



# Apache Hive Subqueries Views & Indexs

by Sumit Mittal



# IMPORTANT

## Copyright Infringement and Illegal Content Sharing Notice

All course content designs, video, audio, text, graphics, logos, images are Copyright© and are protected by India and international copyright laws. All rights reserved.

Permission to download the contents (wherever applicable) for the sole purpose of individual reading and preparing yourself to crack the interview only. Any other use of study materials – including reproduction, modification, distribution, republishing, transmission, display – without the prior written permission of Author is strictly prohibited.

**Trendytech Insights** legal team, along with thousands of our students, actively searches the Internet for copyright infringements. Violators subject to prosecution.

# Subqueries in Hive



# Subqueries

Subqueries are queries which return a result set which are nested within other queries.

Subqueries in Hive can be used:

- FROM clause
- WHERE clause.

## Create a table named *products* inside trendytech database:

```
create table products (  
  id string,  
  title string,  
  cost float  
)  
row format delimited  
fields terminated by ','  
stored as textfile;
```



The screenshot shows a terminal window titled "cloudera@quickstart:~". The terminal has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The command being executed is:

```
hive> create table products (  
  > id string,  
  > title string,  
  > cost float  
  > )  
  > row format delimited  
  > fields terminated by ','  
  > stored as textfile;
```

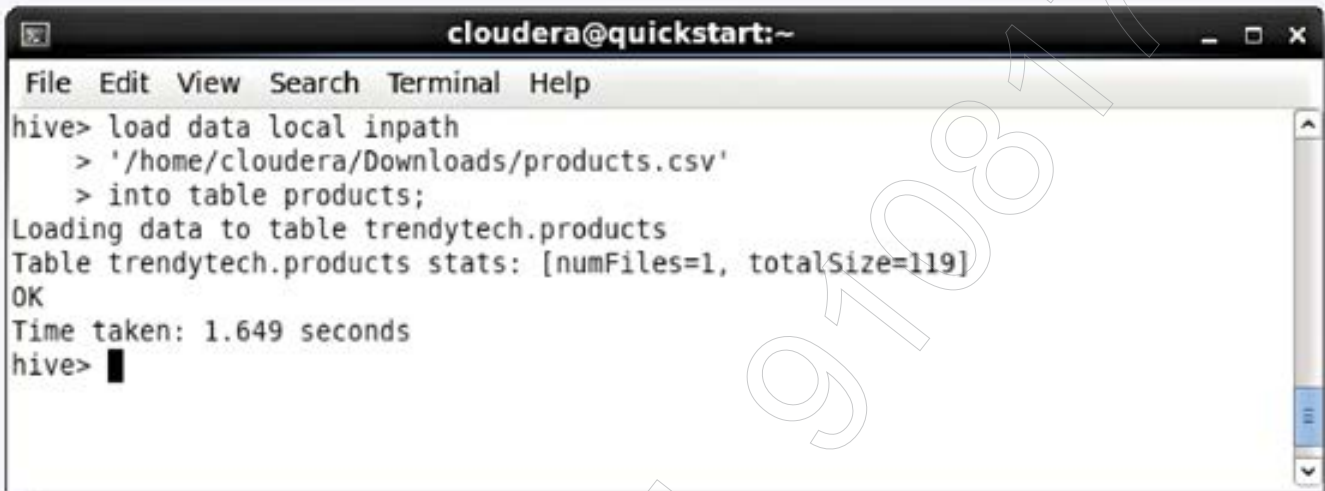
The output of the command is:

```
OK  
Time taken: 0.609 seconds  
hive> █
```



## Load data into *products* table:

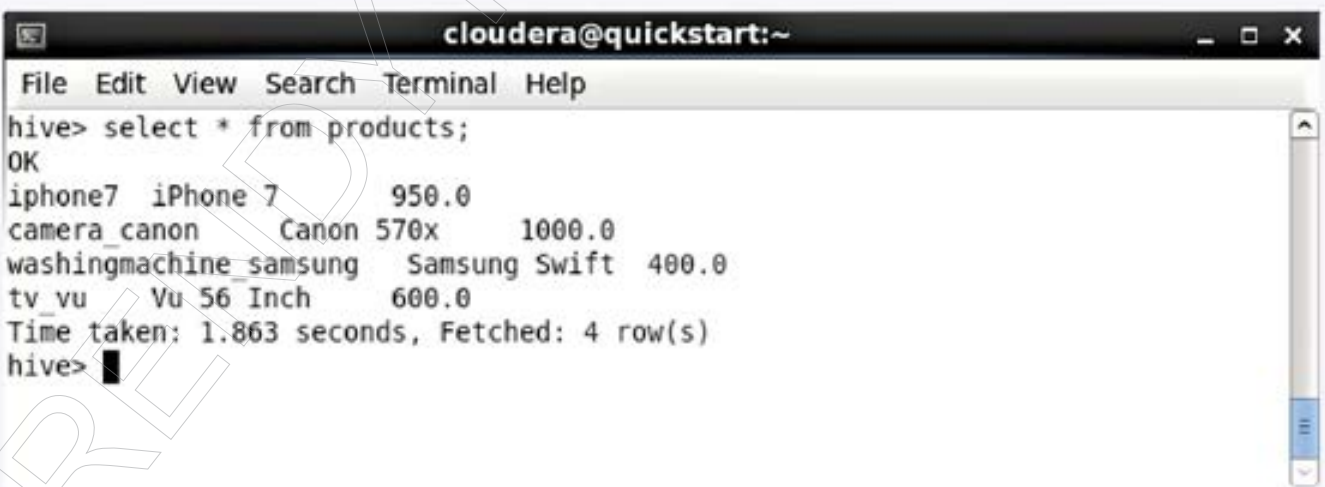
```
load data local inpath  
'/home/cloudera/Downloads/products.csv'  
into table products;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> load data local inpath  
  > '/home/cloudera/Downloads/products.csv'  
  > into table products;  
Loading data to table trendytech.products  
Table trendytech.products stats: [numFiles=1, totalSize=119]  
OK  
Time taken: 1.649 seconds  
hive> █
```

