Database Project Report

Traffic Flow Analysis for Urban Planning in Zurich | Team ACID

Ramiro Díez-Liébana, Valeska Blank, Dongyuan Gao, Cyriel Van Helleputte

2025-11-22

Contents

1 Introduction & Context (Proposal)

The City of Zurich faces growing challenges related to traffic congestion, mobility planning, and urban development. As the population increases and commuting patterns evolve, many intersections and arterial roads experience significant pressure, particularly during peak hours. Congestion affects not only travel times but also road safety, air quality, and the overall effectiveness of the city's transport network.

Urban planning authorities must therefore make complex decisions about where to prioritize infrastructure investments, how to optimize traffic signals, and which areas require redesign or alternative mobility solutions. These decisions rely heavily on an accurate understanding of traffic flows, temporal patterns, directional imbalances, and long-term trends. High-quality traffic data

and transparent analytical methods are essential to support evidence-based planning, enable more targeted interventions, and ensure that resources are allocated efficiently.

To contribute to this goal, this project focuses on deriving meaningful insights and key performance indicators (KPIs) that can inform data-driven urban planning decisions in line with the OECD Data Value Cycle, which emphasizes the transformation of raw data into actionable value for public administration.

2 Project Idea & Use Case (Proposal)

To support Zurich's urban planners in addressing congestion and mobility challenges, this project aims to transform traffic count data into meaningful indicators that can guide evidence-based decision-making. The central use case focuses on identifying traffic pressure points at intersections and understanding how traffic patterns vary by time, location, and direction.

3 Key Performance Indicators (KPIs)

The KPI suite was first developed collaboratively by the team to ensure that every indicator directly supports the Zurich urban-planning use case. The procedure followed three steps: (1) define the analytical questions for districts and counting sites, (2) design SQL-ready calculations and classification thresholds, and (3) allow minor adjustments later in the project as additional data nuances emerged. This predefined framework now guides all downstream analytics and remains stable unless new evidence requires a targeted refinement.

3.1 KPI 1: District-Level Stress Index

Purpose. Compare traffic volume growth with population growth for each district to detect commuter hubs and residential pressure zones.

KPI 1.1 – Aggregate Population by District & Year. Join `Population` with `Quarter` and sum `population_count` per `city_district` for 2012 vs. 2025 (standard `SELECT city_district, SUM(population_count)` grouped by district and year).

KPI 1.2 – Aggregate Traffic Volume by District & Year. Use the `TrafficMeasurement → MeasurementSite → CountingSite → Quarter` join path to sum measured hourly `vehicle_count` per district for 2012 and 2025 (group by `city_district` and `YEAR(timestamp)`).

KPI 1.3 – Calculate Growth Rates.

- Population Growth (%) = ((Pop_2025 - Pop_2012) / Pop_2012) × 100
- Traffic Growth (%) = ((Traffic_2025 - Traffic_2012) / Traffic_2012) × 100

KPI 1.4 – Stress Index. `Stress Index = Traffic Growth % - Population Growth %`

Interpretation: positive values highlight commuter hubs; negative values show residential pressure; near zero indicates balanced growth.

KPI 1.5 – Classification.

- Stress Index > +10% : High commuter pressure
- Stress Index < -10% : High residential pressure
- -10% <= Stress Index <= +10% : Balanced

Sample output:

District	Pop Growth %	Traffic Growth %	Stress Index	Classification
1	15.2%	8.3%	-6.9%	Balanced
2	12.5%	22.1%	+9.6%	Commuter Hub

3.2 KPI 2: Peak-Hour Bottleneck Identification *(only if deeper analysis is required)*

Purpose. Detect counting sites where peak-hour traffic consistently exceeds capacity.

KPI 2.1 – Average Hourly Traffic per Counting Site. Calculate `AVG(vehicle_count)` grouped by `counting_site_id` and `HOUR(timestamp)` for measured rows from 2023–2025.

KPI 2.2 – Peak Hour per Site. For each counting site (e.g., Seestrasse, Wollishofen), pick the hour with the maximum average volume.

