ACADEMIC YEAR 2021-2022

BIGDATA LABORATORY

Report on,

Learning Activity II-Programming Assignment

Submitted by,

Karthik Senthil(1NT18IS201)

submitted to,

Mr. PRASHANTH B S,
Assistant Professor,
Department of Information Science and Engineering
Nitte Meenakshi Institute of Technology
Bangalore-064

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(An autonomous institution with A+ Grade by NAAC /UGC, Affiliated to Visvesvaraya Technological University, Belgaum, Approved by UGC/AICTE/Govt. of Karnataka)

Yelahanka, Bengaluru-560064

NITTE EDUCATION TRUST | NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

BD LAB

LA II

> Name: Karthik Senthil
> USN: 1NT18IS201
> Section: B
> Sem: 6
> Date: 09/06/21

Exercise-I

Create a dataset in excel as a .csv file and it should contain the following fields with at least 20 sample datasets in it.

Use the Hadoop MapReduce programming framework to come up with a Program which will take the data from this .csv file and computes the following.

1. Total number of employees who are eligible for the pay raise.
2. Total number of cumulative awards the company had this year.
3. How many total awards were obtained by the employee whose salary is 30000
4. Count the number of employees who had paid the Tax

Dataset :- employee.csv

Foden	76394	500000	3	YES	YES
Grealish	87404	200000	0	NO	YES
Saka	34398	30000	2	YES	YES
Rashford	92834	500000	3	YES	NO
Maguire	34973	300000	0	NO	NO
Calvert	89458	80000	0	NO	YES
Kane	28572	800000	6	YES	YES
Philips	23857	30000	0	NO	YES
Alexander	67584	200000	1	YES	NO
Reece	48928	30000	1	NO	YES
Chilwell	58958	200000	1	NO	NO
Shaw	83249	200000	2	YES	YES
Mount	23894	300000	2	NO	YES
Sterling	49859	500000	4	YES	NO
Pickford	39847	30000	2	NO	NO
Hendersoı	59724	100000	0	NO	NO
Trippier	68734	100000	0	YES	NO
Walker	93452	30000	2	YES	NO
Godfrey	79834	50000	0	NO	YES
Lingard	83975	100000	6	YES	YES

Queries

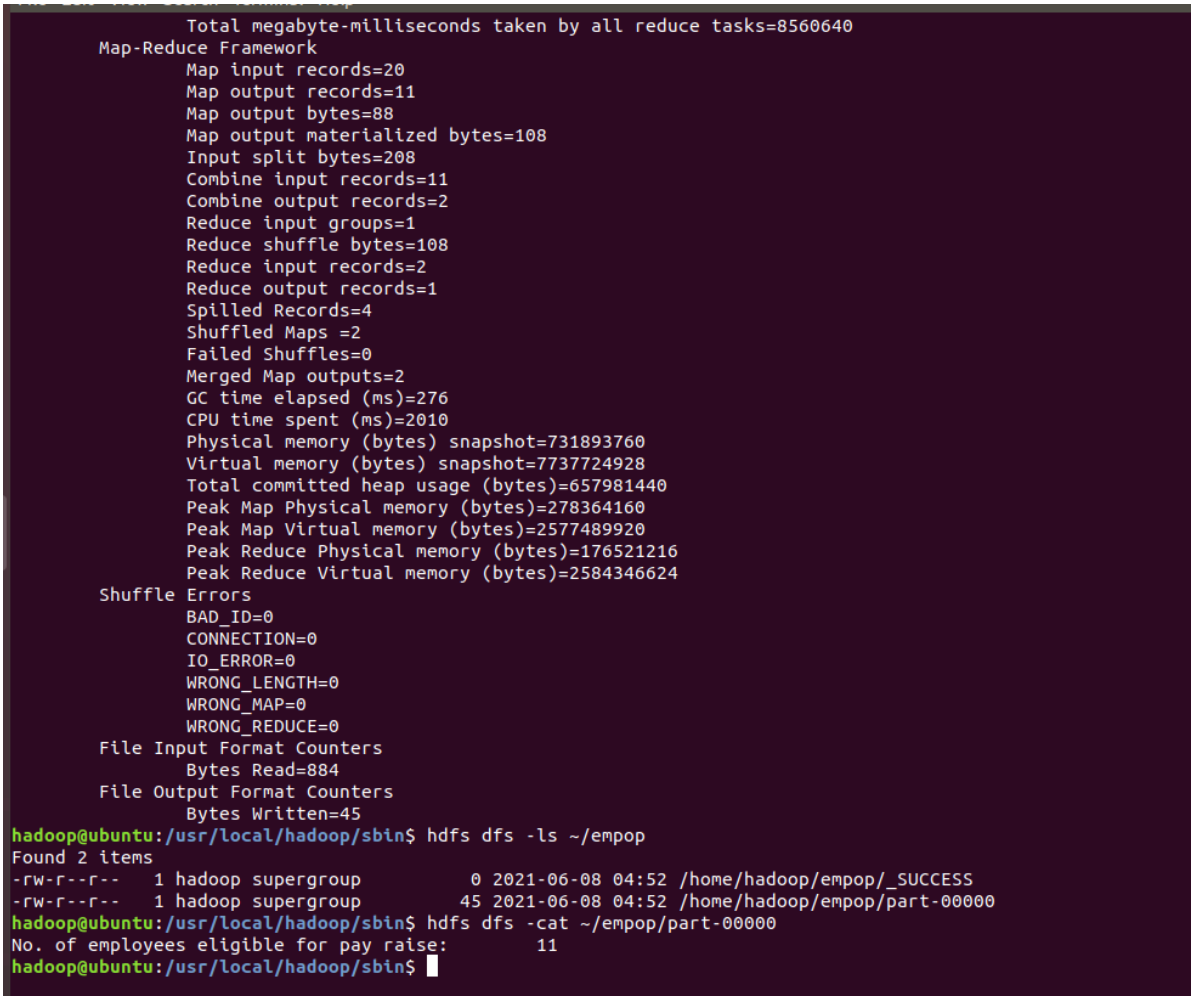
1. Total number of employees who are eligible for the pay raise.

