BIG DATA ANALYSIS PROJECT

Domain for the project:-Movie Industry

Each year entertainment industry produces many short and large duration movies for audience attention. But every movie does not earn a good profit at box-office. So it's important for the industry to apply big data analytics in their industry to achieve more profit and give the audience the best to watch.

By analyzing large amount of movies and information related to movie it is possible to predict the result of movie at the box-office and reactions of the viewer.

Hollywood is using this technology from past many years for business of their movies but India is still behind in using this analytics.

Advantages of analyzing the movie data:-

1. By detecting the anomalies in previous movies producers and directors may get results that will provide meaningful recommendations.
2. It will help in movie development
3. It will help in movie promotion and distribution
4. It will also help the audience to select the movie according to their preferences.

➔ For this project, dataset is around of 25 records for each table and dataset contains two tables related to movie information and two tables related to actors involved in this industry.

Also the dataset contains some movies from the years 2000 to 2019 and the actors during this time.

-> 1. Schema of some important tables.

movie_table

Movie_id: Specific id provided to each movie stored in data.

Yop: The year in which the movie is produced.

Movie_name: The name of the movie.

Main_actor: Actor in main lead in the movie.

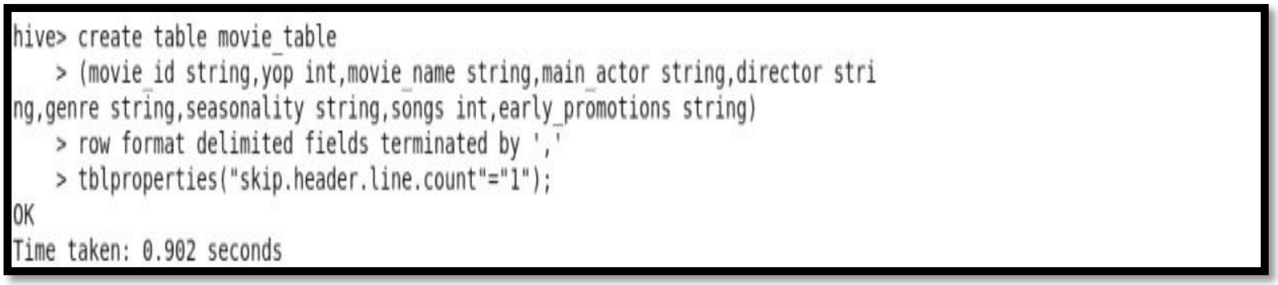
Director: Director of the movie.

Genre: type of the movie.

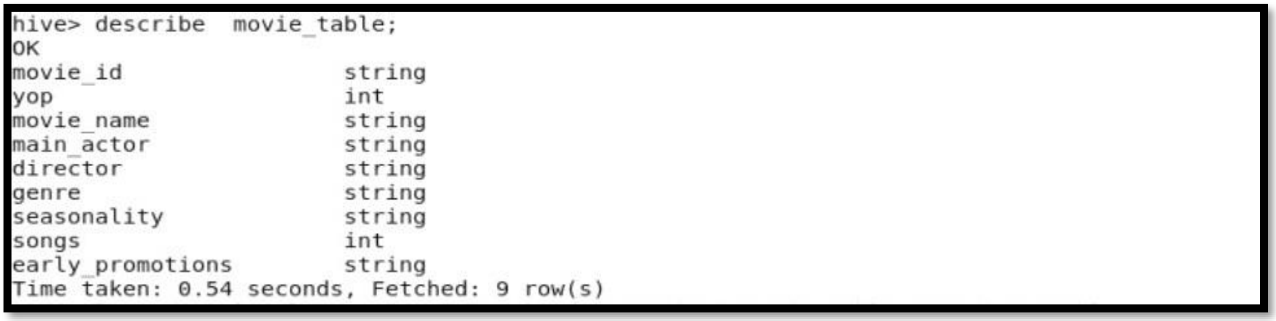
Seasonality: The time of period in which the movie is produced.

Songs: No. of songs in the movie.

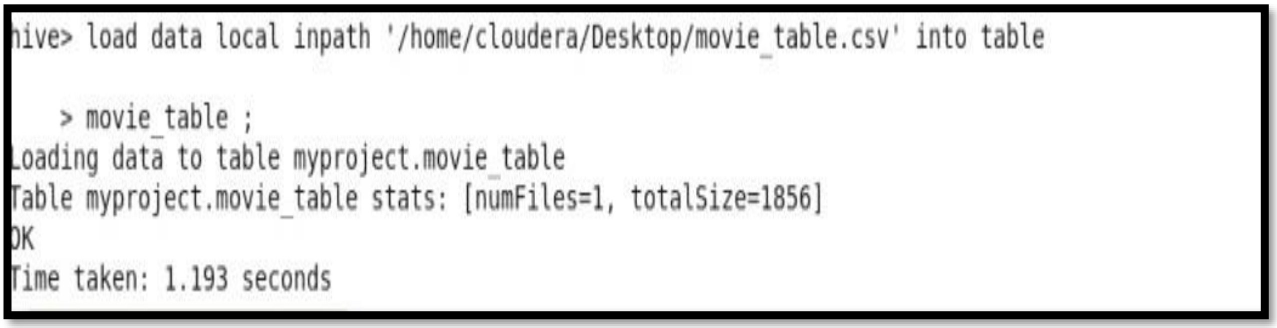
Early_promotions: Is any promotion has been done before the release of the movie.



```
hive> create table movie_table
    > (movie_id string,yop int,movie_name string,main_actor string,director stri
ng,genre string,seasonality string,songs int,early_promotions string)
    > row format delimited fields terminated by ','
    > tblproperties("skip.header.line.count"="1");
OK
Time taken: 0.902 seconds
```



```
hive> describe  movie_table;
OK
movie_id                string
yop                     int
movie_name              string
main_actor              string
director                string
genre                   string
seasonality             string
songs                   int
early_promotions        string
Time taken: 0.54 seconds, Fetched: 9 row(s)
```



```
hive> load data local inpath '/home/cloudera/Desktop/movie_table.csv' into table

    > movie_table ;
Loading data to table myproject.movie_table
Table myproject.movie_table stats: [numFiles=1, totalSize=1856]
OK
Time taken: 1.193 seconds
```

movieprice_table

Movie_id: Specific id provided to each movie stored in data.

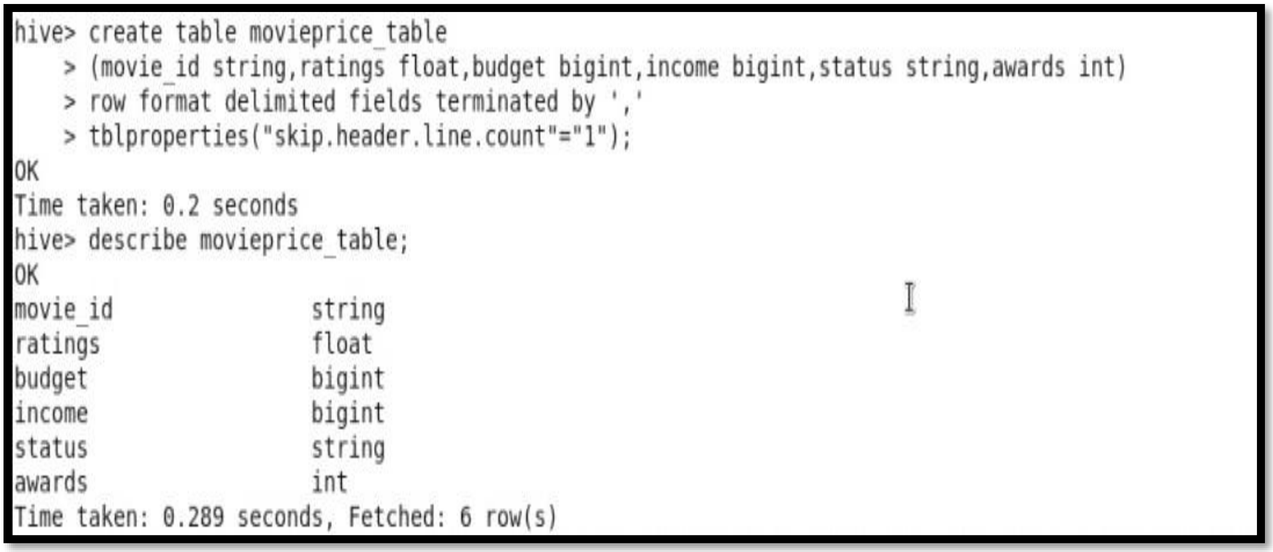
Ratings: The ratings provided by the critics to the movie.

Budget: Amount (rupees) spent on the making and release of the movie.

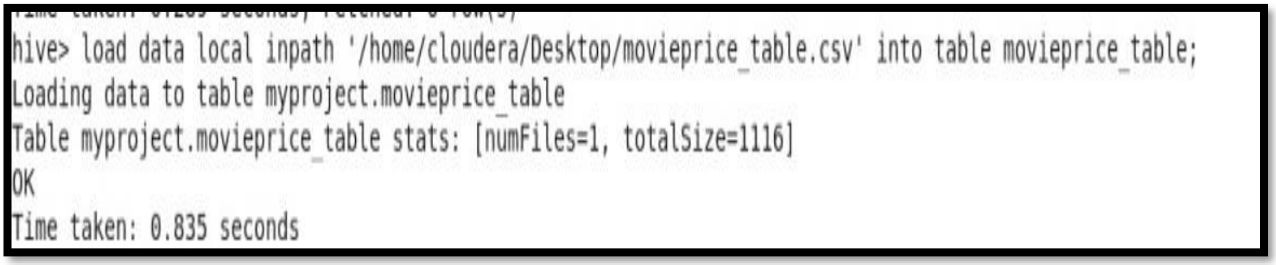
Income: Amount collected by the movie after the release.

Status: The result of the movie at the box office or the reaction of the audience.

Awards: No. of awards collected by the movie at various awards shows.



```
hive> create table movieprice_table
    > (movie_id string,ratings float,budget bigint,income bigint,status string,awards int)
    > row format delimited fields terminated by ','
    > tblproperties("skip.header.line.count"="1");
OK
Time taken: 0.2 seconds
hive> describe movieprice_table;
OK
movie_id                string
ratings                 float
budget                  bigint
income                  bigint
status                  string
awards                  int
Time taken: 0.289 seconds, Fetched: 6 row(s)
```



```
hive> load data local inpath '/home/cloudera/Desktop/movieprice_table.csv' into table movieprice_table;
Loading data to table myproject.movieprice_table
Table myproject.movieprice_table stats: [numFiles=1, totalSize=1116]
OK
Time taken: 0.835 seconds
```

actor_table

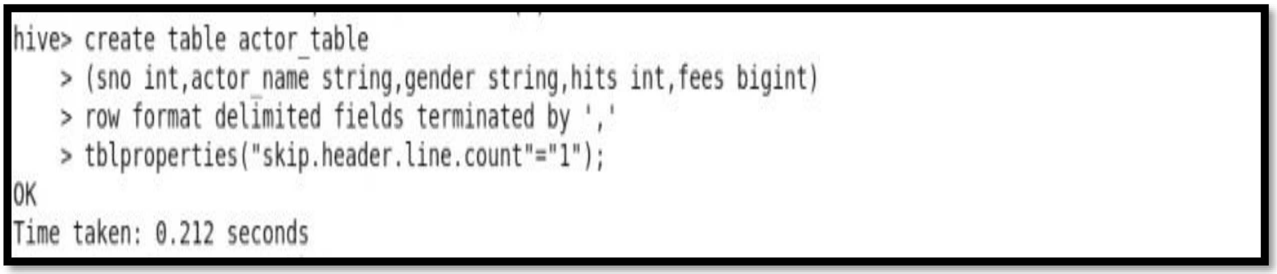
Sno: Serial no is provided to the actor whose data is stored in the table.

Actor_name: Name of the actor.

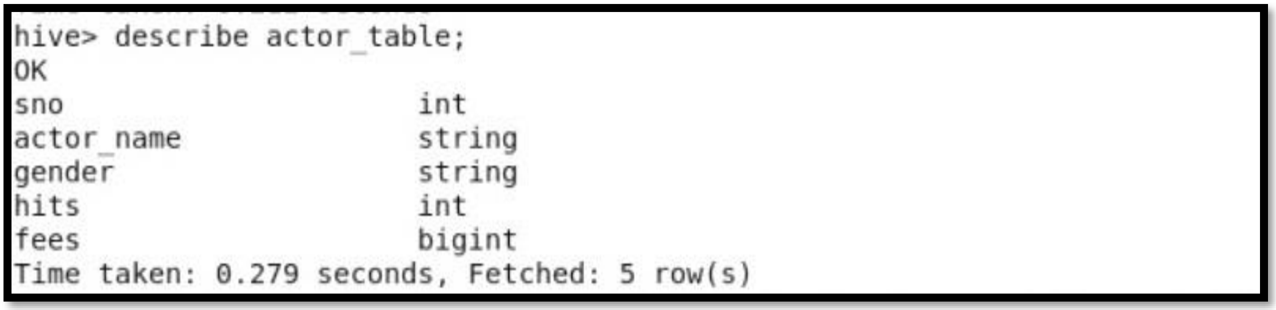
Gender: Gender of the actor.

Hits: No. of hits movie done by the actor.

