NDIIT

NEW DELHI INSTITUTE FOR INFORMATION TECHNOLOGY AND MANAGEMENT

ASSIGNMENT: DATABASE MANAGEMENT SYSTEM(DBMS)

SUBMITTED BY:TEJAS KHURANA

YEAR: BCA 2ND YEAR(2019-2020)

Q1 .What are advantages of DBMS over traditional file-based systems?

Advantages of DBMS over File system –

- Data redundancy and inconsistency – Redundancy is the concept of repetition of data i.e. each data may have more than a single copy. The file system cannot control redundancy of data as each user defines and maintains the needed files for a specific application to run. There may be a possibility that two users are maintaining same files data for different applications. Hence changes made by one user does not reflect in files used by second users, which leads to inconsistency of data. Whereas DBMS controls redundancy by maintaining a single repository of data that is defined once and is accessed by many users. As there is no or less redundancy, data remains consistent.

- Data sharing –
  File system does not allow sharing of data or sharing is too complex. Whereas in DBMS, data can be shared easily due to centralized system.

- Data concurrency –
  Concurrent access to data means more than one user is accessing the same data at the same time. Anomalies occur when changes made by one user gets lost because of changes made by another user. File system does not provide any procedure to stop anomalies. Whereas DBMS provides a locking system to stop anomalies to occur.

- Data searching – For every search operation performed on file system, a different application program has to be written. While

DBMS provides inbuilt searching operations. User only have to write a small query to retrieve data from database.

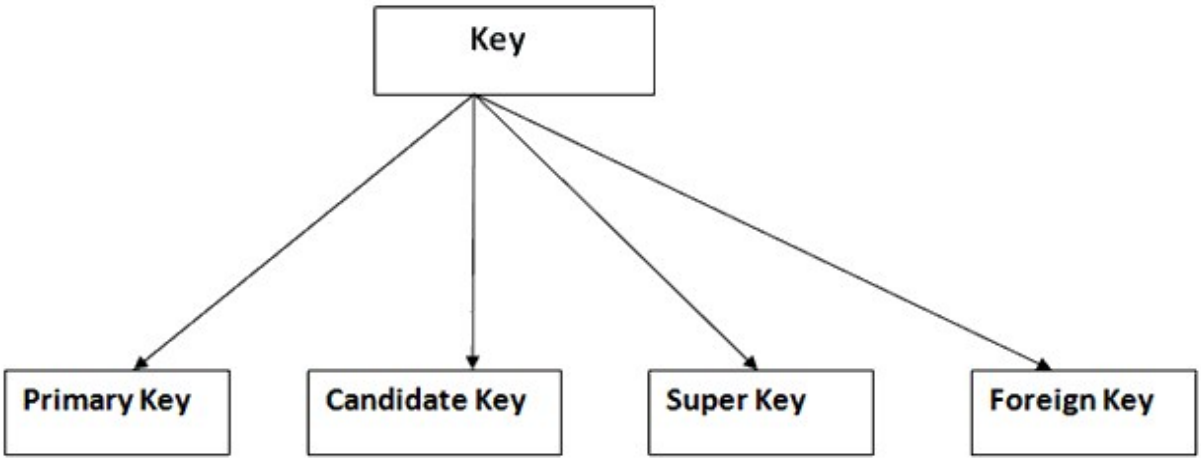
- Data integrity –
  There may be cases when some constraints need to be applied on the data before inserting it in database. The file system does not provide any procedure to check these constraints automatically. Whereas DBMS maintains data integrity by enforcing user defined constraints on data by itself.

- System crashing –
  In some cases, systems might have crashes due to various reasons. It is a bane in case of file systems because once the system crashes, there will be no recovery of the data that's been lost. A DBMS will have the recovery manager which retrieves the data making it another advantage over file systems.

- Data security –
  A file system provides a password mechanism to protect the database but how longer can the password be protected? No one can guarantee that. This doesn't happen in the case of DBMS. DBMS has specialized features that help provide shielding to its data.