Program :Pgm01.java

The map method runs for each line of the CSV input. Once we convert said line of input to a string we split it using "," as
the delimiter to get the values of that row as an array of strings.
In this case we need "pay_inc" column, which is 5th index in the array. We check if this is "YES" in the mapper, and if so, we collect the output and send it to the reducer.
For the reducer, we count the number of key value pairs the mapper has returned by looping through it with a count variable. We then use the key and return the value which is the count.

```
hadoop jar /home/hadoop/Desktop/emp_pgm01.jar ~/empip ~/empop
```

OUTPUT



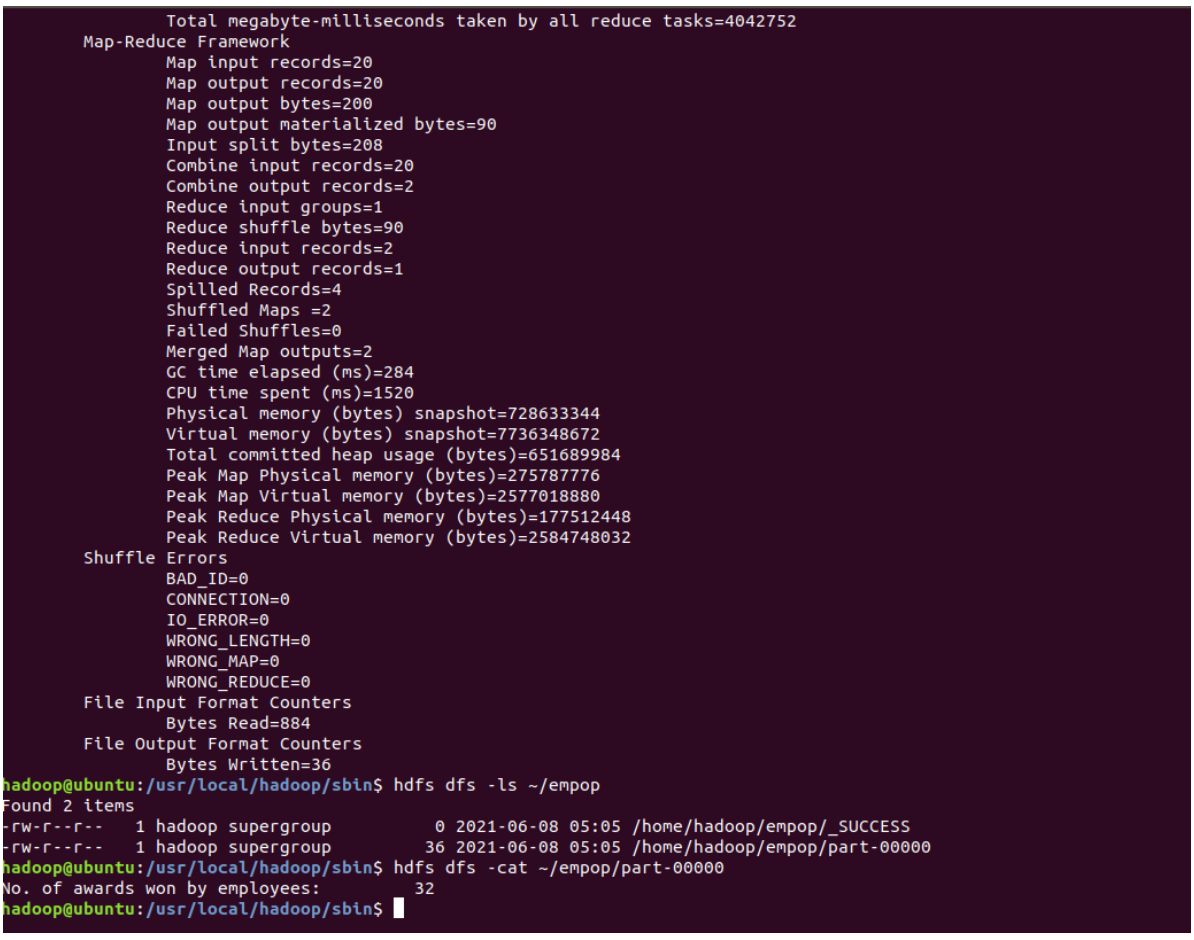
2. Total number of cumulative awards the company had this year

Program : Pgm02.java

The map method runs for each line of the CSV input. Once we convert said line of input to a string we split it using "," as
the delimiter to get the values of that row as an array of strings.
In this case we need "awards" column, which is 3rd index in the array. We convert the data type to IntWritable as that is what we use for the value in the key,value pair and send it to the reducer.
For the reducer, we add all the values in a count variable. We then use the key and return the value which is the count.

```
hadoop jar /home/hadoop/Desktop/emp_pgm02.jar ~/empip ~/empop
```

OUTPUT



3. How many total awards were obtained by the employee whose salary is 30000

Program: Pgm03.java

The map method runs for each line of the CSV input. Once we convert said line of input to a string we split it using "," as
the delimiter to get the values of that row as an array of strings.
In this case we need "awards" column, which is 3rd index in the array. We also need "salary" column, which is 2nd index in the array. We check if it is "30000" using if condition, We convert the data type of awards to IntWritable as that is what we use for the value in the key,value pair and send it to the reducer.
For the reducer, we add all the values in a count variable. We then use the key and return the value which is the count.

```
hadoop jar /home/hadoop/Desktop/emp_pgm03.jar ~/empip ~/empop
```

OUTPUT

```
                Total megabyte-milliseconds taken by all reduce tasks=2786304
        Map-Reduce Framework
                Map input records=20
                Map output records=5
                Map output bytes=50
                Map output materialized bytes=120
                Input split bytes=208
                Combine input records=5
                Combine output records=2
                Reduce input groups=1
                Reduce shuffle bytes=120
                Reduce input records=2
                Reduce output records=1
                Spilled Records=4
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=297
                CPU time spent (ms)=1490
                Physical memory (bytes) snapshot=737398784
                Virtual memory (bytes) snapshot=7740305408
                Total committed heap usage (bytes)=649068544
                Peak Map Physical memory (bytes)=280379392
                Peak Map Virtual memory (bytes)=2577727488
                Peak Reduce Physical memory (bytes)=177729536
                Peak Reduce Virtual memory (bytes)=2585165824
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=882
        File Output Format Counters
                Bytes Written=50
hadoop@ubuntu:/usr/local/hadoop/sbin$ hdfs dfs -ls ~/empop
Found 2 items
-rw-r--r--   1 hadoop supergroup          0 2021-06-08 05:27 /home/hadoop/empop/_SUCCESS
-rw-r--r--   1 hadoop supergroup         50 2021-06-08 05:27 /home/hadoop/empop/part-00000
hadoop@ubuntu:/usr/local/hadoop/sbin$ hdfs dfs -cat ~/empop/part-00000
No. of awards won by employees who earn 30000:  7
hadoop@ubuntu:/usr/local/hadoop/sbin$
```

