



## **Hive, разница между File и Row Format, RCFile**

**Драль Алексей**, [study@bigdatateam.org](mailto:study@bigdatateam.org)

CEO at BigData Team, <https://bigdatateam.org>

<https://www.facebook.com/bigdatateam>



# Hive DDL warm-up

```
create external table tab_dataset (  
    first_column string,  
    second_column string,  
    value int  
) location '/user/<user>/hive_practice_data/';
```

```
$ hdfs dfs -cat /user/<user>/hive_practice_data/*
```

```
first <tab> line <tab> 1  
second <tab> line <tab> 3  
last <tab> line <tab> 5  
\N <tab> \N <tab> 10
```



# Hive DDL warm-up

```
create external table tab_dataset (  
    first_column string,  
    second_column string,  
    value int  
) location '/user/<user>/hive_practice_data/';
```

```
$ hdfs dfs -cat /user/<user>/hive_practice_data/*
```

```
first <tab> line <tab> 1  
second <tab> line <tab> 3  
last <tab> line <tab> 5  
\N <tab> \N <tab> 10
```

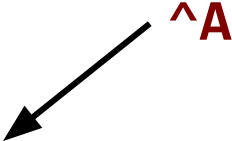
```
$ hive -e "use <user>; select * from tab_dataset;"
```

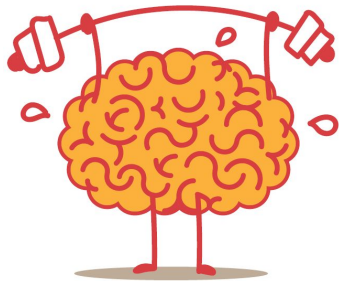
```
first    line    1    NULL    NULL  
second   line    3    NULL    NULL  
last     line    5    NULL    NULL  
\N       \N       10    NULL    NULL
```

```
Time taken: 3.805 seconds, Fetched: 4 row(s)
```



```
CREATE EXTERNAL TABLE tab_dataset (  
    first_column    STRING,  
    second_column  STRING,  
    value          INT  
  
)  
ROW FORMAT DELIMITED  
    FIELDS TERMINATED BY '\001'  
    COLLECTION ITEMS TERMINATED BY '\002'  
    MAP KEYS TERMINATED BY '\003'  
    LINES TERMINATED BY '\n'  
LOCATION '/user/<user>/hive_practice_data/';
```





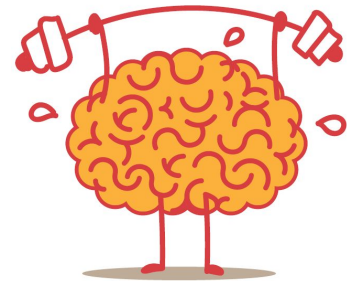
```
CREATE TABLE employees (  
    name          STRING,  
    salary        FLOAT,  
    subordinates  ARRAY<STRING>,  
    deductions    MAP<STRING, FLOAT>,  
    address       STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>);
```

```
John Doe^A100000.0^AMary Smith^BTodd Jones^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A1 Michigan Ave.^BChicago^BIL^B60600
```

```
Mary Smith^A80000.0^ABill King^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A100 Ontario St.^BChicago^BIL^B60601
```



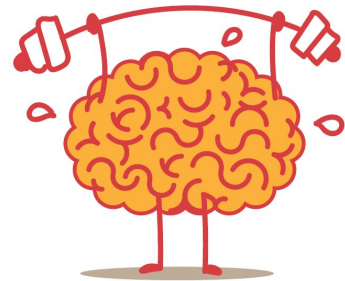
## Hive DDL



```
CREATE TABLE employees (  
    name          STRING,  
    salary        FLOAT,  
    subordinates  ARRAY<STRING>,  
    deductions    MAP<STRING, FLOAT>,  
    address       STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>);
```

```
John Doe^A100000.0^AMary Smith^BTodd Jones^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A1 Michigan Ave.^BChicago^BIL^B60600
```

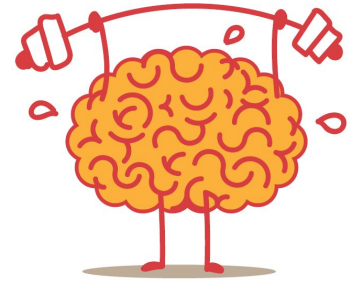
```
Mary Smith^A80000.0^ABill King^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A100 Ontario St.^BChicago^BIL^B60601
```



```
CREATE TABLE employees (  
    name          STRING,  
    salary        FLOAT,  
    subordinates  ARRAY<STRING>,  
    deductions    MAP<STRING, FLOAT>,  
    address       STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>);
```

```
John Doe^A100000.0^AMary Smith^BTodd Jones^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A1 Michigan Ave.^BChicago^BIL^B60600
```

```
Mary Smith^A80000.0^ABill King^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A100 Ontario St.^BChicago^BIL^B60601
```

