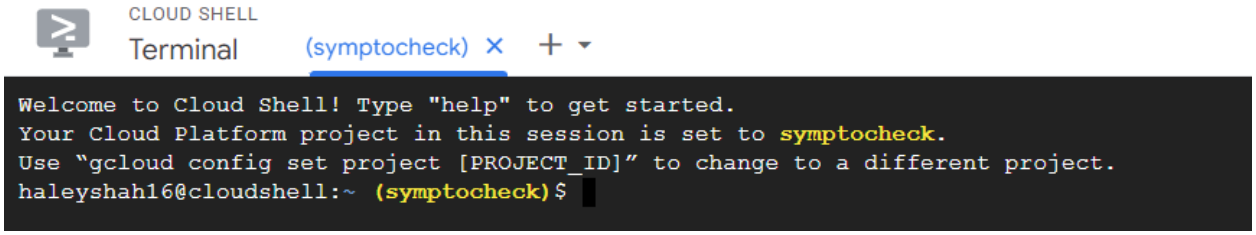
GCP Connection:



DDL Commands:

```
CREATE TABLE Records(
        user_id VARCHAR(255),
        age INT,
        sex VARCHAR(10),
        country VARCHAR(10),
        checkin_date DATETIME,
        trackable_id VARCHAR(255),
        trackable_type VARCHAR(255),
        trackable_name VARCHAR(255),
        trackable_value INT
);

CREATE TABLE Symptoms (
        SymptomId INT AUTO_INCREMENT PRIMARY KEY,
        SymptomName VARCHAR(255) NOT NULL,
        Severity INT
);

CREATE TABLE Treatment (
        TreatmentId INT AUTO_INCREMENT PRIMARY KEY,
        Medicine VARCHAR(255) NOT NULL,
        Dosage VARCHAR(30)
);

CREATE TABLE Illness(
        IllnessId INT AUTO_INCREMENT PRIMARY KEY,
        IllnessName VARCHAR(255) NOT NULL
);

CREATE TABLE Has_Symptom(
        Illness_Symptom_Id INT AUTO_INCREMENT PRIMARY KEY,
        IllnessName VARCHAR(255),
        SymptomName VARCHAR(255),
        Severity INT
);
```

```
CREATE TABLE Has_Treatment(
        Illness_Treatment_Id INT AUTO_INCREMENT PRIMARY KEY,
        IllnessName VARCHAR(255),
        Medicine VARCHAR(255) NOT NULL,
        Dosage VARCHAR(30)
);
```

Insert Into Tables:

```
mysql> SELECT COUNT(user_id) FROM Records;
+----------------+
| COUNT(user_id) |
+----------------+
|        7976223 |
+----------------+
1 row in set (18.99 sec)
```

```
mysql> SELECT COUNT(SymptomId) FROM Symptoms;
+------------------+
| COUNT(SymptomId) |
+------------------+
|            63142 |
+------------------+
1 row in set (0.33 sec)
```

```
mysql> SELECT COUNT(TreatmentId) FROM Treatment;
+--------------------+
| COUNT(TreatmentId) |
+--------------------+
|              19557 |
+--------------------+
1 row in set (0.24 sec)
```

```
mysql> SELECT COUNT(IllnessId) FROM Illness;
+------------------+
| COUNT(IllnessId) |
+------------------+
|             9203 |
+------------------+
1 row in set (0.22 sec)
```

```
mysql> SELECT COUNT(Illness_Symptom_Id) FROM Has_Symptom;
+---------------------------+
| COUNT(Illness_Symptom_Id) |
+---------------------------+
|                  14468863 |
+---------------------------+
1 row in set (2.38 sec)
```

```
mysql> SELECT COUNT(Illness_Treatment_Id) FROM Has_Treatment;
+----------------------------+
| COUNT(Illness_Treatment_Id) |
+----------------------------+
|                     297965 |
+----------------------------+
1 row in set (0.44 sec)
```

Advanced Queries:

1. **Query to get the top conditions/illnesses in which anxiety is a symptom.**

```
SELECT r2.trackable_name,COUNT(r2.user_id) AS
`count_user_id`
FROM Records r1 JOIN Records r2 ON r1.user_id = r2.user_id
WHERE r1.trackable_type = 'Symptom' AND r1.trackable_name =
'Anxiety' AND r2.trackable_type ='Condition' AND
r1.checkin_date = r2.checkin_date
GROUP BY r2.trackable_name
ORDER BY `count_user_id` DESC
LIMIT 15;
```

trackable_name	count_user_id
Anxiety	12982
Fibromyalgia	12650
Depression	11673
Chronic fatigue syndrome	6651
Migraine	5921
IBS	3917
Generalized anxiety disorder	3828
Ehlers-Danlos syndrome	3170
Irritable bowel syndrome	3159
Lyme disease	3120
POTS	2835
Headaches	2663
Asthma	2509
Fatigue	2380
Allergies	2314

15 rows in set (56.83 sec)

EXPLAIN ANALYZE for Original Result:

```
| -> Limit: 15 row(s)  (actual time=56052.569..56052.572 rows=15 loops=1)
-> Sort: count_user_id DESC, limit input to 15 row(s) per chunk  (actual time=56052.567..56052.569 rows=15 loops=1)
      -> Table scan on <temporary>  (actual time=0.003..0.669 rows=2252 loops=1)
            -> Aggregate using temporary table  (actual time=56051.460..56052.282 rows=2252 loops=1)
                  -> Inner hash join (r2.user_id = r1.user_id), (r2.checkin_date = r1.checkin_date)  (cost=598843837.71 rows=619634)
(actual time=26884.533..55823.834 rows=224441 loops=1)
                        -> Filter: (r2.trackable_type = 'Condition')  (cost=6809.60 rows=7872) (actual time=0.082..26251.959 rows=1111517
loops=1)
                              -> Table scan on r2  (cost=6809.60 rows=7871685) (actual time=0.076..25205.263 rows=7976223 loops=1)
                                    -> Hash
                                          -> Filter: ((r1.trackable_type = 'Symptom') and (r1.trackable_name = 'Anxiety'))
(cost=851039.50 rows=78717) (actual time=0.115..26714.584 rows=61602 loops=1)
                                                -> Table scan on r1  (cost=851039.50 rows=7871685) (actual time=0.084..25332.205
rows=7976223 loops=1)
|
```

Indexing #1 with trackable_id:

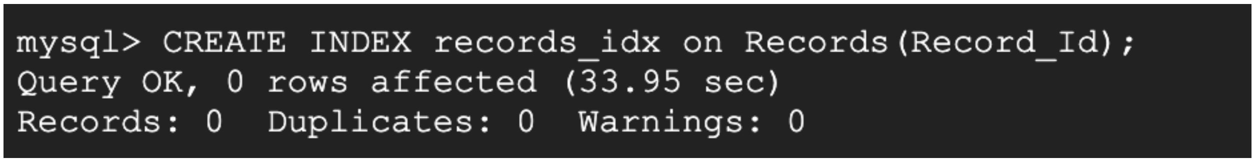
Trackable id is in the where clause and indexing this should give us a slight reduction on runtime as seen below.

```
mysql> CREATE INDEX trackable_id_idx on Records(trackable_id(10));
Query OK, 0 rows affected (49.95 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

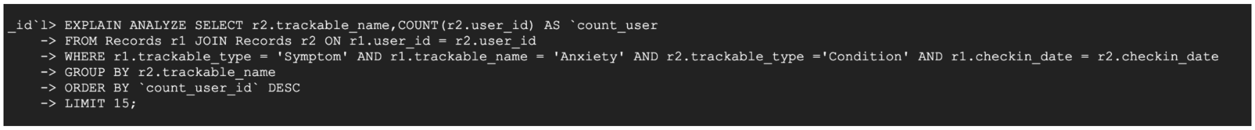
```
mysql> EXPLAIN ANALYZE SELECT r2.trackable_name,COUNT(r2.user_id) AS `count_user_id` FROM Records r1 JOIN Records r2 ON r1.user_id = r
2.user_id WHERE r1.trackable_type = 'Symptom' AND r1.trackable_name = 'Anxiety' AND r2.trackable_type ='Condition' AND r1.checkin_date
 = r2.checkin_date GROUP BY r2.trackable_name ORDER BY `count_user_id` DESC
LIMIT 15;
| EXPLAIN
| -> Limit: 15 row(s)  (actual time=48282.251..48282.253 rows=15 loops=1)
    -> Sort: count_user_id DESC, limit input to 15 row(s) per chunk  (actual time=48282.250..48282.251 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.002..0.534 rows=2252 loops=1)
            -> Aggregate using temporary table  (actual time=48281.346..48282.006 rows=2252 loops=1)
                -> Inner hash join (r2.user_id = r1.user_id), (r2.checkin_date = r1.checkin_date)  (cost=598843837.71 rows=619634) (ac
tual time=22953.869..48063.539 rows=224441 loops=1)
                    -> Filter: (r2.trackable_type = 'Condition')  (cost=6809.60 rows=7872) (actual time=0.075..22666.757 rows=1111517
loops=1)
                        -> Table scan on r2  (cost=6809.60 rows=7871685) (actual time=0.069..21691.393 rows=7976223 loops=1)
                    -> Hash
                        -> Filter: ((r1.trackable_type = 'Symptom') and (r1.trackable_name = 'Anxiety'))  (cost=851039.50 rows=78717)
(actual time=0.117..22825.209 rows=61602 loops=1)
                            -> Table scan on r1  (cost=851039.50 rows=7871685) (actual time=0.098..21531.255 rows=7976223 loops=1)
 |
1 row in set (48.31 sec)
```

Indexing #2 with Record_Id:

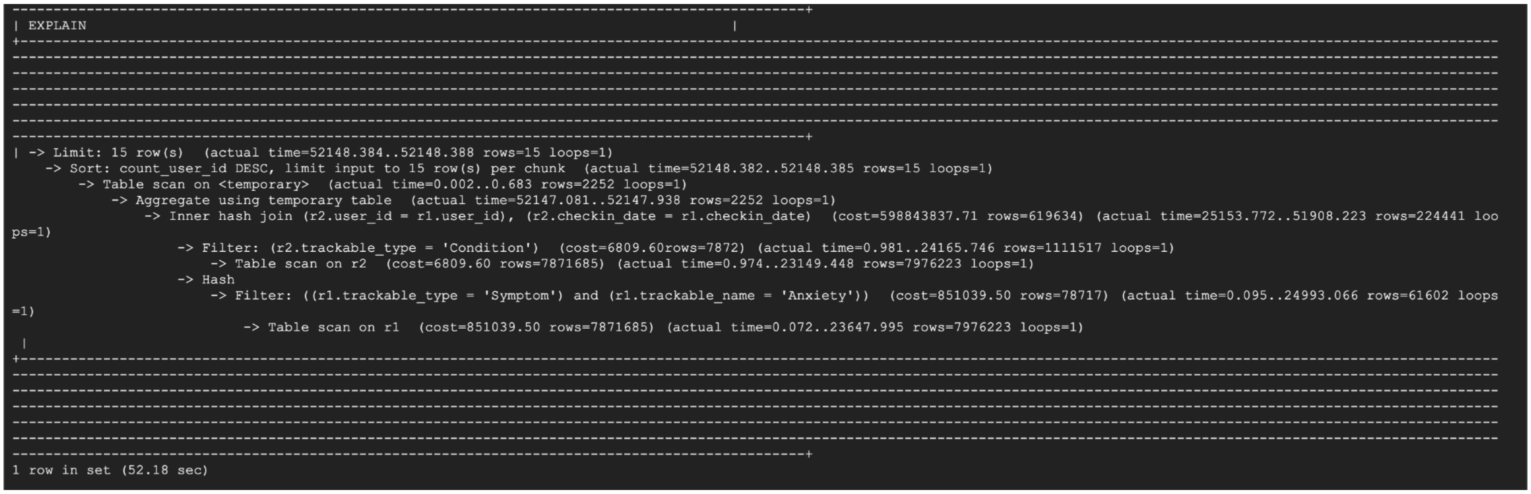
Record id isn’t really seen in this query, so we should not expect a significant reduction in runtime, as seen below.



