Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Определить ближайший дилерский центр для каждого клиента. Маркетологи пытаются повысить вовлеченность клиентов, помогая им найти ближайший к ним дилерский центр. Команда разработчиков также заинтересована в том, чтобы узнать, каково среднее расстояние между каждым покупателем и его ближайшим дилерским центром.

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

1. Во-первых, создадим таблицу с точками долготы и широты для каждого клиента:

```
CREATE TEMP TABLE customer_points AS (
SELECT
     customer_id,
     point(longitude, latitude) AS lng_lat_point
FROM customers
WHERE longitude IS NOT NULL
AND latitude IS NOT NULL
);
```

TEMP TABLE

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Как работает временная таблица в PostgreSQL?

•Временные таблицы невидимы для других транзакций и сеансов базы данных.

•Временные таблицы видны для текущей транзакции или сеанса базы данных, в котором мы создаем таблицу.

•Мы можем создать временную таблицу с тем же именем, что и постоянная таблица в базе данных, что на самом деле не рекомендуется. Временная таблица скрывает постоянную таблицу, определенную с тем же именем, пока она не будет удалена для соответствующего сеанса базы данных или транзакции.

•Мы не можем получить доступ к постоянной таблице, если у нас есть временная таблица с тем же именем, что и постоянная таблица.

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

2. Проверить данные во временной таблице

```
SELECT * FROM customer_points;
```

```
customer_id|lng_lat_point                 |
-----------+-----------------------------+
          2|(-90.2625,38.5814)           |
          3|(-87.2758,30.6143)           |
          4|(-86.8219,36.0986)           |
          5|(-80.4582,25.5584)           |
          6|(-80.3187,25.6364)           |
          7|(-81.2608,28.5663)           |
          8|(-72.9271,41.3087)           |
          9|(-94.83200000000001,38.8999) |
         10|(-106.3,31.6948)             |
         11|(-95.9605,36.3024)           |
         13|(-116.846,33.0169)           |
         14|(-92.3289,34.8337)           |
         15|(-117.8892,33.9187)          |
         16|(-97.0727,33.1428)           |
```

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

3. Создать аналогичную таблицу для каждого дилерского центра:

```
SELECT * FROM customer_points;
CREATE TEMP TABLE dealership_points AS (
SELECT
dealership_id,
point(longitude, latitude) AS lng_lat_point
FROM dealerships
);

SELECT * FROM dealership_points;
```

dealership_id	lng_lat_point
1	(-74.323291,40.79284600000004)
2	(-118.305423,34.057753999999996)
3	(-95.30702,29.963501)
4	(-80.23645400000001,25.801748)
5	(-122.34360900000001,37.524487)
6	(-122.38729,47.537959)
7	(-77.07974,38.839180999999996)
8	(-122.602984,45.575702)
9	(-119.775709,39.527681)
10	(-87.813599,41.963474)
11	(-84.439297,33.789649)
12	(-81.372232,28.520723999999998)
13	(-81.410156,30.316027000000002)
14	(-97.681912,30.515476)
15	(-112.36994299999999,33.623799)

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

3. Объединить эти таблицы, чтобы рассчитать расстояние от каждого клиента до каждого дилерского центра (в милях):

```
CREATE TEMP TABLE customer_dealership_distance AS (
SELECT
      customer_id,
      dealership_id,
      c.lng_lat_point <@> d.lng_lat_point AS distance
FROM customer_points c
CROSS JOIN dealership_points d
);

SELECT * FROM customer_dealership_distance;
```

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

3. Объединить эти таблицы, чтобы рассчитать расстояние от каждого клиента до каждого дилерского центра (в милях):

customer_id	dealership_id	distance
2	1	859.8940793201315
2	2	1585.647794965087
2	3	661.1449933095291
2	4	1058.283498390192
2	5	1738.0755753840901
2	6	1720.7866866604265
2	7	710.3506330216096
2	8	1714.4120709491135
2	9	1577.7526240301158
2	10	266.93721034903587
2	11	463.5443671619498
2	12	862.282848174378
2	13	761.117994881253
2	14	698.6164270950409

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

4. Выбрать ближайший дилерский центр для каждого клиента, используя следующий запрос:

```
CREATE TEMP TABLE closest_dealerships AS (
SELECT DISTINCT ON (customer_id)
     customer_id,
     dealership_id,
     distance
FROM customer_dealership_distance
ORDER BY customer_id, distance
);
SELECT * FROM closest_dealerships;
```

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Шаги для выполнения запроса PostgreSQL

4. Выбрать ближайший дилерский центр для каждого клиента, используя следующий запрос:

customer_id	dealership_id	distance
2	18	231.80698588435425
3	11	274.9885104949864
4	11	208.93337510454796
5	4	21.75674247468588
6	4	12.51912084529485
7	12	7.4605341973673704
8	1	81.00919920405966
9	18	21.56261227624718
10	15	377.32135494582764
11	18	208.45090235163937
13	2	110.6149374875428
14	19	291.6134662380013
15	2	25.707699223220462

Practice 06-2. Геопространственный анализ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Самостоятельное задание

- Провести выгрузку полученного результата из временной таблицы в CSV.
- Построить карту клиентов и сервисных центров в облачной визуализации Yandex DataLence.
- Удалить временные таблицы.

customer_id	dealership_id	distance
2	18	231.80698588435425
3	11	274.9885104949864
4	11	208.93337510454796
5	4	21.75674247468588
6	4	12.51912084529485
7	12	7.4605341973673704
8	1	81.00919920405966
9	18	21.56261227624718
10	15	377.32135494582764
11	18	208.45090235163937
13	2	110.6149374875428
14	19	291.6134662380013
15	2	25.707699223220462