Abdullah Kapadia
22K - 4147
BCS - 5J

Database Assignment#02

Q1.

a)

1. PATIENT
   - PatientID
   - Name
   - ContactNumber
   - DateOfBirth
2. TREATMENT
   - TreatmentID
   - HospitalName
   - StartDate
   - EndDate
3. PROCEDURE
   - ProcedureID
   - ProcedureName
   - ProcedureDate
4. MEDICATION
   - MedicationID
   - MedicationName
   - Dosage
   - Duration

b)

Entity Types:

1. PATIENT
   - PatientID (Primary Key)
   - Name
   - ContactNumber
   - DateOfBirth
2. TREATMENT
   - TreatmentID (Primary Key)
   - HospitalName
   - StartDate
   - EndDate
   - PatientID (Foreign Key referencing PATIENT)
3. PROCEDURE
   - ProcedureID (Primary Key)

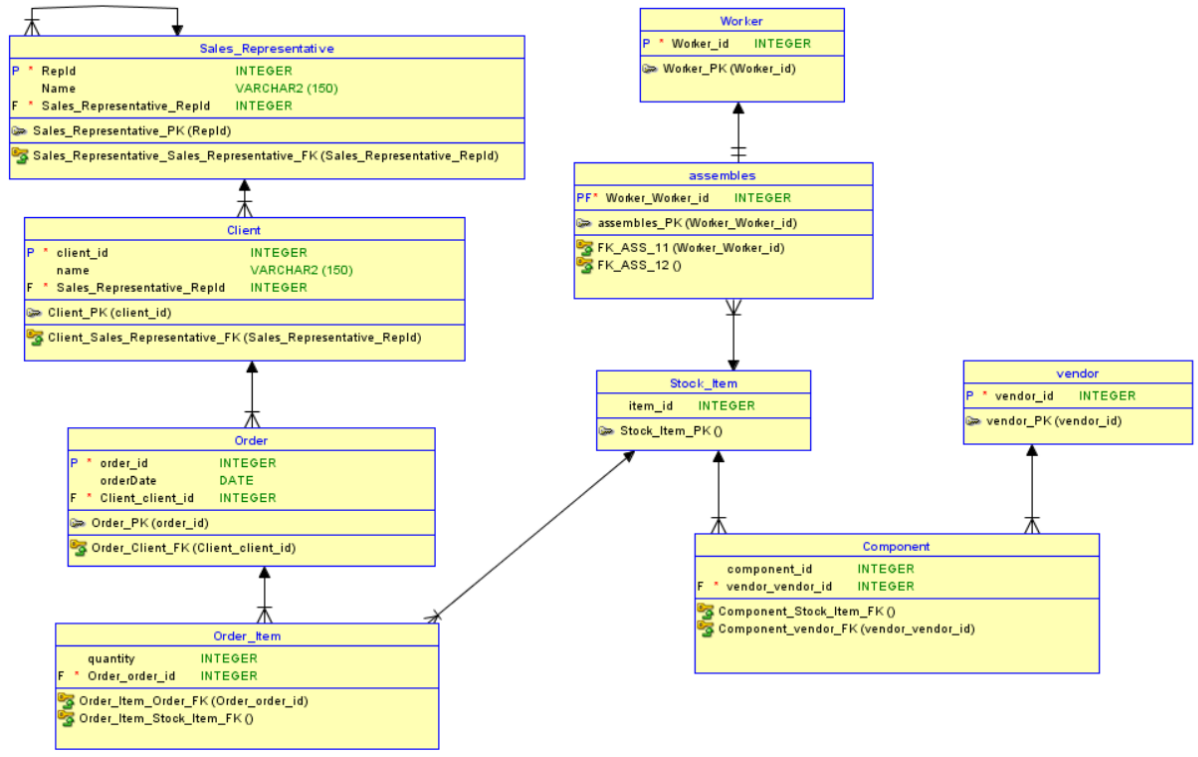
- ProcedureName
- ProcedureDate
- TreatmentID (Foreign Key referencing TREATMENT)

4. MEDICATION
   - MedicationID (Primary Key)
   - MedicationName
   - Dosage
   - Duration
   - TreatmentID (Foreign Key referencing TREATMENT)

