

## 1. Name of Your Company

Smile Database Inc x

## 2. Project Title

Dental Clinic Appointment Management Database

## 3. Team

- Utsav Monga (*MySQL Expert*)
- Frayan Foroughi (*MS SQL Server Expert*)

## 4. Weekly Meeting Hours

Monday: 10:30am to 12:30pm (Approximately 2 hours) in person

Tuesday: 10:30am to 11:00am (Approximately 30 minutes) in person

Wednesday: 12:30pm to 1:30pm (approximately 1 hour) in person/online

Thursday: 10:30am to 11:00am (Approximately 30 minutes) in person

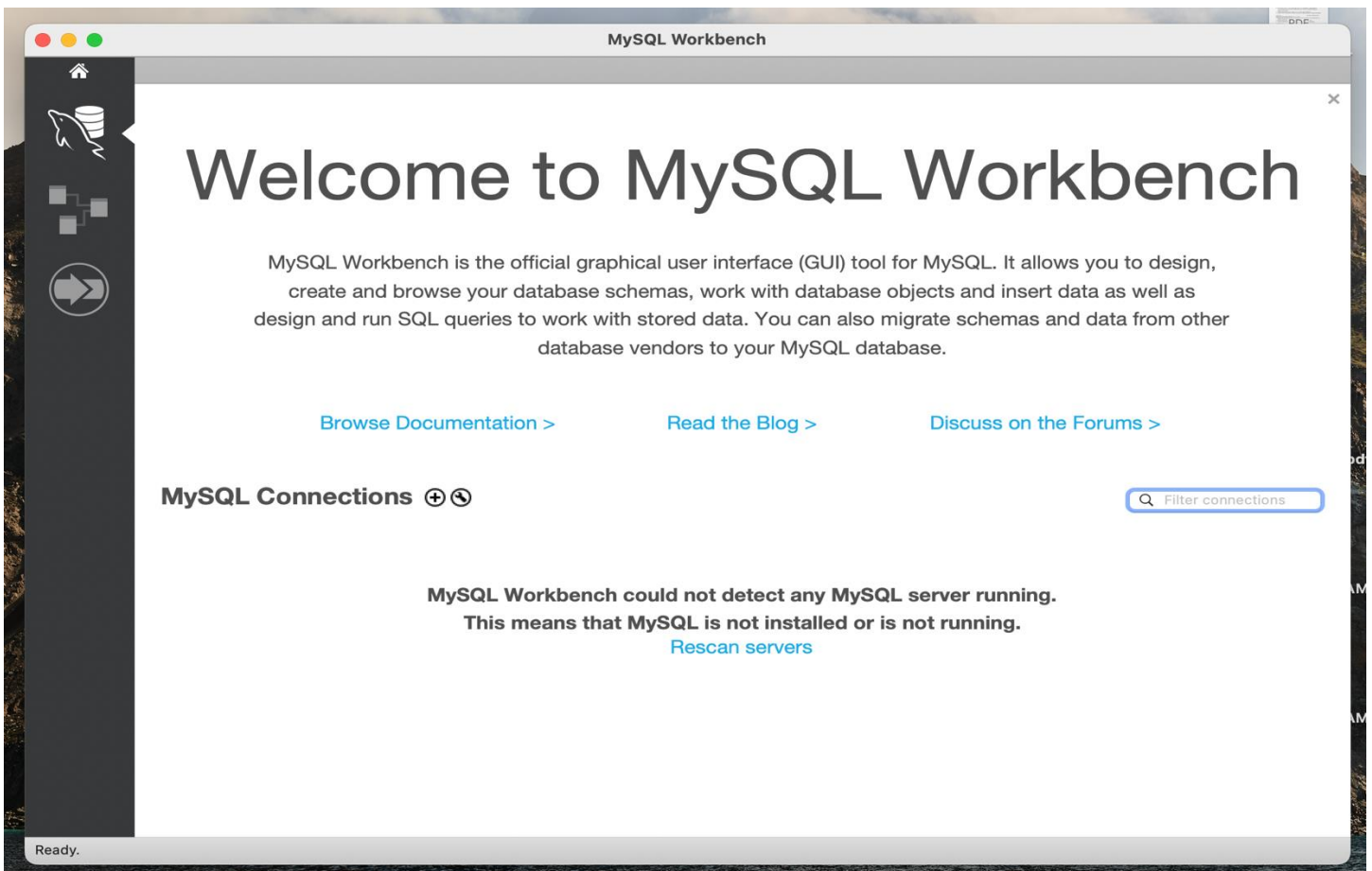
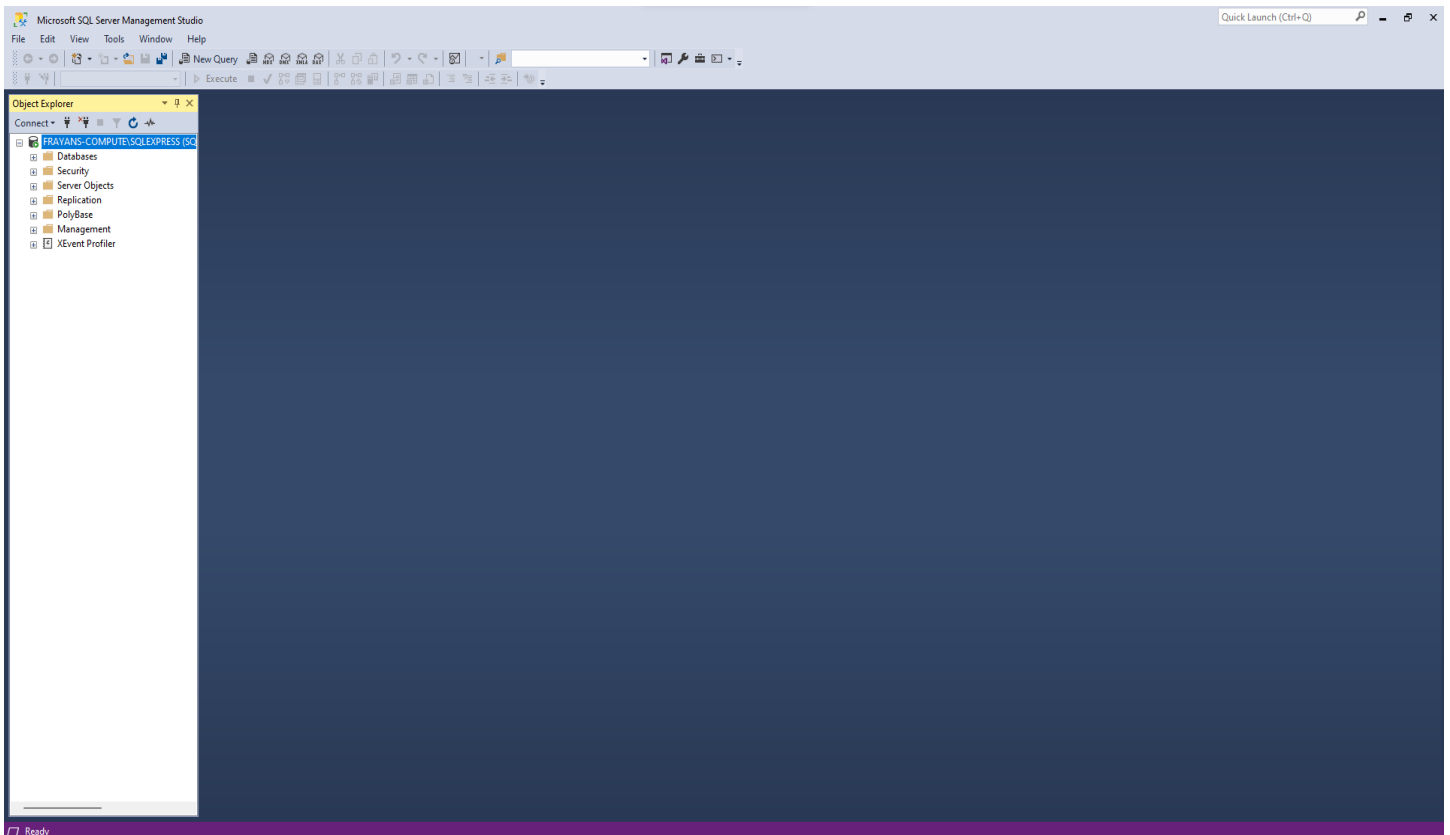
## 5. Project Description

