P3 Result to EXPLAIN

Exercise 2

a)

```
                                QUERY PLAN
-------------------------------------------------------------
 Seq Scan on customers  (cost=0.00..721.00 rows=995 width=156)
   Filter: ((country)::text = 'Japan'::text)
(2 rows)
```

Excercise 3

b)

```
                                QUERY PLAN
-------------------------------------------------------------
 Seq Scan on customers  (cost=0.00..721.00 rows=995 width=156)
   Filter: ((country)::text = 'Japan'::text)
(2 rows)
```

d)

```
                                        QUERY PLAN
-----------------------------------------------------------------------------------------
 Index Scan using customers_country on customers  (cost=0.00..56.66 rows=995 width=156)
   Index Cond: ((country)::text = 'Japan'::text)
(2 rows)
```

Exercise 4

a)

```
                                QUERY PLAN
-------------------------------------------------------------------------
 Hash Join  (cost=733.44..1004.41 rows=597 width=8)
   Hash Cond: (o.customerid = c.customerid)
   ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=12)
   ->  Hash  (cost=721.00..721.00 rows=995 width=4)
         ->  Seq Scan on customers c  (cost=0.00..721.00 rows=995 width=4)
               Filter: ((country)::text = 'Japan'::text)
(6 rows)
```

c)

```
                                QUERY PLAN
-------------------------------------------------------------------------
 Merge Join  (cost=1803.59..1874.53 rows=597 width=8)
   Merge Cond: (c.customerid = o.customerid)
   ->  Sort  (cost=770.54..773.03 rows=995 width=4)
         Sort Key: c.customerid
         ->  Seq Scan on customers c  (cost=0.00..721.00 rows=995 width=4)
               Filter: ((country)::text = 'Japan'::text)
   ->  Sort  (cost=1033.04..1063.04 rows=12000 width=12)
         Sort Key: o.customerid
```

```
        ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=12)
(9 rows)
```

It uses Merge join.
The total cost is 1874.53

d)

```
                                QUERY PLAN
-------------------------------------------------------------------------------
 Nested Loop  (cost=0.00..5749.36 rows=597 width=8)
   ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=12)
   ->  Index Scan using customers_pkey on customers c  (cost=0.00..0.45 rows=1 width=4)
         Index Cond: (c.customerid = o.customerid)
         Filter: ((c.country)::text = 'Japan'::text)
(5 rows)
```

It uses Nested Loop
The total cost is 5749.36

Exercise 5

a)

```
                                QUERY PLAN
-------------------------------------------------------------------------------
 Sort  (cost=1501.33..1501.36 rows=11 width=13)
   Sort Key: (avg(o.totalamount))
   ->  HashAggregate  (cost=1501.00..1501.14 rows=11 width=13)
         ->  Hash Join  (cost=921.00..1441.00 rows=12000 width=13)
               Hash Cond: (o.customerid = c.customerid)
               ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=12)
               ->  Hash  (cost=671.00..671.00 rows=20000 width=9)
                     ->  Seq Scan on customers c  (cost=0.00..671.00 rows=20000 width=9)
(8 rows)
```

```
                                QUERY PLAN

-------------------------------------------------------------------------------------------------
----
 Sort  (cost=2325.52..2325.55 rows=11 width=13)
   Sort Key: (avg(o.totalamount))
   ->  HashAggregate  (cost=2325.19..2325.33 rows=11 width=13)
         ->  Merge Join  (cost=1033.15..2265.19 rows=12000 width=13)
               Merge Cond: (c.customerid = o.customerid)
               ->  Index Scan using customers_pkey on customers c  (cost=0.00..1002.25 rows=20000 width
=9)
               ->  Sort  (cost=1033.04..1063.04 rows=12000 width=12)
                     Sort Key: o.customerid
                     ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=12)
(9 rows)
```

