Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)

Задание:

1) Установить виртуальную машину с ОС Ubuntu в VirtualBox.
2) Установить Elasticsearch, Neo4j, Hadoop, Spark.
3) Вручную создать JSON-файл с 20-30 JSON-документами для предметной области, указанной в варианте.
4) В Elasticsearch создать индекс с анализатором и маппингом, проиндексировать JSON-документы, разработать запросы с вложенной агрегацией, представить результаты в среде Kibana.
5) В Neo4j по данным из Elasticsearch заполнить графовую базу данных, разработать и реализовать запрос к этой БД.
6) В Spark по данным из Elasticsearch сформировать csv-файлы с таблицами и сохранить их в файловой системе HDFS, написать запрос и реализовать его в Spark, проанализировать процесс выполнения запроса с использованием монитора Spark.

Предметная область – Прокат автомобилей

Elasticsearch:

1. Типы документов (json):
Арендатор:
{index, doc_type, id, body: {id_арендатора, сведения_об_арендаторе*, id_аренды, дата_начала_аренды, продолжительность_аренды, стоимость, id_автомабиля}}
Автомобиль:
{index, doc_type, id, body: {сведения_об_автомобиле*, [диагностическая_карта_техосмотра*], [отзыв_об_автомобиле*]}}
2. Требование к анализатору:
поля, отмеченные *, разделить на слова, убрать пунктуацию с помощью токенизатора standart (русский), перевести все токены в нижний регистр, убрать токены, находящиеся в списке стоп-слов, выполнить стемминг оставшихся токенов с помощью фильтра snowball.
3. Запросы с вложенной агрегацией:
- разбить арендаторов по дате начала аренды с периодом 1 год, для каждой группы определить среднюю стоимость аренды по каждому автомобилю,
- определить число грузовых автомобилей в парке, используя поле «сведения_об_автомобиле».

Neo4j:

1. По данным из Elasticsearch заполнить графовую базу данных Арендатор(id_арендатора, сведения_об_арендаторе) - Арендовал(дата_начала_аренды, продолжительность_аренды, стоимость) - Автомобиль(id_автомобиля, сведения_об_автомобиле).
2. Разработать и реализовать запрос: найти автомобиль с максимальной суммарной стоимостью аренды.

Spark:

1. По данным из Elasticsearch сформировать csv-файлы (с внутренней схемой) таблиц «Арендатор», «Аренда», «Автомобиль» и сохранить их в файловой системе HDFS.

2. Написать запрос select: Определить арендатора и автомобиль с максимальной стоимостью аренды.

3. Реализовать этот запрос в Spark. Построить временную диаграмму его выполнения по результатам работы монитора.

					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
					Задание на курсовой проект	Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата				
Разраб.						Лист 1	Листов 11	
Руковод.								
Н. Контр.								

Индексация документов Elasticsearch

Фрагмент маппинга для индекса арендатор

```
ArendatorMapping = {
    "properties":{
        "arendator_id": {
            "type": "text",
            "fielddata": True
        },
        "arendator_data": {
            "type": "text",

"analyzer":"analitic_for_ru",

"search_analyzer":"analitic_for_ru",
            "fielddata": True
        },
        "arenda_id":  {
            "type": "text",
            "fielddata": True
        },
        "date_of_arenda":  {
            "type":   "date",
            "format": "yyyy-MM-dd"
        },
        "numb_days_of_arend": {
            "type": "integer"
        },
        "price": {
            "type": "integer"
        },
        "car_id": {
            "type": "text",
            "fielddata": True
        }
    }
}
```

Маппинг для индекса автомобиль

```
CarMapping = {
    "properties":{
        "car_data": {
            "type": "text",

"analyzer":"analitic_for_ru",

"search_analyzer":"analitic_for_ru",
            "fielddata": True
        "diagnostic_card":{
            "type": "text",

"analyzer":"analitic_for_ru",

"search_analyzer":"analitic_for_ru",
            "fielddata": True
        },
        "car_reviews":{
            "type": "text",

"analyzer":"analitic_for_ru",

"search_analyzer":"analitic_for_ru",
            "fielddata": True
 }   }  }
```

