Assignment-3: Advanced SQL, GEO-1006

Ming-Chieh Hu, Hongyu Ye, Carmem E. F. Aires

November 2024

Project Description

This report was made by:

1. **Ming-Chieh Hu**, student number: 6186416
2. **Hongyu Ye**, student number: 6286240
3. **Carmem E. F. Aires**, student number: 4325893

Contents

1 Question 1

Question: Create at least one index on the dataset: Use the create index SQL statement for this.

Answer:

```sql
create unique index id_log
on bluelog(id);

create unique index id_scanner
on bluetoothlocations(scanner);

select
    tablename, indexname, indexdef
from
    pg_indexes
where
    tablename IN ('bluelog', 'bluetoothlocations');
```

	tablename name	indexname name	indexdef text
1	bluelog	bluelog_pkey	CREATE UNIQUE INDEX bluelog_pkey ON public.bluelog USING btree (id)
2	bluetoothlocations	bluetoothlocations_pkey	CREATE UNIQUE INDEX bluetoothlocations_pkey ON public.bluetoothlocations USING btree (scan…
3	bluelog	idx_bluelog_mac	CREATE INDEX idx_bluelog_mac ON public.bluelog USING btree (mac)
4	bluetoothlocations	idx_bluetoothlocations_scanner	CREATE INDEX idx_bluetoothlocations_scanner ON public.bluetoothlocations USING btree (scann…

2 Question 2

2.1 Data cleansing 1

Question: Remove all entries of Bluetooth devices that are seen only by one scanner because these are probably non mobile Bluetooth device that are positioned close to one of the scanners.

Answer:

```sql
delete from bluelog b2 where b2.mac in (
    select b.mac
    from bluelog b
    group by b.mac
    having count(b.id) = 1
);
```

Name	Value
Updated Rows	852
Query	delete from bluelog b2 where b2.mac in (select b.mac from bluelog b group by b.mac having count(b.id) = 1)
Start time	Tue Nov 26 14:05:14 CET 2024
Finish time	Tue Nov 26 14:05:14 CET 2024

2.2 Data cleansing 2

Question: Remove all entries of Bluetooth devices that are seen over 1 hour because these devices are probably not moving as well.

Answer:

```sql
delete from bluelog b2 where b2.mac in (
    select b.mac
    from bluelog b
    where b.endtime - b.starttime > '01:00:00'
);
```

Name	Value
Updated Rows	9531
Query	delete from bluelog b2 where b2.mac in (select b.mac from bluelog b where b.endtime - b.starttime > '01:00:00')
Start time	Tue Nov 26 14:40:39 CET 2024
Finish time	Tue Nov 26 14:40:39 CET 2024

3 Question 3

3.1 Top 10 routes

Question: Find the top 10 of routes that are taken most frequently. A route is a combination of two consecutive points for the same Bluetooth device.

Answer:

```sql
select bl1."name" as route_start, bl2."name" as route_end, count((b.scanner, b2.scanner)) as freq
from bluelog b
join bluelog b2 on b."next" = b2.id
join bluetoothlocations bl1 on b.scanner = bl1.scanner
join bluetoothlocations bl2 on b2.scanner = bl2.scanner
where b.scanner <> b2.scanner
group by (bl1."name", bl2."name")
order by freq desc limit 10;
```

Grid Text Record

	route_start	route_end	freq
1	Zuidwal/Oude Delft	Hambrug	7,508
2	Hambrug	Zuidwal/Oude Delft	5,956
3	Hambrug	Michiel de Ruyterweg	5,781
4	Koepoortbrug	Oosteinde/Nieuwe lange dijk	3,496
5	Oosteinde/Nieuwe lange dijk	Koepoortbrug	2,999
6	Michiel de Ruyterweg	Hambrug	2,425
7	Michiel de Ruyterweg	Zuidwal/Oude Delft	2,178
8	Zuidwal/Oude Delft	Michiel de Ruyterweg	1,525
9	Parkeergarage zuidpoort	Michiel de Ruyterweg	1,419
10	Hambrug	Parkeergarage zuidpoort	1,403

3.2 Rush hour

Question: Is it really true that people travel faster during rush hour?

