

6. Lambda архитектура. Spark Streaming + Cassandra

1. Подключиться к кластеру, создать таблицу в бд Cassandra с использованием консоли cqlsh.
2. Заполнить таблицу данными.
3. Читаем данные в пакетном режиме через Spark используя коннектор к Cassandra. Подготовить отчет - листинг консоли с результатами выполнения заданий.
4. Доп задание на оценку "Хорошо": Выполнить все необходимые задания в предыдущих пунктах. Создать временное представление в памяти, выполнить запрос с фильтром НЕ по ключевому полю и добиться пуш фильтра в Cassandra.
5. Доп задание на оценку "Отлично": Выполнить все необходимые задания в предыдущих пунктах. Пункт 2 выполнить следующим образом: Скачать dataset среднего размера из любого открытого источника (например www.kaggle.com), написать скрипт на python для импорта dataset в таблицу Cassandra (в качестве коннектора использовать пакет `from cassandra.cluster import Cluster`)

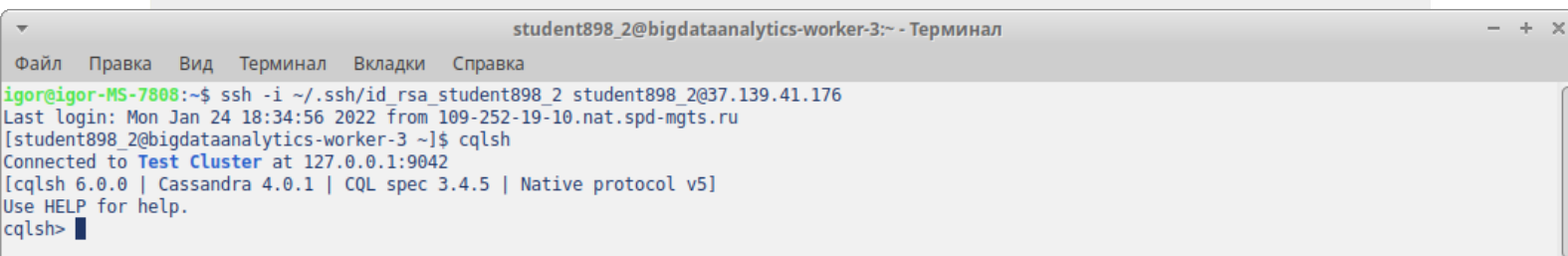
P.S. Задание на "Отлично" возможно будет сложным для Вас, однако это задание приближено к рабочей обстановке. Всех обратившихся ко мне до истечения дедлайна на данное дз, с конструктивными вопросами или проблемами по этому заданию ждет бонус в виде зачета задания 5 вне зависимости от его успешного или неуспешного выполнения. Важно что вы проикнитесь в проблему и сделаете шаги в сторону ее решения.

Поработать с Cassandra через консоль. Протестировать инсерты, селекты с разными ключами. Работать в keyspace lesson7. Там можно создать свои таблички.

Подключаемся к серверу ssh -i ~/.ssh/id_rsa_student898_2 student898_2@37.139.41.176

Запускаем консольный клиент Cassandra

cqlsh



```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
igor@igor-MS-7808:~$ ssh -i ~/.ssh/id_rsa_student898_2 student898_2@37.139.41.176
Last login: Mon Jan 24 18:34:56 2022 from 109-252-19-10.nat.spd-mgts.ru
[student898_2@bigdataanalytics-worker-3 ~]$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.0.0 | Cassandra 4.0.1 | CQL spec 3.4.5 | Native protocol v5]
Use HELP for help.
cqlsh>
```

Далее все команды в терминале кассандры.

Выбираем keyspace:

```
USE lesson7;

DROP TABLE shadrin_animals;

SELECT * FROM
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
Use HELP for help.
cqlsh> USE lesson7;
cqlsh:lesson7> DROP TABLE shadrin_animals;
InvalidRequest: Error from server: code=2200 [Invalid query] message="Table 'lesson7.shadrin_animals' doesn't exist"
cqlsh:lesson7> SELECT * FROM
animals                les_7.                lesson7.                system.                system_schema.        system_virtual_schema.
animals_2              less7.                lesson_7.                system_auth.            system_traces.
keyspace1              lesson7.               projectx.                system_distributed.     system_views.
```

Создаём новую табличку:

```
CREATE TABLE shadrin_animals(id int, name text, size text, primary key (id));

SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> CREATE TABLE shadrin_animals(id int, name text, size text, primary key (id));
cqlsh:lesson7> SELECT * FROM shadrin_animals;

 id | name | size
----+-----+-----
(0 rows)
cqlsh:lesson7>
```