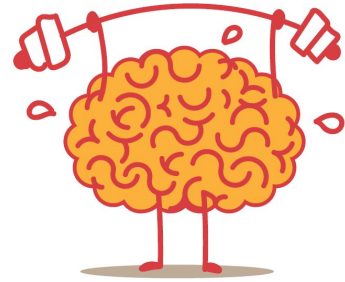


```
CREATE TABLE employees (  
    name          STRING,  
    salary        FLOAT,  
    subordinates  ARRAY<STRING>,  
    deductions    MAP<STRING, FLOAT>,  
    address       STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>);
```

```
John Doe^A100000.0^AMary Smith^BTodd Jones^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A1 Michigan Ave.^BChicago^BIL^B60600
```

```
Mary Smith^A80000.0^ABill King^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A100 Ontario St.^BChicago^BIL^B60601
```





```
CREATE TABLE employees (  
    name          STRING,  
    salary        FLOAT,  
    subordinates  ARRAY<STRING>,  
    deductions    MAP<STRING, FLOAT>,  
    address       STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>);
```

```
John Doe^A100000.0^AMary Smith^BTodd Jones^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A1 Michigan Ave.^BChicago^BIL^B60600
```

```
Mary Smith^A80000.0^ABill King^AFederal Taxes^C.2^BState  
Taxes^C.05^BInsurance^C.1^A100 Ontario St.^BChicago^BIL^B60601
```



# Hive regex SerDe (пример)

```
ADD JAR /usr/local/hive/lib/hive-serde.jar;

CREATE EXTERNAL TABLE ser_de_example (
    ip STRING
)
ROW FORMAT
    serde 'org.apache.hadoop.hive.serde2.RegexSerDe'
    with serdeproperties (
        "input.regex" = "^(\\S*)\\t.*"
    )
STORED AS textfile
LOCATION '/path/to/user_logs';
```



```
CREATE EXTERNAL TABLE tab_dataset (  
    first_column    STRING,  
    second_column  STRING,  
    value           INT  
)  
ROW FORMAT DELIMITED  
    FIELDS TERMINATED BY '\001'  
    COLLECTION ITEMS TERMINATED BY '\002'  
    MAP KEYS TERMINATED BY '\003'  
    LINES TERMINATED BY '\n'  
STORED AS file_format  
LOCATION '/user/<user>/hive_practice_data/';
```

↓

default: **TEXTFILE**



## **Q&A**

Как расшифровывается RCFile?



# Record Columnar File (RCFile)

1. Быстрая загрузка данных в хранилище

[\[Facebook\] 2011 - RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems](#)



# Record Columnar File (RCFile)

1. Быстрая загрузка данных в хранилище
2. Высокая скорость обработки запросов

[\[Facebook\] 2011 - RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems](#)



# Record Columnar File (RCFile)

1. Быстрая загрузка данных в хранилище
2. Высокая скорость обработки запросов
3. Эффективное использование жесткого диска

[\[Facebook\] 2011 - RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems](#)



# Record Columnar File (RCFile)

1. Быстрая загрузка данных в хранилище
2. Высокая скорость обработки запросов
3. Эффективное использование жесткого диска
4. Адаптивность к динамическому изменению паттернов аналитических запросов

[\[Facebook\] 2011 - RCFile: A Fast and Space-efficient Data Placement Structure in MapReduce-based Warehouse Systems](#)



