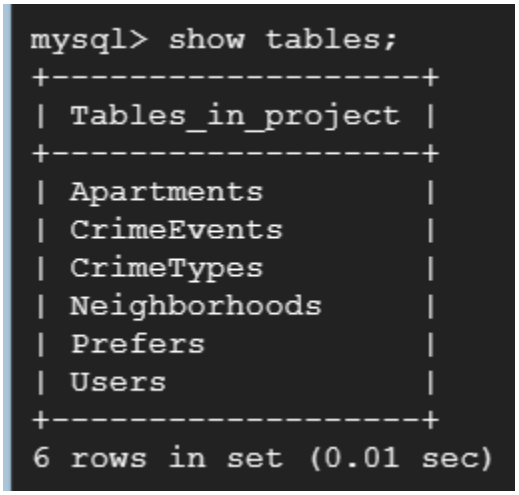
Here is a picture showing the terminal of the GCP with the databases with the tables we implemented. Our database is located on the Google GCP.



DDL Commands For Creating the Table

Users:

```
CREATE TABLE Users (
    UserId VARCHAR(255) PRIMARY KEY,
    Email VARCHAR(255) UNIQUE,
    Password VARCHAR(255),
    Gender VARCHAR(255),
    Age INT DEFAULT -1
);
```

Prefers:

```
CREATE TABLE Prefers(
    UserId VARCHAR(255),
    NeighborhoodName VARCHAR(255),
    FOREIGN KEY (UserId) REFERENCES Users(UserId),
    FOREIGN KEY (NeighborhoodName) REFERENCES Neighborhoods(NeighborhoodName)
);
```

CRIME TYPES:

```
CREATE TABLE CrimeTypes (
   CrimeId INTEGER PRIMARY KEY,
   PrimaryType VARCHAR(255),
   Arrested BOOLEAN
);
```

CRIME EVENTS:

```
CREATE TABLE CrimeEvents (
   CrimeId INTEGER PRIMARY KEY,
   Date DATETIME,
   LocationDescription VARCHAR(255),
   Block VARCHAR(255)
);
```

Neighborhoods:

```
CREATE TABLE Neighborhoods (
   NeighborhoodName VARCHAR(255) PRIMARY KEY,
   AverageRent INTEGER ,
   AverageAge INTEGER ,
   Demographic VARCHAR(255)
);
```

Apartments:

```
CREATE TABLE Apartments (
   ApartmentId INTEGER PRIMARY KEY,
   Neighborhood VARCHAR(255),
   FOREIGN KEY (Neighborhood) REFERENCES Neighborhoods(NeighborhoodName),
   RenterCompanyName VARCHAR(255),
   Rating INTEGER,
   Cost INTEGER
);
```

Entry into the Table Implementation

We decided to implement implement entries for Users, Crime Type and Crime Event

For Table Users, some users are inserted into the table.
INSERT INTO Users (UserId, Email, Password, Gender, Age)
VALUES ('user01', 'paul@google.com', 'password1', 'male', 32);
VALUES ('user02', 'ethan@google.com', 'password2', 'male', 23);
VALUES ('user03', 'alex@google.com', 'password3', 'male', 45);
VALUES ('user04', 'linda@google.com', 'password4', 'female', 37);
VALUES ('user05', 'mike@google.com', 'password5', 'male', 40');

For Table CrimeType, some CrimeTypes are inserted into the table.
INSERT INTO CrimeTypes (CrimeId, PrimaryType , Arrested)
VALUES (12589893, 'SEX OFFENSE', FALSE);
VALUES (12592454, 'OTHER OFFENSE', FALSE);
VALUES (12601676, 'OFFENSE INVOLVING CHILDREN', TRUE);

For Table CrimeEvent, some CrimeEvent are inserted into the table.
INSERT INTO CrimeEvent(CrimeId, Date, LocationDescription, Block)
VALUES (12589893, 1/11/2022 3:00, RESIDENCE, 087XX S KINGSTON AVE);
VALUES (12592454, 1/14/2022 15:55, RESIDENCE, 067XX S MORGAN ST);
VALUES (12601676, 1/13/2022 16:00, STREET, 031XX W AUGUSTA BLVD);

