**Zhastay Yeltay**

**Hive Practice**

CONTENTS

[1. Basic Hive Interaction Using Hue (15 points) 2](#_Toc126593965)

[2. Basic Hive Interaction Using Beeline (10 points) 10](#_Toc126593966)

[3. Managed and External Tables Using Beeline (20 points) 12](#_Toc126593967)

[4. Partitioned Tables (35 points) 17](#_Toc126593968)

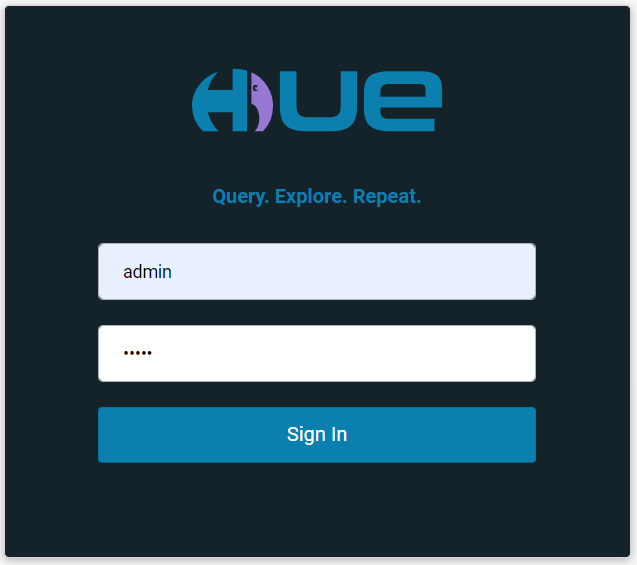
[5. Hive ACID Tables (20 points) 22](#_Toc126593969)

Before I want to say that I’ve changed the image of hive-metastore-postgresql from “peerlesscnn/hive-metastore-postgresql:2.3.0” to “bde2020/hive-metastore-postgresql:2.3.0” in the docker-compose.yml. And after that I created hue user for postgres manually from sql file. Because It doesn’t work without it. As a consequence it works perfectly.

***Note: The requirements for the practical task are highlighted in bold, the other simple text is my answers.***

**Basic Hive Interaction Using Hue (15 points)**

* **Connect to Hue using “admin/admin”**



* **Browse over to the Hive editor and perform the following:**
  + **Show available databases under Hive**

A screenshot of a computer

Description automatically generated

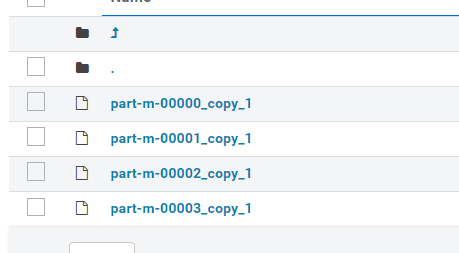
After installing all these things I noticed that there is no tables in default scheme.

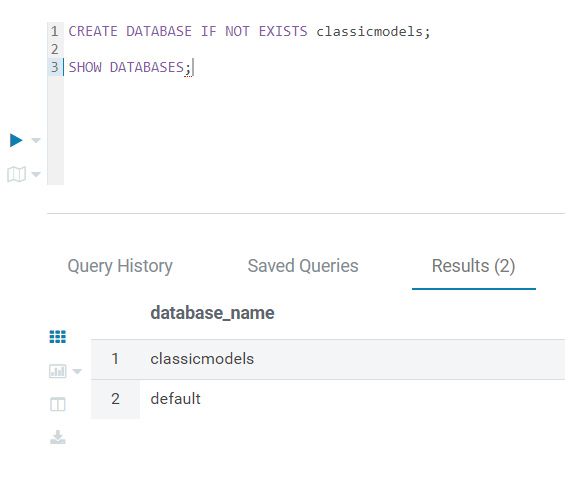
* + **Create the “classicmodels” database and upload the database from the attachment of the Homework module; verify that the separator is considered successfully**

I found the classicmodels.db in the <https://www.mysqltutorial.org/mysql-sample-database.aspx>. Because I needed the column names. And this is really hard to do without it.

Also, I prepared python code for parsing and replacing separator symbol to tab, that means I created CSV files(Actually it is TSV(Tabular separated values)).

Imported files:

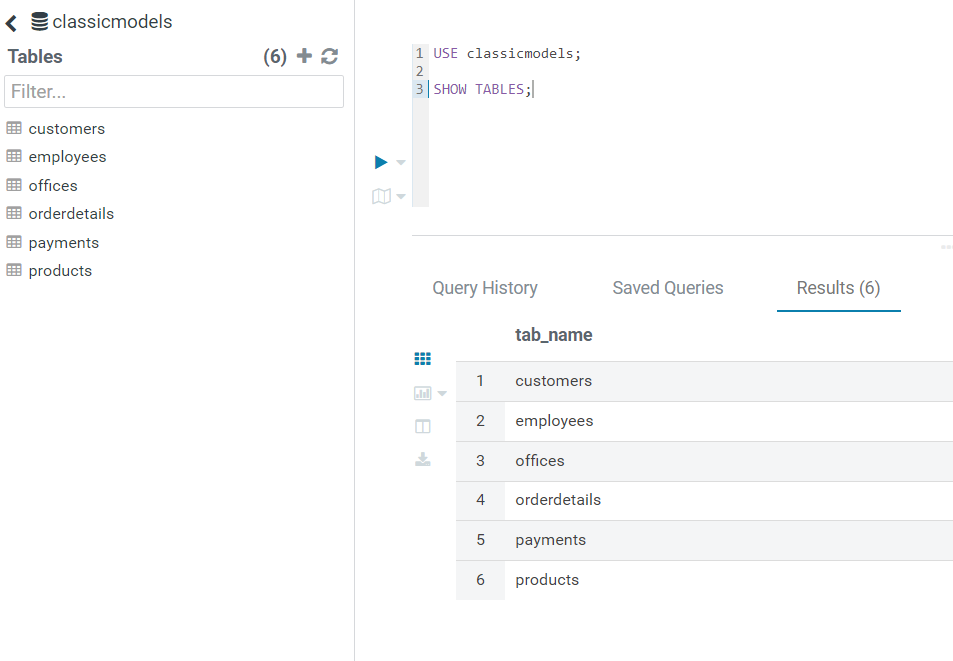




* + **Switch to it**

USE classicmodels;

