

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Ижевский государственный технический университет
имени М.Т. Калашникова»
(ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»)

Факультет «Информатика и вычислительная техника»
Кафедра «Программное обеспечение»
Направление подготовки 09.03.04 Программная инженерия

ОТЧЕТ ПО ПРАКТИКЕ

(наименование типа практики)

(полное наименование профильной организации)

Выполнил обучающийся _____ / _____ /
(подпись) (ФИО, курс, номер группы)

Дата сдачи отчета: «__» _____ 202_г.

Дата аттестации «__» _____ 202_г.

Оценка _____

Руководитель практики от
ИЖГТУ имени М.Т. Калашникова _____ / _____ /
(подпись) (И.О. Фамилия, должность, ученая степень)

Заведующий кафедрой _____ / _____ /
(подпись) (И.О. Фамилия, должность, ученая степень)

2. РАЗРАБОТКА ЗАДАЧИ

2.1. Описание постановки задачи

2.1.1. Характеристика задачи

Для решения поставленной задачи необходимо разработать различные алгоритмы для работы с базой данных key-value, формирования диапазонов индексирования с привязкой к нодам, хранения диапазонов и управления их статусами, отправки данных на индексирование в Apache Solr и слежения за статусом индексирования.

2.1.2. Входная информация

Входная информация сервиса базы данных key-value: характеристика необходимой информации, а именно тип коллекции (`collection_id`) и тип информации(`type_id`), в виде массивов из чисел (см. таблицу 1).

Входная информация сервиса хранения диапазонов: словарь из номера бакета и его максимального размера в формате {номер бакета: размер, ...}

Входная информация сервиса индексирования: объект, содержащий идентификатор диапазона, номер начала, номер конца, номер бакета, номер ноды.

Таблица 1

Название	Тип
<code>collection_id</code>	Array of integers
<code>type_id</code>	Array of integers

2.1.3. Выходная информация

Выходная информация сервиса базы данных key-value: сервис возвращает JSON с размерами всех бакетов, в соответствии с переданными ограничениями. Структура JSON: {номер бакета: размер, ...}

Выходная информация сервиса хранения диапазонов: объект, содержащий идентификатор диапазона, номер начала, номер конца, номер бакета, номер ноды.

Выходная информация сервиса индексирования для Solr: номер начала, номер конца, номер бакета, номер ноды

Выходная информация сервиса индексирования для сервиса диапазонов: статус индексирования, идентификатор диапазона.

Таблица 2

Название	Тип
kvSize	JSON integer: long
range	Object
Range	Object
indexedRange	Object

2.2. Описание алгоритма получения информации о базе данных key-value

2.2.1. Назначение и характеристика алгоритма

Алгоритм предназначен для получения актуального размера каждой таблицы (бакета) и их привязки к серверу (ноде). В базе происходит поиск порядковых номеров последних записей, из которых формируется словарь, содержащий номер бакета и его размер.

2.2.2. Используемая информация

Массивы чисел, содержащие тип коллекции (`collection_id`) и тип информации(`type_id`).

2.2.3. Результаты решения

Возвращает словарь, содержащий номер бакета как ключ и его размер как значение.

2.2.4. Алгоритм решения

Алгоритм получения информации о базе данных key-value:

- 1) проитерироваться по всем бакетам
- 2) получить номер последнего значения в каждом
- 3) собрать номера бакетов и их размеры в словарь

Алгоритм представлен на рисунке 2.1.

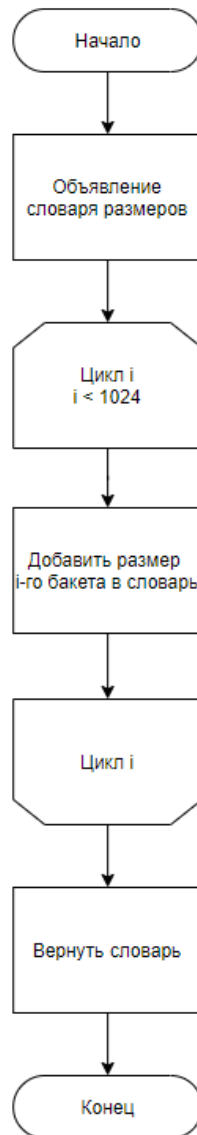


Рис. 2.1 Алгоритм получения информации о базе данных key-value

2.3. Описание алгоритма формирования диапазона

2.3.1. Назначение и характеристика алгоритма

Алгоритм предназначен для формирования диапазонов индексирования на основе двух состояний БД: старого, на момент прошлого индексирования, и нового, полученного во время текущего индексирования.

2.3.2. Используемая информация

Используются 2 словаря с одинаковой структурой: ключ – номер бакета, значение – текущий размер. Первый словарь формируется из содержимого базы диапазонов, второй возвращает сервис информации о базе `ky-value`.

2.3.3. Результат решения

Массив объектов диапазонов, состоящих из идентификатора диапазона, идентификатор бакета, номера значения начала индексирования и номера конца.

2.3.4. Алгоритм решения

Алгоритм формирования диапазона

- 1) получить текущие размеры БД
- 2) получить прошлые размеры БД
- 3) создать новый диапазон на основе них

Алгоритм представлен на рисунках 2.2 и 2.3.

