

|  |
| --- |
| Business Template  **Subject areas** |
| **Logo / Image** |

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 3](#_Toc62212636)

[2.3 Objects 3](#_Toc62212637)

# 

# Business Description

## Business background

The management of the subway metro system which transports millions of passengers every day is a difficult task that if performed to the best of a person’s ability may make a task that appears gargantuan turn into a goose neck experience. However, managing such complexing duties, without the assistance of central networks, is difficult as they require separate or manual logs. Consequently, maintenance schemes take longer to address than necessary, ticket rates are rather unstandardized and the entire process becomes inefficient.

## Problems. Current Situation

Maintenance Issues Analytical Problems: Maintenance and repair details about the train, tracks, tunnels, and the station related and additional upkeep are not centralized and hence become hard to maintain.

Problems Related To Pricing And Ticketing: In regards to ticket pricing, discounts and prices appear to be set apart, which doesn’t allow room for flexibility and revenue loss.

Decision-Making Difficulties: The presence of scatter plots makes the creation of quick and informed decisions relating to maintenance and security rather complex.

Exacerbated Safety Circumstances: The lack of integrated documents increases the risk of neglecting any essential maintenance which causes security problems.

## the Benefits of implementing a database. Project Vision

Organized Approach: The integrating of all database ensures that all the information is accessed in a reliable and timely manner to manage repairs, schedules and ticketing.

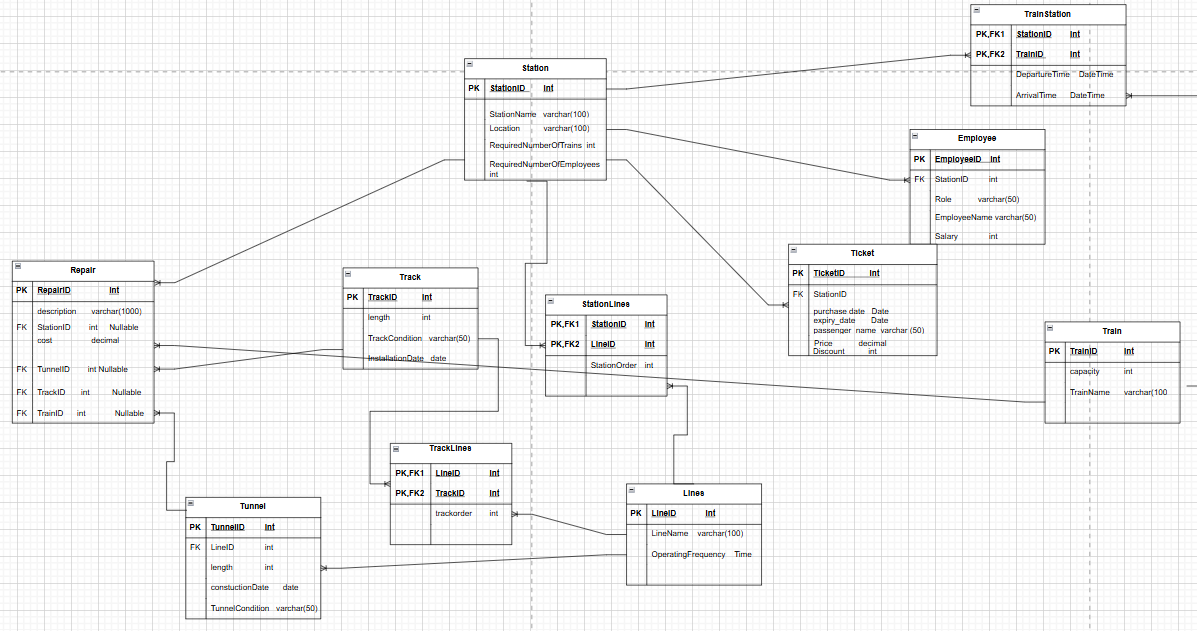
Segmentation Planning: Maintenance schedules and repairs are well managed and recorded, helping to reduce downtime and increasing safety.

The Improvement of Rev Nevertheless, the idea permits considerable flexibility for the pricing, discounts, and advertising of tickets, increasing re.

