Faculty of Computing &
Information Technology

Database Systems

SE Fall 21

Project Report

Submitted to

Sir Asif Sohail

Submitted By

BSEF21M037 –MUHAMMAD SAADULLAH

BSEF21M040 – MUHAMMAD AHMAD

Online movie database system

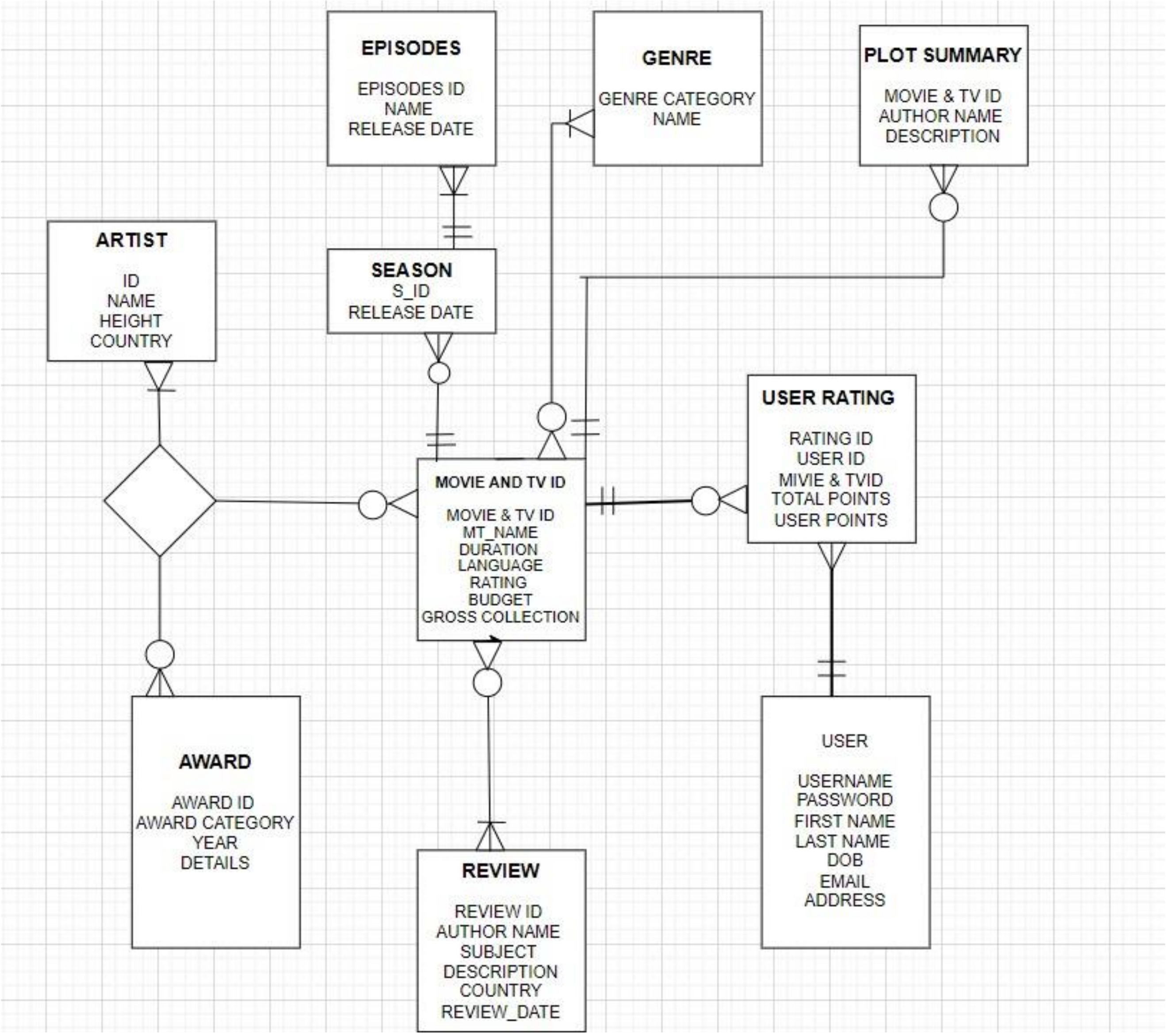
INTRODUTION TO SYSTEM:

Online Movie Database System is an online database of information related to films, television series, and streaming content online including cast, production crew and personal biographies, ratings, and critical reviews. It serves as a valuable resource for movie enthusiasts, professionals in the entertainment industry and general audiences seeking information about their favorite films and TV shows.

Here's an introduction to the workings of ONLINE MOVIE DATABASE MANAGEMET SYSTEM

- Users can **browse movies and TV shows**, view details such as cast and crew, production information, plot summaries, and user ratings.
- **Artists, production houses, and users** are registered in the system with their respective details.
- **Movies** are associated with genres and various **artists**, including **actors, directors, writers, and producers**.
- Users can rate movies, and the system tracks **user ratings** and **awards** received by movies.
- The project allows for easy retrieval of information such as **top-rated movies**, **artist profiles**, and **award-winning films.**

Entity Relationships diagram



Normalization

Movie_and_tv(movi_tv_id, movi_tv_name, duration, language, Rating, Budget, gross_collection, genre_id, genre_category_name, plot_summary, author_name, sum_desc, Review_id, review_subject, author_name, desc, movie_rating_id, movie_rating, noOfVotes, Award_id, Award_category, award_year, actor_id, actor_name, actor_dob, actor_height, actor_country, writer_id, writer_name, writer_dob, writer_height, writer_country, director_id, director_name, director_dob, director_height, director_country, producer_id, producer_name, producer_dob, producer_height, producer_country, tv_season_id, tv_season_name, tv_s_release_date, episode_id, episode_name ,ep_release_date, user_id, user_passward, user_name, user_dob, user_email, user_address, user_rating,)

After performing the normalization of above main table when the repeating groups are removed so the above table splits into below tables like…

1NF

Movie_actor(Movie_TVID, actor_id, actor_name, actor_dob, height, country, type, gender, bio)

Movie_plot_summ(Movie_TVID, summary_id, author_name,sum_desc)

Movie_director(Movie_TVID, dir_id, award_id, name, dob, height, country,award_year)

Movie_writer(Movie_TVID, writer_id ,award_id, name, dob, height, country, award_year)

Movie_producer(Movie_TVID, producer_id,award_id, name, dob, height, country, award_year)

Movie_and_tv(Movie_TVID,movie_tv_name,duration,language, budget, gross_collection)

Movie_rating(Movie_tvID, rating, noOfVotes)

Movie_genre(Movie_TVID, genre_id, genre_name)

User_rating(Movie_TVID,user_id, user_name, dob, email, address, user_rating, total_points,user_points)

Season(Movie_TVID, season_id, episode_id,season_no, season_name, episode_name, epi_release_date)

2NF

As there are same attributes in producer , writer , actor and director so we can combine it to form one single relation artist, That is parent relation. After performing 2NF the finalized tables are present on the other page.

3NF

All tables are already in 3NF. Because Extends the normalization process by eliminating transitive dependencies. A transitive dependency exists when a non-prime (non-key) attribute depends on another non-prime attribute, rather than depending directly on the primary key. To achieve 3NF, a table must be in 2NF, and there should be no transitive dependencies.

Finalized relations

Artist(*artist_id, actor_name, actor_dob, height, country, gender, bio)*

Movie_actor(Movie_TVID, actor_id,character_name)

Movie_director(Movie_TVID, dir_id)

Movie_writer(Movie_TVID, writer_id)

Movie_producer (Movie_TVID, producer_id)

Award(award_id, award_category_name)

Movie_artist_award(artist_id,movi_tvID,year)

Movie_plot_summ(Movie_TVID, summary_id, author_Fname, Lname,sum_desc)

Movie_review(movie_tviD, review_subject, author_name, desc)

Movie_and_tv(Movie_TVID,movie_tv_name,duration,language, budget, gross_collection)

Movie_rating(rating_id,Movie_tvID, rating, noOfVotes)

Movie_genre(Movie_TVID, genre_id)

Genre(genre_id, genre_name)

User_rating(Movie_TVID, user_rating, total_points,user_points)

user(user_id, user_name, dob, email, address)

TV_Season(Movie_tvID, season_id , season_no, season_name)

TV_episodes(season_id, episode_name, release_date)

All tables are upto 3NF because all of them meet successfully the conditions of 3NF. The table is in 3NF when it is already in 1NF and 2NF. So all of the above tables are satisfying the condition of no repeating group , no partial dependency and no transitive dependency among the non_key attributes of all the relations.

