Sample SQL Queries from Thesis

```sql
DELETE FROM dbo.AirbnbONLYneedLONDON
WHERE Market <> 'London';
```

```sql
SELECT DISTINCT area FROM dbo.housing_in_london_yearly_variables
WHERE area NOT IN (
    SELECT G.NAME
    FROM dbo.GIS_London_Polygons_Names AS G
    WHERE G.NAME IS NOT NULL
);
```

```sql
DELETE FROM dbo.housing_in_london_yearly_variables
WHERE area NOT IN (
    SELECT G.NAME
    FROM dbo.GIS_London_Polygons_Names AS G
    WHERE G.NAME IS NOT NULL
);
```

```sql
SELECT COUNT (Price) AS Policzone,
"Neighbourhood Cleansed"
FROM dbo.AirbnbONLYneedLONDON
GROUP BY "Neighbourhood Cleansed"
ORDER BY Policzone;
```

```sql
DELETE FROM dbo.AirbnbONLYneedLONDON
WHERE [Neighbourhood Cleansed] NOT IN (
    SELECT G.NAME
    FROM dbo.GIS_London_Polygons_Names AS G
    WHERE G.NAME IS NOT NULL
);
```

```sql
SELECT DISTINCT M.area, Y.area, G.NAME, L.borough, A.[Neighbourhood Cleansed]
FROM dbo.housing_in_london_monthly_variables M
JOIN dbo.GIS_London_Polygons_Names G ON M.area = G.NAME
JOIN dbo.housing_in_london_yearly_variables Y ON G.NAME = Y.area
JOIN dbo.london_crime_by_lsoa L ON G.NAME = L.borough
JOIN dbo.AirbnbONLYneedLONDON A ON G.NAME = A.[Neighbourhood Cleansed]
ORDER BY M.area, Y.area, G.Name, L.borough ASC;
```

```sql
SELECT A.ID, A.[Neighbourhood Cleansed], G.GSS_CODE, A.Latitude, A.Longitude, A.[Property Type],
    A.[Room Type], A.Accommodates, A.Bathrooms, A.Bedrooms, A.Beds, A.Price, A.[Review Scores Rating],
    A.[Reviews per Month]
INTO dbo.AirBnbGSS_Code
FROM dbo.AirbnbONLYneedLONDON A
JOIN dbo.GIS_London_Polygons_Names G ON G.NAME = A.[Neighbourhood Cleansed];

SELECT M.date, M.area, M.average_price, M.houses_sold, M.no_of_crimes,
G.GSS_CODE
INTO dbo.housing_monthly_GSS_Code
FROM dbo.housing_in_london_monthly_variables M
JOIN dbo.GIS_London_Polygons_Names G ON G.NAME = M.area;

SELECT Y.area, G.GSS_CODE, Y.date, Y.median_salary, Y.life_satisfaction, Y.mean_salary, Y.recycling_pct,
Y.population_size, Y.number_of_jobs, Y.area_size, Y.no_of_houses
INTO dbo.housing_yearly_GSS_Code
FROM dbo.housing_in_london_yearly_variables Y
JOIN dbo.GIS_London_Polygons_Names G ON G.NAME = Y.area;

SELECT C.borough, G.GSS_CODE, C.major_category,
C.minor_category, C.value, C.year, C.month
INTO dbo.crime_GSS_Code
FROM dbo.london_crime_by_lsoa C
JOIN dbo.GIS_London_Polygons_Names G ON C.borough = G.NAME;
```

```sql
UPDATE dbo.AirBnbGSS_Code
SET [Reviews per Month] = NULL
WHERE [Reviews per Month] = ' ';
```

```sql
UPDATE dbo.AirBnbGSS_Code
SET [Reviews per Month] = NULL
WHERE TRY_CAST([Reviews per Month] AS FLOAT) IS NULL
   AND [Reviews per Month] IS NOT NULL;
```

```sql
--Dodaj nową kolumnę do tabeli
ALTER TABLE dbo.AirBnbGSS_Code
ADD AverageRevenueMonth FLOAT;

--Ustawienie wartości dla nowej kolumny na podstawie warunku:
UPDATE dbo.AirBnbGSS_Code
SET AverageRevenueMonth = [Reviews per Month] * 10 * Price
WHERE [Reviews per Month] > 0;
```

```sql
-- Zmiana znaków # na NULL
UPDATE dbo.housing_yearly_GSS_Code
SET mean_salary = NULL
WHERE mean_salary = '#';

-- Zmiana typu danych z VAR na INT
ALTER TABLE dbo.housing_yearly_GSS_Code
ALTER COLUMN mean_salary INT;
```

