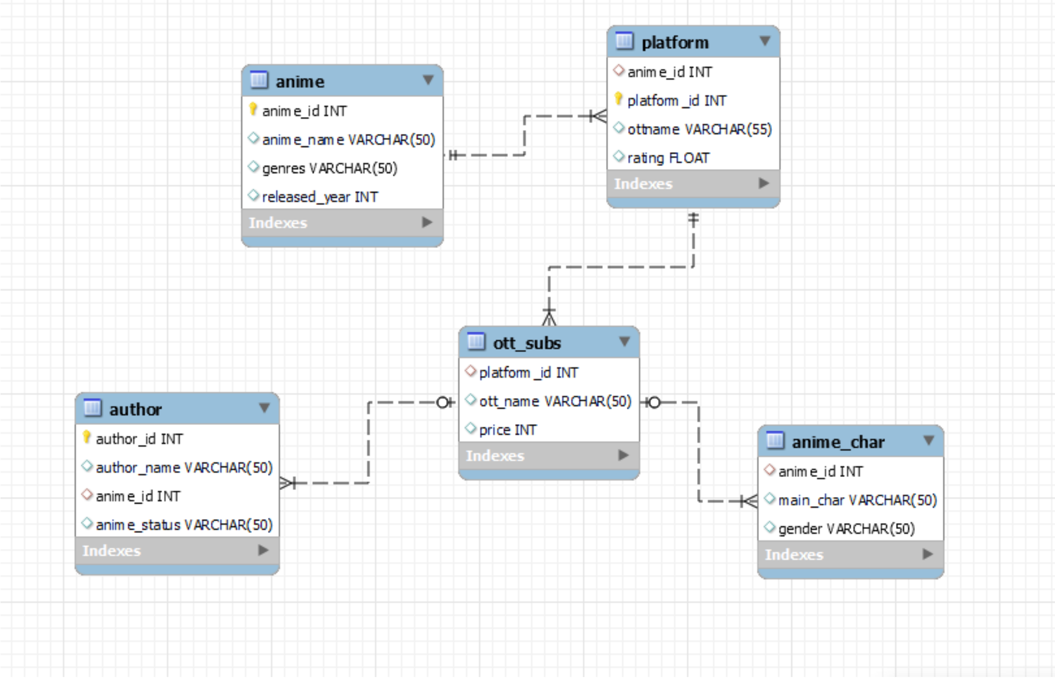
# ANIME DETAILS-CASE STUDY

I made a database for anime. Anime is the japnese style of making flim with animationswhich is enjoyed by viewers of all ages.

The database is modelled relationally as much as possible. I have made as few attributes as I can in each table. The Character_list table contains the name of the character, its gender, name of the anime it has appeared in and its rating (of the character). The Author_list table contains the name of the creator, its gender, name of the anime he/she and its rating , each table of the anime/case study database have a unique key value and the proper discryption so that the user can easily retrieve the data.

## ER(Entity Relationship) diagram:



I created the database in the MySQL workbench and I named it as "**casestudy** created the tables under the databases related to the topic '**anime**'.

Tables created in the database are

- Anime
- Author
- Platform
- Main_char
- Ott_subs

Lets start the case study with the following questions

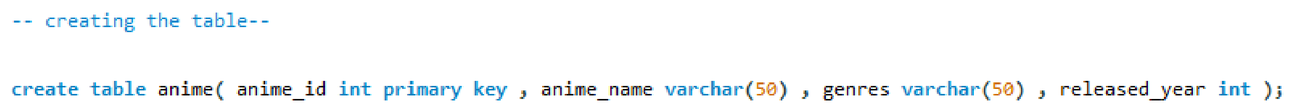
1. Creating tables.
2. Inserting values in the table.
3. Updating the values in table.
4. Creating the another table using **auto increment** in coulmns.
5. Using TCL commands like Commit and rollback to retrieve value.
6. Using aggregate functions to find sum , count, avg of coulmns.
7. List out the author name whose name contains 'misashi' in it.
8. List out all animes that are released in between 1997 and 2014(return values in ascending order)
9. Find the price of subscribing the platform for all the animes(return value by ascending order)
10. Find the price of subscribing the aniwatch platform for watching the anime
11. Retrieve the least and expensive price for each ottname.
12. Identify the author name whose anime has the most rating in the platform table.
13. Display all the anime which have the ranking above 9.8 in the platform.
14. Show the main characters of the animes with their respective anime name.
15. Show the anime name and their character name who are female.
16. List the main character name of all anime whose are male with their anime name.
17. Show the main characters of anime who have more/maximum rating.
18. Show ratings of all anime whose status are completed with their author name.