* + **Show all the available tables in the database**

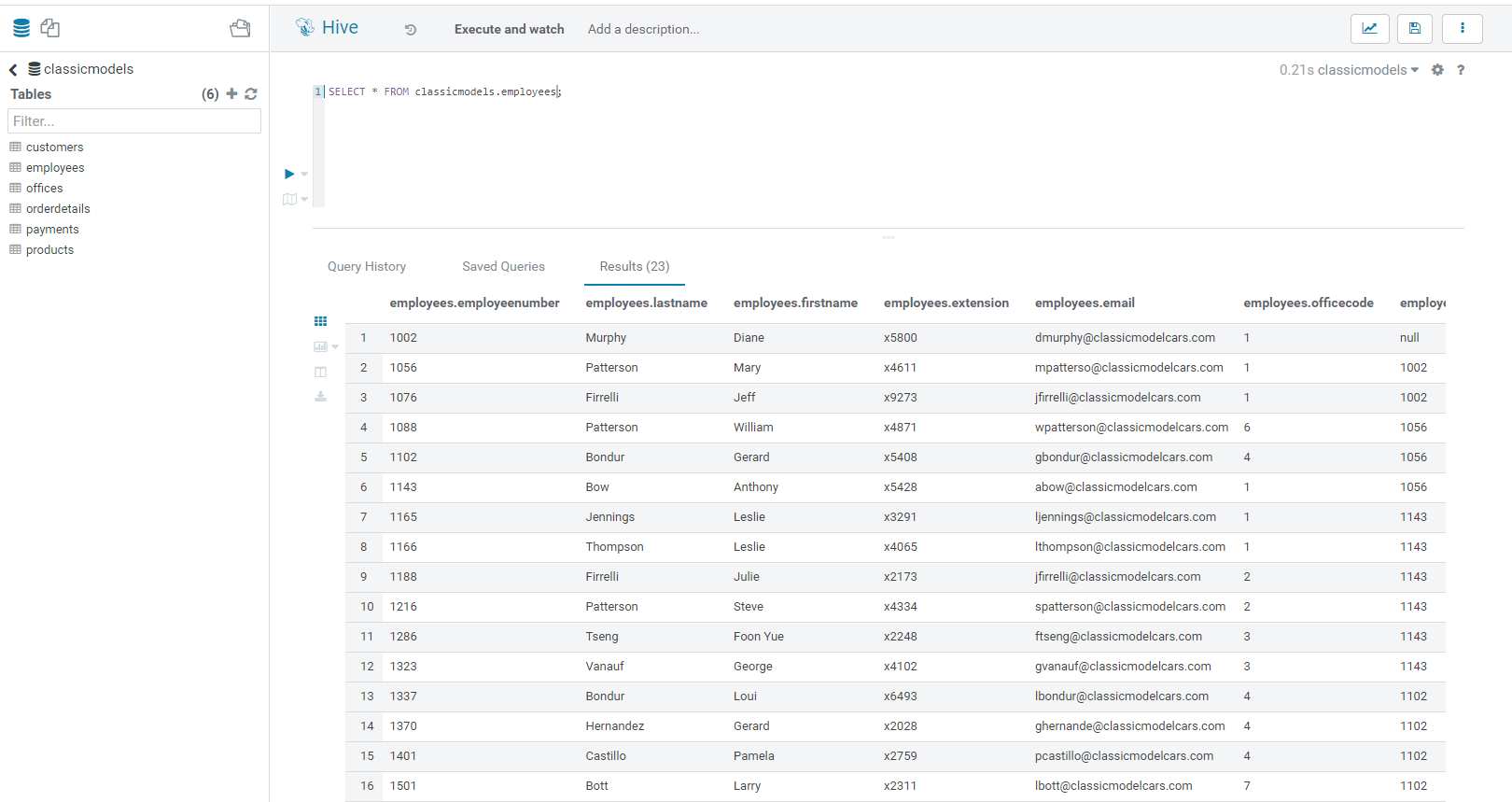


* + **Expand the “Customers” table and view its columns and data types**

**A screenshot of a computer

Description automatically generated**

* **Perform the following queries:**
  + **Query all rows from the “Employees” table**