Вставка записи:

```
INSERT INTO shadrin_animals (id, name, size) VALUES ( 3, 'Deer', 'Big');

SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> INSERT INTO shadrin_animals (id, name, size) VALUES ( 3, 'Deer', 'Big');
cqlsh:lesson7> SELECT * FROM shadrin_animals;

 id | name | size
----+-----+-----
  3 | Deer | Big
(1 rows)
cqlsh:lesson7>
```

Апдейт записи с `id = 3`:

```
insert into shadrin_animals (id, name) values (3, 'Doe');

SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> insert into shadrin_animals (id, name) values (3, 'Doe');
cqlsh:lesson7> SELECT * FROM shadrin_animals;

 id | name | size
----+----+----
  3 | Doe  | Big

(1 rows)
cqlsh:lesson7>
```

Вставка ещё одной записи:

```
insert into shadrin_animals (id, name) values (5, 'Snake');
```

```
SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> insert into shadrin_animals (id, name) values (5, 'Snake');
cqlsh:lesson7> SELECT * FROM shadrin_animals;

 id | name | size
----+----+----
  5 | Snake | null
  3 | Doe  | Big

(2 rows)
cqlsh:lesson7>
```

Удаление по ключу не отработает. Это особенность консольной утилиты.

```
delete id from shadrin_animals where id = 3;
```

Удалить запись можно, затерев старые значения.

```
insert into shadrin_animals (id, name, size) values (3, null, null);
```

```
SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> delete id from shadrin_animals where id = 3;
InvalidRequest: Error from server: code=2200 [Invalid query] message="Invalid identifier id for deletion (should not be a PRIMARY KEY part)"
cqlsh:lesson7> insert into shadrin_animals (id, name, size) values (3, null, null);
cqlsh:lesson7> SELECT * FROM shadrin_animals;

 id | name | size
----+----+----
  5 | Snake | null
  3 | null  | null

(2 rows)
cqlsh:lesson7>
```

В конце удалим табличку:

```
drop table shadrin_animals;
```

```
SELECT * FROM shadrin_animals;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:lesson7> drop table shadrin animals;
cqlsh:lesson7> SELECT * FROM shadrin_animals;
InvalidRequest: Error from server: code=2200 [Invalid query] message="table shadrin_animals does not exist"
cqlsh:lesson7>
```

Проверим как выполняется `count` по большой таблице.

```
use keyspace1;
```

```
SELECT table_name FROM system_schema.tables where keyspace_name = 'keyspace1';
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:keyspace1> SELECT table_name FROM system_schema.tables where keyspace_name = 'keyspace1';

table_name
-----
users_unknown

(1 rows)
cqlsh:keyspace1>
```

```
SELECT table_name FROM system_schema.tables;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

peers
peers_v2
prepared_statements
repairs
size_estimates
sstable_activity
table_estimates
transferred_ranges
transferred_ranges_v2
view_builds_in_progress
events
sessions

(48 rows)
cqlsh:keyspace1>
```

```
SELECT * FROM system_schema.tables;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

system_traces | events | 99p | 0.01 | {'keys': 'ALL', 'rows_per_partition': 'NO
NE'} | null | tracing events |
{'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'} | {
'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'} | 1 | 0 |
0 | 0 | 128 | BLOCKING | 0 | 8826e8e9-e16a-3728-8753-3bc1fc713c25 | 2048 |
0 | 99p
system_traces | sessions | 99p | 0.01 | {'keys': 'ALL', 'rows_per_partition': 'NO
NE'} | null | tracing sessions |
{'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'} | {
'chunk_length_in_kb': '16', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'} | 1 | 0 |
0 | 0 | 128 | BLOCKING | 0 | c5e99f16-8677-3914-b17e-960613512345 | 2048 |
0 | 99p

(48 rows)
cqlsh:keyspace1>
```

```
SELECT keyspace_name, table_name FROM system_schema.tables;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

system | peers
system | peers_v2
system | prepared_statements
system | repairs
system | size_estimates
system | sstable_activity
system | table_estimates
system | transferred_ranges
system | transferred_ranges_v2
system | view_builds_in_progress
system_traces | events
system_traces | sessions

(48 rows)
cqlsh:keyspace1>
```

```
select * from users unknown limit 10;
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

user_id | age   | c    | gender | ma1 | ma2 | mi1 | mi2 | s1 | s2 | segment
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
35262   | young| null | M      | null| null| null| null| null| null| null
39433   | midage| null| F      | null| null| null| null| null| null| null
37032   | young| null | M      | null| null| null| null| null| null| null
48451   | midage| null| F      | null| null| null| null| null| null| null
40239   | young| null | F      | null| null| null| null| null| null| null
47076   | young| null | M      | null| null| null| null| null| null| null
41114   | old   | null | M      | null| null| null| null| null| null| null
34323   | young| null | F      | null| null| null| null| null| null| null
35243   | old   | null | F      | null| null| null| null| null| null| null
43690   | midage| null| M      | null| null| null| null| null| null| null


(10 rows)
cqlsh:keyspace1>
```

