ШКОЛА БОЛЬШИХ ДАННЫХ

# Apache Hive

## Руководство по лабораторным работам

## День 1

версия 2.1 от 04/12/2019

**Москва, 2019**
www.bigdataschool.ru

# Содержание

## Подключение к Hive на кластере Hadoop на платформе EC2 AWS.

1. Получите у преподавателя индивидуальный номер ученика XY
2. Установите подключение через OpenVPN с лабораторной сетью. Используйте настройки в файле **Student0X.ovpn**
3. Запустите терминал PuTTY и настройте SSH соединение с узлом кластера указанный вам преподавателем. Для аутентификации используйте файл **Hadoop01.руь**. При входе укажите имя пользователя **ec2-user**.
4. Запустите Hive командой:

```
hive
```

5. Или командой ( поменяйте IP на имя сервера с установленным экземпляром Hive Server 2):

```
beeline jdbc:hive2://172.31.53.147:10000 ec2-user ec2-user
```

6. Просмотрите список доступных баз данных с помощью команды:

```
show databases;
```

## Работа с внешними (external) таблицами в Hive

1. Создайте папку s**tocksXY** в **HDFS** командой:

```
!hadoop fs -mkdir stocksXY;
```

при работе через **beeline** используйте другой формат:

```
!sh hadoop fs -mkdir stocksXY
```

2. Загрузите файл **sample.txt** из локальной папки **input** в созданную папку с помощью команды:

```
!hadoop fs -put input/sample.txt stocksXY/sample.txt;
```

3. Создайте новую базу данных **hadoopXY**:

```
create database hadoopXY;
```

4. Подключитесь к созданной базе данных:

```
use hadoopXY;
```

5. Создайте внешнюю таблицу в созданной базе данных:

```
CREATE EXTERNAL TABLE stocks_tb (
exch STRING,
symbol STRING,
ymd STRING,
price_open FLOAT,
price_high FLOAT,
price_low FLOAT,
price_close FLOAT,
volume INT,
price_adj_close FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/ec2-user/stocksXY';
```

6. Сделайте выборку 10 первых значений из таблицы stocks_tb (используйте выражение Limit 10 в конце запроса)
7. Удалите файл sample.txt из папки stocksXY в файловой системе HDFS
8. Сделайте выборку 10 первых значений из таблицы stocks_tb
9. Скопируйте в папку stocksXY файл medium.txt
10. Проверьте количество строк в таблице. Повторите проверку последовательно добавляя файлы sample.txt и stocks.txt
11. Какие выводы вы можете сделать по работе с внешними таблицами Hive. (Очередность загрузки данных? Дублирующие значения? Приниципы работы с внешними таблицами?)

## Работа с управляемыми (managed) таблицами в Hive

1. Создайте управляемую таблицу **stocks_mg** в созданной ранее базе данных:

```
CREATE TABLE stocks_mg (
exch STRING,
symbol STRING,
ymd STRING,
price_open FLOAT,
price_high FLOAT,
```

```
price_low FLOAT,
price_close FLOAT,
volume INT,
price_adj_close FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

2. Сделайте выборку 10 первых значений из таблицы stocks_mg. Данная таблица не имеет значений поэтому нужно загрузить текстовый файл для наполнения таблицы
3. Создайте папку inputXY в HDFS
4. Загрузите файл stocks.txt из локальной папки input в созданную папку inputXY
5. Загрузите данные в таблицу командой:

```
LOAD DATA INPATH 'InputXY'
INTO TABLE stocks_mg;
```

6. Сделайте выборку 10 первых значений из таблицы stocks_mg. Проверьте содержимое директории inputXY. Сделайте выводы.
7. Проверьте свойства созданных вами внешней и управляемой таблицы с помощью команды:

