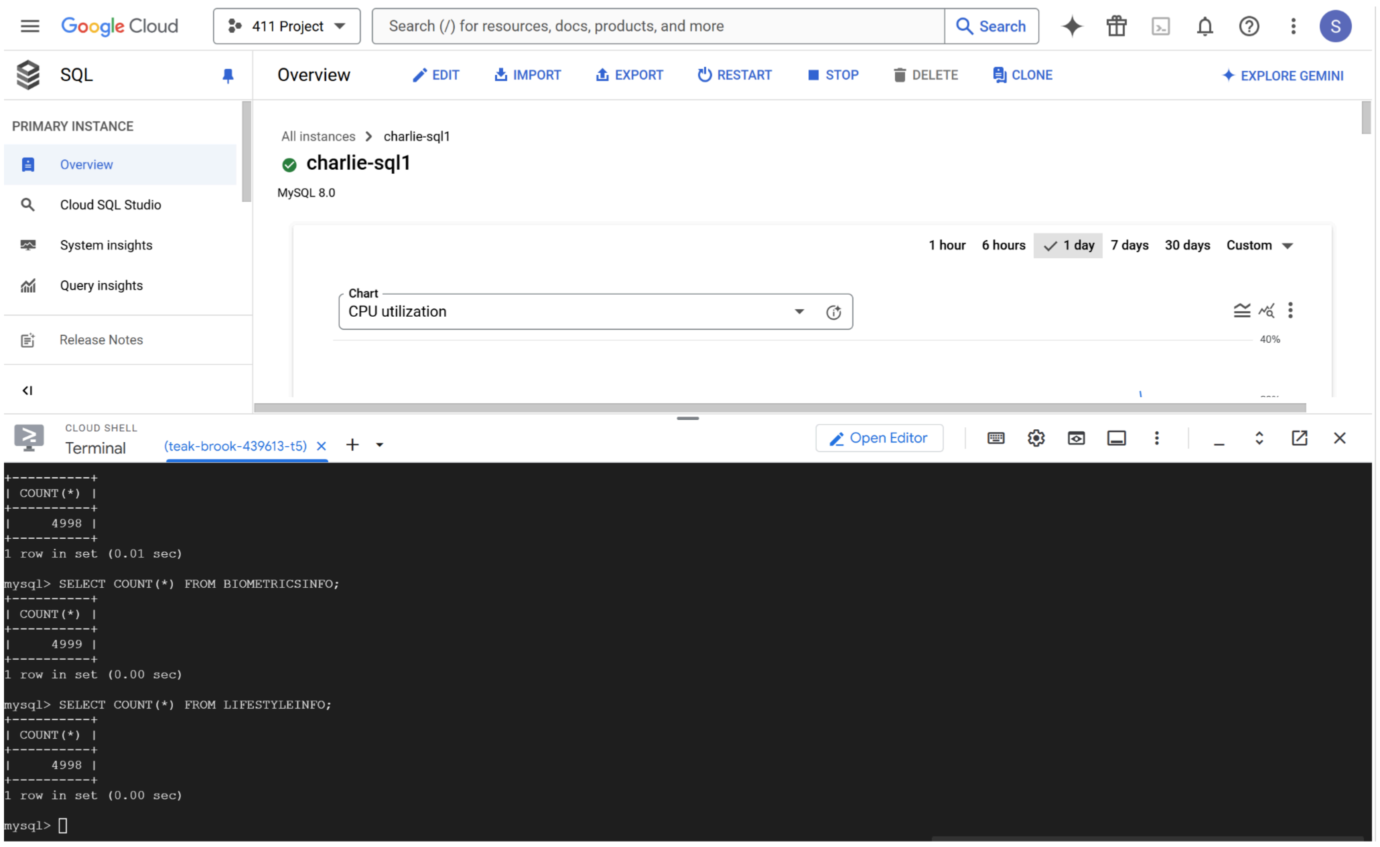
Stage 3 Deliverables:

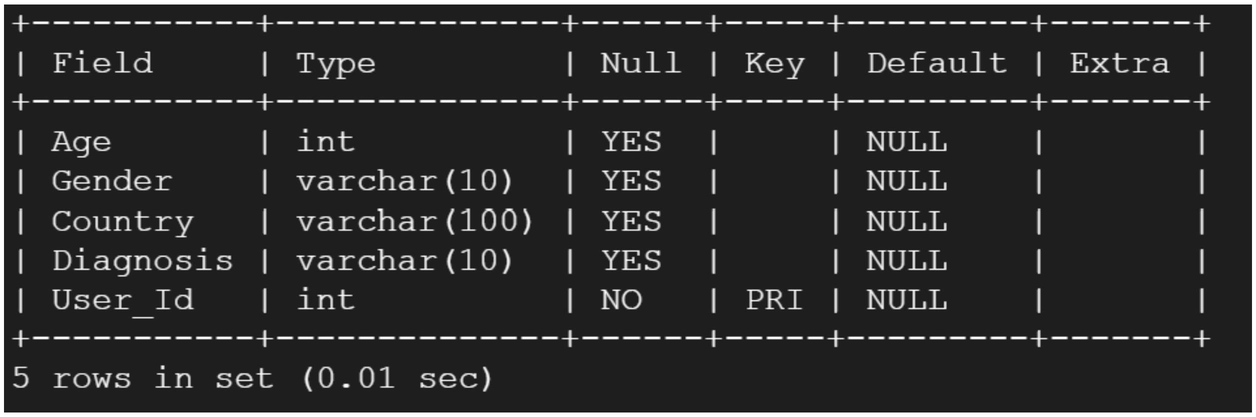
[Database Design Document / DDL Commands]

Proof of Connection to GCP:

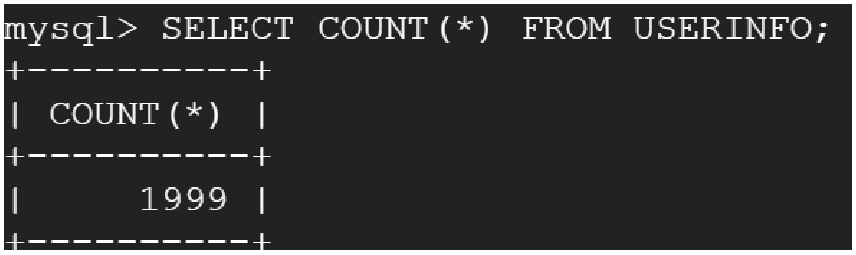


```
+----------+
| COUNT(*) |
+----------+
|     4998 |
+----------+
1 row in set (0.01 sec)

mysql> SELECT COUNT(*) FROM BIOMETRICSINFO;
+----------+
| COUNT(*) |
+----------+
|     4999 |
+----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM LIFESTYLEINFO;
+----------+
| COUNT(*) |
+----------+
|     4998 |
+----------+
1 row in set (0.00 sec)

mysql> 
```

USERINFO table:



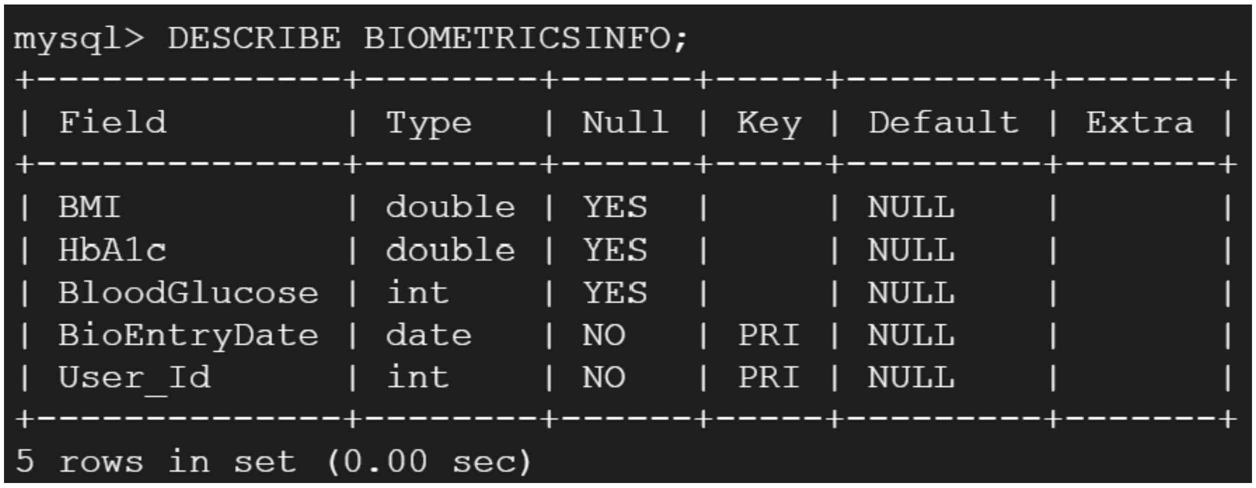
```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| Age       | int          | YES  |     | NULL    |       |
| Gender    | varchar(10)  | YES  |     | NULL    |       |
| Country   | varchar(100) | YES  |     | NULL    |       |
| Diagnosis | varchar(10)  | YES  |     | NULL    |       |
| User_Id   | int          | NO   | PRI | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```



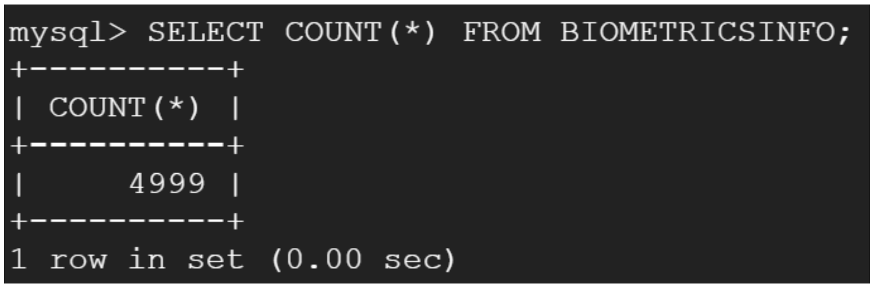
```
mysql> SELECT COUNT(*) FROM USERINFO;
+----------+
| COUNT(*) |
+----------+
|     1999 |
+----------+
```

2000 rows

BIOMETRICSINFO table:



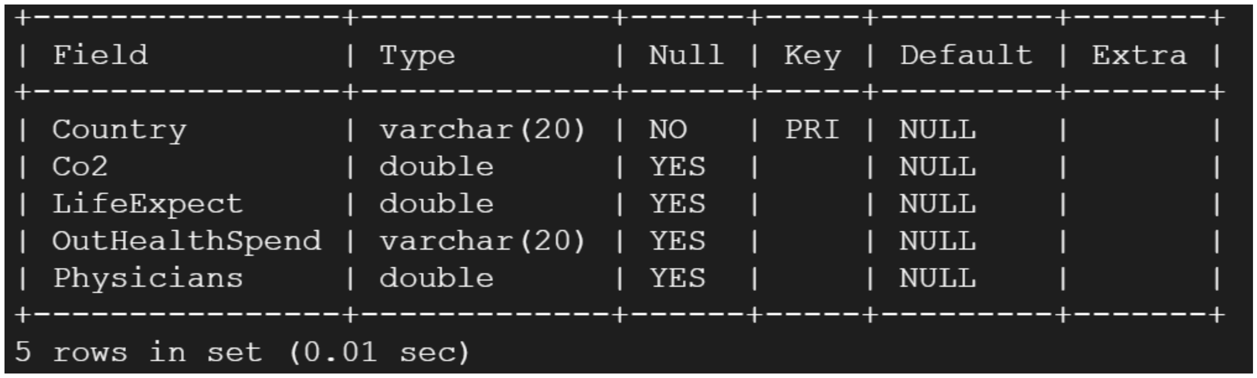
```
mysql> DESCRIBE BIOMETRICSINFO;
+--------------+--------+------+-----+---------+-------+
| Field        | Type   | Null | Key | Default | Extra |
+--------------+--------+------+-----+---------+-------+
| BMI          | double | YES  |     | NULL    |       |
| HbA1c        | double | YES  |     | NULL    |       |
| BloodGlucose | int    | YES  |     | NULL    |       |
| BioEntryDate | date   | NO   | PRI | NULL    |       |
| User_Id      | int    | NO   | PRI | NULL    |       |
+--------------+--------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```



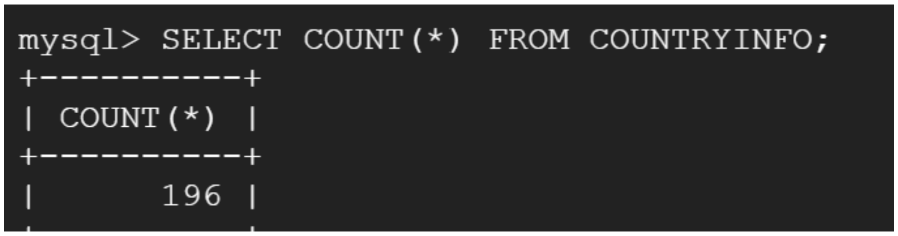
```
mysql> SELECT COUNT(*) FROM BIOMETRICSINFO;
+----------+
| COUNT(*) |
+----------+
|     4999 |
+----------+
1 row in set (0.00 sec)
```

5000 rows

COUNTRYINFO table:



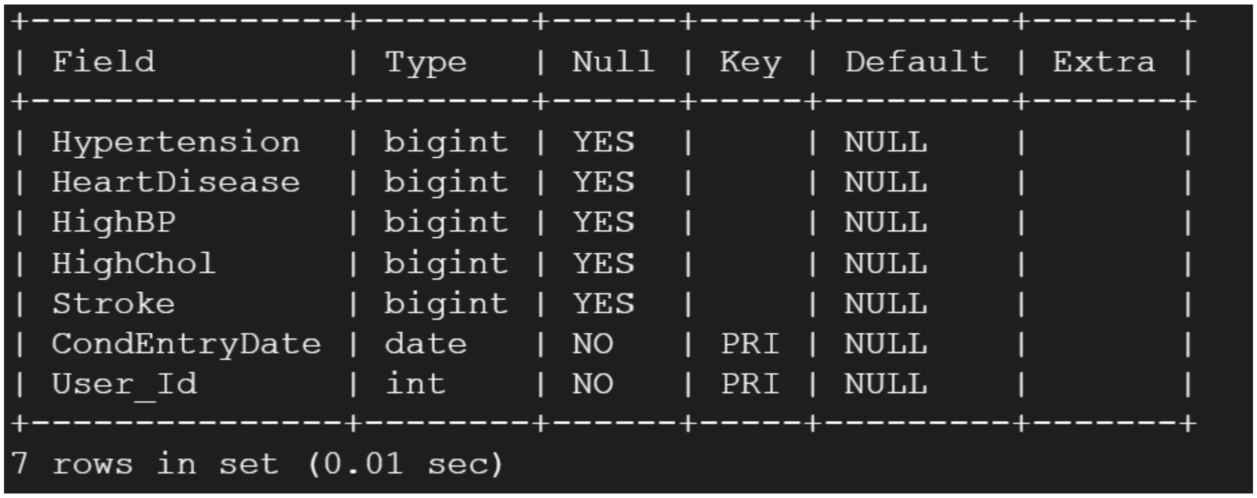
```
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| Country        | varchar(20) | NO   | PRI | NULL    |       |
| Co2            | double      | YES  |     | NULL    |       |
| LifeExpect     | double      | YES  |     | NULL    |       |
| OutHealthSpend | varchar(20) | YES  |     | NULL    |       |
| Physicians     | double      | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```



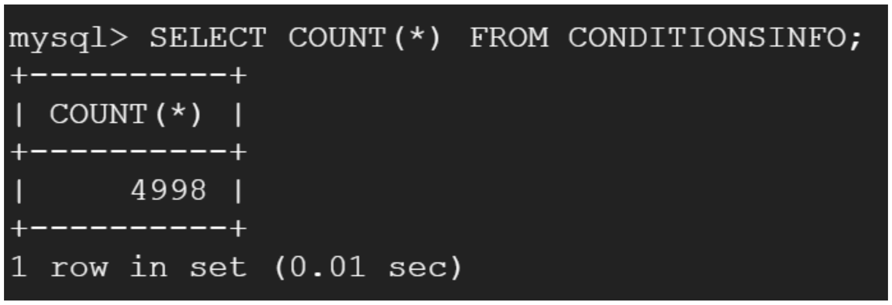
```
mysql> SELECT COUNT(*) FROM COUNTRYINFO;
+----------+
| COUNT(*) |
+----------+
|      196 |
```

200 rows

CONDITIONSINFO table:



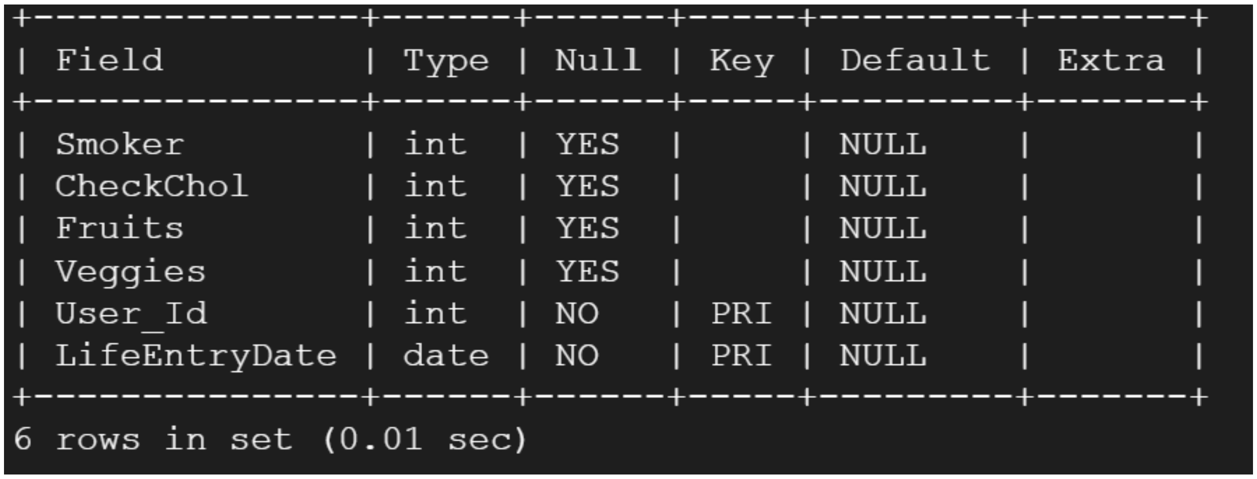
```
+---------------+--------+------+-----+---------+-------+
| Field         | Type   | Null | Key | Default | Extra |
+---------------+--------+------+-----+---------+-------+
| Hypertension  | bigint | YES  |     | NULL    |       |
| HeartDisease  | bigint | YES  |     | NULL    |       |
| HighBP        | bigint | YES  |     | NULL    |       |
| HighChol      | bigint | YES  |     | NULL    |       |
| Stroke        | bigint | YES  |     | NULL    |       |
| CondEntryDate | date   | NO   | PRI | NULL    |       |
| User_Id       | int    | NO   | PRI | NULL    |       |
+---------------+--------+------+-----+---------+-------+
7 rows in set (0.01 sec)
```

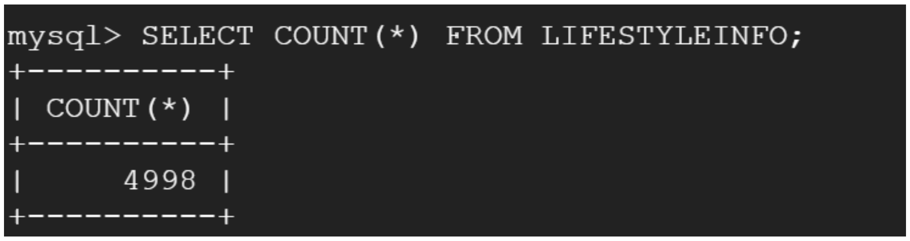


```
mysql> SELECT COUNT(*) FROM CONDITIONSINFO;
+----------+
| COUNT(*) |
+----------+
|     4998 |
+----------+
1 row in set (0.01 sec)
```

5000 rows

LIFESTYLEINFO table;


```
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| Smoker        | int  | YES  |     | NULL    |       |
| CheckChol     | int  | YES  |     | NULL    |       |
| Fruits        | int  | YES  |     | NULL    |       |
| Veggies       | int  | YES  |     | NULL    |       |
| User_Id       | int  | NO   | PRI | NULL    |       |
| LifeEntryDate | date | NO   | PRI | NULL    |       |
+---------------+------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

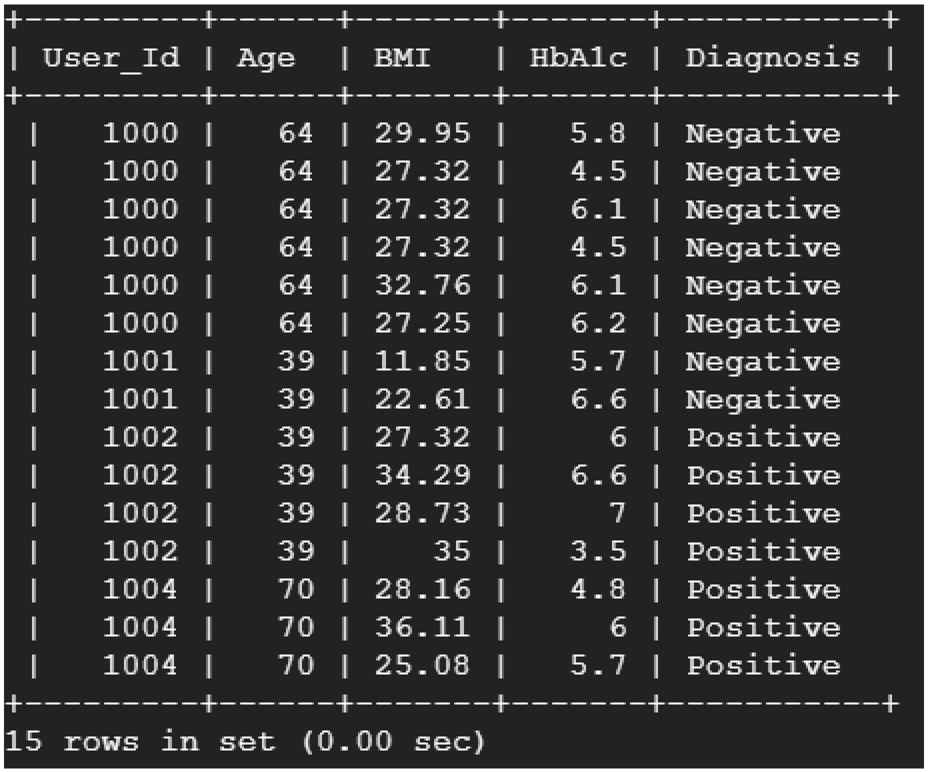

```
mysql> SELECT COUNT(*) FROM LIFESTYLEINFO;
+----------+
| COUNT(*) |
+----------+
|     4998 |
+----------+
```
5000 rows

Advanced Queries:

Query 1:

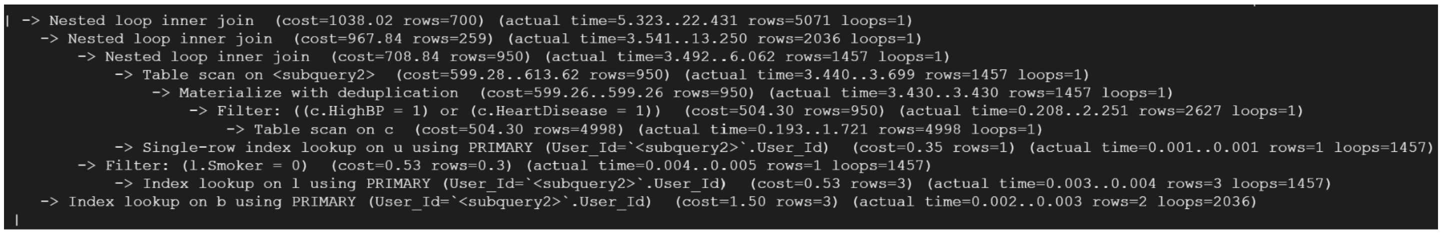
```
SELECT u.User_Id, u.Age, b.BMI, b.HbA1c, u.Diagnosis
FROM USERINFO u
JOIN LIFESTYLEINFO l ON u.User_Id = l.User_Id
JOIN BIOMETRICSINFO b ON u.User_Id = b.User_Id
WHERE l.Smoker = 0
AND EXISTS (
  SELECT 1
  FROM CONDITIONSINFO c
  WHERE c.User_Id = u.User_Id
  AND (c.HighBP = 1 OR c.HeartDisease = 1)
);
```