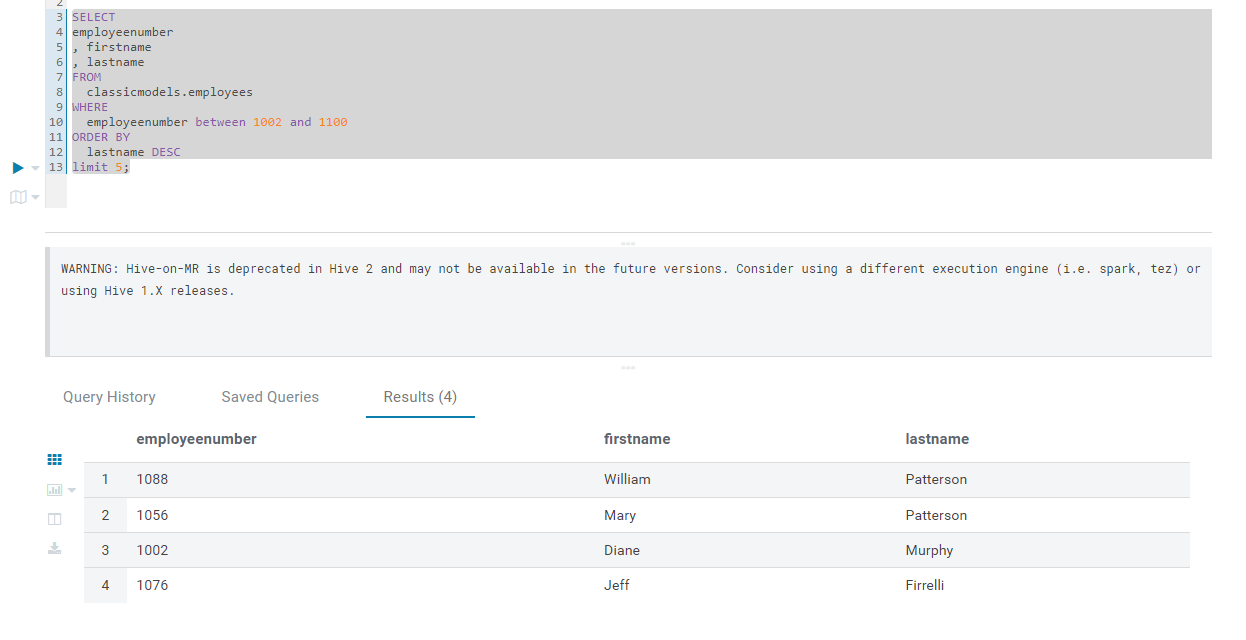
****

* + **Alter the previous query to fetch only the first 10 rows**

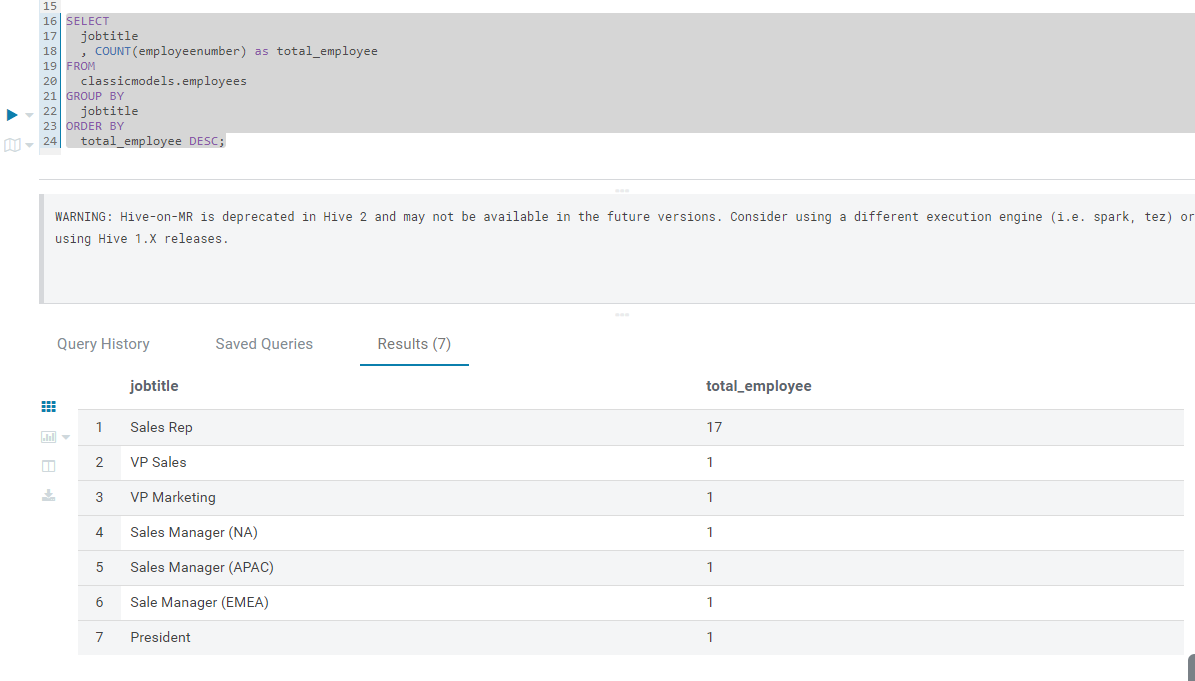
**A screenshot of a computer

Description automatically generated**

* + **Write a query to fetch the following:**
    - **The employee ID: first name and last name**
    - **The employee number should be between 1002 and 1100**
    - **Order by last name in descending order**
    - **Fetch only first five rows**

****

* + **Write a query to fetch the number of employees per job title, ordered by number of employees in descending order**

****

* + **Export the query output to a text file**

**A screenshot of a computer

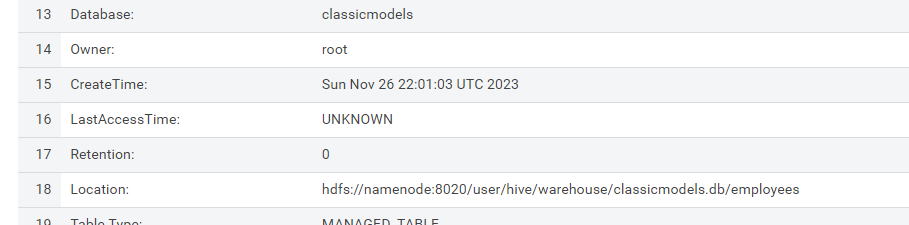
Description automatically generated**

* **Check which HDFS folder the “employees” table points at**

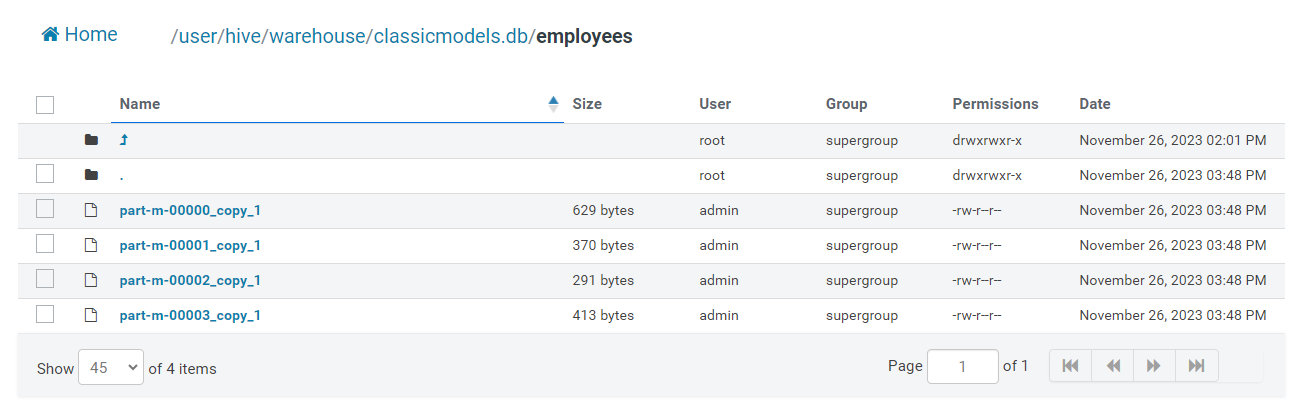
**Hint: Use the practice guide to see how you can view such details for Hive tables.**

I used “DESCRIBE FORMATTED employees;”

I got “'hdfs://namenode:8020/user/hive/warehouse/classicmodels.db/employees'”

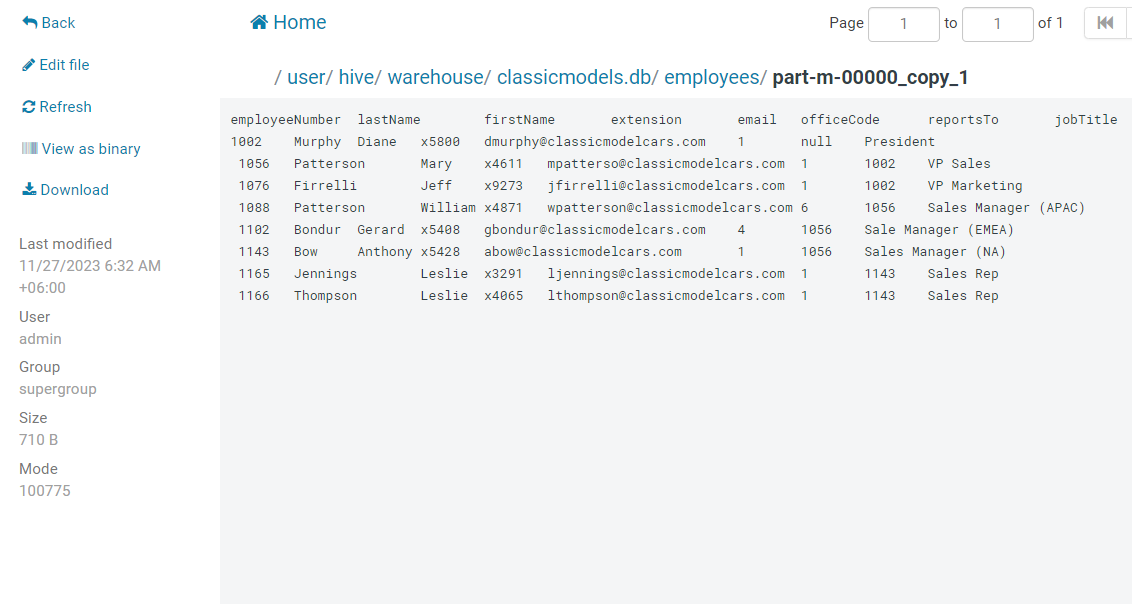


* **HDFS Browse**
  + **Use the HDFS browser in Hue to browse over to the HDFS folder and examine its contents**

****

* + **Click on one of the files to examine the file contents. Check the following:**
    - **If the file is human-readable**

Yes, as you can see below the picture shows textual data.



* + - **Which Hive table property is responsible for this**

I think answer is TextFormat

**Note: Use the CLI to browse HDFS and view file contents in case Hue HDFS browser is not working in your environment (If ports 50070 and 50075 were not mapped, this will be causing an error message)**

Also, I can use localhost:50070 to view folders.