```
mysql> CREATE INDEX records_idx on Records(Record_Id);
Query OK, 0 rows affected (33.95 sec)
Records: 0  Duplicates: 0  Warnings: 0
```



```
_id`l> EXPLAIN ANALYZE SELECT r2.trackable_name,COUNT(r2.user_id) AS `count_user
    -> FROM Records r1 JOIN Records r2 ON r1.user_id = r2.user_id
    -> WHERE r1.trackable_type = 'Symptom' AND r1.trackable_name = 'Anxiety' AND r2.trackable_type ='Condition' AND r1.checkin_date = r2.checkin_date
    -> GROUP BY r2.trackable_name
    -> ORDER BY `count_user_id` DESC
    -> LIMIT 15;
```



```
-------------------------------------------------------------------------------------------------+
| EXPLAIN                                                                              |
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=52148.384..52148.388 rows=15 loops=1)
    -> Sort: count_user_id DESC, limit input to 15 row(s) per chunk  (actual time=52148.382..52148.385 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.002..0.683 rows=2252 loops=1)
            -> Aggregate using temporary table  (actual time=52147.081..52147.938 rows=2252 loops=1)
                -> Inner hash join (r2.user_id = r1.user_id), (r2.checkin_date = r1.checkin_date)  (cost=598843837.71 rows=619634) (actual time=25153.772..51908.223 rows=224441 loo
ps=1)
                    -> Filter: (r2.trackable_type = 'Condition')  (cost=6809.60rows=7872) (actual time=0.981..24165.746 rows=1111517 loops=1)
                        -> Table scan on r2  (cost=6809.60 rows=7871685) (actual time=0.974..23149.448 rows=7976223 loops=1)
                    -> Hash
                        -> Filter: ((r1.trackable_type = 'Symptom') and (r1.trackable_name = 'Anxiety'))  (cost=851039.50 rows=78717) (actual time=0.095..24993.066 rows=61602 loops
=1)

                            -> Table scan on r1  (cost=851039.50 rows=7871685) (actual time=0.072..23647.995 rows=7976223 loops=1)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------+
1 row in set (52.18 sec)
```

Indexing #3 with trackable_name:

The where clause uses trackable_name from Records and we know that what’s in the where clause affects the runtime. By adding an index, we significantly reduce the time as seen in the EXPLAIN ANALYZE:

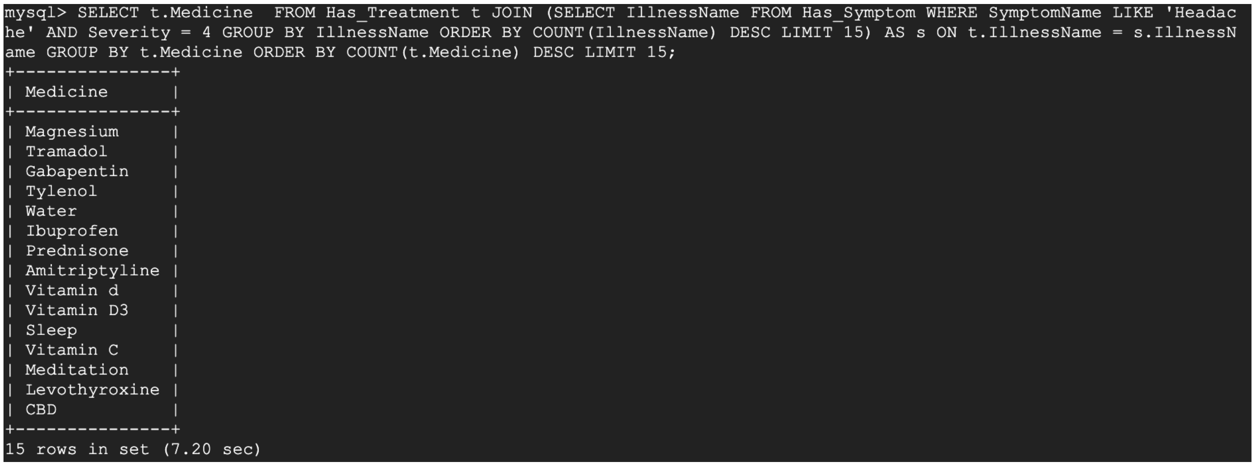
```
mysql> CREATE INDEX track_name_idx on Records(trackable_name(5));
Query OK, 0 rows affected (49.73 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> EXPLAIN ANALYZE SELECT r2.trackable_name,COUNT(r2.user_id) AS `count_user_id` FROM Records r1 JOI
N Records r2 ON r1.user_id = r2.user_id WHERE r1.trackable_type = 'Symptom' AND r1.trackable_name = 'Anx
iety' AND r2.trackable_type ='Condition' AND r1.checkin_date = r2.checkin_date GROUP BY r2.trackable_nam
e ORDER BY `count_user_id` DESC LIMIT 15;
+-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
```

