

DATA MODELLING

Hitesh Pantha

30362043 MIT federation

Contents

Q.1.....	1
Q.2.....	3
Q.3.....	33
Q.No 4	34
Q.NO 5.....	34
Q.NO 6.....	35
Q.No 7	35
Q.no 8.....	36
Q,n0 9.....	37
Q.10.....	37
q.no 11	39
Q.No 12	40
Q.no 13.....	40
Q.no 14.....	41
Q.no 15.....	41
Q.No 16	42
Conclusion:.....	43
References:	43

Q.1

Create table actor

```
(  
act_id integer  
act_fname char (20),  
act_lname char (20),  
act_gender char (1),  
Primary key (act_id)  
);
```

Create table director

```
(  
    dir_id integer,  
    mov_id integer,  
    Primary key (dir_id)  
);
```

Create reviewer

```
(  
    rev_id integer,  
    rev_name char (30),  
    Primary key (rev_id)  
);
```

Create table actor

```
(  
    gen_id integer,  
    gen_titlechar (20),  
    Primary key (gen_id)  
);
```

Create table movie

```
(  
    mov_id integer,  
    mov_title char (50),  
    mov_year integer,  
    mov_time integer,  
    mov_lang char(50)  
    mov_dt_rel date,  
    mov_rel_country char(5),  
    Primary key (mov_id)  
);
```

```

create table rating
(
    mov_id integer references movie (mov_id),
    rev_id integer references reviewer(rev_id),
    rev_stars varchar(30),
    num_0_rating integer
);

create table movie_direction
(
    dir_id integer REFERENCES director (dir_id),
    mov_id integer REFERENCES movie (mov_id)
);

create table movie_cast
(
    act_id integer references actor (act_id),
    mov_id integer references movie (mov_id),
    role varchar
)

create table movie_genres
(
    gen_id integer references genres (gen_id),
    mov_id integer references movie (mov_id)
)

```

Q.2

```

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('101', 'James', 'Stewart', 'M');

INSERT INTO public.actor(

```

```
    act_id, act_fname, act_lname, act_gender)
VALUES ('102', 'Deborah', 'Kerr', 'F');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('103', 'Peter', 'OToole', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('104', 'Robert', 'De Niro', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('105', 'F.Murray', 'Abraham', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('106', 'Harrison', 'Ford', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('107', 'Nicole', 'Kidman', 'F');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('108', 'Stephan', 'Baldwin', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('109', 'Jack', 'Nicholson');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('110', 'Mark', 'Wahlberg', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('111', 'Woody', 'Allen', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
```

```
VALUES ('112', 'Clarie', 'Danes', 'F');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('113', 'Tim', 'Robbens', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('114', 'Kevin', 'Spacy', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('115', 'Kate', 'Winslet', 'F');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('116', 'Robins', 'Willaims', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('117', 'Jon', 'Voight', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('118', 'Ewan', 'McGregor', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('119', 'Christian', 'Bale', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('120', 'Maggie', 'hallGyllen', 'F');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('121', 'Dev', 'Patel', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('122', 'Signorney', 'Weaver', 'F');
```

```

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('123', 'David', 'Aston', 'M');

INSERT INTO public.actor(
    act_id, act_fname, act_lname, act_gender)
VALUES ('124', 'Ali', 'Astin', 'F');

```

The screenshot shows the pgAdmin 4 interface with the following details:

- Toolbar:** Includes icons for back, forward, search, and refresh, along with a URL bar set to `http://127.0.0.1:50814/browser/`.
- Header:** Shows "Admin" and various menu options like File, Object, Tools, Help, and a "Dependencies" tab.
- Left Panel (Object Browser):**
 - Shows the database structure under "nuser" and "public".
 - The "Tables (9)" section is expanded, with "actor" selected.
 - Under "actor", the following items are listed: Columns (4), Constraints (1), Indexes, RLS Policies, Rules, Triggers, director, genres, movie, and movie_cast.
- Central Panel (Query Editor):**
 - Shows the query `select * from actor`.
 - Shows the results of the query in a table format:

act_id	[PK] integer	act_fname	character varying	act_lname	character varying	act_gender	character varying
1	107	Nicole		Kidman		F	
2	108	Stephan		Baldwin		M	
3	109	Jack		Nicholson		M	
4	110	Mark		Wahlberg		M	
5	111	Woody		Allen		M	
6	112	Clarie		Danes		F	
7	113	Tim		Robbens		M	
8	114	Kevin		Spacy		M	
9	115	Kate		Winslet		F	
10	116	Robins		Willaims		M	
11	117	Jon		Voight		M	
12	118	Ewan		McGregor		M	
13	119	Christian		Bale		M	
14	120	Mannie		hallullivan		F	

The screenshot shows the MySQL Workbench application. The 'Dependencies' tab is active. The left sidebar lists database objects: Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Procedures, Sequences, and Tables (9). Under 'Tables (9)', 'actor' is selected, and its details like Columns (4), Constraints (1), Indexes, RLS Policies, Rules, and Triggers are shown. The main pane contains a 'Query Editor' tab with the SQL command 'select * from actor'. The results are displayed in a table with columns: act_id [PK] integer, act_fname character varying, act_lname character varying, and act_gender character varying. The data shows 24 rows of actor information.

act_id	act_fname	act_lname	act_gender
13	Christian	Bale	M
14	Maggie	McGilligan	F
15	Dev	Patel	M
16	James	Stewart	M
17	Signorini	Weaver	F
18	David	Aston	M
19	Ali	Astin	F
20	Deborah	Kerr	F
21	Peter	O'Toole	M
22	Robert	De Niro	M
23	F.Murray	Abraham	M
24	Harrison	Ford	M

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('101', '901', 'John Scottie Ferguson');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('102', '902', 'Miss Giddens');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('103', '903', 'T.E Lawrence');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('104', '904', 'Micheal');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('105', '905', 'Antonia Salieri');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('106', '906', 'Rich Deckard');
```

