**Dynamic Partition in Hive**

**Overview :**

Hive is a storage system which is developed by Facebook, it is basically a data warehouse system that performs operations on structured data. It is mainly used to perform operations like analytics and MapReduce jobs.it also provides the functionality of reading, writing, and managing large datasets. Hive is mainly popular because it can serve as an alternative to the traditional approach of database operations. Hive uses indexing which helps internally the queries to be efficient, it can also operate on compressed data which is stored in the ecosystem of Hadoop.

**Types of partitioning:**

There are two types of partitioning are as follows.

1. Static Partitioning

Partitioning in Apache Hive is very much needed to improve performance while scanning the Hive tables. It allows a user working on the hive to query a small or desired portion of the Hive tables.

Suppose we have a table *student* that contains 5000 records, and we want to only process data of students belonging to the ‘A’ section only. However, the student table contains student records belonging to all the sections (A, B, C, D) but with partitioning, we do not need to process all those 5000 records. Here, partitioning helps us in separating data of students according to their sections. By doing so the time to execute the query will be increased, and we do not need to scan all the other unnecessary data available inside the ‘student’ table.

Partitioning in the hive can be static or dynamic. In this article, we will be implementing static partitioning in the hive.

### **Features of Static Partitioning**

* Partitions are manually added so it is also known as manual partition
* Data Loading in static partitioning is faster as compare to dynamic partitioning so static partitioning is preferred when we have massive files to load.
* In static partitioning individual files are loaded as per the partition we want to set.
* where clause is used in order to use limit in static partition
* Altering the partition in the static partition is allowed whereas dynamic partition doesn’t support Alter statement.

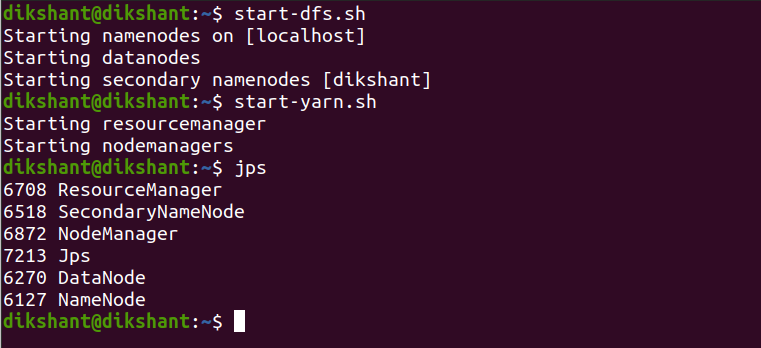
To perform the below operation make sure your hive is running. Below are the steps to launch a hive on your local system.

**Step 1:** Start all your Hadoop Daemon

start-dfs.sh # this will start namenode, datanode and secondary namenode

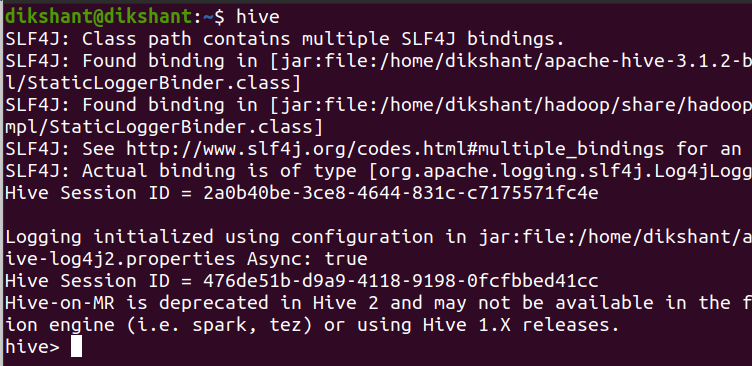
start-yarn.sh # this will start node manager and resource manager

jps # To check running daemons



**Step 2:** Launch hive from terminal

hive



Now, we are all set to perform the quick demo.