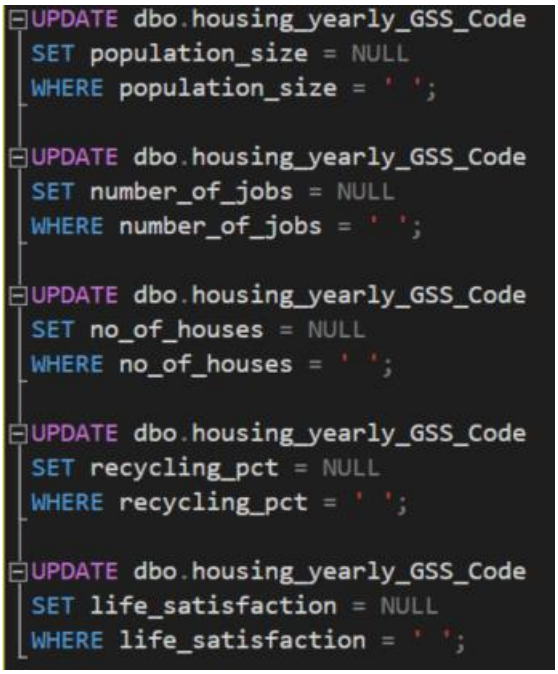
```sql
SELECT area, year,
COALESCE(mean_salary, (LAG(mean_salary) OVER (ORDER BY area) +
                       LEAD(mean_salary) OVER (ORDER BY area)) / 2) AS mean_salary,
LAG(mean_salary) OVER (ORDER BY area) AS Preview,
LEAD(mean_salary) OVER (ORDER BY area) AS Future
FROM dbo.housing_yearly_GSS_Code;
```

```sql
WITH Podzapytanie AS (
SELECT mean_salary,
       LAG(mean_salary) OVER(ORDER BY area) AS Previous,
       LEAD(mean_salary) OVER(ORDER BY area) AS Future
FROM dbo.housing_yearly_GSS_Code
)

UPDATE Podzapytanie
SET mean_salary = COALESCE(mean_salary, (Previous + Future) / 2);
```

```sql
UPDATE dbo.housing_yearly_GSS_Code
SET mean_salary = 37193
WHERE area IN ('hackney') AND year = 2000;

UPDATE dbo.housing_yearly_GSS_Code
SET mean_salary = 34757
WHERE area IN ('hackney') AND year = 2001;
```

```sql
-- Zamieniamy puste wartości na NULL
UPDATE dbo.housing_yearly_GSS_Code
SET median_salary = NULL
WHERE median_salary = ' ';

-- Zmieniamy typ danych z VARCHAR na INTEGER
ALTER TABLE dbo.housing_yearly_GSS_Code
ALTER COLUMN median_salary INT;
```

```sql
WITH Podzapytanie AS (
SELECT median_salary,
       LAG(median_salary) OVER(ORDER BY area) AS Previous,
       LEAD(median_salary) OVER(ORDER BY area) AS Future
FROM dbo.housing_yearly_GSS_Code
)

UPDATE Podzapytanie
SET median_salary = COALESCE(median_salary, (Previous + Future) / 2);
```