**A screenshot of a computer

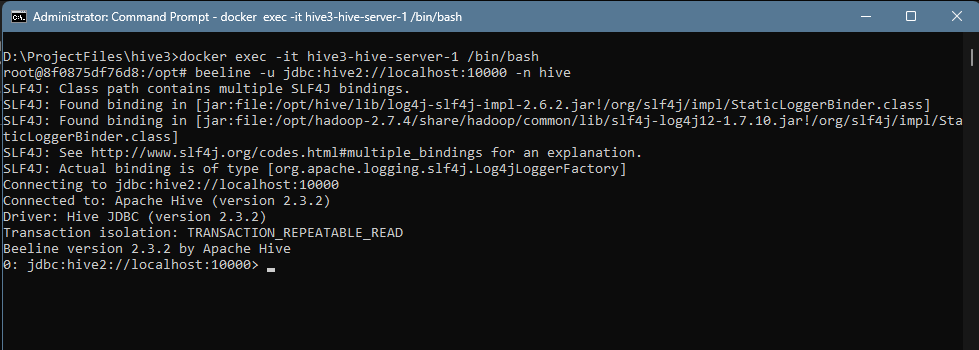
Description automatically generated**

**Basic Hive Interaction Using Beeline (10 points)**

* **Open a BASH session to the practice environment and connect to Hive using Beeline**

docker exec -it hive3-hive-server-1 /bin/bash

beeline -u jdbc:hive2://localhost:10000 -n hive

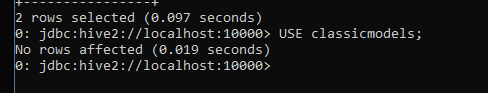


* **Show available databases (Verify you see the “ClassisModels” database)**

A screen shot of a computer

Description automatically generated

* **Switch to use the “ClassisModels” database**

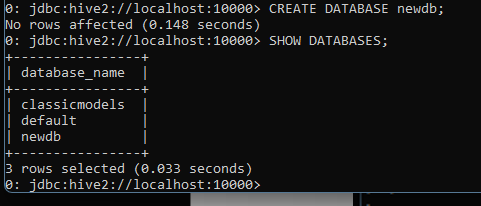
****

* **Show all tables in this database**

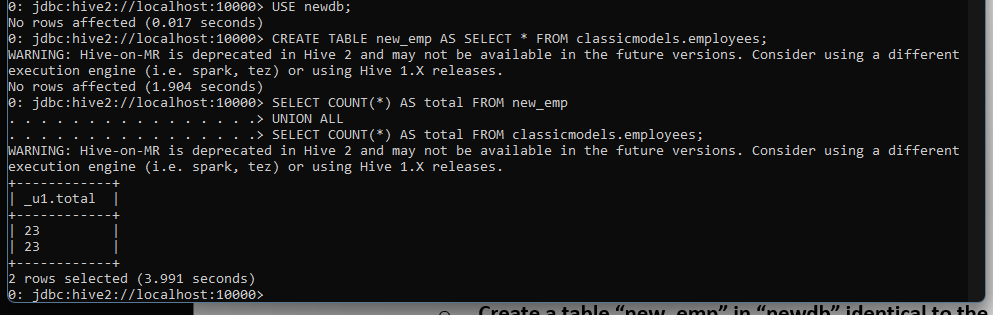
**A screen shot of a computer

Description automatically generated**

* **Create a new database called “newdb” and verify the database was created**

****

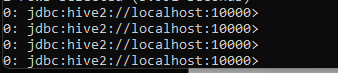
* **Create a table “new\_emp” in “newdb” identical to the “Employees” table in “ClassicModels” database (Both schema and data), and run a COUNT(\*) to verify the table is populated**

****

**Managed and External Tables Using Beeline (20 points)**

* **Open a BASH session to the practice environment and connect to Hive using Beeline**

It is opened:



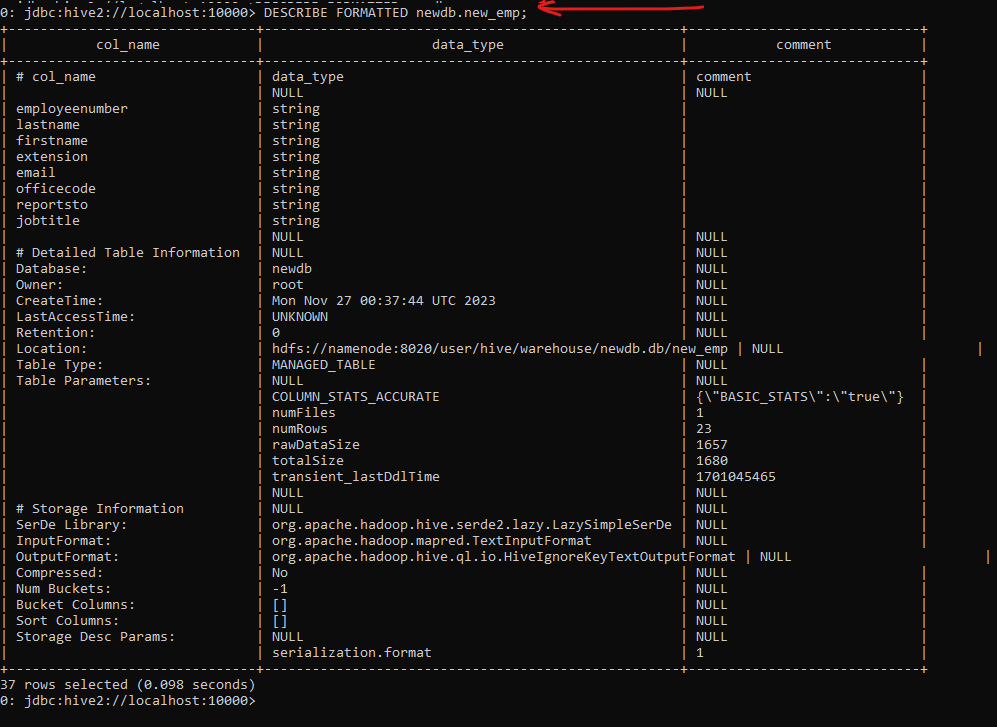
* **Run a COUNT(\*) to verify that “newdb.new\_emp” is in place and populated**

**A screenshot of a computer

Description automatically generated**

* **Check using the table properties (without browsing HDFS):**

DESCRIBE FORMATTED newdb.new\_emp;



* + **Where in HDFS the data for this table is located**

hdfs://namenode:8020/user/hive/warehouse/newdb.db/new\_emp

* + **The file type for this table**

File type is TextFormat.

**A black background with white text

Description automatically generated**

* + **If the table Managed or External?**

It is managed:

****

* + **How many physical data files belong to this table**

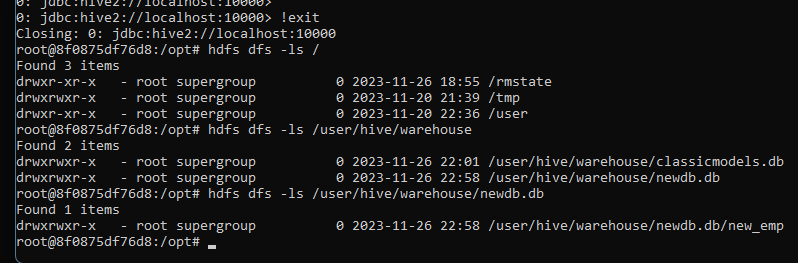
Only 1.

