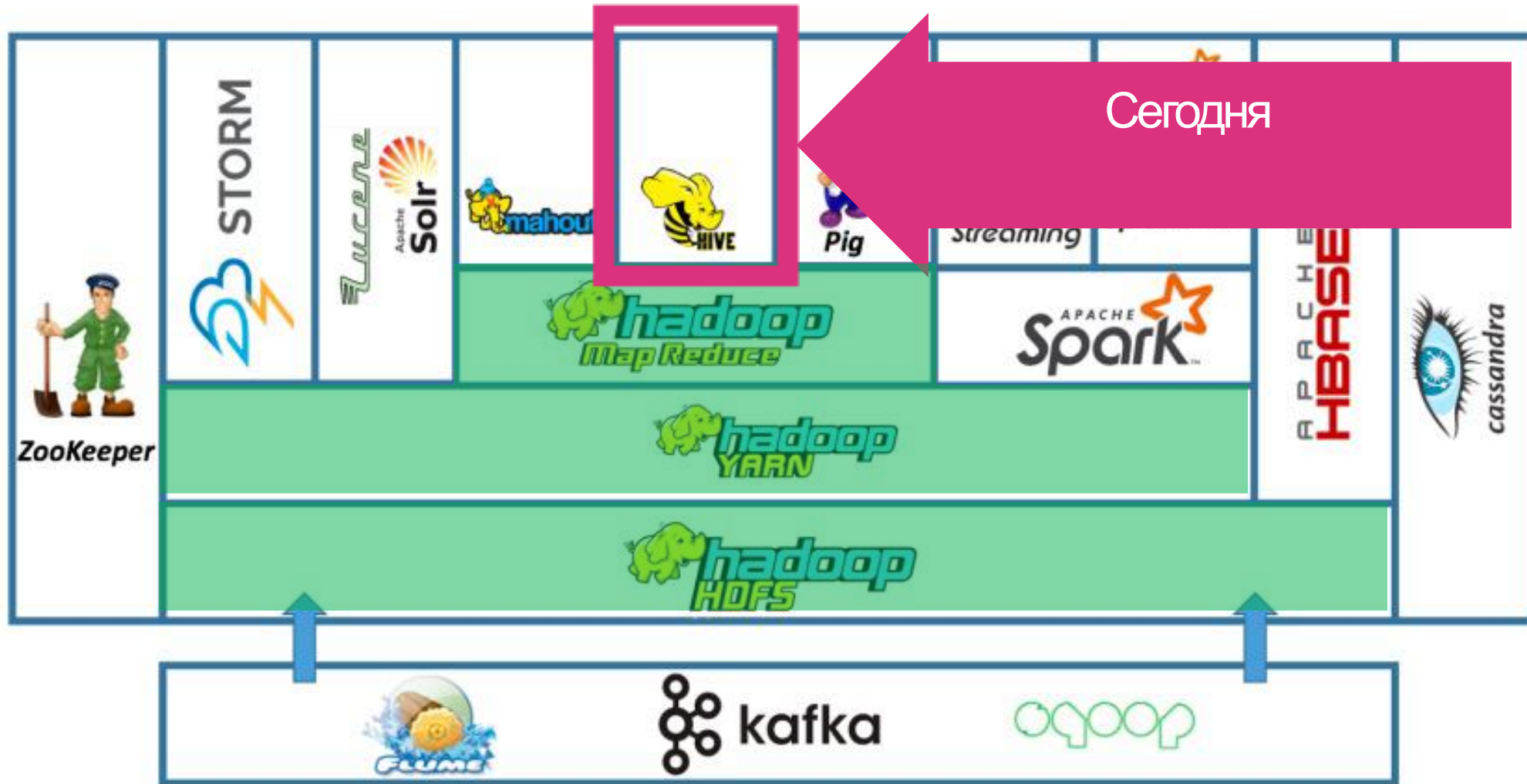


# КУРС

## «Инструменты и корпоративные платформы для хранения и анализа больших данных»

# Hive

# The Hadoop Ecosystem



# Apache Hive



## Apache Hive

- Initial Release in October 2010
- written in Java
- current Version: 3.1.2

<http://hive.apache.org>

- **Hive YES:**
  - прост в использовании (HiveQL)
  - на основе Hadoop (HDFS, YARN)
  - хорошие возможности горизонтального масштабирования (например, путем разделения HDFS и YARN)
- лучше всего использовать для:
  - (BigData) задачи хранения данных
  - озеро данных (а DataLake)
  - интерфейс для аналитиков, специалистов по данным, разработчиков
  - специальные (пакетные) запросы, агрегация и анализ больших объемов данных (PB!) и сотен узлов
- **Hive NOT:**
  - транзакционная база данных
  - highly responsive

# HiveQL

- SQL like **query language**
- Поддерживается: **Hive CLI** (устарело), **Beeline CLI** и большинством клиентов **JDBC**.
- Поддерживаемые Hive форматы файлов HDFS :
  - **Text File** (даже сжатый gzip или bzip2)
  - **Sequence File**
  - **RC File**
  - **ORC**
  - **Parquet**
  - **Avro**

## Hive Text File Format

**Hive Text** является форматом хранения по умолчанию. Используется текстовый формат для обмена данными с другим клиентским приложением. Данные хранятся в строках, каждая строка является записью. Каждая строка заканчивается символом новой строки (`\n`).

