

Part 1

Implementation of Database Table using GCP

```
mysql> show databases;
+-----+
| Database |
+-----+
| Pokemon |
| Sighting |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
6 rows in set (0.05 sec)

mysql> show tables;
+-----+
| Tables_in_Pokemon |
+-----+
| Location |
| Organizations |
| Pokemon |
| Sighting |
| StatsCP |
+-----+
5 rows in set (0.00 sec)

mysql> 
```

DDL Commands

Tables:

```
CREATE TABLE Pokemon (
    pokemon_id INT PRIMARY KEY,
    pokemon_name VARCHAR(255),
    base_attack INT,
    base_defense INT,
    base_stamina INT,
    type VARCHAR(255),
    rarity VARCHAR(255),
);
```

```
CREATE TABLE StatsCP (  
    base_attack INT,  
    base_defense INT,  
    base_stamina INT,  
    max_cp INT,  
    PRIMARY KEY (base_attack, base_defense, base_stamina)  
);
```

```
CREATE TABLE Location (  
    longitude DECIMAL(20, 18),  
    latitude DECIMAL(15, 12),  
    city VARCHAR(255),  
    population_density DECIMAL(10,2),  
    closeToWater BOOLEAN,  
    PRIMARY KEY(longitude, latitude)  
);
```

```
CREATE TABLE Sighting (  
    sightingId VARCHAR(255),  
    pokemon_id INT,  
    longitude DECIMAL(20, 18),  
    latitude DECIMAL(15, 12),  
    appearedTimeOfDay VARCHAR(255),  
    weather VARCHAR(255),  
    temperature DECIMAL(4,2),  
    windSpeed DECIMAL(4,2),  
    PRIMARY KEY (sightingId),  
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(pokemon_id) ON DELETE  
CASCADE  
);
```

```
CREATE TABLE User (  
    userId INT PRIMARY KEY,  
    password VARCHAR(255),  
    role VARCHAR(255),  
    organizationName VARCHAR(255),  
    FOREIGN KEY organizationName REFERENCES Organizations(organizationName)  
ON DELETE CASCADE  
);
```

```
CREATE TABLE Reports (  
    reportId INT PRIMARY KEY,
```

```

    sightingId INT,
    userId INT,
    eventId INT,
    status VARCHAR(255),
    notes TEXT,
    time DATETIME,
    FOREIGN KEY sightingId REFERENCES Sighting(sightingId) ON DELETE CASCADE,
    FOREIGN KEY userId REFERENCES User(userId) ON DELETE CASCADE,
    FOREIGN KEY eventId REFERENCES Event(eventId) ON DELETE CASCADE

);

CREATE TABLE Events (
    eventId INT PRIMARY KEY,
    eventName VARCHAR(255),
    description TEXT,
    location VARCHAR(255),
    time DATETIME,
    participantCount INT,
    organizationName VARCHAR(255),
    FOREIGN KEY organizationName REFERENCES Organizations(organizationName)
ON DELETE CASCADE
);

CREATE TABLE Organizations (
    organizationName VARCHAR(255) PRIMARY KEY
);

```

1000 rows in three tables

```
5 rows in set (0.00 sec)

mysql> SELECT COUNT(*) FROM Location;
+-----+
| COUNT(*) |
+-----+
|    143919 |
+-----+
1 row in set (0.75 sec)

mysql> SELECT COUNT(*) FROM Pokemon
-> ;
+-----+
| COUNT(*) |
+-----+
|     1007 |
+-----+
1 row in set (0.11 sec)

mysql> SELECT COUNT(*) FROM Sighting;
+-----+
| COUNT(*) |
+-----+
|    296021 |
+-----+
1 row in set (2.15 sec)

mysql> 
```

Generalized/Plan SQL Queries:

1. IF 1 Filter SPECIFIED;

User defines X filter,

SELECT Pokemon, Longitude, Latitude

FROM Pokemon Natural Join StatsCP, Natural JOIN Sighting Natural Join Location

GROUP BY Pokemon.type

HAVING Distance >= (LongitudeUSER, LatitudeUSER) - (LongitudeSighting, LatitudeSighting)