KPI 2.3 – Bottleneck Threshold. Flag sites where peak-hour volume exceeds 800 vehicles/hour. Example:

Counting Site Name	Peak Hour	Avg Volume	Status
Hardbrücke	08:00	1,245	Bottleneck
Bellevue	17:00	1,102	Bottleneck
Seestrasse (Wollishofen)	07:00	650	Normal

3.3 KPI 3: Directional Imbalance *(optional, counting-site level)*

Purpose. Reveal inbound vs. outbound inbalance using `MeasurementSite.direction`. Aggregate inbound and outbound totals per counting site (e.g., `SUM(CASE WHEN direction='inbound' ...)`), compute the ratio, and highlight sites where the inbound/outbound ratio is above 1.5 or below 0.67.

3.4 KPI 4: Dashboard & Visualization Plan

- **Visual 1: District Stress Index Map** – Heatmap based on KPI 1 results.
- **Visual 2: Peak-Hour Heatmap** – Time-of-day heatmap per counting site using KPI 2 outputs.
- **Visual 3: Growth Trend Comparison** – Grouped bar chart/scatterplot (population vs. traffic growth per district).
- **Additional Visuals** – e.g., Any additional visualizations that the team discovers while doing the analysis.

3.5 KPI 5: Provide Actionable Insights for Planning Decisions

Purpose. Convert KPI outputs into budget and intervention guidance for the Urban Planning Office.

KPI 5.1 – District Priorities.

Priority	Stress Index Range	Recommended Action
High	< -15% (residential pressure)	Expand public transit, add lanes, optimize signals
High	> +15% (commuter pressure)	Improve inbound capacity, park-and-ride facilities
Medium	-15% to -10% or +10% to +15%	Monitor trends, reassess in two years
Low	-10% to +10% (balanced)	Maintain current infrastructure

KPI 5.2 – Site(street)-Level Interventions.

- Top five bottleneck streets : immediate signal-timing optimization or capacity studies.
- Streets with directional imbalance > 1.5 : directional lane adjustments or reversible lanes.

KPI 5.3 – Budget Allocation Guide.

1. Short-term (1–2 years): immediate action on the top ten bottleneck intersections flagged in KPI 2.
2. Long-term (3–5 years): invest in infrastructure with persistent stress index extremes.

Together, these five KPIs provide a structured, predefined guideline for our project and analysis.

4 Data Model & Database Schema

4.1 Data Preprocessing & Preparation

- Hourly traffic count data, address data (inkluding city district and quarter) and population data from the City of Zurich's open data portal was collected.
- The raw datasets consisted of multiple CSV files (2012–2025).
- All CSV files were loaded with Python using the Pandas library.

4.1.1 Raw Dataset Columns: Traffic Data

- MSID
- MSName
- ZSID
- ZSName
- Achse
- HNr
- Hoehe
- EKoord
- NKoord
- Richtung
- Knummer
- Kname
- AnzDetektoren
- D1ID–D4ID
- MessungDatZeit
- LieferDat
- AnzFahrzeuge

- AnzFahrzeugeStatus

Because the raw dataset included a mixture of analytical attributes and highly technical metadata, each field was examined to determine its relevance for traffic-flow analysis. The review was guided by the perspective of the City of Zurich's urban planning department, which is primarily interested in temporal and spatial traffic patterns at specific locations and intersections.

4.1.2 Removed Fields

- **HNr**, due to inconsistent content.
- **D1ID–D4ID**, as these detector identifiers are technical metadata without analytical value.
- **LieferDat**, which is a delivery timestamp not required for traffic analysis.

All fields describing the measurement location, the measurement configuration, and the traffic counts themselves were retained. To improve clarity and support further processing and database integration, the remaining column names were translated into English equivalents. In a further preprocessing step, several categorical values contained in German were translated to English. After all preprocessing steps were completed, the resulting cleaned dataset contained 21,721,493 rows and 14 columns.

