

|  |
| --- |
| 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

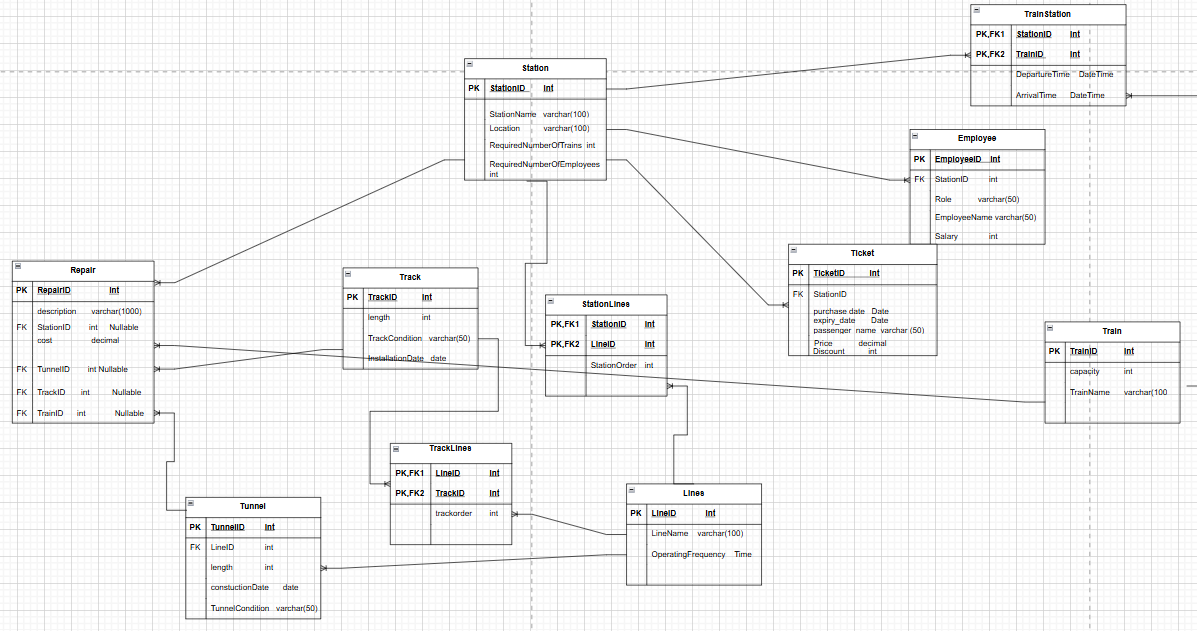
## Problems. Current Situation

## the Benefits of implementing a database. Project Vision

# 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 and I already talked about that.. Because of that I added TrainID as FK. Ticket also has price and its type is integer and I also added Discount attribute, because in the text discount was mentioned,also it has passangerName, PurchaseDate of the ticket and Expiry\_date and I made some constraints for them.

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.

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, to be honest I don’t know if trains have names but I added it anyway 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. as I have already mentioned we have Many-to-Many relationship between table Train and table Station. We have another Many-to-Many relationship between table Train and table Lines, because there can be many trains at one line and one train can be at many lines. We have One-to-Many relationship between table Train and table Ticket and because of that I added TrainID in Ticket table. 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 again I already talked about in 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 table Track TrackID is PK. length which tells us track length. Table Track has One-to-Many relationship with table Repair and has Many-to-Many relationship with table Lines.

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) |

Table tunnel has TrackID as a PK and LineID and non-prime attributes ConstructionDate and TunnelCondition., It has One-to-Many relationship with table Lines and that’s why we need LineID in this table and has One-to-Many relationship with table Repair.

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 |

Table Repair has RepairID as PK, attribute description explains what was damaged and where, RepairDate says what is it for so does cost. This table has four One-to-Many relationship with table Station,Track,Tunnel,Train.

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. 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 |

Summary: to sum up everything that I change it would be that I add some constrains for attributes, also changed the structure of the graph.

You made 8 comments about what I have done wrong and I will answer those comments now.

1. I added example data tables for all tables now.
2. About the dual foreign keys. I changed the One-to-One relationship between Table repair and Tables track,tunnel,station and train and made One-to-Many relationship. So because of that I added these four tables primary keys in table repair to keep track what is damaged.
3. I think I also talked about that
4. I deleted the bridge table between table Train and table Lines, because I think we will still have the kind of information what we need without that bridge table. Because we have Train->TrainStation->Station->StationLines->Lines.
5. Now I have no relationship between table Ticket and Table train because that business model was unrealistic and added attributes that you told me to add. And added some realistic attributes for table Tunnel and table Track.
6. I added constraints for some attributes.
7. I corrected my mistake and changed name to TrackLines
8. I changed attributes NumberOfTrains and NumberOfEmployees to RequiredNumberOfTrains and RequiredNumberOfEmployees so now we know maximum how many trains and employees do we need.