# Model description

## Definitions & Acronyms

## Logical Scheme



## Objects

Table Description

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Station | StationID | PK | Int |
| Location |  | Varchar(100) |
|  | RequiredNumberOfTrains |  | Int Constraint(>=0) |
|  | RequiredNumberOfEmployees |  | Int Constraint(>=0) |

Table station have One-to-Many relationship with table ticket because at one station there can be many tickets bought but you can buy exactly one ticket at one station. Because of that table Ticket needs StationID. Also it have One-to-Many relationship with Table employee because at one station there can be many employees but one employee works at one station, because of that Table Employee needs StationID as FK and I also added it in Employee table. Table station has Many-to-Many relationship with Table Train, because there can be many trains at one station and train can go at many station so because of that I created bridge table and Named it TrainStation and I will talk about it later. there is One-to-Many relation between table Station and table Repair, And lastly there is again Many-to-Many relationship with table Line. Because one station can be at many lines and line can be at many stations. Because of that I created another bridge table and added StationID and LineID as composite primary key and also marked them as foreign keys.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| StationID | Location | NumberOfTrains | NumberOfEmployees |
| 1 | New-york | 123 | 1234 |

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Ticket | TicketID | PK | Int |
| StationID | FK | Int |
| PurchaseDate |  | date |
| Expirty\_date |  | Date Constraint(Expiry\_date>purchaseDate) |
|  | Price |  | Int constraint(price>0) |
|  | Discount |  | Decimal-constraint  Discount between 0 and 100 |
|  | PassengerName |  | Varchar(50) |

Ticket table has One-to-Many relationship with Station because you can buy only 1 ticket at 1 station but at 1 station there can be many tickets to be bought .Ticket also has price, discount, purchase date

And its passenger name. I made constraints for some attributes. First one is expiry\_date and it must be

Greater than purchaseDate. Second one is Discount and it should be between 0 and 100. And last one is price and it should be greater than 0.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TicketID | StationID | PurchaseDate | ExpiryDate | Price | Discount | PassengerName |
| 1 | 23 | 11-01-2021 | 12-01-2021 | 30 | 12.65 | ‘Daviti’ |

Table Employee

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Employee | EmployeeID | PK | Int |
| StationID | FK | Int |
| Role |  | Varchar(50) |
| EmployeeName |  | varchar |
|  | Salary |  | Int Constraint(salary>=0) |

Employee table has EmployeeID as PK so we can identify every employee uniquely. We have StationID as FK that’s because there is One-to-Many relationship between Employee and Station Tables, so if we want to know in which station does the employee work we need to add StationID at the table. Also employees have roles and that’s why I added it as a attribute, they also have name and Salary.

I made constraint on salary attribute. It should be greater or equal to zero.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EmployeeID | StationID | Role | EmployeeName | Salary |
| 1 | 23 | mechanic | daviti | 3000 |

Table Train

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Train | TrainID | PK | Int |
| TrainName |  | Varchar(50) |
| capacity |  | Int Constraint(Capacity>0) |

In Train table we have TrainID as PK and it uniquely identifies every train, Another attribute is TrainName,and in my logic there can be trains with the same name so TrainName doesn’t uniquely identifies train that’s why it is not part of the PK. Train also has capacity and I added it as a attribute with data type int and it should be greater than 0. we have Many-to-Many relationship between table Train and table Station. and lastly we have One-to-Many relationship between table Repair and table Train.

Example with data

|  |  |  |
| --- | --- | --- |
| TrainID | TrainName | capacity |
| 1 | Easter express | 200 |

Table Lines

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Lines | LineID | PK | Int |
| LineName |  | Varchar(50) |
| OperatingFrequency |  | time |

In table Lines LineID is PK. I also added LineName as an attribute and operatingFrequency which means how often do trains come and go and its type is time. there is Many-to-Many relationship between table Lines and table Station. There is also another Many-to-Many relationship between Table Track and table lines. Because one line can pass through multiple Tracks and one track can be used by multiple lines. Also there is One-to-Many relationship between Table Lines and table Tunnel. One tunnel is associated with one line and one line can pass through multiple tunnels.

Example with data

|  |  |  |
| --- | --- | --- |
| LineID | LineName | OperatingFrequency |
| 1 | Saburtalo Line | 07:00 |

Table Track

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Track | TrackID | PK | Int |
| InstallationDate |  | Date-constraint(InstallationDate<=currentDate) |
| length |  | Int-constraint(length>0) |
|  | TrackCondition |  | Varchar(50) |

In the Track table, TrackID is the primary key (PK). There is also a Length attribute that indicates the track length, along with InstallationDate and TrackCondition. The InstallationDate should be less than or equal to the current date. The Track table has a One-to-Many relationship with the Repair table because each track may require multiple repairs over its lifespan.and a Many-to-Many relationship with the Lines table.

