RAILWAY RESERVATION SYSTEM

ABSTRACT

The Railway Reservation System facilitates the passengers to enquire about the trains available based on source and destination, Booking and Cancellation of tickets, enquire about the status of the booked ticket, etc. The aim of the case study is to design and develop a database maintaining the records of different trains, train status, and passengers.

This project contains Introduction to the Railways reservation system. It is the computerized system of reserving the seats of train seats in advanced. Online reservation has made the process for the reservation of seats very much easier than ever before.

Then this project contains an entity-relationship model diagram based on the railway reservation system and an introduction to the relation model. There is also the design of the database of the railway reservation system based on the relation model. Example of some SQL queries to retrieve data from rail management database.

INTRODUCTION

Database is an organized collection of data. The data is typically organized to model aspects of reality in a way that supports processes requiring information.

A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data.

The main purpose of maintaining database for Railway Reservation System is to reduce the manual errors involved in the booking and cancelling of tickets and make it convenient for the customers and providers to maintain the data about their customers and about the seats available at them. Due to automation many loopholes that exist in the manual maintenance of the records can be removed. The speed of obtaining and processing the data will be fast. For future expansion the proposed system can be web enabled so that clients can make various enquiries about trains between stations. Due to this, sometimes a lot of problems occur, and they face many disputes with customers. To solve the above problem, we have designed a database which includes customer details, availability of seats on trains, no of trains and their details.

PROJECT DESCRIPTION

This project is about creating the database about Railway Reservation System.

The railway reservation system facilitates the passengers to enquire about the trains available based on source and destination, booking and cancellation of tickets, enquire about the status of the booked ticket, etc. The aim of this study is to design and develop a database maintaining the records of different trains, train status, and passengers. The record of the train includes its number, name, source, destination, and the number of seats available. Passengers can book their tickets for the train in which seats are available.

For this, the passenger must provide the desired train number, desired stations of travel and the date for which the ticket is to be booked. Before booking a ticket for a passenger, the validity of the train number and booking date is checked. Once the train number and booking date are validated, it is checked whether the seat is available. If yes, the ticket is booked with confirm status and corresponding ticket ID is generated which is stored along with other details of the passenger. The ticket once booked can be cancelled at any time. For this, the passenger must provide the TicketId (the unique key). The TicketId is searched, and the corresponding record is deleted. The list of assumptions is very large since the reservation system is very large in reality, it is not feasible to develop the case study to that extent and prepare documentation at that level. Therefore, a small sample case study has been created to demonstrate the working of the reservation system.

List of trains must be maintained. Detailed passenger information is to be maintained in the booking procedure, the train number, train date, and Seat number are read from the passenger. Based on the values provided by the passenger, the corresponding record is retrieved from the Ticket. If a ticket can be booked, then passenger details are read and stored in the Ticket table. In the cancellation procedure, TicketId is read from the passenger and corresponding record is searched in the Passenger. If the record exists, it is deleted, and the refund is processed.

Introduction to the Organization:

The railway reservation system aims to automate the booking and cancellation processes for passengers, reducing manual errors and improving efficiency. It also provides facilities for passengers to enquire about train availability, status, and make reservations conveniently.

Problems the Database Solves:

- Manual errors in booking and cancellation of tickets.
- Inefficient management of train availability and passenger records.

- Lack of real-time information for passengers and staff.
- Difficulty in handling ticket cancellations.

Related Business Rules:

1. A user can make only one booking using this system.
2. Trains are listed as based on daily operation and only five trains are taken as sample.
3. Seat Availability is for the assumed date of operation of the train.
4. Only one category of seat is mentioned.
5. A ticket will be issued upon successful reservation including passenger details.
6. A booked ticket can be cancelled providing the paymentId.

Related Use Cases:

- Train Enquiry: Passengers can enquire about trains based on source and destination.
- Ticket Booking: Passengers can book tickets for available trains.
- Ticket Cancellation: Passengers can cancel their booked tickets.

