DATAMASS

Big Data Academy 2019

Kainos

Lab 3 - Hive/Impala

datamass.io

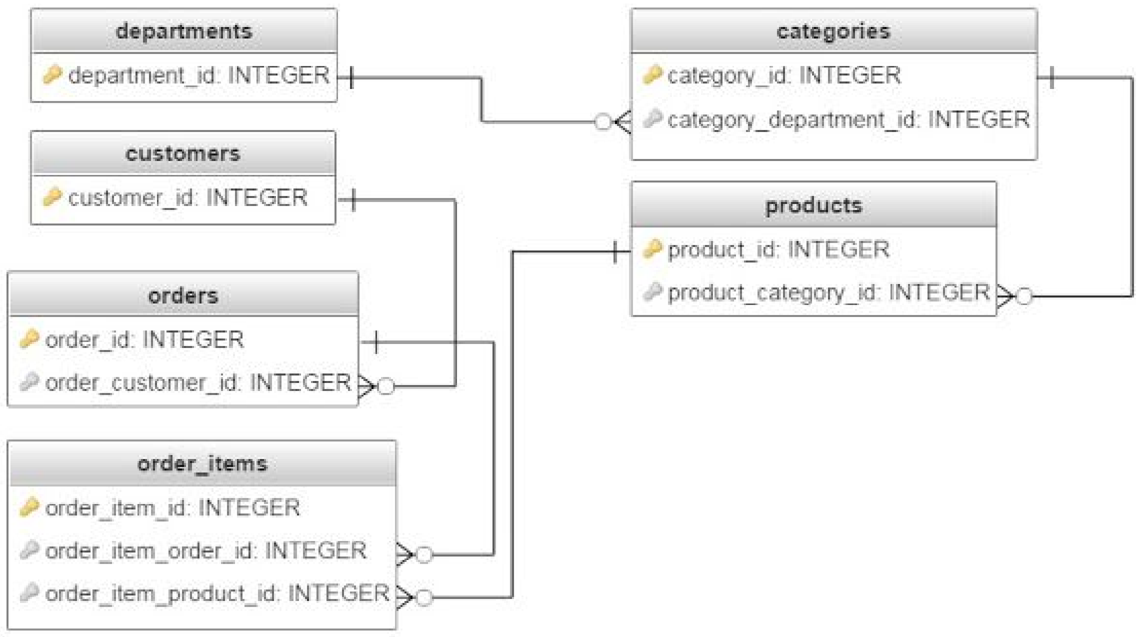
Introduction

The aim of the laboratory classes is to familiarize participants with the Hive / Impala environment allowing for analyzing data placed in the distributed HDFS file system. The analysis will be carried out using the HQL language (Hive Query Language).

***NOTE**: All operations will be carried out on the database located on a virtual machine created for the purposes of laboratory classes.*

db-name: retail_db
db-user: retail_dba
db-passwd: cloudera

The relation diagram in **retail_db** database looks as follows:



HIVE/IMPALA

CLI

- Access to the Hive / Impala environment is presented below. It is worth noting that both these environments have rich interpreters that allow them to be fed with many input parameters.

```
[cloudera@quickstart ~]$ hive
```

```
[cloudera@quickstart ~]$ impala-shell
```

The basic input parameters are shown in the table below:

Run Query	`hive -e 'select a.col from tab1 a'`
Run Query Silent Mode	`hive -S -e 'select a.col from tab1 a'`
Set Hive Config Variables	`hive -e 'select a.col from tab1 a' -hiveconf hive.root.logger=DEBUG,console`
Use Initialization Script	`hive -i initialize.sql`
Run Non-Interactive Script	`hive -f script.sql`

- Hive/Imapla allows to create databases

```
hive> CREATE DATABASE IF NOT EXISTS customers;
OK
Time taken: 2.531 seconds
```

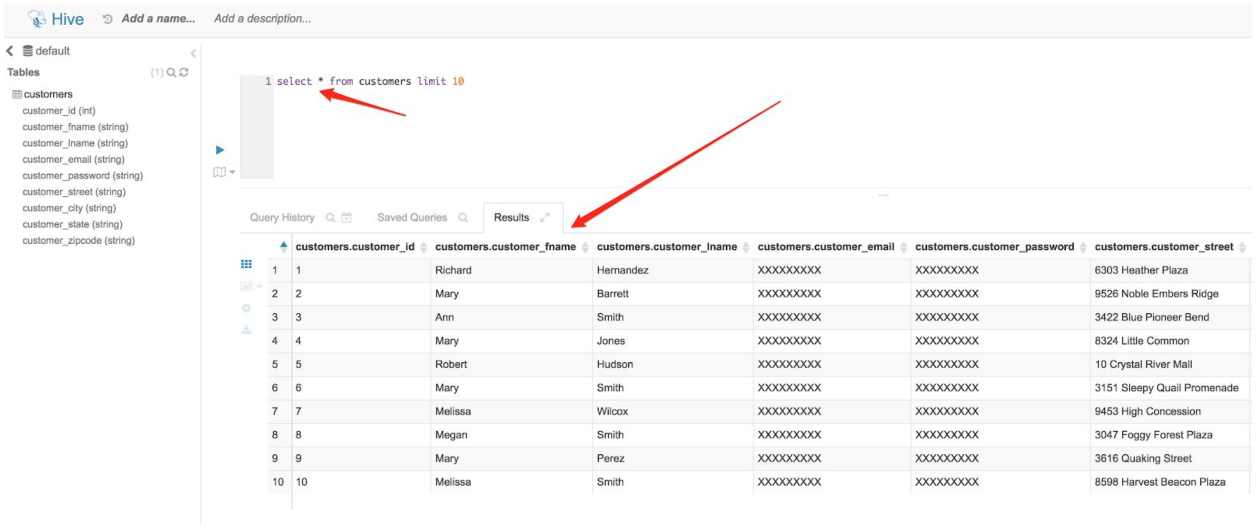
- Syntax associated with creating a table