сделаем подсчет сколько мужчин и сколько женщин

```
select gender, count(*) from users_unknown group by gender;
```

мы не сможем по значению, только общее кол-во

```
select count(*) from users unknown group by user id;
```



The screenshot shows a terminal window with a title bar that reads "student898_2@bigdataanalytics-worker-3:~ - Терминал". The window has a menu bar with options: "Файл", "Правка", "Вид", "Терминал", "Вкладки", and "Справка". The terminal content consists of 15 lines, each containing the number "1". At the bottom of the terminal, the text "---MORE---" is visible, indicating that the output has been truncated.

```
select count(*) from users unknown;
```

exit

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
---MORE---select count(*) from users_unknown;
<Error from server: code=000a [Protocol error] message="Invalid value for the paging state">
cqlsh:keyspace1> select count(*) from users_unknown;

count
-----
19625

(1 rows)

Warnings :
Aggregation query used without partition key

cqlsh:keyspace1> exit
[student898_2@bigdataanalytics-worker-3 ~]$
```

HBASE

Запускаем консольный клиент:

```
hbase shell
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
cqlsh:keyspace1> exit
[student898_2@bigdataanalytics-worker-3 ~]$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/phoenix/phoenix-5.0.0.3.1.4.0-315-server.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.0.2.3.1.4.0-315, r, Fri Aug 23 05:15:48 UTC 2019
Took 0.0020 seconds
hbase(main):001:0>
```

Создаём новую табличку:

```
create 'lesson6:shadrin_animals', 'name', 'size'
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit', REGION_REPLICATION => 2, CONFIGURATION => {'hbase.hregion.scan.loadCo
lumnFamiliesOnDemand' => 'true'}}
hbase> create 't1', {NAME => 'f1', DFS_REPLICATION => 1}

You can also keep around a reference to the created table:

hbase> t1 = create 't1', 'f1'

Which gives you a reference to the table named 't1', on which you can then
call methods.

Took 1.3397 seconds
hbase(main):002:0>
```

Вставка записи:

```
put 'lesson6:shadrin_animals', '3', 'name', 'Deer'
```

exit

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

ERROR: В соединении отказано

Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates. To put a cell value into table 'ns1:t1' or 't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

hbase> put 'ns1:t1', 'r1', 'c1', 'value'
hbase> put 't1', 'r1', 'c1', 'value'
hbase> put 't1', 'r1', 'c1', 'value', ts1
hbase> put 't1', 'r1', 'c1', 'value', {ATTRIBUTES=>{'mykey'=>'myvalue'}}
hbase> put 't1', 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|SECRET'}
```

The same commands also can be run on a table reference. Suppose you had a reference
t to table 't1', the corresponding command would be:

```
hbase> t.put 'r1', 'c1', 'value', ts1, {ATTRIBUTES=>{'mykey'=>'myvalue'}}
```

Took 138.8503 seconds
hbase(main):003:0>

Запускаем `r pyspark` с указанием библиотеки для работы с `cassandra`.

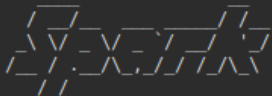
```
export SPARK KAFKA VERSION=0.10
```

```
/opt/spark-2.4.8/bin/pyspark --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5,com.datastax.spark:spark-cassandra-connector_2.11:2.4.2
```

```
student898_2@bigdataanalytics-worker-3: ~ - Терминал
```

Файл	Правка	Вид	Терминал	Вкладки	Справка									
	default		14		0		0		0		14		0	

```
:: retrieving :: org.apache.spark#spark-submit-parent-212ef652-8bf5-4589-bc37-0c666095f3bb  
  confs: [default]  
    0 artifacts copied, 14 already retrieved (0kB/10ms)  
22/01/25 21:02:49 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
22/01/25 21:02:50 WARN util.Utills: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.  
22/01/25 21:02:50 WARN util.Utills: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.  
22/01/25 21:02:50 WARN util.Utills: Service 'SparkUI' could not bind on port 4042. Attempting port 4043.  
Welcome to
```

 version 2.4.8

```
Using Python version 2.7.5 (default, Nov 16 2020 22:23:17)  
SparkSession available as 'spark'.  
>>>
```

