Data and Applications – Project

Introduction to mini-world:

Our mini world is the South-Central Railway System.

Purpose of the Database:

This database gives the information and details about the Railway Timings, Train availability, and Ticketing information of the passengers.

Users of the Database:

- Passengers boarding the railways
- Employees working in the railway department
- Maintainers of the railway database

Applications of the Database:

- Storing and processing data about the running trains and travelling passengers can be done efficiently using this approach
- Ticket cancellations and bookings can be handled in an organised manner
- The queries/requests by the users to enquire about the available trains can be efficiently processed using different filters and searching techniques
- We can perform several statistical analyses and predict the train availability on a particular day, the most boarded train in a season

Database Requirements:

Assumptions:

- All the names of the trains and stations are distinct
- There exists at least one train between any two stations
- An employee can work only in one station
- The capacity of one compartment of a train is 8 passengers

Strong Entity Types:

- TRAINs: contains info about the trains
  - Train Name (atomic and key attribute and primary key): name of the train [Char(50), NOT NULL, unique]
  - Train No (atomic and key attribute): age of the player [Int(20), NOT NULL, unique]
- STATIONs: contains info about railway stations
  - Station Name (key and atomic attribute and primary key): name of the station [Char(50), NOT NULL, unique]

- Station Id (key and atomic attribute): Id assigned to the station [Int(10), NOT NULL, unique]

- Station Master (composite attribute): master of the station [Char(20), NOT NULL, unique]

- TICKETs: contains information about all possible tickets
  - Source Station (atomic attribute): starting point of the journey [Char(50), NOT NULL]
  - Destination Station (atomic attribute): ending point of the journey [Char(50), NOT NULL]
  - Price (atomic attribute): Price of the ticket [Int(10), NOT NULL]
- RAILWAY-QUARTERs: contains data about the accommodation of the RAILWAY-EMPLOYEEs
  - Quarters-Number (atomic and key attribute and primary key): ID of the quarters [Int(10), NOT NULL, unique]
  - Location (composite and key attribute): address of the quarters [Char(100), NOT NULL, unique]

Weak Entity Types:

- RAILWAY-EMPLOYEEs: employees working in the railway-dept
  - Name (composite attribute): name of the employee [Char(20), NOT NULL]
  - Designation (atomic attribute): designation of the employee [Char(20), NOT NULL]
  - Age (atomic attribute): age of the employee [Int(10), NOT NULL]
- PASSENGERs: contains data about the passengers
  - Name (composite attribute): name of the passenger [Char(20), NOT NULL]
  - DOB (atomic attribute): date of birth of the passenger [Char(10), NOT NULL]
  - Age (derived attribute): age of the passenger [Int(10), NOT NULL]
  - Date of Journey (atomic attribute): date of journey of the passenger [Char(10), NOT NULL]
  - Destination (atomic attribute and foreign key): Destination station of the passenger [Char(50), NOT NULL]
  - Seat Information (composite attribute): coach and seat number of the passenger [Char(50), NOT NULL]

Relationship Types:

- Trains **run through** Stations
  1) Degree = 2
  2) Participating entity types: Trains, Stations
  3) Cardinality ratio = N:M
  4) Constraints: Trains (1, M) - Stations (1, N)
  5) Participation constraint = Trains (Total) – Stations (Total)
- Railway-Employees **work in** Stations
  1) Degree = 2
  2) Participating entity types: Railway-Employees, Stations

3) Cardinality ratio = N:1
4) Constraints: Railway-Employees (1, 1) - Stations (1, N)
5) Participation constraint = Railway-Employees (Total) – Stations (Total)

- Passengers **belong to the same** compartment
  1) Degree = 2 (Recursive Relationship Type)
  2) Participating entity types = Audience, Seats
  3) Cardinality ratio = 1:1

4) Constraints: Passengers (1,1) - Passengers (1,1)
5) Participation constraint = Passengers (Total) – Passengers (Total)

Quaternary Relationship type:

- The journey of a **passenger** is described with a **ticket**, in a **train**, through **stations**
  1) Degree = 4
  2) Participating entity types = Passengers, Trains, Stations, Tickets
  3) Cardinality ratio = N:M:K:1
  4) Constraints: Passengers (1, 1) - Tickets (1, 1) - Trains (1, N) – Stations (1, M)
  5) Participation constraint = Passengers (Total) - Tickets (Total) - Trains (Total) – Stations (Total)

Functional Requirements:

1) Retrieval:

a) Selection:
- Retrieving the data of the passengers travelling from Bangalore to Chennai on a particular day (Diwali day for example)
- Retrieving the Number of trains available from source to destination with respect to user on a particular day
- Retrieving the data of the passenger using PNR number

b) Projection:
- Number of trains departing from a particular source at a particular time

c) Aggregate:
- Stations from which, maximum/minimum number of passengers are travelling so that the number of trains between those two stations can be increased/decreased
- On which day of the week (including the time of the day) the trains are running at high capacity

d) Search:
- Search for all the names of trains and stations starting with the letter 'S'
- Search for a passenger's data using the first few digits of his PNR number

e) Analysis:
- Number of trains that ran on time without any delay over a past period of time

2) Modifications

a) Insert:
- Insert the data in the PASSENGERs entity type

Verification of Constraints:

1. Referential Integrity Constraint:
   In the passengers entity type, there exists only one foreign key "Destination" which refers to the "Station Names" attribute in the stations entity type.
   If an insert operation is performed to passengers such that the destination is not present in the station names, then it is a violation of Referential Integrity

Constraint because the newly inserted destination name doesn't refer to any station name.

2. Entity Integrity Constraint:
   There exists no primary keys in passengers entity type since it is a weak attribute. Hence Entity Integrity Constraint is not violated.
3. Domain Constraint:
   If an insertion is made such that the values are out of the domain, then the Domain Constraint is violated.

b) Update:

- Update the new passenger's information for a particular PNR after the journey of the previous passenger is finished.

Verification of Constraints:

1. Referential Integrity Constraint:
   In the passengers entity type, there exists only one foreign key "Destination" which refers to the "Station Names" attribute in the stations entity type.
   If an insert operation is performed to passengers such that the destination is not present in the station names, then it is a violation of Referential Integrity Constraint because the newly inserted destination name doesn't refer to any station name.
2. Entity Integrity Constraint:
   There exists no primary keys in passengers entity type since it is a weak attribute. Hence Entity Integrity Constraint is not violated.
3. Domain Constraint:
   If an insertion is made such that the values are out of the domain, then the Domain Constraint is violated.

c) Delete:

- If a passenger has cancelled his ticket, his data is deleted from the PASSENGERs entity type
- Delete a station name from station entity type

Verification of Constraints:

1. Referential Integrity Constraint:
   If a station name is deleted from the stations entity type, then the destination (foreign key attribute in the passengers entity type) that was referring to this station name might not be referring to anything now. Hence Referential Integrity Constraint can be violated.

Summary:

Our database represents the South-Central Railway System. It stores all the relevant and necessary data of the Railways.