## Display all records of *products* table:

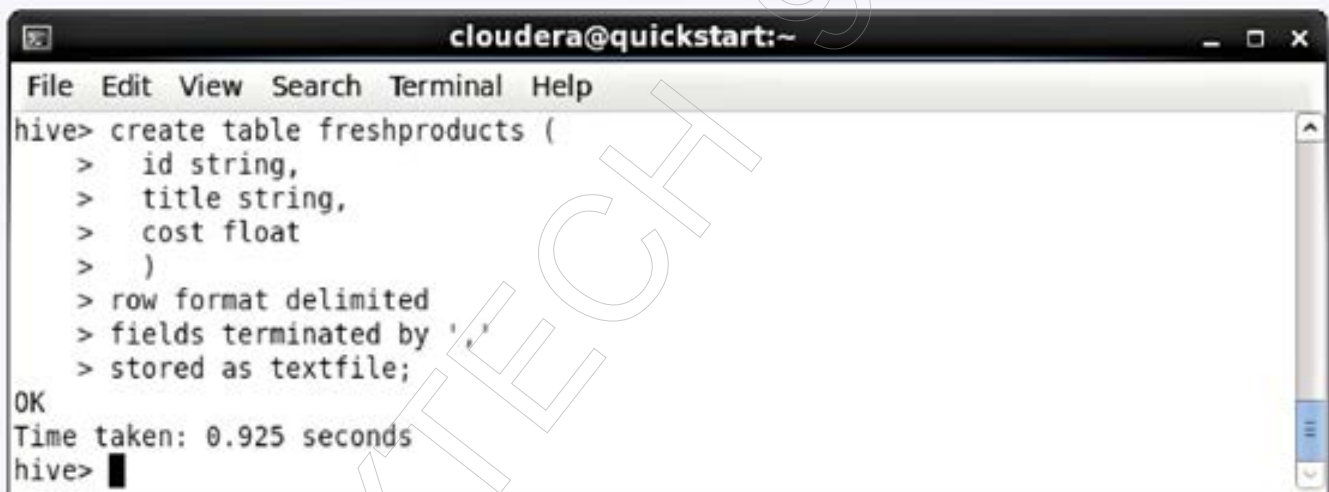
```
select * from products;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> select * from products;  
OK  
iphone7  iPhone 7      950.0  
camera_canon  Canon 570x      1000.0  
washingmachine_samsung  Samsung Swift  400.0  
tv_vu  Vu 56 Inch      600.0  
Time taken: 1.863 seconds, Fetched: 4 row(s)  
hive> █
```

## Create another table named *freshproducts* inside trendytech database:

```
create table freshproducts (  
  id string,  
  title string,  
  cost float  
)  
row format delimited  
fields terminated by ','  
stored as textfile;
```

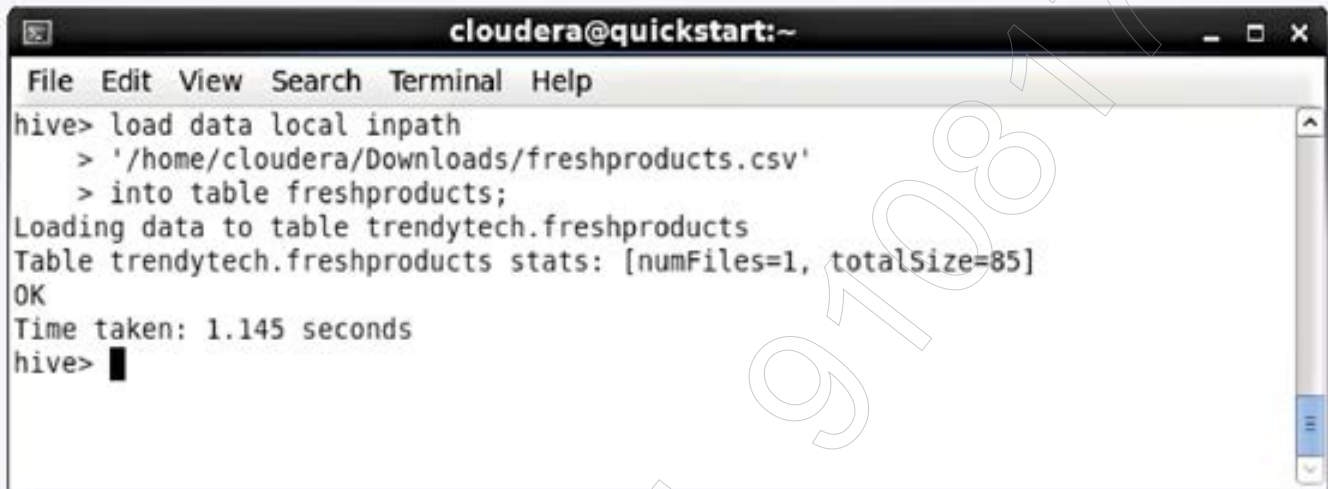


A screenshot of a terminal window titled "cloudera@quickstart:~". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal shows the following command sequence:

```
hive> create table freshproducts (  
  > id string,  
  > title string,  
  > cost float  
  > )  
  > row format delimited  
  > fields terminated by ','  
  > stored as textfile;  
OK  
Time taken: 0.925 seconds  
hive> █
```

## Load data into *freshproducts* table:

```
load data local inpath  
'/home/cloudera/Downloads/freshproducts.csv'  
into table freshproducts;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> load data local inpath  
      > '/home/cloudera/Downloads/freshproducts.csv'  
      > into table freshproducts;  
Loading data to table trendytech.freshproducts  
Table trendytech.freshproducts stats: [numFiles=1, totalSize=85]  
OK  
Time taken: 1.145 seconds  
hive> █
```

## Display all records of *freshproducts* table:

```
select * from products;
```



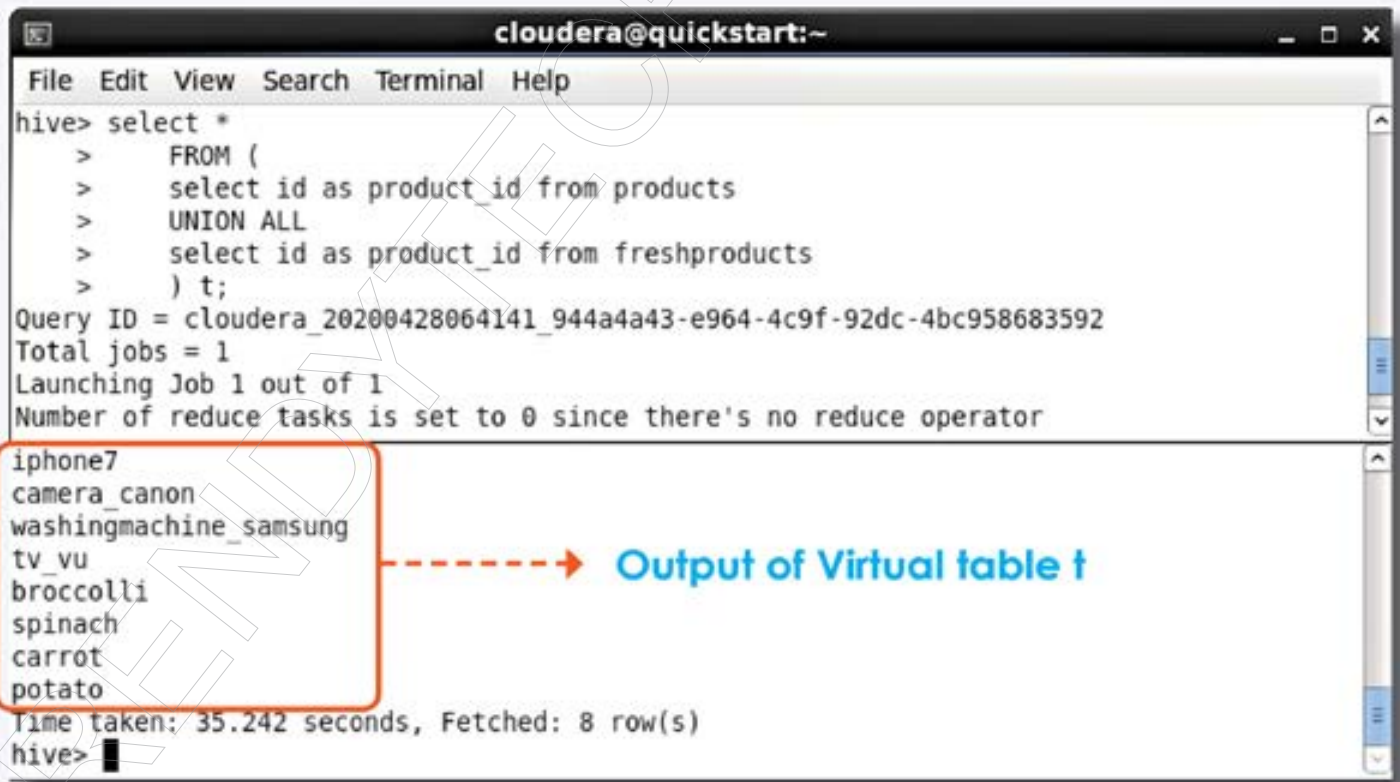
```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> select * from freshproducts;  
OK  
broccoli  Broccoli  5.0  
spinach  Spinach  7.0  
carrot    Local Carrots  4.0  
potato    Idaho Potatoes  4.0  
Time taken: 0.112 seconds, Fetched: 4 row(s)  
hive> █
```