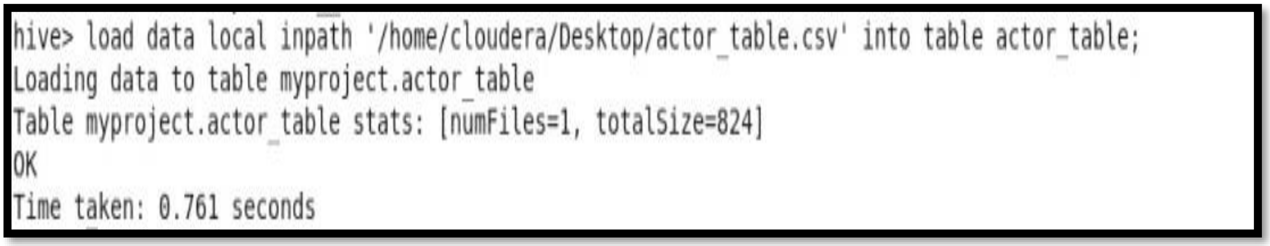
Fees: The amount received by the actor for each movie.



```
hive> create table actor_table
    > (sno int,actor_name string,gender string,hits int,fees bigint)
    > row format delimited fields terminated by ','
    > tblproperties("skip.header.line.count"="1");
OK
Time taken: 0.212 seconds
```



```
hive> describe actor_table;
OK
sno                     int
actor_name              string
gender                  string
hits                    int
fees                    bigint
Time taken: 0.279 seconds, Fetched: 5 row(s)
```



```
hive> load data local inpath '/home/cloudera/Desktop/actor_table.csv' into table actor_table;
Loading data to table myproject.actor_table
Table myproject.actor_table stats: [numFiles=1, totalSize=824]
OK
Time taken: 0.761 seconds
```

actor_personal

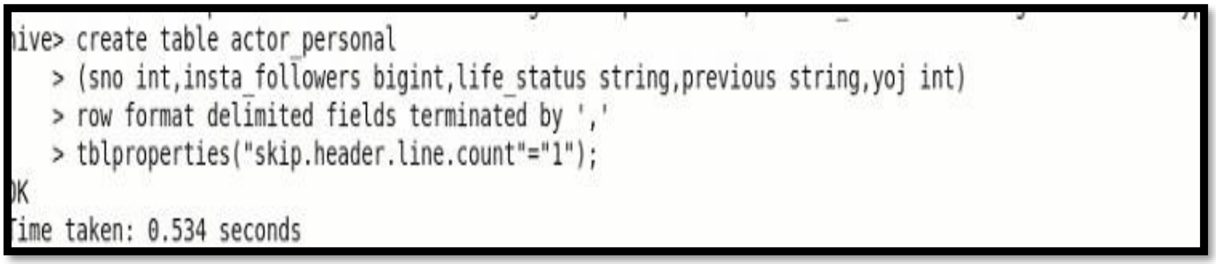
Sno: Serial no is provided to the actor whose data is stored in the table.

Insta_followers: No. of followers of the movie star on social media instagram.

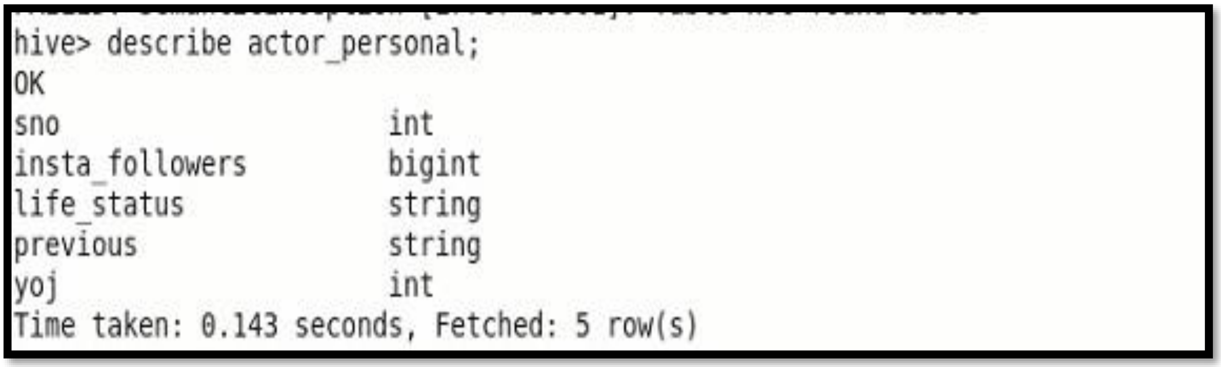
life_status: The movie star is married or not married.

Previous: Whether the movie actor belongs to star family or not.

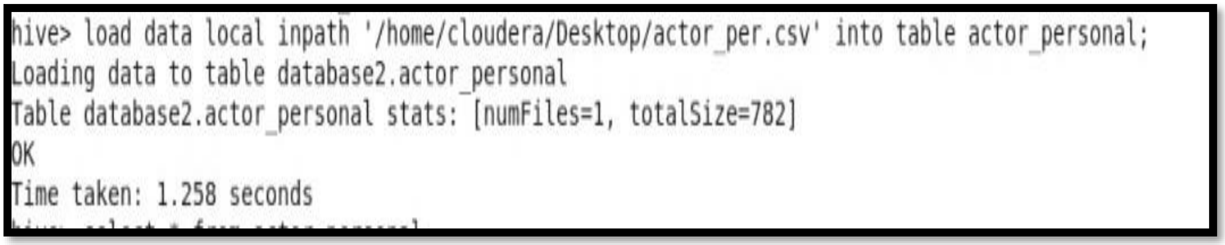
Yoj: The year in which the actor joined the industry.



```
ive> create table actor_personal
   > (sno int,insta_followers bigint,life_status string,previous string,yoj int)
   > row format delimited fields terminated by ','
   > tblproperties("skip.header.line.count"="1");
K
ime taken: 0.534 seconds
```



```
hive> describe actor_personal;
OK
sno                     int
insta_followers         bigint
life_status             string
previous                string
yoj                     int
Time taken: 0.143 seconds, Fetched: 5 row(s)
```

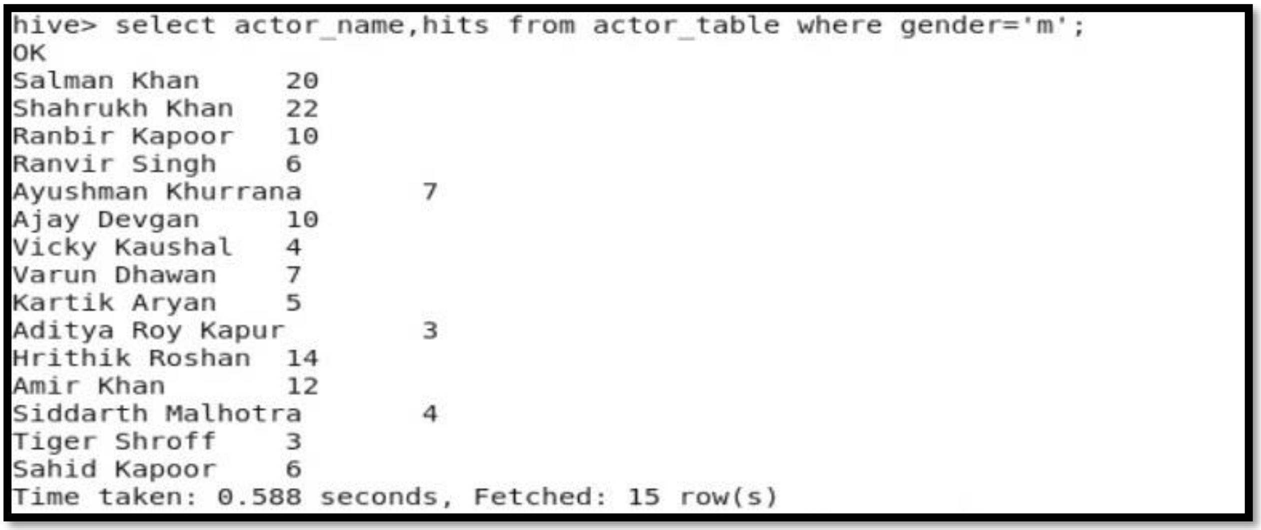


```
hive> load data local inpath '/home/cloudera/Desktop/actor_per.csv' into table actor_personal;
Loading data to table database2.actor_personal
Table database2.actor_personal stats: [numFiles=1, totalSize=782]
OK
Time taken: 1.258 seconds
```

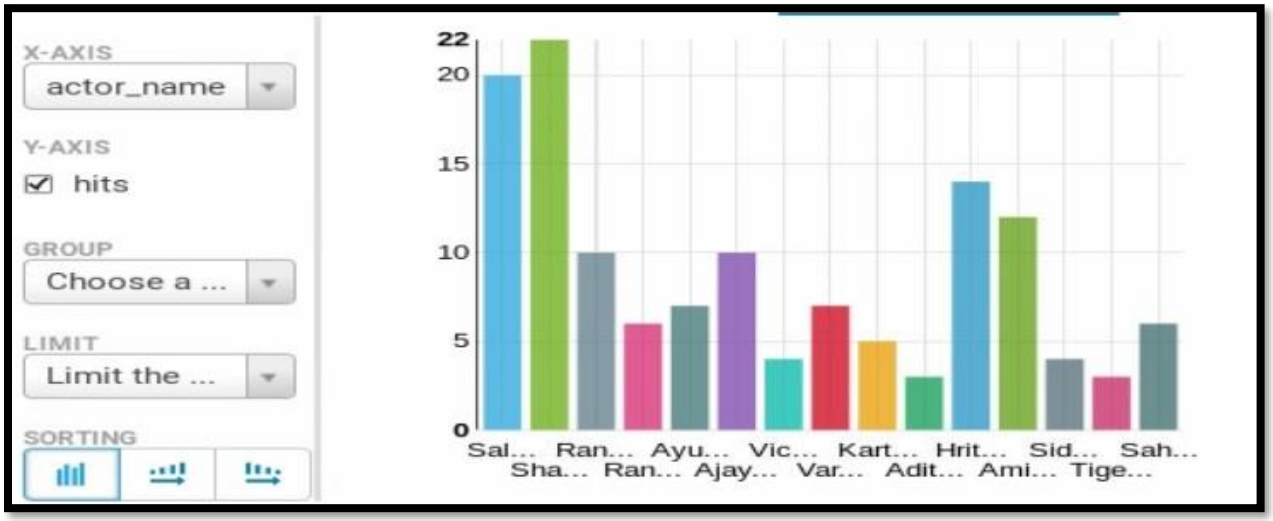
2. Identify 10 challenges/problems that an individual might face.

Challenge 1: How many hit movies has been done by each actor (both male and female) in the past?

-> Select actor_name,hits from actor_table from actor_table where gender='m';



```
hive> select actor_name,hits from actor_table where gender='m';
OK
Salman Khan     20
Shahrukh Khan   22
Ranbir Kapoor   10
Ranvir Singh    6
Ayushman Khurrana       7
Ajay Devgan     10
Vicky Kaushal   4
Varun Dhawan    7
Kartik Aryan    5
Aditya Roy Kapur        3
Hrithik Roshan  14
Amir Khan       12
Siddarth Malhotra       4
Tiger Shroff    3
Sahid Kapoor    6
Time taken: 0.588 seconds, Fetched: 15 row(s)
```



X-AXIS actor_name

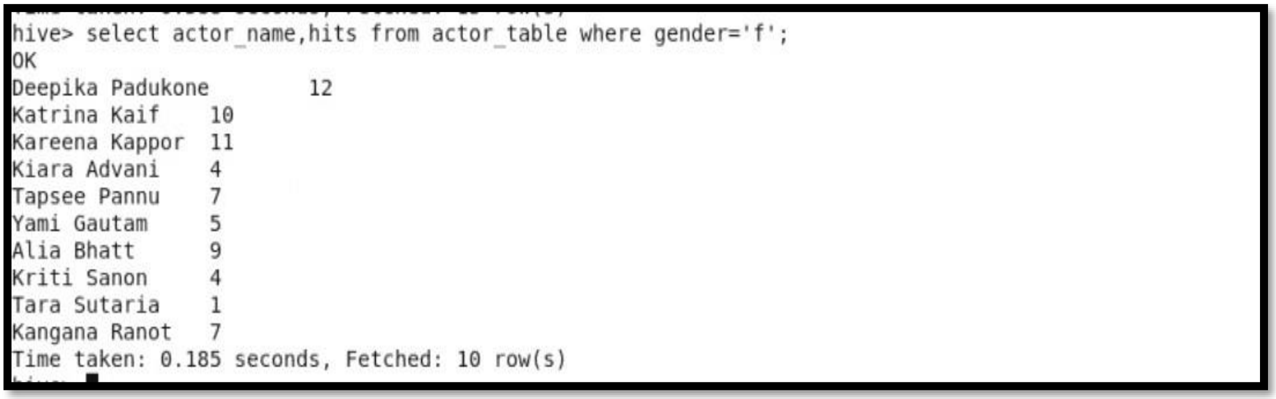
Y-AXIS ☑ hits

GROUP Choose a ...

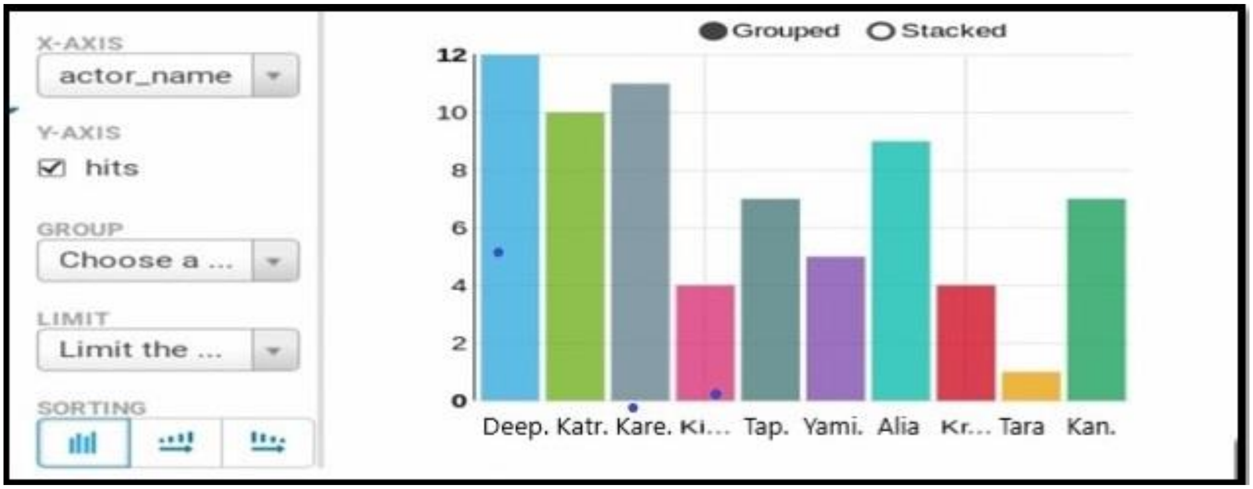
LIMIT Limit the ...