2. IF X(i) Stats filters SPECIFIED;

User defines Xi filters,Y Distance

SELECT Pokemon, Longitude, Latitude

FROM Pokemon Natural Join StatsCP Natural JOIN Sighting Natural Join Location

```
GROUP BY X1,X2...Xi
HAVING Distance >= (LongitudeUSER, LatitudeUSER) - (LongitudeSighting, LatitudeSighting)
AND Xis in ranges of filters
```

3. IF X(i) Sighting filters SPECIFIED;
 User defines Xi filters,Y Distance
 SELECT Pokemon, Longitude, Latitude
 FROM Pokemon Natural Join StatsCP Natural JOIN Sighting Natural Join Location
 GROUP BY X1,X2...Xi
 HAVING Distance >= (LongitudeUSER, LatitudeUSER) - (LongitudeSighting, LatitudeSighting)
 AND Xis in ranges of filters

4. IF Pokemon SPECIFIED;
 User defines X as pokemon name,

```
-- params: :X, :latitudeUSER, :longitudeUSER, :DistanceUSER (km)
SELECT p.pokemon_name AS Pokemon, l.longitude, l.latitude,
  2 * 6371 * ASIN(
    SQRT(
      POWER(SIN(RADIANS(l.latitude - :latitudeUSER) / 2), 2) +
      COS(RADIANS(:latitudeUSER)) * COS(RADIANS(l.latitude)) *
      POWER(SIN(RADIANS(l.longitude - :longitudeUSER) / 2), 2)
    )
  ) AS actual
FROM Pokemon p NATURAL JOIN StatsCP NATURAL JOIN Sighting s NATURAL JOIN
Location l
WHERE p.pokemon_name = :X AND actual < :DistanceUSER
ORDER BY actual;
```

Specific Queries + 15 Results

First Query: finds number of each water type pokemon's sightings within 100 miles of Urbana-Champaign

```
SELECT Pokemon.pokemon\_name, COUNT(Pokemon.pokemon_name)
FROM Pokemon NATURAL JOIN Sighting
WHERE Pokemon.type = 'Water'
AND 100 >=
(SELECT ST_Distance_Sphere(
  POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
```

```

POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34)
GROUP BY Pokemon.pokemon\_name;

```

```

mysql> SELECT Pokemon.pokemon_name, COUNT(Pokemon.pokemon_name)
-> FROM Pokemon NATURAL JOIN Sighting
-> WHERE Pokemon.type = 'Water'
-> AND 100 >=
-> (SELECT ST_Distance_Sphere(
->     POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
->     POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34)
-> GROUP BY Pokemon.pokemon_name
-> ;

```

pokemon_name	COUNT(Pokemon.pokemon_name)
Krabby	3
Seel	1
Goldeen	2
Horsea	1
Poliwag	1

```

5 rows in set (9.51 sec)

```

Less than 15 rows in output

Second Query: finds average distance of each water type pokemon's sightings within 100 miles of Urbana-Champaign

```

SELECT Pokemon.pokemon\_name, AVG(ST_Distance_Sphere(
    POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
    POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34)
FROM Pokemon NATURAL JOIN Sighting
WHERE Pokemon.type = 'Water'
AND 100 >= (SELECT ST_Distance_Sphere(
    POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
    POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34)
GROUP BY Pokemon.pokemon\_name;

```

```
mysql> SELECT Pokemon.pokemon_name, AVG(ST_Distance_Sphere(
-> POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
-> POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34)
-> FROM Pokemon NATURAL JOIN Sighting
-> WHERE Pokemon.type = 'Water'
-> AND 100 >= (SELECT ST_Distance_Sphere(
-> POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
-> POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34)
-> GROUP BY Pokemon.pokemon_name
-> ;
```

pokemon_name	AVG(ST_Distance_Sphere(POINT(-88.2073, 40.1106), POINT(Sighting.longitude, Sighting.latitude)) / 1609.34)
Krabby	98.71722803754314
Seel	98.73747065903535
Goldene	87.33306333227473
Horsea	98.71891370230698
Poliwag	87.29944236020435