Relations Description:

ARTIST:-

Attribute	*Data Type*	*Size*	*Constraints*
Artist_id	*number*	*5*	*primary key*
Artist_name	*Varchar2*	*20*	*Not null*
dob	*date*		
height	*Number*	*2*	
country	*Varchar2*	*20*	
gender	*char*	*1*	*Check(gender in (‘M’,’F’N’)*

moviedirector

Attribute	*Data Type*	*Size*	*Constraints*
Movie_tv_ID	*number*	*3*	*FK references movie_tv, PK*
Artist_id	*number*	*3*	*FK references artist*

moviedirector

Attribute	*Data Type*	*Size*	*Constraints*
Movie_tvID	*number*	*3*	*FK references movie_tv,PK*
Artist_id	*number*	*3*	*FK references artist,PK*

movieProducer

Attribute	*Data Type*	*Size*	*Constraints*
Movie_tvID	*number*	*3*	*PK, FK references movie_tv*
Artist_id	*number*	*3*	*PK , FK references artist*

moviewriter

Attribute	*Data Type*	*Size*	*Constraints*
Movie_tvID	*number*	*3*	*PK, FK references movie_tv*
Artist_id	*number*	*3*	*PK, FK references artist*

Movie_tv

Attribute	*Data Type*	*Size*	*Constraints*
Movie_tvID	*number*	*3*	*Primary key ,*
name	*Varchar2*	*20*	*Not null*

type	*char*	*5*	*check(type in('MOVIE','TV')),*
Duration_min	*number*	*5*	
language	*Varchar2*	*20*	
Budget_us	*number*	*10*	
originCountry	*Varchar2*	*20*	
Gross_collection	*Varchar2*	*50*	

Genre:-

Attribute	*Data Type*	*Size*	*Constraints*
Genre_id	*number*	*3*	*Primary key*
Genre_categoryName	*Varchar2*	*20*	*Not null*

Movie_genre:-

Attribute	*Datatype*	*Size*	*Constraints*
Movie_TVID	*number*	*3*	*PK, FK references movie_tv*
Genre_id	*number*	*3*	*PK, FK references genre*

Plot summary

Attribute	*Data Type*	*Size*	*Constraints*
Plot_summaryID	*number*	*3*	*Primary key*
Movie_tvID	*number*	*3*	*FK references movie_tv*
authorName	*Varchar2*	*20*	*Not null*
description	*Varchar2*	*200*	

Movie_TVReview

Attribute	*Data Type*	*Size*	*Constraints*
ReviewID	*number*	*3*	*Primary key*
Movie_tvID	*number*	*3*	*FK references movie_tv*
subject	*Varchar2*	*20*	*Not null*
authorname	*Varchar2*	*20*	
description	*Varchar2*	*200*	
country	*Varchar2*	*20*	
reviewDate	*date*		

User_rating

Attribute	*Data Type*	*Size*	*Constraints*
User_ratingID	*number*	*3*	*PK, FK references movie_tv*
User_id	*number*	*3*	*FK references user_record*
Movie_tvID	*Number3*	*3*	*Not null*

Rating	number	2	Not null

User record:-

Attribute	Data Type	Size	Constraints
User_id	number	3	Primary key
User_name	Varchar2	20	Not null
email	Varchar2	30	check(email like '%@email.com'),
country	Varchar2	20	

Award:-

Attribute	Data Type	Size	Constraints
Award_id	number	3	Primary key
Category_name	Varcahr2	20	Not null

Movie_artist_award

Attribute	Data Type	Size	Constraints
Movie_tvID	number	3	FK references movie_tv,PK
Artist_id	number	3	FK references artist ,PK
Award_id	number	3	FK references award ,PK
Award_date	date		

TV_season

Attribute	Data Type	Size	Constraints
Tv_season_id	number	3	PK
Movie_tvID	number	3	FK references movie_tv
Season_no	number	2	Not null
Season_name	Varchar2	20	

TV_EPISODE

Attribute	Data Type	Size	Constraints
TV_EPISODE_ID	NUMBER	3	PK
SEASON_ID	NUMBER	3	FK references tv_season
TV_EPISODEnAME	VARCHAR2	20	
RELEASE_DATE	DATE		

Create table commands

```
create table artist(
artist_id number(5) constraint art_artID_pk primary key,
artist_name varchar2(20) constraint art_artName_nk not null,
dob date,
country varchar2(20),
gender char(1) constraint art_gender_ck check(gender in('M' , 'F' , 'N')),
height number(2),
bio varchar2(200)
);
```

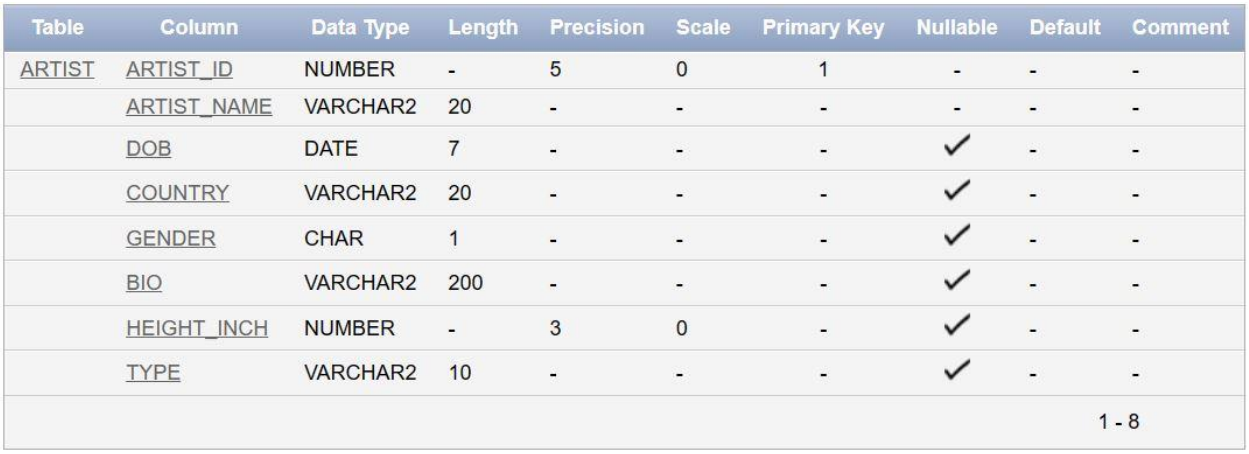


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ARTIST	ARTIST_ID	NUMBER	-	5	0	1	-	-	-
	ARTIST_NAME	VARCHAR2	20	-	-	-	-	-	-
	DOB	DATE	7	-	-	-	✓	-	-
	COUNTRY	VARCHAR2	20	-	-	-	✓	-	-
	GENDER	CHAR	1	-	-	-	✓	-	-
	BIO	VARCHAR2	200	-	-	-	✓	-	-
	HEIGHT_INCH	NUMBER	-	3	0	-	✓	-	-
	TYPE	VARCHAR2	10	-	-	-	✓	-	-

1 - 8

```
create table movieActor(
movie_tvID number(3),
artist_id number(3),
character_name varchar2(20),
constraint movie_actor_movie_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tvID),
constraint movie_actor_artist_id_fk foreign key(artist_id) references artist(artist_id),
constraint mov_actor_mov_tvID_artID_pk primary key(movie_tvID, artist_id)
)
```

