

|  |
| --- |
| Business Template  **Subway (metro)** |
|  |



Contents

[1 Business Description 3](#_Toc161175101)

[1.1 Business background 3](#_Toc161175102)

[1.2 Problems. Current Situation 3](#_Toc161175103)

[1.3 the Benefits of implementing a database. Project Vision 3](#_Toc161175104)

[2 Model description 4](#_Toc161175105)

[2.1 Definitions & Acronyms 4](#_Toc161175106)

[2.2 Logical scheme 5](#_Toc161175107)

[2.3 Data objects 6](#_Toc161175108)

[2.5 Examples of data 9](#_Toc161175109)

[2.6 table relationships 10](#_Toc161175110)

# 

# Business Description

## Business background

The company called Subwaypolis is a company which have a subway (metro) system of a city called Budapest. The company have a large history, it’s operation started when the first subway system setted up in 1894. Millions of passengers uses the system each year. The company now owns 4 different metro lines (M1, M2, M3, M4) and having more than 20 trains and hundreds of employees.

The business itself is fairly large and complex, consisting of subway lines, each with multiple stations. The trains run on these lines by following a schedule, carrying passengers from one station to another within a line. The metro system can be used by those who purchase a ticket, there are different types of ticket with various prices and discounts or promotions. The business itself is responsible to maintain and repair the infrastructure where they are involved, the tunnels, lines and stations.

As the company grows, the larger the system becomes the harder become to control, maintain and supervise the operations. Due to the old tracking systems, there are continuous errors occurring which highly influence the operation of the company badly, which causes an increase in unsatisfied passengers through the years.

## Problems. Current Situation

There are 3 main problems which brings hundreds of minor ones.

1. **Information is scattered** across different systems, leading to inconsistencies and making it difficult to get a unified view of operations.
2. **Manual data entry** and **lack of automation** lead to operational inefficiencies and increased likelihood of errors.
3. **Difficulty in accessing** and **analyzing data** limits the company’s ability to make data-driven decisions and improvements.

## the Benefits of implementing a database. Project Vision

Implementing a database is inevitable nowadays from a business perspective. By having a centralized and clear management of all information related to the subway system, will increase the efficiency in the long-term and even immediately. Tracking and optimization is the key goal in this project, and to reach that, we must create a logical data model for the subway system which will be the base for the future data interpretations.   
If I would highlight 5 major strength of database integration, these are the following:

1. **Centralized Data Management**
2. **Efficiency and Accuracy**
3. **Data-Driven decision-making**
4. **Scalability**
5. **Automatization**

The vision for this project is to design and implement a comprehensive database system that accurately represents Subwaypolis’s operations and facilitates efficient data management. The database will include tables for stations, lines, schedules, tickets, infrastructure, trains, employees, routes, and more, with appropriate relationships and constraints to ensure data integrity. We will collaborate with managers and key members of the company to meet the needs and give space to test, review and share ideas about the project. Once it will be completed, the benefits of this project couldn’t be questioned, it will be clearly visible for the employees and customers immediately.

# Model description

## Definitions & Acronyms

**Train** – The vehicle which transports passengers, all trains on a line are identical and interchangeable.

**Station** – A place where passengers can go on and of a train, in each station there are one ore more employees

**Line** – This subway line is the main path within the subway system, it is predefined route which followed by the train and its connects multiple stations. In example if the subway system of Budapest, the “M4” line connects stations between Újbuda-központ and Keleti-pályaudvar.

**Route –** The route represents a specific path that can be taken by the train. It is possible that for example let assume that on weekdays the train stops at every station in the “M4”line between Újbuda-központ and Keleti pályaudvar, however on weekends from 7pm until 11 pm, the train skip some station within the “M4” because of the lack of passengers at that period. Or it can happen that there is a demonstration for example in Keleti-pályaudvar, and the police decide to stop the operation of the station Keleti-pályaudvar for a few hours. So in this case in a line there can be many different routes.

**Schedule** – Information about when the trains arrive and depart from each station

**Ticket** – A pass that allows passengers to use the subway

**TicketType –**  The types of ticket which contains the discount receivable for the already set types: Adult, Child, Student, Senior, Disabled.