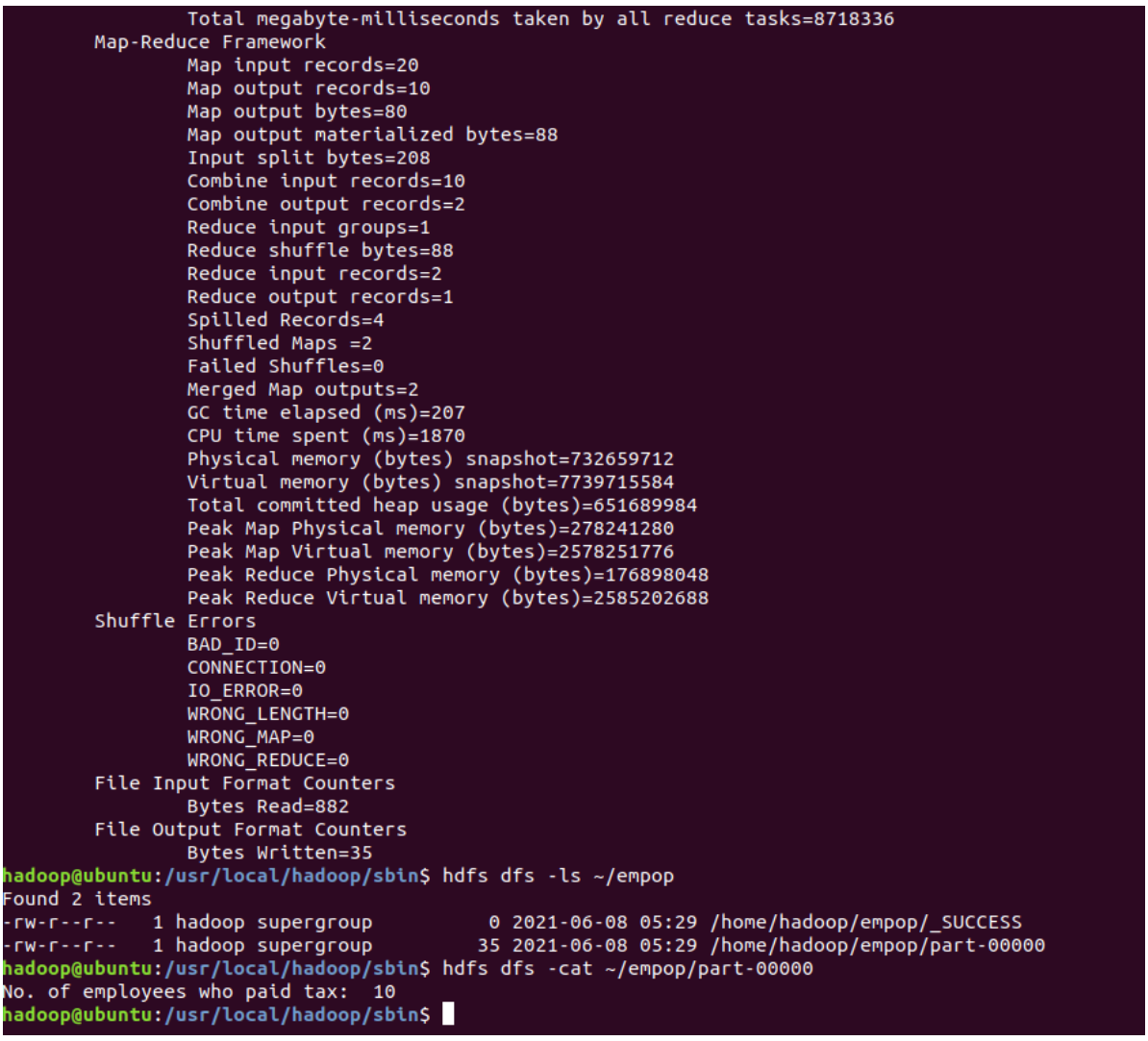
4. Count the number of employees who had paid the Tax

Program : Pgm04.java

The map method runs for each line of the CSV input. Once we convert said line of input to a string we split it using "," as
the delimiter to get the values of that row as an array of strings.
In this case we need "tax_paid" column, which is 4th index in the array. We check if this is "YES" in the mapper, and if so, we collect the output and send it to the reducer.
For the reducer, we count the number of key value pairs the mapper has returned by looping through it with a count variable. We then use the key and return the value which is the count.

```
hadoop jar /home/hadoop/Desktop/emp_pgm04.jar ~/empip ~/empop
```

OUTPUT



Exercise-II

Use the above dataset in .csv file and create a database called as EmployeeDB. Create a table under the database called as Employee using HIVEQL. The table fields are same, that is,

Name SSN Salary Awards Tax paid Eligible for Pay raise
Ajay 63300 30000 2 YES YES

Use the HiveQL language to perform the following Query based Map-reduce operations,

1. Insert 5 records using INSERT command.
2. Demonstrate the Alter command for the following cases,
   a. Rename the table name to “Emp”.
   b. Rename the column name “Eligible for Pay raise” to“Eligibility”.
3. Count the number of Employee who are eligible for pay raise who had paid the tax.
4. Extract all the users ordered by the Name who had paid the tax but are not eligible for pay raise.
5. Create separate view containing “SSN and Salary” and call the view as sal_ssn_view.
6. Display (name, eligibility) fields grouped by the SSN.
7. Display the (Name, SSN) of employees whose salary is >40000 but < 48000.
8. Create Another table called orders with the following fields (custssn = SSN in the Employee) and perform the following joins (outer,left outer, right outer) over custssn

Dataset : emp.csv

Foden	76394	500000	3	YES	YES
Grealish	87404	200000	0	NO	YES
Saka	34398	30000	2	YES	YES
Rashford	92834	500000	3	YES	NO
Maguire	34973	300000	0	NO	NO
Calvert	89458	80000	0	NO	YES
Kane	28572	800000	6	YES	YES
Philips	23857	30000	0	NO	YES
Alexander	67584	200000	1	YES	NO
Reece	48928	30000	1	NO	YES
Chilwell	58958	200000	1	NO	NO
Shaw	83249	200000	2	YES	YES
Mount	23894	300000	2	NO	YES
Sterling	49859	500000	4	YES	NO
Pickford	39847	30000	2	NO	NO
Hendersoı	59724	100000	0	NO	NO
Trippier	68734	100000	0	YES	NO
Walker	93452	30000	2	YES	NO
Godfrey	79834	50000	0	NO	YES
Lingard	83975	100000	6	YES	YES

Procedure

1. Create a DB named employeedb

```
create database employeedb;
use employee;
```

```
hive> create database EmployeeDB;
OK
Time taken: 0.505 seconds
hive> show databases;
OK
default
employeedb
salesdb
Time taken: 0.074 seconds, Fetched: 3 row(s)
hive> use employeedb
    > ;
OK
Time taken: 0.073 seconds
hive>
```