Answer:

```sql
-- if a person travels more than 10 hours,
-- we assume he/she isn't commuting in Delft but rather to other cities.
create view out_of_delft as (
    select b.*
    from bluelog b join bluelog b2 on b."next" = b2.id
    where b2.starttime - b.endtime > '10:00:00'
);
```

```sql
-- Rush hour
-- Excluding view out_of_delft
select avg(b2.starttime - b.endtime) as rush_hour_avg_time
from bluelog b join bluelog b2 on b."next" = b2.id
where b.scanner <> b2.scanner
and (
    (date_part('hour', b.endtime) >= 7 and date_part('hour', b2.starttime) < 9)
    or (date_part('hour', b.endtime) >= 17 and date_part('hour', b2.starttime) < 19)
)
and b.id not in (
    select od.id
f   rom out_of_delft od
);
```

```sql
-- NON rush hour
-- Excluding view out_of_delft
select avg(b2.starttime - b.endtime) as non_rush_hour_avg_time
from bluelog b join bluelog b2 on b."next" = b2.id
where b.scanner <> b2.scanner
and not (
    (date_part('hour', b.endtime) >= 7 and date_part('hour', b2.starttime) < 9)
    or (date_part('hour', b.endtime) >= 17 and date_part('hour', b2.starttime) < 19)
)
and b.id not in (
    select od.id
    from out_of_delft od
);
```

Query result for rush hour

	rush_hour_avg_time
1	00:20:48.981935

Query result for non-rush hour

	non_rush_hour_avg_time
1	00:39:27.206561

3.3 Commuting time

Implement at least one other interesting SQL query on the dataset.

Question: How much time do Commuters spend on traveling to other cities? (assume commuters go out in the morning from 7:00 - 12:00 and go home after 17:00).

Answer:

```sql
-- Only counting trips in view out_of_delft
select avg(b2.starttime - b.endtime) as avg_time
from bluelog b join bluelog b2 on b."next" = b2.id
where (
    (date_part('hour', b.endtime) between 7 and 11) -- not after noon
    and (date_part('hour', b2.starttime) between 17 and 23) -- not after midnight
)
and b.id in (
    select od.id
    from out_of_delft od
)
and b2.starttime - b.endtime < '23:59:59';
```

	avg_time
1	11:29:05.735616

4 Question 4

Question: Please create a PL/pgSQL function 'isphone' that returns true if the device-type is a Phone and false otherwise. So that you can use the isPhone function in your scrips.

Answer:

```sql
-- SQL version
create or replace function isPhone(devicetype text) returns bool as $$
select
    case
        when substring(devicetype::int::bit(12)>>8,8) = B'00010' then true
        else false
    end;
$$ language SQL;


-- plpgsql version (with DECLARE, BEGIN, and END)
create or replace function isPhone(devicetype text) returns bool as $$
declare
begin
    return case
            when substring(devicetype::int::bit(12)>>8,8) = B'00010' then true
            else false
        end;
exception
    when invalid_text_representation then
    return false;
end;
$$ language plpgsql;

-- select isPhone
select isPhone(b.devicetype)
from bluelog b
```

	isphone
1	[v]
2	[v]
3	[v]
4	[v]
5	[v]
6	[v]
7	[v]
8	[v]
9	[v]
10	[v]

5 Question 5

Question: Write a PL/pgSQL function activityDuration that calculates the duration of device activity in minutes. The function should take the starttime and endtime as inputs and return the duration in minutes as an integer.

Answer:

```
create or replace function activityDuration(starttime timestamp , endtime timestamp) returns
↪  integer as $$
declare
    days integer;
    hours integer;
    minutes integer;
    time_diff timestamp;
begin
    time_diff := endtime - starttime;
    days := date_part('day', time_diff);
    hours := date_part('hour', time_diff);
    minutes := date_part('minute', time_diff);
    return days*3600 + hours*60 + minutes;
end;

-- example
select activityDuration('2012-10-06 12:11:57', '2012-10-07 15:24:03');
$$ language plpgsql;
```

	123 activityduration
1	3,792

6 Question 6

Question: Write a PL/pgSQL function mostCommonDeviceType that identifies the most frequently occurring device type in a given time frame. The function should accept two timestamps defining the start and end of the period and return the most common devicetype within that period.

Answer:

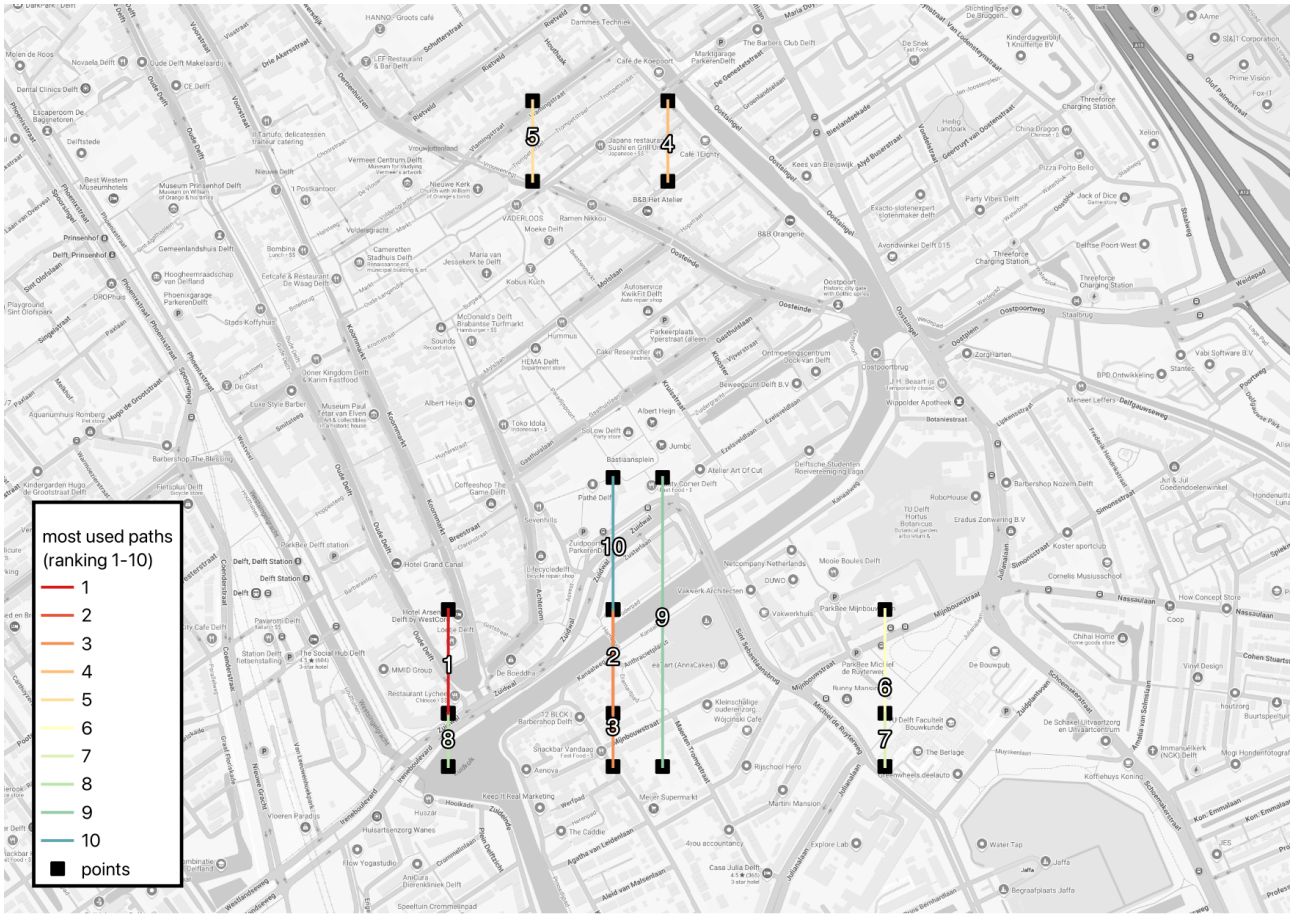
```
create or replace function mostCommonDeviceType(_starttime timestamp , _endtime timestamp)
↪  returns text as $$
declare
begin
    return b.devicetype
    from bluelog b
    where b.endtime > _starttime and b.starttime > _endtime
    group by b.devicetype
    order by count(b.devicetype) desc limit 1;
end;
$$ language plpgsql;

-- example
select mostCommonDeviceType('2012-10-06 12:13:57', '2012-10-06 12:14:03');
```

