Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 6)

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 – Салон красоты

Elasticsearch:

1. Типы JSON-документов:
Процедура:
{index, doc_type, id, body: {сведения_о_процедуре*, стоимость}}
Пациент:
{index, doc_type, id, body: {id_пациента, персональные_данные*, путёвка, дата_прибытия, продолжительность_прибывания, [диагноз*], [id_процедуры]}}
2. Требования к анализатору: для полей, отмеченных символом *, должен быть разработан анализатор со следующими требованиями: разделить текст на слова, убрать пунктуацию с помощью токенизатора standard (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 пациентов по дате прибытия с периодом 1 год, для каждой «корзины» определить количество пациентов, прошедших каждую процедуру;
– определить стоимость процедуры по заданным ключевым словам.

Neo4j:

1. По данным из Elasticsearch заполнить графовую базу данных Пациент(id_пациента, дата_прибытия, персональные_данные) -Прошёл(стоимость) - Процедура(id_процедуры).
2. Разработать и реализовать запрос: найти процедуру с максимальной суммарной стоимостью.

Spark:

1. По данным из Elasticsearch сформировать csv-файлы (с внутренней схемой) таблиц «Пациент», «Назначение», «Процедура» и сохранить их в файловой системе HDFS.
2. Написать запрос select: определить суммарное число назначений по каждой процедуре.
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 № 24)			
					Задание на курсовой проект	Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата				
Разраб.								
Руковод.						Лист 1	Листов 10	
Н. Контр.								

Индексация документов Elasticsearch

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"
            ]
        }
    }
}
```

Маппинг для индекса процедура

```
ProcedureMapping = {
    "properties":{
        "name": {
                "type": "text",
                "fielddata": True,
        "analyzer":"analitic_for_ru",
        "search_analyzer":"analitic_for
_ru"
        },

        "description": {
                "type": "text",
                "fielddata": True,

        "analyzer":"analitic_for_ru",
        "search_analyzer":"analitic_for
_ru"
        },
        "price": {
                "type": "integer"
        }
    }
}
```

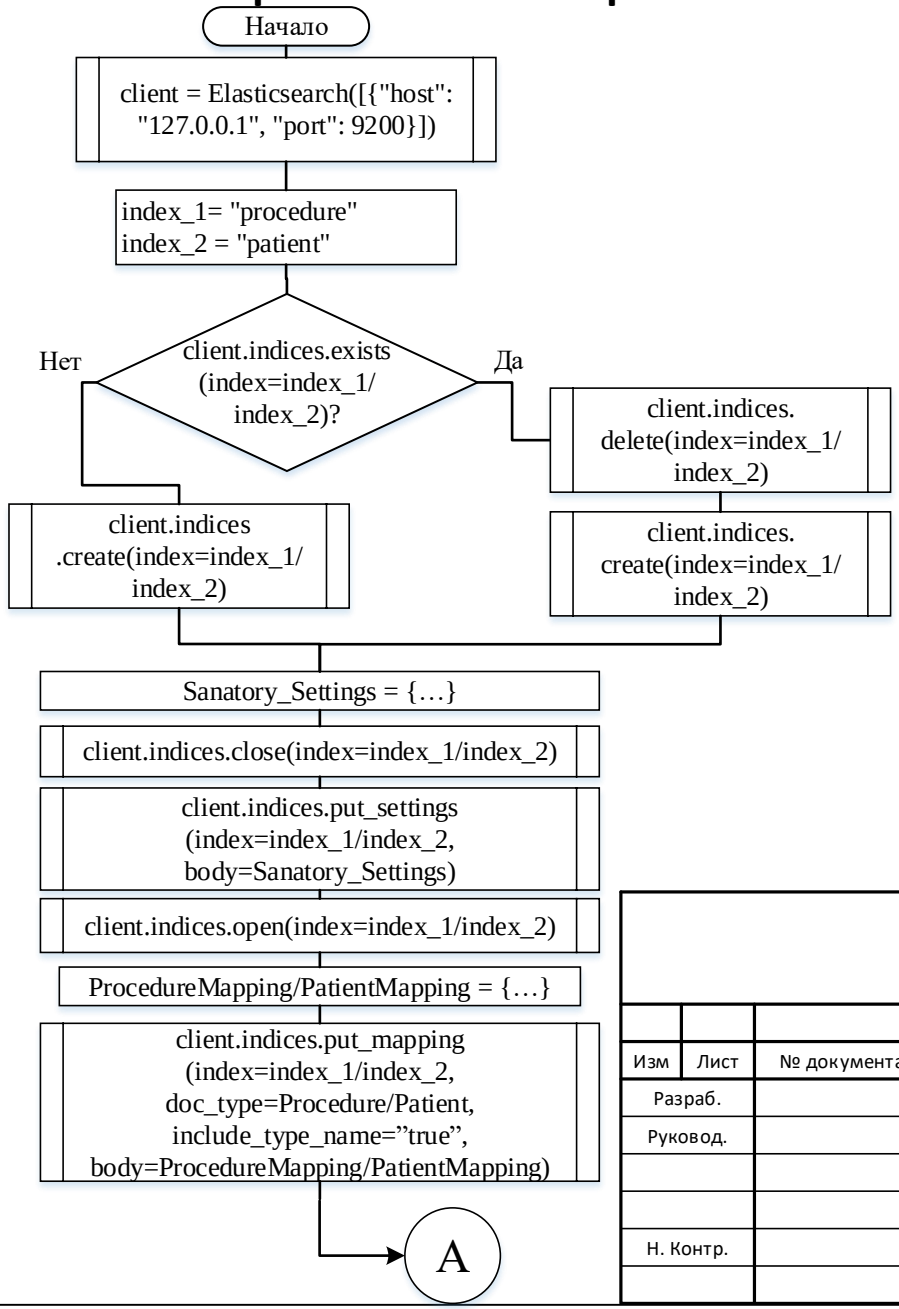
Фрагмент маппинга для индекса пациент

