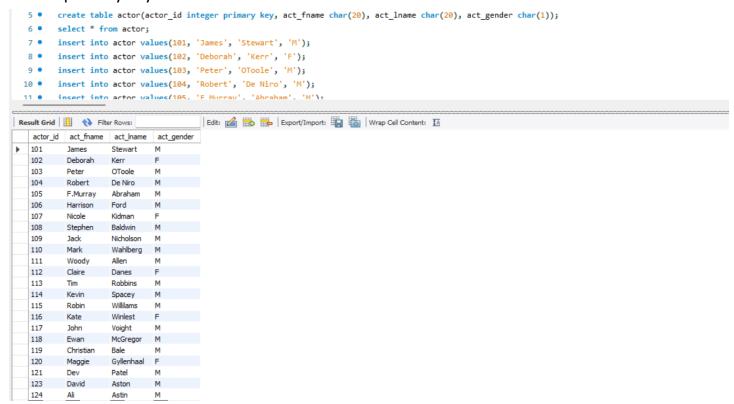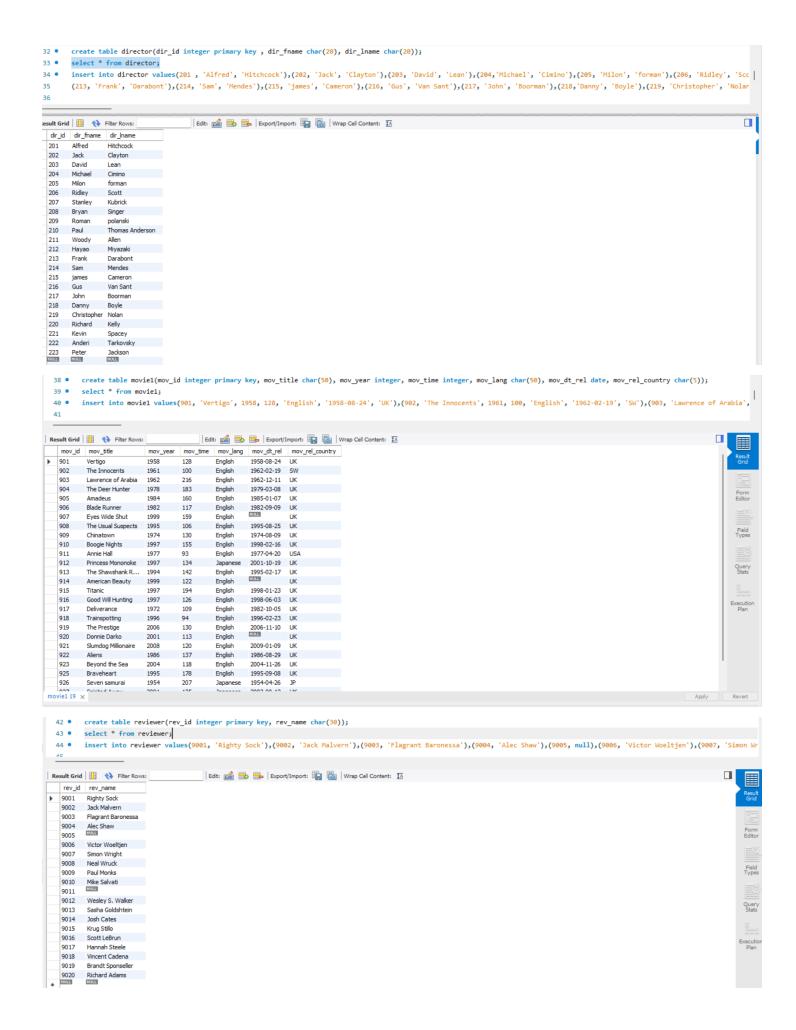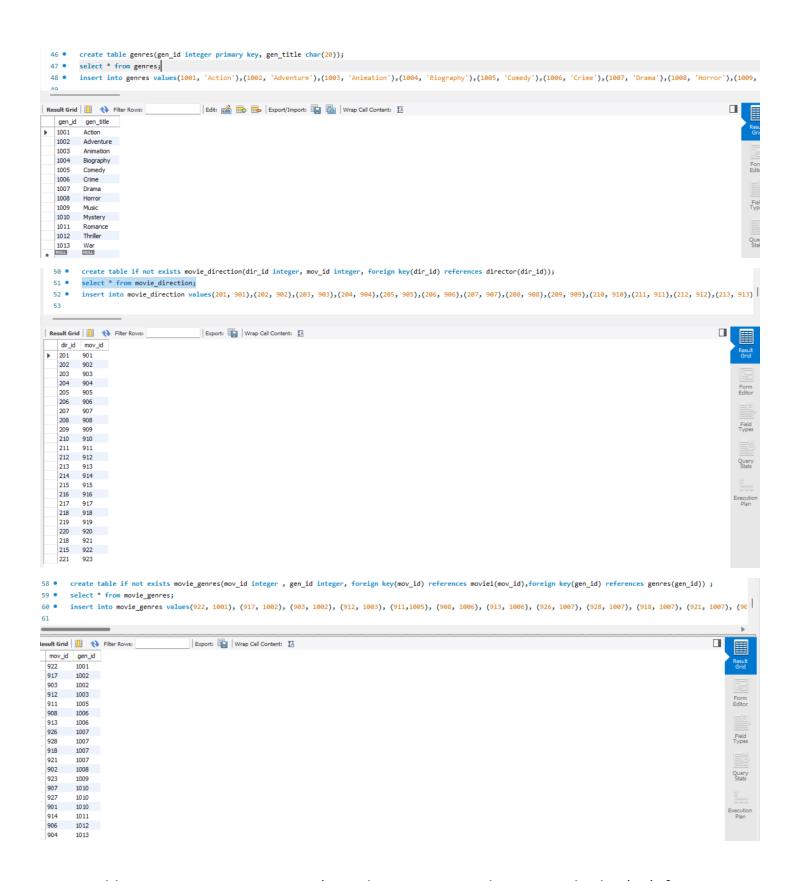# SBA 3 ASSESSMENT

