
HIVE

Hive

- Работает поверх Hadoop
- Предоставляет SQL-подобный язык (*HiveQL)*
- Разработан в Facebook в 2007 году
- Проект Apache Hadoop

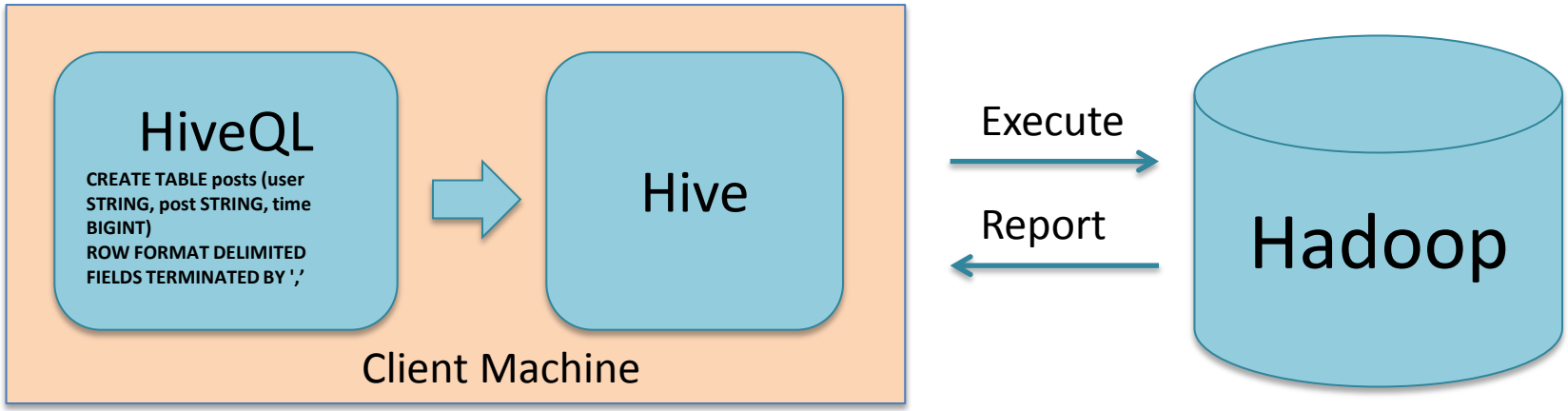
  http://hive.apache.org

Hive предоставляет

- Возможность структурировать различные форматы данных
- Простой интерфейс для запросов, анализа и обобщения больших объемов данных
- Доступ к данных из различных источников, таких как HDFS и HBase

