A logo of a globe with yellow rings around it

Description automatically generated

Group Assignment

Technology Park Malaysia

**CT042-3-1-IDB**

Introduction to Database

Date: 7/16/2024

Lecturer Name: Lai Chew Ping

Group Number: 19

Group Members:

|  |  |  |
| --- | --- | --- |
| **No.** | **Student Name** | **TP Number** |
| 1. | Ruba Munaf Hasan Alshaikh | TP078528 |
| 2. | Manreen Kaur A/P Jagjit Singh | TP071290 |
| 3. | Hasibul Islam | TP077662 |
| 4. | Ahmed AlaaEldin Mohmed Elhusseini Seada | TP077921 |

Leader Name: Ruba Munaf Hasan Alshaikh

[Introduction 1](#_Toc1647378128)

[Database and Database Management System 1](#_Toc968530931)

[Disadvantages of the file-based system, relate your discussion accordingly to the case study 1](#_Toc1215837314)

[Advantages of Database and DBMS, relate your discussion accordingly to the case study 1](#_Toc1556470483)

[Business Rules & Normalization 1](#_Toc1110383046)

[Generate a list of complete business rules 1](#_Toc2086825060)

[Provide an example of UNF and perform normalization up to 3NF clearly showing all the steps with an explanation 1](#_Toc255608222)

[Unnormalized Form (UNF) 1](#_Toc1324682977)

[First Normal Form (1NF) 1](#_Toc1936452798)

[Second Normal Form(2NF) 1](#_Toc1653447292)

[Third Normal Form (3NF) 1](#_Toc1701051436)

[Explanation of the Step 1](#_Toc629407847)

[1) Unnormalized Form to First Normal Form (UNF to 1NF): 1](#_Toc743747906)

[2) First Normal Form to Second Normal Form (1NF to 2NF): 1](#_Toc1282572529)

[3) Second Normal Form to Third Normal Form (2NF to 3NF): 1](#_Toc904203943)

[1) Unnormalized Form to First Normal Form (UNF to 1NF) 1](#_Toc2044488721)

[2) First Normal Form to Second Normal Form (1NF to 2NF) 1](#_Toc21248347)

[3) Second Normal Form to Third Normal Form (2NF to 3NF) 1](#_Toc1495331254)

[Entity Relationship Diagram 1](#_Toc1154059856)

[References: 1](#_Toc636144755)

# Introduction

Keretapi Tanah Melayu (KTM), Malaysia’s premier train service, operates three major routes: the Northern Track (NA1709), Southern Track (SA1865), and East Coast Track (CA8760). These routes connect key states, including Kuala Lumpur, Perak, Pulau Pinang, Kedah, Perlis, Melaka, Negeri Sembilan, Johor, Pahang, Terengganu, and Kelantan. Each track features multiple daily trains, offering both business and economy classes.

Trains are staffed by a captain, co-captain, and up to three attendants, ensuring a high standard of service. Passengers can start their journey from any station and book multiple trips. Customer details, including names, addresses, phone numbers, and emails, are managed in a comprehensive reservation system. Each reservation includes a unique number, travel dates, train details, seat numbers, and total charges. This central database streamlines bookings, enhancing operational efficiency and customer satisfaction for KTM’s extensive rail network.

# Database and Database Management System

## **Disadvantages of the file-based system, relate your discussion accordingly to the case study**

### 

A file-based system is a straightforward system organizing data into individual files stored in a file system. Nonetheless, this approach exhibits several disadvantages detrimental in complex and dynamic situations, especially in the train reservation system for Keretapi Tanah Melayu (KTM).

First and foremost, data redundancy and inconsistency are major pitfalls in file-based systems. In KTM’s situation, the lack of a centralized database implies that most files contain duplicate passenger data, train schedules, and reservation details. These files lack indexing, necessitating searches of multiple files for information. Besides, this redundancy consumes storage capacity and boosts the likelihood of inconsistent outcomes if updates are not executed consistently on all pertinent files. For instance, manually updating booking information across all pertinent files can be laborious and error-prone if passengers change their information. (GeeksforGeeks, 2022)