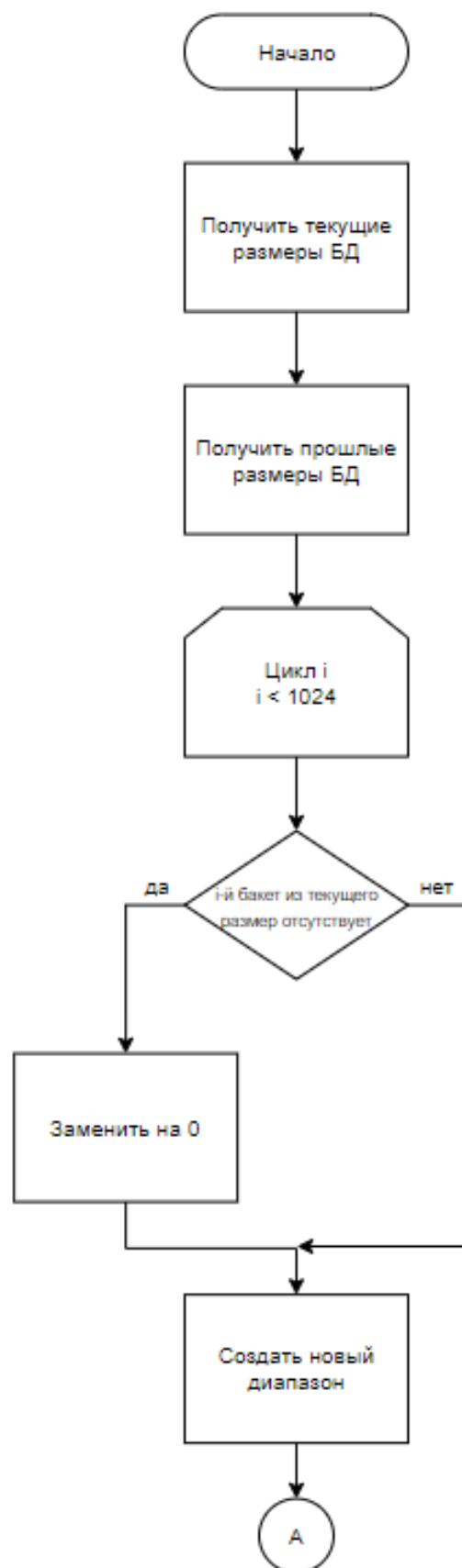


Рис. 2.2 Алгоритм формирования диапазона



Рис. 2.3 Алгоритм формирования диапазона

2.4. Описание алгоритма сохранения статуса

2.4.1. Назначение и характеристика алгоритма

Так как в процессе индексирования могут возникнуть ошибки из-за неверных данных или неполадки сервера, необходимо отслеживать, какие диапазоны успешно попали в хранилище Solr, а какие не смогли пройти индексирования. Для этого используются статусы: «ожидает», «завершён», «ошибка». Диапазоны со статусом «ожидает» и «ошибка» будут отправлены на индексацию при первой возможности. При добавлении диапазона ему ставится статус «ожидает». Статусы «завершён» и «ошибка» устанавливаются на основе ответа от Solr.

2.4.2. Используемая информация

- 1) Идентификатор диапазона
- 2) Статус индексирования от Solr
- 3) Статус диапазона от сервиса индексации

2.4.3. Результат решения

Изменение статуса диапазона на основе ответа от сервиса индексации.

2.4.4. Алгоритм решения

- 1) Получить статус от Solr
- 2) Проанализировать сообщение с ответом
- 3) Отправить в сервис диапазонов
- 4) Обновить статус в БД на основе ответа

Алгоритм представлен на рисунке 2.4.

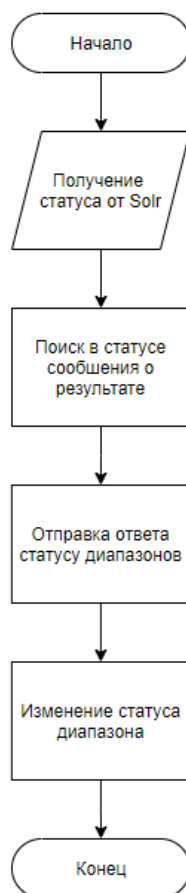


Рис. 2.4 Алгоритм сохранения статуса

2.5. Описание алгоритма отправки данных на индексацию

2.5.1. Назначение и характеристика алгоритма

Отправка данных в Solr происходит в несколько потоков. Потоки работают в паре: «читатель» и «писатель». «Писатель» принимает первый доступный диапазон от сервиса хранения диапазонов и записывает информацию из БД в очередь, группируя информацию по ключу номера данных. «Читатель» формирует пакет данных и отправляет его в Solr, предварительно преобразовав их в подходящий формат. Конец работы происходит при появлении в очереди терминального объекта или слишком долгом простое одного из потоков.

2.5.2. Используемая информация

Диапазон индексируемых данных из базы key-value.

2.5.3. Результат решения

Проиндексированные данные в Solr.

2.5.4. Алгоритм решения

- 1) Создать несколько потоков «читателей» и «писателей»
- 2) Создать очереди, объединяющие потоки в пары
- 3) Начать чтение данных
- 4) Преобразовать группу данных в одно значение с необходимым форматом
- 5) Отправить в Solr

Алгоритм представлен на рисунках 2.5 и 2.6.