```
INSERT INTO public.movie_cast(
```

```
    act_id, mov_id, role)
VALUES ('107', '907', 'Alis Harford');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('108', '908', 'McManus');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('110', '910', 'Eddie Adams');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('111', '911', 'Alvy Singer');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('112', '912', 'San');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('113', '913', 'Andy Dufresne');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('114', '914', 'Lester Burnham');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('115', '915', 'Rose FeWitt Bukater');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('116', '916', 'Sean Maguire');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('117', '917', 'Ed');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
```

```
VALUES ('118', '918', 'Renton');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('120', '920', 'Elizabeth Darko');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('121', '921', 'Older Jamei');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('122', '922', 'Ripley');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('123', '923', 'Bobby Darin');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('109', '909', 'J.J Gittes');

INSERT INTO public.movie_cast(
    act_id, mov_id, role)
VALUES ('119', '919', 'Alfred Borden');
```

gAdmin

File ▾ Object ▾ Tools ▾ Help ▾

Properties SQL Statistics Dependents Assignmnet 2/...

Query Editor

```
1 select * from movie_cast
2
```

Query Editor Query History Notifications

Data Output Explain Messages

	act_id integer	mov_id integer	role character varying
1	101	901	John Scottie Ferguson
2	102	902	Miss Giddens
3	103	903	T.E Lawrence
4	104	904	Micheal
5	105	905	Antonia Salieri
6	106	906	Rich Deckard
7	107	907	Alis Harford
8	108	908	McManus
9	110	910	Eddie Adams
10	111	911	Alvy Singer
11	112	912	San

Type here to search

Properties SQL Statistics Dependents Assignmnet 2/...

Properties SQL Statistics Dependents Assignmnet 2/...

Query Editor

```
1 select * from movie_cast
2
```

Query Editor Query History Notifications

Data Output Explain Messages

	act_id integer	mov_id integer	role character varying
12	113	913	Andy Dufresne
13	114	914	Lester Burnham
14	115	915	Rose FeWitt Bukater
15	116	916	Sean Maguire
16	117	917	Ed
17	118	918	Renton
18	120	920	Elizabeth Darko
19	121	921	Older Jamel
20	122	922	Ripley

The screenshot shows the MySQL Workbench interface. On the left, the Object Navigator displays the schema structure under the 'public' database, including tables like actor, director, genres, movie, movie_cast, movie_direction, movie_genres, rating, reviewer, and their respective columns and constraints. The 'director' table is currently selected. The main area contains the Query Editor with the following SQL code:

```

1 select * from movie_cast
2

```

The Data Output tab is active, showing the results of the query:

	act_id	mov_id	role
	integer	integer	character varying
15	116	916	Sean Maguire
16	117	917	Ed
17	118	918	Renton
18	120	920	Elizabeth Darko
19	121	921	Older Jamel
20	122	922	Ripley
21	123	923	Bobby Darin
22	109	909	J.J Gittes
23	119	919	Alfred Borden

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('901', 'Vertigo', '1958', '128', 'English', '1958-08-21', 'UK');
```

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('902', 'The Innocents', '1961', '100', 'English', '1962-02-19', 'SW');
```

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('903', 'Lawremnce of Arabia', '1962', '216', 'English', '1962-12-11', 'UK');
```

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('904', 'The Deer Hunter', '1978', '183', 'English', '1979-03-08', 'UK');
```

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('905', 'Amadeus', '1984', '160', 'English', '1985-01-07', 'UK');
```

INSERT INTO public.movie(

```

    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
```

```
VALUES ('906', 'Blade Runner', '1982', '117', 'English', '1982-09-09', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('907', 'Eye Wide Shut', '1999', '159', 'English', '1995-08-25', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('908', 'The Usual Suspects', '1995', '159', 'English', '', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('909', 'Chinatown', '1974', '130', 'English', '1975-08-09', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('910', 'Bogie Nights', '1997', '155', 'English', '1998-02-16', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('911', 'Annie Hall', '1977', '93', 'English', '1977-04-20', 'USA');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('912', 'Princess Mononoke', '1997', '155', 'Japanese', '2001-10-19', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('913', 'The Shawshank Redemption', '1994', '142', 'English', '1995-02-17', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('914', 'American Beauty', '1999', '122', 'English', '', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('915', 'Titanic', '1997', '194', 'English', '1998-01-23', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('916', 'Good Will Hunting', '1977', '126', 'English', '1998-06-03', 'UK');
```

```
INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('917', 'Deliverance', '1972', '109', 'English', '1982-10-05', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('918', 'Trainspotting', '1996', '94', 'English', '1996-02-23', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('919', 'The Prestige', '2006', '130', 'English', '2006-11-10', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('920', 'Donnie Darko', '2001', '113', 'English', '', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('921', 'Slumdog Millionaire', '2008', '120', 'English', '2009-01-09', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('922', 'Aliens', '1986', '137', 'English', '1986-08-29', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('923', 'Beyound the Sea', '2004', '118', 'English', '2004-11-26', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('924', 'Avatar', '2009', '162', 'English', '2009-12-17', 'UK');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('926', 'Seven Samurai', '1954', '207', 'Japanese', '1954-04-26', 'JP');

INSERT INTO public.movie(
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
VALUES ('927', 'Spirited Away', '2001', '125', 'Japanese', '2003-09-12', 'UK');

INSERT INTO public.movie()
```

```
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
```

```
VALUES ('928', 'Back to the Future', '185', '116', 'English', '1985-12-04', 'UK');
```

```
INSERT INTO public.movie(
```

```
    mov_id, mov_title, mov_year, mov_time, mov_lang, mov_dt_rel, mov_rel_country)
```

```
VALUES ('925', 'Braveheart', '1995', '178', 'English', '1995-09-08', 'UK');
```

The screenshot shows the pgAdmin 4 interface with the 'Data Output' tab selected for the 'movie' table. The table has 14 rows of data, each with a unique ID from 901 to 914, along with movie titles, years, times, languages, release dates, and countries.