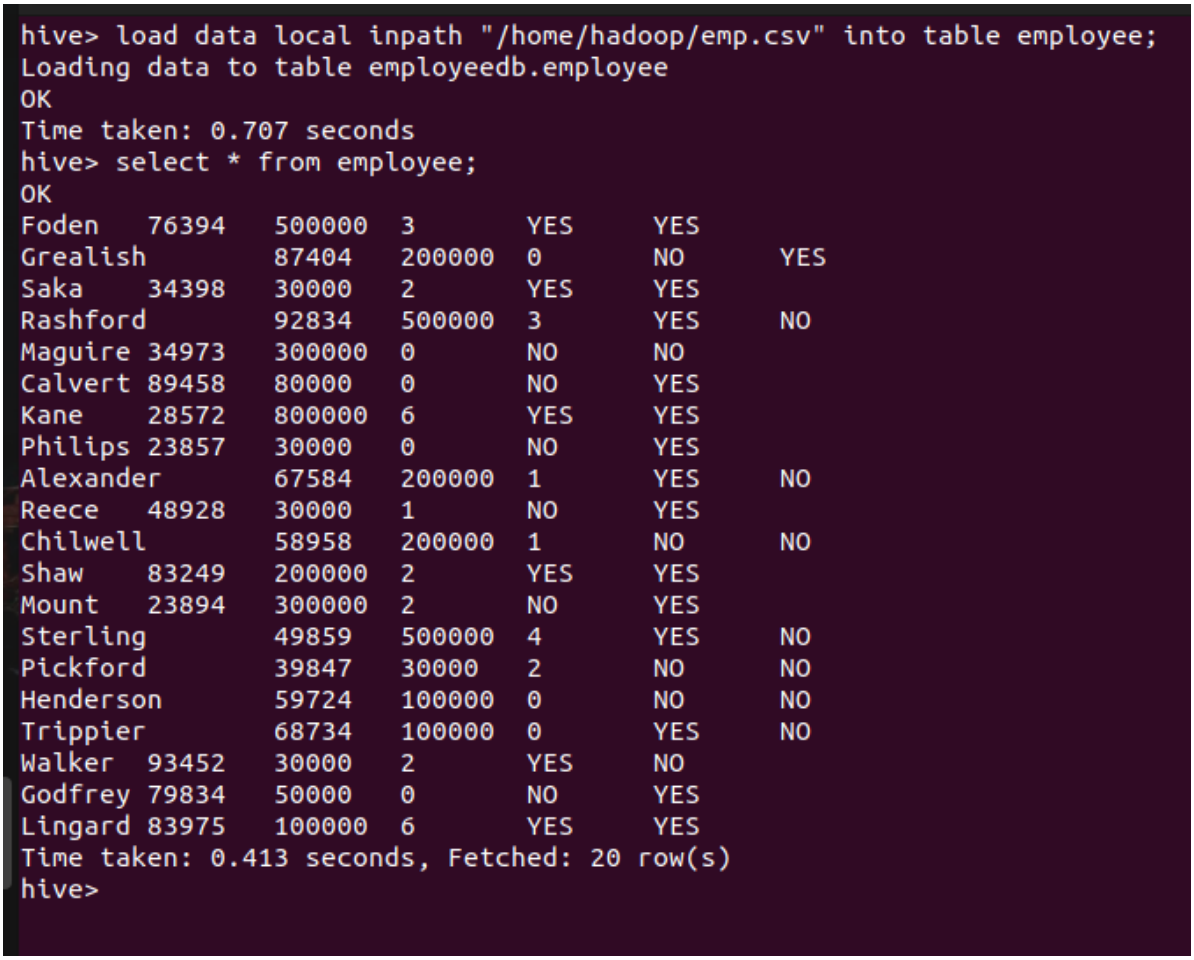
2. Create table employee

```
create table employee(Name string, SSN string, Salary int, Awards int, tax_paid
string, pay_inc string)
>row format delimited
>fields terminated by ",";
```

```
hive> create table employee(name string, ssn string, salary int, awards int, tax_paid string, pay_inc string)
    > row format delimited
    > fields terminated by ",";
OK
Time taken: 1.831 seconds
hive>
```

3. Insert Data from CSV file

```
LOAD DATA local INPATH "/home/hadoop/Desktop/emp.csv" into table employee;
```



```
hive> load data local inpath "/home/hadoop/emp.csv" into table employee;
Loading data to table employeedb.employee
OK
Time taken: 0.707 seconds
hive> select * from employee;
OK
Foden    76394    500000   3        YES      YES
Grealish         87404    200000   0        NO       YES
Saka     34398    30000    2        YES      YES
Rashford         92834    500000   3        YES      NO
Maguire  34973    300000   0        NO       NO
Calvert  89458    80000    0        NO       YES
Kane     28572    800000   6        YES      YES
Philips  23857    30000    0        NO       YES
Alexander        67584    200000   1        YES      NO
Reece    48928    30000    1        NO       YES
Chilwell         58958    200000   1        NO       NO
Shaw     83249    200000   2        YES      YES
Mount    23894    300000   2        NO       YES
Sterling         49859    500000   4        YES      NO
Pickford         39847    30000    2        NO       NO
Henderson        59724    100000   0        NO       NO
Trippier         68734    100000   0        YES      NO
Walker   93452    30000    2        YES      NO
Godfrey  79834    50000    0        NO       YES
Lingard  83975    100000   6        YES      YES
Time taken: 0.413 seconds, Fetched: 20 row(s)
hive>
```

Queries

1. Insert 5 records using INSERT command.

```
insert into employee values
> ("Smith-Rowe","45676",30000,1,"YES","YES"),
> ("Sancho","56784",300000,3,"NO","YES"),
> ("Watkins","54747",50000,0,"NO","NO"),
> ("Dean","87348",100000,0,"YES","NO"),
> ("Bellingham","78393",80000,1,"NO","YES");
```