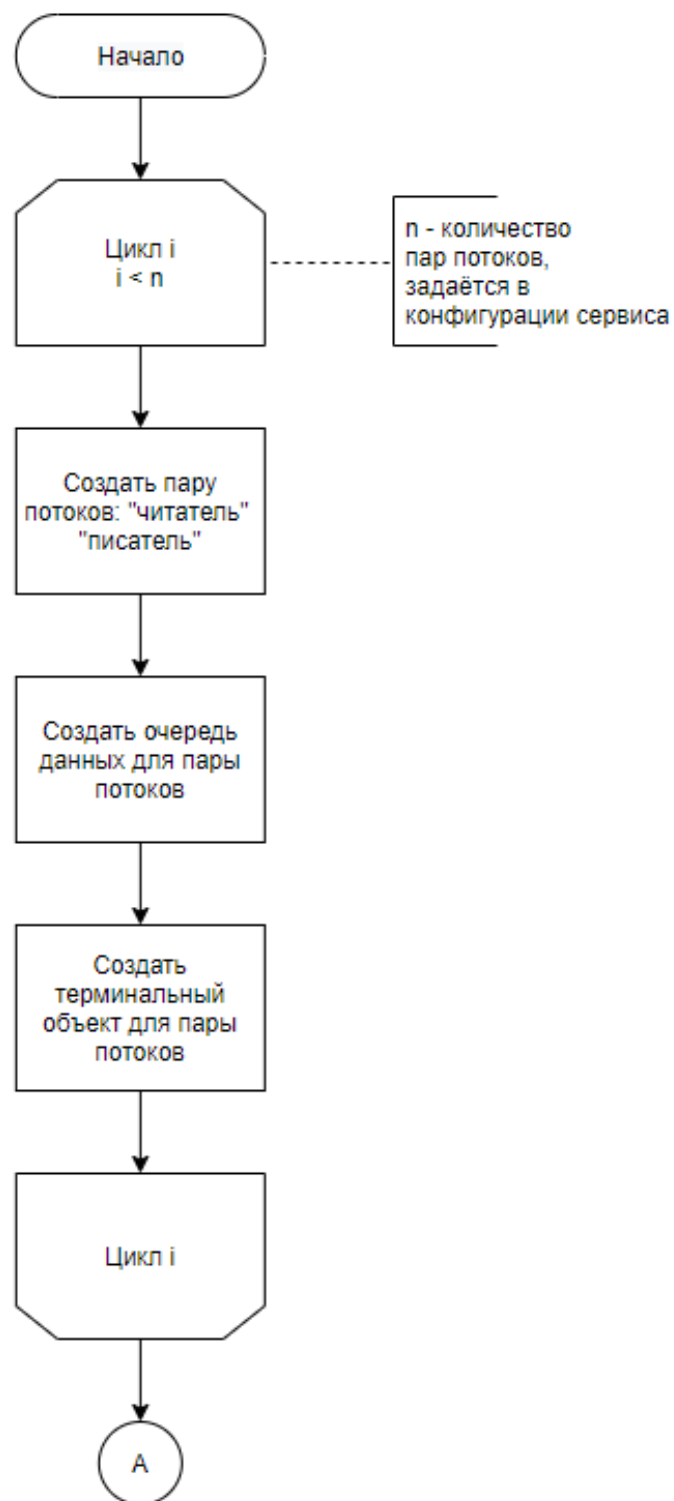


Рис. 2.5 Алгоритм отправки данных на индексацию

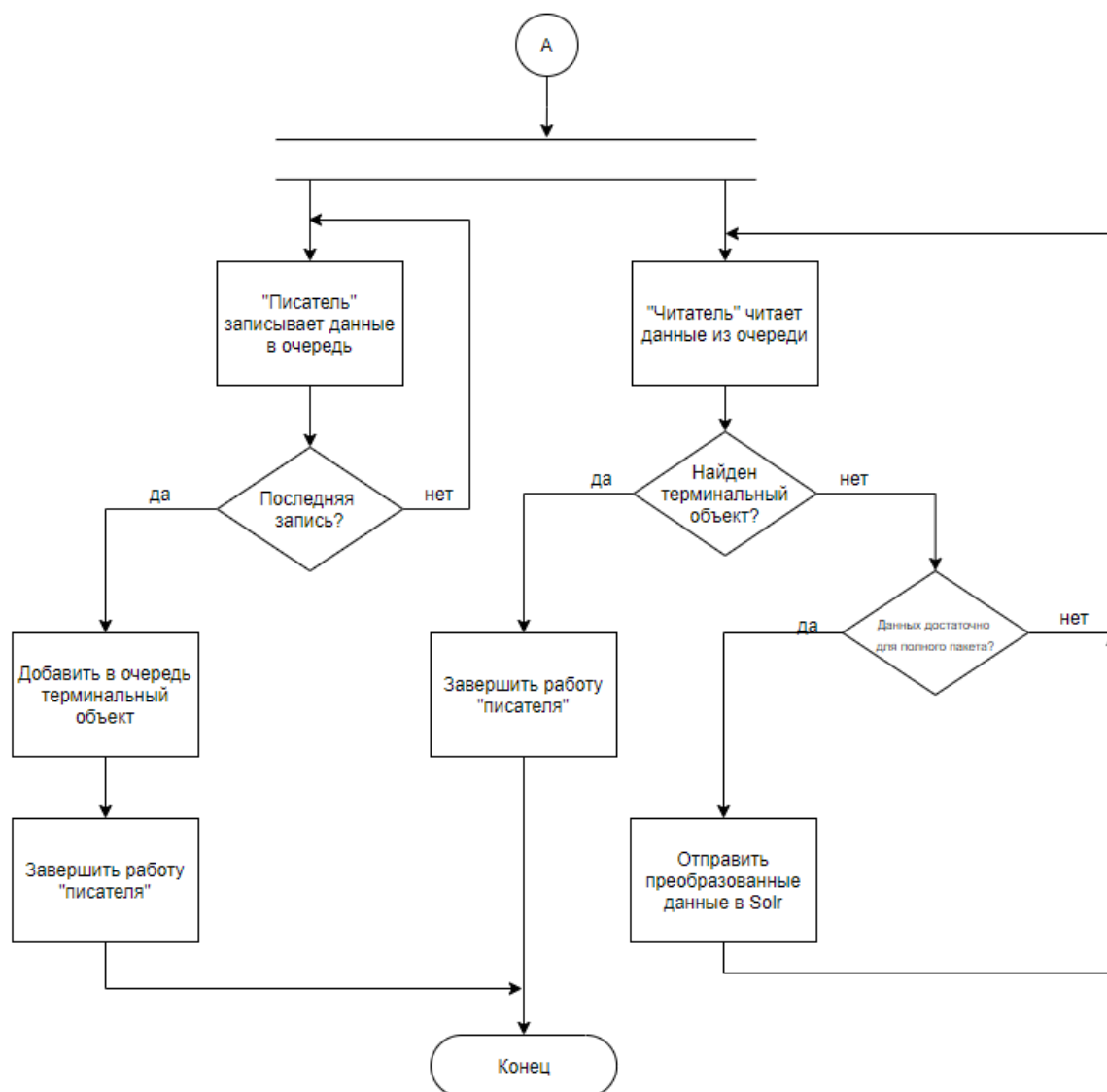


Рис. 2.6 Алгоритм отправки данных на индексацию

2.6. Описание алгоритма преобразования группы данных

2.6.1. Назначение и характеристика алгоритма

Данные в БД key-value приходят из разных источников и в разное время, из-за чего возможно появление однотипных данных с разной степенью полноты и минорными отличиями в текстовых полях. Пример - несколько объектов с данными одного судебного дела, взятые с разных сайтов. Но в Solr необходимо хранить только одну, наиболее полную редакцию дела. Для этого разработан алгоритм группировки и слияния подобных объектов.

2.6.2. Используемая информация

Данные одного дела сгруппированы по UUID ключу в БД union_key.

Данные дела хранятся в виде JSON объекта, с вариативным набором полей. Пример данных показан на рисунке 2.7.

```
{
  "caseNumber": "05-1008/405/2019",
  "court": {
    "name": "Судебный участок № 405 Кунцевского судебного района г. Москвы",
    "type": "МС"
  },
  "judge": "Нефедычева Л.М.",
  "caseType": "А",
