Login :

ssh -l bigdata01 linux.scudc.scu.edu

cd $HIVE\_HOME

Data copy from local to ssh directory

scp -r data [bigdata01@linux.scudc.scu.edu](mailto:bigdata01@linux.scudc.scu.edu):/home/bigdata01/

setup cdh-5.12

cd $HIVE\_HOME

cd bin/

hive

<https://pdfs.semanticscholar.org/ae03/35f556cb67692d35ec1262155c3b0861104f.pdf>

Check what is in our hdfs

hdfs dfs -ls /user/bigdata01

Copy data from local to hdfs:

hdfs dfs -mkdir visdata

hdfs dfs -copyFromLocal testData.csv /user/bigdata01/visdata/

Check if csv file copied:

hdfs dfs -ls /user/bigdata01/visdata

Table creation command :

CREATE EXTERNAL TABLE IF NOT EXISTS test1(EmployeeID INT,FirstName STRING)

COMMENT 'Employee details'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LINES TERMINATED BY '\n'

STORED AS TEXTFILE

LOCATION '/user/bigdata01/visdata/';

Check data exists in table :

select \* from test1;

Create our data tables

* Move data from your machine to linux bigdata01

scp -r data [bigdata01@linux.scudc.scu.edu](mailto:bigdata01@linux.scudc.scu.edu):/home/bigdata01/

* Remove header from all csv files

sed -i 1d data.csv

sed -i 1d free-zipcode-database.csv

sed -i 1d movie.csv

sed -i 1d occupation.csv

sed -i 1d user.csv

sed -i 1d zipcode.csv

* Create directory for each csv file in hdfs

hdfs dfs -mkdir dataDir

hdfs dfs -mkdir freeZipcodeDir

hdfs dfs -mkdir movieDir

hdfs dfs -mkdir occupationDir

hdfs dfs -mkdir userDir

hdfs dfs -mkdir zipcodeDir

* Check all new directories in hdfs

hdfs dfs -ls /user/bigdata01

* Copy data from local to hdfs

cd data

hdfs dfs -copyFromLocal data.csv /user/bigdata01/dataDir/

hdfs dfs -copyFromLocal free-zipcode-database.csv /user/bigdata01/freeZipcodeDir/

hdfs dfs -copyFromLocal movie.csv /user/bigdata01/movieDir/

hdfs dfs -copyFromLocal occupation.csv /user/bigdata01/occupationDir/

hdfs dfs -copyFromLocal user.csv /user/bigdata01/userDir/

hdfs dfs -copyFromLocal zipcode.csv /user/bigdata01/zipcodeDir/

* Check all files are in hdfs

hdfs dfs -ls /user/bigdata01/dataDir

hdfs dfs -ls /user/bigdata01/freeZipcodeDir

hdfs dfs -ls /user/bigdata01/movieDir

hdfs dfs -ls /user/bigdata01/occupationDir

hdfs dfs -ls /user/bigdata01/userDir

hdfs dfs -ls /user/bigdata01/zipcodeDir

* Create tables hql

**Data\_T1**

● Userid NUMBER  
● Itemid NUMBER  
● Rating NUMBER  
● Timestamp NUMBER

CREATE EXTERNAL TABLE IF NOT EXISTS Data\_T1(

userid INT,

itemid INT,

rating INT,

timestamp INT)  
COMMENT 'Data details'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE  
LOCATION '/user/bigdata01/dataDir/';

**Movie\_T1**  
● movieid (theta join with item id) NUMBER  
● Title VARCHAR2(255)  
● Release\_date VARCHAR2(30)  
● Unknown NUMBER  
● Action NUMBER  
● Adventure NUMBER  
● Animation NUMBER  
● Childrens NUMBER  
● Comedy NUMBER  
● Crime NUMBER  
● Documentary NUMBER  
● Drama NUMBER  
● Fantasy NUMBER  
● Film\_Noir NUMBER  
● Horror NUMBER  
● Musical NUMBER  
● Mystery NUMBER  
● Romance NUMBER  
● Sci\_Fi NUMBER  
● Thriller NUMBER  
● War NUMBER  
● Western NUMBER

CREATE EXTERNAL TABLE IF NOT EXISTS Movie\_T1(

movieid INT,  
title STRING,  
release\_date STRING,  
unknown INT,  
action INT,  
adventure INT,  
animation INT,  
childrens INT,  
comedy INT,  
crime INT,  
documentary INT,  
drama INT,  
fantasy INT,  
film\_Noir INT,  
horror INT,  
musical INT,  
mystery INT,  
romance INT,  
sci\_fi INT,  
thriller INT,  
war INT,  
western INT)  
COMMENT 'Movie details'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE  
LOCATION '/user/bigdata01/movieDir/';