```
PatientMapping = {
    "properties":{
        "patient_id": {
            "type": "text",
            "fielddata": True
        },

"personal_data": {
            "type": "text",
            "analyzer":"analitic_for_ru",

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

"precedures_id": {
            "type": "text",
            "fielddata": True
        }
    }
}
```

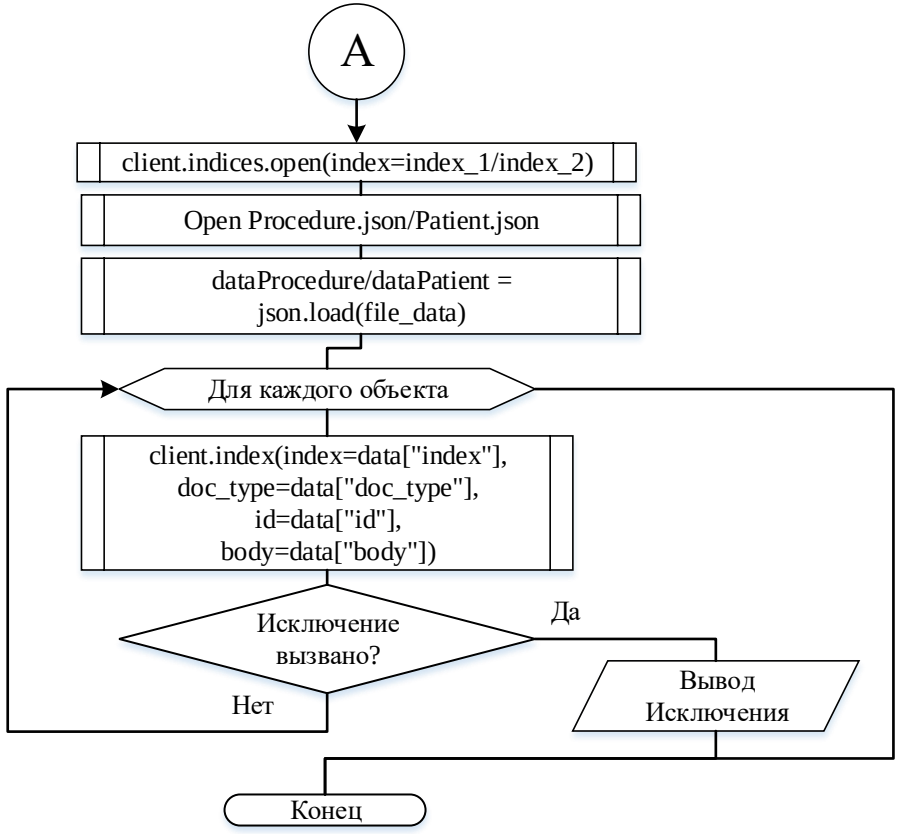
					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)	
Изм	Лист	№ документа	Подпись	Дата	Индексация документов Elasticsearch. Маппинг и анализатор	Литер. / Масса / Масштаб
Разраб.						Лист 2 / Листов 10
Руковод.						
