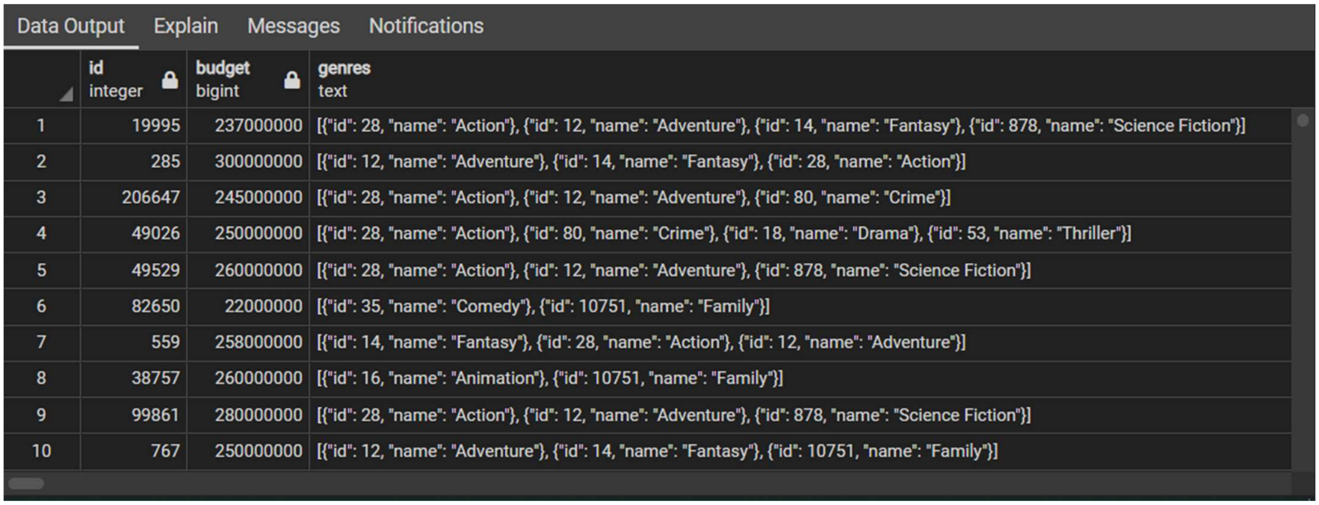
Data Cleaning in SQL

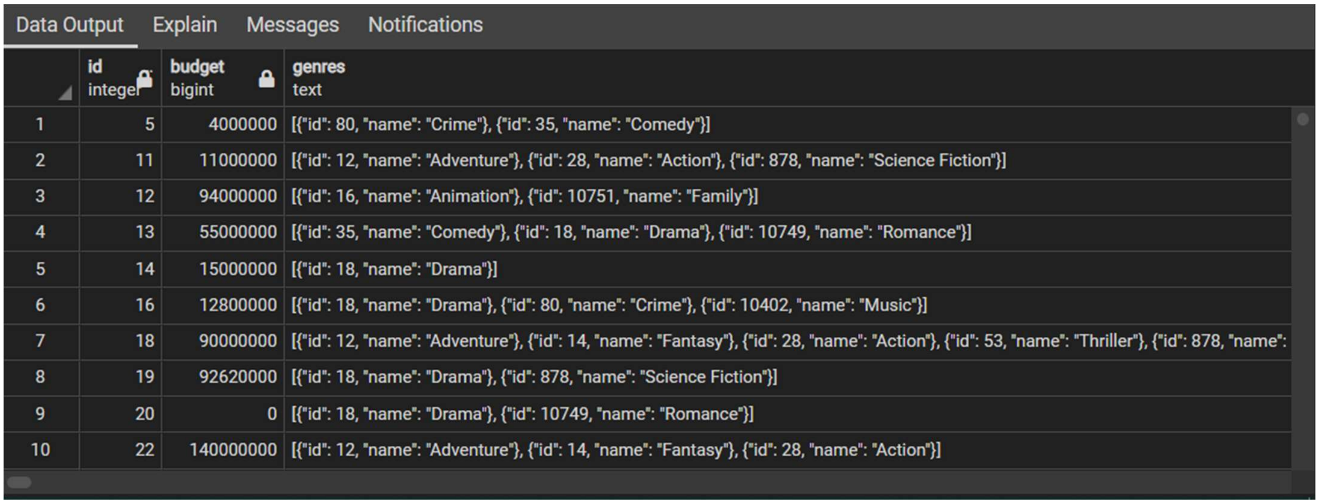
Task 1 : Cleaning the “movies” table (in PostGreSQL)

1) Output after Subtask 1



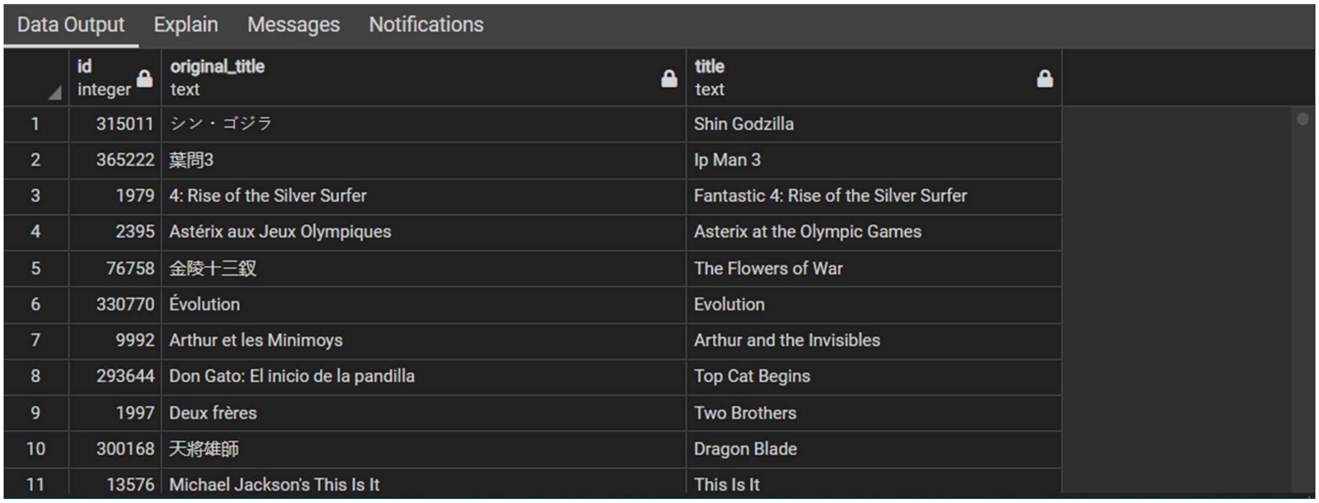
	id integer	budget bigint	genres text
1	19995	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 878, "name": "Science Fiction"}]
2	285	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}]
3	206647	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 80, "name": "Crime"}]
4	49026	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "name": "Crime"}, {"id": 18, "name": "Drama"}, {"id": 53, "name": "Thriller"}]
5	49529	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 878, "name": "Science Fiction"}]
6	82650	22000000	[{"id": 35, "name": "Comedy"}, {"id": 10751, "name": "Family"}]
7	559	258000000	[{"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]
8	38757	260000000	[{"id": 16, "name": "Animation"}, {"id": 10751, "name": "Family"}]
9	99861	280000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 878, "name": "Science Fiction"}]
10	767	250000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 10751, "name": "Family"}]

2) Output after Subtask 2



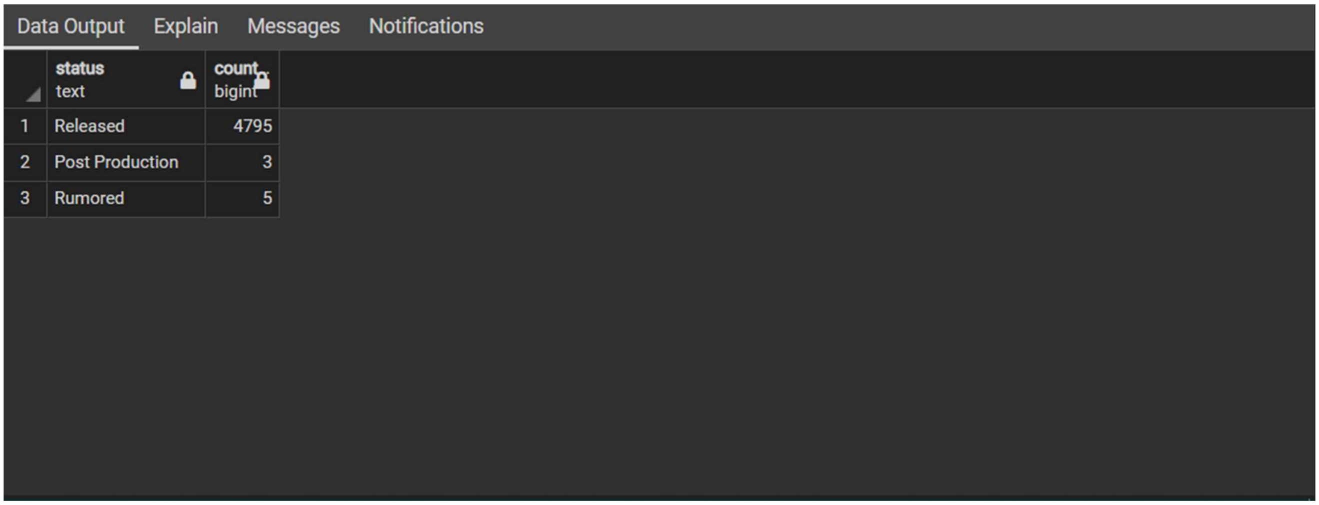
	id integer	budget bigint	genres text
1	5	4000000	[{"id": 80, "name": "Crime"}, {"id": 35, "name": "Comedy"}]
2	11	11000000	[{"id": 12, "name": "Adventure"}, {"id": 28, "name": "Action"}, {"id": 878, "name": "Science Fiction"}]
3	12	94000000	[{"id": 16, "name": "Animation"}, {"id": 10751, "name": "Family"}]
4	13	55000000	[{"id": 35, "name": "Comedy"}, {"id": 18, "name": "Drama"}, {"id": 10749, "name": "Romance"}]
5	14	15000000	[{"id": 18, "name": "Drama"}]
6	16	12800000	[{"id": 18, "name": "Drama"}, {"id": 80, "name": "Crime"}, {"id": 10402, "name": "Music"}]
7	18	90000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}, {"id": 53, "name": "Thriller"}, {"id": 878, "name":
8	19	92620000	[{"id": 18, "name": "Drama"}, {"id": 878, "name": "Science Fiction"}]
9	20	0	[{"id": 18, "name": "Drama"}, {"id": 10749, "name": "Romance"}]
10	22	140000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 28, "name": "Action"}]