b)

```
                                                QUERY PLAN
-----------------------------------------------------------------------------------------------------
 Merge Join  (cost=1033.15..2265.19 rows=12000 width=192)
   Merge Cond: (c.customerid = o.customerid)
   ->  Index Scan using customers_pkey on customers c  (cost=0.00..1002.25 rows=20000 width=156)
   ->  Sort  (cost=1033.04..1063.04 rows=12000 width=36)
         Sort Key: o.customerid
         ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=36)
(6 rows)

                                        QUERY PLAN
-------------------------------------------------------------------------------
 Sort  (cost=3783.54..3813.54 rows=12000 width=192)
   Sort Key: c.customerid
   ->  Hash Join  (cost=370.00..1861.00 rows=12000 width=192)
         Hash Cond: (c.customerid = o.customerid)
         ->  Seq Scan on customers c  (cost=0.00..671.00 rows=20000 width=156)
         ->  Hash  (cost=220.00..220.00 rows=12000 width=36)
               ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=36)
(7 rows)
```

Exercise 6

a)

```
                                QUERY PLAN
-------------------------------------------------------------------------
 Seq Scan on customers c  (cost=0.00..5001021.00 rows=6667 width=15)
   Filter: (4 < (SubPlan 1))
   SubPlan 1
     ->  Aggregate  (cost=250.00..250.01 rows=1 width=0)
           ->  Seq Scan on orders o  (cost=0.00..250.00 rows=1 width=0)
                 Filter: (customerid = $0)
(6 rows)
```

The total cost is 5001021.00

b)

```
                                QUERY PLAN
-------------------------------------------------------------------------
 Hash Join  (cost=1231.00..1703.29 rows=8996 width=15)
   Hash Cond: (o.customerid = c.customerid)
   ->  HashAggregate  (cost=310.00..467.43 rows=8996 width=4)
         Filter: (count(*) > 4)
         ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=4)
   ->  Hash  (cost=671.00..671.00 rows=20000 width=15)
         ->  Seq Scan on customers c  (cost=0.00..671.00 rows=20000 width=15)
(7 rows)
```

Exercise 7

a)

```
                                                       QUERY PLAN
----------------------------------------------------------------------------------------------------
 Sort  (cost=614926.51..614927.01 rows=199 width=130)
   Sort Key: ordercounts1.customerid
   ->  Subquery Scan ordercounts1  (cost=1008.88..614918.91 rows=199 width=130)
         Filter: (5 >= (SubPlan 1))
         ->  HashAggregate  (cost=1008.88..1016.35 rows=597 width=15)
               ->  Hash Join  (cost=733.44..1004.41 rows=597 width=15)
                     Hash Cond: (o.customerid = c.customerid)
                     ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=4)
                     ->  Hash  (cost=721.00..721.00 rows=995 width=15)
                           ->  Seq Scan on customers c  (cost=0.00..721.00 rows=995 width=15)
                                 Filter: ((country)::text = 'Japan'::text)
         SubPlan 1
           ->  Aggregate  (cost=1028.29..1028.30 rows=1 width=0)
                 ->  HashAggregate  (cost=1010.38..1020.83 rows=597 width=15)
                       Filter: ($0 < count(*))
                       ->  Hash Join  (cost=733.44..1004.41 rows=597 width=15)
                             Hash Cond: (o.customerid = c.customerid)
                             ->  Seq Scan on orders o  (cost=0.00..220.00 rows=12000 width=4)
                             ->  Hash  (cost=721.00..721.00 rows=995 width=15)
                                   ->  Seq Scan on customers c  (cost=0.00..721.00 rows=995 width=15)
                                         Filter: ((country)::text = 'Japan'::text)
(21 rows)
```