Plus, data discrepancy induces duplicate data entries, raising the probability of future inconsistencies. KTM’s repeated entries of the same passenger or train schedule produce booking errors, overbooked trains, or canceled reservations, which frustrates clients and leads to complaints and loss of business. Additionally, inconsistencies in the system reduce data quality. For example, inaccurate passenger records and erroneous schedule information will cause a decline in service quality, contributing to customer dissatisfaction. (GeeksforGeeks, 2023)

For Keretapi Tanah Melayu (KTM), transitioning from a file-based system to a Database Management System (DBMS) is essential to tackling the issues brought on by data redundancy and inconsistency. Maintaining consistency is a recurring problem in KTM's present file-based architecture, as reservation and passenger records are replicated across many files. Manual synchronization across separate files is necessary for updates or alterations to customer records or train schedules, which increases the risk of errors and operational inefficiencies. These discrepancies have an immediate impact on customer satisfaction in addition to internal processes. Inaccuracies in reservation information or out-of-date client records may cause delays, confusion while traveling, and passenger discontent. Such errors affect KTM's attempts to deliver dependable service and continually satisfy consumer expectations. Data redundancy causes data inconsistency. Using the previous example, we have two addresses stored for a student who is enrolled in two courses. If the student requests to change his address, only one of the addresses is updated, which could result in data inconsistency. (Beginners Book, 2015)

## **Advantages of Database and DBMS, relate your discussion accordingly to the case study**

A Database Management System (DBMS) provides a robust solution for managing the intricacies of the KTM train reservation system by providing a centralized and structured approach to managing data. Several advantages are found in using database management systems in managing a train reservation system for Keretapi Tanah Melayu (KTM).

Data integrity is primarily enhanced by the features of a Database Management System (DBMS), such as transactions and relational constraints. By enforcing regulations such as mandating unique constraints, such as email addresses for clients will mitigate the risk of data redundancy and inconsistency in DBMS. Furthermore, maintaining referential integrity between related databases such as reservations and train schedules guarantees that the data remains consistent and accurate throughout the system. (Geeksforgeeks,2019) This dependability is vital for KTM to manage accurate records of passenger reservations and operational schedules. Transactions in a DBMS ensure that all operations are executed completely and correctly, or not at all, preventing incomplete updates that could result in data inconsistencies. Relational constraints and transaction management guarantee precise and reliable data handling for improved data integrity. (Naeem, 2024)

These factors highlight why transitioning from a file-based system to a strong database management system would enhance KTM's train reservation operations by boosting data management, security, and operational efficiencies.

One other advantage of the DBMS for KTM is scalability. The DBMS uses effective indexing and query optimization algorithms to optimize data retrieval and administration as database size and transaction volume expand. This scalability guarantees quick reaction times and seamless operations even during periods of high travel demand or system expansions. Furthermore, data consistency and integrity are improved by DBMS's centralized architecture. The uniform application of updates and revisions to all pertinent records reduces the possibility of inaccuracies or conflicts in the data. This dependability is essential to maintaining high service standards and operational effectiveness, freeing KTM to concentrate on providing high-quality service without being impeded by difficulties with data administration. Data is managed using database systems to ensure quick response times and easy accessibility. The DBMS can still provide faster data access and update even with large databases. (Beginners Book, 2015)

# Business Rules & Normalization

## **Generate a list of complete business rules**

A list of complete business rules is listed below:

1. A train reservation can be made for one or more trains.
2. Each train route is assigned a unique train number.
3. The train includes various classes, including business and economy, each with a particular

number of seats.

1. Each train route operates on one track: the Northern Track, the Southern Track, or the East

Coast Track.

1. Each track has a varied number of trains traveling every day.
2. Only one train departs a single station at the same time and date.
3. A train must be operated by two people which are the captain and the co-captain.
4. Each train can have a maximum of three attendants manning the train.
5. Train operators may operate numerous trains.
6. Train operators must complete 20,000 operation hours to be positioned as senior operators.
7. Each train operator is assigned a unique employment number and associated details that include name, age, experience, and salary.
8. Each train attendant is assigned a unique ID which contains their name, position, salary, phone number, and residence address.
9. Customers can travel from any state or city in Malaysia.
10. Customers can reserve multiple trains in a single booking.
11. Each customer has a unique email address, but their first and last names are not required to be unique
12. A reservation has a unique reservation number and details including reservation status, reservation date, travel date, train number, reserved seats, departure and destination stations, and total charges.
13. More than twenty customers book trains within the same month.
14. Each train station represents a state and serves as a halt for trains on that path.
15. Customers may have zero or more phone numbers and email addresses.
16. Customers may or may not have made bookings at any given time.
17. The system must keep track of the overall number of seats in business and economy classes on each train.
18. The system must maintain track of the train operators and attendants assigned to each train.

