Part A:

A.Q1 Run the following sql query. What is the execution plan of the query? Use the EXPLAIN statement of postgresql to find out the execution plan.

"Seq Scan on played_in (cost=0.00..1120.00 rows=3618 width=19) (actual time=0.016..5.958 rows=3624 loops=1)"

" Filter: ("position" = 1)"

"Total runtime: 6.049 ms"

A.Q2 What is the estimated cost of the plan in A.1?

0.00 To start the plan and 1120.00 to execute it.

A.Q3 Build an index on position. Run the same query in A.1. Now, what is the execution plan of the query?

""Bitmap Heap Scan on played_in (cost=72.30..500.52 rows=3618 width=19) (actual time=0.363..0.982 rows=3624 loops=1)"

" Recheck Cond: ("position" = 1)"

" -> Bitmap Index Scan on a3 (cost=0.00..71.39 rows=3618 width=0) (actual time=0.317..0.317 rows=3624 loops=1)"

"   Index Cond: ("position" = 1)"

"Total runtime: 1.058 ms"4- 4.29..12.76

A.Q4 What is the estimated cost of the plan in A.Q3?

Cost to start the heap scan 72.30 and the cost to run the heap scan 500.52, while the index scan cost 0 to start and 71.39 to compute.

A.Q5 Did the execution plan and the cost change after the addition of the index? Explain why they are the same or different.

The costs did change as the index changed the way postgres searches for data rather than using SEQ Scan, post gres estimated that a heap scan and an index scan would be more efficient.

Part B:

B.Q1 Run the following sql query. What is the execution plan of the query?

```
"Seq Scan on played_in  (cost=0.00..1120.00 rows=114 width=19) (actual time=0.139..6.478
rows=118 loops=1)"

"  Filter: ((name)::text ~~ '%pele%'::text)"

"Total runtime: 6.497 ms"
```

B.Q2 What is the estimated cost of the plan in B.Q1?

0.00 to initiate the plan and 1120 to execute the plan.

B.Q3 Build an index on the name attribute of the relation played_in. Run the same query specified in B.Q1. Now, what is the execution plan of the query?

```
"Seq Scan on played_in  (cost=0.00..1120.00 rows=114 width=19) (actual
time=0.138..5.896 rows=118 loops=1)"
"  Filter: ((name)::text ~~ '%pele%'::text)"
"Total runtime: 5.911 ms"
```

B.Q4 What is the estimated cost of the plan in B.3?

0 to initiate plan and 1120 to execute it.

B.Q5 Did the execution plan and the cost change after the addition of the index? Explain why they are the same or different.

No it did not change as postgres estimated a Seq Scan would still be the most appropriate for execution.

Part C:

C.Q1: Run the SQL query. What is the Execution Plan of the query?

```
"Seq Scan on cup_matches  (cost=0.00..61.20 rows=893 width=28) (actual time=0.015..0.826 rows=1002 loops=1)"
"  Filter: ((rating * 3::numeric) > 20::numeric)"
"Total runtime: 0.858 ms"
```

C.Q2: What is the estimated cost of the plan in C.Q1?

0.00 to initiate the plan and 61.20 to execute it .

C.Q3: Create the index on the rating attribute of the relation cup_matches. Run the query shown in C.Q1. Now, what is the execution plan of the query?

```
"Seq Scan on cup_matches  (cost=0.00..61.20 rows=893 width=28) (actual time=0.006..0.794 rows=1002 loops=1)"
"  Filter: ((rating * 3::numeric) > 20::numeric)"
"Total runtime: 0.826 ms"
```

C.Q4: What is the estimated cost of the plan in C.Q3?

0.00 to initiate the plan and 61.20 to execute it .

C.Q5: Did the execution plan and the cost change after the addition of the index? Explain why they are the same or different.

No it did not change as postgres estimated a Seq Scan would still be the most appropriate for execution since there is a unique index for each unique name which will eventually be as costly as the seq scan.

Part D:

D.01: Run the following sql query. What is the execution plan of the query? What is the estimated cost of the plan?

```
"Hash Join  (cost=81.30..51956.41 rows=4390101 width=47) (actual time=0.490..613.702 rows=4392789 loops=1)"

"  Hash Cond: (played_in.year = cup_matches.year)"

"  ->  Seq Scan on played_in  (cost=0.00..972.60 rows=58960 width=19) (actual time=0.005..2.780 rows=58960 loops=1)"

"  ->  Hash  (cost=47.80..47.80 rows=2680 width=28) (actual time=0.471..0.471 rows=2680 loops=1)"

"        ->  Seq Scan on cup_matches  (cost=0.00..47.80 rows=2680 width=28) (actual time=0.002..0.150 rows=2680 loops=1)"

"Total runtime: 678.087 ms"
```