```
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=35244.750..35244.752 rows=15 loops=1)
    -> Sort: count_user_id DESC, limit input to 15 row(s) per chunk  (actual time=35244.749..35244.750 r
ows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.003..0.515 rows=2252 loops=1)
            -> Aggregate using temporary table  (actual time=35243.851..35244.500 rows=2252 loops=1)
                -> Inner hash join (r2.user_id = r1.user_id), (r2.checkin_date = r1.checkin_date)  (cost
=176155208.42 rows=186734) (actual time=13328.789..35054.591 rows=224441 loops=1)
                    -> Filter: (r2.trackable_type = 'Condition')  (cost=6633.29 rows=7872) (actual time=
0.541..19139.657 rows=1111517 loops=1)
                        -> Table scan on r2  (cost=6633.29 rows=7871685) (actual time=0.537..18316.824 r
ows=7976223 loops=1)
                    -> Hash
                        -> Filter: ((r1.trackable_type = 'Symptom') and (r1.trackable_name = 'Anxiety'))
  (cost=126473.48 rows=23722) (actual time=146.759..13258.510 rows=61602 loops=1)
                            -> Index lookup on r1 using track_name_idx (trackable_name='Anxiety')  (cost
=126473.48 rows=237222) (actual time=146.750..13220.761 rows=114619 loops=1)
 |
+-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
1 row in set (35.37 sec)
```

2. Query to show the most frequent treatment method for some symptoms

```
SELECT t.Medicine  FROM Has_Treatment t JOIN (SELECT IllnessName FROM Has_Symptom
WHERE SymptomName LIKE 'Headache' AND Severity = 4 GROUP BY IllnessName ORDER BY
COUNT(IllnessName) DESC LIMIT 15) AS s ON t.IllnessName = s.IllnessName GROUP BY
t.Medicine ORDER BY COUNT(t.Medicine) DESC LIMIT 15;
```



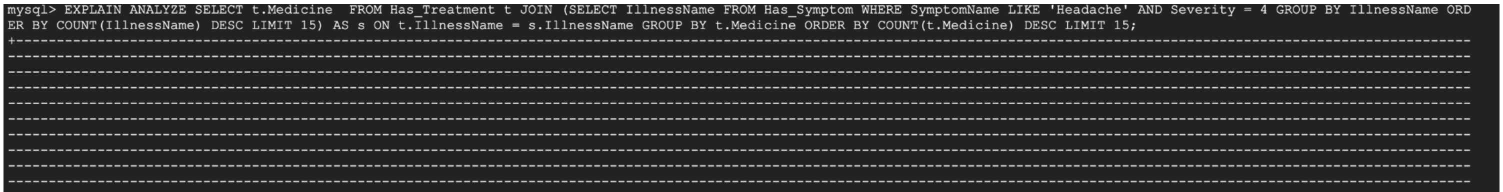
```
mysql> SELECT t.Medicine  FROM Has_Treatment t JOIN (SELECT IllnessName FROM Has_Symptom WHERE SymptomName LIKE 'Headac
he' AND Severity = 4 GROUP BY IllnessName ORDER BY COUNT(IllnessName) DESC LIMIT 15) AS s ON t.IllnessName = s.IllnessN
ame GROUP BY t.Medicine ORDER BY COUNT(t.Medicine) DESC LIMIT 15;
+---------------+
| Medicine      |
+---------------+
| Magnesium     |
| Tramadol      |
| Gabapentin    |
| Tylenol       |
| Water         |
| Ibuprofen     |
| Prednisone    |
| Amitriptyline |
| Vitamin d     |
| Vitamin D3    |
| Sleep         |
| Vitamin C     |
| Meditation    |
| Levothyroxine |
| CBD           |
+---------------+
15 rows in set (7.20 sec)
```

When we run explain analyze with no indices, we get a time of 7.077 seconds:

```
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=7077.643..7077.644 rows=15 loops=1)
    -> Sort: `count(t.Medicine)` DESC, limit input to 15 row(s) per chunk  (actual time=7077.642..7077.643 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.004..1.041 rows=5383 loops=1)
            -> Aggregate using temporary table  (actual time=7075.612..7077.109 rows=5383 loops=1)
                -> Nested loop inner join  (cost=172540.45 rows=0) (actual time=6749.949..7040.195 rows=47624 loops=1)
                    -> Filter: (t.IllnessName is not null)  (cost=28976.95 rows=287127) (actual time=0.057..109.132 rows=297965 loops=1)
                        -> Table scan on t  (cost=28976.95 rows=287127) (actual time=0.055..88.295 rows=297965 loops=1)
                    -> Index lookup on s using <auto_key0> (IllnessName=t.IllnessName)  (actual time=0.000..0.000 rows=0 loops=297965)
                        -> Materialize  (cost=0.00..0.00 rows=0) (actual time=5882.708..5889.982 rows=15 loops=1)
                            -> Limit: 15 row(s)  (actual time=5736.047..5736.049 rows=15 loops=1)
                                -> Sort: `count(Has_Symptom.IllnessName)` DESC, limit input to 15 row(s) per chunk  (actual
time=5736.046..5736.048 rows=15 loops=1)
                                    -> Table scan on <temporary>  (actual time=0.004..0.300 rows=1401 loops=1)
                                        -> Aggregate using temporary table  (actual time=5735.515..5735.892 rows=1401 loops=1)
                                            -> Filter: ((Has_Symptom.Severity = 4) and (Has_Symptom.SymptomName like 'Headache'))
(cost=1369135.60 rows=150643) (actual time=0.652..5693.328 rows=22525 loops=1)
                                                -> Table scan on Has_Symptom  (cost=1369135.60 rows=13559196) (actual time=0.042..4562.376
rows=14468863 loops=1)
|
+-------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------
```

Indexing analysis 1: Add index on Medicine (improved time: 5.91 sec)

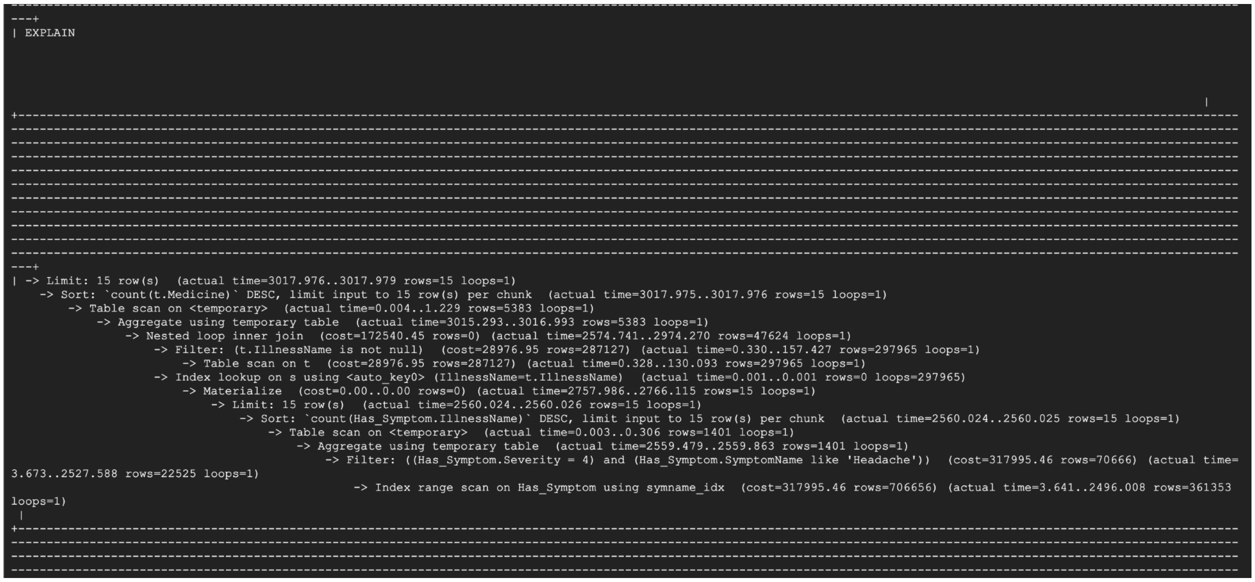
We will be showing users potential medicines more frequently, and relative to the number of entries (which is in the millions) medicine will be much smaller. Adding an index on Medicine should speed this up.