5 rows in set (9.11 sec)

Less than 15 rows in output

Third Query: finds number of pokemon sightings with the constraint of $1000 \leq \text{max_cp}$ ≤ 2000 and within 100 miles of Urbana-Champaign

```
SELECT Pokemon.pokemon\_name, MAX(StatsCP.max_cp) as max_cp,
COUNT(Pokemon.pokemon_name)
FROM Pokemon
JOIN StatsCP
ON Pokemon.base_attack = StatsCP.base_attack AND Pokemon.base_defense =
StatsCP.base_defense AND Pokemon.base_stamina = StatsCP.base_stamina
NATURAL JOIN Sighting
WHERE StatsCP.max_cp >= 1000 AND StatsCP.max_cp <= 2000
AND 100 >= (SELECT ST_Distance_Sphere(
    POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
    POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34)
GROUP BY Pokemon.pokemon\_name;
```

```
mysql> SELECT Pokemon.pokemon_name, MAX(StatsCP.max_cp) as max_cp, COUNT(Pokemon.pokemon_name)
-> FROM Pokemon
-> JOIN StatsCP
-> ON Pokemon.base_attack = StatsCP.base_attack AND Pokemon.base_defense = StatsCP.base_defense AND Pokemon.base_stamina = StatsCP.base_stamina
-> NATURAL JOIN Sighting
-> WHERE StatsCP.max_cp >= 1000 AND StatsCP.max_cp <= 2000
-> AND 100 >= (SELECT ST_Distance_Sphere(
-> POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
-> POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34)
-> GROUP BY Pokemon.pokemon_name;
```

pokemon_name	max_cp	COUNT(Pokemon.pokemon_name)
Pidgeotto	1366	3
Raticate	1950	3
Oddish	1405	2
Paras	1164	3
Venonat	1148	9
Bellsprout	1182	3
Seel	1111	1
Gastly	1406	1
Drowzee	1190	10
Krabby	1785	3
Horsea	1208	1
Goldene	1318	2
Eevee	1225	13
Dratini	1149	1

14 rows in set (1 min 14.04 sec)

Less than 15 rows in output

Fourth Query: finding pokemon that have been sighted within 100 miles of Urbana-Champaign and within 100 miles of San Francisco

```
((SELECT Pokemon.pokemon\_name
FROM Pokemon NATURAL JOIN Sighting
WHERE 100 >= (SELECT ST_Distance_Sphere(
    POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
    POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34))
INTERSECT
(SELECT Pokemon.pokemon\_name
FROM Pokemon NATURAL JOIN Sighting
WHERE 100 >= (SELECT ST_Distance_Sphere(
    POINT(-99.1332, 19.4326), -- Mexico City (lon, lat)
    POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
) / 1609.34)))
LIMIT 15;
```



```
mysql> ((SELECT Pokemon.pokemon_name
-> FROM Pokemon NATURAL JOIN Sighting
-> WHERE 100 >= (SELECT ST_Distance_Sphere(
-> POINT(-88.2073, 40.1106), -- Urbana-Champaign (lon, lat)
-> POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34))
-> INTERSECT
-> (SELECT Pokemon.pokemon_name
-> FROM Pokemon NATURAL JOIN Sighting
-> WHERE 100 >= (SELECT ST_Distance_Sphere(
-> POINT(-99.1332, 19.4326), -- Mexico City (lon, lat)
-> POINT(Sighting.longitude, Sighting.latitude) -- Sighting (lon, lat)
-> ) / 1609.34)))
-> LIMIT 15
-> ;
```

pokemon_name
Eevee
Weedle
Pidgey
Oddish
Rattata
Venonat
Dratini
Drowzee
Raticate
Bellsprout
Krabby
Metapod
Pidgeotto
Spearow
Paras

15 rows in set (14.18 sec)