Now then lets see how the syntax are created and queries are executed:

Every questions have their queries that are executed in the workspace in the application called MySQL workbench.

## 1.creating tables

SYNTAX



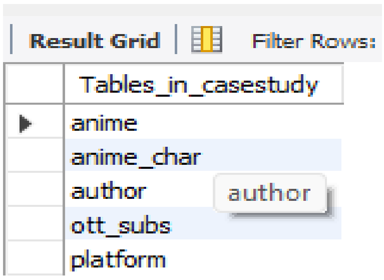
```
-- creating the table--

create table anime( anime_id int primary key , anime_name varchar(50) , genres varchar(50) , released_year int );
```

- SHOW TABLES

  It shows how many tables are created

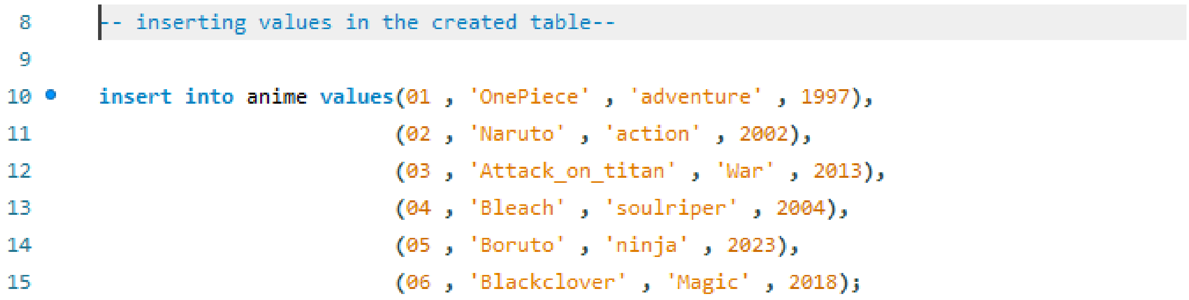
OUTPUT:



| Tables_in_casestudy |
|---|
| anime |
| anime_char |
| author |
| ott_subs |
| platform |

## 2. Inserting values in the table.

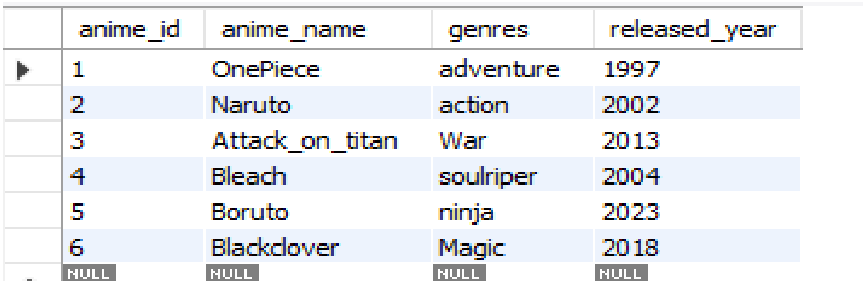
- Syntax



```
-- inserting values in the created table--

insert into anime values(01 , 'OnePiece' , 'adventure' , 1997),
                        (02 , 'Naruto' , 'action' , 2002),
                        (03 , 'Attack_on_titan' , 'War' , 2013),
                        (04 , 'Bleach' , 'soulriper' , 2004),
                        (05 , 'Boruto' , 'ninja' , 2023),
                        (06 , 'Blackclover' , 'Magic' , 2018);
```

OUTPUT



| anime_id | anime_name | genres | released_year |
|---|---|---|---|
| 1 | OnePiece | adventure | 1997 |
| 2 | Naruto | action | 2002 |
| 3 | Attack_on_titan | War | 2013 |
| 4 | Bleach | soulriper | 2004 |
| 5 | Boruto | ninja | 2023 |
| 6 | Blackclover | Magic | 2018 |
| NULL | NULL | NULL | NULL |

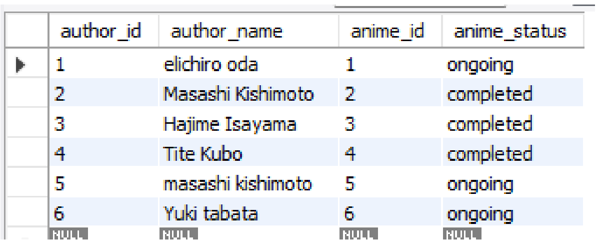
## 3. Updating the values in  table.