	mov_id	mov_title	mov_time	mov_year	mov_lang	mov_dt_rel	mov_rel_country
1	901	Vertigo	128	1958	English	1958-08-21	UK
2	902	The Innocents	100	1961	English	1962-02-19	SW
3	903	Lawrence of Arabia	216	1962	English	1962-12-11	UK
4	904	The Deer Hunter	183	1978	English	1979-03-08	UK
5	905	Amadeus	160	1984	English	1985-01-07	UK
6	906	Blade Runner	117	1982	English	1982-09-09	UK
7	907	Eye Wide Shut	159	1999	English	1995-08-25	UK
8	908	The Usual Suspects	159	1995	English	[null]	UK
9	909	Chinatown	130	1974	English	1975-08-09	UK
10	910	Bogie Nights	155	1997	English	1998-02-16	UK
11	911	Annie Hall	93	1977	English	1977-04-20	USA
12	912	Princess Mononoke	155	1997	Japanese	2001-10-19	UK
13	913	The Shawshank Redem...	142	1994	English	1995-02-17	UK
14	914	American Beauty	122	1999	English	[null]	UK

The screenshot shows the pgAdmin 4 interface with the 'Data Output' tab selected for the 'movie' table. The table has 28 rows of data, each with a unique ID from 915 to 928, along with movie titles, years, times, languages, release dates, and countries.

	mov_id	mov_title	mov_time	mov_year	mov_lang	mov_dt_rel	mov_rel_country
1	915	Titanic	194	1997	English	1998-01-23	UK
16	916	Good Will Hunting	126	1977	English	1998-06-03	UK
17	917	Deliverance	109	1972	English	1982-10-05	UK
18	918	Trainspotting	94	1996	English	1996-02-23	UK
19	919	The Prestige	130	2006	English	2006-11-10	UK
20	920	Donnie Darko	113	2001	English	[null]	UK
21	921	Slumdog Millionaire	120	2008	English	2009-01-09	UK
22	922	Aliens	137	1986	English	1986-08-29	UK
23	923	Beyond the Sea	118	2004	English	2004-11-26	UK
24	924	Avatar	162	2009	English	2009-12-17	UK
25	926	Seven Samuri	207	1954	Japanese	1954-04-26	JP
26	927	Spirited Away	125	2011	Japanese	2003-09-12	UK
27	928	Back to the Future	116	185	English	1995-12-04	UK
28	925	Braveheart	178	1995	English	1995-09-08	UK

```
INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('201', 'Fred', 'Caravanhitch');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('202', 'Jackie', 'Claytonburry');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('203', 'Greene', 'Lyon');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('204', 'Miguel', 'Camino');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('205', 'George', 'Forman');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('206', 'Antartic', 'Scott');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('207', 'Stanlee', 'Carbrick');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('208', 'Bryon', 'Sanger');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('209', 'Roman', 'Polanski');
```

```
INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('210', 'Paul', 'Thomus Anderson');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('211', 'Woody', 'Allan');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('212', 'Hayao', 'Miyazaki');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('213', 'Frank', 'Darabont');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('214', 'Sam', 'Mandes');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('215', 'James', 'Cameron');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('216', 'Fred', 'Gun Van Sant');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('217', 'John', 'Boreman');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('218', 'Danne', 'Boyle');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('219', 'Christoher', 'Nolan');

INSERT INTO public.director(
```

```

dir_id, dir_fname, dir_lname)
VALUES ('220', 'Richard', 'Kelly');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('221', 'Kevin', 'Spacey');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('222', 'Andrie', 'Tarkovsky');

INSERT INTO public.director(
    dir_id, dir_fname, dir_lname)
VALUES ('223', 'Peter', 'Jackson');

```

The screenshot shows the pgAdmin 4 interface. The left sidebar lists database objects: Materialized Views, Procedures, Sequences, Tables (9), rating, reviewer, and Trigger Functions. The 'reviewer' table is currently selected. The main area has tabs for Properties, SQL, Statistics, Dependents, and Data Output. The Data Output tab is active, showing a table with columns: dir_id [PK] integer, dir_fname character varying, and dir_lname character varying. The table contains 15 rows of data, starting with 1, 201, Fred and ending with 15, 215, James. Below the table, a query editor window is open, displaying several SQL statements related to the movie table.

dir_id	dir_fname	dir_lname
1	201	Fred
2	202	Jackie
3	203	Greene
4	204	Miguel
5	205	George
6	206	Antartic
7	207	Stanlee
8	208	Bryon
9	209	Roman
10	210	Paul
11	211	Woody
12	212	Hayao
13	213	Frank
14	214	Sam
15	215	James

dir_id	dir_fname	dir_lname
10	Paul	Thomus Anderson
11	Woody	Allan
12	Hayao	Miyazaki
13	Frank	Darabont
14	Sam	Mandes
15	James	Cameron
16	Fred	Gun Van Sant
17	John	Boreman
18	Danne	Boyle
19	Christoher	Nolan
20	Richard	Kelly
21	Kevin	Spacey
22	Andrie	Tarkovsky
23	Peter	Jackson

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('201', '901');

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('202', '902');

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('203', '903');

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('204', '904');

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('205', '905');

INSERT INTO public.movie_direction(

 dir_id, mov_id)

 VALUES ('206', '906');

INSERT INTO public.movie_direction(

```
        dir_id, mov_id)
VALUES ('207', '907');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('208', '908');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('209', '909');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('210', '910');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('211', '911');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('212', '912');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('213', '913');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('214', '914');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('215', '915');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
VALUES ('218', '918');

INSERT INTO public.movie_direction(
        dir_id, mov_id)
```

```

VALUES ('216', '916');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('217', '917');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('219', '919');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('220', '920');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('218', '921');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('215', '922');

INSERT INTO public.movie_direction(
    dir_id, mov_id)
VALUES ('221', '923');

```

http://127.0.0.1:64267/browser/

Apps Maps News Gmail FedUni - Federation... YouTube Barista>All rou... Netflix Barcats - The Future... My Applications...

gAdmin File Object Tools Help

Assignmnet 2/postgres@PostgreSQL 12 *

Browser

- > actor
- > director
- > genres
- > movie
- > movie_cast
- > movie_direction
- > movie_genres
- > rating
- > reviewer
 - > Columns
 - > Constraints
 - > Indexes
 - > RLS Policies
 - > Rules

