СОГЛАСОВАНО

_____/ /
«_____» _____ **2022 г.**

УТВЕРЖДАЮ
Генеральный директор
ООО «Эквирон»

_____ /Селиверстов М.Н./
«_17_» ___04___ **2022 г.**

# СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ «ЕНИСЕЙ»

Обеспечение программное для администрирования баз данных

Руководство программиста

Прикладной программный интерфейс

Версия системы 1.0

Редакция 1

RU.ВРМН.620111-01 33 01

Лист утверждения

Подп. и дата | Инв. № дубл. | Взам. Инв. № | Подп. и дата | Инв. № подл.

2022

УТВЕРЖДЕН

RU.ВРМН.620111-01 33 01-ЛУ

# СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ «ЕНИСЕЙ»

Обеспечение программное для администрирования баз данных

Руководство программиста

Прикладной программный интерфейс

Версия системы 1.0

Редакция 1

RU.ВРМН.620111-01 33 01

Листов 474



2022

## АННОТАЦИЯ

Настоящий документ «Обеспечение программное для администрирования баз данных. Руководство программиста» RU.BPMH.620111-01 33 01 предназначен для ознакомления с функциями прикладного программного интерфейса (ППИ) БД «Енисей». Документ разработан в соответствии с ГОСТ 19.504-79 «Единая система программной документации. Руководство программиста».

## СОДЕРЖАНИЕ

# 1. ОБЩИЕ СВЕДЕНИЯ

## 1.1. Назначение программы

Программное изделие СУБД «Енисей» представляет собой документо-ориентированную систему управления базами данных, относящуюся к типу «No SQL».

## 1.2. Функции программы

Программное изделие СУБД «Енисей» представляет собой систему управления базами данных, относящуюся к типу «No SQL» и реализующую следующие целевые функции:

1) Предоставление прикладного программного интерфейса, ППИ (английское наименование — application programming interface, API) вида «API REST», основанного на протоколе HTTP, обеспечивающем легкость взаимодействия с базой данных СУБД.
2) Обеспечение легкости освоения и использования СУБД посредством простоты структуры ресурсов и методов протокола HTTP (GET, PUT, DELETE).
3) Хранение данных в гибкой, основанной на документе структуре.
4) Предоставление пользователям эффективных средств сопоставления данных, позволяющих запрашивать, комбинировать и фильтровать информацию.
5) Обеспечение простой в использовании репликации с несколькими мастерами (multi-master репликация), с помощью которой можно копировать, обмениваться и синхронизировать данные между базами данных, развернутых на группе средств вычислительной техники (СВТ).
6) Использование модели данных со следующими ключевыми особенностями:
– База данных является самой внешней структурой данных / контейнером в СУБД «Енисей».
– Каждая база данных представляет собой набор независимых документов.
– Каждый документ поддерживает свои собственные данные и автономную схему.
– Метаданные документа содержат информацию о редакции, что позволяет объединить различия, возникшие при отключении баз данных.
– СУБД «Енисей» реализует управление несколькими версиями параллелизма, чтобы избежать необходимости блокировать поле базы данных во время записи.

# 2. УСЛОВИЯ ПРИМЕНЕНИЯ ПРОГРАММЫ

## 2.1. Требуемые характеристики технических средств

### 2.1.1 Требуемые характеристики средств вычислительной техники коллективного пользования

Требуемые характеристики СВТ коллективного пользования:

1) Минимальные требования:

– Центральный процессор:

(1) Архитектура — Intel x86-64.

(2) Число ядер, не менее — 2.

(3) Тактовая частота, ГГц, не менее — 2.

(4) Поддержка набора команд SSE4.2.

– ОЗУ:

(1) Емкость, Гбайт, не менее — 4.

– Накопитель данных:

(1) Емкость, Гбайт, не менее — 8.

2) Рекомендуемые требования:

– Центральный процессор:

(1) Архитектура — Intel x86-64.

(2) Число ядер, не менее — 4 (6 при репликации данных между ЦОД).

(3) Тактовая частота, ГГц, не менее — 3.

(4) Поддержка набора команд SSE4.2.

– ОЗУ:

(1) Емкость, Гбайт, не менее — 16.

– Накопитель данных:

(1) Емкость, Гбайт, не менее — 16.

### 2.1.2 Требуемые характеристики средств вычислительной техники индивидуального пользования

Требуемые характеристики СВТ индивидуального пользования (АРМ оператора):

1) При разработке программных изделий на базе функционала, предоставляемого СУБД «Енисей»:

– ПЭВМ с сетевым адаптером, обеспечивающим инфокоммуникационный канал, и характеристиками, соответствующими рекомендуемым требованиям операционной системы:

(1) Microsoft Windows версии не ниже 10.

(2) Apple macOS версии не ниже 11 «Big Sur».

2) При эксплуатации программных изделий на базе функционала, предоставляемого СУБД «Енисей»:

– ПЭВМ с сетевым адаптером, обеспечивающим инфокоммуникационный канал, и характеристиками, соответствующими рекомендуемым требованиям операционных систем, обеспечивающих функционирование браузеров:

(1) В среде ОС семейства Microsoft Windows:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

(2) В среде ОС семейства Apple macOS:

– Google Chrome версии не ниже 67.

– Apple Safari версии не ниже 11.1.

– Mozilla Firefox версии не ниже 67.

## 2.2. Программное обеспечение, необходимое для функционирования программы

### 2.2.1 Программное обеспечение, устанавливаемое на СВТ коллективного пользования

Для эксплуатации СУБД «Енисей» необходимо следующее программное обеспечение, устанавливаемое на СВТ коллективного пользования:

1) Операционная система — Astra Linux 2.12, REDOS 7.3, Debian 11, Ubuntu 20.04 LTS и выше.

2) SSH Server (режим аутентификации по имени и паролю).

3) Пакеты утилит командной строки и общесистемных программных средств — bash, ifconfig, sysctl, curl, yum, systemctl, yum-config-manager, unzip.

Для установки СУБД «Енисей» необходимо установить операционную систему Linux, настроить SSH Server, пакеты утилит командной строки и общесистемных программных средств, указанных в перечне выше.

### 2.2.2 Программное обеспечение, устанавливаемое на СВТ индивидуального пользования

Для эксплуатации СУБД «Енисей» необходимо следующее программное обеспечение, устанавливаемое на СВТ индивидуального пользования (АРМ оператора):

1) При разработке программных изделий на базе функционала, предоставляемого СУБД «Енисей»:

– Операционные системы:

(1) Microsoft Windows версии не ниже 10.

(2) Apple macOS версии не ниже 11 «Big Sur».

2) При эксплуатации программных изделий на базе функционала, предоставляемого СУБД «Енисей»:

– Сочетание операционных систем и браузеров:

(1) В среде ОС семейства Microsoft Windows:

– Google Chrome версии не ниже 67.

– Microsoft Edge версии не ниже 80.

– Mozilla Firefox версии не ниже 67.

(2) В среде ОС семейства Apple macOS:

– Google Chrome версии не ниже 67.

– Apple Safari версии не ниже 11.1.

– Mozilla Firefox версии не ниже 67.

# 3. ХАРАКТЕРИСТИКИ ПРОГРАММЫ

## 3.1. Временные характеристики

Программное изделие СУБД «Енисей» предоставляет средства доступа к хранящимся внутри СУБД данным, таким образом, при его функционировании должны обеспечиваться временные характеристики, перечень которых отображает Таблица 1.

**Таблица 1 — Перечень временных характеристик, которым должна соответствовать СУБД «Енисей»**

| Временные характеристики, которым должна соответствовать СУБД «Енисей»[1)] | Значение характеристик |
|---|---|
| Количество операций чтения для твердотельного накопителя, операций в секунду | Не менее 300 000 |
| Количество операций чтения для ОЗУ, операций в секунду | Не менее 300 000 |
| Количество операций записи для твердотельного накопителя, операций в секунду | Не менее 200 000 |
| Количество операций записи для ОЗУ, операций в секунду | Не менее 500 000 |
| Средняя задержка, мкс | Не более 4 000 |
| Минимальная задержка, мкс | Не более 200 |
| Максимальная задержка, мкс | Не более 300 000 |
| Задержка, в которую уложились 95% операций, мс | Не более 12 |
| Задержка, в которую уложились 99% операций, мс | Не более 22 |

———————

[1)] Указанные временные характеристики достигаются при использовании средств вычислительной техники со следующими показателями производительности:

Процессор Intel Xeon, число ядер, не менее — 8.

Емкость ОЗУ, Гбайт, не менее — 32.

Емкость твердотельного накопителя данных (суммарно 4 накопителя), Гбайт, не менее — 120.

## 3.2. Режим работы

Режим функционирования БД «Енисей» — круглосуточный круглогодичный (24/7/365).

## 3.3. Средства контроля правильности выполнения и самовосстанавливаемости программы

### 3.3.1 Средства контроля правильности выполнения

Контроль правильности выполнения БД «Енисей» осуществляется посредством:

1) Внутренних средств диагностики.

### 3.3.2 Средства самовосстанавливаемости

Самовосстанавливаемость БД «Енисей» осуществляется посредством внутренних репликаций.

1) Внутренних средств восстановления.

1) Сторонних средств автоматического резервного копирования.

2) Встроенных инструментов операционной системы.

# 4. ОБРАЩЕНИЕ К ПРОГРАММЕ

## 4.1. Введение

Компоненты пути URL-адреса ППИ помогают определить часть сервера СУБД «Енисей», к которой осуществляется доступ. В результате структура запроса URL идентифицирует и эффективно описывает область базы данных, к которой производится обращение.

Как и во всех URL-адресах, отдельные компоненты разделяются прямой косой чертой.

Как правило, компоненты URL и поля JSON, начинающиеся с символа _ (подчеркивание), представляют собой специальный компонент или сущность на сервере или в возвращаемом объекте. Например, сегмент URL /_all_dbs получает список всех баз данных экземпляра СУБД «Енисей».

## 4.2. Основы ППИ

ППИ БД «Енисей» является основным методом взаимодействия с экземпляром БД «Енисей». Запросы выполняются с помощью HTTP-протокола, и запросы используются для получения информации из базы данных, сохранения новых данных, а также для выполнения представлений и форматирования информации, хранящейся в документах.

Запросы к ППИ можно классифицировать по различным областям системы БД «Енисей», к которым производится обращение, и по методу HTTP, используемому для отправки запроса. Различные методы подразумевают различные операции, например, получение информации из базы данных обычно обрабатывается операцией GET, а обновление запросом POST или PUT. Существуют некоторые различия между информацией, которую необходимо предоставить для различных методов.

Почти для всех операций передаваемые данные и возвращаемая структура данных определяются в объекте JavaScript Object Notation (JSON).

Ошибки при доступе к ППИ БД «Енисей» сообщаются с помощью стандартных кодов состояния HTTP.

При обращении к определенным областям ППИ БД «Енисей» предоставляется конкретная информация и примеры по методам HTTP и запросам, структурам JSON и кодам ошибок.

### 4.2.1 Формат запроса и ответы

БД «Енисей» поддерживает следующие методы HTTP-запросов, которые отображает Таблица 2.

**Таблица 2 – Методы HTTP-запросов**

| Метод | Описание |
|---|---|
| GET | Запрос указанного элемента. Как и в обычных HTTP-запросах, формат URL определяет то, что будет возвращено. В БД «Енисей» это могут быть статические элементы, документы базы данных, конфигурационная и статистическая информация. В большинстве случаев информация возвращается в виде документа JSON. |
| HEAD | Метод HEAD используется для получения HTTP-заголовка GET-запроса без тела ответа. |
| POST | Загрузка данных. В БД «Енисей» POST используется для установки значений, включая загрузку документов, установку значений документов и запуск определенных команд администрирования. |
| PUT | Используется для размещения указанного ресурса. В БД «Енисей» PUT используется для создания новых объектов, включая базы данных, документы, представления и конструкторские документы. |
| DELETE | Удаляет указанный ресурс, включая документы, представления и конструкторские документы. |
| COPY | Специальный метод, который можно использовать для копирования документов и объектов. |

Если используется неподдерживаемый тип HTTP-запроса с URL-адресом, который не поддерживает указанный тип, то будет возвращено сообщение 405 Method Not Allowed, в котором перечислены поддерживаемые HTTP-методы. Пример ответа отображает Рисунок 1.

```
{
    "error":"method_not_allowed",
    "reason":"Only GET,HEAD allowed"
}
```

**Пример ответа**
**Рисунок 1**

### 4.2.2 HTTP-заголовки

Поскольку БД «Енисей» использует HTTP для всех коммуникаций, необходимо убедиться, что передаются правильные HTTP-заголовки (и обрабатываются при получении), чтобы были получены правильные формат и кодировка. Различные среды и клиенты будут более или менее строги к действию этих HTTP-заголовков (особенно если они отсутствуют). Необходимо быть максимально конкретными.

#### 4.2.2.1. Формат запроса

Определяет список принимаемых типов данных, которые будут возвращены сервером (т.е. которые принимаются/понимаются клиентом). Формат должен представлять собой список одного или нескольких MIME-типов, разделенных двоеточиями.

Определяет тип содержимого информации, предоставляемой в запросе.

Параметры HTTP заголовка запроса отображает Таблица 3.

**Таблица 3 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип |
|---|---|---|
| Accept | application/json,<br>*/* | Строка (String) |
| Content-type | application/json,<br>текст,<br>соответствующий тип MIME для вложения,<br>application/octet-stream | Строка (String) |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) |

Пример отправки запроса без явного заголовка Accept или при указании */* отображает Рисунок 2.

```
GET /recipes HTTP/1.1
Host: couchdb:5984
Authorization: Basic YWRtaW46YWRtaW4=
Accept: */*
```

**Пример JSON запроса**
**Рисунок 2**

Пример отправки запроса с явным указанием заголовка Accept отображает Рисунок 3.

```
GET /recipes HTTP/1.1
Host: couchdb:5984
Authorization: Basic YWRtaW46YWRtaW4=
Accept: application/json
```

**Пример JSON запроса**
**Рисунок 3**

#### 4.2.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 4.

**Таблица 4 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Cache-control | must-revalidate | Строка (String) | Да |

Заголовок HTTP ответного сообщения Cache-Control содержит предложение для клиентских механизмов кэширования, то есть как относиться к возвращаемой информации.

Параметры HTTP заголовка ответа отображает Таблица 5.

**Таблица 5 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип |
|---|---|---|
| Content-length | длина (в байтах) возвращаемого содержимого | Число (int64) |
| Content-type | text/plain;charset=utf-8 | Строка (String) |
| Etag | отображения ревизии документа или представления. | Строка (String) |

ETag присваивается группе map/reduce (совокупность представлений в одном проектном документе). Любое изменение индексов для этих представлений будет генерировать новый ETag для всех URL представлений в одном проектном документе, даже если результаты конкретного представления не изменились.

Каждый URL _view имеет свой собственный ETag, который обновляется только при внесении изменений в базу данных, влияющих на этот индекс. Если индекс для конкретного представления не изменяется, то это представление сохраняет исходное направление ETag (поэтому чаще отправляется обратно 304 Not Modified).

Параметры HTTP заголовка ответа отображает Таблица 6.

**Таблица 6 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип |
|---|---|---|
| Etag | кодировка | Строка (String) |

Transfer-Encoding: chunked означает, что ответ отправляется частями, этот метод известен как chunked transfer encoding. Он используется, когда СУБД «Енисей» не знает заранее размер данных, которые он будет отправлять.

Параметры HTTP заголовка ответа отображает Таблица 7.

**Таблица 7 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип |
|---|---|---|
| Etag | время получения тела запроса в миллисекундах | Число (int64) |

Доступно, когда содержимое тела включено в запрос.

Параметры HTTP заголовка ответа отображает Таблица 8.

**Таблица 8 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип |
|---|---|---|
| Etag | Уникальный идентификатор для запроса | Строка (String) |

Пример ответа на запрос с явным указанием заголовка Accept отображает Рисунок 4.

```
HTTP/1.1 200 OK
Server: Yenisei/1.0.0 (Erlang OTP/24)
Date: Thu, 13 Jan 2013 13:40:11 GMT
Content-Type: application/json
Content-Length: 227
Cache-Control: must-revalidate
```

**Пример ответа**
**Рисунок 4**

Пример ответа на запрос без явного заголовка Accept или при указании */* отображает Рисунок 5.

```
HTTP/1.1 200 OK
Server: Yenisei/1.0.0 (Erlang OTP/24)
Date: Thu, 13 Jan 2011 13:39:34 GMT
Content-Type: text/plain;charset=utf-8
Content-Length: 227
Cache-Control: must-revalidate
```

**Пример ответа**
**Рисунок 5**

### 4.2.3 Основы JSON

Большинство запросов и ответов используют JavaScript Object Notation (JSON) для форматирования содержимого и структуры данных и ответов.

JSON поддерживает те же основные типы, что и JavaScript, которые отображает Таблица 9.

**Таблица 9 – Параметры тела запроса**

| Тип | Значение | Пример |
|---|---|---|
| Массив (Array) | Список значений, заключенных в квадратные скобки | ["one", "two", "three"] |
| Логический тип (Boolean) | Значение true или false | { "value": true} |
| Число (int64) | Целое число или число с плавающей точкой | |
| Объект (Object) | Набор пар ключ/значение (т.е. ассоциативный массив, или хэш). Ключ должен быть строкой, а значение может быть любым из поддерживаемых значений JSON. | {<br>"servings" : 4,<br>"subtitle" : "Easy to make in advance, and then cook when ready",<br>"cooktime" : 60,<br>"title" : "Chicken Coriander"<br>} |
| Строка (String) | Строка, должна быть заключена в двойные кавычки, поддерживать символы Unicode и экранирование обратной косой чертой. | "A String" |

Разбор JSON в объект JavaScript поддерживается с помощью функции JSON.parse() в JavaScript или с помощью различных библиотек, которые выполнят разбор содержимого в объект JavaScript. Библиотеки для разбора и генерации JSON доступны на многих языках, включая Perl, Python, Ruby, Erlang и другие.

#### 4.2.3.1. Работа с числами

Любые числа, определенные в JSON, которые содержат десятичную точку или экспоненту, будут переданы через Erlang VM в типе данных «double». Любые числа, используемые в представлениях, будут проходить через представление сервера о числе (в распространенном случае JavaScript даже целые числа передаются как double из-за определения числа в JavaScript).

Пример записываемого документа отображает Рисунок 6.

```
{
    "_id":"30b3b38cdbd9e3a587de9b8122000cff",
    "number": 1.1
}
```

**Пример документа**
**Рисунок 6**

Пример записываемого документа в БД «Енисей» отображает Рисунок 7.

```
{
    "_id":"30b3b38cdbd9e3a587de9b8122000cff",
    "_rev":"1-f065cee7c3fd93aa50f6c97acde93030",
    "number":1.1000000000000000888
}
```

**Пример документа из БД «Енисей»**
**Рисунок 7**

В Erlang мы имеем следующее соотношение, которое отображает Рисунок 8.

```
ejson:encode(ejson:decode(<<"1.1">>)).
<<"1.1000000000000000888">>
```

**Пример отображения Erlang**
**Рисунок 8**

Журнал для пары наиболее распространенных библиотек JSON:

Ejson (текущий парсер БД «Енисей») на БД «Енисей» sha 168a663b отображает Рисунок 9

```
$ ./utils/run -i
Erlang R14B04 (erts-5.8.5) [source] [64-bit] [smp:2:2] [rq:2]
[async-threads:4] [hipe] [kernel-poll:true]

Eshell V5.8.5  (abort with ^G)
1> ejson:encode(ejson:decode(<<"1.01234567890123456789012345678901234567890">>)).
<<"1.0123456789012346135">>
2> F = ejson:encode(ejson:decode(<<"1.01234567890123456789012345678901234567890">>)).
<<"1.0123456789012346135">>
3> ejson:encode(ejson:decode(F)).
<<"1.0123456789012346135">>
```

**Ejson**
**Рисунок 9**

Node отображает Рисунок 10.

```
$ node -v
v0.6.15
$ node
JSON.stringify(JSON.parse("1.012345678901234567890123456789012345678901234567890"))
'1.0123456789012346'
var f = JSON.stringify(JSON.parse("1.012345678901234567890123456789012345678901234567890"))
undefined
JSON.stringify(JSON.parse(f))
'1.0123456789012346'
```

**Node**
**Рисунок 10**

Python отображает Рисунок 11.

```
$ python
Python 2.7.2 (default, Jun 20 2012, 16:23:33)
[GCC 4.2.1 Compatible Apple Clang 4.0 (tags/Apple/clang-418.0.60)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
import json
json.dumps(json.loads("1.012345678901234567890123456789012345678901234567890"))
'1.0123456789012346'
f = json.dumps(json.loads("1.012345678901234567890123456789012345678901234567890"))
json.dumps(json.loads(f))
'1.0123456789012346'
```

**Python**
**Рисунок 11**

Ruby отображает Рисунок 12.

```
$ irb --version
irb 0.9.5(05/04/13)
require 'JSON'
=> true
JSON.dump(JSON.load("[1.012345678901234567890123456789012345678901234567890]"))
=> "[1.01234567890123]"
f = JSON.dump(JSON.load("[1.012345678901234567890123456789012345678901234567890]"))
=> "[1.01234567890123]"
JSON.dump(JSON.load(f))
=> "[1.01234567890123]"
```

**Ruby**
**Рисунок 12**

Spidermonkey отображает Рисунок 13.

```
$ js -h 2>&1 | head -n 1
JavaScript-C 1.8.5 2011-03-31
$ js
js> JSON.stringify(JSON.parse("1.012345678901234567890123456789012345678901234567890"))
"1.0123456789012346"
js> var f = JSON.stringify(JSON.parse("1.012345678901234567890123456789012345678901234567890"))
js> JSON.stringify(JSON.parse(f))
"1.0123456789012346"
```

**Spidermonkey**
**Рисунок 13**

## 4.3. Сервер

Интерфейс группы «Сервер» СУБД «Енисей» предоставляет методы для получения информации о базе данных, а также получения и настроек конфигурации.

### 4.3.1 Получить информацию об экземпляре базы данных

Метод предоставляет метаинформацию об экземпляре СУБД «Енисей». Ответное сообщение содержащую информацию о сервере, включая приветственное сообщение и версию СУБД «Енисей».

#### 4.3.1.1. Формат запроса

Параметры запроса отображает Таблица 10.

**Таблица 10 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/ |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 11.

**Таблица 11 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 14.

```
GET / HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 14**

#### 4.3.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 12.

**Таблица 12 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 13.

**Таблица 13 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 15. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 179
Content-Type: application/json
Date: Sat, 10 Aug 2013 06:33:33 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "couchdb": "Welcome",
    "uuid": "85fb71bf700c17267fef77535820e371",
    "vendor": {
        "name": "The Apache Software Foundation",
        "version": "1.3.1"
    },
    "version": "1.3.1"
}
```

**Ответ**

**Рисунок 15**

### 4.3.2 Получить список запущенных заданий

Метод предоставляет список запущенных заданий, включая тип задания, имя, статус и идентификатор процесса. Результатом выполнения запроса является JSON-массив текущих запущенных задач, причем каждая задача описывается одним объектом. В зависимости от типа операции набор полей объекта ответа может быть различным.

#### 4.3.2.1. Формат запроса

Параметры запроса отображает Таблица 14.

**Таблица 14 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_active_tasks |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 15.

**Таблица 15 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 16.

```
GET /_active_tasks HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 16**

#### 4.3.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 16.

**Таблица 16 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 17.

**Таблица 17 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| changes_done | Обработанные изменения | Число (int64) | Нет |
| database | Исходная база данных | Строка (String) | Да |
| pid | Идентификатор процесса | Строка (String) | Да |
| progress | Текущий прогресс в процентах | Число (int64) | Да |
| type | Тип операции, перечень типов заданий отображает подпункт 5.3.1.1 | Строка (String) | Да |
| started_on | Время начала задачи в виде временной метки unix | Число (int64) | Да |
| status | Сообщение о состоянии задачи | Строка (String) | Нет |
| task | Имя задачи | Строка (String) | Нет |
| total_changes | Общее количество изменений в процессе | Число (int64) | Нет |
| updated_on | Временная метка Unix последнего обновления операции | Число (int64) | Нет |
| checkpointed_source_seq | Идентификатор последовательности источника, который был успешно реплицирован в последний раз | Число (int64) | Нет |
| design_document | Проектный документ | Строка (String) | Нет |
| continuous | Непрерывная | Логический тип (Boolean) | Нет |
| doc_id | Идентификатор документа | Строка (String) | Нет |
| doc_write_failures | Количество документов, которые не удалось записать на цель | Число (int64) | Нет |
| docs_read | Количество документов, которые были прочитаны из источника | Число (int64) | Нет |
| docs_written | Количество документов, которые были записаны в цель | Число (int64) | Нет |
| missing_revisions_found | Количество ревизий, которые были найдены на источнике, но отсутствуют в цели | Число (int64) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| replication_id | Идентификатор репликации | Строка (String) | Нет |
| revisions_checked | Количество ревизий, которые были проверены с момента начала этой репликации | Число (int64) | Нет |
| source | Исходная база данных | Строка (String) | Нет |
| target | Целевая база данных | Строка (String) | Нет |

Коды состояния отображает Таблица 18.

**Таблица 18 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора |

Ответ в случае успешного завершения запроса отображает Рисунок 17. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 1690
Content-Type: application/json
Date: Sat, 10 Aug 2013 06:37:31 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

[
    {
        "changes_done": 64438,
        "database": "mailbox",
        "pid": "<0.12986.1>",
        "progress": 84,
        "started_on": 1376116576,
        "total_changes": 76215,
        "type": "database_compaction",
        "updated_on": 1376116619
    },
    {
        "changes_done": 14443,
        "database": "mailbox",
        "design_document": "c9753817b3ba7c674d92361f24f59b9f",
        "pid": "<0.10461.3>",
        "progress": 18,
        "started_on": 1376116621,
        "total_changes": 76215,
        "type": "indexer",
        "updated_on": 1376116650
    },
    {
        "changes_done": 5454,
        "database": "mailbox",
        "design_document": "_design/meta",
        "pid": "<0.6838.4>",
        "progress": 7,
        "started_on": 1376116632,
        "total_changes": 76215,
        "type": "indexer",
        "updated_on": 1376116651
    },
    {
        "checkpointed_source_seq": 68585,
        "continuous": false,
        "doc_id": null,
        "doc_write_failures": 0,
        "docs_read": 4524,
        "docs_written": 4524,
        "missing_revisions_found": 4524,
        "pid": "<0.1538.5>",
        "progress": 44,
        "replication_id": "9bc1727d74d49d9e157e260bb8bbd1d5",
        "revisions_checked": 4524,
        "source": "mailbox",
        "source_seq": 154419,
        "started_on": 1376116644,
        "target": "http://mailsrv:5984/mailbox",
        "type": "replication",
        "updated_on": 1376116651
    }
]
```

**Ответ**

**Рисунок 17**

### 4.3.3 Получить список всех баз данных в экземпляре

Возвращает список всех баз данных в экземпляре БД «Енисей».

#### 4.3.3.1. Формат запроса

Параметры запроса отображает Таблица 19.

**Таблица 19 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_all_dbs |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 20.

**Таблица 20 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 21.

**Таблица 21 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| descending | Возвращать базы данных в порядке убывания по ключу. По умолчанию false. | Логический тип (Boolean) | Нет |
| endkey | Прекратить возврат баз данных при достижении указанного ключа. | Объект (Json Object) | Нет |
| end_key | Псевдоним для параметра endkey. | Объект (Json Object) | Нет |
| limit | Ограничить количество возвращаемых баз данных указанным числом. | Число (int64) | Нет |
| skip | Пропустить указанное число баз данных перед началом возврата результатов. По умолчанию «0». | Число (int64) | Нет |
| startkey | Возвращать базы данных, начинающиеся с указанного ключа. | Объект (Json Object) | Нет |
| start_key | Псевдоним для startkey. | Объект (Json Object) | Нет |

Пример запроса отображает Рисунок 18.

```
GET /_all_dbs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 18**

#### 4.3.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 22.

**Таблица 22 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 23.

**Таблица 23 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 19. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 52
Content-Type: application/json
Date: Sat, 10 Aug 2013 06:57:48 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

[
   "_users",
   "contacts",
   "docs",
   "invoices",
   "locations"
]
```

**Ответ**

**Рисунок 19**

### 4.3.4 Получить список информации обо всех базах данных в экземпляре

Метод возвращает список информации обо всех базах данных в экземпляре БД «Енисей».

#### 4.3.4.1. Формат запроса

Параметры запроса отображает Таблица 24.

**Таблица 24 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_dbs_info |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 25.

**Таблица 25 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 26.

**Таблица 26 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| descending | Возвращать базы данных в порядке убывания по ключу. По умолчанию false. | Логический тип (Boolean) | Нет |
| endkey | Прекратить возврат баз данных при достижении указанного ключа. | Объект (Json Object) | Нет |
| end_key | Псевдоним для параметра endkey. | Объект (Json Object) | Нет |
| limit | Ограничить количество возвращаемых баз данных указанным числом. | Число (int64) | Нет |
| skip | Пропустить указанное число баз данных перед началом возврата результатов. По умолчанию 0. | Число (int64) | Нет |
| startkey | Возвращать базы данных, начинающиеся с указанного ключа. | Объект (Json Object) | Нет |
| start_key | Псевдоним для startkey. | Объект (Json Object) | Нет |

Пример запроса отображает Рисунок 20.

```
GET /_dbs_info HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 20**

#### 4.3.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 27.

**Таблица 27 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 28.

**Таблица 28 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 21. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Thu, 18 Nov 2021 14:37:35 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

[
  {
    "key": "animals",
    "info": {
      "db_name": "animals",
      "update_seq": "52232",
      "sizes": {
        "file": 1178613587,
        "external": 1713103872,
        "active": 1162451555
      },
      "purge_seq": 0,
      "doc_del_count": 0,
      "doc_count": 52224,
      "disk_format_version": 6,
      "compact_running": false,
      "cluster": {
        "q": 8,
        "n": 3,
        "w": 2,
        "r": 2
      },
      "instance_start_time": "0"
    }
  }
]
```

**Ответ**

**Рисунок 21**

### 4.3.5 Получить информацию о списке указанных баз данных в экземпляре

Метод возвращает информацию о списке указанных баз данных в экземпляре БД «Енисей». Данный метод позволяет запрашивать информацию о нескольких базах данных в одном запросе, вместо нескольких запросов GET /{db}.

#### 4.3.5.1. Формат запроса

Параметры запроса отображает Таблица 29.

**Таблица 29 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_dbs_info |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 30.

**Таблица 30 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 31.

**Таблица 31 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| keys | Массив имен баз данных, которые необходимо запросить | Массив (Array) | Да |

Пример запроса отображает Рисунок 22.

```
POST /_dbs_info HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
Content-Type: application/json

{
    "keys": [
        "animals",
        "plants"
    ]
}
```

**Запрос**
**Рисунок 22**

Поддерживаемое количество указанных баз данных в списке может быть ограничено путем изменения параметра max_db_number_for_dbs_info_req в конфигурационном файле. По умолчанию ограничение составляет 100. Увеличение лимита может повлиять на нагрузку на сервер.

#### 4.3.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 32.

**Таблица 32 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Коды состояния отображает Таблица 33.

**Таблица 33 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Отсутствуют ключи или превышено количество ключей в запросе |

Ответ в случае успешного завершения запроса отображает Рисунок 23. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 20 Dec 2017 06:57:48 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

[
  {
    "key": "animals",
    "info": {
      "db_name": "animals",
      "update_seq": "52232",
      "sizes": {
        "file": 1178613587,
        "external": 1713103872,
        "active": 1162451555
      },
      "purge_seq": 0,
      "doc_del_count": 0,
      "doc_count": 52224,
      "disk_format_version": 6,
      "compact_running": false,
      "cluster": {
        "q": 8,
        "n": 3,
        "w": 2,
        "r": 2
      },
      "instance_start_time": "0"
    }
  },
  {
    "key": "plants",
    "info": {
      "db_name": "plants",
      "update_seq": "303",
      "sizes": {
        "file": 3872387,
        "external": 2339,
        "active": 67475
      },
      "purge_seq": 0,
      "doc_del_count": 0,
      "doc_count": 11,
      "disk_format_version": 6,
      "compact_running": false,
      "cluster": {
        "q": 8,
        "n": 3,
        "w": 2,
        "r": 2
      },
      "instance_start_time": "0"
    }
  }
]
```

**Ответ**

**Рисунок 23**

### 4.3.6 Получить статус узла или кластера

Метод возвращает статус узла или кластера в соответствии с мастером настройки кластера.

#### 4.3.6.1. Формат запроса

Параметры запроса отображает Таблица 34.

**Таблица 34 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_cluster_setup |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 35.

**Таблица 35 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 36.

**Таблица 36 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ensure_dbs_exist | Список системных баз данных, существование которых необходимо обеспечить на узле/кластере. По умолчанию ["_users", "_replicator"]. | Массив (Array) | Нет |

Пример запроса отображает Рисунок 24.

```
GET /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**

**Рисунок 24**

#### 4.3.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 37.

**Таблица 37 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 38.

**Таблица 38 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Текущее состояние узла и/или кластера, возвращаемые состояния отображает подпункт 5.3.1.2 | Строка (String) | Да |

Коды состояния отображает Таблица 39.

**Таблица 39 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 25. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
X-CouchDB-Body-Time: 0
X-Couch-Request-ID: 5c058bdd37
Server: Yenisei/1.0.0 (Erlang OTP/24)
Date: Sun, 30 Jul 2017 06:33:18 GMT
Content-Type: application/json
Content-Length: 29
Cache-Control: must-revalidate

{"state":"cluster_enabled"}
```

**Ответ**
**Рисунок 25**

### 4.3.7 Настроить узел

Метод позволяет настроить узел как отдельный (автономный) узел, как часть кластера или завершение настройки кластера.

#### 4.3.7.1. Формат запроса

Параметры запроса отображает Таблица 40.

**Таблица 40 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_cluster_setup |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 41.

**Таблица 41 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 42.

**Таблица 42 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| action | Перечень значений отображает подпункт 5.3.1.3 | Строка (String) | Да |
| bind_address | IP-адрес, к которому следует привязать текущий узел. Специальное значение 0.0.0.0 может быть указано для привязки ко всем интерфейсам узла. (только для enable_cluster и enable_single_node) | Строка (String) | Да |
| username | Имя пользователя администратора сервера для создания. (только для enable_cluster и enable_single_node), или имя пользователя администратора удаленного сервера (add_node) | Строка (String) | Да |
| password | Пароль для создаваемого администратора сервера. (только для enable_cluster и enable_single_node), или имя пользователя администратора удаленного сервера (add_node) | Строка (String) | Да |
| port | TCP-порт, к которому нужно привязать этот узел (только для enable_cluster и enable_single_node) или TCP-порт, к которому нужно привязать удаленный узел (только для add_node) | Число (int64) | Да |
| node_count | Общее число узлов, которые будут объединены в кластер, включая данный узел. Используется для определения значения n кластера, максимум до 3. (только для enable_cluster). remote_node (строка): IP-адрес удаленного узла, который нужно настроить как часть списка узлов этого кластера. (только для enable_cluster) | Число (int64) | Да |
| remote_node | IP-адрес удаленного узла, который нужно настроить как часть списка узлов этого кластера. (только для enable_cluster) | Строка (String) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| remote_current_user | Имя пользователя администратора сервера, авторизованного на удаленном узле. (только для enable_cluster) | Строка (String) | Нет |
| remote_current_password | Пароль администратора сервера, авторизованного на удаленном узле. (только для enable_cluster) host (строка): IP-адрес удаленного узла, который необходимо добавить в кластер. (только для add_node) | Строка (String) | Нет |
| ensure_dbs_exist | Список системных баз данных, существование которых необходимо убедиться на узле/кластере. По умолчанию ["_users", "_replicator"]. | Массив (Array) | Нет |

После установки и начальной настройки/конфигурации мы можем настроить кластер. На каждом узле необходимо выполнить команду для настройки узла.

Пример запроса отображает Рисунок 26.

```
POST /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
Content-Type: application/json
{
    "action": "enable_cluster",
    "bind_address":"0.0.0.0",
    "username": "admin",
    "password":"password",
    "node_count":"3"
}
```

**Запрос**
**Рисунок 26**

После этого мы можем объединить все узлы вместе. Выбирается один узел в качестве «узла координации настройки», на котором будут выполняться все эти команды. Этот «узел координации установки» только управляет установкой и требует, чтобы все остальные узлы могли видеть его и наоборот. Он не имеет никакого специального назначения за пределами процесса установки; БД «Енисей» не имеет концепции «главного» узла в кластере.

Установка не будет работать с недоступными узлами. Все узлы должны быть онлайн и правильно сконфигурированы до начала процесса настройки кластера.

Чтобы присоединить узел к кластеру, необходимо выполнить эти команды для каждого узла, который нужно добавить.

Пример запроса отображает Рисунок 27 и Рисунок 28.

```
POST /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: <setup-coordination-node>:5984
Content-Type: application/json
{
    "action": "enable_cluster",
    "bind_address":"0.0.0.0",
    "username": "admin",
    "password":"password",
    "port": 5984,
    "node_count": "3",
    "remote_node": "<remote-node-ip>",
    "remote_current_user": "<remote-node-username>",
    "remote_current_password": "<remote-node-password>"
}
```

**Запрос**
**Рисунок 27**

```
POST /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: <setup-coordination-node>:5984
Content-Type: application/json
{
    "action": "add_node",
    "host":"<remote-node-ip>",
    "port": "<remote-node-port>",
    "username": "admin",
    "password":"password"
}
```

**Запрос**
**Рисунок 28**

Это позволит соединить два узла вместе. После этого необходимо выполнить следующую команду, чтобы завершить настройку кластера и добавить системные базы данных:

Пример запроса отображает Рисунок 29.

```
POST /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: <setup-coordination-node>:5984
Content-Type: application/json
{
    "action": "finish_cluster"
}
```

**Запрос**
**Рисунок 29**

Необходимо проверить установку

Пример запроса отображает Рисунок 30.

```
POST /_cluster_setup HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: <setup-coordination-node>:5984
Content-Type: application/json
```

**Запрос**
**Рисунок 30**

Необходимо убедиться, что все кластеры подключены

Пример запроса отображает Рисунок 31.

```
POST /_ membership HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: <setup-coordination-node>:5984
Content-Type: application/json
```

**Запрос**
**Рисунок 31**

#### 4.3.7.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 43.

**Таблица 43 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Ответ на запрос проверки установки отображает Рисунок 32.

```
{"state":"cluster_finished"}
```

**Ответ**
**Рисунок 32**

Ответ на запрос проверки подключения кластеров отображает Рисунок 33.

```
{
    "all_nodes": [
        "couchdb@couch1.test.com",
        "couchdb@couch2.test.com",
        "couchdb@couch3.test.com",
    ],
    "cluster_nodes": [
        "couchdb@couch1.test.com",
        "couchdb@couch2.test.com",
        "couchdb@couch3.test.com",
    ]
}
```

**Ответ**
**Рисунок 33**

### 4.3.8 Получить список всех событий базы данных в экземпляре

Метод возвращает список всех событий в экземпляре БД «Енисей». Для использования этой конечной точки необходимо существование базы данных _global_changes.

#### 4.3.8.1. Формат запроса

Параметры запроса отображает Таблица 44.

**Таблица 44 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_db_updates |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 45.

**Таблица 45 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 46.

**Таблица 46 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| feed | normal: Возвращает все изменения БД, затем закрывает соединение. По умолчанию.<br>longpoll: Закрывает соединение после первого события.<br>continuous: Отправка строки JSON на каждое событие. Держит сокет открытым до истечения времени.<br>eventsource: Как и continuous, но отправляет события в формате EventSource. | Строка (String) | Нет |
| timeout | Количество секунд до того, как БД «Енисей» закроет соединение. По умолчанию 60. | Число (int64) | Нет |
| heartbeat | Период в миллисекундах, после которого в результатах отправляется пустая строка. Применимо только для фидов longpoll, continuous и eventsource. Отменяет любой тайм-аут, чтобы фид оставался живым неограниченное время. По умолчанию 60000. Может иметь значение true, чтобы использовать значение по умолчанию. | Число (int64) | Нет |
| since | Возвращает только обновления с указанного идентификатора последовательности. Если идентификатор последовательности указан, но не существует, возвращаются все изменения. Может быть строкой now, чтобы начать показывать только новые обновления. | Строка (String) | Нет |

Пример запроса отображает Рисунок 34.

```
GET /_db_updates HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**

**Рисунок 34**

#### 4.3.8.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 47.

**Таблица 47 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 48.

**Таблица 48 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| results | Массив событий базы данных. Для режимов longpoll и continuous весь ответ представляет собой содержимое массива results. | Массив (Array) | Да |
| last_seq | Идентификатор последней сообщенной последовательности. | Строка (String) | Да |

JSON объект отображает Таблица 49.

**Таблица 49 – JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| db_name | Имя базы данных | Строка (String) | Да |
| type | Событие базы данных одно из created, updated, deleted | Строка (String) | Да |
| seq | Последовательность обновления события | Объект (Json Object) | Да |

Коды состояния отображает Таблица 50.

**Таблица 50 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 35. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 18 Mar 2017 19:01:35 GMT
Etag: "C1KU98Y6H0LGM7EQQYL6VSL07"
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
X-Couch-Request-ID: ad87efc7ff
X-CouchDB-Body-Time: 0

{
    "results":[
        {"db_name":"mailbox","type":"created","seq":"1-
g1AAAAFReJzLYWBg4MhgTmHgzcvPy09JdcjLz8gvLskBCjMlMiTJ____PyuDOZExFyjAnmJhkWaeaIquGIf2JAUgmWQPMiGRAZcaB5
CaePxqEkBq6vGqyWMBkgwNQAqobD4h"},
        {"db_name":"mailbox","type":"deleted","seq":"2-
g1AAAAFReJzLYWBg4MhgTmHgzcvPy09JdcjLz8gvLskBCjMlMiTJ____PyuDOZEpFyjAnmJhkWaeaIquGIf2JAUgmWQPMiGRAZcaB5
CaePxqEkBq6vGqyWMBkgwNQAqobD4hdQsg6vYTUncAou4-IXUPIOpA7ssCAIFHa60"},
    ],
    "last_seq": "2-
g1AAAAFReJzLYWBg4MhgTmHgzcvPy09JdcjLz8gvLskBCjMlMiTJ____PyuDOZEpFyjAnmJhkWaeaIquGIf2JAUgmWQPMiGRAZcaB5
CaePxqEkBq6vGqyWMBkgwNQAqobD4hdQsg6vYTUncAou4-IXUPIOpA7ssCAIFHa60"
}
```


**Ответ**

**Рисунок 35**

### 4.3.9 Получить узлы, входящие в кластер

Отображает узлы, входящие в кластер, как cluster_nodes. В поле all_nodes отображаются все узлы (all_node), о которых знает данный узел, включая те, которые являются частью кластера.

#### 4.3.9.1. Формат запроса

Параметры запроса отображает Таблица 51.

**Таблица 51 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_membership |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 52.

**Таблица 52 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 36.

```
GET /_membership HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 36**

#### 4.3.9.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 53.

**Таблица 53 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 54.

**Таблица 54 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 37. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 11 Jul 2015 07:02:41 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Content-Length: 142

{
    "all_nodes": [
        "node1@127.0.0.1",
        "node2@127.0.0.1",
        "node3@127.0.0.1"
    ],
    "cluster_nodes": [
        "node1@127.0.0.1",
        "node2@127.0.0.1",
        "node3@127.0.0.1"
    ]
}
```

**Ответ**
**Рисунок 37**

### 4.3.10 Управление операцией репликации

Запрос, настройка или остановка операции репликации.

#### 4.3.10.1. Формат запроса

Параметры запроса отображает Таблица 55.

**Таблица 55 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_replicate |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 56.

**Таблица 56 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Нет |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 57.

**Таблица 57 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| cancel | Отмена репликации. | Логический тип (Boolean) | Нет |
| continuous | Настроить репликацию на непрерывную работу. | Логический тип (Boolean) | Нет |
| create_target | Создает целевую базу данных. Требуются привилегии администратора на целевом сервере. | Логический тип (Boolean) | Нет |
| create_target_params | Объект, содержащий параметры, которые будут использоваться при создании целевой базы данных. Может включать стандартные параметры q и n. | Объект (Json Object) | Нет |
| doc_ids | Массив идентификаторов документов, подлежащих синхронизации. doc_ids, filter и selector являются взаимоисключающими. | Массив (Array) | Нет |
| filter | Имя функции фильтрации. doc_ids, filter и selector являются взаимоисключающими. | Строка (String) | Нет |
| selector | Селектор для фильтрации документов для синхронизации. Имеет такое же поведение, как и объекты selector в документах репликации. doc_ids, filter и selector являются взаимоисключающими. | Объект (Json Object) | Нет |
| source_proxy | Адрес прокси-сервера, через который должна происходить репликация из источника (протокол может быть «http» или «socks5»). | Строка (String) | Нет |
| target_proxy | Адрес прокси-сервера, через который должна происходить репликация на цель (протокол может быть «http» или «socks5»). | Строка (String) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| source | Полный URL базы данных источника или объект, который содержит полный URL базы данных источника с дополнительными параметрами, такими как заголовки. Например: „http://example.com/source_db_name“ или {«url»: «url здесь», «headers»: { «header1»: «value1», … . }}. | Строка (String) / Объект (Object) | Да |
| target | Полный URL целевой базы данных или объект, содержащий полный URL целевой базы данных с дополнительными параметрами, такими как заголовки. Например: „http://example.com/target_db_name“ или {«url»: «url здесь», «headers»: { «header1»: «value1», … . }}. | Строка (String) / Объект (Object) | Да |

Для параметров источника и цели репликации требуются полностью определенные URL.

Пример запроса отображает Рисунок 38.

```
POST /_replicate HTTP/1.1
Accept: application/json
Content-Length: 80
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "source": "http://127.0.0.1:5984/db_a",
    "target": "http://127.0.0.1:5984/db_b"
}
```

**Запрос**
**Рисунок 38**

#### 4.3.10.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 58.

**Таблица 58 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 59.

**Таблица 59 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| history | История репликации. | Массив объектов HistoryItem (Array of HistoryItem) | Да |
| ok | Статус репликации. | Логический тип (Boolean) | Да |
| replication_id_version | Версия протокола репликации | Число (int64) | Да |
| session_id | Уникальный идентификатор сессии | Строка (String) | Да |
| source_last_seq | Последний номер последовательности, считанный из исходной базы данных. | Число (int64) | Да |

История репликации представляет собой массив объектов со структурой, которую отображает Таблица 60.

**Таблица 60 – JSON объект HistoryItem**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| doc_write_failures | Количество сбоев записи документов. | Число (int64) | Да |
| docs_read | Число прочитанных документов. | Число (int64) | Да |
| docs_written | Число документов, записанных на цель | Число (int64) | Да |
| end_last_seq | Номер последней последовательности в потоке изменений | Число (int64) | Да |
| end_time | Дата/время завершения операции репликации в формате RFC 2822 | Строка (String) | Да |
| missing_checked | Количество проверенных отсутствующих документов | Число (int64) | Да |
| missing_found | Число найденных отсутствующих документов | Число (int64) | Да |
| recorded_seq | Номер последней записанной последовательности | Число (int64) | Да |
| session_id | Идентификатор сессии для данной операции репликации | Строка (String) | Да |
| start_last_seq | Первый порядковый номер в потоке изменений | Число (int64) | Да |
| start_time | Дата/время начала операции репликации в формате RFC 2822 | Строка (String) | Да |

Коды состояния отображает Таблица 61.

**Таблица 61 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 202 Accepted | Запрос на непрерывную репликацию был принят |
| 400 Bad Request | Неверные данные JSON |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Не найдена исходная или целевая БД или попытка отменить неизвестное задание репликации |
| 500 Internal Server Error | Спецификация JSON недействительна |

Ответ в случае успешного завершения запроса отображает Рисунок 39. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 692
Content-Type: application/json
Date: Sun, 11 Aug 2013 20:38:50 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "history": [
        {
            "doc_write_failures": 0,
            "docs_read": 10,
            "docs_written": 10,
            "end_last_seq": 28,
            "end_time": "Sun, 11 Aug 2013 20:38:50 GMT",
            "missing_checked": 10,
            "missing_found": 10,
            "recorded_seq": 28,
            "session_id": "142a35854a08e205c47174d91b1f9628",
            "start_last_seq": 1,
            "start_time": "Sun, 11 Aug 2013 20:38:50 GMT"
        },
        {
            "doc_write_failures": 0,
            "docs_read": 1,
            "docs_written": 1,
            "end_last_seq": 1,
            "end_time": "Sat, 10 Aug 2013 15:41:54 GMT",
            "missing_checked": 1,
            "missing_found": 1,
            "recorded_seq": 1,
            "session_id": "6314f35c51de3ac408af79d6ee0c1a09",
            "start_last_seq": 0,
            "start_time": "Sat, 10 Aug 2013 15:41:54 GMT"
        }
    ],
    "ok": true,
    "replication_id_version": 3,
    "session_id": "142a35854a08e205c47174d91b1f9628",
    "source_last_seq": 28
}
```

**Ответ**

**Рисунок 39**

### 4.3.10.3. Операции репликации

Цель репликации заключается в том, чтобы в конце процесса все активные документы в исходной базе данных были также в базе данных назначения, а все документы, которые были удалены в исходных базах данных, были также удалены (если они существуют) в целевой базе данных.

Репликация может быть описана как push или pull репликация:

Репликация Pull это когда источником является удаленный экземпляр БД «Енисей», а целью локальная база данных.

Вытягивающая репликация является наиболее полезным решением, если исходная база данных имеет постоянный IP-адрес, а целевая (локальная) база данных может иметь динамически назначаемый IP-адрес (например, через DHCP). Это особенно важно, если осуществляется репликация данных на мобильное или другое устройство с центрального сервера.

Репликация Push — это когда источником является локальная база данных, а целью удаленная база данных.

#### 4.3.10.3.1. Указание исходной и целевой базы данных

Необходимо использовать URL-спецификацию базы данных БД «Енисей», если хотите выполнить репликацию в одной из следующих двух ситуаций:

Репликация с удаленной базой данных (т.е. с другим экземпляром БД «Енисей» на том же хосте или на другом хосте).

Репликация с базой данных, требующей аутентификации.

##### 4.3.10.3.1.1. Формат запроса

Чтобы запросить репликацию между базой данных, локальной для экземпляра БД «Енисей», которому посылается запрос, и удаленной базой данных, используется запрос, который отображает Рисунок 40.

```
POST http://couchdb:5984/_replicate HTTP/1.1
Content-Type: application/json
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "source" : "recipes",
    "target" : "http://coucdb-remote:5984/recipes",
}
```

**Запрос**
**Рисунок 40**

Добавив поле create_target в объект запроса, можно создать целевую базу данных, что отображает Рисунок 41.

```
POST http://couchdb:5984/_replicate HTTP/1.1
Content-Type: application/json
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "create_target" : true
    "source" : "recipes",
    "target" : "http://couchdb-remote:5984/recipes",
}
```

**Запрос**
**Рисунок 41**

#### 4.3.10.3.1.2. Формат ответа на запрос

Во всех случаях запрашиваемые базы данных в спецификации источника и цели должны существовать. Если это не так, в объекте JSON будет возвращена ошибка. Ответ с ошибкой отображает Рисунок 42.

```
{
    "error" : "db_not_found"
    "reason" : "could not open http://couchdb-remote:5984/ol1ka/",
}
```

**Ответ**
**Рисунок 42**

Поле create_target не является разрушительным. Если база данных уже существует, репликация проходит в обычном режиме.