**Syntax**

-
  ```
  insert into author values(1 , 'elichiro oda',1,'ongoing'),
                           (2 , 'Masashi Kishimoto',2,'completed'),
                           (3 , 'Hajime Isayama ',3,'completed'),
                           (4 , 'Tite Kubo',4,'completed'),
                           (5 , 'Masahi kishimoto',5,'ongoing'),
                           (6 , 'Yuki tabata',6,'ongoing');
  -- update the valaues in tables --
  ```

-
  ```
  update author set author_name='masashi kishimoto' where author_id=5;
  ```

**OUTPUT**



| author_id | author_name | anime_id | anime_status |
|---|---|---|---|
| 1 | elichiro oda | 1 | ongoing |
| 2 | Masashi Kishimoto | 2 | completed |
| 3 | Hajime Isayama | 3 | completed |
| 4 | Tite Kubo | 4 | completed |
| 5 | masashi kishimoto | 5 | ongoing |
| 6 | Yuki tabata | 6 | ongoing |
| NULL | NULL | NULL | NULL |

## 4. Creating the another table using auto increment in coulmns

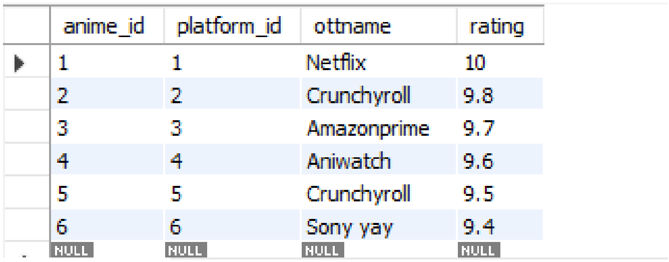
**SYNATX**

```
-- creating another table  using auto increment --
```

-
  ```
  create table platform( anime_id int, platform_id int primary key auto_increment, ottname varchar(55), rating float(2))
  ```

-
  ```
  insert into platform (anime_id , ottname  ,rating )
                      values(01,'Netflix',10),
                         (02,'Crunchyroll',9.8),
                             (03,'Amazonprime',9.7),
                             (04,'Aniwatch',9.6),
                             (05,'Crunchyroll',9.5),
                             (06,'Sony yay',9.4);
  ```

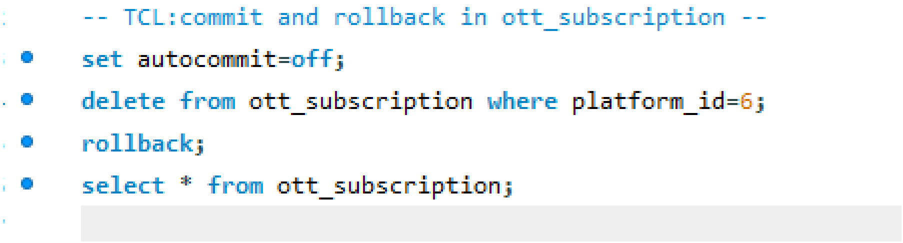
OUTPUT



| anime_id | platform_id | ottname | rating |
|---|---|---|---|
| 1 | 1 | Netflix | 10 |
| 2 | 2 | Crunchyroll | 9.8 |
| 3 | 3 | Amazonprime | 9.7 |
| 4 | 4 | Aniwatch | 9.6 |
| 5 | 5 | Crunchyroll | 9.5 |
| 6 | 6 | Sony yay | 9.4 |
| NULL | NULL | NULL | NULL |

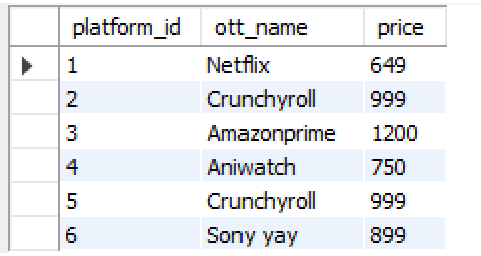
## 5.Using TCL commands like Commit and rollback to retrieve value.

