Database Lab – HW3

سوال اول

به وسیله این دستور میتوان فیلم های موجود را بر اساس ژانر، دسته بندی کرد به طوری که مجموع box_office و budget کلی این فیلم ها را خروجی گرفت. همانطور که مشاهده میکنید برای ژانر دوم (romance) دو فیلم وجود دارد:

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

```
1  -- Question 1
2  select genre_id, sum(box_office) box_office, sum(budget) budget, count(budget) count
3  from movies
4  group by genre_id;
5
```

Data Output | Explain | Messages | Notifications

	genre_id integer	box_office money	budget money	count bigint
1	3	$547,400,000.00	$35,000,000.00	1
2	0	$755,900,000.00	$30,000,000.00	2
3	2	$69,400,000.00	$50,000,000.00	1

در دستور بعدی میانگین امتیاز imdb و rotten_tomatoes هر ژانر آورده شده که مانند دستور قبل بر اساس شماره ژانر گروه بندی شده اند:

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

```
1  -- Question 1
2  select genre_id, sum(box_office) box_office, sum(budget) budget, count(budget) count
3  from movies
4  group by genre_id;
5
6  select genre_id, avg(imdb) imdb, avg(rotten_tomatoes) rotten_tomatoes, count(budget) count
7  from movies
8  group by genre_id;
9
```

Data Output | Explain | Messages | Notifications

	genre_id integer	imdb numeric	rotten_tomatoes numeric	count bigint
1	3	7.3000000000000000	72.0000000000000000	1
2	0	7.8500000000000000	87.5000000000000000	2
3	2	7.3000000000000000	51.0000000000000000	1

در آخرین دستور میانگین مدت زمان هر ژانر خروجی داده شده. به طوری که ژانر شماره 2 که action است بیشتر مدت زمان فیلم را دارد:

```
10  select genre_id, avg(duration) avg_duration, count(budget) count
11  from movies
12  group by genre_id;
```

Data Output | Explain | Messages | Notifications

	genre_id integer	avg_duration numeric	count bigint
1	3	134.0000000000000000	1
2	0	127.0000000000000000	2
3	2	144.0000000000000000	1

سوال دوم

در این سوال از دستور intersect برای اشتراک گیری بین دو select استفاده شد و خروجی به ترتیب نزولی دو ستون عددی نمایش داده شده است:

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

```
1  (
2      (
3          select *
4          from movies
5      )
6      INTERSECT
7      (
8          select *
9          from movies
10         where imdb < 8
11     )
12 ) order by imdb DESC, rotten_tomatoes DESC
```

Data Output | Explain | Messages | Notifications

	id integer	title character varying (50)	release_date date	imdb numeric (3,1)	rotten_tomatoes smallint	box_office money	director character varying (30)	duration smallint	budget money	genre_id integer
1	0	The Fault in Our Stars	2014-06-06	7.7	81	$307,000,000.00	Josh Boone	126	$8,500,000.00	0
2	2	American Sniper	2014-02-16	7.3	72	$547,400,000.00	Clint Eastwood	134	$35,000,000.00	3
3	1	13 Hours	2016-03-04	7.3	51	$69,400,000.00	Michael Bay	144	$50,000,000.00	2

سوال سوم

ابتدا از جدول genres ژانر های مربوطه را با جدول movies یکی کرده و در دستور دوم 3 ستون خاص را میخوانیم:

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

```
with cte_movie as (
    select title, release_date, imdb, rotten_tomatoes, box_office, director, duration, budget, genre
    from movies
    natural join genres
)
select title, director, genre
from cte_movie
```

Data Output | Explain | Messages | Notifications

	title character varying (50)	director character varying (30)	genre character varying (20)
1	The Fault in Our Stars	Josh Boone	romance
2	13 Hours	Michael Bay	Horror
3	American Sniper	Clint Eastwood	Action
4	La La Land	Damien Chazelle	Drama

سوال چهارم

در این دستور با استفاده از inner join مشخصات تمام فیلم هایی که ژانر دارند و ژانر آنها در جدول genres وجود دارد چاپ میشود:

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

```
select title, release_date, imdb, genre, duration
from movies
inner join genres
    on genres.id = movies.genre_id
```

Data Output | Explain | Messages | Notifications

	title character varying (50)	release_date date	imdb numeric (3,1)	genre character varying (20)	duration smallint
1	The Fault in Our Stars	2014-06-06	7.7	romance	126
2	13 Hours	2016-03-04	7.3	Action	144
3	American Sniper	2014-02-16	7.3	Drama	134
4	La La Land	2016-07-01	8.0	romance	128

در این دستور با استفاده از right outer join تمام ژانر ها – جدا از اینکه آیا فیلمی داریم که این ژانر را داشته باشند یا خیر – چاپ شده اند. برای دو ژانر comedy و horror چون فیلمی نداریم پس ستون های نظیر مقدار null دارند:

```sql
select title, release_date, imdb, genre, duration
from movies
right outer join genres
    on genres.id = movies.genre_id;
```

Data Output | Explain | Messages | Notifications

	title character varying (50)	release_date date	imdb numeric (3,1)	genre character varying (20)	duration smallint
1	The Fault in Our Stars	2014-06-06	7.7	romance	126
2	13 Hours	2016-03-04	7.3	Action	144
3	American Sniper	2014-02-16	7.3	Drama	134
4	La La Land	2016-07-01	8.0	romance	128
5	[null]	[null]	[null]	Comedy	[null]
6	[null]	[null]	[null]	Horror	[null]

سوال پنجم

MoviesDB/postgres@PostgreSQL 13

Query Editor | Query History

genres | genre_id | gene

```sql
select title, release_date, imdb, rotten_tomatoes, box_office, director, duration, budget, genre
from movies
full outer join genres                  -- This will add 2 extra rows with many null values
    on genres.id = movies.genre_id      -- The full outer join is redundant helps use all 6 conditions
where
    director is not null                -- This will remove the 2 extra added rows by filtering the NULL values
    and title like '_________%'         -- This will remove 1 row that doesn't satisfy the given condition
    and genre = any (
        select genre from genres        -- This is redundant, but makes use of 1 condition (:
    )
    and imdb = all (
        select imdb from movies
        where imdb < 7.5                -- This will filter 1 row that doesn't satisfy the given condition
    )
    and exists (
        select *
        from movies
        where
            rotten_tomatoes > 60
    )
```

Data Output | Explain | Messages | Notifications

	title character varying (50)	release_date date	imdb numeric (3,1)	rotten_tomatoes smallint	box_office money	director character varying (30)	duration smallint	budget money	genre character varying (20)
1	American Sniper	2014-02-16	7.3	72	$547,400,000.00	Clint Eastwood	134	$35,000,000.00	Drama

برای این سوال:

- از دستور full outer join استفاده شد تا دو ردیف null برگردانده شود که در دستور بعدی استفاده خواهد شد.
- از دستور is not null برای حذف کردن دو ردیف اضافه شده توسط دستور قبل استفاده شد.
- از دستور like '_____%' استفاده شد تا یک ردیف را حذف کنیم که در شرط صدق نمیکرد.
- برای استفاده از دستور any چک کردیم تا تمام ژانر های فیلم ها در جدول genres موجود باشند.

- برای دستور all تمام فیلم هایی را برگرداندیم که imdb کمتر از 7.5 داشته باشند (یک ردیف حذف شد)
- برای دستور exists تمام فیلم هایی را برگرداندیم که rotten tomatoes بیشتر از 60 دارند.
- برای مقایسه ها از دستور < استفاده شد.