1. Create Database project_movie_data and deploy table data according to ER diagram.

```
1 •    create database project_movie_data;
2 •    use project_movie_data;
3
4
```

2. Create 9 tables which are presented in ER model and load the data with their foreign key and primary key values.

```
5 •    create table actor(actor_id integer primary key, act_fname char(20), act_lname char(20), act_gender char(1));
6 •    select * from actor;
7 •    insert into actor values(101, 'James', 'Stewart', 'M');
8 •    insert into actor values(102, 'Deborah', 'Kerr', 'F');
9 •    insert into actor values(103, 'Peter', 'OToole', 'M');
10 •   insert into actor values(104, 'Robert', 'De Niro', 'M');
11 •   insert into actor values(105, 'F.Murray', 'Abraham', 'M');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: IA

actor_id	act_fname	act_lname	act_gender
101	James	Stewart	M
102	Deborah	Kerr	F
103	Peter	OToole	M
104	Robert	De Niro	M
105	F.Murray	Abraham	M
106	Harrison	Ford	M
107	Nicole	Kidman	F
108	Stephen	Baldwin	M
109	Jack	Nicholson	M
110	Mark	Wahlberg	M
111	Woody	Allen	M
112	Claire	Danes	F
113	Tim	Robbins	M
114	Kevin	Spacey	M
115	Robin	Willilams	M
116	Kate	Winlest	F
117	John	Voight	M
118	Ewan	McGregor	M
119	Christian	Bale	M
120	Maggie	Gyllenhaal	F
121	Dev	Patel	M
123	David	Aston	M
124	Ali	Astin	M