3) Output after Subtask 3



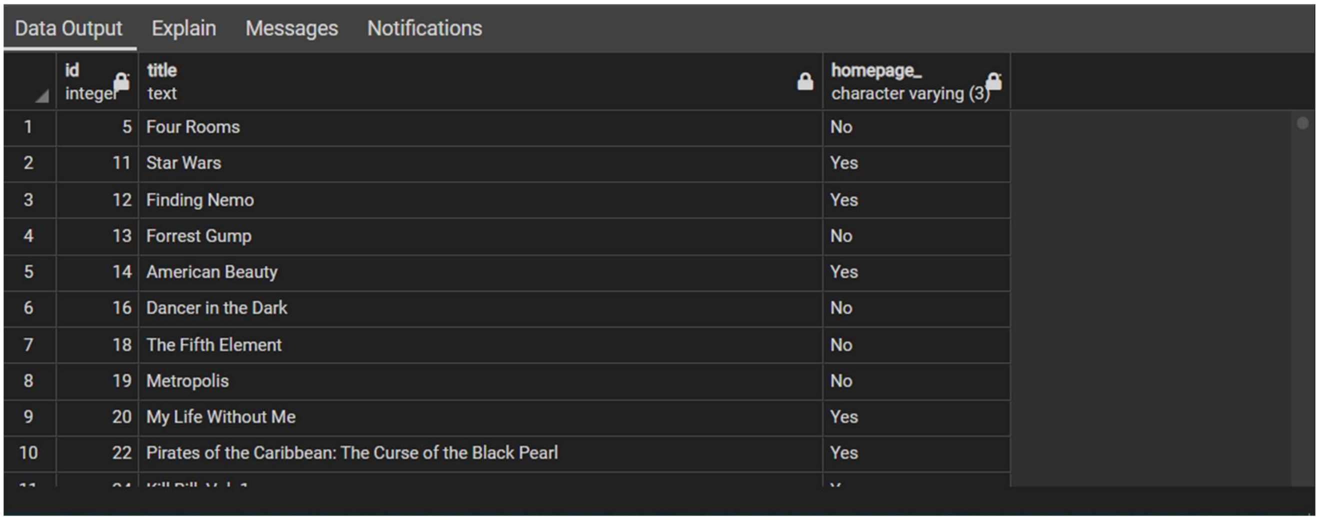
	id integer	original_title text	title text
1	315011	シン・ゴジラ	Shin Godzilla
2	365222	葉問3	Ip Man 3
3	1979	4: Rise of the Silver Surfer	Fantastic 4: Rise of the Silver Surfer
4	2395	Astérix aux Jeux Olympiques	Asterix at the Olympic Games
5	76758	金陵十三釵	The Flowers of War
6	330770	Évolution	Evolution
7	9992	Arthur et les Minimoys	Arthur and the Invisibles
8	293644	Don Gato: El inicio de la pandilla	Top Cat Begins
9	1997	Deux frères	Two Brothers
10	300168	天將雄師	Dragon Blade
11	13576	Michael Jackson's This Is It	This Is It

4) Output after Subtask 4



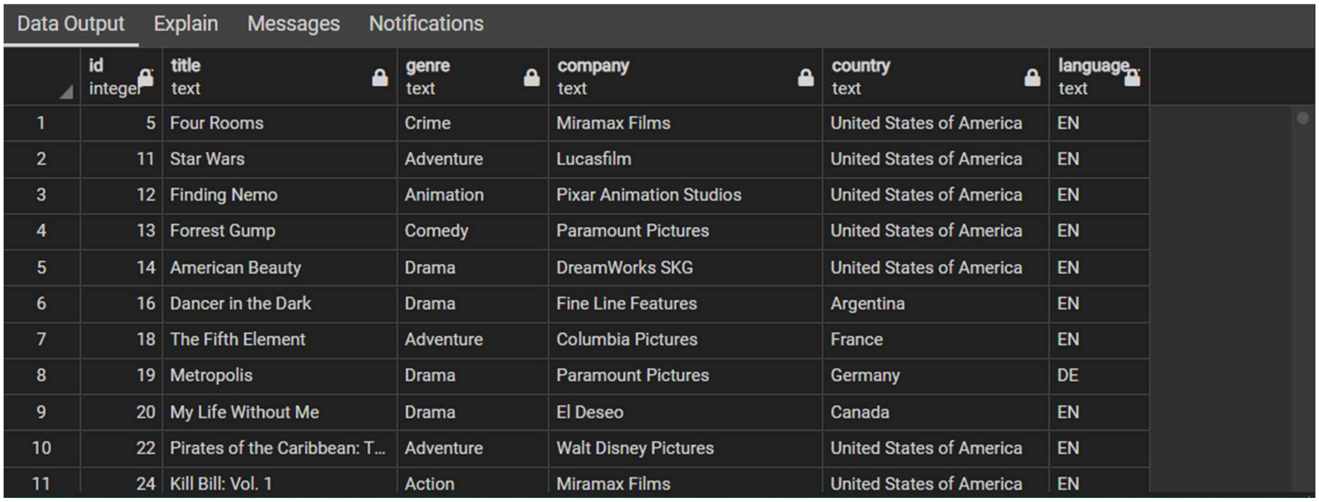
	status text	count bigint
1	Released	4795
2	Post Production	3
3	Rumored	5

5) Output after Subtask 6



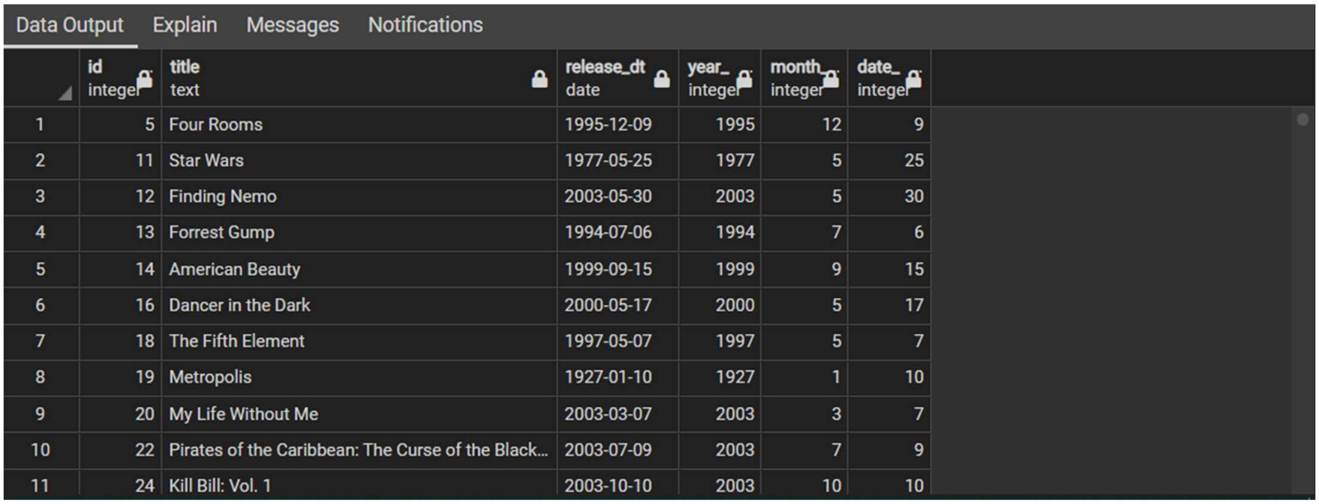
	id integer	title text	homepage_ character varying (3)
1	5	Four Rooms	No
2	11	Star Wars	Yes
3	12	Finding Nemo	Yes
4	13	Forrest Gump	No
5	14	American Beauty	Yes
6	16	Dancer in the Dark	No
7	18	The Fifth Element	No
8	19	Metropolis	No
9	20	My Life Without Me	Yes
10	22	Pirates of the Caribbean: The Curse of the Black Pearl	Yes
11	24	Kill Bill: Vol. 1	Y

6) Output after Subtask 7



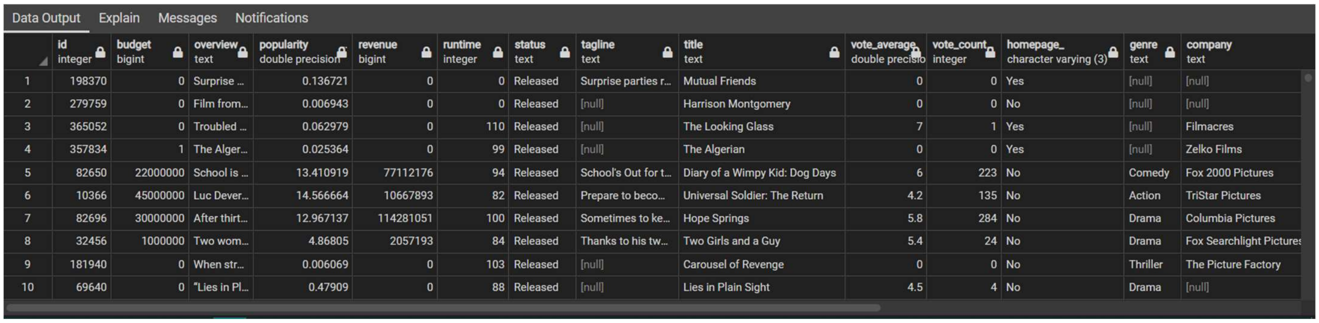
	id integer	title text	genre text	company text	country text	language text
1	5	Four Rooms	Crime	Miramax Films	United States of America	EN
2	11	Star Wars	Adventure	Lucasfilm	United States of America	EN
3	12	Finding Nemo	Animation	Pixar Animation Studios	United States of America	EN
4	13	Forrest Gump	Comedy	Paramount Pictures	United States of America	EN
5	14	American Beauty	Drama	DreamWorks SKG	United States of America	EN
6	16	Dancer in the Dark	Drama	Fine Line Features	Argentina	EN
7	18	The Fifth Element	Adventure	Columbia Pictures	France	EN
8	19	Metropolis	Drama	Paramount Pictures	Germany	DE
9	20	My Life Without Me	Drama	El Deseo	Canada	EN
10	22	Pirates of the Caribbean: T...	Adventure	Walt Disney Pictures	United States of America	EN
11	24	Kill Bill: Vol. 1	Action	Miramax Films	United States of America	EN

7) Output after Subtask 10

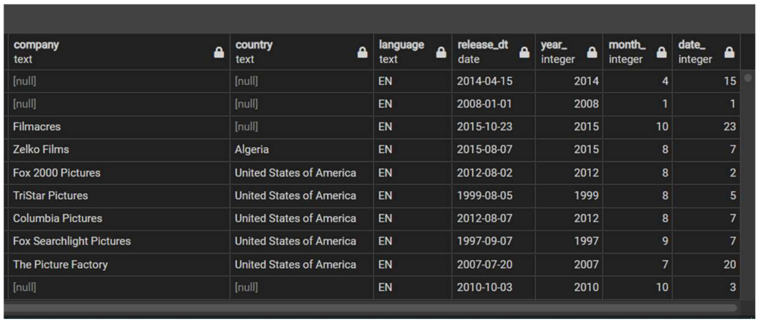


	id integer	title text	release_dt date	year_ integer	month_ integer	date_ integer
1	5	Four Rooms	1995-12-09	1995	12	9
2	11	Star Wars	1977-05-25	1977	5	25
3	12	Finding Nemo	2003-05-30	2003	5	30
4	13	Forrest Gump	1994-07-06	1994	7	6
5	14	American Beauty	1999-09-15	1999	9	15
6	16	Dancer in the Dark	2000-05-17	2000	5	17
7	18	The Fifth Element	1997-05-07	1997	5	7
8	19	Metropolis	1927-01-10	1927	1	10
9	20	My Life Without Me	2003-03-07	2003	3	7
10	22	Pirates of the Caribbean: The Curse of the Black...	2003-07-09	2003	7	9
11	24	Kill Bill: Vol. 1	2003-10-10	2003	10	10