OUTPUT

```
hive> insert into employee values
    > ("Smith-Rowe","45676",30000,1,"YES","YES"),
    > ("Sancho","56784",300000,3,"NO","YES"),
    > ("Watkins","54747",50000,0,"NO","NO"),
    > ("Dean","87348",100000,0,"YES","NO"),
    > ("Bellingham","78393",80000,1,"NO","YES");
Query ID = hadoop_20210609123417_1498cbe6-1148-4156-b770-f8bd230fd2be
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1623221371091_0002, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0002/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-09 12:34:32,608 Stage-1 map = 0%,  reduce = 0%
2021-06-09 12:34:50,229 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.8 sec
2021-06-09 12:34:58,591 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.8 sec
MapReduce Total cumulative CPU time: 7 seconds 800 msec
Ended Job = job_1623221371091_0002
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9000/user/hive/warehouse/employeedb.db/employee/.hive-staging_hive_2021-06-09_12-34-17_757_2415168219059067962-1/-ext-10000
Loading data to table employeedb.employee
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.8 sec   HDFS Read: 20849 HDFS Write: 616 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 800 msec
OK
Time taken: 43.702 seconds
hive>
```

```
hive> select * from employee;
OK
Smith-Rowe      45676   30000   1       YES     YES
Sancho  56784   300000  3       NO      YES
Watkins 54747   50000   0       NO      NO
Dean    87348   100000  0       YES     NO
Bellingham      78393   80000   1       NO      YES
Foden   76394   500000  3       YES     YES
Grealish        87404   200000  0       NO      YES
Saka    34398   30000   2       YES     YES
Rashford        92834   500000  3       YES     NO
Maguire 34973   300000  0       NO      NO
Calvert 89458   80000   0       NO      YES
Kane    28572   800000  6       YES     YES
Philips 23857   30000   0       NO      YES
Alexander       67584   200000  1       YES     NO
Reece   48928   30000   1       NO      YES
Chilwell        58958   200000  1       NO      NO
Shaw    83249   200000  2       YES     YES
Mount   23894   300000  2       NO      YES
Sterling        49859   500000  4       YES     NO
Pickford        39847   30000   2       NO      NO
Henderson       59724   100000  0       NO      NO
Trippier        68734   100000  0       YES     NO
Walker  93452   30000   2       YES     NO
Godfrey 79834   50000   0       NO      YES
Lingard 83975   100000  6       YES     YES
Time taken: 0.412 seconds, Fetched: 25 row(s)
hive>
```

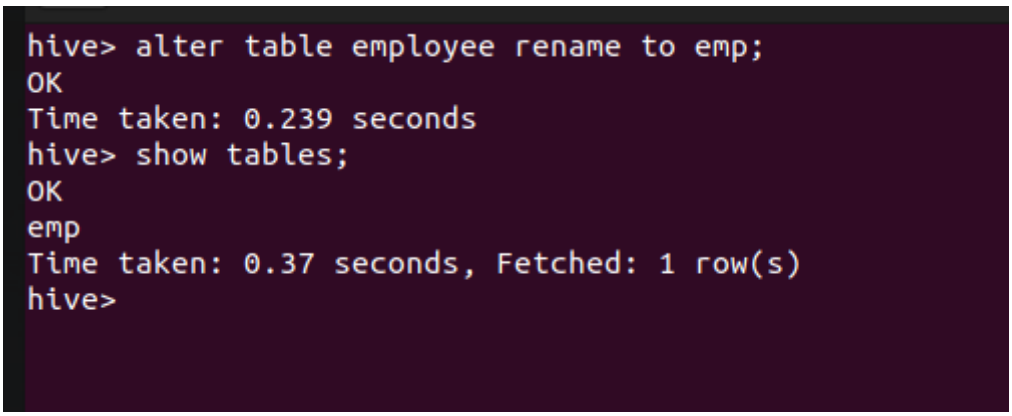
2. Demonstrate the Alter command for the following cases

a. Rename the table name to “emp”.

b. Rename the column name “Eligible for Pay raise” to“Eligibility”.

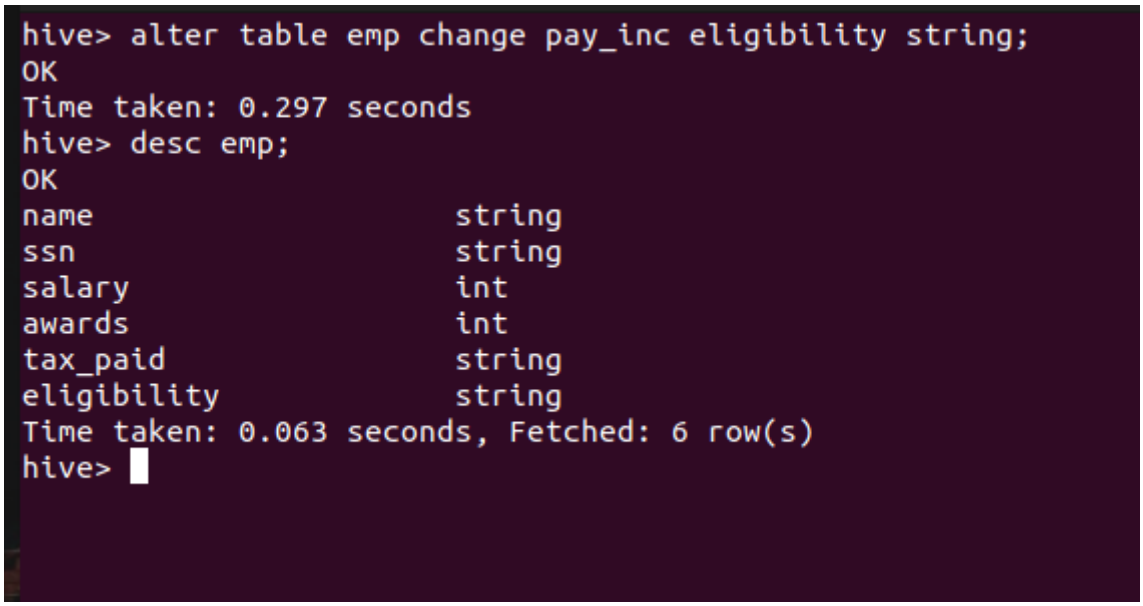
```
ALTER TABLE employee RENAME TO emp;
```

OUTPUT




```
ALTER TABLE emp CHANGE pay_inc eligibility string;
```

OUTPUT



3. Count the number of Employee who are eligible for pay raise who had paid the tax.

```
select count(*) from emp
> where tax_paid="YES" and eligibility="YES";
```