Q2 What are super, primary, candidate and foreign keys?

Answer - Types of keys



Primary key

- It is the first key which is used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys as we saw in PERSON table. The key which is most suitable from those lists become a primary key.
- In the EMPLOYEE table, ID can be primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License Number and Passport Number as primary key since they are also unique.
- For each entity, selection of the primary key is based on requirement and developers.

Candidate key

- A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.

- The remaining attributes except for primary key are considered as a candidate key. The candidate keys are as strong as the primary key.

Super Key

Super key is a set of an attribute which can uniquely identify a tuple. Super key is a superset of a candidate key.

For example: In the above EMPLOYEE table, for (EMPLOEE_ID, EMPLOYEE_NAME) the name of two employees can be the same, but their EMPLYEE_ID can't be the same. Hence, this combination can also be a key.

The super key would be EMPLOYEE-ID, (EMPLOYEE_ID, EMPLOYEENAME), etc.

Foreign key

- Foreign keys are the column of the table which is used to point to the primary key of another table.
- In a company, every employee works in a specific department, and employee and department are two different entities. So, we can't store the information of the department in the employee table. That's why we link these two tables through the primary key of one table.
- We add the primary key of the DEPARTMENT table, Department_Id as a new attribute in the EMPLOYEE table.

Now in the EMPLOYEE table, Department_Id is the foreign key, and both the tables are related.

****

Q3. What is the difference between primary key and unique constraints?

Answer -

Key Differences Between Primary key and Unique key:

- Primary key will not accept NULL values whereas Unique key can accept one NULL value.
- A table can have only primary key whereas there can be multiple unique keys on a table.
- A Clustered index automatically created when a primary key is defined whereas Unique key generates the non-clustered index.

PARAMENTER	PRIMARY KEY	UNIQUE KEY
Basic	Used to serve as a unique identifier for each row in a table.	Uniquely determines a row which isn't primary key.
NULL value acceptance	Cannot accept NULL values.	Can accept one NULL value.
Number of keys that can be defined in the table	Only one primary key	More than one unique key
Index	Creates clustered index	Creates nonclustered index

Q4. What is database normalization?

ANSWER-

NORMALIZATION is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships. The purpose of Normalization in SQL is to eliminate redundant (repetitive) data and ensure data is stored logically.

The inventor of the relational model Edgar Codd proposed the theory of normalization with the introduction of the First Normal Form, and he continued to extend theory with Second and Third Normal Form. Later he joined Raymond F. Boyce to develop the theory of Boyce-Codd Normal Form.

Database Normal Forms

Here is a list of Normal Forms

- 1NF (First Normal Form)
- 2NF (Second Normal Form)
- 3NF (Third Normal Form) BCNF (Boyce-Codd Normal Form)
- 4NF (Fourth Normal Form)
- 5NF (Fifth Normal Form)
- 6NF (Sixth Normal Form)

Q5 What are the Difference Between DDL, DML and DCL Commands?

SQL statements are divided into two major categories: data definition language (DDL) and data manipulation language (DML).

Data Definition Language (DDL) statements are used to define the database structure or schema. Some examples:

- CREATE - to create objects in the database
- ALTER - alters the structure of the database
- DROP - delete objects from the database
- TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed
- COMMENT - add comments to the data dictionary
- RENAME - rename an object

Data Manipulation Language (DML) statements are used for managing data within schema objects. Some examples:

- SELECT - retrieve data from a database
- INSERT - insert data into a table
- UPDATE - updates existing data within a table
- DELETE - deletes all records from a table, the space for the records remain
- MERGE - UPSERT operation (insert or update)
- CALL - call a PL/SQL or Java subprogram
- EXPLAIN PLAN - explain access path to data
- LOCK TABLE - control concurrency

Data Control Language (DCL) statements. Some examples:

- GRANT - gives user's access privileges to database • REVOKE - withdraw access privileges given with the GRANT command

Transaction Control (TCL) statements are used to manage the changes made by DML statements. It allows statements to be grouped together into logical transactions.