### Static Partitioning

In static partitioning, we partition the table based on some attribute. The attributes or columns we use to separate records are not present in the actual data we load to our table but we separate them using the *partition* statement available in Hive. The partitions are manually partitioned that’s why static partition is also known as manual partition. below is the well-explained example that helps you out to understand it well.

**Step 1:** Let’s [create a table](https://www.geeksforgeeks.org/how-to-create-table-in-hive) ‘student’ with the following attributes (student\_name, father\_name, and percentage ) and partition it using ‘section’ in our default database.

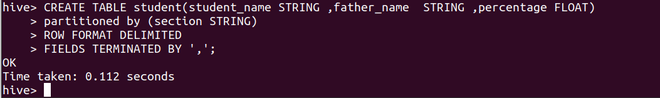
**Note:** Do not provide the partitioned columns name in *create table <table-name>* statement. Once you mentioned the names in the *partitioned by* the statement they will automatically partition with their respective attributes in the hive table.

CREATE TABLE student(student\_name STRING ,father\_name STRING ,percentage FLOAT)

partitioned by (section STRING)

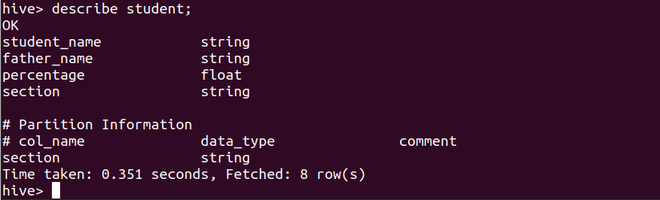
ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',';



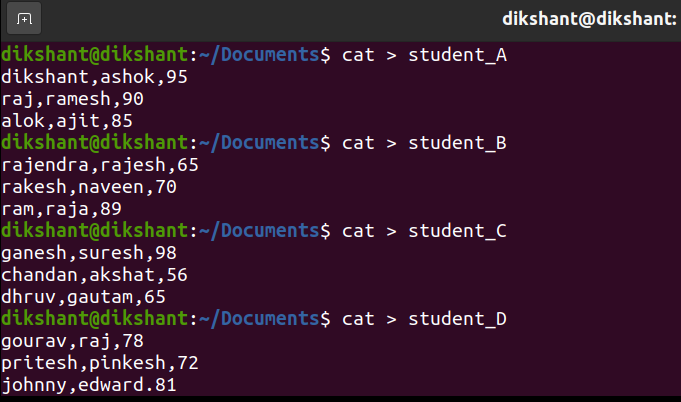
**Step 2:** Describe the table to see information about table attributes and partitioned columns

describe student;



Here we can see that the *section* column has been marked as the partition attribute and also has been added to the table attribute list.

**Step 3:** Create 4 different files containing data of students from respective sections (student-A, student-B, student-C, student-D) make sure that *section* column which is our partition column is never added to the actual table.



Creating student table according to their sections

**Step 4:** Load the data from 4 different files containing data of student section-wise to our student table along with the partitioned attribute value.

**a.** Loading data from student\_A partitioned with section ‘A’

LOAD DATA LOCAL INPATH '/home/dikshant/Documents/student\_A' INTO TABLE student

partition(section = "A");

**b.** Loading data from student\_B partitioned with section ‘B’

LOAD DATA LOCAL INPATH '/home/dikshant/Documents/student\_B' INTO TABLE student

partition(section = 'B');

**c.** Loading data from student\_C partitioned with section ‘C’

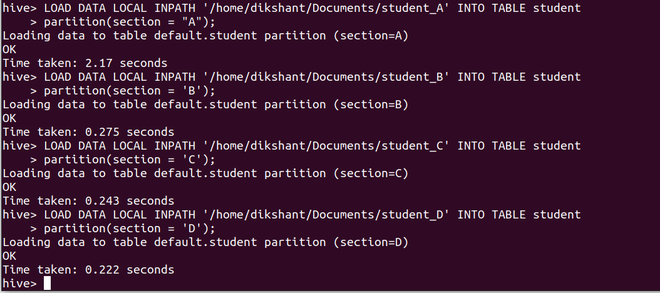
LOAD DATA LOCAL INPATH '/home/dikshant/Documents/student\_C' INTO TABLE student

partition(section = 'C');