D.Q2 Make an index on cup_matches.year, and run the query shown in D.Q1. Show the plan and the estimated cost.

```
"Hash Join  (cost=81.30..51956.41 rows=4390101 width=47) (actual time=0.472..607.333 rows=4392789 loops=1)"

"  Hash Cond: (played_in.year = cup_matches.year)"

"  ->  Seq Scan on played_in  (cost=0.00..972.60 rows=58960 width=19) (actual time=0.003..2.558 rows=58960 loops=1)"

"  ->  Hash  (cost=47.80..47.80 rows=2680 width=28) (actual time=0.462..0.462 rows=2680 loops=1)"

"        ->  Seq Scan on cup_matches  (cost=0.00..47.80 rows=2680 width=28) (actual time=0.002..0.142 rows=2680 loops=1)"

"Total runtime: 672.240 ms"
```

D.Q3 Did the execution plan and the cost change after the addition of the index at D.Q2? Explain why they are the same or different.

No they didn’t not change as postgres estimated that the hash scan would still be the best way to do the search.

D.Q4 Make an additional index on played_in.year. Now there are two indexes created, each for cup_matches.year and played_in.year. Run the query shown in D.Q1. Show the plan and the estimated cost.

```
"Hash Join  (cost=81.30..51956.41 rows=4390101 width=47) (actual time=0.491..611.477 rows=4392789 loops=1)"

"  Hash Cond: (played_in.year = cup_matches.year)"

"  ->  Seq Scan on played_in  (cost=0.00..972.60 rows=58960 width=19) (actual time=0.004..2.537 rows=58960 loops=1)"

"  ->  Hash  (cost=47.80..47.80 rows=2680 width=28) (actual time=0.479..0.479 rows=2680 loops=1)"

"        ->  Seq Scan on cup_matches  (cost=0.00..47.80 rows=2680 width=28) (actual time=0.002..0.145 rows=2680 loops=1)"

"Total runtime: 676.091 ms"
```

D.Q5 Did the execution plan and the cost change after the addition of the index at D.Q4? Explain why they are the same or different.

No they didn’t not change as postgres estimated that the hash scan would still be the best way to do the search so multiple indexes did not make any difference.

Part E:

E.Q1 Run the following sql query. What is the execution plan of the query? What is the estimated cost of the plan? Which join algorithm is used?

```
1- "Hash Join  (cost=81.30..2012.00 rows=58960 width=47) (actual time=0.463..18.537 rows=58960 loops=1)"

"  Hash Cond: (played_in.mid = cup_matches.mid)"

"  ->  Seq Scan on played_in  (cost=0.00..972.60 rows=58960 width=19) (actual time=0.006..2.700 rows=58960 loops=1)"

"  ->  Hash  (cost=47.80..47.80 rows=2680 width=28) (actual time=0.451..0.451 rows=2680 loops=1)"

"        ->  Seq Scan on cup_matches  (cost=0.00..47.80 rows=2680 width=28) (actual time=0.002..0.157 rows=2680 loops=1)"

"Total runtime: 19.575 ms"
```

<u>It used Hash Join</u>

E.Q2 Disable the join algorithm used in the plan of the query in E.Q1, by using the "SET" command in postgresql. Now, what is the execution plan of the query? What is the estimated cost of the plan? Which join algorithm is used?

```
"Merge Join  (cost=0.00..4472.81 rows=58960 width=47) (actual time=0.026..28.682 rows=58960 loops=1)"

"  Merge Cond: (cup_matches.mid = played_in.mid)"

"  ->  Index Scan using cup_matches_pkey on cup_matches  (cost=0.00..104.45 rows=2680 width=28) (actual time=0.013..0.506 rows=2680 loops=1)"

"  ->  Index Scan using played_in_pkey on played_in  (cost=0.00..3624.66 rows=58960 width=19) (actual time=0.011..19.960 rows=58960 loops=1)"

"Total runtime: 29.676 ms"
```

<u>It used Merge Join</u>

E.Q3 Disable the join algorithm used in the plan in E.Q2, by using the "SET" command in postgresql. At this point, you disabled the two join algorithms. Now, what is the execution plan of the query? What is the estimated cost of the plan? Which join algorithm is used?

```
"Nested Loop  (cost=0.00..5258.97 rows=58960 width=47) (actual time=0.016..26.108 rows=58960 loops=1)"
```

```
"  ->  Seq Scan on cup_matches  (cost=0.00..47.80 rows=2680 width=28) (actual
time=0.005..0.145 rows=2680 loops=1)"

"  ->  Index Scan using played_in_pkey on played_in  (cost=0.00..1.67 rows=22 width=19)
(actual time=0.002..0.007 rows=22 loops=2680)"

"        Index Cond: (played_in.mid = cup_matches.mid)"

"Total runtime: 27.122 ms"
```

<u>It Used Nested Loop</u>