4.1.3 Cleaned Dataset Columns

- **measurement_site_id**: Unique technical identifier of the measurement site. A measurement site represents a specific traffic-flow direction or lane at a counting location.
- **measurement_site_name**: Technical name of the measurement site. In this dataset, this field contains "Unknown" for all entries.
- **counting_site_id**: Identifier of the counting site, representing the physical location where traffic measurements are collected.
- **counting_site_name**: Human-readable name of the counting site, describing the location (street).
- **axis**: Categorization of the counting site into a traffic axis (street).
- **position_description**: A textual descriptor indicating where along the street segment the measurement site is located. Contains "Unknown" for many entries.
- **east_coordinate**: The east coordinate of the measurement site in the Swiss CH1903+ / LV95 reference system.
- **north_coordinate**: The north coordinate of the measurement site in the Swiss CH1903+ / LV95 reference system.
- **direction**: The direction of traffic flow being measured (e.g., "inbound", "outbound").
- **signal_id**: Identifier of the associated traffic signal or intersection controller regulating traffic at the measurement site.
- **signal_name**: Name of the associated traffic signal or intersection.
- **num_detectors**: Number of detectors installed at the measurement site.
- **timestamp**: The timestamp indicating the start of the hourly measurement interval (ISO-8601 format).
- **vehicle_count**: The number of vehicles recorded during the hourly measurement interval.
- **vehicle_count_status**: Indicates how the vehicle count was produced: "Measured", "Missing", or "Imputed".

4.1.4 Raw Dataset Columns: Quarters

The raw quarter dataset contained address-level information used by the City of Zurich for administrative and statistical purposes. The fields included:

- adresse
- anzahl_fla_projektiert
- anzahl_fla_real
- flaeche_projektiert
- flaeche_real
- flaeche_total
- gwr_egid
- hausnummer
- lokalisationsname
- objectid
- stadtkreis
- statistisches_quartier

4.1.4.1 Cleaning and Transformation To link traffic data with geographic units, the raw fields required several preprocessing steps:

- **Address splitting and cleaning**
  Street names and house numbers were embedded in a single free-text field (e.g., `"Heinrich-Federer-Strasse 12A"`, `"Widmerstrasse 88"`, `"Seeblickstrasse 17d"`).
  Using regular expressions, house numbers (including suffixes such as `12b`, `17d`) were removed, leaving a clean street-level identifier.
- **Data-type normalisation**
  `city_district` was converted to a nullable integer type, and `statistical_quarter` was normalised as a trimmed string. Inconsistent labels such as `"Schwamend.-Mitte"` were harmonised to `"Schwamendingen-Mitte"` to ensure joinability with the population dataset.

4.1.4.2 Additional Standardisation for Street Name Matching To ensure that traffic measurement locations could be linked reliably to the quarter dataset, several street names from the traffic dataset required manual standardisation. While most addresses were harmonised through regex-based cleaning, a small number of street names contained compound forms that could not be resolved automatically (e.g., combined street names, hyphenated patterns, or slash-separated names). These were corrected manually to create a consistent set of street identifiers.

Examples of manual mappings include:

- `Sood-/Leimbachstrasse` → `Soodstrasse`
- `Tobelhof-/Dreiwiesenstrasse` → `Tobelhofstrasse`
- `Manessestrasse - Schimmelstrasse` → `Manessestrasse`
- `Angererstrasse Tunnelstrasse` → `Angererstrasse`

Additionally, several counting site names referred to non-street features such as bridges, tunnels or motorway segments (e.g., *Quaibrücke*, *Milchbucktunnel*, *A1L*). For these cases, the closest corresponding street-level name from the quarter dataset was identified manually.

These manual adjustments were essential to achieve a fully joinable set of street names between the

traffic and quarter datasets and ensured that all counting sites could be assigned to a statistical quarter.

4.1.4.3 Cleaned Quarter Dataset Columns

- **address**: Original full address as provided in the raw data.
- **house_number**: Extracted house number component (kept for completeness).
- **street_name**: Cleaned and standardised street name used as the spatial key.
- **city_district**: Official Zurich district number (1–12).
- **statistical_quarter**: Statistical quarter ("Statistisches Quartier") providing fine-grained spatial units.