```
DESCRIBE FORMATTED table_name;
```

8. Найдите значение аттрибута location и проверьте файловую систему на наличие данных для обоих таблиц в указанном location
9. Какие выводы вы можете сделать по работе с внешними и внутренними таблицами Hive? (Очередность загрузки данных? Дублирующие значения? Приниципы работы с внешними и внутренними таблицами? Где хранятся данные таблиц?)
   **Примечание:** внешние таблицы надо использовать, когда данные должны быть доступны не только для HIVE, но и для внешних компонент и приложений.

## Создание таблиц с использование SELECT/INSERT

1. Создайте таблицу с помощью выполнения SQL запроса с SELECT:

```
CREATE TABLE stocks_ctas
AS
SELECT * FROM stocks_mg;
```

2. Обратите внимание что создание табюлицы SELECTом из существующей выполняется с помощью MapReduce задачи и требует двух операций копирования данных (временный файл и файл таблицы stocks_ctas)
3. Напишите запрос, выводящий количество строк в таблице stocks_ctas
4. Другой вариант создания таблицы запросом SELECT:

```
INSERT INTO TABLE stocks_ctas
SELECT s.* FROM stocks_mg s;
```

5. Повторите запрос, выводящий количество строк в таблице stocks_ctas. Обратите внимание что количество строк в таблице удвоилось.
6. Для того чтобы перезаписать существующую таблицу данных используйте команду OVERWRITE:

```
INSERT OVERWRITE TABLE stocks_ctas
SELECT s.* FROM stocks_mg s;
```

7. Вместо использования расположения по умолчанию при создании таблиц указывайте при создании каталог в файловой системе HDFS где будут храниться файлы таблиц Hive для исключения операций копирования-перемещения при загрузке значений в таблицу:

```
CREATE TABLE stocks_mg (
exch STRING,
symbol STRING,
ymd STRING,
price_open FLOAT,
price_high FLOAT,
price_low FLOAT,
price_close FLOAT,
volume INT,
price_adj_close FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/ec2-user/input/hive/stocks_db';
```

## Стандартные операции SELECT

1. Операции SELECT имеют стандартный синтаксис SQL. Простой SELECT:

```
SELECT * FROM stocks_mg
WHERE symbol = 'KBK';
```

2. SELECT с выбором значений колонки symbol, равным 'KBK' или 'K9B':

```
SELECT * FROM stocks_mg
WHERE symbol IN ('KBK', 'K9B');
```

3. SELECT с выбором значений из нескольких колонок где колонка exch начинается с ABC и колонка symbol имеет значение K.E где "." любой символ:

```
SELECT * FROM stocks_mg
WHERE exch LIKE 'ABC%' and symbol RLIKE 'K.E';
```

4. Сложная выборка с формированием дополнительного поля диапазонами CASE:

```
SELECT symbol, price_open, price_close, volume,
        CASE
                WHEN volume < 20000 THEN 'low'
                WHEN volume >= 20000 AND volume < 40000 THEN 'middle'
                WHEN volume >= 40000 AND volume < 60000 THEN 'high'
                ELSE 'very high'
END AS volume_level
FROM stocks_mg
WHERE symbol = 'КВК';
```

5. Использование DISTINCT для поиска уникальных комбинаций значений:

```
SELECT DISTINCT exch, symbol FROM stocks_mg;
```

6. Группировка по значению GROUP BY (в данном запросе мы выводим значения колонок "год" и "символ" с подсчетом среднего значения обьема сделок по ним avg(volume) и последующей группировкой по этим колонкам):

```
SELECT year(ymd), symbol, avg(volume) FROM stocks_mg
GROUP BY year(ymd), symbol;
```

7. Группировка по значению GROUP BY только строк, где значение удовлетворяет условию "среднее значение объема> 400000":

```
SELECT year(ymd), symbol, avg(volume) FROM stocks_mg
GROUP BY year(ymd), symbol
HAVING avg(volume) > 400000;
```

8. Иногда вывод необходимо сохранить для последующего использования в виде таблицы, хранящейся в локальной файловой системе или в HDFS:

```
INSERT OVERWRITE LOCAL DIRECTORY 'output/hive/stocks'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT year(ymd), symbol, avg(volume) FROM stocks_mg
GROUP BY year(ymd), symbol
HAVING avg(volume) > 400000;
```

9. LOCAL указывает, что файл необходимо сохранить в локальной файловой системе. При подключении через beeline, необходимо сохранить вывод в файловой системе HDFS, а затем скачать его командой hadoop fs -get. Ключ OVERWRITE указывает, что данная директория будет перезаписанна

## Использование ORDER BY/ SORT BY/ DISTRIBUTE BY

1. Hive использует MapReduce для выполнения выражений по разному. Выполним SELECT с параметром ORDER BY для таблицы stocks_mg:

```
SELECT * FROM stocks_mg
ORDER BY price_close DESC;
```

2. Обратите внимание что для выполнения запроса используется 1 Reducer, так как ORDER BY обращается глобально ко всем данным Dataset это может вызвать проблемы с нехваткой памяти и значительным снижением производительности (время выполнения запроса).
3. Изменить параметр выделения памяти для процессов Mapper и Reducer а также количество процессов mapper и reducer при запуске отдельной задачи Hive можно либо редактированием файла конфигурации hive-site.xml или установкой

```
SET mapreduce.job.reduces=3;
```

4. Проверьте выполнение запроса SELECT .... ORDER BY снова. Сколько reducers используется сейчас?
5. Заменим ORDER BY на SORT BY. Сколько reducers используется сейчас?
6. Для анализа выходных данных вставим данные выборки в таблицу в локальной файловой системе

```
INSERT OVERWRITE LOCAL DIRECTORY '/home/ec2-user/output/hive/stocks'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT ymd, symbol, price_close
FROM stocks_big WHERE year(ymd) = '2003'
SORT BY symbol ASC, price_close DESC;
```

7. Проверим выбранную директорию /home/ec2-user/output/hive/stocks на наличие файлов выборки. В зависимости от количества reducers мы получим такое же количество выходных файлов (3 в нашем случае). Проверьте каждый файл поделитесь своими наблюдениями с остальными.
8. Для правильной сортировки данных по "своим" reducers (без дублирования symbols в разных reducers) необходимо провести распределение по symbol. Измените запрос добавив:

```
DISTRIBUTE BY symbol
SORT BY symbol ASC, price_close DESC;
```

9. Выполните обновленный запрос и проверьте полученные результаты каждого файла-выборки.
10. Учитывая, что размер нашей выборки не большой, наличие дополнительных reducers не сказывается на производительности запроса. Попробуйте создать таблицу с максимальной выборкой и провести сравнение производительности исполнения запроса с ORDER BY и DISTRIBUTE BY & SORT BY
11. Если выполняете распределение (DISTRIBUTE BY) и сортировку(SORT BY) по одной колонке в нарастающем порядке (ASC order) то вы можете заменить данные выражения на CLUSTER BY:

```
INSERT OVERWRITE LOCAL DIRECTORY '/home/ec2-user/output/hive/stocks'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT ymd, symbol, price_close
FROM stocks_big WHERE year(ymd) = '2003'
CLUSTER BY symbol;
```

12. **Самостоятельное задание: возьмите файл stocks-big для создания таблицы stock**

## Создание партиций (PARTITION)

1. Для разбиения данных на разделы и улучшения скорости выполнения запросов к данным принадлежащим одному разделу мы можем организовать хранение больших dataset в партициях.
2. Выполните SELECT по symbol="ZZZ" и времени '2001-07-03'.
3. Создадим новую таблицу с использование PARTITIONED BY

```
CREATE TABLE stocks_partition (
        exch STRING,
        symbol STRING,
        ymd STRING,
        price_open FLOAT,
        price_high FLOAT,
        price_low FLOAT,
        price_close FLOAT,
        volume INT,
        price_adj_close FLOAT)
PARTITIONED BY (sym STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

4. Посмотрим формат созданной таблицы с помощью DESCRIBE FORMATTED table_name; В качестве партиций будет использоваться значение колонки sym (не путать с symbol).
5. Для загрузки данных в stocks_partition: загружаем данные в партицию, статически указывая желаемую партицию с помощью INSERT OVERWRITE:

```
INSERT OVERWRITE TABLE stocks_partition
PARTITION (sym = 'B7J')
SELECT * FROM stocks s
WHERE s.symbol = 'B7J'
```