8) Output after Subtask 12

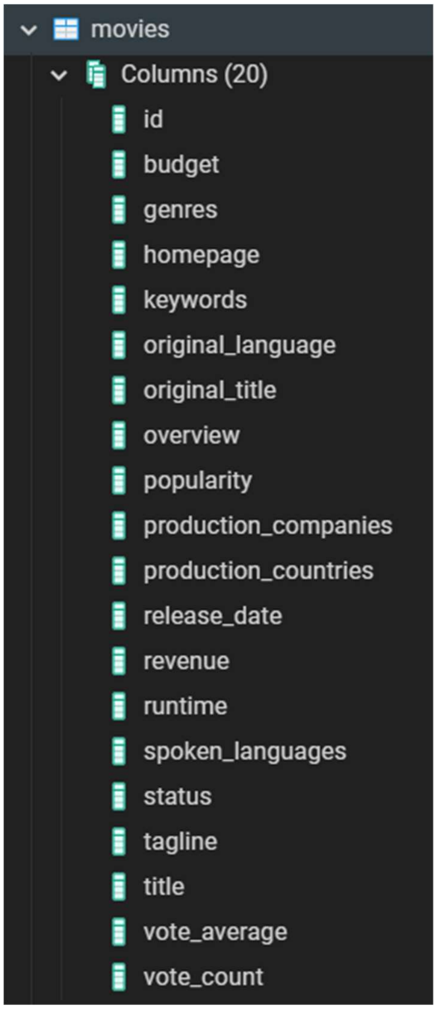


	id integer	budget bigint	overview text	popularity double precision	revenue bigint	runtime integer	status text	tagline text	title text	vote_average double precisio	vote_count integer	homepage_ character varying (3)	genre text	company text
1	198370	0	Surprise ...	0.136721	0	0	Released	Surprise parties r...	Mutual Friends	0	0	Yes	[null]	[null]
2	279759	0	Film from...	0.006943	0	0	Released	[null]	Harrison Montgomery	0	0	No	[null]	[null]
3	365052	0	Troubled ...	0.062979	0	110	Released	[null]	The Looking Glass	7	1	Yes	[null]	Filmacres
4	357834	1	The Alger...	0.025364	0	99	Released	[null]	The Algerian	0	0	Yes	[null]	Zelko Films
5	82650	22000000	School is ...	13.410919	77112176	94	Released	School's Out for t...	Diary of a Wimpy Kid: Dog Days	6	223	No	Comedy	Fox 2000 Pictures
6	10366	45000000	Luc Dever...	14.566664	10667893	82	Released	Prepare to beco...	Universal Soldier: The Return	4.2	135	No	Action	TriStar Pictures
7	82696	30000000	After thirt...	12.967137	114281051	100	Released	Sometimes to ke...	Hope Springs	5.8	284	No	Drama	Columbia Pictures
8	32456	1000000	Two wom...	4.86805	2057193	84	Released	Thanks to his tw...	Two Girls and a Guy	5.4	24	No	Drama	Fox Searchlight Pictures
9	181940	0	When str...	0.006069	0	103	Released	[null]	Carousel of Revenge	0	0	No	Thriller	The Picture Factory
10	69640	0	"Lies in Pl...	0.47909	0	88	Released	[null]	Lies in Plain Sight	4.5	4	No	Drama	[null]

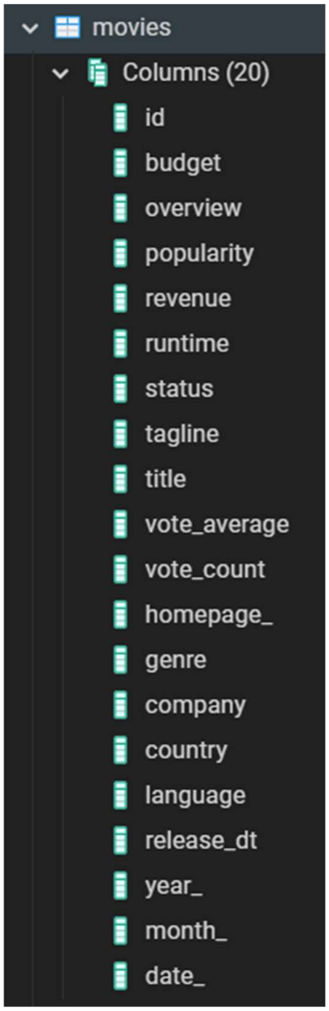


company text	country text	language text	release_dt date	year_ integer	month_ integer	date_ integer
[null]	[null]	EN	2014-04-15	2014	4	15
[null]	[null]	EN	2008-01-01	2008	1	1
Filmacres	[null]	EN	2015-10-23	2015	10	23
Zelko Films	Algeria	EN	2015-08-07	2015	8	7
Fox 2000 Pictures	United States of America	EN	2012-08-02	2012	8	2
TriStar Pictures	United States of America	EN	1999-08-05	1999	8	5
Columbia Pictures	United States of America	EN	2012-08-07	2012	8	7
Fox Searchlight Pictures	United States of America	EN	1997-09-07	1997	9	7
The Picture Factory	United States of America	EN	2007-07-20	2007	7	20
[null]	[null]	EN	2010-10-03	2010	10	3

Table columns before and after data cleaning:



movies
Columns (20)
id
budget
genres
homepage
keywords
original_language
original_title
overview
popularity
production_companies
production_countries
release_date
revenue
runtime
spoken_languages
status
tagline
title
vote_average
vote_count



movies
Columns (20)
id
budget
overview
popularity
revenue
runtime
status
tagline
title
vote_average
vote_count
homepage_
genre
company
country
language
release_dt
year_
month_
date_

Script for Task 1:

```
---- Project: DATA CLEANING IN SQL --------------------
-------------------------------
---- Description: In this project we clean the TMDB
movies dataset in SQL Server. ----
---- Task 1 is performed with this script: Cleaning the
"movies" table. --------------
---- Note: "id" column in the "movies" source data has
been moved to first place. ----
---- Note: The functions and commands are consistent
with PostGreSQL environment. ----
-------------------------------------------------------
-------------------------------

-- 1) Let's look at the "movies" table.
        SELECT *
        FROM movies

-- 2) Let's look at the table sorted by the "id" column
to find any discrepancies.
        SELECT *
        FROM movies
        ORDER BY id
        -- Observations: There are some missing "id"
entries.
```

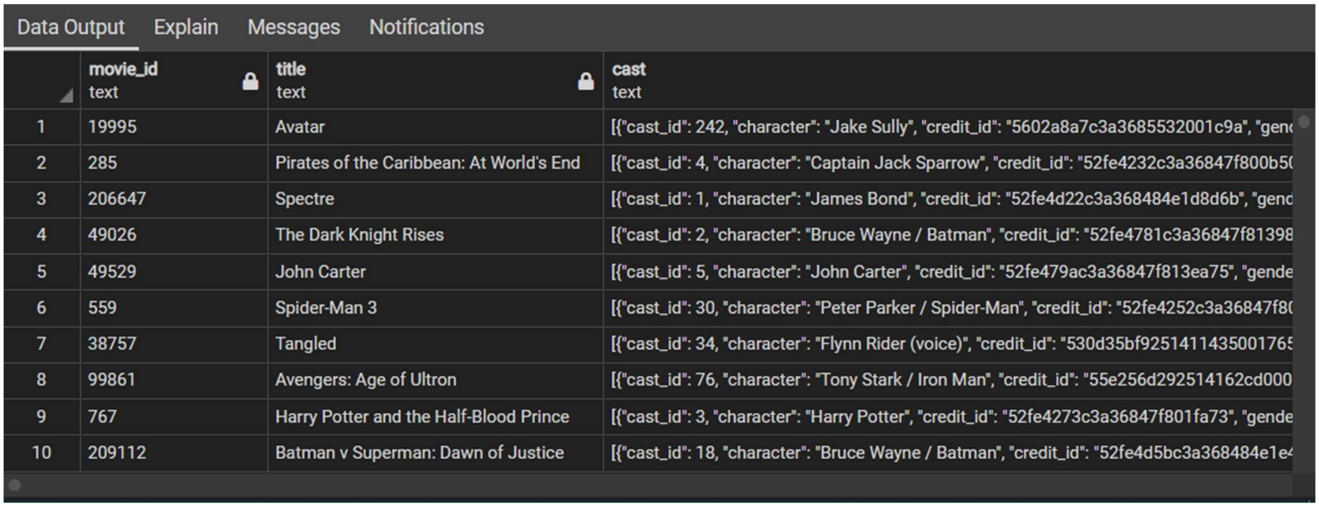
```
-- 3) Let's focus on the "movies" table and try to
clean it for further use.
        SELECT id, original_title, title FROM movies
        WHERE original_title <> title
        -- Let's drop the original_title column, as it
has special characters and we already have title
column.
        ALTER TABLE movies
        DROP COLUMN original_title

-- 4) Let's look at the "status" column. There are no
NULL values but almost all the entries are "Released".
        SELECT status, COUNT(status)
        FROM movies
        GROUP BY status

-- 5) Let's drop the keywords column as we do not
intend to use it further.
        ALTER TABLE movies
        DROP COLUMN keywords

-- 6) Let's change the homepage column to a Yes/No
column according to if the movie has a homepage or not.
```

```
        ALTER TABLE movies
        ADD homepage_ VARCHAR(3);

        UPDATE movies
        SET homepage_ = 'Yes'
        WHERE homepage IS NOT NULL

        UPDATE movies
        SET homepage_ = 'No'
        WHERE homepage IS NULL

        ALTER TABLE movies
        DROP COLUMN homepage

-- 7) Let's extract the most prominent values in JSON
populated columns
--    "genres", "production_companies", and
"production_countries",
--    and store those values in the new columns
--    "genre", "company", and "country".
--    We'll also convert the values in
"original_language" column
--    to upper-case and store them in new "language"
column.
        ALTER TABLE movies
        ADD COLUMN genre TEXT, ADD COLUMN company TEXT,
ADD COLUMN country TEXT, ADD COLUMN language TEXT;

        UPDATE movies
        SET genre = NULL
        WHERE genres = '[]'

        UPDATE movies
        SET genre = SUBSTRING(genres,
STRPOS(genres,'e":')+5, STRPOS(genres,'}')-
STRPOS(genres,'e":')-6)
        WHERE genres <> '[]'

        UPDATE movies
        SET company = NULL
        WHERE production_companies = '[]'

        UPDATE movies
        SET company = SUBSTRING(production_companies,
STRPOS(production_companies,'e":')+5,
STRPOS(production_companies,',')-
STRPOS(production_companies,'e":')-6)
        WHERE production_companies <> '[]'

        UPDATE movies
        SET country = NULL
        WHERE production_countries = '[]'

        UPDATE movies
        SET country = SUBSTRING(production_countries,
STRPOS(production_countries,'e":')+5,
STRPOS(production_countries,'}')-
STRPOS(production_countries,'e":')-6)
        WHERE production_countries <> '[]'

        UPDATE movies
        SET language = UPPER(original_language)

-- 8) Let's drop the old columns
--    "genres", "production_companies",
"production_countries", "spoken_languages", and
"original_language".
        ALTER TABLE movies
        DROP COLUMN genres, DROP COLUMN
original_language, DROP COLUMN production_companies,
        DROP COLUMN production_countries, DROP COLUMN
spoken_languages

-- 9) Let's change the "release_date" type from
DATETIME to DATE and store the values in new column
"release_dt".
        ALTER TABLE movies
        ADD release_dt DATE

        UPDATE movies
        SET release_dt = DATE(release_date)

-- 10) Let's extract year, month, and date information
from "release_dt" column,
--     and store the values in newly created "year_",
"month_", and "date_" columns.
        ALTER TABLE movies
        ADD COLUMN year_ int, ADD COLUMN month_ int,
ADD COLUMN date_ int;

        UPDATE movies
        SET year_ = EXTRACT(year FROM release_dt)

        UPDATE movies
        SET month_ = EXTRACT(month FROM release_dt)

        UPDATE movies
        SET date_ = EXTRACT(day FROM release_dt)

        ALTER TABLE movies
        DROP COLUMN release_date

-- 11) Let's delete entries with no "id" (zero such
entries are present.)
        DELETE FROM movies
        WHERE id IS NULL

-- 12) Let's take a final look at the "movies" table.
        SELECT *
        FROM movies
```