**Infrastructure** – It consist the physical structures that are necessary for the operations, the most important are the Trains, Stations and Lines

**Employee** – A person who works for the subway(metro) system’s business.

**RouteSchedule** – A junction table that implements a many-to-many relationship between Route and Schedule.

**TicketSale** - A junction table that implements many-to-many relationship between Ticket and Schedule

## Logical scheme

A diagram of a network

Description automatically generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type | Constraint |
| **Station** | StationID | Unique identifier for each station | INT | Primary Key |
| StationName | The name of the station | VARCHAR(100) | Not Null |
| Location | Location of the station | VARCHAR(100) | Not Null |
| InfrastructureID | A Foreign Key referencing the “Infrastructure” table (FK) | INT | Not Null |
| EmployeeID | A Foreign Key referencing the “Employee” table (FK) | INT | Not Null |
| **Line** | LineID | A Unique identifier for each station | INT | Primary Key |
| LineName | The name of the line | VARCHAR(100) | Not Null |
| OperatingFrequency | The frequency at which trains operate on this line | INT | Not Null |
| NumberOfTrains | Number of trains operating on this line | INT | Not Null, Greater than 0 |
| NumberOfEmployees | Number of employees working on this line | INT | Not Null, Greater than 0 |
| InfrastructureID | A foreign key referencing the “Infrastructure” table (FK) | INT | Foreign Key ref. Infrastructure.InfrastructureID, Not Null |
| **Schedule** | ScheduleID | A unique identifier for each schedule (PK) | INT | Primary key |
| LineID | A foreign key referencing the “Line” table (FK) | INT | Foreign Key ref. Station.StationID, Not Null |
| StationID | A foreign key referencing the “Station” table | INT | Not Null |
| ArrivalTime | The time when the trains arrive at the station | TIME | Not Null |
| DepartureTime | The time when trains depart from the station | TIME | Not Null |
| **Ticket** | TicketID | A unique identifier for each ticket | INT | Primary Key |
| TypeID | The type of the Ticket | INT | Foreign Key ref. TicketType.TypeID, Not Null |
| Price | Price of the ticket | DECIMAL (5,2) | Not Null, Greater than 0 |

## Data objects

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type | Constraint | |
| **Infrastructure** | InfrastructureID | A unique identifier for each piece of infrastructure (PK) | INT | | Primary Key |
| Type | This field can have values of: line, train, station | VARCHAR(50) | | Not Null |
| Status | Current status of the infrastructure it can be changed overtime | VARCHAR(100) | | Not Null |
| LastMaintenance | The date of the last maintenance | DATE | | Not Null |
| NextMaintenance | The date of the next scheduled maintenance | DATE | | Not Null |
| **Employee** | EmployeeID | A unique identifier for each employee (PK) | INT | | Primary Key |
| Name | The name of the employee | VARCHAR(100) | | Not Null |
| Position | The position of the employee | VARCHAR(100) | | Not Null |
|  |  |  | |  |
| **Route** | RouteID | A unique identifier for each route (PK) | INT | | Primary Key |
| LineID | A foreign key referencing the “Line” table(FK) | INT | | Foreign Key ref. Line.LineID, Not Null |
|  |  |  | |  |
|  |  |  | |  |
| **Train** | TrainID | A unique identifier for each train (FK) | INT | | Primary Key |
| LineID | A foreign key referencing the “Line” table (FK) | INT | | Foreign Key ref. Line.LineID, Not Null |
| InfrastructureID | A foreign key referencing the “Infrastructure” table (FK) | INT | | Foreign Key ref. Infrastructure. infrastructureID,  Not Null |
|  |  |  | |  |
| **TicketType** | TypeID | Unique identifier for each type of Ticket | INT | | Primary Key |
| TypeName | The name of the ticket type | VARCHAR(100) | | Not Null |
| Discount | The percentage discount rate of the ticket | Decimal(3, 2) | | Greater or Equal to 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type | Constraint |
| **RouteSchedule**  (It is a junction table for many-to-many relationship between “Route” and “Schedule”) | RouteID | A foreign key referencing the “Route” table (FK) | INT | Foreign Key ref Route.RouteID, Not Null |
| ScheduleID | A foreign key referencing the “Schedule” table | INT | Foreign Key ref. Schedule.ScheduleID, Not Null |
| **TicketSale**  (It is a junction table for many-to-many relationship between “Ticket” and “Schedule”) | TicketID | A foreign key referencing the “Ticket” table | INT | Foreign Key ref. Ticket.TicketID, Not Null |
| ScheduleID | A foreign key referencing the “Schedule” table | INT | Foreign Key ref. Schedule.ScheduleID, Not Null |
| SaleDate | The date of the ticket sale | DATE | Not Null |
| **StationRoute** (It is a junction table for many-to-many relationship between “Route” and “Station” | RouteID | A foreign key referencing the “Route” table (FK) | INT | Foreign Key ref Route.RouteID, Not Null |
| StationID | A foreign key referencing the “Station” table (FK) | INT | Foreign Key ref Station.StationID, Not Null |
| **TrainRoute**  (It is a junction table for many-to-many relationship between “Route” and “Train” | RouteID | A foreign key referencing the “Route” table (FK) | INT | Foreign Key ref Route.RouteID, Not Null |
| TrainID | A foreign key referencing the “Train” table (FK) | INT | Foreign Key ref Train.TrainID, Not Null |
| **EmployeeLine**  (It is a junction table for many-to-many relationship between “Employee” and “Line” | EmployeeID | A foreign key referencing the “Employee” table (FK) | INT | Foreign Key ref Employee.Employee ID, Not Null |
| LineID | A foreign key referencing the “Line” table (FK) | INT | Foreign Key ref Line.LineID, Not Null |
| EmployeeStation  It is a junction table for many-to-many relationship betweenemployee and station | EmployeeID | A foreign key referencing the “Employee” table (FK) | INT | Foreign Key ref employee.employeeID, Not Null |
| StationID | A foreign key referencing the “Station” table (FK) | INT | Foreign Key ref station.stationID, Not Null |