Output (First 15 rows):



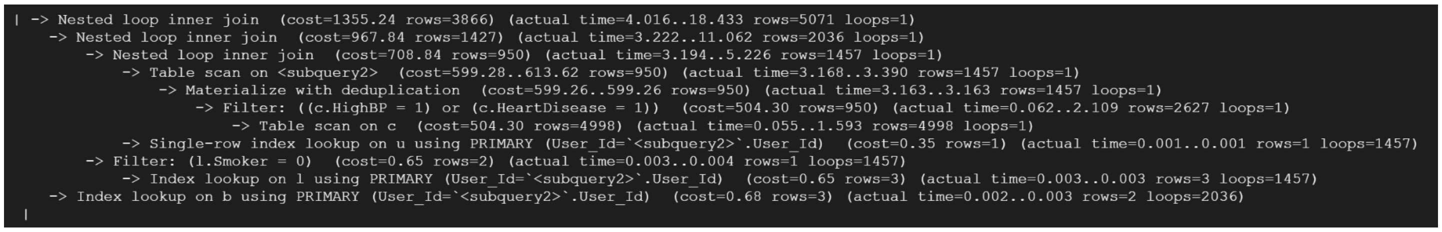
```
+---------+------+-------+-------+----------+
| User_Id | Age  | BMI   | HbA1c | Diagnosis |
+---------+------+-------+-------+----------+
 |   1000 |   64 | 29.95 |   5.8 | Negative
 |   1000 |   64 | 27.32 |   4.5 | Negative
 |   1000 |   64 | 27.32 |   6.1 | Negative
 |   1000 |   64 | 27.32 |   4.5 | Negative
 |   1000 |   64 | 32.76 |   6.1 | Negative
 |   1000 |   64 | 27.25 |   6.2 | Negative
 |   1001 |   39 | 11.85 |   5.7 | Negative
 |   1001 |   39 | 22.61 |   6.6 | Negative
 |   1002 |   39 | 27.32 |     6 | Positive
 |   1002 |   39 | 34.29 |   6.6 | Positive
 |   1002 |   39 | 28.73 |     7 | Positive
 |   1002 |   39 |    35 |   3.5 | Positive
 |   1004 |   70 | 28.16 |   4.8 | Positive
 |   1004 |   70 | 36.11 |     6 | Positive
 |   1004 |   70 | 25.08 |   5.7 | Positive
+---------+------+-------+-------+----------+
15 rows in set (0.00 sec)
```

Default explain analyze (total cost 1038.02):



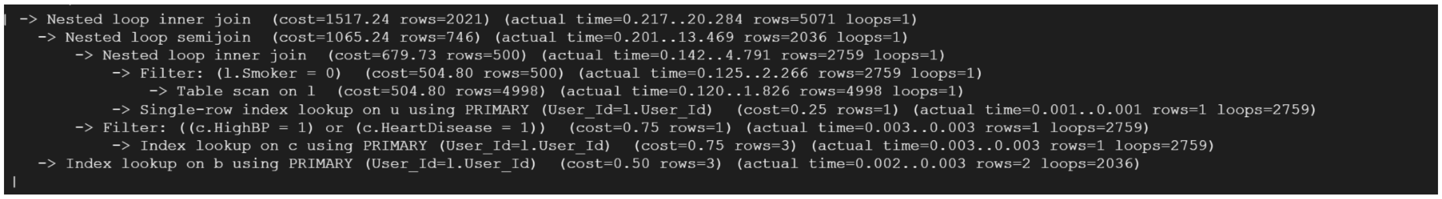
```
| -> Nested loop inner join  (cost=1038.02 rows=700) (actual time=5.323..22.431 rows=5071 loops=1)
    -> Nested loop inner join  (cost=967.84 rows=259) (actual time=3.541..13.250 rows=2036 loops=1)
        -> Nested loop inner join  (cost=708.84 rows=950) (actual time=3.492..6.062 rows=1457 loops=1)
            -> Table scan on <subquery2>  (cost=599.28..613.62 rows=950) (actual time=3.440..3.699 rows=1457 loops=1)
                -> Materialize with deduplication  (cost=599.26..599.26 rows=950) (actual time=3.430..3.430 rows=1457 loops=1)
                    -> Filter: ((c.HighBP = 1) or (c.HeartDisease = 1))  (cost=504.30 rows=950) (actual time=0.208..2.251 rows=2627 loops=1)
                        -> Table scan on c  (cost=504.30 rows=4998) (actual time=0.193..1.721 rows=4998 loops=1)
            -> Single-row index lookup on u using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=0.35 rows=1) (actual time=0.001..0.001 rows=1 loops=1457)
        -> Filter: (l.Smoker = 0)  (cost=0.53 rows=0.3) (actual time=0.004..0.005 rows=1 loops=1457)
            -> Index lookup on l using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=0.53 rows=3) (actual time=0.003..0.004 rows=3 loops=1457)
    -> Index lookup on b using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=1.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2036)
 |
```

Create index on LIFESTYLEINFO Smoker attribute (total cost 1355.24):



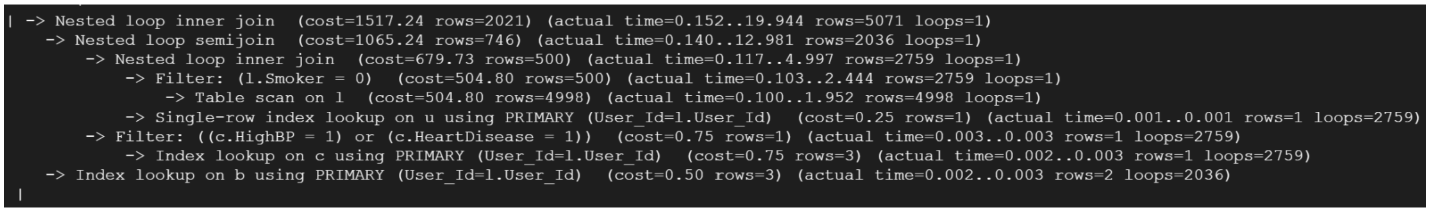
```
| -> Nested loop inner join  (cost=1355.24 rows=3866) (actual time=4.016..18.433 rows=5071 loops=1)
    -> Nested loop inner join  (cost=967.84 rows=1427) (actual time=3.222..11.062 rows=2036 loops=1)
        -> Nested loop inner join  (cost=708.84 rows=950) (actual time=3.194..5.226 rows=1457 loops=1)
            -> Table scan on <subquery2>  (cost=599.28..613.62 rows=950) (actual time=3.168..3.390 rows=1457 loops=1)
                -> Materialize with deduplication  (cost=599.26..599.26 rows=950) (actual time=3.163..3.163 rows=1457 loops=1)
                    -> Filter: ((c.HighBP = 1) or (c.HeartDisease = 1))  (cost=504.30 rows=950) (actual time=0.062..2.109 rows=2627 loops=1)
                        -> Table scan on c  (cost=504.30 rows=4998) (actual time=0.055..1.593 rows=4998 loops=1)
            -> Single-row index lookup on u using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=0.35 rows=1) (actual time=0.001..0.001 rows=1 loops=1457)
        -> Filter: (l.Smoker = 0)  (cost=0.65 rows=2) (actual time=0.003..0.004 rows=1 loops=1457)
            -> Index lookup on l using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=0.65 rows=3) (actual time=0.003..0.003 rows=3 loops=1457)
    -> Index lookup on b using PRIMARY (User_Id=`<subquery2>`.User_Id)  (cost=0.68 rows=3) (actual time=0.002..0.003 rows=2 loops=2036)
 |
```