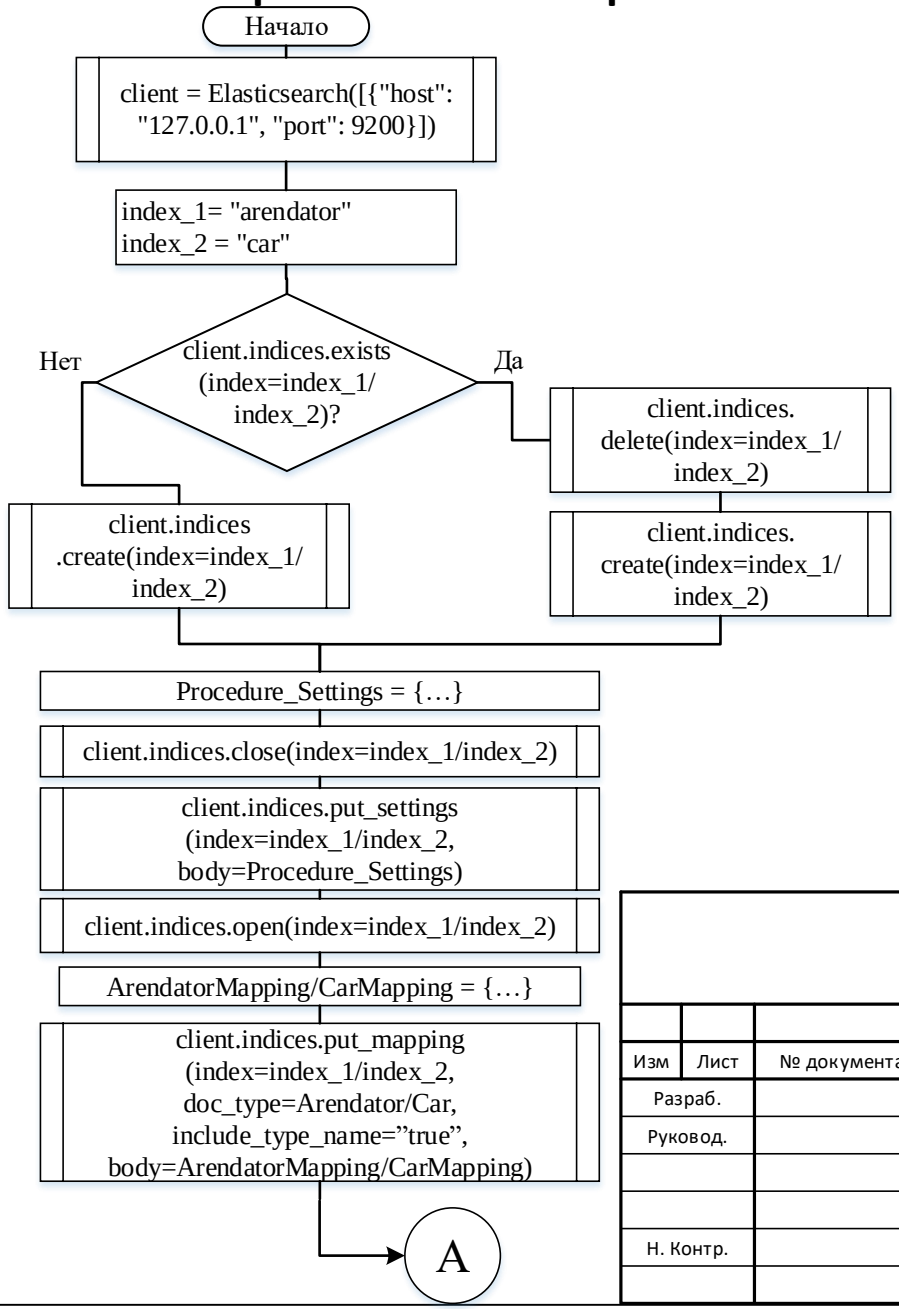
Анализатор для индексов

```
"analysis" : {
    "filter": {
        "russian_stop_words": {
            "type": "stop",
            "stopwords": "_russian_"
        },
        "filter_ru_sn": {
            "type": "snowball",
            "language":"Russian"
        }
    },
    "analyzer":
    {
        "analitic_for_ru":
        {
            "type": "custom",
            "tokenizer": "standard",
            "filter": [
                "lowercase",
                "russian_stop_words",
                "filter_ru_sn"
            ]
        }
    }
}
```

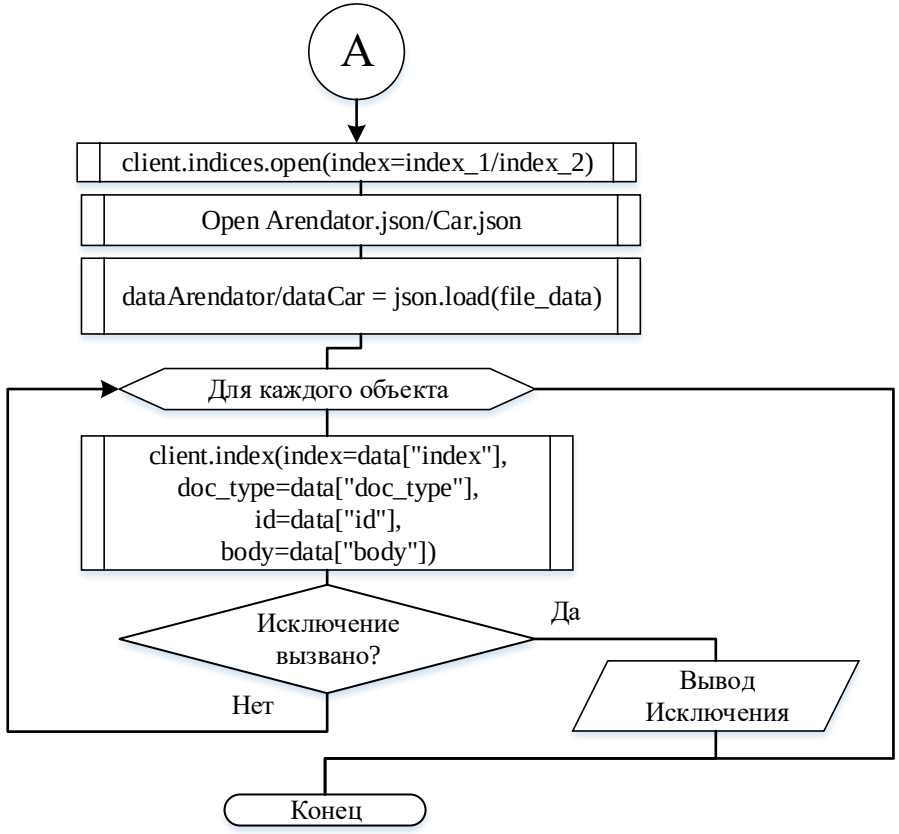
					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
Изм	Лист	№ документа	Подпись	Дата	Индексация документов Elasticsearch. Маппинг и анализатор	Литер.	Масса	Масштаб
Разраб.								
Руковод.						Лист 2	Листов 11	
Н. Контр.								