  "startDate": "2019-09-24T00:00:00",
  "endDate": "2019-09-24T00:00:00",
  "region": 77,
  "sides": [
    {
      "side": "Карелин Д.А.",
      "role": "DEFENDANT",
      "roleName": "Привлекаемое лицо"
    }
  ],
  "articles": [
    {
      "caseArticleSource": "КОАП",
      "numbers": [
        19, 4
      ],
      "part": 1
    }
  ],
  "links": [
    {
      "type": "ROOT",
      "link": "http://mos-sud.ru/ms/405"
    }, {
      "type": "CARD",
      "link": "http://mos-sud.ru/ms/405/consideration/as/?pn\u003d0\u0026id\u003d"
    }
  ],
  "result": "Завершено",
  "engine": "MOS_MIR",
  "type": "HEADER"
}
```

Рис.2.7 Пример данных судебного дела

2.6.3. Результат решения

Один объект дела, содержащий наиболее полную информацию из нескольких редакций дела.

2.6.4. Алгоритм решения

Алгоритм слияние данных:

- 1) Взять первый объект из группы как основной
- 2) Начать итерацию по полям следующего объекта
- 3) Если поле следующего объекта более полное, то дополнить им основной
- 4) Если поле отсутствует в основном объекте, то добавить его без изменений

Алгоритм представлен на рисунках 2.8 и 2.9.