## **Provide an example of UNF and perform normalization up to 3NF clearly showing all the steps with an explanation**

## **Unnormalized Form (UNF)**

|  |  |  |
| --- | --- | --- |
| Reservation \_Number | 1 | 2 |
| Custmer\_ID | 101 | 102 |
| Customer\_Name | John Doe | Jane Smith |
| Email | [john@mail.com](mailto:john@mail.com) | |  | | --- | | [jane@mail.com](mailto:jane@mail.com) | |
| Train\_Number | NA1709 | SA1865 |
| Train\_Details | Date: 2024-07-01,  Time: 09:00,  Seats: 5E, 10B | Date: 2024-07-02,  Time: 10:00,  Seats: 7E, 12B |
| Opretor\_Details | Captain: Alice,  Co-Captain: Bob | Captain: Charlie,  Co-Captain: David |
| Attendent\_Detail | Attendant: Eve,  Attendant: Adam | Attendant: Eve,  Attendant: Frank |
| Departure\_Station | Kuala Lumpur | Kuala Lumpur |
| Arrival\_Station | Perlis | Johor |
| Reservation\_Date | 2024-06-25 | 2024-06-26 |
| Traveling\_Date | 2024-7-01 | 2024-07-02 |
| Total\_Charges | 150.00 | 200.00 |

Table Unnormalized Form (UNF)

### **First Normal Form (1NF)**

This table is named UNF and has different fields with repeating groups such as Train\_Details, Operator\_Details, and Attendant\_Details.  
  
undefined  
  
1NF bans repeating groups and as such, each field that is a member of a composite value contains atomic values only. We will now split the Train\_Details, Operator\_Details, and Attendant\_Details attributes into three different attributes.

**Train\_Reservation (1NF)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Reservation\_Number** | 1 | 1 | 2 | 2 |
| **Customer\_ID** | 101 | 101 | 102 | 102 |
| **Customer\_Name** | John Doe | John Doe | Jane Smith | Jane Smith |
| **Email** | [john@mail.com](mailto:john@mail.com) | [john@mail.com](mailto:john@mail.com) | [jane@mail.com](mailto:jane@mail.com) | [jane@mail.com](mailto:jane@mail.com) |
| **Train\_Number** | NA1709 | NA1709 | SA1865 | SA1865 |
| **Date** | 2024-07-01 | 2024-07-01 | 2024-07-02 | 2024-07-02 |
| **Time** | 09:00 | 09:00 | 10:00 | 10:00 |
| **Seat\_Class** | Economy | Business | Economy | Business |
| **Seat\_Number** | 5E | 10B | 7E | 12B |
| **Departure\_Station** | Kuala Lumpur | Kuala Lumpur | Kuala Lumpur | Kuala Lumpur |
| **Arrival\_Station** | Perlus | Perlus | Johor | Johor |
| **Reservation\_Date** | 2024-06-25 | 2024-06-25 | 2024-06-26 | 2024-06-26 |
| **Traveling\_Date** | 2024-07-01 | 2024-07-01 | 2024-07-02 | 2024-07-02 |
| **Total\_Charges** | 150.00 | 150.00 | 200:00 | 200:00 |

Table First Normal Form (1NF): Train\_Reservation

### **Second Normal Form(2NF)**

2NF helps to control the partial dependencies by making sure all the non-key attributes are fully reliant on the primary key. In our case A needs to be partitioned in more than two ways to make all the attributes fully functionally dependent on the primary key.