SORTING

Select actor_name,hits from actor_table from actor_table where gender='f';



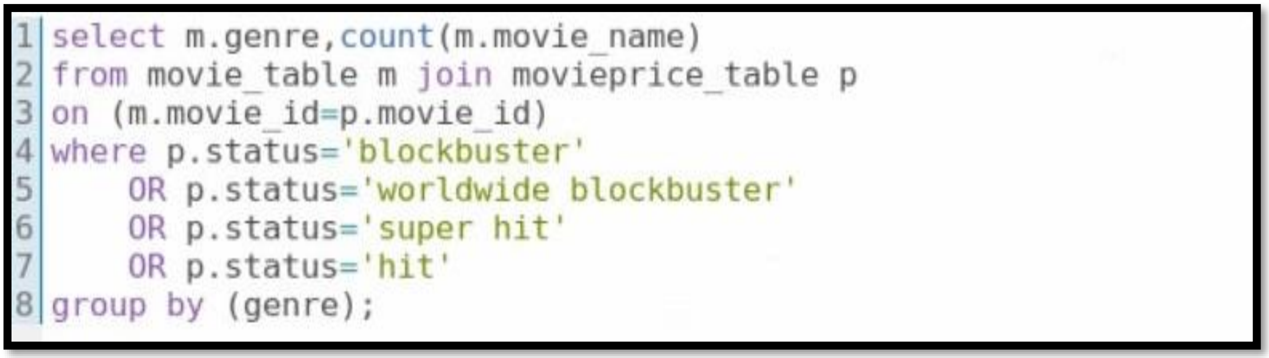
```
hive> select actor_name,hits from actor_table where gender='f';
OK
Deepika Padukone        12
Katrina Kaif    10
Kareena Kappor  11
Kiara Advani    4
Tapsee Pannu    7
Yami Gautam     5
Alia Bhatt      9
Kriti Sanon     4
Tara Sutaria    1
Kangana Ranot   7
Time taken: 0.185 seconds, Fetched: 10 row(s)
```



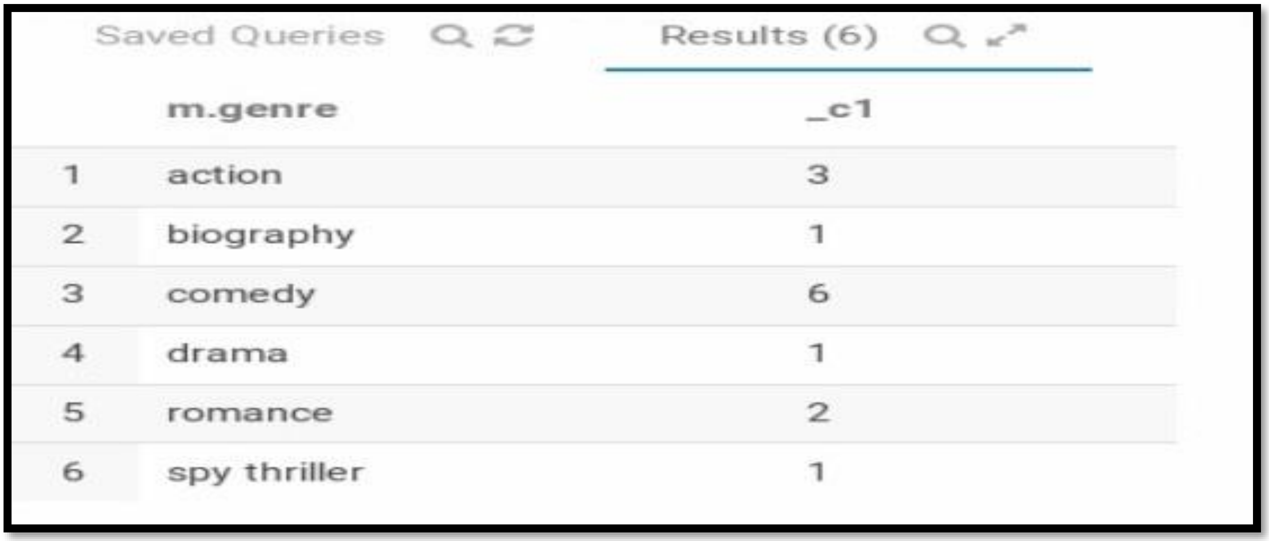
Insights: This data will help the producer and directors to cast the actors in their further movies and will also help the audiences to decide for which movie they have to go based on cast in that movie.

Challenge 2: Which type of movie (genre) is more liked by the audience?

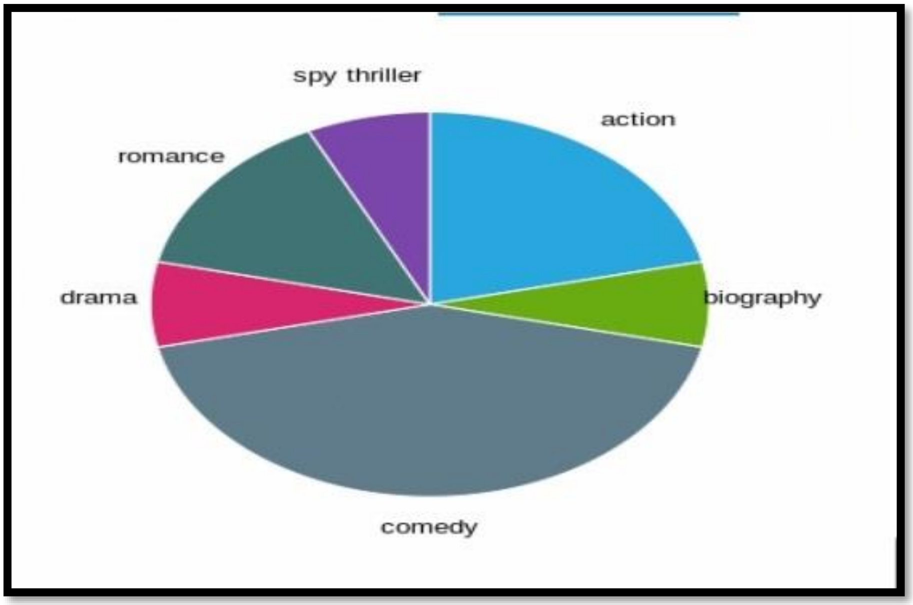
-> select m.genre,count(m.movie_name) from movie_table m join movieprice_table p on (m.movie_id=p.movie_id) where p.status='blockbuster' or p.status='worldwide blockbuster' or p.status='super hit' or p.status='hit' group by (genre);



```
select m.genre,count(m.movie_name)
from movie_table m join movieprice_table p
on (m.movie_id=p.movie_id)
where p.status='blockbuster'
    OR p.status='worldwide blockbuster'
    OR p.status='super hit'
    OR p.status='hit'
group by (genre);
```



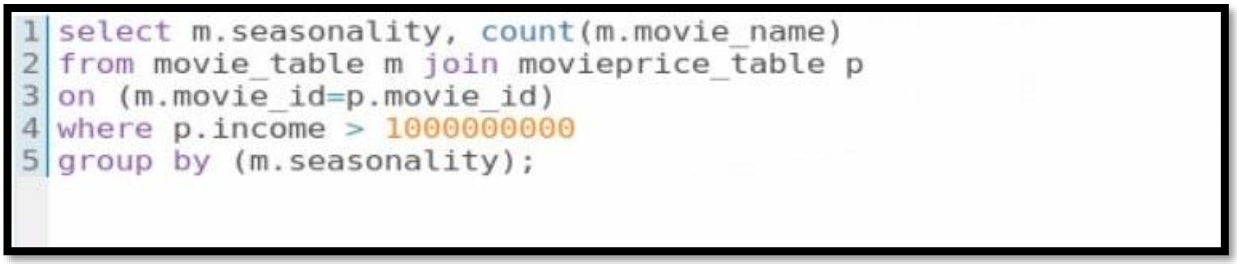
	m.genre	_c1
1	action	3
2	biography	1
3	comedy	6
4	drama	1
5	romance	2
6	spy thriller	1



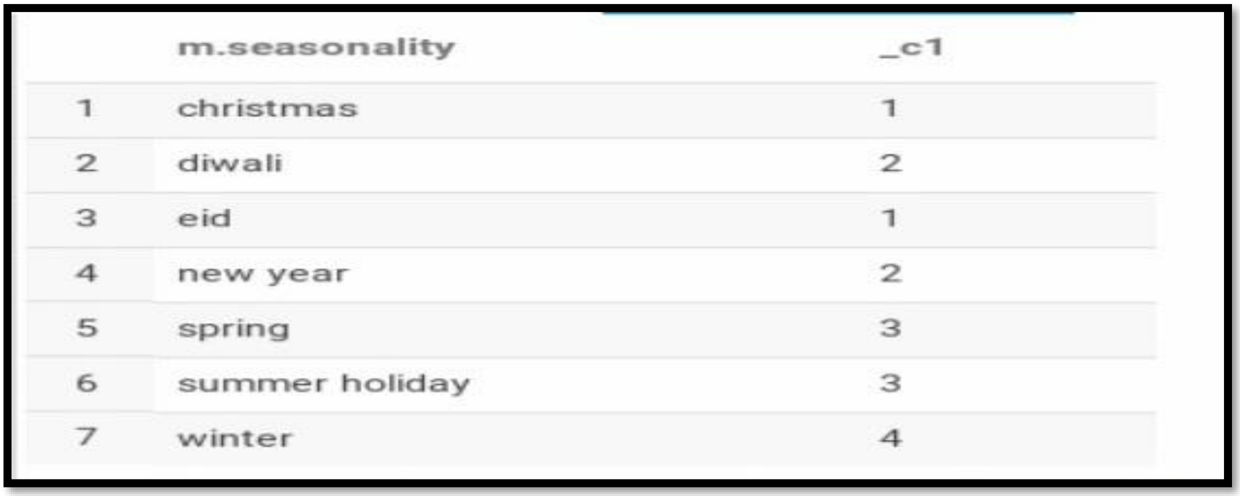
Insights: In today’s life everyone have their own interest and according to their choice they prefer the movie. From this data we get the result that most of the people like to watch comedy movies and they earn more compared to others movies. This data will help the directors to make the movie on the comedy topic because it is liked more by the audience.

Challenge 3: Does movie release day affect the business of the movie at box office (income)?

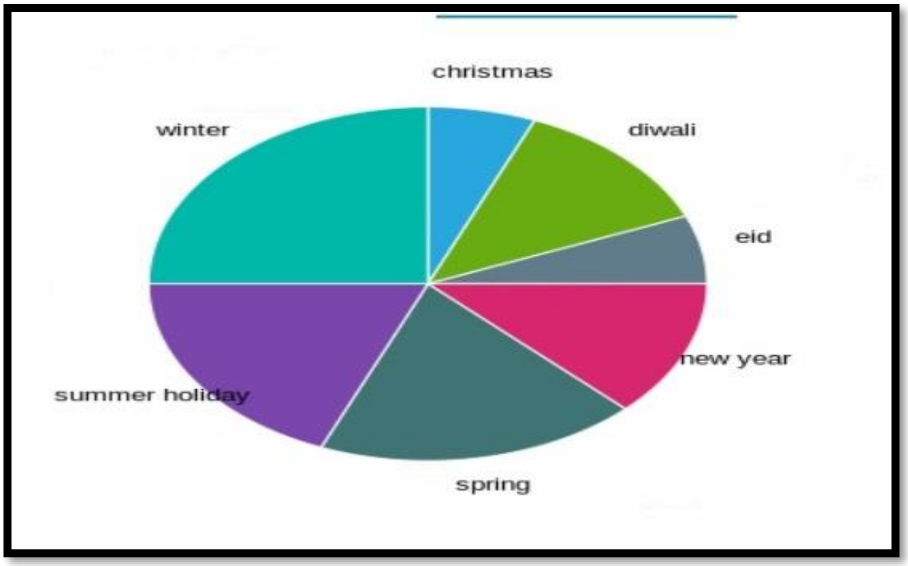
-> select m.seasonality,count(m.movie_name) from movie_table m join movieprice_table p on(m.movie_id=p.movie_table) where income>1000000000 group by(m.sesaonality);



```
select m.seasonality, count(m.movie_name)
from movie_table m join movieprice_table p
on (m.movie_id=p.movie_id)
where p.income > 1000000000
group by (m.seasonality);
```



	m.seasonality	_c1
1	christmas	1
2	diwali	2
3	eid	1
4	new year	2
5	spring	3
6	summer holiday	3
7	winter	4

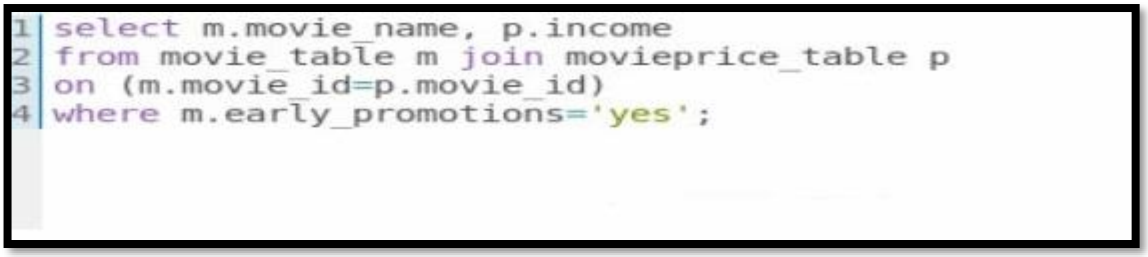


Insights: In past, movie release day was not a big deal but now release day of movie is very important.Most of the directors and producers prefer to release their movie on a specific days like festival,eves and in holidays. From the above data this can be clearly seen that the movie who earns more than 100 crores relaese on the festival ,eves or holidays in winter or summer.