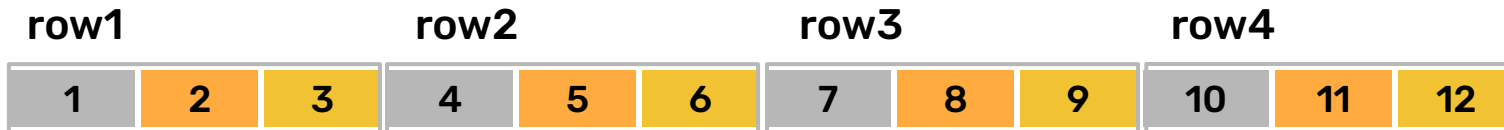


# Record Columnar File (RCFile)

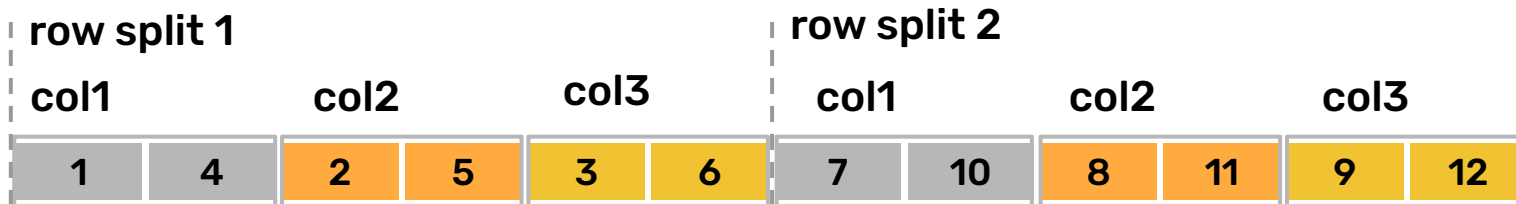
## Логическая структура

	col1	col2	col3
row1	1	2	3
row2	4	5	6
row3	7	8	9
row4	10	11	12

## Row-oriented layout



## Column-oriented layout (RCFile)





# Record Columnar File (RCFile)

Логическая структура

	col1	col2	col3
row1	1	2	3
row2	4	5	6
row3	7	8	9
row4	10	11	12

Row-oriented layout

row1	row2	row3	row4
1 2 3	4 5 6	7 8 9	10 11 12

Column-oriented layout (RCFile)

row split 1	row split 2
col1 col2 col3	col1 col2 col3
1 4 2 5 3 6	7 10 8 11 9 12

date	user	order
2017-05-19 17:53	Alex	100
2017-05-19 17:59	Andrey	200
2017-05-19 18:02	Artyom	50
2017-05-20 10:27	Victoria	350

2017-05-19 17:53,2017-05-19  
17:59, 2017-05-19 18:02,2017-05-20  
10:27; Alex,Andrey,Artyom,Victoria;  
100,200,50,350



# Кодирование и сжатие данных

2017-05-19 17:53,2017-05-19  
17:59,2017-05-19 18:02,2017-05-20  
10:27; Alex, Andrey, Artyom, Victoria;  
100,200,50,350



compression algorithm<sub>1</sub>



compression algorithm<sub>2</sub>



compression algorithm<sub>3</sub>



# Кодирование Datetime

2017-05-19 17:53,2017-05-19  
17:59,2017-05-19 18:02,2017-05-20  
10:27; Alex,Andrey,Artyom,Victoria;  
100,200,50,350



compression algorithm<sub>1</sub>  
2017-05-19 17:53  
2017-05-19 17:59  
2017-05-19 18:02  
2017-05-20 10:27



1495205580  
1495205940  
1495206120  
1495265220



# Delta-кодирование

2017-05-19 17:53, 2017-05-19  
17:59, 2017-05-19 18:02, 2017-05-20  
10:27; Alex, Andrey, Artyom, Victoria;  
100, 200, 50, 350

compression algorithm<sub>1</sub>  
2017-05-19 17:53  
2017-05-19 17:59  
2017-05-19 18:02  
2017-05-20 10:27

1495205580  
+360  
+180  
+59100

$\delta$ -encoding

1495205580  
1495205940  
1495206120  
1495265220



# Кодирование по словарю

2017-05-19 17:53,2017-05-19  
17:59,2017-05-19 18:02,2017-05-20  
10:27; Alex,Andrey,Artyom,Victoria;  
100,200,50,350