Conceptual Database Design:

The following ER Diagram shows the relationship between different entities:

One to One relation:

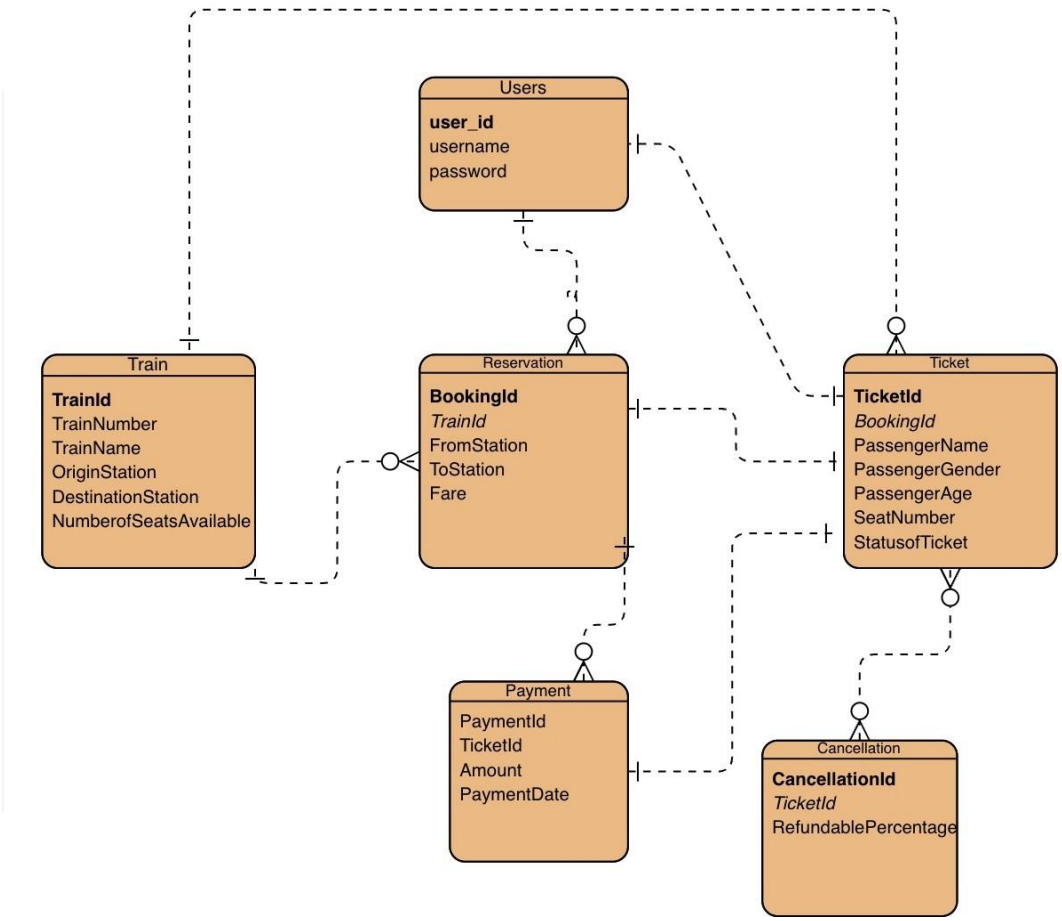
- Reservation – Ticket: This relation indicates that each reservation corresponds to exactly one ticket, and vice versa.
- Ticket - Payment: Similarly, this relation implies that each ticket is associated with exactly one payment, and each payment corresponds to exactly one ticket.
- User – Ticket: Each user can have a single ticket while each ticket is associated with one user.

One to Many Relations:

- User - Reservation: This relation suggests that each user can have multiple reservations, indicating a one-to-many relationship between users and reservations.
- Reservation - Payment: This relation implies that each reservation can have multiple payments (e.g., for partial payments or installment plans), suggesting a one-to-many relationship between reservations and payments.
- Train - Reservation: This relation indicates that each train can have multiple reservations, representing a one-to-many relationship between trains and reservations.
- Train - Ticket: Similarly, each train can have multiple tickets associated with it, suggesting a one-to-many relationship between trains and tickets.

Many to Many relations:

- Ticket – Cancellation: This relation suggests a many-to-many relationship between tickets and cancellations, indicating that multiple tickets can be associated with multiple cancellations, and vice versa.