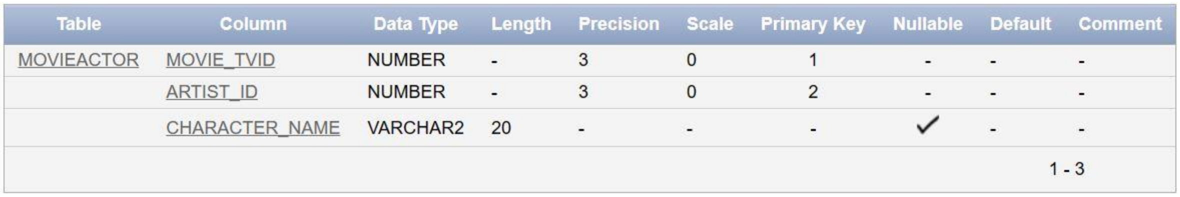


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIEACTOR	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	ARTIST_ID	NUMBER	-	3	0	2	-	-	-
	CHARACTER_NAME	VARCHAR2	20	-	-	-	✓	-	-
								1 - 3	

```
create table moviedirector(
movie_tvID number(3),
artist_id number(3),
constraint movie_dir_movie_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tvID),
constraint movie_dir_artist_id_fk foreign key(artist_id) references artist(artist_id),
constraint mov_dir_mov_tvID_artID_pk primary key(movie_tvID, artist_id)
)
```

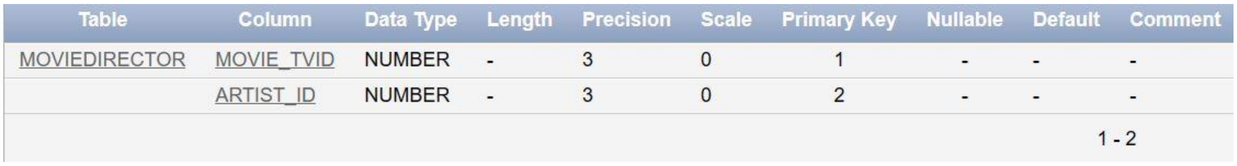


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIEDIRECTOR	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	ARTIST_ID	NUMBER	-	3	0	2	-	-	-
								1 - 2	

```
create table movieProducer(
movie_tvID number(3),
artist_id number(3),
constraint movie_pro_movie_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tvID),
constraint movie_pro_artist_id_fk foreign key(artist_id) references artist(artist_id),
constraint mov_pro_mov_tvID_artID_pk primary key(movie_tvID, artist_id)
)
```

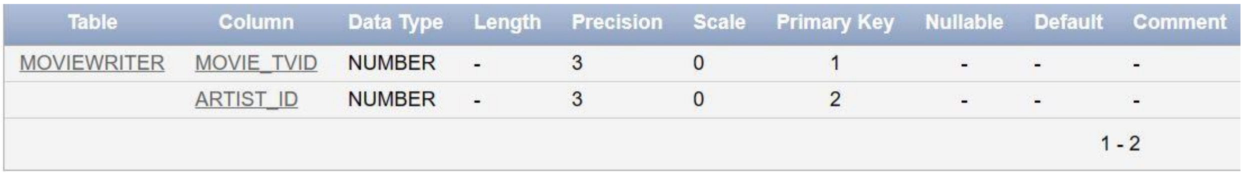


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIEWRITER	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	ARTIST_ID	NUMBER	-	3	0	2	-	-	-
								1 - 2	

```
create table moviewriter(
movie_tvID number(3),
artist_id number(3),

constraint movie_wri_movie_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tvID),

constraint movie_wri_artist_id_fk foreign key(artist_id) references artist(artist_id),

constraint mov_wri_mov_tvID_artID_pk primary key(movie_tvID, artist_id)

)
```

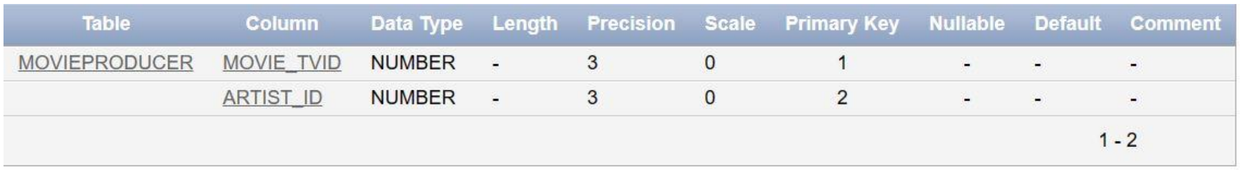


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIEPRODUCER	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	ARTIST_ID	NUMBER	-	3	0	2	-	-	-
									1 - 2

```
create table movie_TV(

movie_tvID number(3) constraint mov_tv_mov_tvID_pk primary key,

name varchar2(20) constraint mov_tv_name_nk not null,

type char(1) constraint mov_tv_type check(type in('MOVIE','TV')),

duration_min number(5),

language varchar2(20),

budget_$ number(10),

originCountry varchar2(20),

gross_collection number(10)

);
```

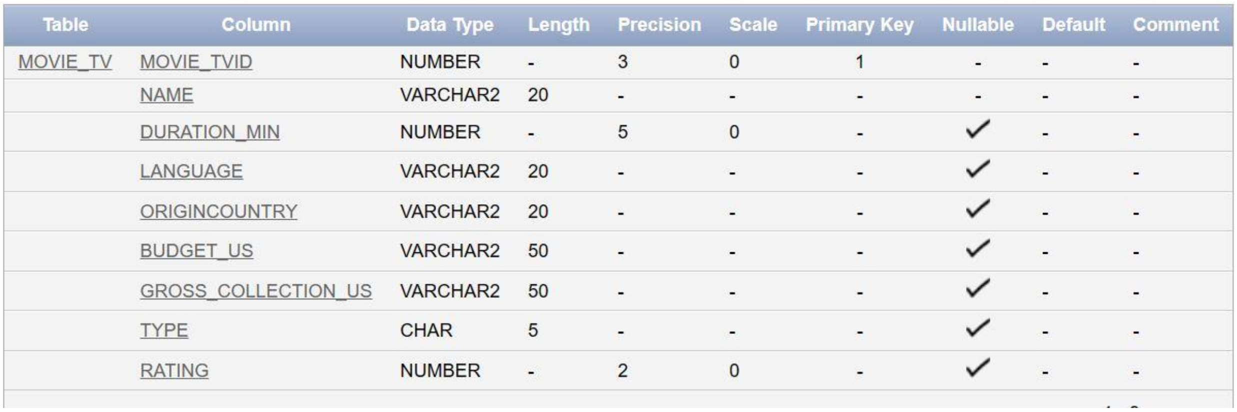


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIE_TV	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	NAME	VARCHAR2	20	-	-	-	-	-	-
	DURATION_MIN	NUMBER	-	5	0	-	✓	-	-
	LANGUAGE	VARCHAR2	20	-	-	-	✓	-	-
	ORIGINCOUNTRY	VARCHAR2	20	-	-	-	✓	-	-
	BUDGET_US	VARCHAR2	50	-	-	-	✓	-	-
	GROSS_COLLECTION_US	VARCHAR2	50	-	-	-	✓	-	-
	TYPE	CHAR	5	-	-	-	✓	-	-
	RATING	NUMBER	-	2	0	-	✓	-	-

```
create table genre(
genre_id number(3) constraint genre_genre_id_pk primary key,
genre_categoryName varchar(20) constraint gen_gen_cateName_nk not null
)
```

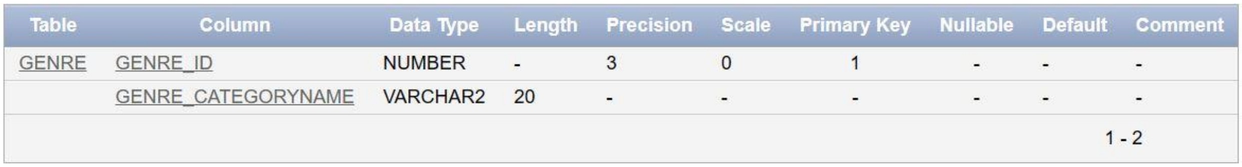


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GENRE	GENRE_ID	NUMBER	-	3	0	1	-	-	-
	GENRE_CATEGORYNAME	VARCHAR2	20	-	-	-	-	-	-
									1 - 2

```
create table movie_genre(
movie_TVID number(3),
genre_ID number(3),
constraint mov_gen_mov_tvid_gen_id primary key(movie_tvID , genre_ID),
constraint mov_gen_mov_tvid_fk foreign key(movie_tvID) references movie_tv(movie_tvID),
constraint mov_gen_gen_id_fk foreign key(genre_id) references genre(genre_id)
)
```