- **COMMIT -** save work done
- **SAVEPOIN**T - identify a point in a transaction to which you can later roll back
- **ROLLBACK** - restore database to original since the last COMMIT
- **SET TRANSACTION** - Change transaction options like isolation level and what rollback segment to use

Q 6 What is the difference between having and where clause?

Answer- Difference Between WHERE and HAVING Clause

Comparison Parameters	Where Clause	Having Clause
Implementation	This clause is implemented in row operations.	This clause is implemented in column operations.
Application	This clause is applied to a single row.	This clause is applied to summarized rows or groups.
Data extraction	It fetches the selected	Complete data is
	data from the table based on the condition.	fetched together and separated based on condition later.
Aggregate Functions	These functions don't appear in Where clause.	These functions can appear in Having clause.
Used along	It can be used with SELECT and other statements like Update, Insert, Delete etc.	It cannot be used without a SELECT query.
Acts as	It acts as the pre-filter.	It acts as the postfilter.
Group By	Group by clause comes after Where.	Group by clause comes after Having.

Q7 How to print duplicate rows in a table?

ANSWER - SQL Query to find duplicate records in a table in MySQL

```
+-------+----------+
| name   | phone     |
+-------+----------+
| NAMAN | 90784523 |
| TANU| 90563478|
| KAMNA | 24536414 |
```

```
| NONI    | 89567856 |
| NAMAN| 90784523 |
| TANU | 90563478|
| KAMNA| 24536414|
| NONI    | 89567856|
| NAMAN | 90784523|
| TANU | 90563478 |
| KAMNA | 24536414 |
| NONI  | 89567856 |
| NAMAN | 90563478 |
| TANU | 90784523 |
| KAMNA | 24536414 |
| NONI  | 89567856 |
| AVI |  8965342 |
| AVI  |  6888342 | +-------+----------+
```

Following SELECT query will **only find duplicates records based on the name** which might not be correct if two contacts of same but different numbers are stored, as in the following result set Ruby is shown as duplicate which is incorrect.

MySQL> select name, count(name) from contacts group by name, phone;

```
+-------+-------------+
| name  | count(name) |
+-------+-------------+
| NAMAN|            4 |
| TANU|           4   |
| KAMNA|           4   |
```

```
| NONI   |             4 |
| AVI    |           1   |
| AVI    |           1 |
+-------+-------------+
```

Q8 What is a view in SQL? How to create one?

SQL | Views

Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.

In this article we will learn about creating, deleting and updating Views.
Sample Tables:

Student Details

S_ID	NAME	ADDRESS
1	Harsh	Kolkata
2	Ashish	Durgapur
3	Pratik	Delhi
4	Dhanraj	Bihar
5	Ram	Rajasthan

Student Marks

ID	NAME	MARKS	AGE
1	Harsh	90	19
2	Suresh	50	20
3	Pratik	80	19
4	Dhanraj	95	21
5	Ram	85	18

We can create View using **CREATE VIEW** statement. A View can be created from a single table or multiple tables.

Syntax:

CREATE VIEW view_name AS SELECT column1, column2.....FROM table_nameWHERE condition;view_name: Name for the Viewtable_name: Name of the tablecondition:
Condition to select rows

Examples:

Creating View from a single table:

- In this example we will create a View named DetailsView from the table StudentDetails

Query:CREATE VIEW DetailsView ASSELECT NAME, ADDRESSFROM StudentDetailsWHERE S_ID < 5;

- To see the data in the View, we can query the view in the same manner as we query a table.
- SELECT * FROM DetailsView; • **Output:**

NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar

Q9 What is a transaction? What are ACID properties?

ANSWER -
DBMS – Transaction

A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.

Let's take an example of a simple transaction. Suppose a bank employee transfers Rs 500 from A's account to B's account. This very simple and small transaction involves several lowlevel tasks.

