

|  |
| --- |
| **Data Analytics engineering Mentoring program AU25** |



Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 3](#_Toc62212634)

[2.1 Definitions & Acronyms 3](#_Toc62212635)

[2.2 Logical Scheme 4](#_Toc62212636)

[2.3 Objects 4](#_Toc62212637)

# 

# Business Description

## Business background

This database supports the operations of a metropolitan subway system, including trains, stations, ticketing, infrastructure, and scheduling. Its objective is to centralize all operational data, enabling efficient planning of routes, sales, maintenance, and passenger services.

## Problems. Current Situation

Before database integration, information was scattered across different tools and spreadsheets. This caused duplication of data, inconsistent train schedules, difficulties tracking maintenance, and slow access to reliable operational reports.

## the Benefits of implementing a database. Project Vision

The database provides a single source of truth for metro operations. It improves data consistency, automates scheduling, ensures accurate ticketing, supports planning and maintenance, and increases overall operational reliability and customer satisfaction.

# Model description

## Definitions & Acronyms

**Logical Scheme**

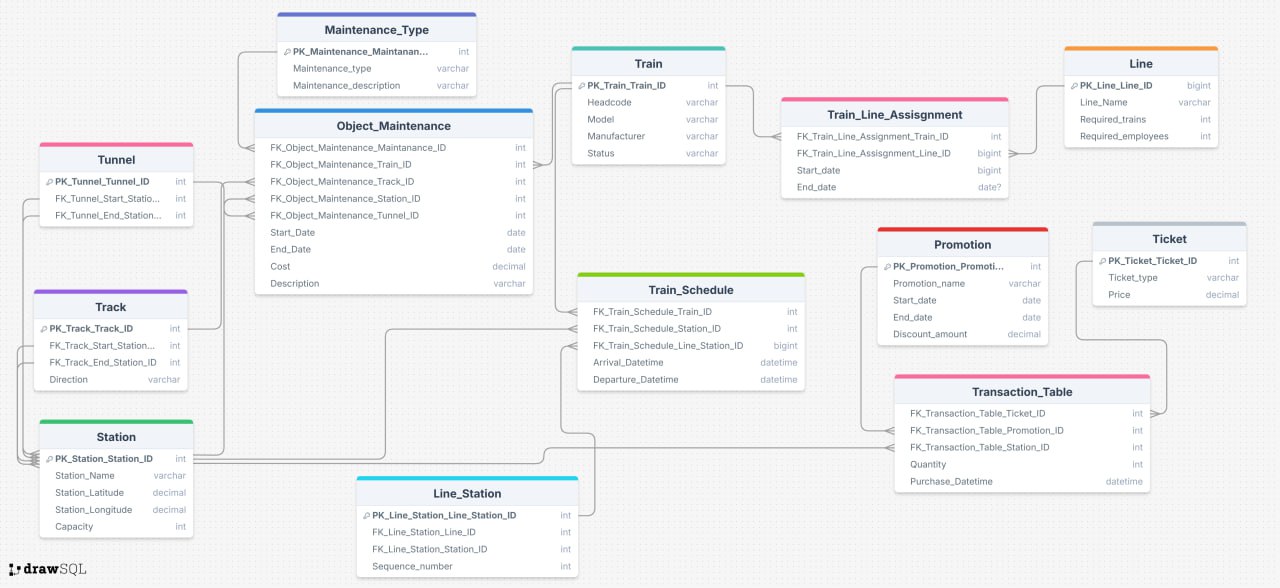
The model supports core metro processes:

* train movement and scheduling
* ticket sales and promotions
* tracking stations along multiple lines
* maintaining physical infrastructure (tracks, tunnels, trains)

**Key Design Decisions**

* **M:N Relationship (Train ↔️ Line):** Because a train can serve multiple lines over time and each line can have multiple trains, a junction table (Train\_Line\_Assignment) is used.
* **Historical accuracy:** Start\_Date and End\_Date allow tracking changes over time.
* **3NF Applied:** Each table stores one concept, all non-key attributes depend on the key, and no transitive dependencies exist.