Dependencies

	Name	Restriction
schema	public	normal

Query Editor

```
1 select * from movie_direction
```

Data Output Explain Messages

	dir_id	mov_id
1	201	901
2	202	902
3	203	903
4	204	904
5	205	905
6	206	906
7	207	907
8	208	908
9	209	909
10	210	910
11	211	911
12	212	912
13	213	913

The screenshot shows the pgAdmin 4 interface. On the left, there's a tree view of the database schema under 'Tables'. A table named 'movie_direction' is selected. To the right of the tree view is a 'Properties' tab and a 'SQL' tab. Below these is a 'Data Output' table showing the results of a query:

	dir_id	mov_id
9	209	909
10	210	910
11	211	911
12	212	912
13	213	913
14	214	914
15	215	915
16	218	918
17	216	916
18	217	917
19	219	919
20	220	920
21	218	921
22	215	922
23	221	923

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
```

```
    VALUES ('1001', 'Action');
```

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
```

```
    VALUES ('1002', 'Adventure');
```

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
```

```
    VALUES ('1003', 'Aviation');
```

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
```

```
    VALUES ('1004', 'Biography');
```

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
```

```
    VALUES ('1005', 'Comedy');
```

```
INSERT INTO public.genres(
```

```
    gen_id, gen_title)
VALUES ('1006', 'Crime');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1007', 'Drama');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1008', 'Horror');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1009', 'Music');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1010', 'Mystery');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1011', 'Romance');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1012', 'Thriller');

INSERT INTO public.genres(
    gen_id, gen_title)
VALUES ('1013', 'War');
```

The screenshot shows a PostgreSQL database interface. On the left, a tree view of schema objects is visible, with 'genres' selected. Below it, a table named 'public.genres' is shown with columns 'gen_id' and 'gen_title'. The data output shows 13 rows of genre names. In the center, the 'Query Editor' contains several SQL statements related to genres and movie_direction. The bottom status bar indicates 'Run here to search'.

gen_id	[PK] integer	gen_title
1	1001	Action
2	1002	Adventure
3	1003	Aviation
4	1004	Biography
5	1005	Comedy
6	1006	Crime
7	1007	Drama
8	1008	Horror
9	1009	Music
10	1010	Mystery
11	1011	Romance
12	1012	Thriller
13	1013	War

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('922', '1001');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('917', '1002');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('903', '1002');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('912', '1003');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('911', '1005');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('908', '1006');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('913', '1006');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('926', '1007');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('928', '1007');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('918', '1007');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('921', '1007');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('902', '1008');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('923', '1009');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('907', '1010');

INSERT INTO public.movie_genres(
    mov_id, gen_id)
VALUES ('927', '1010');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('901', '1010');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('914', '1011');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('906', '1012');
```

```
INSERT INTO public.movie_genres(
```

```
    mov_id, gen_id)
```

```
VALUES ('904', '1013');
```

The screenshot shows the pgAdmin 4 interface. On the left, there's a sidebar with various database objects like cast, direction, genres, r, imns, constraints, indexes, Policies, and ts. A 'Restriction' section is set to 'normal'. The main area has a 'Tools' menu at the top. The central part is a 'Query Editor' window with the following content:

```
1 select * from movie_genres
```

Below the editor is a 'Data Output' tab showing the results of the query:

	mov_id	gen_id
1	922	1001
2	917	1002
3	903	1002
4	912	1003
5	911	1005
6	908	1006
7	913	1006
8	926	1007
9	928	1007
10	918	1007
11	921	1007
12	902	1008

At the bottom of the editor, there are buttons for 'Copy' and 'Copy to Query Editor'. Below the editor, there's a history of previous queries:

- select * from movie_genres 01:46:42
- INSERT INTO public.movie_genres(mov_id, gen... 01:46:06
- select * from genres

```
INSERT INTO public.reviewer(
```

```
    rev_id, rev_name)
```

```
VALUES ('9001', 'Righty Sock');
```

```
INSERT INTO public.reviewer(
```

```
    rev_id, rev_name)
```

```
VALUES ('9002', 'Jack Malvern');
```

```
INSERT INTO public.reviewer(
```

```
    rev_id, rev_name)
```

```
VALUES ('9003', 'Flagrant Baronessa');
```

```
INSERT INTO public.reviewer(
```

```
    rev_id, rev_name)
```

```
VALUES ('9004', 'Alec Shaw');
```

```
INSERT INTO public.reviewer(
```

```
    rev_id)
```

```
VALUES ('9005');
```

```
INSERT INTO public.reviewer(
```

```
    rev_id, rev_name)
```

```
VALUES ('9006', 'Victor Woeltjen');
```

```
INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9007', 'Simon Wright');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9008', 'Neal Wruck');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9009', 'Paul Monks');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9010', 'Mike Salvati');

INSERT INTO public.reviewer(
    rev_id)
VALUES ('9011');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9012', 'Wesley S. Walker');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9013', 'Sasha GOldshtein');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9014', 'Josh Cates');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9015', 'Krug Stillo');

INSERT INTO public.reviewer(
    rev_id, rev_name)
VALUES ('9016', 'Scott LeBrun');

INSERT INTO public.reviewer(
```

```

rev_id, rev_name)
VALUES ('9017', 'Hannah Steele');

INSERT INTO publicreviewer(
    rev_id, rev_name)
VALUES ('9018', 'Vincent Cadena');

INSERT INTO publicreviewer(
    rev_id, rev_name)
VALUES ('9019', 'Brandit Sponseller');

INSERT INTO publicreviewer(
    rev_id, rev_name)
VALUES ('9020', 'Richard Adams');

```

The screenshot shows a database management interface with a sidebar containing a tree view of tables and a main area for querying data.

Sidebar (Tables):

- Tables (9)
 - actor
 - director
 - genres
 - movie
 - movie_cast
 - movie_direction
 - movie_genres
 - rating
 - reviewer

Query Editor:

```

1 select * from reviewer

```

Data Output

	rev_id [PK] integer	rev_name character varying
1	9001	Rightly Sock
2	9002	Jack Malvern
3	9003	Flagrant Baronessa
4	9004	Alec Shaw
5	9005	[null]
6	9006	Victor Woeltjen
7	9007	Simon Wright
8	9008	Neal Wruck
9	9009	Paul Monks
10	9010	Mike Salvati
11	9011	[null]
12	9012	Wesley S. Walker
13	9013	Sasha GOldshein

The screenshot shows a database interface with a sidebar containing a tree view of tables: actor, director, genres, movie, movie_cast, movie_direction, movie_genres, rating, and reviewer. The 'reviewer' table is selected and expanded, showing options for Columns, Constraints, Indexes, RLS Policies, and Rules.