### 4.3.10.3.2. Одноразовая репликация

#### 4.3.10.3.2.1. Формат запроса

Можно запросить репликацию базы данных, чтобы синхронизировать две базы данных. По умолчанию процесс репликации происходит один раз и синхронизирует две базы данных вместе. Например, можно запросить однократную синхронизацию между двумя базами данных, указав поля source и target в содержимом JSON запроса, что отображает Рисунок 43.

```
POST http://couchdb:5984/_replicate HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "source" : "recipes",
    "target" : "recipes-snapshot"
}
```

**Запрос**
**Рисунок 43**

#### 4.3.10.3.2.2. Формат ответа на запрос

В приведенном выше примере будут синхронизированы базы данных recipes и recipes-snapshot. Эти базы данных являются локальными для экземпляра БД «Енисей», к которому был сделан запрос. Ответом будет JSON-структура, содержащая информацию об успехе (или неудаче) процесса синхронизации, а также статистику о процессе, что отображает Рисунок 44.

```
{
    "ok" : true,
    "history" : [
        {
            "docs_read" : 1000,
            "session_id" : "52c2370f5027043d286daca4de247db0",
            "recorded_seq" : 1000,
            "end_last_seq" : 1000,
            "doc_write_failures" : 0,
            "start_time" : "Thu, 28 Oct 2010 10:24:13 GMT",
            "start_last_seq" : 0,
            "end_time" : "Thu, 28 Oct 2010 10:24:14 GMT",
            "missing_checked" : 0,
            "docs_written" : 1000,
            "missing_found" : 1000
        }
    ],
    "session_id" : "52c2370f5027043d286daca4de247db0",
    "source_last_seq" : 1000
}
```

**Ответ**
**Рисунок 44**

#### 4.3.10.3.3. Непрерывная репликация

Синхронизация базы данных с помощью ранее описанных методов происходит только один раз, во время выполнения запроса на репликацию. Чтобы целевая база данных постоянно реплицировалась с исходной, необходимо установить поле continuous объекта JSON в запросе в true.

##### 4.3.10.3.3.1. Формат запроса

При непрерывной репликации изменения в исходной базе данных будут реплицироваться в целевую базу данных бесконечно, пока репликация не будет специально прекращена, Пример запроса отображает Рисунок 45.

```
POST http://couchdb:5984/_replicate HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "continuous" : true
    "source" : "recipes",
    "target" : "http://couchdb-remote:5984/recipes",
}
```

**Запрос**
**Рисунок 45**

Изменения будут реплицироваться между двумя базами данных до тех пор, пока между двумя экземплярами доступно сетевое соединение.

Чтобы две базы данных синхронизировались друг с другом, необходимо настроить репликацию в обоих направлениях; то есть, необходимо реплицировать с источника на цель и отдельно с цели на источник.

#### 4.3.10.3.4. Отмена непрерывной репликации

Можно отменить непрерывную репликацию, добавив поле cancel в объект JSON запроса и установив его значение в true. Следует обратить внимание, что структура запроса должна быть идентична исходной, чтобы запрос на отмену был удовлетворен. Например, если была запрошена непрерывная репликация, запрос на отмену также должен содержать поле continuous.

##### 4.3.10.3.4.1. Формат запроса

Запрос на репликацию отображает Рисунок 46

```
POST http://couchdb:5984/_replicate HTTP/1.1
Content-Type: application/json
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "source" : "recipes",
    "target" : "http://couchdb-remote:5984/recipes",
    "create_target" : true,
    "continuous" : true
}
```

**Запрос**
**Рисунок 46**

Запрос на отмену отображает Рисунок 47.

```
POST http://couchdb:5984/_replicate HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "cancel" : true,
    "continuous" : true
    "create_target" : true,
    "source" : "recipes",
    "target" : "http://couchdb-remote:5984/recipes",
}
```

**Запрос**
**Рисунок 47**

Запрос на отмену несуществующей репликации приводит к ошибке 404.

### 4.3.11 Получить список заданий репликации

Список заданий репликации. Включает репликации, созданные через конечную точку /_replicate, а также созданные из документов репликации. Не включает репликации, которые были завершены или не были запущены из-за неправильного оформления документов репликации. Описание каждого задания будет включать информацию об источнике и цели, идентификатор репликации, историю последних событий и некоторые другие сведения.

#### 4.3.11.1. Формат запроса

Параметры запроса отображает Таблица 62.

**Таблица 62 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_scheduler/jobs |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 63.

**Таблица 63 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 64.

**Таблица 64 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| limit | Количество возвращаемых результатов. | Число (int64) | Нет |
| skip | Количество пропускаемых результатов, начиная с начала, упорядоченных по идентификатору репликации. | Число (int64) | Нет |

Пример запроса отображает Рисунок 48.

```
GET /_scheduler/jobs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 48**

#### 4.3.11.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 65.

**Таблица 65 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

JSON объект отображает Таблица 66.

**Таблица 66 – JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Количество пропущенных результатов. | Число (int64) | Да |
| total_rows | Общее количество заданий репликации. | Число (int64) | Да |
| id | Идентификатор репликации | Строка (String) | Да |
| database | База данных документов репликации | Строка (String) | Да |
| doc_id | Идентификатор документа репликации | Строка (String) | Да |
| history | История событий с временной меткой в виде списка объектов | Строка (String) | Да |
| pid | Идентификатор процесса репликации | Строка (String) | Да |
| node | Узел кластера, на котором выполняется задание | Строка (String) | Да |
| source | Источник репликации | Строка (String) | Да |
| target | Цель репликации | Строка (String) | Да |
| start_time | Временная метка начала репликации | Строка (String) | Да |

Коды состояния отображает Таблица 67.

**Таблица 67 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображают Рисунок 49 и Рисунок 50. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 1690
Content-Type: application/json
Date: Sat, 29 Apr 2017 05:05:16 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "jobs": [
        {
            "database": "_replicator",
            "doc_id": "cdyno-0000001-0000003",
            "history": [
                {
                    "timestamp": "2017-04-29T05:01:37Z",
                    "type": "started"
                },
                {
                    "timestamp": "2017-04-29T05:01:37Z",
                    "type": "added"
                }
            ],
            "id": "8f5b1bd0be6f9166ccfd36fc8be8fc22+continuous",
            "info": {
                "changes_pending": 0,
                "checkpointed_source_seq": "113-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE01ygQLsZsYGqcamiZjKcRqRxwIkGRqA1H-
oSbZgk1KMLCzTDE0wdWUBAF6HJIQ",
                "doc_write_failures": 0,
                "docs_read": 113,
                "docs_written": 113,
                "missing_revisions_found": 113,
                "revisions_checked": 113,
                "source_seq": "113-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE01ygQLsZsYGqcamiZjKcRqRxwIkGRqA1H-
oSbZgk1KMLCzTDE0wdWUBAF6HJIQ",
                "through_seq": "113-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE01ygQLsZsYGqcamiZjKcRqRxwIkGRqA1H-
oSbZgk1KMLCzTDE0wdWUBAF6HJIQ"
            },
            "node": "node1@127.0.0.1",
            "pid": "<0.1850.0>",
            "source": "http://myserver.com/foo",
            "start_time": "2017-04-29T05:01:37Z",
            "target": "http://adm:*****@localhost:15984/cdyno-0000003/",
            "user": null
        },
        {
            "database": "_replicator",
            "doc_id": "cdyno-0000001-0000002",
            "history": [
                {
                    "timestamp": "2017-04-29T05:01:37Z",
                    "type": "started"
                },
                {
                    "timestamp": "2017-04-29T05:01:37Z",
                    "type": "added"
                }
            ],
```

**Ответ (начало)**
**Рисунок 49**

```
            "id": "e327d79214831ca4c11550b4a453c9ba+continuous",
            "info": {
                "changes_pending": null,
                "checkpointed_source_seq": 0,
                "doc_write_failures": 0,
                "docs_read": 12,
                "docs_written": 12,
                "missing_revisions_found": 12,
                "revisions_checked": 12,
                "source_seq": "12-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE1lzgQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSexgk4yMkhITjS0wdWUBADfEJBg",
                "through_seq": "12-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE1lzgQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSexgk4yMkhITjS0wdWUBADfEJBg"
            },
            "node": "node2@127.0.0.1",
            "pid": "<0.1757.0>",
            "source": "http://myserver.com/foo",
            "start_time": "2017-04-29T05:01:37Z",
            "target": "http://adm:*****@localhost:15984/cdyno-0000002/",
            "user": null
        }
    ],
    "offset": 0,
    "total_rows": 2
}
```


**Ответ (окончание)**
**Рисунок 50**

### 4.3.12 Получить список состояний документов репликации

Список состояний документов репликации. Содержит информацию обо всех документах, даже в состояниях завершено (completed) и неудачно (failed). Для каждого документа возвращается идентификатор документа, база данных, идентификатор репликации, источник и цель, а также другая информация.

#### 4.3.12.1. Формат запроса

Параметры запроса отображает Таблица 68.

**Таблица 68 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_scheduler/docs |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 69.

**Таблица 69 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 70.

**Таблица 70 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| limit | Количество возвращаемых результатов. | Число (int64) | Нет |
| skip | Количество пропускаемых результатов, начиная с начала, упорядоченных по идентификатору репликации. | Число (int64) | Нет |

Пример запроса отображает Рисунок 51.

```
GET /_scheduler/docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 51**

### 4.3.12.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 71.

**Таблица 71 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | |

Объекты JSON ответа отображает Таблица 72.

**Таблица 72 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Количество пропущенных результатов. | Число (int64) | Да |
| total_rows | Общее количество заданий репликации. | Число (int64) | Да |
| id | Идентификатор репликации | Строка (String) | Да |
| database | База данных документов репликации | Строка (String) | Да |
| state | Одно из следующих состояний: initializing, running, completed, pending, crashing, error, failed. | Строка (String) | Да |
| doc_id | Идентификатор документа репликации | Строка (String) | Да |
| node | Узел кластера, на котором выполняется задание | Строка (String) | Да |
| source | Источник репликации | Строка (String) | Да |
| target | Цель репликации | Строка (String) | Да |
| start_time | Временная метка начала репликации | Строка (String) | Да |
| last_updated | Временная метка последнего обновления состояния | Строка (String) | Да |
| info | Содержит дополнительную информацию о состоянии. Для ошибок это будет объект с полем "error" и строковым значением. Для состояний успеха информация указана далее | Объект (Json Object) | Да |
| error_count | Количество ошибок. Показывает, сколько раз подряд произошел сбой репликации. Репликация будет повторена с экспоненциальным отступлением, основанным на этом числе. При успешной репликации это число сбрасывается в 0. Этот параметр можно использовать для получения представления о том, почему конкретная репликация не продвигается | Число (int64) | Да |

Информационное поле документа планировщика отображает Таблица 73.

**Таблица 73 –JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| revisions_checked | Количество ревизий, которые были проверены с момента начала этой репликации | Число (int64) | Да |
| missing_revisions_found | Количество ревизий, которые были найдены на источнике, но отсутствуют в цели | Число (int64) | Да |
| docs_read | Количество документов, которые были прочитаны из источника | Число (int64) | Да |
| docs_written | Количество документов, которые были записаны в цель | Число (int64) | Да |
| changes_pending | Количество изменений, которые еще не реплицированы | Число (int64) | Да |
| doc_write_failures | Количество документов, которые не удалось записать на цель | Число (int64) | Да |
| checkpointed_source_seq | Идентификатор последовательности источника, который был успешно реплицирован в последний раз | Объект (Json Object) | Да |

Коды состояния отображает Таблица 74.

**Таблица 74 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображают Рисунок 52 и Рисунок 53. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Sat, 29 Apr 2017 05:10:08 GMT
Server: Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked

{
    "docs": [
        {
            "database": "_replicator",
            "doc_id": "cdyno-0000001-0000002",
            "error_count": 0,
            "id": "e327d79214831ca4c11550b4a453c9ba+continuous",
            "info": {
                "changes_pending": 15,
                "checkpointed_source_seq": "60-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYEyVygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSSpgk4yMkhITjS0wdWUBAENCJEg",
                "doc_write_failures": 0,
                "docs_read": 67,
                "docs_written": 67,
                "missing_revisions_found": 67,
                "revisions_checked": 67,
                "source_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8",
                "through_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8"
            },
            "last_updated": "2017-04-29T05:01:37Z",
            "node": "node2@127.0.0.1",
            "source_proxy": null,
            "target_proxy": null,
            "source": "http://myserver.com/foo",
            "start_time": "2017-04-29T05:01:37Z",
            "state": "running",
            "target": "http://adm:*****@localhost:15984/cdyno-0000002/"
        },
        {
            "database": "_replicator",
            "doc_id": "cdyno-0000001-0000003",
            "error_count": 0,
```

**Ответ**

**Рисунок 52**

```
            "id": "8f5b1bd0be6f9166ccfd36fc8be8fc22+continuous",
            "info": {
                "changes_pending": null,
                "checkpointed_source_seq": 0,
                "doc_write_failures": 0,
                "docs_read": 12,
                "docs_written": 12,
                "missing_revisions_found": 12,
                "revisions_checked": 12,
                "source_seq": "12-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE1lzgQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSexgk4yMkhITjS0wdWUBADfEJBg",
                "through_seq": "12-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE1lzgQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSexgk4yMkhITjS0wdWUBADfEJBg"
            },
            "last_updated": "2017-04-29T05:01:37Z",
            "node": "node1@127.0.0.1",
            "source_proxy": null,
            "target_proxy": null,
            "source": "http://myserver.com/foo",
            "start_time": "2017-04-29T05:01:37Z",
            "state": "running",
            "target": "http://adm:*****@localhost:15984/cdyno-0000003/"
        }
    ],
    "offset": 0,
    "total_rows": 2
}
```


**Ответ**

**Рисунок 53**

### 4.3.13 Получить информацию о документах репликации из базы данных репликатора

Получение информации о документах репликации из базы данных репликатора. По умолчанию используется база данных репликаторов _replicator, но могут существовать и другие базы данных репликаторов, если их имя заканчивается суффиксом /_replicator.

#### 4.3.13.1. Формат запроса

Параметры запроса отображает Таблица 75.

**Таблица 75 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_scheduler/docs/{replicator_db} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 76.

**Таблица 76 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 77.

**Таблица 77 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| limit | Количество возвращаемых результатов. | Число (int64) | Нет |
| skip | Количество пропускаемых результатов, начиная с начала, упорядоченных по идентификатору репликации. | Число (int64) | Нет |

Пример запроса отображает Рисунок 54.

```
GET /_scheduler/docs/other/_replicator HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 54**

### 4.3.13.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 78.

**Таблица 78 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 79.

**Таблица 79 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Количество пропущенных результатов. | Число (int64) | Да |
| total_rows | Общее количество заданий репликации. | Число (int64) | Да |
| id | Идентификатор репликации | Строка (String) | Да |
| database | База данных документов репликации | Строка (String) | Да |
| state | Одно из следующих состояний: initializing, running, completed, pending, crashing, error, failed. | Строка (String) | Да |
| doc_id | Идентификатор документа репликации | Строка (String) | Да |
| node | Узел кластера, на котором выполняется задание | Строка (String) | Да |
| source | Источник репликации | Строка (String) | Да |
| target | Цель репликации | Строка (String) | Да |
| start_time | Временная метка начала репликации | Строка (String) | Да |
| last_updated | Временная метка последнего обновления состояния | Строка (String) | Да |
| info | Содержит дополнительную информацию о состоянии. Для ошибок это будет объект с полем "error" и строковым значением. Для состояний успеха информация приведена далее | объект | Да |
| error_count | Количество ошибок. Показывает, сколько раз подряд произошел сбой репликации. Репликация будет повторена с экспоненциальным отступлением, основанным на этом числе. При успешной репликации это число сбрасывается в 0. Этот параметр можно использовать для получения представления о том, почему конкретная репликация не продвигается. | Число (int64) | Да |

Информационное поле документа планировщика отображает Таблица 80.

**Таблица 80 –JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| revisions_checked | Количество ревизий, которые были проверены с момента начала этой репликации. | Число (int64) | Да |
| missing_revisions_found | Количество ревизий, которые были найдены на источнике, но отсутствуют в цели. | Число (int64) | Да |
| docs_read | Количество документов, которые были прочитаны из источника. | Число (int64) | Да |
| docs_written | Количество документов, которые были записаны в цель. | Число (int64) | Да |
| changes_pending | Количество изменений, которые еще не реплицированы. | Число (int64) | Да |
| doc_write_failures | Количество документов, которые не удалось записать на цель. | Число (int64) | Да |
| checkpointed_source_seq | Идентификатор последовательности источника, который был успешно реплицирован в последний раз. | объект | Да |

Коды состояния отображает Таблица 81.

**Таблица 81 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 55. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Sat, 29 Apr 2017 05:10:08 GMT
Server: Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked

{
    "docs": [
        {
            "database": "other/_replicator",
            "doc_id": "cdyno-0000001-0000002",
            "error_count": 0,
            "id": "e327d79214831ca4c11550b4a453c9ba+continuous",
            "info": {
                "changes_pending": 0,
                "checkpointed_source_seq": "60-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYEyVygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSSpgk4yMkhITjS0wdWUBAENCJEg",
                "doc_write_failures": 0,
                "docs_read": 67,
                "docs_written": 67,
                "missing_revisions_found": 67,
                "revisions_checked": 67,
                "source_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8",
                "through_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8"
            },
            "last_updated": "2017-04-29T05:01:37Z",
            "node": "node2@127.0.0.1",
            "source_proxy": null,
            "target_proxy": null,
            "source": "http://myserver.com/foo",
            "start_time": "2017-04-29T05:01:37Z",
            "state": "running",
            "target": "http://adm:*****@localhost:15984/cdyno-0000002/"
        }
    ],
    "offset": 0,
    "total_rows": 1
}
```

**Ответ**

**Рисунок 55**

### 4.3.14 Получить информацию о конкретном документе репликации

#### 4.3.14.1. Формат запроса

Параметры запроса отображает Таблица 82.

**Таблица 82 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_scheduler/docs/{replicator_db}/{docid} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 83.

**Таблица 83 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 56.

```
GET /_scheduler/docs/other/_replicator/cdyno-0000001-0000002 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 56**

#### 4.3.14.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 84.

**Таблица 84 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 85.

**Таблица 85 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | ID репликации, или null, если состояние завершено или не выполнено | Строка (String) | Да |
| state | Одно из следующих состояний: инициализация (initializing), выполнение (running), завершено (completed), ожидание (pending), сбой (crashing), ошибка (error), не выполнено (failed) | Строка (String) | Да |
| database | База данных, из которой был получен документ репликации | Строка (String) | Да |
| doc_id | Идентификатор документа репликации | Строка (String) | Да |
| node | Узел кластера, на котором выполняется задание | Строка (String) | Да |
| source | Источник репликации | Строка (String) | Да |
| target | Цель репликации | Строка (String) | Да |
| start_time | Временная метка начала репликации | Строка (String) | Да |
| last_updated | Временная метка последнего обновления состояния | Строка (String) | Да |
| info | Содержит дополнительную информацию о состоянии. Для ошибок это будет объект с полем "error" и строковым значением. Для состояний успеха информация приведена далее | объект | Да |
| error_count | Количество ошибок. Показывает, сколько раз подряд произошел сбой репликации. Репликация будет повторена с экспоненциальным отступлением, основанным на этом числе. При успешной репликации это число сбрасывается в 0. Этот параметр можно использовать для получения представления о том, почему конкретная репликация не продвигается. | Число (int64) | Да |

Информационное поле документа планировщика отображает Таблица 86.

**Таблица 86 –JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| revisions_checked | Количество ревизий, которые были проверены с момента начала этой репликации. | Число (int64) | Да |
| missing_revisions_found | Количество ревизий, которые были найдены на источнике, но отсутствуют в цели. | Число (int64) | Да |
| docs_read | Количество документов, которые были прочитаны из источника. | Число (int64) | Да |
| docs_written | Количество документов, которые были записаны в цель. | Число (int64) | Да |
| changes_pending | Количество изменений, которые еще не реплицированы. | Число (int64) | Да |
| doc_write_failures | Количество документов, которые не удалось записать на цель. | Число (int64) | Да |
| checkpointed_source_seq | Идентификатор последовательности источника, который был успешно реплицирован в последний раз. | объект | Да |

Коды состояния отображает Таблица 87.

**Таблица 87 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 57. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Sat, 29 Apr 2017 05:10:08 GMT
Server: Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked

{
    "database": "other/_replicator",
    "doc_id": "cdyno-0000001-0000002",
    "error_count": 0,
    "id": "e327d79214831ca4c11550b4a453c9ba+continuous",
    "info": {
        "changes_pending": 0,
        "checkpointed_source_seq": "60-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYEyVygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSSpgk4yMkhITjS0wdWUBAENCJEg",
        "doc_write_failures": 0,
        "docs_read": 67,
        "docs_written": 67,
        "missing_revisions_found": 67,
        "revisions_checked": 67,
        "source_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8",
        "through_seq": "67-
g1AAAACTeJzLYWBgYMpgTmHgz8tPSTV0MDQy1zMAQsMckEQiQ1L9____szKYE2VygQLsBsZm5pZJJpjKcRqRxwIkGRqA1H-
oSepgk4yMkhITjS0wdWUBAEVKJE8"
    },
    "last_updated": "2017-04-29T05:01:37Z",
    "node": "node2@127.0.0.1",
    "source_proxy": null,
    "target_proxy": null,
    "source": "http://myserver.com/foo",
    "start_time": "2017-04-29T05:01:37Z",
    "state": "running",
    "target": "http://adm:*****@localhost:15984/cdyno-0000002/"
}
```

**Ответ**

**Рисунок 57**

### 4.3.15 Получение информации о конечной точке /_node/{node-name}

Конечная точка /_node/{node-name} может быть использована для подтверждения имени узла Erlang сервера, обрабатывающего запрос. Это наиболее полезно при обращении к /_node/_local для получения этой информации. Повторное получение этой информации для конечной точки БД «Енисей»может быть полезно для определения того, правильно ли кластер БД «Енисей» проксируется через обратный балансировщик нагрузки.

#### 4.3.15.1. Формат запроса

Параметры запроса отображает Таблица 88.

**Таблица 88 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 89.

**Таблица 89 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 58.

```
GET /_node/_local HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 58**

#### 4.3.15.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 90.

**Таблица 90 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 91.

**Таблица 91 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 59. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 27
Content-Type: application/json
Date: Tue, 28 Jan 2020 19:25:51 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Request-ID: 5b8db6c677
X-CouchDB-Body-Time: 0

{"name":"node1@127.0.0.1"}
```

**Ответ**
**Рисунок 59**

### 4.3.16 Получить статистику для запущенного сервера

#### 4.3.16.1. Формат запроса

Параметры запроса отображает Таблица 92.

**Таблица 92 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_stats |
| Метод | GET |

Ресурс _stats возвращает объект JSON, содержащий статистику для запущенного сервера. Объект структурирован по секциям верхнего уровня, объединяющими статистику для ряда записей, при этом каждая отдельная статистика легко идентифицируется, а содержание каждой статистики является самоописывающимся.

Статистика отбирается по внутренней выборке с настраиваемым интервалом. При мониторинге конечной точки _stats для получения точных результатов необходимо использовать частоту опроса не менее чем в два раза выше. Например, если интервал составляет 10 секунд, опрашивайте _stats не реже, чем каждые 5 секунд.

Литеральная строка _local служит псевдонимом для имени локального узла, поэтому для всех URL-адресов stats {node-name} можно заменить на _local, чтобы взаимодействовать со статистикой локального узла.

Параметры HTTP заголовка запроса отображает Таблица 93.

**Таблица 93 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 60.

```
GET /_node/_local/_stats/couchdb/request_time HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 60**

#### 4.3.16.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 94.

**Таблица 94 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 95.

**Таблица 95 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 61. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 187
Content-Type: application/json
Date: Sat, 10 Aug 2013 11:41:11 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
  "value": {
    "min": 0,
    "max": 0,
    "arithmetic_mean": 0,
    "geometric_mean": 0,
    "harmonic_mean": 0,
    "median": 0,
    "variance": 0,
    "standard_deviation": 0,
    "skewness": 0,
    "kurtosis": 0,
    "percentile": [
      [
        50,
        0
      ],
      [
        75,
        0
      ],
      [
        90,
        0
      ],
      [
        95,
        0
      ],
      [
        99,
        0
      ],
      [
        999,
        0
      ]
    ],
    "histogram": [
      [
        0,
        0
      ]
    ],
    "n": 0
  },
  "type": "histogram",
  "desc": "length of a request inside CouchDB without MochiWeb"
}
```

**Ответ**
**Рисунок 61**

В полях указывается текущее, минимальное и максимальное значения, а также набор статистических средств и величин. Количество в каждом случае не определено, но приведенные ниже описания обеспечивают достаточную детализацию для определения единиц измерения.

Статистика представляется по «группам», разделы и тип статистики отображает подпункт 5.3.1.3.

Можно также получить доступ к отдельным статистикам, указывая разделы статистики и ID статистики как часть пути URL. Например, чтобы получить статистику request_time в разделе БД «Енисей» для целевого узла, можно использовать:

GET /_node/_local/_stats/couchdb/request_time HTTP/1.1

Возвращает целый объект статистики, как и в случае полного запроса, но содержащий только запрошенную индивидуальную статистику.

### 4.3.17 Получить ответ в формате text/plain с помощью ресурса _prometheus

Ресурс _prometheus возвращает ответ в формате text/plain, который объединяет конечные точки /_node/{node-name}/_stats и /_node/{node-name}/_system.

#### 4.3.17.1. Формат запроса

Параметры запроса отображает Таблица 96.

**Таблица 96 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_prometheus |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 97.

**Таблица 97 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 62.

```
GET /_node/_local/_prometheus HTTP/1.1
Accept: text/plain
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 62**

Если указана дополнительная опция конфигурации порта, то клиент может вызвать этот ППИ, используя порт, который не требует аутентификации. По умолчанию эта опция имеет значение false``(OFF). При значении ``true (ON), порты по умолчанию для 3-узлового кластера 17986, 27986, 37986.

Пример запроса отображает Рисунок 63.

```
GET /_node/_local/_prometheus HTTP/1.1
Accept: text/plain
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:17986
```

**Запрос**
**Рисунок 63**

#### 4.3.17.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 98.

**Таблица 98 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 99.

**Таблица 99 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображают Рисунок 64, Рисунок 65 и Рисунок 66. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 187
Content-Type: text/plain; version=2.0
Date: Sat, 10 May 2020 11:41:11 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

# TYPE couchdb_couch_log_requests_total counter
couchdb_couch_log_requests_total{level="alert"} 0
couchdb_couch_log_requests_total{level="critical"} 0
couchdb_couch_log_requests_total{level="debug"} 0
couchdb_couch_log_requests_total{level="emergency"} 0
couchdb_couch_log_requests_total{level="error"} 0
couchdb_couch_log_requests_total{level="info"} 8
couchdb_couch_log_requests_total{level="notice"} 51
couchdb_couch_log_requests_total{level="warning"} 0
# TYPE couchdb_couch_replicator_changes_manager_deaths_total counter
couchdb_couch_replicator_changes_manager_deaths_total 0
# TYPE couchdb_couch_replicator_changes_queue_deaths_total counter
couchdb_couch_replicator_changes_queue_deaths_total 0
# TYPE couchdb_couch_replicator_changes_read_failures_total counter
couchdb_couch_replicator_changes_read_failures_total 0
# TYPE couchdb_couch_replicator_changes_reader_deaths_total counter
couchdb_couch_replicator_changes_reader_deaths_total 0
# TYPE couchdb_couch_replicator_checkpoints_failure_total counter
couchdb_couch_replicator_checkpoints_failure_total 0
# TYPE couchdb_couch_replicator_checkpoints_total counter
couchdb_couch_replicator_checkpoints_total 0
# TYPE couchdb_couch_replicator_cluster_is_stable gauge
couchdb_couch_replicator_cluster_is_stable 1
# TYPE couchdb_couch_replicator_connection_acquires_total counter
couchdb_couch_replicator_connection_acquires_total 0
# TYPE couchdb_couch_replicator_connection_closes_total counter
couchdb_couch_replicator_connection_closes_total 0
# TYPE couchdb_couch_replicator_connection_creates_total counter
couchdb_couch_replicator_connection_creates_total 0
# TYPE couchdb_couch_replicator_connection_owner_crashes_total counter
couchdb_couch_replicator_connection_owner_crashes_total 0
# TYPE couchdb_couch_replicator_connection_releases_total counter
couchdb_couch_replicator_connection_releases_total 0
# TYPE couchdb_couch_replicator_connection_worker_crashes_total counter
couchdb_couch_replicator_connection_worker_crashes_total 0
# TYPE couchdb_couch_replicator_db_scans_total counter
couchdb_couch_replicator_db_scans_total 1
# TYPE couchdb_couch_replicator_docs_completed_state_updates_total counter
couchdb_couch_replicator_docs_completed_state_updates_total 0
# TYPE couchdb_couch_replicator_docs_db_changes_total counter
couchdb_couch_replicator_docs_db_changes_total 0
# TYPE couchdb_couch_replicator_docs_dbs_created_total counter
couchdb_couch_replicator_docs_dbs_created_total 0
# TYPE couchdb_couch_replicator_docs_dbs_deleted_total counter
couchdb_couch_replicator_docs_dbs_deleted_total 0
# TYPE couchdb_couch_replicator_docs_dbs_found_total counter
couchdb_couch_replicator_docs_dbs_found_total 2
# TYPE couchdb_couch_replicator_docs_failed_state_updates_total counter
couchdb_couch_replicator_docs_failed_state_updates_total 0
# TYPE couchdb_couch_replicator_failed_starts_total counter
couchdb_couch_replicator_failed_starts_total 0
# TYPE couchdb_couch_replicator_jobs_adds_total counter
couchdb_couch_replicator_jobs_adds_total 0
```

**Ответ (начало)**
**Рисунок 64**

```
# TYPE couchdb_couch_replicator_jobs_crashed gauge
couchdb_couch_replicator_jobs_crashed 0
# TYPE couchdb_couch_replicator_jobs_crashes_total counter
couchdb_couch_replicator_jobs_crashes_total 0
# TYPE couchdb_couch_replicator_jobs_duplicate_adds_total counter
couchdb_couch_replicator_jobs_duplicate_adds_total 0
# TYPE couchdb_couch_replicator_jobs_pending gauge
couchdb_couch_replicator_jobs_pending 0
# TYPE couchdb_couch_replicator_jobs_removes_total counter
couchdb_couch_replicator_jobs_removes_total 0
# TYPE couchdb_couch_replicator_jobs_running gauge
couchdb_couch_replicator_jobs_running 0
# TYPE couchdb_couch_replicator_jobs_starts_total counter
couchdb_couch_replicator_jobs_starts_total 0
# TYPE couchdb_couch_replicator_jobs_stops_total counter
couchdb_couch_replicator_jobs_stops_total 0
# TYPE couchdb_couch_replicator_jobs_total gauge
couchdb_couch_replicator_jobs_total 0
# TYPE couchdb_couch_replicator_requests_total counter
couchdb_couch_replicator_requests_total 0
# TYPE couchdb_couch_replicator_responses_failure_total counter
couchdb_couch_replicator_responses_failure_total 0
# TYPE couchdb_couch_replicator_responses_total counter
couchdb_couch_replicator_responses_total 0
# TYPE couchdb_couch_replicator_stream_responses_failure_total counter
couchdb_couch_replicator_stream_responses_failure_total 0
# TYPE couchdb_couch_replicator_stream_responses_total counter
couchdb_couch_replicator_stream_responses_total 0
# TYPE couchdb_couch_replicator_worker_deaths_total counter
couchdb_couch_replicator_worker_deaths_total 0
# TYPE couchdb_couch_replicator_workers_started_total counter
couchdb_couch_replicator_workers_started_total 0
# TYPE couchdb_auth_cache_requests_total counter
couchdb_auth_cache_requests_total 0
# TYPE couchdb_auth_cache_misses_total counter
couchdb_auth_cache_misses_total 0
# TYPE couchdb_collect_results_time_seconds summary
couchdb_collect_results_time_seconds{quantile="0.5"} 0.0
couchdb_collect_results_time_seconds{quantile="0.75"} 0.0
couchdb_collect_results_time_seconds{quantile="0.9"} 0.0
couchdb_collect_results_time_seconds{quantile="0.95"} 0.0
couchdb_collect_results_time_seconds{quantile="0.99"} 0.0
couchdb_collect_results_time_seconds{quantile="0.999"} 0.0
couchdb_collect_results_time_seconds_sum 0.0
couchdb_collect_results_time_seconds_count 0
# TYPE couchdb_couch_server_lru_skip_total counter
couchdb_couch_server_lru_skip_total 0
# TYPE couchdb_database_purges_total counter
couchdb_database_purges_total 0
# TYPE couchdb_database_reads_total counter
couchdb_database_reads_total 0
# TYPE couchdb_database_writes_total counter
couchdb_database_writes_total 0
```

**Ответ (продолжение)**
**Рисунок 65**

```
# TYPE couchdb_db_open_time_seconds summary
couchdb_db_open_time_seconds{quantile="0.5"} 0.0
couchdb_db_open_time_seconds{quantile="0.75"} 0.0
couchdb_db_open_time_seconds{quantile="0.9"} 0.0
couchdb_db_open_time_seconds{quantile="0.95"} 0.0
couchdb_db_open_time_seconds{quantile="0.99"} 0.0
couchdb_db_open_time_seconds{quantile="0.999"} 0.0
couchdb_db_open_time_seconds_sum 0.0
couchdb_db_open_time_seconds_count 0
# TYPE couchdb_dbinfo_seconds summary
couchdb_dbinfo_seconds{quantile="0.5"} 0.0
couchdb_dbinfo_seconds{quantile="0.75"} 0.0
couchdb_dbinfo_seconds{quantile="0.9"} 0.0
couchdb_dbinfo_seconds{quantile="0.95"} 0.0
couchdb_dbinfo_seconds{quantile="0.99"} 0.0
couchdb_dbinfo_seconds{quantile="0.999"} 0.0
couchdb_dbinfo_seconds_sum 0.0
couchdb_dbinfo_seconds_count 0
# TYPE couchdb_document_inserts_total counter
couchdb_document_inserts_total 0
# TYPE couchdb_document_purges_failure_total counter
couchdb_document_purges_failure_total 0
# TYPE couchdb_document_purges_success_total counter
couchdb_document_purges_success_total 0
# TYPE couchdb_document_purges_total_total counter
couchdb_document_purges_total_total 0
# TYPE couchdb_document_writes_total counter
couchdb_document_writes_total 0
# TYPE couchdb_httpd_aborted_requests_total counter
couchdb_httpd_aborted_requests_total 0
# TYPE couchdb_httpd_all_docs_timeouts_total counter
couchdb_httpd_all_docs_timeouts_total 0
# TYPE couchdb_httpd_bulk_docs_seconds summary
couchdb_httpd_bulk_docs_seconds{quantile="0.5"} 0.0
couchdb_httpd_bulk_docs_seconds{quantile="0.75"} 0.0
couchdb_httpd_bulk_docs_seconds{quantile="0.9"} 0.0
couchdb_httpd_bulk_docs_seconds{quantile="0.95"} 0.0
couchdb_httpd_bulk_docs_seconds{quantile="0.99"} 0.0
couchdb_httpd_bulk_docs_seconds{quantile="0.999"} 0.0
couchdb_httpd_bulk_docs_seconds_sum 0.0
couchdb_httpd_bulk_docs_seconds_count 0
...remaining couchdb metrics from _stats and _system
```

**Ответ (окончание)**
**Рисунок 66**

### 4.3.18 Получить статистические данные системного уровня для запущенного сервера

Ресурс _system возвращает объект JSON, содержащий различные статистические данные системного уровня для запущенного сервера. Объект структурирован по секциям верхнего уровня, в которых собрана статистика для ряда записей, при этом каждая отдельная статистика легко идентифицируется, а содержание каждой статистики является самоописывающимся.

Литеральная строка _local служит псевдонимом для имени локального узла, поэтому для всех URL статистики {node-name} можно заменить на _local, чтобы взаимодействовать со статистикой локального узла.

#### 4.3.18.1. Формат запроса

Параметры запроса отображает Таблица 100.

**Таблица 100 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_system |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 101.

**Таблица 101 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 67.

```
GET /_node/_local/_system HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 67**

#### 4.3.18.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 102.

**Таблица 102 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 103.

**Таблица 103 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 68. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 187
Content-Type: application/json
Date: Sat, 10 Aug 2013 11:41:11 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
  "uptime": 259,
  "memory": {
  ...
}
```

**Ответ**
**Рисунок 68**

### 4.3.19 Перезапустить /_node/{node-name}

Этот ППИ предназначен только для упрощения интеграционного тестирования и не предназначен для использования в производстве.

#### 4.3.19.1. Формат запроса

Параметры запроса отображает Таблица 104.

**Таблица 104 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_restart |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 105.

**Таблица 105 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 69.

```
POST /_node/_local/_restart HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 69**

#### 4.3.19.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 106.

**Таблица 106 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 107.

**Таблица 107 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 70. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Sat, 10 Aug 2013 11:41:11 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true
}
```

**Ответ**
**Рисунок 70**

### 4.3.20 Проверяет результаты токенизации анализатора Lucene

Конечные точки поиска требуют наличия запущенного поискового плагина, подключенного к каждому узлу кластера

Проверяет результаты токенизации анализатора Lucene на образце текста.

#### 4.3.20.1. Формат запроса

Параметры запроса отображает Таблица 108.

**Таблица 108 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_search_analyze |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 109.

**Таблица 109 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры тела запроса отображает Таблица 110.

**Таблица 110 – Параметры тела запроса**

| Параметр | Описание |
|---|---|
| field | Проверяет результаты токенизации анализатора |
| text | Токен анализатора, который необходимо проверить |

Пример запроса отображает Рисунок 71.

```
POST /_search_analyze HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
Content-Type: application/json

{"analyzer":"english", "text":"running"}
```

**Запрос**
**Рисунок 71**

#### 4.3.20.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 111.

**Таблица 111 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 112.

**Таблица 112 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Тело запроса неверно (неправильно сформировано или отсутствует одно из обязательных полей) |
| 500 Internal Server Error | Произошла ошибка сервера (или ошибка другого рода). |

Ответ в случае успешного завершения запроса отображает Рисунок 72. Коды ошибок приведены в пункте 6.1.2.

```
{
    "tokens": [
        "run"
    ]
}
```

**Ответ**
**Рисунок 72**

### 4.3.21 Получить подтверждение работы сервера

Подтверждает, что сервер работает, запущен и готов отвечать на запросы. Если maintenance_mode равен true или nolb, конечная точка вернет ответ 404.

#### 4.3.21.1. Формат запроса

Параметры запроса отображает Таблица 113.

**Таблица 113 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_up |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 114.

**Таблица 114 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 73.

```
GET /_up HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 73**

#### 4.3.21.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 115.

**Таблица 115 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Коды состояния отображает Таблица 116.

**Таблица 116 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 404 Not Found | Сервер в данный момент недоступен для запросов |

Ответ в случае успешного завершения запроса отображает Рисунок 74. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 16
Content-Type: application/json
Date: Sat, 17 Mar 2018 04:46:26 GMT
Server: CouchDB/2.2.0-f999071ec (Erlang OTP/19)
X-Couch-Request-ID: c57a3b2787
X-CouchDB-Body-Time: 0

{
    "status":"ok"
}
```

**Ответ**
**Рисунок 74**

### 4.3.22 Получить уникальные идентификаторы

Запрашивает один или несколько уникальных идентификаторов (UUID) у экземпляра БД «Енисей». Ответом является объект JSON, содержащий список UUID.

#### 4.3.22.1. Формат запроса

Параметры запроса отображает Таблица 117.

**Таблица 117 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_uuids |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 118.

**Таблица 118 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 119.

**Таблица 119 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| count | Количество возвращаемых UUID. По умолчанию 1. | Число (int64) | Нет |

Пример запроса отображает Рисунок 75.

```
GET /_uuids?count=10 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 75**

Тип UUID определяется настройкой алгоритма UUID в конфигурации БД «Енисей».

Тип UUID может быть изменен в любое время через ППИ конфигурации. Например, тип UUID может быть изменен на случайный путем отправки этого HTTP-запроса, который отображает Рисунок 76

```
PUT http://couchdb:5984/_node/nonode@nohost/_config/uuids/algorithm HTTP/1.1
Content-Type: application/json
Accept: */*
Authorization: Basic YWRtaW46YWRtaW4=

"random"
```

**Запрос**
**Рисунок 76**

#### 4.3.22.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 120.

**Таблица 120 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | Хэш ответа | Строка (String) | Да |
| ETag | application/json | Строка (String) | Да |

Коды состояния отображает Таблица 121.

**Таблица 121 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Запрошено больше UUID, чем разрешено получить |

Ответ в случае успешного завершения запроса отображает Рисунок 77. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Length: 362
Content-Type: application/json
Date: Sat, 10 Aug 2013 11:46:25 GMT
ETag: "DGRWWQFLUDWN5MRKSLKQ425XV"
Expires: Fri, 01 Jan 1990 00:00:00 GMT
Pragma: no-cache
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "uuids": [
        "75480ca477454894678e22eec6002413",
        "75480ca477454894678e22eec600250b",
        "75480ca477454894678e22eec6002c41",
        "75480ca477454894678e22eec6003b90",
        "75480ca477454894678e22eec6003fca",
        "75480ca477454894678e22eec6004bef",
        "75480ca477454894678e22eec600528f",
        "75480ca477454894678e22eec6005e0b",
        "75480ca477454894678e22eec6006158",
        "75480ca477454894678e22eec6006161"
    ]
}
```

**Ответ**

**Рисунок 77**

Ответ на изменение типа UUID отображает Рисунок 78.

```
{
    "uuids" : [
        "031aad7b469956cf2826fcb2a9260492",
        "6ec875e15e6b385120938df18ee8e496",
        "cff9e881516483911aa2f0e98949092d",
        "b89d37509d39dd712546f9510d4a9271",
        "2e0dbf7f6c4ad716f21938a016e4e59f"
    ]
}
```

**Ответ**
**Рисунок 78**

### 4.3.23 Получить информацию о состоянии заданий

Возвращает число завершенных, не выполненных, запущенных, остановленных и общее количество заданий, а также состояние решардинга на кластере.

#### 4.3.23.1. Формат запроса

Параметры запроса отображает Таблица 122.

**Таблица 122 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 123.

**Таблица 123 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 79.

```
GET /_reshard HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 79**

#### 4.3.23.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 124.

**Таблица 124 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

JSON объект отображает Таблица 125.

**Таблица 125 – JSON объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Остановлен (stopped) или запущен (running) | Строка (String) | Да |
| state_reason | Ноль (null) или строка, описывающая дополнительную информацию или причину, связанную с состоянием | Строка (String) | Да |
| completed | Количество завершенных заданий решардинга | Число (int64) | Да |
| failed | Количество не выполненных заданий решардинга | Число (int64) | Да |
| running | Количество запущенных заданий решардинга | Число (int64) | Да |
| stopped | Счетчик остановленных заданий решардинга | Число (int64) | Да |
| total | Общее количество заданий решардинга | Число (int64) | Да |

Коды состояния отображает Таблица 126.

**Таблица 126 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 80. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "completed": 21,
    "failed": 0,
    "running": 3,
    "state": "running",
    "state_reason": null,
    "stopped": 0,
    "total": 24
}
```

**Ответ**
**Рисунок 80**

### 4.3.24 Получить информацию о состоянии решардинга

Возвращает состояние решардинга и необязательную информацию о состоянии.

#### 4.3.24.1. Формат запроса

Параметры запроса отображает Таблица 127.

**Таблица 127 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/state |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 128.

**Таблица 128 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 81.

```
GET /_reshard/state HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 81**

#### 4.3.24.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 129.

**Таблица 129 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 130.

**Таблица 130 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Остановлен (stopped) или запущен (running) | Строка (String) | Да |
| state_reason | Дополнительная информация или причина, связанная с состоянием. | Строка (String) | Да |

Коды состояния отображает Таблица 131.

**Таблица 131 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 82. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "reason": null,
    "state": "running"
}
```

**Ответ**

**Рисунок 82**

### 4.3.25 Изменить состояние решардинга на кластере

Изменение состояния решардинга на кластере. Состояния могут быть остановлен (stopped) или запущен (running). Это запускает и останавливает глобальную решардинг на всех узлах кластера. Если есть работающие задания, они будут остановлены, когда состояние изменится на остановленное. Когда состояние снова изменится на запущен (running), эти задания продолжат выполняться.

#### 4.3.25.1. Формат запроса

Параметры запроса отображает Таблица 132.

**Таблица 132 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/state |
| Метод | PUT |

Параметры HTTP заголовка запроса отображает Таблица 133.

**Таблица 133 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 134.

**Таблица 134 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Остановлен (stopped) или запущен (running) | Строка (String) | Да |
| state_reason | Дополнительная информация или причина, связанная с состоянием. | Строка (String) | Да |

Пример запроса отображает Рисунок 83.

```
PUT /_reshard/state HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "state": "stopped",
    "reason": "Rebalancing in progress"
}
```

**Запрос**
**Рисунок 83**

#### 4.3.25.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 135.

**Таблица 135 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 136.

**Таблица 136 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | true | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 137.

**Таблица 137 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверный запрос. Может быть плохое или отсутствующее имя состояния |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 84. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "ok": true
}
```

**Ответ**
**Рисунок 84**

### 4.3.26 Получить информацию о состоянии заданий решардинга

Форма ответа и, в частности, поле total_rows и offset должны соответствовать конечной точке _scheduler/jobs.

#### 4.3.26.1. Формат запроса

Параметры запроса отображает Таблица 138.

**Таблица 138 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 139.

**Таблица 139 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 85.

```
GET /_reshard/jobs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
```

**Запрос**
**Рисунок 85**

#### 4.3.26.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 140.

**Таблица 140 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 141.

**Таблица 141 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| jobs | Массив json-объектов, по одному на каждое задание решардинга. Поля каждого задания указаны в конечной точке /_reshard/job/{jobid}. | Список (List) | Да |
| offset | Смещение в объекте списка заданий. В настоящее время закодировано на 0. | Число (int64) | Да |
| total_rows | Общее количество заданий решардинга на кластере. | Число (int64) | Да |

Коды состояния отображает Таблица 142.

**Таблица 142 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 86. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "jobs": [
        {
            "history": [
                {
                    "detail": null,
                    "timestamp": "2019-03-28T15:28:02Z",
                    "type": "new"
                },
                {
                    "detail": "initial_copy",
                    "timestamp": "2019-03-28T15:28:02Z",
                    "type": "running"
                },
                ...
            ],
            "id": "001-171d1211418996ff47bd610b1d1257fc4ca2628868def4a05e63e8f8fe50694a",
            "job_state": "completed",
            "node": "node1@127.0.0.1",
            "source": "shards/00000000-1fffffff/d1.1553786862",
            "split_state": "completed",
            "start_time": "2019-03-28T15:28:02Z",
            "state_info": {},
            "target": [
                "shards/00000000-0fffffff/d1.1553786862",
                "shards/10000000-1fffffff/d1.1553786862"
            ],
            "type": "split",
            "update_time": "2019-03-28T15:28:08Z"
        },
        ...
    ],
    "offset": 0,
    "total_rows": 24
}
```

**Ответ**

**Рисунок 86**

### 4.3.27 Получить информацию о задании решардинга

Получение информации о задании решардинга, идентифицированном по jobid.

#### 4.3.27.1. Формат запроса

Параметры запроса отображает Таблица 143.

**Таблица 143 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs/{jobid} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 144.

**Таблица 144 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 87.

```
GET /_reshard/jobs/001-171d1211418996ff47bd610b1d1257fc4ca2628868def4a05e63e8f8fe50694a HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
```

**Запрос**
**Рисунок 87**

### 4.3.27.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 145.

**Таблица 145 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 146.

**Таблица 146 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор задания. | Строка (String) | Да |
| type | В настоящее время реализовано только разделение (split). | Строка (String) | Да |
| job_state | Текущее состояние задания. Может быть одним из следующих: новое (new), выполняется (running), остановлено (stopped), завершено (completed) или не выполнено (failed). | Строка (String) | Да |
| split_state | Детали состояния, характерного для разделения сегментов. Указывает, насколько продвинулось разделение сегмента, и может быть одним из следующих: new, initial_copy, topoff1, build_indices, topoff2, copy_local_docs, update_shardmap, wait_source_close, topoff3, source_delete или completed. | Строка (String) | Да |
| state_info | Необязательная дополнительная информация, связанная с текущим состоянием. | Объект (Json Object) | Да |
| source | Для разделенных (split) заданий это будет исходный сегмент. | Строка (String) | Да |
| target | Для разделенных (split) заданий это будет список из двух или более целевых сегментов. | Список (List) | Да |
| history | Список json-объектов, записывающих историю перехода задания из одного состояния в другое. | Список (List) | Да |

Коды состояния отображает Таблица 147.

**Таблица 147 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 88. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{

    "id": "001-171d1211418996ff47bd610b1d1257fc4ca2628868def4a05e63e8f8fe50694a",
    "job_state": "completed",
    "node": "node1@127.0.0.1",
    "source": "shards/00000000-1fffffff/d1.1553786862",
    "split_state": "completed",
    "start_time": "2019-03-28T15:28:02Z",
    "state_info": {},
    "target": [
        "shards/00000000-0fffffff/d1.1553786862",
        "shards/10000000-1fffffff/d1.1553786862"
    ],
    "type": "split",
    "update_time": "2019-03-28T15:28:08Z",
    "history": [
        {
            "detail": null,
            "timestamp": "2019-03-28T15:28:02Z",
            "type": "new"
        },
        {
            "detail": "initial_copy",
            "timestamp": "2019-03-28T15:28:02Z",
            "type": "running"
        },
        ...
    ]
}
```

**Ответ**
**Рисунок 88**

### 4.3.28 Создать задания решардинга

В зависимости от того, какие поля указаны в запросе, будет создано одно или несколько заданий решардинга. Ответ представляет собой массив результатов в формате json. Каждый объект результата представляет собой одно задание решардинга для определенного узла и диапазона. Некоторые из ответов могут быть успешными, а некоторые неудачными. Успешные результаты будут иметь ключ и значение "ok": true, а не выполненые задания будут иметь ключ и значение "error": "{error_message}".

#### 4.3.28.1. Формат запроса

Параметры запроса отображает Таблица 148.

**Таблица 148 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 149.

**Таблица 149 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 150.

**Таблица 150 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| type | Тип задания. В настоящее время принимается только "split". | Строка (String) | Да |
| db | База данных для разделения. Это поле является взаимоисключающим с полем "shard". | Строка (String) | Да |
| node | Разделить сегменты на определенном узле. Это необязательный параметр. Значением должен быть один из узлов, возвращаемых конечной точкой _membership. | Строка (String) | Нет |
| range | Разделить копии сегментов в заданном диапазоне. Формат диапазона hhhhhhhh-hhhhhhhh, где h шестнадцатеричная цифра. Такой формат используется, поскольку именно так диапазоны представлены в файловой системе. Этот параметр необязателен и является взаимоисключающим с полем "shard". | Строка (String) | Да |
| shard | Разделить определенный сегмент. Сегмент должен быть указан как "shards/ {range}/{db}.{suffix}". Где range имеет формат hhhhhhhhhhhhhhhhhhhhhhhhhhhhhhhh, db имя базы данных, а suffix суффикс создания сегмента (временная метка). | Строка (String) | Нет |
| error | Сообщение об ошибке, если задание не может быть создано. | Строка (String) | Нет |
| node | Узел кластера, на котором было создано и выполняется задание. | Строка (String) | Нет |

Пример запроса отображает Рисунок 89.