SYNTAX



```
-- TCL:commit and rollback in ott_subscription --
set autocommit=off;
delete from ott_subscription where platform_id=6;
rollback;
select * from ott_subscription;
```

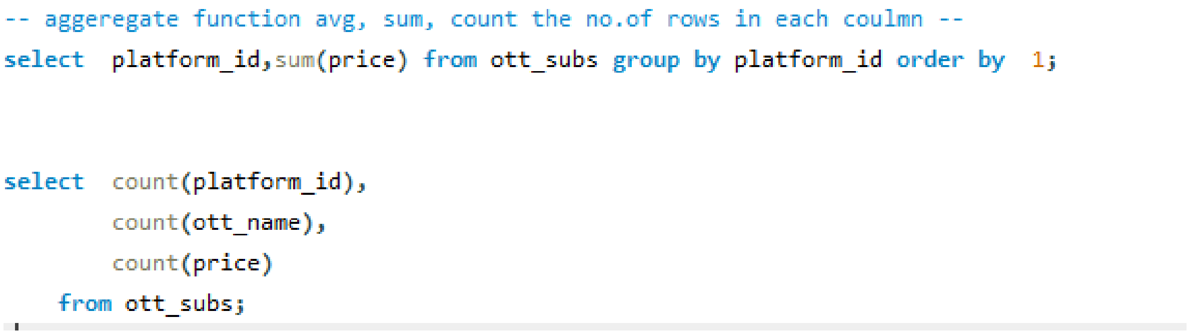
OUTPUT



| platform_id | ott_name | price |
|---|---|---|
| 1 | Netflix | 649 |
| 2 | Crunchyroll | 999 |
| 3 | Amazonprime | 1200 |
| 4 | Aniwatch | 750 |
| 5 | Crunchyroll | 999 |
| 6 | Sony yay | 899 |

## 6. Using aggregate functions to find sum,avg and count of each rows.

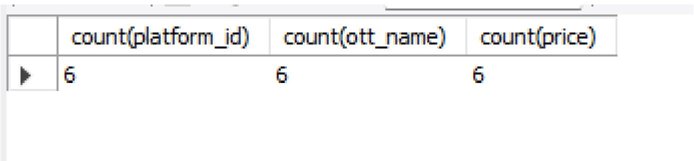
SYNATX



```
-- aggeregate function avg, sum, count the no.of rows in each coulmn --
select  platform_id,sum(price) from ott_subs group by platform_id order by  1;


select  count(platform_id),
        count(ott_name),
        count(price)
    from ott_subs;
```

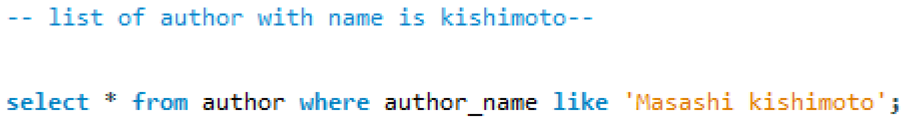
OUTPUT



| count(platform_id) | count(ott_name) | count(price) |
|---|---|---|
| 6 | 6 | 6 |

## 7. List out the author name whose name contains ‘misashi’ in it

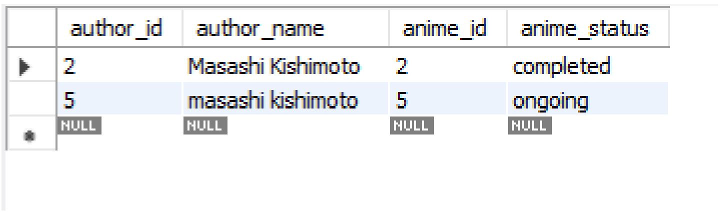
## SYNTAX



```
-- list of author with name is kishimoto--

select * from author where author_name like 'Masashi kishimoto';
```