Н. Контр.						

Индексация документов Elasticsearch

Алгоритм добавления маппинга и настройки анализатора



Алгоритм индексации



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)
Изм	Лист	№ документа	Подпись	Дата	Индексация документов Elasticsearch. Алгоритмы индексациии
Разраб.					Литер. / Масса / Масштаб
Руковод.					Лист 3 / Листов 10
Н. Контр.					

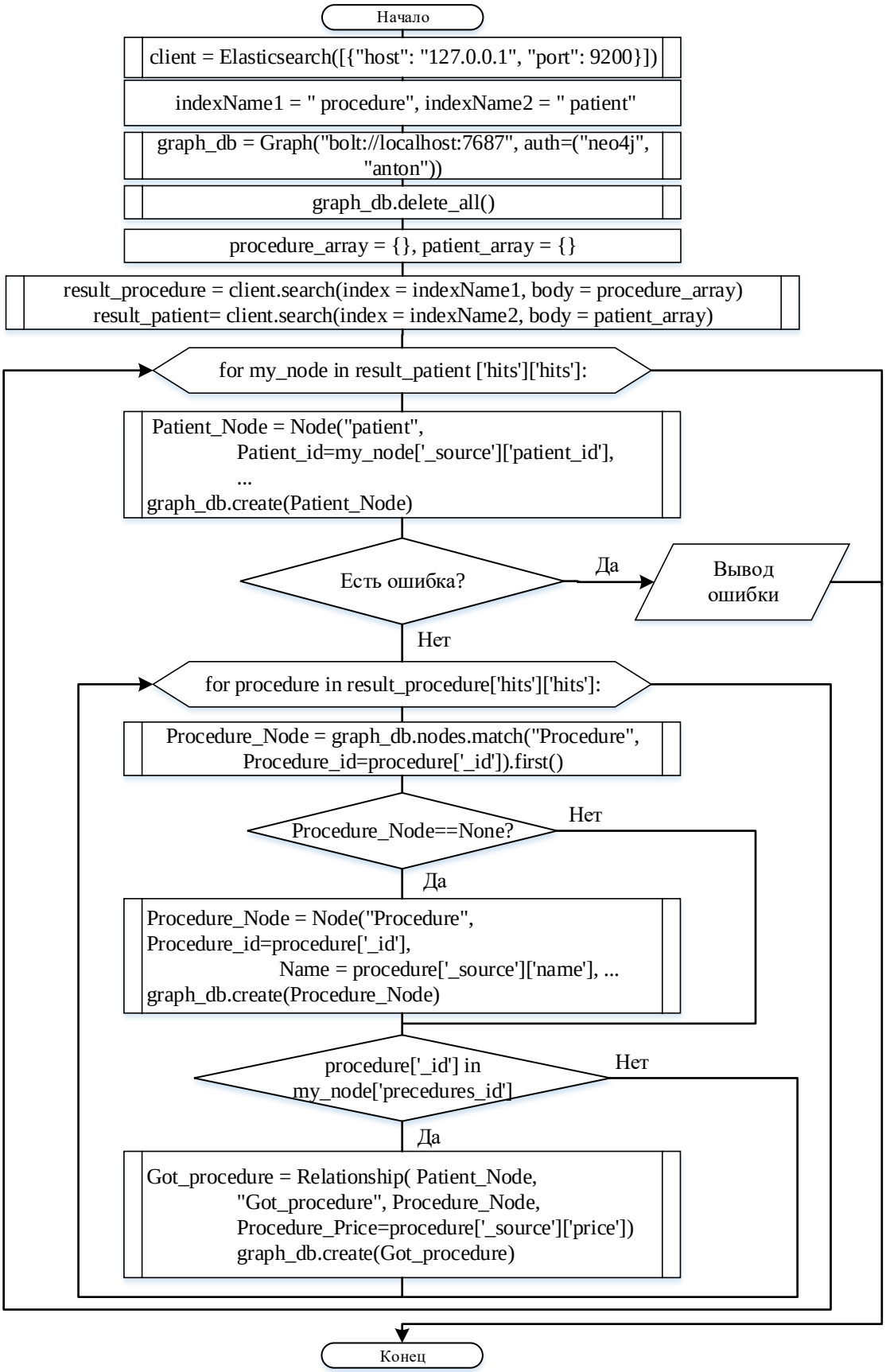
Elasticsearch. Запросы

Первый запрос: разбить пациентов по дате прибытия с периодом 1 год, для каждой «корзины» определить количество пациентов, прошедших каждую процедуру.

Второй запрос: определить стоимость процедуры по заданным ключевым словам.

