Name of Your Company

Rosal Analytics

Project Title

Home Care Company Database

Team

- Diana Nalivaiko
- Roberto Rojas Perez
- Asal Mojtabaei

We are going to use MySQL

System Choice

MySQL

Weekly Meeting Hours

We will meet and work on the project on Thursdays at 3:30 pm

Project Description

Home Care Company provides health care assistance to seniors and disabled people. Many registered nurses work for this company. Patients (Clients) make a contract with the company. In the contract information about the client, schedule of nurse visit at home, address of the client, phone number, brief description of the illness of the patient… is stored.

We store information about nurses too. Name, phone number, email address, Health Ministry registration number, date of birth …

Nurses must record a report about their visit of the patient. They store in the database if they met the client on the scheduled time, or they missed, the health condition of the client and services they provided to the client.

There is a possibility that a nurse cannot visit a client and the company substitute the nurse with another nurse for that temporary period. We must store this information too.

(This description is a draft description and must be completed gradually as the database is developed to reflect the complete description of the final database. The description should contain information about entities, attributes, relationships, cardinality ratios, and constraints…)

Assumptions about Cardinality and Participations

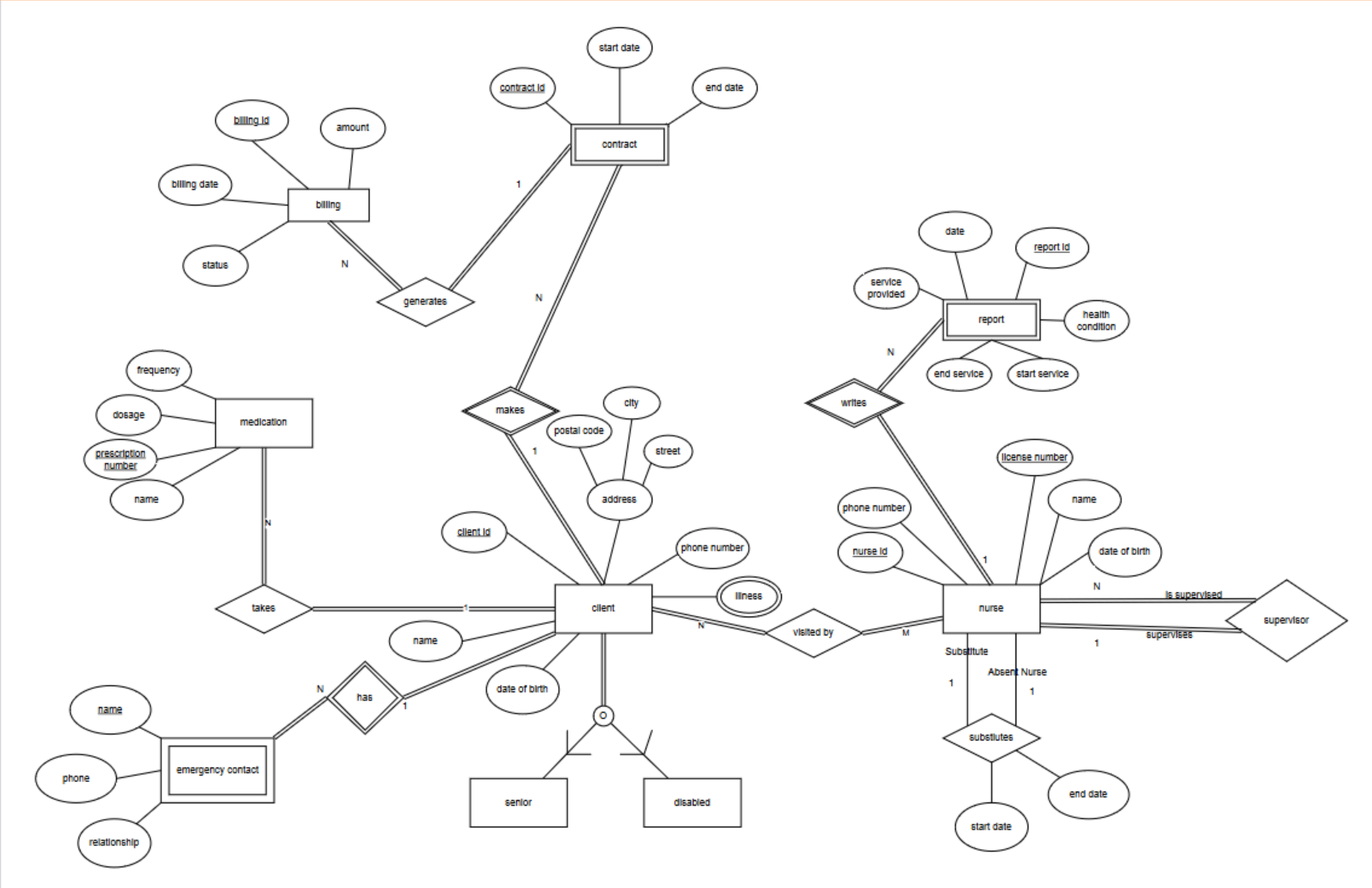
You can write all the assumptions about Cardinality and Participations (total/partial) here.