```
POST /_reshard/jobs HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=

{
    "db": "db3",
    "range": "80000000-ffffffff",
    "type": "split"
}
```

**Запрос**

**Рисунок 89**

#### 4.3.28.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 151.

**Таблица 151 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 152.

**Таблица 152 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | true, если задание создано успешно | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 153.

**Таблица 153 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Одно или несколько заданий были успешно созданы |
| 400 Bad Request | Неверный запрос. Возможно, не прошла проверка параметров |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Не найдена база данных, узел, диапазон или сегмент |

Ответ в случае успешного завершения запроса отображает Рисунок 90. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Content-Type: application/json

[
    {
        "id": "001-30d7848a6feeb826d5e3ea5bb7773d672af226fd34fd84a8fb1ca736285df557",
        "node": "node1@127.0.0.1",
        "ok": true,
        "shard": "shards/80000000-ffffffff/db3.1554148353"
    },
    {
        "id": "001-c2d734360b4cb3ff8b3feaccb2d787bf81ce2e773489eddd985ddd01d9de8e01",
        "node": "node2@127.0.0.1",
        "ok": true,
        "shard": "shards/80000000-ffffffff/db3.1554148353"
    }
]
```

**Ответ**

**Рисунок 90**

### 4.3.29 Остановить и удалить задания решардинга

Если задание запущено, остановит его, а затем удалит.

#### 4.3.29.1. Формат запроса

Параметры запроса отображает Таблица 154.

**Таблица 154 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs/{jobid} |
| Метод | DELETE |

Параметры HTTP заголовка запроса отображает Таблица 155.

**Таблица 155 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 91.

```
DELETE /_reshard/jobs/001-171d1211418996ff47bd610b1d1257fc4ca2628868def4a05e63e8f8fe50694a HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
```

**Запрос**
**Рисунок 91**

#### 4.3.29.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 156.

**Таблица 156 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 157.

**Таблица 157 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | true, если задание создано успешно | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 158.

**Таблица 158 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Задание не найдено |

Ответ в случае успешного завершения запроса отображает Рисунок 92. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "ok": true
}
```

**Ответ**
**Рисунок 92**

### 4.3.30 Получить состояние выполнения задания решардинга

Возвращает состояние выполнения задания решардинга, идентифицированного по jobid.

#### 4.3.30.1. Формат запроса

Параметры запроса отображает Таблица 159.

**Таблица 159 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs/{jobid}/state |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 160.

**Таблица 160 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 93.

```
GET /_reshard/jobs/001-b3da04f969bbd682faaab5a6c373705cbcca23f732c386bb1a608cfbcfe9faff/state HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 93**

#### 4.3.30.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 161.

**Таблица 161 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 162.

**Таблица 162 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Одно из новое (new), выполняется (running), остановлено (stopped), завершено (completed) или не выполнено (failed). | Строка (String) | Нет |
| state_reason | Дополнительная информация, связанная с состоянием | Строка (String) | Нет |

Коды состояния отображает Таблица 163.

**Таблица 163 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Задание не найдено |

Ответ в случае успешного завершения запроса отображает Рисунок 94. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "reason": null,
    "state": "running"
}
```

**Ответ**
**Рисунок 94**

### 4.3.31 Изменить состояние задания решардинга

Изменение состояния конкретного задания решардинга, идентифицированного по jobid. Состояние может быть изменено с остановленного (stopped) на работающее (running) или с работающего (running) на остановленное (stopped). Если отдельное задание остановлено (stopped) с помощью этого ППИ, оно останется остановленным (stopped) даже после того, как глобальное состояние решардинга будет переключено с остановленного (stopped) на запущенное (running). Если задание уже завершено (completed), его состояние останется завершенным (completed).

#### 4.3.31.1. Формат запроса

Параметры запроса отображает Таблица 164.

**Таблица 164 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_reshard/jobs/{jobid}/state |
| Метод | PUT |

Параметры HTTP заголовка запроса отображает Таблица 165.

**Таблица 165 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 166.

**Таблица 166 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| state | Остановлен stopped или запущен running | Строка (String) | Да |
| state_reason | Необязательная строка, описывающая дополнительную информацию или причину, связанную с состоянием. | Строка (String) | Да |

Пример запроса отображает Рисунок 95.

```
PUT /_reshard/state/001-
b3da04f969bbd682faaab5a6c373705cbcca23f732c386bb1a608cfbcfe9faff/state HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "state": "stopped",
    "reason": "Rebalancing in progress"
}
```

**Запрос**
**Рисунок 95**

### 4.3.31.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 167.

**Таблица 167 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 168.

**Таблица 168 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | true | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 169.

**Таблица 169 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверный запрос. Возможно, неправильное имя состояния |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Задание не найдено |

Ответ в случае успешного завершения запроса отображает Рисунок 96. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
     "ok": true
}
```

**Ответ**
**Рисунок 96**

### 4.3.32 Аутентификация

Интерфейсы для получения данных сеанса и авторизации

#### 4.3.32.1. Базовая аутентификация

Базовая аутентификация (RFC 2617) — это быстрый и простой способ аутентификации в БД «Енисей». Основным недостатком является необходимость отправки учетных данных пользователя при каждом запросе, что может быть небезопасно и может снизить производительность работы (поскольку БД «Енисей» должен вычислять хэш пароля при каждом запросе).

Параметры HTTP заголовка запроса отображает Таблица 170.

**Таблица 170 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

#### 4.3.32.1.1. Формат запроса

Пример запроса отображает Рисунок 97

```
GET / HTTP/1.1
Accept: application/json
Authorization: Basic cm9vdDpyZWxheA==
Host: localhost:5984
```

**Запрос**
**Рисунок 97**

#### 4.3.32.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 171.

**Таблица 171 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 98. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 177
Content-Type: application/json
Date: Mon, 03 Dec 2012 00:44:47 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "couchdb":"Welcome",
    "uuid":"0a959b9b8227188afc2ac26ccdf345a6",
    "version":"1.3.0",
    "vendor": {
        "version":"1.3.0",
        "name":"The Apache Software Foundation"
    }
}
```

**Ответ**
**Рисунок 98**

#### 4.3.32.2. Аутентификация Cookie

Для аутентификации с помощью Cookie (RFC 2109) БД «Енисей» генерирует маркер, который клиент может использовать для следующих нескольких запросов к БД «Енисей». Токены действительны до истечения времени ожидания. Когда БД «Енисей» видит действительный маркер в последующем запросе, он аутентифицирует пользователя по этому маркеру, не запрашивая пароль повторно. По умолчанию Cookies действительны в течение 10 минут, но это можно перенастроить. Также можно сделать cookie постоянными.

Чтобы получить первый токен и таким образом аутентифицировать пользователя в первый раз, имя пользователя и пароль должны быть отправлены в _session ППИ.

##### 4.3.32.2.1. Инициировать новую сессию для указанных учетных данных пользователя

Инициирует новую сессию для указанных учетных данных пользователя, предоставляя значение Cookie.

###### 4.3.32.2.1.1. Формат запроса

Параметры запроса отображает Таблица 172.

**Таблица 172 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_session |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 173.

**Таблица 173 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/x-www-form-urlencoded, application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 174.

**Таблица 174 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| next | Обеспечивает перенаправление после успешного входа в систему на указанное место. Это расположение является относительным по отношению к корню сервера. | Строка (String) | Нет |

Параметры формы отображает Таблица 175.

**Таблица 175 – Параметры формы**

| Поле | Описание |
|---|---|
| name | Имя пользователя |
| password | Пароль |

Пример запроса отображает Рисунок 99.

```
POST /_session HTTP/1.1
Accept: application/json
Content-Length: 24
Content-Type: application/x-www-form-urlencoded
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

name=root&password=relax
```

**Запрос**
**Рисунок 99**

Запрос в формате JSON отображает Рисунок 100.

```
POST /_session HTTP/1.1
Accept: application/json
Content-Length: 37
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "name": "root",
    "password": "relax"
}
```

**Запрос в формате JSON**
**Рисунок 100**

Если был предоставлен следующий параметр запроса, то в случае успешной аутентификации ответ вызовет перенаправление на указанное место:

Пример запроса отображает Рисунок 101.

```
POST /_session?next=/blog/_design/sofa/_rewrite/recent-posts HTTP/1.1
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

name=root&password=relax
```

**Запрос**
**Рисунок 101**

#### 4.3.32.2.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 176.

**Таблица 176 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Set-Cookie | Маркер авторизации | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 177.

**Таблица 177 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |
| name | Имя пользователя | Строка (String) | Да |
| roles | Список ролей пользователя | Массив (Array) | Да |

Коды состояния отображает Таблица 178.

**Таблица 178 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 302 Found | Перенаправление после успешной аутентификации |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |

Ответ в случае успешного завершения запроса отображает Рисунок 102. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 43
Content-Type: application/json
Date: Mon, 03 Dec 2012 01:23:14 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Set-Cookie: AuthSession=cm9vdDo1MEJCRkYwMjq0LO0ylOIwShrgt8y-UkhI-c6BGw; Version=1; Path=/; HttpOnly

{"ok":true,"name":"root","roles":["_admin"]}
```

**Ответ**
**Рисунок 102**

Ответ на запрос с параметром отображает Рисунок 103.

```
HTTP/1.1 302 Moved Temporarily
Cache-Control: must-revalidate
Content-Length: 43
Content-Type: application/json
Date: Mon, 03 Dec 2012 01:32:46 GMT
Location: http://localhost:5984/blog/_design/sofa/_rewrite/recent-posts
Server: Yenisei/1.0.0 (Erlang OTP/24)
Set-Cookie: AuthSession=cm9vdDo1MEJDMDEzRTp7Vu5GKCkTxTVxwXbpXsBARQWnhQ; Version=1; Path=/; HttpOnly

{"ok":true,"name":null,"roles":["_admin"]}
```

**Ответ**
**Рисунок 103**

#### 4.3.32.2.2. Получить информацию об аутентифицированном пользователе

Возвращает информацию об аутентифицированном пользователе, включая объект контекста пользователя, метод аутентификации и базу данных, которые были использованы, а также список настроенных обработчиков аутентификации на сервере.

##### 4.3.32.2.2.1. Формат запроса

Параметры запроса отображает Таблица 179.

**Таблица 179 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_session |
| Метод | GET |

Параметры строки запроса отображает Таблица 180.

**Таблица 180 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| basic | Принять базовый аутентификатор при запросе этого ресурса. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 104.

```
GET /_session HTTP/1.1
Host: localhost:5984
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Cookie: AuthSession=cm9vdDo1MEJDMDQxRDpqb-Ta9QfP9hpdPjHLxNTKg_Hf9w
```

**Запрос**
**Рисунок 104**

#### 4.3.32.2.2.2. Формат ответа на запрос

Объекты JSON ответа отображает Таблица 181.

**Таблица 181 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |
| userCtx | Пользовательский контекст для текущего пользователя | Объект (Json Object) | Да |
| info | Конфигурация аутентификации сервера | Объект (Json Object) | Да |

Коды состояния отображает Таблица 182.

**Таблица 182 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |

Ответ в случае успешного завершения запроса отображает Рисунок 105. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 175
Content-Type: application/json
Date: Fri, 09 Aug 2013 20:27:45 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Set-Cookie: AuthSession=cm9vdDo1MjA1NTBDMTqmX2qKt1KDR--GUC80DQ6-Ew_XIw; Version=1; Path=/; HttpOnly

{
    "info": {
        "authenticated": "cookie",
        "authentication_db": "_users",
        "authentication_handlers": [
            "cookie",
            "default"
        ]
    },
    "ok": true,
    "userCtx": {
        "name": "root",
        "roles": [
            "_admin"
        ]
    }
}
```

**Ответ**
**Рисунок 105**

#### 4.3.32.2.3. Закрыть сессию пользователя

Закрывает сессию пользователя, давая браузеру команду очистить Cookie. Это не аннулирует сессию с точки зрения сервера, так как нет способа сделать это, поскольку Cookie БД «Енисей» не имеют статических данных. Это означает, что вызов этой конечной точки является необязательным с точки зрения клиента и не защищает от кражи Cookie сессии.

##### 4.3.32.2.3.1. Формат запроса

Параметры запроса отображает Таблица 183.

**Таблица 183 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_session |
| Метод | DELETE |

Пример запроса отображает Рисунок 106.

```
DELETE /_session HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Cookie: AuthSession=cm9vdDo1MjA1NEVGMDo1QXNQkqC_0Qmgrk8Fw61_AzDeXw
Host: localhost:5984
```

**Запрос**
**Рисунок 106**

#### 4.3.32.2.3.2. Формат ответа на запрос

Коды состояния отображает Таблица 184.

**Таблица 184 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешно закрыта сессия |

Ответ в случае успешного завершения запроса отображает Рисунок 107. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Fri, 09 Aug 2013 20:30:12 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Set-Cookie: AuthSession=; Version=1; Path=/; HttpOnly

{
    "ok": true
}
```

**Ответ**
**Рисунок 107**

### 4.3.32.3. Прокси-аутентификация

Для использования этого метода аутентификации убедитесь, что значение {chttpd_auth, proxy_authentication_handler} добавлено в список активных chttpd/authentication_handlers, что отображает Рисунок 108.

```
[chttpd]
authentication_handlers = {chttpd_auth, cookie_authentication_handler}, {chttpd_auth, proxy_authentica
tion_handler}, {chttpd_auth, default_authentication_handler}
```

**Настройка метода аутентификации**
**Рисунок 108**

Прокси-аутентификация очень полезна в том случае, если приложение уже использует какой-либо внешний сервис аутентификации, и нет необходимости дублировать пользователей и их роли в БД «Енисей».

Этот метод аутентификации позволяет создать объект User Context Object для удаленного аутентифицированного пользователя. По умолчанию клиенту просто нужно передать БД «Енисей» определенные заголовки с соответствующими запросами:

X-Auth-CouchDB-UserName: имя пользователя;

X-Auth-CouchDB-Roles: разделенный запятыми (,) список ролей пользователя;

X-Auth-CouchDB-Token: токен аутентификации. Когда proxy_use_secret установлен (что настоятельно рекомендуется!), этот заголовок предоставляет HMAC имени

пользователя для аутентификации и секретный токен для предотвращения запросов из ненадежных источников. (Используйте SHA1 имени пользователя и подписывайте секретом).

Создание токена (пример с openssl) отображает Рисунок 109.

```
echo -n "foo" | openssl dgst -sha1 -hmac "the_secret"
# (stdin)= 22047ebd7c4ec67dfbcbad7213a693249dbfbf86
```

**Создание токена**
**Рисунок 109**

#### 4.3.32.3.1. Формат запроса

Пример запроса отображает Рисунок 110.

```
GET /_session HTTP/1.1
Host: localhost:5984
Accept: application/json
Content-Type: application/json; charset=utf-8
X-Auth-CouchDB-Roles: users,blogger
X-Auth-CouchDB-UserName: foo
X-Auth-CouchDB-Token: 22047ebd7c4ec67dfbcbad7213a693249dbfbf86
```

**Запрос**
**Рисунок 110**

#### 4.3.32.3.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 111. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 190
Content-Type: application/json
Date: Fri, 14 Jun 2013 10:16:03 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "info": {
        "authenticated": "proxy",
        "authentication_db": "_users",
        "authentication_handlers": [
            "cookie",
            "proxy",
            "default"
        ]
    },
    "ok": true,
    "userCtx": {
        "name": "foo",
        "roles": [
            "users",
            "blogger"
        ]
    }
}
```

**Ответ**
**Рисунок 111**

Не нужно запрашивать сессию для аутентификации этим методом, если все необходимые HTTP-заголовки предоставлены.

#### 4.3.32.4. Аутентификация JWT

Чтобы использовать этот метод аутентификации, убедитесь, что значение {chttpd_auth, jwt_authentication_handler} добавлено в список активных chttpd/authentication_handlers, что отображает Рисунок 112.

```
[chttpd]
authentication_handlers = {chttpd_auth, cookie_authentication_handler}, {chttpd_auth, proxy_authentica
tion_handler}, {chttpd_auth, default_authentication_handler}
```

**Настройка метода аутентификации**
**Рисунок 112**

Аутентификация JWT позволяет БД «Енисей» использовать генерируемые извне токены JWT вместо определения пользователей или ролей в базе данных _users.

Обработчик аутентификации JWT требует, чтобы все токены JWT были подписаны ключом, которому БД «Енисей» настроен доверять (нет поддержки алгоритма «NONE» JWT).

Кроме того, БД «Енисей» может быть настроен на отклонение JWT-токенов, в которых отсутствует настраиваемый набор утверждений (например, администратор БД «Енисей» может настаивать на утверждении exp).

Проверяются только утверждения, перечисленные в обязательных проверках. Дополнительные утверждения будут проигнорированы.

Для настройки аутентификации JWT существуют два раздела конфигурации;

Конфигурация required_claims представляет собой список дополнительных обязательных утверждений JWT, разделенных запятыми, которые должны присутствовать в любом представленном токене JWT. Если какое-либо из них отсутствует, отправляется 400 Bad Request.

Утверждение alg является обязательным, поскольку оно используется для поиска правильного ключа для проверки подписи.

Утверждение sub является обязательным и используется в качестве имени пользователя БД «Енисей», если JWT-токен действителен.

Частное утверждение _couchdb.roles является необязательным. Если оно представлено в виде массива строк JSON, то используется в качестве списка ролей пользователя БД «Енисей», пока действителен токен JWT. Настройку аутентификации отображает Рисунок 113.

```
; [jwt_keys]
; Configure at least one key here if using the JWT auth handler.
; If your JWT tokens do not include a "kid" attribute, use "_default"
; as the config key, otherwise use the kid as the config key.
; Examples
; hmac:_default = aGVsbG8=
; hmac:foo = aGVsbG8=
; The config values can represent symmetric and asymmetric keys.
; For symmetric keys, the value is base64 encoded;
; hmac:_default = aGVsbG8= # base64-encoded form of "hello"
; For asymmetric keys, the value is the PEM encoding of the public
; key with newlines replaced with the escape sequence \n.
; rsa:foo = -----BEGIN PUBLIC KEY-----\nMIIBIjAN...IDAQAB\n-----END PUBLIC KEY-----\n
; ec:bar = -----BEGIN PUBLIC KEY-----\nMHYwEAYHK...AzztRs\n-----END PUBLIC KEY-----\n
```

**Настройка аутентификации**
**Рисунок 113**

В разделе jwt_key перечислены все ключи, которым доверяет данный сервер БД «Енисей». Необходимо убедиться, что все узлы кластера имеют одинаковый список.

JWT-токены, не содержащие утверждения kid, будут проверяться по ключу $alg:_default.

В целях безопасности обязательно указывать алгоритм, связанный с каждым ключом (например, представление токена с подписью HMAC, использующего открытый ключ RSA или EC, которому доверяет сервер: https://auth0.com/blog/ critical-vulnerabilities-in-json-web-token-libraries/).

#### 4.3.32.4.1. Формат запроса

Пример запроса отображает Рисунок 114.

```
GET /_session HTTP/1.1
Host: localhost:5984
Accept: application/json
Content-Type: application/json; charset=utf-8
Authorization: Bearer <JWT token>
```

**Запрос**
**Рисунок 114**

#### 4.3.32.4.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 115. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 188
Content-Type: application/json
Date: Sun, 19 Apr 2020 08:29:15 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "info": {
        "authenticated": "jwt",
        "authentication_db": "_users",
        "authentication_handlers": [
            "cookie",
            "proxy",
            "default"
        ]
    },
    "ok": true,
    "userCtx": {
        "name": "foo",
        "roles": [
            "users",
            "blogger"
        ]
    }
}
```

**Ответ**
**Рисунок 115**

Не нужно запрашивать сеанс аутентификации этим методом, если предоставлен необходимый HTTP-заголовок.

### 4.3.33 Конфигурация

ППИ конфигурации сервера БД «Енисей» предоставляет интерфейс для запроса и обновления различных значений конфигурации в работающем экземпляре БД «Енисей».

#### 4.3.33.1. Доступ к конфигурации локального узла

Литеральная строка _local служит псевдонимом для имени локального узла, поэтому во всех URL-адресах конфигурации {node-name} можно заменить на _local, чтобы взаимодействовать с конфигурацией локального узла.

#### 4.3.33.2. Получить всю конфигурацию сервера

Возвращает всю конфигурацию сервера БД «Енисей» в виде структуры JSON. Структура организована по различным разделам конфигурации с индивидуальными значениями.

##### 4.3.33.2.1. Формат запроса

Параметры запроса отображает Таблица 185.

**Таблица 185 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 186.

**Таблица 186 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 116.

```
GET /_node/nonode@nohost/_config HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 116**

#### 4.3.33.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 187.

**Таблица 187 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 188.

**Таблица 188 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 117. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 4148
Content-Type: application/json
Date: Sat, 10 Aug 2013 12:01:42 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "attachments": {
        "compressible_types": "text/*, application/javascript, application/json,  application/xml",
        "compression_level": "8"
    },
    "couchdb": {
        "users_db_suffix": "_users",
        "database_dir": "/var/lib/couchdb",
        "max_attachment_chunk_size": "4294967296",
        "max_dbs_open": "100",
        "os_process_timeout": "5000",
        "uri_file": "/var/lib/couchdb/couch.uri",
        "util_driver_dir": "/usr/lib64/couchdb/erlang/lib/couch-1.5.0/priv/lib",
        "view_index_dir": "/var/lib/couchdb"
    },
    "chttpd": {
        "allow_jsonp": "false",
        "backlog": "512",
        "bind_address": "0.0.0.0",
        "port": "5984",
        "require_valid_user": "false",
        "socket_options": "[{sndbuf, 262144}, {nodelay, true}]",
        "server_options": "[{recbuf, undefined}]",
        "secure_rewrites": "true"
    },
    "httpd": {
        "authentication_handlers": "{couch_httpd_auth, cookie_authentication_handler}, {couch_httpd_au
th, default_authentication_handler}",
        "bind_address": "192.168.0.2",
        "max_connections": "2048",
        "port": "5984",
    },
    "log": {
        "writer": "file",
        "file": "/var/log/couchdb/couch.log",
        "include_sasl": "true",
        "level": "info"
    },
    "query_server_config": {
        "reduce_limit": "true"
    },
    "replicator": {
        "max_http_pipeline_size": "10",
        "max_http_sessions": "10"
    },
    "stats": {
        "interval": "10"
    },
    "uuids": {
        "algorithm": "utc_random"
    }
}
```

**Ответ**

**Рисунок 117**

#### 4.3.33.3. Получить структуру конфигурации для одной секции

##### 4.3.33.3.1. Формат запроса

Параметры запроса отображает Таблица 189.

**Таблица 189 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config/{section} |
| Метод | GET |

Параметры строки запроса отображает Таблица 190.

**Таблица 190 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| section | Имя раздела конфигурации |

Параметры HTTP заголовка запроса отображает Таблица 191.

**Таблица 191 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 118.

```
GET /_node/nonode@nohost/_config/httpd HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 118**

##### 4.3.33.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 192.

**Таблица 192 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 193.

**Таблица 193 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |

Ответ в случае успешного завершения запроса отображает Рисунок 119. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 444
Content-Type: application/json
Date: Sat, 10 Aug 2013 12:10:40 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "authentication_handlers": "{couch_httpd_auth, cookie_authentication_handler}, {couch_httpd_auth,
default_authentication_handler}",
    "bind_address": "127.0.0.1",
    "default_handler": "{couch_httpd_db, handle_request}",
    "port": "5984"
}
```

**Ответ**
**Рисунок 119**

### 4.3.33.4. Получить значение конфигурации из определенного раздела конфигурации

#### 4.3.33.4.1. Формат запроса

Параметры запроса отображает Таблица 194.

**Таблица 194 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config/{section}/{key} |
| Метод | GET |

Параметры строки запроса отображает Таблица 195.

**Таблица 195 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| section | Имя раздела конфигурации |
| key | Имя параметра конфигурации |

Параметры HTTP заголовка запроса отображает Таблица 196.

**Таблица 196 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 120.

```
GET /_node/nonode@nohost/_config/log/level HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 120**

#### 4.3.33.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 197.

**Таблица 197 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 198.

**Таблица 198 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |

Ответ в случае успешного завершения запроса отображает Рисунок 121. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 8
Content-Type: application/json
Date: Sat, 10 Aug 2013 12:12:59 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

"debug"
```

**Ответ**
**Рисунок 121**

Возвращаемым значением будет JSON значения, которое может быть строкой или числовым значением, массивом или объектом. Некоторые клиентские среды могут не воспринимать простые строки или числовые значения как правильный JSON.

#### 4.3.33.5. Обновить значение конфигурации

Обновляет значение конфигурации. Новое значение должно быть предоставлено в теле запроса в соответствующем формате JSON. Если задается строковое значение, необходимо предоставить корректную строку JSON. В ответ БД «Енисей» отправляет старое значение для ключа целевой секции.

##### 4.3.33.5.1. Формат запроса

Параметры запроса отображает Таблица 199.

**Таблица 199 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config/{section}/{key}¶ |
| Метод | PUT |

Параметры строки запроса отображает Таблица 200.

**Таблица 200 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| section | Имя раздела конфигурации |
| key | Имя параметра конфигурации |

Параметры HTTP заголовка запроса отображает Таблица 201.

**Таблица 201 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 122.

```
PUT /_node/nonode@nohost/_config/log/level HTTP/1.1
Accept: application/json
Content-Length: 7
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

"info"
```

**Запрос**
**Рисунок 122**

#### 4.3.33.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 202.

**Таблица 202 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 203.

**Таблица 203 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 400 Bad Request | Неверное тело запроса JSON |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |
| 500 Internal Server Error | Ошибка при настройке конфигурации |

Ответ в случае успешного завершения запроса отображает Рисунок 123. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 8
Content-Type: application/json
Date: Sat, 10 Aug 2013 12:12:59 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

"debug"
```

**Ответ**
**Рисунок 123**

#### 4.3.33.6. Удалить значение конфигурации

Удаляет значение конфигурации. Возвращаемый JSON будет значением параметра конфигурации до его удаления.

##### 4.3.33.6.1. Формат запроса

Параметры запроса отображает Таблица 204.

**Таблица 204 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config/{section}/{key} |
| Метод | DELETE |

Параметры строки запроса отображает Таблица 205.

**Таблица 205 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| section | Имя раздела конфигурации |
| key | Имя параметра конфигурации |

Параметры HTTP заголовка запроса отображает Таблица 206.

**Таблица 206 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 124.

```
DELETE /_node/nonode@nohost/_config/log/level HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 124**

#### 4.3.33.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 207.

**Таблица 207 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 208.

**Таблица 208 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успешная аутентификация |
| 401 Unauthorized | Имя пользователя или пароль не были распознаны |
| 404 Not Found | Указанный параметр конфигурации не найден |

Ответ в случае успешного завершения запроса отображает Рисунок 125. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 7
Content-Type: application/json
Date: Sat, 10 Aug 2013 12:29:03 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

"info"
```

**Ответ**
**Рисунок 125**

### 4.3.33.7. Перезагрузить конфигурацию с диска

Перезагружает конфигурацию с диска. Побочным эффектом является очистка всех изменений конфигурации в памяти, которые не были зафиксированы на диске.

#### 4.3.33.7.1. Формат запроса

Параметры запроса отображает Таблица 209.

**Таблица 209 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_node/{node-name}/_config/_reload |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 210.

**Таблица 210 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 126.

```
POST /_node/nonode@nohost/_config/_reload HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 126**

#### 4.3.33.7.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 211.

**Таблица 211 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 127. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Tues, 21 Jan 2020 11:09:35
Server: CouchDB/3.0.0 (Erlang OTP)

{"ok":true}
```

**Ответ**

**Рисунок 127**

## 4.4. Базы данных

Конечная точка базы данных предоставляет интерфейс к целой базе данных в БД «Енисей». Это запросы на уровне базы данных, а не на уровне документов.

Для всех этих запросов имя базы данных в пути URL должно быть именем базы данных, с которой необходимо выполнить операцию. Например, чтобы получить метаинформацию для базы данных (recipes), можно использовать HTTP-запрос:

```
GET /recipes
```

Для наглядности в путях URL используется форма, приведенная ниже:

```
GET /db
```

где `db` — имя любой базы данных.

### 4.4.1 Получить HTTP-заголовки с информацией о базе данных

Возвращает HTTP-заголовки, содержащие минимальное количество информации об указанной базе данных. Поскольку тело ответа пустое, использование метода HEAD это легкий способ проверить, существует ли уже база данных или нет.

#### 4.4.1.1. Формат запроса

Параметры запроса отображает Таблица 212.

**Таблица 212 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db} |
| Метод | HEAD |

Параметры строки запроса отображает Таблица 213.

**Таблица 213 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 214.

**Таблица 214 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Коды состояния отображает Таблица 215.

**Таблица 215 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | База данных существует |
| 404 Not Found | Запрашиваемая база данных не найдена |

Пример запроса отображает Рисунок 128.

```
HEAD /test HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 128**

#### 4.4.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 216.

**Таблица 216 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Нет |

Ответ в случае успешного завершения запроса отображает Рисунок 129. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Mon, 12 Aug 2013 01:27:41 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
```

**Ответ**
**Рисунок 129**

### 4.4.2 Получить информацию о базе данных

#### 4.4.2.1. Формат запроса

Параметры запроса отображает Таблица 217.

**Таблица 217 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db} |
| Метод | GET |

Параметры строки запроса отображает Таблица 218.

**Таблица 218 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 219.

**Таблица 219 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 130.

```
GET /receipts HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 130**

#### 4.4.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 220.

**Таблица 220 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 221.

**Таблица 221 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| cluster.n | Реплики. Количество копий каждого документа | Число (int64) | Да |
| cluster.q | Шарды. Количество разделов диапазона | Число (int64) | Да |
| cluster.r | Считывает кворум. Количество согласованных копий документа, которые должны быть прочитаны до успешного ответа | Число (int64) | Да |
| cluster.w | Записывает кворум. Количество копий документа, которые должны быть записаны для успешного ответа | Число (int64) | Да |
| compact_running | Устанавливается в true, если процедура уплотнения базы данных работает с этой базой данных | Логический тип (Boolean) | Да |
| db_name | Имя базы данных | Строка (String) | Да |
| disk_format_version | Версия физического формата, используемого для данных при хранении на диске | Число (int64) | Да |
| doc_count | Количество документов в указанной базе данных | Строка (String) | Да |
| doc_del_count | Количество удаленных документов | Число (int64) | Да |
| instance_start_time | Всегда "0". (Возвращается по унаследованным причинам). | Число (int64) | Да |
| purge_seq | Непрозрачная строка, описывающая состояние очистки базы данных. Не полагайтесь на эту строку для подсчета количества операций очистки | Строка (String) | Да |
| sizes.active | Размер реальных данных внутри базы данных, в байтах. | Число (int64) | Да |
| sizes.external | Размер содержимого базы данных без сжатия в байтах. | Число (int64) | Да |
| sizes.file | Размер файла базы данных на диске в байтах. Индексы представлений не включаются в расчет | Число (int64) | Да |
| update_seq | Строка, описывающая состояние базы данных. Не полагайтесь на эту строку для подсчета количества обновлений. | Строка (String) | Да |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| props.partitioned | Если присутствует и имеет значение true, это указывает на то, что база данных разбита на разделы. | Логический тип (Boolean) | Нет |

Коды состояния отображает Таблица 222.

**Таблица 222 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | База данных создана успешно (кворум соблюден). |
| 202 Accepted | Принята (по крайней мере, одним узлом) |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 412 Precondition Failed | База данных уже существует |

Ответ в случае успешного завершения запроса отображает Рисунок 131. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 258
Content-Type: application/json
Date: Mon, 12 Aug 2013 01:38:57 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "cluster": {
        "n": 3,
        "q": 8,
        "r": 2,
        "w": 2
    },
    "compact_running": false,
    "db_name": "receipts",
    "disk_format_version": 6,
    "doc_count": 6146,
    "doc_del_count": 64637,
    "instance_start_time": "0",
    "props": {},
    "purge_seq": 0,
    "sizes": {
        "active": 65031503,
        "external": 66982448,
        "file": 137433211
    },
    "update_seq": "292786-g1AAAAF..."
}
```

**Ответ**
**Рисунок 131**

### 4.4.3 Создать новую базу данных

Создает новую базу данных. Имя базы данных {db} должно быть составлено по следующим правилам:

1) Имя должно начинаться со строчной буквы (a-z)

2) Строчные символы (a-z)

3) Цифры (0-9)

4) Любой из символов _, $, (, ), +, -, и /.

Вышеприведенные правила можно записать как ^[a-z][a-z0-9_$()+/ -]*$.

#### 4.4.3.1. Формат запроса

Параметры запроса отображает Таблица 223.

**Таблица 223 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db} |
| Метод | PUT |

Параметры строки запроса отображает Таблица 224.

**Таблица 224 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры тела запроса отображает Таблица 225.

**Таблица 225 – Параметры тела запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| q | Сегменты, количество разделов диапазона. По умолчанию 8, если это не переопределено в конфигурации кластера | Число (int64) | Нет |
| n | Реплики. Количество копий базы данных в кластере. По умолчанию 3, если это не переопределено в конфигурации кластера | Число (int64) | Нет |
| partitioned | Создавать ли базу данных с разделами. По умолчанию false | Логический тип (Boolean) | Нет |

Параметры HTTP заголовка запроса отображает Таблица 226.

**Таблица 226 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 132.

```
PUT /db HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 132**

Если повторить тот же запрос к БД «Енисей», он ответит 412, поскольку база данных уже существует:

Пример запроса отображает Рисунок 133.

```
PUT /db HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 133**

Если указано недопустимое имя базы данных, БД «Енисей» возвращает ответ 400:

Пример запроса отображает Рисунок 134.

```
PUT /_db HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 134**

#### 4.4.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 227.

**Таблица 227 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | Расположение URI базы данных | Строка (String) | Нет |

Объекты JSON ответа отображает Таблица 228.

**Таблица 228 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции. Доступно в случае успеха | Логический тип (Boolean) | Нет |
| error | Тип ошибки. Доступно, если код ответа 4xx | Строка (String) | Нет |
| reason | Описание ошибки. Доступно, если код ответа равен 4xx | Строка (String) | Нет |

Коды состояния отображает Таблица 229.

**Таблица 229 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | База данных создана успешно (кворум соблюден). |
| 202 Accepted | Принята (по крайней мере, одним узлом) |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 412 Precondition Failed | База данных уже существует |

Ответ в случае успешного завершения запроса отображает Рисунок 135. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 08:01:45 GMT
Location: http://localhost:5984/db
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 135**

Ответ на запрос создания уже существующей базы данных отображает Рисунок 136.

```
HTTP/1.1 412 Precondition Failed
Cache-Control: must-revalidate
Content-Length: 95
Content-Type: application/json
Date: Mon, 12 Aug 2013 08:01:16 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "error": "file_exists",
    "reason": "The database could not be created, the file already exists."
}
```

**Ответ**
**Рисунок 136**

Ответ на запрос создания базы данных с недопустимым именем отображает Рисунок 137.

```
HTTP/1.1 400 Bad Request
Cache-Control: must-revalidate
Content-Length: 194
Content-Type: application/json
Date: Mon, 12 Aug 2013 08:02:10 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "error": "illegal_database_name",
    "reason": "Name: '_db'. Only lowercase characters (a-z), digits (0-
9), and any of the characters _, $, (, ), +, -, and / are allowed. Must begin with a letter."
}
```

**Ответ**
**Рисунок 137**

### 4.4.4 Удалить базу данных

Удаляет указанную базу данных, а также все содержащиеся в ней документы и вложения.

Чтобы избежать удаления базы данных, БД «Енисей» будет отвечать HTTP кодом состояния 400, если URL запроса включает параметр ?rev=. Это говорит о том, что человек хочет удалить документ, но забыл добавить идентификатор документа в URL.

#### 4.4.4.1. Формат запроса

Параметры запроса отображает Таблица 230.

**Таблица 230 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db} |
| Метод | DELETE |

Параметры строки запроса отображает Таблица 231.

**Таблица 231 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 232.

**Таблица 232 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 138.

```
DELETE /db HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**

**Рисунок 138**

#### 4.4.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 233.

**Таблица 233 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 234.

**Таблица 234 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Нет |

Коды состояния отображает Таблица 235.

**Таблица 235 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | База данных удалена успешно (кворум соблюден и база данных удалена хотя бы одним узлом) |
| 202 Accepted | Принято (удалена хотя бы одним из узлов, кворум еще не достигнут) |
| 400 Bad Request | Неверное имя базы данных или случайно забыт идентификатор документа |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | База данных не существует или неверное имя базы данных |

Ответ в случае успешного завершения запроса отображает Рисунок 139. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 08:54:00 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**

**Рисунок 139**

### 4.4.5 Создать новый документ в базе данных

Создает новый документ в указанной базе данных, используя предоставленную структуру документа JSON.

Если структура JSON включает поле _id, то документ будет создан с указанным ID документа.

Если поле _id не указано, то будет сгенерирован новый уникальный ID, в соответствии с алгоритмом UUID, настроенным для данного сервера.

#### 4.4.5.1. Формат запроса

Параметры запроса отображает Таблица 236.

**Таблица 236 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db} |
| Метод | POST |

Параметры строки запроса отображает Таблица 237.

**Таблица 237 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 238.

**Таблица 238 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 239.

**Таблица 239 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| batch | Хранить документ в пакетном режиме Возможные значения: ok. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 140.

```
POST /db HTTP/1.1
Accept: application/json
Content-Length: 81
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
{
    "servings": 4,
    "subtitle": "Delicious with fresh bread",
    "title": "Fish Stew"
}
```

**Запрос**
**Рисунок 140**

#### 4.4.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 240.

**Таблица 240 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, text/plain; charset=utf-8 | Строка (String) | Да |
| Location | Расположение URI документа | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 241.

**Таблица 241 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Информация о пересмотре | Строка (String) | Да |

Коды состояния отображает Таблица 242.

**Таблица 242 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Документ создан и сохранен на диске |
| 202 Accepted | Данные документа приняты, но еще не сохранены на диске |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | База данных не существует |
| 409 Conflict | Конфликтующий документ с таким же ID уже существует |

Ответ в случае успешного завершения запроса отображает Рисунок 141. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 95
Content-Type: application/json
Date: Tue, 13 Aug 2013 15:19:25 GMT
Location: http://localhost:5984/db/ab39fe0993049b84cfa81acd6ebad09d
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "id": "ab39fe0993049b84cfa81acd6ebad09d",
    "ok": true,
    "rev": "1-9c65296036141e575d32ba9c034dd3ee"
}
```

**Ответ**
**Рисунок 141**

#### 4.4.5.3. Указание идентификатора документа

Идентификатор документа можно указать, включив поле _id в JSON отправляемой записи. Следующий запрос создаст тот же документ с идентификатором FishStew.

##### 4.4.5.3.1. Формат запроса

Пример запроса отображает Рисунок 142

```
POST /db HTTP/1.1
Accept: application/json
Content-Length: 98
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
{
    "_id": "FishStew",
    "servings": 4,
    "subtitle": "Delicious with fresh bread",
    "title": "Fish Stew"
}
```

**Запрос**
**Рисунок 142**

#### 4.4.5.3.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 143. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 71
Content-Type: application/json
Date: Tue, 13 Aug 2013 15:19:25 GMT
ETag: "1-9c65296036141e575d32ba9c034dd3ee"
Location: http://localhost:5984/db/FishStew
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "id": "FishStew",
    "ok": true,
    "rev": "1-9c65296036141e575d32ba9c034dd3ee"
}
```

**Ответ**
**Рисунок 143**

#### 4.4.5.4. Пакетный режим записи

Можно записывать документы в базу данных с более высокой скоростью, используя опцию пакетного режима. При этом записи документов собираются вместе в памяти (на основе каждого пользователя) до их фиксации на диске. Это увеличивает риск того, что документы не будут сохранены в случае сбоя, поскольку документы не записываются на диск немедленно.

Пакетный режим не подходит для критически важных данных, но может быть идеальным для таких приложений, как журнальные данные, когда риск некоторой потери данных в результате сбоя является приемлемым.

Чтобы использовать пакетный режим, добавьте аргумент запроса batch=ok к URL запроса POST /{db}, PUT /{db}/{docid}, или DELETE /{db}/{docid}. Сервер БД «Енисей» немедленно ответит кодом ответа HTTP 202 Accepted.

Создание или обновление документов в пакетном режиме не гарантирует, что все документы будут успешно сохранены на диске. Например, отдельные документы могут быть не сохранены из-за конфликтов, отклонения функцией валидации или по другим причинам, даже если в целом пакет был успешно отправлен.

#### 4.4.5.4.1. Формат запроса

Пример запроса отображает Рисунок 144.

```
POST /db?batch=ok HTTP/1.1
Accept: application/json
Content-Length: 98
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
{
    "_id": "FishStew",
    "servings": 4,
    "subtitle": "Delicious with fresh bread",
    "title": "Fish Stew"
}
```

**Запрос**
**Рисунок 144**

#### 4.4.5.4.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 145. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 202 Accepted
Cache-Control: must-revalidate
Content-Length: 28
Content-Type: application/json
Date: Tue, 13 Aug 2013 15:19:25 GMT
Location: http://localhost:5984/db/FishStew
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "id": "FishStew",
    "ok": true
}
```

**Ответ**
**Рисунок 145**

### 4.4.6 Получить все документы из базы данных

При запуске встроенного представления _all_docs view, будут возвращены все документы из базы данных. За исключением параметров URL (описанных ниже), эта конечная точка работает так же, как и любое другое представление.

#### 4.4.6.1. Формат запроса

Параметры запроса отображает Таблица 243.

**Таблица 243 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_all_docs |
| Метод | GET |

Параметры строки запроса отображает Таблица 244.

**Таблица 244 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 245.

**Таблица 245 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 146.

```
GET /db/_all_docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 146**

#### 4.4.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 246.

**Таблица 246 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Коды состояния отображает Таблица 247.

**Таблица 247 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 404 Not Found | База данных не существует |

Ответ в случае успешного завершения запроса отображает Рисунок 147. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 10 Aug 2013 16:22:56 GMT
ETag: "1W2DJUZFZSZD9K78UFA3GZWB4"
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "16e458537602f5ef2a710089dffd9453",
            "key": "16e458537602f5ef2a710089dffd9453",
            "value": {
                "rev": "1-967a00dff5e02add41819138abb3284d"
            }
        },
        {
            "id": "a4c51cdfa2069f3e905c431114001aff",
            "key": "a4c51cdfa2069f3e905c431114001aff",
            "value": {
                "rev": "1-967a00dff5e02add41819138abb3284d"
            }
        },
        {
            "id": "a4c51cdfa2069f3e905c4311140034aa",
            "key": "a4c51cdfa2069f3e905c4311140034aa",
            "value": {
                "rev": "5-6182c9c954200ab5e3c6bd5e76a1549f"
            }
        },
        {
            "id": "a4c51cdfa2069f3e905c431114003597",
            "key": "a4c51cdfa2069f3e905c431114003597",
            "value": {
                "rev": "2-7051cbe5c8faecd085a3fa619e6e6337"
            }
        },
        {
            "id": "f4ca7773ddea715afebc4b4b15d4f0b3",
            "key": "f4ca7773ddea715afebc4b4b15d4f0b3",
            "value": {
                "rev": "2-7051cbe5c8faecd085a3fa619e6e6337"
            }
        }
    ],
    "total_rows": 5
}
```

**Ответ**
**Рисунок 147**

### 4.4.7 Получить все документы из базы данных, указав параметры строки запроса в теле запроса

Функциональность представления POST _all_docs поддерживает идентичные параметры и поведение, указанные в ППИ GET /{db}/_all_docs, но позволяет указывать параметры строки запроса в качестве ключей в объекте JSON в теле запроса POST.

#### 4.4.7.1. Формат запроса

Параметры запроса отображает Таблица 248.

**Таблица 248 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_all_docs |
| Метод | POST |

Пример запроса отображает Рисунок 148.

```
POST /db/_all_docs HTTP/1.1
Accept: application/json
Content-Length: 70
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "keys" : [
        "Zingylemontart",
        "Yogurtraita"
    ]
}
```

**Запрос**
**Рисунок 148**

#### 4.4.7.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 149. Коды ошибок приведены в пункте 6.1.2.

```
{
    «total_rows" : 2666,
    «rows" : [
        {
            «value" : {
                «rev" : «1-a3544d296de19e6f5b932ea77d886942"
            },
            «id" : «Zingylemontart",
            «key" : «Zingylemontart"
        },
        {
            «value" : {
                «rev" : «1-91635098bfe7d40197a1b98d7ee085fc"
            },
            «id" : «Yogurtraita",
            «key" : «Yogurtraita"
        }
    ],
    «offset" : 0
}
```

**Ответ**
**Рисунок 149**

### 4.4.8 Получить JSON-структуру всех проектных документов в данной базе данных

GET /{db}/_design_docs возвращает JSON-структуру всех проектных документов в данной базе данных. Информация возвращается в виде JSON-структуры, содержащей метаинформацию о возвращаемой структуре, включая список всех проектных документов и основное содержание, состоящее из ID, ревизии и ключа. Ключом является _id проектного документа.

#### 4.4.8.1. Формат запроса

Параметры запроса отображает Таблица 249.

**Таблица 249 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design_docs |
| Метод | GET |

Параметры строки запроса отображает Таблица 250.

**Таблица 250 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 251.

**Таблица 251 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры строки запроса отображает Таблица 252.

**Таблица 252 – Параметры строки запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| conflicts | Включает информацию о конфликтах в ответе. Игнорируется, если include_docs не равен true. По умолчанию установлено значение false. | Логический тип (Boolean) | Нет |
| descending | Возвращает по ключу проектные документы в порядке убывания. По умолчанию установлено значение false. | Логический тип (Boolean) | Нет |
| endkey | Останавливает возврат записей при достижении указанного ключа. Необязательный параметр. | Строка (String) | Нет |
| end_key | Псевдоним для параметра endkey. | Строка (String) | Нет |
| endkey_docid | Останавливает возврат записей при достижении указанного идентификатора проектного документа. Необязательный параметр. | Строка (String) | Нет |
| end_key_doc_id | Псевдоним для параметра endkey_docid. | Строка (String) | Нет |
| include_docs | Возвращает полное содержание проектной документации. По умолчанию установлено значение false. | Логический тип (Boolean) | Нет |
| inclusive_end | Указывает, должен ли указанный конечный ключ быть включен в результат. По умолчанию установлено значение true. | Логический тип (Boolean) | Нет |
| key | Возвращает только те проектные документы, которые соответствуют указанному ключу. Необязательный параметр. | Строка (String) | Нет |
| keys | Возвращает только те проектные документы, которые соответствуют указанным ключам. Необязательный параметр. | Строка (String) | Нет |
| limit | Ограничивает количество возвращаемых проектных документов указанным числом. Необязательный параметр. | Число (int64) | Нет |
| skip | Пропускает указанное количество записей перед началом возврата результатов. По умолчанию установлено значение 0. | Число (int64) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| startkey | Возвращает записи, начинающиеся с указанного ключа. Необязательный параметр. | Строка (String) | Нет |
| start_key | Псевдоним для параметра startkey. | Строка (String) | Нет |
| startkey_docid | Возвращает записи, начинающиеся с указанного идентификатора проектного документа. Необязательный параметр. | Строка (String) | Нет |
| start_key_doc_id | Псевдоним для параметра startkey_docid. | Строка (String) | Нет |
| update_seq | Ответ включает значение параметра update_seq, указывающее, какой идентификатор последовательности основной базы данных отражает представление. По умолчанию установлено значение false. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 150.

```
GET /db/_design_docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 150**

#### 4.4.8.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 253.

**Таблица 253 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 254.

**Таблица 254 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Смещение начала списка проектных документов | Число (int64) | Да |
| rows | Массив объектов строк представления. По умолчанию возвращаемая информация содержит только идентификатор и ревизию проектного документа. | Массив (Array) | Да |
| total_rows | Количество проектных документов в базе данных. Следует обратить внимание, что это не количество строк, возвращаемых в фактическом запросе. | Число (int64) | Да |
| update_seq | Текущая последовательность обновлений для базы данных. | Число (int64) | Нет |

Коды состояния отображает Таблица 255.

**Таблица 255 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 404 Not Found | Запрашиваемая база данных не найдена |

Ответ в случае успешного завершения запроса отображает Рисунок 151. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
ETag: "1W2DJUZFZSZD9K78UFA3GZWB4"
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "_design/ddoc01",
            "key": "_design/ddoc01",
            "value": {
                "rev": "1-7407569d54af5bc94c266e70cbf8a180"
            }
        },
        {
            "id": "_design/ddoc02",
            "key": "_design/ddoc02",
            "value": {
                "rev": "1-d942f0ce01647aa0f46518b213b5628e"
            }
        },
        {
            "id": "_design/ddoc03",
            "key": "_design/ddoc03",
            "value": {
                "rev": "1-721fead6e6c8d811a225d5a62d08dfd0"
            }
        },
        {
            "id": "_design/ddoc04",
            "key": "_design/ddoc04",
            "value": {
                "rev": "1-32c76b46ca61351c75a84fbcbceece2f"
            }
        },
        {
            "id": "_design/ddoc05",
            "key": "_design/ddoc05",
            "value": {
                "rev": "1-af856babf9cf746b48ae999645f9541e"
            }
        }
    ],
    "total_rows": 5
}
```

**Ответ**

**Рисунок 151**

### 4.4.9 Получить JSON-структуру всех проектных документов в данной базе данных, указав параметры строки запроса в теле запроса

Функциональность представления POST _design_docs поддерживает идентичные параметры и поведение, указанные в ППИ GET /{db}/_design_docs, но позволяет указывать параметры строки запроса в качестве ключей в объекте JSON в теле запроса POST.

#### 4.4.9.1. Формат запроса

Параметры запроса отображает Таблица 256.

**Таблица 256 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design_docs |
| Метод | POST |

Пример запроса отображает Рисунок 152.

```
POST /db/_design_docs HTTP/1.1
Accept: application/json
Content-Length: 70
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "keys" : [
        "_design/ddoc02",
        "_design/ddoc05"
    ]
}
```

**Запрос**
**Рисунок 152**

#### 4.4.9.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 153. Коды ошибок приведены в пункте 6.1.2.

```
{
    "total_rows" : 5,
    "rows" : [
        {
            "value" : {
                "rev" : "1-d942f0ce01647aa0f46518b213b5628e"
            },
            "id" : "_design/ddoc02",
            "key" : "_design/ddoc02"
        },
        {
            "value" : {
                "rev" : "1-af856babf9cf746b48ae999645f9541e"
            },
            "id" : "_design/ddoc05",
            "key" : "_design/ddoc05"
        }
    ],
    "offset" : 0
}
```

**Ответ**
**Рисунок 153**

#### 4.4.9.3. Отправка нескольких запросов к базе данных

Выполняет несколько запросов просмотра всех документов в этой базе данных. Это позволяет вместо нескольких запросов POST /{db}/_all_docs выполнить их в одном.

Множественные запросы также поддерживаются в /db/_local_docs/queries и /db/_design_docs/queries (аналогично /db/_all_docs/queries).

#### 4.4.9.3.1. Формат запроса

Параметры запроса отображает Таблица 257.

**Таблица 257 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_all_docs/queries |
| Метод | POST |

Параметры строки запроса отображает Таблица 258.