Текстовый формат представляет собой простой формат плоского файла. Вы можете использовать сжатие ( *BZIP2* ) в текстовом файле, чтобы уменьшить пространство для хранения.

команда **Hive CREATE TABLE:**

```
Create table textfile_table  
(column_specs)  
stored as textfile;
```

## Hive Sequence File Format

**Файлы последовательности** — это плоские файлы **Hadoop**, в которых значения хранятся в виде двоичных пар «ключ значение». Файлы последовательности имеют двоичный формат, и эти файлы можно разбивать.

Основным преимуществом использования файла последовательности является **объединение двух или более файлов в один файл**.

команда **Hive CREATE TABLE:**

```
Create table sequencefile_table  
(column_specs)  
stored as sequencefile;
```

## Hive RC File Format

**RCFile** — это формат файла со столбцами строк. Это еще одна форма формата файла **Hive**, которая обеспечивает высокую степень сжатия на уровне строк. Если вам нужно выполнять несколько строк одновременно, вы можете использовать формат **RCFile**.

**RCFile** очень похож на формат файла последовательности. Этот формат файла также хранит данные в виде пар ключ-значение.

команда **Hive CREATE TABLE:**

```
Create table RCfile_table  
(column_specs)  
stored as rcfile;
```



## Hive AVRO File Format

**AVRO** — это проект с открытым исходным кодом, который предоставляет услуги сериализации данных и обмена данными для **Hadoop**.

Можно обмениваться данными между экосистемой **Hadoop** и программой, написанной на любом языке программирования.

**Avro** — один из популярных форматов файлов в приложениях на основе **Big Data Hadoop**.

команда **Hive CREATE TABLE:**

```
Create table avro_table  
(column_specs)  
stored as avro;
```

## Hive ORC File Format

**ORC file** — формат файла **Optimized Row Columnar**.

Формат файла **ORC** обеспечивает высокоэффективный способ хранения данных в таблице **Hive**. Эта файловая система фактически была разработана для преодоления ограничений других форматов файлов **Hive**.

Использование файлов **ORC** повышает производительность, когда **Hive** читает, записывает и обрабатывает данные из больших таблиц.

команда **Hive CREATE TABLE:**

```
Create table orc_table  
(column_specs)  
stored as orc;
```

## Hive Parquet File Format

**Parquet** — это формат двоичных файлов, ориентированный на столбцы.

Паркет высокоэффективен для типов объемных запросов.

**Parquet** особенно хорош для запросов, сканирующих определенные столбцы в определенной таблице.

Таблица **Parquet** использует сжатие **Snappy, gzip**.

команда **Hive CREATE TABLE:**

```
Create table parquet_table  
(column_specs)  
stored as parquet;
```

# HDFS/Hive - Wordcount


- пример **WordCount**, полученный с использованием **Hive** и **HiveQL**:

```
CREATE TABLE faust (line STRING);

LOAD DATA INPATH '/user/hadoop/faust' OVERWRITE INTO TABLE faust;

CREATE TABLE word_counts AS
SELECT word, count(1) AS count FROM
(SELECT explode(split(line, '\\s')) AS word FROM faust) temp
GROUP BY word ORDER BY word;
```