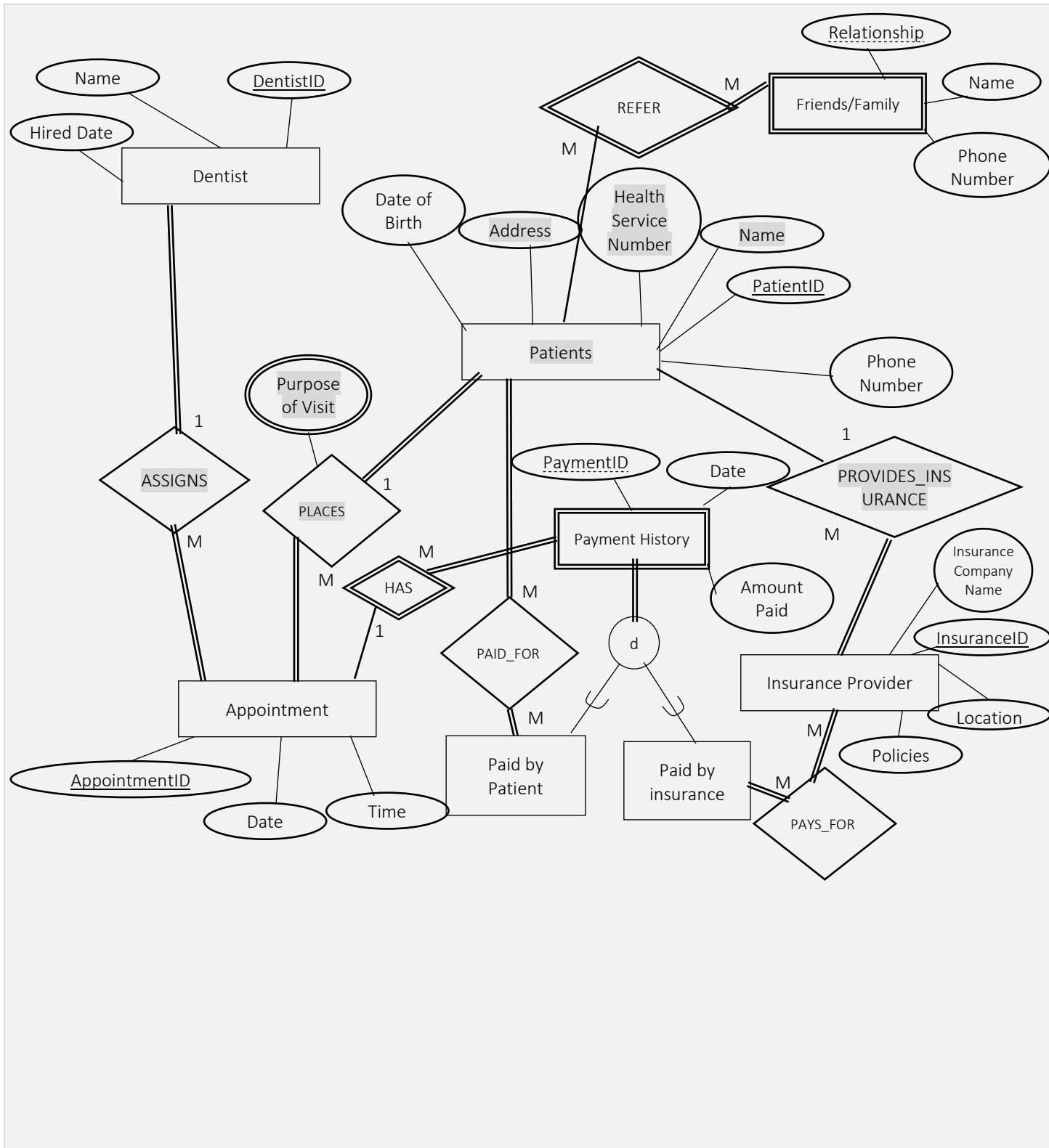
- There is a dental clinic managed by several dentists. Each dentist receives some patients. For each patient several appointments are booked in different days and times. Times will be based on when the clinic opens and closes. Whenever a patient wants to book an appointment, they are required to give their name, date of birth and phone number. Each patient may have one or more purposes of visit. We will need to store Each patient may have 1 or more insurance policies. We need to store the information of insurance policies of the patients. This will include the insurance provider and their information. This information may include InsuranceID, Insurance Company Name, location, and other requirements. Some patients may not have an insurance policy and they pay the expenses from their pocket. In this case, they will deposit how much they are paying. We need to store the payment history of the patient as well. Patients can refer their friends and families and we store their information in the database too.