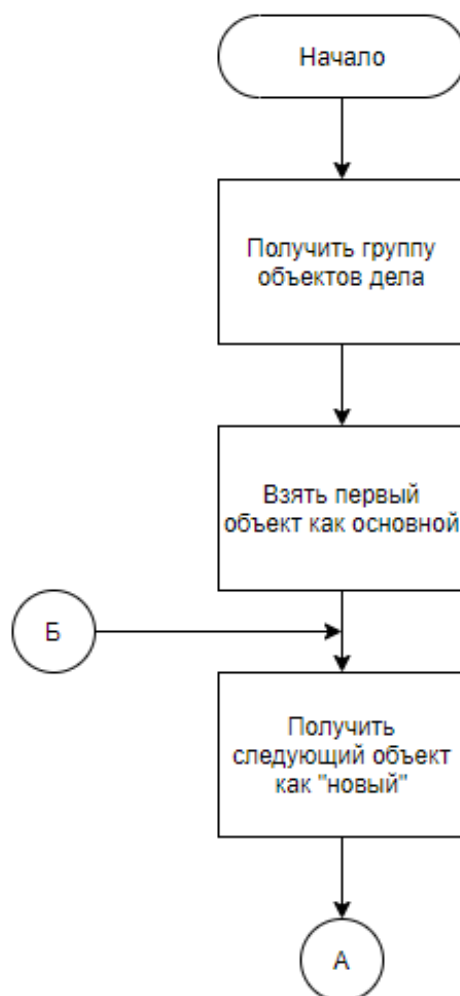


Рис.2.8 Алгоритм преобразования группы данных

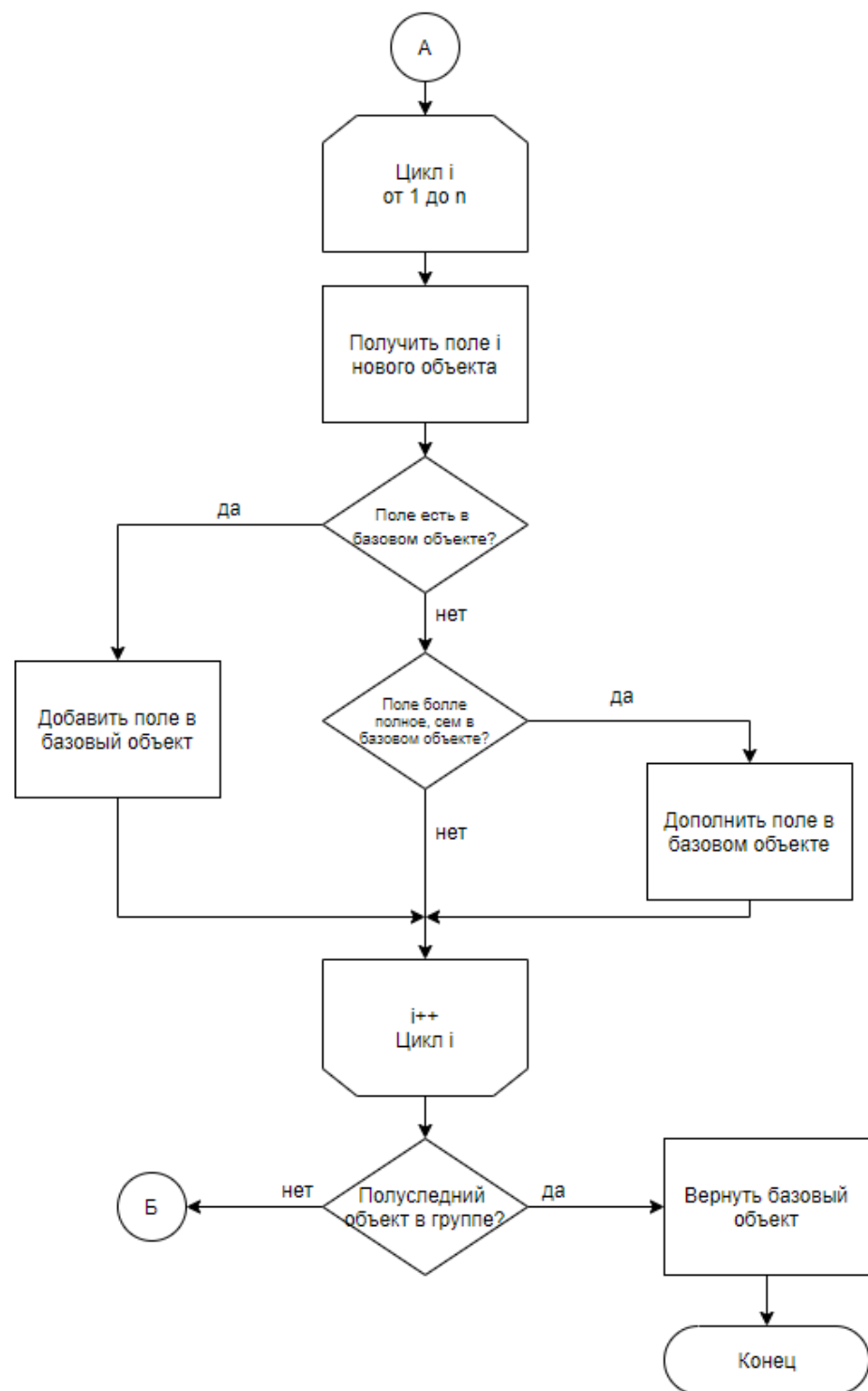


Рис.2.9 Алгоритм преобразования группы данных

2.7. Описание программы для сервиса базы данных key-value

2.7.1. Функциональное назначение

Распределённая база данных kv (key-value) состоит из 4 реляционных баз данных, распределённых по отдельным серверам. В каждой базе 1024 таблицы, из которых заполнены 256, остальные необходимы для поддержания структуры распределённой системы как единого хранилища.

Сервис базы данных key-value предназначен для получения по HTTP запросу текущего размера каждой таблицы-бакета в БД и привязки бакетов к нодам. БД распределена по нескольким серверам, из-за чего необходимо прописывать конфигурации и код переключения подключений в каждом проекте, требующем соединения с данной БД.