```sql
32 •   create table director(dir_id integer primary key , dir_fname char(20), dir_lname char(20));
33 •   select * from director;
34 •   insert into director values(201 , 'Alfred', 'Hitchcock'),(202, 'Jack', 'Clayton'),(203, 'David', 'Lean'),(204,'Michael', 'Cimino'),(205, 'Milon', 'forman'),(206, 'Ridley', 'Sco
35     (213, 'Frank', 'Darabont'),(214, 'Sam', 'Mendes'),(215, 'james', 'Cameron'),(216, 'Gus', 'Van Sant'),(217, 'John', 'Boorman'),(218,'Danny', 'Boyle'),(219, 'Christopher', 'Nolar
36
```

dir_id	dir_fname	dir_lname
201	Alfred	Hitchcock
202	Jack	Clayton
203	David	Lean
204	Michael	Cimino
205	Milon	forman
206	Ridley	Scott
207	Stanley	Kubrick
208	Bryan	Singer
209	Roman	polanski
210	Paul	Thomas Anderson
211	Woody	Allen
212	Hayao	Miyazaki
213	Frank	Darabont
214	Sam	Mendes
215	james	Cameron
216	Gus	Van Sant
217	John	Boorman
218	Danny	Boyle
219	Christopher	Nolan
220	Richard	Kelly
221	Kevin	Spacey
222	Anderi	Tarkovsky
223	Peter	Jackson
NULL	NULL	NULL

```sql
38 •   create table movie1(mov_id integer primary key, mov_title char(50), mov_year integer, mov_time integer, mov_lang char(50), mov_dt_rel date, mov_rel_country char(5));
39 •   select * from movie1;
40 •   insert into movie1 values(901, 'Vertigo', 1958, 128, 'English', '1958-08-24', 'UK'),(902, 'The Innocents', 1961, 100, 'English', '1962-02-19', 'SW'),(903, 'Lawrence of Arabia',
41
```

mov_id	mov_title	mov_year	mov_time	mov_lang	mov_dt_rel	mov_rel_country
901	Vertigo	1958	128	English	1958-08-24	UK
902	The Innocents	1961	100	English	1962-02-19	SW
903	Lawrence of Arabia	1962	216	English	1962-12-11	UK
904	The Deer Hunter	1978	183	English	1979-03-08	UK
905	Amadeus	1984	160	English	1985-01-07	UK
906	Blade Runner	1982	117	English	1982-09-09	UK
907	Eyes Wide Shut	1999	159	English	NULL	UK
908	The Usual Suspects	1995	106	English	1995-08-25	UK
909	Chinatown	1974	130	English	1974-08-09	UK
910	Boogie Nights	1997	155	English	1998-02-16	UK
911	Annie Hall	1977	93	English	1977-04-20	USA
912	Princess Mononoke	1997	134	Japanese	2001-10-19	UK
913	The Shawshank R...	1994	142	English	1995-02-17	UK
914	American Beauty	1999	122	English	NULL	UK
915	Titanic	1997	194	English	1998-01-23	UK
916	Good Will Hunting	1997	126	English	1998-06-03	UK
917	Deliverance	1972	109	English	1982-10-05	UK
918	Trainspotting	1996	94	English	1996-02-23	UK
919	The Prestige	2006	130	English	2006-11-10	UK
920	Donnie Darko	2001	113	English	NULL	UK
921	Slumdog Millionaire	2008	120	English	2009-01-09	UK
922	Aliens	1986	137	English	1986-08-29	UK
923	Beyond the Sea	2004	118	English	2004-11-26	UK
925	Braveheart	1995	178	English	1995-09-08	UK
926	Seven samurai	1954	207	Japanese	1954-04-26	JP

movie1 19 ×

```sql
42 •   create table reviewer(rev_id integer primary key, rev_name char(30));
43 •   select * from reviewer;
44 •   insert into reviewer values(9001, 'Righty Sock'),(9002, 'Jack Malvern'),(9003, 'Flagrant Baronessa'),(9004, 'Alec Shaw'),(9005, null),(9006, 'Victor Woeltjen'),(9007, 'Simon Wr
```

rev_id	rev_name
9001	Righty Sock
9002	Jack Malvern
9003	Flagrant Baronessa
9004	Alec Shaw
9005	NULL
9006	Victor Woeltjen
9007	Simon Wright
9008	Neal Wruck
9009	Paul Monks
9010	Mike Salvati
9011	NULL
9012	Wesley S. Walker
9013	Sasha Goldshtein
9014	Josh Cates
9015	Krug Stillo
9016	Scott LeBrun
9017	Hannah Steele
9018	Vincent Cadena
9019	Brandt Sponseller
9020	Richard Adams
NULL	NULL

