

HW #9: Cassandra + Spark

1. Описание задания	2
2. Критерии оценивания	2
3. Описание данных	3
4. Задание #1, Task ID: cassandra.create_keyspace	4
5. Задание #2, Task ID: cassandra.movies_ddl	4
6. Задание #3, Task ID: cassandra.movies_by_genre	5
7. Задание #4, Task ID: cassandra.movies_with_genre_index	5
8. Задание #5, Task ID: cassandra.movies_by_genre_rating	6
9. Правила оформления задания	7



1. Описание задания

В этом задании вы будете разрабатывать схему данных в Cassandra для видео-сервиса, основанного на данных [Movielens](#). От вас потребуется:

1. спроектировать и реализовать в Cassandra схему данных для получения ответов на поставленные вопросы;
2. написать запросы для получения ответов;
3. написать на Spark SQL код загрузки данных из HDFS в Cassandra.

Дополнительная информация

Запуск cqlsh на brain-client:

```
$ cqlsh brain-node1
```

Запуск Spark со Spark Cassandra Connector:

```
$ <options to run pyspark or spark-submit> --conf  
spark.cassandra.connection.host=brain-node1 --packages  
com.datastax.spark:spark-cassandra-connector_2.11:2.4.2 --conf  
spark.cassandra.auth.username=<username>
```

Запуск CQL скриптов в автоматическом режиме будет производиться с помощью:

```
cqlsh ${cassandra_endpoint} -k ${keyspace_name} -f task_*.cql
```

Запуск PySpark скриптов в автоматическом режиме будет производиться с помощью:

```
spark-submit --conf  
spark.cassandra.connection.host=${cassandra_endpoint} --packages  
com.datastax.spark:spark-cassandra-connector_2.11:2.4.2 task_*.py  
${keyspace_name}1
```

Как создать DataFrame из таблицы Cassandra можно прочитать [здесь](#)

2. Критерии оценивания

Балл за задачу складывается из:

- **100%** - правильное решение задачи

¹ Таким образом внутри PySpark скрипта рекомендуется использовать `sys.argv[1]` для получения keyspace для записи данных



- **0%** - поддерживаемость и читаемость кода
 - в общем случае см. Clean Code и [Google Python Style Guide](#)
- **0%** - эффективность решения (такие как потребляемые CPU-ресурсы, скорость выполнения (в предположении свободного кластера)).

Бонусы и штрафы:

- **100%** за плагиат в решениях (всем участникам процесса)
- **100%** за посылку решения после deadline
- **5%** за каждую новую посылку (одна дополнительная посылка бесплатно)

Формула подсчета финальной оценки²:

$\max(0, 0.95^{\max(0, \# \text{доп.посылок} - 1)} * (1 - \text{штраф. за. дедлайн. и. списывание})) * \text{последняя. оценка. из. grader}$

3. Описание данных

3.1 Данные о фильмах

movies.csv:

- Путь на кластере: полный датасет - `/data/movielens/movies.csv`
- Формат: текст
- В каждой строке находятся следующие поля, разделенные запятой:
 1. `movieId` - id фильма
 2. `title` - заголовок фильма (содержит год выпуска)
 3. `genres` - список жанров в виде строке слов, разделенных символом " | "

Пример:

```
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
```

3.2 Данные о рейтингах

ratings.csv:

- Путь на кластере: полный датасет - `/data/movielens/ratings.csv`
- Формат: текст
- В каждой строке находятся следующие поля, разделенные запятой:
 1. `userId` - id пользователя
 2. `movieId` - id фильма

² результат округляется до целого



3. rating- оценка
4. timestamp - время оценивания

Пример:

```
1,1441,4.0,945544871
1,1609,3.0,945544824
1,1961,3.0,945544871
```

4. Задание #1, Task ID: `cassandra.create_keyspace`

Создайте keyspace со стратегией репликации SimpleStrategy и фактором репликации

2. Назовите keyspace в соответствии с аргументом командной строки:

```
bash task*_create_keyspace.sh ${cassandra_endpoint} ${keyspace_name}
```

5. Задание #2, Task ID: `cassandra.movies_ddl`

movieid	int
title	text
year	int
genres	set<text>

1. Создайте таблицу `movies` со схемой, приведенной выше. Таблица должна давать возможность выбрать всю информацию по конкретному названию фильма. Поскольку у одного фильма может быть несколько версий выхода на экран, то нужно предоставить возможность искать информацию по фильму и году одновременно. Заметьте, что жанры в исходном файле представляют из себя строку слов, разделенных символом вертикальной черты "|". В таблице в Cassandra жанры должны быть представлены типом [set](#).
2. Напишите Spark SQL код, который будет считывать файл `movies.csv`, приводить DataFrame к правильному формату и запишет данные в таблицу `movies` в Cassandra. Обратите внимание, что в датасете могут быть ошибки, поэтому ошибки записи в Cassandra могут быть устранены предварительной очисткой данных³.