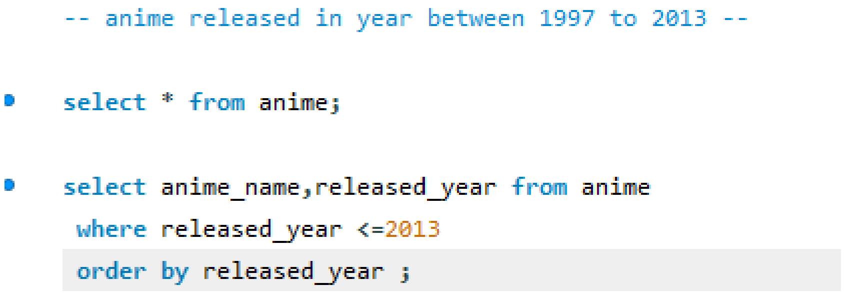
OUTPUT



| author_id | author_name | anime_id | anime_status |
|---|---|---|---|
| 2 | Masashi Kishimoto | 2 | completed |
| 5 | masashi kishimoto | 5 | ongoing |
| NULL | NULL | NULL | NULL |

## 8.List out all animes that are released in between 1997 and 2014(return values in ascending order)

SYNTAX

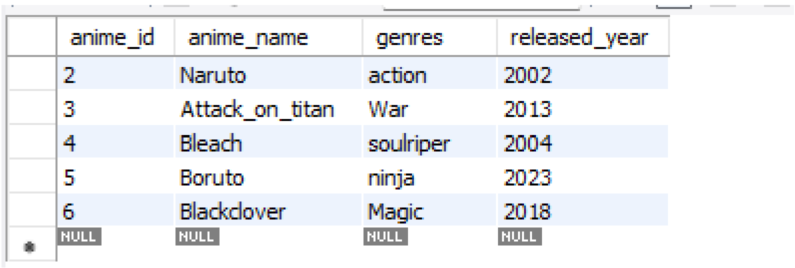


```
-- anime released in year between 1997 to 2013 --

select * from anime;

select anime_name,released_year from anime
 where released_year <=2013
 order by released_year ;
```

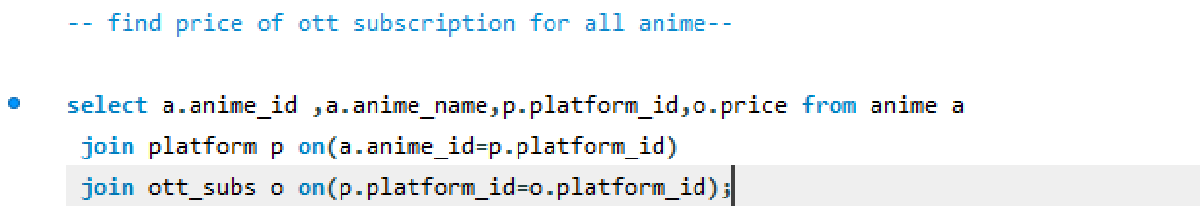
OUTPUT



| anime_id | anime_name | genres | released_year |
|---|---|---|---|
| 2 | Naruto | action | 2002 |
| 3 | Attack_on_titan | War | 2013 |
| 4 | Bleach | soulriper | 2004 |
| 5 | Boruto | ninja | 2023 |
| 6 | Blackclover | Magic | 2018 |
| NULL | NULL | NULL | NULL |

## 9. Find the price of subscribing the platform for all the animes(return value by ascending order)

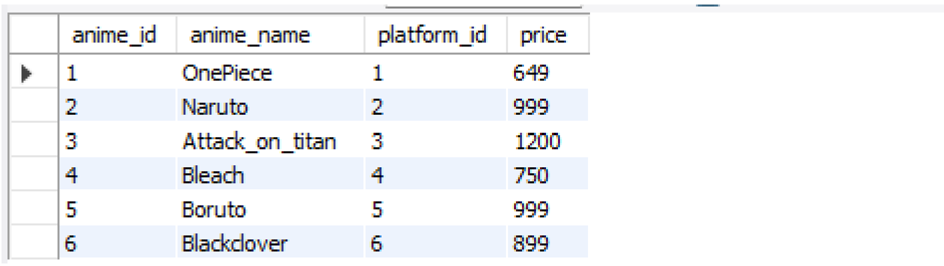
SYNTAX



```
-- find price of ott subscription for all anime--

select a.anime_id ,a.anime_name,p.platform_id,o.price from anime a
 join platform p on(a.anime_id=p.platform_id)
 join ott_subs o on(p.platform_id=o.platform_id);
```