Индексация документов Elasticsearch

Алгоритм добавления маппинга и настройки анализатора



Алгоритм индексации



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)
Изм	Лист	№ документа	Подпись	Дата	Индексация документов Elasticsearch. Алгоритмы индексациии
Разраб.					
Руковод.					
Н. Контр.					

Литер.	Масса	Масштаб
Лист 3	Листов 11	

Elasticsearch. Запросы

Первый запрос: разбить арендаторов по дате начала аренды с периодом 1 год, для каждой группы определить среднюю стоимость аренды по каждому автомобилю.

Второй запрос: определить число грузовых автомобилей в парке, используя поле «сведения_об_автомобиле».

```
"aggregations" : {
  "year_period" : {
    "buckets" : [
      {
        "key_as_string" : "2019-01-01",
        "key" : 1546300800000,
        "doc_count" : 1,
        "Arendator" : {
          "doc_count_error_upper_bound" : 0,
          "sum_other_doc_count" : 0,
          "buckets" : [
            {
              "key" : "cr03",
              "doc_count" : 1,
              "mean_arenda_price" : {
                "value" : 7000.0
              }
            }
          ]
        }
      },
      {
        "key_as_string" : "2020-01-01",
        "key" : 1577836800000,
        "doc_count" : 7,
        "Arendator" : {
          "doc_count_error_upper_bound" : 0,
          "sum_other_doc_count" : 0,
          "buckets" : [
            {
              "key" : "cr01",
              "doc_count" : 1,
              "mean_arenda_price" : {
                "value" : 1800.0
              }
            },
            {
              "key" : "cr05",
              "doc_count" : 2,
              "mean_arenda_price" : {
                "value" : 2000.0
              }
```

```
GET arendator/_search
{
"size": 0,
"aggregations": {
  "year_period": {
      "date_histogram": {
      "field": "date_of_arenda",
      "calendar_interval": "year",
      "format": "yyyy-MM-dd"
      },
   "aggregations": {
      "Arendator": {
       "terms": {
            "field": "car_id",
          "order": {
              "_key": "asc"
          }
       },
      "aggregations": {
            "mean_arenda_price": {
            "avg": {
                "field": "price"
            }
        }
      }
} } } } }
```

```
"hits" : {
  "total" : {
    "value" : 15,
    "relation" : "eq"
  },
  "max_score" : 0.7457469,
  "hits" : [
    {
      "_index" : "car",
      "_type" : "Car",
      "_id" : "CR04",
      "_score" : 0.7457469,
      "_source" : {
        "car_data" : "Грузовик:
          Chevrolet Silverado, 2017г"
      }
    },
    {
      "_index" : "car",
      "_type" : "Car",
      "_id" : "CR06",
      "_score" : 0.7457469,
      "_source" : {
        "car_data" : "Грузовик: Ford
          Ranger, 2015г"
      }
    },
    {
      "_index" : "car",
      "_type" : "Car",
      "_id" : "CR08",
      "_score" : 0.7457469,
      "_source" : {
        "car_data" : "Грузовик: Toyota
          Tacoma, 2016г"
      }
```

```
GET car/_search
{
  "query": {
    "match_phrase": {
      "car_data": "грузовик"
    }
  },
  "_source": ["car_data"],
  "aggs":{
    "count": {
          "value_count": {
                "field": "_id"
          }
      }
  }
}
```

