

Кафедра алгоритмов и технологий
программирования

SQL over BigData



Ивченко Олег



MapReduce на SQL

- SELECT
- WHERE
- ORDER BY / SORT BY
- GROUP BY
- JOIN
- HAVING



MapReduce на SQL

- SELECT - map
- WHERE - map
- ORDER BY / SORT BY - reduce
- GROUP BY - shuffle & sort (или счетчики) + тривиальный редьюсер
- JOIN - map + reduce
- HAVING - map + reduce + [map]



Hive



- Появился в 2007 г. в Facebook
 - SQL-обёртка (**не СУБД**)
 - Не только MapReduce (Spark...)
- Как структурировать данные?
 - Schema on read



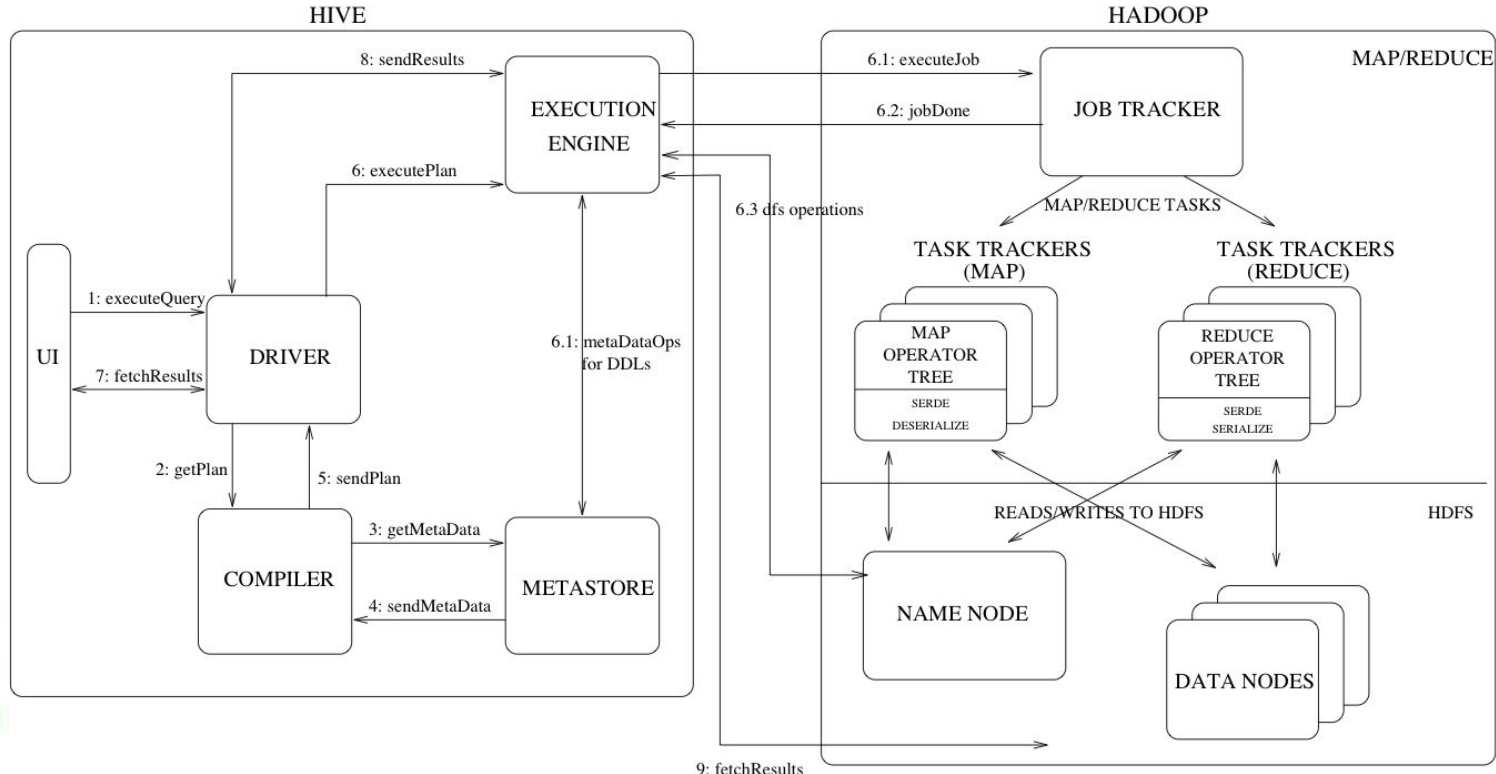
Hive. Выполнение запроса

- Parser - проверка синтаксиса
- Semantic Analyzer - проверка типов данных, раскрытие *, неявная конвертация типов
- Logical Plan Operator - построение дерева выполнения запросов, оптимизация запросов:
 - два join-а в 1 mapreduce задачу
 - reduce side join -> map side join
 - group by можно сделать разными способами (часть на map-стадии)
- Query Plan Generator - формирование данных по дереву выполнения запросов с учетом хранения данных в HDFS.