Limited to 15 rows in output

Part 2. Indexing

Query 1

```
| -> Table scan on <temporary> (actual time=5276..5276 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=5276..5276 rows=5 loops=1)
|     -> Nested loop inner join (cost=319315 rows=29549) (actual time=1175..5275 rows=8 loops=1)
|       -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is not nu
11)) (cost=31213 rows=295489) (actual time=218..5255 rows=170 loops=1)
|         -> Table scan on Sighting (cost=31213 rows=295489) (actual time=0.22..4506 rows=296021 loops=1)
|           -> Filter: (Pokemon.type = 'Water') (cost=0.875 rows=0.1) (actual time=0.12..0.12 rows=0.0471 loops=170)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.113..0.113 rows=1 loops=170)
```

Original

```
| -> Table scan on <temporary> (actual time=7166..7166 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=7166..7166 rows=5 loops=1)
|     -> Nested loop inner join (cost=245765 rows=29549) (actual time=1481..7166 rows=8 loops=1)
|       -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is not nu
11)) (cost=31536 rows=295489) (actual time=174..7161 rows=170 loops=1)
|         -> Table scan on Sighting (cost=31536 rows=295489) (actual time=94.9..6432 rows=296021 loops=1)
|           -> Filter: (Pokemon.type = 'Water') (cost=0.625 rows=0.1) (actual time=0.0268..0.0268 rows=0.0471 loops=170)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.625 rows=1) (actual time=0.0202..0.0203 rows=1 loops=170)
```

Type Index

```

-----+
| -> Table scan on <temporary> (actual time=32607..32607 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=32607..32607 rows=5 loops=1)
|     -> Nested loop inner join (cost=138518 rows=163953) (actual time=12952..32607 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=25.7 rows=73) (actual time=20.6..148 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1676 rows=2246) (actual time=345..445 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1676 rows=2246) (actual time=40.6..443 rows=473 loops=73)
|
|-----+

```

Type, maxCP Indexes

```

-----+
| -> Table scan on <temporary> (actual time=26147..26147 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=26147..26147 rows=5 loops=1)
|     -> Nested loop inner join (cost=137024 rows=163953) (actual time=8895..26147 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=25.7 rows=73) (actual time=17..115 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1655 rows=2246) (actual time=276..357 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1655 rows=2246) (actual time=41.3..355 rows=473 loops=73)
|
|-----+

```

Type, maxcp, base_attack Indexes

```

-----+
| -> Table scan on <temporary> (actual time=22655..22655 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=22655..22655 rows=5 loops=1)
|     -> Nested loop inner join (cost=115482 rows=163953) (actual time=8301..22655 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=20.4 rows=73) (actual time=0.0775..54.9 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1360 rows=2246) (actual time=262..310 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1360 rows=2246) (actual time=22.8..308 rows=473 loops=73)
|
|-----+

```

Type, maxcp, base_attack, base_defense Indexes

Query 2

```

-----+
| -> Table scan on <temporary> (actual time=7166..7166 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=7166..7166 rows=5 loops=1)
|     -> Nested loop inner join (cost=245765 rows=29549) (actual time=1481..7166 rows=8 loops=1)
|       -> Filter: ((100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is not null)) (cost=31536 rows=295489) (actual time=474..7161 rows=170 loops=1)
|       -> Table scan on Sighting (cost=31536 rows=295489) (actual time=94.9..6432 rows=296021 loops=1)
|       -> Filter: (Pokemon.type = 'Water') (cost=0.625 rows=0.1) (actual time=0.0268..0.0268 rows=0.0471 loops=170)
|       -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.625 rows=1) (actual time=0.0202..0.0203 rows=1 loops=170)
|
|-----+

```

Original

```

-----+
| -> Table scan on <temporary> (actual time=27856..27856 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=27856..27856 rows=5 loops=1)
|     -> Nested loop inner join (cost=142001 rows=163953) (actual time=13179..27855 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=20.4 rows=73) (actual time=0.0848..175 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1723 rows=2246) (actual time=311..379 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1723 rows=2246) (actual time=34.7..378 rows=473 loops=73)
|
|-----+