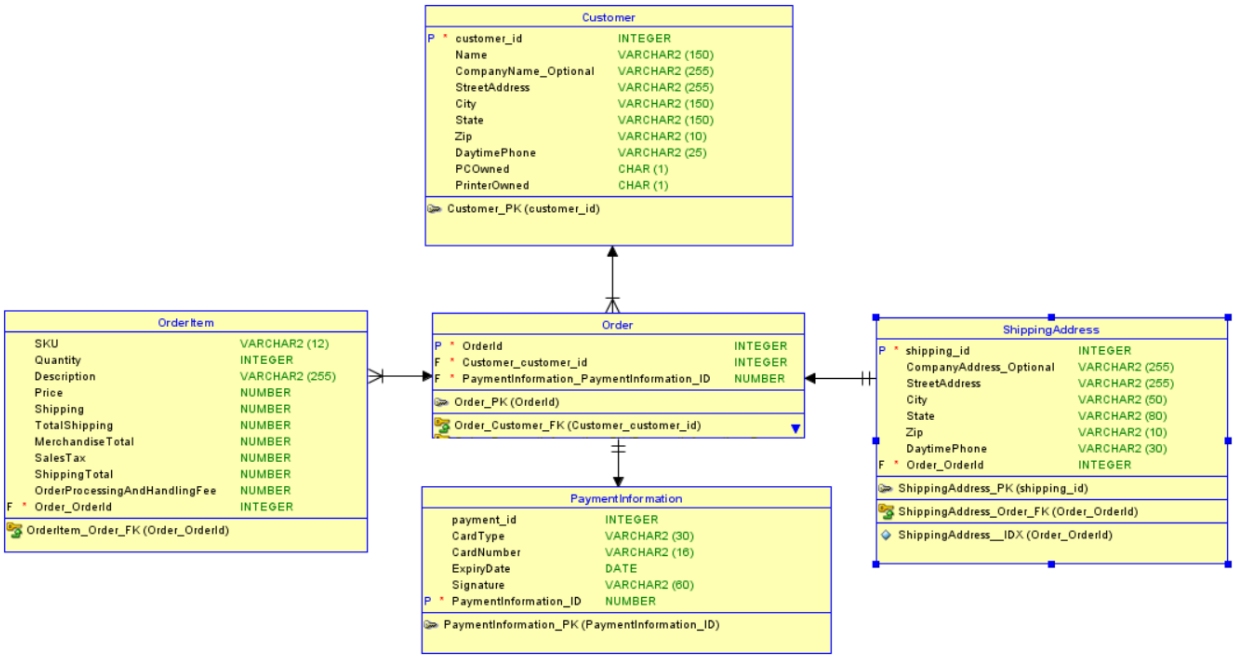
Relationship Types:

1. PATIENT to TREATMENT: One-to-Many
   - One patient can have multiple treatments
   - Each treatment is associated with one patient
2. TREATMENT to PROCEDURE: One-to-Many
   - One treatment can involve multiple procedures
   - Each procedure is associated with one treatment
3. TREATMENT to MEDICATION: One-to-Many
   - One treatment can involve multiple medications
   - Each medication prescription is associated with one treatment

Q2)



Q3)



Q4 PART A)

a) In the current design, since all details are stored in a single table, it is not feasible to insert the information of a new supplier without assigning any cars. This is because the table is structured around car entries, and each entry must include details about the car and the associated supplier. Adding a supplier without a car would result in rows with many null or incomplete values, which is not ideal.

   Issue**:** The primary issue here is that the table does not support the independence of supplier data from car data. This leads to the inability to record suppliers who do not yet provide cars, causing a limitation in database flexibility and scalability.

b) If the phone number for "Hertz" is updated in some rows but not all, it will lead to inconsistent data across the database. This situation is known as an update anomaly.

   Update anomaly: If we update only some rows but not all, we end up with inconsistent data.

   Implications: This can lead to data inconsistency, where different rows show different phone numbers for the same supplier, making it unclear which is correct.

c) If the last car rented from "Avis" (C104) is deleted from the system, the supplier information for "Avis" would also be lost if there are no other cars in the database from "Avis."

Delete Anomaly: This leads to a delete anomaly where removing a car entry also unintentionally removes valuable supplier information. The potential issues include loss of contact information for the supplier and historical data that might be needed for reports or audits.

Q4 PART B)

1. First Normal Form (1NF): The table is already in 1NF as all attributes contain atomic values.
2. Second Normal Form (2NF): To achieve 2NF, we need to remove partial dependencies. The current primary key is CarID. Separate Supplier details into a new table where SupplierID is the primary key, removing the dependency of supplier information on the car entry.
3. Third Normal Form (3NF): To Remove Transitive Dependencies, Separate out the Category details into another table to ensure that non-key attributes (Category Name) depend only on the primary key (Category ID).