**Users**● Userid NUMBER  
● Age NUMBER  
● Gender VARCHAR2(1)  
● Occupationid NUMBER  
● Zipcode VARCHAR2(30)

CREATE EXTERNAL TABLE IF NOT EXISTS Users\_T1(  
userid INT,  
age INT,  
gender STRING,  
occupationid INT,  
zipcode STRING)  
COMMENT 'User details'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE  
LOCATION '/user/bigdata01/userDir/';

**Occupation**  
● Occupationid NUMBER  
● Occupation VARCHAR2(30)

CREATE EXTERNAL TABLE IF NOT EXISTS Occupation\_T1(  
occupationid INT,  
occupation STRING)  
COMMENT 'Occupation details'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE  
LOCATION '/user/bigdata01/occupationDir/';

Zipcode  
● Zipcode NUMBER  
● City VARCHAR2(30)  
● State VARCHAR2(30)

CREATE EXTERNAL TABLE IF NOT EXISTS Zipcode\_T1(  
zipcode INT,  
city STRING,

state STRING)  
COMMENT 'Zipcode details'  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
STORED AS TEXTFILE  
LOCATION '/user/bigdata01/zipcodeDir/';

1. **Select movie where comedy is 1**

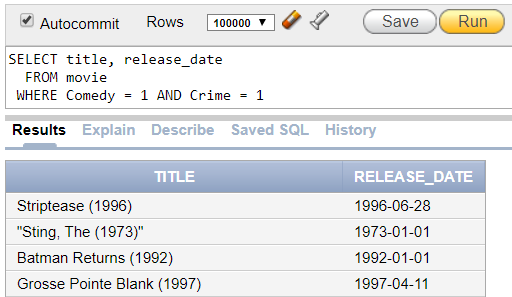
SELECT title, release\_date

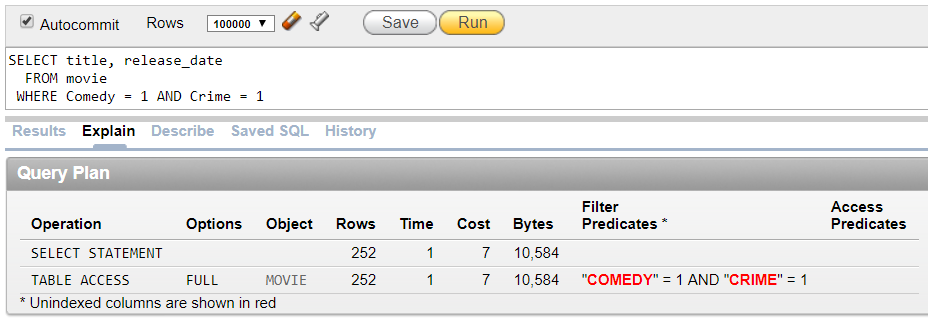
FROM Movie\_T1

WHERE Comedy = 1 and Crime=1;

Q1 Output : Time taken: 29.671 seconds, Fetched: 437 row(s)

