Exercise 1

Total cost that take to fetch the data using command:

```
EXPLAIN SELECT * FROM public.customer
```

Output:

```
Seq Scan on customer  (cost=0.00..4034.00 rows=100000 width=211)
```

For 3 different query:

1)

```
EXPLAIN ANALYZE SELECT *
FROM public.customer
WHERE id>10 AND id<100;
```

```
Seq Scan on customer  (cost=0.00..328.00 rows=90 width=211)
```

2)

```
EXPLAIN SELECT *
FROM public.customer
WHERE name = 'Jason Lopez';
```

```
Seq Scan on customer  (cost=0.00..4534.00 rows=2 width=211)
```

3)

```
EXPLAIN SELECT *
FROM public.customer
WHERE address = 'USS Myers';
```

```
Seq Scan on customer  (cost=0.00..4784.00 rows=1 width=211)
```

Creating name index by following command:

```
CREATE INDEX idx_customer_name ON customer(name)
```

Afer Indexing: 1)

```
Index Scan using customer_pkey on customer  (cost=0.29..12.16 rows=87 width=211) (actual time=0.009..
```

2)

```
->  Bitmap Index Scan on idx_customer_name  (cost=0.00..4.43 rows=2 width=0)
```

3)

```
Index Scan using idx_customer_address on customer  (cost=0.42..8.44 rows=1 width=211)
```

```
## Exercise 2

### 1-TASK

```sql
SELECT * FROM category as cat, film_category a
WHERE (cat.name = 'Sci-Fi' or cat.name = 'Horr
  AND cat.category_id = f_c.category_id
  AND f_c.film_id = f.film_id
    AND (f.rating = 'R' OR f.rating = 'PG-13')
  AND NOT EXISTS (SELECT * FROM rental WHERE i
inventory_id)
```

ANALYZING:

```
"Nested Loop  (cost=606.27..693.69 rows=47 wid
"  ->  Hash Join  (cost=606.00..627.00 rows=12
"        Hash Cond: (f_c.category_id = cat.cat
"        ->  Seq Scan on film_category f_c  (c
width=12)"
"        ->  Hash  (cost=605.97..605.97 rows=2
"              ->  Nested Loop  (cost=510.99..
"                    ->  Hash Anti Join  (cost
"                          Hash Cond: (i.inven
"                          ->  Seq Scan on inv
rows=4581 width=16)"
"                          ->  Hash  (cost=310
```

Exercise 2

1-TASK

```
SELECT * FROM category as cat, film_category as f
WHERE (cat.name = 'Sci-Fi' or cat.name = 'Horror'
    AND cat.category_id = f_c.category_id
    AND f_c.film_id = f.film_id
    AND (f.rating = 'R' OR f.rating = 'PG-13')
    AND NOT EXISTS (SELECT * FROM rental WHERE i.
```

ANALYZING:

```
"Nested Loop  (cost=606.27..693.69 rows=47 width=
"  ->  Hash Join  (cost=606.00..627.00 rows=125 w
"        Hash Cond: (f_c.category_id = cat.catego
"        ->  Seq Scan on film_category f_c  (cost
"        ->  Hash  (cost=605.97..605.97 rows=2 wi
"              ->  Nested Loop  (cost=510.99..605
"                    ->  Hash Anti Join  (cost=51
"                          Hash Cond: (i.inventor
```
```

```
"                              ->  Hash  (cost=310
"                                    ->  Seq Scan
rows=16044 width=4)"
"                        ->  Seq Scan on category
width=80)"
"                              Filter: (((name)::t
::text = 'Horror'::text))"
"  ->  Index Scan using film_pkey on film f  (
"        Index Cond: (film_id = f_c.film_id)"
"        Filter: ((rating = 'R'::mpaa_rating)
)"
```

2-Task

```sql
WITH store_sum AS (SELECT st.*, SUM(pay.amount
          JOIN staff AS s ON pay.staff_id = s.
          JOIN store AS st ON s.store_id = st.
          WHERE TO_DATE('020107', 'MMDDYY') <
           AND pay.payment_date < TO_DATE('030
          GROUP BY st.store_id)
SELECT * FROM store_sum AS s
JOIN (SELECT ad.city_id, MAX(s.sum) FROM store
   INNER JOIN address AS ad ON s.address_id =
   GROUP BY ad.city_id)
   AS tot ON s.sum = tot.max
```