****

**Hints:**

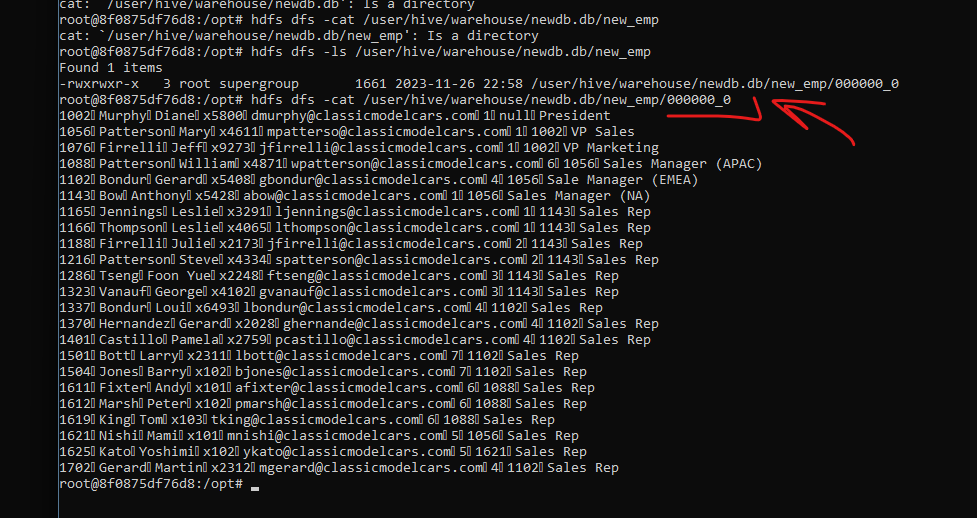
* + - **Use the practice guide to see how you can view such details for Hive tables**
    - **Use a more verbose command to view all properties**
* **Exit Beeline and use the HDFS CLI to examine the HDFS directory for this table. Do the following:**
  + **Check how many files there are**

There are 3 folders.

****

* + **View the file contents and see if they are readable; explain why**

No, they are folders in hdfs directory. But I tried to examine a 000000\_0 fil in the new\_emp table. And yes, it is readable.



* **Go back to Beeline and drop the “new\_emp” table**

**A computer screen with text and symbols

Description automatically generated**

* **Check in the HDFS CLI again if the HDFS directory and files still exist; explain why**

There is no directory and files.

**A screenshot of a computer program

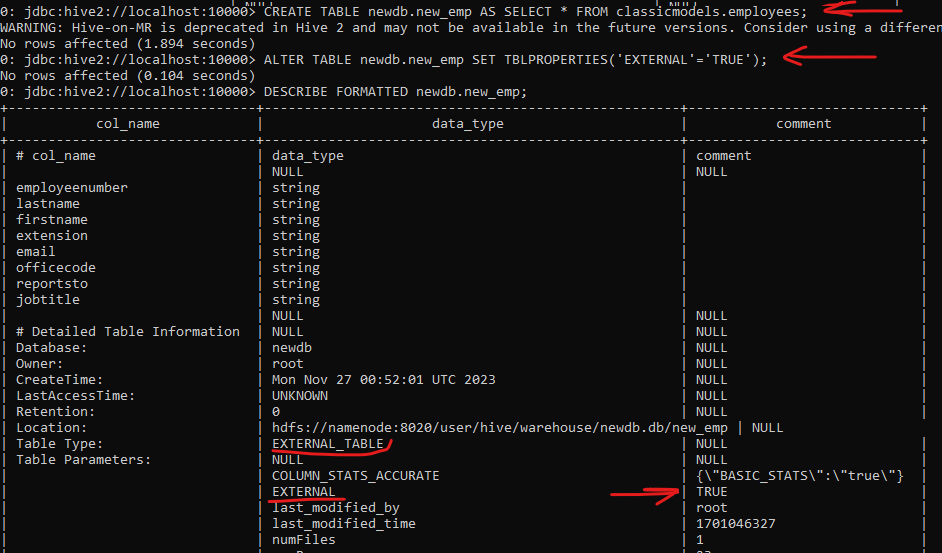
Description automatically generated**

* **Go back to Beeline and create the “new\_emp” table again; this time, create it as an EXTERNAL table. Check what error you received and explain why.**

****

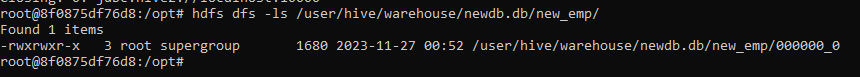
Because we cannot use it when we select. We can try it when data is settled somewhere, for example, when we read from file.

* **Create the table as MANAGED (this is the default), and change it manually to EXTERNAL after its creation.**
* **Check again:**
  + **Where in HDFS the data for this table is located**
  + **The file type for this table**
  + **If the table Managed or External**

****

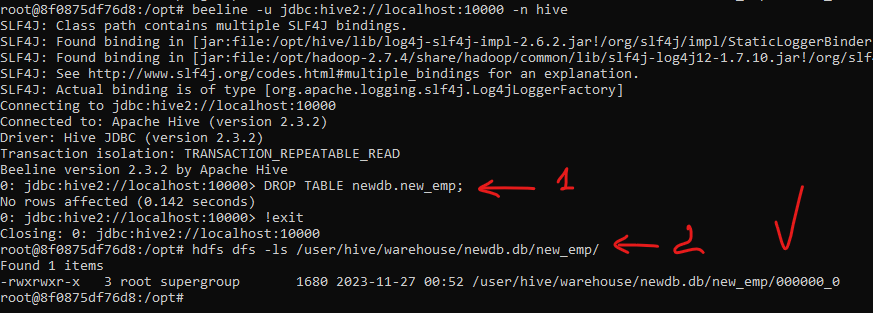
* **Exit Beeline and use the HDFS CLI to examine the HDFS directory for this table; check how many files there are**

Only 1 file.

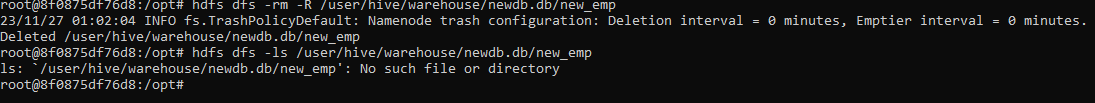
****

* **Go back to Beeline, and drop the “new\_emp” table**
* **Check again in the HDFS CLI if the HDFS directory and file still exist; explain why**

We changed type of table, and I lost control. *We can not delete external data, because system creates link to this data and after deleting table*, we remove just link to this data. That is it about External tables. Also we can connect different tables or file for one time using and that is best option for this case.