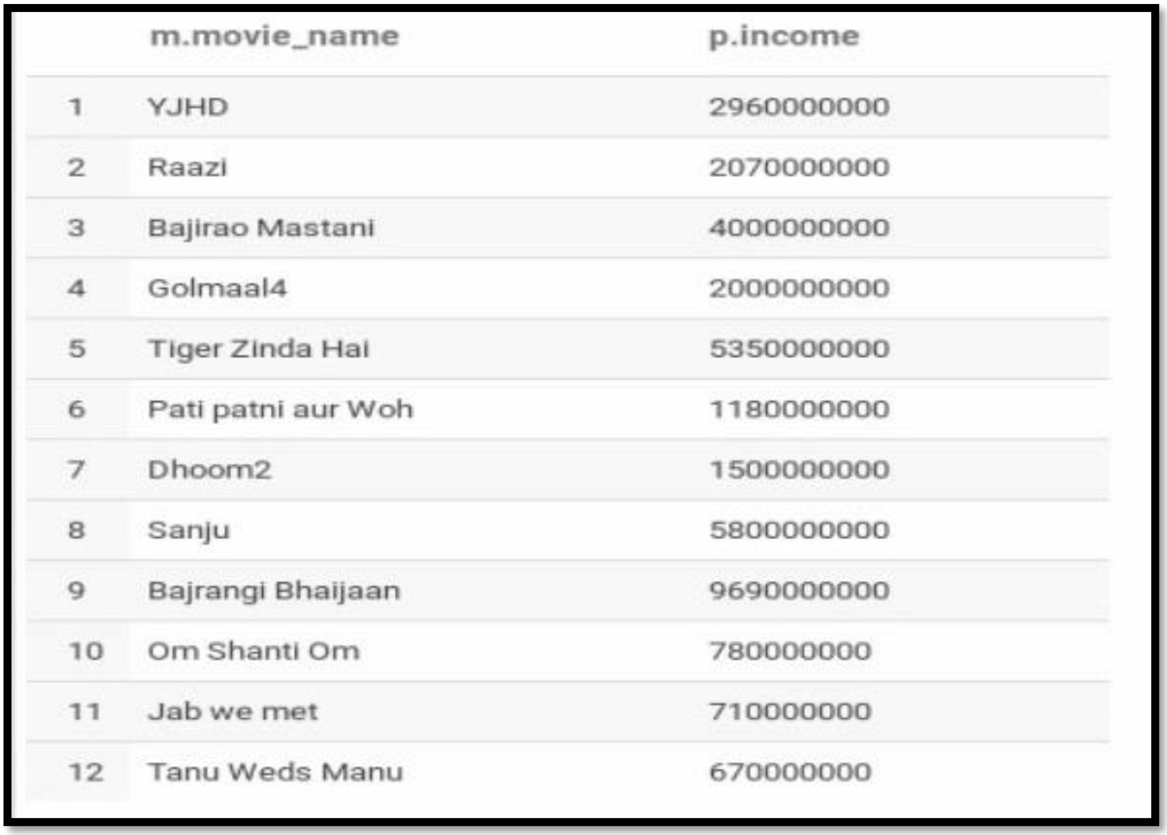
Challenge 4: Do early promotions help the movie to earn more?

-> select m.movie_name,p.income from movie_table m join movieprice_table p on(m.movie_id=p.movieprice_table)

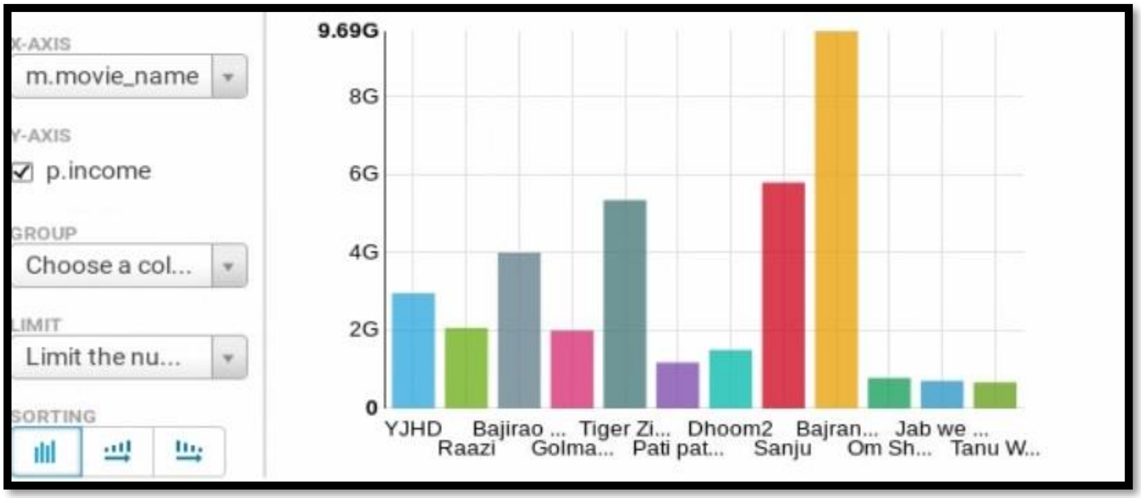
Where early_promotions='yes';



```
select m.movie_name, p.income
from movie_table m join movieprice_table p
on (m.movie_id=p.movie_id)
where m.early_promotions='yes';
```



	m.movie_name	p.income
1	YJHD	2960000000
2	Raazi	2070000000
3	Bajirao Mastani	4000000000
4	Golmaal4	2000000000
5	Tiger Zinda Hai	5350000000
6	Pati patni aur Woh	1180000000
7	Dhoom2	1500000000
8	Sanju	5800000000
9	Bajrangi Bhaijaan	9690000000
10	Om Shanti Om	780000000
11	Jab we met	710000000
12	Tanu Weds Manu	670000000



X-AXIS
m.movie_name

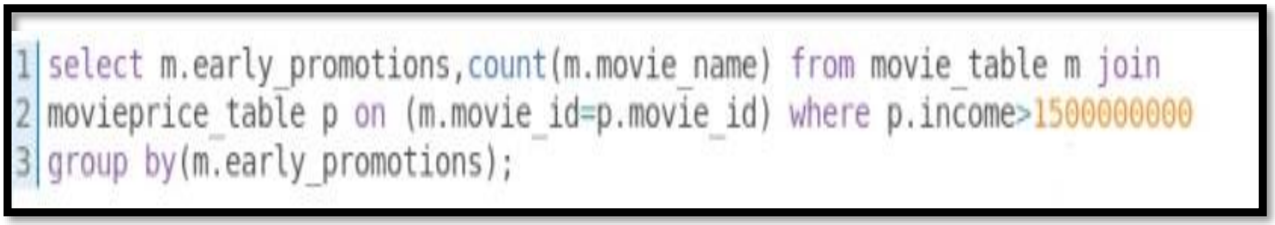
Y-AXIS
☑ p.income

GROUP
Choose a col...

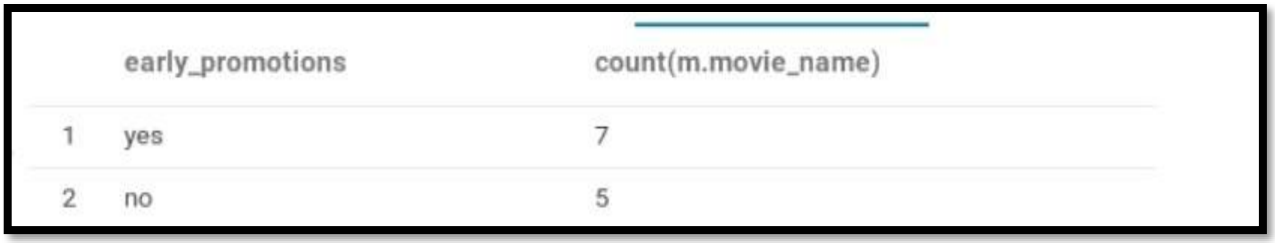
LIMIT
Limit the nu...

SORTING

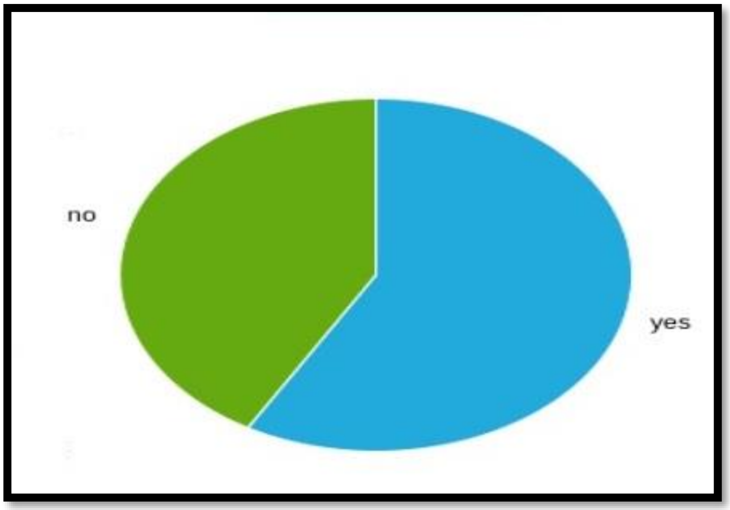
Insights: It is every important to make acknowledge the people about the movie before release.From the above data it can be visulaized that the movie who done early promotions earn a good amount of money.



```
select m.early_promotions,count(m.movie_name) from movie_table m join
movieprice_table p on (m.movie_id=p.movie_id) where p.income>1500000000
group by(m.early_promotions);
```



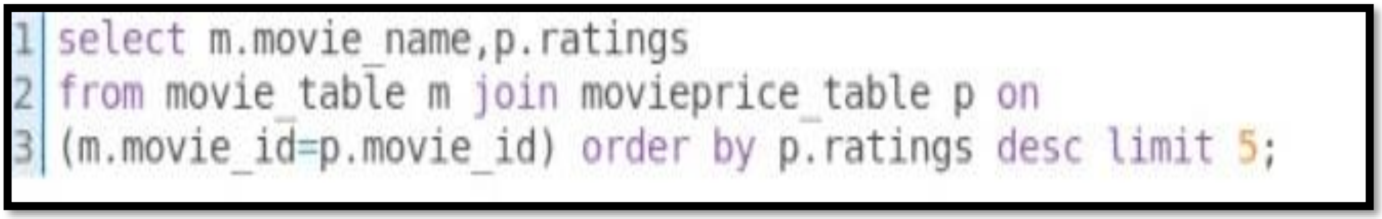
	early_promotions	count(m.movie_name)
1	yes	7
2	no	5



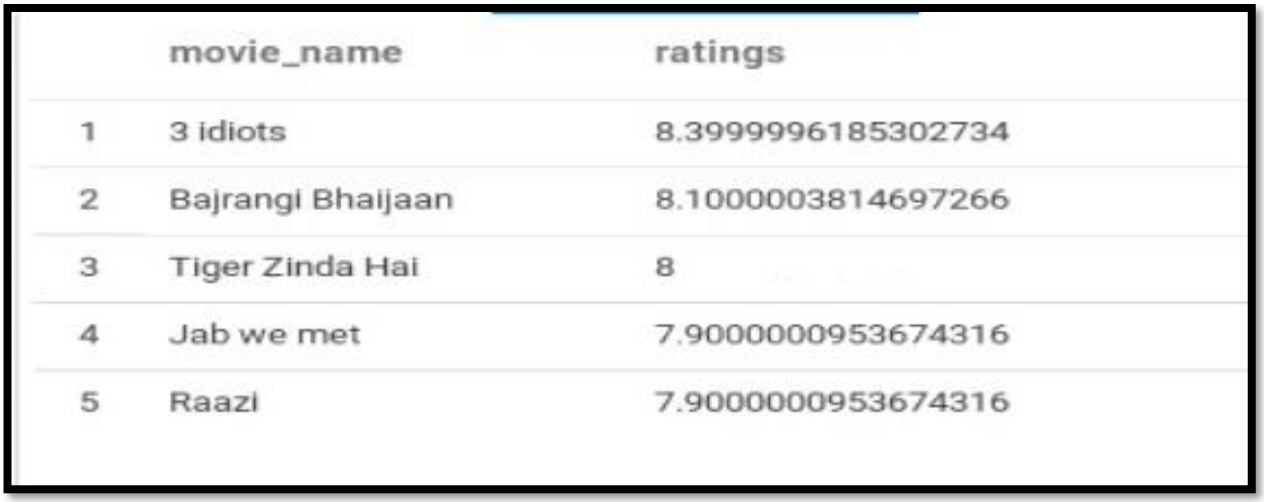
Insights: From the above data it can be seen that the movie who earn more done early promotions but still there are some which do not done early promotions but still earn agood amount of money. So, its good for movie business to do promotions before release.

Challenge 5: Which are the top five movies of bollywood according to the ratings provided by critics(imdb) and profit at box office ?