Чтобы избежать повторения кода, был создан данный сервис, уже сконфигурированный для подключения и переключения соединений, предоставляющий интерфейс REST API для других сервисов.

2.7.2. Описание информации

Входные данные для системы:

- 1) тип коллекции (collection_id) – массив чисел
- 2) тип информации (type_id) – массив чисел
- 3) конфигурация системы (таблица 3)

Таблица 3

Название	Тип	Назначение
kv.threads	Integer	Количество потоков для чтения из бд
kv.nodes.id	Integer	Идентификатор ноды (номер)
kv.nodes.url	String	Адрес хоста базы данных (адрес:порт/имяБД)

kv.nodes.username	String	Логин для входа в базу данных
kv.nodes.password	String	Пароль для входа в базу данных
kv.nodes.bucket- range	String	Диапазон бакетов, содержащих данные

Параметры с префиксом kv.nodes являются списком, описывающим все рабочие ноды базы данных.

Выходные данные системы:

JSON с размерами всех бакетов

2.7.3. Описание логики

При запуске сервиса приложение считывает конфигурацию, настраивая подключение ко всем нодам базы данных. При поступлении запроса сервис итерируется по всем бакетам, получая их размеры и формируя результирующий словарь.

Исходя из конфигурации, каждому бакету соответствует одна нода. Во время итерации, на основе номера бакета определяется необходимая нода и соответствующее ей соединение с сервером БД.

Приложение является веб-сервисом, поэтому основная работа происходит по GET запросу.

Алгоритм работы системы:

- 1) Запуск
- 2) Парсинг конфигурации
- 3) Настройка подключений к нодам
- 4) Ожидание GET запроса
- 5) Считывание размеров каждого бакета

6) Формирование словаря запросов

7) Ответ на GET запрос

Алгоритм работы сервиса представлен на рисунке 2.10.



Рис. 2.10 Алгоритм работы сервиса базы данных key-value

2.8. Описание программы для сервиса хранения диапазонов

2.8.1. Функциональное назначение

Данный сервис предназначен для хранения информации о базе данных key-value, а именно о связях БД с нодами и бакетами. Также сервис хранит

диапазоны индексирования, отправляет их по GET запросу и управляет их статусами, полученными из POST запроса.

Основной объект сервиса – диапазон индексирования. Диапазон представляет собой объект, содержащий идентификатор, номер ноды, номер бакета, статус, номер начала диапазона и номер конца, тип коллекции и информации. Другие сервисы, на основе этого объекта, получают определённый «срез» таблицы, что и позволяет проводить индексацию по частям.

2.8.2. Описание информации

Входные данных системы:

- 1) Таблицы с информацией о базах данных
- 2) Типы коллекций
- 3) Типы информации
- 4) Словарь из номера бакета и его размера
- 5) Статус индексирования диапазона

Таблицы с информацией о базах данных:

- 1) Базы данных (таблица 4)
- 2) Ноды (таблица 5)
- 3) Бакеты (таблица 6)
- 4) Ядра Solr (таблица 7)

Таблица 4

Название колонки	Расшифровка	Тип
id	Идентификатор	Big integer
name	Название базы	Text

Таблица 5

Название колонок	Расшифровка	Тип
id	Идентификатор	Big integer
database_id	ИД базы данных	Big integer
index	Индекс	Big integer

Таблица 6

Название колонок	Расшифровка	Тип
id	Идентификатор	Big integer
node_id	ИД ноды	Big integer
index	Индекс	Big integer

Таблица 7

Название колонок	Расшифровка	Тип
id	Идентификатор	Big integer
name	Название	Text
database_id	ИД базы данных	Big integer

Таблицы заполняются вручную и содержат информацию и структуре хранилища.

Выходная информация:

Объект диапазона в формате JSON. Структура:

- 1) id – идентификатор
- 2) bucket_index – номер бакета
- 3) node_index – номер ноды

- 4) from_order_value – номер начала индексирования
- 5) to_order_value – номер конца индексирования

2.8.3. Описание логики

Сервис работает как веб-приложение и предоставляет несколько функций: заполнить базу диапазонов актуальными данными, получить первый доступный диапазон (со статусом «ошибка» или «ожидает») и установить статус.

Заполнить базу – получить словарь с размерами БД из ответа сервиса базы данных key-value. Установка статуса – получение статуса индексирования от сервиса индексирования и сохранение его в БД диапазонов.

Основной алгоритм работы:

- 1) Запуск
- 2) Запросить размеры БД key-value
- 3) Сохранить диапазоны на основе размеров
- 4) Ожидание GET запроса диапазона
- 5) Отправить первый доступный диапазон
- 6) Ожидание POST запроса со статусом индексирования
- 7) Сохранить статус диапазона

Алгоритм работы сервиса представлен на рисунках 2.11 и 2.12.