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name
  (col_name data_type [COMMENT 'col_comment'], ...)
  [PARTITIONED BY (col_name data_type [COMMENT 'col_comment'], ...)]
  [COMMENT 'table_comment']
  [WITH SERDEPROPERTIES ('key1'='value1', 'key2'='value2', ...)]
  [
   [ROW FORMAT row_format] [STORED AS file_format]
  ]
  [LOCATION 'hdfs_path']
  [TBLPROPERTIES ('key1'='value1', 'key2'='value2', ...)]
  [CACHED IN 'pool_name' [WITH REPLICATION = integer] | UNCACHED]
```

- Creating a table in Hive / Impala using the Sqoop environment:

```
[cloudera@quickstart ~]$ sqoop import
--connect "jdbc:mysql://quickstart.cloudera:3306/retail_db"
--username retail_dba
--password cloudera
--table customers
--target-dir /user/cloudera/mysql/customers_hive
--fields-terminated-by "\t"
--hive-import
```

- The imported data can be seen in HUE using Hive / Impala or Metastore manager.



```
select * from customers limit 10
```

	customers.customer_id	customers.customer_fname	customers.customer_lname	customers.customer_email	customers.customer_password	customers.customer_street
1	1	Richard	Hernandez	XXXXXXXXX	XXXXXXXXX	6303 Heather Plaza
2	2	Mary	Barrett	XXXXXXXXX	XXXXXXXXX	9526 Noble Embers Ridge
3	3	Ann	Smith	XXXXXXXXX	XXXXXXXXX	3422 Blue Pioneer Bend
4	4	Mary	Jones	XXXXXXXXX	XXXXXXXXX	8324 Little Common
5	5	Robert	Hudson	XXXXXXXXX	XXXXXXXXX	10 Crystal River Mall
6	6	Mary	Smith	XXXXXXXXX	XXXXXXXXX	3151 Sleepy Quail Promenade
7	7	Melissa	Wilcox	XXXXXXXXX	XXXXXXXXX	9453 High Concession
8	8	Megan	Smith	XXXXXXXXX	XXXXXXXXX	3047 Foggy Forest Plaza
9	9	Mary	Perez	XXXXXXXXX	XXXXXXXXX	3616 Quaking Street
10	10	Melissa	Smith	XXXXXXXXX	XXXXXXXXX	8598 Harvest Beacon Plaza

- Create a Hive / Impala table based on a file located in a specific location on the HDFS.

```
create external table customers_limit (
customer_id INT,
customer_fname STRING,
customer_email STRING,
customer_state STRING)
row format delimited
fields terminated by '\t'
location '/user/cloudera/mysql/customers_limit'
```

NOTE: In the case of Impala, access to the table is possible after pressing the refresh button (Hue, Impala editor) or entering the INVALIDATE METADATA command in the Impala editor.

Exercise 1

In what type of table (external, managed) sqoop automatically stores the data visible after the completion of the loading process?

Exercise 2

Load the **departments** table using the SQOOP tool. Place the table in a database named **metadb**. Data related to the table should be placed on HDFS in the following location **/user/hive/warehouse/departments**. As a separator use '\001'.

Exercise 3

Load data from the **categories** table into the HDFS file system to the following location **/user/kainos/mysql/categories**. Use the separator '\t'. Place data in a single output file. Use the compression and codec *org.apache.hadoop.io.compress.SnappyCodec*.

Exercise 4

Create a **category** table in Hive based on the data saved in the following location on HDFS **/user/kainos/mysql/categories.**

Exercise 5

Execute JOIN on the **categories** and **departments** tables. Display the following columns *category_name*, *department_name* and *guid_id*[1]. Save the result to the **cat_dep_join** table. Start the Impala environment and count the number of items found in each department (department_name). Sort descending by count.

Exercise 6

Perform the export of the result set created in the previous task to the relational database (MySQL). Use the SQOOP tool for export. Name the mysql table **cat_dep_join**.

Exercise 7

Assign the appropriate departments to the products, then sort by *department_name* ascending, *product_name* descending, and finally group by categories.

[1] use function **reflect("java.util.UUID", "randomUUID")**