**Таблица 258 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 259.

**Таблица 259 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Accept | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 260.

**Таблица 260 – Объекты JSON запроса**

| Поле | Описание |
|---|---|
| queries | Массив объектов запросов с полями по параметрам каждого отдельного запроса представления, который будет выполняться. Имена полей и их значения аналогичны параметрам запроса обычного запроса _all_docs. |

Пример запроса отображает Рисунок 154.

```
POST /db/_all_docs/queries HTTP/1.1
Content-Type: application/json
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "queries": [
        {
            "keys": [
                "meatballs",
                "spaghetti"
            ]
        },
        {
            "limit": 3,
            "skip": 2
        }
    ]
}
```

**Запрос**
**Рисунок 154**

#### 4.4.9.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 261.

**Таблица 261 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа Transfer-Encoding, chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 262.

**Таблица 262 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| results | Массив объектов результатов по одному для каждого запроса. Каждый объект результата содержит те же поля, что и ответ на обычный запрос _all_docs. | Массив (Array) | Да |

Коды состояния отображает Таблица 263.

**Таблица 263 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Указанная база данных отсутствует |
| 500 Internal Server Error | Ошибка выполнения запроса |

Ответ в случае успешного завершения запроса отображает Рисунок 155. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 20 Dec 2017 11:17:07 GMT
ETag: «1H8RGBCK3ABY6ACDM7ZSC30QK»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "results" : [
        {
            "rows": [
                {
                    "id": "meatballs",
                    "key": "meatballs",
                    "value": 1
                },
                {
                    "id": "spaghetti",
                    "key": "spaghetti",
                    "value": 1
                }
            ],
            "total_rows": 3
        },
        {
            "offset" : 2,
            "rows" : [
                {
                    "id" : "Adukiandorangecasserole-microwave",
                    "key" : "Aduki and orange casserole microwave",
                    "value" : [
                        null,
                        "Aduki and orange casserole microwave"
                    ]
                },
                {
                    "id" : "Aioli-garlicmayonnaise",
                    "key" : "Aioli garlic mayonnaise",
                    "value" : [
                        null,
                        "Aioli garlic mayonnaise"
                    ]
                },
                {
                    "id" : "Alabamapeanutchicken",
                    "key" : "Alabama peanut chicken",
                    "value" : [
                        null,
                        "Alabama peanut chicken"
                    ]
                }
            ],
            "total_rows" : 2667
        }
    ]
}
```

**Ответ**

**Рисунок 155**

### 4.4.10 Получить несколько документов

Данный метод может быть вызван для массового запроса нескольких документов. Он хорошо подходит для получения определенной ревизии документов, как это делают, например, репликаторы, или для получения истории ревизий.

#### 4.4.10.1. Формат запроса

Параметры запроса отображает Таблица 264.

**Таблица 264 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_bulk_get |
| Метод | POST |

Параметры строки запроса отображает Таблица 265.

**Таблица 265 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 266.

**Таблица 266 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>multipart/related,<br>multipart/mixed | Строка (String) | Да |
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 267.

**Таблица 267 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| revs | Предоставляет историю ревизий | Логический тип (Boolean) | Нет |
| docs | Список объектов документов, с id, и опциональными параметрами rev и atts_since | Массив (Array) | Да |

Пример запроса отображает Рисунок 156.

```
POST /db/_bulk_get HTTP/1.1
Accept: application/json
Content-Type:application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "docs": [
        {
            "id": "foo"
            "rev": "4-753875d51501a6b1883a9d62b4d33f91",
        },
        {
            "id": "foo"
            "rev": "1-4a7e4ae49c4366eaed8edeaea8f784ad",
        },
        {
            "id": "bar",
        }
        {
            "id": "baz",
        }
    ]
}
```

**Запрос**
**Рисунок 156**

Пример запроса отображает Рисунок 157.

```
POST /db/_bulk_get HTTP/1.1
Accept: application/json
Content-Type:application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "docs": [
        {
            "id": "a"
        }
    ]
}
```

**Запрос**
**Рисунок 157**

#### 4.4.10.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 268.

**Таблица 268 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 269.

**Таблица 269 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| results | Массив результатов для каждой запрошенной пары документ/ревизия. Идентификатор ключа id содержит идентификатор запрошенного документа ID, docs содержит массив объектов из одного элемента, каждый из которых имеет либо ключ ошибки и значение, описывающее ошибку, либо ключ ok и связанное с ним значение запрошенного документа, с дополнительным свойством _revisions, в котором перечислены родительские ревизии, если revs=true. | Объект (Json Object) | Да |

Коды состояния отображает Таблица 270.

**Таблица 270 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | В запросе недопустимые данные JSON или недопустимый параметр запроса |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Неверное имя базы данных |
| 415 Unsupported Media Type | Плохое значение Content-Type |

Ответ в случае успешного завершения запроса отображают Рисунок 158 и Рисунок 159. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Mon, 19 Mar 2018 15:27:34 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "results": [
    {
      "id": "foo",
      "docs": [
        {
          "ok": {
            "_id": "foo",
            "_rev": "4-753875d51501a6b1883a9d62b4d33f91",
            "value": "this is foo",
            "_revisions": {
              "start": 4,
              "ids": [
                "753875d51501a6b1883a9d62b4d33f91",
                "efc54218773c6acd910e2e97fea2a608",
                "2ee767305024673cfb3f5af037cd2729",
                "4a7e4ae49c4366eaed8edeaea8f784ad"
              ]
            }
          }
        }
      ]
    },
    {
      "id": "foo",
      "docs": [
        {
          "ok": {
            "_id": "foo",
            "_rev": "1-4a7e4ae49c4366eaed8edeaea8f784ad",
            "value": "this is the first revision of foo",
            "_revisions": {
              "start": 1,
              "ids": [
                "4a7e4ae49c4366eaed8edeaea8f784ad"
              ]
            }
          }
        }
      ]
    },
```

**Ответ (начало)**
**Рисунок 158**

```
    {
      "id": "bar",
      "docs": [
        {
          "ok": {
            "_id": "bar",
            "_rev": "2-9b71d36dfdd9b4815388eb91cc8fb61d",
            "baz": true,
            "_revisions": {
              "start": 2,
              "ids": [
                "9b71d36dfdd9b4815388eb91cc8fb61d",
                "309651b95df56d52658650fb64257b97"
              ]
            }
          }
        }
      ]
    },
    {
      "id": "baz",
      "docs": [
        {
          "error": {
            "id": "baz",
            "rev": "undefined",
            "error": "not_found",
            "reason": "missing"
          }
        }
      ]
    }
  ]
}
```


**Ответ (окончание)**
**Рисунок 159**

Пример ответа с конфликтующим документом:

Ответ в случае успешного завершения запроса отображает Рисунок 160. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Mon, 19 Mar 2018 15:27:34 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "results": [
    {
      "id": "a",
      "docs": [
        {
          "ok": {
            "_id": "a",
            "_rev": "1-23202479633c2b380f79507a776743d5",
            "a": 1
          }
        },
        {
          "ok": {
            "_id": "a",
            "_rev": "1-967a00dff5e02add41819138abb3284d"
          }
        }
      ]
    }
  ]
}
```

**Ответ**
**Рисунок 160**

### 4.4.11 Создать или обновить несколько документов

ППИ для большей части документов позволяет создавать и обновлять несколько документов одновременно в рамках одного запроса. Основная операция аналогична созданию или обновлению одного документа, за исключением того, что передается структура и информация документа.

При создании новых документов идентификатор документа (_id) является необязательным.

Для обновления существующих документов необходимо указать идентификатор документа, информацию о ревизии (_rev) и новые значения документа.

В случае пакетного удаления документов все поля, такие как ID документа, информация о ревизии и статус удаления (_deleted), являются обязательными для заполнения.

#### 4.4.11.1. Формат запроса

Параметры запроса отображает Таблица 271.

**Таблица 271 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_bulk_docs |
| Метод | POST |

Параметры строки запроса отображает Таблица 272.

**Таблица 272 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 273.

**Таблица 273 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 274.

**Таблица 274 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| docs | Список объектов документов | Массив (Array) | Да |
| new_edits | Если указано значение false, то данный параметр запрещает базе данных присваивать им новые идентификаторы ревизий. По умолчанию установлено значение true. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 161.

```
POST /db/_bulk_docs HTTP/1.1
Accept: application/json
Content-Length: 109
Content-Type:application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "docs": [
        {
            "_id": "FishStew"
        },
        {
            "_id": "LambStew",
            "_rev": "2-0786321986194c92dd3b57dfbfc741ce",
            "_deleted": true
        }
    ]
}
```

**Запрос**
**Рисунок 161**

#### 4.4.11.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 275.

**Таблица 275 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Массив объектов JSON ответа отображает Таблица 276.

**Таблица 276 – Массив объектов JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа | Строка (String) | Да |
| rev | Маркер новой ревизии документа. Доступен, если документ сохранен без ошибок | Строка (String) | Нет |
| error | Тип ошибки | Строка (String) | Нет |
| reason | Причина ошибки | Строка (String) | Нет |

Коды состояния отображает Таблица 277.

**Таблица 277 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Документ был создан или обновлен |
| 400 Bad Request | В запросе JSON недопустимые данные |
| 404 Not Found | Запрашиваемая база данных не найдена |

Ответ в случае успешного завершения запроса отображает Рисунок 162. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 144
Content-Type: application/json
Date: Mon, 12 Aug 2013 00:15:05 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
[
    {
        "ok": true,
        "id": "FishStew",
        "rev":" 1-967a00dff5e02add41819138abb3284d"
    },
    {
        "ok": true,
        "id": "LambStew",
        "rev": "3-f9c62b2169d0999103e9f41949090807"
    }
]
```

**Ответ**
**Рисунок 162**

### 4.4.11.3. Массовая вставка документов

Каждый раз, когда документ сохраняется или обновляется в БД «Енисей», обновляется внутреннее B-дерево. Массовая вставка обеспечивает повышение эффективности как в области хранения, так и во времени за счет консолидации многих обновлений на промежуточных узлах B-дерева.

Он не предназначен для выполнения ACID-подобных транзакций в БД «Енисей», единственной границей транзакции в БД «Енисей» является одно обновление в одной базе данных.

Для массовой вставки документов в базу данных необходимо предоставить JSON-структуру с массивом документов, которые необходимо добавить в базу данных. Можно либо включить идентификатор документа, либо разрешить автоматическое создание идентификатора документа.

#### 4.4.11.3.1. Формат запроса

Например, следующее обновление вставляет три новых документа, два с предоставленными идентификаторами документов и один, для которого будет сгенерирован идентификатор документа:

Пример запроса отображает Рисунок 163.

```
POST /source/_bulk_docs HTTP/1.1
Accept: application/json
Content-Length: 323
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "docs": [
        {
            "_id": "FishStew",
            "servings": 4,
            "subtitle": "Delicious with freshly baked bread",
            "title": "FishStew"
        },
        {
            "_id": "LambStew",
            "servings": 6,
            "subtitle": "Serve with a whole meal scone topping",
            "title": "LambStew"
        },
        {
            "servings": 8,
            "subtitle": "Hand-made dumplings make a great accompaniment",
            "title": "BeefStew"
        }
    ]
}
```

**Запрос**
**Рисунок 163**

#### 4.4.11.3.2. Формат ответа на запрос

При массовой вставке будет возвращен ответ 201 Created (Создан), при этом содержимое возвращаемой структуры указывает на определенный успех или другие сообщения для каждого документа.

Возвращаемая структура из приведенного выше примера содержит список созданных документов, с комбинацией идентификаторов и их ревизий:

Ответ в случае успешного завершения запроса отображает Рисунок 164. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 215
Content-Type: application/json
Date: Sat, 26 Oct 2013 00:10:39 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
[
    {
        "id": "FishStew",
        "ok": true,
        "rev": "1-6a466d5dfda05e613ba97bd737829d67"
    },
    {
        "id": "LambStew",
        "ok": true,
        "rev": "1-648f1b989d52b8e43f05aa877092cc7c"
    },
    {
        "id": "00a271787f89c0ef2e10e88a0c0003f0",
        "ok": true,
        "rev": "1-e4602845fc4c99674f50b1d5a804fdfa"
    }
]
```

**Ответ**

**Рисунок 164**

### 4.4.11.4. Обновление документов в массовом порядке

Процедура массового обновления документов аналогична процедуре вставки, за исключением того, что необходимо указать идентификатор документа и текущую ревизию для каждого документа в строке JSON для оператора массового обновления.

#### 4.4.11.4.1. Формат запроса

Пример запроса отображает Рисунок 165.

```
POST /recipes/_bulk_docs HTTP/1.1
Accept: application/json
Content-Length: 464
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "docs": [
        {
            "_id": "FishStew",
            "_rev": "1-6a466d5dfda05e613ba97bd737829d67",
            "servings": 4,
            "subtitle": "Delicious with freshly baked bread",
            "title": "FishStew"
        },
        {
            "_id": "LambStew",
            "_rev": "1-648f1b989d52b8e43f05aa877092cc7c",
            "servings": 6,
            "subtitle": "Serve with a whole meal scone topping",
            "title": "LambStew"
        },
        {
            "_id": "BeefStew",
            "_rev": "1-e4602845fc4c99674f50b1d5a804fdfa",
            "servings": 8,
            "subtitle": "Hand-made dumplings make a great accompaniment",
            "title": "BeefStew"
        }
    ]
}
```

**Запрос**
**Рисунок 165**

#### 4.4.11.4.2. Формат ответа на запрос

Возвращаемая структура представляет собой JSON обновленных документов с новой ревизией и информацией об идентификаторе ID.

Ответ в случае успешного завершения запроса отображает Рисунок 166. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 215
Content-Type: application/json
Date: Sat, 26 Oct 2013 00:10:39 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
[
    {
        "id": "FishStew",
        "ok": true,
        "rev": "2-2bff94179917f1dec7cd7f0209066fb8"
    },
    {
        "id": "LambStew",
        "ok": true,
        "rev": "2-6a7aae7ac481aa98a2042718d09843c4"
    },
    {
        "id": "BeefStew",
        "ok": true,
        "rev": "2-9801936a42f06a16f16c30027980d96f"
    }
]
```

**Ответ**
**Рисунок 166**

Можно также удалять документы во время массового обновления, добавив поле _deleted со значением true к каждой комбинации ID документа/ревизии в переданной структуре JSON.

Тип возврата при массовой вставке будет 201 Created, а содержимое возвращаемой структуры будет указывать на конкретные сообщения об успехе или отказе для каждого документа.

#### 4.4.11.5. Семантика транзакций массовых документов

Операции с массовыми документами являются неделимыми. Это означает, что БД «Енисей» не гарантирует сохранение любого отдельнго документа, включенного в массовое обновление (или вставку) при отправке запроса.

##### 4.4.11.5.1. Формат ответа на запрос

Ответ будет содержать список документов, успешно вставленных или обновленных во время процесса. В случае сбоя часть документов может быть успешно сохранена, а другая потеряна. Параметр new _rev, указывающий на создание новой ревизии документа, покажет в структуре ответа, был ли документ обновлен. Если обновление не удалось, будет выдана ошибка типа conflict.

Ответ в случае успешного завершения запроса отображает Рисунок 167. Коды ошибок приведены в пункте 6.1.2.

```
[
    {
        "id" : "FishStew",
        "error" : "conflict",
        "reason" : "Document update conflict."
    },
    {
        "id" : "LambStew",
        "error" : "conflict",
        "reason" : "Document update conflict."
    },
    {
        "id" : "BeefStew",
        "error" : "conflict",
        "reason" : "Document update conflict."
    }
]
```


**Ответ**
**Рисунок 167**

В этом случае новая ревизия не была создана, и для обновления документа необходимо отправить обновление документа с правильным тегом ревизии.

Репликация документов не зависит от типа вставки или обновления. Документы и ревизии, созданные во время массовой вставки или обновления, реплицируются таким же образом, как и любые другие документы.

### 4.4.11.6. Валидация массового документа и ошибки конфликта

JSON, возвращаемый операцией _bulk_docs, состоит из массива структур JSON, по одной для каждого документа в исходном представлении. Необходимо изучить возвращенную структуру JSON, чтобы убедиться, что все документы, представленные в исходном запросе, были успешно добавлены в базу данных.

Если документ (или ревизия документа) не был правильно занесен в базу данных из-за ошибки, необходимо проверить поле ошибки, чтобы определить её тип и порядок действий. Ошибки будут одного из следующих типов:

1) Conflict.

Представленный документ находится в конфликте. Новая редакция не будет создана, и нужно будет повторно отправить документ в базу данных.

Разрешение конфликтов документов, добавленных с помощью интерфейса bulk docs, идентично процедурам разрешения, используемым при разрешении ошибок конфликтов во время репликации

2) forbidden

Записи с этим типом ошибки указывают на то, что процедура проверки, примененная к документу во время отправки, вернула ошибку.

Например, процедура проверки включает следующее:

```
throw({forbidden: 'invalid recipe ingredient'});
```

#### 4.4.11.6.1. Формат ответа на запрос

Ответ на ошибку отображает Рисунок 168.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 80
Content-Type: application/json
Date: Sat, 26 Oct 2013 00:05:17 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
[
    {
        "id": "LambStew",
        "error": "forbidden",
        "reason": "invalid recipe ingredient"
    }
]
```

**Ответ**
**Рисунок 168**

### 4.4.12 Найти документ

Поиск документов с использованием декларативного синтаксиса запросов в формате JSON. Запросы могут использовать встроенный индекс _all_docs или пользовательские индексы, указанные с помощью конечной точки _index.

#### 4.4.12.1. Формат запроса

Параметры запроса отображает Таблица 278.

**Таблица 278 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_find |
| Метод | POST |

Параметры строки запроса отображает Таблица 279.

**Таблица 279 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 280.

**Таблица 280 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 281.

**Таблица 281 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| selector | JSON-объект, описывающий критерии, используемые для отбора документов. | Объект (Json Object) | Да |
| limit | Максимальное количество возвращаемых результатов. По умолчанию 25. | Число (int64) | Нет |
| skip | Пропустить первые „n“ результатов, где „n“ указанное значение. | Число (int64) | Нет |
| sort | Массив JSON, соответствующий синтаксису сортировки. | Объект (Json Object) | Нет |
| fields | Массив JSON, указывающий, какие поля каждого объекта должны быть возвращены. Если является пустым, то возвращается весь объект. Более подробная информация представлена в разделе о фильтрации полей. | Массив (Array) | Нет |
| use_index | Указание запросу использовать определенный индекс. Указывается либо как «<design_document>», либо как [«<design_document>», «<index_name>»]. | Строка (String)\| Массив (Array) | Нет |
| conflicts | Включать документы с конфликтами, если значение true. Предназначен для легкого поиска противоречивых документов без индекса или представления. По умолчанию false. | Логический тип (Boolean) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| r | Кворум чтения, необходимый для результата. По умолчанию это значение равно 1, в этом случае возвращается документ, найденный по индексу. Если задано большее значение, каждый документ считывается как минимум из такого количества копий, прежде чем он будет возвращен в результате. Это может занять больше времени, чем использование только документа, хранящегося локально с индексом. Необязательный параметр, по умолчанию: 1 | Число (int64) | Нет |
| bookmark | Строка, позволяющая указать, какая страница результатов нужна. Используется для просмотра наборов результатов. Каждый запрос возвращает строку под ключом bookmark, которую можно передать обратно в запрос для получения следующей страницы результатов. Если какая-либо часть селекторного запроса изменяется между запросами, результаты неопределены. По умолчанию: null | Строка (String) | Нет |
| update | Обновлять ли индекс перед возвращением результата. По умолчанию: true. | Логический тип (Boolean) | Нет |
| stable | Должны ли результаты представления возвращаться из «постоянного» набора сегментов. | Логический тип (Boolean) | Нет |
| stale | Комбинация опций update=false и stable=true. Возможные варианты: «ok», false | Логический тип (Boolean) | Нет |
| execution_stats | Включить статистику выполнения в ответ на запрос. По умолчанию: false | Логический тип (Boolean) | Нет |

Пример тела запроса для поиска документов с помощью индекса отображает Рисунок 169.

```
POST /movies/_find HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: 168
Host: localhost:5984

{
    "selector": {
        "year": {"$gt": 2010}
    },
    "fields": ["_id", "_rev", "year", "title"],
    "sort": [{"year": "asc"}],
    "limit": 2,
    "skip": 0,
    "execution_stats": true
}
```

**Запрос**
**Рисунок 169**

### 4.4.12.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 282.

**Таблица 282 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 283.

**Таблица 283 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| docs | Массив документов, соответствующих поиску. В каждом документе перечисляются поля, указанные в части fields тела запроса, вместе с их значениями. | Объект (Json Object) | Да |
| warning | Предупреждения о выполнении | Строка (String) | Нет |
| execution_stats | Статистика выполнения | Объект (Json Object) | Да |
| bookmark | Строка, используемая для просмотра наборов результатов. Подробности использования отображает поле bookmark в запросе | Строка (String) | Нет |

Коды состояния отображает Таблица 284.

**Таблица 284 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Запрашиваемая база данных не найдена |
| 500 Internal Server Error | Ошибка выполнения запроса |

Значения limit и skip соответствуют ожиданиям. Хотя функция skip существует, она не предназначена для пролистывания. Причина в том, что функция закладок более эффективна.

Ответ в случае успешного завершения запроса отображает Рисунок 170. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Thu, 01 Sep 2016 15:41:53 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked

{
    "docs": [
        {
            "_id": "176694",
            "_rev": "1-54f8e950cc338d2385d9b0cda2fd918e",
            "year": 2011,
            "title": "The Tragedy of Man"
        },
        {
            "_id": "780504",
            "_rev": "1-5f14bab1a1e9ac3ebdf85905f47fb084",
            "year": 2011,
            "title": "Drive"
        }
    ],
    "execution_stats": {
        "total_keys_examined": 0,
        "total_docs_examined": 200,
        "total_quorum_docs_examined": 0,
        "results_returned": 2,
        "execution_time_ms": 5.52
    }
}
```

**Ответ**
**Рисунок 170**

#### 4.4.12.3. Синтаксис селектора

Селекторы выражаются в виде объекта JSON, описывающего интересующие документы. Внутри этой структуры можно применять условную логику, используя специально названные поля.

Хотя селекторы имеют некоторые сходства с документами запросов MongoDB, но они выражаются в сходстве целей и не обязательно распространяются на общность функций или результатов.

##### 4.4.12.3.1. Основы селекторов

Элементарный синтаксис селектора требует указания одного или нескольких полей и соответствующих значений, необходимых для этих полей. Этот селектор ищет все документы, поле «director» которых имеет значение «Lars von Trier».

Селектор отображает Рисунок 171.

```
{
    "director": "Lars von Trier"
}
```

**Селектор**
**Рисунок 171**

Простой селектор, проверяющий определенные поля отображает Рисунок 175.

```
"selector": {
  "title": "Live And Let Die"
},
"fields": [
  "title",
  "cast"
]
```

**Селектор**
**Рисунок 172**

Можно создать более сложные селекторные выражения, комбинируя операторы. Для достижения наилучшей производительности лучше всего комбинировать операторы «комбинации» или «логического массива», такие как $regex, с операторами равенства, такими как $eq, $gt, $gte, $lt и $lte (но не $ne).

##### 4.4.12.3.2. Селектор с 2 полями

Этот селектор соответствует любому документу с полем имени, содержащим «Paul», а также с полем местоположения со значением «Boston».

Селектор отображает Рисунок 173.

```
{
    "name": "Paul",
    "location": "Boston"
}
```

**Селектор**
**Рисунок 173**

#### 4.4.12.3.3. Подполя

Более сложный селектор позволяет указать значения для полей вложенных объектов, или подполей. Например, для указания поля и подполя можно использовать стандартную структуру JSON.

Пример селектора поля и подполя с использованием стандартной структуры JSON отображает Рисунок 174.

```
{
    "imdb": {
        "rating": 8
    }
}
```

**Селектор**
**Рисунок 174**

В сокращенном эквиваленте используется точечная нотация для объединения имен поля и подполя в одно имя, что отображает Рисунок 175.

```
{
    "imdb.rating": 8
}
```

**Селектор**
**Рисунок 175**

#### 4.4.12.3.4. Операторы

Операторы идентифицируются с помощью префикса в виде знака доллара ($) в поле имени.

В синтаксисе селектора есть два основных типа операторов:

Комбинированные операторы

Операторы условия

В общем случае операторы сочетания применяются на самом верхнем уровне выбора. Они используются для объединения условий или для создания комбинаций условий в одном селекторе.

Каждый явный оператор имеет вид, как отображает Рисунок 176.

```
{"$operator": argument}
```

**Селектор**
**Рисунок 176**

Селектор без явного оператора считается имеющим неявный оператор. Точный неявный оператор определяется структурой выражения селектора.

#### 4.4.12.3.5. Неявные операторы

Существует два неявных оператора:

Равенство

И

В селекторе любое поле, содержащее значение JSON, но не имеющее операторов, считается условием равенства. Неявный тест на равенство применяется также для полей и подполей.

Любой объект JSON, который не является аргументом оператора условия, представляет собой неявный оператор $and для каждого поля.

В приведенном ниже примере, который отображает Рисунок 177 мы используем оператор для поиска любого документа, в котором поле «year» имеет значение больше, чем 2010

Каждый явный оператор имеет вид, как отображает Рисунок 176.

```
{
    "year": {
        "$gt": 2010
    }
}
```

**Селектор**
**Рисунок 177**

В следующем примере в соответствующем документе должно быть поле «director», и это поле должно иметь значение, в точности равное «Lars von Trier». Пример отображает .Рисунок 178.

```
{
    "director": "Lars von Trier"
}
```

**Селектор**
**Рисунок 178**

Можно также сделать оператор равенства явным, что отображает Рисунок 179.

```
{
    "director": {
        "$eq": "Lars von Trier"
    }
}
```

**Селектор**
**Рисунок 179**

В следующем примере с использованием подполей обязательное поле «imdb» в соответствующем документе должно также иметь подполе «rating», и это подполе должно иметь значение, равное 8.

Пример применения неявного оператора к подполям отображает Рисунок 180.

```
{
    "imdb": {
        "rating": 8
    }
}
```

**Селектор**
**Рисунок 180**

Опять же, можно сделать оператор равенства явным, что отображает Рисунок 181.

```
{
    "imdb": {
        "rating": { "$eq": 8 }
    }
}
```

**Селектор**
**Рисунок 181**

Пример использования оператора $eq при полнотекстовом индексировании отображает Рисунок 182.

```
{
  "selector": {
    "year": {
      "$eq": 2001
    }
  },
  "sort": [
    "title:string"
  ],
  "fields": [
    "title"
  ]
}
```

**Селектор**
**Рисунок 182**

Пример использования оператора $eq с базой данных, проиндексированной по полю «year» отображает Рисунок 183.

```
{
  "selector": {
    "year": {
      "$eq": 2001
    }
  },
  "sort": [
    "year"
  ],
  "fields": [
    "year"
  ]
}
```

**Селектор**
**Рисунок 183**

В этом примере поле «director» должно присутствовать и содержать значение «Lars von Trier», а поле «year» должно существовать и иметь значение 2003, что отображает Рисунок 184.

```
{
    "director": "Lars von Trier",
    "year": 2003
}
```

**Селектор**
**Рисунок 184**

Можно сделать явными как оператор $and, так и оператор равенства.

Пример использования явных операторов $and и $eq отображает Рисунок 185.

```
{
    "$and": [
        {
            "director": {
                "$eq": "Lars von Trier"
            }
        },
        {
            "year": {
                "$eq": 2003
            }
        }
    ]
}
```

**Селектор**
**Рисунок 185**

#### 4.4.12.3.6. Явные операторы

Все операторы, кроме «Равенство» и «И», должны быть указаны в явном виде.

#### 4.4.12.3.7. Комбинированные операторы

Комбинированные операторы используются для объединения селекторов. Помимо обычных булевых операторов, встречающихся в большинстве языков программирования, есть три комбинированных оператора ($all, $elemMatch и $allMatch), которые помогают работать с массивами JSON, и один, работающий с картами JSON ($keyMapMatch).

Комбинированный оператор принимает один аргумент. Аргументом является либо другой селектор, либо массив селекторов.

Список комбинированных операторов отображает Таблица 285.

**Таблица 285 – Список комбинированных операторов**

| Оператор | Аргумент | Назначение |
|---|---|---|
| $and | Массив (Array) | Работает, если все селекторы в массиве совпадают. |
| $or | Массив (Array) | Работает, если совпадает любой из селекторов в массиве. Все селекторы должны использовать один и тот же индекс. |
| $not | Селектор | Работает, если заданный селектор не совпадает. |
| $nor | Массив (Array) | Работает, если ни один из селекторов в массиве не совпадает. |
| $all | Массив (Array) | Сравнивает значение массива, если оно содержит все элементы массива аргумента. |
| $elemMatch | Селектор | Подбирает и возвращает все документы, содержащие поле массива с хотя бы одним элементом, который соответствует всем заданным критериям запроса. |
| $allMatch | Селектор | Подбирает и возвращает все документы, содержащие поле массива, все элементы которого соответствуют всем заданным критериям запроса. |
| $keyMapMatch | Селектор | Подбирает и возвращает все документы, содержащие карту, в которой есть хотя бы один ключ, соответствующий всем заданным критериям запроса. |

##### 4.4.12.3.7.1. Оператор $and

Оператор $and используется с двумя полями, что отображает Рисунок 186.

```
{
  "selector": {
    "$and": [
      {
        "title": "Total Recall"
      },
      {
        "year": {
          "$in": [1984, 1991]
        }
      }
    ]
  },
  "fields": [
    "year",
    "title",
    "cast"
  ]
}
```

**Пример использования оператора $and**
**Рисунок 186**

Оператор $and работает, если все селекторы в массиве совпадают. Ниже приведен пример с использованием первичного индекса (_all_docs), что отображает Рисунок 187.

```
{
    "$and": [
        {
            "_id": { "$gt": null }
        },
        {
            "year": {
                "$in": [2014, 2015]
            }
        }
    ]
}
```

**Пример использования оператора $and с использование первичного индекса**
**Рисунок 187**

### 4.4.12.3.7.2. Оператор $or

Оператор $or используется, если совпадает любой из селекторов в массиве. Ниже приведен пример с индексом по полю «year», что отображает Рисунок 188.

```
{
    "year": 1977,
    "$or": [
        { "director": "George Lucas" },
        { "director": "Steven Spielberg" }
    ]
}
```

**Пример использования оператора $or**
**Рисунок 188**

#### 4.4.12.3.7.3. Оператор $not

Оператор $not используется, если заданный селектор не совпадает. Ниже приведен пример с индексом по полю «year», что отображает Рисунок 189.

```
{
    "year": {
        "$gte": 1900
    },
    "year": {
        "$lte": 1903
    },
    "$not": {
        "year": 1901
    }
}
```

**Пример использования оператора $not**
**Рисунок 189**

#### 4.4.12.3.7.4. Оператор $nor

Оператор $nor используется, если заданный селектор не совпадает. Ниже приведен пример с индексом по полю «year», что отображает Рисунок 190.

```
{
    "year": {
        "$gte": 1900
    },
    "year": {
        "$lte": 1910
    },
    "$nor": [
        { "year": 1901 },
        { "year": 1905 },
        {  "year": 1907 }
    ]
}
```

**Пример использования оператора $nor**
**Рисунок 190**

#### 4.4.12.3.7.5. Оператор $all

Оператор $all соответствует значению массива, если он содержит все элементы массива аргумента. Ниже приведен пример использования с первичным индексом (_all_docs), что отображает Рисунок 191.

```
{
    «_id»: {
        «$gt»: null
    },
    «genre»: {
        «$all»: [«Comedy»,»Short»]
    }
}
```

**Пример использования оператора $all**
**Рисунок 191**

#### 4.4.12.3.7.6. Оператор $elemMatch

Оператор $elemMatch подбирает и возвращает все документы, содержащие поле массива, хотя бы один элемент которого соответствует заданным критериям запроса. Ниже приведен пример с первичным индексом (_all_docs), что отображает Рисунок 192.

```
{
    "_id": {
        "$gt": null
    },
    "genre": {
        "$all": ["Comedy","Short"]
    }
}
```

**Пример использования оператора $elemMatch**

**Рисунок 192**

#### 4.4.12.3.7.7. Оператор $allMatch

Оператор $allMatch подбирает и возвращает все документы, содержащие поле массива, все элементы которого соответствуют заданным критериям запроса. Ниже приведен пример использования с первичным индексом (_all_docs), что отображает Рисунок 193.

```
{
    "_id": { "$gt": null },
    "genre": {
        "$allMatch": {
            "$eq": "Horror"
        }
    }
}
```

**Пример использования оператора $allMatch**

**Рисунок 193**

#### 4.4.12.3.7.8. Оператор $keyMapMatch

Оператор $keyMapMatch подбирает и возвращает все документы, содержащие карту, в которой есть хотя бы один ключ, соответствующий всем заданным критериям запроса. Ниже приведен пример с первичным индексом (_all_docs), что отображает Рисунок 194.

```
{
    "_id": { "$gt": null },
    "cameras": {
        "$keyMapMatch": {
            "$eq": "secondary"
        }
    }
}
```

**Пример использования оператора $keyMapMatch**

**Рисунок 194**

#### 4.4.12.3.8. Операторы условия

Операторы условия специфичны для поля и используются для оценки значения, хранящегося в этом поле. Например, базовый оператор $eq работает, если указанное поле содержит значение, равное заданному аргументу.

Чтобы оператор условия работал правильно, поле должно существовать в документе, чтобы селектор совпал. Например, $ne означает, что указанное поле должно существовать и не равно значению аргумента.

Поддерживаются основные операторы равенства и неравенства, характерные для большинства языков программирования. Используется строгое соответствие типов.

Кроме того, доступны некоторые «мета» операторы условий. Некоторые операторы условия принимают в качестве аргумента любое допустимое содержимое JSON. Другие операторы условия требуют, чтобы аргумент был в определенном формате JSON.

Список операторов условия отображает Таблица 286.

**Таблица 286 – Список операторов условия**

| Тип оператора | Оператор | Аргумент | Назначение |
|---|---|---|---|
| (Не)равенство | $lt | Объект (Json Object) | Поле меньше, чем аргумент |
| | $lte | Объект (Json Object) | Поле меньше или равно аргументу. |
| | $eq | Объект (Json Object) | Поле равно аргументу |
| | $ne | Объект (Json Object) | Поле не равно аргументу. |
| | $gte | Объект (Json Object) | Поле больше или равно аргументу. |
| | $gt | Объект (Json Object) | Поле больше, чем аргумент. |
| Объект (Object) | $exists | Логический тип (Boolean) | Проверяет, существует ли поле, независимо от его значения. |
| | $type | Строка (String) | Проверяет тип поля документа. Допустимые значения: «null», «boolean», «number», «string», «array» и «object». |
| Массив (Array) | $in | Массив значений Json (Array of Json values) | Поле документа должно существовать в представленном списке. |
| | $nin | Массив значений Json (Array of Json values) | Поле документа не должно существовать в представленном списке. |
| | $size | Число (int64) | Специальное условие для соответствия длине поля массива в документе. Поля, не являющиеся массивами, не могут соответствовать этому условию. |

| Тип оператора | Оператор | Аргумент | Назначение |
|---|---|---|---|
| Разное | $mod | [Делитель, Остаток] | Делитель и остаток являются целыми положительными или отрицательными числами. Нецелые значения приводят к ошибке 404. Подбирает документы, в которых поле % делитель == остаток является истинным, и только в том случае, если поле документа является целым числом. |
| | $regex | Строка (String) | Шаблон регулярного выражения для сопоставления с полем документа. Выполняется только в том случае, если поле является строковым значением и соответствует заданному регулярному выражению. Алгоритмы сопоставления основаны на библиотеке Perl Compatible Regular Expression (PCRE). |

Регулярные выражения не работают с индексами, поэтому их не следует использовать для фильтрации больших наборов данных. Однако их можно использовать для ограничения частичного индекса.

#### 4.4.12.3.9. Создание селекторных выражений

В общем случае, когда оператор принимает аргумент, этот аргумент может быть другим оператором со своими аргументами. Это позволяет нам строить более сложные селекторные выражения.

Однако в качестве основы запроса можно использовать только операторы равенства, такие как $eq, $gt, $gte, $lt и $lte (но не $ne). Необходимо включить в селектор хотя бы один из них.

##### 4.4.12.3.9.1. Формат запроса

Если попытаться выполнить запрос на поиск всех документов, имеющих поле afieldname, содержащее значение, начинающееся с буквы A, это вызовет предупреждение, поскольку ни один индекс не может быть использован, и база данных выполнит полное сканирование первичного индекса.

Пример запроса отображает Рисунок 195.

```
POST /movies/_find HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: 112
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "selector": {
        "afieldname": {"$regex": "^A"}
    }
}
```

**Запрос**
**Рисунок 195**

#### 4.4.12.3.9.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 196. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Thu, 01 Sep 2016 17:25:51 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "warning":"no matching index found, create an index to optimize
    query time",
    "docs":[
    ]
}
```

**Ответ**
**Рисунок 196**

При развертывании в production всегда рекомендуется создавать соответствующий индекс.

Большинство выражений селектора работают именно так, как можно ожидать для данного оператора. Но, например, сравнение строк выполняется с помощью ICU и может дать неожиданные результаты, если ожидалось ASCII упорядочивание. Более подробную информацию можно посмотреть в разделе Сопоставление представлений.

### 4.4.12.4. Синтаксис сортировки

Поле sort содержит список пар имен полей и направлений, выраженных в виде базового массива. Первая пара имени поля и направления это самый верхний уровень сортировки. Вторая пара, если она указана, является следующим уровнем сортировки.

Поле может быть любым, при желании для полей поддокумента можно использовать пунктирную нотацию.

Значение направления «asc» для восходящего и «desc» для нисходящего. Если опустить значение направления, будет использоваться значение по умолчанию «asc».

Пример сортировки по двум полям:

```
[{«fieldName1»: «desc»}, {«fieldName2»: «desc» }]
```

Пример, сортировка по 2 полям, предполагая направление по умолчанию для обоих:

```
[«fieldNameA», «fieldNameB»]
```

Типичным требованием является поиск некоторого содержимого с помощью селектора, а затем сортировка результатов в соответствии с указанным полем в нужном направлении.

Чтобы использовать сортировку, следует убедиться, что:

1) Хотя бы одно из полей сортировки включено в селектор.

2) Уже определен индекс, в котором все поля сортировки расположены в том же порядке.

3) Каждый объект в массиве сортировки имеет один ключ.

Если объект в массиве сортировки не имеет единственного ключа, результирующий порядок сортировки зависит от реализации и может измениться.

Find не поддерживает несколько полей с разными порядками сортировки, поэтому направления должны быть либо все по возрастанию, либо все по убыванию.

Для имен полей в текстовых сортировках поиска иногда необходимо указывать тип поля, например:

```
{ «<fieldname>:string»: «asc»}
```

Если возможно, делается попытка определить тип поля на основе селектора. В неоднозначных случаях тип поля должен быть указан явно.

Порядок сортировки не определен, если поля содержат разные типы данных. Это важное различие между текстовыми индексами и индексами представления.

#### 4.4.12.4.1. Формат запроса

Простой запрос с использованием сортировки отображает Рисунок 197.

```
{
    "selector": {"Actor_name": "Robert De Niro"},
    "sort": [{"Actor_name": "asc"}, {"Movie_runtime": "asc"}]
}
```

**Запрос**
**Рисунок 197**

#### 4.4.12.5. Фильтрация полей

Можно точно указать, какие поля возвращаются для документа при выборе из базы данных. Это дает два преимущества:

1) Результаты ограничиваются только теми частями документа, которые необходимы для разрабатываемого приложения.
2) Уменьшение размера ответа.

Возвращаемые поля задаются в виде массива.

В ответ включаются только определенные поля фильтра. При включении списка полей не происходит автоматического включения _id или других полей метаданных.

##### 4.4.12.5.1. Формат запроса

Пример выборочного извлечения полей из совпадающих документов отображает Рисунок 198.

```
{
    "selector": { "Actor_name": "Robert De Niro" },
    "fields": ["Actor_name", "Movie_year", "_id", "_rev"]
}
```

**Запрос**
**Рисунок 198**

#### 4.4.12.6. Пагинация

Запросы Mango поддерживают пагинацию через поле закладок. Каждый ответ _find содержит закладку маркер, который БД «Енисей» использует для определения того, откуда возобновлять поиск при последующих запросах. Чтобы получить следующий набор результатов запроса, добавьте к следующему запросу закладку, полученную в предыдущем ответе. Не забудьте оставить селектор прежним, иначе можно получить неожиданные результаты. Для постраничной перемотки назад можно использовать предыдущую закладку, чтобы вернуть предыдущий набор результатов.

Следует обратить внимание, что наличие закладки не гарантирует, что результатов будет больше. Можно проверить, удалось ли достигнуть конца набора результатов, сравнив количество возвращенных результатов с размером запрашиваемой страницы если количество возвращенных результатов < лимита, то больше результатов нет.

#### 4.4.12.7. Статистика выполнения

Find может возвращать основную статистику выполнения для конкретного запроса. В сочетании с конечной точкой _explain это должно дать некоторое представление о том, эффективно ли используются индексы.

Поля статистики выполнения отображает Таблица 287.

**Таблица 287 – Статистика выполнения**

| Тип оператора | Оператор |
|---|---|
| total_keys_examined | Количество исследуемых индексных ключей. В настоящее время всегда равно 0. |
| total_docs_examined | Количество документов, извлеченных из базы данных/индекса, эквивалентно использованию include_docs=true в представлении. Затем они могут быть отфильтрованы в памяти для дальнейшего сужения по набору результатов на основе селектора. |
| total_quorum_docs_examined | Количество документов, извлеченных из базы данных с помощью внеполосной выборки документов. Это значение ненулевое, только когда в параметрах запроса указан кворум чтения > 1. |
| results_returned | Количество результатов, возвращенных из запроса. В идеале это число не должно быть значительно меньше, чем общее количество рассмотренных документов/ключей. |
| execution_time_ms | Общее время выполнения в миллисекундах, измеренное базой данных. |
| total_keys_examined | Количество исследуемых индексных ключей. В настоящее время всегда равно 0. |
| total_docs_examined | Количество документов, извлеченных из базы данных/индекса, эквивалентно использованию include_docs=true в представлении. Затем они могут быть отфильтрованы в памяти для дальнейшего сужения по набору результатов на основе селектора. |
| total_quorum_docs_examined | Количество документов, извлеченных из базы данных с помощью внеполосной выборки документов. Это значение ненулевое, только когда в параметрах запроса указан кворум чтения > 1. |
| results_returned | Количество результатов, возвращенных из запроса. В идеале это число не должно быть значительно меньше, чем общее количество рассмотренных документов/ключей. |
| execution_time_ms | Общее время выполнения в миллисекундах, измеренное базой данных. |

### 4.4.13 Создать новый индекс для базы данных

Mango это декларативный язык запросов JSON для баз данных БД «Енисей». Mango включает в себя несколько типов индексов, начинающихся с первичного индекса. Индексы Mango с типом индекса json строятся с помощью MapReduce Views.

#### 4.4.13.1. Формат запроса

Параметры запроса отображает Таблица 288.

**Таблица 288 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_index |
| Метод | POST |

Параметры строки запроса отображает Таблица 289.

**Таблица 289 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 290.

**Таблица 290 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 291.

**Таблица 291 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| index | JSON-объект, определяющий индекс для создания. | Объект (Json Object) | Да |
| ddoc | Имя документа проектирования, в котором будет создан индекс. По умолчанию каждый индекс будет создан в своем собственном проектном документе. Для повышения эффективности индексы могут быть сгруппированы в проектные документы. Однако изменение одного индекса в проектном документе приведет к аннулированию всех остальных индексов в том же документе (аналогично представлениям). | Строка (String) | Нет |
| name | Имя индекса. Если имя не указано, оно будет сгенерировано автоматически. | Строка (String) | Нет |
| type | Принимает значения «json» или «text». По умолчанию используется json. Геопространственные индексы будут поддерживаться в будущем. Необязательные текстовые индексы поддерживаются через стороннюю библиотеку. | Строка (String) | Нет |
| partitioned | Определяет, является ли JSON-индекс разделенным или глобальным. Значением по умолчанию partitioned является свойство partitioned базы данных. Чтобы создать глобальный индекс разделенной базе данных, укажите false для поля «partitioned». Если указать true для поля «partitioned» для неразделенной базы данных, произойдет ошибка. | Логический тип (Boolean) | Нет |

Пример создания нового индекса для поля с именем foo отображает Рисунок 199.

```
POST /db/_index HTTP/1.1
Content-Type: application/json
Content-Length: 116
Host: localhost:5984
{
    "index": {
        "fields": ["foo"]
    },
    "name" : "foo-index",
    "type" : "json"
}
```

**Запрос**
**Рисунок 199**

Пример создания индекса с использованием всех доступных параметров запроса, что отображает Рисунок 200.

```
POST /db/_index HTTP/1.1
Content-Type: application/json
Content-Length: 396
Host: localhost:5984
{
    "index": {
        "partial_filter_selector": {
            "year": {
                "$gt": 2010
            },
            "limit": 10,
            "skip": 0
        },
        "fields": [
            "_id",
            "_rev",
            "year",
            "title"
        ]
    },
    "ddoc": "example-ddoc",
    "name": "example-index",
    "type": "json",
    "partitioned": false
}
```

**Запрос**
**Рисунок 200**

#### 4.4.13.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 292.

**Таблица 292 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 293.

**Таблица 293 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| result | Флаг, показывающий, был ли создан индекс или он уже существует. Принимает значения «created» или «exists». | Строка (String) | Да |
| id | Id документа проектирования, в котором был создан индекс. | Строка (String) | Да |
| name | Имя созданного индекса. | Строка (String) | Да |

Коды состояния отображает Таблица 294.

**Таблица 294 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Индекс создан успешно или уже существует |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение администратора |
| 404 Not Found | База данных не найдена |
| 500 Internal Server Error | Ошибка выполнения |

Объект Index представляет собой JSON-объект с полями, которые отображает Таблица 295.

**Таблица 295 – Объекты JSON**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| fields | Массив имен полей в соответствии с синтаксисом сортировки <find/sort>. Также допускаются вложенные поля, например, «person.name». | Массив (Array) | Нет |
| partial_filter_selector | Селектор для применения к документам во время индексирования, создающий частичный индекс <find/partial_indexes> | Объект (Json Object) | Нет |

Возвращаемый JSON подтверждает, что индекс был создан, что отображает Рисунок 201.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 96
Content-Type: application/json
Date: Thu, 01 Sep 2016 18:17:48 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "result":"created",
    "id":"_design/a5f4711fc9448864a13c81dc71e660b524d7410c",
    "name":"foo-index"
}
```

**Ответ**
**Рисунок 201**

По умолчанию индекс JSON будет включать все документы, в которых присутствуют индексируемые поля, включая те, которые имеют нулевые значения.

### 4.4.13.3. Частичные индексы

Частичные индексы позволяют фильтровать документы во время индексирования, что потенциально обеспечивает значительное повышение производительности для селекторов запросов, которые не могут быть четко отображены на диапазон в индексе.

#### 4.4.13.3.1. Формат запроса

Пример запроса отображает отображает Рисунок 202.

```
{
  "selector": {
    "status": {
      "$ne": "archived"
    },
    "type": "user"
  }
}
```

**Запрос**
**Рисунок 202**

Без частичного индекса для поиска всех документов «type»: «user», которые не имеют статуса «archived» (архивировано), потребуется полное сканирование индекса. Это происходит потому, что обычный индекс может использоваться только для поиска смежных строк, а оператор «$ne» не может этого гарантировать.

Чтобы улучшить время отклика, мы можем создать индекс, который исключает документы со статусом «status»: { «$ne»: «archived» } во время индексирования с помощью поля «partial_filter_selector», что отображает Рисунок 203.

```
POST /db/_index HTTP/1.1
Content-Type: application/json
Content-Length: 144
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
  "index": {
    "partial_filter_selector": {
      "status": {
        "$ne": "archived"
      }
    },
    "fields": ["type"]
  },
  "ddoc" : "type-not-archived",
  "type" : "json"
}
```

**Запрос**
**Рисунок 203**

Частичные индексы в настоящее время не используются планировщиком запросов, если они не указаны полем «use_index», поэтому нам необходимо изменить исходный запрос, что отображает Рисунок 204.

```
{
  "selector": {
    "status": {
      "$ne": "archived"
    },
    "type": "user"
  },
  "use_index": "type-not-archived"
}
```

**Запрос**
**Рисунок 204**

Технически, нам не нужно включать фильтр по полю «status» (статус) в селектор запроса частичный индекс гарантирует, что это всегда так но включение фильтра делает намерение селектора более ясным и облегчает улучшения в планировании запросов (например, автоматический выбор частичных индексов).

Индекс с полями используется только тогда, когда селектор включает все индексируемые поля. Например, если индекс содержит [«a». «b»], но селектор требует, чтобы в соответствующих документах существовало только поле [«a»], индекс не будет действителен для запроса. Однако все индексы можно рассматривать так, как будто они включают специальные поля _id и _rev. Их никогда не нужно указывать в селекторе запроса.

### 4.4.14 Получить список всех индексов в базе данных

GET-запрос к /db/_index позволяет получить список всех индексов в базе данных. В дополнение к информации, доступной через этот ППИ, индексы также хранятся в проектных документах.

#### 4.4.14.1. Формат запроса

Параметры запроса отображает Таблица 296.

**Таблица 296 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_index |
| Метод | GET |

Параметры строки запроса отображает Таблица 297.

**Таблица 297 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 298.

**Таблица 298 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 205.

```
GET /db/_index HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 205**

#### 4.4.14.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 299.

**Таблица 299 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 300.

**Таблица 300 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| total_rows | Количество индексов | Число (int64) | Да |
| indexes | Массив определений индексов | Объект (Json Object) | Да |

Коды состояния отображает Таблица 301.

**Таблица 301 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успех |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 500 Internal Server Error | Ошибка выполнения |

Формат объектов индексов отображает Таблица 302.

**Таблица 302 – Формат объектов индексов**

| Поле | Описание |
|---|---|
| ddoc | Идентификатор проектного документа, к которому относится индекс. Этот идентификатор может быть использован для получения проектного документа, содержащего индекс, путем выполнения GET-запроса к /db/ddoc, где ddoc значение этого поля. |
| name | Имя индекса. |
| type | Тип индекса. В настоящее время поддерживается только тип «json». |
| def | Определение индекса, содержащее индексируемые поля и порядок сортировки: по возрастанию или по убыванию. |

Ответ в случае успешного завершения запроса отображает Рисунок 206. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 238
Content-Type: application/json
Date: Thu, 01 Sep 2016 18:17:48 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "total_rows": 2,
    "indexes": [
    {
        "ddoc": null,
        "name": "_all_docs",
        "type": "special",
        "def": {
            "fields": [
                {
                    "_id": "asc"
                }
            ]
        }
    },
    {
        "ddoc": "_design/a5f4711fc9448864a13c81dc71e660b524d7410c",
        "name": "foo-index",
        "type": "json",
        "def": {
            "fields": [
                {
                    "foo": "asc"
                }
            ]
        }
    }
  ]
}
```

**Ответ**

**Рисунок 206**

### 4.4.15 Удалить индекс

#### 4.4.15.1. Формат запроса

Параметры запроса отображает Таблица 303.

**Таблица 303 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/_index/{designdoc}/json/{name} |
| Метод | DELETE |

Параметры строки запроса отображает Таблица 304.