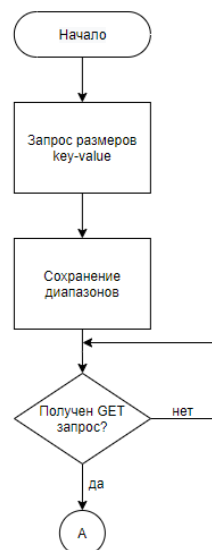


Рис. 2.11 Алгоритм работы сервиса диапазонов

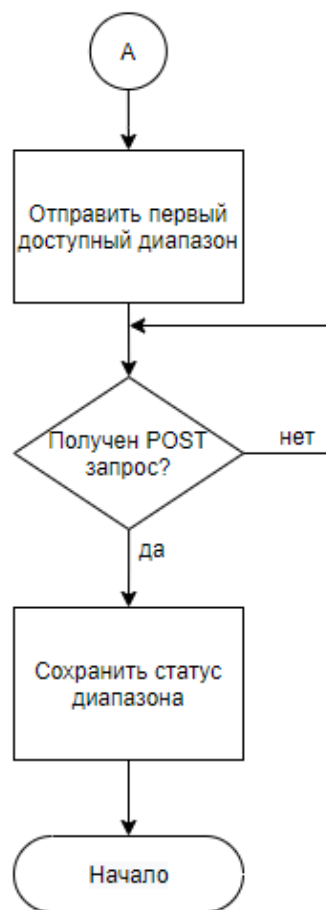


Рис. 2.12 Алгоритм работы сервиса диапазонов

2.9. Описание программы для сервиса индексирования

2.9.1. Функциональное назначение

Сервис предназначен для отправки данных на индексирования в Apache Solr по протоколу HTTP, а также для слежения за ходом индексации и получения статусов диапазонов с последующей отправкой в сервис хранения диапазонов.

2.9.2. Описание информации

Входная информация:

- 1) Объект диапазона (см. 2.7.2)
- 2) Данные из базы key-value
- 3) «Сырой» ответ от Solr со статусом

Выходная информация:

- 1) Обработанные данные из базы key-value
- 2) Обработанный статус индексирования диапазона

2.9.3. Описание логики

При запуске программы создаётся несколько потоков, читающих данные и отправляющие их в Solr. Основной алгоритм индексации описан в п. 2.5.5. После индексации каждого диапазона Solr возвращает ответ с результатом индексирования. Сообщение с результатом содержит предложение на английском языке, понятное для человека, но не подходящее для программы. Из-за этого, программа пытается найти ключевые слова в предложении, описывающие результат (например «failed», «done» «fetched»). На основе этого составляется ответ для сервиса хранения диапазонов.

Алгоритм программы:

- 1) Создание потоков «читателей» и «писателей»
- 2) Запрос доступного диапазона
- 3) Чтение данных из БД key-value
- 4) Отправка данных в Solr
- 5) Ожидание ответа Solr
- 6) Форматирование на основе ответа сообщения для сервиса хранения
- 7) Отправка ответа

Алгоритм представлен на рис. 2.13.

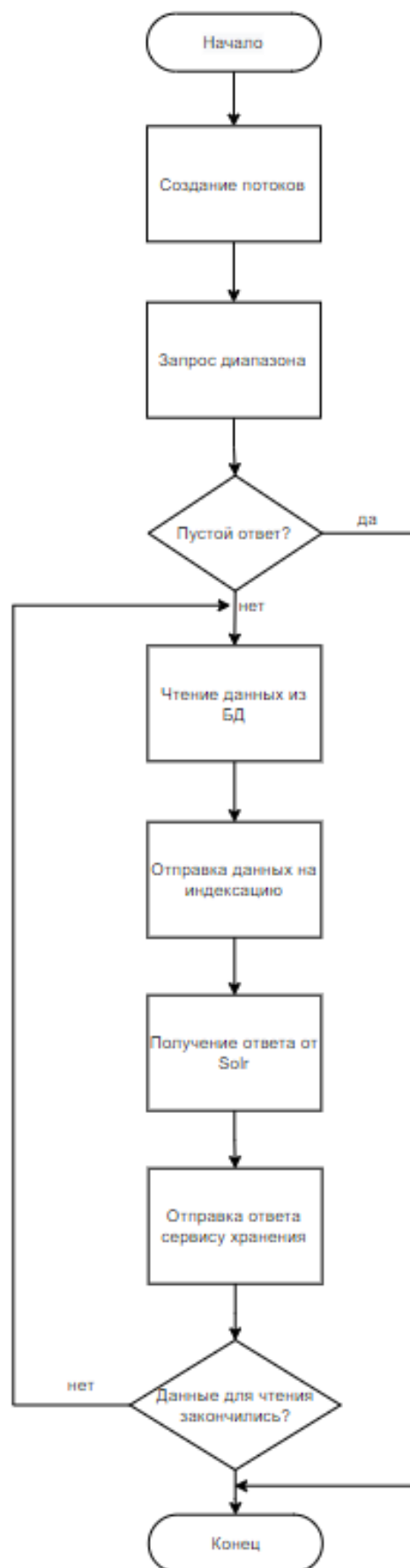


Рис.2.13 Алгоритм работы сервиса индексации

2.10. Описание контрольного примера

2.10.1. Назначение

Контрольный пример служит для проверки работоспособности системы.