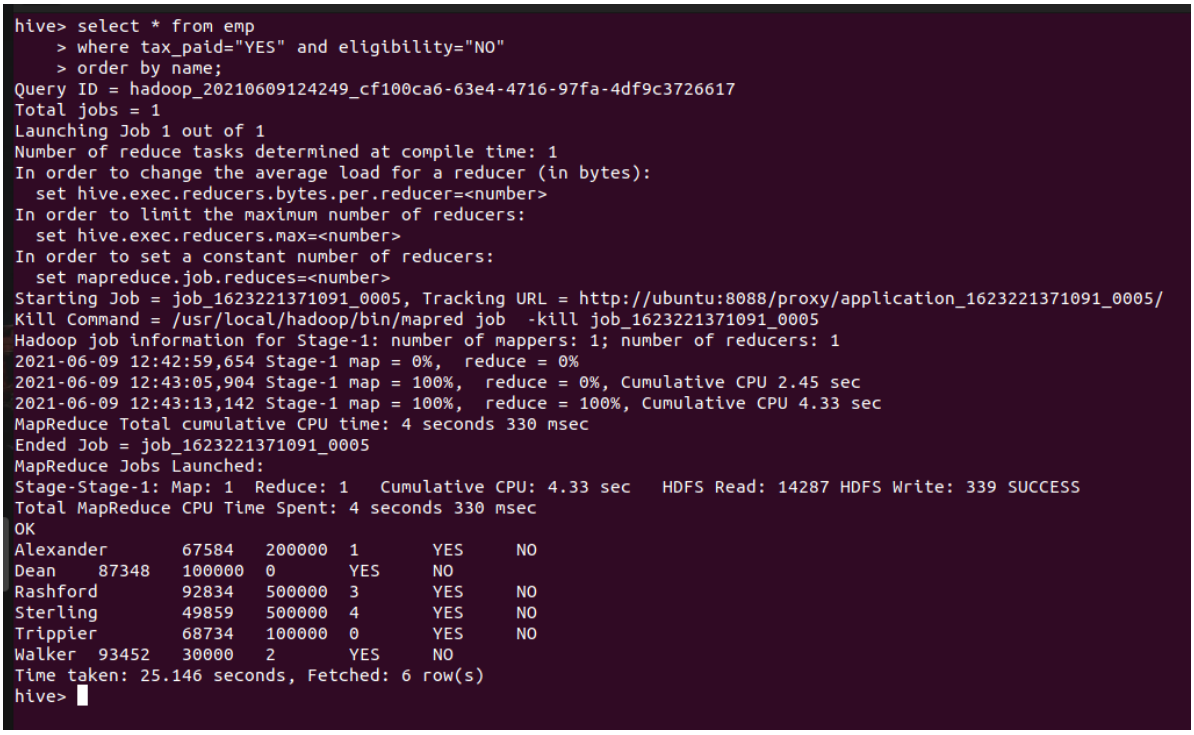
OUTPUT

```
hive> select count(*) from emp
    > where tax_paid="YES" and eligibility="YES";
Query ID = hadoop_20210609123908_371d96fa-e3d4-4eb6-849e-c466b2a6a0f6
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1623221371091_0003, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0003/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-09 12:39:25,877 Stage-1 map = 0%,  reduce = 0%
2021-06-09 12:39:55,835 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 12.83 sec
2021-06-09 12:40:07,286 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 15.27 sec
MapReduce Total cumulative CPU time: 15 seconds 270 msec
Ended Job = job_1623221371091_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 15.27 sec   HDFS Read: 15693 HDFS Write: 101 SUCCESS
Total MapReduce CPU Time Spent: 15 seconds 270 msec
OK
6
Time taken: 60.9 seconds, Fetched: 1 row(s)
hive>
```

4. Extract all the users ordered by the Name who had paid the tax but are not eligible for pay raise.

```
select * from emp
> where taxpaid="YES" and eligibility="NO"
> order by name;
```

OUTPUT

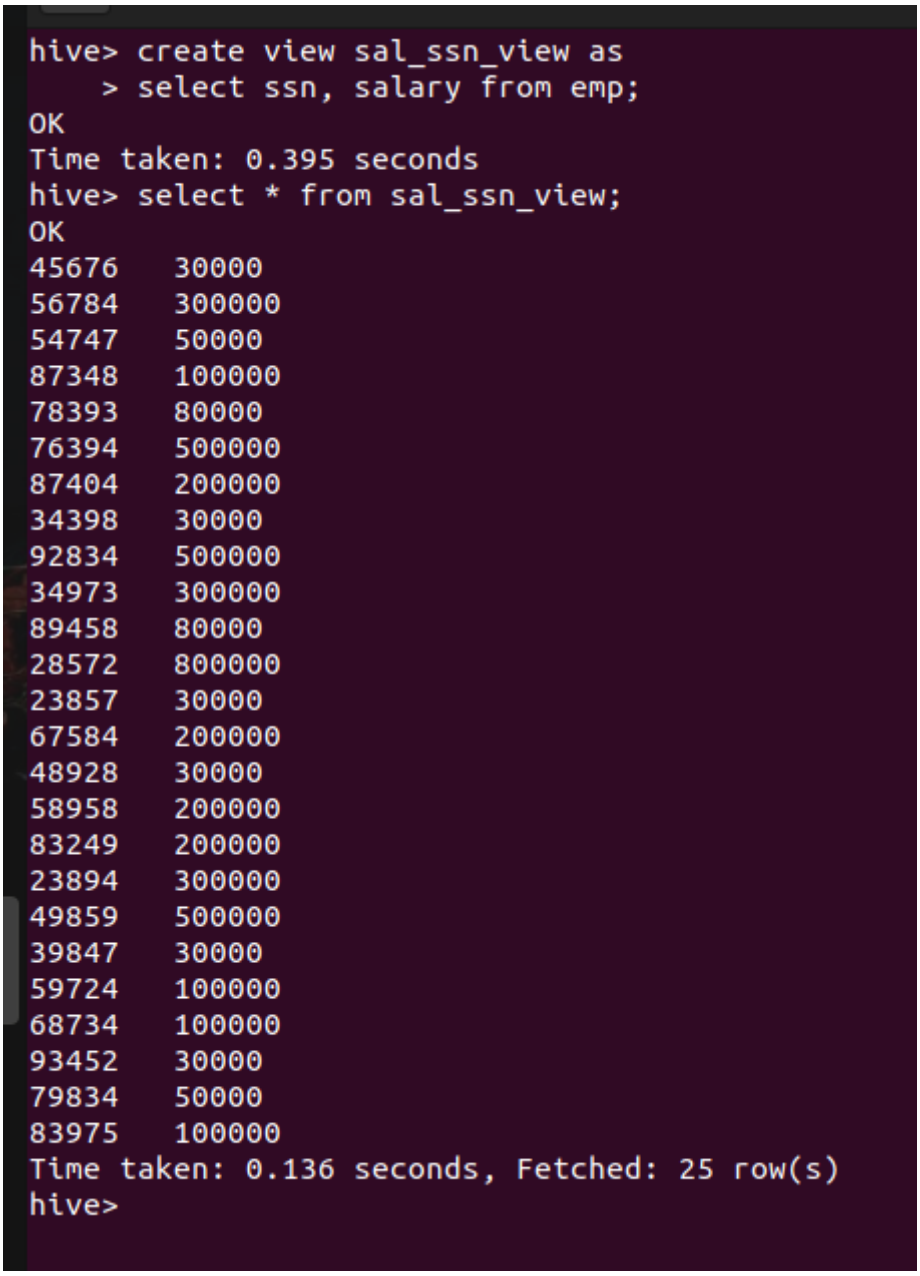


5. Create separate view containing “SSN and Salary” and call the view as sal_ssn_view

```
create view sal_ssn_view as
> select ssn,salary from emp;

select * from sal_ssn_view;
```