The cleaned quarter dataset provides the geographic reference layer used to integrate traffic measurements with demographic information.

4.1.5 Raw Dataset Columns: Population

The raw population dataset consisted of quarterly demographic counts published by the City of Zurich. The original fields included:

- **StichtagDatJahr**: Year of the reference date.
- **StichtagDatMM**: Month of the reference date (numeric).
- **StichtagDatMonat**: Month of the reference date (German label).
- **StichtagDat**: Full reference date (e.g., "1998-03-31"), indicating quarterly population snapshots.
- **SexCd** / **SexLang**: Numeric code and German label representing the sex category.
- **AlterV20ueber80Sort_noDM**, **AlterV20ueber80Cd_noDM**, **AlterV20ueber80Kurz_noDM**: Age-group fields (not relevant for this project).
- **HerkunftCd** / **HerkunftLang**: Numeric code and German label representing population origin.
- **KreisCd** / **KreisLang**: District number and district name.
- **QuarCd** / **QuarLang**: Statistical quarter code and label.
- **DatenstandCd**, **DatenstandLang**: Metadata describing the publication state.
- **AnzBestWir**: Population count.

4.1.5.1 Cleaning and Transformation To prepare the dataset for integration and analysis, several cleaning and normalisation steps were performed:

- **Column reduction**
  Multiple date-related fields were redundant. `StichtagDatJahr` was retained and renamed to `reference_date_year`.
- Age-group columns and metadata fields were removed.
- **Category translation and code preservation**
  German labels for sex and origin (e.g., `"männlich"`, `"weiblich"`, `"Schweizer*in"`, `"Ausländer*in"`) were translated to English.
  The numeric codes (`SexCd`, `HerkunftCd`) were preserved and renamed to `sex_code` and `origin_code`, as these form stable keys for later relational integration.
- **Data-type normalisation**
  - `KreisCd` was converted from float to nullable integer and renamed to `city_district`.

– `QuarLang` was cleaned and normalised (whitespace trimming, harmonisation of abbreviations), then renamed to `statistical_quarter`.

4.1.5.2 Cleaned Population Dataset Columns

- **reference_date_year**: Year of the population count, extracted from the quarterly reference date.
- **sex_code**: Numeric code representing the sex category.
- **sex**: Human-readable sex label derived from `sex_code`, such as "male", "female", or "unknown".
- **origin_code**: Numeric code representing the origin category.
- **origin**: Human-readable origin label derived from `origin_code`, such as "Swiss" or "Foreign".
- **city_district**: Zurich district number (1–12).
- **statistical_quarter**: Statistical quarter ("Statistisches Quartier"), harmonised to match the Quarter dataset.
- **population_count**: Number of residents in the demographic segment.

4.2 Entity–Relationship Model

The cleaned and harmonised datasets were integrated into a relational data model that reflects the structure of Zurich's traffic measurement infrastructure, its geographic units, and the associated demographic information. The final model was created in *dbdiagram.io* and is shown in the figure below.