- One or more nurses can visit multiple clients. N:M , both have a total relationship.
- Zero or one nurse substitutes another nurse, and zero or one nurse is substituted by another nurse. 1:1. a nurse has a partial relationship with the other.

- One nurse must write a report, and one or more reports are written by one nurse. 1:N, Both have a total participation.
- One patient makes one or more contracts with the company, and one or more contracts is made by one patient. N:1 , Both have total participation.
- One client can have multiple emergency contacts. 1:N , both have a total relationship.
- One client can have multiple medications. 1:N , both have total participation.
- One contract can have multiple billing records. 1:N both have total participation.
- One nurse can supervise multiple nurses, but each nurse can have only one supervisor.1:N, Both have total relationship (recursive relationship).
- Seniors and Disabled are sub-classes of Client: all clients are disabled or seniors or both.
- A client can have multiple illnesses.

EER Modeling Diagram

In the following drawing canvas, EER Modeling shapes have been provided. You can copy and replicate them (Ctrl+C to copy and Ctrl+V to paste. You can also select a shape, then press Ctrl button and drag and drop to copy a shape) and edit them to build your diagram.



ER-Model Mapping to Database Relational Schema

Client(clientID, name , dateOfBirth , phone number , postal code , city, street)

Senior (**clientID)**

Disabled (**clientID**)

Medication(prescriptionNumber , name , dosage , frequency, **clientID**)

Nurse(nurseID, LicenseNumber, Name, DOB, Phone)

Client_contract(**ClientID** , ContractID , StartDate , EndDate)

Client_Ilness(**clientID** , illness)

Client_EmergencyContract(**ClientID**, name , phone, relationship)

Billing(billingID , amount , billing date , status , **contractID**)

Client_visitedBy_nurse(**nurseID , LicenseNumber, clientID**)

Substitute_Substitutes_AbsentNurse (**Substitute_nurse_id, AbsentNurse_nurse_id,** start date, end date)

isSupervised_Supervisor_Supervises (**isSupervised_nurse_id, Supervisor_nurse_id**)

Nurse_Report (**nurse id**, **license_number,** report id, date, service provided, end service, start service)

Normalization

All relations must be normalized up to BCNF. You must explain why you believe every relation in your database in normalized.

BCNF requirements:

Each functional dependency has its left side as a superkey.

There are no partial dependencies or transitive dependencies that violate the BCNF conditions.