**Таблица 304 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| designdoc | Имя документа проектирования |
| name | Имя индекса |

Параметры HTTP заголовка запроса отображает Таблица 305.

**Таблица 305 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 207.

```
DELETE /db/_index/_design/a5f4711fc9448864a13c81dc71e660b524d7410c/json/foo-index HTTP/1.1
Accept: */*
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 207**

#### 4.4.15.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 306.

**Таблица 306 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 307.

**Таблица 307 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | «true» в случае успеха. | Строка (String) | Да |

Коды состояния отображает Таблица 308.

**Таблица 308 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Успех |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение писателя |
| 404 Not Found | Индекс не найден |
| 500 Internal Server Error | Ошибка выполнения |

Ответ в случае успешного завершения запроса отображает Рисунок 208. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Thu, 01 Sep 2016 19:21:40 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 208**

### 4.4.16 Показать используемый в запросе индекс

Показывает, какой индекс используется в запросе. Параметры те же, что и у _find

#### 4.4.16.1. Формат запроса

Параметры запроса отображает Таблица 309.

**Таблица 309 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_explain |
| Метод | POST |

Параметры строки запроса отображает Таблица 310.

**Таблица 310 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 311.

**Таблица 311 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 209.

```
POST /movies/_explain HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-Length: 168
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "selector": {
        "year": {"$gt": 2010}
    },
    "fields": ["_id", "_rev", "year", "title"],
    "sort": [{"year": "asc"}],
    "limit": 2,
    "skip": 0
}
```

**Запрос**
**Рисунок 209**

### 4.4.16.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 312.

**Таблица 312 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 313.

**Таблица 313 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| dbname | Имя базы данных | Строка (String) | Да |
| index | Индекс, используемый для выполнения запроса | Объект (Json Object) | Да |
| selector | Используемый селектор запроса | Объект (Json Object) | Да |
| opts | Используемые параметры запроса | Объект (Json Object) | Да |
| limit | Используемый параметр limit | Число (int64) | Да |
| skip | Используемый параметр пропуска | Число (int64) | Да |
| fields | Поля, возвращаемые запросом | Массив (Array) | Да |
| range | Параметры диапазона, передаваемые базовому представлению | Объект (Json Object) | Да |

Коды состояния отображает Таблица 314.

**Таблица 314 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Некорректный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 500 Internal Server Error | Ошибка выполнения |

Ответ в случае успешного завершения запроса отображает Рисунок 210. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Thu, 01 Sep 2016 15:41:53 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "dbname": "movies",
    "index": {
        "ddoc": "_design/0d61d9177426b1e2aa8d0fe732ec6e506f5d443c",
        "name": "0d61d9177426b1e2aa8d0fe732ec6e506f5d443c",
        "type": "json",
        "def": {
            "fields": [
                {
                    "year": "asc"
                }
            ]
        }
    },
    "selector": {
        "year": {
            "$gt": 2010
        }
    },
    "opts": {
        "use_index": [],
        "bookmark": "nil",
        "limit": 2,
        "skip": 0,
        "sort": {},
        "fields": [
            "_id",
            "_rev",
            "year",
            "title"
        ],
        "r": [
            49
        ],
        "conflicts": false
    },
    "limit": 2,
    "skip": 0,
    "fields": [
        "_id",
        "_rev",
        "year",
        "title"
    ],
    "range": {
        "start_key": [
            2010
        ],
        "end_key": [
            {}
        ]
    }
}
```

**Ответ**

**Рисунок 210**

#### 4.4.16.3. Index selection

Поле индекса описывает Таблица 315.

**Таблица 315 – Поле индекса**

| Поле | Описание |
|---|---|
| _find | выбирает, какой индекс использовать для ответа на запрос, если не указать индекс во время запроса. |

Планировщик запросов просматривает раздел селектора и находит индекс с наиболее близким соответствием операторам и полям, используемым в запросе. Если есть два или более индекса типа json, которые совпадают, предпочтение отдается индексу с наименьшим количеством полей в индексе. Если все еще есть два или более индекса-кандидата, выбирается индекс с первым по алфавиту именем.

Хорошей практикой является явное указание индексов в запросах. Это предотвращает воздействие на существующие запросы новыми индексами, которые могут быть добавлены в рабочей среде.

### 4.4.17 Получить список сегментов базы данных

Ответ будет содержать список сегментов базы данных. Каждый сегмент будет иметь свой внутренний диапазон базы данных, а также узлы, на которых хранятся реплики этих сегментов.

#### 4.4.17.1. Формат запроса

Параметры запроса отображает Таблица 316.

**Таблица 316 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_shards |
| Метод | GET |

Параметры строки запроса отображает Таблица 317.

**Таблица 317 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 318.

**Таблица 318 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 211.

```
GET /db/_shards HTTP/1.1
Accept: */*
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 211**

#### 4.4.17.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 319.

**Таблица 319 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 320.

**Таблица 320 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| shards | Отображение диапазонов сегментов на отдельные реплики сегментов на каждом узле кластера | Объект (Json Object) | Да |

Коды состояния отображает Таблица 321.

**Таблица 321 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуется привилегия чтения |
| 415 Unsupported Media Type | Плохое значение Content-Type |
| 500 Internal Server Error | Внутренняя ошибка сервера или таймаут |

Ответ в случае успешного завершения запроса отображает Рисунок 212. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 621
Content-Type: application/json
Date: Fri, 18 Jan 2019 19:55:14 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "shards": {
    "00000000-1fffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "20000000-3fffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "40000000-5fffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "60000000-7fffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "80000000-9fffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "a0000000-bfffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "c0000000-dfffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ],
    "e0000000-ffffffff": [
      "couchdb@node1.example.com",
      "couchdb@node2.example.com",
      "couchdb@node3.example.com"
    ]
  }
}
```

**Ответ**
**Рисунок 212**

### 4.4.18 Получить информацию о конкретном сегменте

Возвращает информацию о конкретном сегменте, в котором хранится данный документ, а также информацию об узлах, на которых этот сегмент имеет реплику.

#### 4.4.18.1. Формат запроса

Параметры запроса отображает Таблица 322.

**Таблица 322 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_shards/{docid} |
| Метод | GET |

Параметры строки запроса отображает Таблица 323.

**Таблица 323 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 324.

**Таблица 324 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 213.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 94
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Date: Fri, 18 Jan 2019 20:08:07 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
```

**Запрос**
**Рисунок 213**

### 4.4.18.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 325.

**Таблица 325 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 326.

**Таблица 326 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| range | Диапазон сегментов, в котором хранится документ | Строка (String) | Да |
| nodes | Список узлов, обслуживающих реплику сегмента | Массив (Array) | Да |

Коды состояния отображает Таблица 327.

**Таблица 327 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуется привилегия на чтение |
| 404 Not Found | База данных или документ не найдены |
| 500 Internal Server Error | Внутренняя ошибка сервера или тайм-аут |

Ответ в случае успешного завершения запроса отображает Рисунок 214. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 94
Content-Type: application/json
Date: Fri, 18 Jan 2019 20:26:33 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "range": "e0000000-ffffffff",
  "nodes": [
    "node1@127.0.0.1",
    "node2@127.0.0.1",
    "node3@127.0.0.1"
  ]
}
```

**Ответ**

**Рисунок 214**

### 4.4.19 Запустить внутреннюю синхронизацию сегментов для всех реплик

Для заданной базы данных принудительно запускает внутреннюю синхронизацию сегментов для всех реплик всех сегментов базы данных.

Обычно используется только при обслуживании кластера, таком как перемещение сегмента.

#### 4.4.19.1. Формат запроса

Параметры запроса отображает Таблица 328.

**Таблица 328 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_sync_shards |
| Метод | POST |

Параметры строки запроса отображает Таблица 329.

**Таблица 329 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 330.

**Таблица 330 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 215.

```
POST /db/_sync_shards HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
Accept: */*
```

**Запрос**

**Рисунок 215**

#### 4.4.19.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 331.

**Таблица 331 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 332.

**Таблица 332 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции. Доступно в случае успеха | Логический тип (Boolean) | Нет |
| error | Тип ошибки. Доступно, если код ответа 4xx | Строка (String) | Нет |
| reason | Описание ошибки. Доступно, если код ответа 4xx | Строка (String) | Нет |

Коды состояния отображает Таблица 333.

**Таблица 333 – Коды состояния**

| Код | Описание |
|---|---|
| 202 Accepted | Запрос принят |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | База данных не найдена |
| 500 Internal Server Error | Внутренняя ошибка сервера или таймаут |

Ответ в случае успешного завершения запроса отображает Рисунок 216. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 202 Accepted
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Fri, 18 Jan 2019 20:19:23 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Request-ID: 14f0b8d252
X-CouchDB-Body-Time: 0
{
    "ok": true
}
```

**Ответ**
**Рисунок 216**

Администраторы могут захотеть увеличить значение [mem3] sync_concurrency до большего значения на время синхронизации сегментов.

### 4.4.20 Получить отсортированный список изменений, внесенных в документы

Возвращает отсортированный список изменений, внесенных в документы в базе данных по порядку, может быть получен из ресурса _changes базы данных. Гарантируется предоставление только самого последнего изменения для данного документа, например, если в документ были добавлены поля, а затем удалены, клиент ППИ, проверяющий изменения, не обязательно получит промежуточное состояние добавленных документов.

Можно использовать для просмотра обновлений и изменений в базе данных для последующей обработки или синхронизации, а для практических целей постоянно подключенный канал _changes является разумным подходом для создания журнала в реальном времени большинству приложений.

#### 4.4.20.1. Формат запроса

Параметры запроса отображает Таблица 334.

**Таблица 334 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_changes |
| Метод | GET |

Параметры строки запроса отображает Таблица 335.

**Таблица 335 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 336.

**Таблица 336 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/event-stream,<br>text/plain | Строка (String) | Да |
| Last-Event-ID | ID последнего события, полученного сервером при предыдущем соединении. Переопределяет параметр запроса since. | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 337.

**Таблица 337 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| doc_ids | Список идентификаторов документов для фильтрации потока изменений в виде правильного массива JSON. Используется с фильтром _doc_ids. Поскольку длина URL ограничена, лучше использовать POST /{db}/_changes. | Массив (Array) | Нет |
| conflicts | Включает в ответ информацию о конфликтах. Игнорируется, если include_docs не true. По умолчанию false. | Логический тип (Boolean) | Нет |
| descending | Возвращает результаты изменений в порядке убывания (самое последнее изменение первым). По умолчанию false. | Логический тип (Boolean) | Нет |
| feed | normal Устанавливает обычный режим опроса. Все прошлые изменения возвращаются немедленно. Параметр, установленный по умолчанию.<br>longpoll Устанавливает режим длительного опроса. Ждет, пока произойдет хотя бы одно изменение, отправляет изменение, затем закрывает соединение. Чаще всего используется в сочетании с since=now для ожидания следующего изменения.<br>continuous Устанавливает непрерывный режим. Отправляет строку JSON на каждое событие. Держит сокет открытым до истечения времени.<br>eventsource Устанавливает режим источника событий. Работает так же, как и в непрерывном режиме, но отправляет события в формате EventSource. | Строка (String) | Нет |
| filter | Ссылка на функцию фильтрации из документа проектирования, которая будет фильтровать весь поток, выдавая только отфильтрованные события. | Строка (String) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| heartbeat | Период в миллисекундах, по истечению которого в результатах отправляется пустая строка. Применяется только для потоков longpoll, continuous и eventsource. Отменяет любой таймаут, чтобы поток оставался действующим неограниченное время. По умолчанию 60000. Может иметь значение true, чтобы использовать значение по умолчанию. | Число (int64) | Нет |
| include_docs | Включает связанный документ в каждый результат. Если есть конфликты, возвращается только победившая ревизия. По умолчанию false. | Логический тип (Boolean) | Нет |
| attachments | Включает Base64-кодированное содержимое вложений в документы, которые включаются, если include_docs равно true. Игнорируется, если include_docs не является true. По умолчанию значение параметра false. | Логический тип (Boolean) | Нет |
| att_encoding_info | Включает информацию о кодировке в заглушки вложений, если include_docs true и данное вложение сжато. Игнорируется, если значение include_docs не true. По умолчанию false. | Логический тип (Boolean) | Нет |
| last-event-id | Псевдоним заголовка Last-Event-ID. | Число (int64) | Нет |
| limit | Ограничивает количество строк результатов указанным значением (Следует обратить внимание, что использование 0 здесь имеет тот же эффект, что и 1). | Число (int64) | Нет |
| since | Запускает результаты из изменений сразу после заданной последовательности обновления. Может быть действительной последовательностью обновления или значением now. По умолчанию 0. | | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| style | Указывает, сколько ревизий будет возвращено в массив изменений. По умолчанию, main_only, будет возвращаться только текущая «выигрышная» ревизия; all_docs будет возвращать все ревизии листа (включая конфликты и удаленные бывшие конфликты). | Строка (String) | Нет |
| timeout | Максимальный период в миллисекундах для ожидания изменений перед отправкой ответа, даже если результатов нет. Применимо только для longpoll или continuous. Значение по умолчанию задается параметром конфигурации chttpd/changes_timeout. Следует обратить внимание, что значение 60000 также является максимальным таймаутом по умолчанию для предотвращения необнаруженных неработающих соединений. | Число (int64) | Нет |
| view | Позволяет использовать функции просмотра в качестве фильтров. Документы считаются «пройденными» для фильтра вида в том случае, если функция map выдает для них хотя бы одну запись. | Строка (String) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| seq_interval | При пакетной выборке изменений установка параметра seq_interval указывает БД «Енисей» вычислять seq обновления только после каждого N-го полученного результата. Установка параметра seq_interval=<batch size> , где <batch size> количество результатов, запрашиваемых на пакет, позволяет снизить нагрузку на исходную базу данных БД «Енисей»; вычисление значения seq по большому количеству шардов (особенно в базах данных с высоким уровнем разделения) является дорогостоящим в высоконагруженном кластере БД «Енисей». | Число (int64) | Нет |

Пример запроса отображает Рисунок 217.

```
GET /db/_changes?style=all_docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 217**

#### 4.4.20.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 338.

**Таблица 338 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Cache-Control | no-cache, если лента изменений является источником событий | Строка (String) | Да |
| Content-Type | application/json,<br>text/event-stream,<br>text/plain; charset=utf-8. | Строка (String) | Да |
| ETag | хэш ответа, если лента изменений является нормальной. | Строка (String) | Да |
| Transfer-Encoding | chunked. | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 339.

**Таблица 339 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| last_seq | Последовательность обновления последних изменений | Объект (Json Object) | Да |
| pending | Количество оставшихся элементов в ленте | Число (int64) | Да |
| results | Изменения, внесенные в базу данных | Массив (Array) | Да |

Коды состояния отображает Таблица 340.

**Таблица 340 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неправильный запрос |

Поле результатов изменений в базе данных отображает Таблица 341.

**Таблица 341 – JSON-объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| changes | Список листьев документа с одним полем rev. | Массив (Array) | Да |
| id | Идентификатор документа. | Строка (String) | Да |
| seq | Последовательность обновления. | Объект (Json Object) | Да |
| deleted | true, если документ удален. | Логический тип (Boolean) | Нет |

Ответ в случае успешного завершения запроса отображает Рисунок 218. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Mon, 12 Aug 2013 00:54:58 GMT
ETag: «6ASLEKEMSRABT0O5XY9UPO9Z»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "last_seq": "5-g1AAAAIreJyVkEsKwjAURZ-
toI5cgq5A0sQ0OrI70XyppcaRY92J7kR3ojupaSPUUgotgRd4yTlwbw4A0zRUMLdnpaMkwmyF3Ily9xBwEIuiKLI05KOTW0wkV4r
ruP29UyGWbordzwKVxWBNOGMKZhertDlarbr5pOT3DV4gudUC9-
MPJX9tpEAYx4TQASns2E24ucuJ7rXJSL1BbEgf3vTwpmedCZkYa7Pulck7Xt7x_usFU2aIHOD4eEfVTVA5KMGUkqhNZV-8_o5i",
    "pending": 0,
    "results": [
        {
            "changes": [
                {
                    "rev": "2-7051cbe5c8faecd085a3fa619e6e6337"
                }
            ],
            "id": "6478c2ae800dfc387396d14e1fc39626",
            "seq": "3-
g1AAAAG3eJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MSGXAqSVIAkkn2IFUZzIkMuUAee5pRqnGiuXkKA2dp
XkpqWmZeagpu_Q4g_fGEbEkAqaqH2sIItsXAyMjM2NgUUwdOU_JYgCRDA5ACGjQfn30QlQsgKvcjfGaQZmaUmmZClM8gZhyAmHGf
sG0PICrBPmQC22ZqbGRqamyIqSsLAAArcXo"
        },
        {
            "changes": [
                {
                    "rev": "3-7379b9e515b161226c6559d90c4dc49f"
                }
            ],
            "deleted": true,
            "id": "5bbc9ca465f1b0fcd62362168a7c8831",
            "seq": "4-
g1AAAAHXeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBMZc4EC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HqQ_kQG3qgSQqnoUtxoYGZkZG5uS4NY8FiDJ0ACkgAbNx2cfROUCiMr9CJ8ZpJkZpaaZ
EOUziBkHIGbcJ2zbA4hKsA-ZwLaZGhuZmhobYurKAgCz33kh"
        },
        {
            "changes": [
                {
                    "rev": "6-460637e73a6288cb24d532bf91f32969"
                },
                {
                    "rev": "5-eeaa298781f60b7bcae0c91bdedd1b87"
                }
            ],
            "id": "729eb57437745e506b333068fff665ae",
            "seq": "5-g1AAAAIReJyVkE0OgjAQRkcwUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloRid3MMkEUoJHbXbOxVy6
arc_SxQWQzRVHCuYHaxSpuj1aqbj0t-3-AlSrZakn78oeSvjRSIkIhSNiCFHbsKN3c50b02mURvEB-
yD296eNOzzoRMRLRZ98rkHS_veGcC_nR-fGe1gaCaxihhjOI2lX0BhniHaA"
        }
    ]
}
```

**Ответ**

**Рисунок 218**

Если указанные реплики шардов в каком-либо значении since недоступны, выбираются альтернативные реплики, и используется последняя известная контрольная точка между ними. Если это произойдет, Можно снова увидеть изменения, которые уже наблюдались ранее. Поэтому приложение, использующее поток _changes, должно быть «идемпотентным», то есть способным получать одни и те же данные несколько раз безопасно.

Cloudant Sync и PouchDB уже оптимизируют процесс репликации, устанавливая параметр seq_interval на количество результатов, ожидаемых в одной партии. Этот параметр увеличивает пропускную способность за счет уменьшения задержки между последовательными запросами при массовой передаче документов. Это позволило повысить производительность репликации до 20% в базах данных с высокой степенью задержки.

Использование параметра вложений для включения вложений в ленту изменений не рекомендуется при больших размерах вложений. Также Следует обратить внимание, что используемое Base64-кодирование приводит к 33%-ному превышению (т.е. одной трети) размера передачи для вложений.

Результаты, возвращаемые _changes, частично упорядочены. Другими словами, не гарантируется сохранение порядка при многократных вызовах.

### 4.4.21 Запросить ленту изменений в базе данных с параметром запроса

Запрашивает ленту изменений в базе данных так же, как и GET /{db}/_changes, но широко используется с параметром запроса ?filter=_doc_ids и позволяет передать больший список идентификаторов документов для фильтрации.

#### 4.4.21.1. Формат запроса

Параметры запроса отображает Таблица 342.

**Таблица 342 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_changes |
| Метод | POST |

Пример запроса отображает Рисунок 219.

```
POST /recipes/_changes?filter=_doc_ids HTTP/1.1
Accept: application/json
Content-Length: 40
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "doc_ids": [
        "SpaghettiWithMeatballs"
    ]
}
```

**Запрос**
**Рисунок 219**

#### 4.4.21.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 220. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 28 Sep 2013 07:23:09 GMT
ETag: «ARIHFWL3I7PIS0SPVTFU6TLR2»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "last_seq": "5-g1AAAAIreJyVkEsKwjAURZ-
toI5cgq5A0sQ0OrI70XyppcaRY92J7kR3ojupaSPUUgotgRd4yTlwbw4A0zRUMLdnpaMkwmyF3Ily9xBwEIuiKLI05KOTW0wkV4r
ruP29UyGWbordzwKVxWBNOGMKZhertDlarbr5pOT3DV4gudUC9-
MPJX9tpEAYx4TQASns2E24ucuJ7rXJSL1BbEgf3vTwpmedCZkYa7Pulck7Xt7x_usFU2aIHOD4eEfVTVA5KMGUkqhNZV8_o5i",
    "pending": 0,
    "results": [
        {
            "changes": [
                {
                    "rev": "13-bcb9d6388b60fd1e960d9ec4e8e3f29e"
                }
            ],
            "id": "SpaghettiWithMeatballs",
            "seq":  "5-g1AAAAIReJyVkE0OgjAQRkcwUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloRid3MMkEUoJHbXbOxVy6
arc_SxQWQzRVHCuYHaxSpuj1aqbj0t-3-AlSrZakn78oeSvjRSIkIhSNiCFHbsKN3c50b02mURvEB-
yD296eNOzzoRMRLRZ98rkHS_veGcC_nR-fGe1gaCaxihhjOI2lX0BhniHaA"
        }
    ]
}
```

**Ответ**
**Рисунок 220**

#### 4.4.21.3. Изменения

#### 4.4.21.3.1. Опрос

По умолчанию все изменения немедленно возвращаются в теле JSON:

#### 4.4.21.3.1.1. Формат запроса

Пример запроса отображает Рисунок 221.

```
GET /somedatabase/_changes HTTP/1.1
```

**Запрос**
**Рисунок 221**

results это список изменений в последовательном порядке. Новые и измененные документы отличаются только значением rev; удаленные документы включают атрибут «deleted»: true. (В режиме style=all_docs, deleted применяется только к текущей/победившей ревизии. Другие перечисленные ревизии могут быть удалены даже при отсутствии свойства deleted; чтобы убедиться в этом, необходимо получить (GET) их по отдельности).

last_seq последовательность обновления последнего возвращенного обновления (эквивалентно последнему элементу в результатах).

Отправку параметра since в строке запроса, который позволяет пропустить все изменения до заданной последовательности обновлений включительно отображает Рисунок 222.

```
GET /somedatabase/_changes?since=4-
g1AAAAHXeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBMZc4EC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HqQ_kQG3qgSQqnoUtxoYGZkZG5uS4NY8FiDJ0ACkgAbNx2cfROUCiMr9CJ8ZpJkZpaaZ
EOUziBkHIGbcJ2zbA4hKsA-ZwLaZGhuZmhobYurKAgCz33kh HTTP/1.1
```

**Запрос**
**Рисунок 222**

#### 4.4.21.3.1.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 223. Коды ошибок приведены в пункте 6.1.2.

```
{"results":[
{"seq":"1-
g1AAAAF9eJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P__7MSGXAqSVIAkkn2IFUZzIkMuUAee5pRqnGiuXkKA2dpX
kpqWmZeagpu_Q4g_fGEbEkAqaqH2sIItsXAyMjM2NgUUwdOU_JYgCRDA5ACGjQfn30QlQsgKvcTVnkAovI-
YZUPICpBvs0CAN1eY_c",
"id":"fresh",
"changes":
[{
"rev":"1-967a00dff5e02add41819138abb3284d"}]},
{
"seq":"3-
g1AAAAG3eJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MSGXAqSVIAkkn2IFUZzIkMuUAee5pRqnGiuXkKA2dp
XkpqWmZeagpu_Q4g_fGEbEkAqaqH2sIItsXAyMjM2NgUUwdOU_JYgCRDA5ACGjQfn30QlQsgKvcjfGaQZmaUmmZClM8gZhyAmHGf
sG0PICrBPmQC22ZqbGRqamyIqSsLAAArcXo",
"id":"updated",
"changes":
[{
"rev":"2-
7051cbe5c8faecd085a3fa619e6e6337CFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloRid3MMkEUoJHbXbOxVy6
arc_SxQWQzRVHCuYHaxSpuj1aqbj0t-3-AlSrZakn78oeSvjRSIkIhSNiCFHbsKN3c50b02mURvEB-
yD296eNOzzoRMRLRZ98rkHS_veGcC_nR-fGe1gaCaxihhjOI2lX0BhniHaA",
"id":"deleted",
"changes":
[{
"rev":"2-eec205a9d413992850a6e32678485900"}],
"deleted":true}
],
"last_seq":"5-g1AAAAIreJyVkEsKwjAURZ-
toI5cgq5A0sQ0OrI70XyppcaRY92J7kR3ojupaSPUUgotgRd4yTlwbw4A0zRUMLdnpaMkwmyF3Ily9xBwEIuiKLI05KOTW0wkV4r
ruP29UyGWbordzwKVxWBNOGMKZhertDlarbr5pOT3DV4gudUC9-
MPJX9tpEAYx4TQASns2E24ucuJ7rXJSL1BbEgf3vTwpmedCZkYa7Pulck7Xt7x_usFU2aIHOD4eEfVTVA5KMGUkqhNZV-8_o5i",
"pending": 0}
```


**Ответ**
**Рисунок 223**

Структура возврата для режимов normal и longpoll представляет собой массив JSON объектов changes и последовательность последнего обновления.

В формате возврата для непрерывного (continuous) режима сервер отправляет строку с разделителем CRLF (возврат каретки, перевод строки) для каждого изменения. Каждая строка содержит объект JSON, описанный выше.

Можно также запросить полное содержимое каждого изменения документа (вместо только уведомления об изменении) с помощью параметра include_docs, что отображает Рисунок 224.

```
{
    "last_seq": "5-g1AAAAIreJyVkEsKwjAURZ-
toI5cgq5A0sQ0OrI70XyppcaRY92J7kR3ojupaSPUUgotgRd4yTlwbw4A0zRUMLdnpaMkwmyF3Ily9xBwEIuiKLI05KOTW0wkV4r
ruP29UyGWbordzwKVxWBNOGMKZhertDlarbr5pOT3DV4gudUC9-
MPJX9tpEAYx4TQASns2E24ucuJ7rXJSL1BbEgf3vTwpmedCZkYa7Pulck7Xt7x_usFU2aIHOD4eEfVTVA5KMGUkqhNZV-8_o5i",
    "pending": 0,
    "results": [
        {
            "changes": [
                {
                    "rev": "2-eec205a9d413992850a6e32678485900"
                }
            ],
            "deleted": true,
            "id": "deleted",
            "seq":  "5-g1AAAAIReJyVkE0OgjAQRkcwUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloRid3MMkEUoJHbXbOxVy6
arc_SxQWQzRVHCuYHaxSpuj1aqbj0t-3-
AlSrZakn78oeSvjRSIkIhSNiCFHbsKN3c50b02mURvEByD296eNOzzoRMRLRZ98rkHS_veGcC_nR-
fGe1gaCaxihhjOI2lX0BhniHaA",
        }
    ]
}
```


**Ответ**

**Рисунок 224**

### 4.4.21.3.2. Длительный опрос

Лента longpoll, вероятно, наиболее применима для браузера, является более эффективной формой опроса, которая ожидает изменения перед отправкой ответа. longpoll позволяет избежать необходимости часто опрашивать БД «Енисей», чтобы обнаружить, что ничего не изменилось!

Запрос к серверу будет оставаться открытым до тех пор, пока в базе данных не произойдет изменение, которое впоследствии будет передано, после чего соединение закроется. Это низкая нагрузка как для сервера, так и для клиента.

Ответ это в основном тот же JSON, который отправляется для обычной ленты.

Поскольку ожидание изменения может быть значительным, Можно установить таймаут перед автоматическим закрытием соединения (аргумент timeout). Можно также задать интервал (с помощью аргумента запроса heartbeat), через который посылается новая строка для поддержания соединения активным.

Помните, что heartbeat означает «Посылать строку каждые x мс, если не поступает никаких изменений, и удерживать соединение неограниченное время», в то время как timeout означает «Держать это соединение открытым в течение x мс, и если за это время не поступит никаких изменений, закрыть сокет». heartbeat имеет приоритет над timeout.

#### 4.4.21.3.3. Непрерывность

Постоянный опрос сервера БД «Енисей» не идеален установка новых HTTP-соединений только для того, чтобы сообщить клиенту, что ничего не произошло, создает ненужную нагрузку на БД «Енисей».

Continuous остается открытым и подключенным к базе данных, пока явно не будет закрыт, а изменения отправляются клиенту по мере их возникновения, т.е. практически в реальном времени.

Как и в случае с типом longpoll, Можно установить тайм-аут и интервалы, чтобы гарантировать, что соединение будет оставаться открытым для новых изменений и обновлений.

Имейте в виду, что heartbeat означает «Посылать строчную подачу каждые x мс, если не поступает никаких изменений, и удерживать соединение неограниченное время», а timeout означает «Держать это соединение открытым в течение x мс, и если за это время не поступит никаких изменений, закрыть сокет». heartbeat имеет приоритет над timeout.

Ответ continuous немного отличается от других типов, чтобы упростить работу клиента каждая строка ответа либо пуста, либо представляет собой объект JSON, представляющий одно изменение, как в результатах обычной ленты.

##### 4.4.21.3.3.1. Формат запроса

Если был указан лимит, лента завершится объектом { last_seq }. Пример запроса отображает Рисунок 225.

```
GET /somedatabase/_changes?feed=continuous HTTP/1.1
```

**Запрос**
**Рисунок 225**

##### 4.4.21.3.3.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 226. Коды ошибок приведены в пункте 6.1.2.

```
{"seq":"1-
g1AAAAF9eJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MSGXAqSVIAkkn2IFUZzIkMuUAee5pRqnGiuXkKA2dp
XkpqWmZeagpu_Q4g_fGEbEkAqaqH2sIItsXAyMjM2NgUUwdOU_JYgCRDA5ACGjQfn30QlQsgKvcTVnkAovI-
YZUPICpBvs0CAN1eY_c","id":"fresh","changes":[{"rev":"5-
g1AAAAHxeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D666H6GcH6DYyMzIyNTUnwRR4LkGRoAFJAg-
YjwiMtOdXCwJyU8ICYtABi0n6EnwzSzIxS00yI8hPEjAMQM-
5nJTIQUPkAovI_UGUWAA0SgOI","id":"updated","changes":[{"rev":"2-7051cbe5c8faecd085a3fa619e6e6337"}]}
{"seq":"3-
g1AAAAHReJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D660H6Exlwqspj AZIMDUAKqHA-
yCZGiEuTUy0MzEnxL8SkBRCT9iPcbJBmZpSaZkKUmyFmHICYcZ-
wux9AVIJ8mAUABgp6XQ","id":"deleted","changes":[{"rev":"2-
eec205a9d413992850a6e32678485900"}],"deleted":true}
... tum tee tum ...
{"seq":"6-
g1AAAAIreJyVkEsKwjAURWMrqCOXoCuQ9MU0OrI70XyppcaRY92J7kR3ojupaVNopRQsgRd4yTlwb44QmqahQnN7VjpKImAr7E6U
u4eAI7EoiiJLQx6c3GIiuVJcx93vvQqxdFPsaguqLAY04YwpNLtYpc3RatXPJyW__-EFllst4D_-UPLXmh9VPAaICaEDUtixm-
jmLie6N30YqTeYDenDmx7e9GwyYRODNuu_MnnHyzverV6AMkPkAMfHO1rdUAKUkqhLZV-
_0o5j","id":"updated","changes":[{"rev":"3-825cb35de44c433bfb2df415563a19de"}]}
```


**Ответ**
**Рисунок 226**

Очевидно, что … tum tee tum … не появляется в фактическом ответе, но представляет собой длинную паузу перед тем, как произошло изменение с seq 6.

#### 4.4.21.3.4. Источник события

eventsource предоставляет push-уведомления, которые могут быть использованы в виде событий DOM в браузере. Для получения более подробной информации обратитесь к спецификации W3C eventsource. БД «Енисей» также учитывает параметр Last-Event-ID.

##### 4.4.21.3.4.1. Формат запроса

Пример запроса отображает Рисунок 227.

```
GET /somedatabase/_changes?feed=eventsource HTTP/1.1
```

**Запрос**
**Рисунок 227**

Если установить интервал (используя аргумент запроса heartbeat), БД «Енисей» будет отправлять события heartbeat, которые Можно отслеживать:

Пример запроса отображает Рисунок 228.

```
source.addEventListener('heartbeat', function () {}, false);
```

**Запрос**
**Рисунок 228**

Это может отслеживаться клиентским приложением для перезапуска соединения EventSource при необходимости (например, если TCP-соединение застряло в полуоткрытом состоянии).

На соединения EventSource распространяются ограничения на кросс-оригинальный обмен ресурсами. Возможно, потребуется настроить поддержку CORS, чтобы EventSource работал в приложении.

#### 4.4.21.3.4.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 229. Коды ошибок приведены в пункте 6.1.2.

```
// define the event handling function
if (window.EventSource) {
    var source = new EventSource(«/somedatabase/_changes?feed=eventsource»);
    source.onerror = function(e) {
        alert('EventSource failed.');
    };
    var results = [];
    var sourceListener = function(e) {
        var data = JSON.parse(e.data);
        results.push(data);
    };
    // start listening for events
    source.addEventListener('message', sourceListener, false);
    // stop listening for events
    source.removeEventListener('message', sourceListener, false);
}
```

**Ответ**
**Рисунок 229**

### 4.4.21.4. Фильтрация

Можно фильтровать содержимое ленты изменений несколькими способами. Самый простой способ указать в запросе один или несколько идентификаторов документов. В результате возвращаемое значение структуры будет содержать только изменения для указанных идентификаторов. Следует обратить внимание, что значение этого аргумента запроса должно быть массивом в формате JSON.

Можно также отфильтровать поток _changes, определив функцию фильтра в проектном документе. Спецификация фильтра такая же, как и для фильтров репликации. Указывается имя функции фильтра в параметре filter, указывая имя проектного документа и имя фильтра.

#### 4.4.21.4.1. Формат запроса

Пример запроса отображает Рисунок 230.

```
GET /db/_changes?filter=design_doc/filtername HTTP/1.1
```

**Запрос**
**Рисунок 230**

#### 4.4.21.4.2. _doc_ids

Этот фильтр принимает только изменения для документов, ID которых указан в параметре запроса doc_ids или в массиве объектов полезной нагрузки. Пример можно посмотреть в POST /{db}/_changes.

#### 4.4.21.4.3. _selector

Этот фильтр принимает только изменения для документов, которые соответствуют заданному селектору, определенному с помощью того же синтаксиса селектора, который используется для _find.

Это значительно эффективнее, чем использование функции фильтра JavaScript, и является рекомендуемым вариантом при фильтрации только по атрибутам документа.

Следует обратить внимание, что, в отличие от фильтров JavaScript, селекторы не имеют доступа к объекту запроса

##### 4.4.21.4.3.1. Формат запроса

Пример запроса отображает Рисунок 231.

```
POST /recipes/_changes?filter=_selector HTTP/1.1
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "selector": { "_id": { "$regex": "^_design/" } }
}
```

**Запрос**
**Рисунок 231**

##### 4.4.21.4.3.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 232. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Tue, 06 Sep 2016 20:03:23 GMT
Etag: «1H8RGBCK3ABY6ACDM7ZSC30QK»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "last_seq": "11-g1AAAAIreJyVkEEKwjAQRUOrqCuPoCeQZGIaXdmbaNIk1FLjyrXeRG-
iN9Gb1LQRaimFlsAEJnkP_s8RQtM0VGhuz0qTmABfYXdI7h4CgeSiKIosDUVwcotJIpQSOmp_71TIpZty97OgymJAU8G5QrOLVdo
crVbdfFzy-
wYvcbLVEvrxh5K_NlJggIhSNiCFHbmJbu5yonttMoneYD6kD296eNOzzoRNBNqse2Xyjpd3vP96AcYNTQY4Pt5RdTOuHIwCY5S0q
ewLwY6OaA",
    "pending": 0,
    "results": [
        {
            "changes": [
                {
                    "rev": "10-304cae84fd862832ea9814f02920d4b2"
                }
            ],
            "id": "_design/ingredients",
            "seq": "8-
g1AAAAHxeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D666H6GcH6DYyMzIyNTUnwRR4LkGRoAFJAg-
ZnJTIQULkAonI_ws0GaWZGqWkmRLkZYsYBiBn3Cdv2AKIS7ENWsG2mxkampsaGmLqyAOYpgEo"
        },
        {
            "changes": [
                {
                    "rev": "123-6f7c1b7c97a9e4f0d22bdf130e8fd817"
                }
            ],
            "deleted": true,
            "id": "_design/cookbook",
            "seq": "9-
g1AAAAHxeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D661F8YWBkZGZsbEqCL_JYgCRDA5ACGjQ_K5GBgMoFEJX7EW42
SDMzSk0zIcrNEDMOQMy4T9i2BxCVYB-ygm0zNTYyNTU2xNSVBQDnK4BL"
        },
        {
            "changes": [
                {
                    "rev": "6-5b8a52c22580e922e792047cff3618f3"
                }
            ],
            "deleted": true,
            "id": "_design/meta",
            "seq": "11-g1AAAAIReJyVkE0OgjAQRiegUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloQhO7mGSCKWEjtrtnQq5d
FXufhaoLIZoKjhXMLtYpc3RatXNxyW_b_ASJVstST_-
UPLXRgpESEQpG5DCjlyFm7uc6F6bTKI3iA_Zhzc9vOlZZ0ImItqse2Xyjpd3vDMBfzo_vrPawLiaxihhjOI2lX0BirqHbg"
        }
    ]
}
```

**Запрос**
**Рисунок 232**

#### 4.4.21.4.3.3. Отсутствующий селектор

Если объект селектора отсутствует в теле запроса, сообщение об ошибке будет похоже на следующий пример, который отображает Рисунок 233.

```
{
   "error": "bad request",
   "reason": "Selector must be specified in POST payload"
}
```

**Сообщение об ошибке**
**Рисунок 233**

#### 4.4.21.4.3.4. Недопустимый объект JSON

Если объект селектора не является правильно сформированным объектом JSON, сообщение об ошибке будет похоже на следующий пример, который отображает Рисунок 234.

```
{
   "error": "bad request",
   "reason": "Selector error: expected a JSON object"
}
```

**Сообщение об ошибке**
**Рисунок 234**

#### 4.4.21.4.3.5. Недопустимый селектор

Если объект selector не содержит действительного выражения выбора, сообщение об ошибке будет похоже на следующий пример, который отображает Рисунок 235.

```
{
   "error": "bad request",
   "reason": "Selector error: expected a JSON object"
}
```

**Сообщение об ошибке**
**Рисунок 235**

### 4.4.21.4.4. _design

Фильтр _design принимает только изменения для любого проектного документа в запрашиваемой базе данных.

#### 4.4.21.4.4.1. Формат запроса

Пример запроса отображает Рисунок 236.

```
GET /recipes/_changes?filter=_design HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 236**

#### 4.4.21.4.4.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 237. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Tue, 06 Sep 2016 12:55:12 GMT
ETag: «ARIHFWL3I7PIS0SPVTFU6TLR2»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "last_seq": "11-g1AAAAIreJyVkEEKwjAQRUOrqCuPoCeQZGIaXdmbaNIk1FLjyrXeRG-
iN9Gb1LQRaimFlsAEJnkP_s8RQtM0VGhuz0qTmABfYXdI7h4CgeSiKIosDUVwcotJIpQSOmp_71TIpZty97OgymJAU8G5QrOLVdo
crVbdfFzy-
wYvcbLVEvrxh5K_NlJggIhSNiCFHbmJbu5yonttMoneYD6kD296eNOzzoRNBNqse2Xyjpd3vP96AcYNTQY4Pt5RdTOuHIwCY5S0q
ewLwY6OaA",
    "pending": 0,
    "results": [
        {
            "changes": [
                {
                    "rev": "10-304cae84fd862832ea9814f02920d4b2"
                }
            ],
            "id": "_design/ingredients",
            "seq": "8-
g1AAAAHxeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D666H6GcH6DYyMzIyNTUnwRR4LkGRoAFJAg-
ZnJTIQULkAonI_ws0GaWZGqWkmRLkZYsYBiBn3Cdv2AKIS7ENWsG2mxkampsaGmLqyAOYpgEo"
        },
        {
            "changes": [
                {
                    "rev": "123-6f7c1b7c97a9e4f0d22bdf130e8fd817"
                }
            ],
            "deleted": true,
            "id": "_design/cookbook",
            "seq": "9-
g1AAAAHxeJzLYWBg4MhgTmHgz8tPSTV0MDQy1zMAQsMcoARTIkOS_P___7MymBOZcoEC7MmJKSmJqWaYynEakaQAJJPsoaYwgE1J
M0o1TjQ3T2HgLM1LSU3LzEtNwa3fAaQ_HkV_kkGyZWqSEXH6E0D661F8YWBkZGZsbEqCL_JYgCRDA5ACGjQ_K5GBgMoFEJX7EW42
SDMzSk0zIcrNEDMOQMy4T9i2BxCVYB-ygm0zNTYyNTU2xNSVBQDnK4BL"
        },
        {
            "changes": [
                {
                    "rev": "6-5b8a52c22580e922e792047cff3618f3"
                }
            ],
            "deleted": true,
            "id": "_design/meta",
            "seq": "11-g1AAAAIReJyVkE0OgjAQRiegUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloQhO7mGSCKWEjtrtnQq5d
FXufhaoLIZoKjhXMLtYpc3RatXNxyW_b_ASJVstST_-
UPLXRgpESEQpG5DCjlyFm7uc6F6bTKI3iA_Zhzc9vOlZZ0ImItqse2Xyjpd3vDMBfzo_vrPawLiaxihhjOI2lX0BirqHbg"
        }
    ]
}
```

**Ответ**

**Рисунок 237**

#### 4.4.21.4.5. _view

Специальный фильтр _view позволяет использовать существующую функцию map в качестве фильтра. Если функция map выдает что-либо для обрабатываемого документа, это считается принятым, и событие изменений передается в ленту. Для большинства практических случаев использования функции фильтра очень похожи на функции map, поэтому эта возможность позволяет сократить количество дублирующегося кода.

Хотя функции map не обрабатывают проектную документацию, использование фильтра _view заставляет их это делать. Необходима уверенность, что они готовы без проблем обрабатывать документы с чужеродной структурой.

Использование фильтра _view filter не запрашивает индексные файлы представления, поэтому невозможно использовать обычные параметры запроса представления для дополнительной фильтрации потока изменений по индексному ключу. Кроме того, БД «Енисей» не возвращает результат мгновенно, как это делается для представлений в качестве фильтра действительно используется указанная функция map.

Более того, невозможно сделать такие фильтры динамическими, например, обрабатывать параметры запроса или работать с Контекстный объект пользователя функция map работает только с документом.

##### 4.4.21.4.5.1. Формат запроса

Пример запроса отображает Рисунок 238.

```
GET /recipes/_changes?filter=_view&view=ingredients/by_recipe HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 238**

##### 4.4.21.4.5.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 239. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Tue, 06 Sep 2016 12:57:56 GMT
ETag: «ARIHFWL3I7PIS0SPVTFU6TLR2»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "last_seq": "11-g1AAAAIreJyVkEEKwjAQRUOrqCuPoCeQZGIaXdmbaNIk1FLjyrXeRG-
iN9Gb1LQRaimFlsAEJnkP_s8RQtM0VGhuz0qTmABfYXdI7h4CgeSiKIosDUVwcotJIpQSOmp_71TIpZty97OgymJAU8G5QrOLVdo
crVbdfFzy-
wYvcbLVEvrxh5K_NlJggIhSNiCFHbmJbu5yonttMoneYD6kD296eNOzzoRNBNqse2Xyjpd3vP96AcYNTQY4Pt5RdTOuHIwCY5S0q
ewLwY6OaA",
    "results": [
        {
            "changes": [
                {
                    "rev": "13-bcb9d6388b60fd1e960d9ec4e8e3f29e"
                }
            ],
            "id": "SpaghettiWithMeatballs",
            "seq": "11-g1AAAAIReJyVkE0OgjAQRiegUVceQU9g-
mOpruQm2tI2SLCuXOtN9CZ6E70JFmpCCCFCmkyTdt6bfJMDwDQNFcztWWkcY8JXyB2cu49AgFwURZGloQhO7mGSCKWEjtrtnQq5d
FXufhaoLIZoKjhXMLtYpc3RatXNxyW_b_ASJVstST_-
UPLXRgpESEQpG5DCjlyFm7uc6F6bTKI3iA_Zhzc9vOlZZ0ImItqse2Xyjpd3vDMBfzo_vrPawLiaxihhjOI2lX0BirqHbg"
        }
    ]
}
```

**Ответ**
**Рисунок 239**

### 4.4.22 Запросить сжатие указанной базы данных

Запрос на сжатие указанной базы данных. Сжимает дисковый файл базы данных, выполняя следующие операции:

Записывает новую, оптимизированную, версию файла базы данных, удаляя неиспользуемые секции из новой версии во время записи. Поскольку для этой цели временно создается новый файл, для завершения процедуры сжатия может потребоваться вдвое больше места в текущем хранилище указанной базы данных.

Удаляет из базы данные полей всех нелистовых ревизий документов.

Удаляет старую историю ревизий сверх лимита, указанного параметром базы данных _revs_limit.

Сжатие может быть запрошено только для отдельной базы данных; невозможно уплотнить все базы данных для экземпляра БД «Енисей». Процесс сжатия выполняется как фоновый процесс.

Можно определить, работает ли процесс сжатия над базой данных, получив метаинформацию базы данных; значение compact_running в возвращаемой структуре базы данных будет установлено в true.

Можно также получить список запущенных процессов, чтобы определить, запущен ли процесс сжатия в данный момент.

#### 4.4.22.1. Формат запроса

Параметры запроса отображает Таблица 343.

**Таблица 343 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_compact |
| Метод | POST |

Параметры строки запроса отображает Таблица 344.

**Таблица 344 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 345.

**Таблица 345 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 240.

```
POST /db/_compact HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 240**

#### 4.4.22.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 346.

**Таблица 346 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 347.

**Таблица 347 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 348.

**Таблица 348 – Коды состояния**

| Код | Описание |
|---|---|
| 202 Accepted | Запрос на сжатие был принят |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 415 Unsupported Media Type | Плохое значение Content-Type |

Ответ в случае успешного завершения запроса отображает Рисунок 241. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 202 Accepted
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 09:27:43 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**

**Рисунок 241**

### 4.4.23 Запросить сжатие индексов представления

Сжимает индексы представления, связанные с указанным проектным документом. Может оказаться, что сжатие большого представления может вернуть больше памяти, чем сжатие фактической базы данных. Таким образом, можно использовать этот параметр вместо сжатия всей базы данных, если известно, что определенный набор индексов представления был затронут недавним изменением базы данных.

#### 4.4.23.1. Формат запроса

Параметры запроса отображает Таблица 349.

**Таблица 349 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_compact/{ddoc} |
| Метод | POST |

Параметры строки запроса отображает Таблица 350.

**Таблица 350 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |

Параметры HTTP заголовка запроса отображает Таблица 351.

**Таблица 351 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 242.

```
POST /db/_compact/posts HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 242**

#### 4.4.23.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 352.

**Таблица 352 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 353.

**Таблица 353 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 354.

**Таблица 354 – Коды состояния**

| Код | Описание |
|---|---|
| 202 Accepted | Запрос на сжатие был принят |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 404 Not Found | Документ проектирования не найден |
| 415 Unsupported Media Type | Плохое значение Content-Type |

Ответ в случае успешного завершения запроса отображает Рисунок 243. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 202 Accepted
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 09:36:44 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 243**

Индексы представлений хранятся в отдельном файле .couch, основанном на хэше соответствующих функций документа проектирования, в подкаталоге, где расположены основные файлы базы данных .couch.

### 4.4.24 Удалить индексные файлы представлений¶

Удаляет индексные файлы представлений, которые больше не требуются БД «Енисей» в результате изменения представлений в проектных документах. Поскольку имя файла представления основано на хэше функций представления, со временем старые представления будут оставаться, потребляя хранилище. Этот вызов очищает кэшированный вывод представления на диске для данного представления.

#### 4.4.24.1. Формат запроса

Параметры запроса отображает Таблица 355.

**Таблица 355 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_view_cleanup |
| Метод | POST |

Параметры строки запроса отображает Таблица 356.

**Таблица 356 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 357.

**Таблица 357 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json, | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 244.

```
POST /db/_view_cleanup HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 244**

#### 4.4.24.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 358.

**Таблица 358 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 359.

**Таблица 359 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 360.

**Таблица 360 – Коды состояния**

| Код | Описание |
|---|---|
| 202 Accepted | Запрос на сжатие был принят |
| 400 Bad Request | Неверное имя базы данных |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |
| 415 Unsupported Media Type | Плохое значение Content-Type |

Ответ в случае успешного завершения запроса отображает Рисунок 245. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 202 Accepted
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 09:27:43 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true

}
```

**Ответ**
**Рисунок 245**

### 4.4.25 Получить текущий объект безопасности из указанной базы данных

Возвращает текущий объект безопасности из указанной базы данных.

Объект безопасности состоит из двух обязательных элементов, admins и members, которые используются для указания списка пользователей и/или ролей, имеющих права администратора и участника БД соответственно:

1) members: могут читать все типы документов из БД, а также записывать (и редактировать) документы в БД, за исключением конструкторских документов.
2) admins: имеют все привилегии участников (members) плюс привилегии: писать (и редактировать) проектные документы, добавлять/удалять администраторов и участников БД и устанавливать лимит ревизий БД. Они не могут ни создавать базу данных, ни удалять базу данных.

Объекты members и admins содержат два поля типа массив:

1) names: Список имен пользователей БД «Енисей»
2) roles: Список ролей пользователей

Любые дополнительные поля в объекте безопасности являются необязательными. Весь объект security становится доступным для проверки и других внутренних функций, чтобы база данных могла контролировать и ограничивать функциональность.

Если поля names и roles свойств admins или members являются пустыми массивами или не существуют, это означает, что у базы данных нет администраторов или участников.

При отсутствии members только admins сервера (с зарезервированной ролью _admin) могут обновлять проектные документы и вносить другие изменения на уровне admins.

При отсутствии участников или ролей любой пользователь может писать обычные документы (любые документы, не относящиеся к дизайну) и читать документы из базы данных.

Вновь создаваемые базы данных по умолчанию имеют роль _admin для предотвращения непреднамеренного доступа.

Если для базы данных определены имена участников или роли, то только аутентифицированным пользователям, имеющим соответствующее имя или роль, разрешается читать документы из базы данных (или выполнять вызов GET /{db}).

Если объект безопасности для базы данных никогда не был установлен, то возвращаемое значение будет пустым.

Также следует обратить внимание, что объекты безопасности не являются обычными версионными документами (то есть, на них не распространяются правила MVCC). Это выбор подхода для ускорения проверки авторизации (позволяет избежать обхода B-дерева документов базы данных).

#### 4.4.25.1. Формат запроса

Параметры запроса отображает Таблица 361.

**Таблица 361 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_security |
| Метод | GET |

Параметры строки запроса отображает Таблица 362.

**Таблица 362 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 363.

**Таблица 363 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 246.

```
GET /db/_security HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 246**

#### 4.4.25.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 364.

**Таблица 364 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 365.

**Таблица 365 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| admins | Объект с двумя полями имена и роли. | Объект (Json Object) | Да |
| members | Объект с двумя полями в виде имен и ролей. | Объект (Json Object) | Да |

Коды состояния отображает Таблица 366.