Logical Database Design:

ENTITIES	ATTRIBUTES
Users	user_id username password

Train	TrainId TrainNumber TrainName OriginStation DestinationStation NumberofSeatsAvailable
Reservation	BookingId TrainId {FK} FromStation ToStation TravelDate Fare
Ticket	TicketId BookingId {FK} PassengerName PassengerGender PassengerAge SeatNumber StatusofTicket
Payment	PaymentId TicketId {FK} Amount PaymentDate
Cancellation	CancellationId TicketId {FK} RefundablePercentage

Physical Database Design:

(1) TABLE Users:

```sql
CREATE TABLE Users (
    user_id INT PRIMARY KEY,
    username VARCHAR(50) UNIQUE,
    password VARCHAR(100)
)
```

```
Table created.
```

```sql
INSERT INTO Users(user_id, username, password) VALUES (1, 'Suhani Shashwat', 'SSH');
INSERT INTO Users(user_id, username, password) VALUES (2, 'Milan Patel', 'MPP');
INSERT INTO Users(user_id, username, password) VALUES (3, 'Vishal Mishra', 'VIMR');
INSERT INTO Users(user_id, username, password) VALUES (4, 'Sagar Thakre', 'STRK');
INSERT INTO Users(user_id, username, password) VALUES (5, 'Kashish Nai', 'KASN');
INSERT INTO Users(user_id, username, password) VALUES (6, 'Mahek Desai', 'MAHSE');
INSERT INTO Users(user_id, username, password) VALUES (7, 'Yugesha Karkar', 'YKAR');
INSERT INTO Users(user_id, username, password) VALUES (8, 'Chirag Muntashir', 'CHMUR');
INSERT INTO Users(user_id, username, password) VALUES (9, 'Aditya Sanap', 'ADTDP');
INSERT INTO Users(user_id, username, password) VALUES (10, 'Dhairya Amlani', 'DHAM');

```

(2) TABLE Train:

```sql
CREATE TABLE Train (
    TrainId INT PRIMARY KEY,
    TrainNumber VARCHAR(20),
    TrainName VARCHAR (20),
    OriginStation VARCHAR(100),
    DestinationStation VARCHAR(100),
    NumberofSeatsAvailable INT )
```

```
Table created.
```

```
INSERT INTO Train (TrainId, TrainNumber, TrainName, OriginStation, DestinationStation, NumberofSeatsAvailable)
VALUES (1, '101', 'Vivek Express', 'Dibrugarh', 'KanyaKumari', 48);
INSERT INTO Train (TrainId, TrainNumber, TrainName, OriginStation, DestinationStation, NumberofSeatsAvailable)
VALUES (2, '102', 'Kutch Express', 'Madgaon', 'Kutch', 28);
INSERT INTO Train (TrainId, TrainNumber, TrainName, OriginStation, DestinationStation, NumberofSeatsAvailable)
VALUES (3, '103', 'Tejas Express', 'Mumbai Central', 'Amritsar Junction', 59);
INSERT INTO Train (TrainId, TrainNumber, TrainName, OriginStation, DestinationStation, NumberofSeatsAvailable)
VALUES (5, '105', 'Vande Bharat Express', 'Ajmer Junction', 'Kolkata Junction', 10);
INSERT INTO Train (TrainId, TrainNumber, TrainName, OriginStation, DestinationStation, NumberofSeatsAvailable)
VALUES (4, '104', 'Shatabdi Express', 'Thiruvananthapuram Central', 'Delhi Junction', 63);
```

(3) TABLE Reservation:

```
CREATE TABLE Reservation (
    BookingId INT PRIMARY KEY,
    TrainId INT,
    FromStation VARCHAR (20),
    ToStation VARCHAR (20),
    TravelDate DATE,
    Fare DECIMAL(10,2),
    FOREIGN KEY (TrainId) REFERENCES Train(TrainId)
)
```

```
Table created.
```

```
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (1, 1, 'Navsari', 'Vadodara', TO_DATE('2024-04-20', 'YYYY-MM-DD'), 50.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (2, 2, 'Bandra Junction', 'Amritsar Junction', TO_DATE('2024-04-21', 'YYYY-MM-DD'), 175.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (3, 4, 'Madgaon', 'Jaipur', TO_DATE('2024-04-22', 'YYYY-MM-DD'), 193.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (4, 3, 'Bihar Central', 'Kolkata Junction', TO_DATE('2024-04-23', 'YYYY-MM-DD'), 187.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (5, 5, 'Tamil Nadu Junction', 'Ahmedabad Junction', TO_DATE('2024-04-24', 'YYYY-MM-DD'), 230.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (6, 2, 'Pune', 'Delhi Central', TO_DATE('2024-04-25', 'YYYY-MM-DD'), 237.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (7, 4, 'Rajkot', 'Kerela', TO_DATE('2024-04-26', 'YYYY-MM-DD'), 276.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (8, 1, 'Punjab', 'Porbandar', TO_DATE('2024-04-27', 'YYYY-MM-DD'), 198.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (9, 3, 'Ghatkopar', 'Chennai', TO_DATE('2024-04-28', 'YYYY-MM-DD'), 149.00);
INSERT INTO Reservation (BookingId, TrainId, FromStation, ToStation, TravelDate, Fare)
VALUES (10, 5, 'Assam', 'Indore', TO_DATE('2024-04-29', 'YYYY-MM-DD'), 288.00);
```