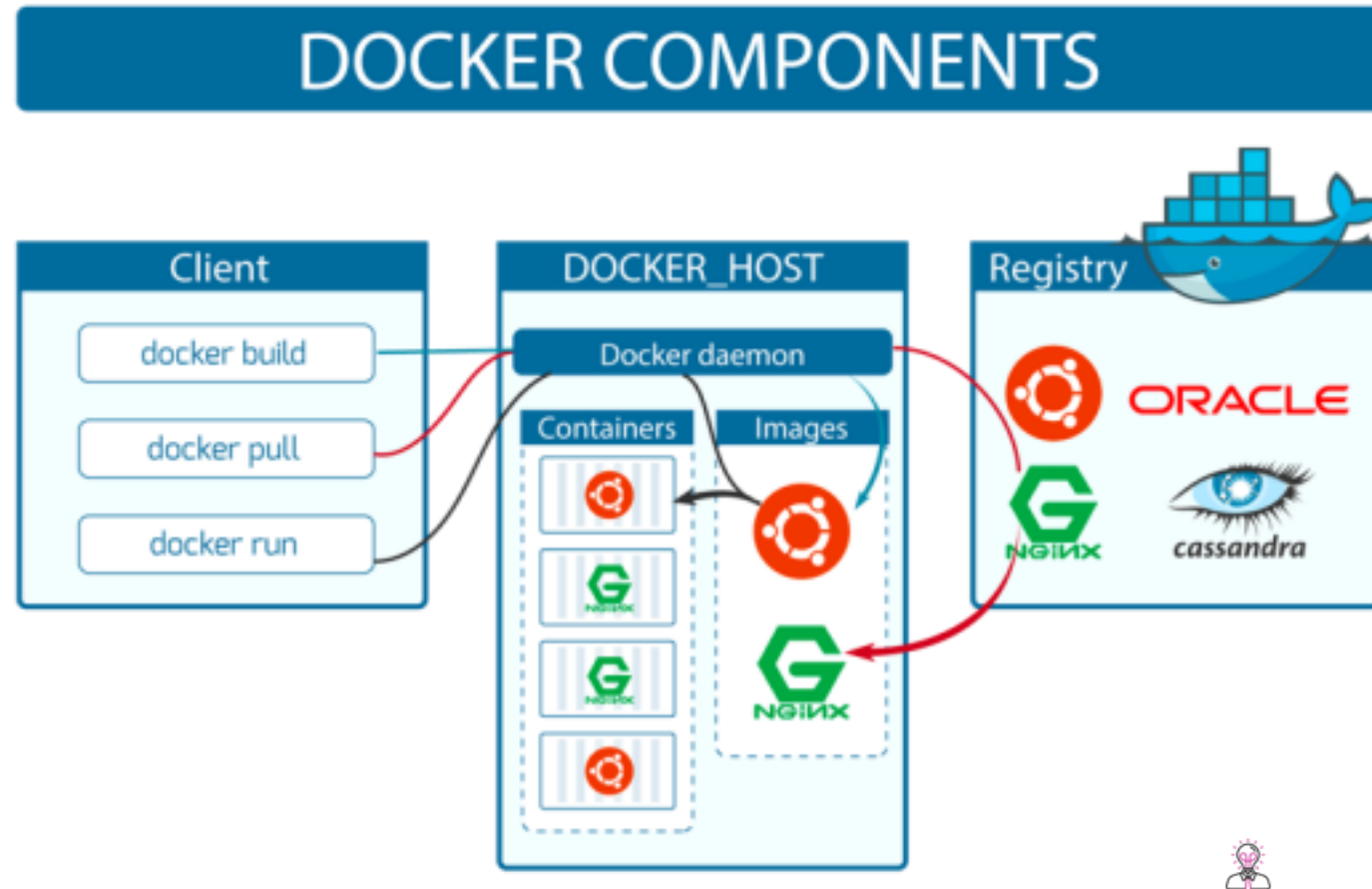
```
SELECT * from word_counts ORDER BY count DESC LIMIT 10;
```



word	count
<i>und</i>	509
<i>die</i>	463
<i>der</i>	440
<i>ich</i>	435
<i>Und</i>	400
<i>nicht</i>	346
<i>zu</i>	319
[...]	

# Docker

Чтобы ускорить процесс и не тратить время на установку и настройку **Hive** и других инструментов, воспользуемся уже подготовленным докер-контейнером.



# Docker Images/Dockerfiles



github.com/BosenkoTM/RepoData/tree/main/docker

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main RepoData / docker

BosenkoTM Add files via upload

Name	Last commit message
..	
airflow	Add files via upload
hadoop_base	Add files via upload
hive_base	Add files via upload
hiveserver_base	Add files via upload
pentaho	Add files via upload
spark_base	Add files via upload
README.md	Add files via upload

README.md

## Docker образы

В этой папке содержатся все изображения Docker, использованные в этом курсе. Дополнительные сведения см. в файлах readme образов Docker (в подпапках), например: как их запустить, остановить и использовать.

образы:

- [Hadoop Base Image](#) Hadoop 3.1.2 Base Image (Ubuntu 18.04)
- [Hadoop and Hive Base Image](#) Hadoop 3.1.2 and Hive 3.1.2 Base Image (Ubuntu 18.04)
- [Hadoop, Hive and HiveServer2 Base Image](#) Hadoop 3.1.2, Hive 3.1.2 and HiveServer2 Base Image (Ubuntu 18.04)
- [Spark Base Image](#) Spark 2.3.4 on Hadoop 3.1.2 as well as Hive 3.1.2 and HiveServer2 Base Image (Ubuntu 18.04)
- [Airflow Base Image](#) Airflow 1.10.5 with PostgreSQL 10.10 as Metadata Store Base Image (Ubuntu 18.04)
- [Pentaho Data Integration Base Image](#) Pentaho Data Integration 8.0 Base Image (Ubuntu 18.04)

<https://github.com/BosenkoTM/RepoData/tree/main/docker>

docker hub

Explore Repositories Organizations Get Help marcelmittelstaedt

Repositories Using 1 of 1 private repositories. [Get more](#)

marcelmittelstaedt Filter by repository name... Create Repository +

REPOSITORY	DESCRIPTION	LAST MODIFIED
marcelmittelstaedt / airflow	Airflow 10.1.5 Image using PostgreSQL 10.10 for Metadata (Ubuntu...	2 days ago
marcelmittelstaedt / spark_base	Hadoop 3.1.2 and Spark 2.3.4 Base Image (Ubuntu 18.04)	5 days ago
marcelmittelstaedt / hiveserver_base	Hive 3.1.2, Hadoop 3.1.2 and HiveServer2 Base Image (Ubuntu 18...	5 days ago
marcelmittelstaedt / hive_base	Hive 3.1.2 and Hadoop 3.1.2 Base Image (Ubuntu 18.04)	6 days ago
marcelmittelstaedt / hadoop_base	Hadoop 3.1.2 Base Image (Ubuntu 18.04)	6 days ago
marcelmittelstaedt / ubuntu_18_04_base	Ubuntu 18.04 Base Image created from scratch using debootstrap...	8 days ago