FDs:

ClientID->Name, DateOfBirth, PhoneNumber, PostalCode, City, Street

ClientID->Senior

ClientID->Disabled

PrescriptionNumber->Name, Dosage, Frequency, ClientID

NurseID , LicenseNumber -> Name, DOB, Phone

ContractID, ClientID->StartDate, EndDate

ClientID, Illness->IllnessDetails

ClientID, NameOfContact->Phone, Relationship

BillingID->Amount, BillingDate, Status, ContractID

NurseID, LicenseNumber, ClientID ->VisitDetails

substitute_nurse_id -> absentNurse_nurse_id, startDate, endDate

isSupervised_nurse_id -> supervisor_nurse_id

ReportID, NurseID, LicenseNumber ->Date, ServiceProvided, EndService, StartService

Tables:

Clients(ClientID,Name,DateOfBirth,PhoneNumber,PostalCode,City,Street) ClientID is a superkey.

ClientSenior(ClientID,Senior) ClientID is a superkey

ClientDisabled(ClientID,Disabled) ClientID is a superkey

Prescriptions(PrescriptionNumber,Name,Dosage,Frequency,ClientID) PrescriptionNumber is a superkey.

Nurses(NurseID,LicenseNumber,Name,DOB,Phone) NurseID,LicenseNumber is a composite superkey.

Contracts(ContractID, ClientID , StartDate,EndDate) ContractID,ClientID is the composite superkey.

ClientIllnesses(ClientID,Illness,IllnessDetails) ClientID,Illness is the composite primary key.

ClientContacts(ClientID,NameOfContact,Phone,Relationship) ClientID,NameOfContact is the composite primary key.

Billing(BillingID,Amount,BillingDate,Status, ContractID) BillingID is a superkey.

VisitDetails(NurseID,LicenseNumber,ClientID,VisitDetails) NurseID,LicenseNumber,ClientID is the composite primary key.

SubstituteNurses(substitute_nurse_id, absentNurse_nurse_id, startDate, endDate) substitute_nurse_id is superkey

SupervisorNurses(isSupervised_nurse_id, supervisor_nurse_id) isSupervised_nurse_id is superkey.

Reports(ReportID,NurseID,LicenseNumber,Date,ServiceProvided,EndService,StartService)
ReportID,NurseID,LicenseNumber is composite primary key.

Determining Data Types (Domain) and Constraints

You explain why you choose a certain data type for a field and why you apply certain constraints

```
CREATE TABLE Client(
   clientID INT AUTO_INCREMENT,
         name VARCHAR(100) NOT NULL,
         dataOfBirth DATE NOT NULL,
         phoneNumber CHAR(10) NOT NULL,
         postalCode CHAR(6) NOT NULL,
         city VARCHAR(20),
         street VARCHAR(100),
         clientType ENUM('Senior', 'Disabled') NOT NULL,
         PRIMARY KEY(clientID)
);
```

- Client ID’s can be an incremental primary key, names can be up to 100 due to usually not being any longer, DOB can be a DATE data type that mandatory, phones can be mandatory 10 chars due to not being any longer, postal codes are required but can just be 6 chars long, city names are always under 20 chars, street names need a bit more space, and client types can be set to seniors or disabled since those are the types we serve

```
CREATE TABLE Illness (
        illnessID INT AUTO_INCREMENT ,
        clientID INT,
        illness VARCHAR(100)NOT NULL,
        PRIMARY KEY(illnessID),
        FOREIGN KEY(clientID) REFERENCES Client(clientID)
);
```

- For IDs we chose integer data type because IDs are whole numbers. ‘illnessID’ has auto increment to allow a unique number to be auto-generated when a new record is inserted into the table. Since all patients of Home Care company have an illness, ‘illness’ is not null and is up to 100 characters long.

```
CREATE TABLE Medication(
   prescriptionNO  INT AUTO_INCREMENT,
        name VARCHAR(30) NOT NULL,
        dosage VARCHAR(10) ,
        frequency VARCHAR(100),
        PRIMARY KEY(prescriptionNO)
);
```