**Таблица 366 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 247. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 109
Content-Type: application/json
Date: Mon, 12 Aug 2013 19:05:29 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "admins": {
        "names": [
            "superuser"
        ],
        "roles": [
            "admins"
        ]
    },
    "members": {
        "names": [
            "user1",
            "user2"
        ],
        "roles": [
            "developers"
        ]
    }
}
```

**Ответ**
**Рисунок 247**

### 4.4.26 Установить объект безопасности для данной базы данных

#### 4.4.26.1. Формат запроса

Параметры запроса отображает Таблица 367.

**Таблица 367 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_security |
| Метод | PUT |

Параметры строки запроса отображает Таблица 368.

**Таблица 368 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 369.

**Таблица 369 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 370.

**Таблица 370 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| admins | Объект с двумя полями имена и роли. | Объект (Json Object) | Да |
| members | Объект с двумя полями в виде имен и ролей. | Объект (Json Object) | Да |

Пример запроса отображает Рисунок 248.

```
shell> curl http://localhost:5984/pineapple/_security -X PUT -H ‘content-type: application/json’ -
H ‘accept: application/json’ -
d ‘{«admins»:{«names»:[«superuser»],»roles»:[«admins»]},»members»:{«names»: [«user1»,»user2»],»roles
»: [«developers»]}}’
PUT /db/_security HTTP/1.1
Accept: application/json
Content-Length: 121
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "admins": {
        "names": [
            "superuser"
        ],
        "roles": [
            "admins"
        ]
    },
    "members": {
        "names": [
            "user1",
            "user2"
        ],
        "roles": [
            "developers"
        ]
    }
}
```

**Запрос**
**Рисунок 248**

#### 4.4.26.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 371.

**Таблица 371 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 372.

**Таблица 372 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 373.

**Таблица 373 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 401 Unauthorized | Требуются привилегии администратора сервера БД «Енисей» |

Ответ в случае успешного завершения запроса отображает Рисунок 249. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Tue, 13 Aug 2013 11:26:28 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 249**

### 4.4.27 Удалить ссылки на документы в базе данных

Очистка базы данных навсегда удаляет ссылки на документы в базе данных. Обычное удаление документа в БД «Енисей» не удаляет документ из базы данных, вместо этого документ помечается как _deleted=true (и создается новая ревизия). Это делается для того, чтобы удаленные документы могли быть реплицированы в другие базы данных как удаленные. Это также означает, что Можно проверить статус документа и определить, что документ был удален, по его отсутствию.

Запрос на очистку должен включать идентификаторы документов, а для каждого идентификатора документа одну или несколько ревизий, которые должны быть очищены. Документы могут быть ранее удалены, но это не обязательно. Ревизии должны быть ревизиями листа.

Ответ будет содержать список идентификаторов документов и ревизий, которые были успешно очищены.

#### 4.4.27.1. Формат запроса

Параметры запроса отображает Таблица 374.

**Таблица 374 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_purge |
| Метод | POST |

Параметры строки запроса отображает Таблица 375.

**Таблица 375 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 376.

**Таблица 376 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 377.

**Таблица 377 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| object | Сопоставление идентификатора документа со списком ревизий для очистки | Объект (Json Object) | Да |

Пример запроса отображает Рисунок 250.

```
POST /db/_purge HTTP/1.1
Accept: application/json
Content-Length: 76
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "c6114c65e295552ab1019e2b046b10e": [
        "3-b06fcd1c1c9e0ec7c480ee8aa467bf3b",
        "3-c50a32451890a3f1c3e423334cc92745"
    ]
}
```

**Запрос**

**Рисунок 250**

#### 4.4.27.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 378.

**Таблица 378 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 379.

**Таблица 379 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| purge_seq | Строка последовательности очистки | Строка (String) | Да |
| purged | Сопоставление идентификатора документа со списком очищенных ревизий | Объект (Json Object) | Да |

Коды состояния отображает Таблица 380.

**Таблица 380 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Запрос успешно завершен |
| 202 Accepted | Запрос был принят и успешно завершен хотя бы на одной реплике, но кворум не был достигнут |
| 400 Bad Request | Неверное имя базы данных или полезная нагрузка JSON |
| 415 Unsupported Media Type | Плохое значение Content-Type |
| 500 Internal Server Error | Внутренняя ошибка сервера или тайм-аут |

Ответ в случае успешного завершения запроса отображает Рисунок 251. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 107
Content-Type: application/json
Date: Fri, 02 Jun 2017 18:55:54 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "purge_seq": null,
  "purged": {
    "c6114c65e295552ab1019e2b046b10e": [
        "3-c50a32451890a3f1c3e423334cc92745"
      ]
  }
}
```

**Ответ**
**Рисунок 251**

Дерево редактирования документа 1 отображает Рисунок 252.

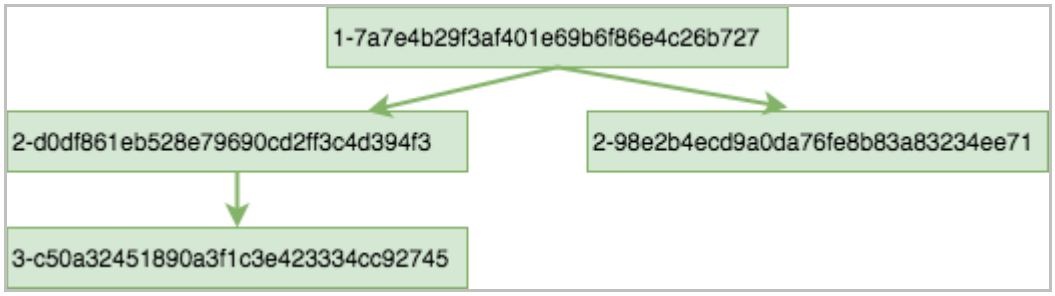

**Дерево редактирования документа 1**
**Рисунок 252**

Например, учитывая приведенное выше дерево очистки и выдачу вышеуказанного запроса на очистку, весь документ будет очищен, так как он содержит только одну ветвь с ревизией листа 3-c50a32451890a3f1c3e423334cc92745, которая будет очищена. В результате этой операции очистки документ с _id:c6114c65e295552ab1019e2b046b10e будет полностью удален из b+дерева документов и b+дерева последовательностей базы данных. Он не будет доступен через конечные точки _all_docs или _changes, как будто этого документа никогда не существовало. Также в результате операции очистки будут увеличены значения purge_seq и update_seq базы данных.

Следует обратить внимание, как ревизия 3-b06fcd1c1c9e0ec7c480ee8aa467bf3b была проигнорирована. Ревизии, которые уже были очищены, и нелистовые ревизии игнорируются в запросе на очистку.

Если документ имеет две конфликтующие ревизии со следующей историей ревизий:

Дерево редактирования документа 2 отображает Рисунок 253.




**Дерево редактирования документа 2**
**Рисунок 253**

Вышеприведенный запрос на очистку очистит только одну ветвь, оставив дерево ревизий документа только с одной ветвью:

Дерево редактирования документа 3 отображает Рисунок 254.


**Дерево редактирования документа 3**
**Рисунок 254**

В результате этой операции очистки новая обновленная версия документа будет доступна в _all_docs и _changes, создавая новую запись в _changes. Периодичность очистки базы данных (purge_seq) и обновления базы данных (update_seq) увеличится.

#### 4.4.27.3. Внутренняя репликация

Очистка автоматически реплицируется между копиями одной и той же базы данных. Каждая база данных имеет внутреннее дерево чисток, в котором хранится определенное количество последних чисток. Это позволяет осуществлять внутреннюю синхронизацию между репликами одной и той же базы данных.

#### 4.4.27.4. Внешняя репликация

Операции очистки не реплицируются на другие внешние базы данных. Внешняя репликация работает путем определения ревизий документа источника, которые отсутствуют в целевой базе, и копирования этих ревизий из источника в целевую базу. Операция очистки полностью удаляет ревизии из дерева очистки документа, делая внешнюю репликацию ревизий невозможной.

Если нужно, чтобы очистка была эффективной для нескольких эффективных баз данных, необходимо запустить очистку отдельно для каждой из баз данных.

#### 4.4.27.5. Обновление индексов

Количество чисток в базе данных отслеживается с помощью последовательности чисток. Она используется индексатором представлений для оптимизации обновления представлений, содержащих очищенные документы.

Каждый внутренний индексатор базы данных, включая индексатор представлений, хранит свою собственную последовательность очистки. Последовательность очистки, хранящаяся в индексе, может быть намного меньше, чем последовательность очистки базы данных, вплоть до количества запросов на очистку, разрешенных к хранению в деревьях очистки базы данных. Несколько запросов на очистку могут быть обработаны индексатором без перестройки индекса. Индекс будет обновляться в соответствии с этими запросами на очистку.

Индекс документов основан на победителе дерева ревизий. В зависимости от того, какая ревизия указана в запросе на очистку, обновление индекса происходит следующим образом:

1) Если в запросе на очистку победитель дерева ревизий не указан, то индексная запись этого документа не изменяется.
2) Если победитель дерева ревизий указан в запросе на очистку, и после очистки осталась одна ревизия, индексная запись документа будет построена в соответствии с новым победителем дерева ревизий.
3) Если в запросе на очистку указаны все ревизии документа, индексная запись документа будет удалена. Документ больше не будет найден в поиске.

### 4.4.28 Получить текущие настройки

Получает текущую настройку purged_infos_limit (лимит очищенных документов), максимальное количество исторических очисток (ID очищенных документов с их ревизиями), которые могут храниться в базе данных.

#### 4.4.28.1. Формат запроса

Параметры запроса отображает Таблица 381.

**Таблица 381 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_purged_infos_limit |
| Метод | GET |

Параметры строки запроса отображает Таблица 382.

**Таблица 382 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 383.

**Таблица 383 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 255.

```
GET /db/_purged_infos_limit HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 255**

#### 4.4.28.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 384.

**Таблица 384 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 385.

**Таблица 385 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 386.

**Таблица 386 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 256. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 5
Content-Type: application/json
Date: Wed, 14 Jun 2017 14:43:42 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
1000
```

**Ответ**
**Рисунок 256**

### 4.4.29 Установить максимальное количество очищенных документов, которые будут отслеживаться в базе данных даже после сжатия

Устанавливает максимальное количество очищенных документов (запрошенные очищенные идентификаторы с их ревизиями), которые будут отслеживаться в базе данных даже после сжатия. Можно установить лимит очищенных документов в базе данных, используя в качестве тела запроса скалярное целое число лимита, который требуется установить.

Значение по умолчанию для исторических хранимых чисток составляет 1000. Это означает, что до 1000 очисток может быть синхронизировано между репликами одной и той же базы данных в случае, если одна из реплик не работала, когда происходила очистка.

Этот запрос устанавливает мягкий предел для хранимых чисток. Во время сжатия БД «Енисей» попытается сохранить в базе данных только _purged_infos_limit чисток, но иногда количество сохраненных чисток может превысить это значение. Если база данных не завершила синхронизацию чисток с активными индексами или активными внутренними репликациями, она может временно хранить большее количество исторических чисток.

#### 4.4.29.1. Формат запроса

Параметры запроса отображает Таблица 387.

**Таблица 387 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_purged_infos_limit |
| Метод | PUT |

Параметры строки запроса отображает Таблица 388.

**Таблица 388 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 389.

**Таблица 389 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 257.

```
PUT /db/_purged_infos_limit HTTP/1.1
Accept: application/json
Content-Length: 4
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
1500
```

**Запрос**
**Рисунок 257**

### 4.4.29.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 390.

**Таблица 390 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 391.

**Таблица 391 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ok | Статус операции | Логический тип (Boolean) | Да |

Коды состояния отображает Таблица 392.

**Таблица 392 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверные данные JSON |

Ответ в случае успешного завершения запроса отображает Рисунок 258. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Wed, 14 Jun 2017 14:45:34 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 258**

### 4.4.30 Получить ревизии документа, несуществующие в базе данных

При заданном списке ревизий документа возвращает ревизии документа, которые не существуют в базе данных.

#### 4.4.30.1. Формат запроса

Параметры запроса отображает Таблица 393.

**Таблица 393 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_missing_revs |
| Метод | POST |

Параметры строки запроса отображает Таблица 394.

**Таблица 394 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 395.

**Таблица 395 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 396.

**Таблица 396 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| object | Отображение идентификатора документа на список ревизий для поиска | Объект (Json Object) | Да |

Пример запроса отображает Рисунок 259.

```
POST /db/_missing_revs HTTP/1.1
Accept: application/json
Content-Length: 76
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "c6114c65e295552ab1019e2b046b10e": [
        "3-b06fcd1c1c9e0ec7c480ee8aa467bf3b",
        "3-0e871ef78849b0c206091f1a7af6ec41"
    ]
}
```

**Запрос**
**Рисунок 259**

#### 4.4.30.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 397.

**Таблица 397 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 398.

**Таблица 398 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| missing_revs | Сопоставление идентификатора документа со списком пропущенных ревизий | Объект (Json Object) | Да |

Коды состояния отображает Таблица 399.

**Таблица 399 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверное имя базы данных или полезная нагрузка JSON |

Ответ в случае успешного завершения запроса отображает Рисунок 260. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 64
Content-Type: application/json
Date: Mon, 12 Aug 2013 10:53:24 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "missing_revs":{
        "c6114c65e295552ab1019e2b046b10e": [
            "3-b06fcd1c1c9e0ec7c480ee8aa467bf3b"
        ]
    }
}
```

**Ответ**

**Рисунок 260**

### 4.4.31 Получить подмножество идентификаторов документов/ревизий, не соответствующих ревизиям базы данных

Учитывая набор идентификаторов документа/ревизии, возвращает подмножество тех из них, которые не соответствуют ревизиям, хранящимся в базе данных.

В основном используется репликатором в качестве важной оптимизации: после получения набора новых идентификаторов ревизий из исходной базы данных репликатор отправляет этот набор в _revs_diff базы назначения, чтобы узнать, какие из них уже существуют там. После этого он может избежать получения и отправки уже известных тел документов.

Как содержание запроса, так и содержание ответа представляют собой объекты JSON, ключами которых являются идентификаторы документов, но значения структурированы по-разному:

1) В запросе значение это массив идентификаторов ревизий для данного документа.
2) В ответе значение представляет собой объект с ключом missing:, значением которого является список идентификаторов ревизий для этого документа (те, которые не хранятся в базе данных) и, по желанию, ключ possible_ancestors, значением которого является массив известных идентификаторов ревизий, которые могут быть предками отсутствующих ревизий.

#### 4.4.31.1. Формат запроса

Параметры запроса отображает Таблица 400.

**Таблица 400 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_revs_diff |
| Метод | POST |

Параметры строки запроса отображает Таблица 401.

**Таблица 401 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 402.

**Таблица 402 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 403.

**Таблица 403 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| object | Отображение идентификатора документа на список ревизий для поиска | Объект (Json Object) | Да |

Пример запроса отображает Рисунок 261.

```
POST /db/_revs_diff HTTP/1.1
Accept: application/json
Content-Length: 113
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "190f721ca3411be7aa9477db5f948bbb": [
        "3-bb72a7682290f94a985f7afac8b27137",
        "4-10265e5a26d807a3cfa459cf1a82ef2e",
        "5-067a00dff5e02add41819138abb3284d"
    ]
}
```

**Запрос**
**Рисунок 261**

#### 4.4.31.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 404.

**Таблица 404 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 405.

**Таблица 405 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| missing | Список пропущенных ревизий для указанного документа | Массив (Array) | Да |
| possible_ancestors | Список ревизий, которые могут быть предками для указанного документа и его текущей ревизии в запрашиваемой базе данных | Массив (Array) | Да |

Коды состояния отображает Таблица 406.

**Таблица 406 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверное имя базы данных или полезная нагрузка JSON |

Ответ в случае успешного завершения запроса отображает Рисунок 262. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 88
Content-Type: application/json
Date: Mon, 12 Aug 2013 16:56:02 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "190f721ca3411be7aa9477db5f948bbb": {
        "missing": [
            "3-bb72a7682290f94a985f7afac8b27137",
            "5-067a00dff5e02add41819138abb3284d"
        ],
        "possible_ancestors": [
            "4-10265e5a26d807a3cfa459cf1a82ef2e"
        ]
    }
}
```

**Ответ**
**Рисунок 262**

### 4.4.32 Получить текущую настройку revs_limit

#### 4.4.32.1. Формат запроса

Параметры запроса отображает Таблица 407.

**Таблица 407 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_revs_limit |
| Метод | GET |

Параметры строки запроса отображает Таблица 408.

**Таблица 408 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 409.

**Таблица 409 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 263.

```
GET /db/_revs_limit HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 263**

#### 4.4.32.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 410.

**Таблица 410 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 411.

**Таблица 411 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 268. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 5
Content-Type: application/json
Date: Mon, 12 Aug 2013 17:27:30 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
1000
```

**Ответ**
**Рисунок 264**

### 4.4.33 Установить максимальное количество ревизий документа, которое будет отслеживаться БД «Енисей» после сжатия

Устанавливает максимальное количество ревизий документа, которое будет отслеживаться БД «Енисей» даже после сжатия. Можно установить лимит ревизий для базы данных, используя в качестве тела запроса скалярное целое число лимита, который требуется установить.

#### 4.4.33.1. Формат запроса

Параметры запроса отображает Таблица 412.

**Таблица 412 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_revs_limit |
| Метод | PUT |

Параметры строки запроса отображает Таблица 413.

**Таблица 413 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 414.

**Таблица 414 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 265.

```
PUT /db/_revs_limit HTTP/1.1
Accept: application/json
Content-Length: 5
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
1000
```

**Запрос**
**Рисунок 265**

#### 4.4.33.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 415.

**Таблица 415 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 416.

**Таблица 416 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверное имя базы данных или полезная нагрузка JSON |

Ответ в случае успешного завершения запроса отображает Рисунок 266. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 12
Content-Type: application/json
Date: Mon, 12 Aug 2013 17:47:52 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "ok": true
}
```

**Ответ**
**Рисунок 266**

## 4.5. Документы

Подробная информация о том, как создавать, читать, обновлять и удалять документы в базе данных.

### 4.5.1 Получить HTTP заголовки с информацией о документе

Возвращает HTTP заголовки, содержащие минимальное количество информации об указанном документе. Метод поддерживает те же аргументы запроса, что и метод GET /{db}/{docid}, но возвращается только информация заголовка (включая размер документа и ревизию в виде ETag).

Заголовок ETag показывает текущую ревизию запрашиваемого документа, а Content-Length указывает длину данных, если документ запрашивается полностью.

Если добавить любой из аргументов запроса, то результирующие HTTP-заголовки будут соответствовать тому, что будет возвращено.

#### 4.5.1.1. Формат запроса

Параметры запроса отображает Таблица 417.

**Таблица 417 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid} |
| Метод | HEAD |

Параметры строки запроса отображает Таблица 418.

**Таблица 418 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 419.

**Таблица 419 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| If-None-Match | Маркер ревизии документа в двойных кавычках | Строка (String) | |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 267.

```
HEAD /db/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 267**

#### 4.5.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 420.

**Таблица 420 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | Размер документа | Строка (String) | Да |
| ETag | Маркер ревизии документа, заключенный в двойные кавычки | Строка (String) | Да |

Коды состояния отображает Таблица 421.

**Таблица 421 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ существует |
| 304 Not Modified | Документ не был изменен с указанной ревизии |
| 401 Unauthorized | Требуется привилегия чтения |
| 404 Not Found | Документ не найден |

Ответ в случае успешного завершения запроса отображает Рисунок 268. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 660
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "12-151bb8678d45aaa949ec3698ef1c7e78"
Server: Yenisei/1.0.0 (Erlang OTP/24)
```

**Ответ**

**Рисунок 268**

### 4.5.2 Получить документ по указанному docid

Возвращает документ по указанному docid из указанной базы данных. Если не указана конкретная ревизия, будет возвращена последняя ревизия документа.

#### 4.5.2.1. Формат запроса

Параметры запроса отображает Таблица 422.

**Таблица 422 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid} |
| Метод | GET |

Параметры строки запроса отображает Таблица 423.

**Таблица 423 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 424.

**Таблица 424 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>multipart/related<br>multipart/mixed<br>text/plain | Строка (String) | Да |
| If-None-Match | Маркер ревизии документа в двойных кавычках | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 425.

**Таблица 425 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| attachments | Включать тела вложений в ответ. По умолчанию false | Логический тип (Boolean) | Нет |
| att_encoding_info | Включает информацию о кодировке в заглушки вложений, если конкретное вложение сжато. По умолчанию false. | Логический тип (Boolean) | Нет |
| atts_since | Включает вложения только с указанных ревизий. Не включает вложения для указанных ревизий. | Массив (Array) | Нет |
| conflicts | Включает информацию о конфликтах в документе. По умолчанию false | Логический тип (Boolean) | Нет |
| deleted_conflicts | Включает информацию об удаленных конфликтующих ревизиях. По умолчанию false | Логический тип (Boolean) | Нет |
| latest | Принуждает извлекать последнюю ревизию «листа», независимо от того, какая ревизия была запрошена. По умолчанию false | Логический тип (Boolean) | Нет |
| local_seq | Включает последовательность последних обновлений для документа. По умолчанию false meta (boolean) Действует так же, как указание параметров запроса all conflicts, deleted_conflicts и revs_info. По умолчанию false | Логический тип (Boolean) | Нет |
| open_revs | Извлекает документы указанных ревизий листа. Дополнительно принимает значение all, чтобы вернуть все ревизии листа. | Массив (Array) | Нет |
| rev | Извлекает документ указанной ревизии. | Строка (String) | Нет |
| revs | Включает список всех известных ревизий документа. По умолчанию false | Логический тип (Boolean) | Нет |
| revs_info | Включает подробную информацию обо всех известных ревизиях документа. По умолчанию false | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 269.

```
GET /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 269**

#### 4.5.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 426.

**Таблица 426 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Length | application/json<br>multipart/related<br>multipart/mixed<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Маркер ревизии документа, заключенный в двойную кавычку. Недоступен при извлечении информации, связанной с конфликтами | Строка (String) | Да |
| Transfer-Encoding | chunked. Доступно при запросе с параметром запроса open_revs | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 427.

**Таблица 427 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| _id | Идентификатор документа | Строка (String) | Да |
| _rev | Токен MVCC пересмотра | Строка (String) | Да |
| _deleted | Флаг удаления. Доступен, если документ был удален | Логический тип (Boolean) | Нет |
| _attachments | Заглушки вложений. Доступно, если документ имеет какие-либо вложения | Объект (Json Object) | Нет |
| _conflicts | Список конфликтующих ревизий. Доступен при запросе с параметром запроса conflicts=true | Массив (Array) | Нет |
| _deleted_conflicts | Список удаленных конфликтующих ревизий. Доступен при запросе с параметром запроса deleted_conflicts=true. | Массив (Array) | Нет |
| _local_seq | Последовательность обновления документа в текущей базе данных. Доступно при запросе с параметром запроса local_seq=true | Строка (String) | Нет |
| _revs_info | Список объектов с информацией о локальных ревизиях и их статусе. Доступен при запросе с параметром запроса open_revs | Массив (Array) | Нет |
| _revisions | Список объектов без маркеров локальных ревизий. Доступен при запросе с параметром запроса revs=true | Объект (Json Object) | Нет |

Коды состояния отображает Таблица 428.

**Таблица 428 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ существует |
| 304 Not Modified | Документ не был изменен с указанной ревизии |
| 400 Bad Request | Формат запроса или ревизии был неверным |
| 401 Unauthorized | Требуется привилегия чтения |
| 404 Not Found | Документ не найден |

Ответ в случае успешного завершения запроса отображает Рисунок 270. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 660
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "1-917fa2381192822767f010b95b45325b"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_id": "SpaghettiWithMeatballs",
    "_rev": "1-917fa2381192822767f010b95b45325b",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**
**Рисунок 270**

### 4.5.3 Создать новый именованный документ или новую редакцию

Метод PUT создает новый именованный документ или создает новую редакцию существующего документа. В отличие от POST /{db}, необходимо указать идентификатор документа в URL запроса.

При обновлении существующего документа текущая ревизия документа должна быть включена в документ (т.е. в содержание запроса), в параметр запроса rev или в заголовок запроса If-Match.

#### 4.5.3.1. Формат запроса

Параметры запроса отображает Таблица 429.

**Таблица 429 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid} |
| Метод | PUT |

Параметры строки запроса отображает Таблица 430.

**Таблица 430 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 431.

**Таблица 431 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json,<br>multipart/related | Строка (String) | Да |
| If-Match | Ревизия документа. Альтернатива параметру запроса rev или ключу документа | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 432.

**Таблица 432 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия документа, если обновляется существующий документ. Альтернатива заголовку If-Match или ключу документа. | Строка (String) | Нет |
| batch | Хранить документ в пакетном режиме. Возможные значения: ok. | Строка (String) | Нет |
| new_edits | Предотвращает вставку противоречивого документа. Возможные значения: true (по умолчанию) и false. Если значение false, в документ должно быть включено хорошо сформированное _rev. new_edits=false используется репликатором для вставки документов в целевую базу данных, даже если это приводит к возникновению конфликтов. Необязательный параметр, значение false предназначено для использования только репликатором.. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 271.

```
PUT /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Content-Length: 196
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Запрос**
**Рисунок 271**

### 4.5.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 433.

**Таблица 433 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8<br>multipart/related | Строка (String) | Да |
| ETag | Новая ревизия цитируемого документа | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 434.

**Таблица 434 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Токен MVCC ревизии | Строка (String) | Да |

Коды состояния отображает Таблица 435.

**Таблица 435 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Документ создан и сохранен на диске |
| 202 Accepted | Данные документа приняты, но еще не сохранены на диске |
| 400 Bad Request | Неверное содержание запроса или параметры |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | Указанная база данных или идентификатор документа не существует |
| 409 Conflict | Документ с указанным ID уже существует или указанная ревизия не является последней для целевого документа |

Ответ в случае успешного завершения запроса отображает Рисунок 272. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 85
Content-Type: application/json
Date: Wed, 14 Aug 2013 20:31:39 GMT
ETag: "1-917fa2381192822767f010b95b45325b"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs",
    "ok": true,
    "rev": "1-917fa2381192822767f010b95b45325b"
}
```

**Ответ**
**Рисунок 272**

### 4.5.4 Пометить указанный документ как удаленный

Помечает указанный документ как удаленный, добавляя поле _deleted со значением true. Документы с таким полем больше не будут возвращаться в запросах, но останутся в базе данных. Необходимо указать текущую (последнюю) ревизию, либо используя параметр rev, либо используя заголовок If-Match для указания ревизии.

БД «Енисей» не удаляет указанный документ полностью. Вместо этого он оставляет метаданные с очень простой информацией о документе. Эти метаданные необходимы для того, чтобы действие удаления можно было реплицировать между базами данных.

#### 4.5.4.1. Формат запроса

Параметры запроса отображает Таблица 436.

**Таблица 436 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid} |
| Метод | DELETE |

Параметры строки запроса отображает Таблица 437.

**Таблица 437 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 438.

**Таблица 438 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| If-Match | Ревизия документа. Альтернатива параметру запроса rev или ключу документа | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 439.

**Таблица 439 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Фактическая ревизия документа. | Строка (String) | Да |
| batch | Хранить документ в пакетном режиме. Возможные значения: ok. | Строка (String) | Нет |

Пример запроса отображает Рисунок 273.

```
DELETE /recipes/FishStew?rev=1-9c65296036141e575d32ba9c034dd3ee HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 273**

В качестве альтернативы вместо параметра запроса rev можно использовать заголовок If-Match, что отображает Рисунок 274.

```
DELETE /recipes/FishStew HTTP/1.1
Accept: application/json
If-Match: 1-9c65296036141e575d32ba9c034dd3ee
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 274**

### 4.5.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 440.

**Таблица 440 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8<br>multipart/related | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 441.

**Таблица 441 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Токен MVCC ревизии | Строка (String) | Да |

Коды состояния отображает Таблица 442.

**Таблица 442 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ успешно удален |
| 202 Accepted | Запрос принят, но изменения еще не сохранены на диске |
| 400 Bad Request | Неверное содержание запроса или параметры |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | Указанная база данных или идентификатор документа не существует |
| 409 Conflict | Указанная ревизия не является последней для целевого документа |

Ответ в случае успешного завершения запроса отображает Рисунок 275. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 71
Content-Type: application/json
Date: Wed, 14 Aug 2013 12:23:13 GMT
ETag: "2-056f5f44046ecafc08a2bc2b9c229e20"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "FishStew",
    "ok": true,
    "rev": "2-056f5f44046ecafc08a2bc2b9c229e20"
}
```

**Ответ**
**Рисунок 275**

### 4.5.5 Копировать документ в новый или существующий

#### 4.5.5.1. Формат запроса

Параметры запроса отображает Таблица 443.

**Таблица 443 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid} |
| Метод | COPY |

Метод COPY (не стандартный HTTP) копирует существующий документ в новый или уже существующий документ. Копирование документа возможно только в рамках одной базы данных.

Источник документа указывается в строке запроса с заголовком Destination запроса, указывающего целевой документ.

Параметры строки запроса отображает Таблица 444.

**Таблица 444 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 445.

**Таблица 445 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Destination | Документ назначения. Должен содержать идентификатор целевого документа и, необязательно, версию целевого документа, если копируется в существующий документ | Строка (String) | Нет |
| If-Match | Ревизия документа источника. Альтернативно параметру запроса | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 446.

**Таблица 446 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия для копирования из нее. | Строка (String) | Нет |
| batch | Хранить документ в пакетном режиме. Возможные значения: ok | Строка (String) | Нет |

Пример запроса отображает Рисунок 276.

```
COPY /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Destination: SpaghettiWithMeatballs_Italian
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 276**

#### 4.5.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 447.

**Таблица 447 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 448.

**Таблица 448 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Токен MVCC ревизии | Строка (String) | Да |

Коды состояния отображает Таблица 449.

**Таблица 449 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Документ успешно создан |
| 202 Accepted | Запрос принят, но изменения еще не сохранены на диске |
| 400 Bad Request | Неверное содержание запроса или параметры |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | Указанная база данных или идентификатор документа не существует |
| 409 Conflict | Указанная ревизия не является последней для целевого документа |

Ответ в случае успешного завершения запроса отображает Рисунок 277. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 93
Content-Type: application/json
Date: Wed, 14 Aug 2013 14:21:00 GMT
ETag: "1-e86fdf912560c2321a5fcefc6264e6d9"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs_Italian
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs_Italian",
    "ok": true,
    "rev": "1-e86fdf912560c2321a5fcefc6264e6d9"
}
```

**Ответ**
**Рисунок 277**

### 4.5.5.3. Вложения

Если документ содержит вложения, то возвращаемая структура будет содержать сводку вложений, связанных с документом, но не сами данные вложений.

JSON для возвращаемого документа будет включать поле _attachments с одним или несколькими определениями вложений.

Ключами объекта _attachments являются имена вложений, а значениями - информационные объекты со следующей структурой, которую отображает Таблица 450.

**Таблица 450 – Информационные объекты**

| Поле | Описание | Тип |
|---|---|---|
| content_type | MIME-тип вложения | Строка (String) |
| data | Содержимое в Base64-кодировке. Доступен, если содержимое вложений запрашивается с помощью следующих параметров запроса: attachments=true при запросе документа attachments=true&include_docs=true при запросе ленты изменений или представления atts_since. | Строка (String) |
| digest | Хэш-дайджест содержимого. Он начинается с префикса, который объявляет тип хэша (md5-), и продолжается хэш-дайджестом в Base64-кодировке. | Строка (String) |
| encoded_length | Размер сжатого вложения в байтах. Доступно, если content_type находится в списке сжимаемых типов при добавлении вложения и указаны следующие параметры запроса: att_encoding_info=true при запросе документа att_encoding_info=true&include_docs=true при запросе ленты изменений или представления | Число (int64) |
| encoding | Кодек сжатия. Доступен, если content_type находится в списке сжимаемых типов при добавлении вложения и указаны следующие параметры запроса: att_encoding_info=true при запросе документа att_encoding_info=true&include_docs=true при запросе ленты изменений или представления. | Строка (String) |
| length | Реальный размер вложения в байтах. Недоступен, если запрашивается содержимое вложения | Число (int64) |
| revpos | Номер ревизии, когда было добавлено вложение | Число (int64) |
| stub | Имеет значение true, если объект содержит информацию о заглушке и не содержит содержимого. В противном случае опущен в ответе | Логический тип (Boolean) |

#### 4.5.5.3.1. Основная информация о вложениях

##### 4.5.5.3.1.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 451.

**Таблица 451 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 278.

```
GET /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 278**

#### 4.5.5.3.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 452.

**Таблица 452 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 279. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 660
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "5-fd96acb3256302bf0dd2f32713161f2a"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_attachments": {
        "grandma_recipe.txt": {
            "content_type": "text/plain",
            "digest": "md5-Ids41vtv725jyrN7iUvMcQ==",
            "length": 1872,
            "revpos": 4,
            "stub": true
        },
        "my_recipe.txt": {
            "content_type": "text/plain",
            "digest": "md5-198BPPNiT5fqlLxoYYbjBA==",
            "length": 85,
            "revpos": 5,
            "stub": true
        },
        "photo.jpg": {
            "content_type": "image/jpeg",
            "digest": "md5-7Pv4HW2822WY1r/3WDbPug==",
            "length": 165504,
            "revpos": 2,
            "stub": true
        }
    },
    "_id": "SpaghettiWithMeatballs",
    "_rev": "5-fd96acb3256302bf0dd2f32713161f2a",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**
**Рисунок 279**

### 4.5.5.3.2. Получение содержимого прикрепленных файлов

С помощью параметра запроса attachments=true можно получить документ со всем содержимым прикрепленных файлов.

#### 4.5.5.3.2.1. Формат запроса

Пример запроса отображает Рисунок 280.

```
GET /db/pixel?attachments=true HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 280**

Получить содержимое прикрепленных файлов с определенной ревизии с помощью параметра запроса atts_since, что отображает Рисунок 281.

```
GET /recipes/SpaghettiWithMeatballs?atts_since=[%224-874985bc28906155ba0e2e0538f67b05%22]  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 281**

#### 4.5.5.3.2.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 282. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 553
Content-Type: application/json
Date: Wed, 14 Aug 2013 11:32:40 GMT
ETag: "4-f1bcae4bf7bbb92310079e632abfe3f4"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_attachments": {
        "pixel.gif": {
            "content_type": "image/gif",
            "data": "R0lGODlhAQABAIAAAAAAAP///yH5BAEAAAAALAAAAAABAAEAAAIBRAA7",
            "digest": "md5-2JdGiI2i2VELZKnwMers1Q==",
            "revpos": 2
        },
        "pixel.png": {
            "content_type": "image/png",
            "data": "iVBORw0KGgoAAAANSUhEUgAAAAEAAAABAQMAAAAl21bKAAAAAXNSR0IArs4c6QAAAANQTFRFAAAAp3o
92gAAAAF0Uk5TAEDm2GYAAAABYktHRACIBR1IAAAACXBIWXMAAAsTAAALEwEAmpwYAAAAB3RJTUUH3QgOCx8VHgmcNwAAAApJREF
UCNdjYAAAAAIAAeIhvDMAAAAASUVORK5CYII=",
            "digest": "md5-Dgf5zxgGuchWrve73evvGQ==",
            "revpos": 3
        }
    },
    "_id": "pixel",
    "_rev": "4-f1bcae4bf7bbb92310079e632abfe3f4"
}
```

**Ответ**
**Рисунок 282**

Ответ на запрос содержимого прикрепленных файлов с определенной ревизии отображает Рисунок 283.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 760
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "5-fd96acb3256302bf0dd2f32713161f2a"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_attachments": {
        "grandma_recipe.txt": {
            "content_type": "text/plain",
            "digest": "md5-Ids41vtv725jyrN7iUvMcQ==",
            "length": 1872,
            "revpos": 4,
            "stub": true
        },
        "my_recipe.txt": {
            "content_type": "text/plain",
            "data": "MS4gQ29vayBzcGFnaGV0dGkKMi4gQ29vayBtZWV0YmFsbHMKMy4gTWl4IHRoZW0KNC4gQWRkIHRvbWF
0byBzYXVjZQo1LiAuLi4KNi4gUFJPRklUIQ==",
            "digest": "md5-198BPPNiT5fqlLxoYYbjBA==",
            "revpos": 5
        },
        "photo.jpg": {
            "content_type": "image/jpeg",
            "digest": "md5-7Pv4HW2822WY1r/3WDbPug==",
            "length": 165504,
            "revpos": 2,
            "stub": true
        }
    },
    "_id": "SpaghettiWithMeatballs",
    "_rev": "5-fd96acb3256302bf0dd2f32713161f2a",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**

**Рисунок 283**

#### 4.5.5.3.3. Эффективное извлечение нескольких вложений

Как было отмечено выше, получение документа с attachments=true возвращает большой объект JSON со всеми вложениями. Когда документ и файлы меньше, это нормально, но если было прикреплено что-то большее, например, медиафайлы (аудио/видео), разбор такого ответа может быть очень дорогим.

Чтобы решить эту проблему, БД «Енисей» позволяет получать документы в формате multipart/related.

##### 4.5.5.3.3.1. Формат запроса

Пример запроса отображает Рисунок 284.

```
GET /recipes/secret?attachments=true HTTP/1.1
Accept: multipart/related
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 284**

#### 4.5.5.3.3.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 285. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Length: 538
Content-Type: multipart/related; boundary="e89b3e29388aef23453450d10e5aaed0"
Date: Sat, 28 Sep 2013 08:08:22 GMT
ETag: "2-c1c6c44c4bc3c9344b037c8690468605"
Server: Yenisei/1.0.0 (Erlang OTP/24)

--e89b3e29388aef23453450d10e5aaed0
Content-Type: application/json

{"_id":"secret","_rev":"2-
c1c6c44c4bc3c9344b037c8690468605","_attachments":{"recipe.txt":{"content_type":"text/plain","revpos"
:2,"digest":"md5-HV9aXJdEnu0xnMQYTKgOFA==","length":86,"follows":true}}}
--e89b3e29388aef23453450d10e5aaed0
Content-Disposition: attachment; filename="recipe.txt"
Content-Type: text/plain
Content-Length: 86

1. Take R
2. Take E
3. Mix with L
4. Add some A
5. Serve with X

--e89b3e29388aef23453450d10e5aaed0--
```

**Ответ**
**Рисунок 285**

В этом ответе документ содержит только информацию о корешке вложений, и довольно короткое время все вложения идут как отдельные сущности, что уменьшает объем памяти и накладные расходы на обработку (вы ведь заметили, что содержимое вложений идет как необработанные данные, а не в кодировке base64?)

#### 4.5.5.3.4. Получение информации о кодировке вложений¶

Используя параметр запроса att_encoding_info=true, Можно получить информацию о размере сжатых вложений и использованном кодеке.

Пример запроса отображает Рисунок 286.

#### 4.5.5.3.4.1.1 Формат запроса

```
GET /recipes/SpaghettiWithMeatballs?att_encoding_info=true HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 286**

#### 4.5.5.3.4.1.2 Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 287. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 736
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "5-fd96acb3256302bf0dd2f32713161f2a"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_attachments": {
        "grandma_recipe.txt": {
            "content_type": "text/plain",
            "digest": "md5-Ids41vtv725jyrN7iUvMcQ==",
            "encoded_length": 693,
            "encoding": "gzip",
            "length": 1872,
            "revpos": 4,
            "stub": true
        },
        "my_recipe.txt": {
            "content_type": "text/plain",
            "digest": "md5-198BPPNiT5fqlLxoYYbjBA==",
            "encoded_length": 100,
            "encoding": "gzip",
            "length": 85,
            "revpos": 5,
            "stub": true
        },
        "photo.jpg": {
            "content_type": "image/jpeg",
            "digest": "md5-7Pv4HW2822WY1r/3WDbPug==",
            "length": 165504,
            "revpos": 2,
            "stub": true
        }
    },
    "_id": "SpaghettiWithMeatballs",
    "_rev": "5-fd96acb3256302bf0dd2f32713161f2a",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**
**Рисунок 287**

#### 4.5.5.3.5. Получение информации о кодировке вложений

Чтобы создать документ с несколькими вложениями с помощью одного запроса, достаточно вставить данные вложений в кодировке base64 в текстовое поле документа, что отображает Рисунок 288.

```
{
  "_id":"multiple_attachments",
  "_attachments":
  {
    "foo.txt":
    {
      "content_type":"text\/plain",
      "data": "VGhpcyBpcyBhIGJhc2U2NCBlbmNvZGVkIHRleHQ="
    },

   "bar.txt":
    {
      "content_type":"text\/plain",
      "data": "VGhpcyBpcyBhIGJhc2U2NCBlbmNvZGVkIHRleHQ="
    }
  }
}
```

**Текстовое поле документа**
**Рисунок 288**

Кроме того, Можно загрузить документ с вложениями более эффективно в формате multipart/related. Это позволяет избежать необходимости Base64-кодирования вложений, что экономит процессор и пропускную способность. Для этого в заголовке Content-Type запроса PUT /{db}/{docid} установите значение multipart/related.

Первым MIME-телом будет сам документ, который должен иметь свой Content-Type application/json". Оно также должно включать объект метаданных _attachments, в котором каждый объект вложения имеет ключ follows со значением true.

Последующие MIME-тела являются вложениями.

##### 4.5.5.3.5.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 453.

**Таблица 453 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | multipart/related | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 289.

```
PUT /temp/somedoc HTTP/1.1
Accept: application/json
Content-Length: 372
Content-Type: multipart/related;boundary="abc123"
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
User-Agent: HTTPie/0.6.0

--abc123
Content-Type: application/json

{
    "body": "This is a body.",
    "_attachments": {
        "foo.txt": {
            "follows": true,
            "content_type": "text/plain",
            "length": 21
        },
        "bar.txt": {
            "follows": true,
            "content_type": "text/plain",
            "length": 20
        }
    }
}

--abc123

this is 21 chars long
--abc123

this is 20 chars lon
--abc123--
```

**Запрос**
**Рисунок 289**

#### 4.5.5.3.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 454.

**Таблица 454 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 290. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 72
Content-Type: application/json
Date: Sat, 28 Sep 2013 09:13:24 GMT
ETag: "1-5575e26acdeb1df561bb5b70b26ba151"
Location: http://localhost:5984/temp/somedoc
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "somedoc",
    "ok": true,
    "rev": "1-5575e26acdeb1df561bb5b70b26ba151"
}
```

**Ответ**
**Рисунок 290**

#### 4.5.5.4. Получение списка ревизий

Можно получить список редакций для данного документа, добавив параметр revs=true в URL запроса

##### 4.5.5.4.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 455.

**Таблица 455 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 291.

```
GET /recipes/SpaghettiWithMeatballs?revs=true  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 291**

#### 4.5.5.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 456.

**Таблица 456 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Возвращаемая структура JSON включает исходный документ, в том числе структуру _revisions, которая включает информацию о ревизиях в следующей форме, которую отображает Таблица 457.

**Таблица 457 – JSON-объекты**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ids | Массив действительных идентификаторов ревизий, в обратном порядке (сначала последняя). | Массив (Array) | Да |
| start | Префиксный номер последней ревизии | Число (int64) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 292. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 584
Content-Type: application/json
Date: Wed, 14 Aug 2013 11:38:26 GMT
ETag: "5-fd96acb3256302bf0dd2f32713161f2a"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_id": "SpaghettiWithMeatballs",
    "_rev": "8-6f5ad8db0f34af24a6e0984cd1a6cfb9",
    "_revisions": {
        "ids": [
            "6f5ad8db0f34af24a6e0984cd1a6cfb9",
            "77fba3a059497f51ec99b9b478b569d2",
            "136813b440a00a24834f5cb1ddf5b1f1",
            "fd96acb3256302bf0dd2f32713161f2a",
            "874985bc28906155ba0e2e0538f67b05",
            "0de77a37463bf391d14283e626831f2e",
            "d795d1b924777732fdea76538c558b62",
            "917fa2381192822767f010b95b45325b"
        ],
        "start": 8
    },
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**

**Рисунок 292**

### 4.5.5.5. Получение расширенной истории ревизий

Можно получить дополнительную информацию о ревизиях для данного документа, указав в запросе аргумент revs_info.

#### 4.5.5.5.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 458.

**Таблица 458 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 293.

```
GET /recipes/SpaghettiWithMeatballs?revs_info=true  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 293**

#### 4.5.5.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 459.

**Таблица 459 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 294. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 802
Content-Type: application/json
Date: Wed, 14 Aug 2013 11:40:55 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_id": "SpaghettiWithMeatballs",
    "_rev": "8-6f5ad8db0f34af24a6e0984cd1a6cfb9",
    "_revs_info": [
        {
            "rev": "8-6f5ad8db0f34af24a6e0984cd1a6cfb9",
            "status": "available"
        },
        {
            "rev": "7-77fba3a059497f51ec99b9b478b569d2",
            "status": "deleted"
        },
        {
            "rev": "6-136813b440a00a24834f5cb1ddf5b1f1",
            "status": "available"
        },
        {
            "rev": "5-fd96acb3256302bf0dd2f32713161f2a",
            "status": "missing"
        },
        {
            "rev": "4-874985bc28906155ba0e2e0538f67b05",
            "status": "missing"
        },
        {
            "rev": "3-0de77a37463bf391d14283e626831f2e",
            "status": "missing"
        },
        {
            "rev": "2-d795d1b924777732fdea76538c558b62",
            "status": "missing"
        },
        {
            "rev": "1-917fa2381192822767f010b95b45325b",
            "status": "missing"
        }
    ],
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**

**Рисунок 294**

Возвращаемый документ содержит поле _revs_info с расширенной информацией о ревизии, включая наличие и статус каждой ревизии. Это поле массива содержит объекты со следующей структурой, которые отображает Таблица 460.

**Таблица 460 – Объекты поля массива**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Полная строка ревизии | Строка (String) | Да |
| status | Статус ревизии. Может быть одним из:<br>available (доступен): Ревизия доступна для получения с помощью параметра запроса rev<br>missing (отсутствует): Пересмотр недоступен<br>deleted (удален): Пересмотр принадлежит удаленному документу | Строка (String) | Да |

#### 4.5.5.6. Получение конкретной ревизии

Чтобы получить конкретную ревизию, используйте аргумент rev в запросе и укажите полный номер ревизии. Будет возвращена указанная ревизия документа, включая поле _rev, указывающее ревизию, которая была запрошена.

##### 4.5.5.6.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 461.

**Таблица 461 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 295.

```
GET /recipes/SpaghettiWithMeatballs?rev=6-136813b440a00a24834f5cb1ddf5b1f1  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 295**

#### 4.5.5.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 462.

**Таблица 462 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 296. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 271
Content-Type: application/json
Date: Wed, 14 Aug 2013 11:40:55 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_id": "SpaghettiWithMeatballs",
    "_rev": "6-136813b440a00a24834f5cb1ddf5b1f1",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs"
}
```

**Ответ**
**Рисунок 296**

#### 4.5.5.6.3. Извлечение удаленных документов

БД «Енисей» фактически не удаляет документы через DELETE /{db}/{docid}. Вместо этого, он оставляет метаданные с очень базовой информацией о документе. Если просто используется GET /{db}/{docid} БД «Енисей» вернет ответ 404 Not Found.

#### 4.5.5.6.3.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 463.

**Таблица 463 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 297.

```
GET /recipes/FishStew  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 297**

Однако Можно получить метаданные документа, используя параметр запроса rev в запросе GET /{db}/{docid}:

Пример запроса отображает Рисунок 298.

```
GET /recipes/FishStew?rev=2-056f5f44046ecafc08a2bc2b9c229e20  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 298**

#### 4.5.5.6.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 464.

**Таблица 464 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 299. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 404 Object Not Found
Cache-Control: must-revalidate
Content-Length: 41
Content-Type: application/json
Date: Wed, 14 Aug 2013 12:23:27 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "error": "not_found",
    "reason": "deleted"
}
```

**Ответ**
**Рисунок 299**

Ответ на запрос с использованием rev отображает Рисунок 300.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 79
Content-Type: application/json
Date: Wed, 14 Aug 2013 12:30:22 GMT
ETag: "2-056f5f44046ecafc08a2bc2b9c229e20"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_deleted": true,
    "_id": "FishStew",
    "_rev": "2-056f5f44046ecafc08a2bc2b9c229e20"
}
```

**Ответ**
**Рисунок 300**

#### 4.5.5.7. Обновление существующего документа

Чтобы обновить существующий документ, необходимо указать номер текущей редакции в параметре _rev.

##### 4.5.5.7.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 465.

**Таблица 465 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| If-Match | Ревизия документа. Альтернатива параметру запроса rev или ключу документа | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 301.

```
PUT /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Content-Length: 258
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "_rev": "1-917fa2381192822767f010b95b45325b",
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs",
    "serving": "hot"
}
```

**Запрос**
**Рисунок 301**

В качестве альтернативы Можно указать номер текущей ревизии в HTTP-заголовке If-Match запроса:

Пример запроса отображает Рисунок 302.

```
PUT /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Content-Length: 258
Content-Type: application/json
If-Match: 1-917fa2381192822767f010b95b45325b
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

{
    "description": "An Italian-
American dish that usually consists of spaghetti, tomato sauce and meatballs.",
    "ingredients": [
        "spaghetti",
        "tomato sauce",
        "meatballs"
    ],
    "name": "Spaghetti with meatballs",
    "serving": "hot"
}
```

**Запрос**
**Рисунок 302**

### 4.5.5.7.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 466.

**Таблица 466 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 303. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 85
Content-Type: application/json
Date: Wed, 14 Aug 2013 20:33:56 GMT
ETag: "2-790895a73b63fb91dd863388398483dd"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs",
    "ok": true,
    "rev": "2-790895a73b63fb91dd863388398483dd"
}
```

**Ответ**
**Рисунок 303**

### 4.5.5.8. Копирование из определенной редакции

Чтобы скопировать из определенной версии, используйте аргумент rev в строке запроса или If-Match.

#### 4.5.5.8.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 467.

**Таблица 467 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| If-Match | Ревизия документа. Альтернатива параметру запроса rev или ключу документа | Строка (String) | Нет |
| Destination | Документ назначения. Должен содержать идентификатор целевого документа и, необязательно, версию целевого документа, если копируется в существующий документ | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 304.

```
COPY /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Destination: SpaghettiWithMeatballs_Original
If-Match: 1-917fa2381192822767f010b95b45325b
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 304**

#### 4.5.5.8.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 468.

**Таблица 468 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 305. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 93
Content-Type: application/json
Date: Wed, 14 Aug 2013 14:21:00 GMT
ETag: "1-917fa2381192822767f010b95b45325b"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs_Original
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs_Original",
    "ok": true,
    "rev": "1-917fa2381192822767f010b95b45325b"
}
```

**Ответ**
**Рисунок 305**

### 4.5.5.9. Копирование в существующий документ

Для копирования в существующий документ необходимо указать текущую строку ревизии для целевого документа, добавив параметр rev к строке заголовка Destination.

#### 4.5.5.9.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 469.

**Таблица 469 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Destination | Документ назначения. Должен содержать идентификатор целевого документа и, необязательно, версию целевого документа, если копируется в существующий документ | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 306.

```
COPY /recipes/SpaghettiWithMeatballs?rev=8-6f5ad8db0f34af24a6e0984cd1a6cfb9 HTTP/1.1
Accept: application/json
Destination: SpaghettiWithMeatballs_Original?rev=1-917fa2381192822767f010b95b45325b
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 306**

#### 4.5.5.9.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 470.

**Таблица 470 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 307. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 93
Content-Type: application/json
Date: Wed, 14 Aug 2013 14:21:00 GMT
ETag: "2-62e778c9ec09214dd685a981dcc24074"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs_Original
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs_Original",
    "ok": true,
    "rev": "2-62e778c9ec09214dd685a981dcc24074"
}
```

**Ответ**
**Рисунок 307**

### 4.5.6 Получить HTTP-заголовки, содержащие минимальное количество информации об указанном вложении

Возвращает HTTP-заголовки, содержащие минимальное количество информации об указанном вложении. Метод поддерживает те же аргументы запроса, что и метод GET /{db}/{docid}/{attname}, но возвращается только информация заголовка (включая размер вложения, кодировку и MD5-хэш в виде ETag).