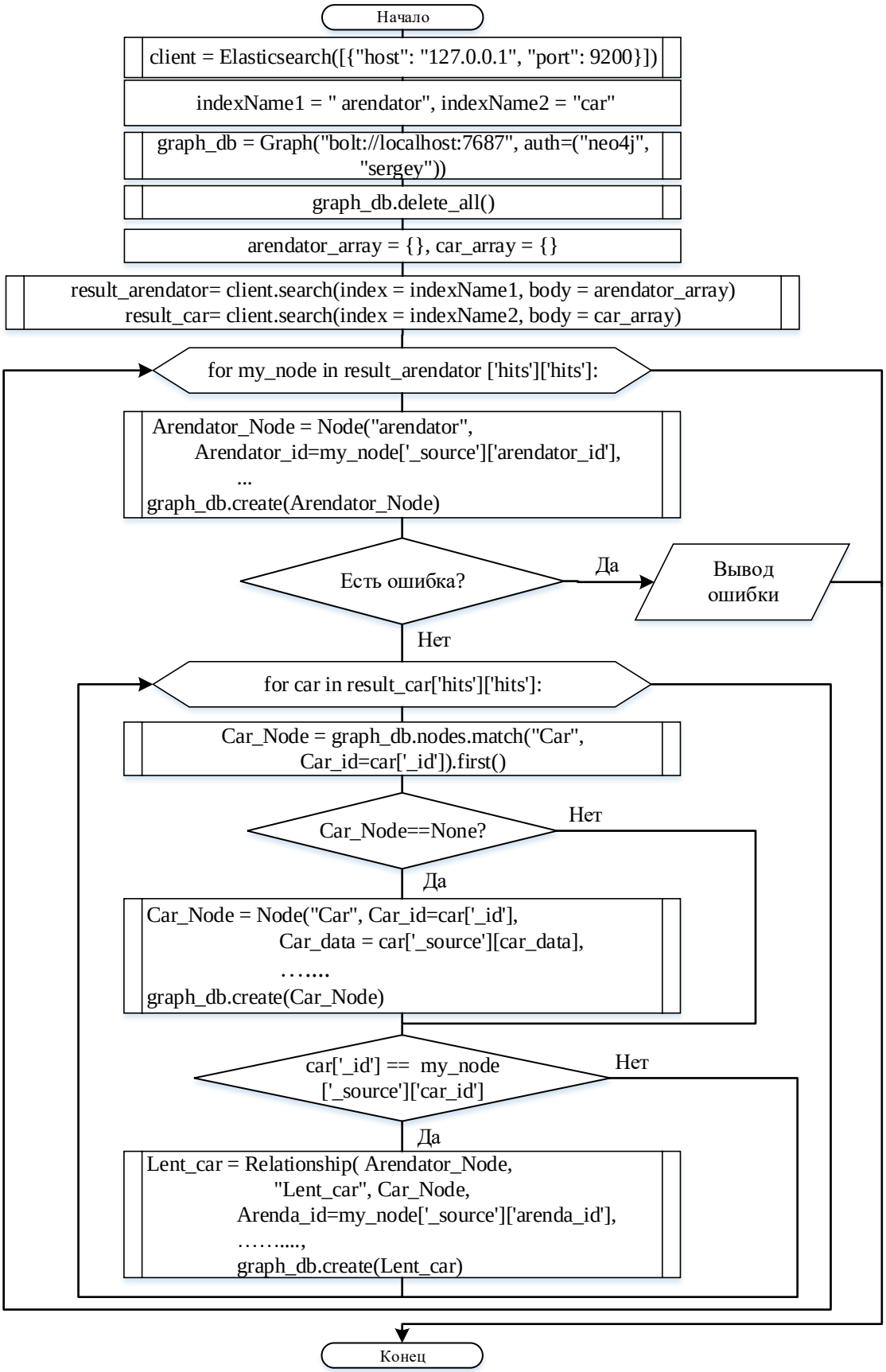
Изм	Лист	№ документа	Подпись	Дата
Разраб.				
Руковод.				
Н. Контр.				

Разработка макета аналитической системы на основе баз данных NoSQL (вариант №21)