## Logical Scheme



## Objects

**Table Description**

**OPERATIONS TABLES**

|  |  |
| --- | --- |
| Table Name | Description |
| Train | Stores information about each train, including its model, headcode, manufacturer, status, and the line it currently operates on. Each train runs on one line at a time. |
| Train\_Schedule | Defines the arrival and departure timetable of trains at stations. It connects a train to a specific station with arrival and departure datetimes and references the sequence of that station on its line. |
| Line | Represents a railway line. Each line has a name and required resource information such as number of trains and employees needed to operate it. |
| Line\_Station | A junction table that defines the ordered sequence of stations along a railway line. It ensures a line has multiple stations in the correct route order. |
| Station | Stores station details such as name, coordinates (latitude & longitude), capacity, and the line it belongs to. |

**INFRASTRUCTURE TABLES**

|  |  |
| --- | --- |
| Table Name | Description |
| Track | Defines an above-ground track segment between two stations, including direction information (e.g., Northbound/Southbound). |
| Tunnel | Defines a tunnel segment between two stations, similar to tracks, but for underground routes. |
| Object\_Maintenance | Tracks all maintenance performed on infrastructure or trains. It links to trains, tracks, stations, or tunnels and includes start/end dates, cost, and a description. |
| Maintanance\_Type | Lists types of maintenance (e.g., electrical, engine service, track repair) along with a description of each maintenance category. |

**TICKETING & PROMOTION TABLES**

|  |  |
| --- | --- |
| Table Name | Description |
| Ticket | Represents a ticket type (e.g., standard, express, VIP), including its base price. |
| Promotion | Defines discounts available during specific dates (e.g., holiday sale, weekend offers). Contains name, validity period, and discount amount. |
| Transaction\_Table | Logs completed ticket purchases. It records which ticket was bought, which promotion (if any) was applied, which station the purchase occurred at, total price, quantity, and purchase timestamp. |

**Table detailed description**

1. **Train\_Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Train\_Schedule | FK\_Train\_Schedule\_ Train\_ID | References the Train table (Train\_ID). Part of composite PK (FK) | Int |
| FK\_Train\_Schedule\_Station \_ID | References the Station table (Station\_ID). Part of composite PK (FK) | Int |
| FK\_Train\_Schedule\_Line\_Station\_ID | References the Line\_Station table (Line\_Station\_ID). Part of composite PK (FK) | Bigint |
| Arrival\_Datetime | Planned or actual arrival timestamp. | Datetime |
|  | Departure\_Datetime | Departure timestamp, may be NULL for terminals. | Datetime (nullable) |

Train ↔Train\_Schedule: One Train has many scheduled stops, so there is one-to-many relationship: 1:N.

Station↔Train\_Schedule: A Station has many schedule rows → 1:N.

Line\_Station↔Train\_Schedule: One Line\_Station has many scheduled calls; each scheduled row is exactly one stop→ 1:N.

**Examples with data.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FK\_Train\_Schedule\_Train\_ID | FK\_Train\_Schedule\_Station\_ID | FK\_Train\_Schedule\_Line\_Station\_ID | Arrival\_Datetime | Departure\_Datetime |
| 12 | 101 | 5001 | 2025-06-01  08:05 | 2025-06-01  08:08 |
| 13 | 102 | 5002 | 2025-06-01 08:12 | 2025-06-01 08:13 |

1. **Line**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Line | PK\_Line\_Line\_ID | Unique identifier of a subway line (PK) | bigint |
| Line\_Name | Human-readable name of the line. | varchar |
| Required\_Trains | Planned number of trains needed to operate the line. | int |
| Required\_Employees | Planned number of employees assigned to the line | int |

Line↔Train\_Line\_Assignment → 1:N

**Examples with data.**

|  |  |  |  |
| --- | --- | --- | --- |
| PK\_Line\_Line\_ID | Line\_Name | Required\_Trains | Required\_Employees |
| 3 | Green | 20 | 160 |
| 5 | Red | 24 | 190 |

1. **Station**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Station | PK\_Station\_Station\_ID | Unique identifier of a station (PK). | Int |
| Station\_Name | Official name of the station. | Varchar |
| Station\_Latituide | Latitude for mapping/GIS | Decimal |
| Station\_Longitude | Longitude for mapping/GIS | decimal |
| Capacity | Maximum passenger capacity (planning metric). | Int |

**Station** ↔**Line\_Station:** One station has many Line\_Station rows, each Line\_Station points to one Station→ 1:N

**Examples with data.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PK\_Station\_Station\_ID | Station\_Name | Station\_Latitude | Station\_Longitude | Capacity |
| 101 | Central Square | 41.4036 | 2.1744 | 60000 |
| 102 | Park Street | 41.4050 | 2.1800 | 45000 |

