James Frierson, Cecile Darwiche, Riyan Ibadah
CSC 423 Database Systems
Group Project Part 2

Part 2: Develop a logical data model based on the following requirements:

- **Derive relations from the conceptual model.**
  - **Client(**ClientNum{pk}, fName, lName, address, number)
  - **Equipment**(eqId{pk}, description, usage, cost)
  - **Employee(**staffNum{pk}, fName, lName, address, salary, number)
  - **Requirement(**reqId{pk}, startD, startD, duration, comments,ClientNum{fk}
    - Foreign key here to connect to clients because of the many to many relationship between Client and Requirement. References Client(ClientNum)
  - **Hired(**reqId{pk}{fk}, staffNum{pk}{fk})
    - Hired comes about because of *:* relationship between staff and requirement
      - reqId **references** Requirement (reqId)
      - staffNum **references** Employee(staffNum)
  - **Usage**(reqId{pk}{fk},eqId{pk}{fk})
    - Usage comes about because of the *:* relationship between requirements and equipment
      - reqId **references** Requirements(reqId)
      - eqId **references** Equipment(eqId)

- **Validate the logical model using normalization to 3NF.**

Clients:

1NF Achieved, each attribute holds atomic values.
2NF Achieved because no attribute is dependent on a subset of the primary key
3 NF Achieved because no transitive decencies exist

Equipment:

1NF Achieved, each attribute holds atomic values.
2NF Achieved because no attribute is dependent on a subset of the primary key
3 NF Achieved because no transitive decencies exist

Employee:

1NF Achieved, each attribute holds atomic values.
2NF Achieved because no attribute is dependent on a subset of the primary key
3 NF Achieved because no transitive decencies exist

Requirement:

1NF Achieved, each attribute holds atomic values.
2NF Achieved because no attribute is dependent on a subset of the primary key
3 NF Achieved because no transitive dependencies exist

Usage:

1NF: Composed of foreign keys only, each with atomic values.
2NF: As it contains only foreign keys, it is in 2NF by default

3NF: No additional non-key attributes, so it's in 3NF.

Hired:

1NF: Atomic values in both columns.
2NF: Only foreign keys, thus already in 2NF.
3NF: Directly in 3NF as there are no non-key attributes.

- **Validate the logical model against user transactions.**

1. Query: How much equipment was used in a specific service?
   - Path:
     - Start with the Requirement table to identify the specific service using reqID.
     - Use Requirement_Equipment join table to find all eqID associated with that reqID.
     - Query the Equipment table to count the number of equipment items used.
   - Relationships: The join table links Requirements to Equipment, allowing us to track equipment usage for specific services.

2. Query: Who were the employees working on a job for a particular client?
   - Path:
     - Begin with the Clients table to identify the client using clientNum.
     - Move to the Requirement table to find all reqIDs associated with that client.
     - Use the Employee_Requirement join table to find all staffNum for those reqIDs.
     - Query the Employee table to get details of these employees.
   - Relationships: This path shows the link between Clients, their Requirements, and the Employees assigned to those Requirements.

3. Query: What is the total cost of equipment used in all services this month?
   - Path:
     - Start with the Requirement table to find all services (reqID) within the current month.
     - Use Requirement_Equipment to find all associated eqIDs.
     - Sum the cost from the Equipment table for these eqIDs.
   - Relationships: By connecting Requirements to Equipment, we can calculate the total equipment cost.

4. Query: Find the average duration of services provided to a specific client.
   - Path:
     - Begin with the Clients table to identify the client using clientNum.

    - Access the Requirement table to find all requirements associated with this client.
    - Calculate the average of the duration field from these requirements.
- Relationships: This shows how client information is linked to the services they have received and the duration of these services.

5. Query: How many clients received services involving a specific piece of equipment last year?

- Path:
    - Start from the Equipment table to identify the equipment using eqID.
    - Use Requirement_Equipment to find all reqIDs associated with this eqID.
    - From the Requirement table, filter these reqIDs for those within last year.
    - Count the unique clientNums associated with these requirements.
- Relationships: This connects Equipment to Requirements and then to Clients, focusing on a e.

- <u>**Define integrity constraints:**</u>
    - Primary key constraints.
        - Clients: clientNum - UNIQUE , NOT NULL
        - Equipment: eqID - UNIQUE , NOT NULL
        - Employee: staffNum - UNIQUE, NOT NULL
        - Requirement: reqID - UNIQUE , NOT NULL
        - Hired: Composite key of reqId and staffNum
        - Usage: Composite key of reqId and eqId

    - Referential integrity/Foreign key constraints.

      Hired(<u>reqId{pk}{fk},</u> <u>staffNum{pk}{fk}</u>)
        - reqId **references** Requirement (reqId)
        - staffNum **references** Employee(staffNum)

      Usage(reqId{pk}{fk},eqId{pk}{fk})
        - reqId **references** Requirements(reqId)
        - eqId **references** Equipment(eqId)

      Requirement(ClientNum{fk})
        - ClientNum **references** client(ClientNum)

    - Alternate key constraints (if any).
        - Clients: number (assuming that each client has a unique phone number)

        - Employee: number (assuming that each employee has a unique phone number)

- Required data.