#### 4.5.6.1. Формат запроса

Параметры запроса отображает Таблица 471.

**Таблица 471 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid}/{attname} |
| Метод | HEAD |

Параметры строки запроса отображает Таблица 472.

**Таблица 472 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |
| attname | Имя вложения |

Параметры HTTP заголовка запроса отображает Таблица 473.

**Таблица 473 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| If-Match | Ревизия документа. Альтернатива параметру запроса rev | Строка (String) | Да |
| If-None-Match | Двоичный дайджест MD5 вложения в кодировке base64 | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Дополнительные параметры запроса отображает Таблица 474.

**Таблица 474 – Дополнительные параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия документа. | Строка (String) | Нет |

Пример запроса отображает Рисунок 308.

```
HEAD /recipes/SpaghettiWithMeatballs/recipe.txt HTTP/1.1
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 308**

#### 4.5.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 475.

**Таблица 475 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept-Ranges | Запрос на определение диапазона. Используется для вложений с типом содержимого application/octet-stream | Строка (String) | Да |
| Content-Encoding | Используемый кодек сжатия. Доступен, если тип содержимого content_type вложения находится в списке сжимаемых типов | Строка (String) | Да |
| Content-Length | Размер вложения. Если использовался кодек сжатия, это значение относится к сжатому размеру, а не к фактическому | Строка (String) | Да |
| ETag | Бинарный дайджест MD5 с двойными кавычками в кодировке base64 | Строка (String) | Да |

Коды состояния отображает Таблица 476.

**Таблица 476 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Вложение существует |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | Указанная база данных, документ или вложение не найдены |

Ответ в случае успешного завершения запроса отображает Рисунок 309. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Accept-Ranges: none
Cache-Control: must-revalidate
Content-Encoding: gzip
Content-Length: 100
Content-Type: text/plain
Date: Thu, 15 Aug 2013 12:42:42 GMT
ETag: "vVa/YgiE1+Gh0WfoFJAcSg=="
Server: Yenisei/1.0.0 (Erlang OTP/24)
```

**Ответ**
**Рисунок 309**

### 4.5.7 Получить вложение файла, связанное с документом

Возвращает вложение файла, связанное с документом. Возвращаются необработанные данные связанного вложения (так же, как если бы обращение осуществлялось к статическому файлу). Возвращаемый Content-Type будет соответствовать типу содержимого, установленному при отправке вложения документа в базу данных.

#### 4.5.7.1. Формат запроса

Параметры запроса отображает Таблица 477.

**Таблица 477 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid}/{attname} |
| Метод | GET |

Параметры строки запроса отображает Таблица 478.

**Таблица 478 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |
| attname | Имя вложения |

Параметры HTTP заголовка запроса отображает Таблица 479.

**Таблица 479 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| If-Match | Ревизия документа. Альтернатива параметру запроса rev | Строка (String) | |
| If-None-Match | Двоичный дайджест MD5 вложения в кодировке base64 | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Дополнительные параметры запроса отображает Таблица 480.

**Таблица 480 – Дополнительные параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия документа. | Строка (String) | Нет |

Пример запроса отображает Рисунок 310.

```
GET /recipes/SpaghettiWithMeatballs/recipe.txt HTTP/1.1
If-Match: 3-022371c6aae1aab7a91d8bdaf3a37b40
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 310**

#### 4.5.7.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 481.

**Таблица 481 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept-Ranges | Запрос на определение диапазона. Используется для вложений с типом содержимого application/octet-stream | Строка (String) | |
| Content-Encoding | Используемый кодек сжатия. Доступен, если тип содержимого content_type вложения находится в списке сжимаемых типов | Строка (String) | |
| Content-Length | Размер вложения. Если использовался кодек сжатия, это значение относится к сжатому размеру, а не к фактическому | Строка (String) | |
| ETag | Бинарный дайджест MD5 с двойными кавычками в кодировке base64 | Строка (String) | |

Коды состояния отображает Таблица 482.

**Таблица 482 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Вложение существует |
| 401 Unauthorized | Требуются привилегии на чтение |
| 404 Not Found | Указанная база данных, документ или вложение не найдены |

Ответ в случае успешного завершения запроса отображает Рисунок 311. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Accept-Ranges: none
Cache-Control: must-revalidate
Content-Encoding: gzip
Content-Length: 64
Content-Type: text/plain
Date: Thu, 15 Aug 2013 12:42:42 GMT
ETag: "vVa/YgiE1+Gh0WfoFJAcSg=="
Server: Yenisei/1.0.0 (Erlang OTP/24)
Information
```

**Ответ**
**Рисунок 311**

### 4.5.8 Загрузить содержимое в качестве вложения к документу

Загружает указанное содержимое в качестве вложения к указанному документу. Имя вложения должно быть строкой в кодировке URL. Необходимо предоставить заголовок Content-Type, а для существующего документа предоставить либо аргумент запроса rev, либо HTTP-заголовок If-Match. Если ревизия опущена, будет создан новый, иначе пустой документ с предоставленным вложением, или возникнет конфликт.

Если при загрузке вложения используется существующее имя вложения, БД «Енисей» обновит соответствующее хранимое содержимое базы данных. Поскольку для добавления вложения в документ необходимо предоставить информацию о ревизии, это служит проверкой для обновления существующего вложения.

Загрузка вложения обновляет соответствующую ревизию документа. Ревизии отслеживаются для родительского документа, а не для отдельных вложений.

#### 4.5.8.1. Формат запроса

Параметры запроса отображает Таблица 483.

**Таблица 483 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid}/{attname} |
| Метод | PUT |

Параметры строки запроса отображает Таблица 484.

**Таблица 484 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |
| attname | Имя вложения |

Параметры HTTP заголовка запроса отображает Таблица 485.

**Таблица 485 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | MIME-тип вложения. По умолчанию: application/octet-stream | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 486.

**Таблица 486 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия документа. | Строка (String) | Нет |

Пример запроса отображает Рисунок 312.

```
PUT /recipes/SpaghettiWithMeatballs/recipe.txt HTTP/1.1
Accept: application/json
Content-Length: 86
Content-Type: text/plain
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
If-Match: 1-917fa2381192822767f010b95b45325b

1. Cook spaghetti
2. Cook meatballs
3. Mix them
4. Add tomato sauce
5. ...
6. PROFIT!
```

**Запрос**

**Рисунок 312**

#### 4.5.8.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 487.

**Таблица 487 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| If-Match | Ревизия документа. Альтернатива параметру запроса rev | Строка (String) | Нет |

Объекты JSON ответа отображает Таблица 488.

**Таблица 488 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа. | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Токен MVCC ревизии | Строка (String) | Да |

Коды состояния отображает Таблица 489.

**Таблица 489 – Коды состояния**

| Код | Описание |
|---|---|
| 201 Created | Вложение создано и сохранено на диске |
| 202 Accepted | Запрос был принят, но изменения еще не сохранены на диске |
| 400 Bad Request | Неверное содержание запроса или параметры |
| 401 Unauthorized | Требуются привилегии на чтение |
| 404 Not Found | Указанная база данных, документ или вложение не найдены |
| 409 Conflict | Не указана ревизия документа или она не самая последняя |

Ответ в случае успешного завершения запроса отображает Рисунок 313. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 85
Content-Type: application/json
Date: Thu, 15 Aug 2013 12:38:04 GMT
ETag: "2-ce91aed0129be8f9b0f650a2edcfd0a4"
Location: http://localhost:5984/recipes/SpaghettiWithMeatballs/recipe.txt
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs",
    "ok": true,
    "rev": "2-ce91aed0129be8f9b0f650a2edcfd0a4"
}
```

**Ответ**
**Рисунок 313**

### 4.5.9 Удалить вложение

Удаляет вложение с именем {attname} указанного документа doc. Для удаления вложения необходимо указать параметр запроса rev или If-Match с текущей ревизией.

Удаление вложения обновляет соответствующую ревизию документа. Ревизии отслеживаются для родительского документа, а не для отдельных вложений.

#### 4.5.9.1. Формат запроса

Параметры запроса отображает Таблица 490.

**Таблица 490 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/{docid}/{attname} |
| Метод | DELETE |

Параметры строки запроса отображает Таблица 491.

**Таблица 491 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 492.

**Таблица 492 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | Да |
| If-Match | Ревизия документа. Альтернатива параметру запроса rev | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 493.

**Таблица 493 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rev | Ревизия документа. | Строка (String) | Да |
| batch | Хранить изменения в пакетном режиме. Возможные значения: ok. | Строка (String) | Нет |

Пример запроса отображает Рисунок 314.

```
DELETE /recipes/SpaghettiWithMeatballs?rev=6-440b2dd39c20413045748b42c6aba6e2 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 314**

В качестве альтернативы вместо параметра запроса `rev` можно использовать заголовок If-Match.

Пример запроса отображает Рисунок 315.

```
DELETE /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
If-Match: 6-440b2dd39c20413045748b42c6aba6e2
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 315**

#### 4.5.9.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 494.

**Таблица 494 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Новая ревизия документа с двойными кавычками | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 495.

**Таблица 495 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| id | Идентификатор документа. | Строка (String) | Да |
| ok | Статус операции | Логический тип (Boolean) | Да |
| rev | Токен MVCC ревизии | Строка (String) | Да |

Коды состояния отображает Таблица 496.

**Таблица 496 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Вложение успешно удалено |
| 202 Accepted | Запрос принят, но изменения еще не сохранены на диске |
| 400 Bad Request | Неверное имя запроса или параметры |
| 401 Unauthorized | Требуются привилегии на запись |
| 404 Not Found | Указанная база данных, документ или вложение не найдены |
| 409 Conflict | Не указана ревизия документа или она не самая последняя |

Ответ в случае успешного завершения запроса отображает Рисунок 316. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 85
Content-Type: application/json
Date: Wed, 14 Aug 2013 12:23:13 GMT
ETag: "7-05185cf5fcdf4b6da360af939431d466"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "id": "SpaghettiWithMeatballs",
    "ok": true,
    "rev": "7-05185cf5fcdf4b6da360af939431d466"
}
```

**Ответ**
**Рисунок 316**

### 4.5.9.3. Запросы с диапазоном HTTP

HTTP позволяет указывать диапазоны байтов для запросов. Это позволяет реализовать возобновляемые загрузки и пропускаемые аудио- и видеопотоки. Доступно для всех вложений внутри БД «Енисей».

Осуществить такой запрос позволяет параметр Range. Например, для получения первых 13 байт необходимо указать “Range: bytes=0-12”.

**HTTP поддерживает множество способов указания одиночных и даже множественных диапазонов байтов.**

## 4.6. Проектные документы

В БД «Енисей» проектные документы обеспечивают основной интерфейс для создания приложения БД «Енисей». В проектном документе определяются представления, используемые для извлечения информации из БД «Енисей» через одно или несколько представлений. Проектные документы создаются в вашем экземпляре БД «Енисей» таким же образом, как создаются документы базы данных, но содержание и определение документов отличается. Проектные документы именуются с помощью идентификатора, определяемого с помощью URL-пути документа проектирования, и этот URL-адрес может быть использован для доступа к содержимому базы данных.

Представления и списки работают вместе для обеспечения автоматизированного (и форматированного) вывода из вашей базы данных

### 4.6.1 Получить HTTP-заголовки с информацией о проектном документе

Возвращает HTTP-заголовки, содержащие минимальное количество информации об указанном проектном документе.

#### 4.6.1.1. Формат запроса

Параметры запроса отображает Таблица 497.

**Таблица 497 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc} |
| Метод | HEAD |

Параметры HTTP заголовка запроса отображает Таблица 498.

**Таблица 498 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 317.

```
HEAD /test_people/_design/PeopleInfo HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 317**

#### 4.6.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 499.

**Таблица 499 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | Размер документа | Строка (String) | Да |

Коды состояния отображает Таблица 500.

**Таблица 500 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ существует |
| 401 Unauthorized | Требуется привилегия чтения |
| 404 Not Found | Документ не найден |

Ответ в случае успешного завершения запроса отображает Рисунок 318. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: " 1-09fc8f33e21a187166d2c3ad15c0df55"
Server: Yenisei/1.0.0 (Erlang OTP/24)
```

**Ответ**
**Рисунок 318**

### 4.6.2 Получить содержимое проектного документа

Возвращает содержимое проектного документа, указанного с помощью имени проектного документа и указанной базы данных из URL. Если не будет запрошена конкретная ревизия, всегда будет возвращена последняя ревизия документа.

#### 4.6.2.1. Формат запроса

Параметры запроса отображает Таблица 501.

**Таблица 501 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 502.

**Таблица 502 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса к тестовой базе данных отображает Рисунок 319.

```
GET /test_people/_design/PeopleInfo  HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 319**

#### 4.6.2.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 503.

**Таблица 503 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 320. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "1-09fc8f33e21a187166d2c3ad15c0df55"
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "_id": "_design/PeopleInfo",
    "_rev": "1-09fc8f33e21a187166d2c3ad15c0df55",
    "views": {
        "ListView": {
            "map": "function (doc) {\n  emit(doc._id, {\"Фамилия\": doc.Фамилия, \"Имя\": doc.Имя});
\n}"
        }
    },
    "language": "javascript"
}
```

**Ответ**
**Рисунок 320**

### 4.6.3 Создать новый проектный документ или новую ревизию

Метод PUT создает новый именованный проектный документ или создает новую ревизию существующего проектного документа.

#### 4.6.3.1. Формат запроса

Параметры запроса отображает Таблица 504.

**Таблица 504 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc} |
| Метод | PUT |

Проектные документы имеют определенное соглашение по своим полям и структуре. Параметры запроса отображает Таблица 505.

**Таблица 505 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| language | Определяет сервер запросов для обработки функций конструкторского документа | Строка (String) | Нет |
| options | Опции представления по умолчанию | Объект (Json Object) | Нет |
| filters | Определение функций фильтра | Объект (Json Object) | Нет |
| lists | Определение функций списка. Утратил актуальность. | Объект (Json Object) | Нет |
| rewrites | Определение правил перезаписи. Утратил актуальность. | Массив строк (Array of String) | Нет |
| shows | Показать определение функций. Утратил актуальность. | Объект (Json Object) | Нет |
| updates | Обновить определение функций | Объект (Json Object) | Нет |
| validate_doc_update | Источник функции валидации обновления документа | Строка (String) | Нет |
| views | Определение функций представления. | Объект (Json Object) | Нет |
| autoupdate | Указывает, следует ли автоматически строить индексы, определенные в этом проектном документе. По умолчанию - true. | Логический тип (Boolean) | Нет |

Следует обратить внимание, что для фильтров, списков, показов и обновлений полей объекты являются отображением имени функции на строковый исходный код функции. Для представлений отображение такое же, за исключением того, что значения представляют собой объекты с ключами map и reduce (необязательно), которые также содержат исходный код функций.

Параметры запроса отображает Таблица 506.

**Таблица 506 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса к тестовой базе данных отображает Рисунок 321.

```
PUT  /test_people/_design/PeopleInfo/List?rev=3-88422ba49354b4c6a29b29dfc28f31af HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "_id":"_design/PeopleInfo",
    "_rev":"3-88422ba49354b4c6a29b29dfc28f31af",
    "views":
    {"List":{
        "map":"function (doc) {\n  emit(doc._id, {\"Фамилия\": doc.second_name, \"Имя\": doc.first_nam
e});\n}"}},
        "language":"javascript",
        "_attachments":{
            "List":{
                "content_type":"application/json",
                "revpos":3,
                "digest":"md5-BLk3abJhKmqgwh9PlFaisQ==",
                "length":228,
                "stub":true
          }
     }
}
```

**Запрос**
**Рисунок 321**

### 4.6.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 507.

**Таблица 507 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | Новое расположение URI | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 324. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 81
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
Location: http://localhost:5984/test_people/_design%2FPeopleInfo/List
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfo",
    "rev": "3-88422ba49354b4c6a29b29dfc28f31af"
}
```

**Ответ**
**Рисунок 322**

### 4.6.4 Удалить документ из базы данных

Удаляет указанный документ из базы данных. Необходимо предоставить текущую (последнюю) ревизию, либо с помощью параметра rev указать ревизию.

#### 4.6.4.1. Формат запроса

Параметры запроса отображает Таблица 508.

**Таблица 508 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc} |
| Метод | DELETE |

Параметры HTTP заголовка запроса отображает Таблица 509.

**Таблица 509 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 323.

```
DELETE /test_people/_design/PeopleInfoCopied?rev=1-09fc8f33e21a187166d2c3ad15c0df55 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 323**

#### 4.6.4.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 510.

**Таблица 510 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 324. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "1-09fc8f33e21a187166d2c3ad15c0df55"
Location: http://51.250.31.135:5984/test_people/_design%2FPeopleInfoCopied
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfoCopied",
    "rev": "2-2db47b673cf04972102c760dd4ccb32a"
}
```

**Ответ**
**Рисунок 324**

### 4.6.5 Копировать существующий проектный документ

COPY (нестандартный HTTP) копирует существующий проектный документ в новый или существующий документ.

Учитывая, что индексы представлений на диске называются по MD5-хэшу определения представления, и что операция COPY фактически не изменит это определение, скопированные представления не придется восстанавливать. Оба представления будут обслуживаться из одного и того же индекса на диске.

#### 4.6.5.1. Формат запроса

Параметры запроса отображает Таблица 511.

**Таблица 511 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc} |
| Метод | COPY |

Параметры HTTP заголовка запроса отображает Таблица 512.

**Таблица 512 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Destination | Документ назначения. Должен содержать идентификатор целевого документа и, необязательно, версию целевого документа, если копируется в существующий документ | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 325.

```
COPY /test_people/_design/PeopleInfo  HTTP/1.1
Accept: application/json
Destination: “_design/PeopleInfoCopied”
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 325**

##### 4.6.5.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 513.

**Таблица 513 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 326. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: "1-09fc8f33e21a187166d2c3ad15c0df55"
Location: http://51.250.31.135:5984/test_people/_design%2FPeopleInfoCopied
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfoCopied",
    "rev": "1-09fc8f33e21a187166d2c3ad15c0df55"
}
```

**Ответ**
**Рисунок 326**

### 4.6.6 Получить HTTP-заголовки с информацией о вложении

Возвращает HTTP-заголовки, содержащие минимальное количество информации об указанном вложении.

#### 4.6.6.1. Формат запроса

Параметры запроса отображает Таблица 514.

**Таблица 514 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/{attname} |
| Метод | HEAD |

Параметры HTTP заголовка запроса отображает Таблица 515.

**Таблица 515 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 327.

```
HEAD /test_people/_design/PeopleInfo/AdditionalInformation.txt HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 327**

#### 4.6.6.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 516.

**Таблица 516 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 328. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 39
Content-Type: text/plain
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: " hFwJIamdJ3MRnYy+i6AQSA=="
Location: http://51.250.31.135:5984/test_people/_design%2FPeopleInfo/AdditionalInformation.txt
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfo",
    "rev": "2-2d445dca120462447ce034cd1f9253f7"
}
```

**Ответ**
**Рисунок 328**

### 4.6.7 Получить вложение файла

Возвращает вложение файла, связанное с проектным документом. Возвращаются исходные данные связанного вложения (так же, как если бы обращение осуществлялось к статическому файлу).

#### 4.6.7.1. Формат запроса

Параметры запроса отображает Таблица 517.

**Таблица 517 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/{attname} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 518.

**Таблица 518 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 329.

```
GET /test_people/_design/PeopleInfo/AdditionalInformation.txt HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 329**

#### 4.6.7.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 519.

**Таблица 519 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 330. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 39
Content-Type: text/plain
Date: Tue, 13 Aug 2013 21:35:37 GMT
ETag: " hFwJIamdJ3MRnYy+i6AQSA=="
Server: Yenisei/1.0.0 (Erlang OTP/24)

Additional information
```

**Ответ**
**Рисунок 330**

### 4.6.8 Загрузить содержимое в качестве вложения

Загружает предоставленное содержимое в качестве вложения в указанный проектный документ. Указанное имя вложения должно быть строкой в кодировке URL.

#### 4.6.8.1. Формат запроса

Параметры запроса отображает Таблица 520.

**Таблица 520 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/{attname} |
| Метод | PUT |

Параметры HTTP заголовка запроса отображает Таблица 521.

**Таблица 521 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 331.

```
PUT /test_people/_design/PeopleInfo/AdditionalInformation.txt?rev=1-09fc8f33e21a187166d2c3ad15c0df55
HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 331**

#### 4.6.8.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 522.

**Таблица 522 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 332. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
Location: http://51.250.31.135:5984/test_people/_design%2FPeopleInfo/AdditionalInformation.txt
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfo",
    "rev": "2-2d445dca120462447ce034cd1f9253f7"
}
```

**Ответ**
**Рисунок 332**

### 4.6.9 Удалить вложение

Удаляет вложение указанного проектного документа.

#### 4.6.9.1. Формат запроса

Параметры запроса отображает Таблица 523.

**Таблица 523 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/{attname} |
| Метод | DELETE |

Параметры HTTP заголовка запроса отображает Таблица 524.

**Таблица 524 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 333.

```
DELETE /test_people/_design/PeopleInfo/AdditionalInformation.txt?rev=2-
2d445dca120462447ce034cd1f9253f7 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 333**

#### 4.6.9.1.1.1. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 525.

**Таблица 525 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 334. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 232
Content-Type: application/json
Date: Tue, 13 Aug 2013 21:35:37 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)

{
    "ok": true,
    "id": "_design/PeopleInfo",
    "rev": " 3-0804524d36acec5ad83017efbcb9ea97"
}
```

**Ответ**
**Рисунок 334**

### 4.6.10 Получить информацию о проектном документе

Получает информацию об указанном проектном документе, включая индекс, размер индекса и текущее состояние проектного документа, и связанную с ним информацию об индексе.

#### 4.6.10.1. Формат запроса

Параметры запроса отображает Таблица 526.

**Таблица 526 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_info |
| Метод | GET |

Параметры строки запроса отображает Таблица 527.

**Таблица 527 – Параметры строки запроса**

| Поле | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя конструкторского документа |

Параметры HTTP заголовка запроса отображает Таблица 528.

**Таблица 528 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 335.

```
GET /recipes/_design/recipe/_info HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 335**

В качестве альтернативы вместо параметра запроса rev можно использовать заголовок If-Match.

Пример запроса отображает Рисунок 336.

```
DELETE /recipes/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
If-Match: 6-440b2dd39c20413045748b42c6aba6e2
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 336**

### 4.6.10.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 529.

**Таблица 529 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 530.

**Таблица 530 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| name | Имя документа дизайна | Строка (String) | Да |
| view_index | Информация об индексе представления | Объект (Json Object) | Да |

Коды состояния отображает Таблица 531.

**Таблица 531 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ от GET /{db}/_design/{ddoc}/_info содержит поле view_index (object) со структурой, которую отображает подпункт 5.3.2.1.

Ответ в случае успешного завершения запроса отображает Рисунок 337. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 263
Content-Type: application/json
Date: Sat, 17 Aug 2013 12:54:17 GMT
Server: БД «Енисей» (Erlang/OTP)
{
    "name": "recipe",
    "view_index": {
        "compact_running": false,
        "language": "python",
        "purge_seq": 0,
        "signature": "a59a1bb13fdf8a8a584bc477919c97ac",
        "sizes": {
          "active": 926691,
          "disk": 1982704,
          "external": 1535701
        },
        "update_seq": 12397,
        "updater_running": false,
        "waiting_clients": 0,
        "waiting_commit": false
    }
}
```

**Ответ**
**Рисунок 337**

### 4.6.11 Выполнить функцию представления

Выполняет указанную функцию представления из указанного документа проектирования.

#### 4.6.11.1. Формат запроса

Параметры запроса отображает Таблица 532.

**Таблица 532 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_view/{view} |
| Метод | GET |

Параметры строки запроса отображает Таблица 533.

**Таблица 533 – Параметры строки запроса**

| Поле | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя конструкторского документа |
| view | Имя функции представления |

Параметры HTTP заголовка запроса отображает Таблица 534.

**Таблица 534 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | **Да** |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 535.

**Таблица 535 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| conflicts | Включить информацию о конфликтах в ответ. Игнорируется, если include_docs не является true. По умолчанию false. | Логический тип (Boolean) | Нет |
| descending | Возвращать документы в порядке убывания по ключу. По умолчанию false. | Логический тип (Boolean) | Нет |
| endkey | Остановить возврат записей при достижении указанного ключа. | Объект (Json Object) | Нет |
| end_key | Псевдоним для параметра endkey. | Объект (Json Object) | Нет |
| endkey_docid | Остановить возврат записей при достижении указанного идентификатора документа. Игнорируется, если endkey не задан. | Строка (String) | Нет |
| end_key_doc_id | Псевдоним для endkey_docid. | Строка (String) | Нет |
| group | Сгруппировать результаты с помощью функции reduce в группу или одну строку. Подразумевает, что reduce - true и максимальный уровень group_level. По умолчанию false. | Логический тип (Boolean) | Нет |
| group_level | Укажите уровень группы, который будет использоваться. Подразумевается, что group - true. | Число (int64) | Нет |
| include_docs | Включить связанный документ в каждую строку. По умолчанию false. | Логический тип (Boolean) | Нет |
| attachments | Включить содержимое вложений в Base64-кодировке в документы, которые включаются, если include_docs равно true. Игнорируется, если значение include_docs не true. По умолчанию false. | Логический тип (Boolean) | Нет |
| att_encoding_info | Включать информацию о кодировке в заглушки вложений, если include_docs true и конкретное вложение сжато. Игнорируется, если include_docs не true. По умолчанию false. | Логический тип (Boolean) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| inclusive_end | Указывает, должен ли указанный конечный ключ быть включен в результат. По умолчанию true. | Логический тип (Boolean) | Нет |
| key | Возвращает только документы, соответствующие указанному ключу. | Объект (Json Object) | Нет |
| keys | Возвращает только документы, в которых ключ совпадает с одним из ключей, указанных в массиве. | json-массив (Array of json) | Нет |
| limit | Ограничить количество возвращаемых документов указанным числом. | Число (int64) | Нет |
| reduce | Использовать функцию сокращения. По умолчанию true, если определена функция reduce. | Логический тип (Boolean) | Нет |
| skip | Пропустить указанное количество записей перед началом возврата результатов. По умолчанию равно 0. | Число (int64) | Нет |
| sorted | Сортировать возвращаемые строки | Логический тип (Boolean) | Нет |
| stable | Должны ли результаты представления возвращаться из стабильного набора фргментов. По умолчанию false. | Логический тип (Boolean) | Нет |
| stale | Разрешить использование результатов из устаревшего представления. Поддерживаемые значения: ok и update_after. ok эквивалентно stable=true&update=false. update_after эквивалентно stable=true&update=lazy. Поведение по умолчанию эквивалентно stable=false&update=true. Следует обратить внимание, что этот параметр устарел. Вместо него используйте stable и update. | Строка (String) | Нет |
| startkey | Возвращает записи, начинающиеся с указанного ключа. | Объект (Json Object) | Нет |
| start_key | Псевдоним для startkey. | Объект (Json Object) | Нет |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| startkey_docid | Возвращает записи, начинающиеся с указанного идентификатора документа. Игнорируется, если startkey не задан. | Строка (String) | Нет |
| start_key_doc_id | Псевдоним для параметра startkey_docid. | Строка (String) | Нет |
| update | Следует ли обновлять рассматриваемое представление перед ответом пользователю. Поддерживаемые значения: true, false, lazy. По умолчанию true. | Строка (String) | Нет |
| update_seq | Включать ли в ответ значение update_seq, указывающее на идентификатор последовательности базы данных, которую отражает представление. По умолчанию false. | Логический тип (Boolean) | Нет |

Пример запроса отображает Рисунок 338.

```
GET /recipes/_design/ingredients/_view/by_name HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 338**

### 4.6.11.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 536.

**Таблица 536 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json, text/plain; charset=utf-8 | Строка (String) | |
| ETag | Подпись ответа | Строка (String) | |
| Transfer-Encoding | chunked | | |

Объекты JSON ответа отображает Таблица 537.

**Таблица 537 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Смещение начала списка документов. | Число (int64) | Да |
| rows | Массив объектов строк представления. По умолчанию возвращаемая информация содержит только ID документа и его ревизию. | Массив (Array) | Да |
| total_rows | Количество документов в базе данных/представлении. | Число (int64) | Да |
| update_seq | Текущая последовательность обновления для базы данных. | Объект (Json Object) | Нет |

Коды состояния отображает Таблица 538.

**Таблица 538 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Указанная база данных, проектный документ или представление не найдены |

Ответ в случае успешного завершения запроса отображает Рисунок 339. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:12:06 GMT
ETag: «2FOLSBSW4O6WB798XU4AQYA9B»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "SpaghettiWithMeatballs",
            "key": "meatballs",
            "value": 1
        },
        {
            "id": "SpaghettiWithMeatballs",
            "key": "spaghetti",
            "value": 1
        },
        {
            "id": "SpaghettiWithMeatballs",
            "key": "tomato sauce",
            "value": 1
        }
    ],
    "total_rows": 3
}
```

**Ответ**

**Рисунок 339**

Использование параметра attachments для включения вложений в результаты просмотра не рекомендуется при больших размерах вложений. Также Следует обратить внимание, что используемое Base64-кодирование приводит к 33%-ному превышению (т.е. одной трети) размера передачи для вложений.

### 4.6.12 Выполнить функцию представления с параметрами

Выполняет указанную функцию представления из указанного документа проектирования. Функциональность представления POST поддерживает идентичные параметры и поведение, указанные в GET /{db}/_design/{ddoc}/_view/{view} ППИ, но позволяет передавать параметры строки запроса как ключи в JSON-объекте в теле POST-запроса.

#### 4.6.12.1. Формат запроса

Параметры запроса отображает Таблица 539.

**Таблица 539 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_view/{view} |
| Метод | POST |

Параметры HTTP заголовка запроса отображает Таблица 540.

**Таблица 540 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 340.

```
POST /recipes/_design/ingredients/_view/by_name HTTP/1.1
Accept: application/json
Content-Length: 37
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "keys": [
        "meatballs",
        "spaghetti"
    ]
}
```

**Запрос**
**Рисунок 340**

### 4.6.12.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 541.

**Таблица 541 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |

Коды состояния отображает Таблица 542.

**Таблица 542 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 341. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:14:13 GMT
ETag: «6R5NM8E872JIJF796VF7WI3FZ»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "SpaghettiWithMeatballs",
            "key": "meatballs",
            "value": 1
        },
        {
            "id": "SpaghettiWithMeatballs",
            "key": "spaghetti",
            "value": 1
        }
    ],
    "total_rows": 3
}
```

**Ответ**
**Рисунок 341**

### 4.6.12.3. Опции представления

Существуют две опции индексирования представления, которые могут быть определены в проектном документе как булевы свойства объекта опций. В отличие от других опций запроса, эти параметры не являются параметрами URL, потому что они вступают в силу, когда создается индекс представления, а не когда к нему обращаются. Параметры представления отображает Таблица 543.

**Таблица 543 – Параметры представления**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| local_seq | Делает локальные порядковые номера документов доступными для функций map (как свойство документа _local_seq). | Логический тип (Boolean) | Нет |
| include_design | Позволяет вызывать функции map для проектных документов, а также для обычных документов. | Логический тип (Boolean) | Нет |

### 4.6.12.4. Запрос представлений и индексов

Определение представления в документе проектирования также создает индекс, основанный на ключевой информации, определенной в каждом представлении. Создание и использование индекса значительно увеличивает скорость доступа и поиска или выбора документов из представления.

Однако индекс не обновляется при добавлении или изменении новых документов в базе данных. Вместо этого индекс создается или обновляется либо при первом обращении к представлению, либо при обращении к представлению после обновления документа. В каждом случае индекс обновляется до выполнения запроса представления к базе данных.

Индексы представления обновляются инкрементально в следующих ситуациях:

1) В базу данных добавлен новый документ.

2) Документ был удален из базы данных.

3) Документ в базе данных был обновлен.

Индексы представлений полностью перестраиваются при изменении определения представления. Для этого при обновлении проектного документа создается «отпечаток» определения представления. Если отпечаток изменяется, то индексы представления полностью перестраиваются. Это гарантирует, что изменения в определениях представлений отражаются в индексах представлений.

Перестройка индексов представлений происходит, когда одно представление из одной группы представлений (т.е. все представления, определенные в одном проектном документе) было определено как нуждающееся в перестройке. Например, если имеется проектный документ с различными представлениями, и производится обновление базы данных, все три индекса представления в проектном документе будут обновлены.

Поскольку представление обновляется после того, как к нему был сделан запрос, это может привести к задержке возврата информации при обращении к представлению, особенно если в базе данных имеется большое количество документов, а индекс представления не существует. Существует ряд способов смягчить, но не полностью устранить эти проблемы. К ним относятся:

1) Следует создавать определение представления (и связанные с ним проектные документы) в базе данных до того, как разрешите вставку или обновление документов. Если это разрешить, пока к представлению осуществляется доступ, индекс можно обновлять постепенно.
2) Ручной принудительный запрос представления из базы данных. Можно сделать это либо до того, как пользователям будет разрешено использовать представление, либо получить доступ к представлению вручную после добавления или обновления документов.
3) Использовать ленту изменений для отслеживания изменений в базе данных, а затем получить доступ к представлению, чтобы заставить соответствующий индекс представления быть обновленным.

Ни один из этих способов не может полностью устранить необходимость в перестройке или обновлении индексов при обращении к представлению, но они могут уменьшить влияние обновления индекса на конечных пользователей.

Другой альтернативой является предоставление пользователям доступа к устаревшей версии индекса представления, вместо того чтобы заставлять индекс обновляться и отображать обновленные результаты. Использование устаревшего представления может не вернуть последнюю информацию, но вернет результаты запроса представления с использованием существующей версии индекса.

#### 4.6.12.4.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 544.

**Таблица 544 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Например, чтобы получить доступ к существующему устаревшему представлению by_recipe в документе проектирования recipes, что отображает Рисунок 342.

```
GET /test_people/_design/PeopleInfo/_view/ListView?stale=ok&limit=3 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 342**

#### 4.6.12.4.2. Формат ответа на запрос

Доступ к устаревшему представлению:

1) Не вызывает перестройки индексов представления, даже если с момента последнего доступа произошли изменения.
2) Возвращает текущую версию индекса представления, если такая версия существует.
3) Возвращает пустой набор результатов, если данный индекс представления не существует.

В качестве альтернативы можно использовать значение update_after для параметра stale. Это приводит к тому, что представление возвращается как stale view, но процесс обновления запускается после того, как информация о представлении была возвращена клиенту.

В дополнение к использованию устаревших представлений, Можно также использовать аргумент запроса update_seq. При использовании этого аргумента запроса генерируется информация о представлении, включая последовательность обновления базы данных, из которой было сгенерировано представление. Возвращенное значение можно сравнить с текущей последовательностью обновления, отображенной в информации базы данных (возвращается по GET /{db}).

Параметры HTTP заголовка ответа отображает Таблица 545.

**Таблица 545 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 343. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:14:13 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "total_rows": 15,
    "offset": 0,
    "rows": [
        {
            "id": "Andreev_I",
            "key": "Andreev_I",
            "value": {
                "Фамилия": "Андреев",
                "Имя": "Илья",
                "Возраст": 27
            }
        },
        {
            "id": "b14de8379df4c9d00f919035170219b4222",
            "key": "b14de8379df4c9d00f919035170219b4222",
            "value": {}
        },
        {
            "id": "Frolova_I",
            "key": "Frolova_I",
            "value": {
                "Фамилия": "Фролова",
                "Имя": "Инна",
                "Возраст": 32
            }
        }
    ]
}
```

**Запрос**

**Рисунок 343**

#### 4.6.12.5. Сортировка возвращаемых строк

Каждый элемент в возвращаемом массиве сортируется с использованием собственной сортировки UTF-8 в соответствии с содержимым ключевой части выдаваемого содержимого. Основной порядок вывода следующий:

1) null
2) false
3) true
4) Числа
5) Текст (с учетом регистра, сначала в нижнем регистре)
6) Массивы (в соответствии со значениями каждого элемента, по порядку)
7) Объекты (в соответствии со значениями ключей, в порядке следования)

##### 4.6.12.5.1. Формат запроса

Параметры HTTP заголовка запроса отображает Таблица 546.

**Таблица 546 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 344.

```
GET /db/_design/test/_view/sorting HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 344**

Можно изменить порядок возвращаемой информации о представлении, используя значение запроса descending, установленное в true:

Пример запроса отображает Рисунок 345.

```
GET /db/_design/test/_view/sorting?descending=true HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 345**

#### 4.6.12.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 547.

**Таблица 547 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 346. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 10:09:25 GMT
ETag: «8LA1LZPQ37B6R9U8BK9BGQH27»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "dummy-doc",
            "key": null,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": false,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": true,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 0,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 1,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 10,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 42,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "10",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "hello",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "Hello",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "\u043f\u0440\u0438\u0432\u0435\u0442",
            "value": null
        },
        {
            "id": "dummy-doc",
```

```
            "key": [],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                1,
                2,
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                2,
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": {},
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": {
                "foo": "bar"
            },
            "value": null
        }
    ],
    "total_rows": 17
}
```


**Ответ**
**Рисунок 346**

Ответ к запросу с измененным порядком возвращаемой информации о представлении отображает Рисунок 347.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 10:09:25 GMT
ETag: «Z4N468R15JBT98OM0AMNSR8U»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": 0,
    "rows": [
        {
            "id": "dummy-doc",
            "key": {
                "foo": "bar"
            },
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": {},
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                2,
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [
                1,
                2,
                3
            ],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": [],
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "\u043f\u0440\u0438\u0432\u0435\u0442",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "Hello",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": "hello",
            "value": null
        },
        {
```

```
            "id": "dummy-doc",
            "key": "10",
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 42,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 10,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 1,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": 0,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": true,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": false,
            "value": null
        },
        {
            "id": "dummy-doc",
            "key": null,
            "value": null
        }
    ],
    "total_rows": 17
}
```


**Ответ**
**Рисунок 347**

### 4.6.12.5.3. Порядок сортировки и startkey/endkey

Направление сортировки применяется до фильтрации, применяемой с помощью аргументов запроса startkey и endkey.

#### 4.6.12.5.3.1. Формат запроса

Пример запроса отображает Рисунок 348.

```
GET http://couchdb:5984/recipes/_design/recipes/_view/by_ingredient?startkey=%22carrots%22&endkey=%2
2egg%22 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
```

**Запрос**
**Рисунок 348**

Будет работать корректно при перечислении всех совпадающих записей между carrots и egg. Если порядок вывода изменить на противоположный с помощью аргумента запроса descending, запрос представления не вернет ни одной записи:

Пример запроса отображает Рисунок 349.

```
GET /recipes/_design/recipes/_view/by_ingredient?descending=true&startkey=%22carrots%22&endkey=%22eg
g%22 HTTP/1.1
Accept: application/json
Host: localhost:5984
{
    "total_rows" : 26453,
    "rows" : [],
    "offset" : 21882
}
```

**Запрос**
**Рисунок 349**

Результаты будут пустыми, поскольку записи в представлении инвертируются перед применением фильтра ключей, и поэтому endkey «egg» будет виден перед startkey «carrots», что приведет к пустому списку.

Вместо этого следует изменить значения параметров startkey и endkey, чтобы они соответствовали сортировке по убыванию, применяемой к ключам.

Пример запроса отображает Рисунок 350.

```
GET /recipes/_design/recipes/_view/by_ingredient?descending=true&startkey=%22egg%22&endkey=%22carrot
s%22 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 350**

### 4.6.12.5.4. Необработанная сортировка

По умолчанию БД «Енисей» использует драйвер ICU для сортировки результатов представления. Можно использовать двоичную корелляцию вместо этого для более быстрой сборки представлений, где корелляция Unicode не важна.

Для использования необработанной сортировки добавьте «options»:{«collation»: «raw»} в объект представления документа проектирования. После этого представления будут перегенерированы и для соответствующего представления будет применен новый порядок.

#### 4.6.12.6. Использование ограничений и пропуск строк

По умолчанию представления возвращают все результаты. Это хорошо, когда количество результатов невелико, но это может привести к проблемам, когда результатов миллиарды, так как клиенту придется прочитать их все и занять всю доступную память.

Но можно сократить количество выводимых строк результатов, указав параметр запроса limit. Например, получение списка рецептов с использованием представления by_title и ограничением до 5 возвращает только 5 записей, в то время как всего в представлении 2667 записей.

##### 4.6.12.6.1. Формат запроса

Пример запроса отображает Рисунок 351.

```
GET /recipes/_design/recipes/_view/by_title?limit=5 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 351**

Чтобы опустить некоторые записи, можно использовать параметр запроса skip.

Пример запроса отображает Рисунок 352.

```
GET /recipes/_design/recipes/_view/by_title?limit=3&skip=2 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 352**

##### 4.6.12.6.2. Формат ответа на запрос

Ответ в случае успешного завершения запроса отображает Рисунок 353. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:14:13 GMT
ETag: «9Q6Q2GZKPH8D5F8L7PB6DBSS9»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset" : 0,
    "rows" : [
        {
            "id" : "3-tiersalmonspinachandavocadoterrine",
            "key" : "3-tier salmon, spinach and avocado terrine",
            "value" : [
                null,
                "3-tier salmon, spinach and avocado terrine"
            ]
        },
        {
            "id" : "Aberffrawcake",
            "key" : "Aberffraw cake",
            "value" : [
                null,
                "Aberffraw cake"
            ]
        },
        {
            "id" : "Adukiandorangecasserole-microwave",
            "key" : "Aduki and orange casserole - microwave",
            "value" : [
                null,
                "Aduki and orange casserole - microwave"
            ]
        },
        {
            "id" : "Aioli-garlicmayonnaise",
            "key" : "Aioli - garlic mayonnaise",
            "value" : [
                null,
                "Aioli - garlic mayonnaise"
            ]
        },
        {
            "id" : "Alabamapeanutchicken",
            "key" : "Alabama peanut chicken",
            "value" : [
                null,
                "Alabama peanut chicken"
            ]
        }
    ],
    "total_rows" : 2667
}
```

**Ответ**
**Рисунок 353**

Ответ на запрос с параметром skip отображает Рисунок 354.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:14:13 GMT
ETag: «H3G7YZSNIVRRHO5FXPE16NJHN»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset" : 2,
    "rows" : [
        {
            "id" : "Adukiandorangecasserole-microwave",
            "key" : "Aduki and orange casserole - microwave",
            "value" : [
                null,
                "Aduki and orange casserole - microwave"
            ]
        },
        {
            "id" : "Aioli-garlicmayonnaise",
            "key" : "Aioli - garlic mayonnaise",
            "value" : [
                null,
                "Aioli - garlic mayonnaise"
            ]
        },
        {
            "id" : "Alabamapeanutchicken",
            "key" : "Alabama peanut chicken",
            "value" : [
                null,
                "Alabama peanut chicken"
            ]
        }
    ],
    "total_rows" : 2667
}
```

**Ответ**

**Рисунок 354**

Использование параметров limit и skip не рекомендуется для пагинации результатов. Прочитайте о пагинации, почему это так и как сделать ее лучше.

### 4.6.12.7. Отправка нескольких запросов в представление

Выполняет несколько указанных запросов к представлению против функции представления из указанного документа дизайна.

#### 4.6.12.7.1. Формат запроса

Параметры запроса отображает Таблица 548.

**Таблица 548 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_view/{view}/queries¶ |
| Метод | POST |

Параметры строки запроса отображает Таблица 549.

**Таблица 549 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| view | Имя функции представления |

Параметры HTTP заголовка запроса отображает Таблица 550.

**Таблица 550 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Объекты JSON запроса отображает Таблица 551.

**Таблица 551 – Объекты JSON запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| queries | Массив объектов запросов с полями для параметров каждого отдельного выполняемого запроса представления. Имена полей и их значение аналогичны параметрам обычного запроса представления | Массив (Array) | Да |

Пример запроса отображает Рисунок 355.

```
POST /recipes/_design/recipes/_view/by_title/queries HTTP/1.1
Content-Type: application/json
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "queries": [
        {
            "keys": [
                "meatballs",
                "spaghetti"
            ]
        },
        {
            "limit": 3,
            "skip": 2
        }
    ]
}
```

**Запрос**
**Рисунок 355**

#### 4.6.12.7.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 552.

**Таблица 552 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 553.

**Таблица 553 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| results | Массив объектов результатов - по одному для каждого запроса. Каждый объект результата содержит те же поля, что и ответ на обычный запрос представления <ППИ/ddoc/view>. | Массив (Array) | Да |

Коды состояния отображает Таблица 554.

**Таблица 554 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Указанная база данных, документ проектирования или представление отсутствует |
| 500 Internal Server Error | Ошибка выполнения функции представления |

Ответ в случае успешного завершения запроса отображает Рисунок 356. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 20 Dec 2016 11:17:07 GMT
ETag: «1H8RGBCK3ABY6ACDM7ZSC30QK»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "results" : [
        {
            "offset": 0,
            "rows": [
                {
                    "id": "SpaghettiWithMeatballs",
                    "key": "meatballs",
                    "value": 1
                },
                {
                    "id": "SpaghettiWithMeatballs",
                    "key": "spaghetti",
                    "value": 1
                },
                {
                    "id": "SpaghettiWithMeatballs",
                    "key": "tomato sauce",
                    "value": 1
                }
            ],
            "total_rows": 3
        },
        {
            "offset" : 2,
            "rows" : [
                {
                    "id" : "Adukiandorangecasserole-microwave",
                    "key" : "Aduki and orange casserole - microwave",
                    "value" : [
                        null,
                        "Aduki and orange casserole - microwave"
                    ]
                },
                {
                    "id" : "Aioli-garlicmayonnaise",
                    "key" : "Aioli - garlic mayonnaise",
                    "value" : [
                        null,
                        "Aioli - garlic mayonnaise"
                    ]
                },
                {
                    "id" : "Alabamapeanutchicken",
                    "key" : "Alabama peanut chicken",
                    "value" : [
                        null,
                        "Alabama peanut chicken"
                    ]
                }
            ],
            "total_rows" : 2667
        }
    ]
}
```

**Ответ**

**Рисунок 356**

### 4.6.13 Выполнить поиск по индексу

Выполняет запрос на поиск по названному индексу в указанном проектном документе.

Для конечных точек поиска требуется запущенный поисковый плагин, подключенный к каждому узлу кластера.

#### 4.6.13.1. Формат запроса

Параметры запроса отображает Таблица 555.

**Таблица 555 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_search/{index} |
| Метод | GET |

Параметры строки запроса отображает Таблица 556.

**Таблица 556 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| view | Имя функции представления |

Параметры HTTP заголовка запроса отображает Таблица 557.

**Таблица 557 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 558.

**Таблица 558 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| bookmark | Закладка, полученная в результате предыдущего поиска. Этот параметр позволяет листать результаты. Если после закладки больше нет результатов, можно получить ответ с пустым массивом строк и такой же закладкой, подтверждающей конец списка результатов. | Строка (String) | |
| counts | Массив имен строковых полей, для которых запрашиваются подсчеты. Ответ содержит подсчеты для каждого уникального значения этого поля среди документов, соответствующих поисковому запросу. Для работы этого параметра необходимо включить фасетный поиск. | Объект (Json Object) | |
| drilldown | Это поле может быть использовано несколько раз. Каждое использование определяет пару с именем поля и значением. Поиск соответствует только документам, содержащим значение, указанное в названном поле. Это отличается от использования «имя поля:значение» в параметре q только тем, что значения не анализируются. Для работы этого параметра фасетный поиск должен быть включен. | Объект (Json Object) | |
| group_field | Поле, по которому группируются совпадения поиска. | Строка (String) | |
| group_limit | Максимальное количество групп. Это поле можно использовать только в том случае, если указано поле group_field. | Число (int64) | |
| group_sort | Это поле определяет порядок групп в поиске, использующем поле group_field. По умолчанию порядок сортировки - релевантность. | Объект (Json Object) | |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| highlight_fields | Указывает, какие поля следует выделить. Если указано, объект результата содержит поле highlights с записью для каждого указанного поля. | Объект (Json Object) | |
| highlight_pre_tag | Строка, которая вставляется перед выделенным словом при выводе результатов выделения. | Строка (String) | |
| highlight_post_tag | Строка, которая вставляется после выделенного слова при выводе результатов выделения. | Строка (String) | |
| highlight_number | Число сегментов, которые возвращаются при выделении. Если поисковый термин встречается реже, чем заданное количество сегментов, возвращаются более длинные сегменты. | Число (int64) | |
| highlight_size | Количество символов в каждом сегменте для выделения. | Число (int64) | |
| include_docs | Включить полное содержание документов в ответ. | Логический тип (Boolean) | |
| include_fields | JSON-массив имен полей для включения в результаты поиска. Все включаемые поля должны быть проиндексированы с помощью опции store:true. | Объект (Json Object) | |
| limit | Ограничить количество возвращаемых документов указанным числом. Для поиска по группам этот параметр ограничивает количество документов в каждой группе. | Число (int64) | |
| q | Псевдоним для запроса. | Строка (String) | |
| query | Строка запроса Lucene. | Строка (String) | Да |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| ranges | Это поле определяет диапазоны для числовых полей поиска. Значение представляет собой объект JSON, в котором имена полей являются числовыми полями поиска, а значения полей - объектами JSON. Имена полей JSON-объектов - это имена диапазонов. Значения - это строки, описывающие диапазон, например «[0 - 10]». | Объект (Json Object) | |
| sort | Определяет порядок сортировки результатов. В сгруппированном поиске) - этот параметр задает порядок сортировки внутри группы. По умолчанию используется порядок сортировки по релевантности. JSON-строка вида «fieldname<type>» или -fieldname<type> для нисходящего порядка, где fieldname - имя строкового или числового поля, а type - число, строка или JSON-массив строк. Часть type является необязательной и по умолчанию имеет значение number. Примеры: «foo», «-foo», «bar<string>», «-foo<number>» и [«-foo<number>», «bar<string>»]. Строковые поля, используемые для сортировки, не должны быть анализируемыми полями. Поля, используемые для сортировки, должны быть проиндексированы тем же индексатором, который используется для поискового запроса. | Объект (Json Object) | |
| stale | Установите значение ok, чтобы разрешить использование устаревшего индекса. | Строка (String) | |

#### 4.6.13.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 559.

**Таблица 559 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | |
| ETag | Подпись ответа | Строка (String) | |
| Transfer-Encoding | chunked | Строка (String) | |

Объекты JSON ответа отображает Таблица 560.

**Таблица 560 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| rows | Массив объектов строк представления. По умолчанию возвращаемая информация содержит только ID документа и ревизию. | Массив (Array) | |
| total_rows | Количество документов в базе данных/представлении. | Число (int64) | |
| bookmark | Непрозрачный идентификатор для включения постраничной навигации. | Строка (String) | |

Коды состояния отображает Таблица 561.

**Таблица 561 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Неверный запрос |
| 401 Unauthorized | Требуется разрешение на чтение |
| 404 Not Found | Указанная база данных, проектный документ или представление не найдены. |

Перед использованием параметров counts, drilldown и ranges необходимо включить фасетный поиск.

Фасетный поиск и группировка не поддерживаются в разделенном поиске, поэтому следующие параметры запроса не должны использоваться в таких запросах: counts, drilldown, ranges и group_field, group_limit, group_sort``.

Не комбинируйте параметры bookmark и stale. Эти опции ограничивают выбор реплик сегментов, которые будут использоваться для ответа. При их совместном использовании могут возникнуть проблемы при попытке связаться с репликами, которые работают медленно или недоступны.

### 4.6.14 Выполнить поиск по имени индекса

Для конечных точек поиска требуется запущенный плагин поиска, подключенный к каждому узлу кластера.

#### 4.6.14.1. Формат запроса

Параметры запроса отображает Таблица 562.

**Таблица 562 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_search_info/{index}¶ |
| Метод | GET |

Параметры строки запроса отображает Таблица 555.

**Таблица 563 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| view | Имя функции представления |

Пример запроса отображает Рисунок 357.