Create index on CONDITIONSINFO HighBP attribute (total cost 1517.24):



```
| -> Nested loop inner join  (cost=1517.24 rows=2021) (actual time=0.217..20.284 rows=5071 loops=1)
  -> Nested loop semijoin  (cost=1065.24 rows=746) (actual time=0.201..13.469 rows=2036 loops=1)
      -> Nested loop inner join  (cost=679.73 rows=500) (actual time=0.142..4.791 rows=2759 loops=1)
          -> Filter: (l.Smoker = 0)  (cost=504.80 rows=500) (actual time=0.125..2.266 rows=2759 loops=1)
              -> Table scan on l  (cost=504.80 rows=4998) (actual time=0.120..1.826 rows=4998 loops=1)
          -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
      -> Filter: ((c.HighBP = 1) or (c.HeartDisease = 1))  (cost=0.75 rows=1) (actual time=0.003..0.003 rows=1 loops=2759)
          -> Index lookup on c using PRIMARY (User_Id=l.User_Id)  (cost=0.75 rows=3) (actual time=0.003..0.003 rows=1 loops=2759)
  -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2036)
|
```

Create index on CONDITIONSINFO HighBP and HeartDisease attributes (total cost 1517.24):



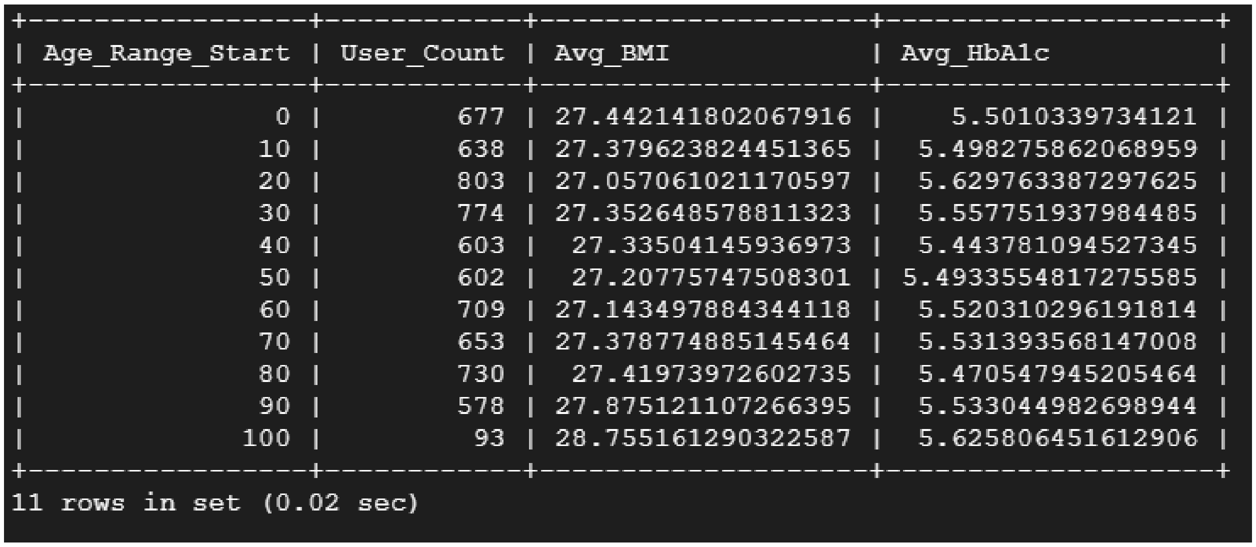
```
| -> Nested loop inner join  (cost=1517.24 rows=2021) (actual time=0.152..19.944 rows=5071 loops=1)
  -> Nested loop semijoin  (cost=1065.24 rows=746) (actual time=0.140..12.981 rows=2036 loops=1)
      -> Nested loop inner join  (cost=679.73 rows=500) (actual time=0.117..4.997 rows=2759 loops=1)
          -> Filter: (l.Smoker = 0)  (cost=504.80 rows=500) (actual time=0.103..2.444 rows=2759 loops=1)
              -> Table scan on l  (cost=504.80 rows=4998) (actual time=0.100..1.952 rows=4998 loops=1)
          -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
      -> Filter: ((c.HighBP = 1) or (c.HeartDisease = 1))  (cost=0.75 rows=1) (actual time=0.003..0.003 rows=1 loops=2759)
          -> Index lookup on c using PRIMARY (User_Id=l.User_Id)  (cost=0.75 rows=3) (actual time=0.002..0.003 rows=1 loops=2759)
  -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2036)
 |
```

Explanation: We want to avoid indexing on primary keys so we don't index on User_Id. We experiment with indexing on the three attributes that show up in the WHERE clause of our query, Smoker from LIFESTYLEINFO and HighBP and HeartDisease from CONDITIONSINFO. Doing so actually increased our final cost for the query. Of these, adding an index on HighBP and HeartDisease had the highest increase to query cost at 1517.24. Indexing Smoker leads to an increased final cost as well at 1355.24. We see no meaningful performance increases in time that can't be attributed to chance, so we settle on the default indexing for this query. The reason for the higher costs with indexing may be related to all of these attributes being binary in nature, which can lead to unique challenges and overhead.

Query 2:

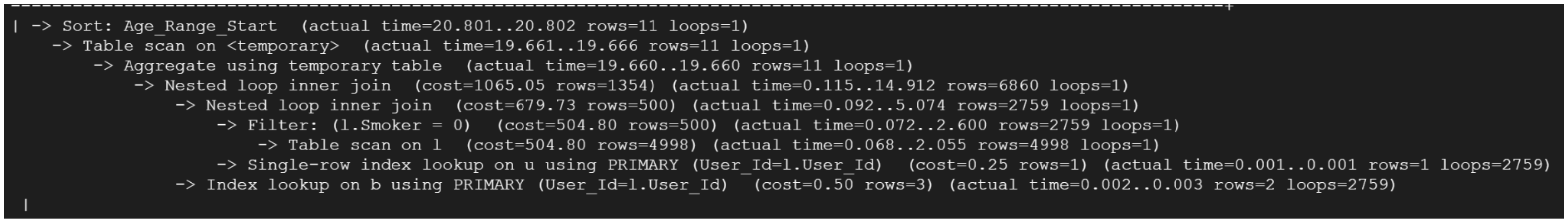
```
SELECT
  FLOOR(u.Age / 10) * 10 AS Age_Range_Start,
  COUNT(u.User_Id) AS User_Count,
  AVG(b.BMI) AS Avg_BMI,
  AVG(b.HbA1c) AS Avg_HbA1c
FROM USERINFO u
JOIN BIOMETRICSINFO b ON u.User_Id = b.User_Id
JOIN LIFESTYLEINFO l ON u.User_Id = l.User_Id
WHERE l.Smoker= 0
GROUP BY Age_Range_Start
ORDER BY Age_Range_Start
```

Output (Less than 15 rows):



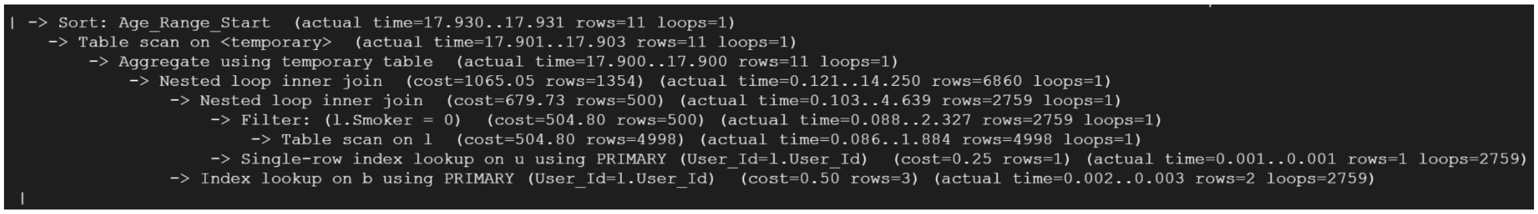
```
+-----------------+------------+--------------------+--------------------+
| Age_Range_Start | User_Count | Avg_BMI            | Avg_HbA1c          |
+-----------------+------------+--------------------+--------------------+
|               0 |        677 | 27.442141802067916 |    5.5010339734121 |
|              10 |        638 | 27.379623824451365 |  5.498275862068959 |
|              20 |        803 | 27.057061021170597 |  5.629763387297625 |
|              30 |        774 | 27.352648578811323 |  5.557751937984485 |
|              40 |        603 |  27.33504145936973 |  5.443781094527345 |
|              50 |        602 |  27.20775747508301 | 5.4933554817275585 |
|              60 |        709 | 27.143497884344118 |  5.520310296191814 |
|              70 |        653 | 27.378774885145464 |  5.531393568147008 |
|              80 |        730 |  27.41973972602735 |  5.470547945205464 |
|              90 |        578 | 27.875121107266395 |  5.533044982698944 |
|             100 |         93 | 28.755161290322587 |  5.625806451612906 |
+-----------------+------------+--------------------+--------------------+
11 rows in set (0.02 sec)
```