OUTPUT

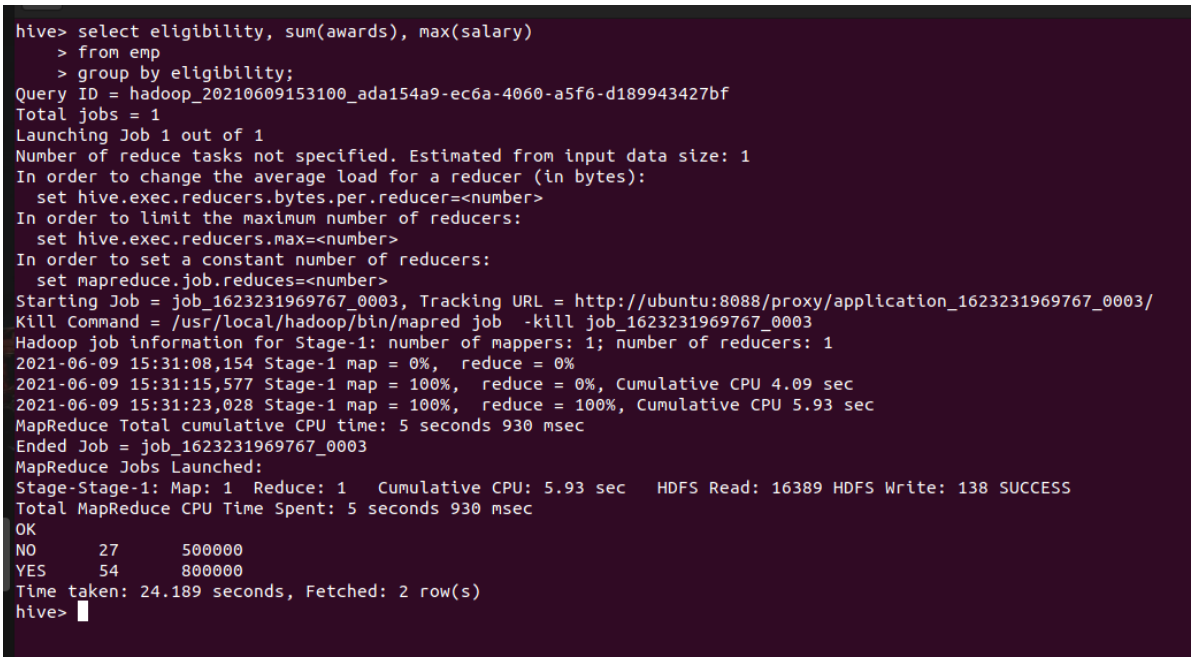


6. Display (name, eligibility) fields grouped by the SSN.

Note: group by clause requires aggregation so utliising sum() and max() functions, grouping by eligibility as opposed to ssn since ssn is unique and therefore has no effect on grouping

```
hive > select eligibility, sum(awards), max(salary)
> from emp
> group eligiblity;
```

OUTPUT

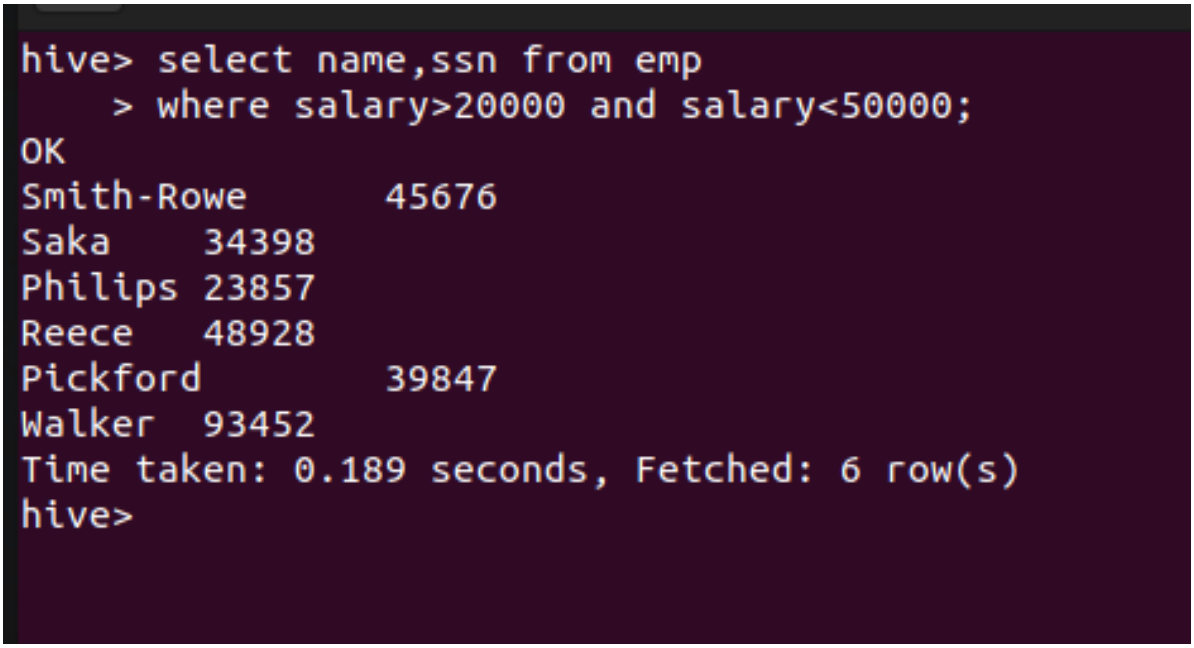


7. Display the (Name, SSN) of employees whose salary is >40000 but < 48000.

Note: due to lack of data entires with salary between 40000 and 48000 as salary, altering the range to 20000 to 50000

```
select name,ssn from emp
> where salary > 20000 and salary< 50000;
```