# Subqueries in the FROM clause:

## Display records using Subqueries using FROM clause:

```
select *  
  FROM (  
    select id as product_id from products  
  UNION ALL  
    select id as product_id from freshproducts  
  ) t;
```

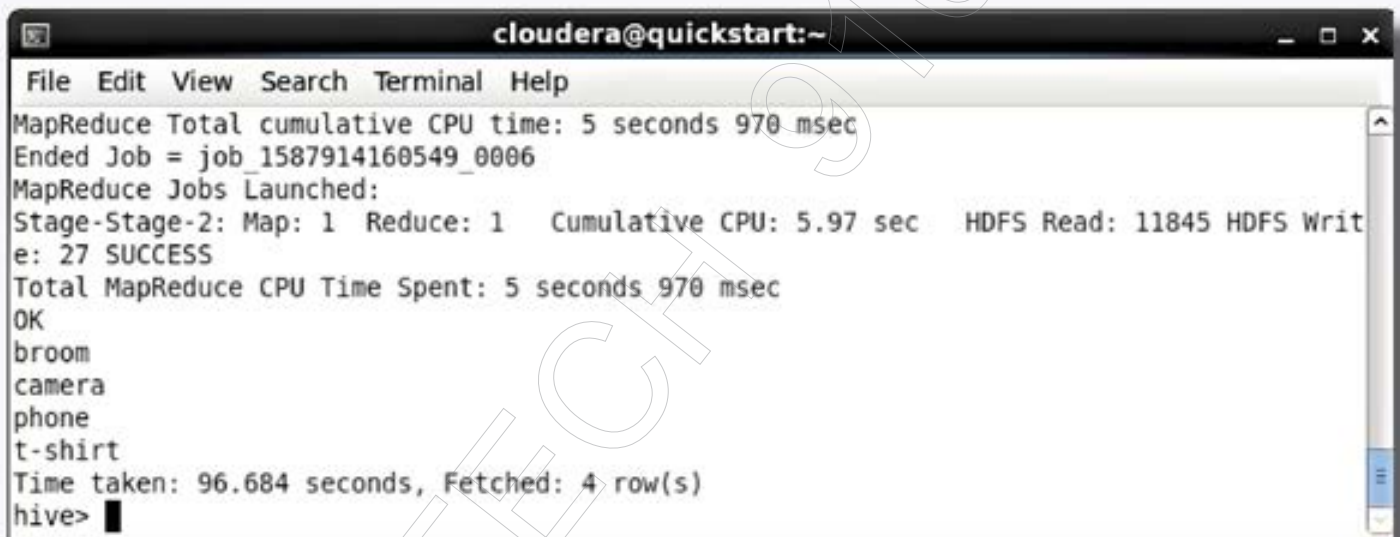


```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> select *  
  > FROM (  
  >   select id as product_id from products  
  >   UNION ALL  
  >   select id as product_id from freshproducts  
  >   ) t;  
Query ID = cloudera_20200428064141_944a4a43-e964-4c9f-92dc-4bc958683592  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks is set to 0 since there's no reduce operator  
iphone7  
camera_canon  
washingmachine_samsung  
tv_vu  
broccoli  
spinach  
carrot  
potato  
Time taken: 35.242 seconds, Fetched: 8 row(s)  
hive>
```

Output of Virtual table t

## One more example of Subqueries using FROM clause:

```
select distinct(t.product_id)
  FROM (
    select product_id from customers
  JOIN
    orders where customers.id=orders.customer_id
  ) t;
```

A screenshot of a terminal window titled 'cloudera@quickstart:~'. The window shows the output of a Hive query. The output includes MapReduce statistics, job ID, and a list of product IDs. The terminal text is as follows:

```
File Edit View Search Terminal Help
MapReduce Total cumulative CPU time: 5 seconds 970 msec
Ended Job = job_1587914160549_0006
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 5.97 sec HDFS Read: 11845 HDFS Write: 27 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 970 msec
OK
broom
camera
phone
t-shirt
Time taken: 96.684 seconds, Fetched: 4 row(s)
hive>
```

## Subqueries in the WHERE clause:

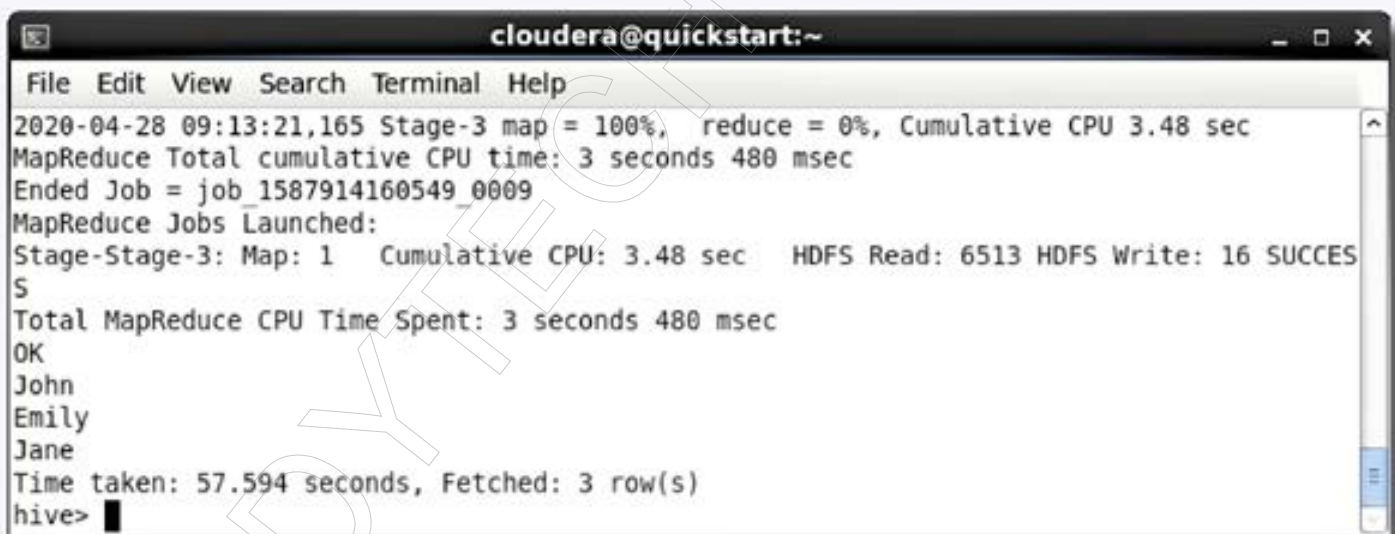
Hive Supports two types of Subqueries in WHERE clause:

- IN / NOT IN
- EXISTS / NOT EXISTS

# Subqueries with "IN & NOT IN" in the WHERE clause:

## Subquery with "IN" clause within WHERE clause:

```
select name from customers
WHERE customers.id IN
(
select customer_id from orders
);
```

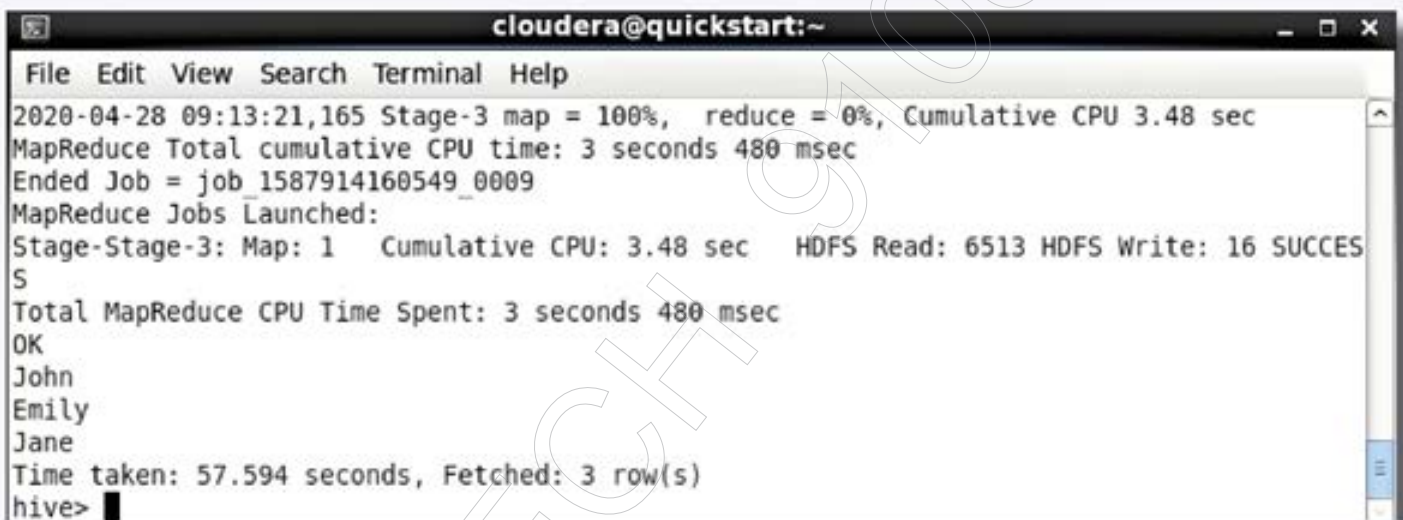
A screenshot of a terminal window titled 'cloudera@quickstart:~'. The window shows the output of a Hive query. The output includes job status information: '2020-04-28 09:13:21,165 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.48 sec', 'MapReduce Total cumulative CPU time: 3 seconds 480 msec', 'Ended Job = job\_1587914160549\_0009', and 'MapReduce Jobs Launched:'. It also shows the results of the query: 'Stage-Stage-3: Map: 1 Cumulative CPU: 3.48 sec HDFS Read: 6513 HDFS Write: 16 SUCCESS', 'Total MapReduce CPU Time Spent: 3 seconds 480 msec', 'OK', and the list of names 'John', 'Emily', 'Jane'. Finally, it shows 'Time taken: 57.594 seconds, Fetched: 3 row(s)' and the prompt 'hive>'.

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
2020-04-28 09:13:21,165 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.48 sec  
MapReduce Total cumulative CPU time: 3 seconds 480 msec  
Ended Job = job_1587914160549_0009  
MapReduce Jobs Launched:  
Stage-Stage-3: Map: 1 Cumulative CPU: 3.48 sec HDFS Read: 6513 HDFS Write: 16 SUCCESS  
OK  
John  
Emily  
Jane  
Time taken: 57.594 seconds, Fetched: 3 row(s)  
hive>
```



## Subquery with "NOT IN" clause within WHERE clause:

```
select name from customers
WHERE customers.id NOT IN
(
select customer_id from orders
);
```



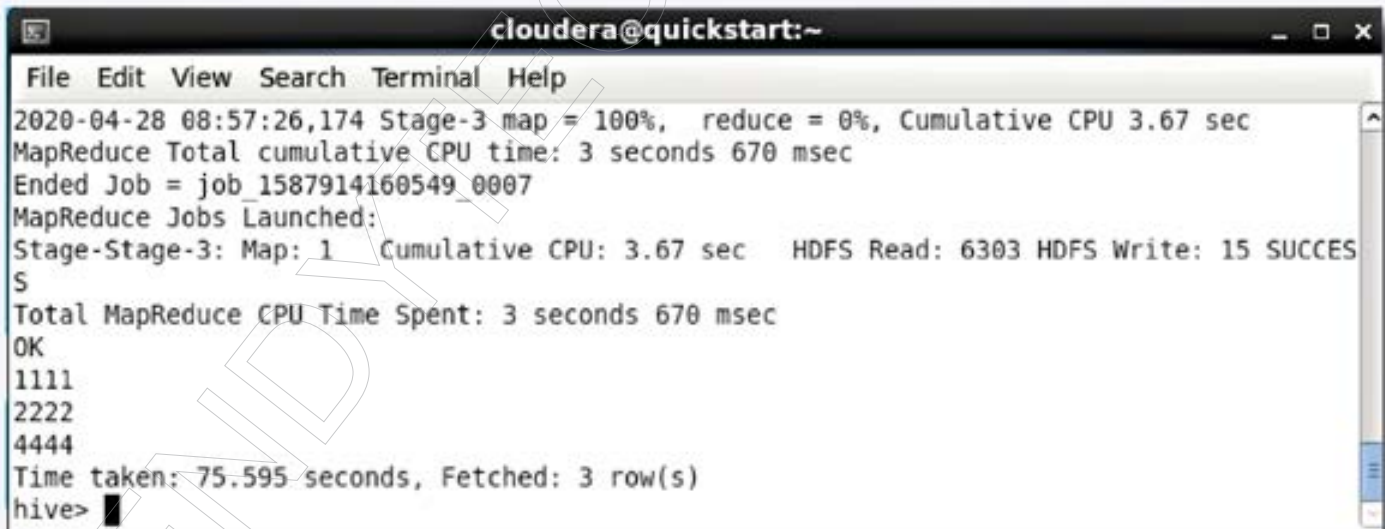
The screenshot shows a terminal window titled "cloudera@quickstart:~". The terminal output displays the results of a Hive query execution. It includes details about the job (job\_1587914160549\_0009), the number of maps (1), the cumulative CPU time (3.48 sec), and the HDFS read/write statistics (6513 reads, 16 writes). The query returned 3 rows, which are listed as John, Emily, and Jane. The total time taken for the query was 57.594 seconds.

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
2020-04-28 09:13:21,165 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.48 sec  
MapReduce Total cumulative CPU time: 3 seconds 480 msec  
Ended Job = job_1587914160549_0009  
MapReduce Jobs Launched:  
Stage-Stage-3: Map: 1 Cumulative CPU: 3.48 sec HDFS Read: 6513 HDFS Write: 16 SUCCESS  
Total MapReduce CPU Time Spent: 3 seconds 480 msec  
OK  
John  
Emily  
Jane  
Time taken: 57.594 seconds, Fetched: 3 row(s)  
hive>
```