**Reservation:**

|  |  |  |
| --- | --- | --- |
| **Reservation\_Number** | 1 | 2 |
| **Customer\_ID** | 101 | 102 |
| **Train\_Number** | NA1709 | SA1865 |
| **Reservation\_Date** | 2024-06-25 | 2024-06-26 |
| **Traveling\_Date** | 2024-07-01 | 2024-07-02 |
| **Total\_Charges** | 150:00 | 200:00 |

Table Second Normal Form (2NF): Reservation

**Customer:**

|  |  |  |
| --- | --- | --- |
| **Customer\_ID** | 101 | 102 |
| |  | | --- | | **Customer\_Name** | | John Doe | [john@mail.com](mailto:john@mail.com) |
| **Email** | Jane Smith | [jane@mail.com](mailto:jane@mail.com) |

Table Second Normal Form (2NF): Customer

**Train\_Details:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train\_Number** | NA1709 | NA1709 | SA1865 | SA1865 |
| **Date** | 2024-07-01 | 2024-07-01 | 2024-07-02 | 2024-07-02 |
| **Time** | 09:00 | 09:00 | 10:00 | 10:00 |
| **Seat\_Class** | Economy | Business | Economy | Business |
| **Seat\_Number** | 5E | 10B | 7E | 12B |
| **Departure\_Station** | Kuala Lumpur | Kuala Lumpur | Kuala Lumpur | Kuala Lumpur |
| **Arrival\_Station** | Perlis | Perlis | Johor | Johor |

Table Second Normal Form (2NF): Train\_Details

**Operator:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train\_Number** | NA1709 | NA1709 | SA1865 | SA1865 |
| **Role** | Captain | Co-Captain | Captain | Co-Captain |
| | **Operator\_Name** | | --- | | Alice | Bob | Charlie | David |

Table Second Normal Form (2NF): Operator

**Attendant:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train\_Number** | NA1709 | NA1709 | SA1865 | SA1865 |
| **Attendant\_Name** | Eve | Adam | Ahmed | Frank |

Table Second Normal Form (2NF): Attendant

### **Third Normal Form (3NF)**

3NF minimizes transitive dependencies deleting dependencies wherein non-key attributes rest on other attributes that are not the primary key.

**Reservation:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Reservation\_Number** | **Customer\_ID** | **Train\_Number** | **Reservation\_Date** | **Traveling\_Date** | **Total\_Charges** |
| 1 | 101 | NA1709 | 2024-06-25 | 2024-07-01 | 150:00 |
| 2 | 102 | SA1865 | 2024-06-26 | 2024-07-02 | 200:00 |

Table Third Normal Form (3NF): Reservation

**Customer:**

|  |  |  |
| --- | --- | --- |
| **Customer\_ID** | **Customer\_Name** | **Email** |
| 101 | John Doe | [john@mail.com](mailto:john@mail.com) |
| 102 | Jane Smith | [jane@mail.com](mailto:jane@mail.com) |

Table Third Normal Form (3NF): Customer

**Train:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train\_Number** | **Departure\_Date** | **Departure\_Time** | **Departure\_Station** | **Arrival\_Station** |
| NA1709 | 2024-07-01 | 09:00 | Kuala Lumpur | Perlis |
| SA1865 | 2024-07-02 | 10:00 | Kuala Lumpur | Johor |

Table Third Normal Form (3NF): Train

**Seat:**

|  |  |  |
| --- | --- | --- |
| **Train\_Number** | **Seat\_Class** | **Seat\_Number** |
| NA1709 | Economy | 5E |
| NA1709 | Business | 10B |
| SA1865 | Economy | 7E |
| SA1865 | Business | 12B |

Table Third Normal Form (3NF): Seat

**Operator:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator\_ID** | **Train\_Number** | **Role** | **Operator\_Name** | **Lead\_Hours** | **Salary** |
| 1 | NA1709 | Captain | Alice | 25000 | 8000RM |
| **2** | **NA1709** | **Co-Captain** | **Hazil** | **20000** | **6000RM** |
| **3** | **SA1865** | **Captain** | **Ahmed** | **22000** | **7500RM** |
| **4** | **SA1865** | **Co-Captain** | **David** | **21000** | **6500RM** |

Table Third Normal Form (3NF): Operator

**Attendant:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attendant\_ID** | **Train\_Number** | **Attendant\_Name** | **Phone\_Number** | **Address** | **Salary** |
| 1 | NA1709 | Nor | 0116652524 | 221 Add St | 3000RM |
| 2 | NA1709 | Azul | 0104453288 | 332 Ola St | 3000RM |
| 3 | SA1865 | Frank | 0172233929 | 565 Main St | 3000RM |
| 4 | SA1865 | Eve | 0128859409 | 679 Klo St | 3500RM |