Oracle Rows fetched: 0.29 seconds, 505



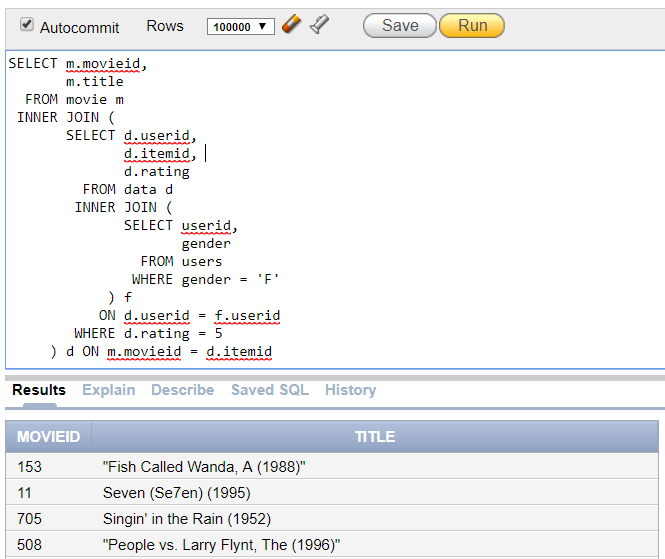


1. **Most popular movie among women**

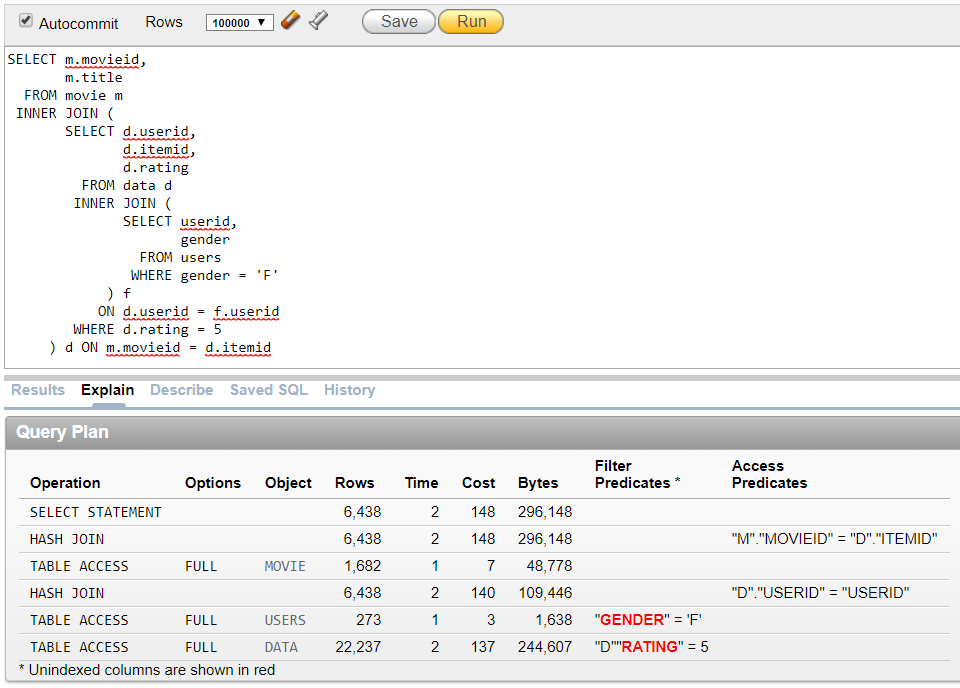
SELECT m.movieid,  
m.title FROM Movie\_T1 m  
INNER JOIN ( SELECT d.userid,  
d.itemid,  
d.rating FROM Data\_T1 d INNER JOIN (  
SELECT userid, gender FROM Users\_T1  
WHERE gender = 'F' )f  
ON d.userid = f.userid WHERE d.rating = 5  
)d  
ON m.movieid = d.itemid​

Time taken: 23.028 seconds, Fetched: 5975 row(s)