# Subqueries with "EXIST & NOT EXIST" in the WHERE clause:

## Subquery with EXISTS clause within WHERE clause:

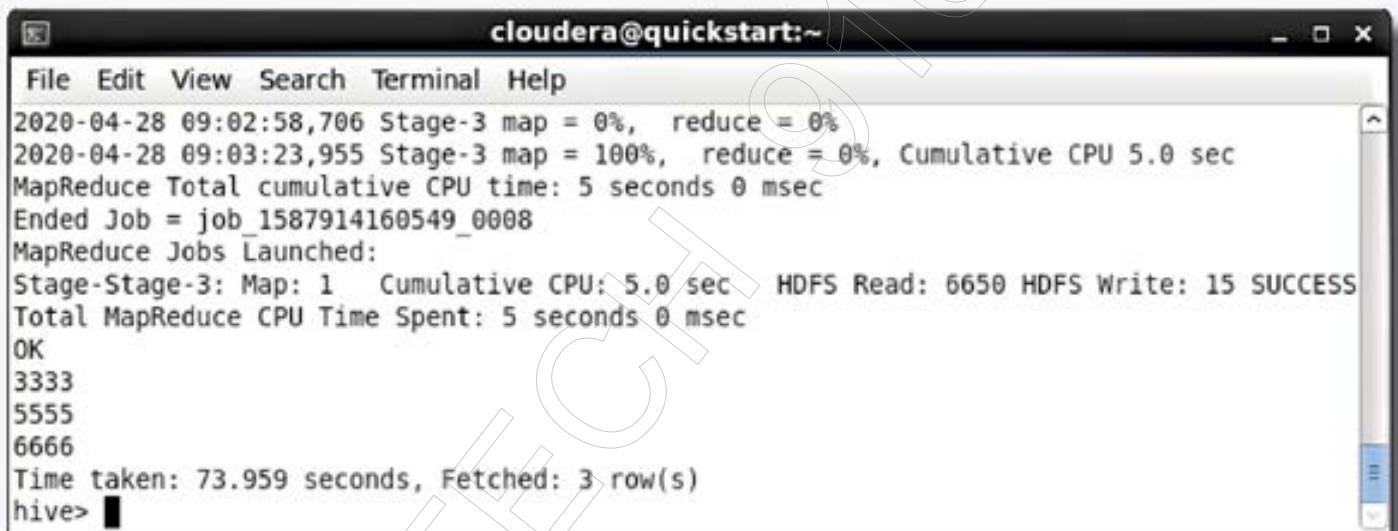
```
select id from customers
WHERE EXISTS
(
select customer_id from orders
where orders.customer_id = customers.id
);
```

A screenshot of a terminal window titled 'cloudera@quickstart:~'. The window shows the output of a Hive query. The output includes job status information: '2020-04-28 08:57:26,174 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.67 sec', 'MapReduce Total cumulative CPU time: 3 seconds 670 msec', 'Ended Job = job\_1587914160549\_0007', 'MapReduce Jobs Launched:', 'Stage-Stage-3: Map: 1 Cumulative CPU: 3.67 sec HDFS Read: 6303 HDFS Write: 15 SUCCESS', 'Total MapReduce CPU Time Spent: 3 seconds 670 msec', 'OK', and the query results: '1111', '2222', '4444'. At the bottom, it says 'Time taken: 75.595 seconds, Fetched: 3 row(s)' and 'hive>'.

```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
2020-04-28 08:57:26,174 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.67 sec  
MapReduce Total cumulative CPU time: 3 seconds 670 msec  
Ended Job = job_1587914160549_0007  
MapReduce Jobs Launched:  
Stage-Stage-3: Map: 1 Cumulative CPU: 3.67 sec HDFS Read: 6303 HDFS Write: 15 SUCCESS  
OK  
1111  
2222  
4444  
Time taken: 75.595 seconds, Fetched: 3 row(s)  
hive>
```

## Subquery with NOT EXISTS clause within WHERE clause:

```
select id from customers
WHERE NOT EXISTS
(
select customer_id from orders
where orders.customer_id = customers.id
);
```

A screenshot of a terminal window titled 'cloudera@quickstart:~'. The window shows the output of a Hive query. The output includes progress information for Stage-3, cumulative CPU time, and the results of the query. The results are three rows: 3333, 5555, and 6666. The terminal also shows the time taken and the number of rows fetched.

```
cloudera@quickstart:~
File Edit View Search Terminal Help
2020-04-28 09:02:58,706 Stage-3 map = 0%, reduce = 0%
2020-04-28 09:03:23,955 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 5.0 sec
MapReduce Total cumulative CPU time: 5 seconds 0 msec
Ended Job = job_1587914160549_0008
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1 Cumulative CPU: 5.0 sec HDFS Read: 6650 HDFS Write: 15 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 0 msec
OK
3333
5555
6666
Time taken: 73.959 seconds, Fetched: 3 row(s)
hive>
```

# Views in Hive





# Views

A view is a virtual table, which provides access to a subset of data from one or more table.

- Stored as a query in Hive's metastore
- Executed when used
- Updated when data in the underlying table changes
- Contains data from single or multiple tables
- Frozen in time, not affected by table changes.