Default explain analyze (total cost 1065.05):



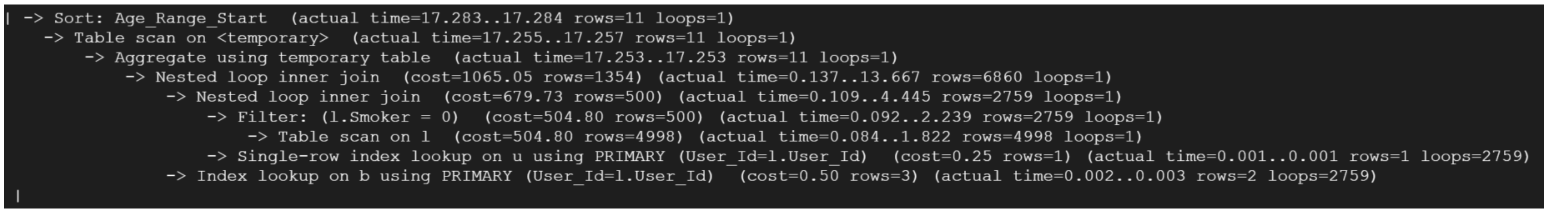
```
| -> Sort: Age_Range_Start  (actual time=20.801..20.802 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=19.661..19.666 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=19.660..19.660 rows=11 loops=1)
            -> Nested loop inner join  (cost=1065.05 rows=1354) (actual time=0.115..14.912 rows=6860 loops=1)
                -> Nested loop inner join  (cost=679.73 rows=500) (actual time=0.092..5.074 rows=2759 loops=1)
                    -> Filter: (l.Smoker = 0)  (cost=504.80 rows=500) (actual time=0.072..2.600 rows=2759 loops=1)
                        -> Table scan on l  (cost=504.80 rows=4998) (actual time=0.068..2.055 rows=4998 loops=1)
                    -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
                -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2759)
 |
```

Create index on BIOMETRICSINFO BMI (total cost 1065.05):



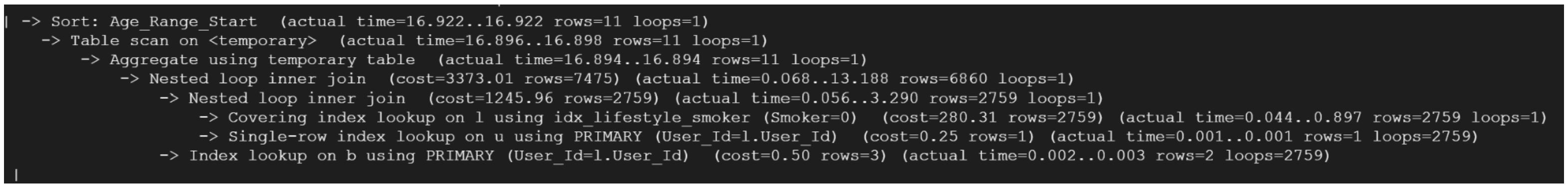
```
| -> Sort: Age_Range_Start  (actual time=17.930..17.931 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=17.901..17.903 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=17.900..17.900 rows=11 loops=1)
            -> Nested loop inner join  (cost=1065.05 rows=1354) (actual time=0.121..14.250 rows=6860 loops=1)
                -> Nested loop inner join  (cost=679.73 rows=500) (actual time=0.103..4.639 rows=2759 loops=1)
                    -> Filter: (l.Smoker = 0)  (cost=504.80 rows=500) (actual time=0.088..2.327 rows=2759 loops=1)
                        -> Table scan on l  (cost=504.80 rows=4998) (actual time=0.086..1.884 rows=4998 loops=1)
                    -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
                -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2759)
 |
```

Create index on BIOMETRICSINFO HbA1c (total cost 1065.05):



```
| -> Sort: Age_Range_Start  (actual time=17.283..17.284 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=17.255..17.257 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=17.253..17.253 rows=11 loops=1)
            -> Nested loop inner join  (cost=1065.05 rows=1354) (actual time=0.137..13.667 rows=6860 loops=1)
                -> Nested loop inner join  (cost=679.73 rows=500) (actual time=0.109..4.445 rows=2759 loops=1)
                    -> Filter: (l.Smoker = 0)  (cost=504.80 rows=500) (actual time=0.092..2.239 rows=2759 loops=1)
                        -> Table scan on l  (cost=504.80 rows=4998) (actual time=0.084..1.822 rows=4998 loops=1)
                    -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
                -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2759)
 |
```

Create index on LIFESTYLEINFO Smoker (total cost 3373.01)



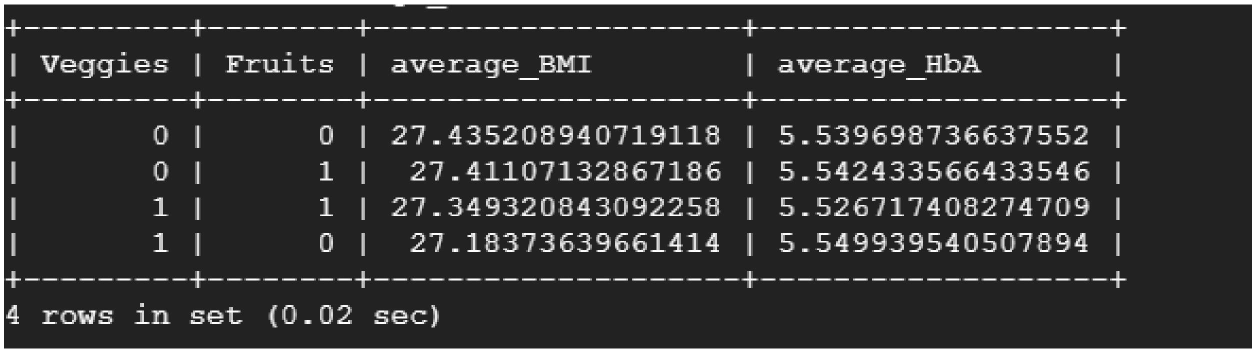
```
| -> Sort: Age_Range_Start  (actual time=16.922..16.922 rows=11 loops=1)
    -> Table scan on <temporary>  (actual time=16.896..16.898 rows=11 loops=1)
        -> Aggregate using temporary table  (actual time=16.894..16.894 rows=11 loops=1)
            -> Nested loop inner join  (cost=3373.01 rows=7475) (actual time=0.068..13.188 rows=6860 loops=1)
                -> Nested loop inner join  (cost=1245.96 rows=2759) (actual time=0.056..3.290 rows=2759 loops=1)
                    -> Covering index lookup on l using idx_lifestyle_smoker (Smoker=0)  (cost=280.31 rows=2759) (actual time=0.044..0.897 rows=2759 loops=1)
                    -> Single-row index lookup on u using PRIMARY (User_Id=l.User_Id)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=2759)
                -> Index lookup on b using PRIMARY (User_Id=l.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=2759)
 |
```

Explanation: We want to avoid indexing on primary keys so we don't index on User_Id. We experiment with indexing on the three attributes Smoker, HbA1c and BMI. Of these, adding an index on HighBP and HbA1c both led to no changes in cost, likely because they weren't present in meaningful joins or conditions in the query. Indexing Smoker leads to an increased final cost at 3373.01. We see no meaningful performance increases in time that can't be attributed to chance, <u>so we settle on the default indexing for this query</u>. The reason for the higher costs with indexing in Smoker may be related to this attribute being binary in nature, which can lead to unique challenges and overhead.

