Explain the below concepts with an example in brief.

**● Hive Data Definitions**

Create Database Statement

A database in Hive is a namespace or a collection of tables.

1. hive**>** CREATE SCHEMA userdb;
2. hive**>** SHOW DATABASES;

Drop database

1. hive**>** DROP DATABASE IF EXISTS userdb;

Creating Hive Tables

Create a table called Sonoo with two columns, the first being an integer and the other a string.

1. hive**>** CREATE TABLE Sonoo(foo INT, bar STRING);

Create a table called HIVE\_TABLE with two columns and a partition column called ds. The partition column is a virtual column. It is not part of the data itself but is derived from the partition that a particular dataset is loaded into.By default, tables are assumed to be of text input format and the delimiters are assumed to be ^A(ctrl-a).

1. hive**>** CREATE TABLE HIVE\_TABLE (foo INT, bar STRING) PARTITIONED BY (ds STRING);

Browse the table

1. hive**>**  Show tables;

Altering and Dropping Tables

1. hive**>** ALTER TABLE Sonoo RENAME TO Kafka;
2. hive**>** ALTER TABLE Kafka ADD COLUMNS (col INT);
3. hive**>** ALTER TABLE HIVE\_TABLE ADD COLUMNS (col1 INT COMMENT 'a comment');
4. hive**>** ALTER TABLE HIVE\_TABLE REPLACE COLUMNS (col2 INT, weight STRING, baz INT COMMENT 'baz replaces new\_col1');

● Hive Data Manipulations

There are multiple ways to modify data in Hive:

* LOAD
* INSERT
* into Hive tables from queries
* into directories from queries
* into Hive tables from SQL
* UPDATE
* DELETE
* MERGE
* EXPORT and IMPORT commands are also available

**Loading files into tables**

LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename [PARTITION (partcol1=val1, partcol2=val2 ...)]

**Inserting data into Hive Tables from queries**

INSERT OVERWRITE TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...) [IF NOT EXISTS]] select\_statement1 FROM from\_statement;

INSERT INTO TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...)] select\_statement1 FROM from\_statement;

**Writing data into the filesystem from queries**

INSERT OVERWRITE [LOCAL] DIRECTORY directory1

  [ROW FORMAT row\_format] [STORED AS file\_format] (Note: Only available starting with Hive 0.11.0)

  SELECT ... FROM …

**Inserting values into tables from SQL**

CREATE TABLE students (name VARCHAR(64), age INT, gpa DECIMAL(3, 2))

CLUSTERED BY (age) INTO 2 BUCKETS STORED AS ORC;

INSERT INTO TABLE students

VALUES ('fred flintstone', 35, 1.28), ('barney rubble', 32, 2.32)

**Update action**

UPDATE tablename SET column = value [, column = value ...] [WHERE expression]

**Delete query**

DELETE FROM tablename [WHERE expression]

**Merge**

MERGE INTO <target table> AS T USING <source expression/table> AS S

ON <boolean expression1>

WHEN MATCHED [AND <boolean expression2>] THEN UPDATE SET <set clause list>

WHEN MATCHED [AND <boolean expression3>] THEN DELETE

WHEN NOT MATCHED [AND <boolean expression4>] THEN INSERT VALUES<value list>

**HiveQL Manipulations**

HiveQL, the Hive query language, focusing on the data manipulation language parts that are used to put data into tables and to extract data from tables to the filesystem.

This chapter uses SELECT ... WHERE clauses extensively when we discuss populating tables with data queried from other tables. So, why aren’t we covering SELECT ... WHERE clauses first, instead of waiting until the next chapter, Chapter 6?

Since we just finished discussing how to create tables, we wanted to cover the next obvious topic: how to get data into these tables so you’ll have something to query! We assume you already understand the basics of SQL, so these clauses won’t be new to you. If they are, please refer to Chapter 6 for details.

Loading Data into Managed 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.

We saw an example of how to load data into a managed table in Partitioned, Managed Tables, which we repeat here with an addition, the use of the OVERWRITE keyword:

LOAD DATA LOCAL INPATH '${env:HOME}/california-employees'

OVERWRITE INTO TABLE employees

PARTITION (country = 'US', state = 'CA');