Elasticsearch. Запросы

Литер.	Масса	Масштаб

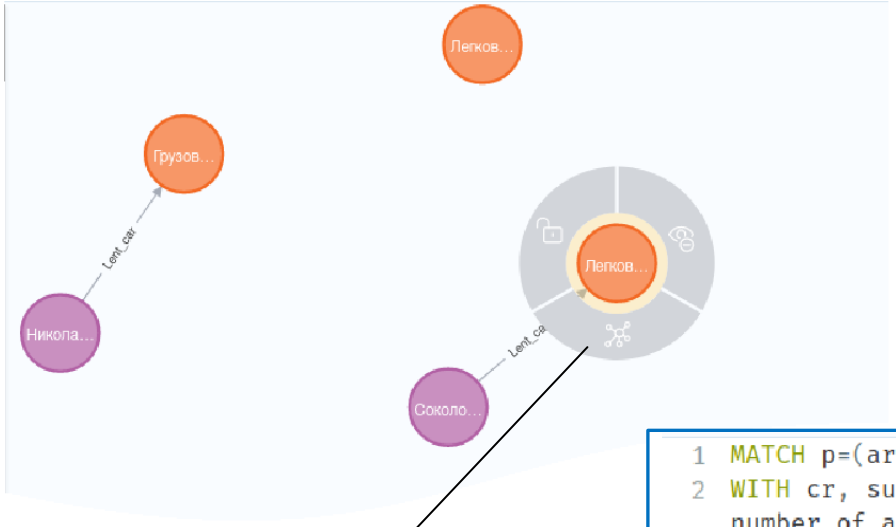
Лист 4 | Листов 11

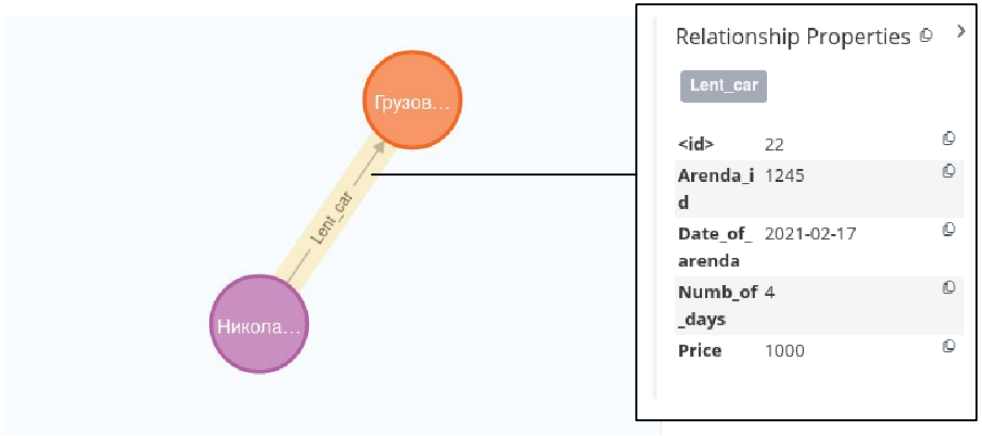


					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
						Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата	Neo4j. Алгоритм создания и заполнения графовой БД			
Разраб.						Лист 5	Листов 11	
Руковод.								
Н. Контр.								

Neo4j. Запрос

Связь «Арендатор арендовал автомобиль» = Lent_car





Node Properties

Car

<id>	8
Car_data	Легковой автомобиль: Volkswagen Golf, 2015г
Car_id	CR25
Car_reviews	[Экономичный и надежный автомобиль,Частые поломки, требуется ремонт,Комфортная и практичная машина для ежедневной езды]
Diagnostic_card	[Проблемы с системой впрыска

Запрос: найти автомобиль с максимальной суммарной стоимостью аренды.

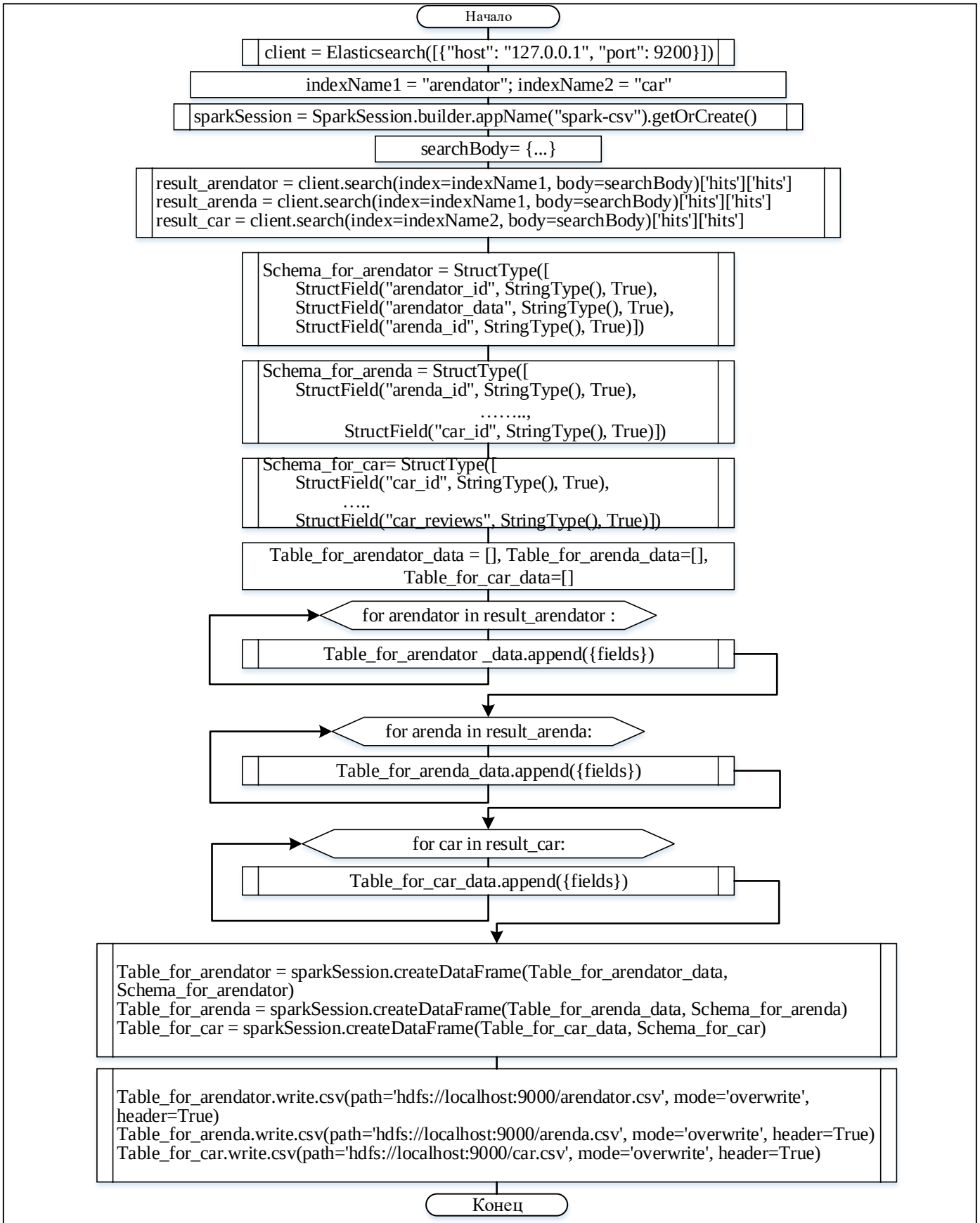
```
1 MATCH p=(ar:arendator)-[r:Lent_car]→(cr:Car)
2 WITH cr, sum(toInteger(r.Price)) as car_sum, count(r) as
  number_of_arendators
3 ORDER BY car_sum desc
4 RETURN cr.Car_id, cr.Car_data, car_sum, number_of_arendators
5 LIMIT 1
```