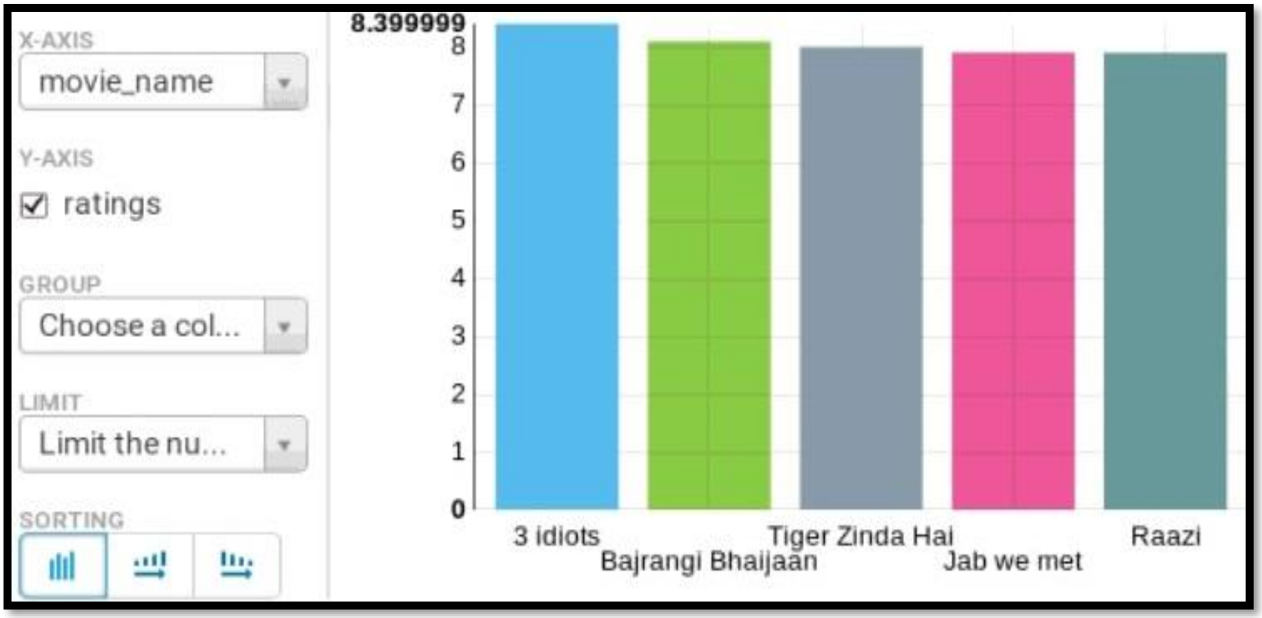
->select m.movie_name,p.ratings from movie_table m join movieprice_table p on (m.movie_id=p.movie_id) order by p.ratings desc limit 5;



```
select m.movie_name,p.ratings
from movie_table m join movieprice_table p on
(m.movie_id=p.movie_id) order by p.ratings desc limit 5;
```



	movie_name	ratings
1	3 idiots	8.3999996185302734
2	Bajrangi Bhaijaan	8.1000003814697266
3	Tiger Zinda Hai	8
4	Jab we met	7.9000000953674316
5	Raazi	7.9000000953674316



X-AXIS
movie_name

Y-AXIS
☑ ratings

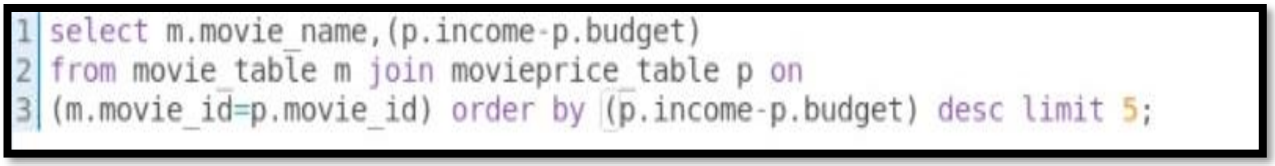
GROUP
Choose a col...

LIMIT
Limit the nu...

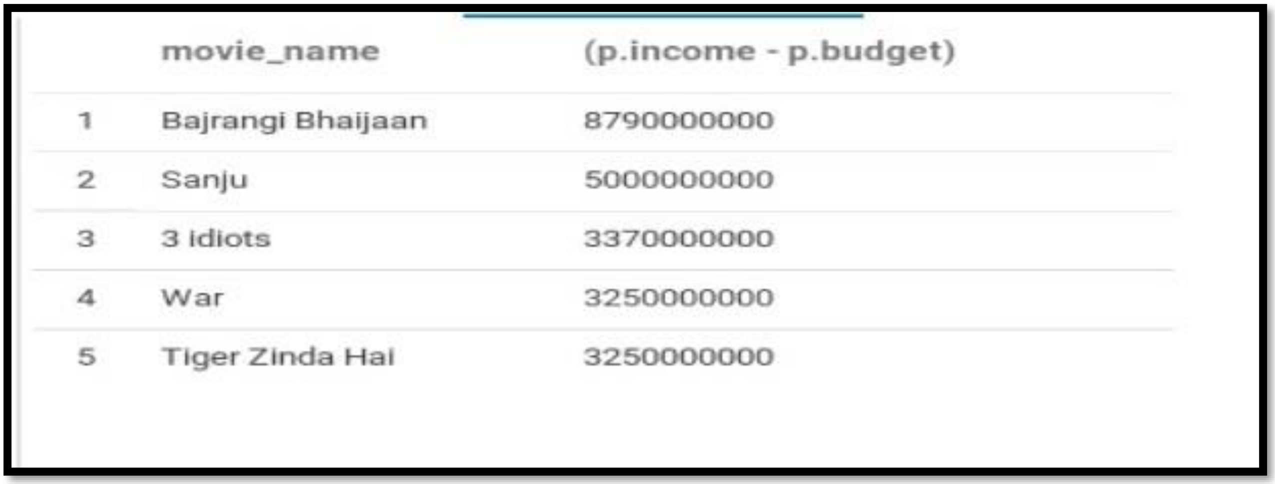
SORTING

select m.movie_name,(p.income-p.budget) from movie_table m join movieprice_table p on(m.movie_id=p.movie_id) order by (p.income-p.budget) desc limit 5;

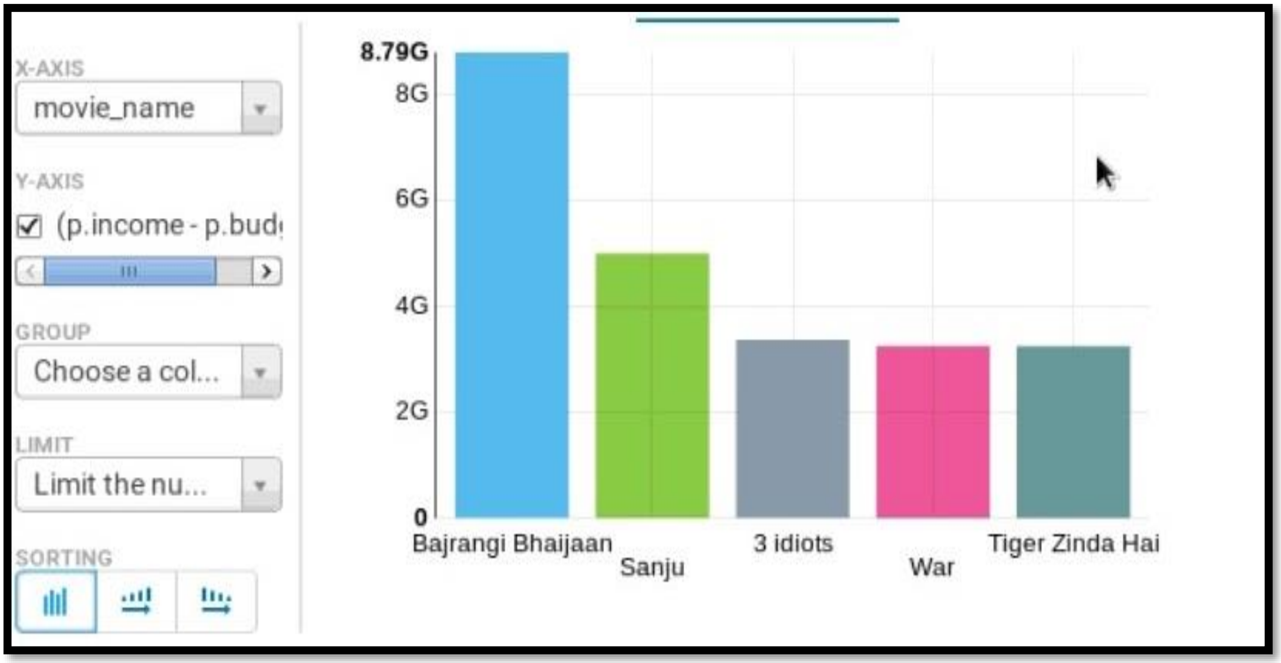
Insights: Many persons believes on critics and thus the rating given by them.So whenever they want to watch a movie they first see its ratings. Ratings is essential aspect for the movie business.



```
select m.movie_name,(p.income-p.budget)
from movie_table m join movieprice_table p on
(m.movie_id=p.movie_id) order by (p.income-p.budget) desc limit 5;
```



	movie_name	(p.income - p.budget)
1	Bajrangi Bhaijaan	8790000000
2	Sanju	5000000000
3	3 idiots	3370000000
4	War	3250000000
5	Tiger Zinda Hai	3250000000



X-AXIS
movie_name

Y-AXIS
☑ (p.income - p.bud

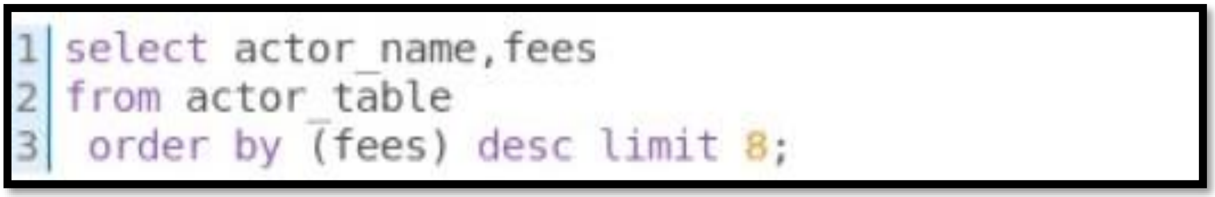
GROUP
Choose a col...

LIMIT
Limit the nu...

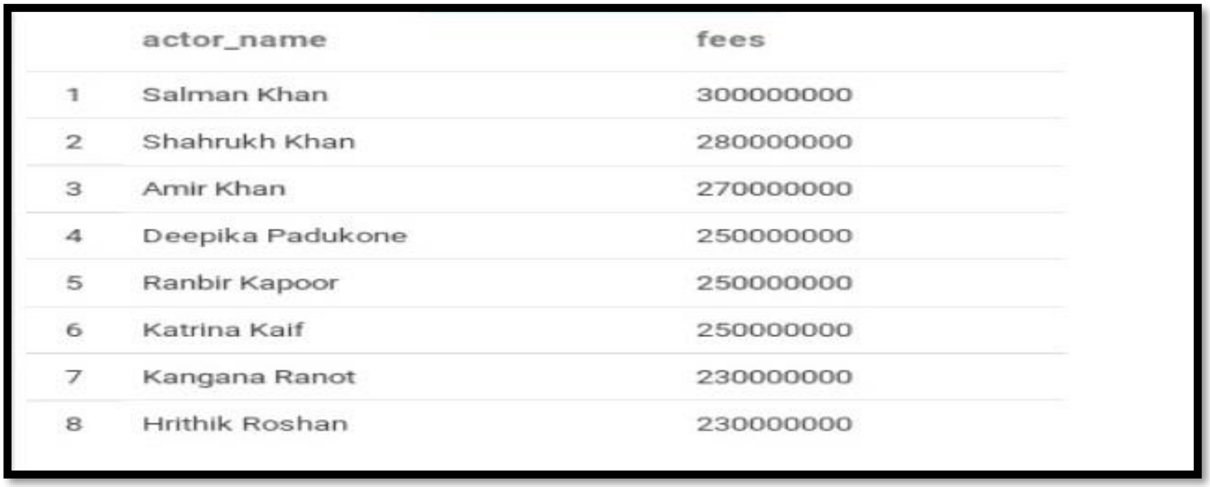
SORTING

Challenge 6: Which are the top most paid actors of bollywood?

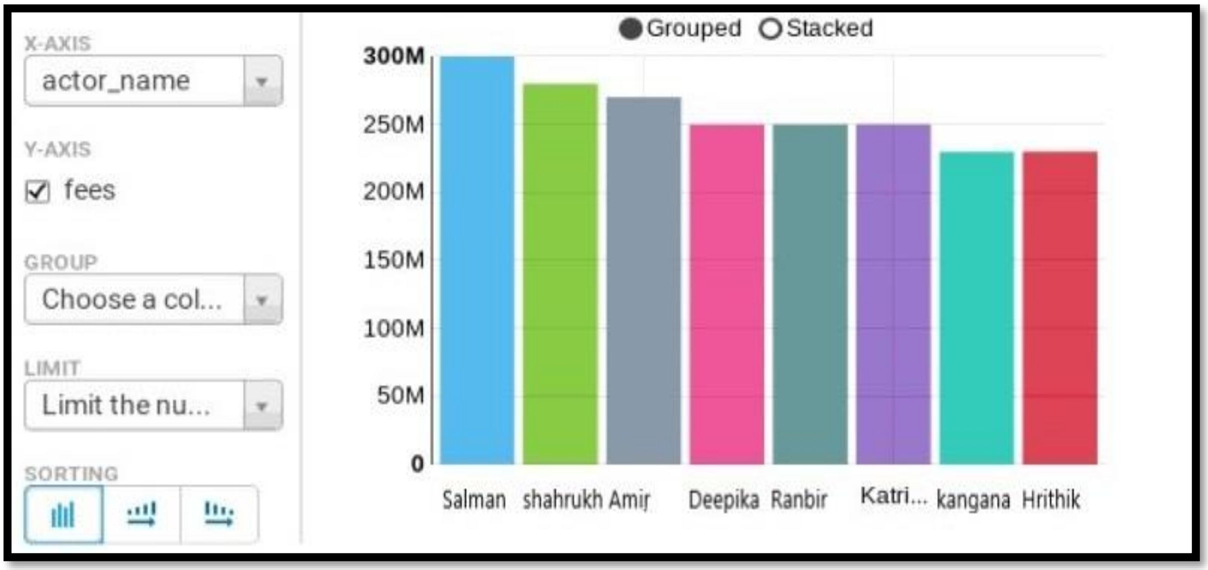
-> select actor_name,fees from actor_table order by fees desc limit 8 ;



```
select actor_name,fees
from actor_table
 order by (fees) desc limit 8;
```



	actor_name	fees
1	Salman Khan	300000000
2	Shahrukh Khan	280000000
3	Amir Khan	270000000
4	Deepika Padukone	250000000
5	Ranbir Kapoor	250000000
6	Katrina Kaif	250000000
7	Kangana Ranot	230000000
8	Hrithik Roshan	230000000



X-AXIS
actor_name

Y-AXIS
☑ fees

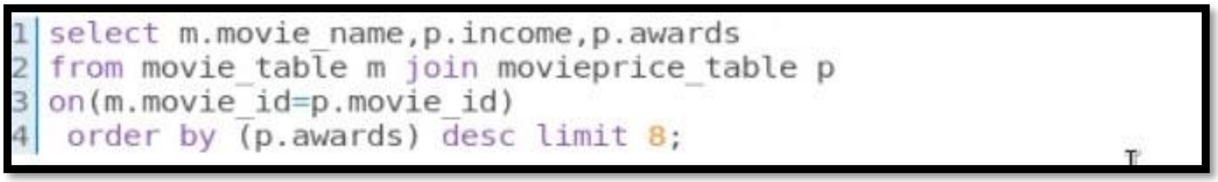
GROUP
Choose a col...