Data Cleaning in SQL

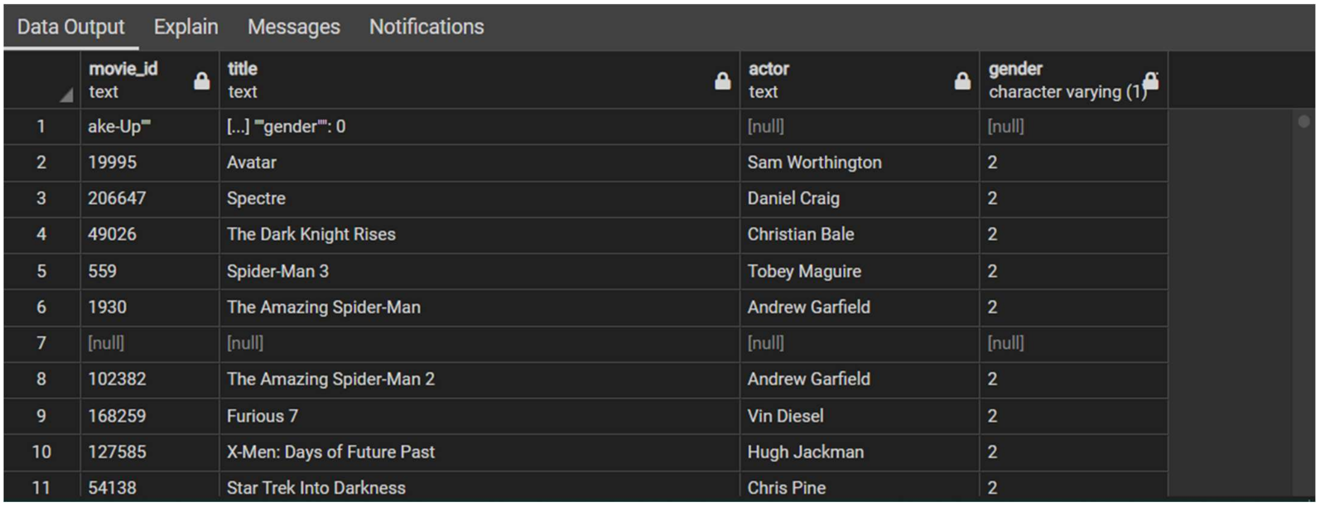
Task 2 : Cleaning the “credits” table (in PostGreSQL)

1) Output after Subtask 1



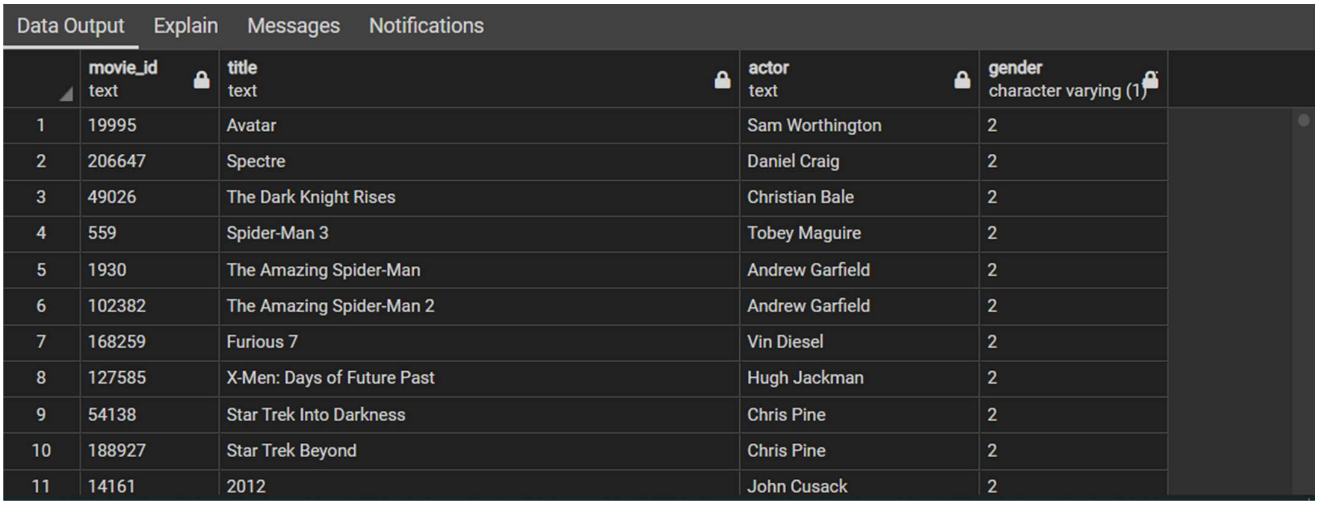
	movie_id text	title text	cast text
1	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "credit_id": "5602a8a7c3a3685532001c9a", "gen
2	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Sparrow", "credit_id": "52fe4232c3a36847f800b5
3	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "credit_id": "52fe4d22c3a368484e1d8d6b", "gend
4	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Batman", "credit_id": "52fe4781c3a36847f81398
5	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "credit_id": "52fe479ac3a36847f813ea75", "gende
6	559	Spider-Man 3	[{"cast_id": 30, "character": "Peter Parker / Spider-Man", "credit_id": "52fe4252c3a36847f8
7	38757	Tangled	[{"cast_id": 34, "character": "Flynn Rider (voice)", "credit_id": "530d35bf925141143500176
8	99861	Avengers: Age of Ultron	[{"cast_id": 76, "character": "Tony Stark / Iron Man", "credit_id": "55e256d292514162cd000
9	767	Harry Potter and the Half-Blood Prince	[{"cast_id": 3, "character": "Harry Potter", "credit_id": "52fe4273c3a36847f801fa73", "gende
10	209112	Batman v Superman: Dawn of Justice	[{"cast_id": 18, "character": "Bruce Wayne / Batman", "credit_id": "52fe4d5bc3a368484e1e

2) Output after Subtask 3



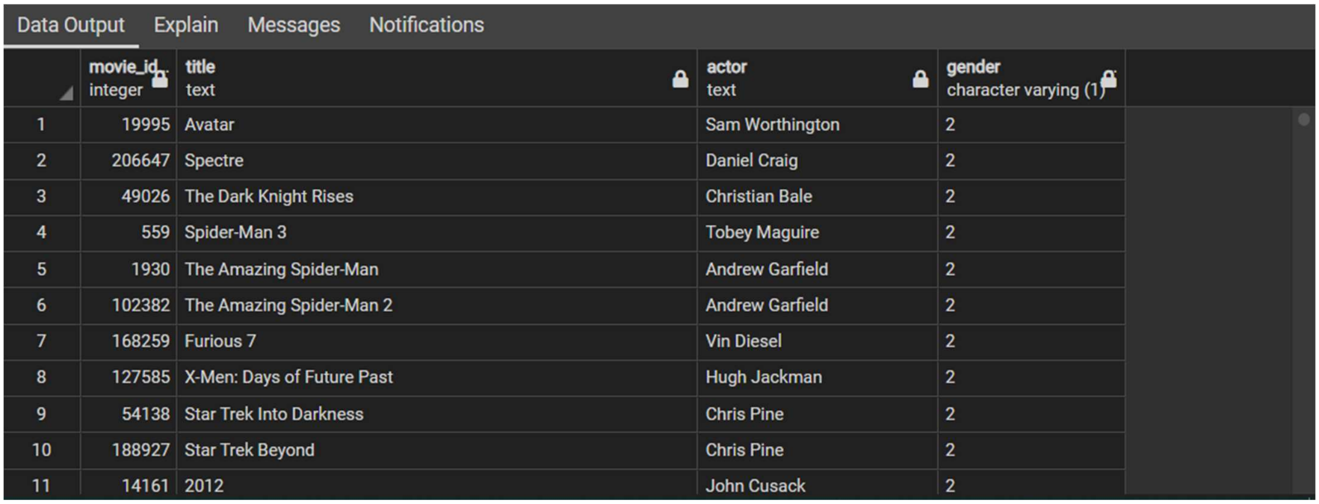
	movie_id text	title text	actor text	gender character varying (1)
1	ake-Up""	[...] ""gender"": 0	[null]	[null]
2	19995	Avatar	Sam Worthington	2
3	206647	Spectre	Daniel Craig	2
4	49026	The Dark Knight Rises	Christian Bale	2
5	559	Spider-Man 3	Tobey Maguire	2
6	1930	The Amazing Spider-Man	Andrew Garfield	2
7	[null]	[null]	[null]	[null]
8	102382	The Amazing Spider-Man 2	Andrew Garfield	2
9	168259	Furious 7	Vin Diesel	2
10	127585	X-Men: Days of Future Past	Hugh Jackman	2
11	54138	Star Trek Into Darkness	Chris Pine	2

3) Output after Subtask 4



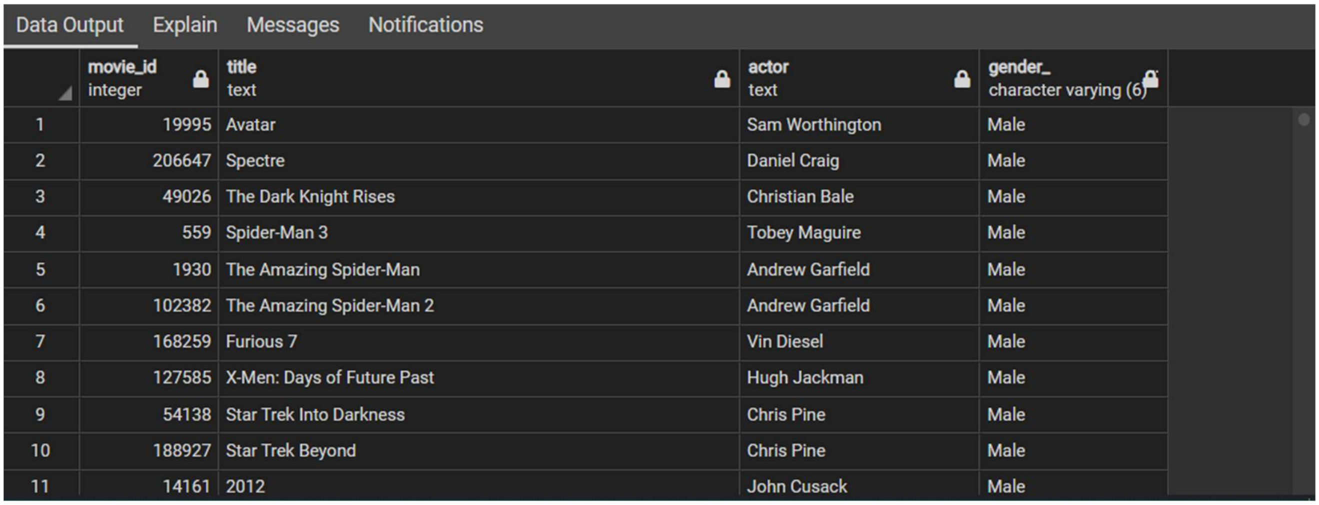
	movie_id text	title text	actor text	gender character varying (1)
1	19995	Avatar	Sam Worthington	2
2	206647	Spectre	Daniel Craig	2
3	49026	The Dark Knight Rises	Christian Bale	2
4	559	Spider-Man 3	Tobey Maguire	2
5	1930	The Amazing Spider-Man	Andrew Garfield	2
6	102382	The Amazing Spider-Man 2	Andrew Garfield	2
7	168259	Furious 7	Vin Diesel	2
8	127585	X-Men: Days of Future Past	Hugh Jackman	2
9	54138	Star Trek Into Darkness	Chris Pine	2
10	188927	Star Trek Beyond	Chris Pine	2
11	14161	2012	John Cusack	2