Table

Text

"cr.Car_id"	"cr.Car_data"	"car_sum"	"number_of_arendators"
"CR03"	"Легковой автомобиль: Toyota Camry, 2018г"	8800	2

					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)	
Изм	Лист	№ документа	Подпись	Дата	Neo4j. Запрос	Литер. / Масса / Масштаб
Разраб.						
Руковод.						Лист 6 / Листов 11
Н. Контр.						



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
						Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата	Spark. Алгоритм создания CSV-файлов с таблицами			
Разраб.						Лист 7	Листов 11	
Руковод.								
Н. Контр.								

Spark. Запрос

Запрос: определить арендатора и автомобиль с максимальной стоимостью аренды.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-from pyspark.sql import SparkSession

from pyspark.sql import *

sparkSession=SparkSession.builder.appName("Python Spark SQL basic example").config("spark.sql.shuffle.partitions","10").getOrCreate()

Arendator_Table = sparkSession.read.load(path='hdfs://localhost:9000/arendator.csv', format='csv', sep=',', inferSchema="true", header="true")
Arenda_Table = sparkSession.read.load(path='hdfs://localhost:9000/arenda.csv', format='csv', sep=',', inferSchema="true", header="true")
Car_Table = sparkSession.read.load(path='hdfs://localhost:9000/car.csv', format='csv', sep=',', inferSchema="true", header="true")

Arendator_Table.registerTempTable("arendator")
Arenda_Table.registerTempTable("arenda")
Car_Table.registerTempTable("car")


df = sparkSession.sql("

SELECT arendator.arendator_data, car.car_data, arenda.price
        FROM arendator, arenda, car
        WHERE (arenda.price = (SELECT MAX(price) FROM arenda))
                and (arendator.arenda_id = arenda.arenda_id)
                and (arenda.car_id = car.car_id)

        ").show()

input('Ctrl C')
```

Вывод результата запроса

```
+-------------------------+----------------------------------------+-----+
|arendator_data           |car_data                                |price|
+-------------------------+----------------------------------------+-----+
|Шаповалов Игорь Максимович|Легковой автомобиль: Toyota Camry, 2018г|7000 |
+-------------------------+----------------------------------------+-----+