Query 3:

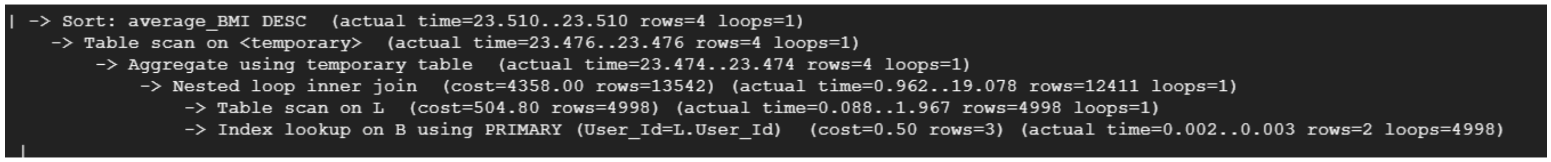
SELECT L.Veggies, L.Fruits, AVG(B.BMI) AS average_BMI, AVG(B.HbA1c) AS average_HbA
FROM BIOMETRICSINFO B
JOIN LIFESTYLEINFO L ON B.User_Id = L.User_Id
GROUP BY L.Veggies, L.Fruits
ORDER BY average_BMI DESC;

Output (less than 15 rows):



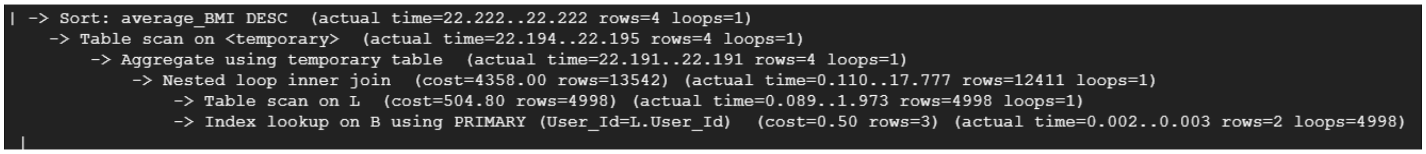
```
+---------+--------+--------------------+-------------------+
| Veggies | Fruits | average_BMI        | average_HbA       |
+---------+--------+--------------------+-------------------+
|       0 |      0 | 27.435208940719118 | 5.539698736637552 |
|       0 |      1 |  27.41107132867186 | 5.542433566433546 |
|       1 |      1 | 27.349320843092258 | 5.526717408274709 |
|       1 |      0 |  27.18373639661414 | 5.549939540507894 |
+---------+--------+--------------------+-------------------+
4 rows in set (0.02 sec)
```

Default explain analyze (total cost 4358.00):



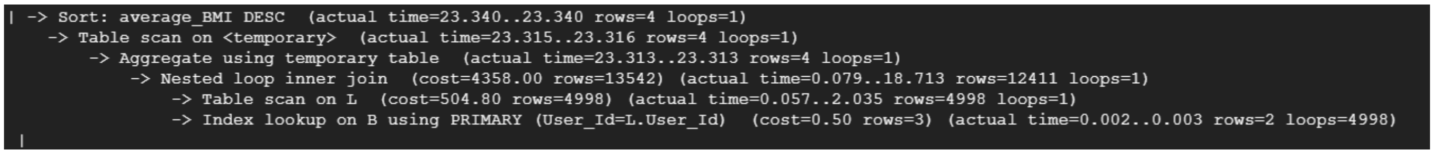
```
| -> Sort: average_BMI DESC  (actual time=23.510..23.510 rows=4 loops=1)
    -> Table scan on <temporary>  (actual time=23.476..23.476 rows=4 loops=1)
        -> Aggregate using temporary table  (actual time=23.474..23.474 rows=4 loops=1)
            -> Nested loop inner join  (cost=4358.00 rows=13542) (actual time=0.962..19.078 rows=12411 loops=1)
                -> Table scan on L  (cost=504.80 rows=4998) (actual time=0.088..1.967 rows=4998 loops=1)
                -> Index lookup on B using PRIMARY (User_Id=L.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=4998)
 |
```

Create index on LIFESTYLEINFO Veggies (total cost 4358.00):



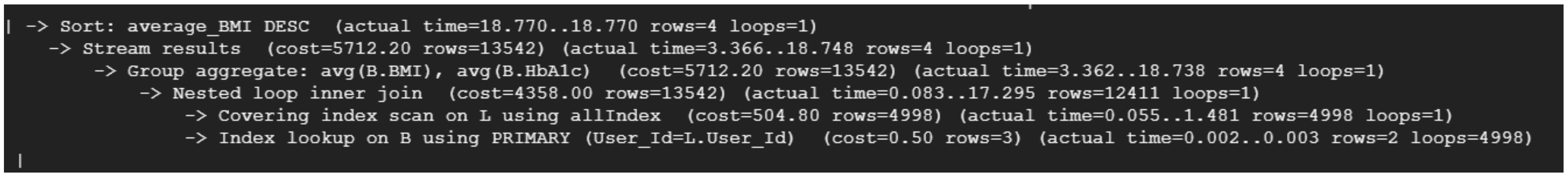
```
| -> Sort: average_BMI DESC  (actual time=22.222..22.222 rows=4 loops=1)
    -> Table scan on <temporary>  (actual time=22.194..22.195 rows=4 loops=1)
        -> Aggregate using temporary table  (actual time=22.191..22.191 rows=4 loops=1)
            -> Nested loop inner join  (cost=4358.00 rows=13542) (actual time=0.110..17.777 rows=12411 loops=1)
                -> Table scan on L  (cost=504.80 rows=4998) (actual time=0.089..1.973 rows=4998 loops=1)
                -> Index lookup on B using PRIMARY (User_Id=L.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=4998)
 |
```

Create index on LIFESTYLEINFO Fruits (total cost 4358.00):



```
| -> Sort: average_BMI DESC  (actual time=23.340..23.340 rows=4 loops=1)
    -> Table scan on <temporary>  (actual time=23.315..23.316 rows=4 loops=1)
        -> Aggregate using temporary table  (actual time=23.313..23.313 rows=4 loops=1)
            -> Nested loop inner join  (cost=4358.00 rows=13542) (actual time=0.079..18.713 rows=12411 loops=1)
                -> Table scan on L  (cost=504.80 rows=4998) (actual time=0.057..2.035 rows=4998 loops=1)
                -> Index lookup on B using PRIMARY (User_Id=L.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=4998)
 |
```

Create index on LIFESTYLEINFO (Veggies, Fruits) (total cost (5712.20):



```
| -> Sort: average_BMI DESC  (actual time=18.770..18.770 rows=4 loops=1)
    -> Stream results  (cost=5712.20 rows=13542) (actual time=3.366..18.748 rows=4 loops=1)
        -> Group aggregate: avg(B.BMI), avg(B.HbA1c)  (cost=5712.20 rows=13542) (actual time=3.362..18.738 rows=4 loops=1)
            -> Nested loop inner join  (cost=4358.00 rows=13542) (actual time=0.083..17.295 rows=12411 loops=1)
                -> Covering index scan on L using allIndex  (cost=504.80 rows=4998) (actual time=0.055..1.481 rows=4998 loops=1)
                -> Index lookup on B using PRIMARY (User_Id=L.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=2 loops=4998)
 |
```

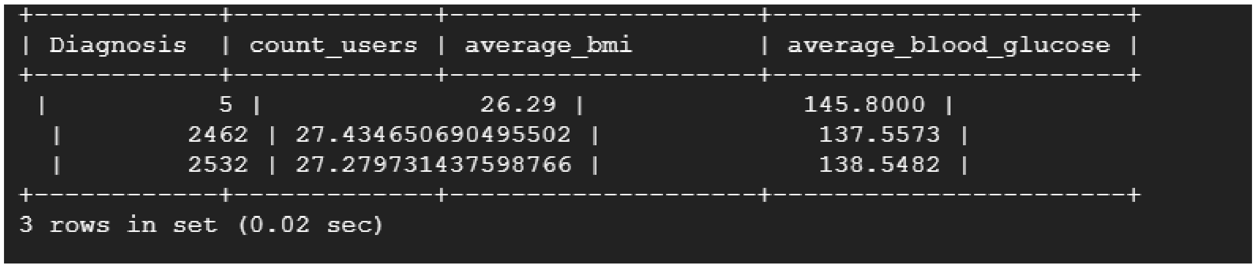
Explanation: We want to avoid indexing on primary keys so we don't index on User_Id. We experiment with indexing on the two attributes Veggies and Fruits from the table