```

Type, maxcp Indexes

```

-----+
| -> Table scan on <temporary> (actual time=27853..27853 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=27853..27853 rows=5 loops=1)
|     -> Nested loop inner join (cost=140124 rows=163953) (actual time=12620..27853 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=25.7 rows=73) (actual time=17.7..114 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1698 rows=2246) (actual time=298..380 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1698 rows=2246) (actual time=37.8..379 rows=473 loops=73)
|
|-----+

```

Type, maxcp, base_attack Indexes

```

-----+
| -> Table scan on <temporary> (actual time=25443..25443 rows=5 loops=1)
|   -> Aggregate using temporary table (actual time=25443..25443 rows=5 loops=1)
|     -> Nested loop inner join (cost=138906 rows=163953) (actual time=8528..25442 rows=8 loops=1)
|       -> Index lookup on Pokemon using type_index (type='Water') (cost=25.7 rows=73) (actual time=24.1..142 rows=73 loops=1)
|       -> Filter: (100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1681 rows=2246) (actual time=286..347 rows=0.11 loops=73)
|       -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1681 rows=2246) (actual time=31.5..345 rows=473 loops=73)
|
|-----+

```

Type, maxcp, base_attack, base_defense Indexes

Query 3

```

-----+
| -> Table scan on <temporary> (actual time=7810..7810 rows=14 loops=1)
|   -> Aggregate using temporary table (actual time=7810..7810 rows=14 loops=1)
|     -> Nested loop inner join (cost=321278 rows=30189) (actual time=427..7808 rows=55 loops=1)
|       -> Nested loop inner join (cost=226163 rows=271758) (actual time=427..7774 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(- (88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is not null)) (cost=29138 rows=271758) (actual time=427..7767 rows=170 loops=1)
|         -> Table scan on Sighting (cost=29138 rows=271758) (actual time=43.2..6908 rows=296021 loops=1)
|         -> Filter: ((Pokemon.base_attack is not null) and (Pokemon.base_defense is not null) and (Pokemon.base_stamina is not null)) (cost=0.625 rows=1) (actual time=0.0385..0.0386 rows=1 loops=170)
|         -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.625 rows=1) (actual time=0.0368..0.0368 rows=1 loops=170)
|         -> Filter: ((StatsCP.max_cp >= 1000) and (StatsCP.max_cp <= 2000)) (cost=0.25 rows=0.111) (actual time=0.201..0.201 rows=0.324 loops=170)
|         -> Single-row index lookup on StatsCP using PRIMARY (base_attack=Pokemon.base_attack, base_defense=Pokemon.base_defense, base_stamina=Pokemon.base_stamina) (cost=0.25 rows=1) (actual time=0.198..0.198 rows=1 loops=170)
|
|-----+

```

Original

```
-----+-----
| -> Table scan on <temporary> (actual time=6237..6237 rows=14 loops=1)
|   -> Aggregate using temporary table (actual time=6237..6237 rows=14 loops=1)
|     -> Nested loop inner join (cost=321253 rows=86188) (actual time=482..6235 rows=55 loops=1)
|       -> Nested loop inner join (cost=226137 rows=271758) (actual time=482..6232 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29113 rows=271758) (actual time=482..6228 rows=170 loops=1)
|           -> Table scan on Sighting (cost=29113 rows=271758) (actual time=35.4..5462 rows=296021 loops=1)
|             -> Filter: ((Pokemon.base_attack is not null) and (Pokemon.base_defense is not null) and (Pokemon.base_stamina is not null)) (cost=0.625 rows=1) (actual time=0.0
249..0.025 rows=1 loops=170)
|               -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.625 rows=1) (actual time=0.0228..0.0229 rows=1 loops=170)
|                 -> Filter: ((StatsCP.max_cp >= 1000) and (StatsCP.max_cp <= 2000)) (cost=0.25 rows=0.317) (actual time=0.0176..0.0176 rows=0.324 loops=170)
|                   -> Single-row index lookup on StatsCP using PRIMARY (base_attack=Pokemon.base_attack, base_defense=Pokemon.base_defense, base_stamina=Pokemon.base_stamina) (cost
=0.25 rows=1) (actual time=0.0143..0.0143 rows=1 loops=170)
|-----+-----
```