**Before creating a view describe the *customers* and *orders* table to display schema informations:**

`describe customers;`



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> Describe customers;  
OK  
id                bigint  
name              string  
address           string  
Time taken: 0.161 seconds, Fetched: 3 row(s)  
hive> █
```

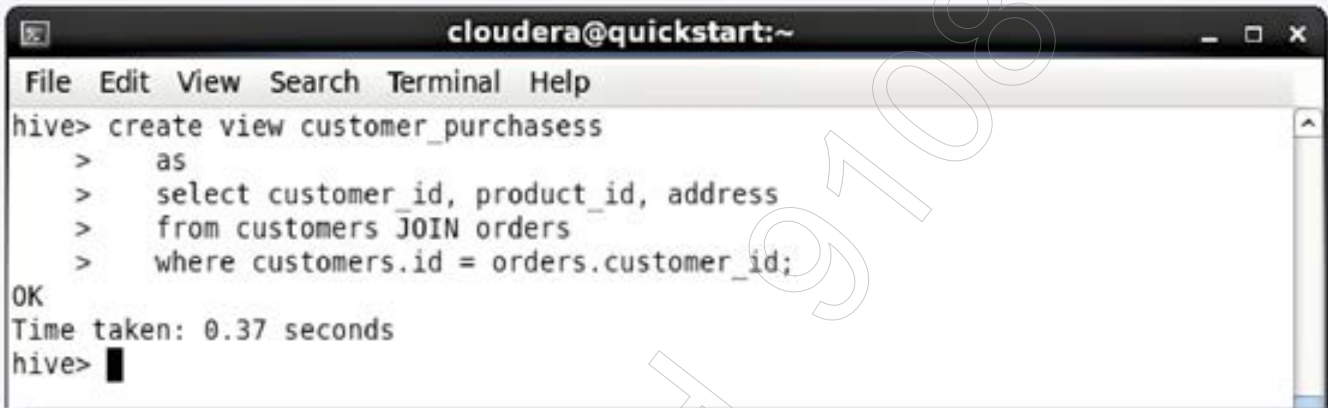
`describe orders;`



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> Describe orders;  
OK  
id                bigint  
product_id        string  
customer_id        bigint  
quantity           int  
amount             double  
Time taken: 0.095 seconds, Fetched: 5 row(s)  
hive> █
```

## Create a view on *customers* and *orders* table:

```
create view customer_purchases  
as  
select customer_id, product_id, address  
from customers JOIN orders  
where customers.id = orders.customer_id;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> create view customer_purchases  
> as  
> select customer_id, product_id, address  
> from customers JOIN orders  
> where customers.id = orders.customer_id;  
OK  
Time taken: 0.37 seconds  
hive> █
```

## To display the view (virtual) table:

```
show tables;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> show tables;  
OK  
all orders  
customer_purchases  
customers  
freshproducts  
mobilephones  
mobilephones_new  
orders  
orders_no_partition  
orders_no_partition1  
products  
Time taken: 0.497 seconds, Fetched: 10 row(s)  
hive> █
```



## Describe the view table to display schema informations:

```
describe customer_purchases;
```



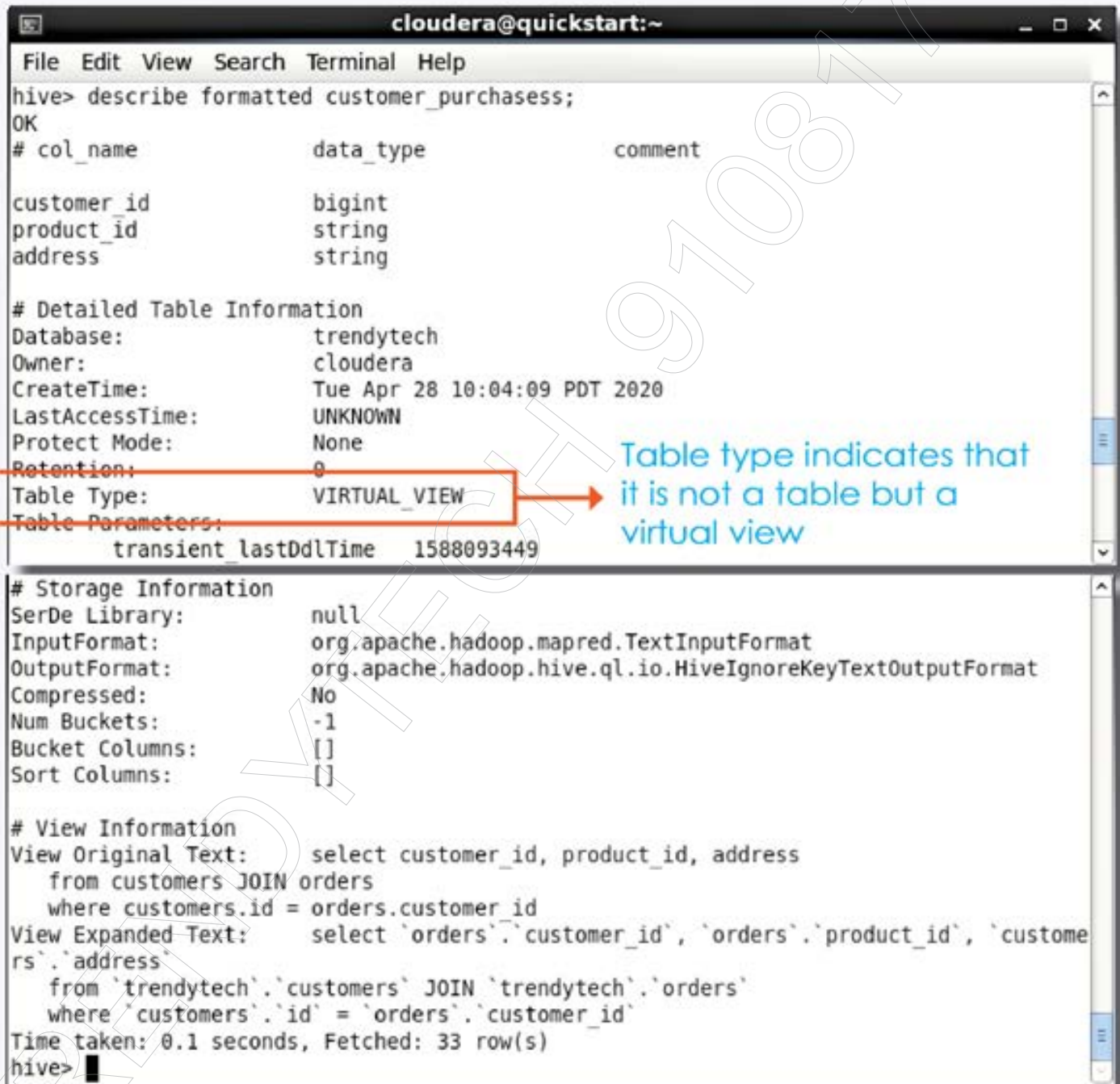
```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> describe customer_purchases;  
OK  
customer_id      bigint  
product_id       string  
address          string  
Time taken: 0.306 seconds, Fetched: 3 row(s)  
hive>
```