4) Output after Subtask 5



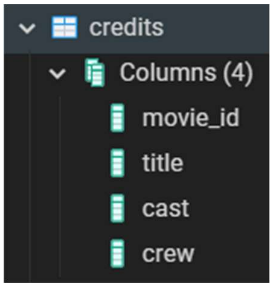
	movie_id integer	title text	actor text	gender character varying (1)
1	19995	Avatar	Sam Worthington	2
2	206647	Spectre	Daniel Craig	2
3	49026	The Dark Knight Rises	Christian Bale	2
4	559	Spider-Man 3	Tobey Maguire	2
5	1930	The Amazing Spider-Man	Andrew Garfield	2
6	102382	The Amazing Spider-Man 2	Andrew Garfield	2
7	168259	Furious 7	Vin Diesel	2
8	127585	X-Men: Days of Future Past	Hugh Jackman	2
9	54138	Star Trek Into Darkness	Chris Pine	2
10	188927	Star Trek Beyond	Chris Pine	2
11	14161	2012	John Cusack	2

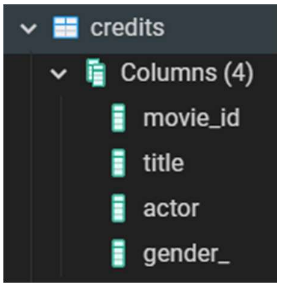
5) Output after Subtask 7



	movie_id integer	title text	actor text	gender_ character varying (6)
1	19995	Avatar	Sam Worthington	Male
2	206647	Spectre	Daniel Craig	Male
3	49026	The Dark Knight Rises	Christian Bale	Male
4	559	Spider-Man 3	Tobey Maguire	Male
5	1930	The Amazing Spider-Man	Andrew Garfield	Male
6	102382	The Amazing Spider-Man 2	Andrew Garfield	Male
7	168259	Furious 7	Vin Diesel	Male
8	127585	X-Men: Days of Future Past	Hugh Jackman	Male
9	54138	Star Trek Into Darkness	Chris Pine	Male
10	188927	Star Trek Beyond	Chris Pine	Male
11	14161	2012	John Cusack	Male

Table columns before and after data cleaning:





Script for Task 2:

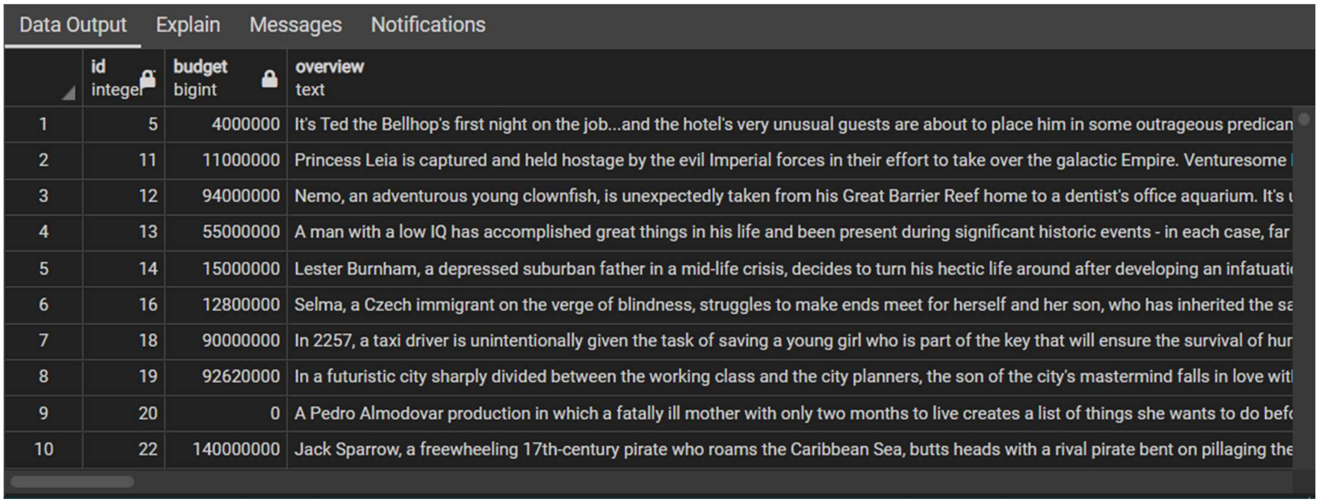
```
---- Project: DATA CLEANING IN SQL --------------------
-------------------------------
---- Description: In this project we clean the TMDB
movies dataset in SQL Server. ----
---- Task 2 is performed with this script: Cleaning the
"credits" table. -------------
---- Note: The functions and commands are consistent
with PostGreSQL environment. ----
-------------------------------------------------------
-------------------------------

-- 1) Let's look at the "credits" table.
        SELECT *
        FROM credits

-- 2) Let's extract the main actor & their gender, and
store them in columns "actor" and "gender".
        ALTER TABLE credits
        ADD COLUMN actor TEXT, ADD COLUMN gender
VARCHAR(1);

        UPDATE credits
        SET actor = SUBSTRING("cast",
STRPOS("cast",'"name"')+9, STRPOS("cast",'"order"')-
STRPOS("cast",'"name"')-12)
        WHERE "cast" LIKE '%"character"%'

        UPDATE credits
        SET actor = NULL
        WHERE "cast" NOT LIKE '%"character"%'

        UPDATE credits
        SET gender = SUBSTRING("cast",
STRPOS("cast",'"gender"')+10, 1)
        WHERE "cast" LIKE '%"character"%'

        UPDATE credits
        SET gender = NULL
        WHERE actor IS NULL
```

```
-- 3) Let's drop the old columns "cast" and "crew".
        ALTER TABLE credits
        DROP COLUMN "cast", DROP COLUMN crew;

-- 4) Let's delete the rows with "actor" value as NULL.
        DELETE FROM credits
        WHERE actor IS NULL

-- 5) Let's change the data type of "movie_id" column
from TEXT to INT.
        ALTER TABLE credits
        ALTER COLUMN movie_id TYPE INT USING
movie_id::integer

-- 6) Let's create new column "gender_" and store the
values as
--    Female, Male, or NULL according to the 1, 2, or
NULL values in "gender" column.
--    Let's also drop the "gender" column afterwards.
        ALTER TABLE credits
        ADD COLUMN gender_ VARCHAR(6)

        UPDATE credits
        SET gender_ = 'Female' WHERE gender = '1'

        UPDATE credits
        SET gender_ = 'Male' WHERE gender = '2'

        UPDATE credits
        SET gender_ = NULL WHERE gender NOT IN
('1','2')

        ALTER TABLE credits
        DROP COLUMN gender

-- 7) Let's take a final look at the "credits" table.
        SELECT *
        FROM credits
```

Data Cleaning in SQL

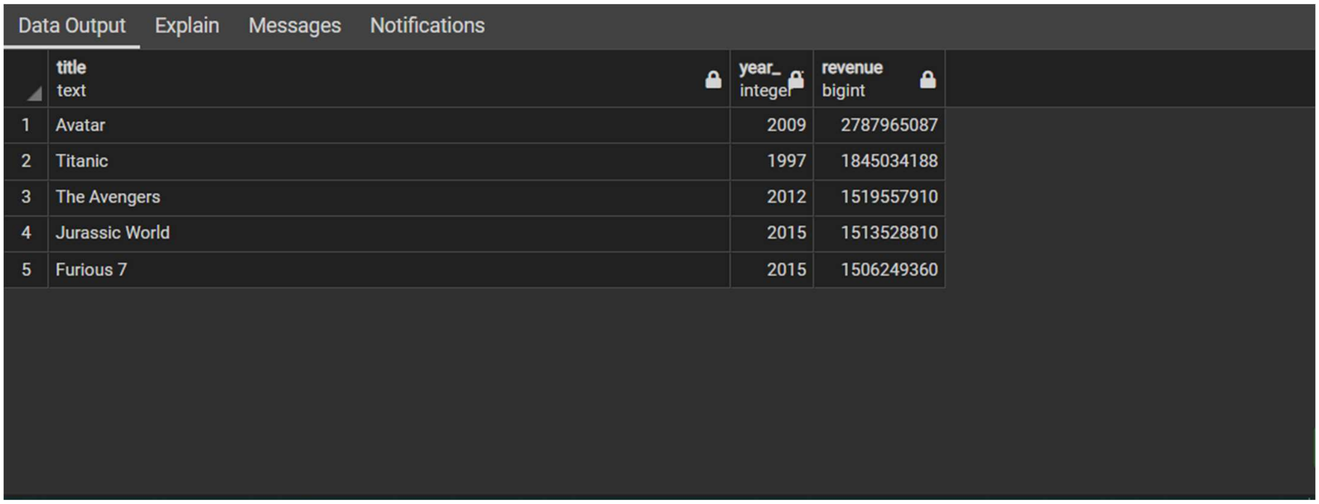
Task 3 : Exploring the dataset (in PostGreSQL)

1) Output after Subtask 1



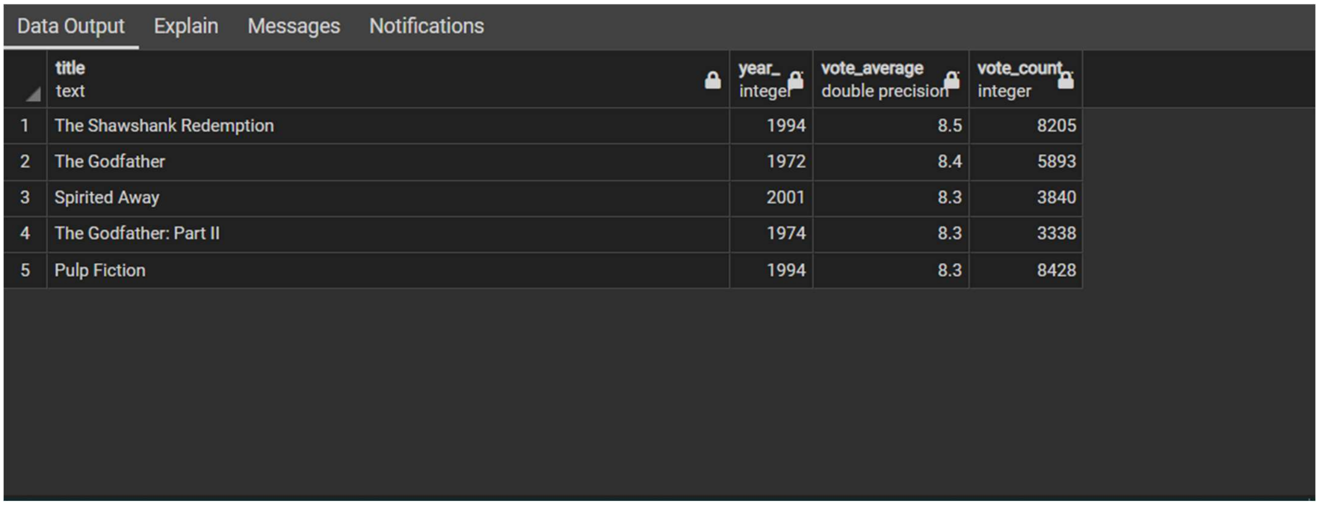
	id integer	budget bigint	overview text
1	5	4000000	It's Ted the Bellhop's first night on the job...and the hotel's very unusual guests are about to place him in some outrageous predican
2	11	11000000	Princess Leia is captured and held hostage by the evil Imperial forces in their effort to take over the galactic Empire. Venturesome
3	12	94000000	Nemo, an adventurous young clownfish, is unexpectedly taken from his Great Barrier Reef home to a dentist's office aquarium. It's
4	13	55000000	A man with a low IQ has accomplished great things in his life and been present during significant historic events - in each case, far
5	14	15000000	Lester Burnham, a depressed suburban father in a mid-life crisis, decides to turn his hectic life around after developing an infatuati
6	16	12800000	Selma, a Czech immigrant on the verge of blindness, struggles to make ends meet for herself and her son, who has inherited the sa
7	18	90000000	In 2257, a taxi driver is unintentionally given the task of saving a young girl who is part of the key that will ensure the survival of hur
8	19	92620000	In a futuristic city sharply divided between the working class and the city planners, the son of the city's mastermind falls in love wit
9	20	0	A Pedro Almodovar production in which a fatally ill mother with only two months to live creates a list of things she wants to do bef
10	22	140000000	Jack Sparrow, a freewheeling 17th-century pirate who roams the Caribbean Sea, butts heads with a rival pirate bent on pillaging the