Type, maxcp Indexes

```
-----+-----
| -> Table scan on <temporary> (actual time=7339..7339 rows=14 loops=1)
|   -> Aggregate using temporary table (actual time=7339..7339 rows=14 loops=1)
|     -> Nested loop inner join (cost=219379 rows=30169) (actual time=618..7337 rows=55 loops=1)
|       -> Nested loop inner join (cost=124264 rows=271758) (actual time=618..7275 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29149 rows=271758) (actual time=618..7271 rows=170 loops=1)
|           -> Table scan on Sighting (cost=29149 rows=271758) (actual time=42.3..6491 rows=296021 loops=1)
|             -> Filter: ((Pokemon.base_attack is not null) and (Pokemon.base_defense is not null) and (Pokemon.base_stamina is not null)) (cost=0.25 rows=1) (actual time=0.02
25..0.0226 rows=1 loops=170)
|               -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.25 rows=1) (actual time=0.0209..0.0209 rows=1 loops=170)
|                 -> Filter: ((StatsCP.max_cp >= 1000) and (StatsCP.max_cp <= 2000)) (cost=0.25 rows=0.111) (actual time=0.364..0.364 rows=0.324 loops=170)
|                   -> Single-row index lookup on StatsCP using PRIMARY (base_attack=Pokemon.base_attack, base_defense=Pokemon.base_defense, base_stamina=Pokemon.base_stamina) (cost
=0.25 rows=1) (actual time=0.361..0.361 rows=1 loops=170)
|-----+-----
```

Type index

```
-----+-----
| -> Table scan on <temporary> (actual time=76684..76684 rows=14 loops=1)
|   -> Aggregate using temporary table (actual time=76684..76684 rows=14 loops=1)
|     -> Nested loop inner join (cost=35892 rows=34475) (actual time=11621..76682 rows=55 loops=1)
|       -> Nested loop inner join (cost=507 rows=153) (actual time=2.33..281 rows=313 loops=1)
|         -> Filter: ((StatsCP.max_cp >= 1000) and (StatsCP.max_cp <= 2000)) (cost=63 rows=307) (actual time=2.23..11.1 rows=307 loops=1)
|           -> Covering index range scan on StatsCP using maxcp index over (1000 <= max_cp <= 2000) (cost=63 rows=307) (actual time=1.4..9.97 rows=307 loops=1)
|             -> Filter: ((Pokemon.base_stamina = StatsCP.base_stamina) and (Pokemon.base_defense = StatsCP.base_defense)) (cost=1.03 rows=0.05) (actual time=0.098..0.879 rows
=1.02 loops=307)
|               -> Index lookup on Pokemon using attack index (base_attack=StatsCP.base_attack) (cost=1.03 rows=4.13) (actual time=0.0948..0.876 rows=6.66 loops=307)
|                 -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=2095 rows=2246) (actual time
=155..244 rows=0.176 loops=313)
|                   -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=2095 rows=2246) (actual time=22.2..243 rows=348 loops=313)
|-----+-----
```

Type, maxcp, base_attack Indexes

```
-----+-----
| -> Table scan on <temporary> (actual time=82108..82108 rows=14 loops=1)
|   -> Aggregate using temporary table (actual time=82108..82108 rows=14 loops=1)
|     -> Nested loop inner join (cost=26171 rows=34475) (actual time=10788..82106 rows=55 loops=1)
|       -> Nested loop inner join (cost=507 rows=153) (actual time=14..391 rows=313 loops=1)
|         -> Filter: ((StatsCP.max_cp >= 1000) and (StatsCP.max_cp <= 2000)) (cost=63 rows=307) (actual time=0.0864..28 rows=307 loops=1)
|           -> Covering index range scan on StatsCP using maxcp index over (1000 <= max_cp <= 2000) (cost=63 rows=307) (actual time=0.0825..27.7 rows=307 loops=1)
|             -> Filter: ((Pokemon.base_stamina = StatsCP.base_stamina) and (Pokemon.base_defense = StatsCP.base_defense)) (cost=1.03 rows=0.05) (actual time=0.531..1.18 rows=
1.02 loops=307)
|               -> Index lookup on Pokemon using attack index (base_attack=StatsCP.base_attack) (cost=1.03 rows=4.13) (actual time=0.524..1.18 rows=6.66 loops=307)
|                 -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) (cost=1462 rows=2246) (actual time
=163..261 rows=0.176 loops=313)
|                   -> Index lookup on Sighting using pokemon_id (pokemon_id=Pokemon.pokemon_id) (cost=1462 rows=2246) (actual time=21.4..260 rows=348 loops=313)
|-----+-----
```