Ctrl C
```

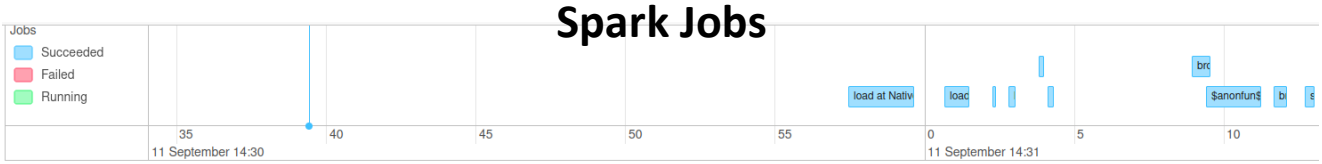
					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
Изм	Лист	№ документа	Подпись	Дата	Spark. Запрос	Литер.	Масса	Масштаб
Разраб.								
Руковод.						Лист 8	Листов 11	
Н. Контр.								

Spark. Мониторинг выполнения запроса

Выполненные SQL-запросы

ID ▾	Description	Submitted	Duration
6	showString at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:08	5 s
5	createOrReplaceTempView at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:05	6 ms
4	createOrReplaceTempView at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:05	1 ms
3	createOrReplaceTempView at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:05	11 ms
2	load at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:03	0.4 s
1	load at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:31:02	0.3 s
0	load at NativeMethodAccessorImpl.java:0 +details	2023/09/11 14:30:53	7 s

Spark Jobs

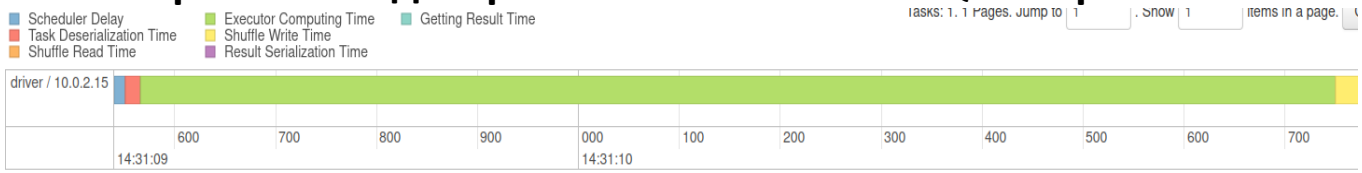


▾ Completed Jobs (10)

Page: 1 — 1 Pages. Jump to 1 . Show 100 items in a page. Go

Job Id (Job Group) ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
9	showString at NativeMethodAccessorImpl.java:0 showString at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:12	0.3 s	1/1	1/1
8 (86c64aa5-4089-4112-92bd-2d66d93d8297)	broadcast exchange (runId 86c64aa5-4089-4112-92bd-2d66d93d8297) $anonfun$withThreadLocalCaptured$1 at FutureTask.java:266	2023/09/11 14:31:11	0.5 s	1/1	1/1
7	$anonfun$withThreadLocalCaptured$1 at FutureTask.java:266 $anonfun$withThreadLocalCaptured$1 at FutureTask.java:266	2023/09/11 14:31:09	2 s	2/2	2/2
6 (8776fc04-2d51-49be-acab-ba2e2b8afdee)	broadcast exchange (runId 8776fc04-2d51-49be-acab-ba2e2b8afdee) $anonfun$withThreadLocalCaptured$1 at FutureTask.java:266	2023/09/11 14:31:08	0.6 s	1/1	1/1