<https://hub.docker.com/u/marcelmittelstaedt>

# Установка Docker Container



## 1. Установка Docker на Ubuntu

**Шаг 1.** Обновить пакеты, которые уже существуют:

```
sudo apt update
```

**Шаг 2.** Установить пакеты, которые назначат право для “apt” пользоваться пакетами по протоколу HTTPS:

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

**Шаг 3.** Установить GPG ключ для репозитория Docker в систему:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

# Установка Docker Container



## 1. Установка Docker на Ubuntu

**Шаг 4.** Добавить в APT sources репозиторий Docker:

```
sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu focal stable"
```

**Шаг 5.** После чего обновить базу данных пакетами Docker из добавленного репозитория:

```
sudo apt update
```

**Шаг 6.** Убедиться, что установка будет производиться именно из репозитория Docker:

```
apt-cache policy docker-ce
```



# Установка Docker Container



## 1. Установка Docker на Ubuntu

Вы увидите данные строки кода(номер версии для Docker может быть другим):

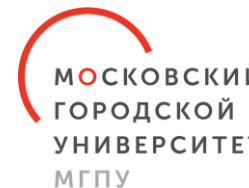
```
docker-ce:
Installed: (none)
Candidate: 5:19.03.9~3-0~ubuntu-focal
Version table:
5:19.03.9~3-0~ubuntu-focal 500
500 https://download.docker.com/linux/ubuntu focal/stable amd64
Packages
```

Обратите внимание, что docker-ce не установлен, но находится в ожидании на установку в репозитории Docker для Ubuntu 20.04

**Шаг 7.** После устанавливаем сам Docker:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-  
buildx-plugin docker-compose-plugin
```

# Установка Docker Container



## 2. Загрузить Hadoop с помощью Hive Image

```
mgpu@mgpu-VirtualBox:~$ sudo docker pull marcelmittelstaedt/hive_base:latest
```

## 3. Запустить контейнер из полученного образа:

```
mgpu@mgpu-VirtualBox:~$ sudo docker run -dit --name hive_base_container -p 8088:8088 -p 9870:9870 -p 9864:9864 marcelmittelstaedt/hive_base:latest
```

# Установка Docker Container



## 4. Просмотреть работающие контейнеры:

mgpu@mgpu-VirtualBox:~\$ **sudo docker ps -a**

```
mgpu@mgpu-VirtualBox:~$ sudo docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
85ace08fabbe	marcelmittelstaedt/hive_base:latest	"/startup.sh"	2 minutes ago	Up 2 minutes	0.0.0.0:8088->8088/tcp, :::8088->8088/tcp, 0.0.0.0:9864->9864/tcp, :::9864->9864/tcp, 0.0.0.0:9870->9870/tcp, :::9870->9870/tcp	hive_base_container

## 5. Просмотреть журнал контейнера (дождитесь завершения):

mgpu@mgpu-VirtualBox:~\$ **sudo docker logs hive\_base\_container**

```
Initialization script completed
schemaTool completed
executing stop-all.sh
WARNING: Stopping all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: Use CTRL-C to abort.
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [85ace08fabbe]
Stopping nodemanagers
Stopping resourcemanager
Container Startup finished.
```

# Установка Docker Container

## 6. Перейти в контейнер:

```
mgpu@mgpu-VirtualBox:~$ sudo docker exec -it hive_base_container bash
```

```
mgpu@mgpu-VirtualBox:~$ sudo docker exec -it hive_base_container bash  
[sudo] password for mgpu:  
root@85ace08fabbe:/#
```

## 7. Переключиться на пользователя **hadoop** :

```
root@85ace08fabbe:/# sudo su hadoop  
hadoop@85ace08fabbe:/$ cd
```

```
root@85ace08fabbe:/# sudo su hadoop  
hadoop@85ace08fabbe:/$ cd
```

## 8. Пуск DFS и YARN:

```
hadoop@85ace08fabbe:~$ start-all.sh
```

```
hadoop@85ace08fabbe:~$ start-all.sh  
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.  
WARNING: This is not a recommended production deployment configuration.  
WARNING: Use CTRL-C to abort.  
Starting namenodes on [localhost]  
Starting datanodes  
Starting secondary namenodes [85ace08fabbe]  
Starting resourcemanager  
Starting nodemanagers
```

# Проверка Hive

## 9. Проверить доступность Hive. Запустить Hive:

hadoop@85ace08fabbe:~\$ **hive**

```
hadoop@85ace08fabbe:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/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.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 763fa944-fb1b-4cac-aeb9-6cfc8c0379d

Logging initialized using configuration in jar:file:/home/hadoop/hive/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Hive Session ID = 7467a491-c400-4ccf-a224-c0b3384f6fa4
hive> █
```

# Настройка Hive

10. Выполните первый SQL-запрос:

```
hive> show databases;
```

```
hive> show databases;  
OK  
default  
Time taken: 0.538 seconds, Fetched: 1 row(s)
```

# Hive: создание внешних таблиц и работа с ними

Использование общедоступного набора данных IMDb.com.