```sql
46 •  create table genres(gen_id integer primary key, gen_title char(20));
47 •  select * from genres;
48 •  insert into genres values(1001, 'Action'),(1002, 'Adventure'),(1003, 'Animation'),(1004, 'Biography'),(1005, 'Comedy'),(1006, 'Crime'),(1007, 'Drama'),(1008, 'Horror'),(1009,
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

gen_id	gen_title
1001	Action
1002	Adventure
1003	Animation
1004	Biography
1005	Comedy
1006	Crime
1007	Drama
1008	Horror
1009	Music
1010	Mystery
1011	Romance
1012	Thriller
1013	War
NULL	NULL

```sql
50 •  create table if not exists movie_direction(dir_id integer, mov_id integer, foreign key(dir_id) references director(dir_id));
51 •  select * from movie_direction;
52 •  insert into movie_direction values(201, 901),(202, 902),(203, 903),(204, 904),(205, 905),(206, 906),(207, 907),(208, 908),(209, 909),(210, 910),(211, 911),(212, 912),(213, 913)
53
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

dir_id	mov_id
201	901
202	902
203	903
204	904
205	905
206	906
207	907
208	908
209	909
210	910
211	911
212	912
213	913
214	914
215	915
216	916
217	917
218	918
219	919
220	920
218	921
215	922
221	923

```sql
58 •  create table if not exists movie_genres(mov_id integer , gen_id integer, foreign key(mov_id) references movie1(mov_id),foreign key(gen_id) references genres(gen_id)) ;
59 •  select * from movie_genres;
60 •  insert into movie_genres values(922, 1001), (917, 1002), (903, 1002), (912, 1003), (911,1005), (908, 1006), (913, 1006), (926, 1007), (928, 1007), (918, 1007), (921, 1007), (90
61
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

mov_id	gen_id
922	1001
917	1002
903	1002
912	1003
911	1005
908	1006
913	1006
926	1007
928	1007
918	1007
921	1007
902	1008
923	1009
907	1010
927	1010
901	1010
914	1011
906	1012
904	1013

```sql
create table IF NOT exists movie_cast(act_id integer, mov_id integer, role char(30), foreign
key(act_id) references actor(act_id), foreign key(mov_id) references movie1(mov_id));
select * from movie_cast;
insert into movie_cast values(101, 901, 'John Scottie Ferguson'),
(102, 902, 'Miss Giddens'),
```

(103, 903, 'T.E.Lawrence')
,(104, 904, 'Michael'),
(105, 905, 'Antonio Salieri'),
(106, 906, 'Dick Deckard'),
(107, 907, 'Alice Harford'),
(108, 908, 'McManus'),
 (110, 910, 'Eddie Adams'),
(111, 911, 'Alvy Singer'),
(112, 912, 'San'),
(113, 913, 'Andy Dufresne'),
(114, 914, 'Lester Burnham'),
(115, 915, 'Rose DeWitt Bukater'),
(116, 916, 'Sean Maguire'),
(117, 917, 'Ed'),(118, 918, 'Renton'),
(120, 920, 'Elizabeth Darko'),(121, 921, 'Older Jamal'),
(122, 922, 'Ripley'),
(114, 923, 'Bobby Darin'),
(109, 909, 'J.J Gittes'),
(119, 919, 'Alfred Borden');

act_id	mov_id	role
101	901	John Scottie Fergus
102	902	Miss Giddens
103	903	T.E. Lawrence
104	904	Michael
105	905	Antonio Salieri
106	906	Rick Deckard
107	907	Alice Harford
108	908	McManus
110	910	Eddie Adams
111	911	Alvy Singer
112	912	San
113	913	Andy Dufresne
114	914	Lester Burnham
115	915	Rose DeWitt Bukater
116	916	Sean Maguire
117	917	Ed