6. Проверяем детали созданной партиции с помощью команды:

```
SHOW PARTITIONS stocks_partition;
```

7. и проверяем LOCATION указанный в выводе команды DESCRIBE FORMATTED для таблицы stocks_partition

8. Добавляем в ручном режиме данные в партицию 'ZZZ'  и 'KJT'.
9. Добавлять партиции можно и по LOCATION (расположение каталога файлов). С помощью INSERT OVERWRITE DIRECTORY создадим каталог с выборкой по symbol ='ZXY':

```
INSERT OVERWRITE DIRECTORY 'output/hive/stockspart-zxy'
SELECT *
FROM stocks WHERE symbol='ZXY';
```

10. изменим таблицу stocks_partition указав новую партицию с помощью LOCATION:

```
ALTER TABLE stocks_partition ADD IF NOT EXISTS
PARTITION (sym = 'ZXY') LOCATION 'output/hive/stockspart-zxy';
```

11. При необходимости можно удалить партицию с помощью DROP:

```
ALTER TABLE stocks_partition DROP IF EXISTS PARTITION (sym = 'ZXY');
```

## Создание динамических партиций (DYNAMIC PARTITION)

1. Для работы с большими данными и в автоматическом режиме нужно включить поддержку динамических партиций. Устанавливаем параметры:

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
```

2. Импортируем данные из таблицы stocks с автоматическим созданием партиций по колонке symbol:

```
INSERT OVERWRITE TABLE stocks_partition
PARTITION (sym)
SELECT s.*, s.symbol
FROM stocks s;
```

## Создание многоуровневых динамических партиций (DYNAMIC PARTITION)

1. На практике естественно вас интересует создание многоуровневых партиций в автоматическом режиме. Для примера рассмотрим создание таблицы с динамическими партициями по названию биржи (exch_name='ABCSE'), году (yr) и символу ценных бумаг (sym):

```
CREATE TABLE IF NOT EXISTS stocks_dynamic_partition (
        exch STRING,
        symbol STRING,
        ymd STRING,
        price_open FLOAT,
```

```
        price_high FLOAT,
        price_low FLOAT,
        price_close FLOAT,
        volume INT,
        price_adj_close FLOAT)
PARTITIONED BY (exch_name STRING, yr STRING, sym STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

2. Проверьте свойства таблицы, созданной вами с применением динамических партиций:

```
DESCRIBE FORMATTED table_name;
```

3. Обратите внимание на созданные партиции
4. Для загрузки данных используем INSERT:

```
INSERT OVERWRITE TABLE stocks_dynamic_partition
PARTITION (exch_name='ABCSE', yr, sym)
SELECT *, year(ymd), symbol
FROM stocks_mg;
```

5. Обратите внимание что мы не загружаем таблицу stocks, которая содержит порядка 8 миллионов строк и 836 комбинаций биржа-сток-символ, а вместо этого используем таблицу меньшего размера на 500 тысяч строк stocks_mg.
6. Для увеличения количества разделов, создаваемых максимально на кластере и количества на datanode, измените следующие параметры:

```
SET hive.exec.max.dynamic.partitions=1000;
SET hive.exec.max.dynamic.partitions.pernode=500;
```

7. Однако даже после снижения количества данных мы все равно не сможем создать нужное количество разделов вследствии того, что кластер проверяет возможность создания партиций и размер файловой системы для хранения данных в 3х репликах для каждой партиции. Поэтому мы сделаем выборочную загрузку данных только по 2001-2003 году и символам начинающимся с symbol 'B..'. Составьте query самостоятельно используя пример запроса на загрузку данных выше.
8. Проверьте созданные вами разделы на файловой системе и с помощью команды:

```
SHOW PARTITIONS stocks_dynamic_partition;
```

9. Выполнение запросов на таблицах с разделами также отличается от простого SELECT. Установите режим:

```
SET hive.mapred.mode=strict;
```

10. Выполните запрос:

```
SELECT * FROM stocks_dynamic_part
WHERE yr=2003 and volume > 10000;
```

11. Попробуйте выполнить запрос простого SELECT без партиций:

```
SELECT * FROM stocks_dynamic_partition
WHERE volume > 10000;
```

12. Примечание: при работе с Partitioned Table в strict режиме простые запросы без указания разделов исполняться не будут (ошибка). Для анализа скорости выполнения запросов для datasets одной размерности необходимо наличие dataset предполагающего распределенное хранение данных на нескольких datanode и использующего сложные запросы обработки данных с несколькими mapper и reducer.