# Get IMDb Data And Move It To HDFS



1. Получить данные с портала IMDb (<https://www.imdb.com/interfaces/>):

```
mgpu@mgpu-VirtualBox:~$ sudo docker exec -it hive_base_container bash
root@85ace08fabbe:/# sudo su hadoop
hadoop@85ace08fabbe:/# cd
hadoop@85ace08fabbe:~$ wget https://datasets.imdbws.com/title.basics.tsv.gz
hadoop@85ace08fabbe:~$ wget https://datasets.imdbws.com/title.ratings.tsv.gz
```

2. Разархивировать данные IMDb:

```
hadoop@85ace08fabbe:~$ gunzip title.basics.tsv.gz
hadoop@85ace08fabbe:~$ gunzip title.ratings.tsv.gz
hadoop@85ace08fabbe:~$ ls
```

3. Создайте каталог HDFS для данных IMDb:

```
hadoop@85ace08fabbe:~$ hadoop fs -mkdir /user/hadoop/imdb
hadoop@85ace08fabbe:~$ hadoop fs -mkdir /user/hadoop/imdb/title_basics
hadoop@85ace08fabbe:~$ hadoop fs -mkdir /user/hadoop/imdb/title_ratings
```



# Create External Tables In Hive

4. Перенесите файлы данных IMDb в HDFS :

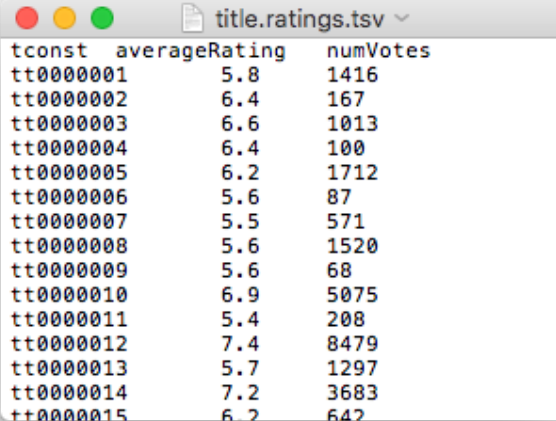
```
hadoop@85ace08fabbe:~$ hadoop fs -put title.basics.tsv /user/hadoop/imdb/title_basics/title.basics.tsv  
hadoop@85ace08fabbe:~$ hadoop fs -put title.ratings.tsv  
/user/hadoop/imdb/title_ratings/title.ratings.tsv
```