****

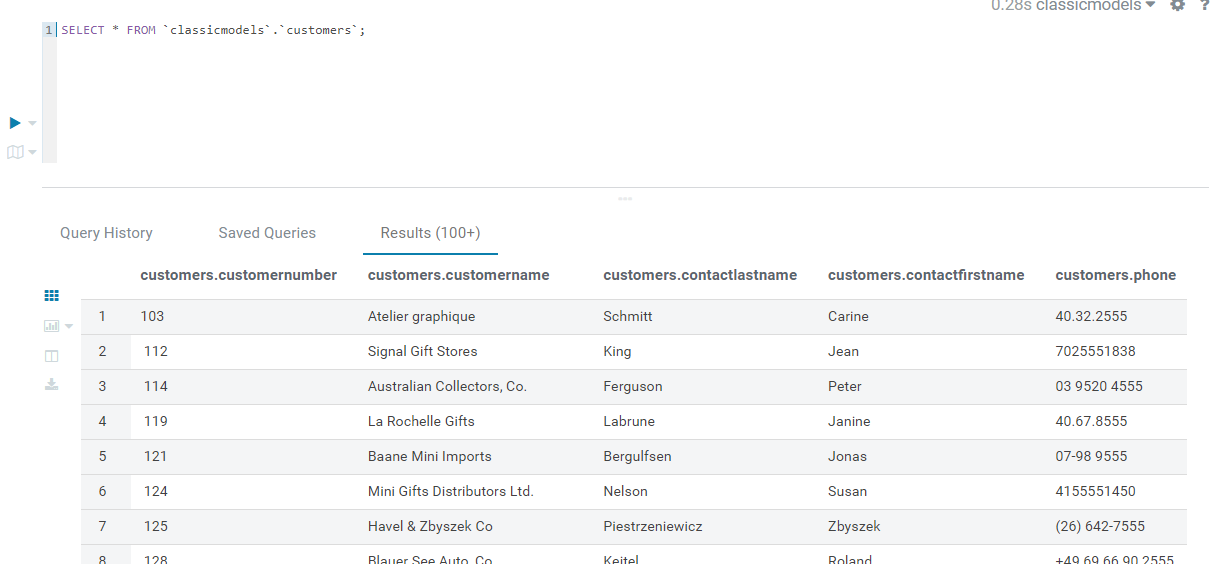
* **Remove the HDFS folder manually using the HDFS command CLI (Careful…). and verify that the directory does not exist**

****

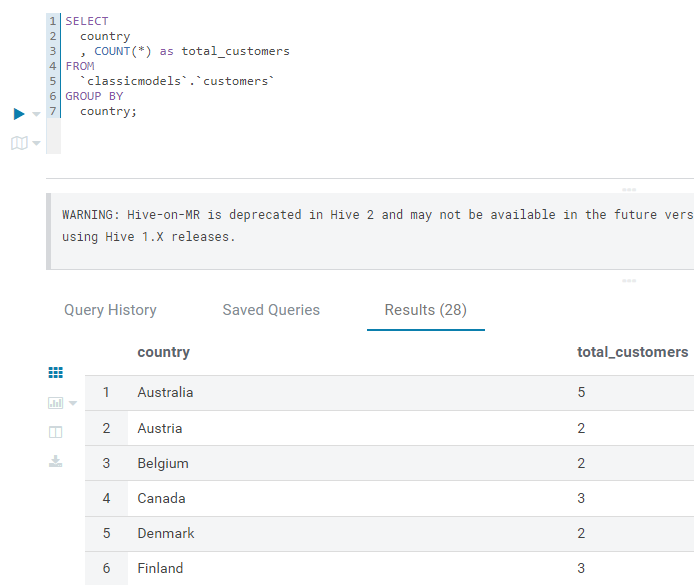
**Partitioned Tables (35 points)**

**You can complete this task through Hue or Beeline. Use the command prompt to perform HDFS tasks if the HDFS browser is not available in your Hue environment.**

* **Query the customers using a simple “SELECT \*” to view the sample data**

****

* **Run another query, this time to see the number of customers in each country  
  (GROUP BY..)**

****

* **Get the DDL of the customers' table**

**Hint: Use the practice guide to see how you can check out such details for Hive tables.**

SHOW CREATE TABLE classicmodels.customers;

* **Copy only the main CREATE TABLE section to a text editor, without all the properties:**

*CREATE TABLE classicmodels.customers(*

*customernumber string,*

*customername string,*

*contactlastname string,*

*contactfirstname string,*

*phone string,*

*addressline1 string,*

*addressline2 string,*

*city string,*

*state string,*

*postalcode string,*

*country string,*

*salesrepemployeenumber string,*

*creditlimit string)*

* **Modify the CREATE TABLE command, to create a new table with the following characteristics:**
  + **DB Name: “ClassicModels”**
  + **Table name: “cust\_country”**
  + **Partitioned By: “country” column**
  + **File type: AVRO**

****

* **Verify the table was created properly and view its properties**

|  |  |  |
| --- | --- | --- |
| **col\_name** | **data\_type** | **comment** |
| # col\_name | data\_type | comment |
|  | NULL | NULL |
| customernumber | string |  |
| customername | string |  |
| contactlastname | string |  |
| contactfirstname | string |  |
| phone | string |  |
| addressline1 | string |  |
| addressline2 | string |  |
| city | string |  |
| state | string |  |
| postalcode | string |  |
| salesrepemployeenumber | string |  |
| creditlimit | string |  |
|  | NULL | NULL |
| # Partition Information | NULL | NULL |
| # col\_name | data\_type | comment |
|  | NULL | NULL |
| country | string |  |
|  | NULL | NULL |
| # Detailed Table Information | NULL | NULL |
| Database: | classicmodels | NULL |
| Owner: | root | NULL |
| SerDe Library: | org.apache.hadoop.hive.serde2.avro.AvroSerDe | NULL |
| InputFormat: | org.apache.hadoop.hive.ql.io.avro.AvroContainerInputFormat | NULL |
| OutputFormat: | org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat | NULL |

* **Insert data into the “Cust\_Country” from the “Customers” table (Limit to 50 rows) so that partitions will be generated and populated dynamically.**

**Note: If you get an error related to “MoveTask” you can ignore it.   
This error is related to the practice environment. Using the following practices, you can verify that rows and partitions were actually created in the target table. This is what is required for this practice.**

set hive.exec.dynamic.partition.mode=nonstrict;

INSERT INTO TABLE classicmodels.cust\_country

PARTITION(country)