|  |  |  |
| --- | --- | --- |
| Table Name | Field name | Field data example |
| **Station** | StationID | STN1 |
| StationName | “Deák Ferenc tér” |
| Location | “Budapest, Hungary” |
| InfrastructureID | INF1 |
| **Line** | LineID | LN1 |
| LineName | “M1 – Yellow Line” |
| OperatingFrequency | 5 (minutes) |
| NumberOfTrains | 10 |
| NumberOfEmployees | 30 |
| InfrastructureID | INF1 |
| **Schedule** | ScheduleID | SCH1 |
| LineID | LN1 |
| StationID | STN1 |
| ArrivalTime | “06:00” |
| DepartureTime | “06:02” |
| **Ticket** | TicketID | TK1 |
| TypeID | TTYP2 |
| Price | 350.0 (HUF) |
| **TicketType** | TypeID | TTYP2 |
| TypeName | “Student” |
| Discount | 50.0 |
| **Infrastructure** | InfrastructureID | INF1 |
| Type | “Train 01” |
| Status | “Operating” |
| LastMaintenance | “2024-01-01” |
| NextMaintenance | “2024-04-01” |
| **Employee** | EmployeeID | EMP1 |
| Name | “Kovács Emil” |
| Position | “Train driver” |
|  |  |

## Examples of data

|  |  |  |
| --- | --- | --- |
| Table Name | Field name | Field data example |
| **Route** | RouteID | RTE1 |
| LineID | LN1 |
|  |  |
| **Train** | TrainID | TRN1 |
| LineID | LN1 |
| InfrastructureID | INF1 |
| **RouteSchedule**  (It is a junction table for many-to-many relationship between “Route” and “Schedule” | RouteID | RTE1 |
| ScheduleID | SCH1 |
| **TicketSale**  (It is a junction table for many-to-many relationship between “Ticket” and “Schedule” | TicketID | TK1 |
| ScheduleID | SCH1 |
| SaleDate | “2024-03-12” |
| **StationRoute** (It is a junction table for many-to-many relationship between “Route” and “Station” | RouteID | RTE1 |
| StationID | STN1 |
| **TrainRoute**  (It is a junction table for many-to-many relationship between “Route” and “Train” | RouteID | RTN1 |
| TrainID | TRN1 |
| **EmployeeLine**  (It is a junction table for many-to-many relationship between “Employee” and “Line” | EmployeeID | EMP1 |
| LineID | LN1 |
| **EmployeeStation** It is a junction table for many-to-many relationship betweenemployee and station | EmployeeID | EMP1 |
| StationID | STN1 |

## table relationships

|  |  |  |
| --- | --- | --- |
| Table 1 | Relationship | Table 2 |
| Ticket | Many-to-many | TicketSale |
| TicketType | One-to-many | Ticket |
| TicketSale | One-to-one | Schedule |
| Schedule | One-to-Many | Station |
| Schedule | One-to-many | RouteSchedule |
| Schedule | One-to-many | Line |
| Station | One-to-one | Infrastructure |
| Route | One-to-many | RouteSchedule |
| Line | One-to-Many | Route |
| Infrastructure | One-to-one | Line |
| Infrastructure | One-to-one | Train |
| Line | One-to-many | Train |
| Train | One-to-many | TrainRoute |
| Employee | One-to-Many | EmployeeLine |
| Station | One-to-Many | EmployeeLine |
| Employee | One-to-Many | EmployeeLine |
| Line | One-to-Many | EmployeeLine |
| Route | One-to-Many | TrainRoute |
| Route | One-to-Many | StationRoute |
| Station | One-to-Many | StationRoute |