UPDATE dbo.housing_yearly_GSS_Code
SET population_size = NULL
WHERE population_size = ' ';
UPDATE dbo.housing_yearly_GSS_Code
SET number_of_jobs = NULL
WHERE number_of_jobs = ' ';
UPDATE dbo.housing_yearly_GSS_Code
SET no_of_houses = NULL
WHERE no_of_houses = ' ';
UPDATE dbo.housing_yearly_GSS_Code
SET recycling_pct = NULL
WHERE recycling_pct = ' ';
UPDATE dbo.housing_yearly_GSS_Code
SET life_satisfaction = NULL
WHERE life_satisfaction = ' ';

=((Population_2018 – Population_2017) / Population_2018 + 1) * Population_2018

Population_2018

```
LAG(population_size) OVER (PARTITION BY area ORDER BY year ASC) AS Preview
```

Population_2017

```
LAG(population_size, 2) OVER (PARTITION BY area ORDER BY year ASC) AS Preview2
```

EKSPLORACJA PROBLEMU

```
;WITH virtual_table AS (
      SELECT area, year, population_size
      FROM dbo.housing_yearly_GSS_Code
      WHERE year IN (2017,2018,2019)
      GROUP BY area, year, population_size),
virtual_table2 AS (
      SELECT area, year, population_size,
      LAG(population_size) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview,
      LAG(population_size, 2) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview2
      FROM virtual_table)

SELECT area, year, population_size,
COALESCE(population_size,
((Preview - Preview2)/Preview + 1) *
Preview, 0) AS Kalkulacja
FROM virtual_table2
ORDER BY area, year ASC;
```

To jest tylko zapytanie które nie wprowadza zmian w bazie danych, lecz pozwala wyświelić informacje w oczekiwany sposób i umożliwić dokonanie analizy

	area	year	population_size	Kalkulacja
1	barking and dagenham	2017	210711	210711.000000
2	barking and dagenham	2018	211998	211998.000000
3	barking and dagenham	2019	NULL	213285.000000
4	barnet	2017	387803	387803.000000
5	barnet	2018	392140	392140.000000
6	barnet	2019	NULL	396477.000000

PROBLEM

ROZWIĄZANIE NA BAZIE DANYCH

```
;WITH virtual_table AS (
      SELECT area, year, population_size
      FROM dbo.housing_yearly_GSS_Code
      WHERE year IN (2017,2018,2019)
      GROUP BY area, year, population_size),
virtual_table2 AS (
      SELECT area, year, population_size,
      LAG(population_size) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview,
      LAG(population_size, 2) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview2
      FROM virtual_table),
virtual_table3 AS (
      SELECT area, year, population_size,
      COALESCE(population_size,
      ((Preview - Preview2)/Preview + 1) *
      Preview, 0) AS Kalkulacja
      FROM virtual_table2)

UPDATE dbo.housing_yearly_GSS_Code
SET population_size = vt3.Kalkulacja
FROM dbo.housing_yearly_GSS_Code h
JOIN virtual_table3 vt3 ON
h.area = vt3.area AND h.year = vt3.year
WHERE h.year = 2019;
```

REZULTAT

	area	year	population_size
1	barking and dagenham	2017	210711
2	barking and dagenham	2018	211998
3	barking and dagenham	2019	213285
4	barnet	2017	387803
5	barnet	2018	392140
6	barnet	2019	396477

ROZWIĄZANY

= ((((number_of_jobs_2018 - number_of_jobs_2000) / number_of_jobs_2000) / 18) + 1) * number_of_jobs_2018

number_of_jobs_2018

LAG(number_of_jobs) OVER (PARTITION BY area ORDER BY year ASC) AS Preview3

number_of_jobs_2000

LAG(number_of_jobs,2) OVER (PARTITION BY area ORDER BY year ASC) AS Preview4

	area	year	number_of_jobs	Kalkulacja
1	barking and dagenham	2000	57000	57000.000000
2	barking and dagenham	2018	66000	66000.000000
3	barking and dagenham	2019	NULL	66578.947368
4	barnet	2000	138000	138000.000000
5	barnet	2018	170000	170000.000000
6	barnet	2019	NULL	172190.016103