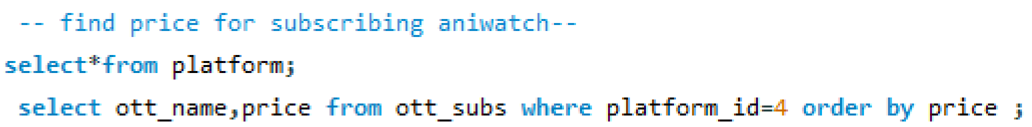
OUTPUT



| anime_id | anime_name | platform_id | price |
|---|---|---|---|
| 1 | OnePiece | 1 | 649 |
| 2 | Naruto | 2 | 999 |
| 3 | Attack_on_titan | 3 | 1200 |
| 4 | Bleach | 4 | 750 |
| 5 | Boruto | 5 | 999 |
| 6 | Blackclover | 6 | 899 |

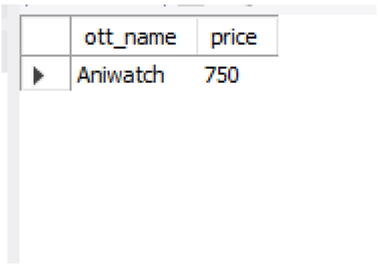
## 10. Find the price of subscribing the aniwatch platform for watching the anime

SYNTAX



```
 -- find price for subscribing aniwatch--
select*from platform;
 select ott_name,price from ott_subs where platform_id=4 order by price ;
```

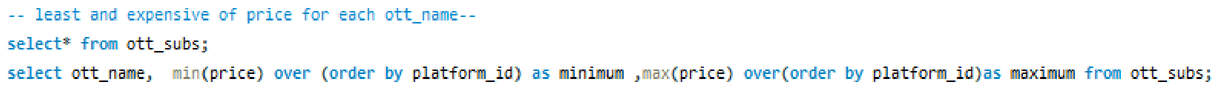
OUTPUT



| ott_name | price |
|---|---|
| Aniwatch | 750 |

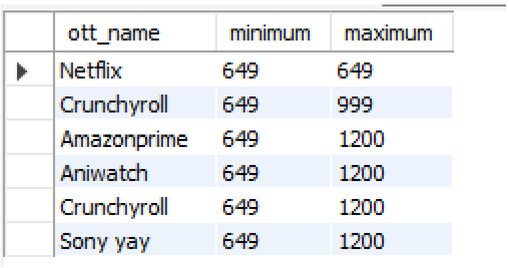
## 11. Retrieve the least and expensive price for each ottname.

SYNTAX



```
-- least and expensive of price for each ott_name--
select* from ott_subs;
select ott_name,  min(price) over (order by platform_id) as minimum ,max(price) over(order by platform_id)as maximum from ott_subs;
```

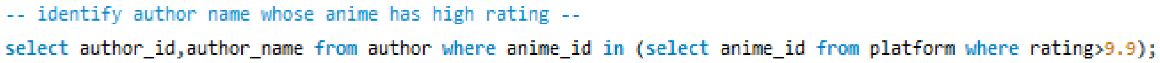
OUTPUT



| ott_name | minimum | maximum |
|---|---|---|
| Netflix | 649 | 649 |
| Crunchyroll | 649 | 999 |
| Amazonprime | 649 | 1200 |
| Aniwatch | 649 | 1200 |
| Crunchyroll | 649 | 1200 |
| Sony yay | 649 | 1200 |

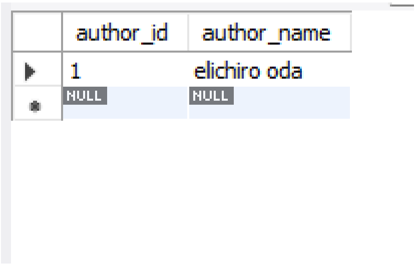
## 12. Identify the author name whose anime has the most rating in the platform table.