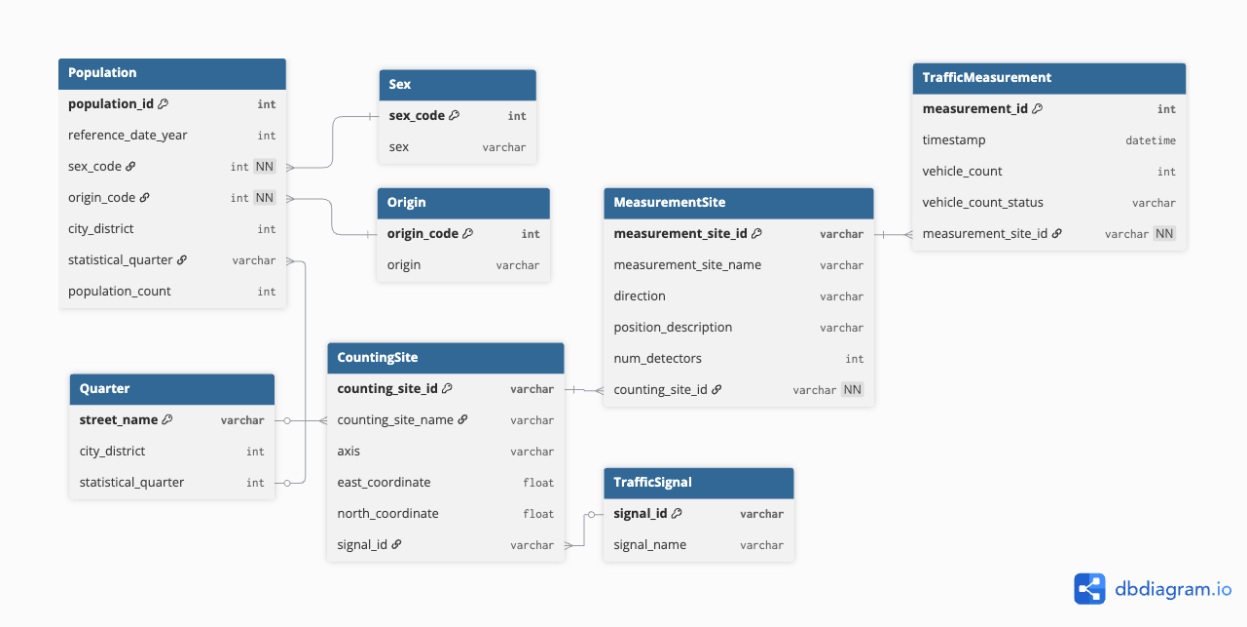

Figure 1: Entity–Relationship Diagram of the integrated Traffic–Quarter–Population model.

4.2.1 Overview

The model is centred around the **traffic measurement process**, which is represented by three core entities:

- **CountingSite**
  Represents a physical traffic counting location (e.g., a road segment or intersection). It stores geographic coordinates, the human-readable location name, and the traffic axis.
- **MeasurementSite**
  Represents a directional or lane-specific measurement point located within a counting site. Each measurement site records traffic in a single direction and may contain multiple detectors.
- **TrafficMeasurement**
  Stores the hourly measured vehicle counts, including measurement status and timestamp.

These three entities form a clear 1:n:n hierarchy:
CountingSite → MeasurementSite → TrafficMeasurement.

4.2.2 Traffic Signal Dimension

Some counting sites are associated with a traffic signal or intersection controller. This metadata is normalised into the **TrafficSignal** table. Since not every counting site is linked to a signal, the relationship is **optional**.

4.2.3 Geographic Dimension: Quarter

The **Quarter** table provides spatial context by assigning each cleaned street and address to:

- a **city district** (1–12), and
- a **statistical quarter** (fine-grained geographic unit).

Counting sites may refer to road infrastructure that is not part of the standard quarter dataset (e.g., motorway segments, tunnels, bridges). Therefore, the relationship between **CountingSite** and **Quarter** is modelled as **optional (0–1)**.

4.2.4 Demographic Dimension: Population

The **Population** table contains demographic counts by sex, origin, and year, referenced at both district and statistical-quarter level. To normalise categorical attributes, the model includes two lookup tables:

- **Sex** (sex_code → sex)
- **Origin** (origin_code → origin)

Every population record references a sex and origin category (mandatory), while the link to a statistical quarter is **optional**. This reflects the presence of entries such as *"Unbekannt (Stadt Zürich)"*, which cannot be assigned to a specific spatial unit.

4.2.5 Optional Relationships (`--0--<`)

Several foreign-key relationships are intentionally optional to reflect real characteristics of the source data:

- Some counting sites lie outside the quarter system.
- Some population entries cannot be mapped to a statistical quarter.
- Not all counting sites are linked to a traffic signal.

By modelling these relationships as optional, the ERD mirrors the true structure and limitations of the underlying open data and avoids enforcing artificial or incorrect mappings.