```
mysql> EXPLAIN ANALYZE SELECT t.Medicine  FROM Has_Treatment t JOIN (SELECT IllnessName FROM Has_Symptom WHERE SymptomName LIKE 'Headache' AND Severity = 4 GROUP BY IllnessName ORD
ER BY COUNT(IllnessName) DESC LIMIT 15) AS s ON t.IllnessName = s.IllnessName GROUP BY t.Medicine ORDER BY COUNT(t.Medicine) DESC LIMIT 15;
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Indexing analyis 2: Index on SymptomName (time: 3.017s)

The Where clause uses Symptom name in this advanced query and we can add an index here to reduce the time searching. This is proven true when we run EXPLAIN ANALYZE:



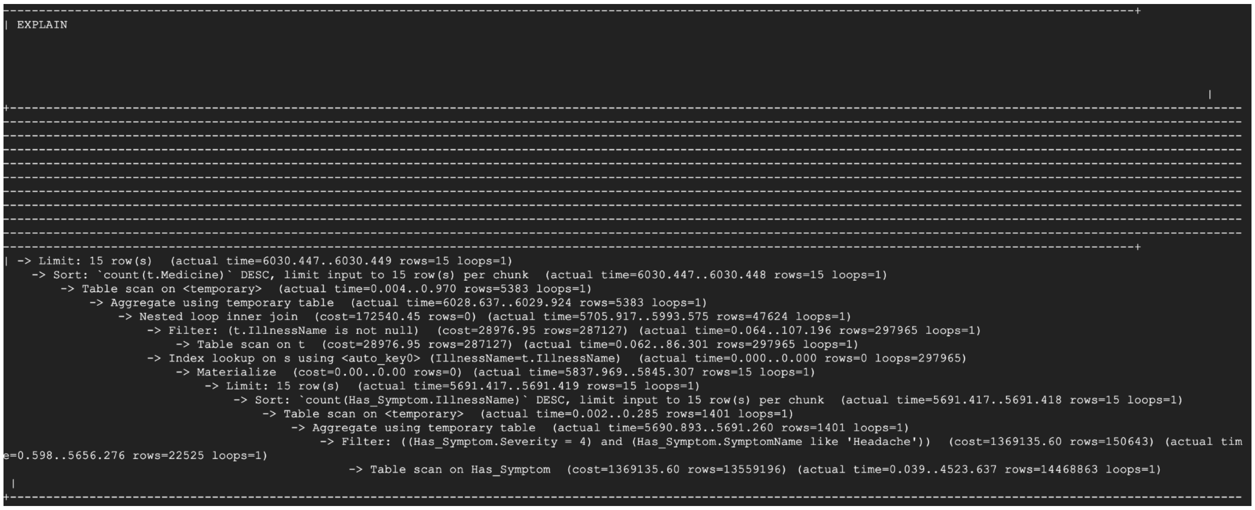
```
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---+
| EXPLAIN




                                                                                                                                                                      |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---+
| -> Limit: 15 row(s)  (actual time=3017.976..3017.979 rows=15 loops=1)
    -> Sort: `count(t.Medicine)` DESC, limit input to 15 row(s) per chunk  (actual time=3017.975..3017.976 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.004..1.229 rows=5383 loops=1)
            -> Aggregate using temporary table  (actual time=3015.293..3016.993 rows=5383 loops=1)
                -> Nested loop inner join  (cost=172540.45 rows=0) (actual time=2574.741..2974.270 rows=47624 loops=1)
                    -> Filter: (t.IllnessName is not null)  (cost=28976.95 rows=287127) (actual time=0.330..157.427 rows=297965 loops=1)
                        -> Table scan on t  (cost=28976.95 rows=287127) (actual time=0.328..130.093 rows=297965 loops=1)
                    -> Index lookup on s using <auto_key0> (IllnessName=t.IllnessName)  (actual time=0.001..0.001 rows=0 loops=297965)
                        -> Materialize  (cost=0.00..0.00 rows=0) (actual time=2757.986..2766.115 rows=15 loops=1)
                            -> Limit: 15 row(s)  (actual time=2560.024..2560.026 rows=15 loops=1)
                                -> Sort: `count(Has_Symptom.IllnessName)` DESC, limit input to 15 row(s) per chunk  (actual time=2560.024..2560.025 rows=15 loops=1)
                                    -> Table scan on <temporary>  (actual time=0.003..0.306 rows=1401 loops=1)
                                        -> Aggregate using temporary table  (actual time=2559.479..2559.863 rows=1401 loops=1)
                                            -> Filter: ((Has_Symptom.Severity = 4) and (Has_Symptom.SymptomName like 'Headache'))  (cost=317995.46 rows=70666) (actual time=
3.673..2527.588 rows=22525 loops=1)
                                                -> Index range scan on Has_Symptom using symname_idx  (cost=317995.46 rows=706656) (actual time=3.641..2496.008 rows=361353
loops=1)
 |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Indexing analysis 3: Index on IllnessName

IllnessName is used in the subquery, so indexing this column should help reduce our time. Looking at the EXPLAIN ANALYZE, we can see that there is a slight time reduction, although not as significant as adding an index on SymptomName.