PROBLEM ↗

REZULTAT

	area	year	number_of_jobs
1	barking and dagenham	2000	57000
2	barking and dagenham	2018	66000
3	barking and dagenham	2019	66579
4	barnet	2000	138000
5	barnet	2018	170000
6	barnet	2019	172190

ROZWIĄZANIE

```
;WITH virtual_table4 AS (
      SELECT area,year, number_of_jobs
      FROM dbo.housing_yearly_GSS_Code
      WHERE year IN (2000, 2018, 2019)
      GROUP BY area, year, number_of_jobs),
virtual_table5 AS (
      SELECT area, year, number_of_jobs,
      LAG(number_of_jobs) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview3,
      LAG(number_of_jobs,2) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview4
      FROM virtual_table4),
virtual_table6 AS(
      SELECT area, year, number_of_jobs,
      COALESCE(number_of_jobs, ((((Preview3 - Preview4)/
      Preview4)/18)+1)*Preview3,0) AS Kalkulacja
      FROM virtual_table5)
UPDATE dbo.housing_yearly_GSS_Code
SET number_of_jobs = vt6.Kalkulacja
FROM dbo.housing_yearly_GSS_Code h
JOIN virtual_table6 vt6 ON
h.area = vt6.area AND h.year = vt6.year
WHERE h.year = 2019;
```

ROZWIĄZANIE 1999

```
;WITH virtual_table7 AS (
      SELECT area,year, number_of_jobs
      FROM dbo.housing_yearly_GSS_Code
      WHERE year IN (1999, 2000, 2018)
      GROUP BY area, year, number_of_jobs),
virtual_table8 AS (
      SELECT area, year, number_of_jobs,
      LAG(number_of_jobs) OVER
      (PARTITION BY area ORDER BY year DESC) AS Preview5,
      LAG(number_of_jobs,2) OVER
      (PARTITION BY area ORDER BY year DESC) AS Preview6
      FROM virtual_table7),
virtual_table9 AS(
      SELECT area, year, number_of_jobs,
      COALESCE(number_of_jobs, ((((Preview3 - Preview4)/
      Preview4)/18)+1)*Preview3,0) AS Kalkulacja
      FROM virtual_table8)
UPDATE dbo.housing_yearly_GSS_Code
SET number_of_jobs = vt9.Kalkulacja
FROM dbo.housing_yearly_GSS_Code h
JOIN virtual_table9 vt9 ON h.area = vt9.area AND h.year = vt9.year
WHERE h.year = 1999;
```

POPRZEDNIE ROZWIĄZANIE 2019

```
;WITH virtual_table4 AS (
      SELECT area,year, number_of_jobs
      FROM dbo.housing_yearly_GSS_Code
      WHERE year IN (2000, 2018, 2019)
      GROUP BY area, year, number_of_jobs),
virtual_table5 AS (
      SELECT area, year, number_of_jobs,
      LAG(number_of_jobs) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview3,
      LAG(number_of_jobs,2) OVER
      (PARTITION BY area ORDER BY year ASC) AS Preview4
      FROM virtual_table4),
virtual_table6 AS(
      SELECT area, year, number_of_jobs,
      COALESCE(number_of_jobs, ((((Preview3 - Preview4)/
      Preview4)/18)+1)*Preview3,0) AS Kalkulacja
      FROM virtual_table5)
UPDATE dbo.housing_yearly_GSS_Code
SET number_of_jobs = vt6.Kalkulacja
FROM dbo.housing_yearly_GSS_Code h
JOIN virtual_table6 vt6 ON
h.area = vt6.area AND h.year = vt6.year
WHERE h.year = 2019;
```