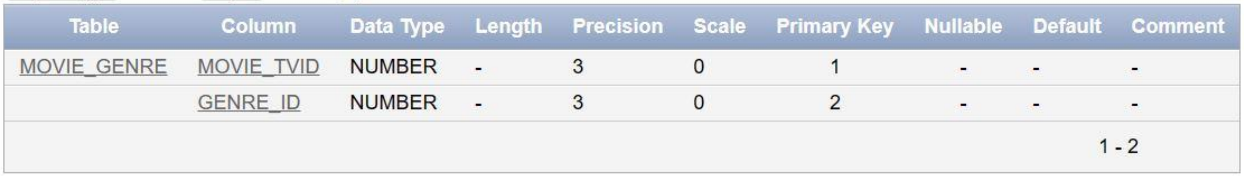


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIE_GENRE	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	GENRE_ID	NUMBER	-	3	0	2	-	-	-
									1 - 2

```
create table plot_Summary(
plot_summaryID number(3) constraint plot_sum_plot_sumID_pk primary key,
movie_tvID number(3),
authorName varchar2(20) constraint plot_sum_autName_nk not null,
description varchar2(200),
constraint plot_sum_mov_tvid_fk foreign key(movie_tvID) references movie_tv(movie_tvID)
)
```

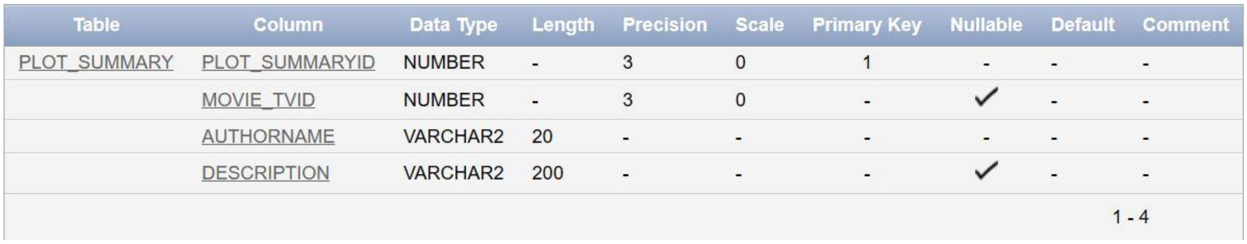


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLOT_SUMMARY	PLOT_SUMMARYID	NUMBER	-	3	0	1	-	-	-
	MOVIE_TVID	NUMBER	-	3	0	-	✓	-	-
	AUTHORNAME	VARCHAR2	20	-	-	-	-	-	-
	DESCRIPTION	VARCHAR2	200	-	-	-	✓	-	-

1 - 4

create table movie_TVReview(

reviewID number(3) constraint mov_TVRev_reviewID_pk primary key,

movie_tvID number(3),

subject varchar2(20) constraint mov_tvRev_subject_nk not null,

authorName varchar2(20),

description varchar2(200),

country varchar2(20),

reviewDate date,

constraint mov_tvRev_mov_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tvID)

)

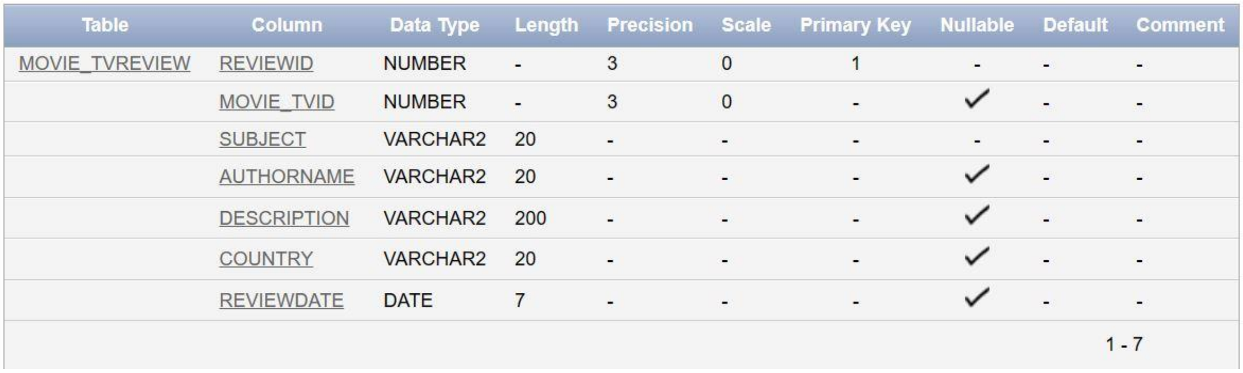


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIE_TVREVIEW	REVIEWID	NUMBER	-	3	0	1	-	-	-
	MOVIE_TVID	NUMBER	-	3	0	-	✓	-	-
	SUBJECT	VARCHAR2	20	-	-	-	-	-	-
	AUTHORNAME	VARCHAR2	20	-	-	-	✓	-	-
	DESCRIPTION	VARCHAR2	200	-	-	-	✓	-	-
	COUNTRY	VARCHAR2	20	-	-	-	✓	-	-
	REVIEWDATE	DATE	7	-	-	-	✓	-	-

1 - 7

create table Movie_tvRating(

rating_ID number(3) constraint mov_tvrating_ratingID_pk primary key,

movie_tvID number(3),

rating number(2), constraint mov_tvrating_rating check(rating between 1 and 10),

noOfVotes number(5),

constraint mov_tvrating_movie_tv_id_fk foreign key(movie_tvid) references movie_tv(movie_tvID)

)

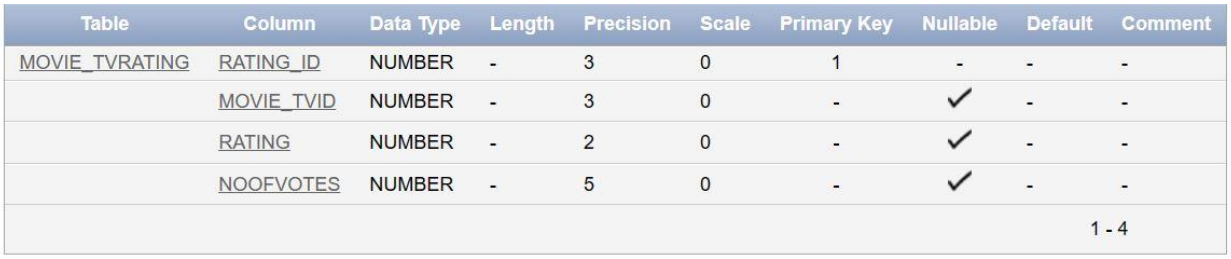


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIE_TVRATING	RATING_ID	NUMBER	-	3	0	1	-	-	-
	MOVIE_TVID	NUMBER	-	3	0	-	✓	-	-
	RATING	NUMBER	-	2	0	-	✓	-	-
	NOOFVOTES	NUMBER	-	5	0	-	✓	-	-
									1 - 4

```
create table user_rating(
user_ratingID number(3) constraint user_rating_user_ratID_pk primary key,
user_ID number(3),
movie_tvID number(3) constraint movie_tvID_nk not null,
rating number(2) constraint user_rating_rating_nk not null,
constraint user_rat_user_ID_fk foreign key(user_id) references user_record(user_id),
constraint user_rating_movie_tvID foreign key(movie_tvID) references movie_tv(movie_tvID)
)
```