Table Third Normal Form (3NF): Attendant

### **Explanation of the Step**

#### **Unnormalized Form to First Normal Form (UNF to 1NF):**

1. Eliminate the repeating groups by assigning each of them into new rows and retain all the other fields holding atomic data.
2. Often in the Unnormalized Form (UNF) of data, one can find repeating groups as well as nested structures. This makes it impossible to query and maintain the integrity of data that is stored inside a relational database. For a table converted to 1NF – it is required that attributes should consist of atomic values only, and all records in the table must be distinguishable.
3. In this case, for our train reservation system, it means separating the repeating groups and locating their values in separate rows. For instance, a single reservation could involve many seats; many operators; or many attendants. In UNF, all these may constitute a single record hence the chances of having duplicate data as well as contradictions.

**To convert to 1NF:**

Identify Repeating Groups: These are often fields comprising several coefficient values or lists. In our case, these object schemes are Seat\_Details, Operator\_Details and Attendant\_Details.

Separate into Multiple Rows: It is necessary to separate each occurrence of a repeating group to its own row. This means that each row should correspond to a single circumstance of the identified entity. For instance, where there are two seats in each reservation, there should be two rows: the first row identifying one seat.

Ensure Atomicity: Every field should not be compound, which means it cannot be split into other fields. For instance, there should be distinct fields: Train\_Details instead of the one which contains both date and time.

#### **First Normal Form to Second Normal Form (1NF to 2NF):**

1. Eliminate all partial dependencies by developing all the non-key attributes of the table into depending on its major key.
2. To get from 1NF to 2NF, some remodeling is required, namely, to handle partial dependencies. A partial dependency occurs where non-key attributes are dependent on some of the composite primary key fields but not all of them. This results in overwriting and creating strange records when updating, inserting, or deleting information.
3. Regarding a composite primary key, it consists of more than one attribute, for instance Reservation\_Number and Seat\_Number in the given case study. Thus, to make all non-key attributes function wholly on the composite primary key, there is a need to remove partial dependencies. This means that each non-key attribute must be linked to the entire record of the primary key attributes rather than a part of it.
4. On the physical level, a person should distribute the data into several interrelated tables to ensure that none of the attributes depends on each other except for the primary key
5. The process of attaining 2NF includes splitting the initial table into various related tables.
6. Therefore, the final table cannot be achieved from the initial table directly. Every table should correspond with one subject or an entity, and any non-key attributes within the tables should be fully functionally dependent on the entire primary key. This involves the determination of entities and the relations between them and the development of a distinct table for the distinct entity.

#### **Second Normal Form to Third Normal Form (2NF to 3NF):**

1. Normalize by removing transitive dependency by making certain that every one of the non-key attributes is not dependent on other non-key attributes and rather is dependent upon the primary key.
2. For ascending from the 2NF to the 3NF it is necessary to remove transitive dependencies. A transitive dependency arises when a non-key field depends on any non-key field other than the primary key field of the table. This tends to create repeated entries and problem areas because changes made to the non-key characteristics may call for alteration in other areas.
3. While analyzing our case study, we must determine whether any of the non-atomic attributes are functionally dependent on at least one other non-key attribute while they should be functionally dependent solely on the primary key. For example, if an attribute such as Customer\_Name depends on the Customer\_ID, and the latter is the primary key of the table, then it is allowed. But if Customer\_Email is dependent on Customer\_Name, where in turn Customer\_Name is dependent on Customer\_ID, we have a transitive dependency.
4. In this case go to a further extent to analyze the tables in a way that each attribute of the relation depends on the primary key only.
5. The third normal form is obtained when we have to split our tables even more to ensure the attributes depend on the primary key and not on any other attribute other than the primary key. The primary key uniqueness also guarantees that the totality of a non-key attribute depends on the given whole primary key and not on part of it.