SELECT * FROM CrimeEvents;

```
                  |0-00 00:00:00 |                                                    | 026XX N Burling St
                    |00 00:00:00 | RESIDENCE                                          | 070XX W IMLAY ST
                    |00 00:00:00 | RESIDENCE                                          | 066XX S LOWE AVE
                   |0-00 00:00:00 |                                                    | 006XX W MADISON ST
                    |-00 00:00:00 | RESIDENCE                                          | 101XX S YATES AVE
                 |0-00-00 00:00:00 | APARTMENT                                          | 038XX S LAKE PARK AVE
                  |00-00 00:00:00 | RESIDENCE                                          | 065XX S SANGAMON ST
                      |0 00:00:00 | APARTMENT                                          | 011XX E 82ND ST
                 |0-00-00 00:00:00 | OTHER (SPECIFY)                                    | 035XX N GREENVIEW AVE
                   |0-00 00:00:00 | RESIDENCE                                          | 060XX S AUSTIN AVE
                   |0-00 00:00:00 | RESIDENCE                                          | 117XX S JUSTINE ST
                   |0-00 00:00:00 |                                                    | 109XX S VERNON AVE
                  |00-00 00:00:00 | AUTO / BOAT / RV DEALERSHIP                        | 019XX W PERSHING RD
                 |-00-00 00:00:00 | RESIDENCE                                          | 080XX S EXCHANGE AVE
                 |0-00-00 00:00:00 | RESIDENCE - PORCH / HALLWAY                        | 037XX W FULLERTON AVE
                 |-00-00 00:00:00 | HOSPITAL BUILDING / GROUNDS                        | 057XX W ROOSEVELT RD
                 |-00-00 00:00:00 | RESIDENCE                                          | 054XX W ROSEDALE AVE
                 |-00-00 00:00:00 | COMMERCIAL / BUSINESS OFFICE                       | 042XX S KILDARE BLVD
                  |00-00 00:00:00 | APARTMENT                                          | 021XX N KILDARE AVE
                  |00-00 00:00:00 | RESTAURANT                                         | 057XX N CENTRAL AVE
                   |0-00 00:00:00 | STREET                                             | 031XX N HALSTED ST
                 |0-00-00 00:00:00 | APARTMENT                                          | 021XX S PRINCETON AVE
                     |00 00:00:00 | SIDEWALK                                           | 031XX N BROADWAY
                    |-00 00:00:00 | ABANDONED BUILDING                                 | 021XX W GRAND AVE
                  |00-00 00:00:00 | STREET                                             | 009XX W RANDOLPH ST
                  |00-00 00:00:00 | OTHER (SPECIFY)                                    | 077XX S EMERALD AVE
                   |0-00 00:00:00 | APARTMENT                                          | 016XX W PIERCE AVE
                    |-00 00:00:00 | APARTMENT                                          | 056XX W NORTH AVE
                 |-00-00 00:00:00 | COMMERCIAL / BUSINESS OFFICE                       | 032XX W LAWRENCE AVE
                 |0-00-00 00:00:00 |                                                    | 035XX N CLAREMONT AVE
+----------+--------------------+----------------------------------------------------+--------------------------------------+
1001 rows in set (0.00 sec)
```

```
mysql> show tables;
+-------------------+
| Tables_in_project |
+-------------------+
| Apartments        |
| CrimeEvents       |
| CrimeTypes        |
| Neighborhoods     |
| Prefers           |
| Users             |
+-------------------+
6 rows in set (0.01 sec)

mysql> SELECT COUNT(*) FROM USERS;
ERROR 1146 (42S02): Table 'project.USERS' doesn't exist
mysql> SELECT COUNT(*) FROM Users;
+----------+
| COUNT(*) |
+----------+
|      644 |
+----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM CrimeTypes;
+----------+
| COUNT(*) |
+----------+
|     1001 |
+----------+
1 row in set (0.01 sec)

mysql> SELECT COUNT(*) FROM CrimeEvents;
+----------+
| COUNT(*) |
+----------+
|     1001 |
+----------+
1 row in set (0.00 sec)
```

Users is a CSV with 1001 entries in it, but it is currently an error with showing 644.
Crime types and Crime events are working as intended.

Advanced Queries

The Following queries have empty sets as the data was automatically generated and thus not fitting into the criteria of the Advanced Queries.

1. **Find what crime type was most committed between 10 and 11 AM.**

```
SELECT CrimeTypes.PrimaryType, COUNT(CrimeEvents.CrimeId) AS CrimeNum
FROM CrimeTypes
JOIN CrimeEvents ON CrimeTypes.CrimeId = CrimeEvents.CrimeId
WHERE TIME(CrimeEvents.Date) BETWEEN '00:10:00' AND '00:11:00'
GROUP BY CrimeTypes.PrimaryType
ORDER BY CrimeNum DESC
LIMIT 15;
```

```
mysql> SELECT CrimeTypes.PrimaryType, COUNT(CrimeEvents.CrimeId) AS CrimeNum
    -> FROM CrimeTypes
    -> JOIN CrimeEvents ON CrimeTypes.CrimeId = CrimeEvents.CrimeId
    -> WHERE TIME(CrimeEvents.Date) BETWEEN '00:10:00' AND '00:11:00'
    -> GROUP BY CrimeTypes.PrimaryType
    -> ORDER BY CrimeNum DESC
    -> LIMIT 15;
Empty set (0.00 sec)
```

This Query counts the number of crimes between the times of 10 and 11 AM and returns the most common crime type in that span of time.