Делаем стандартные импорты и читаем табличку. В формате чтения указываем коннектор к базе данных cassandra. Параметры подключения к БД заданы в конфигах ruyspark, поэтому здесь их не указываем.

```
```python
```

```
from pyspark.sql.types import StructType, StringType, IntegerType, TimestampType
```



```
from pyspark.sql import functions as F

cass_animals_df = spark.read \

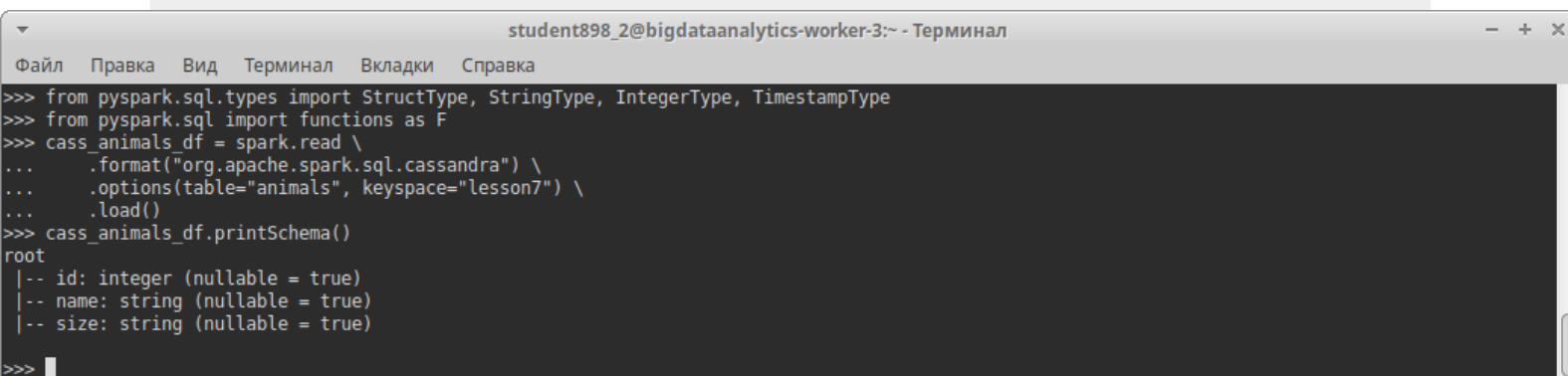
 .format("org.apache.spark.sql.cassandra") \

 .options(table="animals", keyspace="lesson7") \

 .load()

смотрим схему датафрейма

cass_animals_df.printSchema()
```



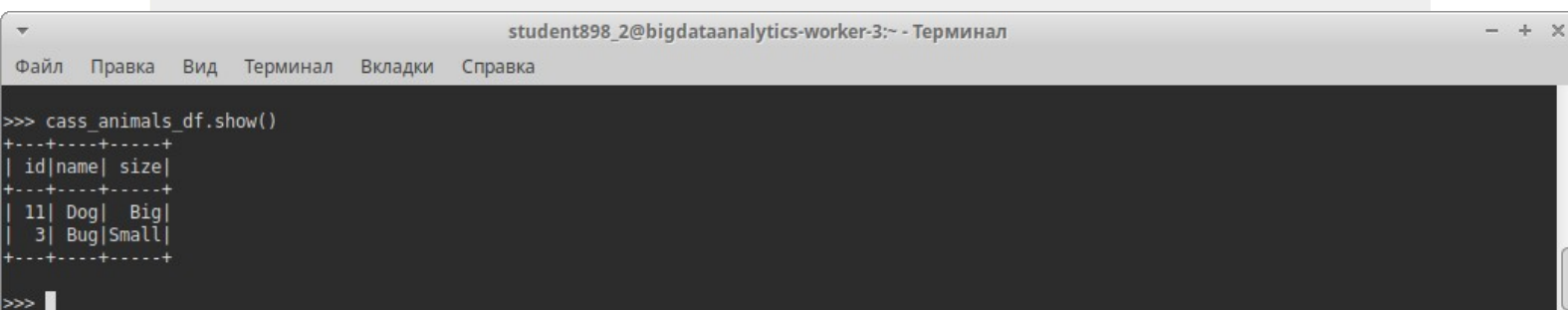
A terminal window titled "student898\_2@bigdataanalytics-worker-3:~ - Терминал" showing the execution of the first code block. The output shows the schema of the DataFrame.

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> from pyspark.sql.types import StructType, StringType, IntegerType, TimestampType
>>> from pyspark.sql import functions as F
>>> cass_animals_df = spark.read \
... .format("org.apache.spark.sql.cassandra") \
... .options(table="animals", keyspace="lesson7") \
... .load()
>>> cass_animals_df.printSchema()
root
 |-- id: integer (nullable = true)
 |-- name: string (nullable = true)
 |-- size: string (nullable = true)
>>>
```