1. **Line\_Station**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Line\_Station | PK\_Line\_Station\_Line\_Station\_ID | Unique identifier of a station-on-a-line entry (PK). | Int |
| FK\_Line\_Station\_Line\_ID | References the table Line (Line\_ID). Identifies which line the station belongs to (FK). | Bigint |
| FK\_Line\_Station\_Station\_ID | References to the table Station and its primary key. Identifies the station on that line (FK). | Int |
| Sequence\_Number | Order of the station along the line (1,2,3,…). | int |

Line\_Station↔Train\_Schedule: One Line\_Station has many scheduled calls → 1:N

**Examples with data**

|  |  |  |  |
| --- | --- | --- | --- |
| PK\_Line\_Station\_LIne\_Station\_ID | FK\_Line\_Station\_Line\_ID | FK\_Line\_Station\_Station\_ID | Sequence\_Number |
| 5001 | 3 | 101 | 1 |
| 5002 | 3 | 102 | 2 |

1. **Train**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Train | PK\_Train\_Train\_ID | Unique identifier of a train (PK). | Int |
| Headcode | Operation code / run identifier | varchar |
| Model | Train model. | varchar |
| Manufacturer | Train manufacturer. | varchar |
| Status | Fleet status (e.g. Active, Out\_of\_service). | varchar |

Train↔Object\_Maintenance:→ 1:N

Train↔Train\_Schedule: → 1:N

**Examples with data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PK\_Train\_Train\_ID | Headcode | Model | Manufacturer | Status |
| 12 | G-220 | CityRunner | Siemens | Active |
| 18 | R-300 | MetroFlex | Alstom | Active |

1. **Train\_Line\_Assignment**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Train\_Line\_Assignment | FK\_Train\_Line\_Assignment\_Train\_ID | References the Train table (Train\_ID), Identifies which train is assigned to the line, FK | Int |
| FK\_Train\_Line\_Assignment\_Line\_Line\_ID | References the Line Table (Line\_ID), identifies the railway line that the train is assigned to, FK | Bigint |
| Start\_Date | The date when the train starts operating on the assigned line, used to track historical changes and prevent overlapping assignments. | Date |
| End\_Date | The date when the train stops operating on that line. If Null, it indicates the train is currently assigned to that line. | Date (Nullabe) |

A single **Train** can serve multiple Lines over time, and a **Line** can operate multiple Trains. Because this is a many-to-many relationship, it is modeled using the **Train\_Line\_Assignmnet** table(Junction table) → M:N

**Composite PK =TrainID+Line\_ID+Start\_Date**

**Examples with data**

|  |  |  |  |
| --- | --- | --- | --- |
| FK\_Train\_Line\_Assignment\_Train\_ID | FK\_Train\_Line\_Assignment\_Line\_ID | Start\_Date | End\_Date |
| 12 | 3 | 2025-01-10 | 2025-05-30 |
| 12 | 5 | 2025-06-01 | NULL |

1. **Ticket**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Ticket | PK\_Ticket\_Ticket\_ID | Unique identifier of a ticket product (PK). | int |
| Ticket\_Type | Product typw (Single, Day\_Pass, etc.). | varchar |
| Price | Base price before discounts. | decimal |

Ticket↔Transaction\_Table:→ 1:N

**Examples with data**

|  |  |  |
| --- | --- | --- |
| PK\_Ticket\_Ticket\_ID | Ticket\_Type | Price |
| 10 | Single | 2.50 |
| 20 | Day\_Pass | 7.00 |

1. **Promotion**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Promotion | PK\_Promotion\_Promotion\_ID | Unique identifier of a promotion (PK). | int |
| Promotion\_Name | Name of the promotion. | varchar |
| Start\_Date | Start date when the promotion is valid. | date |
| End\_Date | End date when the promotion expires. | date |
| Discount amount | Amount discounted. | Decimal |

Promotion↔Transaction\_Table: → 1:N

**Examples with data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PK\_Promotion\_Promotion\_ID | Promotion\_Name | Start\_Date | End\_Date | Discount\_Amount |
| 200 | Spring\_Sale | 2025-03-01 | 2025-03-31 | 0.50 |
| 210 | Weekend Deal | 2025-06-01 | 2025-06-02 | 1.00 |

1. **Transaction\_Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Transaction\_Table | FK\_Transaction\_Table\_Ticket\_ID | References Ticket (Ticket\_ID). Product purchased (FK) | int |
| FK\_Transaction\_Table\_Promotion\_ID | References Promotion (Promotion\_ID). Applied promo if any (FK, nullable). | int (nullable) |
| FK\_Transaction\_Table\_Station\_ID | References Station (Station\_ID) (FK). | int |
| Quantity | Number of tickets in the transaction. | int |
| Purchase\_Datetime | Timestamp of purchase. | datetime |