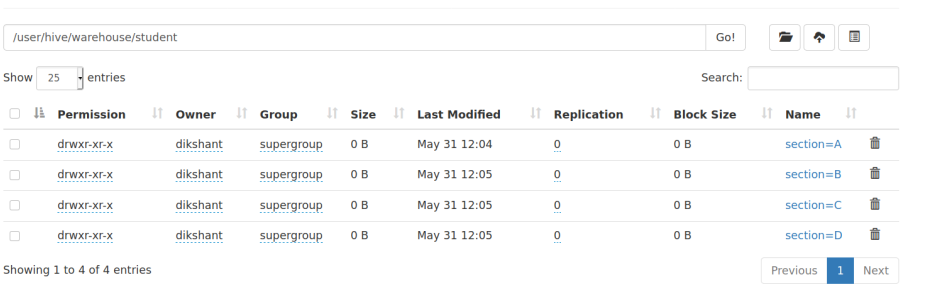
**d**. Loading data from student\_D partitioned with section ‘D’

LOAD DATA LOCAL INPATH '/home/dikshant/Documents/student\_D' INTO TABLE student

partition(section = 'D');



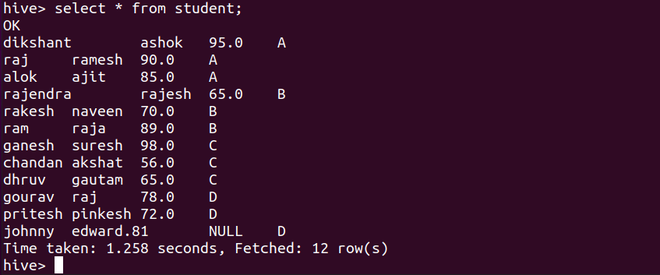
**Step 5:** Now go to your [HDFS](https://www.geeksforgeeks.org/hadoop-hdfs-hadoop-distributed-file-system/)(/user/hive/warehouse/) and check the student table to see how the partitions are made.



Here we can easily observe that the student table is partitioned and contain data of student in accordance to their section. Now if we want to process data of student belongs to section\_A we need not traverse the whole table since it is partitioned. Each partition contains the data of the table as per the mentioned partition.

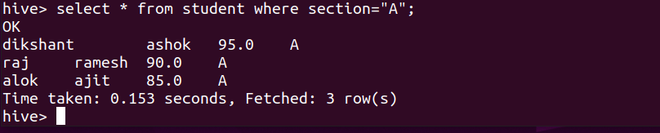
Below select query will select everything from the *student* table. As we can observe we haven’t added any section column but since it is mentioned manually in a partition so it is added.

select \* from student;



Now if we operate on this student table with where clause as seen below. Hive will never query all those 12 records. It is simply perform processing on the partition made by us in this student table. In our case, it will only query **section=A** partition.

select \* from student where section="A";



This is how we perform static partitioning in the hive.

1. **Dynamic Partitioning**

**Dynamic Partitioning :**

Dynamic partitioning is the strategic approach to load the data from the non-partitioned table where the single insert to the partition table is called a dynamic partition. In dynamic partitioning, the values of the partitioned tabled are existed by default so there is no need to pass the value for those columns manually.

**Features of Dynamic Partitioning :**

Some of the features of Dynamic Partitioning are as follows.

* To handle large amounts of data that are stored in distributed storage that cases dynamic partitioning is the best choice.
* Dynamic partitioning can be performed on the hive external table and managed table.
* In Dynamic partitioning, there is no requirement of the where clause.
* If you want to perform partition on the tables without knowing the number of columns in that case you can use Dynamic partitioning.

**Operations of Dynamic Partitioning :**

Now let’s perform some of the operations of Dynamic Partitioning You can prefer ubuntu which would be fast for the operations are as follows.