SYNTAX



```
-- identify author name whose anime has high rating --
select author_id,author_name from author where anime_id in (select anime_id from platform where rating>9.9);
```

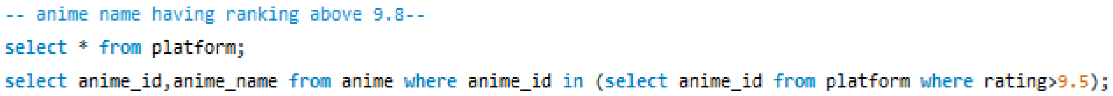
OUTPUT



| author_id | author_name |
|---|---|
| 1 | elichiro oda |
| NULL | NULL |

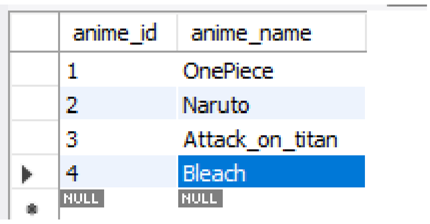
## 13.Display all the anime which have the ranking above 9.8 in the platform.

SYNTAX



```
-- anime name having ranking above 9.8--
select * from platform;
select anime_id,anime_name from anime where anime_id in (select anime_id from platform where rating>9.5);
```

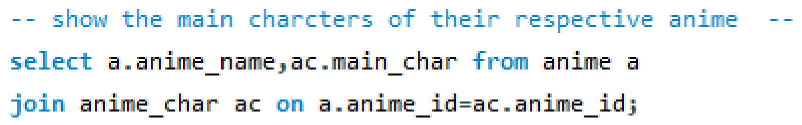
OUTPUT



| anime_id | anime_name |
|---|---|
| 1 | OnePiece |
| 2 | Naruto |
| 3 | Attack_on_titan |
| 4 | Bleach |
| NULL | NULL |

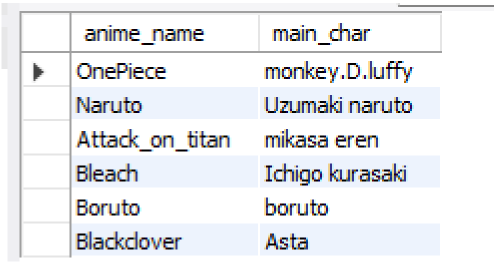
## 14.Show the main characters of the animes with their respective anime name.

SYNTAX



```
-- show the main charcters of their respective anime  --
select a.anime_name,ac.main_char from anime a
join anime_char ac on a.anime_id=ac.anime_id;
```

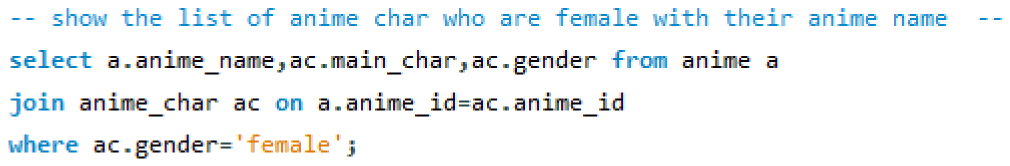
OUTPUT



| | anime_name | main_char |
|---|---|---|
| ▶ | OnePiece | monkey.D.luffy |
| | Naruto | Uzumaki naruto |
| | Attack_on_titan | mikasa eren |
| | Bleach | Ichigo kurasaki |
| | Boruto | boruto |
| | Blackclover | Asta |

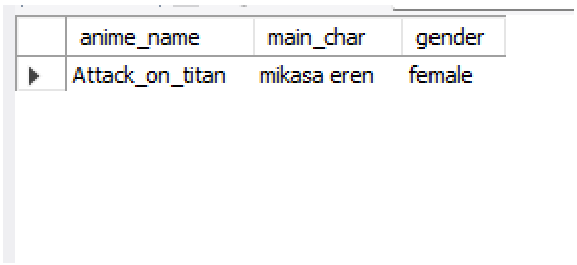
## 15. Show the  anime name and their character name who are female.