	mostcommondevicetype
1	5898756

7 Question 7

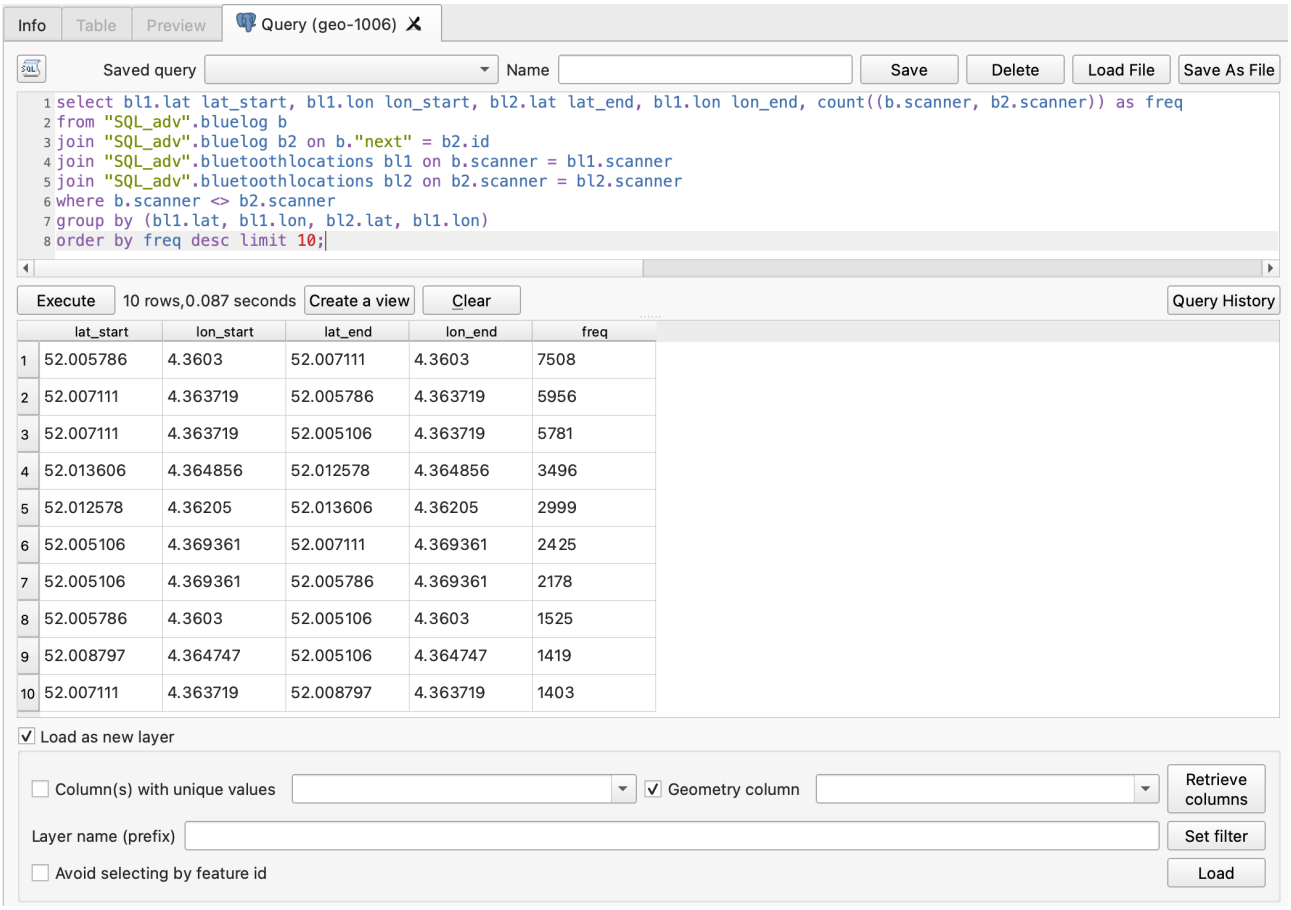
Question: Visualise the results.

Visualization: Top 10 most used routes in Delft
Since Google Fusion Tables and the Fusion Tables API have been discontinued, we decided to visualize the answer of question 3 using QGIS.




```sql
select bl1.lat lat_start, bl1.lon lon_start, bl2.lat lat_end, bl1.lon lon_end, count((b.scanner,
↪  b2.scanner)) as freq
from bluelog b
join bluelog b2 on b."next" = b2.id
join bluetoothlocations bl1 on b.scanner = bl1.scanner
join bluetoothlocations bl2 on b2.scanner = bl2.scanner
where b.scanner <> b2.scanner
group by (bl1.lat, bl1.lon, bl2.lat, bl1.lon)
order by freq desc limit 10;
```

Importing PostgreSQL into QGIS


Info
Table
Preview
Query (geo-1006)
Saved query
Name
Save
Delete
Load File
Save As File
select bl1.lat lat_start, bl1.lon lon_start, bl2.lat lat_end, bl1.lon lon_end, count((b.scanner, b2.scanner)) as freq
from "SQL_adv".bluelog b
join "SQL_adv".bluelog b2 on b."next" = b2.id
join "SQL_adv".bluetoothlocations bl1 on b.scanner = bl1.scanner
join "SQL_adv".bluetoothlocations bl2 on b2.scanner = bl2.scanner
where b.scanner <> b2.scanner
group by (bl1.lat, bl1.lon, bl2.lat, bl1.lon)
order by freq desc limit 10;
Execute
10 rows,0.087 seconds
Create a view
Clear
Query History
lat_start lon_start lat_end lon_end freq
1 52.005786 4.3603 52.007111 4.3603 7508
2 52.007111 4.363719 52.005786 4.363719 5956
3 52.007111 4.363719 52.005106 4.363719 5781
4 52.013606 4.364856 52.012578 4.364856 3496
5 52.012578 4.36205 52.013606 4.36205 2999
6 52.005106 4.369361 52.007111 4.369361 2425
7 52.005106 4.369361 52.005786 4.369361 2178
8 52.005786 4.3603 52.005106 4.3603 1525
9 52.008797 4.364747 52.005106 4.364747 1419
10 52.007111 4.363719 52.008797 4.363719 1403
Load as new layer
Column(s) with unique values
Geometry column
Retrieve columns
Layer name (prefix)
Set filter
Avoid selecting by feature id
Load