**Step-1 :**

First of all, we need to create a database in which you want to perform the operation of the creation of a table.

hive>Create database dynamic\_Demo;

hive>use dynamic\_demo //here we have selected the above created database.

**Step-2 :**

After selection of database from the available list. Now we will enable the dynamic partition

using the following commands are as follows.

hive> set hive.exec.dynamic.partition=true;

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

**Step-3 :**

Create any table with a suitable table name to store the data.

hive>CREATE TABLE food\_prices (

series\_reference string,

Period string,

data\_value int,

status string,

units string,

subject string,

product string,

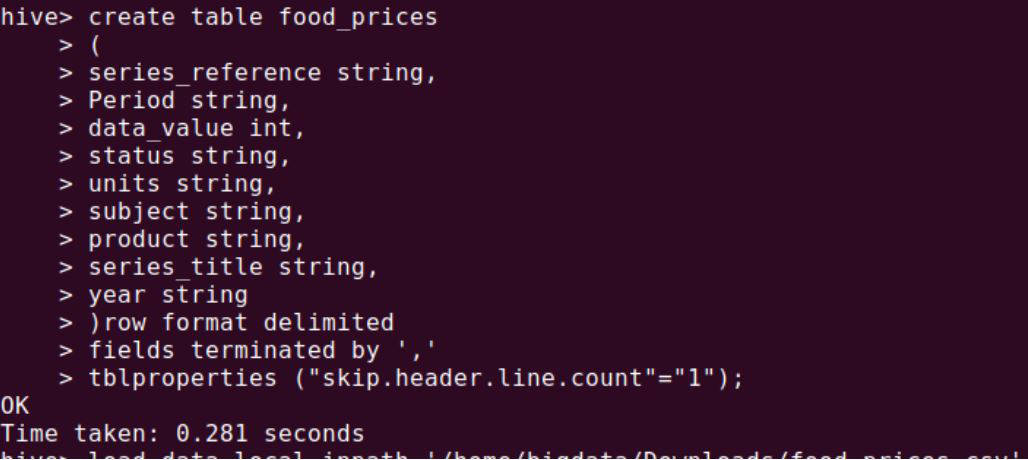
series\_title string,

year string

)ROW FORMAT DELIMITED

fields terminated by ','

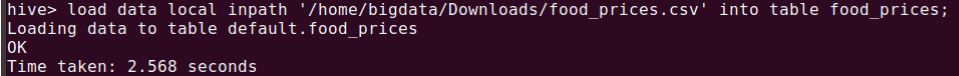
TBLPROPERTIES ("skip.header.line.count"="1");



**Step-4 :**   
Now at the next step let’s load the data which we have created by the following command.

hive>load data local inpath '/home/bigdata/Downloads/food\_prices.csv'

into table food\_prices;



**Step-5 :**    
To create a partitioned table we have to follow the below command.

create table partitioned\_food\_prices

(

series\_reference string,

Period string,

data\_value int,

status string,

units string,

subject string,

product string,

series\_title string

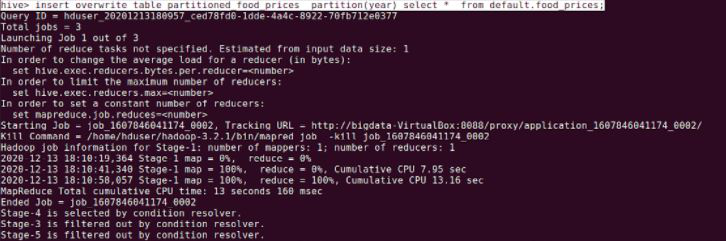
)partitioned by (year string);

**Step-6 :**

Now we will load the data into the partitioned table using the following command.

hive> insert overwrite table partitioned\_food\_prices partition(year)

select \* from default.food\_prices;



**Step-7 :**  
let’s perform some query operations.

hive>select \* partitioned\_food\_prices limit 5



**Step-8 :**   
The final step is to delete the dynamic partition column.

hive>alter table partioned\_food\_prices drop partition (year = '2019')