A's Account

```
Open_Account(A)
Old_Balance = A.balance
New_Balance = Old_Balance – 500
A.balance = New_Balance
Close_Account(A)
```

B's Account

```
Open_Account(B)
Old_Balance = B.balance
```

```
New_Balance = Old_Balance + 500
B.balance = New_Balance
Close_Account(B)
```

ACID Properties

A transaction is a very small unit of a program and it may contain several low-level tasks. A transaction in a database system must maintain **A**tomicity, **C**onsistency, **I**solation, and **D**urability – commonly known as ACID properties – in order to ensure accuracy, completeness, and data integrity.

- **Atomicity** — This property states that a transaction must be treated as an atomic unit, that is, either all of its operations are executed or none. There must be no state in a database where a transaction is left partially completed. States should be defined either before the execution of the transaction or after the execution/abortion/failure of the transaction.

- **Consistency** — The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on the data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.

- **Durability** — The database should be durable enough to hold all its latest updates even if the system fails or restarts. If a transaction updates a chunk of data in a database and commits, then the database will hold the modified data. If a transaction commits but the system fails before the data could be written on to the disk, then that data will be updated once the system springs back into action.

- **Isolation** — In a database system where more than one transaction is being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is the only transaction in the system. No transaction will affect the existence of any other transaction.

Q10 What are clustered and nonclustered Indexes?

Clustered Index - Table is created with primary key constraints then database engine automatically creates clustered index. In this data sort or store in the table or view based on their key and values.

- **Non-Clustered Index** - Table is created with UNIQUE constraints then database engine automatically creates non-clustered index. A non-clustered index contains the non-clustered index key values and each key value entry has a pointer to the data row that contains the key value

Sr. No.	Key	Clustered Index	Non-Clustered Index
1	Basic	Its created on primary key	It can be created on any key
2	Ordering	Store data physically according to the order	It doesn't impact the order

3	Number of indexes	Only one clustered index can be there in a table	There can be any number of non-clustered indexes in a table
4	Space	No extra space is required to store logical structure	Extra space is required to store logical structure
5	Performance	Data retrieval	Data update is
		is faster than non-cluster index	faster than clustered index

CASE STUDY :

Aim: XYZ hospital is a multi-specialty hospital that includes a number of departments, rooms, doctors, nurses, compounders, and other staff working in the hospital. Patients having different kinds of ailments come to the hospital and get checkup done from the concerned doctors. If required they are admitted in the hospital and discharged after treatment.

The aim of this case study is to design and develop a database for the hospital to maintain the records of various departments, rooms, and doctors in the hospital. It also maintains records of the regular patients, patients admitted in the hospital, the

check up of patients done by the doctors, the patients that have been operated, and patients discharged from the hospital.

SOLUTION -

Description: In hospital, there are many departments like Orthopedic, Pathology, Emergency,
Dental, Gynecology, Anesthetics, I.C.U., Blood Bank,
Operation Theater, Laboratory, M.R.I., Neurology, Cardiology, Cancer Department, Corpse, etc. There is an OPD where patients come and get a card (that is, entry card of the patient) for check up from the concerned doctor. After making entry in the card, they go to the concerned doctor's room and the doctor checks up their ailments. According to the ailments, the doctor either prescribes medicine or admits the patient in the concerned department. The patient may choose either private or general room according to his/her need. But before getting admission in the hospital, the patient has to fulfill certain formalities of the hospital like room charges, etc. After the treatment is completed, the doctor discharges the patient. Before discharging from the hospital, the patient again has to complete certain formalities of the hospital like balance charges, test charges, operation charges (if any), blood charges, doctors' charges, etc.

Next we talk about the doctors of the hospital. There are two types of the doctors in the
hospital, namely, regular doctors and call on doctors.

Regular doctors are those doctors who come to the hospital

daily. Calls on doctors are those doctors who are called by

the hospital if the concerned doctor is not available.

Table Description:

Following are the tables along with constraints used in Hospital Management database.

1. DEPARTMENT: This table consists of details about the various departments in the

hospital. The information stored in this table includes department name, department location, and facilities available in that department.