LIFESTYLEINFO since they are involved in GROUP BY clause in our query. Adding indexes on these attributes individually doesn't change query costs, which stays the same at 4358.00. Adding an index on both at the same time increases query cost to 5712.20. The reason for the higher costs with indexing in this case may be related to these attribute being binary in nature. <u>We settle with the default indexing to avoid incurring unnecessary overhead for no cost improvement.</u>

Query 4:

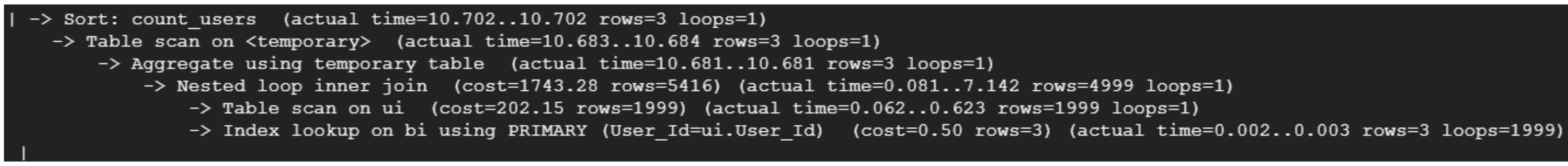
SELECT ui.Diagnosis, COUNT(*) AS count_users, AVG(bi.BMI) AS average_bmi,
AVG(bi.BloodGlucose) AS average_blood_glucose
FROM USERINFO ui
JOIN BIOMETRICSINFO bi ON bi.User_Id = ui.User_Id
GROUP BY ui.Diagnosis
ORDER BY count_users ASC;

Output (less than 15 rows):



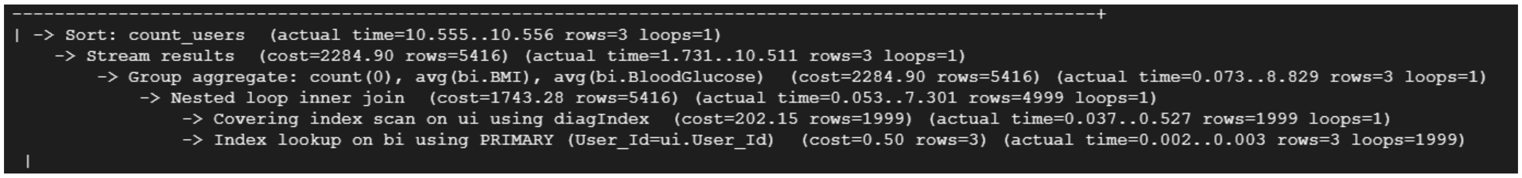
```
+------------+-------------+--------------------+-----------------------+
| Diagnosis  | count_users | average_bmi        | average_blood_glucose |
+------------+-------------+--------------------+-----------------------+
 |          5 |             26.29 |             145.8000 |
  |       2462 | 27.434650690495502 |             137.5573 |
  |       2532 | 27.279731437598766 |             138.5482 |
+------------+-------------+--------------------+-----------------------+
3 rows in set (0.02 sec)
```

Default explain analyze (total cost 1743.28):



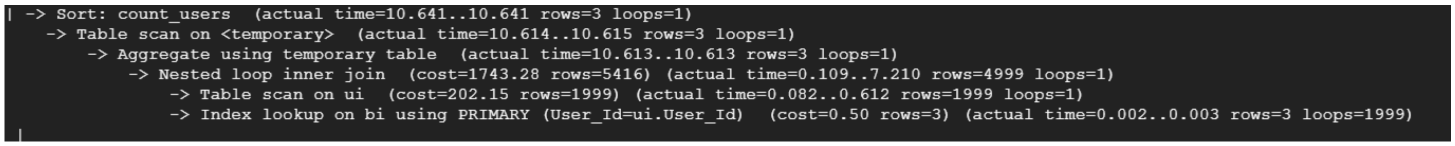
```
| -> Sort: count_users  (actual time=10.702..10.702 rows=3 loops=1)
    -> Table scan on <temporary>  (actual time=10.683..10.684 rows=3 loops=1)
        -> Aggregate using temporary table  (actual time=10.681..10.681 rows=3 loops=1)
            -> Nested loop inner join  (cost=1743.28 rows=5416) (actual time=0.081..7.142 rows=4999 loops=1)
                -> Table scan on ui  (cost=202.15 rows=1999) (actual time=0.062..0.623 rows=1999 loops=1)
                -> Index lookup on bi using PRIMARY (User_Id=ui.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=3 loops=1999)
 |
```

Create index on USERINFO Diagnosis (total cost 2284):



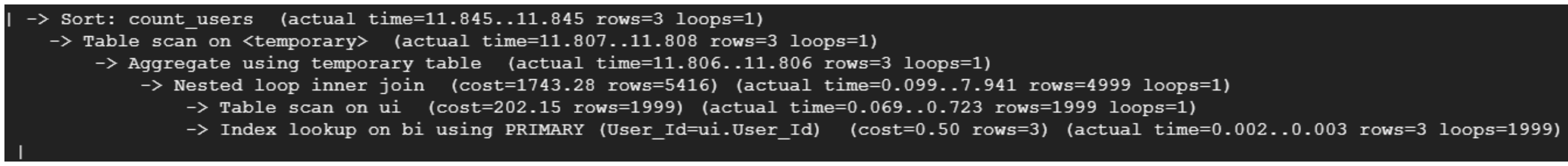
```
------------------------------------------------------------------------------------------------------+
| -> Sort: count_users  (actual time=10.555..10.556 rows=3 loops=1)
    -> Stream results  (cost=2284.90 rows=5416) (actual time=1.731..10.511 rows=3 loops=1)
        -> Group aggregate: count(0), avg(bi.BMI), avg(bi.BloodGlucose)  (cost=2284.90 rows=5416) (actual time=0.073..8.829 rows=3 loops=1)
            -> Nested loop inner join  (cost=1743.28 rows=5416) (actual time=0.053..7.301 rows=4999 loops=1)
                -> Covering index scan on ui using diagIndex  (cost=202.15 rows=1999) (actual time=0.037..0.527 rows=1999 loops=1)
                -> Index lookup on bi using PRIMARY (User_Id=ui.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=3 loops=1999)
 |
```

Create index on BIOMETRICSINFO BloodGlucose (total cost 1743.28):



```
| -> Sort: count_users  (actual time=10.641..10.641 rows=3 loops=1)
    -> Table scan on <temporary>  (actual time=10.614..10.615 rows=3 loops=1)
        -> Aggregate using temporary table  (actual time=10.613..10.613 rows=3 loops=1)
            -> Nested loop inner join  (cost=1743.28 rows=5416) (actual time=0.109..7.210 rows=4999 loops=1)
                -> Table scan on ui  (cost=202.15 rows=1999) (actual time=0.082..0.612 rows=1999 loops=1)
                -> Index lookup on bi using PRIMARY (User_Id=ui.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=3 loops=1999)
 |
```

Create index on BIOMETRICSINFO BMI (total cost 1743.28):



```
| -> Sort: count_users  (actual time=11.845..11.845 rows=3 loops=1)
    -> Table scan on <temporary>  (actual time=11.807..11.808 rows=3 loops=1)
        -> Aggregate using temporary table  (actual time=11.806..11.806 rows=3 loops=1)
            -> Nested loop inner join  (cost=1743.28 rows=5416) (actual time=0.099..7.941 rows=4999 loops=1)
                -> Table scan on ui  (cost=202.15 rows=1999) (actual time=0.069..0.723 rows=1999 loops=1)
                -> Index lookup on bi using PRIMARY (User_Id=ui.User_Id)  (cost=0.50 rows=3) (actual time=0.002..0.003 rows=3 loops=1999)
 |
```

Explanation: We want to avoid indexing on primary keys so we don't index on User_Id. We experiment with indexing on the three attributes Diagnosis BloodGlucose and BMI, since they are involved in GROUP BY clause in our query and are return by our query. Adding an index on Diagnosis increases the query cost to 2284 while indexing on the other attributes doesn't change query costs. Because there are no meaningful cost improvements with these indexing plans, <u>we settle with the default indexing to avoid incurring unnecessary overhead for no cost improvement.</u>