d)

```
                                                                        QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=14629.76..14629.77 rows=5 width=23)
   Sort Key: c.customerid
   ->  Hash Join  (cost=13766.46..14629.70 rows=5 width=23)
         Hash Cond: (c.customerid = o.customerid)
         ->  Nested Loop  (cost=12738.17..13588.50 rows=166 width=19)
               ->  Subquery Scan f  (cost=12738.17..12745.64 rows=166 width=4)
                     Filter: (f.orank <= 5)
                     ->  Sort  (cost=12738.17..12739.42 rows=498 width=4)
                           Sort Key: ((0)::bigint)
                           ->  HashAggregate  (cost=12710.88..12715.86 rows=498 width=4)
                                 ->  Append  (cost=2029.74..12708.39 rows=498 width=4)
                                       ->  Subquery Scan "*SELECT* 1"  (cost=2029.74..2048.43 rows=298
width=4)
                                             ->  Hash Join  (cost=2029.74..2045.45 rows=298 width=4)
                                                   Hash Cond: ((count(*)) = (max((count(*)))))
                                                   ->  HashAggregate  (cost=1007.39..1014.85 rows=597
width=4)
                                                         ->  Hash Join  (cost=733.44..1004.41 rows=597
width=4)
                                                               Hash Cond: (o.customerid = c.customerid)
                                                               ->  Seq Scan on orders o
```

(cost=0.00..220.00 rows=12000 width=4)
-> Hash (cost=721.00..721.00 rows=995 width=4)
-> Seq Scan on customers c (cost=0.00..721.00 rows=995 width=4)
Filter: ((country)::text = 'Japan'::text)
-> Hash (cost=1022.34..1022.34 rows=1 width=8)
-> Aggregate (cost=1022.32..1022.33 rows=1 width=8)
-> HashAggregate (cost=1007.39..1014.85 rows=597 width=4)
-> Hash Join (cost=733.44..1004.41 rows=597 width=4)
Hash Cond: (o.customerid = c.customerid)
-> Seq Scan on orders o (cost=0.00..220.00 rows=12000 width=4)
-> Hash (cost=721.00..721.00 rows=995 width=4)
-> Seq Scan on customers c (cost=0.00..721.00 rows=995 width=4)
Filter: ((country)::text = 'Japan'::text)
-> HashAggregate (cost=10655.46..10657.96 rows=200 width=4)
-> Nested Loop (cost=2028.81..10061.45 rows=118803 width=4)
Join Filter: ((count(*)) < o1.numorders)
-> HashAggregate (cost=1007.39..1014.85 rows=597 width=4)
-> Hash Join (cost=733.44..1004.41 rows=597 width=4)
Hash Cond: (o.customerid = c.customerid)
-> Seq Scan on orders o (cost=0.00..220.00 rows=12000 width=4)
-> Hash (cost=721.00..721.00 rows=995 width=4)
-> Seq Scan on customers c (cost=0.00..721.00 rows=995 width=4)
Filter: ((country)::text = 'Japan'::text)
-> Materialize (cost=1021.42..1027.39 rows=597 width=8)
-> Subquery Scan o1 (cost=1007.39..1020.82 rows=597 width=8)
-> HashAggregate (cost=1007.39..1014.85 rows=597 width=4)
-> Hash Join (cost=733.44..1004.41 rows=597 width=4)

Hash Cond: (o.customerid = c.customerid)
-> Seq Scan on orders o (cost=0.00..220.00 rows=12000 width=4)
-> Hash (cost=721.00..721.00 rows=995 width=4)
-> Seq Scan on customers c (cost=0.00..721.00 rows=995 width=4)
Filter: ((country)::text = 'Japan'::text)
-> Index Scan using customers_pkey on customers c (cost=0.00..5.06 rows=1 width=15)
Index Cond: (c.customerid = f.customerid)
-> Hash (cost=1020.82..1020.82 rows=597 width=12)
-> Subquery Scan o (cost=1007.39..1020.82 rows=597 width=12)
-> HashAggregate (cost=1007.39..1014.85 rows=597 width=4)
-> Hash Join (cost=733.44..1004.41 rows=597 width=4)
Hash Cond: (o.customerid = c.customerid)
-> Seq Scan on orders o (cost=0.00..220.00 rows=12000 width=4)
-> Hash (cost=721.00..721.00 rows=995 width=4)
-> Seq Scan on customers c (cost=0.00..721.00 rows=995 width=4)
Filter: ((country)::text = 'Japan'::text)
(60 rows)