SELECT

customernumber

, customername

, contactlastname

, contactfirstname

, phone

, addressline1

, addressline2

, city

, state

, postalcode

, salesrepemployeenumber

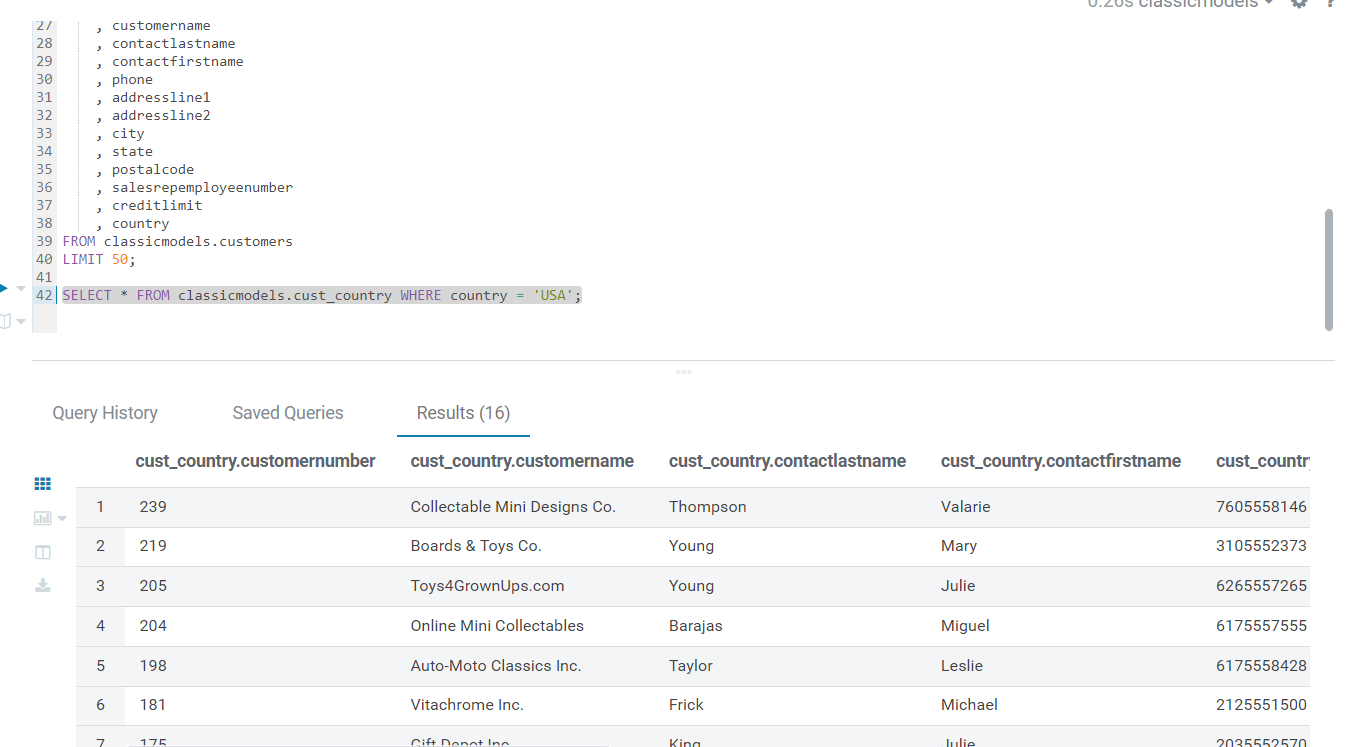
, creditlimit

, country

FROM classicmodels.customers

LIMIT 50;

* **Run a query from “Cust\_Country” to view all customers from “USA”**

****

* **Examine the execution plan of the query to verify partition elimination has occurred. Answer the following questions:**
  + **Which EXPLAIN command was required to view partition related details?**

EXPLAIN DEPENDENCY

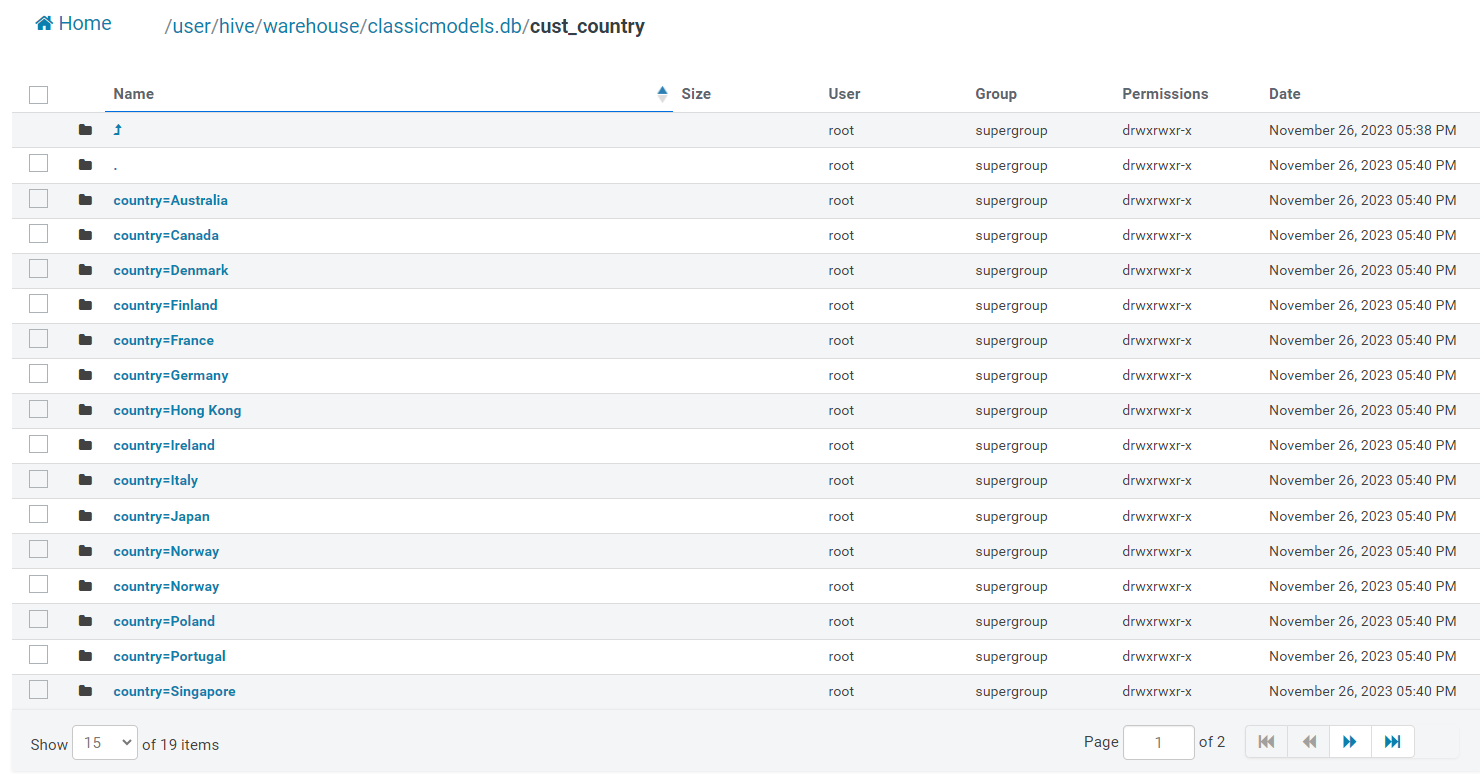
SELECT \* FROM classicmodels.cust\_country WHERE country = 'USA';

|  |
| --- |
| **Explain** |
| {"input\_tables":[{"tablename":"classicmodels@cust\_country","tabletype":"MANAGED\_TABLE"}],"input\_partitions":[{"partitionName":"classicmodels@cust\_country@country=USA"}]} |

* + **Why is partitioning so important for query performance?**

*Since it is easier to find data when it is collected in one place, in the case of the USA we can see that we just chose 19 lines, like 19 pages of a book. This is really good, but if we have a lot of small partitions, then there may be a cases with a small Hadoop file problem. Hadoop is struggling with a small number of files, and for this reason we need to prevent such scenarios.*

* **Go over to HDFS and see the directory structure created for the partitioned table. Answer the following questions:**