Constraint: Department name will be unique for each department.

2. ALL_DOCTORS: This table stores information about all the doctors working for the

hospital and the departments they are associated with. Each doctor is given an identity number starting with DR or DC prefixes only.

Constraint: Identity number is unique for each doctor and the corresponding department should exist in DEPARTMENT table.

3. DOC_REG: This table stores details of regular doctors working in the hospital. Doctors

are referred to by their doctor number. This table also stores personal details of doctors

like name, qualification, address, phone number, salary, date of joining, etc.

Constraint: Doctor's number entered should contain DR only as a prefix and must exist in ALL_DOCTORS table.

4. DOC_ON_CALL: This table stores details of doctors called by hospital when additional

doctors are required. Doctors are referred to by their doctor number. Other personal

details like name, qualification, fees per call, payment due, address, phone number, etc., are also stored.

Constraint: Doctor's number entered should contain DC only as a prefix and must exist in ALL_DOCTORS table.

5. PAT_ENTRY: The record in this table is created when any patient arrives in the hospital

for a check up. When patient arrives, a patient number is generated which acts as a

primary key. Other details like name, age, sex, address, city, phone number, entry date,

name of the doctor referred to, diagnosis, and department name are also stored. After

storing the necessary details patient is sent to the doctor for check up.

Constraint: Patient number should begin with prefix PT. Sex should be M or F only.

Doctor's name and department referred must exist.

6. PAT_CHKUP: This table stores the details about the patients who get treatment from

the doctor referred to. Details like patient number from patient entry table, doctor number, date of check up, diagnosis, and treatment are stored. One more field status is

used to indicate whether patient is admitted, referred for operation or is a regular

patient to the hospital. If patient is admitted, further details are stored in PAT_ADMIT

table. If patient is referred for operation, the further details are stored in PAT_OPR table

and if patient is a regular patient to the hospital, the further details are stored in PAT_REG table.

Constraint: Patient number should exist in PAT_ENTRY table and it should be unique.

7. PAT_ADMIT: When patient is admitted, his/her related details are stored in this table.

Information stored includes patient number, advance payment, mode of payment, room

number, department, date of admission, initial condition, diagnosis, treatment, number

of the doctor under whom treatment is done, attendant name, etc.

Constraint: Patient number should exist in PAT_ENTRY table. Department, doctor number, room number must be valid.

8. PAT_DIS: An entry is made in this table whenever a patient gets discharged from the

hospital. Each entry includes details like patient number, treatment given, treatment advice, payment made, mode of payment, date of discharge, etc.

Constraint: Patient number should exist in PAT_ENTRY table.

9. PAT_REG: Details of regular patients are stored in this table. Information stored includes

date of visit, diagnosis, treatment, medicine recommended, status of treatment, etc.

Constraint: Patient number should exist in patient entry table. There can be multiple

entries of one patient as patient might be visiting hospital repeatedly for check up and there will be entry for patient's each visit.

10. PAT_OPR: If patient is operated in the hospital, his/her details are stored in this table.

Information stored includes patient number, date of admission, date of operation,

number of the doctor who conducted the operation, number of the operation theater in

which operation was carried out, type of operation, patient's condition before and after operation, treatment advice, etc.

Constraint: Patient number should exist in PAT_ENTRY table. Department, doctor number should exist or should be valid.