5	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:04	0.2 s	1/1	1/1
4	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:03	0.2 s	1/1	1/1
3	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:02	0.2 s	1/1	1/1
2	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:02	0.1 s	1/1	1/1
1	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:31:00	0.8 s	1/1	1/1
0	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/09/11 14:30:57	2 s	1/1	1/1

Временная диаграмма выполнения SQL-Запроса



Summary Metrics for 1 Completed Tasks

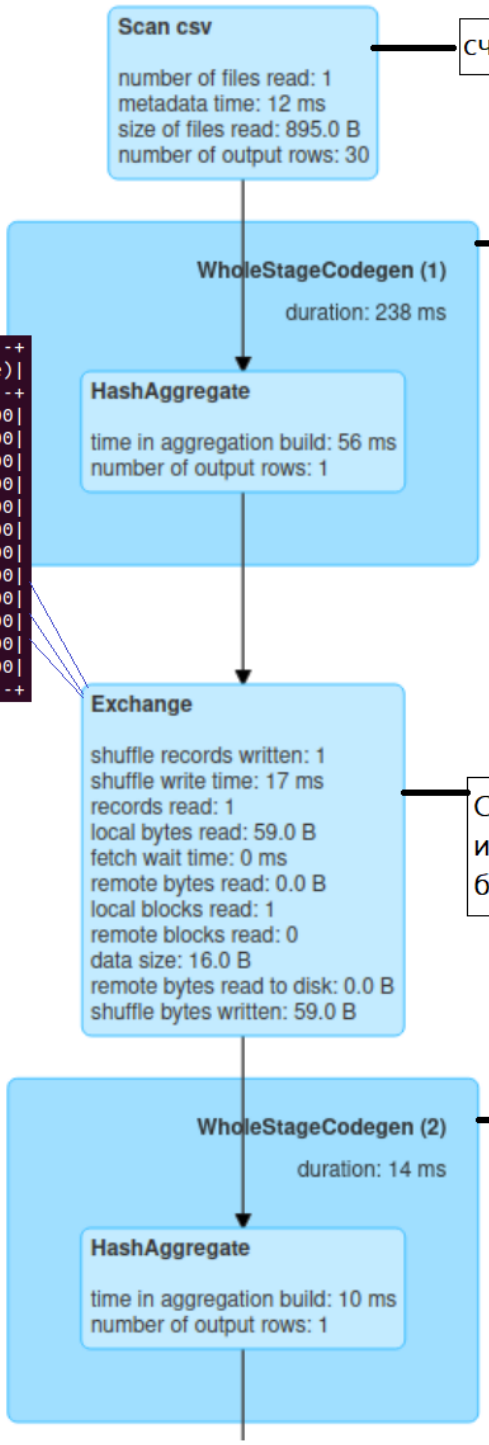
Metric	Min	25th percentile	Median	75th percentile	Max
Duration	1 s	1 s	1 s	1 s	1 s
GC Time	0.6 s	0.6 s	0.6 s	0.6 s	0.6 s
Input Size / Records	895 B / 30	895 B / 30	895 B / 30	895 B / 30	895 B / 30
Shuffle Write Size / Records	59 B / 1	59 B / 1	59 B / 1	59 B / 1	59 B / 1

					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)		
						Литер.	Масса / Масштаб
Изм	Лист	№ документа	Подпись	Дата	Spark. Мониторинг выполнения запроса		
Разраб.						Лист 9	Листов 11
Руковод.							
Н. Контр.							

Spark. DAG SQL-запроса (1 часть)

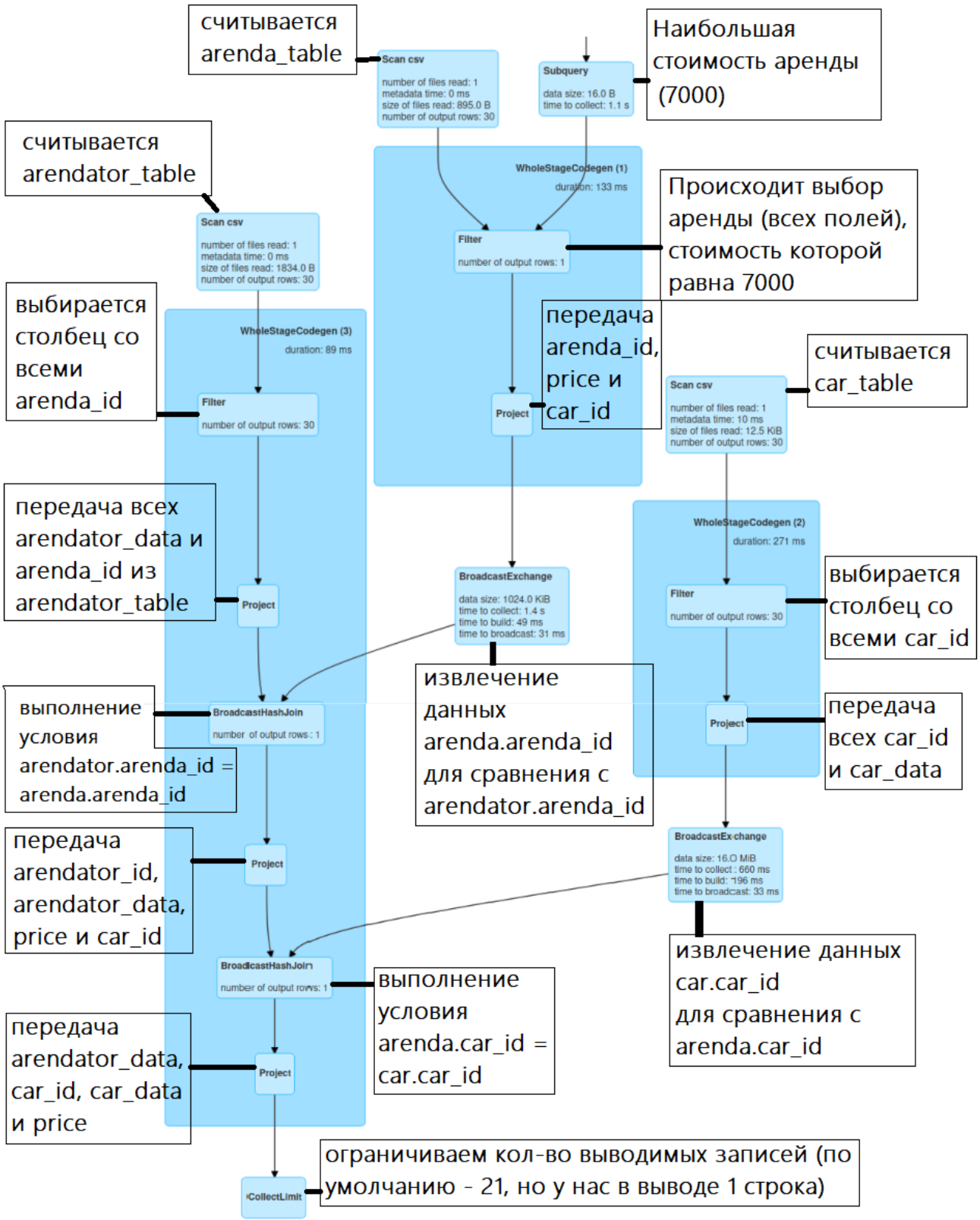
Если бы было разделение, то в каждом блоке получилось бы свое максимальное число, как в этой таблице

numb_days_of_arend	max(price)
12	3500
1	500
6	1800
3	900
5	1400
9	2500
4	1000
8	2200
7	1800
10	7000
11	3200
2	600



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
					Spark. DAG SQL-Запроса (1 часть)	Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата				
Разраб.						Лист 10	Листов 11	
Руковод.								
Н. Контр.								

Spark. DAG SQL-запроса (2 часть)



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 21)			
						Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата	Spark. DAG SQL-Запроса (2 часть)			
Разраб.						Лист 11	Листов 11	
Руковод.								
Н. Контр.								