****

* + **What are the contents of the main directory for this table?**

**Country names.**

* + **What are the names of the subdirectories?**

**Just avro files. Default file name is 000000\_0. Because there is no bucketing and so on…**

**Hive ACID Tables (20 points)**

**This task can be completed through Hue or Beeline.**

**I chose Beeline, it is simple and easy to use.**

**Note:**

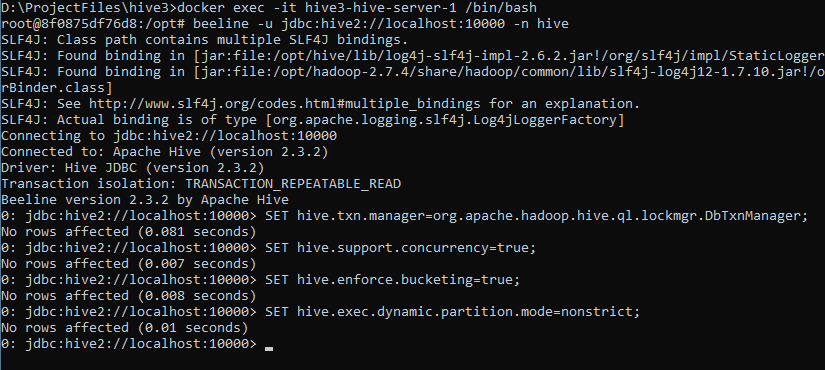
* **The practice environment requires setting the following to support transactions:**

***SET hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;***

***SET hive.support.concurrency=true;***

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

***SET hive.exec.dynamic.partition.mode=nonstrict;***



* **Please run these SET commands in the Hue/Beeline window prior to performing this exercise.**
* **Create a new transactional table called “my\_emp” with the following properties:**
  + **Columns:**
    - **ID – INT**
    - **Name – STRING**
    - **Salary – INT**
  + **File type: ORC**
  + **Transactional…**

**CREATE TABLE classicmodels.my\_emp(**

**id INT,**

**name STRING,**

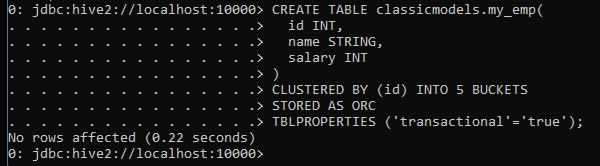
**salary INT**

**)**

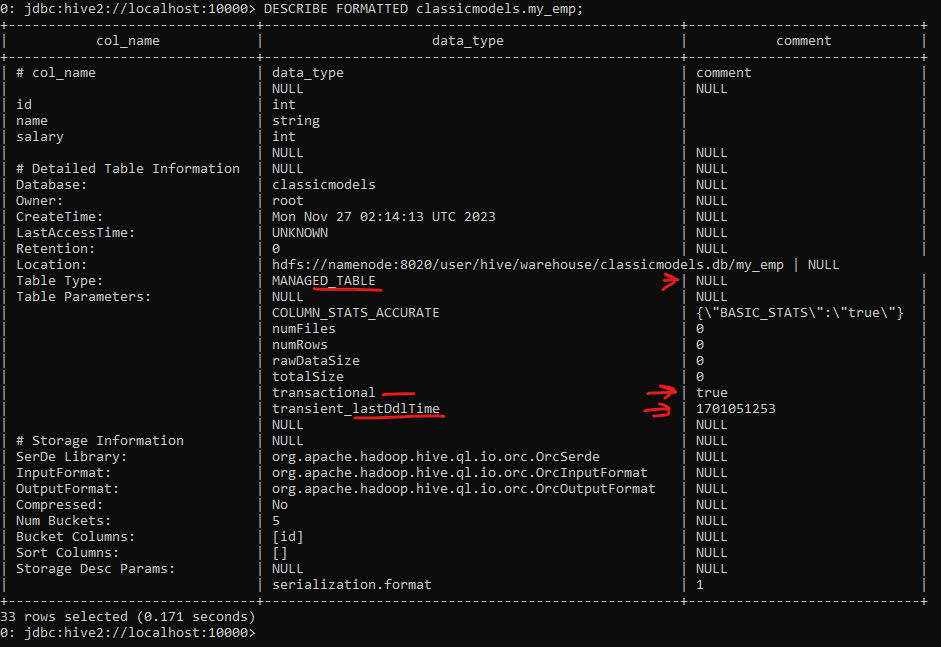
**CLUSTERED BY (id) INTO 5 BUCKETS**

**STORED AS ORC**

**TBLPROPERTIES ('transactional'='true');**

****

* **Check if this table supports DML operations and which DESCRIBE operation is required**

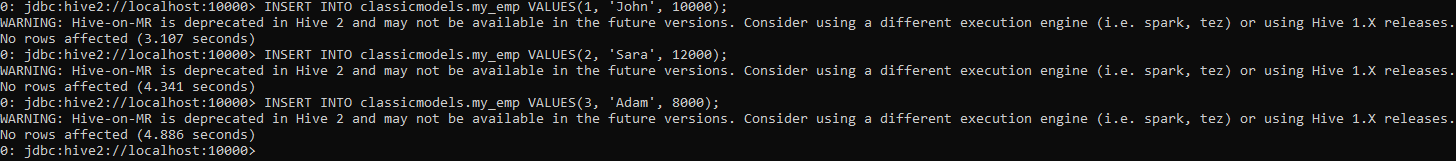
****

* **Insert 3 rows to this table in a single INSERT command:**
  + **1, John, 10000**
  + **2, Sara, 12000**
  + **3, Adam, 8000**

INSERT INTO classicmodels.my\_emp VALUES(1, 'John', 10000);

INSERT INTO classicmodels.my\_emp VALUES(2, 'Sara', 12000);

INSERT INTO classicmodels.my\_emp VALUES(3, 'Adam', 8000);



* **Query the table to verify all rows were inserted**

**A screenshot of a computer

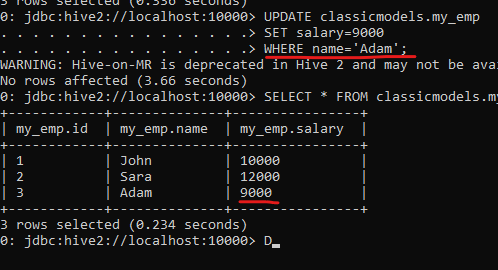
Description automatically generated**

* **Update Adam’s salary in “my\_emp” to 9000**

UPDATE classicmodels.my\_emp

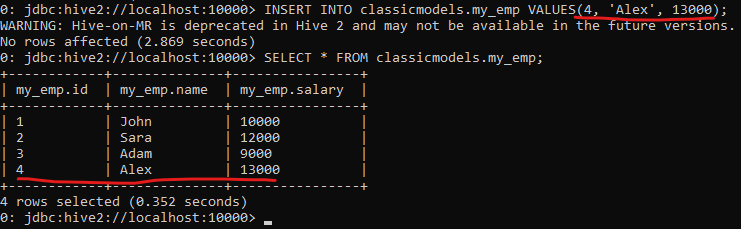
SET salary=9000

WHERE name='Adam';

****

* **Insert a new row to “my\_emp”—4, Alex, 13000**

INSERT INTO classicmodels.my\_emp VALUES(4, 'Alex', 13000);



* **Delete John from the table.**
* **Query “my\_emp” to verify you see all changes performed.**

**A computer screen with white text

Description automatically generated**

**That is all!**

**Thank you for your attention!**

**👍⛈️🏁🫡**