The main area is a 'Query Editor' with the following SQL query:

```
1 select * from reviewer
```

The 'Data Output' tab displays the results of the query:

	rev_id	rev_name
	[PK] integer	character varying
9	9009	Paul Monks
10	9010	Mike Salvati
11	9011	[null]
12	9012	Wesley S. Walker
13	9013	Sasha Goldstein
14	9014	Josh Cates
15	9015	Krug Stilo
16	9016	Scott LeBrun
17	9017	Hannah Steele
18	9018	Vincent Cadena
19	9019	Brandit Sponseller
20	9020	Richard Adams

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars, num_o_rating)
VALUES ('901', '9001', '8.40', '263575');
```

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars, num_o_rating)
VALUES ('902', '9002', '7.90', '20207');
```

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars, num_o_rating)
VALUES ('903', '9003', '8.30', '202778');
```

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars, num_o_rating)
VALUES ('906', '9005', '8.20', '484746');
```

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars)
VALUES ('924', '9006', '7.30');
```

```
INSERT INTO public.rating(
```

```
    rev_id, mov_id, rev_stars, num_o_rating)
VALUES ('908', '9007', '8.60', '779489');
```

```
INSERT INTO public.rating(
    rev_id, mov_id, num_0_rating)
VALUES ('909', '9008', '227235');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('910', '9009', '3.00', '195961');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('911', '9010', '8.10', '203875');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars)
VALUES ('912', '9011', '8.40');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('914', '9013', '7.00', '862618');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('915', '9001', '7.70', '830095');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars)
VALUES ('916', '9014', '4.40');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('925', '9015', '7.70', '81328');

INSERT INTO public.rating(
    rev_id, mov_id, num_0_rating)
VALUES ('918', '9016', '580301');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('920', '9017', '8.10', '609451');

INSERT INTO public.rating(
```

```

rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('921', '9018', '8.00', '609451');
```

```

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('922', '9091', '8.40', '511613');

INSERT INTO public.rating(
    rev_id, mov_id, rev_stars, num_0_rating)
VALUES ('923', '9013', '6.70', '13091');
```

The screenshot shows a database management interface with a sidebar navigation menu and a central workspace. The workspace is divided into several panes: a 'Query Editor' pane containing the SQL code, a 'Data Output' pane displaying the results of the query, and a 'Query History' pane showing previous queries.

Table Structure:

	mov_id	rev_id	rev_stars	num_0_rating
1	9001	901	8.40	263575
2	9002	902	7.90	20207
3	9003	903	8.30	202778
4	9005	906	8.20	484746
5	9006	924	7.30	[null]
6	9007	908	8.60	779489
7	9008	909	[null]	227235
8	9009	910	3.00	195961
9	9010	911	8.10	203875
10	9011	912	8.40	[null]
11	9013	914	7.00	862618
12	9001	915	7.70	830095
13	9014	916	4.40	[null]
14	9015	925	7.70	81328
15	9016	918	[null]	580301
16	9017	920	8.10	609451
17	9018	921	8.00	609451

Query Editor:

```

1 select * from rating;
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

Query History:

```

Today - 10/13/2020
▶ select * from rating;
16:46:44
▶ INSERT INTO public.rating
16:46:18
▶ INSERT INTO public.rating
16:42:21
```

Q ▾ Data Output Explain

	mov_id integer	rev_id integer	rev_stars character varying	num_0_rating integer
ting;	10	9017	920	0.10
	17	9018	921	8.00
	18	9091	922	8.40
	19	9013	923	6.70
	20	9001	901	8.40
	21	9002	902	7.90
	22	9003	903	8.30
s	23	9005	906	8.20
	24	9006	924	7.30
	25	9007	908	8.60
ernally by	26	9008	909	[null]
	27	9009	910	3.00
ing;	28	9010	911	8.10
	29	9011	912	8.40
c.rating	30	9013	914	7.00
				862618
				select * from rating;

SQL Statistics Dependents Assignmnet 2... Assignmnet 2... Assignmnet 2... Assignmnet 2... Assignmnet 2... Back Forward Home

Query Editor

```
1 select * from rating;
2
```

Data Output Explain

	mov_id integer	rev_id integer	rev_stars character varying	num_0_rating integer
	25	9007	908	8.60
	26	9008	909	[null]
	27	9009	910	3.00
	28	9010	911	8.10
	29	9011	912	8.40
	30	9013	914	7.00
	31	9001	915	7.70
				830095
Query Editor Notifications	32	9014	916	4.40
Query History	33	9015	925	7.70
Show queries generated internally by	34	9016	918	[null]
Today - 10/13/2020	35	9017	920	8.10
▶ select * from rating;	36	9018	921	8.00
16:46:44	37	9091	922	8.40
► TNSFRT TNTO public.rating	38	9013	923	6.70

Q.3

Query Editor

```
1 select mov_title, mov_year
2 from movie
3 where mov_title like '%the%'
```

Query Editor Notifications

Data Output Explain

	mov_title	mov_year
1	Beyond the Sea	2004
2	Back to the Future	185

Q.No 4

Properties SQL Statistics Dependents mad assignmnet/postgres@PostgreSQL 12 *

Messages Data Output Explain

Successfully run. Total query runtime: 55 msec.
19 rows affected.

	mov_title	gen_title
1	Aliens	Action
2	Deliverance	Adventure
3	Lawremnce of Arab...	Adventure
4	Princess Mononok...	Aviation
5	Annie Hall	Comedy
6	The Usual Suspect...	Crime
7	The Shawshank Re...	Crime
8	Seven Samuri	Drama
9	Back to the Future	Drama
10	Trainspotting	Drama
11	Slumdog Millionair...	Drama
12	The Innocents	Horror
13	Beyound the Sea	Music
14	Eye Wide Shut	Mysty