5. Create External Table `title_ratings` (file `title.ratings.tsv`) in Hive:

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS title_ratings(  
    tconst STRING,  
    average_rating DECIMAL(2,1),  
    num_votes BIGINT  
    ) COMMENT 'IMDb Ratings'  
    ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' STORED AS  
    TEXTFILE LOCATION '/user/hadoop/imdb/title_ratings'  
    TBLPROPERTIES ('skip.header.line.count'='1');
```

OK

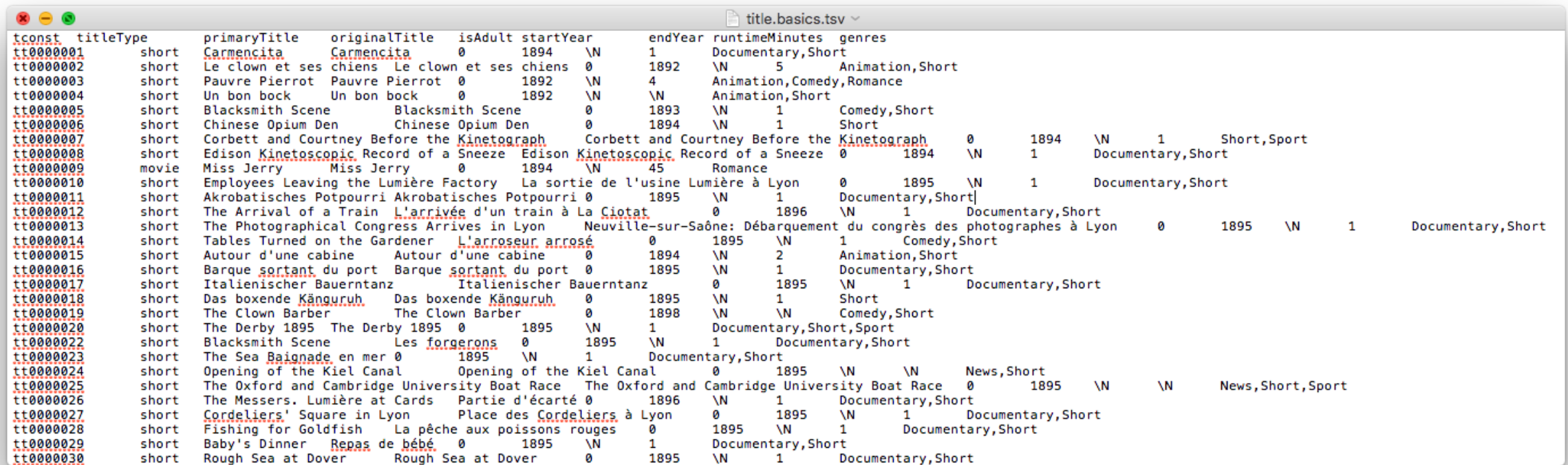
Time taken: 0.699 seconds



tconst	averageRating	numVotes
tt0000001	5.8	1416
tt0000002	6.4	167
tt0000003	6.6	1013
tt0000004	6.4	100
tt0000005	6.2	1712
tt0000006	5.6	87
tt0000007	5.5	571
tt0000008	5.6	1520
tt0000009	5.6	68
tt0000010	6.9	5075
tt0000011	5.4	208
tt0000012	7.4	8479
tt0000013	5.7	1297
tt0000014	7.2	3683
tt0000015	6.2	642

# Create External Tables In Hive

6. Создайте внешнюю таблицу `title_basics` для файла `title.basics.tsv` в Hive.



tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtimeMinutes	genres
tt0000001	short	Carmencita	Carmencita	0	1894	\N	1	Documentary, Short
tt0000002	short	Le clown et ses chiens	Le clown et ses chiens	0	1892	\N	5	Animation, Short
tt0000003	short	Pauvre Pierrot	Pauvre Pierrot	0	1892	\N	4	Animation, Comedy, Romance
tt0000004	short	Un bon bock	Un bon bock	0	1892	\N	\N	Animation, Short
tt0000005	short	Blacksmith Scene	Blacksmith Scene	0	1893	\N	1	Comedy, Short
tt0000006	short	Chinese Opium Den	Chinese Opium Den	0	1894	\N	1	Short
tt0000007	short	Corbett and Courtney Before the Kinetograph	Corbett and Courtney Before the Kinetograph	0	1894	\N	1	Short, Sport
tt0000008	short	Edison Kinetoscopic Record of a Sneeze	Edison Kinetoscopic Record of a Sneeze	0	1894	\N	1	Documentary, Short
tt0000009	movie	Miss Jerry	Miss Jerry	0	1894	\N	45	Romance
tt0000010	short	Employees Leaving the Lumière Factory	La sortie de l'usine Lumière à Lyon	0	1895	\N	1	Documentary, Short
tt0000011	short	Akrobatisches Potpourri	Akrobatisches Potpourri	0	1895	\N	1	Documentary, Short
tt0000012	short	The Arrival of a Train	L'arrivée d'un train à La Ciotat	0	1896	\N	1	Documentary, Short
tt0000013	short	The Photographical Congress Arrives in Lyon	Neuville-sur-Saône: Débarquement du congrès des photographes à Lyon	0	1895	\N	1	Documentary, Short
tt0000014	short	Tables Turned on the Gardener	L'arroseur arrosé	0	1895	\N	1	Comedy, Short
tt0000015	short	Autour d'une cabine	Autour d'une cabine	0	1894	\N	2	Animation, Short
tt0000016	short	Barque sortant du port	Barque sortant du port	0	1895	\N	1	Documentary, Short
tt0000017	short	Italienischer Bauerntanz	Italienischer Bauerntanz	0	1895	\N	1	Documentary, Short
tt0000018	short	Das boxende Känguruh	Das boxende Känguruh	0	1895	\N	1	Short
tt0000019	short	The Clown Barber	The Clown Barber	0	1898	\N	\N	Comedy, Short
tt0000020	short	The Derby 1895	The Derby 1895	0	1895	\N	1	Documentary, Short, Sport
tt0000022	short	Blacksmith Scene	Les forgerons	0	1895	\N	1	Documentary, Short
tt0000023	short	The Sea Baignade en mer		0	1895	\N	1	Documentary, Short
tt0000024	short	Opening of the Kiel Canal	Opening of the Kiel Canal	0	1895	\N	\N	News, Short
tt0000025	short	The Oxford and Cambridge University Boat Race	The Oxford and Cambridge University Boat Race	0	1895	\N	\N	News, Short, Sport
tt0000026	short	The Messers. Lumière at Cards	Partie d'écarté	0	1896	\N	1	Documentary, Short
tt0000027	short	Cordeliers' Square in Lyon	Place des Cordeliers à Lyon	0	1895	\N	1	Documentary, Short
tt0000028	short	Fishing for Goldfish	La pêche aux poissons rouges	0	1895	\N	1	Documentary, Short
tt0000029	short	Baby's Dinner	Repas de bébé	0	1895	\N	1	Documentary, Short
tt0000030	short	Rough Sea at Dover	Rough Sea at Dover	0	1895	\N	1	Documentary, Short

# Create External Tables In Hive

6. Создайте внешнюю таблицу `title_basics` для файла `title.basics.tsv` в Hive.