Посмотрим, что есть в таблице:

```
```python

cass_animals_df.show()
```



A terminal window titled "student898_2@bigdataanalytics-worker-3:~ - Терминал" showing the execution of the second code block. The output shows the first few rows of the DataFrame.

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
>>> cass_animals_df.show()
+----+-----+
| id|name| size|
+----+-----+
| 11| Dog| Big|
| 3| Bug| Small|
+----+-----+
>>> 
```

Создадим запись с ключем 11 и добавим её в таблицу.

```
```python

dog_df = spark.sql("""select 11 as id, "Dog" as name, "Big" as size """)

dog_df.show()
```



```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> dog_df = spark.sql("""select 11 as id, "Dog" as name, "Big" as size """)
>>> dog_df.show()
+---+-----+
| id|name|size|
+---+-----+
| 11| Dog| Big|
+---+-----+
>>>
```

Добавляем с указанием режима `append`.

```
```python
```

```
dog_df.write \

    .format("org.apache.spark.sql.cassandra") \

    .options(table="animals", keyspace="lesson7") \

    .mode("append") \

    .save()
```

```
cass_animals_df.show()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

>>> dog_df.write \
...     .format("org.apache.spark.sql.cassandra") \
...     .options(table="animals", keyspace="lesson7") \
...     .mode("append") \
...     .save()
>>> cass_animals_df.show()
+---+-----+
| id|name| size|
+---+-----+
| 11| Dog|  Big|
|  3| Bug| Small|
+---+-----+
>>> 
```

Не смотря на то что при записи указывался режим `append`, фактически был произведён `update` записи, так как такой ключ уже существовал в таблице.

Теперь прочитаем большой датасет по ключу.

```
```python
```

```
cass_big_df = spark.read \

 .format("org.apache.spark.sql.cassandra") \
```

```
.options(table="users_unknown", keyspace="keyspace1") \
```

```
.load()
```

```
cass_big_df.printSchema()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_big_df = spark.read \
... .format("org.apache.spark.sql.cassandra") \
... .options(table="users_unknown", keyspace="keyspace1") \
... .load()
>>> cass_big_df.printSchema()
root
 |-- user_id: integer (nullable = true)
 |-- age: string (nullable = true)
 |-- c: integer (nullable = true)
 |-- gender: string (nullable = true)
 |-- ma1: integer (nullable = true)
 |-- ma2: integer (nullable = true)
 |-- mi1: integer (nullable = true)
 |-- mi2: integer (nullable = true)
 |-- s1: integer (nullable = true)
 |-- s2: integer (nullable = true)
 |-- segment: string (nullable = true)
>>>
```

```
cass_big_df.show(10)
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

|-- segment: string (nullable = true)

>>> cass_big_df.show(10)
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| ma1| ma2| mi1| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 39729| young|null| F|null|null|null|null|null|null| null|
|31468| midage|null| M|null|null|null|null|null|null| null|
| 37970| old|null| M|null|null|null|null|null|null| null|
|41876| old|null| M|null|null|null|null|null|null| null|
|45293| old|null| F|null|null|null|null|null|null| null|
|36928| midage|null| M|null|null|null|null|null|null| null|
| 34100| old|null| M|null|null|null|null|null|null| null|
| 35078| old|null| M|null|null|null|null|null|null| null|
| 33025| midage|null| F|null|null|null|null|null|null| null|
| 49029| young|null| F|null|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 10 rows

>>>
```

```
cass_big_df.filter(F.col("user_id")=="39729").show()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_big_df.filter(F.col("user_id")=="39729").show()
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| ma1| ma2| mi1| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 39729| young|null| F|null|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