В контрольном примере предусмотрена проверка следующих функциональных возможностей системы:

- 1) Получение данных из БД key-value
- 2) Преобразование данных в соответствии с форматом ядра
- 3) Индексирования данных в Apache Solr
- 4) Формирование диапазонов

2.10.2. Исходные данные

Исходными данными являются:

- 1) Записи из базы данных key-value
- 2) Таблицы с информацией о базах данных
- 3) Типы коллекций и типы данных

2.10.3. Результат расчёта

При запросе актуальных размеров БД key-value сервис возвращает JSON объект, содержащий номер бакета и его размер. В базе на момент тестов были записи только в 2 бакетах. На рисунках 2.14 и 2.15 представлен пример запроса и ответа сервиса.

```
: Request: coll_id=[1, 2], type_id=[1]  
: Response map: size=1024,
```

Рис.2.14 Логи сервиса при запросе


```
{
  "0": 5,
  "1": 1,
  "2": null,
  "3": null,
  "4": null,
  "5": null,
  "6": null,
  "7": null,
  "8": null,
  "9": null,
  "10": null,
  "11": null,
  "12": null,
  "13": null,
  "14": null
}
```

Рис.2.15 Пример ответа

Сервис хранения диапазонов, получив актуальные данные, формирует диапазоны индексирования. Пример сформированных диапазонов, полученных на основе данных из прошлого сервиса (см. рисунок 2.15) показан на рисунке 2.16. Идентификатор и индекса бакета не всегда совпадают.

	id	bucket_id	from_order_value	to_order_value
1	1	1	0	5
2	2	2	0	3
3	3	10	0	12

Рис.2.16 Полученные диапазоны

При запросе первого доступного диапазона сервис формирует объект. Пример на рисунке 2.17.

```
GET http://localhost:9002/get_range

HTTP/1.1 200
Content-Type: application/json
Transfer-Encoding: chunked
Date: Thu, 13 May 2021 06:05:48 GMT
Keep-Alive: timeout=60
Connection: keep-alive

{
  "id": 4,
  "bucket_index": 1,
  "node_index": 1,
  "from_order_value": 0,
  "to_order_value": 5
}
```

Рис.2.17 Объект диапазона

После получения диапазона сервис индексирования начинает читать партии БД и отправляет их на индексирование в Solr. Пример показан на рисунках 2.18. и 2.19.

```
: start reading partition 0
: finish reading partition 0
: partition 0 done, t=403 ms
: start reading partition 1
: finish reading partition 1
: partition 1 done, t=177 ms
: start reading partition 2
: finish reading partition 2
: partition 2 done, t=144 ms
: start reading partition 3
: finish reading partition 3
: partition 3 done, t=144 ms
: start reading partition 4
: finish reading partition 4
: partition 4 done, t=166 ms
: start reading partition 5
: finish reading partition 5
: partition 5 done, t=88 ms
: start reading partition 6
: finish reading partition 6
```

Рис.2.18 Логи чтения партий

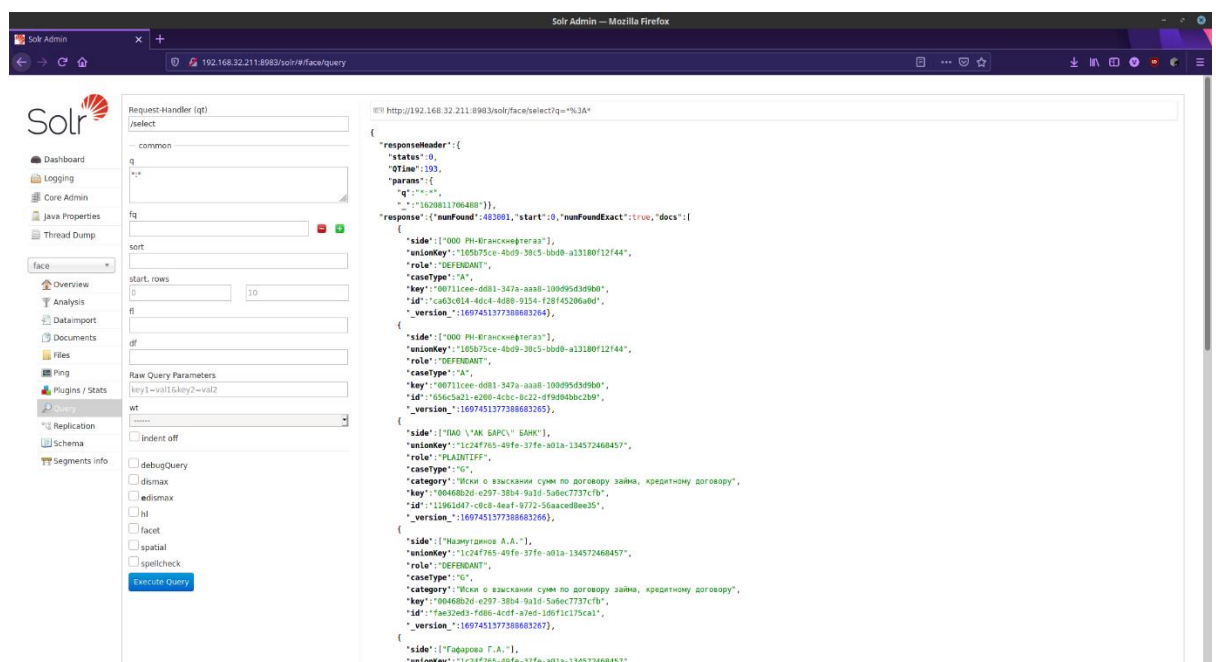


Рис.2.19 Пример проиндексированных данных

2.10.4. Результат испытания

В результате испытания системы можно сделать вывод, что данная программа выполняет все поставленные задачи.