Hive

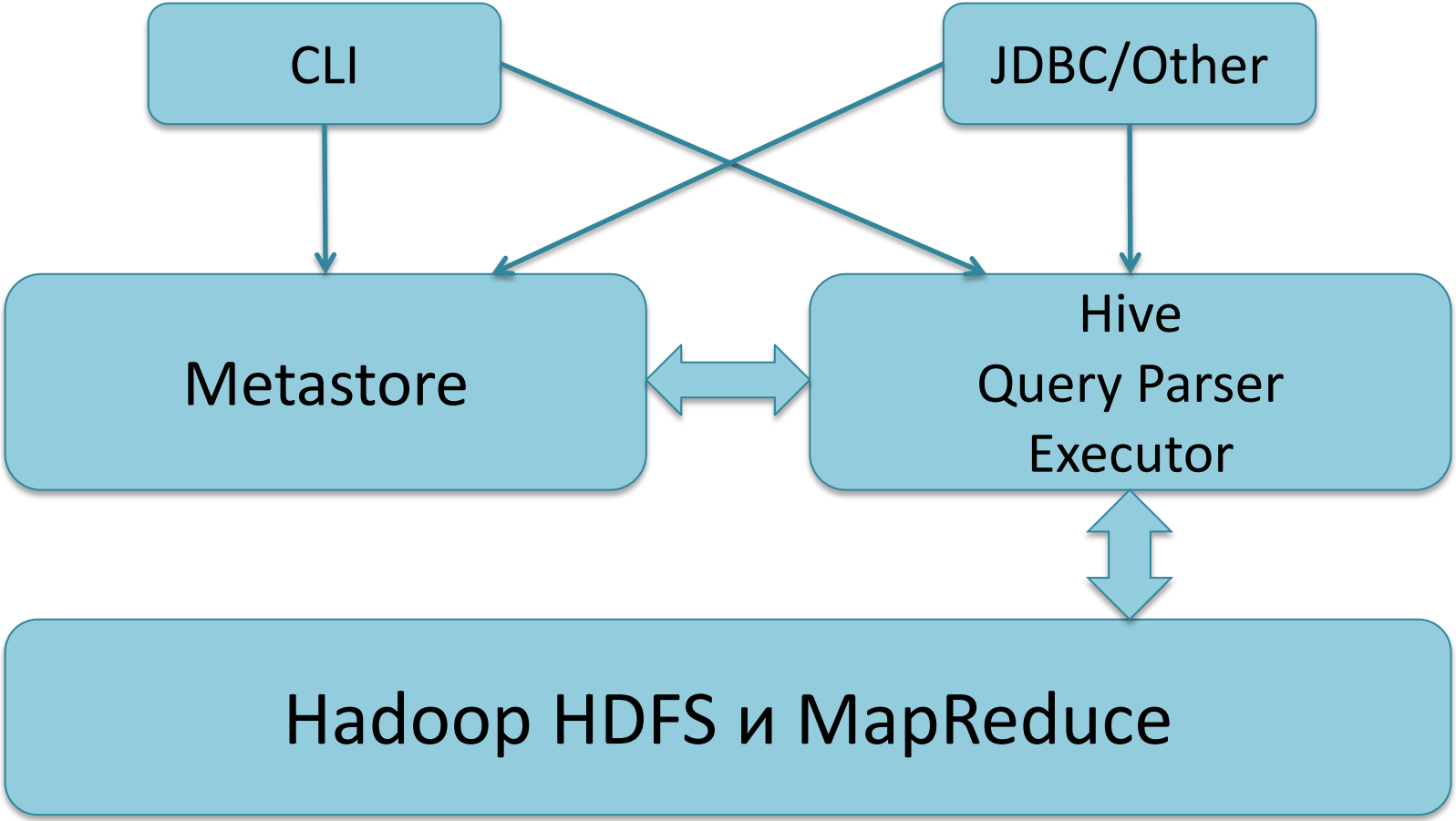
- Hive НЕ предоставляет:
  - Low latency или realtime-запросы
- Запрос даже небольших объемов данных может занять минуты
- Разработан с учетом масштабируемости и легкости использования


HiveQL
CREATE TABLE posts (user STRING, post STRING, time BIGINT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
Hive
Client Machine
Execute
Report
Hadoop

Hive

Hive хранит мета-информацию в реляционной БД

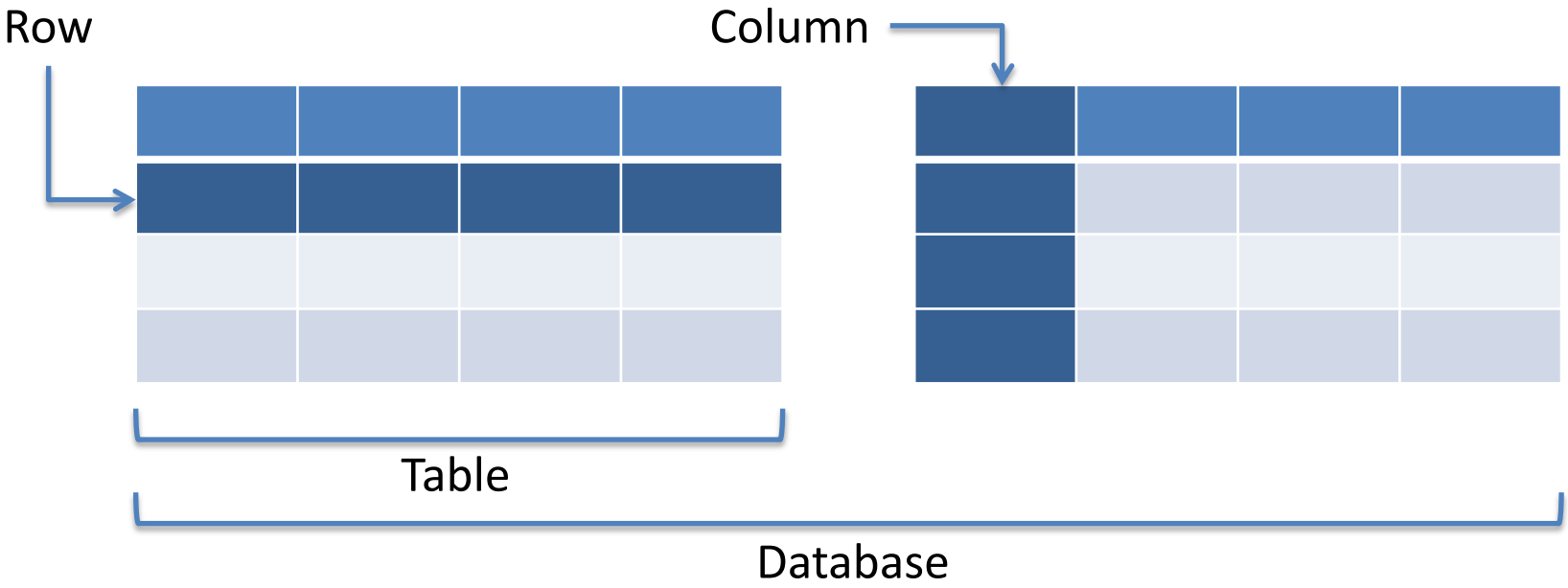
- Поставляется с Derby, “легковесной” встроенной SQL DB
- Можно относительно легко переключиться на другу БД, например, MySQL