Oracle Rows fetched: 5975 rows, 0.09 seconds

****

****

****

1. **Average rating for each gender**

SELECT u.gender,

avg(d.rating) AS avg\_rating

FROM Users\_T1 u

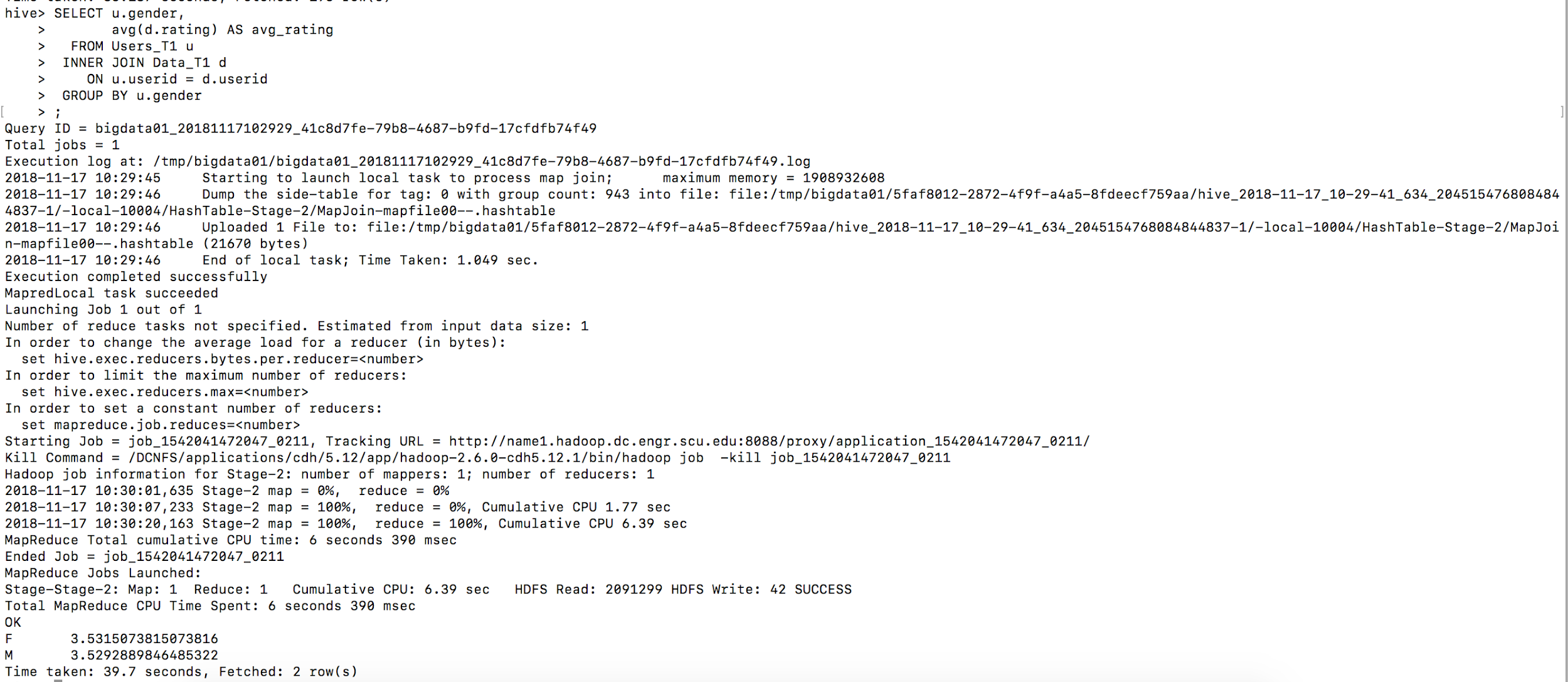
INNER JOIN Data\_T1 d

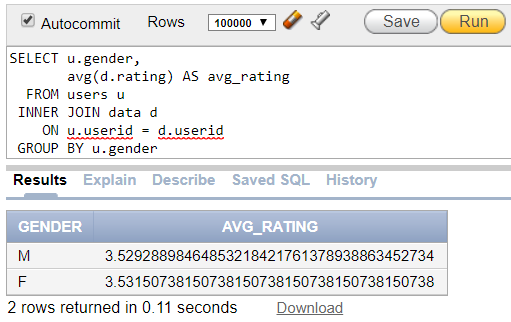
ON u.userid = d.userid

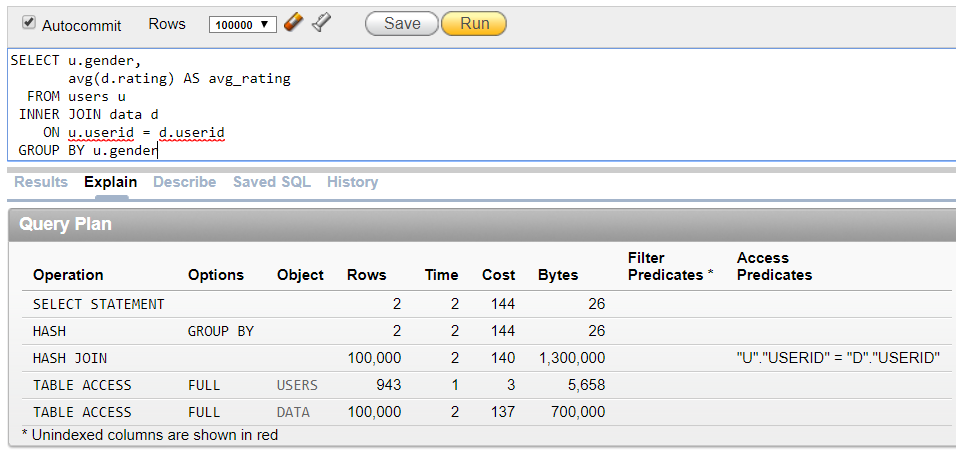
GROUP BY u.gender;

Time taken: 44.283 seconds, Fetched: 2 row(s)