LIMIT
Limit the nu...

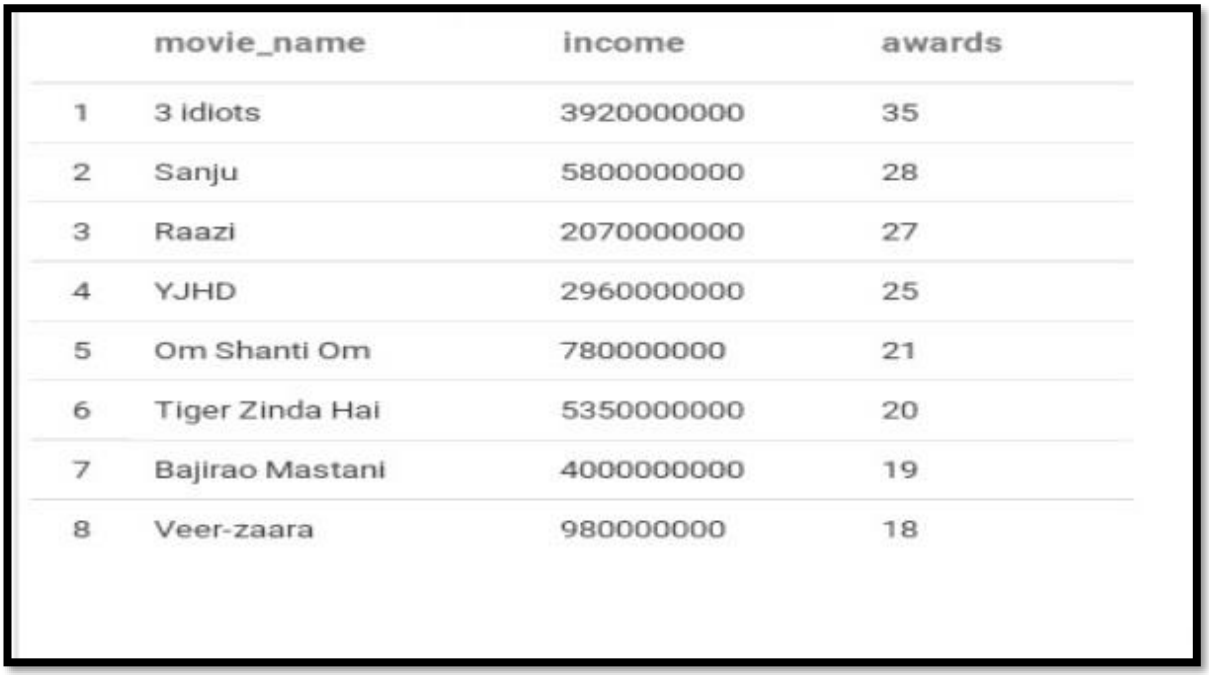
Insights: Directors and producers always want to cast the best actor in their movies and for that they always wanted to know which actor demands how much fees and also people are very curious to know about their fees. So above data will provide help to them.

Challenge 7: Is their any relation between the no of awards and money collected by the movie means it is true or not that the "movie which earns more has more no of awards"?

-> select m.movie_name,p.income,p.awards from movie_table m join movieprice_table p on(m.movie_id=p.movie_id) order by p.awards desc limit 8;



```
select m.movie_name,p.income,p.awards
from movie_table m join movieprice_table p
on(m.movie_id=p.movie_id)
 order by (p.awards) desc limit 8;
```

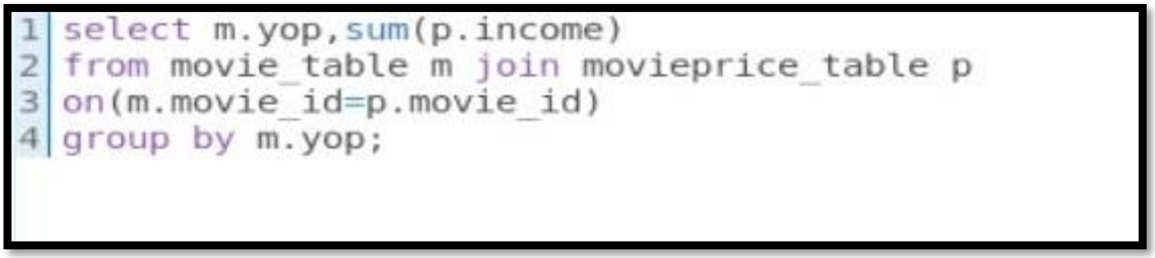


	movie_name	income	awards
1	3 idiots	3920000000	35
2	Sanju	5800000000	28
3	Raazi	2070000000	27
4	YJHD	2960000000	25
5	Om Shanti Om	780000000	21
6	Tiger Zinda Hai	5350000000	20
7	Bajirao Mastani	4000000000	19
8	Veer-zaara	980000000	18

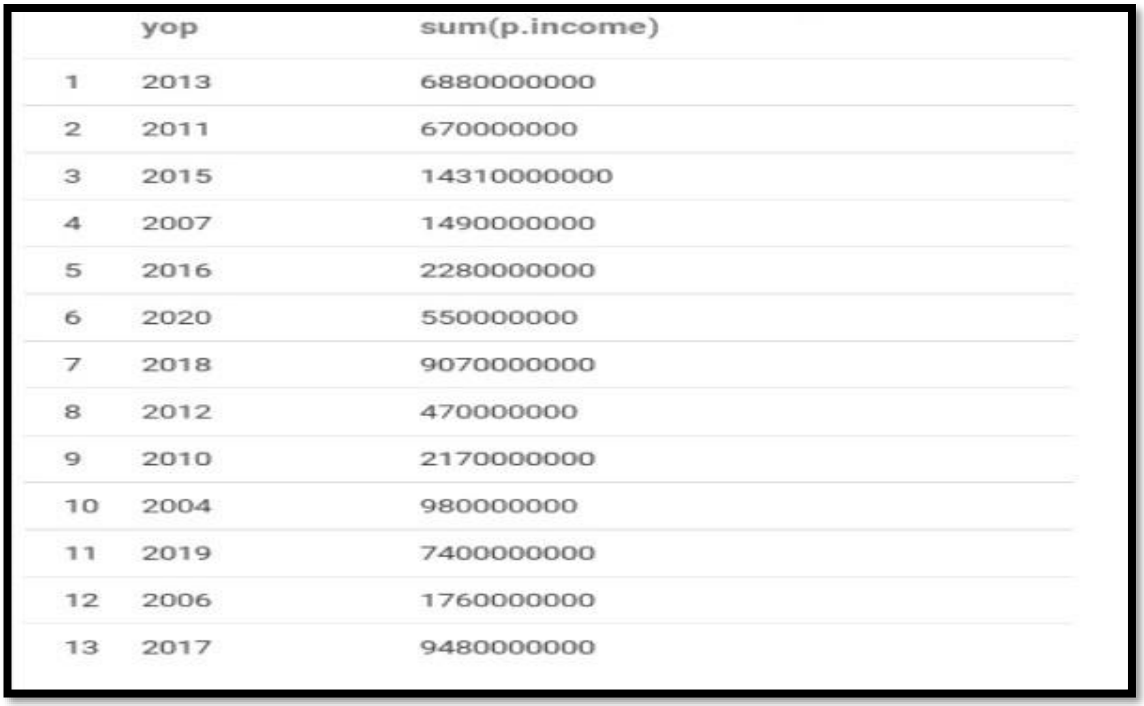
Insights: From the avove data it can be clearly seen that it is not important that the moive which earns more has more no of awards because sometimes movie which has not a good content also earns a lot due to the actors present in the movie.

Challenge 8: No. of movies each year and the income of box office bollywood each year?

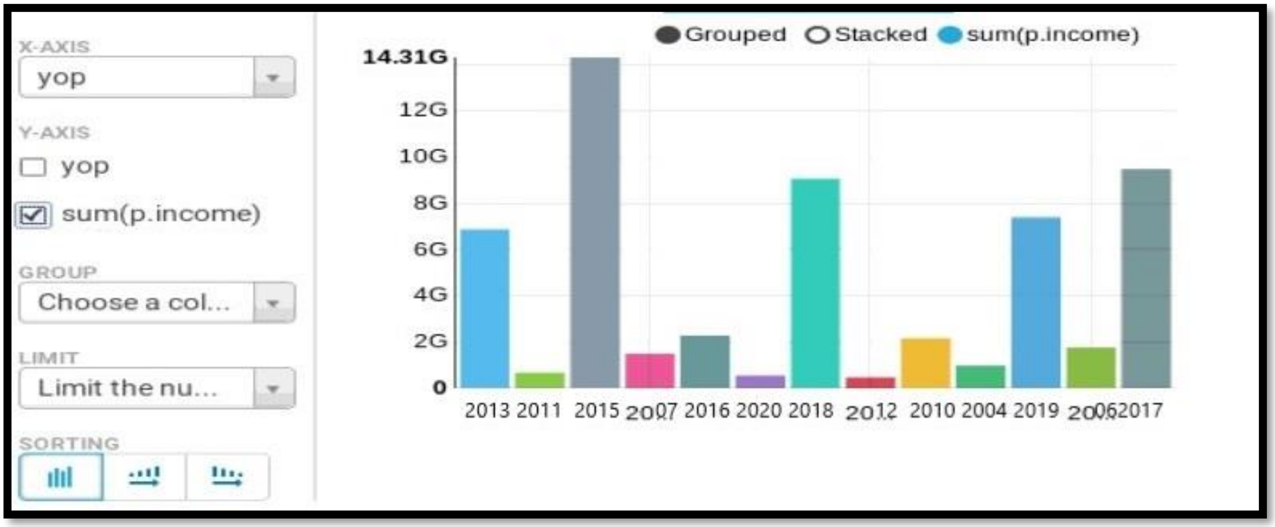
-> select m.yop,sum(p.income) from movie_table m join movieprice_table p on(m.movie_id=p.movie_id) group by m.yop;



```
select m.yop,sum(p.income)
from movie_table m join movieprice_table p
on(m.movie_id=p.movie_id)
group by m.yop;
```



	yop	sum(p.income)
1	2013	6880000000
2	2011	670000000
3	2015	14310000000
4	2007	1490000000
5	2016	2280000000
6	2020	550000000
7	2018	9070000000
8	2012	470000000
9	2010	2170000000
10	2004	980000000
11	2019	7400000000
12	2006	1760000000
13	2017	9480000000



X-AXIS
yop

Y-AXIS
□ yop
☑ sum(p.income)

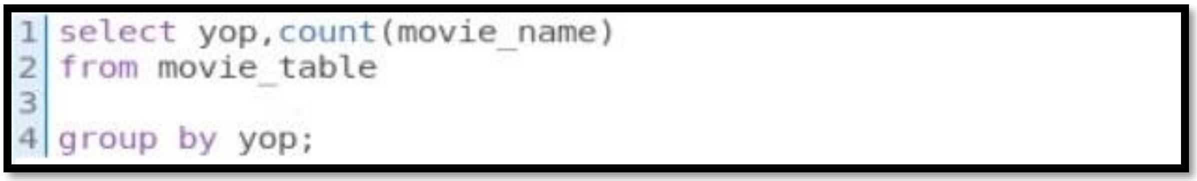
GROUP
Choose a col...

LIMIT
Limit the nu...

SORTING

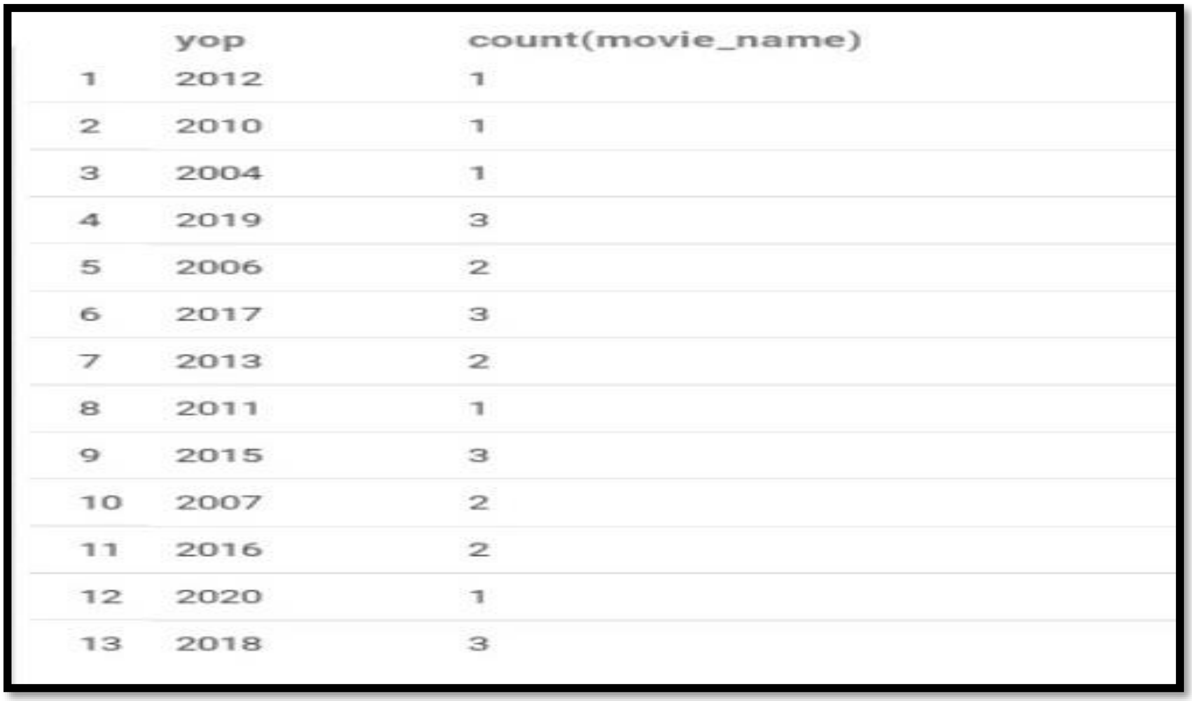
● Grouped ○ Stacked ● sum(p.income)

select yop,count(movie_name) from movie_table group by yop;

