date_time_components_aggregation

```sql
-- Extract the month from date_created and count requests
SELECT date_part('month', date_created) AS month,
    COUNT(*) AS request_count
 FROM evanston311
-- Limit the date range
WHERE date_created >= '2016-01-01'
  AND date_created < '2018-01-01'
-- Group by what to get monthly counts?
GROUP BY month
ORDER BY month;
```

Explanation:

- This SQL query extracts the month from a date_created column in the evanston311 table, counts the number of requests for each month, and then groups the results by month. The WHERE clause filters the data to include only requests between January 1st, 2016 and January 1st, 2018. The date_part function extracts the month from the date. Finally, it orders the results by month for better readability. The added request_count alias makes the output more descriptive.

```sql
-- Get the hour and count requests
SELECT date_part('hour', date_created) AS hour,
    count(*)
 FROM evanston311
GROUP BY hour
-- Order results to select most common
ORDER BY count DESC
LIMIT 1;
```

Explanation:

- This SQL query determines the hour of the day with the most 311 requests in the evanston311 table. It extracts the hour from the date_created column, groups the results by hour, counts the requests within each hour, orders the hours by request count in descending order, and finally returns only the top hour (the one with the most requests).

```sql
-- Count requests completed by hour
SELECT date_part('hour', date_completed) AS hour, -- Extracts the hour from the date_completed column and aliases it as "hour"
    COUNT(*)                    -- Counts the number of rows in each group
 FROM evanston311                    -- Specifies the table to query
```

```
GROUP BY hour                    -- Groups the results by the extracted hour
ORDER BY hour;                   -- Orders the results by hour in ascending order
```

Explanation:

- This SQL query calculates the number of completed requests for each hour of the day from the evanston311 table. It extracts the hour from the date_completed column, groups the data by hour, and then counts the number of requests within each hour group. Finally, it orders the results chronologically by hour.

```
-- Select name of the day of the week the request was created
SELECT to_char(date_created, 'day') AS day,
    -- Select avg time between request creation and completion
    AVG(date_completed - date_created) AS duration
 FROM evanston311
-- Group by the name of the day of the week and
-- integer value of day of week the request was created
GROUP BY day, EXTRACT(DOW FROM date_created)
-- Order by integer value of the day of the week
-- the request was created
ORDER BY EXTRACT(DOW FROM date_created);
```

Explanation:

- This SQL query calculates the average time it takes to complete requests in the evanston311 table, broken down by the day of the week the request was created. It uses to_char to get the day name, AVG to compute the average duration (difference between completion and creation dates), GROUP BY to aggregate results by day name and day number, and ORDER BY to present the results in order of the day of the week (Sunday=0, Saturday=6).

```
-- Aggregate daily counts by month
SELECT date_trunc('month', day) AS month,
    AVG(count)
 -- Subquery to compute daily counts
 FROM (SELECT date_trunc('day', date_created) AS day,
         COUNT(*) AS count
      FROM evanston311
      GROUP BY day) AS daily_count
GROUP BY month
ORDER BY month;
```

Explanation:

This SQL query calculates the average number of records per day for each month. It does this in two steps:

1. **Inner Query:** It first groups the evanston311 table by day (using date_trunc('day', date_created) to extract the date portion), counting the number of records for each day.
2. **Outer Query:** It then takes the results of the inner query and groups them by month (using date_trunc('month', day)), calculating the average daily count for each month using the AVG() function. The results are then ordered by month.