Oracle Rows fetched: 0.11 seconds, 2 rows







1. **Rank movies by average rating**

SELECT m.title,

avg(d.rating) AS avg\_rating

FROM Data\_T1 d

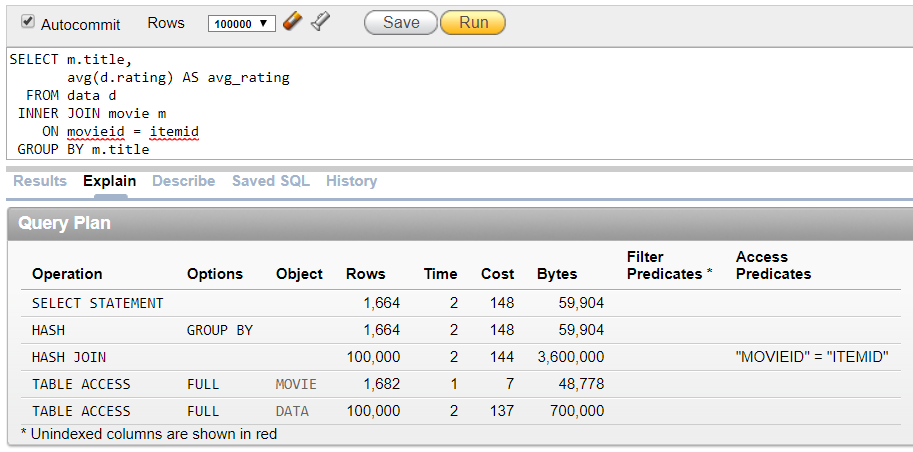
INNER JOIN Movie\_T1 m

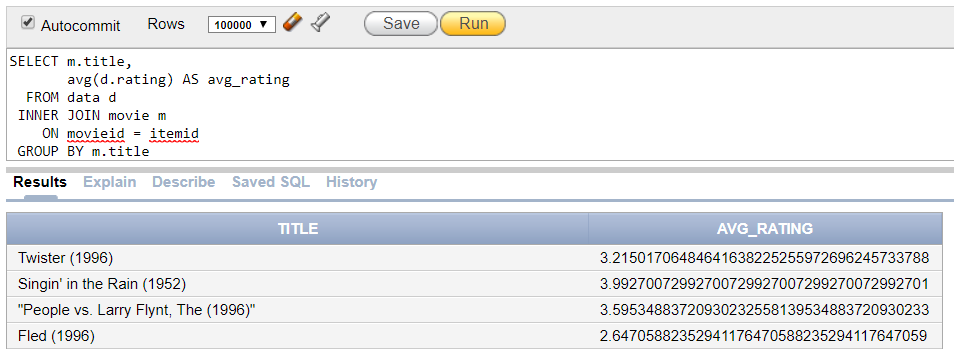
ON movieid = itemid

GROUP BY m.title

Time taken: 51.531 seconds, Fetched: 1661 row(s)

Oracle Rows fetched: 0.30 seconds, 1664 rows

****

****

****

1. **Select movies watched where the users live in Santa Clara and are not an engineer**