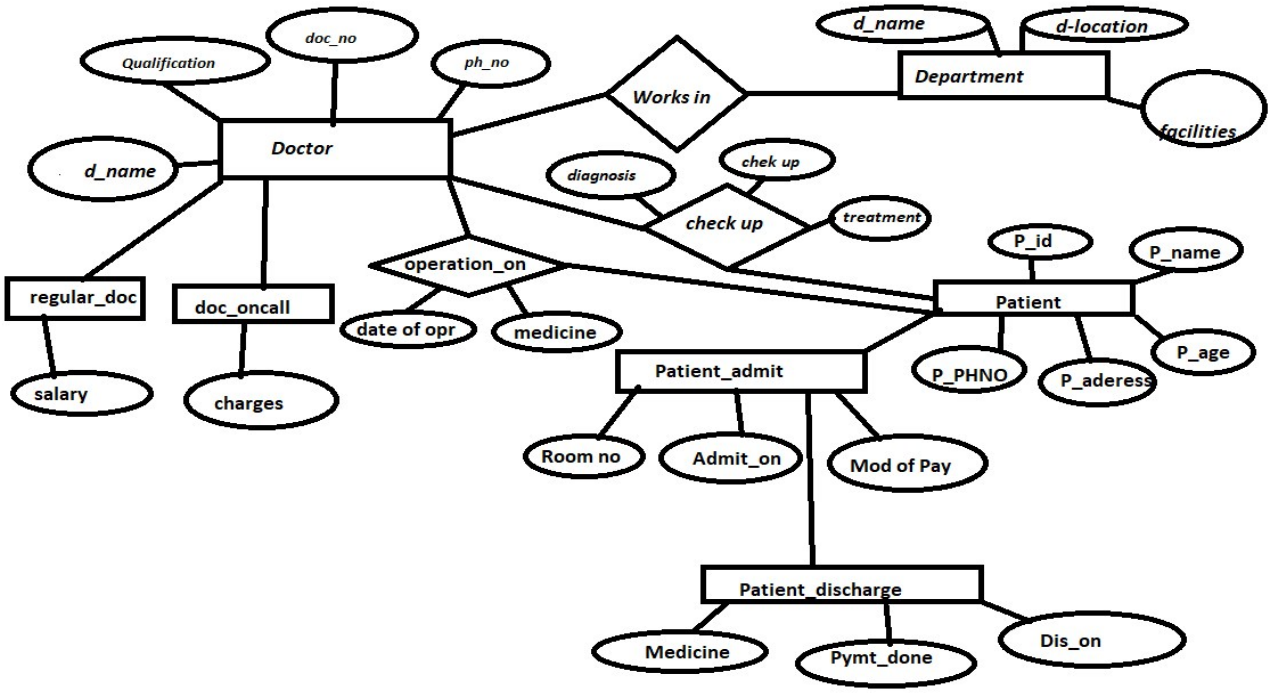
11. ROOM_DETAILS: It contains details of all rooms in the hospital. The details stored in this

table include room number, room type (general or private), status (whether occupied or

not), if occupied, then patient number, patient name, charges per day, etc.

Constraint: Room number should be unique. Room type can only be G or P and status can only be Y or N

E-R Diagram



Relational Database Schema for Case Study

The relational database schema for Hospital Management database is as follows:

- DEPARTMENT (D_NAME, D_LOCATION, FACILITIES)
- ALL_DOCTORS (DOC_NO, DEPARTMENT)
- DOC_REG(DOC_NO, D_NAME, QUALIFICATION, SALARY, ADDRESS,

PH_NO,)

- DOC_ON_CALL (DOC_NO, D_NAME, QUALIFICATION, Charges, ADDRESS,

PH_NO)

- PAT_ENTRY (PAT_NO, PAT_NAME, CHKUP_DT, PT_AGE, SEX, RFRG_CSTNT, DIAGNOSIS,

RFD, ADDRESS, CITY, PH_NO, DEPARTMENT)

- PAT_CHKUP (PAT_NO, DOC_NO, DIAGNOSIS, STATUS, TREATMENT)

- PAT_ADMIT (PAT_NO, ADV_PYMT, MODE_PYMT, ROOM_NO, DEPTNAME, ADMTD_ON, COND_ON, INVSTGTN_DN, TRMT_SDT, ATTDNT_NM)
- PAT_DIS (PAT_NO, TR_ADVS, TR_GVN, MEDICINES, PYMT_GV, DIS_ON)
- PAT_OPR (PAT_NO, DATE_OPR, TY_OPERATION, MEDICINES, DOC_NO, OPTH_NO, OTHER_SUG)
- ROOM_DETAILS (ROOM_NO, TYPE, STATUS, RM_DL_CRG, OTHER_CRG)

Case 2 -

The railway reservation system facilitates the passengers to enquire about the trains available on the basis of source and destination, booking and cancellation of tickets, enquire about the status of the booked ticket, etc. The aim of case study is to design and develop a database maintaining the records of different trains, train status, and passengers. The record of train includes its number, name, source, destination, and days on which it is available, whereas record of train status includes dates for which tickets can be booked, total number of seats available, and number of seats already booked. The database has been developed and tested on the Oracle.