(4) TABLE Ticket:

```sql
CREATE TABLE Ticket (
    TicketId INT PRIMARY KEY,
    BookingId INT,
    PassengerName VARCHAR(30),
    PassengerGender VARCHAR (10),
    PassengerAge INT,
    SeatNumber INT,
    StatusofTicket VARCHAR(10),
    FOREIGN KEY (BookingId) REFERENCES Reservation(BookingId)
    )
```

```
Table created.
```

```sql
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (11, 1, 'Disha Patel', 'Female', 30, 12, 'Confirmed');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (12, 2, 'Krupa Shah', 'Female', 21, 33, 'Waiting');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (13, 3, 'Mack Joe', 'Male', 40, 78, 'Confirmed');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (14, 4, 'John Doe', 'Male', 53, 33, 'Waiting');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (15, 5, 'Sonal Shukla', 'Female', 27, 67, 'Waiting');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (16, 6, 'Ruchi Desai', 'Female', 33, 19, 'Confirmed');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (17, 7, 'Vishva Amlani', 'Female', 28, 45, 'Confirmed');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (18, 8, 'Tanvi Sai', 'Female', 22, 77, 'Waiting');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (19, 9, 'Harsh Joshi', 'Male', 51, 59, 'Confirmed');
INSERT INTO Ticket (TicketId, BookingId, PassengerName, PassengerGender, PassengerAge, SeatNumber, StatusofTicket)
VALUES (20, 10, 'Medha Patil', 'Female', 48, 32, 'Waiting');
```

(5) TABLE Payment:

```sql
CREATE TABLE Payment (
    PaymentId INT PRIMARY KEY,
    TicketId INT,
    Amount DECIMAL(10,2),
    PaymentDate TIMESTAMP,
    FOREIGN KEY (TicketId) REFERENCES Ticket(TicketId)
)
```

```
Table created.
```

```
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (1, 11, 50.00, TO_TIMESTAMP('2024-04-20 10:30:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (2, 12, 175.00, TO_TIMESTAMP('2024-04-21 10:31:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (3, 13, 193.00, TO_TIMESTAMP('2024-04-22 10:32:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (4, 14, 187.00, TO_TIMESTAMP('2024-04-23 10:33:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (5, 15, 230.00, TO_TIMESTAMP('2024-04-24 10:34:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (6, 16, 237.00, TO_TIMESTAMP('2024-04-25 10:35:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (7, 17, 276.00, TO_TIMESTAMP('2024-04-26 10:36:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (8, 18, 198.00, TO_TIMESTAMP('2024-04-27 10:37:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (9, 19, 149.00, TO_TIMESTAMP('2024-04-28 10:38:00', 'YYYY-MM-DD HH24:MI:SS'));
INSERT INTO Payment (PaymentId, TicketId, Amount, PaymentDate)
VALUES (10, 20, 288.00, TO_TIMESTAMP('2024-04-29 10:39:00', 'YYYY-MM-DD HH24:MI:SS'));
```

(6) TABLE Cancellation:

```
CREATE TABLE Cancellation (
    CancellationId INT PRIMARY KEY,
    PaymentId INT,
    RefundablePercentage DECIMAL(5,2),
    FOREIGN KEY (PaymentId) REFERENCES Payment(PaymentId)
)
```

```
Table created.
```

```
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (1, 1, 50.00);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (2, 2, 25.00);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (3, 3, 17.65);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (4, 4, 45.70);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (5, 5, 58.90);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (6, 6, 71.28);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (7, 7, 39.57);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (8, 8, 27.89);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (9, 9, 120.23);
INSERT INTO Cancellation (CancellationId, PaymentId, RefundablePercentage) VALUES (10, 10, 156.30);
```

CONCLUSION

In conclusion, the railway ticket reservation system offers a comprehensive and userfriendly solution for managing ticket bookings, ensuring smooth operations and optimal user experience within the railway service domain.