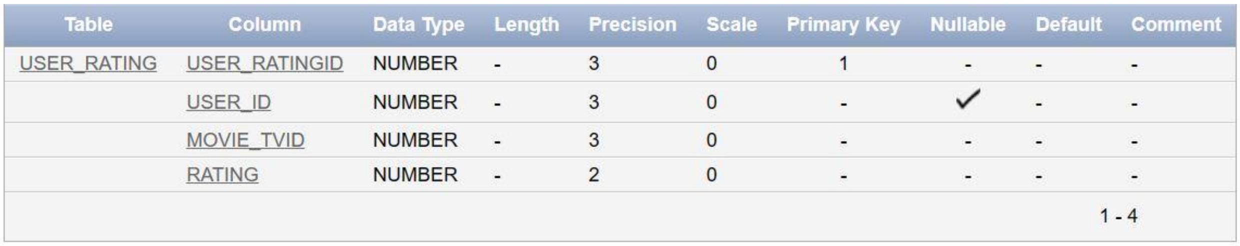


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
USER_RATING	USER_RATINGID	NUMBER	-	3	0	1	-	-	-
	USER_ID	NUMBER	-	3	0	-	✓	-	-
	MOVIE_TVID	NUMBER	-	3	0	-	-	-	-
	RATING	NUMBER	-	2	0	-	-	-	-
									1 - 4

```
create table user_record(
user_id number(3) constraint user_rec_user_id_pk primary key,
user_name varchar2(20) constraint user_rec_user_name_nk not null,
email varchar2(30) constraint user_rec_email_ck check( email like '%@email.com'),
country varchar2(20)
)
```

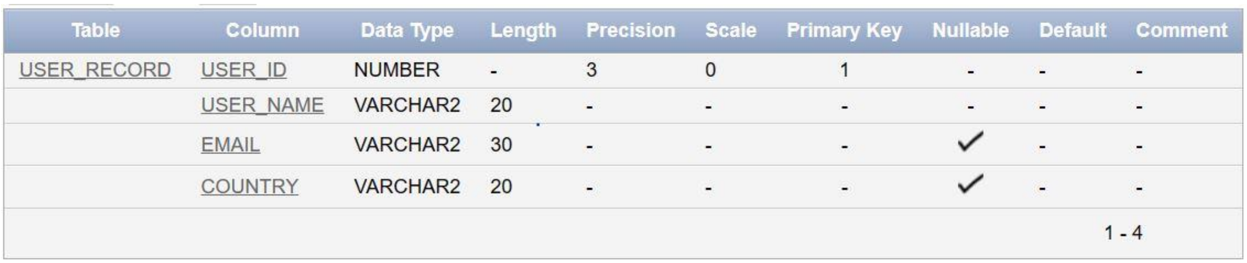


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
USER_RECORD	USER_ID	NUMBER	-	3	0	1	-	-	-
	USER_NAME	VARCHAR2	20	-	-	-	-	-	-
	EMAIL	VARCHAR2	30	-	-	-	✓	-	-
	COUNTRY	VARCHAR2	20	-	-	-	✓	-	-

1 - 4

```
create table award(
award_id number(3) constraint award_award_id_pk primary key,
category_name varchar(20) constraint award_category_name_nk not null
)
```

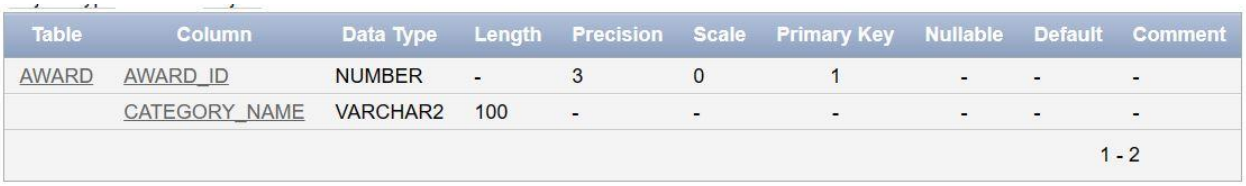


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
AWARD	AWARD_ID	NUMBER	-	3	0	1	-	-	-
	CATEGORY_NAME	VARCHAR2	100	-	-	-	-	-	-

1 - 2

```
create table movie_artist_award (
   movie_tvid number(3),
   artist_id number(3),
   award_id number(3),
   award_date date,
   status varchar2(10),
   constraint movid_artid_awaid_pk primary key (movie_tvid, artist_id, award_id),
   constraint mov_art_awa_movie_tvid_fk foreign key (movie_tvid) references movie_tv (movie_tvid),
   constraint mov_art_awa_artist_id_fk foreign key (artist_id) references artist (artist_id),
   constraint mov_art_awa_award_id_fk foreign key (award_id) references award (award_id)
);
```

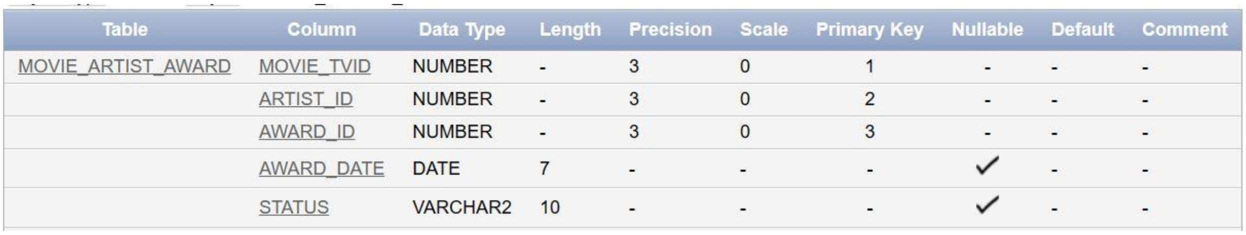


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MOVIE_ARTIST_AWARD	MOVIE_TVID	NUMBER	-	3	0	1	-	-	-
	ARTIST_ID	NUMBER	-	3	0	2	-	-	-
	AWARD_ID	NUMBER	-	3	0	3	-	-	-
	AWARD_DATE	DATE	7	-	-	-	✓	-	-
	STATUS	VARCHAR2	10	-	-	-	✓	-	-

```
create table tv_season(
tv_season_id number(3) constraint tv_sea_tv_season_id_pk primary key,
movie_tvID number(3),
season_no number(2) constraint tv_sea_season_no_nk not null,
season_name varchar2(20),
constraint tv_sea_movie_tvID_fk foreign key(movie_tvID) references movie_tv(movie_tviD)
)
```

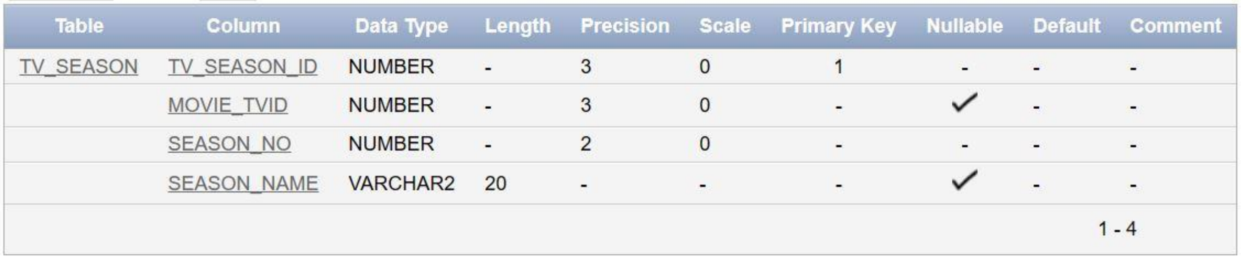


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TV_SEASON	TV_SEASON_ID	NUMBER	-	3	0	1	-	-	-
	MOVIE_TVID	NUMBER	-	3	0	-	✓	-	-
	SEASON_NO	NUMBER	-	2	0	-	-	-	-
	SEASON_NAME	VARCHAR2	20	-	-	-	✓	-	-

1 - 4

```
create table tv_episode(
tv_episode_id number(3) constraint tv_epi_tv_episode_id primary key,
season_id number(3),
tv_episodeName varchar2(20),
release_date date,
constraint tv_episode_season_id_fk foreign key(season_id) references tv_season(tv_season_id)
)
```