**Note:** datatypes are taken from their original tables



## Run *describe formatted* to display detailed view table informations:

```
describe formatted customer_purchases;
```

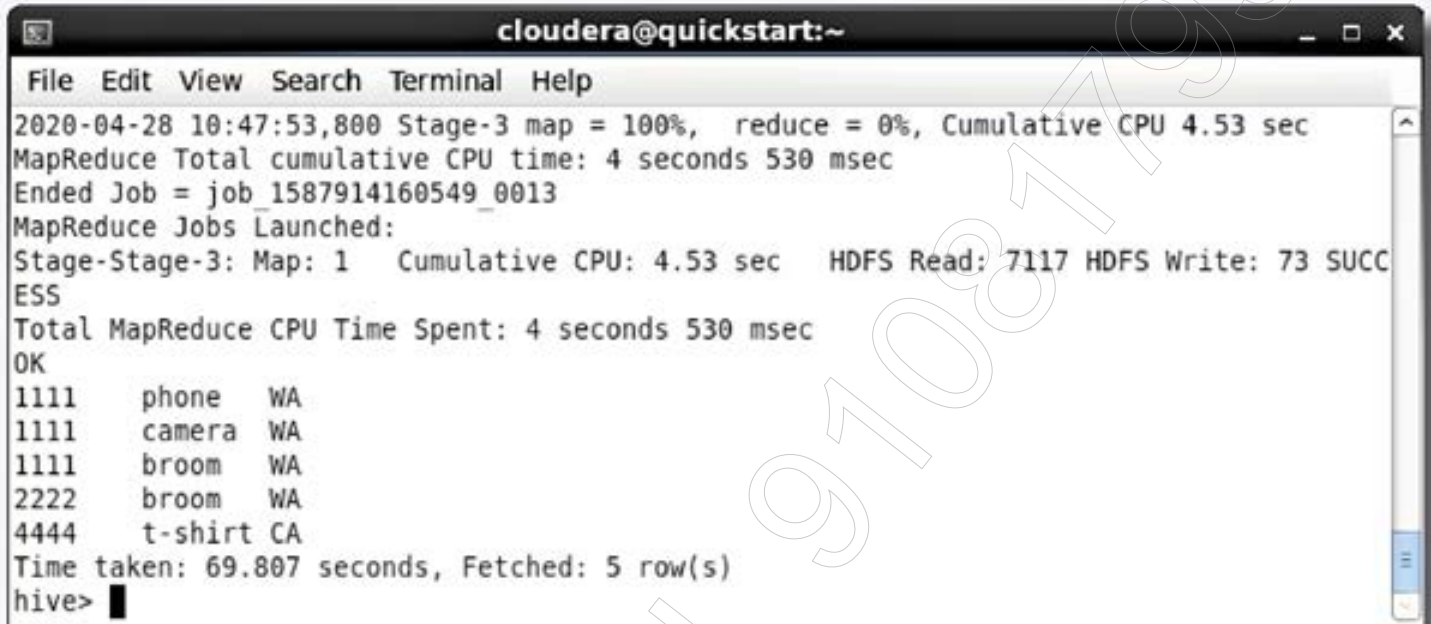


```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> describe formatted customer_purchases;  
OK  
# col_name          data_type          comment  
  
customer_id         bigint  
product_id          string  
address             string  
  
# Detailed Table Information  
Database:           trendytech  
Owner:              cloudera  
CreateTime:         Tue Apr 28 10:04:09 PDT 2020  
LastAccessTime:     UNKNOWN  
Protect Mode:       None  
Retention:          0  
Table Type:         VIRTUAL_VIEW  
Table Parameters:   transient_lastDdlTime 1588093449  
  
# Storage Information  
SerDe Library:      null  
InputFormat:        org.apache.hadoop.mapred.TextInputFormat  
OutputFormat:       org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat  
Compressed:         No  
Num Buckets:        -1  
Bucket Columns:     []  
Sort Columns:       []  
  
# View Information  
View Original Text:  select customer_id, product_id, address  
                    from customers JOIN orders  
                    where customers.id = orders.customer_id  
View Expanded Text:  select `orders`.`customer_id`, `orders`.`product_id`, `customers`.`address`  
                    from `trendytech`.`customers` JOIN `trendytech`.`orders`  
                    where `customers`.`id` = `orders`.`customer_id`  
Time taken: 0.1 seconds, Fetched: 33 row(s)  
hive>
```

Table type indicates that it is not a table but a virtual view

## Run select query to display records of a view:

```
select * from customer_purchases;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
2020-04-28 10:47:53,800 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 4.53 sec  
MapReduce Total cumulative CPU time: 4 seconds 530 msec  
Ended Job = job_1587914160549_0013  
MapReduce Jobs Launched:  
Stage-Stage-3: Map: 1 Cumulative CPU: 4.53 sec HDFS Read: 7117 HDFS Write: 73 SUCCESS  
Total MapReduce CPU Time Spent: 4 seconds 530 msec  
OK  
1111 phone WA  
1111 camera WA  
1111 broom WA  
2222 broom WA  
4444 t-shirt CA  
Time taken: 69.807 seconds, Fetched: 5 row(s)  
hive>
```

**Note:** It will trigger a MapReduce job because it expands the select operation as a subquery in background.

# Hive Index





# Index

Hive index is used to speed up the performance of queries on certain columns of a table.

Without an index, queries with predicates like 'WHERE tab1.col1 = 10' load the entire table or partition and process all the rows. But if an index exists for col1, then only a portion of the file needs to be loaded and processed.

Indexing can be used:

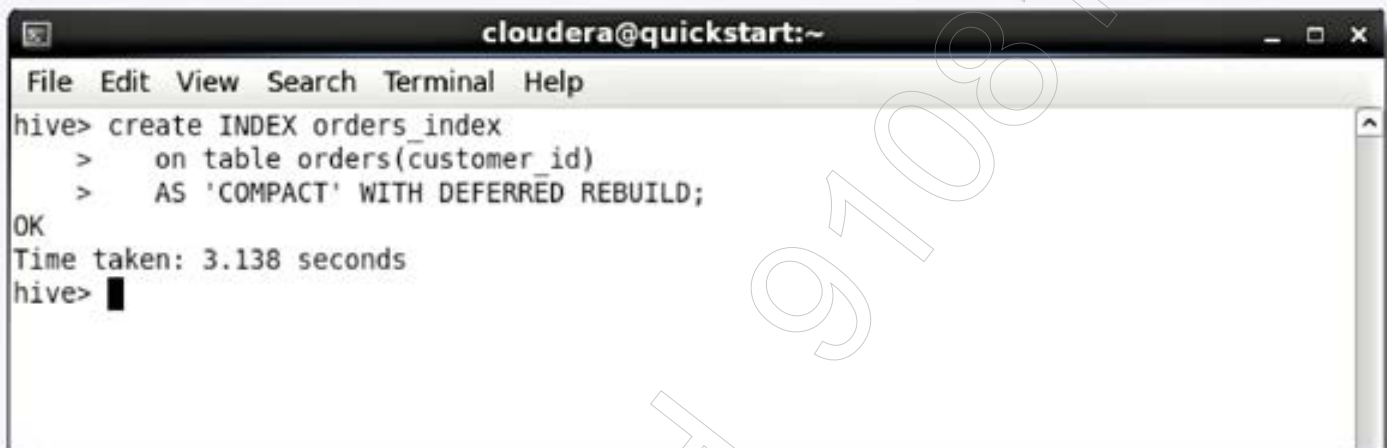
- When the dataset is very large.
- When the query execution is taking more amount of time than expected.
- When a fast query execution is required.