Q.NO 5

select AVG(num_0_rating) AS "avg rating", rev_stars as "avg review"

from rating

group by rev_stars

Properties SQL Statistics Dependents mad assignmnet/postgres@PostgreSQL 12 *

Query Editor

```
select AVG(num_0_rating) AS "avg rating", rev_stars as "avg review"
from rating
group by rev_stars
```

Data Output

	avg rating	avg review
1	8.000000000000	[null]
2	1.000000000000	8.00
3	6.000000000000	8.20
4	8.000000000000	8.30
5	10000000000000	6.70
6	9.000000000000	8.60
7	[null]	7.30

http://127.0.0.1:52189/browser/

pgAdmin File Object Tools Help

Browser

Rating Editor

```
select AVG(num_0_rating) AS "avg rating", rev_stars as "avg review"
from rating
group by rev_stars
```

Data Output

avg rating	avg review
8	3.000000000000
9	1.500000000000
10	7.000000000000
11	1.000000000000
12	4.000000000000
13	2.000000000000
14	8.000000000000

Query Editor Notification

```
Today - 10/14/2020
00:53:35
▶ select AVG(num_0_rating) AS "avg rating", rev_stars as "avg review"
from rating
group by rev_stars
00:42:28
▶ select act_fname,act_lname from actor where ...
00:42:13
▶ select act_fname,act_lname from actor where ...
```

Copy Copy to Query Editor

select AVG(num_0_rating) AS "avg rating", rev_stars as "avg review" from rating group by rev_stars

Messages

Q.NO 6

http://127.0.0.1:52189/browser/

pgAdmin File Object Tools Help

Browser

Tables Editor

```
select actor.act_fname, actor.act_lname, movie.mov_title
from actor
inner join movie_cast
on actor.actor_id=movie_cast.actor_id
inner join movie
on movie_cast.movie_id = movie.movie_id
where not mov_title = 'deliverance'
```

Data Output Explain

act_fname	act_lname	mov_title
James	Stewart	Vertigo
Deborah	Kerr	The Innocents
Peter	O'Toole	Lawrence of Arabia
Robert	De Niro	The Deer Hunter
F.Murray	Abraham	Amadeus
Harrison	Ford	Blade Runner
Nicole	Kidman	Eye Wide Shut
Stephan	Baldwin	The Usual Suspects
Mark	Wahlberg	Bogie Nights
Woody	Allen	Annie Hall
Clarie	Danes	Princess Mononoke
Tim	Robbins	The Shawshank Re...
Kevin	Spacy	American Beauty
Kate	Winslet	Titanic

Query Editor Notifications

```
Today - 10/13/2020
21:54:33
▶ select actor.act_fname, actor.act_lname, movie.mov_title
from actor
inner join movie_cast
on actor.actor_id=movie_cast.actor_id
inner join movie
on movie_cast.movie_id = movie.movie_id
where not mov_title = 'deliverance'
21:46:07
▶ select movie.mov_title,genres.gen_title from ...
21:22:59
▶ INSERT INTO public.rating( mov_id, rev_id, r...
```

Q.No 7

```

SELECT mov_title
FROM reviewer, movie, rating, rating r2
WHERE rating.mov_id= movie.mov_id
AND reviewer.rev_id= rating.rev_id
AND rating.rev_id= r2.rev_id
GROUP By mov_title having count (*) >1 ;

```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the 'Tables' section, including actor, director, genres, movie, movie_cast, rating, and role. The right pane contains a 'Query Editor' window with the provided SQL query. Below the editor is a 'Data Output' pane showing the results of the query:

```

mov_title
Chinatown
Lawrence of Arabia
The Innocents
Aliens
Annie Hall
Good Will Hunting
Beyond the Sea
The Usual Suspects
Amadeus
American Beauty
Bogie Nights
Deliverance
The Shawshank Re...
Vertigo

```

Q.no 8

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the 'Tables' section, including actor, director, movie, and rating. The right pane contains a 'Query Editor' window with the following SQL query:

```

select mov_title, count(act_id) as NumberOfActors
from movie, movie_cast
where movie.mov_id = movie_cast.mov_id
group by mov_title
order by NumberOfActors desc;

```

Below the editor is a 'Data Output' pane showing the results of the query:

mov_title	NumberOfActors
Chinatown	1
Lawrence of Arabia	1
The Innocents	1
Aliens	1
Annie Hall	1
Good Will Hunting	1
Beyond the Sea	1
The Usual Suspects	1
Amadeus	1
American Beauty	1
Bogie Nights	1
Deliverance	1
The Shawshank Re...	1
Vertigo	1

```

1 select mov_title, count(act_id) as NumberOfActors
2 from movie, movie_cast
3 where movie.mov_id = movie_cast.mov_id
4 group by mov_title
5 order by NumberOfActors desc;

```

mov_title	numberofactors
American Beauty	1
Bogie Nights	1
Deliverance	1
The Shawshank Re...	1
Vertigo	1
Titanic	1
Donnie Darko	1
Blade Runner	1
Trainspotting	1
Eye Wide Shut	1
The Prestige	1
Slumdog Millionair...	1
The Deer Hunter	1
Princess Mononok...	1

Q,n0 9

```

select mov_title, rev_stars, act_fname, act_lname
from movie m, rating r, movie_cast mc, actor a
where m.mov_id = r.mov_id
and m.mov_id = mc.mov_id
and mc.act_id = a.act_id
and rev_stars > 8 and act_fname = 'Christian'

```

Q.10

I alter the table to add the new column dob and then insert the file using update command

```
alter table director
```

```
add column dob date;
```

```
UPDATE public.director
```

```

SET dir_id='201', dir_fname='Alfred', dir_lname='Hitchcock', dob = '1964-08-12'
WHERE dir_id = '201';

```

```
UPDATE public.director
```

```
SET dir_id='202', dir_fname='Jack', dir_lname='Clayton', dob = '1955-11-22'
```

```
WHERE dir_id = '202';
```