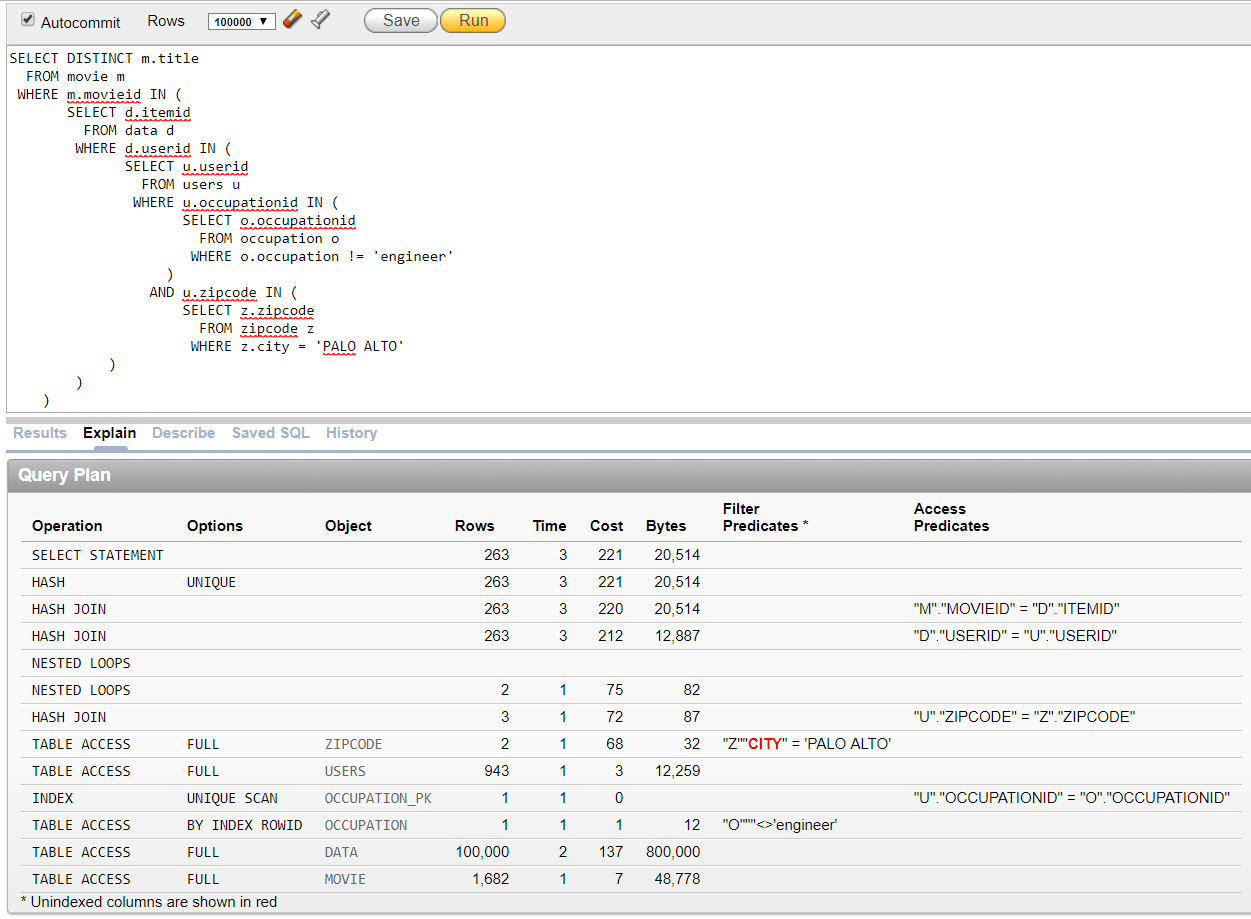
SELECT DISTINCT m.title

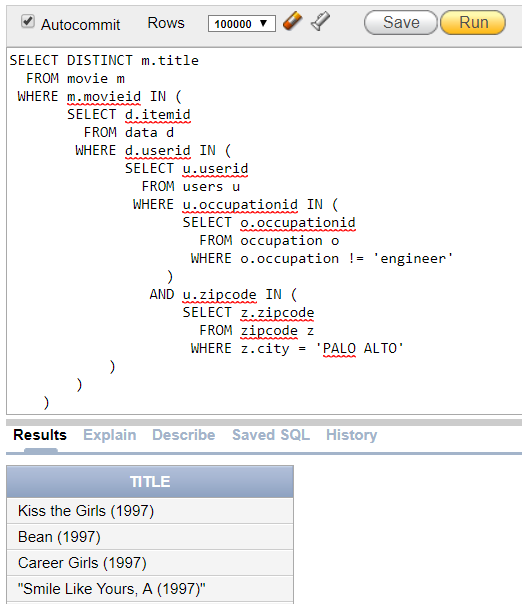
FROM Users\_T1 u, Occupation\_T1 o, Zipcode\_T1 z, Movie\_T1 m, Data\_T1 d

WHERE m.movieid = d.itemid AND d.userid = u.userid AND o.occupationid = u.occupationid AND o.occupation != ‘engineer’ AND z.city = ‘PALO ALTO’ AND u.zipcode = z.zipcode

Time taken: 27.475 seconds

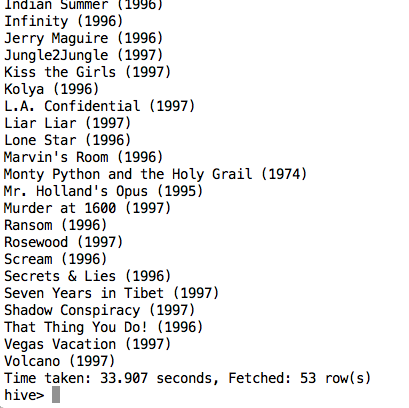
Oracle Rows fetched: 0.06 seconds, 53 rows







Hive:



SELECT m.movieid,  
m.title FROM Movie\_T1 m  
INNER JOIN ( SELECT d.userid,  
d.itemid,  
d.rating FROM Data\_T1 ) d ON (m.movieid = d.itemid​)

hdfs dfs -rm /user/bigdata01/dataDir/data.csv

hdfs dfs -rm /user/bigdata01/freeZipcodeDir/free-zipcode-database.csv

hdfs dfs -rm /user/bigdata01/movieDir/movie.csv

hdfs dfs -rm /user/bigdata01/occupationDir/occupation.csv

hdfs dfs -rm /user/bigdata01/userDir/user.csv

hdfs dfs -rm /user/bigdata01/zipcodeDir/zipcode.csv

**6. 3 Way Join - Average rating by occupation**

SELECT o.occupation, avg(d.rating)

FROM users u

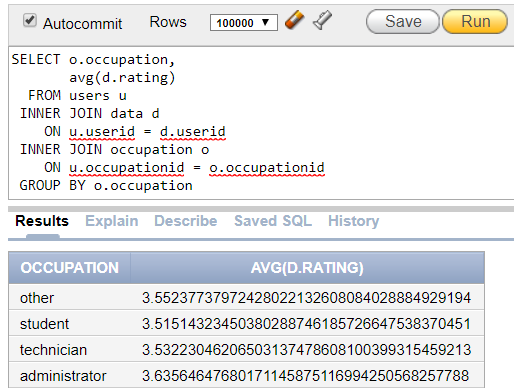
INNER JOIN data d ON u.userid = d.userid

