Assignment 7.3:

Explain the below concepts with an example in brief.

● Hive Data Definitions

● Hive Data Manipulations

● HiveQL Manipulations

**Hive Data definition:**

*Hive* is the Hive query language. Like all SQL dialects in widespread use, it doesn’t fully conform to any particular revision of the ANSI SQL standard. It is perhaps closest to MySQL’s dialect, but with significant differences. Hive offers no support for row-level inserts, updates, and deletes. Hive doesn’t support transactions. Hive adds extensions to provide better performance in the context of Hadoop and to integrate with custom extensions and even external programs.

* CREATE Database, Table – Used to create database and table.

Syntax: create database databasename;

Ex: create database student;

In the above example student is the database name.

Syntax: create table tablename;

Ex: create table studentdetails (id int, name string);

In the above example studentdetails is the name of the table. Create a table called studentdetails with two

columns, the first being an integer and the other a string.

* hive> DROP DATABASE IF EXISTS financials;

The IF EXISTS is optional and suppresses warnings if financials doesn’t exist.

* Alter Database:

You can set key-value pairs in the DBPROPERTIES associated with a database using the ALTER

DATABASE command. No other metadata about the database can be changed, including its name and

directory location:

hive> ALTER DATABASE financials SET DBPROPERTIES ('edited-by' = 'Joe Dba');

* Creating Tables:

The CREATE TABLE statement follows SQL conventions, but Hive’s version offers significant extensions to

Support a wide range of flexibility where the data files for tables are stored, the formats used, etc.

Create table table\_name (column\_name data type.......) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

**Hive Data Manipulations:**

DML refers to "Data Manipulation Language", a subset of SQL statements that modify the data stored in tables. Since Hive has no row-level insert, update, and delete operations, the only way to put data into an table is to use one of the “bulk” load operations. Or you can just write files in the correct directories by other means.

* **LOAD**: Loading data into a Hive table is one of the variants of inserting data into a Hive table. In this method, the entire file is copied/moved to a directory that corresponds to Hive tables. If the table is partitioned, then data is loaded into partitions one at a time.

Syntax: load data [LOCAL] inpath 'filepath' [OVERWRITE] into table table\_name;

Ex: load data LOCAL inpath '/home/acadgild/manjunath/employee\_details.txt' into table employee\_data;

Loading the data from local file system from D:/employee\_details.txt into table employee\_data;

* **INSERT**: Query Results can be inserted into tables by using the insert clause.

Syntax: insert into table table\_name select\_statement1 from from\_statement;

Ex: insert into table employee\_data values ('emp\_name', '12', 'bangalore');

* **UPDATE**: To update the existing table.

Syntax: update table\_name set column = value [, column = value ...] [where expression]

Ex: update employee\_data SET age = 30 WHERE name = ‘emp\_name’;

* **DELETE**: Deletes the data from the table.

Syntax: DELETE FROM table\_name [WHERE expression]

Ex: DELETE FROM employee\_data WHERE id = 2;

**HiveQL Manipulations:**

* Loading Data into Managed Tables: This command will first create the directory for the partition, if it doesn’t already exist, then copy the data to it. If the target table is not partitioned, you omit the PARTITION clause.

Syntax: load data local inpath;

Ex: '${env:HOME}/ag-employee' overwrite into table employees partition (country = 'INDIA', state = 'KA');

* Inserting Data into Tables from Queries: The insert statement lets you load data into a table from a query.

Ex: insert overwrite table employee partition (country = ‘A’, state= 'B') select \* from employees se where se.cnty = 'us' and se.st = 'or';

This is an example for the state of B, where we presume the data is already in another table called employees.

With overwrite, any previous contents of the partition (or whole table if not partitioned) are replaced. If you drop the keyword overwrite or replace it with into, Hive appends the data rather than replaces it.

* Creating Tables and Loading them in query: You can also create a table and insert query results into it in one statement:

Ex: create table employee\_data as select name, salary, address from employees where se.state = 'B';

This table contains just the name, salary, and address columns from the employee table records for employees in B. The schema for the new table is taken from the

Select clause. A common use for this feature is to extract a convenient subset of data from a larger, more unwieldy table.