```
GET /recipes/_design/cookbook/_search_info/ingredients HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 357**

#### 4.6.14.2. Формат ответа на запрос

Коды состояния отображает Таблица 564.

**Таблица 564 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 400 Bad Request | Тело запроса неверно (неправильно сформировано или отсутствует одно из обязательных полей) |
| 500 Internal Server Error | Произошла ошибка сервера (или ошибка другого рода) |

Ответ в случае успешного завершения запроса отображает Рисунок 358. Коды ошибок приведены в пункте 6.1.2.


```
{
    "name": "_design/cookbook/ingredients",
    "search_index": {
        "pending_seq": 7125496,
        "doc_del_count": 129180,
        "doc_count": 1066173,
        "disk_size": 728305827,
        "committed_seq": 7125496
    }
}
```


**Ответ**
**Рисунок 358**

### 4.6.15 Применить функцию show

Применяет функцию show для документа null.
Параметры запроса и ответа зависят от реализации функции.

#### 4.6.15.1. Формат запроса

Параметры запроса отображает Таблица 565.

**Таблица 565 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_show/{func}¶ |
| Метод | GET, POST |

Параметры строки запроса отображает Таблица 566.

**Таблица 566 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| view | Имя функции представления |

Параметры HTTP заголовка запроса отображает Таблица 567.

**Таблица 567 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 568.

**Таблица 568 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| format | Формат возвращаемого ответа. Используется функцией provides() | Строка (String) | Нет |

Функцию отображает Рисунок 359.

```
function(doc, req) {
    if (!doc) {
        return {body: «no doc»}
    } else {
        return {body: doc.description}
    }
}
```

**Функция**
**Рисунок 359**

Пример запроса отображает Рисунок 360.

```
GET /recipes/_design/recipe/_show/description HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 360**

### 4.6.15.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 569.

**Таблица 569 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | text/html; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |

Коды состояния отображает Таблица 570.

**Таблица 570 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 500 Internal Server Error | Ошибка сервера запроса |

Ответ в случае успешного завершения запроса отображает Рисунок 361. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Length: 6
Content-Type: text/html; charset=utf-8
Date: Wed, 21 Aug 2013 12:34:07 GMT
Etag: «7Z2TO7FPEMZ0F4GH0RJCRIOAU»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Vary: Accept
no doc
```

**Ответ**
**Рисунок 361**

### 4.6.16 Применить функцию show для указанного документа

Применяет функцию show для указанного документа.

Параметры запроса и ответа зависят от реализации функции.

#### 4.6.16.1. Формат запроса

Параметры запроса отображает Таблица 571.

**Таблица 571 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_show/{func}/{docid}¶ |
| Метод | GET, POST |

Параметры строки запроса отображает Таблица 572.

**Таблица 572 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| func | Имя функции show |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 573.

**Таблица 573 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| ETag | Подпись ответа | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 574.

**Таблица 574 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| format | Формат возвращаемого ответа. Используется функцией provides() | Строка (String) | Нет |

Функцию отображает Рисунок 362.

```
function(doc, req) {
    if (!doc) {
        return {body: «no doc»}
    } else {
        return {body: doc.description}
    }
}
```

**Функция**
**Рисунок 362**

Пример запроса отображает Рисунок 363.

```
GET /recipes/_design/recipe/_show/description/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**

**Рисунок 363**

### 4.6.16.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 575.

**Таблица 575 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | text/html; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |

Коды состояния отображает Таблица 576.

**Таблица 576 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 500 Internal Server Error | Ошибка сервера запроса |

Ответ в случае успешного завершения запроса отображает Рисунок 364. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Length: 88
Content-Type: text/html; charset=utf-8
Date: Wed, 21 Aug 2013 12:38:08 GMT
Etag: «8IEBO8103EI98HDZL5Z4I1T0C»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Vary: Accept
An Italian-American dish that usually consists of spaghetti, tomato sauce and meatballs.
```

**Ответ**

**Рисунок 364**

### 4.6.17 Применить функцию list для функции view

### 4.6.17.1. Формат запроса

Параметры запроса отображает Таблица 577

**Таблица 577 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_show/{func}/{view}¶ |
| Метод | GET, POST |

Параметры строки запроса отображает Таблица 578.

**Таблица 578 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| func | Имя функции show |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 579.

**Таблица 579 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| ETag | Подпись ответа | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 580.

**Таблица 580 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| format | Формат возвращаемого ответа. Используется функцией provides() | Строка (String) | Нет |

Функцию отображает Рисунок 365.

```
function(head, req) {
    var row = getRow();
    if (!row){
        return ‘no ingredients’
    }
    send(row.key);
    while(row=getRow()){
        send(‘, ‘ + row.key);
    }
}
```

**Функция**
**Рисунок 365**

Пример запроса отображает Рисунок 366.

```
GET /recipes/_design/recipe/_list/ingredients/by_name HTTP/1.1
Accept: text/plain
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 366**

### 4.6.17.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 581.

**Таблица 581 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Коды состояния отображает Таблица 582.

**Таблица 582 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 500 Internal Server Error | Ошибка сервера запроса |

Ответ в случае успешного завершения запроса отображает Рисунок 367. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8
Date: Wed, 21 Aug 2013 12:49:15 GMT
Etag: «D52L2M1TKQYDD1Y8MEYJR8C84»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
Vary: Accept
meatballs, spaghetti, tomato sauce
```

**Ответ**
**Рисунок 367**

### 4.6.18 Применить функцию списка для функции представления

Применяет функцию списка для функции представления <viewfun> из другого проектного документа.

#### 4.6.18.1. Формат запроса

Параметры запроса отображает Таблица 583.

**Таблица 583 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_list/{func}/{other-ddoc}/{view} |
| Метод | GET, POST |

Параметры запроса и ответа зависят от реализации функции.

Параметры строки запроса отображает Таблица 584.

**Таблица 584 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| func | Имя функции show |
| other-ddoc | Имя другого проектного документа, содержащего функцию представления |
| view | Имя функции представления |

Параметры HTTP заголовка запроса отображает Таблица 585.

**Таблица 585 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 586.

**Таблица 586 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| format | Формат возвращаемого ответа. Используется функцией provides() | Строка (String) | Нет |

Функцию отображает Рисунок 368.

```
function(head, req) {
    var row = getRow();
    if (!row){
        return ‘no ingredients’
    }
    send(row.key);
    while(row=getRow()){
        send(‘, ‘ + row.key);
    }
}
```

**Функция**
**Рисунок 368**

Пример запроса отображает Рисунок 369.

```
GET /recipes/_design/ingredient/_list/ingredients/recipe/by_ingredient?key=»spaghetti» HTTP/1.1
Accept: text/plain
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 369**

### 4.6.18.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 587.

**Таблица 587 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | text/plain; charset=utf-8 | Строка (String) | Да |
| ETag | Подпись ответа | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Коды состояния отображает Таблица 588.

**Таблица 588 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |
| 500 Internal Server Error | Ошибка сервера запроса |

Ответ в случае успешного завершения запроса отображает Рисунок 370. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8
Date: Wed, 21 Aug 2013 12:49:15 GMT
Etag: «5L0975X493R0FB5Z3043POZHD»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
Vary: Accept
spaghetti
```

**Ответ**
**Рисунок 370**

### 4.6.19 Выполнить функцию обновления

#### 4.6.19.1. Формат запроса

Параметры запроса отображает Таблица 589.

**Таблица 589 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_update/{func} |
| Метод | POST |

Выполняет функцию обновления на стороне сервера для документа null.

Параметры строки запроса отображает Таблица 590.

**Таблица 590 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| func | Имя функции show |

Параметры HTTP заголовка запроса отображает Таблица 591.

**Таблица 591 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| X-Couch-Id | ID созданного/обновленного документа | Строка (String) | Нет |
| X-Couch-Update-Newrev | Ревизия созданного/обновленного документа | Строка (String) | Нет |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Функцию отображает Рисунок 371.

```
function(doc, req) {
    if (!doc){
      return [null, {‘code’: 400,
                     ‘json’: {‘error’: ‘missed’,
                              ‘reason’: ‘no document to update’}}]
    } else {
        doc.ingredients.push(req.body);
        return [doc, {‘json’: {‘status’: ‘ok’}}];
    }
}
```

**Функция**
**Рисунок 371**

Пример запроса отображает Рисунок 372.

```
POST /recipes/_design/recipe/_update/ingredients HTTP/1.1
Accept: application/json
Content-Length: 10
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984

"something"
```

**Запрос**
**Рисунок 372**

#### 4.6.19.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 592.

**Таблица 592 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Cache-Control | must-revalidate | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |
| Content-Type | text/plain; charset=utf-8 | Строка (String) | Да |

Коды состояния отображает Таблица 593.

**Таблица 593 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ не был создан или обновлен |
| 201 Created | Документ был создан или обновлен |
| 500 Internal Server Error | Ошибка сервера запросов |

Ответ в случае успешного завершения запроса отображает Рисунок 373. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 404 Object Not Found
Cache-Control: must-revalidate
Content-Length: 52
Content-Type: application/json
Date: Wed, 21 Aug 2013 14:00:58 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
    "error": "missed",
    "reason": "no document to update"
}
```

**Ответ**
**Рисунок 373**

### 4.6.20 Выполнить функцию обновления на стороне сервера

Выполняет функцию обновления на стороне сервера для указанного документа.

#### 4.6.20.1. Формат запроса

Параметры запроса отображает Таблица 594.

**Таблица 594 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_update/{func}/{docid} |
| Метод | PUT |

Параметры строки запроса отображает Таблица 595.

**Таблица 595 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя документа проектирования |
| func | Имя функции show |
| docid | Идентификатор документа |

Параметры HTTP заголовка запроса отображает Таблица 596.

**Таблица 596 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Функцию отображает Рисунок 374.

```
function(doc, req) {
    if (!doc){
        return [null, {‘code’: 400,
                       ‘json’: {‘error’: ‘missed’,
                                ‘reason’: ‘no document to update’}}]
    } else {
        doc.ingredients.push(req.body);
        return [doc, {‘json’: {‘status’: ‘ok’}}];
    }
}
```

**Функция**
**Рисунок 374**

Пример запроса отображает Рисунок 375.

```
POST /recipes/_design/recipe/_update/ingredients/SpaghettiWithMeatballs HTTP/1.1
Accept: application/json
Content-Length: 5
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
«love»
```

**Запрос**
**Рисунок 375**

#### 4.6.20.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 597.

**Таблица 597 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Cache-Control | must-revalidate | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |
| Content-Type | text/plain; charset=utf-8 | Строка (String) | Да |
| X-Couch-Id | ID созданного/обновленного документа | Строка (String) | Да |
| X-Couch-Update-Newrev | Ревизия созданного/обновленного документа | Строка (String) | Да |

Коды состояния отображает Таблица 598.

**Таблица 598 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Документ не был создан или обновлен |
| 201 Created | Документ был создан или обновлен |
| 500 Internal Server Error | Ошибка сервера запросов |

Ответ в случае успешного завершения запроса отображает Рисунок 376. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 16
Content-Type: application/json
Date: Wed, 21 Aug 2013 14:11:34 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Id: SpaghettiWithMeatballs
X-Couch-Update-NewRev: 12-a5e099df5720988dae90c8b664496baf
{
    "status": "ok"
}
```

**Ответ**

**Рисунок 376**

### 4.6.21 Переписать указанный путь по правилам

Переписывает указанный путь по правилам, определенным в указанном проектном документе. Правила перезаписи определяются полем rewrites проектного документа. Поле rewrites может быть либо строкой, содержащей функцию перезаписи, либо массивом определений правил.

#### 4.6.21.1. Формат запроса

Параметры запроса отображает Таблица 599.

**Таблица 599 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_design/{ddoc}/_rewrite/{path} |
| Метод | ANY |

#### 4.6.21.2. Использование строковой функции для перезаписи

Когда поле rewrites представляет собой строковую функцию, сервер запросов используется для предварительной обработки и маршрутизации запросов.

Функция принимает объект Объект Request2.

Возвращаемое значение функции заставляет сервер переписать запрос на новое место или немедленно вернуть ответ.

Чтобы переписать запрос, возвращает объект, содержащий следующие свойства, что отображает Таблица 600.

**Таблица 600 – Объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| path | Переписанный путь. | Строка (String) | Да |
| query | Переписанный запрос. Если параметр опущен, используются ключи исходного запроса. | Массив (Array) | Нет |
| headers | Переписанные заголовки. Если параметр опущен, используются оригинальные заголовки запроса. | Объект (Json Object) | Нет |
| method | HTTP-метод переписанного запроса («GET», «POST» и т.д.). Если параметр опущен, используется исходный метод запроса. | Строка (String) | Нет |
| body | Тело запроса для запросов «POST» / «PUT». Если параметр опущен, используется тело исходного запроса. | Строка (String) | Нет |

Чтобы ответить на запрос, возвращает объект, содержащий следующие свойства, что отображает Таблица 601.

**Таблица 601 – Объект**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| code | Возвращаемый код статуса HTTP (200, 404 и т.д.). | Число (int64) | Да |
| body | Тело ответа пользователю. | Строка (String) | Да |

Пример А.

Функцию ограничения доступа отображает Рисунок 377.

```
function(req2) {
  var path = req2.path.slice(4),
    isWrite = /^(put|post|delete)$/i.test(req2.method),
    isFinance = req2.userCtx.roles.indexOf(«finance») > -1;
  if (path[0] == «finance» && isWrite && !isFinance) {
    // Deny writes to  DB «finance» for users
    // having no «finance» role
    return {
      code: 403,
      body: JSON.stringify({
        error: «forbidden».
        reason: «You are not allowed to modify docs in this DB»
      })
    };
  }
  // Pass through all other requests
  return { path: «../../../» + path.join(«/») };
}
```

**Пример А. Функция ограничения доступа**
**Рисунок 377**

Ответ в случае успешного завершения запроса отображает Рисунок 377. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 16
Content-Type: application/json
Date: Wed, 21 Aug 2013 14:11:34 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Id: SpaghettiWithMeatballs
X-Couch-Update-NewRev: 12-a5e099df5720988dae90c8b664496baf
{
    "status": "ok"
}
```

**Пример А. Ответ**
**Рисунок 378**

Пример В. Функцию различных ответов на запросы JSON и HTML отображает Рисунок 379.

```
function(req2) {
  var path = req2.path.slice(4),
    h = headers,
    wantsJson = (h.Accept || «»).indexOf(«application/json») > -1,
    reply = {};
  if (!wantsJson) {
    // Here we should prepare reply object
    // for plain HTML pages
  } else {
    // Pass through JSON requests
    reply.path = «../../../»+path.join(«/»);
  }
  return reply;
}
```

**Пример В. Различные ответы на запросы JSON и HTML**
**Рисунок 379**

#### 4.6.21.3. Использование массива правил для перезаписи

Если поле rewrites представляет собой массив объектов правил, сервер будет переписывать запрос на основе первого подходящего правила в этом массиве.

Каждое правило в массиве представляет собой объект со следующими полями, что отображает Таблица 602.

**Таблица 602 – Объект**

| Поле | Описание | Тип |
|---|---|---|
| method | Метод HTTP-запроса для привязки метода запроса к правилу. Если параметр опущен, используется «*», который соответствует всем методам. | Строка (String) |
| from | Шаблон, используемый для сравнения с URL и определения динамических переменных. | Строка (String) |
| to | Путь, по которому следует переписать URL. Он может содержать переменные, зависящие от переменных привязки, обнаруженных при сопоставлении с шаблоном, и аргументов запроса (URL args и из члена запроса). | Строка (String) |
| query | Аргументы запроса, передаваемые в перезаписанный URL. Они могут содержать динамические переменные | Объект (Json Object) |

Пути to и from могут содержать строковые шаблоны с ведущими символами : или * для определения динамических переменных в совпадении.

Первое правило в массиве rewrites, которое соответствует входящему запросу, используется для определения перезаписи. Чтобы соответствовать входящему запросу, метод правила должен соответствовать HTTP-методу запроса, а from правила должен соответствовать пути запроса, используя следующую логику сопоставления шаблонов.

Шаблон from и URL сначала разделяются на /, чтобы получить список лексем. Например, если поле from имеет вид /somepath/:var/*, а URL - /somepath/a/b/c, то маркерами будут somepath, :var и * для шаблона from и somepath, a, b и c для URL.

Каждая лексема, начинающаяся с : в шаблоне, будет соответствовать соответствующей лексеме в URL и определит новую динамическую переменную, именем которой будет оставшаяся строка после :, а значением - лексема из URL. В данном примере маркер :var будет соответствовать b и установит var = a.

Маркер * в шаблоне будет соответствовать любому количеству маркеров в URL и должен быть последним маркером в шаблоне. Он определит динамическую переменную с оставшимися маркерами. В данном примере маркер * будет соответствовать маркерам b и c и установит * = b/c.

Остальные лексемы должны совпасть в точности, чтобы шаблон считался совпавшим. В данном примере somepath в шаблоне совпадает с somepath в URL, и все маркеры в URL совпали, в результате чего это правило считается совпадением.

Как только правило найдено, URL запроса переписывается с использованием полей to и query. Динамические переменные подставляются в переменные : и * в этих полях, чтобы получить окончательный URL.

Если правило не найдено, возвращается ответ 404 Not Found.

Примеры отображает Таблица 603.

**Таблица 603 – Объект**

| Правило | URL | Переписать на | Знаки |
|---|---|---|---|
| {«from»: «/a», «to»: «/some»} | /a | /some | |
| {«from»: «/a/*», «to»: «/some/*} | /a/b/c | /some/b/c | |
| {«from»: «/a/b», «to»: «/some»} | /a/b?k=v | /some?k=v | k=v |
| {«from»: «/a/b», «to»: «/some/:var»} | /a/b | /some/b?var=b | var=b |
| {«from»: «/a/:foo/», «to»: «/some/:foo/»} | /a/b/c | /some/b/c?foo=b | foo=b |
| {«from»: «/a/:foo», «to»: «/some», «query»: { «k»: «:foo» }} | /a/b | /some/?k=b&foo=b | foo=b |
| {«from»: «/a», «to»: «/some/:foo»} | /a?foo=b | /some/?b&foo=b | foo=b |

Метод запроса, заголовок, параметры запроса, полезная нагрузка запроса и тело ответа зависят от конечной точки, к которой будет переписан URL.

Параметры строки запроса отображает Таблица 604.

**Таблица 604 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| ddoc | Имя проектного документа |
| path | Путь URL для перезаписи |

## 4.7. Базы данных с разделами

Базы данных с разделами позволяют размещать данные в кластере, что обеспечивает значительное повышение производительности запросов, ограниченных одним разделом.

### 4.7.1 Получить информацию, описывающую указанный раздел

Метод предназначен для получения информации, описывающей указанный раздел. Она включает в себя количество документов и удаленных документов, а также размеры внешних и активных данных.

#### 4.7.1.1. Формат запроса

Параметры запроса отображает Таблица 605.

**Таблица 605 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_partition/{partition} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 606.

**Таблица 606 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 380.

```
GET /db/_partition/sensor-260 HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 380**

#### 4.7.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 607.

**Таблица 607 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |

Коды состояния отображает Таблица 608.

**Таблица 608 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 381. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 119
Content-Type: application/json
Date: Thu, 24 Jan 2019 17:19:59 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
{
  "db_name": "my_new_db",
  "doc_count": 1,
  "doc_del_count": 0,
  "partition": "sensor-260",
  "sizes": {
    "active": 244,
    "external": 347
  }
}
```

**Ответ**

**Рисунок 381**

### 4.7.2 Установить границы предоставленного диапазона разделов

Эта конечная точка является удобной конечной точкой для автоматической установки границ предоставленного диапазона разделов. Аналогичных результатов можно добиться, используя глобальную конечную точку /db/_all_docs с соответствующим образом настроенными значениями для start_key и end_key.

#### 4.7.2.1. Формат запроса

Параметры запроса отображает Таблица 609.

**Таблица 609 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port] /{db}/_partition/{partition}/_all_docs |
| Метод | GET |

Параметры строки запроса отображает Таблица 610.

**Таблица 610 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| partition | Имя раздела |

Параметры HTTP заголовка запроса отображает Таблица 611.

**Таблица 611 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 382.

```
GET /db/_partition/sensor-260/_all_docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 382**

#### 4.7.2.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 612.

**Таблица 612 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 383. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 10 Aug 2013 16:22:56 GMT
ETag: «1W2DJUZFZSZD9K78UFA3GZWB4»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
  "offset": 0,
  "rows": [
    {
      "id": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "key": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "value": {
        "rev": "1-05ed6f7abf84250e213fcb847387f6f5"
      }
    }
  ],
  "total_rows": 1
}
```

**Ответ**
**Рисунок 383**

### 4.7.3 Выполнить секционированный запрос

Этот метод предназначен для выполнение секционированного запроса (partitioned query). Возвращаемый результат представления будет содержать только строки с указанным именем раздела.

#### 4.7.3.1. Формат запроса

Параметры запроса отображает Таблица 613.

**Таблица 613 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port] /{db}/_partition/{partition}/_design/{ddoc}/_view/{view} |
| Метод | GET |

Параметры строки запроса отображает Таблица 614.

**Таблица 614 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| partition | Имя раздела |
| ddoc | Идентификатор документа проектирования |
| view | Имя представления |

Параметры HTTP заголовка запроса отображает Таблица 615.

**Таблица 615 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 384.

```
GET /db/_partition/sensor-260/_design/sensor-readings/_view/by_sensor HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 384**

### 4.7.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 616.

**Таблица 616 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 385. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Wed, 21 Aug 2013 09:12:06 GMT
ETag: «2FOLSBSW4O6WB798XU4AQYA9B»
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
  "offset": 0,
  "rows": [
    {
      "id": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "key": [
        "sensor-260",
        "0"
      ],
      "value": null
    },
    {
      "id": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "key": [
        "sensor-260",
        "1"
      ],
      "value": null
    },
    {
      "id": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "key": [
        "sensor-260",
        "2"
      ],
      "value": null
    },
    {
      "id": "sensor-260:sensor-reading-ca33c748-2d2c-4ed1-8abf-1bca4d9d03cf",
      "key": [
        "sensor-260",
        "3"
      ],
      "value": null
    }
  ],
  "total_rows": 4
}
```

**Ответ**
**Рисунок 385**

### 4.7.4 Поиск с указанием идентификатора раздела

Метод предназначен для поиска с указанием идентификатора раздела. Возвращенный результат представления будет содержать только строки с указанным идентификатором раздела.

#### 4.7.4.1. Формат запроса

Параметры запроса отображает Таблица 617.

**Таблица 617 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_partition/{partition_id}/_find |
| Метод | POST |

Параметры строки запроса отображает Таблица 618.

**Таблица 618 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| partition_id | Имя раздела для запроса |

### 4.7.5 Получить используемый индекс в запросе

Эта конечная точка показывает, какой индекс используется в запросе.

#### 4.7.5.1. Формат запроса

Параметры запроса отображает Таблица 619.

**Таблица 619 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_partition/{partition_id}/_explain¶ |
| Метод | POST |

Параметры строки запроса отображает Таблица 620.

**Таблица 620 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |
| partition_id | Имя раздела для запроса |

## 4.8. Локальные (не реплицируемые) документы

Интерфейс локальных (не реплицируемых) документов позволяет создавать локальные документы, которые не реплицируются в другие базы данных. Эти документы можно использовать для хранения конфигурации или другой информации, которая требуется конкретно для локального экземпляра БД «Енисей».

Локальные документы имеют следующие ограничения:

Локальные документы не реплицируются в другие базы данных.

Локальные документы не выводятся представлениями или представлением /{db}/_all_docs.

Локальные документы могут быть перечислены с помощью конечной точки /db/_local_docs.

Локальные документы можно использовать, когда требуется хранить конфигурацию или другую информацию для текущего (локального) экземпляра данной базы данных.

Список доступных методов и пути URL отображает Таблица 621.

**Таблица 621 – Параметры HTTP заголовка запроса**

| Метод | Расположение | Описание |
|---|---|---|
| GET, POST | /db/_local_docs | Возвращает список всех невоспроизводимых документов в базе данных |
| GET | /db/_local/id | Возвращает последнюю редакцию невоспроизведенного документа |
| PUT | /db/_local/id | Вставляет новую версию невоспроизводимого документа |
| DELETE | /db/_local/id | Удаляет невоспроизведенный документ |
| COPY | /db/_local/id | Копирует невоспроизведенный документ |

### 4.8.1 Получить JSON-структуру всех локальных документов в заданной базе данных

Возвращает JSON-структуру всех локальных документов в заданной базе данных. Информация возвращается в виде JSON структуры, содержащей мета информацию о возвращаемой структуре, включая список всех локальных документов и основное содержание, состоящее из ID, ревизии и ключа. Ключом является _id локального документа.

#### 4.8.1.1. Формат запроса

Параметры запроса отображает Таблица 622.

**Таблица 622 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_local_docs |
| Метод | GET |

Параметры строки запроса отображает Таблица 623.

**Таблица 623 – Параметры строки запроса**

| Параметр | Описание |
|---|---|
| db | Имя базы данных |

Параметры HTTP заголовка запроса отображает Таблица 624.

**Таблица 624 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Параметры запроса отображает Таблица 625.

**Таблица 625 – Параметры запроса**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| conflicts | Включает информацию о конфликтах в ответ. Игнорируется, если include_docs не является true. По умолчанию false. | Логический тип (Boolean) | Да |
| descending | Возвращать локальные документы в порядке убывания по ключу. По умолчанию false. | Логический тип (Boolean) | Да |
| endkey | Остановить возврат записей при достижении указанного ключа. Необязательно. | Строка (String) | Да |
| end_key | Псевдоним для параметра endkey. | Строка (String) | Да |
| endkey_docid | Остановить возврат записей при достижении указанного идентификатора локального документа. Необязательно. | Строка (String) | Да |
| end_key_doc_id | Псевдоним для параметра endkey_docid. | Строка (String) | Да |
| include_docs | Включать в возврат полное содержимое локальных документов. По умолчанию false. | Логический тип (Boolean) | Да |
| inclusive_end | Указывает, должен ли указанный конечный ключ быть включен в результат. По умолчанию true. | Логический тип (Boolean) | Да |
| key | Возвращает только те локальные документы, которые соответствуют указанному ключу. Необязательно. | Строка (String) | Да |
| keys | Возвращает только локальные документы, соответствующие указанным ключам. Необязательно. | Строка (String) | Да |
| limit | Ограничить количество возвращаемых локальных документов указанным числом. Необязательно. | Число (int64) | Да |
| skip | Пропустить указанное количество записей перед началом возврата результатов. По умолчанию 0. | Число (int64) | Да |
| startkey | Возвращать записи, начинающиеся с указанного ключа. Необязательно. | Строка (String) | Да |
| start_key | Псевдоним для параметра startkey. | Строка (String) | Да |

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| startkey_docid | Возвращать записи, начинающиеся с указанного локального идентификатора документа. Необязательно. | Строка (String) | Да |
| start_key_doc_id | Псевдоним для параметра startkey_docid. | Строка (String) | Да |
| update_seq | Ответ включает значение update_seq, указывающее, какой идентификатор последовательности базовой базы данных отражает представление. По умолчанию - false. | Логический тип (Boolean) | Да |

Параметры HTTP заголовка запроса отображает Таблица 626.

**Таблица 626 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json,<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 386.

```
GET /db/_local_docs HTTP/1.1
Accept: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 386**

#### 4.8.1.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 627.

**Таблица 627 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |

Объекты JSON ответа отображает Таблица 628.

**Таблица 628 – Объекты JSON ответа**

| Поле | Описание | Тип | Обязательность |
|---|---|---|---|
| offset | Смещение, с которого начался локальный список документов | Число (int64) | Да |
| rows | Массив объектов строк представления. По умолчанию возвращаемая информация содержит только идентификатор локального документа и ревизию. | Массив (Array) | Да |
| total_rows | Количество локальных документов в базе данных. Следует обратить внимание, что это не количество строк, возвращаемых в фактическом запросе. | Число (int64) | Да |
| update_seq | Текущая последовательность обновлений для базы данных. | Число (int64) | Да |

Коды состояния отображает Таблица 629.

**Таблица 629 – Коды состояния**

| Код | Описание |
|---|---|
| 200 OK | Запрос успешно завершен |

Ответ в случае успешного завершения запроса отображает Рисунок 387. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "offset": null,
    "rows": [
        {
            "id": "_local/localdoc01",
            "key": "_local/localdoc01",
            "value": {
                "rev": "0-1"
            }
        },
        {
            "id": "_local/localdoc02",
            "key": "_local/localdoc02",
            "value": {
                "rev": "0-1"
            }
        },
        {
            "id": "_local/localdoc03",
            "key": "_local/localdoc03",
            "value": {
                "rev": "0-1"
            }
        },
        {
            "id": "_local/localdoc04",
            "key": "_local/localdoc04",
            "value": {
                "rev": "0-1"
            }
        },
        {
            "id": "_local/localdoc05",
            "key": "_local/localdoc05",
            "value": {
                "rev": "0-1"
            }
        }
    ],
    "total_rows": null
}
```

**Ответ**

**Рисунок 387**

### 4.8.2 Получить JSON-структуру всех локальных документов в заданной базе данных с параметрами строки запроса

Функциональность POST _local_docs поддерживает идентичные параметры и поведение, указанные в ППИ GET /{db}/_local_docs, но позволяет предоставлять параметры строки запроса в качестве ключей в объекте JSON в теле запроса POST.

#### 4.8.2.1. Формат запроса

Параметры запроса отображает Таблица 630.

**Таблица 630 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_local_docs |
| Метод | POST |

Пример запроса отображает Рисунок 388.

```
POST /db/_local_docs HTTP/1.1
Accept: application/json
Content-Length: 70
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "keys" : [
        "_local/localdoc02",
        "_local/localdoc05"
    ]
}
```

**Запрос**
**Рисунок 388**

#### 4.8.2.2. Формат ответа на запрос

Возвращаемый JSON представляет собой структуру всех документов, но с только выбранными ключами в выводе:

Ответ в случае успешного завершения запроса отображает Рисунок 389. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked


{
    "total_rows" : null,
    "rows" : [
        {
            "value" : {
                "rev" : "0-1"
            },
            "id" : "_local/localdoc02",
            "key" : "_local/localdoc02"
        },
        {
            "value" : {
                "rev" : "0-1"
            },
            "id" : "_local/localdoc05",
            "key" : "_local/localdoc05"
        }
    ],
    "offset" : null
}
```

**Запрос**
**Рисунок 389**

### 4.8.3 Получить локальный документ

Получает указанный локальный документ. Семантика идентична доступу к стандартному документу в указанной базе данных, за исключением того, что документ не реплицируется.

#### 4.8.3.1. Формат запроса

Параметры запроса отображает Таблица 631.

**Таблица 631 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_local/{docid} |
| Метод | GET |

Параметры HTTP заголовка запроса отображает Таблица 632.

**Таблица 632 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Content-Type | application/json | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 390.

```
GET /test_people/_local/ 49a73473ea522ae6d9875200bedd407f HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 390**

#### 4.8.3.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 633.

**Таблица 633 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| Transfer-Encoding | chunked | Строка (String) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 391. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 850
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
ETag: "2FOLSBSW4O6WB798XU4AQYA9B"
Server: Yenisei/1.0.0 (Erlang OTP/24)
Transfer-Encoding: chunked
{
    "_id": "_local/49a73473ea522ae6d9875200bedd407f",
    "_rev": "0-1",
    "session_id": "4033c297c9c508f198fc362666eaa460",
    "source_last_seq": "10-
g1AAAACbeJzLYWBgYMpgTmEQTM4vTc5ISXLIyU9OzMnILy7JAUklMiTV____PyuDOZE5FyjAnmhukJaWmIpNAx5j8liAJEMDkPoP
NY0dbJqpgbGFhYUhNn1ZAJDRMKs",
    "replication_id_version": 4,
    "history": [
        {
            "session_id": "4033c297c9c508f198fc362666eaa460",
            "start_time": "Wed, 24 Aug 2022 10:56:13 GMT",
            "end_time": "Wed, 24 Aug 2022 10:56:18 GMT",
            "start_last_seq": 0,
            "end_last_seq": "10-
g1AAAACbeJzLYWBgYMpgTmEQTM4vTc5ISXLIyU9OzMnILy7JAUklMiTV____PyuDOZE5FyjAnmhukJaWmIpNAx5j8liAJEMDkPoP
NY0dbJqpgbGFhYUhNn1ZAJDRMKs",
            "recorded_seq": "10-
g1AAAACbeJzLYWBgYMpgTmEQTM4vTc5ISXLIyU9OzMnILy7JAUklMiTV____PyuDOZE5FyjAnmhukJaWmIpNAx5j8liAJEMDkPoP
NY0dbJqpgbGFhYUhNn1ZAJDRMKs",
            "missing_checked": 10,
            "missing_found": 10,
            "docs_read": 10,
            "docs_written": 10,
            "doc_write_failures": 0
        }
    ]
}
```

**Ответ**
**Рисунок 391**

### 4.8.4 Сохранить локальный документ

Сохраняет указанный локальный документ. Семантика идентична хранению стандартного документа в указанной базе данных, за исключением того, что документ не реплицируется.

#### 4.8.4.1. Формат запроса

Параметры запроса отображает Таблица 634.

**Таблица 634 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_local/{docid} |
| Метод | PUT |

Параметры HTTP заголовка запроса отображает Таблица 635.

**Таблица 635 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 392.

```
GET /test_people/_local/ id1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
{
    "_id": "id1",
    "title": "doc"
}
```

**Запрос**
**Рисунок 392**

#### 4.8.4.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 636.

**Таблица 636 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 393. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 42
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
Location: “http://localhost:5984/test_people/_local%2Fid1”
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Request-ID: 31b37c16be
{
    "ok": true,
    "id": "_local/id1",
    "rev": "0-1"
}
```

**Ответ**
**Рисунок 393**

### 4.8.5 Удалить локальный документ

Удаляет указанный локальный документ. Семантика идентична удалению стандартного документа в указанной базе данных, за исключением того, что документ не реплицируется.

#### 4.8.5.1. Формат запроса

Параметры запроса отображает Таблица 637.

**Таблица 637 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port] /{db}/_local/{docid} |
| Метод | DELETE |

Параметры HTTP заголовка запроса отображает Таблица 638.

**Таблица 638 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 394.

```
DELETE /test_people/_local/ id1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 394**

#### 4.8.5.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 639.

**Таблица 639 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 395. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 200 OK
Cache-Control: must-revalidate
Content-Length: 42
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Request-ID: fdf4a85f5f
{
    "ok": true,
    "id": "_local/id1",
    "rev": "0-0"
}
```

**Ответ**
**Рисунок 395**

### 4.8.6 Копировать локальный документ

Копирует указанный локальный документ. Семантика идентична копированию стандартного документа в указанной базе данных, за исключением того, что документ не реплицируется.

#### 4.8.6.1. Формат запроса

Параметры запроса отображает Таблица 640.

**Таблица 640 – Параметры запроса**

| Параметр | Описание |
|---|---|
| URL | http://<server>[:port]/{db}/_local/{docid} |
| Метод | COPY |

Параметры HTTP заголовка запроса отображает Таблица 641.

**Таблица 641 – Параметры HTTP заголовка запроса**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Accept | application/json,<br>text/plain | Строка (String) | Да |
| Authorization | Данные авторизации (Basic). Имя пользователя и пароль записываются в формате username:password и данные кодируются в Base64 (YWRtaW46YWRtaW4=)<br>Пример:<br>Authorization: Basic YWRtaW46YWRtaW4= | Строка (String) | Да |

Пример запроса отображает Рисунок 396.

```
COPY /test_people/_local/ id1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Destination: id2
Authorization: Basic YWRtaW46YWRtaW4=
Host: localhost:5984
```

**Запрос**
**Рисунок 396**

### 4.8.6.2. Формат ответа на запрос

Параметры HTTP заголовка ответа отображает Таблица 642.

**Таблица 642 – Параметры HTTP заголовка ответа**

| Параметр | Описание | Тип | Обязательность |
|---|---|---|---|
| Content-Type | application/json<br>text/plain; charset=utf-8 | Строка (String) | Да |
| Cache-Control | must-revalidate | Строка (String) | Да |
| Content-Length | Длина (в байтах) возвращаемого содержимого | Число (int64) | Да |
| ETag | Новая ревизия документа в двойных кавычках | Строка (String) | Да |
| Location | URI документа | Строка (String) | Да |

Ответ в случае успешного завершения запроса отображает Рисунок 397. Коды ошибок приведены в пункте 6.1.2.

```
HTTP/1.1 201 Created
Cache-Control: must-revalidate
Content-Length: 42
Content-Type: application/json
Date: Sat, 23 Dec 2017 16:22:56 GMT
Etag: "0-1"
Location: http://localhost:5984/test_people/id2
Server: Yenisei/1.0.0 (Erlang OTP/24)
X-Couch-Request-ID: e015a0caf2
{
    "ok": true,
    "id": "id2",
    "rev": "1-673aa0eab65ab444a718b40d7fb33f17"
}
```

**Ответ**

**Рисунок 397**

# 5. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

## 5.1. Характер, организация и предварительная подготовка входных и выходных данных

### 5.1.1 Источники информации

Основными источниками информации для СУБД Енисей:

1) Данные, вводимые пользователями системы.

Операции по осуществлению:

1) аналого-цифрового и цифро-аналогового преобразования сигналов;
2) оптического распознавания символов;
3) иных действий по приведению информации к виду, пригодному для обработки на ЭВМ

в составе функций СУБД Енисей не предусмотрены.

### 5.1.2 Методы организации сбора, передачи, контроля и корректировки информации

Сбор массивов информации происходит в процессе эксплуатации СУБД Енисей путём:

1) Получения посредством сервисов взаимодействия в формате JSON-документов.
2) Ввода пользователями информации в экранных формах и её последующего сохранения в базе данных.

Контроль целостности данных реализуется прикладным ПО системы и встроенными в СУБД Енисей средствами (ограничениями, индексами). Ввод и корректировка данных должны осуществляться только через программные компоненты СУБД Енисей.

Основными требованиями, предъявляемыми к процессам организации сбора, передачи, контроля и корректировки информации являются обеспечение достоверности, верифицируемости, конфиденциальности, доступности, оперативности собираемых и передаваемых данных.

Требование достоверности подразумевает организацию процесса сбора и передачи информации, в рамках которого передаваемая и собираемая информация не будет подлежать искажению.

Требование верифицируемости подразумевает организацию процесса сбора и передачи информации, в рамках которого обеспечивается возможность контроля за достоверностью передаваемой информации.

Требование конфиденциальности означает предоставление доступа к информации в строгом соответствии с установленными приоритетами и правилами разграничения доступа.

Требование доступности означает принципиальную возможность получения собираемой информации и ее передачу.

Требование оперативности подразумевает такую организацию процесса сбора и передачи информации, в процессе которого доступная информация будет передана в приемлемые для ее анализа сроки.

## 5.2. Формат, описание и способ кодирования входных и выходных данных при использовании ППИ

Формат, описание и способ кодирования входных и выходных данных при использовании ППИ СУБД Енисей приведены в описании соответствующих методов.

## 5.3. Справочники, доступные посредством ППИ

### 5.3.1 Сервер

#### 5.3.1.1. Перечень типов заданий

Перечень типов заданий отображает Таблица 643.

**Таблица 643 – Типы заданий**

| Параметр | Описание |
|---|---|
| database_compaction | Сжатие базы данных |
| view_compaction | Сжатие представления |
| indexer | Индексатор |
| replication | Репликация |

#### 5.3.1.2. Возвращаемые состояния узла или кластера

Возвращаемое состояние, указывающее на текущее состояние узла или кластера отображает Таблица 644.

**Таблица 644 – Описание состояний**

| Состояние | Описание |
|---|---|
| cluster_disabled | Текущий узел полностью не сконфигурирован |
| single_node_disabled | Текущий узел настроен как одиночный (автономный) узел ([cluster] n=1), но у него либо не определен пользователь-администратор на уровне сервера, либо не созданы стандартные системные базы данных. Если указан параметр запроса ensure_dbs_exist, предоставленный список баз данных отменяет список стандартных системных баз данных по умолчанию. |
| single_node_enabled | Текущий узел настроен как единственный (автономный) узел, определен пользователь-администратор на уровне сервера и создан список баз данных ensure_dbs_exist (явный или по умолчанию). |
| cluster_enabled | Текущий узел имеет [cluster] n > 1, не привязан к 127.0.0.1 и имеет определенного пользователя администратора на уровне сервера. Однако полный набор стандартных системных баз данных еще не создан. Если указан параметр запроса ensure_dbs_exist, предоставленный список баз данных переопределяет список стандартных системных баз данных по умолчанию |
| cluster_finished | Текущий узел имеет [cluster] n > 1, не привязан к 127.0.0.1, у него определен пользователь admin на уровне сервера и создан список баз данных ensure_dbs_exist (явный или по умолчанию). |

#### 5.3.1.3. Поле action настройки узла

Значения поля action отображает Таблица 645.

**Таблица 645 – Значения поля action**

| Поле | Описание |
|---|---|
| enable_single_node | Настроить текущий узел как единственный, автономный сервер БД «Енисей» |
| enable_cluster | Настроить локальный или удаленный узел как один узел, подготавливая его к присоединению к новому кластеру БД «Енисей» |
| add_node | Добавляет указанный удаленный узел в список узлов этого кластера, присоединяя его к кластеру |
| finish_cluster | Завершить работу кластера, создав стандартные системные базы данных |

#### 5.3.1.4. Разделы и типы статистики

Статистика разделена на следующие разделы верхнего уровня, которые отображает Таблица 646.

**Таблица 646 – Разделы статистики**

| Раздел | Описание |
|---|---|
| couch_log | Подсистема протоколирования |
| couch_replicator | Планировщик и подсистема репликации |
| couchdb | Первичные операции с базой данных БД «Енисей» |
| fabric | Операции, связанные с кластером |
| global_changes | Поток глобальных изменений |
| mem3 | Статистика, связанная с участием узла |
| pread | Исключения, связанные с файлами БД «Енисей» |
| rexi | Статистика, связанная с внутренним RPC кластера. |

Тип статистики указывается в поле type, типы отображает Таблица 647.

**Таблица 647 – Типы статистики**

| Тип | Описание |
|---|---|
| Counter (счетчик) | Монотонно увеличивающийся счетчик, сбрасывается при перезапуске |
| histogram (гистограмма) | Биннированный набор значений с определенными делениями. Масштабируется на текущий интервал сбора |
| gauge (монометр) | Одиночное числовое значение, которое может увеличиваться и уменьшаться |

### 5.3.2 Проектные документы

#### 5.3.2.1. Информация об индексе представления

Информацию об индексе представления view_index(object) отображает Таблица 648.

**Таблица 648 – Объекты JSON ответа**

| Поле | Описание | Тип |
|---|---|---|
| compact_running | Указывает, запущена ли в данный момент процедура сжатия для данного представления | Логический тип (Boolean) |
| sizes.active | Размер активных данных внутри представления, в байтах | Число (int64) |
| sizes.external | Размер содержимого представления без сжатия в байтах | Число (int64) |
| sizes.file | Размер в байтах представления, сохраненного на диске | Число (int64) |
| language | Язык для определенных представлений | Строка (String) |
| purge_seq | Последовательность очистки, которая была обработана | Число (int64) |
| signature | MD5-подпись представлений для проектного документа | Строка (String) |
| update_seq | Последовательность обновления соответствующей базы данных, которая была проиндексирована | Число (int64) / Строка (String) |
| updater_running | Указывает, обновляется ли представление в настоящее время | Логический тип (Boolean) |
| waiting_clients | Количество клиентов, ожидающих представления из данного проектного документа | Число (int64) |
| waiting_commit | Указывает, есть ли невыполненные фиксации в базовой базе данных, которые необходимо обработать | Логический тип (Boolean) |

# 6. СООБЩЕНИЯ

## 6.1. Форматы и коды ошибок

### 6.1.1 Форматы и коды ошибок ППИ

Структуру ответа с ошибкой отображает

**Таблица 649 – Формат ответа с ошибкой**

| Атрибут | Описание | Тип данных | Обязательность |
|---|---|---|---|
| error | Краткое описание ошибки | Строка (String) | Да |
| reason | Причина ошибки | Число (int64) | Да |

Пример ошибки, в случае если данные не найдены, отражает Рисунок 398.

```
HTTP/1.1 404
X-Couch-Request-ID: 0a8637d5db
Content-Type: application/json
Content-Length: 58

{
    "error": "not_found",
    "reason": "Database does not exist."
}
```

**Пример JSON ответа**
**Рисунок 398**

Пример ошибки, в случае не пройдена аутентификация, отражает Рисунок 399.

```
HTTP/1.1 401
X-Couch-Request-ID: 6d9faeb609
Content-Type: application/json
Content-Length: 67

{
    "error": "unauthorized",
    "reason": " Name or password is incorrect."
}
```

**Пример JSON ответа**
**Рисунок 399**

Пример ошибки, в случае некорректных данных в запросе отражает Рисунок 399.

```
HTTP/1.1 400
X-Couch-Request-ID: e0403795e5
Content-Type: application/json
Content-Length: 54

{
    "error": "bad_request",
    "reason": "invalid UTF-8 JSON"
}
```

**Пример JSON ответа**
**Рисунок 400**

### 6.1.2 HTTP коды ошибок

В таблице приведен список кодов ошибок, возвращаемых БД «Енисей», и общие описания соответствующих ошибок, которые отображает Таблица 650.

**Таблица 650 – Коды ошибок**

| Код | Описание |
|---|---|
| 202 Accepted | Запрос был принят, но соответствующая операция может быть не завершена. Используется для фоновых операций, таких как сжатие базы данных. |
| 304 Not Modified | Запрошенный дополнительный контент не был изменен. Используется с системой ETag для определения версии возвращаемой информации. |
| 400 Bad Request | Плохая структура запроса. Ошибка может указывать на ошибку в URL запроса, пути или заголовках. Различия в предоставленном хэше MD5 и содержимом также вызывают эту ошибку, так как это может указывать на повреждение сообщения. |
| 401 Unauthorized | Запрашиваемый элемент недоступен с помощью предоставленной авторизации, или авторизация не была предоставлена. |
| 403 Forbidden | Запрашиваемый элемент или операция запрещены. |
| 404 Not Found | Запрошенное содержимое не может быть найдено. Содержимое будет включать дополнительную информацию в виде объекта JSON, если она доступна. Структура будет содержать два ключа: ошибка и причина. Например:<br>{"error":"not_found","reason":"no_db_file"} |
| 405 Method Not Allowed | Запрос был сделан с использованием недопустимого типа HTTP-запроса для запрашиваемого URL. Например, был произведен запрос PUT, в то время как требуется POST. Ошибки этого типа также могут быть вызваны недопустимыми строками URL. |
| 406 Not Acceptable | Запрашиваемый тип содержимого не поддерживается сервером. |
| 409 Conflict | Запрос привел к конфликту обновлений. |
| 412 Precondition Failed | Заголовки запроса от клиента и возможности сервера не совпадают. |
| 413 Request Entity Too Large | Документ превышает настроенное значение couchdb/max_document_size или весь запрос превышает значение chttpd/max_http_request_size. |
| 415 Unsupported Media Type | Поддерживаемые типы содержимого и тип содержимого запрашиваемой или предоставляемой информации указывают, что тип содержимого не поддерживается. |
| 416 Requested Range Not Satisfiable | Диапазон, указанный в заголовке запроса, не может быть удовлетворен сервером. |
| 417 Expectation Failed | При отправке документов операция загрузки завершилась неудачно. |
| 500 Internal Server Error | Запрос был некорректным, либо потому что предоставленный JSON был некорректным, либо некорректная информация была предоставлена как часть запроса. |

# ПРИЛОЖЕНИЕ

# ПЕРЕЧЕНЬ ИЗМЕНЕНИЙ ПРИКЛАДНОГО ПРОГРАММНОГО ИНТЕРФЕЙСА

| *Редакция* | *Дата* | *Версия* | *Список внесенных изменений* |
|---|---|---|---|
| 1 | 04.03.2022 | 1 | Первая редакция |

# ПЕРЕЧЕНЬ ТЕРМИНОВ

В настоящем документе использованы следующие термины:

1) Программное изделие — программа на носителе данных, являющаяся продуктом промышленного производства.

2) Средство вычислительной техники (СВТ) — ПЭВМ (персональная электронно-вычислительная машина) либо другое вычислительное оборудование (мэйнфрейм, мини-ЭВМ, микро-ЭВМ, КПК (карманный персональный компьютер), компьютерный терминал).

3) СВТ индивидуального пользования — вычислительное оборудование, обеспечивающее доступ отдельного пользователя к информационным сервисам, предоставляемым программным изделием:

– Сервер (стоечный или отдельно стоящий).

– Многомашинный вычислительный комплекс (ММВК), то есть серверный кластер.

4) СВТ коллективного пользования — вычислительное оборудование, предназначенное для реализации программным изделием информационных сервисов, предоставляемых всем пользователям, имеющим доступ:

– Автоматизированное рабочее место (АРМ) на базе ПЭВМ.

– Портативный компьютер (ноутбук).

5) Мобильное СВТ — вычислительное оборудование повышенной портативности:

– Карманный персональный компьютер (КПК).

– КПК со встроенным модулем мобильной связи — смартфоны и коммуникаторы.

6) Система управления базами данных — совокупность программных и языковых средств, обеспечивающих управление базами данных.

7) cURL — утилита командной строки, которая позволяет выполнять HTTP-запросы с различными параметрами и методами/

8) Прикладной программный интерфейс — совокупность методов, позволяющих стороннему программному средству получить доступ к функциям программного изделия, обеспечивающим обработку данных, управление ОС, контроль над СВТ и т.д. Английское наименование термина — application programming interface, сокращенно ППИ.

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

| | |
|---|---|
| АРМ | Автоматизированное рабочее место |
| АС | Автоматизированная система |
| АСУТП | Автоматизированная система управления технологическим процессом |
| БД | База данных |
| КПК | Карманный персональный компьютер |
| ММВК | Многомашинный вычислительный комплекс |
| ОС | Операционная система |
| ППИ | Прикладной программный интерфейс |
| ПК | Программный комплекс |
| ПО | Программное обеспечение |
| ПС | Программное средство |
| ПЭВМ | Персональная электронно-вычислительная машина |
| СВТ | Средство вычислительной техники |
| СПО | Специальное программное обеспечение |
| СУБД | Система управления базами данных |
| ППИ | Application programming interface, программный интерфейс приложения |

# ПЕРЕЧЕНЬ РИСУНКОВ

# ПЕРЕЧЕНЬ ТАБЛИЦ