REZULTAT

	area	number_of_jobs	year
1	barking and dagenham	56568	1999
2	barking and dagenham	57000	2000
3	barking and dagenham	66000	2018
4	barnet	136557	1999
5	barnet	138000	2000
6	barnet	170000	2018

```
WHERE year IN (2000, 2018, 2019)
WHERE year IN (1999, 2000, 2018)
```
Zmiana zawartości okien

```
ORDER BY year ASC
ORDER BY year DESC
```
Zmiana kierunku filtracji wewnątrz okien

```
WHERE h.year = 1999;
WHERE h.year = 2019;
```
Zmiana lokalizacji wprowadzenia rezultatu kodu

```
UPDATE dbo.housing_yearly_GSS_Code
SET number_of_jobs = ROUND(number_of_jobs / 1000.0, 0) * 1000;
```

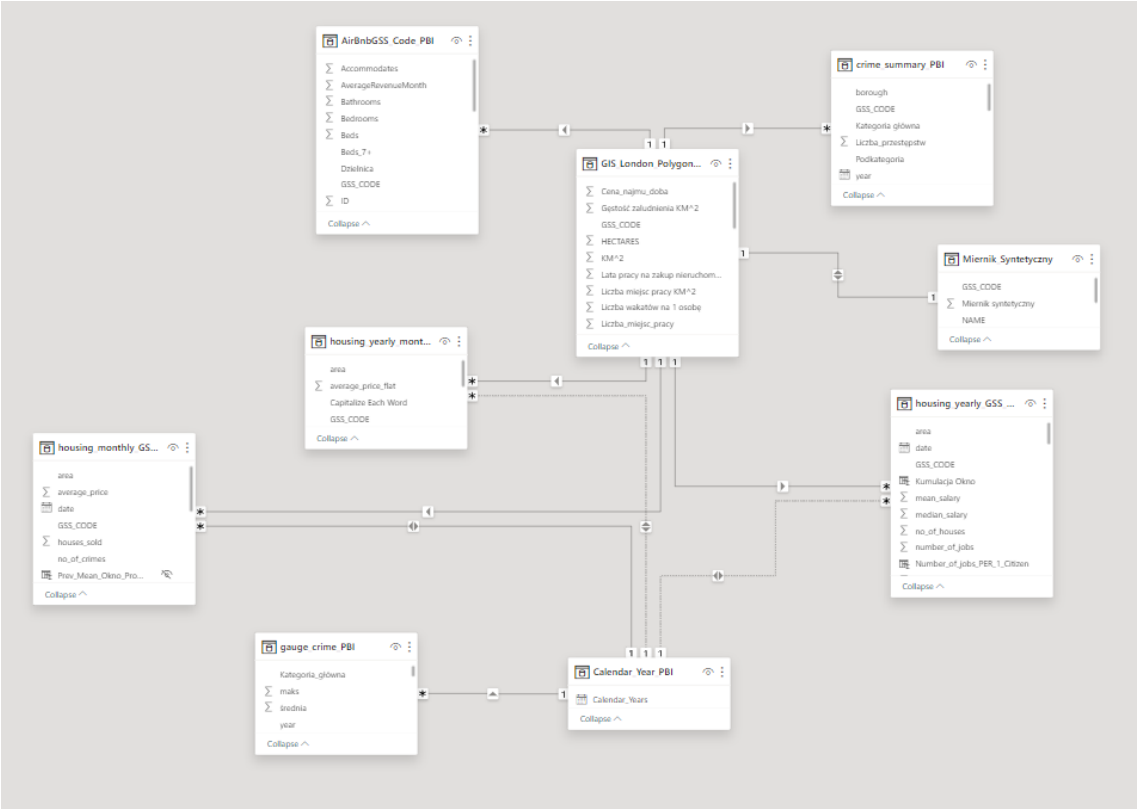
```sql
SELECT L.NAME, H.GSS_CODE, H.life_satisfaction
INTO dbo.GIS_London_Polygons_Names_Updated
FROM dbo.housing_yearly_GSS_Code H
JOIN dbo.GIS_London_Polygons_Names L ON H.GSS_CODE = L.GSS_CODE
WHERE H.year = 2018;

ALTER TABLE dbo.housing_yearly_GSS_Code
DROP COLUMN area_size,
DROP COLUMN life_satisfaction;
```

```sql
SELECT *
INTO AirBnbGSS_Code_PBI
FROM dbo.AirBnbGSS_Code
WHERE AverageRevenueMonth > 0;
```

```sql
;WITH CTE AS (
       SELECT Price, NTILE(7) OVER (ORDER BY Price) AS Grupy
       FROM dbo.AirBnbGSS_Code_PBI),
      CTE2 AS (
       SELECT MAX(Price) AS Obliczone, Grupy
       FROM CTE
       GROUP BY Grupy)
SELECT *
FROM CTE2;
```