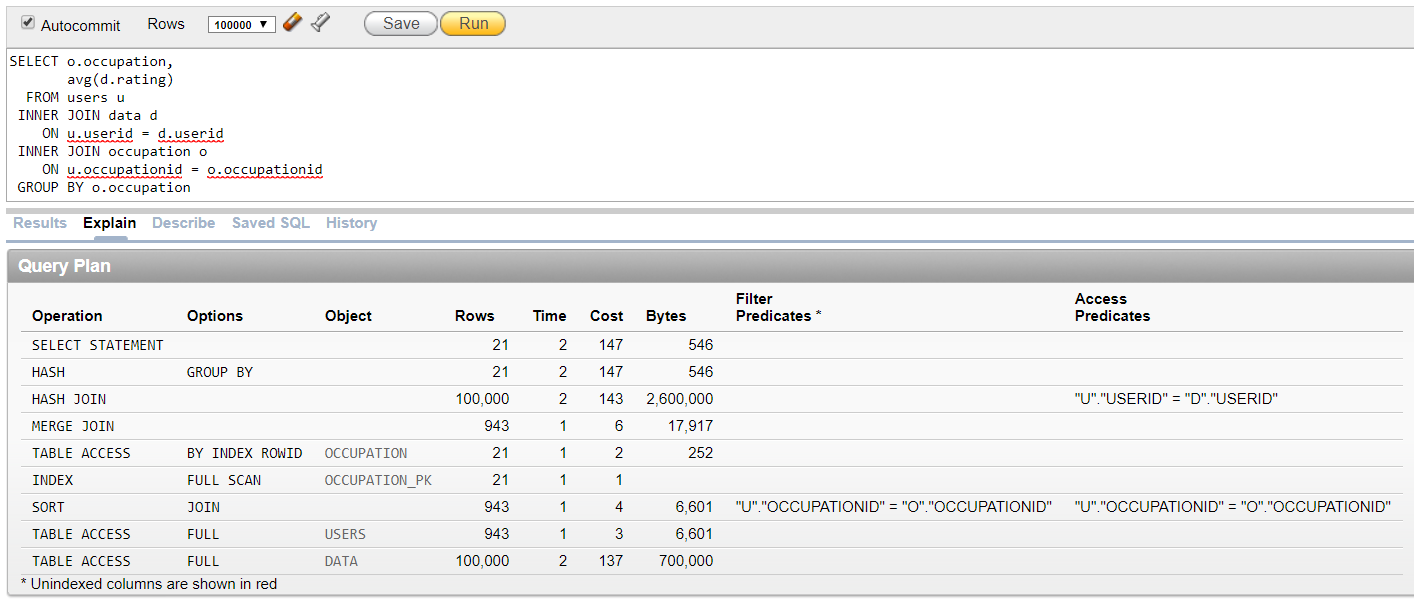
INNER JOIN occupation o ON u.occupationid = o.occupationid

GROUP BY o.occupation

Hive: 21 rows, 41.737 seconds

Oracle: 21 rows, 0.15 seconds





**7. Number of ratings for movies released since Nicholas was born**

SELECT m.title, m.release\_date,

count(d.rating) AS num\_ratings

FROM movie m

INNER JOIN data d

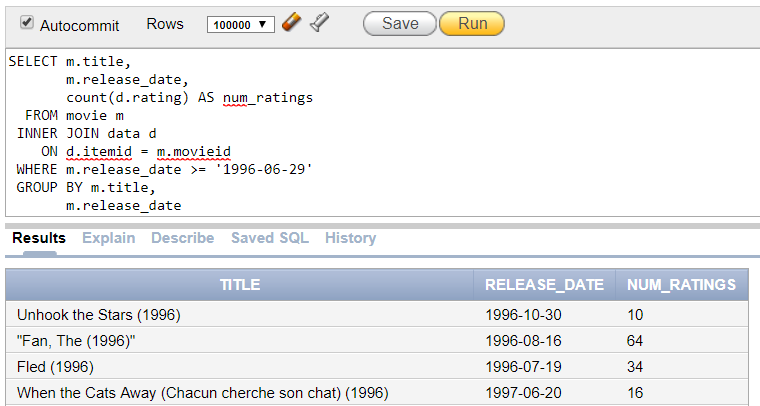
ON d.itemid = m.movieid

WHERE m.release\_date >= '1996-06-29'