```
GET patient/_search
{
"size": 0,
"aggregations": {
  "year_period": {
      "date_histogram": {
      "field": "date_of_arrival",
      "calendar_interval": "year",
      "format": "yyyy-MM-dd"
      },
  "aggregations": {
      "Procedure": {
       "terms": {
            "field": "precedures_id",
           "order": {
                "_key": "asc"
          }
       },
      "aggregations": {
           "number_of_patients": {
           "value_count": {
                "field": "patient_id"
            }
        }
      }
}
}
}
}
}
```

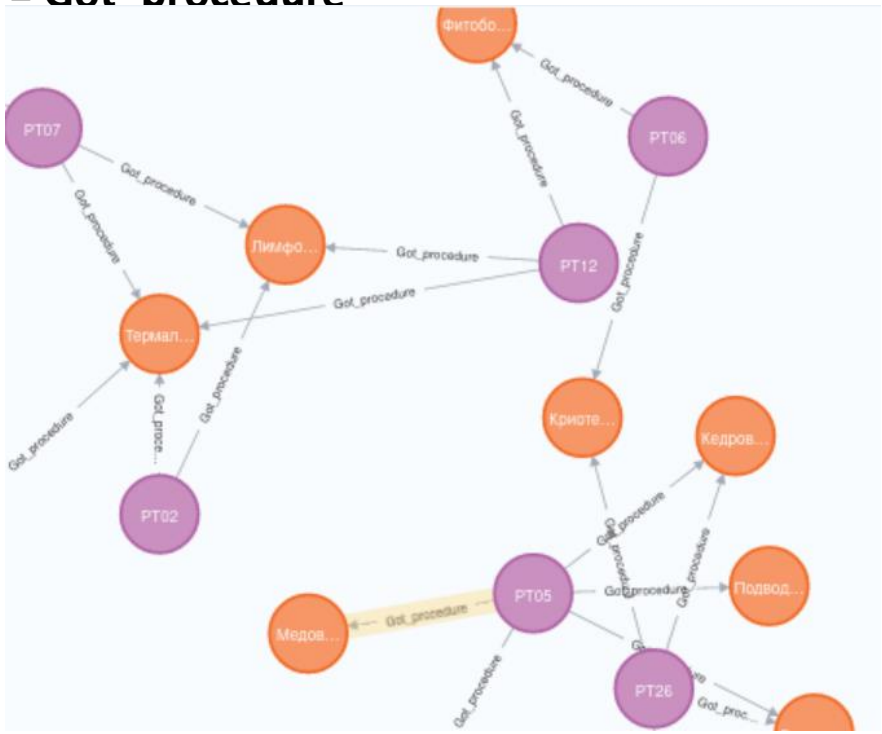
```
GET procedure/_search
{
 "query": {
 "simple_query_string": {
  "query": "травы"
 }
 },
 "_source": [
  "name",
  "price"
 ],
 "highlight": {
 "fields": {
  "description": {}
 }
 }
}
```



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 6)	
						Литер. / Масса / Масштаб
Изм	Лист	№ документа	Подпись	Дата	Neo4j. Алгоритм создания и заполнения графовой БД	
Разраб.						Лист 5 / Листов 10
Руковод.						
Н. Контр.						

Neo4j. Запрос

Связь «пациент получил процедуру» =
= Got_procedure



Запрос: найти процедуру с максимальной суммарной стоимостью.