UPDATE public.director

```
SET dir_id='203', dir_fname='David', dir_lname='Lean', dob = '1969-5-15'
```

```
WHERE dir_id = '203';
```

UPDATE public.director

```
SET dir_id='204', dir_fname='Michael', dir_lname='Cimino', dob = '1959-11-08'
```

```
WHERE dir_id = '204';
```

UPDATE public.director

```
SET dir_id='205', dir_fname='Milos', dir_lname='Forman', dob = '1988-09-12'
```

```
WHERE dir_id = '205';
```

UPDATE public.director

```
SET dir_id='206', dir_fname='Ridley', dir_lname='Scott', dob = '1957-02-07'
```

```
WHERE dir_id = '206';
```

UPDATE public.director

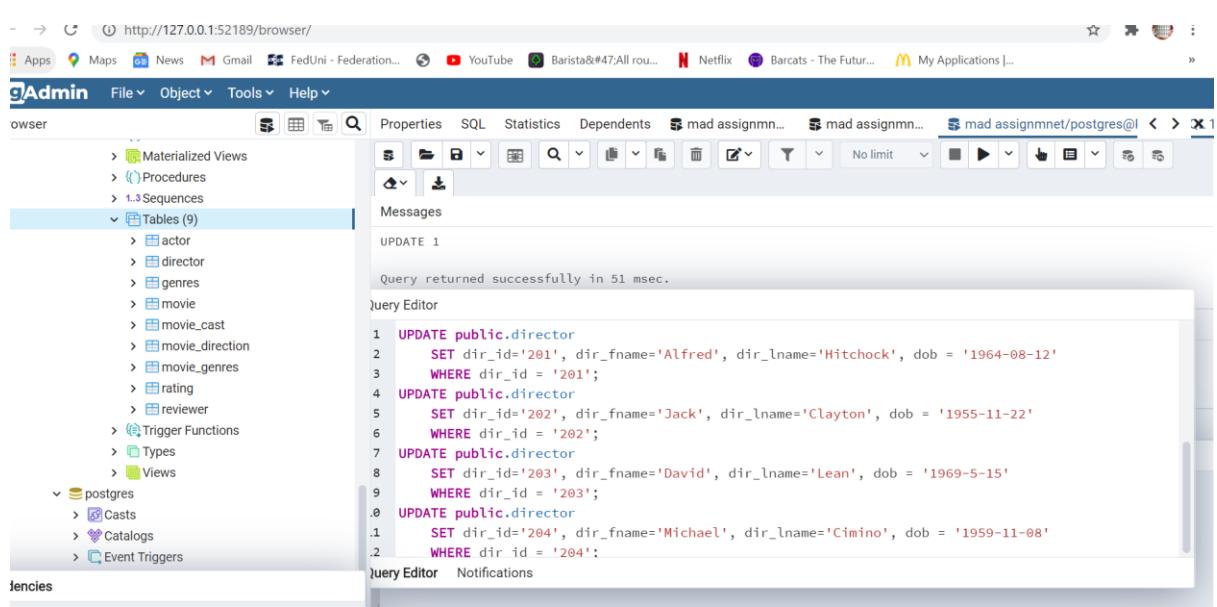
```
SET dir_id='207', dir_fname='Stanley', dir_lname='Kubrick', dob = '1967-11-08'
```

```
WHERE dir_id = '207';
```

UPDATE public.director

```
SET dir_id='208', dir_fname='Bryan', dir_lname='Singer', dob = '1960-02-28'
```

```
WHERE dir_id = '208';
```



The screenshot shows the gAdmin PostgreSQL interface. The left sidebar lists databases (owser, postgres) and their objects (Materialized Views, Procedures, Sequences, Tables, Trigger Functions, Casts, Catalogs, Event Triggers). The 'Tables' section is expanded, showing nine tables: actor, director, genres, movie, movie_cast, movie_direction, movie_genres, rating, reviewer. The main area is the Query Editor, which displays the following SQL code:

```
UPDATE public.director
SET dir_id='201', dir_fname='Alfred', dir_lname='Hitchcock', dob = '1964-08-12'
WHERE dir_id = '201';
UPDATE public.director
SET dir_id='202', dir_fname='Jack', dir_lname='Clayton', dob = '1955-11-22'
WHERE dir_id = '202';
UPDATE public.director
SET dir_id='203', dir_fname='David', dir_lname='Lean', dob = '1969-5-15'
WHERE dir_id = '203';
UPDATE public.director
SET dir_id='204', dir_fname='Michael', dir_lname='Cimino', dob = '1959-11-08'
WHERE dir_id = '204';
```

The message bar at the top right indicates "Query returned successfully in 51 msec."

```

    select * from director
  
```

The screenshot shows the PgAdmin 4 interface. The left sidebar has a tree view of the database schema, including 'Tables (9)' which is expanded to show 'actor', 'director', 'genres', 'movie', 'movie_cast', 'movie_direction', 'movie_genres', 'rating', and 'reviewer'. Below this is a section for 'Trigger Functions', 'Types', and 'Views'. The right side has a 'Properties' tab, an 'SQL' tab with the query editor containing the provided SQL code, a 'Statistics' tab, and a 'Dependents' tab. A 'Data Output' table shows the results of the query, listing 10 rows of director information. The table has columns: dir_id [PK integer], dir_fname character (20), dir_lname character (20), and dob date.

dir_id	dir_fname	dir_lname	dob
14	222	Andrie	Tarkovsky
15	223	Peter	Jackson
16	201	Alfred	Hitchcock
17	202	Jack	Clayton
18	203	David	Lean
19	204	Michael	Cimino
20	205	Milos	Forman
21	206	Ridley	Scott
22	207	Stanley	Kubrick
23	208	Bryan	Singer

q.no 11

```

select director.dir_fname, dir_lname, movie.mov_year
from (director inner join movie_direction on director.dir_id = movie_direction.dir_id)
inner join movie on movie_direction.mov_id = movie.mov_id
where movie.mov_id NOT IN (
  select mov_id from movie where mov_year >= 2000 );
  
```

```

1 select director.dir_fname, dir_lname, movie.mov_year
2 from (director inner join movie_direction on director.dir_id = movie_direction.dir_id)
3 inner join movie on movie_direction.mov_id = movie.mov_id
4 where movie.mov_id NOT IN (
5 select mov_id from movie where mov_year >= 2000 );
  