Type, maxcp, base_attack, base_defense Indexes

Query 4

```
-----+-----
| -> Limit: 15 row(s) (cost=668831..668832 rows=15) (actual time=14298..14298 rows=15 loops=1)
|   -> Table scan on <intersect temporary> (cost=668831..672527 rows=295489) (actual time=14298..14298 rows=15 loops=1)
|     -> Intersect materialize with deduplication (cost=668831..668831 rows=295489) (actual time=14298..14298 rows=26 loops=1)
|       -> Nested loop inner join (cost=319641 rows=295489) (actual time=418..7621 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=31539 rows=295489) (actual time=394..7583 rows=170 loops=1)
|           -> Table scan on Sighting (cost=31539 rows=295489) (actual time=20.6..6805 rows=296021 loops=1)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.217..0.218 rows=1 loops=170)
|               -> Nested loop inner join (cost=319641 rows=295489) (actual time=2.74..6638 rows=3136 loops=1)
|                 -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(99.1332),19.4326)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=31539 rows=295489) (actual time=2.68..6600 rows=3136 loops=1)
|                   -> Table scan on Sighting (cost=31539 rows=295489) (actual time=2.6..5803 rows=296021 loops=1)
|                     -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.0107..0.0107 rows=1 loops=3136)
|-----+-----
```

Original

```
-----+-----
| -> Limit: 15 row(s) (cost=668831..668832 rows=15) (actual time=14298..14298 rows=15 loops=1)
|   -> Table scan on <intersect temporary> (cost=668831..672527 rows=295489) (actual time=14298..14298 rows=15 loops=1)
|     -> Intersect materialize with deduplication (cost=668831..668831 rows=295489) (actual time=14298..14298 rows=26 loops=1)
|       -> Nested loop inner join (cost=319641 rows=295489) (actual time=418..7621 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=31539 rows=295489) (actual time=394..7583 rows=170 loops=1)
|           -> Table scan on Sighting (cost=31539 rows=295489) (actual time=20.6..6805 rows=296021 loops=1)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.217..0.218 rows=1 loops=170)
|               -> Nested loop inner join (cost=319641 rows=295489) (actual time=2.74..6638 rows=3136 loops=1)
|                 -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-(99.1332),19.4326)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=31539 rows=295489) (actual time=2.68..6600 rows=3136 loops=1)
|                   -> Table scan on Sighting (cost=31539 rows=295489) (actual time=2.6..5803 rows=296021 loops=1)
|                     -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.0107..0.0107 rows=1 loops=3136)
|-----+-----
```

Type index

```
-----+
| -> Limit: 15 row(s) (cost=615584..615584 rows=15) (actual time=14531..14531 rows=15 loops=1)
|   -> Table scan on <intersect temporary> (cost=615584..618984 rows=271758) (actual time=14531..14531 rows=15 loops=1)
|     -> Intersect materialize with deduplication (cost=615584..615584 rows=271758) (actual time=14531..14531 rows=26 loops=1)
|       -> Nested loop inner join (cost=294204 rows=271758) (actual time=518..7313 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29240 rows=271758) (actual time=492..7280 rows=170 loops=1)
|           -> Table scan on Sighting (cost=29240 rows=271758) (actual time=96.2..6525 rows=296021 loops=1)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.19..0.19 rows=1 loops=170)
|             -> Nested loop inner join (cost=294204 rows=271758) (actual time=7.43..7182 rows=3136 loops=1)
|               -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-99.1332),19.4326)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29240 rows=271758) (actual time=7.41..7140 rows=3136 loops=1)
|                 -> Table scan on Sighting (cost=29240 rows=271758) (actual time=7.31..6254 rows=296021 loops=1)
|                   -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.875 rows=1) (actual time=0.0119..0.012 rows=1 loops=3136)
|
```