Sloution -

Description:

Passengers can book their tickets for the train in which seats are available. For this, passenger has to provide the desired train number and the date for which ticket is to be booked. Before booking a ticket for a passenger, the validity of 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. After all the available tickets are booked, certain numbers of tickets are booked with waiting status. If waiting lot is also finished, then tickets are not booked and a message of non-availability of seats is displayed.

The ticket once booked can be cancelled at any time. For this, the passenger has to provide the ticket ID (the unique key). The ticket ID is searched and the corresponding record is deleted.

With this, the first ticket with waiting status also gets confirmed.

List of Assumption

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. To implement this sample case study, some assumptions have been made, which are as follows:

1. The number of trains has been restricted to 5.
2. The booking is open only for next seven days from the current date.

3. Only two categories of tickets can be booked, namely, AC and General.
4. The total number of tickets that can be booked in each category (AC and General) is 10.
5. The total number of tickets that can be given the status of waiting is 2.
6. The in-between stoppage stations and their bookings are not considered.

Description of Tables and Procedures

Tables and procedures that will be created are as follows:

1. TrainList: This table consists of details about all the available trains. The information stored in this table includes train number, train name, source, destination, fair for AC ticket, fair for general ticket, and weekdays on which train is available.

Constraint: The train number is unique.

2. Train_Status: This table consists of details about the dates on which ticket can be booked for a train and the status of the availability of tickets. The information stored in this table includes train number, train date, total number of AC seats, total number of general seats, number of AC seats booked, and number of general seats booked.

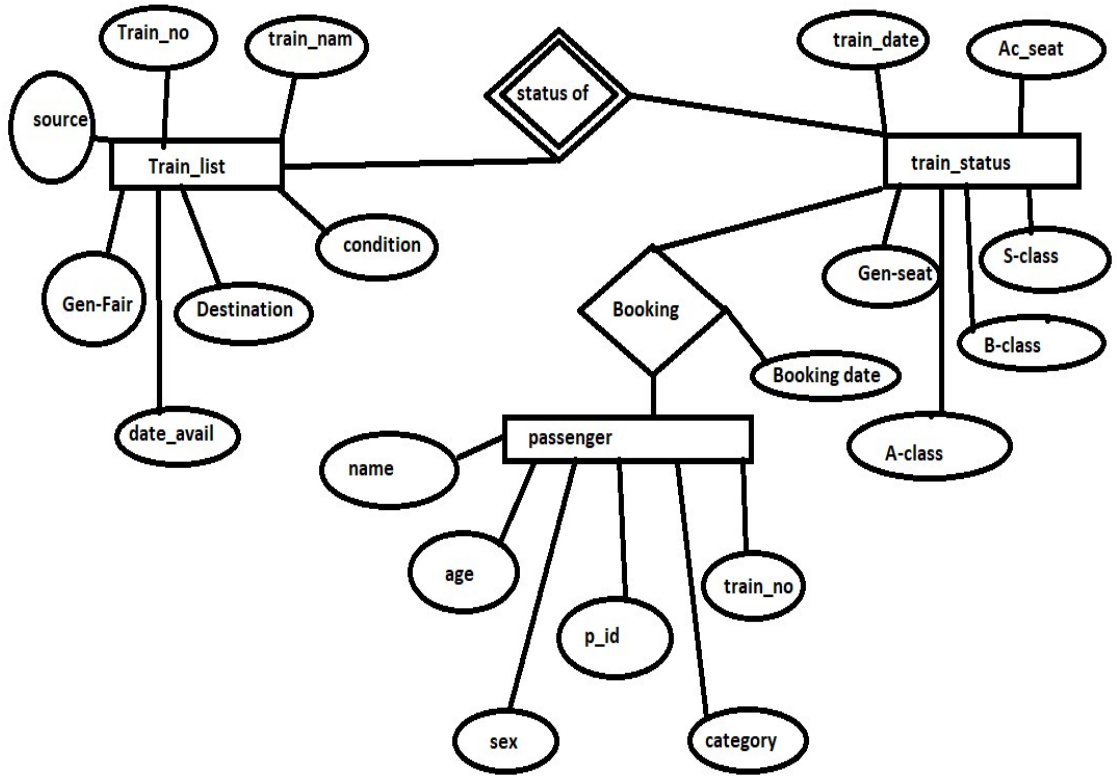
Constraint: Train number should exist in TrainList table.