    Clients
    - Address - NOT NULL need place to clean
    - number NOT NULL

    Equipment:
    - eqID,
    - Description,
    - Usage,
    - cost NOT NULL need to charge) NOT NULL

    Employee :
    - fName, NOT NULL need when hired
    - lName NOT NULL need when hired
    - Salary NOT NULL need to pay an amount
    - Address NOT NULL need when hired

    Requirement (reqID,
    - startD NOT NULL need to know when to start,
    - startT NOT NULL need to know what time to start, =

- Attribute domain constraints

    Clients
    - clientNum: Integer (Primary Key)
    - fName: String (Text)
    - lName: String (Text)
    - address: String (Text)
    - number: Integer (10-digit, assuming US phone number format)

    Equipment

    - eqID: Integer (Primary Key)
    - description: String (Text)
    - usage: String (Text)
    - cost: Numeric (Decimal/Float, must be positive)

    Employee

    - staffNum: Integer (Primary Key)
    - fName: String (Text)
    - lName: String (Text)
    - address: String (Text)
    - salary: Numeric (Decimal/Float, must be positive)
    - number: Integer (10-digit, assuming US phone number format)

Requirement

- reqID: Integer (Primary Key)
- startD: Date
- startT: Time
- duration: Numeric (Integer, non-negative)
- comments: String (Text)

Hired

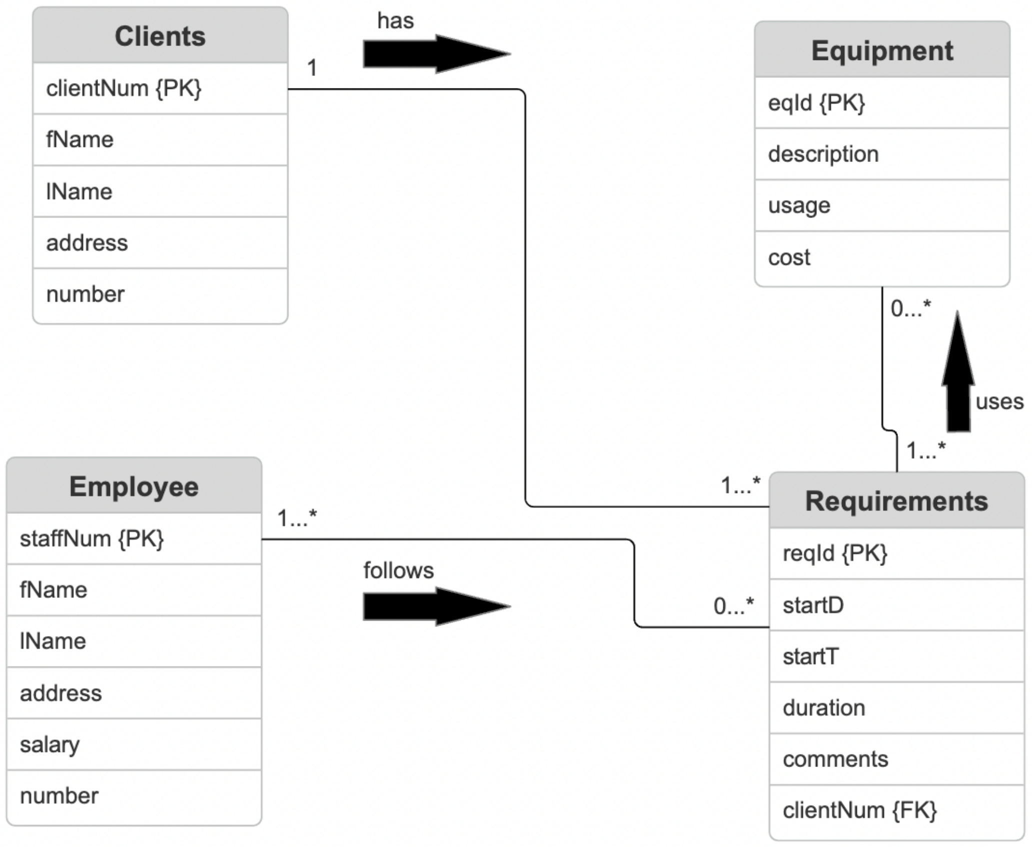
- reqId: Integer (Composite Primary Key, Foreign Key)
- staffNum: Integer (Composite Primary Key, Foreign Key)

Usage

- reqId: Integer (Composite Primary Key, Foreign Key)
- eqId: Integer (Composite Primary Key, Foreign Key)

- General constraints (if any).
    - startD and startT should be valid dates and times.
    - The duration should not be negative,
    - Clients and Employee:
    - Address fields should be strings
    - Names (fName, lName) should be strings, with character restrictions (no numerals or special characters).

- **<u>Generate the E-R diagram for the logical level (contains FKs as attributes)</u>**


Clients
clientNum {PK}
fName
lName
address
number
has
1
Equipment
eqId {PK}
description
usage
cost
0...*
uses
1...*
Employee
staffNum {PK}
fName
lName
address
salary
number
1...*
follows
1...*
0...*
Requirements
reqId {PK}
startD
startT
duration
comments
clientNum {FK}