## Использование BUCKETS для оптимизации PARTITIONS

1. При неправильно выбранном количестве партиций производительность исполнения запросов может быть улучшена использованием BUCKETS для объединения в buckets. Так как основным инструментом до сих пор является DYNAMIC PARTITION незабудьте установить параметры максимального количества партиций на кластер и на узел и, кроме того, включить поддержку buckets командой:

```
SET hive.enforce.bucketing = true;
```

2. Для создания таблицы с поддержкой buckets модифицируйте скрипт создания таблицы, использовавшийся ранее, заменив:

```
PARTITIONED BY (exch_name STRING, yr STRING, sym STRING)
```

3. На:

```
PARTITIONED BY (exch_name STRING, yr STRING)
CLUSTERED BY (symbol) INTO 5 BUCKETS
```

4. создайте таблицу stocks_bucket
5. Проверьте свойства созданной таблицы и расположение таблицы stocks_bucket с помощью

```
DESCRIBE FORMATTED stocks_bucket;
```

6. Загрузите данные в таблицу с помощью запроса

```
INSERT OVERWRITE TABLE stocks_bucket
PARTITION (exch_name='ABCSE', yr)
SELECT *, year(ymd)
FROM stocks WHERE year(ymd) IN ('2001', '2002', '2003') and symbol like 'B%';
```

7. Проверьте загрузку данных в партиции в файловой системе HDFS. Вы увидите по 5 партиций в каждом году.
8. Проверьте выборку в таблице stocks с sampling

```
SELECT *
FROM stocks TABLESAMPLE(BUCKET 3 OUT OF 5 ON symbol) s;
```

9. Проверьте выборку sampling с buckets

```
SELECT * FROM stocks_bucket
TABLESAMPLE(BUCKET 3 OUT OF 5 ON symbol) s;
```

## Соединение таблиц JOIN

В ходе выполнения пунктов задания заполняйте следующую таблицу:

| № эксперимента | Тип соединения | Mappers | Reducers | Time (seconds) |
|---|---|---|---|---|
| **1** | | | | |
| **2** | | | | |
| **3** | | | | |

1. Просмотрите план выполнения запроса:

```
EXPLAIN SELECT s1.exch, s1.symbol, s2.ymd, s2.price_open
FROM stocks s1 JOIN stocks s2
ON s1.exch=s2.exch AND s1.symbol=s2.symbol AND s1.ymd=s2.ymd LIMIT 100;
```

2. Запишите тип соединения.
3. Выполните запрос:

```
SELECT s1.exch, s1.symbol, s2.ymd, s2.price_open
FROM stocks s1 JOIN stocks s2
ON s1.exch=s2.exch AND s1.symbol=s2.symbol AND s1.ymd=s2.ymd LIMIT 100;
```

4. Запишите число Mappers и Reducers.
5. Создайте таблицу dividends:

```
CREATE TABLE dividends(
exch STRING,
symbol STRING,
ymd STRING,
divi FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

6. Загрузите в нее данные из файла **dividends**

7. Просмотрите план выполнения запроса:

```
set hive.auto.convert.join=false;
EXPLAIN SELECT s.exch, s.symbol, s.ymd, d.divi
FROM stocks s JOIN dividends d
ON s.exch=d.exch AND s.symbol=d.symbol AND s.ymd=d.ymd
LIMIT 100;
```

8. Запишите тип соединения.
9. Выполните запрос:

```
SELECT s.exch, s.symbol, s.ymd, d.divi
FROM stocks s JOIN dividends d
ON s.exch=d.exch AND s.symbol=d.symbol AND s.ymd=d.ymd
LIMIT 100;
```

10. Запишите число Mappers и Reducers и время выполнения запроса.
11. Просмотрите план выполнения запроса:

```
set hive.auto.convert.join=true;
EXPLAIN SELECT s.exch, s.symbol, s.ymd, d.divi
FROM stocks s JOIN dividends d
ON s.exch=d.exch AND s.symbol=d.symbol AND s.ymd=d.ymd
LIMIT 100;
```

12. Запишите тип соединения.
13. Выполните запрос:

```
SELECT s.exch, s.symbol, s.ymd, d.divi
FROM stocks s JOIN dividends d
ON s.exch=d.exch AND s.symbol=d.symbol AND s.ymd=d.ymd
LIMIT 100;
```

14. Запишите число Mappers и Reducers и время выполнения запроса.
15. Сравните записанные значения, сделайте выводы.

## Сравнение форматов хранения CSV, AVRO, ORC и PARQUET

В процессе выполнения пунктов задания заполняйте следующую таблицу:

| Формат | Mappers | Reducers | Time (seconds) |
|---|---|---|---|
| CSV | | | |
| PARQUET | | | |
| ORC NONE | | | |
| ORC ZLIB | | | |

| | | | |
|---|---|---|---|
| ORC SNAPPY | | | |
| AVRO | | | |

1. Выполните запрос:

```
SELECT * FROM stocks
WHERE volume=131500
LIMIT 20;
```

2. Запишите число Mappers и Reducers и время выполнения запроса.
3. Создайте таблицу в формате PARQUET запросом:

```
CREATE TABLE stocks_p
STORED AS PARQUET
AS SELECT * FROM stocks;
```

4. Выполните запрос:

```
SELECT * FROM stocks_p
WHERE volume=131500
LIMIT 20;
```

5. Запишите число Mappers и Reducers и время выполнения запроса.
6. Создайте таблицу в формате ORC без сжатия запросом:

```
CREATE TABLE stocks_on
STORED AS ORC TBLPROPERTIES("orc.compress"="NONE")
AS SELECT * FROM stocks;
```

7. Выполните запрос:

```
SELECT * FROM stocks_on
WHERE volume=131500
LIMIT 20;
```

8. Запишите число Mappers и Reducers и время выполнения запроса.
9. Создайте таблицу в формате ORC с сжатием ZLIB запросом:

```
CREATE TABLE stocks_oz
STORED AS ORC TBLPROPERTIES("orc.compress"="ZLIB")
AS SELECT * FROM stocks;
```

10. Выполните запрос:

```
SELECT * FROM stocks_oz
```

```
WHERE volume=131500
LIMIT 20;
```

11. Запишите число Mappers и Reducers и время выполнения запроса.
12. Создайте таблицу в формате ORC с сжатием SNAPPY запросом:

```
CREATE TABLE stocks_os
STORED AS ORC TBLPROPERTIES("orc.compress"="SNAPPY")
AS SELECT * FROM stocks;
```

13. Выполните запрос:

```
SELECT * FROM stocks_os
WHERE volume=131500
LIMIT 20;
```

14. Запишите число Mappers и Reducers и время выполнения запроса.
15. Создайте таблицу в формате AVRO запросом:

```
CREATE TABLE stocks_a
STORED AS AVRO
AS SELECT * FROM stocks;
```

16. Выполните запрос:

```
SELECT * FROM stocks_a
WHERE volume=131500
LIMIT 20;
```

17. Запишите число Mappers и Reducers и время выполнения запроса.
18. Сравните записанные значения, сделайте выводы.
19. Заполните следующую таблицу:

| Формат | Mappers | Reducers | Time (seconds) |
|---|---|---|---|
| CSV | | | |
| PARQUET | | | |
| ORC NONE | | | |
| ORC ZLIB | | | |
| ORC SNAPPY | | | |
| AVRO | | | |

20. Модифицируя и выполняя запрос для таблиц разного типа хранения:

```
SELECT symbol, MAX(volume)
FROM stocks
GROUP BY symbol;
```