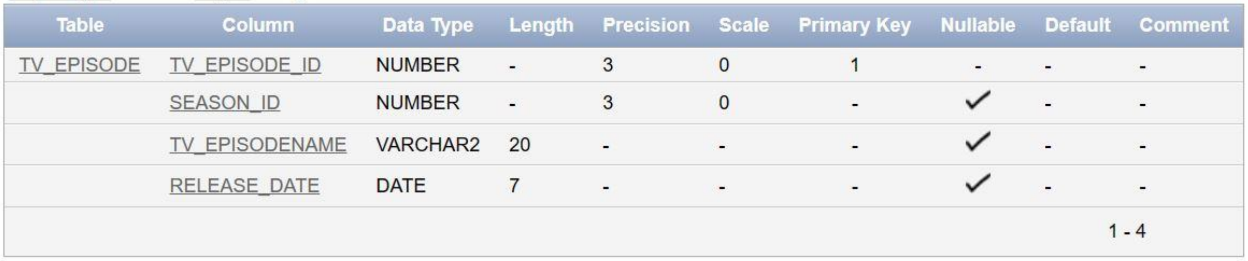
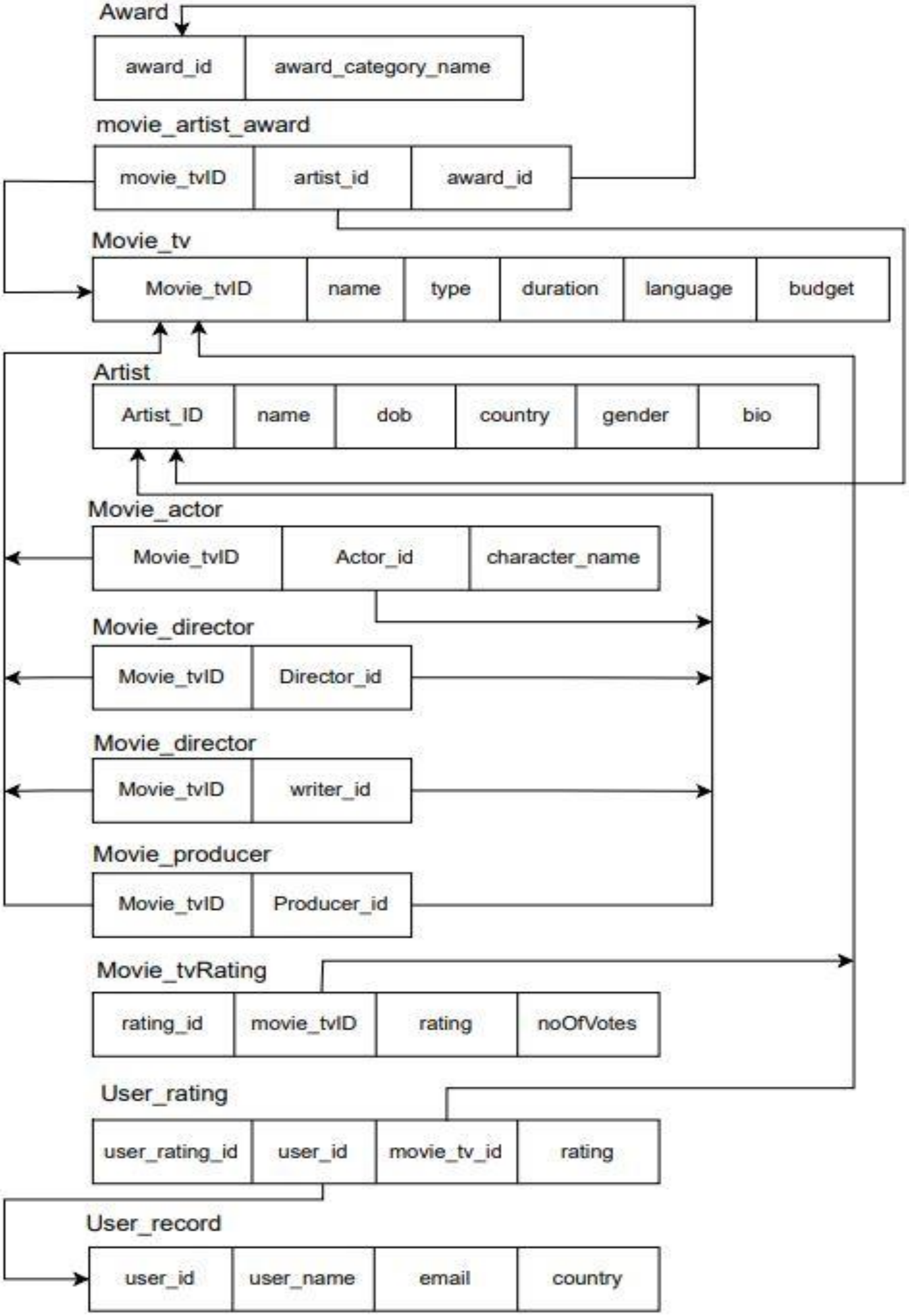


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TV_EPISODE	TV_EPISODE_ID	NUMBER	-	3	0	1	-	-	-
	SEASON_ID	NUMBER	-	3	0	-	✓	-	-
	TV_EPISODENAME	VARCHAR2	20	-	-	-	✓	-	-
	RELEASE_DATE	DATE	7	-	-	-	✓	-	-

1 - 4

8. Relational data model showing the association among different relations of the relational schema



Award

award_id	award_category_name

movie_artist_award

movie_tvID	artist_id	award_id

Movie_tv

Movie_tvID	name	type	duration	language	budget

Artist

Artist_ID	name	dob	country	gender	bio

Movie_actor

Movie_tvID	Actor_id	character_name

Movie_director

Movie_tvID	Director_id

Movie_director

Movie_tvID	writer_id

Movie_producer

Movie_tvID	Producer_id

Movie_tvRating

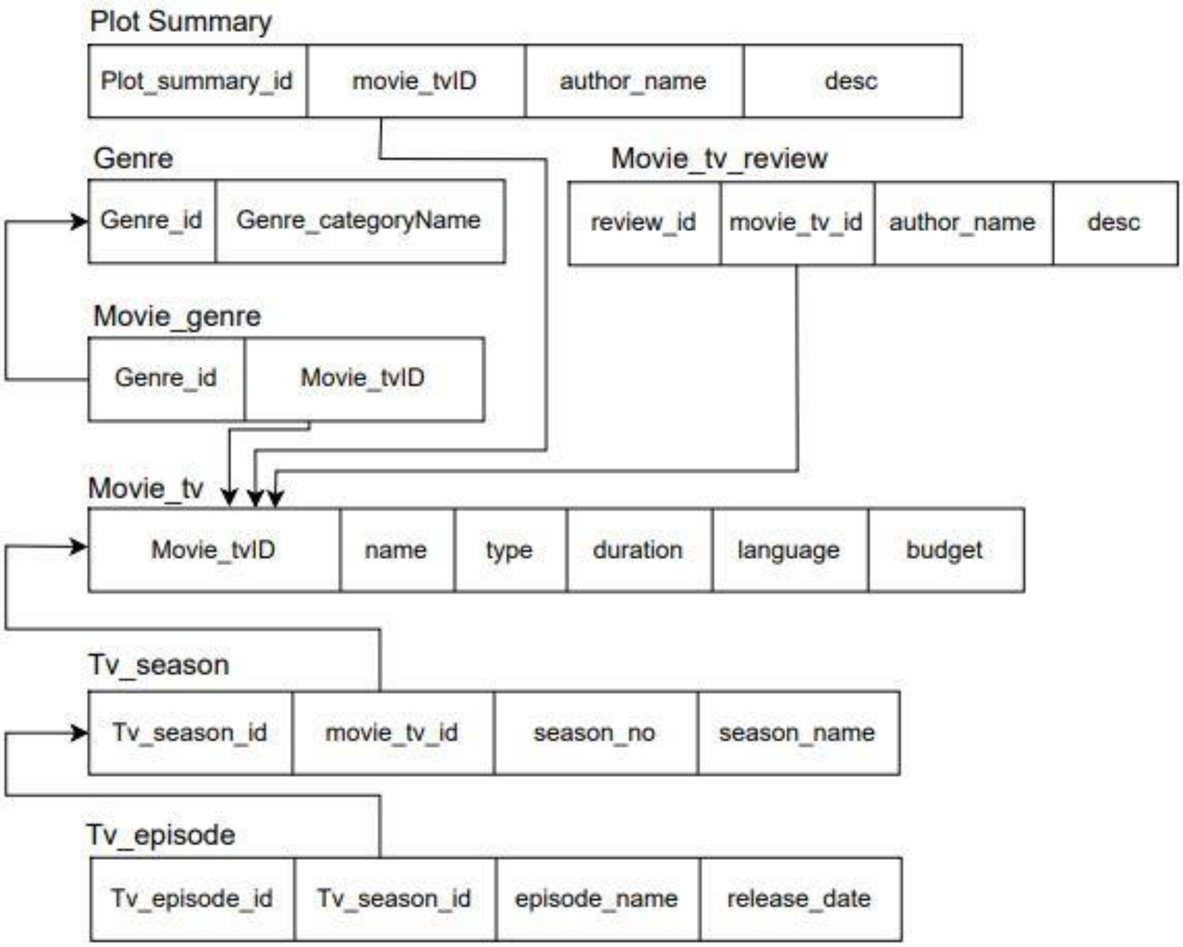
rating_id	movie_tvID	rating	noOfVotes

User_rating

user_rating_id	user_id	movie_tv_id	rating

User_record

user_id	user_name	email	country



Common Procedures & Functions used in system:

- ***Update_votes***

```
create or replace trigger update_votes
after insert OR update on user_rating
for each row
begin

update movie_tvRating mtr
set noOfVotes = noOfVotes - 1
```

```
where mtr.rating = :old.rating
and movie_tvID = :new.movie_tvID;

update movie_tvRating mtr
set noOfVotes = noOfVotes + 1
where mtr.rating = :new.rating and movie_tvID =
:new.movie_tvID;

end;
```

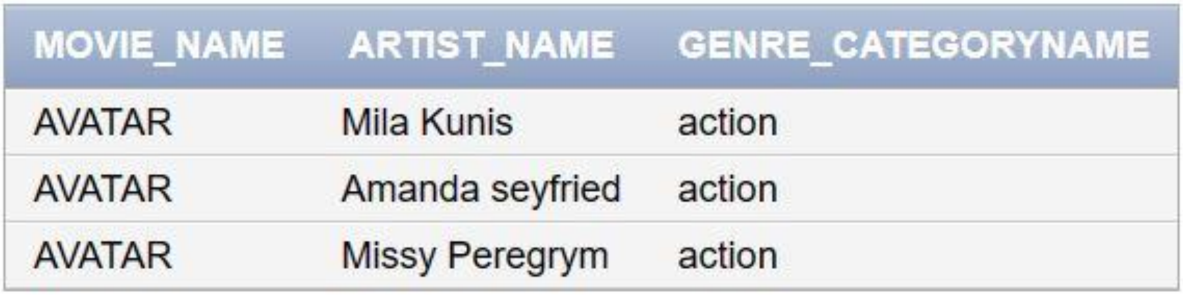
- ***Update_rating***

```
create or replace trigger update_rating
after update OR insert on movie_tvRating
for each row
declare

newRating number(4);
begin

select (sum(rating*noOfVotes))/(sum(noOfvotes)) into
newRating from movie_tvrating
where movie_tvId = :new.movie_tvID;

update movie_tv
set rating = newRating
where movie_tvID = :new.movie_tvId;

end;
```

SELECT statement for five common reports:

- ***Find action movies with female actoress***

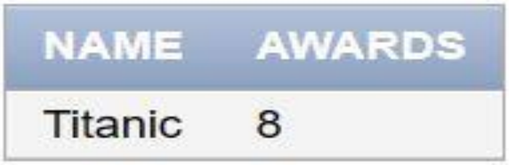
```
select m.name as movie_name, a.artist_name ,
genre_categoryName
from movie_tv m join movie_genre mg
on m.movie_tvID = mg.movie_tvID
join genre g
on g.genre_id = mg.genre_id
join movieActor ma
on ma.movie_tvID = m.movie_tvID
join artist a
on a.artist_id = ma.artist_id
where genre_categoryName = 'action'
and a.gender = 'F';
```



MOVIE_NAME	ARTIST_NAME	GENRE_CATEGORYNAME
AVATAR	Mila Kunis	action
AVATAR	Amanda seyfried	action
AVATAR	Missy Peregrym	action

- ***Higest award wining movie***

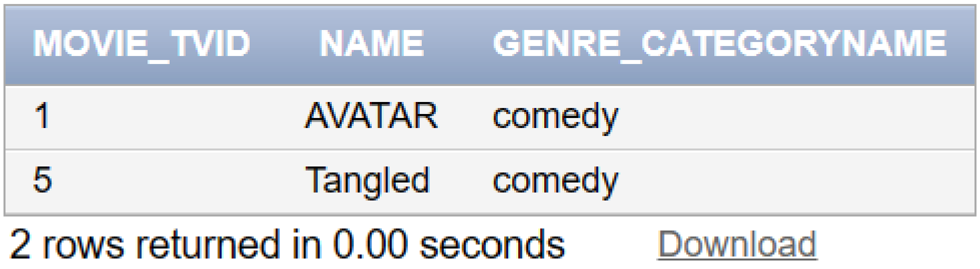
```
select name , count(*) as awards
from movie_tv mt join movie_artist_award maa
on mt.movie_tvID = maa.movie_tvID
group by name
having count(*) =
(select max(count(*)) as awards
from movie_tv mt join movie_artist_award maa
on mt.movie_tvID = maa.movie_tvID
group by name)
```



NAME	AWARDS
Titanic	8

- ***Top comedy movies***

```
select
   m.movie_tvid,
   m.name,
   g.genre_Categoryname
from
   movie_tv m
join movie_genre mg on m.movie_tvid = mg.movie_tvid
join genre g on mg.genre_id = g.genre_id
where
   g.genre_Categoryname = 'comedy';
```

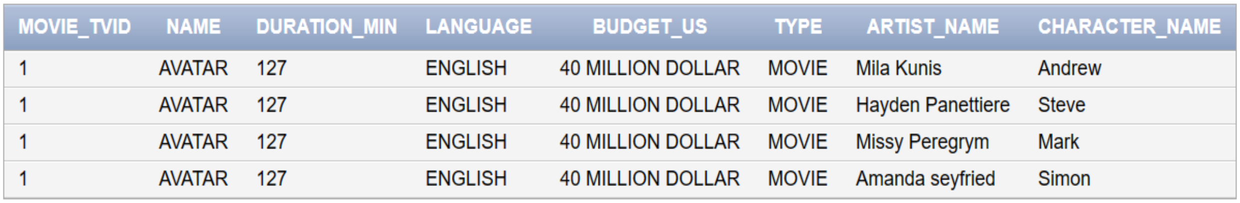


MOVIE_TVID	NAME	GENRE_CATEGORYNAME
1	AVATAR	comedy
5	Tangled	comedy

2 rows returned in 0.00 seconds   Download

- ***Actors with their character name in movie or tv***

```
select
   m.movie_tvid,
   m.name,
   m.duration_min,
   m.language,
   m.budget_us,
   m.type,
   a.artist_name,
   ma.character_name
from
   movie_tv m
join movieActor ma on m.movie_tvid = ma.movie_tvid
join artist a on ma.artist_id = a.artist_id;
```



MOVIE_TVID	NAME	DURATION_MIN	LANGUAGE	BUDGET_US	TYPE	ARTIST_NAME	CHARACTER_NAME
1	AVATAR	127	ENGLISH	40 MILLION DOLLAR	MOVIE	Mila Kunis	Andrew
1	AVATAR	127	ENGLISH	40 MILLION DOLLAR	MOVIE	Hayden Panettiere	Steve
1	AVATAR	127	ENGLISH	40 MILLION DOLLAR	MOVIE	Missy Peregrym	Mark
1	AVATAR	127	ENGLISH	40 MILLION DOLLAR	MOVIE	Amanda seyfried	Simon

Common Procedures & Functions used in system:

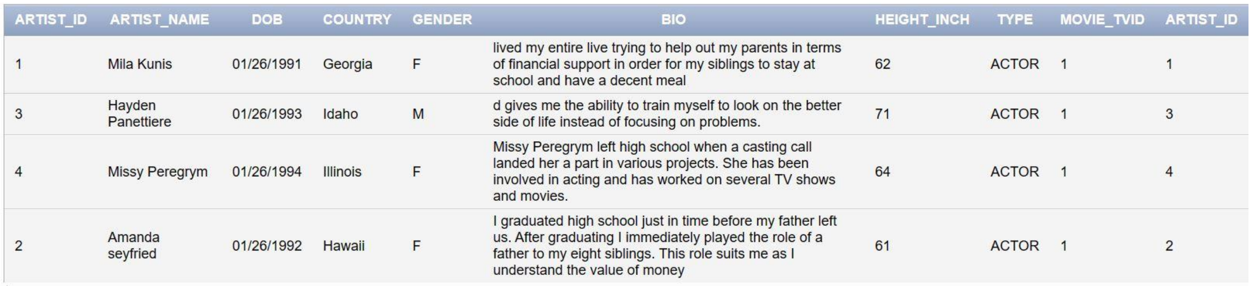
- ***Up_comming_movies***

```
create or replace procedure up_commings is
rec movie_tv%rowtype;
cursor rec_cursor is select * from movie_tv where release_date > sysdate;
begin
loop
fetch rec_cursor into rec;
exit when rec_cursor%notfound;
dbms_output.put_line('Movie_name: ' || rec.name );
end loop;
end;
```

- ***Get_actor_for_Movie***

```
create procedure get_actors_for_movie(in movieid int)
begin
   declare actor_id int;
   declare actor_name varchar(255);
   declare actor_dob date;
   declare height int;
```

```
  declare country varchar(50);
  declare gender varchar(10);
  declare bio text;

  select
    a.artist_id,
    a.actor_name,
    a.actor_dob,
    a.height,
    a.country,
    a.gender,
    a.bio
  into
    actor_id,
    actor_name,
    actor_dob,
    height,
    country,
    gender,
    bio
  from
    artist a
  join movie_actor ma on a.artist_id = ma.actor_id
  where
    ma.movie_tvid = movieid;
end
```



ARTIST_ID	ARTIST_NAME	DOB	COUNTRY	GENDER	BIO	HEIGHT_INCH	TYPE	MOVIE_TVID	ARTIST_ID
1	Mila Kunis	01/26/1991	Georgia	F	lived my entire live trying to help out my parents in terms of financial support in order for my siblings to stay at school and have a decent meal	62	ACTOR	1	1
3	Hayden Panettiere	01/26/1993	Idaho	M	d gives me the ability to train myself to look on the better side of life instead of focusing on problems.	71	ACTOR	1	3
4	Missy Peregrym	01/26/1994	Illinois	F	Missy Peregrym left high school when a casting call landed her a part in various projects. She has been involved in acting and has worked on several TV shows and movies.	64	ACTOR	1	4
2	Amanda seyfried	01/26/1992	Hawaii	F	I graduated high school just in time before my father left us. After graduating I immediately played the role of a father to my eight siblings. This role suits me as I understand the value of money	61	ACTOR	1	2

- ***High_rated_Movie***

```
create or replace function high_rated_movie
```

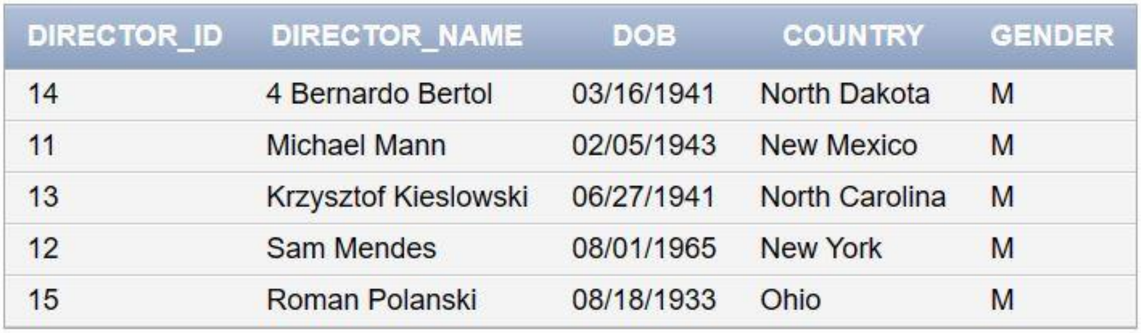
```
    return number is
      v_id number(3);
      v_max_rating number;

    begin
      select movie_tvid, max(rating) into v_id, v_max_rating
      from movie_tv
      where type = 'MOVIE'
      group by movie_tvid;

      if v_id is not null then
        return v_id;
      else
        dbms_output.put_line('No movie has the highest rating.');
        return null;
      end if;

    exception
      when no_data_found then
        dbms_output.put_line('No movie found.');

end high_rated_movie;
```

2 Common views used in system:

- ***Top_movie_directors***

```
create or replace view top_directors
as select a.artist_id as director_id,
artist_name as director_name,
dob,
country,
gender
from artist a join movieDirector md
on a.artist_id = md.artist_id
join movie_tv mt
on md.movie_tvID = mt.movie_tvID
join movie_award ma
on ma.movie_tvID = mt.movie_tvID
```

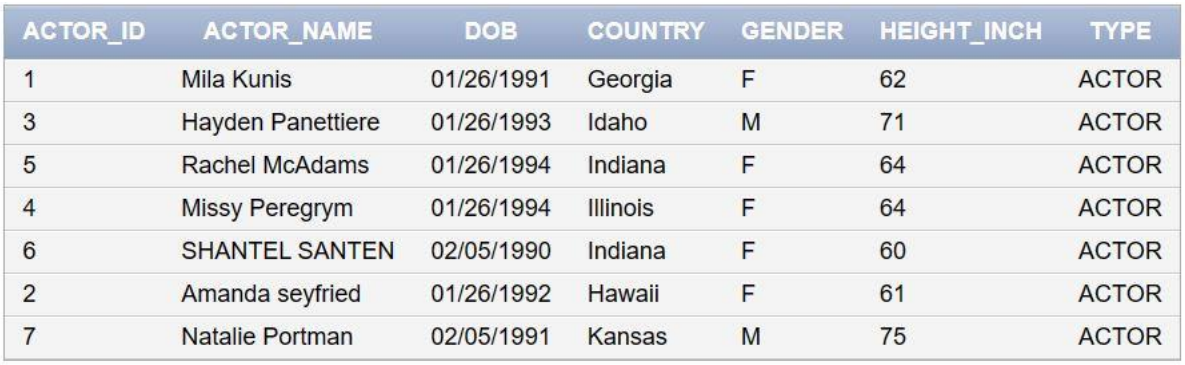
```
where ma.status = ‘WON’
```



DIRECTOR_ID	DIRECTOR_NAME	DOB	COUNTRY	GENDER
14	4 Bernardo Bertol	03/16/1941	North Dakota	M
11	Michael Mann	02/05/1943	New Mexico	M
13	Krzysztof Kieslowski	06/27/1941	North Carolina	M
12	Sam Mendes	08/01/1965	New York	M
15	Roman Polanski	08/18/1933	Ohio	M

- ***Top_movie_actors***

```
create or replace view top_actors
as select
a.artist_id as actor_id,
artist_name as actor_name,
dob,
country,
gender
from artist a
join movieActor md
on a.artist_id = md.artist_id
join movie_tv mt
on md.movie_tvID = mt.movie_tvID
join movie_award ma
on ma.movie_tvID = mt.movie_tvID
where ma.status = ‘WON’
having sum(award_id) > 10;
```



ACTOR_ID	ACTOR_NAME	DOB	COUNTRY	GENDER	HEIGHT_INCH	TYPE
1	Mila Kunis	01/26/1991	Georgia	F	62	ACTOR
3	Hayden Panettiere	01/26/1993	Idaho	M	71	ACTOR
5	Rachel McAdams	01/26/1994	Indiana	F	64	ACTOR
4	Missy Peregrym	01/26/1994	Illinois	F	64	ACTOR
6	SHANTEL SANTEN	02/05/1990	Indiana	F	60	ACTOR
2	Amanda seyfried	01/26/1992	Hawaii	F	61	ACTOR
7	Natalie Portman	02/05/1991	Kansas	M	75	ACTOR