3. Passenger: This table consists of details about the booked tickets. The information stored in this table includes ticket ID, train number, date for which ticket is booked, name, age, sex and address of the passenger, status of reservation (either confirmed or waiting), and category for which ticket is booked. Constraint: Ticket ID is unique and the train number should exist in TrainList table.

4. Booking: In this procedure, the train number, train date, and category is read from the passenger. On the basis of the values provided by the passenger, corresponding record is retrieved from the Train_Status table. If the desired category is AC, then total number of AC seats and number of booked AC seats are compared in order to find whether ticket can be booked or not. Similarly, it can be checked for the general category. If ticket can be booked, then passenger details are read and stored in the Passenger table.

5. Cancel: In this procedure, ticket ID is read from the passenger and corresponding record is searched in the Passenger table. If the record exists, it is deleted from the table. After deleting the record (if it is confirmed), first record with waiting status for the same train and same category are searched from the Passenger table and its status is changed to confirm.

ER-DIAGRAM



Case 3

A local businesswoman has decided to start her own Internet business, called Masterpieces Ltd, hiring paintings to private individuals and commercial companies.

Because of your reputation as a database designer she has called upon your services to design and implement a database to support her new business.

At the initial planning meeting, to discuss the design, the following user requirements were requested.

The system must be able to manage the details of customers, paintings and those paintings currently on hire to customers. Customers are categorized as B (bronze), S (silver), G (gold) or P (platinum). These categories entitle a customer to a discount of 0%, 5%, 10% or 15% respectively.

Customers often request paintings by a particular artist or theme (eg animal, landscape, seascape, naval, still-life, etc). Over time a customer may hire the same painting more than once.

Each painting is allocated a customer monthly rental price defined by the owner. The owner of the painting is then paid 10% of that customer rental price. Any paintings that are not hired within six months are returned to the owner. However, after three months, an owner may resubmit a returned painting.

Each painting can only have one artist associated with it.

Several reports are required from the system. Three main ones are:

1. For each customer, a report showing an overview of all the paintings they have hired or are currently hiring
2. For each artist, a report of all paintings submitted for hire
3. For each artist, a returns report for those paintings not hired over the past six months Remember to identify key attributes and any foreign key attributes.

SOLUTION -

Artist Report

Artist No: _____ Year of Birth: _____ Age: _____

Artist Name: _____ Year of Death: _____ (if applicable)

Country of Birth: _____

Painting No	Painting Title	Theme	Rental Price (monthly)	Owner No No	Owner Name	Owner Tel

Customer Rental Report

Customer No: _____ Customer Category: _____

Customer Name: _____ Category Description: _____

Customer Address: _____ Category Discount: _____

Painting No	Painting Title	Painting Theme	Date of Hire	Date Due Back	Returned (Y/N)

Return To Owner Report

Owner No: _____ Owner Name: _____

Owner Address: _____

_____

_____

Painting No	Painting Title	Return Date
_____	_____	_____
_____	_____	_____
_____	_____	_____
_____	_____	_____
_____	_____	_____