CLI
JDBC/Other
Metastore
Hive
Query Parser
Executor
Hadoop HDFS и MapReduce

Интерфейс Hive

- Command Line Interface (CLI)
- Hive Web Interface
- Java Database Connectivity (JDBC)

Концепция Hive



Hive: пример

1. Создать таблицу
2. Загрузить данные в таблицу
3. Сделать запрос к таблице
4. Удалить таблицу

Hive: Создание таблицы

```
$ hive
hive> !cat data/user-posts.txt;
user1,Funny Story,1343182026191
user2,Cool Deal,1343182133839
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
hive>
```

```
hive> CREATE TABLE posts (user STRING, post STRING, time BIGINT)
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > STORED AS TEXTFILE;
OK
Time taken: 10.606 seconds

hive> show tables;
OK
posts
Time taken: 0.221 seconds

hive> describe posts;
OK
user string
post string
time bigint
Time taken: 0.212 seconds
```

- Создать таблицу из 3-х колонок
- Как будут разделяться поля
- Как сохранять данные

Показать схему таблицы

```
hive> LOAD DATA LOCAL INPATH 'data/user-posts.txt'
    > OVERWRITE INTO TABLE posts;
Copying data from file: data/user-posts.txt
Copying file: data/user-posts.txt
Loading data to table default.posts
Deleted /user/hive/warehouse/posts
OK
Time taken: 5.818 seconds
hive>
```

Существующие записи в таблице *posts* удаляются
Данные из *data/user-posts.txt* загружены в таблицу *posts*

```
$ hdfs dfs -cat /user/hive/warehouse/posts/user-posts.txt
user1,Funny Story,1343182026191
user2,Cool Deal,1343182133839
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
```

По-умолчанию Hive хранит свои таблицы в */user/hive/warehouse*

```
hive> select count(1) from posts;
Total MapReduce jobs = 1
Launching Job 1 out of 1
...
Starting Job = job_1343957512459_0004, Tracking URL =
http://localhost:8088/proxy/application_1343957512459_0004/
Kill Command = hadoop job -Dmapred.job.tracker=localhost:10040 -kill
job_1343957512459_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2014-08-02 22:37:24,962 Stage-1 map = 0%, reduce = 0%
2014-08-02 22:37:31,577 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 0.87 sec
2014-08-02 22:37:32,664 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 2.64 sec
MapReduce Total cumulative CPU time: 2 seconds 640 msec
Ended Job = job_1343957512459_0004
MapReduce Jobs Launched:
Job 0: Map: 1 Reduce: 1 Accumulative CPU: 2.64 sec HDFS Read: 0 HDFS Write: 0
SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 640 msec
OK
4
Time taken: 14.204 seconds
```

- Считаем кол-во записей в таблице *posts*
- HiveQL преобразуется в 1 MapReduce задачу

```
hive> select * from posts where user="user2";
...
...
OK
user2 Cool Deal 1343182133839
Time taken: 12.184 seconds
hive> select * from posts where time<=1343182133839 limit 2;
...
...
OK
user1 Funny Story 1343182026191
user2 Cool Deal 1343182133839
Time taken: 12.003 seconds
hive>
```