**Note:** Indexing Is Removed since Hive version 3.0



## Create an Index on *customer\_id* column of orders table:

```
create INDEX orders_index  
on table orders(customer_id)  
AS 'COMPACT' WITH DEFERRED REBUILD;
```

A screenshot of a terminal window titled "cloudera@quickstart:~". The terminal shows a Hive command being executed: "hive> create INDEX orders\_index on table orders(customer\_id) AS 'COMPACT' WITH DEFERRED REBUILD;". The output shows "OK" and "Time taken: 3.138 seconds". The prompt "hive>" is followed by a cursor.

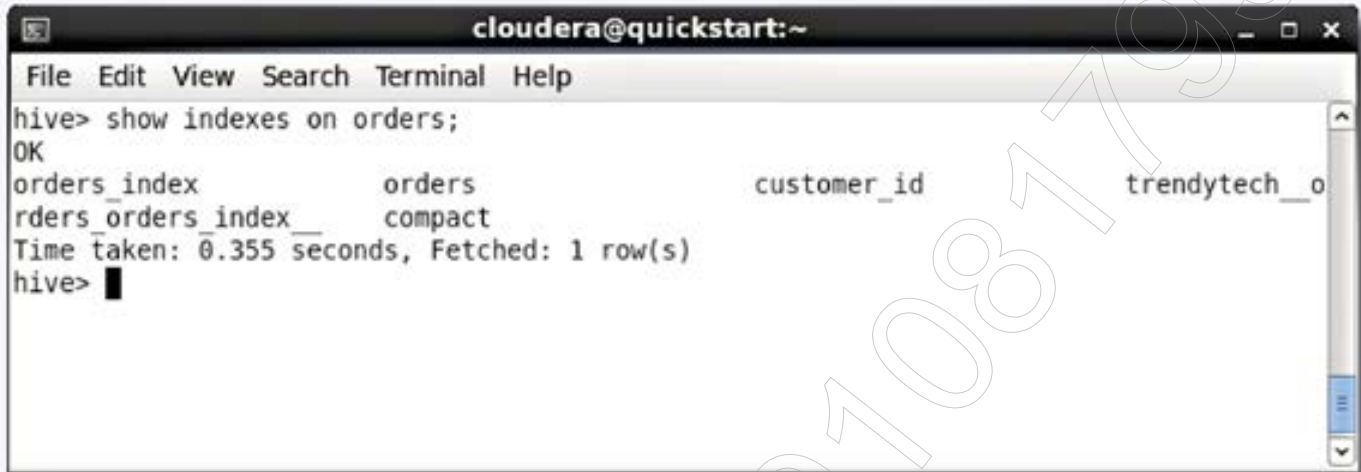
```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> create INDEX orders_index  
  > on table orders(customer_id)  
  > AS 'COMPACT' WITH DEFERRED REBUILD;  
OK  
Time taken: 3.138 seconds  
hive> █
```

**Note:** Here COMPACT means we are creating a compact index for the table.

The WITH DEFERRED REBUILD statement is because we need to alter the index in later stages using this statement.

**Show the Index which we have created on *orders* table:**

`show indexes on orders;`



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> show indexes on orders;  
OK  
orders_index      orders      customer_id      trendytech_o  
rders_orders_index_ compact  
Time taken: 0.355 seconds, Fetched: 1 row(s)  
hive> █
```

**Show table display index table:**

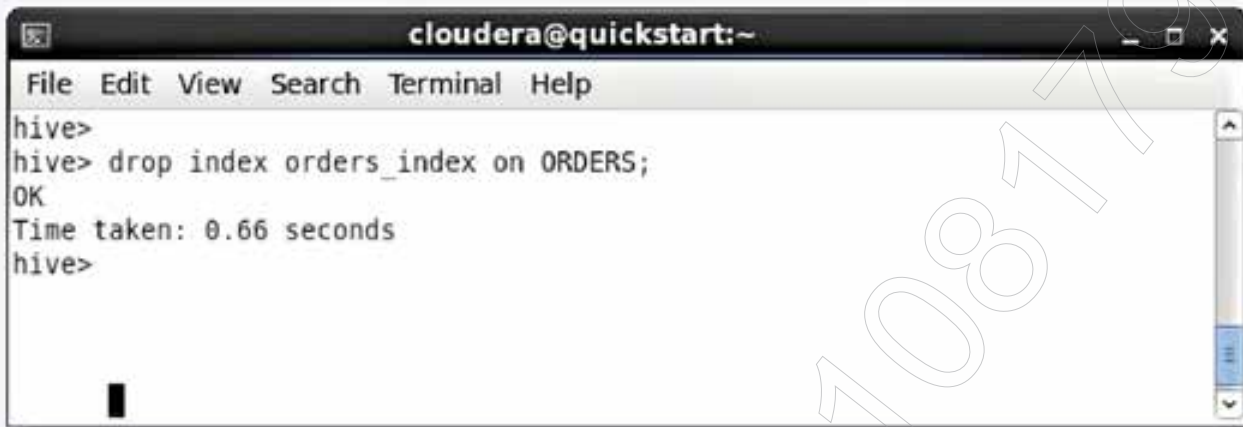
`show tables;`



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive> show tables;  
OK  
all_orders  
customer_purchases  
customers  
freshproducts  
mobilephones  
mobilephones_new  
orders  
orders_no_partition  
orders_no_partition1  
products  
trendytech_orders_orders_index_  
Time taken: 0.027 seconds, Fetched: 11 row(s)  
hive> █
```

## Drop an Index:

```
drop index orders_index on ORDERS;
```



```
cloudera@quickstart:~  
File Edit View Search Terminal Help  
hive>  
hive> drop index orders_index on ORDERS;  
OK  
Time taken: 0.66 seconds  
hive>
```





## 5 Star Google Rated Big Data Course

**LEARN FROM THE EXPERT**



9108179578

**Call for more details**



# Follow US

**Trainer** Mr. Sumit Mittal

**LinkedIn** <https://www.linkedin.com/in/bigdatabysumit/>

**Website** <https://trendytech.in/courses/big-data-online-training/>

**Phone** 9108179578

**Email** [trendytech.sumit@gmail.com](mailto:trendytech.sumit@gmail.com)

**Youtube** TrendyTech

**Twitter** @BigdataBySumit

**Instagram** bigdatabysumit

**Facebook** <https://www.facebook.com/trendytech.in/>