³ (1) предобработка quotes и trailing whitespaces в полях (до и после обработки сырых данных) (2) удаление строк без указания года и других характеристик (3) правильная обработка placeholder для данных, где не указаны жанры (найдите какой это placeholder)



3. Напишите запрос, который посчитает, сколько записей содержится в таблице (проверьте себя - должно получиться 37,968).

Названия CQL и PySpark скриптов:

- task*_movies_ddl.cql
- task*_movies.py
- task*_movies.cql + task*_movies.out (STDOUT)

6. Задание #3, Task ID: cassandra.movies_by_genre

1. Создайте таблицу `movies_by_genre`, которая бы отвечала на вопрос: получить все фильмы в этом жанре (`genre`), отсортированные по убыванию года.
2. Напишите Spark SQL код, который будет брать данные из файла `movies.csv` заполнит таблицу (каждое задание проверяется независимо от других, поэтому предполагается, что `keyspace` создан, но пустой).
3. Напишите запрос, который подсчитает, сколько фильмов жанра `Horror` было снято в период с 1980 по 1990 год (включительно).

Названия CQL и PySpark скриптов:

- task*_movies_by_genre_ddl.cql
- task*_movies_by_genre.py
- task*_movies_by_genre.cql + task*_movies_by_genre.out (STDOUT)

7. Задание #4, Task ID: cassandra.movies_with_genre_index

1. Ту же информацию, что и в задании #3, можно получить из исходной таблицы `movies`, если создать [вторичный индекс](#) на столбце `genres`. Создайте таблицу `movies_with_genre_index` полностью аналогичную "movies" из задания #2. Создайте вторичный индекс (`genre_idx`) на столбце `genres`.
2. Напишите Spark SQL код, который будет брать данные из файла `movies.csv` заполнит таблицу (каждое задание проверяется независимо от других, поэтому предполагается, что `keyspace` создан, но пустой).
3. Напишите запрос, который подсчитает, сколько фильмов жанра `Horror` было снято в период с 1980 по 1990 год (включительно). Заметьте, что такой запрос невозможно выполнить без опции [ALLOW FILTERING](#). Внимательно прочитайте

про эту опцию и запомните, что ее ни в коем случае нельзя использовать в продакшен запросах! Нужно создать другую схему данных.

Названия CQL и PySpark скриптов:

- `task*_movies_with_genre_index_ddl.cql`
- `task*_movies_with_genre_index.py`
- `task*_movies_with_genre_index.cql + task*_movies_with_genre_index.out` (STDOUT)

8. Задание #5, Task ID: `cassandra.movies_by_genre_rating`

1. Создайте таблицу `movies_by_genre_rating`, которая позволит отвечать на такой вопрос: вывести все фильмы данного жанра в заданном диапазоне лет, отсортированные по убыванию среднего рейтинга фильма (поле `rating`, тип `float`).
2. Напишите Spark SQL код, который будет брать данные из файлов `movies.csv` и `ratings.csv` и заполнит таблицу (каждое задание проверяется независимо от других, поэтому предполагается, что `keyspace` создан, но пустой).
3. Подсчитайте минимальный, средний и максимальный рейтинг у фильмов жанра Sci-Fi вышедших в 21 веке (включая 2000 год).

Названия CQL и PySpark скриптов:

- `task*_movies_by_genre_rating_ddl.cql`
- `task*_movies_by_genre_rating.py`
- `task*_movies_by_genre_rating.cql + task*_movies_by_genre_rating.out` (STDOUT)

9. Правила оформления задания

Оформление задания:

- Код задания (Short name): **HW9:NoSQL(Cassandra)**
- Выполненное ДЗ упакуйте в архив `MADEBD2021Q1_<Surname>_<Name>_HW#.zip`, например, для Алексея Драля -- `MADEBD2021Q1_Dral_Alexey_HW9.zip`. (Проверяйте отсутствие пробелов и невидимых символов после копирования)



имени отсюда.⁴) Если ваше решение лежит в папке `my_solution_folder`, то для создания архива `hw.zip` на Linux и Mac OS, выполните команду⁵:

- `zip -r hw.zip my_solution_folder/*`
- На Windows 7/8/10: необходимо выделить все содержимое директории `my_solution_folder/` нажать правую кнопку мыши на одном из выделенных объектов, выбрать в открывшемся меню "Отправить >", затем "Сжатая ZIP-папка". Теперь можно переименовать архив.
- Перед проверкой убедитесь, что дерево вашего архива выглядит так:
 - | `MADEBD2021Q1_<Surname>_<Name>_HW9.zip`
 - | `---- task_<Surname>_<Name>_create_keyspace.sh`
 - | `---- task_<Surname>_<Name>_movies_ddl.cql`
 - | `---- task_<Surname>_<Name>_movies.py`
 - | `---- task_<Surname>_<Name>_movies.cql`
 - | `---- task_<Surname>_<Name>_movies.out`
 - | `---- task_<Surname>_<Name>_movies_by_genre_ddl.cql`
 - | `---- task_<Surname>_<Name>_movies_by_genre.py`
 - | `---- task_<Surname>_<Name>_movies_by_genre.cql`
 - | `---- task_<Surname>_<Name>_movies_by_genre.out`
 - | `---- task_<Surname>_<Name>_movies_with_genre_index_ddl.cql`
 - | `---- task_<Surname>_<Name>_movies_with_genre_index.py`
 - | `---- task_<Surname>_<Name>_movies_with_genre_index.cql`
 - | `---- task_<Surname>_<Name>_movies_with_genre_index.out`
 - | `---- task_<Surname>_<Name>_movies_by_genre_rating_ddl.cql`
 - | `---- task_<Surname>_<Name>_movies_by_genre_rating.py`
 - | `---- task_<Surname>_<Name>_movies_by_genre_rating.cql`
 - | `---- task_<Surname>_<Name>_movies_by_genre_rating.out`
 - При несовпадении дерева вашего архива с представленным деревом, ваше решение будет невозможно автоматически проверить, а значит, и оценить его.
- Для того, чтобы сдать задание необходимо:
 - Зарегистрироваться и залогиниться в сервисе [Everest](#)
 - Перейти на страницу приложения: [BDT-grader-MADE-BD](#)
 - Выбрать вкладку Submit Job (если отображается иная).
 - Выбрать в качестве "Task" значение: **HW9:NoSQL(Cassandra)**⁶
 - Загрузить в качестве "Task solution" файл с решением
 - В качестве Sender ID указать тот, который был выслан по почте
- Если Вы видите надпись "You are not allowed to run this application" во вкладке Submit Job в Everest, то на данный момент сдача закрыта (нет доступных для

⁴ Онлайн инструмент для проверки: <https://www.soscisurvey.de/tools/view-chars.php>

⁵ Флаг `-r` значит, что будет совершен рекурсивный обход по структуре директории

⁶ Сервисный ID: `cassandra.onsite_hw`



сдачи домашних заданий, по техническим причинам или другое). Попробуйте, пожалуйста, еще раз через некоторое время. Если Вы еще ни разу не сдавали, у коллег сдача работает, но Вы видите такое сообщение, сообщите нам об этом.

- Ситуации:
 - * система оценивания показывает оценку (Grade) < 0 , а отчет (Grading report) не помогает решить проблему (пример помощи: в случае неправильно указанного Sender ID система вернет -2 и информацию о том, что его нужно поправить);
 - * показывает 0 и в отчете (Grading report) не указано, какие тесты не пройдены. Если Вы столкнулись с какой-то из них присылайте ссылку на выполненное задание (Job) на почту с темой письма "Short name. ФИО.". Например: **"HW9:NoSQL(Cassandra). Иванов Иван Иванович."**
Пример ссылки: <https://everest.distcomp.org/jobs/67893456230000abc0123def>
- **Внимание:** Если до дедлайна остается меньше суток, и Вы знаете (сами проверили или коллеги сообщили), что сдача решений сломана, обязательно сдайте свое решение и напишите письмо, как написано выше, чтобы мы видели, какое решение Вы имели до дедлайна и смогли его оценить.
- Перед отправкой задания, оставьте, пожалуйста, отзыв о домашнем задании по ссылке: http://rebrand.ly/mailbd2021q1_feedback_hw.. Это позволит нам скорректировать учебную нагрузку по следующим заданиям (в зависимости от того, сколько часов уходит на решение ДЗ), а также ответить на интересующие вопросы.

Любые вопросы / комментарии / предложения можно писать в телеграм-канал курса или на почту bigdata_made2021q1@bigdatateam.org.

Всем удачи!