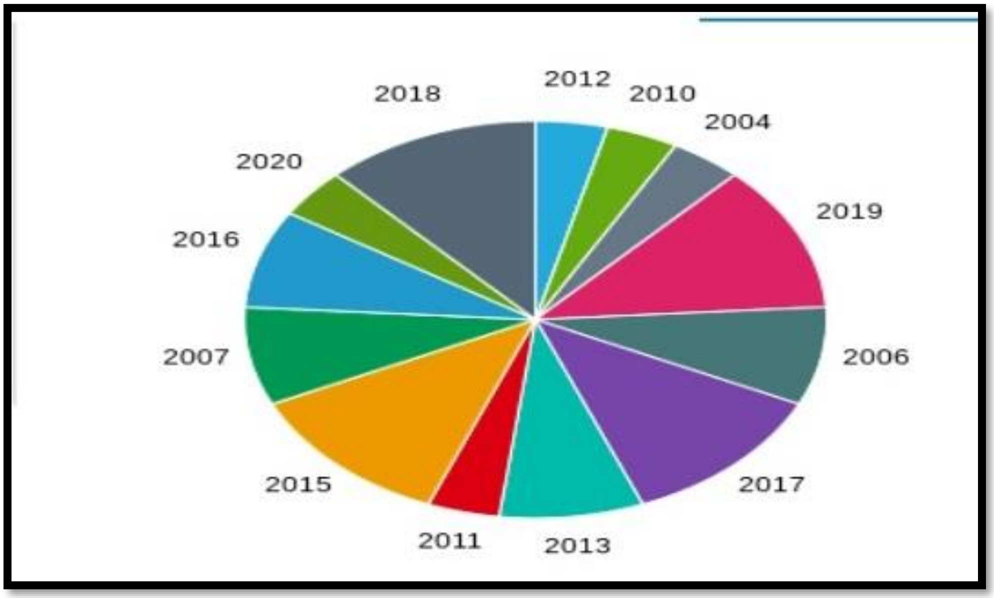


```
select yop,count(movie_name)
from movie_table

group by yop;
```



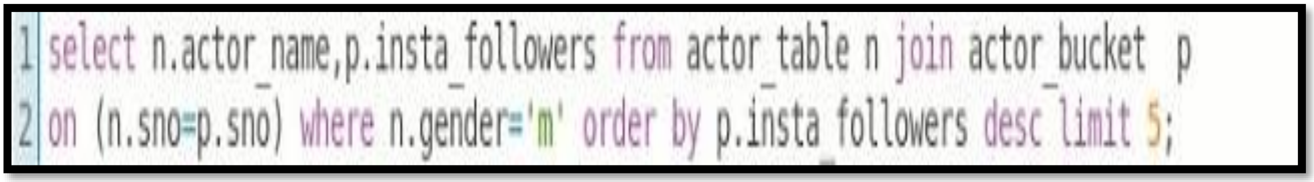
	yop	count(movie_name)
1	2012	1
2	2010	1
3	2004	1
4	2019	3
5	2006	2
6	2017	3
7	2013	2
8	2011	1
9	2015	3
10	2007	2
11	2016	2
12	2020	1
13	2018	3



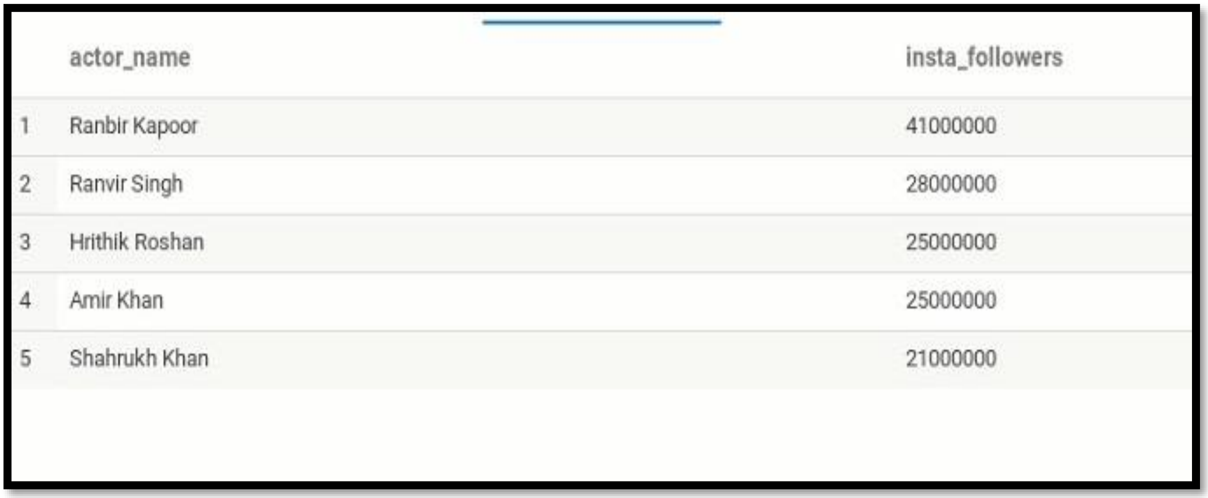
Insights: Analysis of previous data is very crucial to make new improvements in future.

Challenge 9: Which movie actor has more buzz (popularity) in social media?

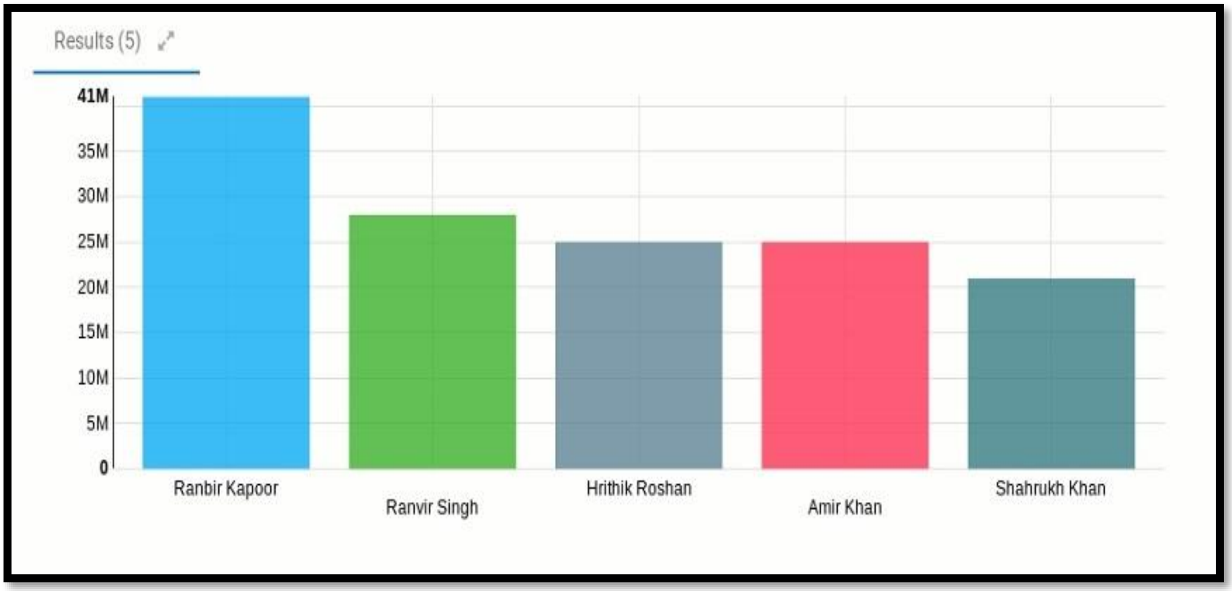
-> select n.actor_name,p.insta_followers from actor_table n join actor_per p on (n.sno=p.sno) where n.gender='m' order by p.insta_followers desc limit 5;



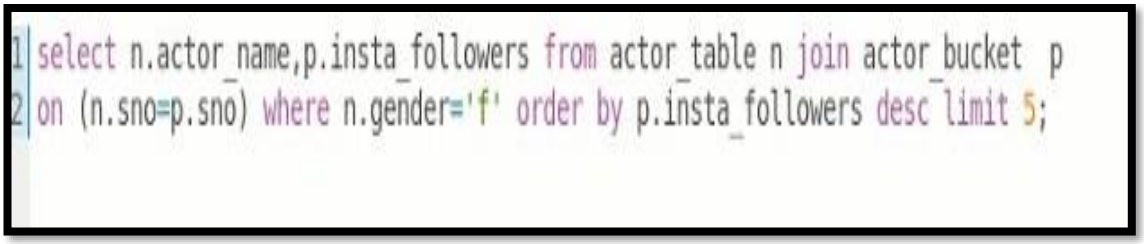
```
select n.actor_name,p.insta_followers from actor_table n join actor_bucket p
on (n.sno=p.sno) where n.gender='m' order by p.insta_followers desc limit 5;
```



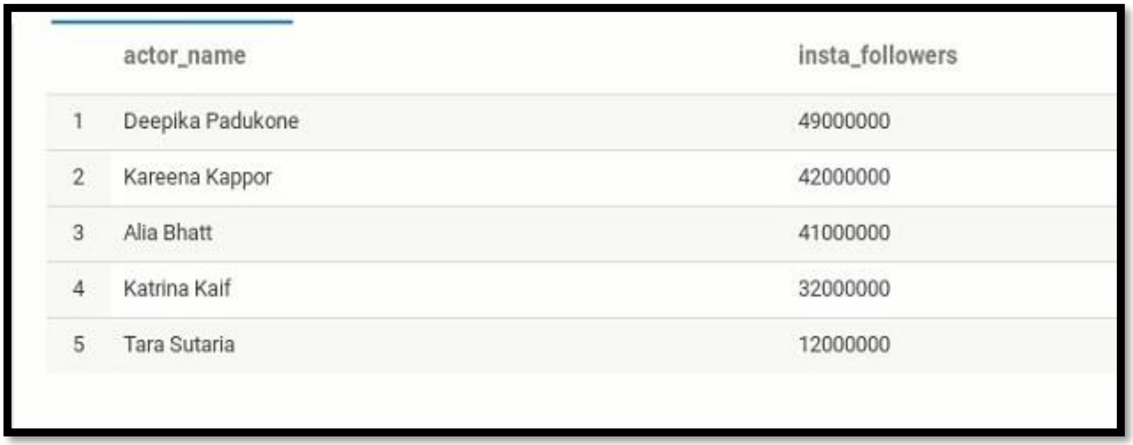
	actor_name	insta_followers
1	Ranbir Kapoor	41000000
2	Ranvir Singh	28000000
3	Hrithik Roshan	25000000
4	Amir Khan	25000000
5	Shahrukh Khan	21000000



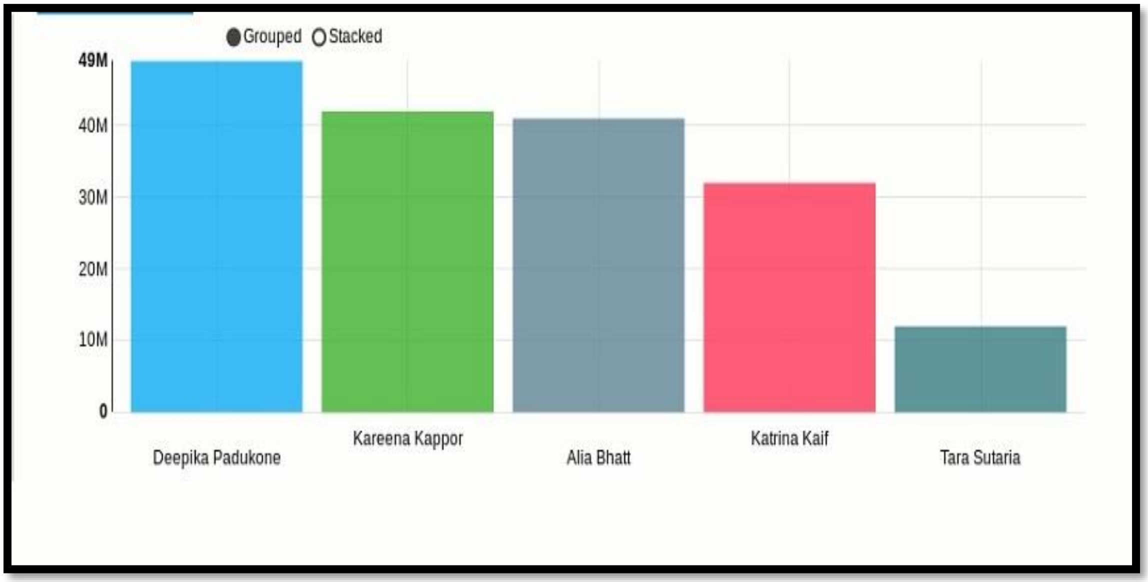
select n.actor_name,p.insta_followers from actor_table n join actor_per p on (n.sno=p.sno) where n.gender='f' order by p.insta_followers desc limit 5;



```
select n.actor_name,p.insta_followers from actor_table n join actor_bucket  p
on (n.sno=p.sno) where n.gender='f' order by p.insta_followers desc limit 5;
```



	actor_name	insta_followers
1	Deepika Padukone	49000000
2	Kareena Kappor	42000000
3	Alia Bhatt	41000000
4	Katrina Kaif	32000000
5	Tara Sutaria	12000000



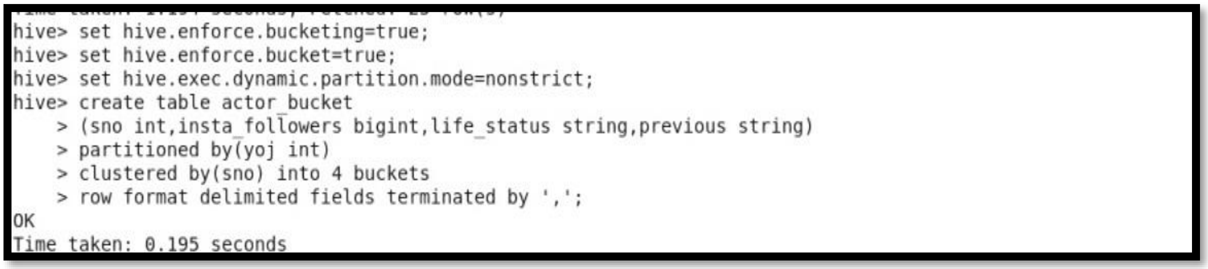
Insights: In today’s time not doing only movies is enough but connecting to your fans is also important and that’s why celebrities are interacting to the their fans through social media . The actor which has more followers means he/she is more popular.