```
CREATE EXTERNAL TABLE IF NOT EXISTS title_basics (  
    tconst STRING,  
    title_type STRING,  
    primary_title STRING,  
    original_title STRING,  
    is_adult DECIMAL(1,0),  
    start_year DECIMAL(4,0),  
    end_year STRING,  
    runtime_minutes INT,  
    genres STRING  
) COMMENT 'IMDb Movies' ROW FORMAT DELIMITED FIELDS TERMINATED BY  
'\t' STORED AS TEXTFILE LOCATION '/user/hadoop/imdb/title_basics'  
TBLPROPERTIES ('skip.header.line.count'='1');
```

# Create External Tables In Hive

7. Query Table `title_basics` in Hive using SQL (HiveQL):

```
hive> select * from title_basics limit 3;
```

```
hive> select * from title_basics limit 3;
```

```
OK
```

tt0000001	short	Carmencita	Carmencita	0	1894	NULL	1	Documentary,Short
tt0000002	short	Le clown et ses chiens	Le clown et ses chiens	0	1892	NULL	5	Animation,Short
tt0000003	short	Pauvre Pierrot	Pauvre Pierrot	0	1892	NULL	4	Animation,Comedy,Romance

8. Query Table `title_ratings` in Hive using SQL (HiveQL):

```
hive> select * from title_ratings limit 3;
```

```
hive> select * from title_ratings limit 3;
```

```
OK
```

```
tt0000001      5.7      2033
```

```
tt0000002      5.7      272
```

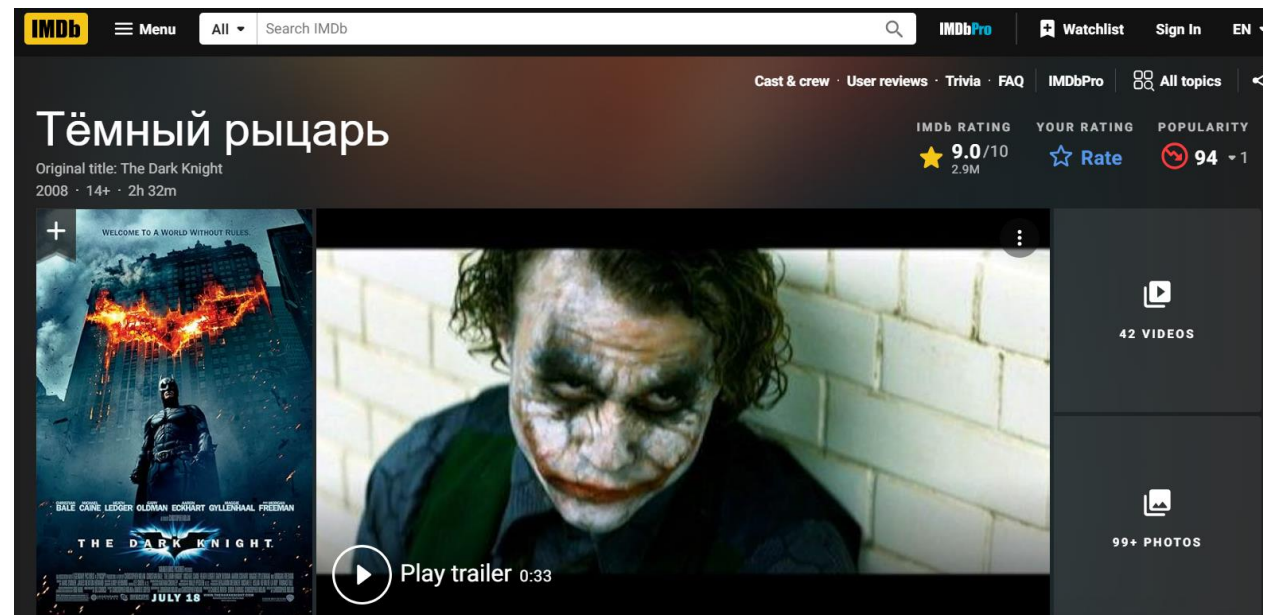
```
tt0000003      6.5      1976
```

```
Time taken: 0.2 seconds, Fetched: 3 row(s)
```

# Create External Tables In Hive

9. Запустите сложный запрос, который запускает задание MapReduce в Yarn, например. получить рейтинг фильма „*The Dark Knight*“:

```
SELECT *  
FROM title_basics b  
JOIN title_ratings r ON (b.tconst=r.tconst)  
WHERE original_title = 'The Dark Knight'  
AND title_type='movie';
```



# Create External Tables In Hive

## 9. Выполнить запрос


hive> **SELECT \* FROM title\_basics b JOIN title\_ratings r ON (b.tconst=r.tconst) WHERE original\_title = 'The Dark Knight' and title\_type='movie';**