create table rating(mov_id integer, rev_id integer, rev_stars integer, num_o_ratings integer,
foreign key(mov_id) references movie1(mov_id), foreign key(rev_id) references
reveiwer(rev_id));
select * from rating;

```
insert into rating values(901, 9001, 8.4, 263575),
 (902, 9002, 7.9, 20207),
 (903, 9003, 8.3, 202778),
 (906, 9005, 8.2, 484746),
 (924, 9006, 7.3, null),
(908, 9007, 8.6, 779489),
 (909, 9008, null, 227235),
 (910, 9009, 3, 195961),
 (911, 9010, 8.1, 203875),
 (912, 9011, 8.4, null),
 (914, 9013, 7, 862618),
 (915, 9001, 7.7, 830095),
(916, 9014, 4, 642132),
 (925, 9015, 7.7, 81328),
 (918, 9016, null, 580301),
 (920, 9017, 8.1, 609451),
 (921, 9018, 8, 667758),
 (922, 9019, 8.4, 511613),
 (923, 9020, 6.7, 13091);
```

mov_id	rev_id	rev_stars	num_o_ratings
901	9001	8.4	263575
902	9002	7.9	20207
903	9003	8.3	202778
906	9005	8.2	484746
924	9006	7.3	NULL
908	9007	8.6	779489
909	9008	NULL	227235
910	9009	3	195961
911	9010	8.1	203875
912	9011	8.4	NULL
914	9013	7	862618
915	9001	7.7	830095
916	9014	4	642132
925	9015	7.7	81328
918	9016	NULL	580301
920	9017	8.1	609451

3. Write a query in SQL to list the Horror movies?

> select mov_title from movie where mov_id in (select mov_id from movie_genres
>
> where gen_id in(select gen_id from genres where gen_title="Horror"));

mov_title
The Innocents

4. Write a query in SQL to find the name of all reviewers who have rated 8 or   more stars?

SELECT REVIEWER.REV_NAME
FROM REVIEWER, RATING
WHERE RATING.REV_ID = REVIEWER.REV_ID
AND RATING.REV_STARS >= 8
AND REVIEWER.REV_NAME IS NOT NULL;

rev_name
Righty Sock
Flagrant Baronessa
Simon Wright
Mike Salvati
Hannah Steele
Vincent Cadena

5. Write a query in SQL to list all the information of the actors who played a role in the movie 'Deliverance'

```
SELECT *
FROM ACTOR
WHERE ACT_ID  IN(
SELECT ACT_ID
FROM MOVIE_CAST
WHERE MOV_ID IN (
SELECT MOV_ID
FROM movie1
WHERE MOVIE_TITLE="Deliverance"
));
```

act_id	act_fname	act_lname	act_gender
117	Jon	Voight	M

6. Write a query in SQL to find the name of the director (first and last names) who directed a movie that casted a role for 'Eyes Wide Shut'. (using subquery)

```
select d.dir_fname, d.dir_lname
from director d, movie_direction md
where d.dir_id = md.dir_id
and md.mov_id in(select mov_id from movie where mov_title = 'Eyes Wide Shut');
```

dir_fname	dir_lname
Stanley	Kubrick

7.Write a query in SQL to find the movie title, year, date of release, director and actor for those movies which reviewer is 'Neal Wruck'.

select mov_title , mov_year, mov_dt_rel, dir_fname, dir_lname, act_fname, act_lname
from movie1 a, movie_direction b, director c, rating d, reviewer e, actor f, movie_cast g
where a.mov_id = b.mov_id
and b.dir_id = c.dir_id
and a.mov_id = d.mov_id
and d.rev_id = e.rev_id
and a.mov_id= g.mov_id
and g.act_id = f.act_id
and e.rev_name = "Neal Wruck";

mov_title	mov_year	mov_dt_rel	dir_fname	dir_lname	act_fname	act_lname
Chinatown	1974	1974-08-09	Roman	Polanski	Jack	Nicholson