- Since every patient has a name, the value of ‘name’ cannot be null and longer than 30 characters. ‘prescriptionNO’ has auto increment to allow a unique number to be auto-generated when a new record is inserted into the table. The common format of a medication dosage is a combination of several numbers and units of measurement, therefore we chose varchar data type up to 10 characters long. Usually, the frequency of taking a medication is represented as a number of pills per day, therefore we chose to use varchar data type up to 100 characters long. Since not every patient of Home Care requires medication, ‘dosage’ and ‘frequency’ might be null.

```
CREATE TABLE Billing (
        billingID INT AUTO_INCREMENT,
        clientID INT,
        billingDate DATE NOT NULL,
        amount INT NOT NULL,
        status VARCHAR(30) NOT NULL,
        PRIMARY KEY(billingID)
);
```

- Billings cans store their individual transactions with an incremental primary key ID number, as well as the date it was made as DATE format, the amount of money they owe, and the status of the bill.

```
CREATE TABLE Emergency(
   emergencyID INT AUTO_INCREMENT,
         clientID INT,
   name VARCHAR(100) NOT NULL,
         phone CHAR(10) NOT NULL,
         relationship VARCHAR(50) NOT NULL,
         PRIMARY KEY(emergencyID),
         FOREIGN KEY(clientID) REFERENCES Client(clientID)
);
```

- Emergency contacts can store their info with their id's as incremental primary keys, along with the clientID that they belong to, while storing sensitive info like their name, relationship and phone number

```
CREATE TABLE Nurse(
   licenseNO INT,
         nurseID INT UNIQUE,
         name VARCHAR(50) NOT NULL,
         dateOfBirth DATE NOT NULL,
         phoneNO VARCHAR(10) NOT NULL,
         PRIMARY KEY(licenseNO)
);
```

- Licences can be stored as an integer for easier data storage, and their individual ID's can be generated with Unique markers to indicate possibly different class types of nurses, as well as storing their personal info like DOB and phone.

```
CREATE TABLE Report(
   reportID INT AUTO_INCREMENT,
         nurseID INT,
         clientID INT ,
         healthCondition VARCHAR(200) NOT NULL,
         startService DATETIME,
         endService DATETIME,
         serviceProvided VARCHAR(200),
         date DATE,
         PRIMARY KEY(reportID),
         FOREIGN KEY(nurseID) REFERENCES Nurse(nurseID),
         FOREIGN KEY(clientID) REFERENCES Client(clientID)
);
```

- reportIDcan be autoincremental primary key, client and nurse id's can be fetched from their respective tables, and a small 200 word description can be done for their current condition, as well as the dates stored with their time signature, and a small description of the service provided, along with the date the service was performed on

```
CREATE TABLE Contract(
   contractID INT AUTO_INCREMENT,
        clientID INT,
        startDate DATETIME,
        endDate DATETIME,
        PRIMARY KEY(contractID),
        foreign key (clientID) REFERENCES Client(clientID)
);
```

- contractID can be an autoincremental, and clientID can be fetched from their respective table, and start and end dates can be DATETIME so that we know what time of day they made it

```
CREATE TABLE Substitute(
        nurseID INT,
        substituteNurseID INT,
        startDate DATE,
        endDate DATE,
        PRIMARY KEY(nurseID, substituteNurseID),
        FOREIGN KEY(nurseID) REFERENCES Nurse(nurseID),
        FOREIGN KEY(substituteNurseID) REFERENCES Nurse(nurseID)
);
```

- Nurse ID and substituteNurseID will refer to the same table as primary keys, and the start and end dates can be stored as a DATE format.

Creating Database and Tables - SQL DDL

CREATE tables script

Inserting Values in Tables

INSERT tables script

SQL Queries

sql queries.txt

Views

You do not need to copy SQL commands here. Save your SQL commands in a script file and just mention the name of the file here. Make sure the script file is stored beside this document within the same folder.