2) Output after Subtask 2



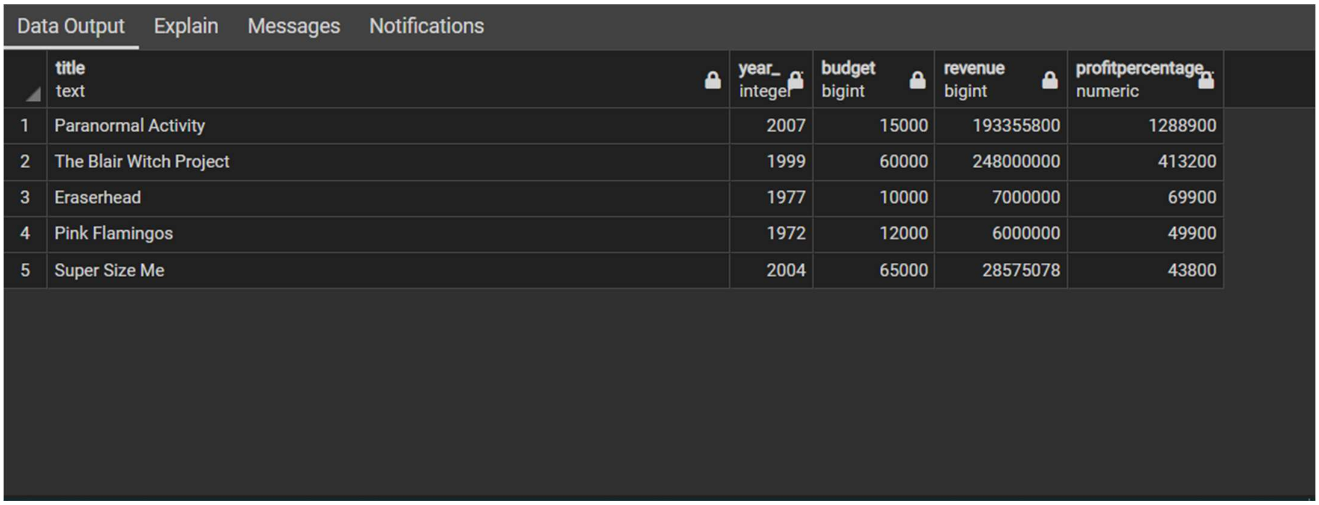
	title text	year_ integer	revenue bigint
1	Avatar	2009	2787965087
2	Titanic	1997	1845034188
3	The Avengers	2012	1519557910
4	Jurassic World	2015	1513528810
5	Furious 7	2015	1506249360

3) Output after Subtask 3



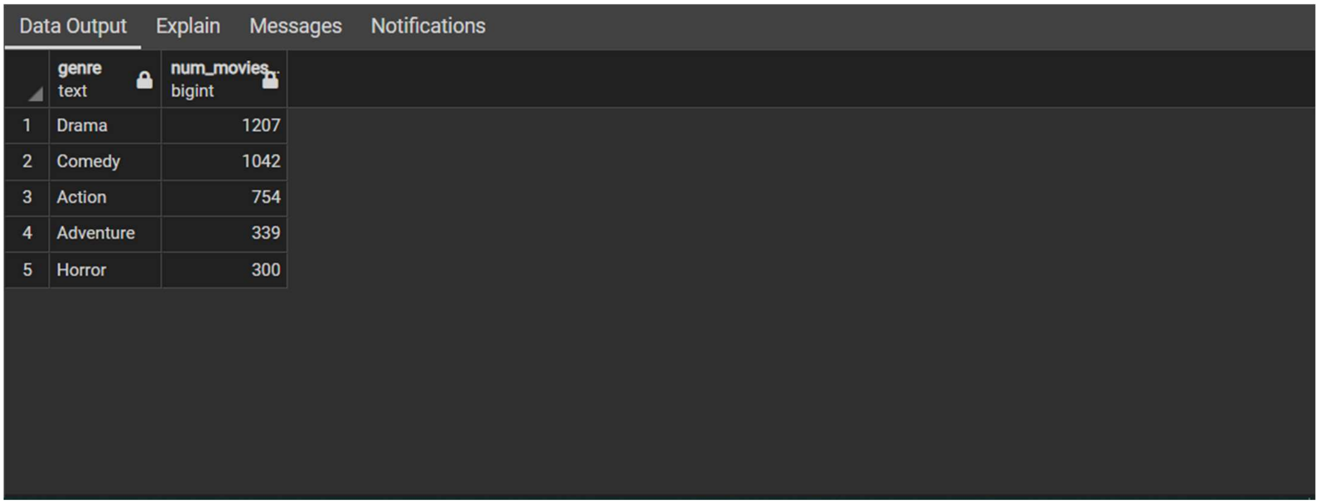
	title text	year_ integer	vote_average double precision	vote_count integer
1	The Shawshank Redemption	1994	8.5	8205
2	The Godfather	1972	8.4	5893
3	Spirited Away	2001	8.3	3840
4	The Godfather: Part II	1974	8.3	3338
5	Pulp Fiction	1994	8.3	8428

4) Output after Subtask 4



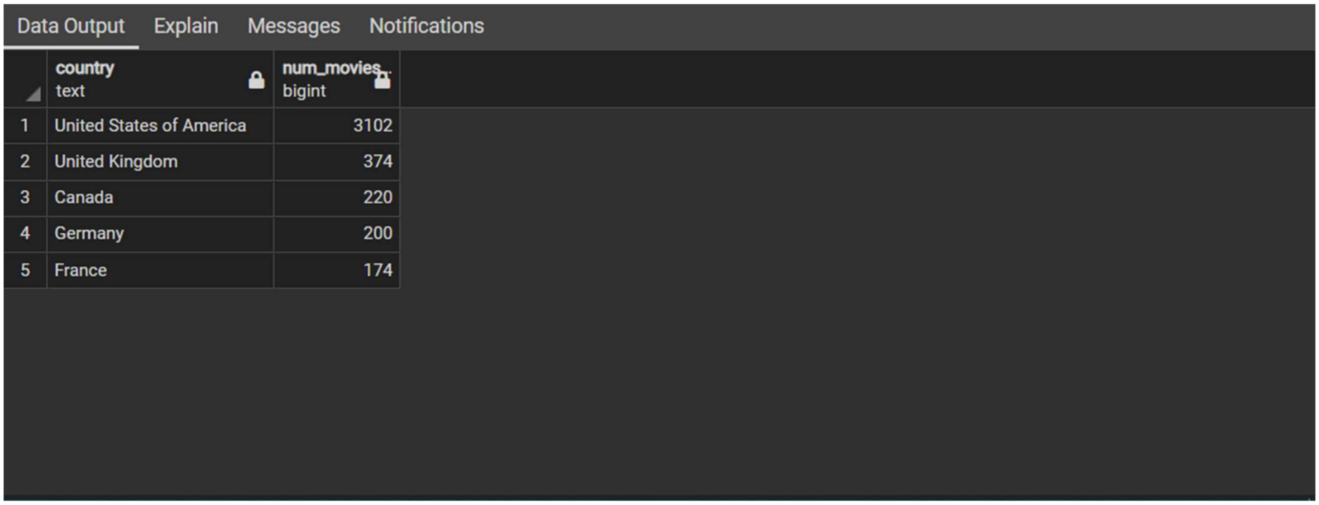
	title text	year_ integer	budget bigint	revenue bigint	profitpercentage numeric
1	Paranormal Activity	2007	15000	193355800	1288900
2	The Blair Witch Project	1999	60000	248000000	413200
3	Eraserhead	1977	10000	7000000	69900
4	Pink Flamingos	1972	12000	6000000	49900
5	Super Size Me	2004	65000	28575078	43800

5) Output after Subtask 5



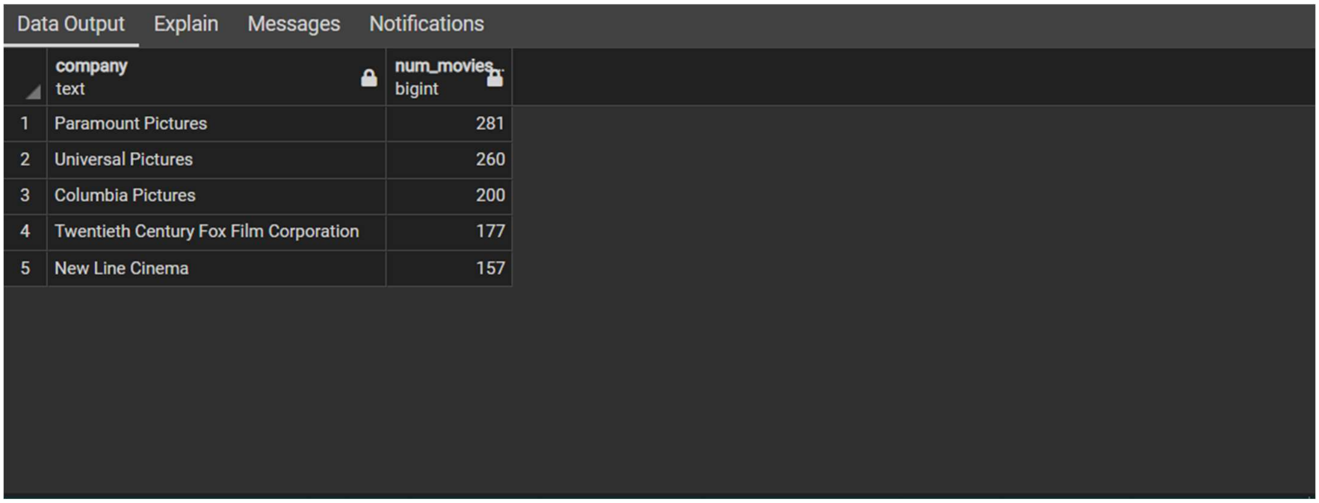
	genre text	num_movies bigint
1	Drama	1207
2	Comedy	1042
3	Action	754
4	Adventure	339
5	Horror	300