```

This screenshot shows the same PgAdmin 4 interface as the first one. The browser pane is identical. The query editor pane now contains a more complex query involving multiple joins and a NOT IN subquery. The results are displayed in a 'Data Output' table, which lists 10 rows of director and movie information. The table has columns: dir_fname character (20), dir_lname character (20), and mov_year integer.

dir_fname	dir_lname	mov_year
Alfred	Hitchcock	1958
Jack	Clayton	1961
David	Lean	1962
Michael	Cimino	1978
Milos	Forman	1984
Ridley	Scott	1982
Stanley	Kubrick	1999
Bryan	Singer	1995
Roman	Polanski	1974
Paul	Thomas Anderson	1997

Q.No 12

```
select movie.mov_title  
from movie_cast  
inner join movie  
on movie_cast.mov_id=movie.mov_id
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the 'Browser' tab, including 'Tables (9)' which contains 'actor', 'director', 'genres', 'movie', 'movie_cast', 'movie_direction', 'movie_genres', 'rating', and 'reviewer'. Below this is 'Trigger Functions', 'Types', and 'Views'. The right side has a 'Query Editor' window with the following SQL code:

```
1 select movie.mov_title  
2 from movie_cast  
3 inner join movie  
4 on movie_cast.mov_id=movie.mov_id
```

Below the Query Editor is a 'Data Output' window showing the results of the query:

mov_title
Vertigo
The Innocents
Lawremnce of Arab...
The Deer Hunter
Amadeus
Blade Runner
Eye Wide Shut
The Usual Suspect...
Bogie Nights
Annie Hall
Princess Mononok...

Q.no 13

```
select sum(movie.mov_time), actor.act_fname  
from (movie_cast inner join movie on movie_cast.mov_id = movie.mov_id)  
inner join actor on movie_cast.act_id = actor.act_id  
group by movie_cast.act_id, actor.act_fname
```

```

1 select sum(movie.mov_time), actor.act_fname
2 from (movie_cast inner join movie on movie_cast.mov_id = movie.mov_id)
3 inner join actor on movie_cast.act_id = actor.act_id
4 group by movie_cast.act_id, actor.act_fname

```

	sum	act_fname
	bigint	character (20)
1	109	Jon
2	142	Tim
3	155	Clarie
4	130	Jack
5	159	Stephan
6	137	Signorey
7	120	Dev
8	128	James
9	122	Kevin
10	183	Robert
11	159	Nicole

Q.no 14

```

select count (rating.rev_id), movie.mov_title, rating.rev_stars
from rating inner join movie on rating.mov_id = movie.mov_id
group by rating.mov_id, movie.mov_title, rating.rev_stars
order by rating.rev_stars

```

```

1 select count (rating.rev_id), movie.mov_title, rating.rev_stars
2 from rating inner join movie on rating.mov_id = movie.mov_id
3 group by rating.mov_id, movie.mov_title, rating.rev_stars
4 order by rating.rev_stars

```

	count	mov_title	rev_stars
	bigint	character (50)	character (90)
1	1	Boggle Nights	3.00
2	1	Good Will Hunting	4.40
3	1	Beyond the Sea	6.70
4	1	American Beauty	7.00
5	1	Avatar	7.30
6	1	Titanic	7.70
7	1	Braveheart	7.70
8	1	The Innocents	7.90
9	1	Slumdog Millionair...	8.00
10	1	Annie Hall	8.10
11	1	Donnie Dark	8.10
12	1	Blade Runner	8.20

Q.no 15

```

select act_fname,act_lname from actor
where act_id IN(
select act_id from movie_cast

```

where mov_id in (select mov_id from movie_direction

where dir_id in(

select dir_id from movie_direction

group by dir_id

having count(dir_id)>1));

```
1 select act_fname,act_lname from actor
2 where act_id IN(
3 select act_id from movie_cast
4 where mov_id in (select mov_id from movie_direction
5 where dir_id in(
6 select dir_id from movie_direction
7 group by dir_id
8 having count(dir_id)>1)));
```

act_fname	act_lname
Kate	Winslet
Ewan	McGregor
Dev	Patel
Signorey	Weaver

Q.No 16

```
1 select dir_fname, dir_lname, count(mov_id) as totalMovies
2 from director d, movie_direction l
3 where d.dir_id=l.dir_id
4 group by dir_fname, dir_lname
5 order by totalMovies desc;
```

dir_fname	dir_lname	totalMovies
James	Cameron	2
Danne	Boyle	2
Fred	Gun Van Sant	1
Hayao	Miyazaki	1
Jackie	Claytonburry	1
Paul	Thomas Anderson ...	1
Kevin	Spacey	1
Frank	Darabont	1
Greene	Lyon	1
Roman	Polanski	1
Miguel	Camino	1
Christoher	Nolan	1

The screenshot shows the pgAdmin 4 application window. On the left, the 'Browser' pane displays a tree view of database objects under the 'Tables (9)' section, including actor, director, genres, movie, movie_cast, movie_direction, movie_genres, rating, and reviewer. Below this is the 'Dependencies' section. The main area contains a 'Messages' tab with the text 'ditor' and a 'Data Output' tab showing a table of director names and counts. The table is as follows:

dir_name	dir_lname	totalmovies
Frank	Darabont	1
Greene	Lyon	1
Roman	Polanski	1
Miguel	Camino	1
Christopher	Nolan	1
John	Boreman	1
Bryon	Sanger	1
George	Forman	1
Fred	Caravanhitch	1
Sam	Mandes	1
Antartic	Scott	1
Woody	Allan	1
Richard	Kelly	1
Stanlee	Carbrick	1

Conclusion:

The assignment 2 has given me a tough time but taught me about the sql code which is most in data science and data modelling. Though the assignment was time consuming but the out put is always pleasure to watch. The use of pgadim in this assignment has given me overview of sql commands as well as importance of data entry. Along with it, it encourages me to explore internet resources.

References:

<https://www.pgadmin.org>

<https://www.w3schools.com/>