>>>
```

```

cass_big_df.filter(F.col("gender")=="F").show()

cass_big_df.filter(F.col("gender")=="F").count()

cass_big_df.filter(F.col("gender")=="M").count()

```

```

student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_big_df.filter(F.col("gender")=="F").show()
+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| ma1| ma2| mi1| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+
| 39729| young|null| F|null|null|null|null|null|null| null|
| 45293| old|null| F|null|null|null|null|null|null| null|
| 33025| midage|null| F|null|null|null|null|null|null| null|
| 49029| young|null| F|null|null|null|null|null|null| null|
| 31525| midage|null| F|null|null|null|null|null|null| null|
| 33367| midage|null| F|null|null|null|null|null|null| null|
| 48819| young|null| F|null|null|null|null|null|null| null|
| 33669| young|null| F|null|null|null|null|null|null| null|
| 47507| old|null| F|null|null|null|null|null|null| null|
| 37185| young|null| F|null|null|null|null|null|null| null|
| 45203| old|null| F|null|null|null|null|null|null| null|
| 34861| midage|null| F|null|null|null|null|null|null| null|
| 31799| old|null| F|null|null|null|null|null|null| null|
| 48083| old|null| F|null|null|null|null|null|null| null|
| 47261| old|null| F|null|null|null|null|null|null| null|
| 42713| old|null| F|null|null|null|null|null|null| null|
| 35389| midage|null| F|null|null|null|null|null|null| null|
| 30689| old|null| F|null|null|null|null|null|null| null|
| 38147| old|null| F|null|null|null|null|null|null| null|
| 30663| young|null| F|null|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

>>> cass_big_df.filter(F.col("gender")=="F").count()
9804
>>> cass_big_df.filter(F.col("gender")=="M").count()
9821
>>>

```

Проверить пошлит ли спарк фильтры в касандру.

Метод `explain`, который показывает логический и физический план запроса. Так как нас интересует только физический план, то будем пользоваться методом `.explain()` у датафрейма.

```
```python
```

```

def explain(self, extended=True):

    if extended:

        print(self._jdf.queryExecution().toString())

    else:

        print(self._jdf.queryExecution().simpleString())

```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка
>>> def explain(self, extended=True):
...     if extended:
...         print(self._jdf.queryExecution().toString())
...     else:
...         print(self._jdf.queryExecution().simpleString())
...
>>>
```

Проверим что находится в PushedFilters в физическом плане при разных запросах.

Запрос 1:

```python

cass\_big\_df.filter(F.col("user\_id")=="39729").explain()

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_big_df.filter(F.col("user_id")=="39729").explain()
== Physical Plan ==
*(1) Filter isnotnull(user_id#49)
+- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,mal#53,ma2#54,mil#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [IsNotNull(user_id), *EqualTo(user_id,39729)], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,mal:int,ma2:int,mil:int,mi2:int,s1:int,s2:int,s...
>>>
```

Здесь применен 1 фильтр \*(1) Filter isnotnull(user\_id#49)

далее смотрим PushedFilters: [IsNotNull(user\_id), \*EqualTo(user\_id,39729)]  
какой сервис тратит ресурсы

получаем датасет из касандры

cass\_big\_df.filter(F.col("user\_id")=="39729").show()

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_big_df.filter(F.col("user_id")=="39729").show()
+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| mal| ma2| mil| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+
| 39729|young|null| F|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+
>>>
```

Запрос 2:

```python

cass_big_df.filter(F.col("gender")=="F").explain()

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

>>> cass_big_df.filter(F.col("gender")=="F").explain()
== Physical Plan ==
*(1) Filter (isNotNull(gender#52) && (gender#52 = F))
+- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,mal#53,ma2#54,mil#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [IsNotNull(gender), EqualTo(gender,F)], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,mal:int,ma2:int,mil:int,mi2:int,s1:int,s2:int,s...
>>>
```

применен фильтр `EqualTo(gender,F)]`

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл  Правка  Вид  Терминал  Вкладки  Справка

>>> cass_big_df.filter(F.col("gender")=="F").explain()
== Physical Plan ==
*(1) Filter (isNotNull(gender#52) && (gender#52 = F))
+- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,mal#53,ma2#54,mil#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [IsNotNull(gender), EqualTo(gender,F)], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,mal:int,ma2:int,mil:int,mi2:int,s1:int,s2:int,s...
>>>
```

Оба запроса спускают фильтр до уровня базы. Фильтр по ключу делается быстро, первый запрос оптимальный. Фильтр по колонке `gender` в кассандре будет выполняться долго, второй запрос не оптимальный. Но если вызывать метод `.show()` у датафрейма, к запросу будет добавлен `limit=20` и всё-таки второй запрос так же выполнится.

Сделаем представление `cass_df` датафрейма `cass_big_df`, чтобы обращаться к нему внутри SQL-выражений.