	Obliczone	Grupy
1	35	1
2	45	2
3	61	3
4	81	4
5	103	5
6	150	6
7	999	7

```sql
CREATE TABLE housing_yearly_monthly_PBI (
    area VARCHAR(100),
    GSS_CODE VARCHAR(10),
    year INT,
    average_price_flat DECIMAL(18, 2)
);

INSERT INTO dbo.housing_yearly_monthly_PBI (area, GSS_CODE, year, average_price_flat)
SELECT area, GSS_CODE, year, SUM(sumowanie)/SUM(houses_sold) AS average_price_flat
FROM (
    SELECT area, average_price, houses_sold, GSS_CODE, (average_price * houses_sold)
AS sumowanie, year
    FROM dbo.housing_monthly_GSS_Code_PBI
) AS cte
GROUP BY area, GSS_CODE, year
ORDER BY area, GSS_CODE, year;
```




```sql
CREATE TABLE crime_summary_PBI (
    borough VARCHAR(50),
    GSS_CODE VARCHAR(10),
    major_category VARCHAR(50),
    minor_category VARCHAR(50),
    count_crime INT,
    year INT
);

INSERT INTO dbo.crime_summary_PBI (borough, GSS_CODE, major_category, minor_category,
count_crime, year)
SELECT borough, GSS_CODE, major_category, minor_category, SUM(value) AS count_crime,
year
FROM dbo.crime_GSS_Code_PBI
WHERE minor_category NOT IN ('Rape', 'Other Sexual', 'Counted per Victim', 'Other
Fraud & Forgery')
GROUP BY borough, major_category, minor_category, year, GSS_CODE
ORDER BY year, borough, count_crime;
```

```sql
;WITH cte AS (
       SELECT borough, major_category, year, SUM(count_crime) AS suma
       FROM dbo.crime_summary_PBI
       GROUP BY major_category, borough, year)
SELECT major_category, year,
MAX(suma) AS maksymalne, AVG(suma) AS srednia
FROM cte
GROUP BY major_category, year
ORDER BY year ASC;
```

```sql
;WITH cte1 AS(
    SELECT borough, GSS_CODE, year, SUM(count_crime) AS suma_przestepstw
    FROM dbo.crime_summary_PBI
    GROUP BY borough, year, GSS_CODE),
cte2 AS ( SELECT area, GSS_CODE, population_size, year
        FROM dbo.housing_yearly_GSS_Code_PBI),
cte3 AS (
    SELECT (suma_przestepstw * 1000)/population_size AS dzialanie, borough, c1.year
    FROM cte1 c1
    JOIN cte2 c2 ON c1.GSS_CODE = c2.GSS_CODE and c1.year = c2.year)
SELECT MAX(dzialanie) AS maksymalna_wartosc, year
FROM cte3
GROUP BY year
ORDER BY year ASC;
```