6) Output after Subtask 6



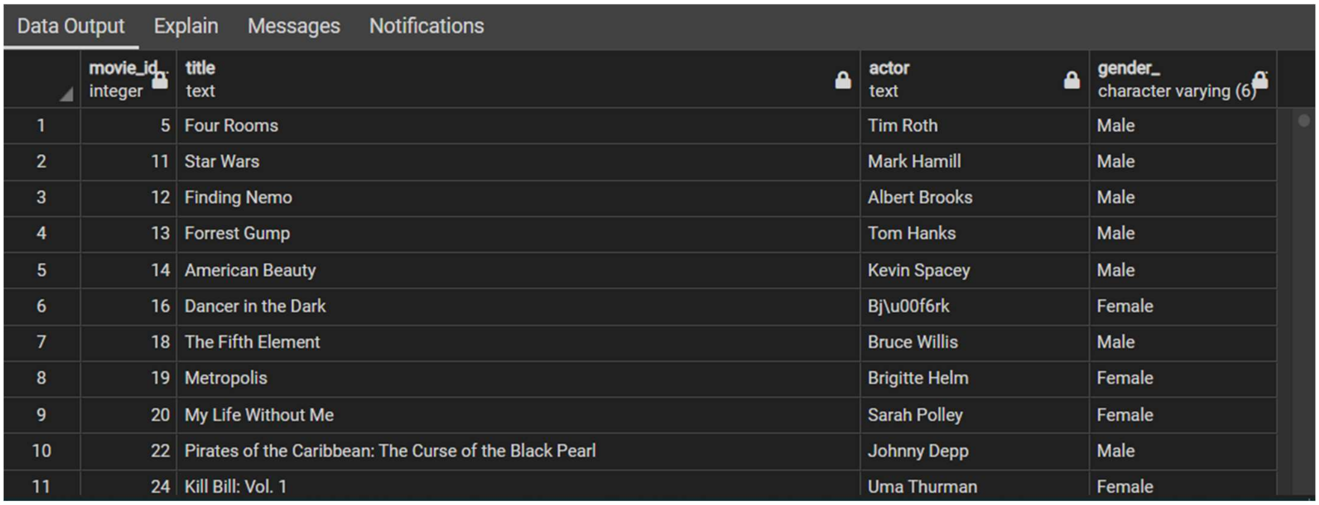
	country text	num_movies bigint
1	United States of America	3102
2	United Kingdom	374
3	Canada	220
4	Germany	200
5	France	174

7) Output after Subtask 7



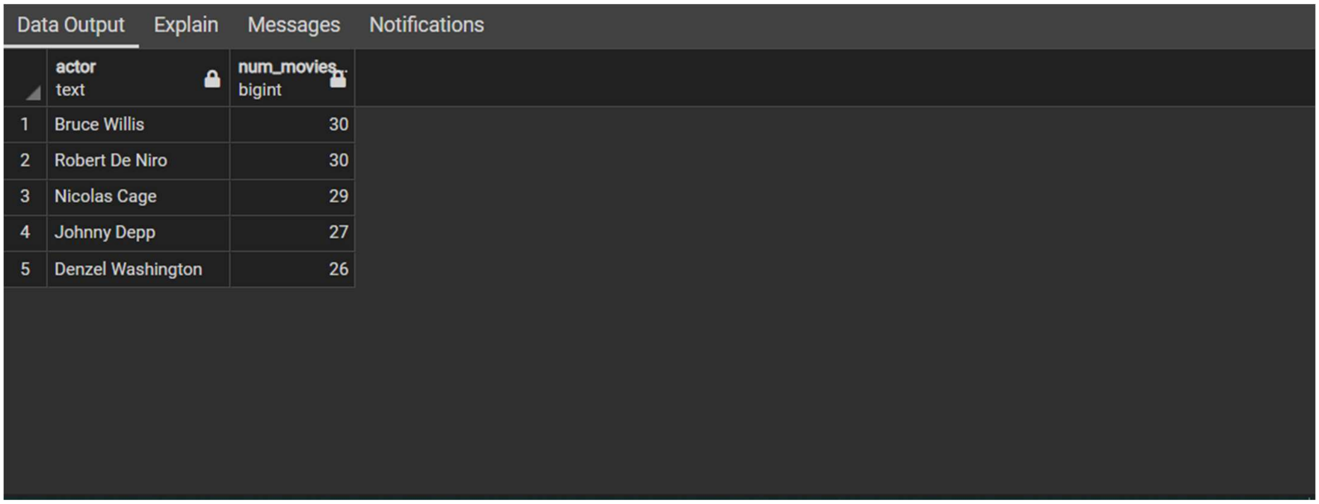
	company text	num_movies bigint
1	Paramount Pictures	281
2	Universal Pictures	260
3	Columbia Pictures	200
4	Twentieth Century Fox Film Corporation	177
5	New Line Cinema	157

8) Output after Subtask 8



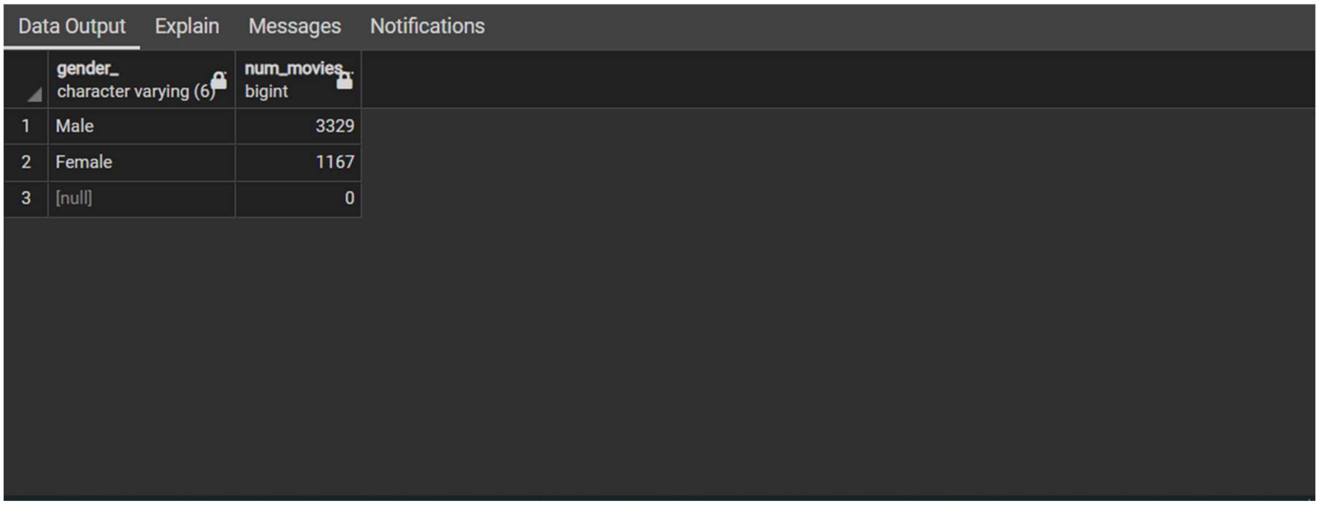
	movie_id integer	title text	actor text	gender_ character varying (6)
1	5	Four Rooms	Tim Roth	Male
2	11	Star Wars	Mark Hamill	Male
3	12	Finding Nemo	Albert Brooks	Male
4	13	Forrest Gump	Tom Hanks	Male
5	14	American Beauty	Kevin Spacey	Male
6	16	Dancer in the Dark	Bj\u00f6rk	Female
7	18	The Fifth Element	Bruce Willis	Male
8	19	Metropolis	Brigitte Helm	Female
9	20	My Life Without Me	Sarah Polley	Female
10	22	Pirates of the Caribbean: The Curse of the Black Pearl	Johnny Depp	Male
11	24	Kill Bill: Vol. 1	Uma Thurman	Female

9) Output after Subtask 9



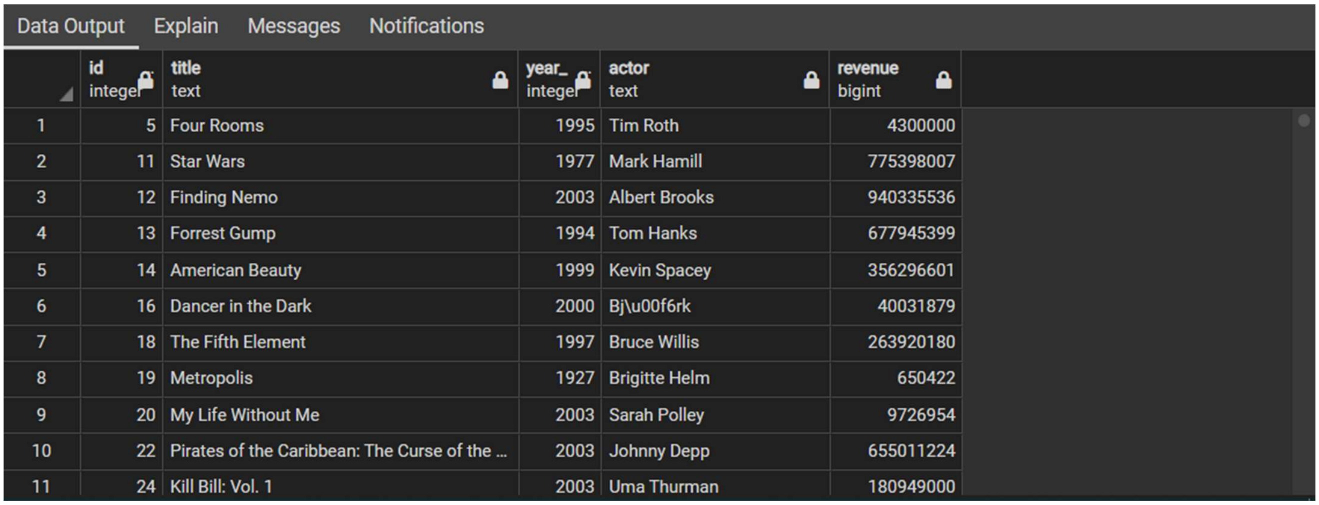
	actor text	num_movies bigint
1	Bruce Willis	30
2	Robert De Niro	30
3	Nicolas Cage	29
4	Johnny Depp	27
5	Denzel Washington	26

10) Output after Subtask 10



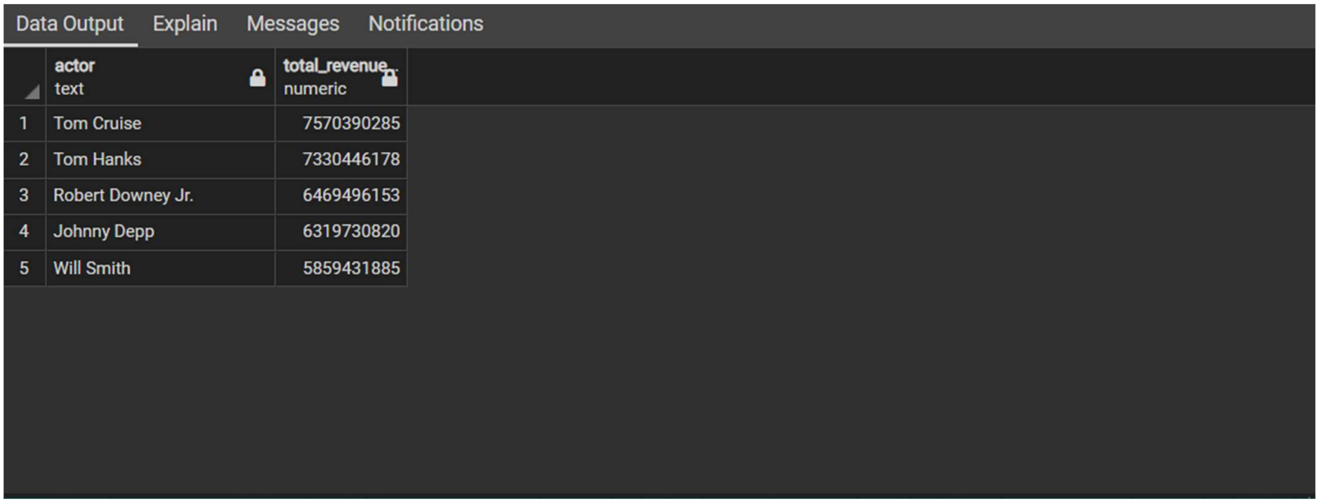
	gender_ character varying (6)	num_movies bigint
1	Male	3329
2	Female	1167
3	[null]	0