```python

`cass_big_df.createOrReplaceTempView("cass_df")`

`cass_df = spark.sql("select * from cass_df limit 10").show()`

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_big_df.createOrReplaceTempView("cass_df")
>>> cass_df = spark.sql("select * from cass_df limit 10").show()
22/01/25 21:54:25 WARN shortcircuit.DomainSocketFactory: The short-circuit local reads feature cannot be used because libhadoop cannot be loaded.
+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| mal| ma2| mil| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+
| 39729| young|null| F|null|null|null|null|null| null|
| 31468| midage|null| M|null|null|null|null|null| null|
| 37970| old|null| M|null|null|null|null|null| null|
| 41876| old|null| M|null|null|null|null|null| null|
| 45293| old|null| F|null|null|null|null|null| null|
| 36928| midage|null| M|null|null|null|null|null| null|
| 34100| old|null| M|null|null|null|null|null| null|
| 35078| old|null| M|null|null|null|null|null| null|
| 33025| midage|null| F|null|null|null|null|null| null|
| 49029| young|null| F|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+-----+
>>>
```

в результате запрос закешировался

```
cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").show()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").show()
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| ma1| ma2| mil| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 33025|midage|null| F|null|null|null|null|null|null| null|
| 33367|midage|null| F|null|null|null|null|null|null| null|
| 33669| young|null| F|null|null|null|null|null|null| null|
| 33220|midage|null| M|null|null|null|null|null|null| null|
| 33379|midage|null| F|null|null|null|null|null|null| null|
| 33494| old|null| M|null|null|null|null|null|null| null|
| 33801| young|null| F|null|null|null|null|null|null| null|
| 33423| young|null| F|null|null|null|null|null|null| null|
| 33444| young|null| M|null|null|null|null|null|null| null|
| 33670|midage|null| M|null|null|null|null|null|null| null|
| 33975| young|null| F|null|null|null|null|null|null| null|
| 33185| old|null| F|null|null|null|null|null|null| null|
| 33408| young|null| M|null|null|null|null|null|null| null|
| 33475|midage|null| F|null|null|null|null|null|null| null|
| 33445|midage|null| F|null|null|null|null|null|null| null|
| 33065| old|null| F|null|null|null|null|null|null| null|
| 33003| young|null| F|null|null|null|null|null|null| null|
| 33756| young|null| M|null|null|null|null|null|null| null|
| 33184|midage|null| M|null|null|null|null|null|null| null|
| 33436|midage|null| M|null|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

>>>
```

```
cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").explain()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").explain()
== Physical Plan ==
*(1) Filter ((isnotnull(user_id#49) && (user_id#49 >= 33000)) && (user_id#49 <= 34000))
+- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,ma1#53,ma2#54,mil#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [IsNotNull(user_id), GreaterThanOrEqual(user_id,33000), LessThanOrEqual(user_id,34000)], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,ma1:int,ma2:int,mil:int,mi2:int,s1:int,s2:int,s...
>>>
```

```
cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").explain()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка

>>> cass_df = spark.sql("select * from cass_df where user_id between 33000 and 34000").explain()
== Physical Plan ==
*(1) Filter ((isnotnull(user_id#49) && (user_id#49 >= 33000)) && (user_id#49 <= 34000))
+- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,ma1#53,ma2#54,mil#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [IsNotNull(user_id), GreaterThanOrEqual(user_id,33000), LessThanOrEqual(user_id,34000)], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,ma1:int,ma2:int,mil:int,mi2:int,s1:int,s2:int,s...
>>>
```

мы видим что выполняется команда пуш-фильтр

```
cass_df = spark.sql("select count(*) from cass_df where user_id between 33000 and 34000").show()
```

```
cass_df = spark.sql("select count(*) from cass_df where user_id between 33000 and 34000").explain()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_df = spark.sql("select count(*) from cass_df where user_id between 33000 and 34000").show()
+-----+
|count(1)|
+-----+
| 981|
+-----+

>>> cass_df = spark.sql("select count(*) from cass_df where user_id between 33000 and 34000").explain()
== Physical Plan ==
*(2) HashAggregate(keys=[], functions=[count(1)])
+- Exchange SinglePartition
 +- *(1) HashAggregate(keys=[], functions=[partial_count(1)])
 +- *(1) Project
 +- *(1) Filter ((isnotnull(user_id#49) && (user_id#49 >= 33000)) && (user_id#49 <= 34000))
 +- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49] PushedFilters: [IsNotNull(user_id), GreaterThanOrEqual(user_id,33000), LessThanOrEqual(user_id,34000)], ReadSchema: struct<user_id:int>
>>>
```

агрегация выполняется на стороне хайв