Type, maxcp indexes

```
-----+
| -> Limit: 15 row(s) (cost=547383..547383 rows=15) (actual time=15385..15385 rows=15 loops=1)
|   -> Table scan on <intersect temporary> (cost=547383..550782 rows=271758) (actual time=15385..15385 rows=15 loops=1)
|     -> Intersect materialize with deduplication (cost=547383..547383 rows=271758) (actual time=15385..15385 rows=26 loops=1)
|       -> Nested loop inner join (cost=260104 rows=271758) (actual time=413..7673 rows=170 loops=1)
|         -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-88.2073),40.1106)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29109 rows=271758) (actual time=364..7619 rows=170 loops=1)
|           -> Table scan on Sighting (cost=29109 rows=271758) (actual time=40.1..6879 rows=296021 loops=1)
|             -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.75 rows=1) (actual time=0.316..0.316 rows=1 loops=170)
|             -> Nested loop inner join (cost=260104 rows=271758) (actual time=3.79..7681 rows=3136 loops=1)
|               -> Filter: ((100 >= (st_distance_sphere(<cache>(point(-99.1332),19.4326)),point(Sighting.longitude,Sighting.latitude)) / 1609.34)) and (Sighting.pokemon_id is no
t null)) (cost=29109 rows=271758) (actual time=3.75..7647 rows=3136 loops=1)
|                 -> Table scan on Sighting (cost=29109 rows=271758) (actual time=3.65..6878 rows=296021 loops=1)
|                   -> Single-row index lookup on Pokemon using PRIMARY (pokemon_id=Sighting.pokemon_id) (cost=0.75 rows=1) (actual time=0.0096..0.00968 rows=1 loops=3136)
|
```

Type, maxcp, base_attack, base_defense Indexes

Final Index Design Report

After analyzing the queries, we implemented indexes on the following attributes:

1. Pokemon.type
 - a. CREATE INDEX type_index ON Pokemon(type)
2. StatsCP.max_cp
 - a. CREATE INDEX maxcp_index ON StatsCP(max_cp)
3. Pokemon.base_attack
 - a. CREATE INDEX attack_index ON Pokemon(base_attack)
4. Pokemon.base_defense
 - a. CREATE INDEX defense_index ON Pokemon(base_defense)

Rationale and Query Analysis

1. Pokemon.type – This index directly benefits Query #1, Query #4 and Query #2, which filter Pokémon by type (WHERE Pokemon.type = 'Water'). By indexing this column, the database can quickly locate only the relevant Pokémon instead of scanning the entire table.

2. StatsCP.max_cp – This index is used in Query #3, which filters Pokémon based on max_cp (WHERE StatsCP.max_cp >= 1000 AND StatsCP.max_cp <= 2000). Indexing max_cp allows the database to efficiently find rows within the specified range without scanning all entries in StatsCP. Although it was only specified/joined in Query #3, it did improve the costs of Query 1-4 by a slight amount, due to StatsCP sharing most of its attributes with the Pokemon table, improving the cache/Pokemon lookups.

3. Pokemon.base_attack and Pokemon.base_defense – These attributes are used to join Pokemon with StatsCP in Query #3. Indexing them improves the join performance by allowing the database to quickly match Pokémon with their corresponding stats. Although it was only specified in Query #3, it did improve the costs of Query 1-4 by 10 fold, due to the Pokemon table being used in every single query.

Observations

After applying these indexes, the performance improvement was very good for all queries, particularly because Queries 1,2,3,4 involves the indexed attributes in join and where clauses. The implemented indexes helped with filtering and joins, but the main computational cost remains calculating distances for each sighting row.