11) Output after Subtask 11



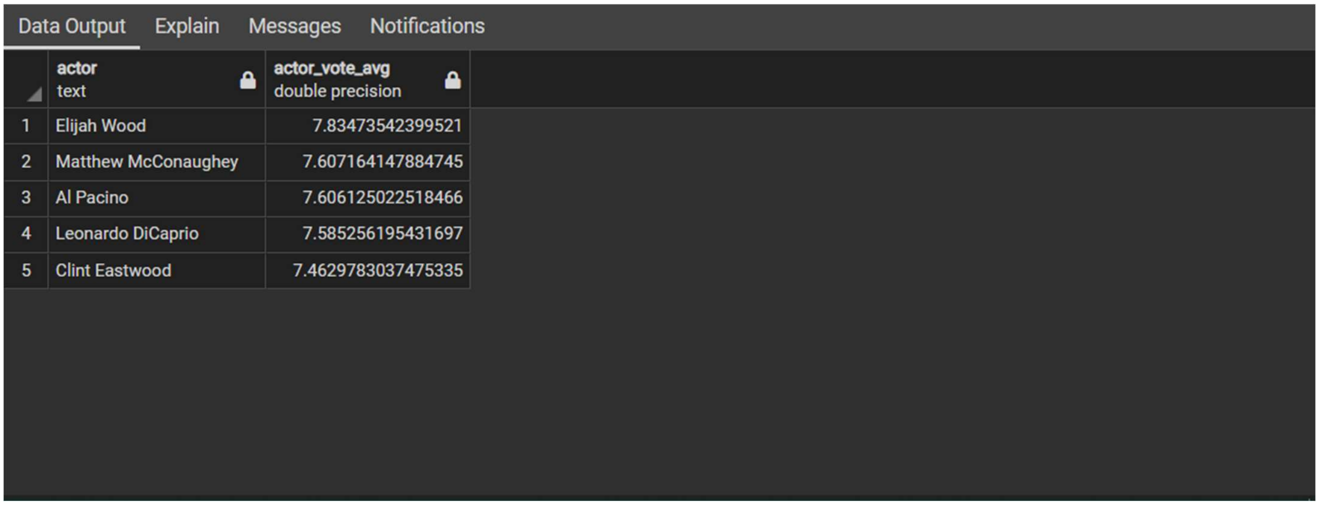
	id integer	title text	year_ integer	actor text	revenue bigint
1	5	Four Rooms	1995	Tim Roth	4300000
2	11	Star Wars	1977	Mark Hamill	775398007
3	12	Finding Nemo	2003	Albert Brooks	940335536
4	13	Forrest Gump	1994	Tom Hanks	677945399
5	14	American Beauty	1999	Kevin Spacey	356296601
6	16	Dancer in the Dark	2000	Bj\u00f6rk	40031879
7	18	The Fifth Element	1997	Bruce Willis	263920180
8	19	Metropolis	1927	Brigitte Helm	650422
9	20	My Life Without Me	2003	Sarah Polley	9726954
10	22	Pirates of the Caribbean: The Curse of the ...	2003	Johnny Depp	655011224
11	24	Kill Bill: Vol. 1	2003	Uma Thurman	180949000

12) Output after Subtask 12



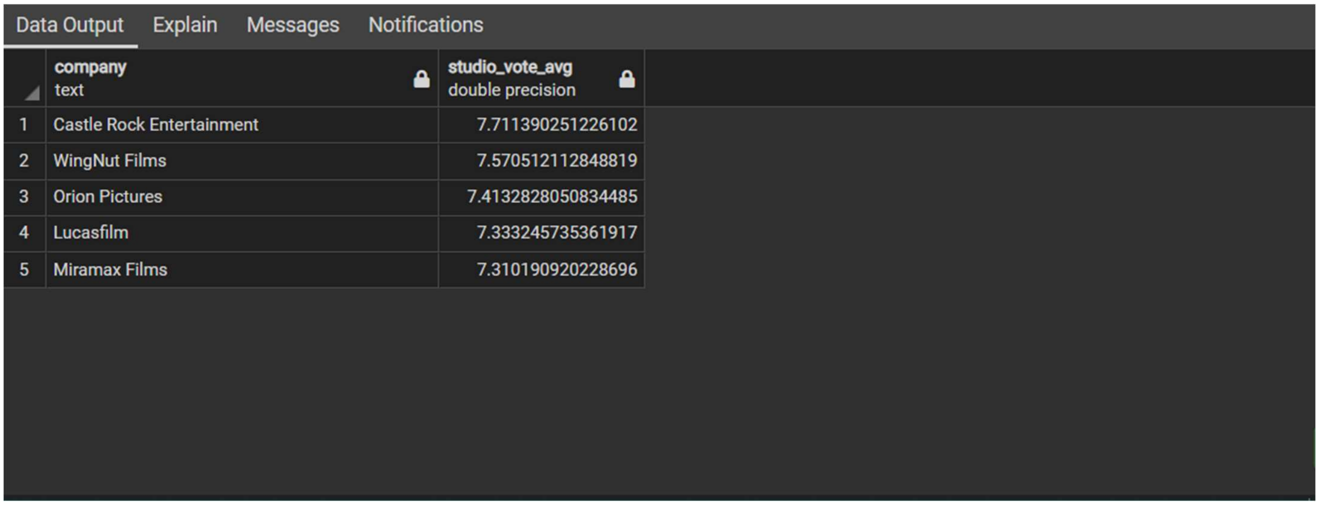
	actor text	total_revenue numeric
1	Tom Cruise	7570390285
2	Tom Hanks	7330446178
3	Robert Downey Jr.	6469496153
4	Johnny Depp	6319730820
5	Will Smith	5859431885

13) Output after Subtask 13



	actor text	actor_vote_avg double precision
1	Elijah Wood	7.83473542399521
2	Matthew McConaughey	7.607164147884745
3	Al Pacino	7.606125022518466
4	Leonardo DiCaprio	7.585256195431697
5	Clint Eastwood	7.4629783037475335

14) Output after Subtask 14



	company text	studio_vote_avg double precision
1	Castle Rock Entertainment	7.711390251226102
2	WingNut Films	7.570512112848819
3	Orion Pictures	7.4132828050834485
4	Lucasfilm	7.333245735361917
5	Miramax Films	7.310190920228696

Script for Task 3:

```
---- Project: DATA CLEANING IN SQL --------------------
------------------------------
---- Description: In this project we clean the TMDB
movies dataset in SQL Server. ----
---- Task 3 is performed with this script: Exploring
the cleaned dataset. ------------
---- Note: The functions and commands are consistent
with PostGreSQL environment. ----
-------------------------------------------------------
------------------------------

-- 1) Let's look at the "movies" table sorted by "id"
column.
        SELECT * FROM movies
        ORDER BY id

-- 2) Top-5 Highest Grossing Movies:
        SELECT title, year_, revenue FROM movies
        ORDER BY revenue DESC
        LIMIT 5

-- 3) Top-5 Highest Rated Movies (with atleast 50
votes):
        SELECT title, year_, vote_average, vote_count
FROM movies
        WHERE vote_count >= 50
        ORDER BY vote_average DESC
        LIMIT 5

-- 4) Top-5 Movies with Highest Profit Percentages
(with a minimum budget of 1000 USD):
        SELECT title, year_, budget, revenue,
ROUND((revenue/budget*100-100),0) AS ProfitPercentage
FROM movies
        WHERE budget >= 1000
        ORDER BY ProfitPercentage DESC
        LIMIT 5

-- 5) Top-5 Most Popular Genres (out of 20 genres):
        SELECT genre, COUNT(genre) AS num_movies
        FROM movies
        GROUP BY genre
        ORDER BY num_movies DESC
        LIMIT 5

-- 6) Top-5 Countries with most movies (out of 70
countries):
        SELECT country, COUNT(country) AS num_movies
        FROM movies
        GROUP BY country
        ORDER BY num_movies DESC
        LIMIT 5

-- 7) Top-5 Companies with most movies (out of 1310
companies):
        SELECT company, COUNT(company) AS num_movies
        FROM movies
        GROUP BY company
        ORDER BY num_movies DESC
        LIMIT 5

-------------------------------------------------------
-------------------------------

-- 8) Let's look at the "credits" table sorted by
"movie_id" column.
        SELECT * FROM credits
        ORDER BY movie_id

-- 9) Top-5 Actors with Most Movies:
        SELECT actor, COUNT(actor) AS num_movies
        FROM credits
        GROUP BY actor
        ORDER BY num_movies DESC
        LIMIT 5

-- 10) Actor Gender distribution:
        SELECT gender_, COUNT(gender_) AS num_movies
        FROM credits
        GROUP BY gender_
        ORDER BY num_movies DESC

-------------------------------------------------------
-------------------------------

-- 11) Let's join both the tables to bring the "actor"
column in from "credits" table.
        SELECT movies.id, movies.title, year_, actor,
movies.revenue FROM movies
        INNER JOIN credits
        ON movies.id = credits.movie_id
        ORDER BY movies.id

-- 12) Top-5 Actors with Highest Total Revenue:
--     (we use the output in subtask 11 as Common Table
Expression (CTE)).
        WITH ActRevenue (id, title, year_, actor,
revenue) AS
        (
        SELECT movies.id, movies.title, year_, actor,
movies.revenue FROM movies
        INNER JOIN credits
        ON movies.id = credits.movie_id
        ORDER BY movies.id
        )
        SELECT actor, SUM(revenue) AS total_revenue
FROM ActRevenue
        GROUP BY actor
        ORDER BY total_revenue DESC
        LIMIT 5

-- 13) Top-5 Actors with Highest Vote Average (with a
minimum of 10 movies):
        WITH ActRating (id, title, actor, vote_average,
vote_count, vote_aggregate) AS
        (
        SELECT movies.id, movies.title, actor,
movies.vote_average, movies.vote_count,
        (movies.vote_average*movies.vote_count) AS
vote_aggregate
        FROM movies
        INNER JOIN credits
        ON movies.id = credits.movie_id
        ORDER BY movies.id
        )
        SELECT actor,
SUM(vote_aggregate)/SUM(vote_count) AS actor_vote_avg
FROM ActRating
        GROUP BY actor
        HAVING SUM(vote_count) > 0 AND
COUNT(vote_count) >= 10
        ORDER BY actor_vote_avg DESC
        LIMIT 5

-- 14) Top-5 Studios with Highest Vote Average (with a
minimum of 10 movies):
        SELECT company,
SUM(vote_average*vote_count)/SUM(vote_count) AS
studio_vote_avg FROM movies
        GROUP BY company
        HAVING SUM(vote_count) > 0 AND
COUNT(vote_count) >= 10
        ORDER BY studio_vote_avg DESC
        LIMIT 5
```

Project Owner: More Shekhar Sanjay

Project Repository: https://github.com/MoreShekharSanjay/project-data-cleaning-in-sql

Email me at: moreshekharsanjay@gmail.com

My LinkedIn: https://www.linkedin.com/in/moreshekharsanjay/

Dataset source: https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata

Software and languages used:
SQL, Microsoft SQL Server Management Studio 18, PostGreSQL Admin V4, Microsoft Excel

---