```
---------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN




                                                                                                                                                                     |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Limit: 15 row(s)  (actual time=6030.447..6030.449 rows=15 loops=1)
    -> Sort: `count(t.Medicine)` DESC, limit input to 15 row(s) per chunk  (actual time=6030.447..6030.448 rows=15 loops=1)
        -> Table scan on <temporary>  (actual time=0.004..0.970 rows=5383 loops=1)
            -> Aggregate using temporary table  (actual time=6028.637..6029.924 rows=5383 loops=1)
                -> Nested loop inner join  (cost=172540.45 rows=0) (actual time=5705.917..5993.575 rows=47624 loops=1)
                    -> Filter: (t.IllnessName is not null)  (cost=28976.95 rows=287127) (actual time=0.064..107.196 rows=297965 loops=1)
                        -> Table scan on t  (cost=28976.95 rows=287127) (actual time=0.062..86.301 rows=297965 loops=1)
                    -> Index lookup on s using <auto_key0> (IllnessName=t.IllnessName)  (actual time=0.000..0.000 rows=0 loops=297965)
                        -> Materialize  (cost=0.00..0.00 rows=0) (actual time=5837.969..5845.307 rows=15 loops=1)
                            -> Limit: 15 row(s)  (actual time=5691.417..5691.419 rows=15 loops=1)
                                -> Sort: `count(Has_Symptom.IllnessName)` DESC, limit input to 15 row(s) per chunk  (actual time=5691.417..5691.418 rows=15 loops=1)
                                    -> Table scan on <temporary>  (actual time=0.002..0.285 rows=1401 loops=1)
                                        -> Aggregate using temporary table  (actual time=5690.893..5691.260 rows=1401 loops=1)
                                            -> Filter: ((Has_Symptom.Severity = 4) and (Has_Symptom.SymptomName like 'Headache'))  (cost=1369135.60 rows=150643) (actual tim
e=0.598..5656.276 rows=22525 loops=1)
                                                -> Table scan on Has_Symptom  (cost=1369135.60 rows=13559196) (actual time=0.039..4523.637 rows=14468863 loops=1)
 |
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
```