ANALYZING:

```
"Hash Join  (cost=458.74..458.82 rows=2 width=
"  Hash Cond: ((max(s_1.sum)) = s.sum)"
"  CTE store_sum"
"    ->  HashAggregate  (cost=442.26..442.28 r
"          Group Key: st.store_id"
"          ->  Hash Join  (cost=2.12..431.53 r
"                Hash Cond: (pay.staff_id = s_
"                ->  Seq Scan on payment pay
width=8)"
"                      Filter: ((to_date('0201
payment_date) AND (payment_date < to_date('030
"                ->  Hash  (cost=2.09..2.09 ro
"                      ->  Nested Loop  (cost=
"                            Join Filter: (s_2
"                            ->  Seq Scan on s
width=6)"
"                            ->  Materialize
"                                  ->  Seq Sca
rows=2 width=16)"
"  ->  GroupAggregate  (cost=16.39..16.42 rows
```

```
"                              ->  Seq Scan on invent
"                              ->  Hash  (cost=310.44
"                                    ->  Seq Scan on
"                        ->  Seq Scan on category cat
"                              Filter: (((name)::text
"  ->  Index Scan using film_pkey on film f  (cos
"        Index Cond: (film_id = f_c.film_id)"
"        Filter: ((rating = 'R'::mpaa_rating) OR
```

2-Task

```
WITH store_sum AS (SELECT st.*, SUM(pay.amount) F
                   JOIN staff AS s ON pay.staff_id
                   JOIN store AS st ON s.store_id
                   WHERE TO_DATE('020107', 'MMDDYY
                    AND pay.payment_date < TO_DATE
                   GROUP BY st.store_id)
SELECT * FROM store_sum AS s
JOIN (SELECT ad.city_id, MAX(s.sum) FROM store_su
     INNER JOIN address AS ad ON s.address_id = a
     GROUP BY ad.city_id)
     AS tot ON s.sum = tot.max
```

ANALYZING:

```
"Hash Join  (cost=458.74..458.82 rows=2 width=82)
"  Hash Cond: ((max(s_1.sum)) = s.sum)"
"  CTE store_sum"
"    ->  HashAggregate  (cost=442.26..442.28 rows
"          Group Key: st.store_id"
"          ->  Hash Join  (cost=2.12..431.53 rows
"                Hash Cond: (pay.staff_id = s_2.s
"                ->  Seq Scan on payment pay  (co
"                      Filter: ((to_date('020107'
"                ->  Hash  (cost=2.09..2.09 rows=
"                      ->  Nested Loop  (cost=0.0
"                            Join Filter: (s_2.st
"                            ->  Seq Scan on staf
"                            ->  Materialize  (co
"                                  ->  Seq Scan o
"  ->  GroupAggregate  (cost=16.39..16.42 rows=2
```

```
"        Group Key: ad.city_id"
"        ->  Sort  (cost=16.39..16.39 rows=2 w
"              Sort Key: ad.city_id"
"              ->  Hash Join  (cost=0.07..16.3
"                    Hash Cond: (ad.address_id
"                    ->  Seq Scan on address a
width=6)"
"                    ->  Hash  (cost=0.04..0.0
"                          ->  CTE Scan on sto
rows=2 width=34)"
"  ->  Hash  (cost=0.04..0.04 rows=2 width=48)
```

```
"        Group Key: ad.city_id"
"        ->  Sort  (cost=16.39..16.39 rows=2 widt
"              Sort Key: ad.city_id"
"              ->  Hash Join  (cost=0.07..16.38 r
"                    Hash Cond: (ad.address_id =
"                    ->  Seq Scan on address ad
"                    ->  Hash  (cost=0.04..0.04 r
"                          ->  CTE Scan on store_
"  ->  Hash  (cost=0.04..0.04 rows=2 width=48)"
```