```
hive> SELECT * FROM title_basics b JOIN title_ratings r ON (b.tconst=r.tconst) WHERE original_title = 'The Dark Knight' and title_type='movie';
Query ID = hadoop_20240310222915_7869d728-7f12-47e7-b425-f727894b3324
Total jobs = 2
Stage-5 is selected by condition resolver.
Stage-1 is filtered out by condition resolver.
SLF4J: Found binding in [jar:file:/home/hadoop/hive/lib/log4j-slf4j-impl-2.10.0.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.apache.logging.slf4j.Log4jLoggerFactory]
2024-03-10 22:29:24 Processing rows: 200000 Hashtable size: 199999 Memory usage: 158212808 percentage: 0.033
2024-03-10 22:29:24 Processing rows: 300000 Hashtable size: 299999 Memory usage: 208670624 percentage: 0.044
2024-03-10 22:29:24 Processing rows: 400000 Hashtable size: 399999 Memory usage: 249595512 percentage: 0.052
2024-03-10 22:29:24 Processing rows: 500000 Hashtable size: 499999 Memory usage: 298705408 percentage: 0.063
Execution completed successfully
MapredLocal task succeeded
Launching Job 2 out of 2
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1710109559597_0001, Tracking URL = http://60a86839349b:8088/proxy/application_1710109559597_0001/
Kill Command = /home/hadoop/hadoop/bin/mapred job -kill job_1710109559597_0001
Hadoop job information for Stage-3: number of mappers: 4; number of reducers: 0
2024-03-10 22:29:42,870 Stage-3 map = 0%, reduce = 0%
2024-03-10 22:30:07,278 Stage-3 map = 13%, reduce = 0%, Cumulative CPU 25.92 sec
2024-03-10 22:30:08,375 Stage-3 map = 75%, reduce = 0%, Cumulative CPU 32.1 sec
2024-03-10 22:30:09,398 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 34.69 sec
MapReduce Total cumulative CPU time: 34 seconds 690 msec
Ended Job = job_1710109559597_0001
MapReduce Jobs Launched:
Stage-Stage-3: Map: 4 Cumulative CPU: 34.69 sec HDFS Read: 911874125 HDFS Write: 463 SUCCESS
Total MapReduce CPU Time Spent: 34 seconds 690 msec
OK
tt0468569 movie The Dark Knight The Dark Knight 0 2008 NULL 152 Action,Crime,Drama tt0468569 9.0 2849163
Time taken: 55.686 seconds, Fetched: 1 row(s)
```



# Create External Tables In Hive

## 9. YARN (<http://XXX.XXX.XXX.XXX:8088/cluster/>):



All Applications

Cluster

[About](#)  
[Nodes](#)  
[Node Labels](#)  
[Applications](#)  
[NEW](#)  
[NEW SAVING](#)  
[SUBMITTED](#)  
[ACCEPTED](#)  
[RUNNING](#)  
[FINISHED](#)  
[FAILED](#)  
[KILLED](#)  
[Scheduler](#)

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Mem
1	0	0	1	0	0 B	16 GB

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes
1	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation
Capacity Scheduler	[memory-mb (unit=M), vcores]	<memory:1024, vCores:1>

Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus
<a href="#">application_1710106073770_0001</a>	hadoop	SELECT * FROM title_ba...title_type='movie' (Stage-3)	MAPREDUCE	default	0	Mon Mar 11 01:07:30 +0300 2024	Mon Mar 11 01:09:10 +0300 2024	FINISHED	SUCCEEDED

# Упражнения

Hive: создание внешних таблиц и работа с ними



# HDFS и Hive QL упражнения - IMDB



1. Подготовить среду Hive.
2. Скачать <https://datasets.imdbws.com/name.basics.tsv.gz>
3. Создать каталог в HDFS /user/hadoop/imdb/name\_basics/ для файла **name.basics.tsv**
4. Создайте внешнюю Hive таблицу `name_basics` для **name.basics.tsv**
5. Используйте HiveQL, чтобы ответить на следующие вопросы::
  - a) Сколько фильмов и сериалов находится в наборе данных IMDB?
  - b) Кто самый молодой актер/сценарист/... в наборе данных?
  - c) Создайте список (`tconst`, `original_title`, `start_year`, `average_rating`, `num_votes`), который состоит из:
    - фильм вышел в 2010 году или позднее;
    - фильм имеет средний рейтинг, равный или превышающий 8,1
    - проголосовали более 100 000 раз за фильм.
  - d) Сколько фильмов находится в списке c)?

# HDFS и Hive QL упражнения - IMDB



6. Используйте **HiveQL**, чтобы ответить на следующие вопросы:

е) Необходимо узнать, в какой промежуток времени(годы) был «великим» для кинематографа. Создайте список, в котором одна строка – год, другая строка – количество фильмов, которые:

- имеют средний рейтинг выше 8;
- были оценены более чем 100 000 раз, сортировку провести в порядке убывания количества фильмов в год.

СПАСИБО ЗА ВНИМАНИЕ