**Examples with data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FK\_Transaction\_Table\_Ticket\_ID | FK\_Transaction\_Table\_Promotion\_ID | FK\_Transaction\_Table\_Station\_ID | Quantity | Purchase\_Datetime |
| 10 | 200 | 101 | 2 | 2025-03-12 07:55 |
| 20 | NULL | 102 | 1 | 2025-06-01 09:02 |

1. **Track**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Track | PK\_Track\_Track\_ID | Unique identifier of a track segment (PK). | int |
| FK\_Track\_Start\_Station\_ID | References Station (Station\_ID). Segment starts (FK). | int |
| FK\_Track\_End\_Station\_ID | References Station (Station\_ID). Segment end (FK) | int |
| Direction | Operational directionality (e.g. N,S, bi-directional). | varchar |

Track↔Object\_Maintenance: → 1:N

**Examples with table**

|  |  |  |  |
| --- | --- | --- | --- |
| PK\_Track\_Track\_ID | FK\_Tarck\_Start\_Station\_ID | FK\_Track\_End\_Station\_ID | Direction |
| 9001 | 101 | 102 | Bi-directional |
| 9002 | 102 | 103 | Bi-directional |

1. **Tunnel**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Tunnel | PK\_Tunnel\_Tunnel\_ID | Unique identifier of a tunnel segment (PK). | Int |
| FK\_Tunnel\_Start\_Station\_ID | References Sttaion (Station\_ID). Segment start (FK). | Int |
| FK\_Tunnel\_End\_Station\_ID | References Station (Station\_ID). Segment end (FK). | Int |

Tunnel↔Object\_Maintenance: → 1:N

**Examples with data**

|  |  |  |
| --- | --- | --- |
| PK\_Tunnel\_Tunnel\_ID | FK\_Tunnel\_Start\_Station\_ID | FK\_Tunnel\_End\_Station\_ID |
| 8001 | 102 | 103 |
| 8002 | 101 | 102 |

1. **Maintenance\_Type**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Maintenance\_Type | PK\_Maintenance\_Type\_Maintenance\_Type\_ID | Unique identifier of a maintenance type (PK). | int |
| Maintenance\_Type | Name of the job. | varchar |
| Maintenance\_Description | Details of the maintenance activity. | int |

Maintenance\_Type↔Object\_Maintenance**:** → 1:N

**Examples with data**

|  |  |  |
| --- | --- | --- |
| PK\_Maintenance\_Type\_Maintenance\_TypeID | Maintanance\_Type | Maintenance\_Description |
| 41 | Brake Inspection | Routine brake check and pad replacement. |
| 12 | Rail Grinding | Smooth rail surface to reduce noise/wear. |

1. **Object\_Maintenance**

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Object\_Maintenance | FK\_Object\_Maintenance\_Maintenance\_Type\_  ID | References Maintenance\_Type (Maintenance\_Type\_ID) (FK). | int |
| FK\_Object\_Maintenance\_Train\_ID | References Train (Train\_ID). The maintained train (FK, nullable). | int (nullable) |
| FK\_Object\_Maintenance\_Track\_ID | References Track (Track\_ID). The maintained track segment (FK, nullable) | int (nullable) |
| FK\_Object\_Maintenance\_Station\_ID | References Station (Station\_ID). The maintained station (FK, nullable). | Int (nullable) |
| FK\_Object\_Maintenance\_Tunnel\_ID | References Tunnel (Tunnel\_ID). The maintained tunnel (FK, nullable). | int (nullable) |
| Start\_Date | Date the maintenance begins. Used in uniqueness and recording. | date |
| End\_Date | Date the maintenance ends (nullable for 1-day tasks). | date (nullable) |
| Cost | Cost of maintenance event. | decimal |
| Description | Free-text description/notes. | Varchar |

**Examplesa with data**

**Fact table, no surrogate PK**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| FK\_Object\_Maintenance\_Maintenance\_Type\_ID | FK\_Object\_Maintenance\_Train\_ID | FK\_Object\_Maintenance\_Track\_ID | FK\_Object\_Maintenance\_Station\_ID | FK\_Object\_Maintenance\_Tunnel\_ID | Start\_Date | | End\_Date | Cost | Description |
| 41 | 12 | NULL | NULL | NULL | 2025-03-10 | 2025-03-10 | | 350.00 | Brake pads replaced |
| 12 | NULL | 9001 | NULL | NULL | 2025-03-08 | 2025-03-09 | | 1200.00 | Rail grinding |