Final Schema Design:

1. Car Table:
    - CarID (Primary Key)
    - Model
    - PricePerDay
    - CategoryID (Foreign Key)
    - SupplierID (Foreign Key)
2. Category Table:
    - CategoryID (Primary Key)
    - CategoryName
3. Supplier Table
    - SupplierID (Primary Key)
    - SupplierName
    - Email
    - Phone

Justification of the Design:

1. Solving anomalies:
    - Insert Anomaly: We can now insert supplier information without needing car data.
    - Update Anomaly: Supplier information is stored in one place, so updates are made once and reflect everywhere.
    - Delete Anomaly: Deleting a car doesn't remove supplier information.
2. Advantages:

- Data Consistency: Information about suppliers and categories is stored in one place, reducing the risk of inconsistencies.
- Data Integrity: Foreign key constraints ensure that cars can't be associated with non-existent suppliers or categories.
- Flexibility: New suppliers can be added without cars, and new categories can be created before assigning cars to them.
- Efficiency: Reduces data redundancy, saving storage space and improving query performance.

Q5)

1. Identifying Functional Dependencies:

   Based on the information provided in the form, we can identify the following functional dependencies:

   - Patient Number: Full Name, Bed Number, Ward Number, Ward Name
   - Drug Number: Name, Description
   - (Patient Number, Drug Number): Dosage, Method of Admin, Units per day, Start Date, Finish Date
2. Normalization Process:
   - First Normal Form (1NF): The data is already in 1NF as all attributes contain atomic values.
   - Second Normal Form (2NF): We need to remove partial dependencies. We can split this into multiple tables:
     - Patient Table: PatientNumber (Primary Key), FullName, BedNumber, WardNumber
     - Ward Table: WardNumber (Primary Key), WardName
     - Drug Table: DrugNumber (Primary Key), DrugName, Description, Dosage
     - Prescription Table: PrescriptionID (Primary Key), PatientNumber (Foreign Key), DrugNumber (Foreign Key), Dosage, MethodOfAdmin, UnitsPerDay, StartDate, FinishDate
   - Third Normal Form (3NF): We need to remove transitive dependencies. The current structure doesn't appear to have any transitive dependencies, so our 2NF relations are already in 3NF.
3. Explanation of Keys:
   - Primary Keys (PK):
     - Patient Table: PatientNumber
     - Ward Table: WardNumber
     - Drug Table: DrugNumber
     - Prescription Table: PrescriptionID (This would be a new auto-generated key)
   - Alternate Keys:
     - There are no clear alternate keys in this schema based on the given information.
   - Foreign Keys (FK):
     - In Patient Table: WardNumber (references Ward Table)

- In Prescription Table:
  - PatientNumber (references Patient Table)
  - DrugNumber (references Drug Table)

Q6)

1. Is this relation in 2NF?
   The relation is not in 2NF due to some partial dependencies.
   Partial dependencies:
   - Room_Number: Room_Type is a partial dependency because Room_Number is only part of the primary key.
   - Service_Code: Service_Charge is a partial dependency because Service_Code is only part of the primary key.

   These dependencies violate the 2NF condition as they depend only on part of the composite key.

   To achieve 2NF, we need to decompose the relation to remove partial dependencies:

   1) **Room**:
      - Attributes: Room_Number, Room_Type
      - Primary Key: Room_Number
   2) **Service**:
      - Attributes: Service_Code, Service_Charge
      - Primary Key: Service_Code
   3) **Reservation**:
      - Attributes: Reservation_ID, Customer_ID, Room_Number, Stay_Date, Service_Code
      - Primary Key: Reservation_ID, Room_Number, Stay_Date
2. Further Normalization to 3NF:
   Since there are no transitive dependencies in any of our 2NF relations, they are already in 3NF. No further normalization is necessary.
   Final Schema:
   1) Room:
      - Room_Number (Primary Key)
      - Room_Type
   2) Service:
      - Service_Code (Primary Key)
      - Service_Charge
   3) Reservation:
      - Reservation_ID (Primary Key)
      - Customer_ID
      - Room_Number (Foreign Key references Room)
      - Stay_Date
      - Service_Code (Foreign Key references Service)