Challenge 10: ->In film industry many actors join this field simultaneously in same year but then in their further coming years they career graph changes differently. Every actor has its own personality and stardom. So according to the year of their joining industry which movie star has more fans? Or

Classify the popular star according to their year of joining?

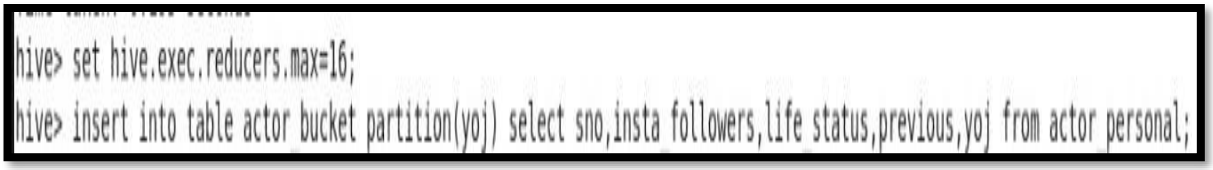
->Step 1: Set the various parameters for partitioning and bucketing.

Step 2: Create a table with partition on the basis of yoj of actors



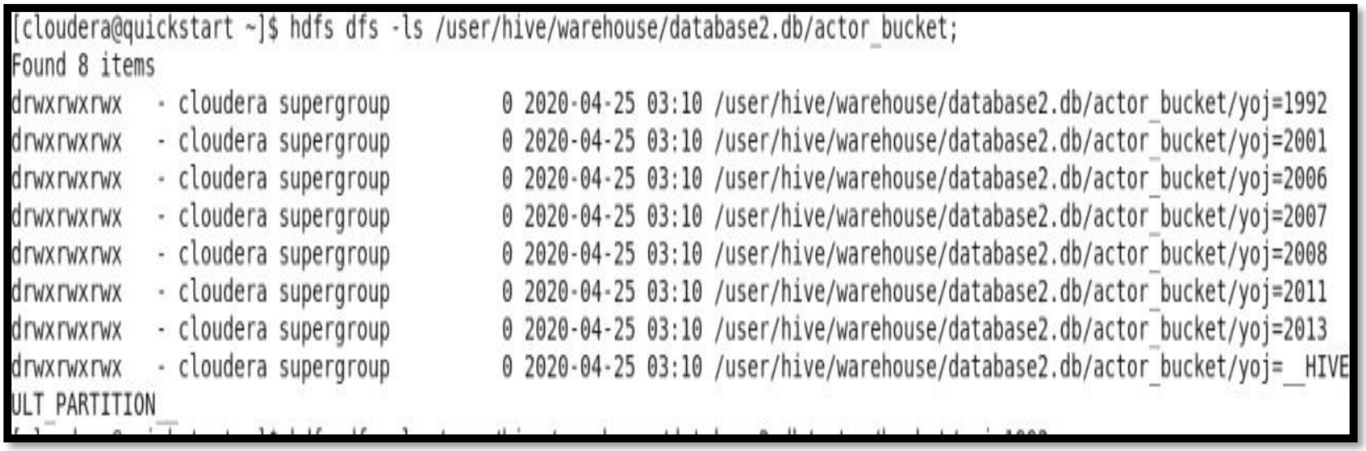
```
hive> set hive.enforce.bucketing=true;
hive> set hive.enforce.bucket=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> create table actor_bucket
    > (sno int,insta_followers bigint,life_status string,previous string)
    > partitioned by(yoj int)
    > clustered by(sno) into 4 buckets
    > row format delimited fields terminated by ',';
OK
Time taken: 0.195 seconds
```

Step 3: Load data into actor_bucket   from actor_personal;

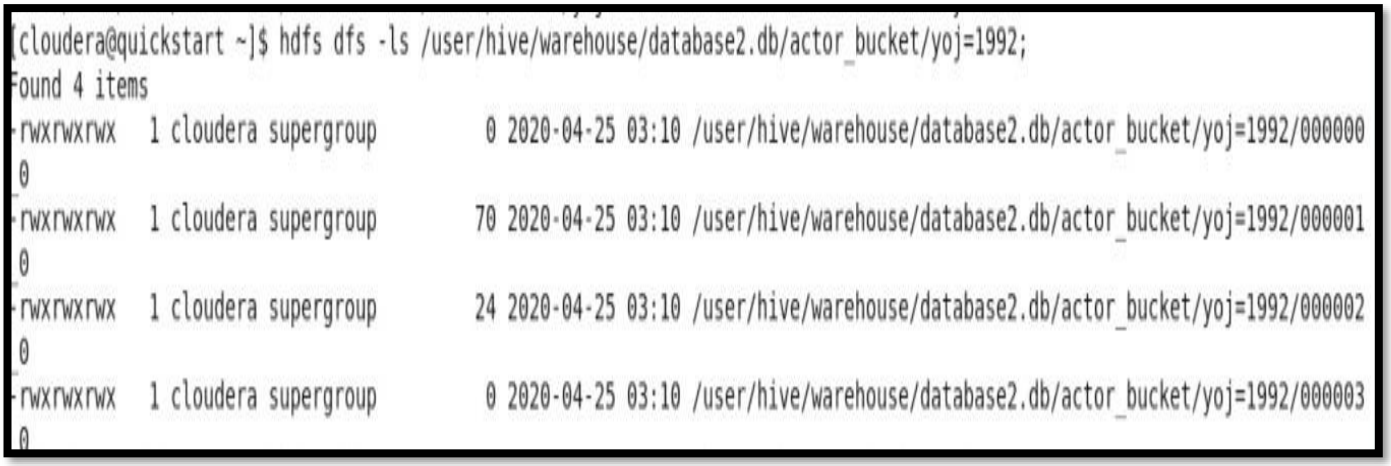


```
hive> set hive.exec.reducers.max=16;
hive> insert into table actor_bucket partition(yoj) select sno,insta_followers,life_status,previous,yoj from actor_personal;
```

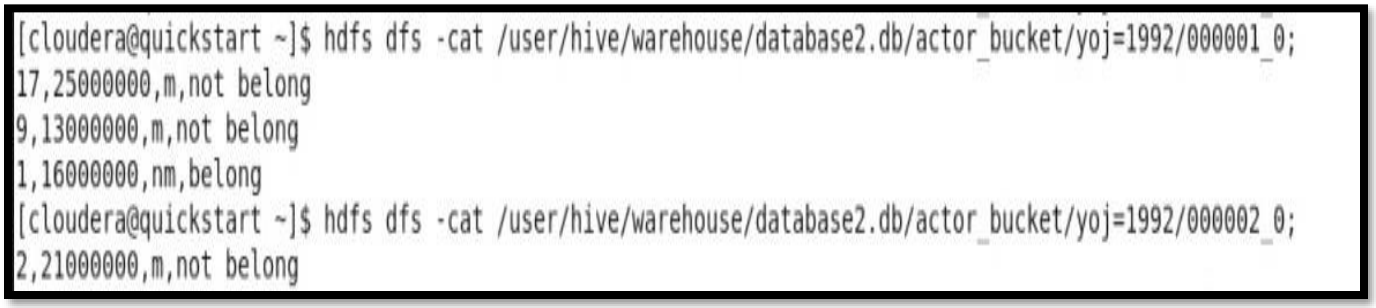
Step 4: Now we can check which actor has joined the industry in which each year and their no of followers



```
[cloudera@quickstart ~]$ hdfs dfs -ls /user/hive/warehouse/database2.db/actor_bucket;
Found 8 items
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=1992
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2001
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2006
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2007
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2008
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2011
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=2013
drwxrwxrwx   - cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=__HIVE
ULT_PARTITION__
```



```
[cloudera@quickstart ~]$ hdfs dfs -ls /user/hive/warehouse/database2.db/actor_bucket/yoj=1992;
Found 4 items
-rwxrwxrwx   1 cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000000
_0
-rwxrwxrwx   1 cloudera supergroup         70 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000001
_0
-rwxrwxrwx   1 cloudera supergroup         24 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000002
_0
-rwxrwxrwx   1 cloudera supergroup          0 2020-04-25 03:10 /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000003
_0
```

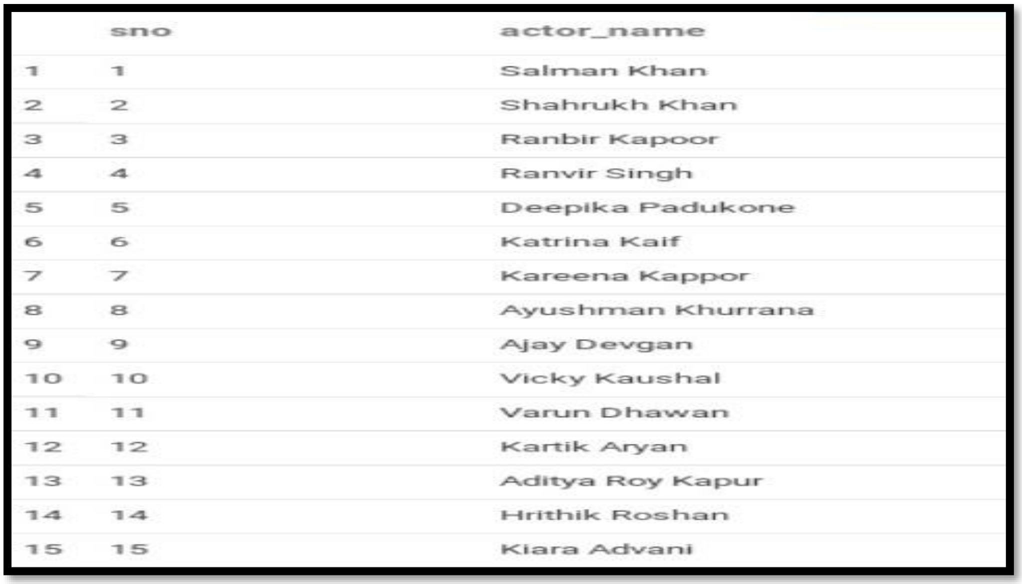


```
[cloudera@quickstart ~]$ hdfs dfs -cat /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000001_0;
17,25000000,m,not belong
9,13000000,m,not belong
1,16000000,nm,belong
[cloudera@quickstart ~]$ hdfs dfs -cat /user/hive/warehouse/database2.db/actor_bucket/yoj=1992/000002_0;
2,21000000,m,not belong
```

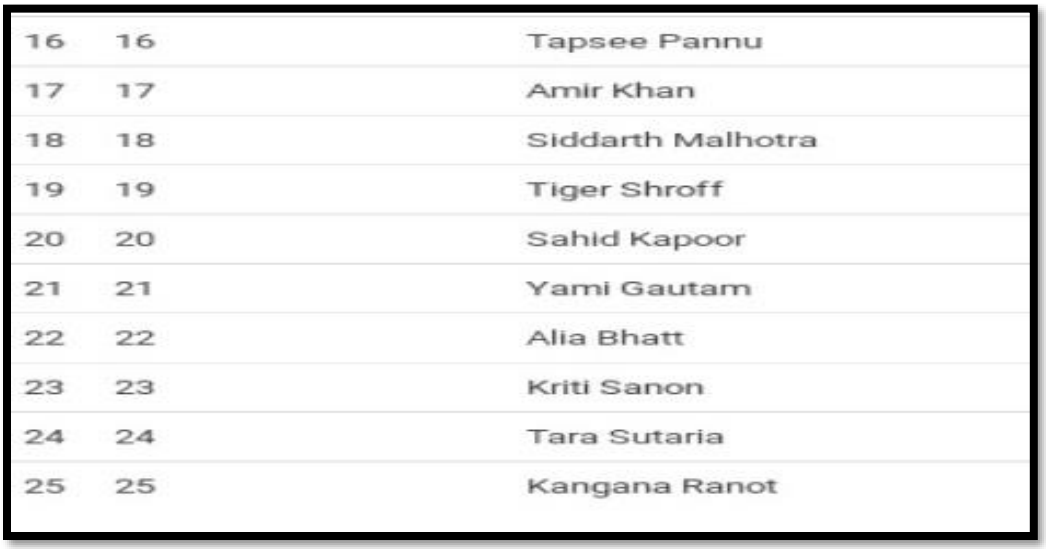
->To view which sno belong to which movie star we can use view

Create view v1 select sno,actor_name from actor_table;

Select * from actor_table;



	sno	actor_name
1	1	Salman Khan
2	2	Shahrukh Khan
3	3	Ranbir Kapoor
4	4	Ranvir Singh
5	5	Deepika Padukone
6	6	Katrina Kaif
7	7	Kareena Kappor
8	8	Ayushman Khurrana
9	9	Ajay Devgan
10	10	Vicky Kaushal
11	11	Varun Dhawan
12	12	Kartik Aryan
13	13	Aditya Roy Kapur
14	14	Hrithik Roshan
15	15	Kiara Advani



16	16	Tapsee Pannu
17	17	Amir Khan
18	18	Siddarth Malhotra
19	19	Tiger Shroff
20	20	Sahid Kapoor
21	21	Yami Gautam
22	22	Alia Bhatt
23	23	Kriti Sanon
24	24	Tara Sutaria
25	25	Kangana Ranot