2. **Get 15 neighborhoods where there have been fewer than 20 arrests**

```
SELECT n.NeighborhoodName
FROM Neighborhoods n NATURAL JOIN CrimeEvents ce NATURAL JOIN CrimeTypes ct
WHERE ct.Arrested = 0
GROUP BY n.NeighborhoodName
HAVING COUNT(*) < 500
LIMIT 15;
```

```
mysql> SELECT n.NeighborhoodName
    -> FROM Neighborhoods n NATURAL JOIN CrimeEvents ce NATURAL JOIN CrimeTypes ct
    -> WHERE ct.Arrested = 0
    -> GROUP BY n.NeighborhoodName
    -> HAVING COUNT(*) < 20
    -> LIMIT 15;
Empty set (0.08 sec)
```

3. **Get 15 neighborhoods with an average rent less than 1000 dollars and an average rating of 7.5 across all renter companies**

```
(SELECT n.NeighborhoodName
FROM Neighborhoods n
```

```
WHERE n.AverageRent < 1000

INTERSECT

SELECT a.Neighborhood
FROM Apartments a
GROUP BY a.Neighborhood
HAVING AVG(Rating) > 7.5)
LIMIT 15;
```

```
mysql> (SELECT n.NeighborhoodName
    -> FROM Neighborhoods n
    -> WHERE n.AverageRent < 1000
    ->
    -> INTERSECT
    ->
    -> SELECT a.Neighborhood
    -> FROM Apartments a
    -> GROUP BY a.Neighborhood
    -> HAVING AVG(Rating) > 7.5)
    -> LIMIT 15;
Empty set (0.01 sec)
```

4. **Get 15 neighborhoods with more than 20 males interested in them and an average age of under 25**

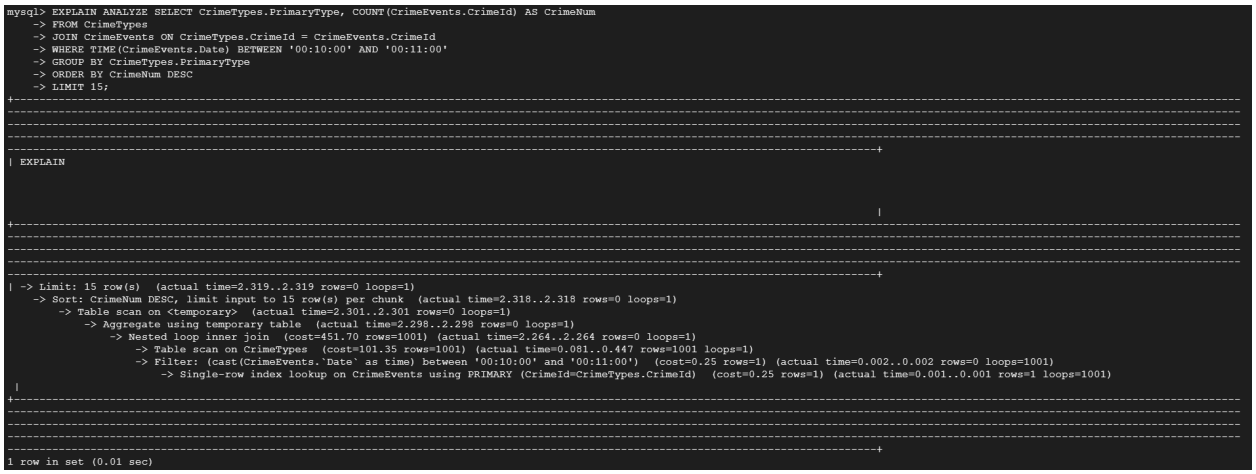
```
SELECT n.NeighborhoodName
FROM Prefers p NATURAL JOIN Users u NATURAL JOIN Neighborhoods n
WHERE n.AverageAge < 25 AND n.NeighborhoodName IN
        (SELECT pref.NeighborhoodName
        FROM Prefers pref
        NATURAL JOIN Users us
        WHERE us.Gender = 'Male'
        GROUP BY pref.NeighborhoodName
        HAVING COUNT(*) > 20)

LIMIT 15;
```

```
mysql> SELECT n.NeighborhoodName
    -> FROM Prefers p NATURAL JOIN Users u NATURAL JOIN Neighborhoods n
    -> WHERE n.AverageAge < 25 AND n.NeighborhoodName IN
    ->     (SELECT pref.NeighborhoodName
    ->     FROM Prefers pref
    ->     NATURAL JOIN Users us
    ->     WHERE us.Gender = 'Male'
    ->     GROUP BY pref.NeighborhoodName
    ->         HAVING COUNT(*) > 20)
    ->
    -> LIMIT 15;
Empty set (0.00 sec)
```

Indexing Analysis

Advanced Query 1

We first use Explain Analyze to find the performance of the original query.



The Cost is 451.70

We then apply 3 indexing methods to try and improve our runtime:

1. Index on CrimeEvent table for the CrimeId

   This index will speed up the JOIN operation between CrimeType and CrimeEvent tables, as it will quickly find the corresponding crimes in the CrimeType Table based on CrimeId.

   mysql> CREATE INDEX idxCrimeId ON CrimeEvent (CrimeId)

After implementing these changes, we noticed a slight improvement

Analysis: Since the CrimeEvent table is joined with the CrimeType table using CrimeId, having an index on CrimeId in CrimeEvents will help speed up the joining process of the two. The database would find the corresponding CrimeId in the CrimeTypes table making the join operation faster.