```
1 MATCH p=(pct:patient)-[r:Got_procedure]→(proc:Procedure)
2 WITH proc, sum(toInteger(r.Procedure_Price)) as procedure_sum, count(r) as
  number_of_client
3 ORDER BY procedure_sum desc
4 RETURN proc.Name, procedure_sum, proc.Price, number_of_client, proc
5 LIMIT 1
```

proc.Name	procedure_sum	proc.Price	number_of_client	proc
"Термальные ванны"	80000	"8000"	10	


```
{
  "identity": 1,
  "labels": [
    "Procedure"
  ],
  "properties": {
"Description": "Процедура, основанная на принятии термальных ванн с использованием теплой или горячей воды с добавлением различных минералов и микроэлементов, что способствует расслаблению
```


patient

<id>	56
Arrival_date	2018-07-17
Diagnosis	[Депрессия,Панические атаки,Психосоматические расстройства]
Numb_of_days	11
Patient_id	PT09
Patient_personal_data	Смирнов Алексей Владимирович
Procedures_id	[PR01,PR03,PR30]
Voucher	4321567

Node Properties ©

Procedure

<id>	36
Description	Процедура, основанная на использовании лечебных трав и растений для оздоровления организма, укрепления иммунной системы и снятия стресса.
Name	Терапия травами
Price	7000
Procedure_id	PR18

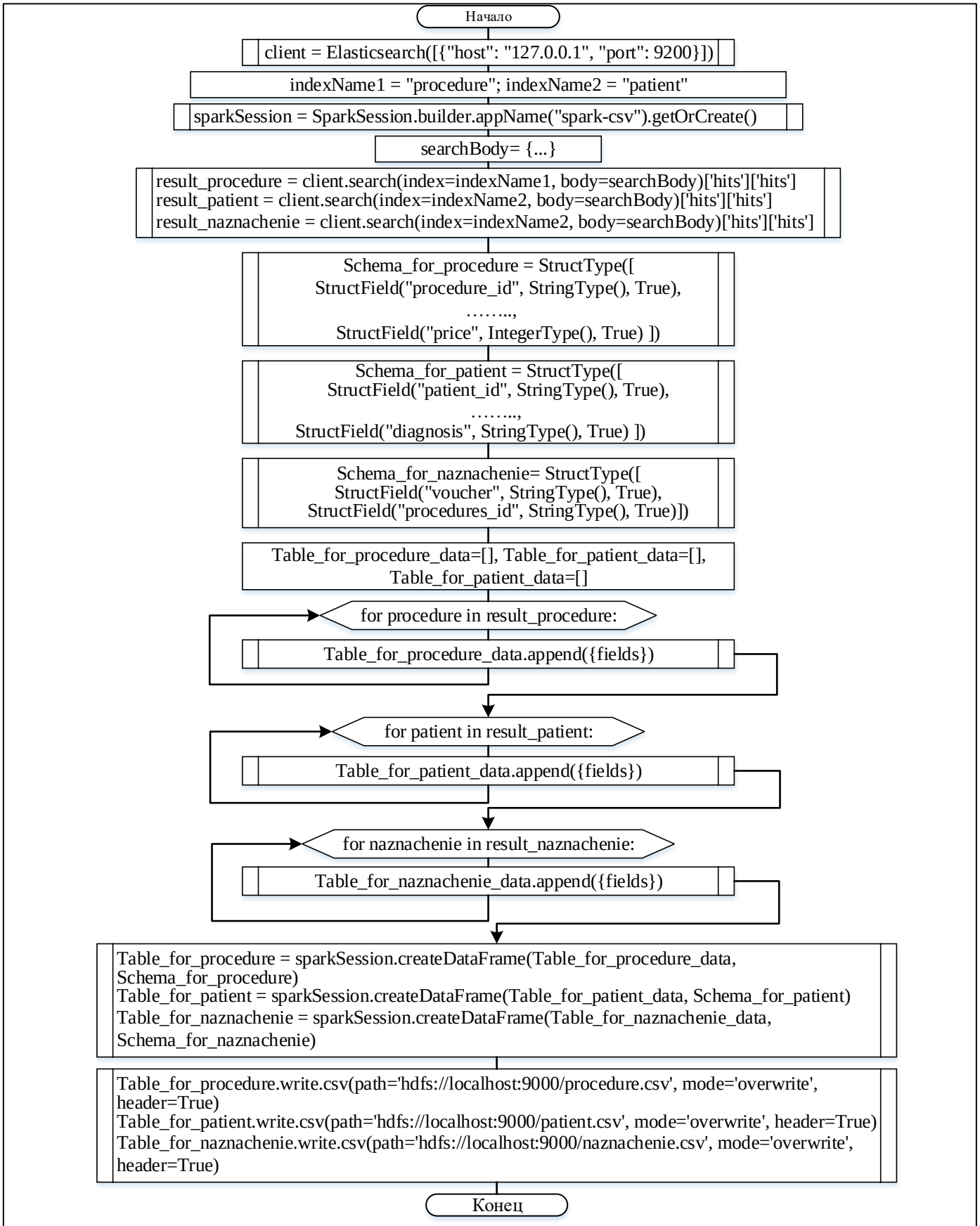
Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)

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

Neo4j. Запрос

Литер.	Масса	Масштаб

Лист 6 | Листов 10



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)			
						Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата	Spark. Алгоритм создания CSV-файлов с таблицами			
Разраб.						Лист 7	Листов 10	
Руковод.								
Н. Контр.								

Spark. Запрос

Запрос: определить число выполненных процедур каждым специалистом

ID_OF_PROCEDURE	NUMBER_OF_NAZNACH
PR01	9
PR02	10
PR03	7
PR04	3
PR05	3
PR06	5
PR07	7
PR08	9
PR09	4
PR10	1
PR11	6
PR12	6
PR13	4
PR14	5
PR15	3
PR16	1
PR17	4
PR18	7
PR19	5
PR20	4
PR22	4
PR23	5
PR24	5
PR25	4
PR26	5
PR27	3
PR28	5
PR29	6
PR30	11