5 Loading & Transforming the Data

5.1 Two-Stage Loading Strategy

Our team implemented workflow to ensure data quality and enable flexible transformations. Raw CSV files were first loaded into staging tables within the `traffic_population_zh_staging` schema, validated, and then transformed into the normalized production schema (`traffic_population_zh`). This approach provided several advantages:

- **Data validation**: Staging tables allowed inspection and cleaning before committing to the production schema.
- **Transformation flexibility**: Adjustments could be still performed via SQL after initial load.
- **Error isolation**: Loading issues in staging will not affect production tables.

5.2 Schema and Table Creation

All production tables were defined using the script shown in Appendix A. The schema follows the Entity-Relationship model described in the previous chapter, implementing 3NF with appropriate primary and foreign keys.

5.2.1 Example: Population Table

```
CREATE TABLE Population (
    population_id INT PRIMARY KEY,
    reference_date_year INT,
    sex_code INT,
    origin_code INT,
    city_district INT,
    statistical_quarter VARCHAR(100),
    population_count INT,
    FOREIGN KEY (sex_code) REFERENCES Sex(sex_code),
    FOREIGN KEY (origin_code) REFERENCES Origin(origin_code)
);
```

5.2.2 Example: TrafficMeasurement Table

```
CREATE TABLE TrafficMeasurement (
    measurement_id INT PRIMARY KEY,
    timestamp DATETIME,
    vehicle_count INT,
    vehicle_count_status VARCHAR(50),
    measurement_site_id VARCHAR(50),
    FOREIGN KEY (measurement_site_id) REFERENCES MeasurementSite(measurement_site_id)
);
```

Lookup tables (dimension tables with no dependencies) (`Sex`, `Origin`, `TrafficSignal`) and geographic reference tables (`Quarter`) were created first to satisfy other tables foregin key requirements, like Population have foreign keys that point to these lookup tables.

5.3 Loading Raw Data into Staging Tables

5.3.1 CSV Accessibility Verification

Before loading, the team verified that MySQL could access the CSV files correctly:

```
USE traffic_population_zh_staging;
SELECT LOAD_FILE('C:\\Users\\RAMIRO\\Downloads\\population_data_cleaned_final.csv')
  IS NOT NULL AS can_read;
```

5.3.2 LOAD DATA INFILE Command

Population data was loaded using MySQL's `LOAD DATA INFILE` statement, it provides efficient loading:

```
LOAD DATA INFILE 'C:\\Users\\RAMIRO\\Downloads\\population_data_cleaned_final.csv'
INTO TABLE population_data_staging
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(reference_date_year, sex_code, sex, origin_code, origin,
 city_district, statistical_quarter, population_count);
```

Key parameters:

- `FIELDS TERMINATED BY ','`: CSV delimiter
- `OPTIONALLY ENCLOSED BY '"'`: Handles quoted fields containing commas
- `IGNORE 1 ROWS`: Skips the CSV header row
- Column list: Maps CSV columns to staging table columns in order

Similar `LOAD DATA` commands were executed for the traffic and quarter datasets, loading them into `stg_traffic_data` and `quarter_data_staging`.

5.4 Transformation into Production Schema

After staging validation, data was transformed and inserted into the normalized production tables. This involved: 1. 2. 3. 4.

Example transformation:

“‘sql code

```
## Challenges and Solutions

Several technical challenges were encountered during the loading process:

- **problem**:
```

- **problem**:

Analyzing & Evaluating Data
- **place holder**:

Efficiency & Query Performance
- **place holder**:

Visualization & Decision Support
- **place holder**:

Conclusions & Lessons Learned
- **place holder**:

Individual Team Member Reflections (required)

Individual Member Structure
- **Member Overview:**
- **Technical Takeaways:**
- **Collaboration Insights:**
- **Next Steps:**

Generative AI Declaration & Guidelines

Generative AI was used as a supplementary tool on top of the assisting material provided in the