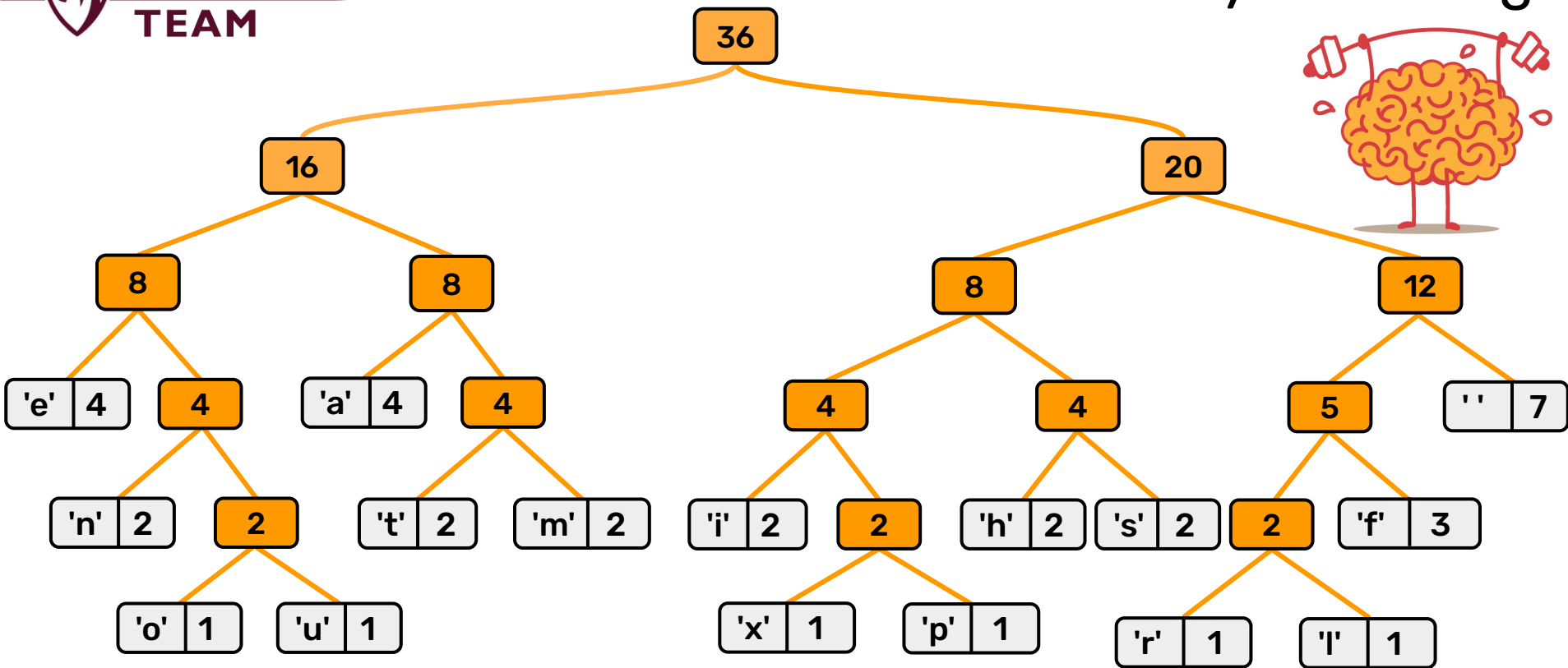
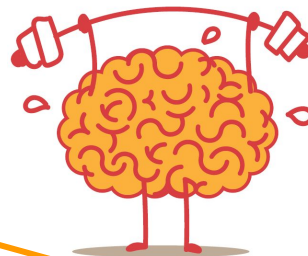
compression algorithm<sub>1</sub>  
(e.g. delta encoding)

dictionary encoding

compression algorithm<sub>3</sub>



# Dictionary Encoding



Код Хаффмана, LZW, ...



# Record Columnar File (RCFile)

2017-05-19 17:53,2017-05-19  
17:59,2017-05-19 18:02,2017-05-20  
10:27; Alex,Andrey,Artyom,Victoria;  
100,200,50,350



compression algorithm<sub>1</sub>  
(e.g. delta encoding)

dictionary encoding

run-length encoding





**BIGDATA**  
**TEAM**

# Резюме по RCFile



1. Быстрая загрузка данных в хранилище





- ± 1.** Быстрая загрузка данных в хранилище



- ±** 1. Быстрая загрузка данных в хранилище
- 2. Высокая скорость обработки запросов



-  1. Быстрая загрузка данных в хранилище
-  2. Высокая скорость обработки запросов



- ±** 1. Быстрая загрузка данных в хранилище
- ✓** 2. Высокая скорость обработки запросов
- 3.** Эффективное использование жесткого диска







- ± 1.** Быстрая загрузка данных в хранилище
- ✓ **2.** Высокая скорость обработки запросов
- ✓ **3.** Эффективное использование жесткого диска



- ± 1.** Быстрая загрузка данных в хранилище
- ✓ **2.** Высокая скорость обработки запросов
- ✓ **3.** Эффективное использование жесткого диска
- 4.** Адаптивность к динамическому изменению паттернов аналитических запросов





-  **1.** Быстрая загрузка данных в хранилище
-  **2.** Высокая скорость обработки запросов
-  **3.** Эффективное использование жесткого диска
-  **4.** Адаптивность к динамическому изменению паттернов аналитических запросов