#### **Unnormalized Form to First Normal Form (UNF to 1NF)**

Unnormalized Form: UNF In UNF, we can indeed have a table like this with repeating groups:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reservation\_ID | |  | | --- | | **Customer\_Name** |  |  | | --- | |  | | |  | | --- | | **Train\_Details** |  |  | | --- | |  | | Seat\_Details | Operator\_Details | Attendant\_Details |
| 1 | John Doe | Train\_1 (2024-07-01) | Seat\_1, Seat\_2 | Operator\_1, Operator\_2 | Attendant\_1, Attendant\_2 |
| 2 | Jane Smith | |  | | --- | | Train\_2 (2024-07-02) |  |  | | --- | |  | | Seat\_3 | Operator\_3 | Attendant\_3 |

Table Unnormalized Form Form to First Normal Form (UNF to 1NF): UNF table

First Normal Form (1NF): We eliminate repeating groups and obtain a relational database with atomic values. Thus, we create separate rows for every repeating group.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Reservation\_ID | Customer\_Name | Train\_Details | Seat\_Details | Operator\_Details | Attendant\_Details |
| 1 | John Doe | Train\_1 (2024-07-01) | Seat\_1 | Operator\_1 | Attendant\_1 |
| 2 | John Doe | Train\_1 (2024-07-01) | Seat\_2 | Operator\_2 | Attendant\_2 |
| 3 | Jane Smith | Train\_2 (2024-07-02) | Seat\_3 | Operator\_3 | Attendant\_3 |

Table Unnormalized Form Form to First Normal Form (UNF to 1NF): 1NF table

#### **First Normal Form to Second Normal Form (1NF to 2NF)**

First Normal Form (1NF): We have the table as shown above.Second Normal Form (2NF): To convert into 2NF, we eliminate partial dependencies. That means a table should have all its non-key attributes fully functionally dependent on the entire primary key. In the given case, identify composite keys and create separate tables to eliminate partial dependencies.Separate the table as given below:

Reservations Table:

|  |  |
| --- | --- |
| **Reservation\_ID** | Train\_Details |
| 1 | John  Doe |
| 2 | Jane Smith |

Table First Normal Form to Second Normal Form (1NF to 2NF): Reservations Table

Train Details Table:

|  |  |
| --- | --- |
| **Reservation\_ID** | Train\_Details |
| 1 | Train\_1 (2024-07-01) |
| 2 | Train\_2 (2024-07-02) |

Table First Normal Form to Second Normal Form (1NF to 2NF): Train Details Table

Seat Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | Seat\_Details |
| 1 | Seat\_1 |
| 2 | Seat\_2 |
| 3 | Seat\_3 |

Table First Normal Form to Second Normal Form (1NF to 2NF): Seat Details Table

Operator Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | Operator\_Details |
| 1 | Operator\_1 |
| 2 | Operator\_2 |
| 3 | Operator\_3 |

Table First Normal Form to Second Normal Form (1NF to 2NF): Operator Details Table

Attendance Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | **Attendant\_Details** |
| 1 | Attendant\_1 |
| 2 | Attendant\_2 |
| 3 | Attendant\_3 |

Table First Normal Form to Second Normal Form (1NF to 2NF): Attendance Details Table

#### 

#### **Second Normal Form to Third Normal Form (2NF to 3NF)**

2NF: We have the following tables as shown above.  
3NF: Turning into 3NF, eliminating the transitive dependencies. That is the dependency of non-key attributes on the primary key alone.Herein, what we must see is that every non-key attribute is dependent upon the primary key only, and not on any other non-key attribute.

Customer Table:

|  |  |
| --- | --- |
| Customer\_ID | Customer\_Name |
| 1 | John  Doe |
| 2 | Jane Smith |

Table Second Normal Form to Third Normal Form (2NF to 3NF): Customer Table

Reservations Table

|  |  |
| --- | --- |
| **Reservation\_ID** | Train\_Details |
| 1 | John  Doe |
| 2 | Jane Smith |

Table Second Normal Form to Third Normal Form (2NF to 3NF): Reservations Table

Seat Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | Seat\_Details |
| 1 | Seat\_1 |
| 2 | Seat\_2 |
| 3 | Seat\_3 |

Table Second Normal Form to Third Normal Form (2NF to 3NF): Seat Details Table

Operator Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | Operator\_Details |
| 1 | Operator\_1 |
| 2 | Operator\_2 |
| 3 | Operator\_3 |

Table Second Normal Form to Third Normal Form (2NF to 3NF): Operator Details Table

Attendance Details Table:

|  |  |
| --- | --- |
| Reservation\_ID | **Attendant\_Details** |
| 1 | Attendant\_1 |
| 2 | Attendant\_2 |
| 3 | Attendant\_3 |

Table Second Normal Form to Third Normal Form (2NF to 3NF): Attendance Details Table

Separate tables for entities and proper relationships between them are maintained, and hence, all non-key attributes will depend on the primary key only. This achieves 3NF.