OUTPUT



8. Create Another table called orders with the following fields (custssn = SSN in the Employee) and perform the following joins (outer,left outer, right outer) over custssn

```
hive> create table orders(orderid int, custssn string, amount int)
    > row format delimited
    > fields terminated by ",";
OK
Time taken: 0.846 seconds
hive> insert into orders values
    > (10,"45676",20000),
    > (20,"87348",35000),
    > (30,"28572",400000),
    > (40,"34398",65000),
    > (50,"58958",89000);
Query ID = hadoop_20210609134612_bf7c7887-b2a0-405a-a6aa-474a3ec3c5f6
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1623221371091_0008, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0008/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-09 13:46:44,048 Stage-1 map = 0%,  reduce = 0%
2021-06-09 13:47:27,958 Stage-1 map = 67%,  reduce = 0%, Cumulative CPU 14.44 sec
2021-06-09 13:47:37,354 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 14.7 sec
2021-06-09 13:47:44,566 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 16.73 sec
MapReduce Total cumulative CPU time: 16 seconds 730 msec
Ended Job = job_1623221371091_0008
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9000/user/hive/warehouse/employeedb.db/orders/.hive-staging_hive_2021-06-09_13-46-12_251_5873766959526653639-1/-ext-10000
Loading data to table employeedb.orders
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 16.73 sec   HDFS Read: 17114 HDFS Write: 422 SUCCESS
Total MapReduce CPU Time Spent: 16 seconds 730 msec
OK
Time taken: 97.186 seconds
hive>
    >
```

```
hive> insert into orders values
    > (10,"45676",20000),
    > (20,"87348",35000),
    > (30,"28572",400000),
    > (40,"34398",65000),
    > (50,"58958",89000);
Query ID = hadoop_20210609134612_bf7c7887-b2a0-405a-a6aa-474a3ec3c5f6
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1623221371091_0008, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0008/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-06-09 13:46:44,048 Stage-1 map = 0%,  reduce = 0%
2021-06-09 13:47:27,958 Stage-1 map = 67%,  reduce = 0%, Cumulative CPU 14.44 sec
2021-06-09 13:47:37,354 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 14.7 sec
2021-06-09 13:47:44,566 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 16.73 sec
MapReduce Total cumulative CPU time: 16 seconds 730 msec
Ended Job = job_1623221371091_0008
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9000/user/hive/warehouse/employeedb.db/orders/.hive-staging_hive_2021-06-09_13-46-12_251_5873766959526653639-1/-ext-10000
Loading data to table employeedb.orders
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 16.73 sec   HDFS Read: 17114 HDFS Write: 422 SUCCESS
Total MapReduce CPU Time Spent: 16 seconds 730 msec
OK
Time taken: 97.186 seconds
hive>
    > select * from orders;
OK
10      45676   20000
20      87348   35000
30      28572   400000
40      34398   65000
50      58958   89000
Time taken: 0.554 seconds, Fetched: 5 row(s)
hive>
```

FULL OUTER JOIN

```
select o.orderid, e.ssn, e.name, o.amount
> from emp e
> FULL OUTER JOIN orders o
> ON (e.ssn = o.custssn);
```

OUTPUT

```
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1623221371091_0009, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0009/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0009
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2021-06-09 13:49:26,456 Stage-1 map = 0%,  reduce = 0%
2021-06-09 13:49:49,213 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 7.13 sec
2021-06-09 13:50:15,668 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 19.23 sec
2021-06-09 13:50:29,641 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 20.88 sec
MapReduce Total cumulative CPU time: 20 seconds 880 msec
Ended Job = job_1623221371091_0009
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 1   Cumulative CPU: 20.88 sec   HDFS Read: 17691 HDFS Write: 899 SUCCESS
Total MapReduce CPU Time Spent: 20 seconds 880 msec
OK
NULL    23857   Philips NULL
NULL    23894   Mount   NULL
30      28572   Kane    400000
40      34398   Saka    65000
NULL    34973   Maguire NULL
NULL    39847   Pickford        NULL
10      45676   Smith-Rowe      20000
NULL    48928   Reece   NULL
NULL    49859   Sterling        NULL
NULL    54747   Watkins NULL
NULL    56784   Sancho  NULL
50      58958   Chilwell        89000
NULL    59724   Henderson       NULL
NULL    67584   Alexander       NULL
NULL    68734   Trippier        NULL
NULL    76394   Foden   NULL
NULL    78393   Bellingham      NULL
NULL    79834   Godfrey NULL
NULL    83249   Shaw    NULL
NULL    83975   Lingard NULL
20      87348   Dean    35000
NULL    87404   Grealish        NULL
NULL    89458   Calvert NULL
NULL    92834   Rashford        NULL
NULL    93452   Walker  NULL
Time taken: 94.532 seconds, Fetched: 25 row(s)
hive>
```

LEFT OUTER JOIN

```
select o.orderid, e.ssn, e.name, o.amount
> from emp e
> LEFT OUTER JOIN orders o
> ON (e.ssn = o.custssn);
```

OUTPUT

```
    > on (e.ssn = o.custssn);
Query ID = hadoop_20210609135119_83b0d8bf-30c4-4569-82a3-b46925fdcf76
Total jobs = 1
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1623221371091_0010, Tracking URL = http://ubuntu:8088/proxy/application_1623221371091_0010/
Kill Command = /usr/local/hadoop/bin/mapred job  -kill job_1623221371091_0010
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 0
2021-06-09 13:52:02,469 Stage-3 map = 0%,  reduce = 0%
2021-06-09 13:52:09,763 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 2.55 sec
MapReduce Total cumulative CPU time: 2 seconds 550 msec
Ended Job = job_1623221371091_0010
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1   Cumulative CPU: 2.55 sec   HDFS Read: 10183 HDFS Write: 899 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 550 msec
OK
10      45676   Smith-Rowe      20000
NULL    56784   Sancho  NULL
NULL    54747   Watkins NULL
20      87348   Dean    35000
NULL    78393   Bellingham      NULL
NULL    76394   Foden   NULL
NULL    87404   Grealish        NULL
40      34398   Saka    65000
NULL    92834   Rashford        NULL
NULL    34973   Maguire NULL
NULL    89458   Calvert NULL
30      28572   Kane    400000
NULL    23857   Philips NULL
NULL    67584   Alexander       NULL
NULL    48928   Reece   NULL
50      58958   Chilwell        89000
NULL    83249   Shaw    NULL
NULL    23894   Mount   NULL
NULL    49859   Sterling        NULL
NULL    39847   Pickford        NULL
NULL    59724   Henderson       NULL
NULL    68734   Trippier        NULL
NULL    93452   Walker  NULL
NULL    79834   Godfrey NULL
NULL    83975   Lingard NULL
Time taken: 58.49 seconds, Fetched: 25 row(s)
hive>
```

RIGHT OUTER JOIN

```
select o.orderid, e.ssn, e.name, o.amount
> from emp e
> RIGHT OUTER JOIN orders o
> ON (e.ssn = o.custssn);
```

OUTPUT