Выбрать записи пользователя *‘user2’*

- Фильтр по timestamp
- Ограничиваем число записей в результате

Hive: Удаление таблицы

```
hive> DROP TABLE posts;
OK
Time taken: 1.234 seconds
hive> exit;


$ hdfs dfs -ls /user/hive/warehouse/
$
```

Hive: Нарушение схемы

```
hive> !cat data/user-posts-error.txt;
user1,Funny Story,1343182026191
user2,Cool Deal,2012-01-05
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
hive> describe posts;
OK
user string
post string
time bigint
Time taken: 0.289 seconds
```

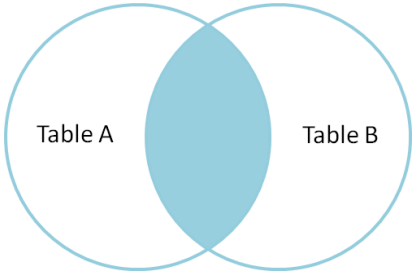
Hive: Нарушение схемы

```
hive> LOAD DATA LOCAL INPATH
    > 'data/user-posts-error.txt '
    > OVERWRITE INTO TABLE posts;
OK
Time taken: 0.612 seconds

hive> select * from posts;
OK
user1 Funny Story 1343182026191
user2 Cool Deal NULL
user4 Interesting Post 1343182154633
user5 Yet Another Blog 13431839394
Time taken: 0.136 seconds
hive>
```

Hive: Joins

- Hive реализует **inner *join*** по-умолчанию



- Имеет поддержку ***outer joins:***
  - left, right и full joins
- Можно объединять множество таблиц

```
hive> select * from posts limit 10;
OK
user1 Funny Story 1343182026191
user2 Cool Deal 1343182133839
user4 Interesting Post 1343182154633
user5 Yet Another Blog 1343183939434
hive> select * from likes limit 10;
OK
user1 12 1343182026191
user2 7 1343182139394
user3 0 1343182154633
user4 50 1343182147364
Time taken: 0.103 seconds
hive> CREATE TABLE posts_likes (user STRING, post STRING,
likes_count INT);
OK
```

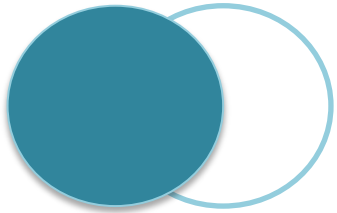
Hive: inner join

```
hive> INSERT OVERWRITE TABLE posts_likes
    > SELECT p.user, p.post, l.count
    > FROM posts p JOIN likes l ON (p.user = l.user);
OK
Time taken: 17.901 seconds
hive> select * from posts_likes limit 10;
OK
user1 Funny Story 12
user2 Cool Deal 7
user4 Interesting Post 50
Time taken: 0.082 seconds
hive>
```

Hive: Outer Joins

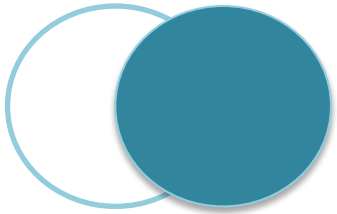
Left Outer

```
SELECT p.*, l.*
FROM posts p LEFT OUTER JOIN likes l ON (p.user = l.user)
limit 10;
```



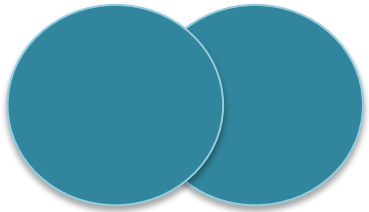
RIGHT Outer

```
SELECT p.*, l.*
FROM posts p RIGHT OUTER JOIN likes l ON (p.user = l.user)
limit 10;
```



FULL Outer

```
SELECT p.*, l.*
FROM posts p FULL OUTER JOIN likes l ON (p.user = l.user)
limit 10;
```



Hive: WordCount

```
CREATE TABLE docs (line STRING);

LOAD DATA INPATH 'docs' OVERWRITE INTO TABLE docs;

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

Pig vs Hive

- PigLatin vs HiveQL
- (Bag, Tuple, Field) vs (Table, Row, Column)
- Схема данных: свободная vs фиксированная