```
#!/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","100").getOrCreate()

Procedure_Table = sparkSession.read.load(path='hdfs://localhost:9000/procedure.csv', format='csv', sep=',',
        inferSchema="true", header="true")
Master_Table = sparkSession.read.load(path='hdfs://localhost:9000/patient.csv', format='csv', sep=',',
        inferSchema="true", header="true")

Procedure_Table.registerTempTable("procedure")
Patient_Table.registerTempTable("patient")

df = sparkSession.sql("SELECT ID_OF_PROCEDURE, COUNT(ID_OF_PROCEDURE) as NUMBER_OF_NAZNACH
FROM (SELECT procedure.procedure_id as ID_OF_PROCEDURE, naznachenie.procedures_id LIKE concat(concat('%',
                procedure.procedure_id),'%') as CHECK,  FROM procedure, naznachenie)
WHERE check=true
GROUP BY ID_OF_PROCEDURE
ORDER BY ID_OF_PROCEDURE").show(30)

input('Ctrl C')
```

					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)			
					Spark. Запрос	Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата				
Разраб.								
Руковод.						Лист 8	Листов 10	
Н. Контр.								

Spark. Мониторинг выполнения запроса

Выполненные SQL-запросы

ID ▾	Description	Submitted	Duration	Job IDs
4	showString at NativeMethodAccessorImpl.java:0 +details	2023/05/06 20:34:58	14 s	[4][5]
3	createOrReplaceTempView at NativeMethodAccessorImpl.java:0 +details	2023/05/06 20:34:57	2 ms	
2	createOrReplaceTempView at NativeMethodAccessorImpl.java:0 +details	2023/05/06 20:34:57	10 ms	
1	load at NativeMethodAccessorImpl.java:0 +details	2023/05/06 20:34:56	0.3 s	[2]
0	load at NativeMethodAccessorImpl.java:0 +details	2023/05/06 20:34:48	6 s	[0]

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

