Mosab Mohamed - B20-04(Online)

Exercise 2:

2.1)

```sql
SELECT f.title, f.rating, c.name
FROM film f,
     inventory i,
     film_category fc,
     category c
WHERE i.film_id = f.film_id
  AND fc.category_id = c.category_id
  AND fc.film_id = f.film_id
  AND (c.name = 'Horror' OR c.name = 'Sci-Fi')
  AND (f.rating = 'PG-13' OR f.rating = 'R');
```

2.2)

```sql
SELECT a.address, SUM(p.amount) as total
FROM address a,
     payment p,
     store s
WHERE s.address_id = a.address_id
  AND p.staff_id = s.manager_staff_id
GROUP BY a.address
ORDER BY total DESC;
```

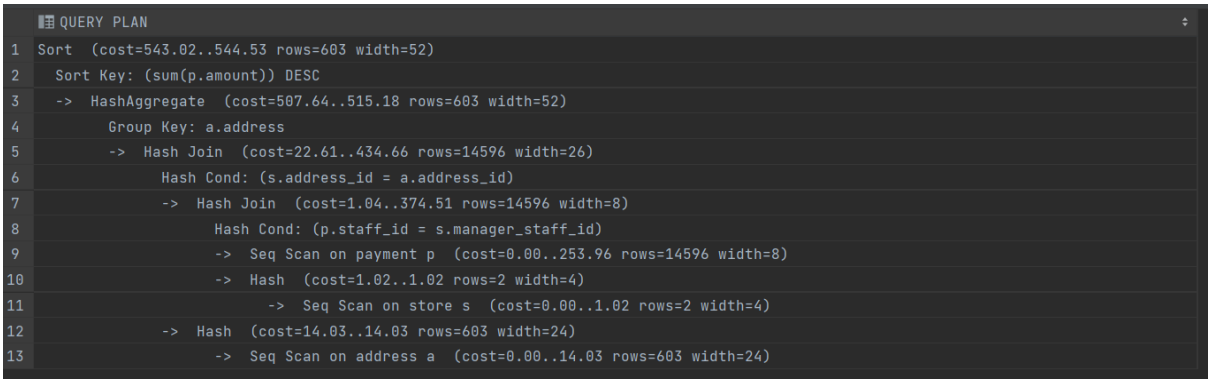
2.3)

```sql
EXPLAIN
SELECT f.title, f.rating, c.name
FROM film f,
     inventory i,
     film_category fc,
     category c
WHERE i.film_id = f.film_id
  AND fc.category_id = c.category_id
  AND fc.film_id = f.film_id
  AND (c.name = 'Horror' OR c.name = 'Sci-Fi')
  AND (f.rating = 'PG-13' OR f.rating = 'R');
```

```
QUERY PLAN
Hash Join  (cost=87.86..177.99 rows=215 width=87)
  Hash Cond: (i.film_id = f.film_id)
  ->  Seq Scan on inventory i  (cost=0.00..70.81 rows=4581 width=2)
  ->  Hash  (cost=87.27..87.27 rows=47 width=93)
        ->  Nested Loop  (cost=1.54..87.27 rows=47 width=93)
              ->  Hash Join  (cost=1.26..20.58 rows=125 width=70)
                    Hash Cond: (fc.category_id = c.category_id)
                    ->  Seq Scan on film_category fc  (cost=0.00..16.00 rows=1000 width=4)
                    ->  Hash  (cost=1.24..1.24 rows=2 width=72)
                          ->  Seq Scan on category c  (cost=0.00..1.24 rows=2 width=72)
                                Filter: (((name)::text = 'Horror'::text) OR ((name)::text = 'Sci-Fi'::text))
              ->  Index Scan using film_pkey on film f  (cost=0.28..0.53 rows=1 width=23)
                    Index Cond: (film_id = fc.film_id)
                    Filter: ((rating = 'PG-13'::mpaa_rating) OR (rating = 'R'::mpaa_rating))
```

The most expensive step is the sequential scan of the inventory to check if the film is in the inventory or not.
Probably there exists a better way to execute the query or a better order that is easier to scan in the tables.

```sql
EXPLAIN
SELECT a.address, SUM(p.amount) as total
FROM address a,
     payment p,
     store s
WHERE s.address_id = a.address_id
  AND p.staff_id = s.manager_staff_id
GROUP BY a.address
ORDER BY total DESC;
```



```
QUERY PLAN
Sort  (cost=543.02..544.53 rows=603 width=52)
  Sort Key: (sum(p.amount)) DESC
  ->  HashAggregate  (cost=507.64..515.18 rows=603 width=52)
        Group Key: a.address
        ->  Hash Join  (cost=22.61..434.66 rows=14596 width=26)
              Hash Cond: (s.address_id = a.address_id)
              ->  Hash Join  (cost=1.04..374.51 rows=14596 width=8)
                    Hash Cond: (p.staff_id = s.manager_staff_id)
                    ->  Seq Scan on payment p  (cost=0.00..253.96 rows=14596 width=8)
                    ->  Hash  (cost=1.02..1.02 rows=2 width=4)
                          ->  Seq Scan on store s  (cost=0.00..1.02 rows=2 width=4)
              ->  Hash  (cost=14.03..14.03 rows=603 width=24)
                    ->  Seq Scan on address a  (cost=0.00..14.03 rows=603 width=24)
```

The most expensive step is sequential scan of the payment table to match it with the staff id of the store table to have a concrete table that has all the different staffs which also means we will have a table that has all the different stores because each staff is uniquely assigned to a single store.
Probably there exists a better way to execute the query or a better order that is easier to scan in the tables.