```
cass_df = spark.sql("select count(*) from cass_df where user_id > 33000 and user_id < 34000").explain()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_df = spark.sql("select count(*) from cass_df where user_id > 33000 and user_id < 34000").explain()
== Physical Plan ==
*(2) HashAggregate(keys=[], functions=[count(1)])
+- Exchange SinglePartition
 +- *(1) HashAggregate(keys=[], functions=[partial_count(1)])
 +- *(1) Project
 +- *(1) Filter ((isnotnull(user_id#49) && (user_id#49 > 33000)) && (user_id#49 < 34000))
 +- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49] PushedFilters: [IsNotNull(user_id), GreaterThan(user_id,33000), LessThan(user_id,34000)], ReadSchema: struct<user_id:int>
>>>
```

```
cass_df = spark.sql("select count(*) from cass_df where user_id in (33408, 33475)").show()
```

```
cass_df = spark.sql("select count(*) from cass_df where user_id in (33408, 33475)").explain()
```

```
student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
>>> cass_df = spark.sql("select count(*) from cass_df where user_id in (33408, 33475)").show()
22/01/25 22:02:32 WARN core.RequestHandler: Query '[2 bound values] SELECT count(*) FROM "keyspace1"."users_unknown" WHERE "user_id" IN (?, ?) ALLOW FILTERING;' generated server side warning(s): Aggregation query used on multiple partition keys (IN restriction)
+-----+
|count(1)|
+-----+
| 2|
+-----+

>>> cass_df = spark.sql("select count(*) from cass_df where user_id in (33408, 33475)").explain()
== Physical Plan ==
*(2) HashAggregate(keys=[], functions=[count(1)])
+- Exchange SinglePartition
 +- *(1) HashAggregate(keys=[], functions=[partial_count(1)])
 +- *(1) Project
 +- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [] PushedFilters: [*In(user_id, [33408,33475])], ReadSchema: struct<>
>>>
```



```

cass_df = spark.sql("select * from cass_df where user_id in (33408,
33475)").show()

cass_df = spark.sql("select * from cass_df where user_id in (33408,
33475)").explain()

cass_df = spark.sql("select count(*) from cass_df where user_id in (33408,
33475)").explain()

```

```

student898_2@bigdataanalytics-worker-3:~ - Терминал
Файл Правка Вид Терминал Вкладки Справка
struct<
>>> cass_df = spark.sql("select * from cass_df where user_id in (33408, 33475)").show()
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|user_id| age| c|gender| ma1| ma2| mi1| mi2| s1| s2|segment|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 33408| young|null| M|null|null|null|null|null|null| null|
| 33475| midage|null| F|null|null|null|null|null|null| null|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
>>> cass_df = spark.sql("select * from cass_df where user_id in (33408, 33475)").explain()
== Physical Plan ==
*(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [user_id#49,age#50,c#51,gender#52,ma1#53,ma2#54,mi1#55,mi2#56,s1#57,s2#58,segment#59] PushedFilters: [*In(user_id, [33408,33475])], ReadSchema: struct<user_id:int,age:string,c:int,gender:string,ma1:int,ma2:int,mi1:int,mi2:int,s1:int,s2:int,s...
>>> cass_df = spark.sql("select count(*) from cass_df where user_id in (33408, 33475)").explain()
== Physical Plan ==
*(2) HashAggregate(keys=[], functions=[count(1)])
+- Exchange SinglePartition
 +- *(1) HashAggregate(keys=[], functions=[partial_count(1)])
 +- *(1) Project
 +- *(1) Scan org.apache.spark.sql.cassandra.CassandraSourceRelation@133c71b8 [] PushedFilters: [*In(user_id, [33408,33475])], ReadSchema:
struct<
>>>

```

Мы рассмотрели как взаимодействовать в батчевом режиме

Создать свой кей-спайс, загрузить свои таблицы/данные используя команды `insert & values`.

Задание со \* взять датасет и загрузить этот дата-сет в базу данных. Написать питон скрипт `cassandra.cluster` из `python3`

т.е. Зайти и почитать как именно туда записать дата-сет (на сервер) рекомендация использовать **csv**. В созданную таблицы в кей-спайс, нужно записать данные и прочитать с помощью команд, приложить скрипт

## Install

```
$ pip install cassandra-csv
```

## Usage

```
from cassandra.cluster import Cluster
```

```
from cassandracs import CassandraCsv

cluster = Cluster()
cassandra_cluster = cluster.connect('database')

result = cassandra_cluster.execute("""SELECT foo FROM bar WHERE
foobar=2""")

CassandraCsv.export(
 result,
 [options]
)
```