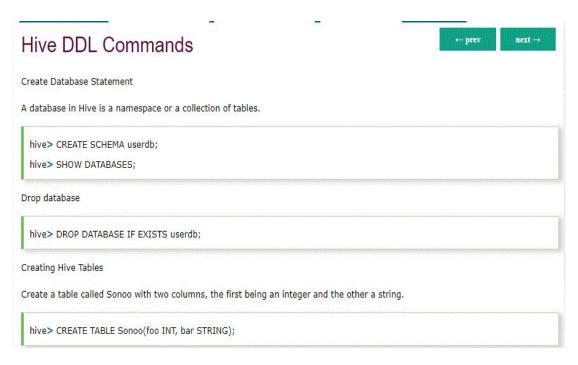
Hive Data Definition Language. Hive Data Definition Language (DDL) is a subset of Hive SQL statements that describe the data structure in Hive by creating, deleting, or altering schema objects such as databases, tables, views, partitions, and buckets.



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).

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

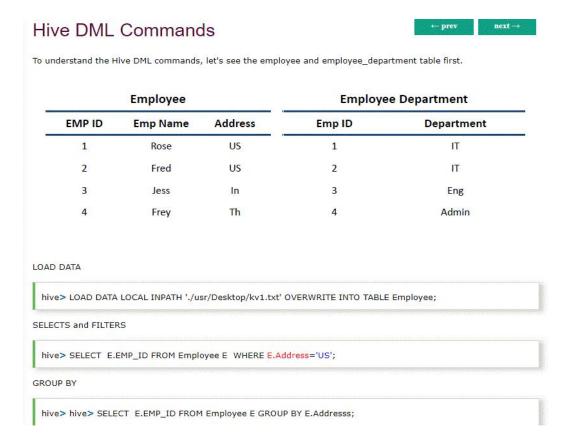
Browse the table

hive> Show tables;

Altering and Dropping Tables

hive> ALTER TABLE Sonoo RENAME TO Kafka;
hive> ALTER TABLE Kafka ADD COLUMNS (col INT);
hive> ALTER TABLE HIVE\_TABLE ADD COLUMNS (col INT comment);
hive> ALTER TABLE HIVE\_TABLE REPLACE COLUMNS (col 2 INT, weight STRING, baz INT COMMENT 'baz replaces new\_col1');

**DML** refers to "Data Manipulation Language", a subset of SQL statements that modify the data stored in tables.



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

# **HiveQL(Hive Query Language)**

Hive provides a command line interface (CLI). Hive provides a CLI to write Hive queries using Hive Query Language (Hive-QL).

Generally, HiveQL syntax is similar to the SQL syntax that most data analysts are familiar with. Hive supports four file formats those are TEXTFILE, SEQUENCEFILE, ORC and RCFILE (Record Columnar File).

- For single user meta data storage Hive uses derby database and
- For multiple user Meta data or shared Meta data case Hive uses MYSQL

# **Built-in operators**

Hive provides Built-in operators for Data operations to be implemented on the tables present inside Hive warehouse.

These operators are used for mathematical operations on operands, and it will return specific value as per the logic applied.

Types of Built-in Operators in HIVE are:

- Relational Operators
- Arithmetic Operators
- Logical Operators
- Operators on Complex types
- Complex type Constructors

## **Relational Operators:**

We use Relational operators for relationship comparisons between two operands.

- · Operators such as equals, Not equals, less than, greater than ...etc
- · The operand types are all number types in these Operators.

The following Table will give us details about Relational operators and its usage.

Built-in Operator	Description	Operand
X = Y	TRUE if expression X is equivalent to expression Y	It takes all primitive types
	Otherwise FALSE.	
X I= Y	TRUE	It takes all
	if expression X is not equivalent to expression Y	primitive types
	Otherwise FALSE.	
X <y< td=""><td>TRUE</td><td>It takes all</td></y<>	TRUE	It takes all
	if expression X is less than expression Y	primitive types
	Otherwise FALSE.	

# **Arithmetic Operators:**

The following Table will give us details about Arithmetic operators

Built-in Operator	Description	Operand	
X + Y It will return the output of adding X and Y value.		It takes all number types	
X - Y	It will return the output of subtracting Y from X value.	It takes all number types	
X * Y	It will return the output of multiplying X and Y values.	It takes all number types	
X/Y	It will return the output of dividing Y from X.	It takes all number types	
X % Y	It will return the remainder resulting from dividing X by Y.	It takes all number types	
X & Y	It will return the output of bitwise AND of X and Y.	It takes all number types	
X Y	It will return the output of bitwise OR of X and Y.	It takes all number types	
X ^ Y	It will return the output of bitwise XOR of X and Y. It takes types		
~X	It will return the output of bitwise NOT of X.	It takes all number types	

#### Logical Operators:

We use Logical operators for performing Logical operations on operands

- · Logical operations such as AND, OR, NOT between operands we use these Operators.
- The operand types all are BOOLEAN type in these Operators

The following Table will give us details about Logical operators

Operators	Description	Operands  Boolean types only	
X AND Y	TRUE if both X and Y are TRUE, otherwise FALSE.		
X && Y	Same as X AND Y but here we using && symbol Boolean		
XORY	TRUE if either X or Y or both are TRUE, otherwise FALSE.	Boolean types only	
XIIY	Same as X OR Y but here we using    symbol Boolean type		
NOT X	TRUE if X is FALSE, otherwise FALSE. Boolean to		
IX	Same as NOT X but here we using! symbol Boolean ty		

## Complex type Constructors:

The following Table will give us details about Complex type Constructors. It will construct instances on complex data types. These are of complex data types such as Array, Map and Struct types in Hive.

In this section, we are going to see the operations performed on Complex type Constructors.

Operators	Operands	Description
array	(val1, val2,)	It will create an array with the given elements as mentioned like val1, val2
Create_ union	(tag, val1, val2,)	It will create a union type with the values that is being mentioned to by the tag parameter
map	(key1, value1, key2, value2,)	It will create a map with the given key/value pairs mentioned in operands
Named_struct	(name1, val1, name2, val2,)	It will create a Struct with the given field names and values mentioned in operands
STRUCT	(val1, val2, val3,)	Creates a Struct with the given field values. Struct field names will be col1, col2, .