Spark Jobs

Job Id (Job Group) ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
5	showString at NativeMethodAccessorImpl.java:0 showString at NativeMethodAccessorImpl.java:0	2023/05/06 20:34:59	12 s	2/2	201/201
4 (466ecc6c-e0a3-412c-ab71-776946e476fc)	broadcast exchange (runId 466ecc6c-e0a3-412c-ab71-776946e476fc) $anonfun$withThreadLocalCaptured$1 at FutureTask.java:266	2023/05/06 20:34:59	0.3 s	1/1	1/1
3	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/05/06 20:34:56	0.2 s	1/1	1/1
2	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/05/06 20:34:56	0.2 s	1/1	1/1
1	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/05/06 20:34:55	0.5 s	1/1	1/1
0	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2023/05/06 20:34:52	2 s	1/1	1/1

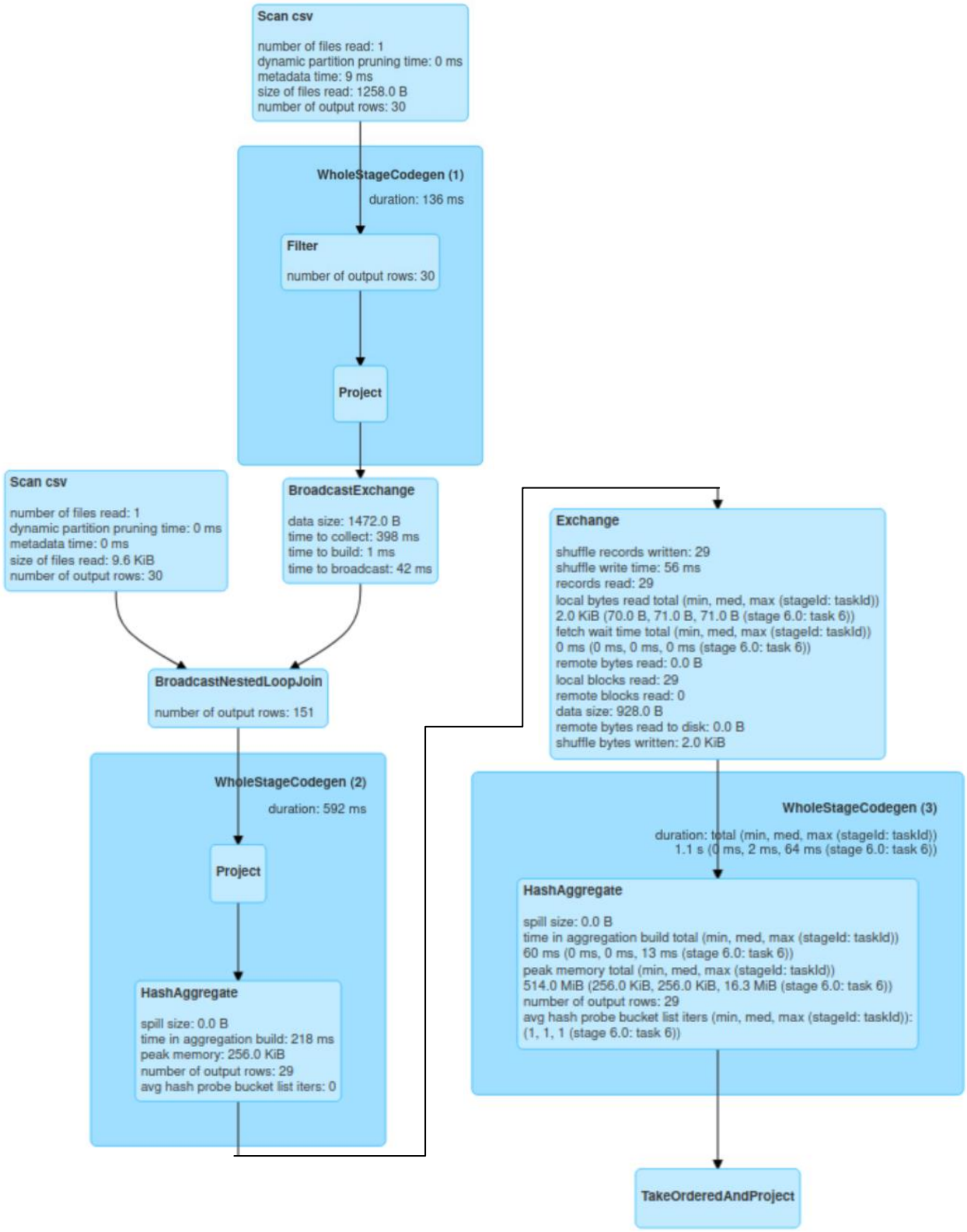
Временная диаграмма выполнения SQL-Запроса

Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	0.7 s	0.7 s	0.7 s	0.7 s	0.7 s
GC Time	0.2 s	0.2 s	0.2 s	0.2 s	0.2 s
Input Size / Records	9.6 KiB / 30	9.6 KiB / 30	9.6 KiB / 30	9.6 KiB / 30	9.6 KiB / 30
Shuffle Write Size / Records	2 KiB / 29	2 KiB / 29	2 KiB / 29	2 KiB / 29	2 KiB / 29

					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)		
						Литер.	Масса	Масштаб
Изм	Лист	№ документа	Подпись	Дата	Spark. Мониторинг выполнения запроса			
Разраб.						Лист 9	Листов 10	
Руковод.								
Н. Контр.								

Spark. DAG SQL-запроса



					Разработка макета аналитической системы на основе баз данных NoSQL (вариант № 24)	
					Spark. DAG SQL-запроса	Литер. / Масса / Масштаб
Изм	Лист	№ документа	Подпись	Дата		Лист 10 / Листов 10
Разраб.						
Руковод.						
Н. Контр.						