# Entity Relationship Diagram

A diagram of a relationship diagram

Description automatically generated

Table Entity Relationship Diagram (ERD)

This Entity Relationship Diagram (ERD) above utilizes Crow's Foot Notation to illustrate the relationships among various entities in a train reservation system. The explanation of the diagram is elucidated below:

This Entity-Relationship Diagram (ERD) utilizes Crow's Foot notation to depict the relationships between the various entities in a train reservation system. Each entity represents a table in a relational database, and the relationships between them are depicted by lines and symbols which indicate the nature of these relationships.

At the core of the diagram is the **Reservation** entity, which captures details about each train reservation. This entity includes attributes such as Reservation\_Number (the primary key), Reservation\_Date, Traveling\_Date, Total\_Charges, Customer\_ID (a foreign key), and Train\_Number (another foreign key). The Reservation entity has a one-to-many relationship with both the Customer and Train entities, indicating that one customer can make multiple reservations, and one train can be associated with multiple reservations.

The **Customer** entity holds information about the passengers making the reservations. It includes attributes like Customer\_ID (the primary key), Customer\_Name, and Email. Each customer can have multiple reservations, as reflected by the one-to-many relationship with the Reservation entity.

The **Train** entity represents individual trains in the system. Its attributes include Train\_Number (the primary key), Departure\_Date, Departure\_Time, Departure\_Station, and Arrival\_Station. A train can have multiple reservations linked to it, showing a one-to-many relationship with the Reservation entity. Additionally, the Train entity connects to other entities, including Seat, TrainOperatorBridge, and TrainAttendantBridge, indicating the various components and personnel associated with each train.

The **Seat** entity records details about the seats available on trains. It includes attributes such as Seat\_Number (the primary key), Seat\_Class, and Train\_Number (a foreign key). Each seat is associated with a specific train, shown by the many-to-one relationship with the Train entity, indicating that multiple seats can belong to one train.

Personnel involved in train operations are represented by the **Operator** and **Attendant** entities. The Operator entity, which includes Operator\_ID (the primary key), Operator\_Name, Role, Lead\_Hours, and Salary, is connected to the Train entity through the TrainOperatorBridge entity. This bridge entity uses a composite key consisting of TrainOperator\_ID (the primary key), Operator\_ID (a foreign key), and Train\_Number (a foreign key), indicating that multiple operators can be associated with multiple trains and vice versa.

Similarly, the **Attendant** entity, which captures details about the train attendants (including Attendant\_ID as the primary key, Attendant\_Name, Phone\_Number, Address, and Salary), is linked to the Train entity via the TrainAttendantBridge entity. The TrainAttendantBridge entity also uses a composite key (TrainAttendant\_ID as the primary key, Attendant\_ID as a foreign key, and Train\_Number as a foreign key), showing that multiple attendants can work on multiple trains and vice versa.

In summary, this ERD provides a comprehensive view of the relationships between various entities in a train reservation system, including customers, reservations, trains, seats, operators, and attendants. The use of Crow's Foot notation clearly illustrates the cardinality and nature of these relationships, helping to understand the database structure and its underlying logic.

# References:

Geeksforgeeks. (2019, November 20). Advantages of Database Management System –

GeeksforGeeks. GeeksforGeeks.

<https://www.geeksforgeeks.org/advantages-of-database-management-system/>

GeeksforGeeks. (2022, April 26). Difference between Data Redundancy and Data Inconsistency.

GeeksforGeeks.

<https://www.geeksforgeeks.org/difference-between-data-redundancy-and-data-inconsistency/>

GeeksforGeeks. (2023, August 28). What is Data Inconsistency in DBMS? GeeksforGeeks.

<https://www.geeksforgeeks.org/what-is-data-inconsistency-in-dbms/>

‌Naeem, T. (2024, March 25). Data integrity in a database - Why is it important | Astera. Astera.

[https://www.astera.com/type/blog/data-integrity-in-a-database/](https://www.astera.com/type/blog/data-integrity-in-a-database/ )

Singh, C. (2015, April 18). Advantages of DBMS over file system. Beginnersbook.com.

<https://beginnersbook.com/2015/04/dbms-vs-file-system/>