Example with data

|  |  |  |  |
| --- | --- | --- | --- |
| TrackID | InstallationDate | TrackCondition | length |
| 1 | 10-01-2020 | Needs repair | 254 |

Table Tunnel

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Tunnel | TunnelID | PK | Int |
| LineID | FK | Int |
| length | In meters | int |
| ConstructioDate |  | Date(ConstructionDate<=currentDate> |
|  | TunnelCondition |  | Varchar(50) |

The tunnel table has TrackID as the primary key (PK) and LineID as a foreign key (FK). The non-prime attributes include ConstructionDate and TunnelCondition. It is important to ensure that ConstructionDate is less than or equal to the current date, maintaining the integrity of the data.

The Tunnel table has a One-to-Many relationship with the Lines table, which is why LineID is included; this indicates the specific line associated with each tunnel. Additionally, there is a One-to-Many relationship between the Tunnel table and the Repair table, as each tunnel may require multiple repairs over its lifespan. The Repair table would therefore include TunnelID as a foreign key to effectively track repairs related to each tunnel.

Example with data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TunnelID | LineID | length | TunnelCondition | ConstructionDate |
| 1 | 7 | 250 | good | 07-11-2021 |

Table Repair

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| Repair | RepairID | PK | Int |
| description |  | Text |
| RepairDate |  | Date |
| cost |  | Decimal constraint(cost>0) |
|  | TrackID | FK Nullable | Int |
|  | TrainID | FK Nullable | int |
|  | StationID | FK Nullable | int |
|  | TunnelID | FK Nullable | int |

In the Repair table, RepairID serves as the primary key (PK). The Description attribute explains what was damaged and where the damage occurred. RepairDate indicates when the repair was completed, and the Cost attribute specifies the expense of the repair, which must be greater than 0. This table has Four One-to-Many relationships with the Station, Track, Tunnel, and Train tables, and the corresponding foreign keys (FKs) have been added to the Repair table to facilitate these relationships.

Example with data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RepairID | Description | Cost | StationID | TunnelID | TrackID | TrainID |
| 1 | TrackRepair | 500 | Null | Null | 101 | Null |

Bridge Table-TrainStation

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| TrainStation | TrainID | PK,FK | Int |
| StationID | PK,FK | Int |
|  | DepartureTime |  | DateTime(DepartureTime< ArrivalTime) |
|  | ArrivalTime |  | DateTime |

Table TrainStation is a bridge table and has TrainID and StationID as composite primary key and they are also foreign keys. Also we have non prime attributes DepartureTime and ArrivalTime. Departure time should be less than ArrivalTime. I changed many-to-many relationship with two one-to-many relationship and that’s why I created this table.

Example with data:

|  |  |  |  |
| --- | --- | --- | --- |
| TrainID | StationID | DepartureTime | ArrivalTime |
| 1 | 7 | 2024-11-01  08:00:00 | 2024-11-01  07:50::00 |

Bridge Table-TrackLines

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| TrackLines | TrackID | PK,FK | Int |
| LineID | PK,FK | Int |
|  | TrackOrder |  | int |

Table TrackLines is a bridge table and has TrackID and LineID as composite primary key and they are also foreign keys. I changed many-to-many relationship with two one-to-many relationship and that’s why I created this table.

|  |  |  |
| --- | --- | --- |
| TrackID | LineID | TrackOrder |
| 101 | 10 | 2 |

Example with data

Bridge Table-StationLines

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field name | Field Description | Data Type |
| StationLines | StationID | PK,FK | Int |
| LineID | PK,FK | Int |
|  | StationOrder |  | int |

Table TrainStation is a bridge table and has TrackID and LineID as composite primary key and they are also foreign keys. I changed many-to-many relationship with two one-to-many relationship and that’s why I created this table. And additionally added one attribute StationOrder to keep track how the stations are ordered in one line.

|  |  |  |
| --- | --- | --- |
| StationID | LineID | StationOrder |
| 1 | 10 | 22 |