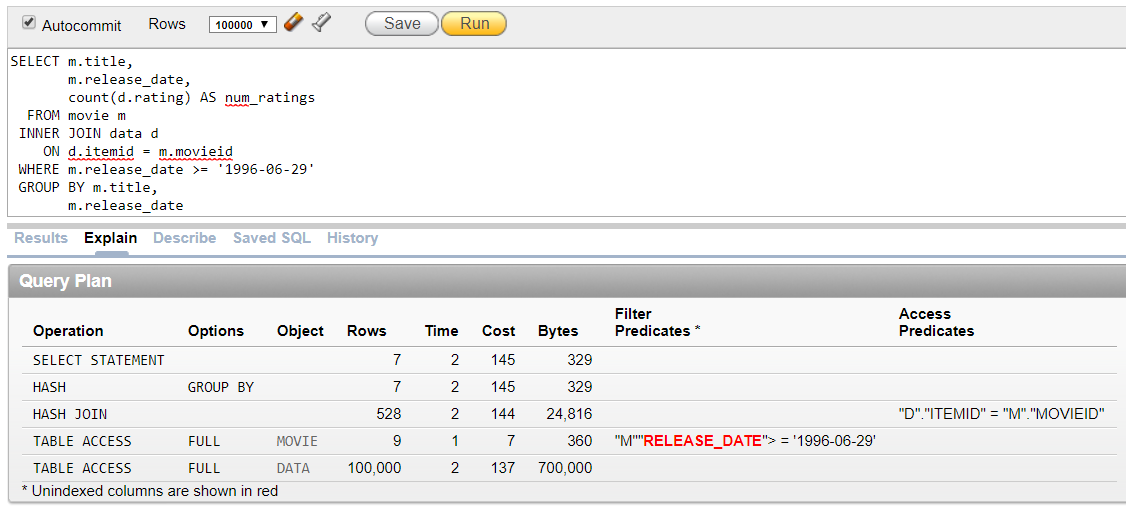
GROUP BY m.title, m.release\_date

Hive:

Oracle: 516 rows in 0.10 seconds







**8. Full location information for users** **using left join**

SELECT u.\*,

z.city,

z.state

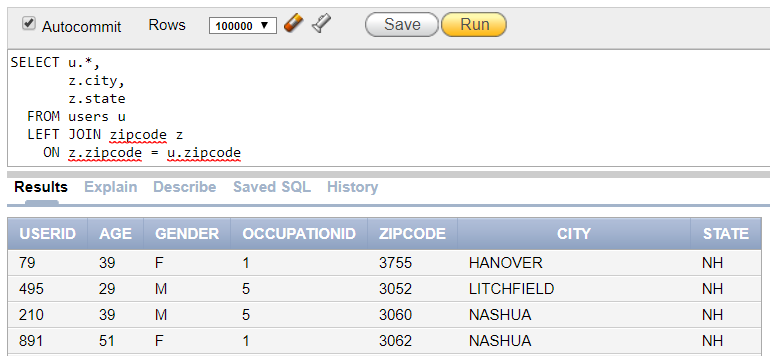
FROM users u

LEFT JOIN zipcode z

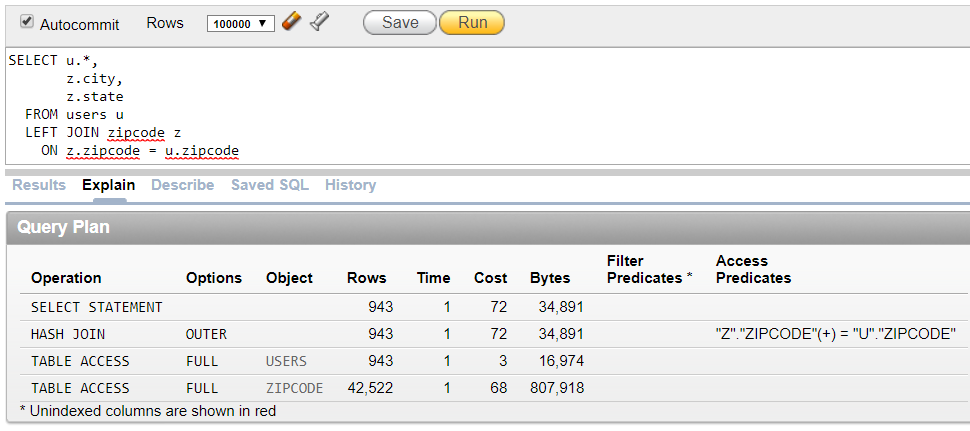
ON z.zipcode = u.zipcode

Hive:

Oracle: 943 rows in 0.14 seconds







<http://www.informit.com/articles/article.aspx?p=2756471&seqNum=4>

Tasks :

SHRADDHA

* Upload data to linux account
* Upload data to hdfs
* Create tables

NICHOLAS

* Create Oracle DB
* Upload data
* Write queries
* Fix queries for Oracle
* Oracle Explain Plan
* Using tool (probably stanford) decide index columns for all tables -- Vikas : Not required spoke to professor.

Added by Vikas Shetty

Oracle Index strategy:

In general, you should create an index on a column in any of the following situations:

The column is queried frequently.

A referential integrity constraint exists on the column.

A UNIQUE key integrity constraint exists on the column.

Some columns are strong candidates for indexing. Columns with one or more of the following characteristics are good candidates for indexing:

Values are unique in the column, or there are few duplicates.

There is a wide range of values (good for regular indexes).

There is a small range of values (good for bitmap indexes).

The column contains many nulls, but queries often select all rows having a value. In this case, a comparison that matches all the non-null values, such as:

WHERE COL\_X >= -9.99 \*power(10,125)

is preferable to

WHERE COL\_X IS NOT NULL

This is because the first uses an index on COL\_X (assuming that COL\_X is a numeric column).

Columns with the following characteristics are less suitable for indexing:

There are many nulls in the column and you do not search on the non-null values.

* Hive sql (all 5)
* Research Paper
* Create presentation for project