Hive

Ссылка с пояснением





Hive metastore

- Metastore хранит метаданные
 - базы данных (database) - пространство имен таблиц
 - таблицы (tables) - список столбцов, создателя, тип хранилища, десериализатор данных

0011150010111000110000101000000111000101111000110110001010000

00001110001011000010101000010100000011100010111000011001000101000000111000101111000110110001010000
00111000101110001011110100001010000001110001011100001100100010100000011100010111100011011000101000
00001110001011000010101000010100000011100010111000011001000101000000111000101111000110110001010000



Hive. Типы таблиц

- По влиянию на данные
 - **External** - не можем менять данные, только метаданные
 - **Managed** - можем менять данные
- По времени жизни
 - **Permanent** - существует всегда
 - **Temporary** - существует во время Hive сессии



АТП

➤ План запроса

Hive. Выполнение запроса

STAGE DEPENDENCIES:

```
Stage-4 is a root stage , consists of Stage-1
Stage-1
Stage-0 depends on stages: Stage-1
```

STAGE PLANS:

```
Stage: Stage-4
  Conditional Operator
```

```
Stage: Stage-1
```

```
  Map Reduce
```

```
    Map Operator Tree:
```

```
      TableScan
```

```
        alias: books
```

```
        Statistics: Num rows: 4469507 Data size: 911779456 Basic stats: COMPLETE Column stats: NONE
```

```
      Filter Operator
```

```
        predicate: (timestamp < 5000000) (type: boolean)
```

```
        Statistics: Num rows: 1489835 Data size: 303926349 Basic stats: COMPLETE Column stats: NONE
```

```
      Reduce Output Operator
```

```
        key expressions: uid (type: string)
```

```
        sort order: +
```

```
        Map-reduce partition columns: uid (type: string)
```

```
        Statistics: Num rows: 1489835 Data size: 303926349 Basic stats: COMPLETE Column stats: NONE
```

```
        value expressions: book_uid (type: string), timestamp (type: int)
```

```
      TableScan
```

```
        alias: users
```

```
        Statistics: Num rows: 422021 Data size: 43890184 Basic stats: COMPLETE Column stats: NONE
```

```
      Reduce Output Operator
```

```
        key expressions: uid (type: string)
```

```
        sort order: +
```

```
        Map-reduce partition columns: uid (type: string)
```

```
        Statistics: Num rows: 422021 Data size: 43890184 Basic stats: COMPLETE Column stats: NONE
```

```
        value expressions: age (type: int)
```

```
    Reduce Operator Tree:
```

```
      Join Operator
```

```
        condition map:
```

```
          Left Outer Join0 to 1
```

```
      keys:
```

```
        0 uid (type: string)
```

```
        1 uid (type: string)
```



Hive. Пример

➤ Данные:

- IP адрес
- маска сети

➤ Датасет:

- `/data/subnets/big/` | 7Gb
- `/data/subnets/small`

```
145.236.79.32 255.255.255.240
56.194.112.48 255.255.255.248
96.72.105.224 255.255.255.224
163.198.20.80 255.255.255.240
210.26.61.0 255.255.255.128
169.211.253.0 255.255.255.0
174.101.220.128 255.255.255.128
242.165.115.184 255.255.255.252
59.213.9.96 255.255.255.224
36.129.209.8 255.255.255.248
```

Посчитать среднее кол-во адресов по сетям.

Выведите план запроса и посчитайте по нему кол-во

MapReduce Job.

1. Создать “базу данных”
2. Создать таблицу (типы и список полей, как парсить данные)
3. Выполнить запрос



Hive. Пример

1. Создать “базу данных”

```
CREATE DATABASE <YOUR_DB> LOCATION '/user/<YOUR_USER>/test_warehouse' ;
```



Hive. Пример

1. Создать “базу данных”

```
CREATE DATABASE <YOUR_DB> LOCATION '/user/<YOUR_USER>/test_dwh';
```

2. Создать таблицу

```
USE <YOUR_DB>;
```

```
DROP TABLE IF EXISTS Subnets;
```

```
CREATE EXTERNAL TABLE Subnets (
```

```
    ip STRING,
```

```
    mask STRING
```

```
) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
```

```
STORED AS TEXTFILE
```

```
LOCATION '/data/subnets/big';
```



Hive. Пример

3. Напишите запрос на SQL (HiveQL \approx SQL) Что показывает **EXPLAIN [EXTENDED]**?

Посчитать среднее кол-во адресов по маскам подсетей. По задаче выведите план запроса и посчитайте по нему кол-во MapReduce Job.

```
SELECT AVG(LENGTH(SUBSTR(mask, 1, 16))) AS avg_mask_length FROM ip_masks
```

```
EXPLAIN
```

```
EXPLAIN EXTENDED
```



Hive. Пример

3. Напишите запрос на SQL (HiveQL \approx SQL)
Что показывает **EXPLAIN**?

4. Т.к. есть структура, можно оптимизировать хранение. См.

- partitioning
- [clustering \(bucketing\)](#)
- несбалансированные данные ([skewed by](#))



Hive. Оптимизации

- Партиционирование
- Кластеризация
- Семплирование
- Map-side Join



Не Hive'ом единым

➤ Apache Pig. Свой язык вместо SQL

```
input_lines = LOAD '/tmp/my-copy-of-all-pages-on-internet' AS (line:chararray);

-- Extract words from each line and put them into a pig bag
-- datatype, then flatten the bag to get one word on each row
words = FOREACH input_lines GENERATE FLATTEN(TOKENIZE(line)) AS word;

-- filter out any words that are just white spaces
filtered_words = FILTER words BY word MATCHES '\\w+';

-- create a group for each word
word_groups = GROUP filtered_words BY word;

-- count the entries in each group
word_count = FOREACH word_groups GENERATE COUNT(filtered_words) AS count, group AS word;

-- order the records by count
ordered_word_count = ORDER word_count BY count DESC;
STORE ordered_word_count INTO '/tmp/number-of-words-on-internet';
```





Не Hive'ом единым

- Cloudera Impala
 - Полноценная база данных,
 - нет поддержки сложных типов,
 - работает намного быстрее (в RAM) для простых запросов
 - входит в поставку Cloudera => легко развернуть



17



Не Hive'ом единым

- Apache Presto
 - Ещё одна Java-based СУБД для BigData
 - **Tutorial по Presto**





Apache Spark



<https://spark.apache.org/>



Вопросы?