SYNTAX



```
-- show the list of anime char who are female with their anime name  --
select a.anime_name,ac.main_char,ac.gender from anime a
join anime_char ac on a.anime_id=ac.anime_id
where ac.gender='female';
```

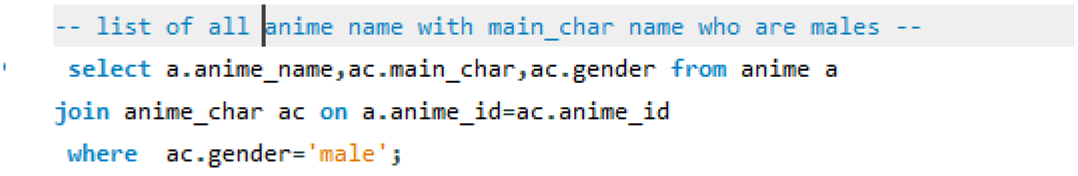
OUTPUT



| | anime_name | main_char | gender |
|---|---|---|---|
| ▶ | Attack_on_titan | mikasa eren | female |

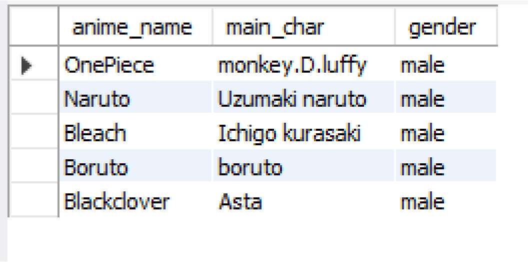
## 16. List the main character name of all anime whose are male with their anime name.

SYNTAX



```
-- list of all anime name with main_char name who are males --
 select a.anime_name,ac.main_char,ac.gender from anime a
join anime_char ac on a.anime_id=ac.anime_id
 where  ac.gender='male';
```

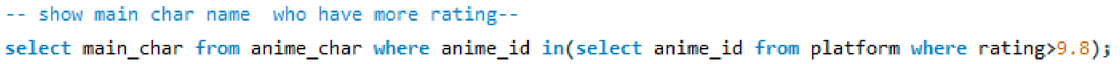
OUTPUT



| anime_name | main_char | gender |
|---|---|---|
| OnePiece | monkey.D.luffy | male |
| Naruto | Uzumaki naruto | male |
| Bleach | Ichigo kurasaki | male |
| Boruto | boruto | male |
| Blackclover | Asta | male |

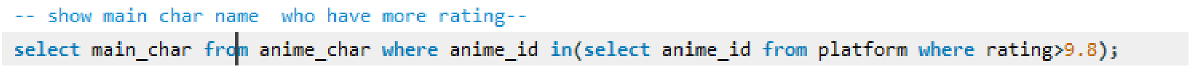
## 17. Show the main characters of anime who have more/maximum rating.

SYNTAX



```
-- show main char name  who have more rating--
select main_char from anime_char where anime_id in(select anime_id from platform where rating>9.8);
```

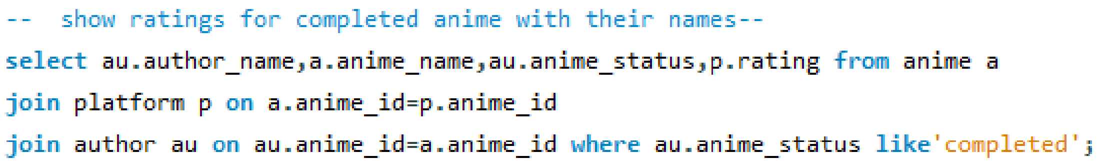
OUTPUT



```
-- show main char name  who have more rating--
select main_char from anime_char where anime_id in(select anime_id from platform where rating>9.8);
```

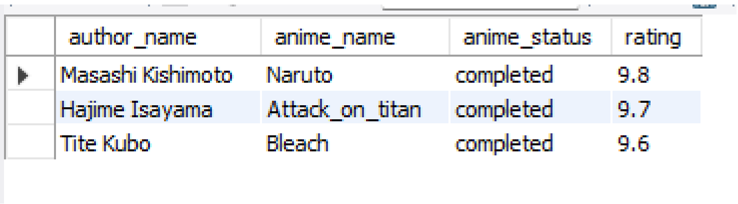
## 18.Show ratings of all anime whose status are completed with their author name.

SYNTAX



```
--  show ratings for completed anime with their names--
select au.author_name,a.anime_name,au.anime_status,p.rating from anime a
join platform p on a.anime_id=p.anime_id
join author au on au.anime_id=a.anime_id where au.anime_status like'completed';
```

OUTPUT



| author_name | anime_name | anime_status | rating |
|---|---|---|---|
| Masashi Kishimoto | Naruto | completed | 9.8 |
| Hajime Isayama | Attack_on_titan | completed | 9.7 |
| Tite Kubo | Bleach | completed | 9.6 |