Guidelines for Responsible and Effective Usage of Gen-AI
1. **Human-in-the-loop verification:** All AI-suggested content was treated as a draft. Text, c
2. **Emphasis on learning:** Gen-AI functioned as a learning companion rather than an automated
3. **Transparency:** The team openly acknowledged where and how AI was used in the workflow so

Use Cases of AI Tools in the DBM Course Context
AI-based assistants were applied in the following specific cases:
1. **Brainstorming:** Elaborating initial ideas, structuring thoughts, and outlining coding app
2. **Debugging & optimization:** Explaining error messages and helping improve self-developed s
3. **Proofreading:** Accelerating grammar and typo checks during documentation.
4. **Information acquisition:** Searching for methodological references or code documentation i

Benefits and Challenges in Using Generative AI Tools
1. **Benefits:** Faster debugging (e.g., resolving R Markdown knitting errors or MySQL query is
2. **Challenges:** The ease of getting answers can create a temptation to trust outputs blindly

How to render this report

A single R command will render the file. In an R console run:
`rmarkdown::render("report.Rmd")`
Make sure your R working directory is the report folder (or use the full path). For PDF output

\newpage

Appendix

This appendix provides complete code listings and supplementary materials referenced in the ma

---

Appendix A: Complete Script for Table Creation

The following SQL script defines all tables in the `traffic_population_zh` production schema.

```sql
USE traffic_population_zh;

CREATE TABLE Sex (
    sex_code INT PRIMARY KEY,
    sex VARCHAR(50)
);

CREATE TABLE Origin (
    origin_code INT PRIMARY KEY,
    origin VARCHAR(100)
);


CREATE TABLE Population (
    population_id INT PRIMARY KEY,
    reference_date_year INT,
    sex_code INT,
    origin_code INT,
    city_district INT,
    statistical_quarter VARCHAR(100),
    population_count INT,
    FOREIGN KEY (sex_code) REFERENCES Sex(sex_code),
    FOREIGN KEY (origin_code) REFERENCES Origin(origin_code)
);


CREATE TABLE Quarter (
    street_name VARCHAR(100) PRIMARY KEY,
    city_district INT,
    statistical_quarter INT
);
```

```
CREATE TABLE CountingSite (
    counting_site_id VARCHAR(50) PRIMARY KEY,
    counting_site_name VARCHAR(100),
    axis VARCHAR(100),
    east_coordinate FLOAT,
    north_coordinate FLOAT,
    signal_id VARCHAR(50)
);

CREATE TABLE MeasurementSite (
    measurement_site_id VARCHAR(50) PRIMARY KEY,
    measurement_site_name VARCHAR(100),
    direction VARCHAR(50),
    position_description VARCHAR(200),
    num_detectors INT,
    counting_site_id VARCHAR(50),
    FOREIGN KEY (counting_site_id) REFERENCES CountingSite(counting_site_id)
);

CREATE TABLE TrafficSignal (
    signal_id VARCHAR(50) PRIMARY KEY,
    signal_name VARCHAR(100)
);

CREATE TABLE TrafficMeasurement (
    measurement_id INT PRIMARY KEY,
    timestamp DATETIME,
    vehicle_count INT,
    vehicle_count_status VARCHAR(50),
    measurement_site_id VARCHAR(50),
    FOREIGN KEY (measurement_site_id) REFERENCES MeasurementSite(measurement_site_id)
);

-- Optional: Add foreign key constraint for traffic signals
-- ALTER TABLE CountingSite
-- ADD FOREIGN KEY (signal_id) REFERENCES TrafficSignal(signal_id);

-- Optional: Performance indexes (discussed in Efficiency chapter)
-- CREATE INDEX idx_population_sex ON Population(sex_code);
-- CREATE INDEX idx_population_origin ON Population(origin_code);
-- CREATE INDEX idx_measurement_timestamp ON TrafficMeasurement(timestamp);
-- CREATE INDEX idx_measurement_site ON TrafficMeasurement(measurement_site_id);
-- CREATE INDEX idx_counting_site_signal ON CountingSite(signal_id);
```

5.5 Appendix B: Complete Script for Table Transformation