Partition Data in Hive

|  |  |
| --- | --- |
| **Exercise Dir** | ~/workspace/data-partition |
| **HDFS Data** | lanier/accounts\_avro |

**In this exercise you will create and load an Hive table with account**

**data, partitioned by area code.**

Previously you imported data from the accounts table using Sqoop, into a table called accounts\_avro. In this exercise, you will create a new table with some of the account data, partitioned by area code (the first three digits of the phone number).

1. Create a new, empty table with Hive:

CREATE EXTERNAL TABLE accounts\_by\_areacode ( acct\_num INT,

first\_name STRING, last\_name STRING, phone\_number STRING)

PARTITIONED BY (areacode STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

1. In order to populate the new table, you will need to extract the area code from the phone number. Try executing the following query to demonstrate:

SELECT acct\_num, first\_name, last\_name,

phone\_number, SUBSTR(phone\_number,1,3) AS areacode

FROM accounts\_avro

1. Use the SELECT statement above in an INSERT INTO TABLE command to copy the specified columns to the new table, dynamically partitioning by area code.
2. Execute a simple query to confirm that the table was populated correctly, such as:

SELECT \* FROM accounts\_by\_areacode LIMIT 10

1. Using Hue or the hdfs command line interface, confirm that the directory structure of the accounts\_by\_areacode table includes partition directories. Review the data in the directories to verify that the partitioning is correct.

**END**

**BELOW IS THE PREREQUISITE LAB**

THIS IS THE PREREQUISITE LAB

**In this exercise, you will use import data in Avro format and create an**

**Impala/Hive table to access it.**

**1.** Change directories to the exercise directory.

* + cd ~/workspace/data-format/

1. Import the accounts table to an Avro data format.

**$** sqoop import \

--connect jdbc:mysql://localhost/loudacre \ --username training --password training \ --table accounts \

--target-dir /loudacre/accounts\_avro \ --null-non-string '\\N' \

**--as-avrodatafile**

1. View the files imported by Sqoop into HDFS using the hdfs command. What do you see when you try to view the content of the data files?

What they will see will depend on what tool they are using. Hue will deserialize the Avro file and display it as properties and values. The HDFS command line

will display mostly garbage because the file contains compressed, binary format

data.

1. Optional: Download one of the generated Avro files to a local (non-HDFS) directory, and use the avro-tools tojson command to view the file contents.

**$** hdfs dfs -get \

/loudacre/accounts\_avro/part-m-00000.avro

**$** avro-tools tojson part-m-00000.avro | more

1. Sqoop generates a schema (in the file accounts.avsc) in the current directory. Review this file and copy it to HDFS below /loudacre/.
   * hdfs dfs -put accounts.avsc /loudacre/
2. In Impala or Hive, create a table using this schema:

CREATE EXTERNAL TABLE accounts\_avro

STORED AS AVRO

LOCATION '/loudacre/accounts\_avro'

TBLPROPERTIES ('avro.schema.url'=

‘hdfs:/loudacre/accounts.avsc')

###REPLACE ABOVE WITH####

create external table accounts\_test stored as avro tblproperties ('avro.schema.url' = ‘hdfs:/user/training/lanier/accounts.avsc')

1. Confirm correct creation of the table by issuing a query, such as:

SELECT \* FROM accounts\_avro LIMIT 10

1. Optional: Use the DESCRIBE or DESCRIBE FORMATTED command to list the columns and data types of the accounts\_avro table created from the Avro schema.

**Bonus Exercise**

Create a new table based on the existing accounts\_avro table data, using Parquet

for the storage format.