## 6. Assumptions about Cardinality and Participations

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

- All patients will have at least one dentist
- All dentists will have many patients
- One patient will have one appointment at a time
- Each patient may book many appointments
- One or more patients will pay expenses from their pocket





## 7. ER-Model Mapping to Database Relational Schema

Dentist(DentistID,Name,HiredDate)

Appointment(AppointmentID,Date,Time,**DentistID**)

Patient(PatientID,HealthServiceNumber,DateOfBirth,Address,Name,PhoneNumber)

InsuranceProvider(InsuranceID,CompanyName,Location,Policies,**PatientID**)

PatientRelationship(PatientID,Relationship)

PatientRelationshipContact(**PatientID**,PatientRelationName,PatientRelationPhone)

Appointment\_PurposeOfVisit(**AppointmentID**,PurposeOfVisit)

AppointmentPatientID(**AppointmentID**,**PatientID**)

AppointmentPaymentHistory(PaymentID,AmountPaid,AmountPaidByPatient,AmountPaidByInsurance,**AppointmentID**)

Patients\_PaidFor\_PaidByPatient(**PatientID**,**PaymentID**)

InsuranceProvider\_PaysFor\_PaidByInsurance(**PaymentID**,**InsuranceID**)

## 8. Normalization

- Dentist(DentistID,Name,HiredDate)
  - 1NF
    - Does not have a composite attribute
    - Does not have a multivalued attribute
    - Does not have nested relations
  - 2NF
    - Already in 2NF because there is no composite key
  - 3NF
    - Already in 3NF because there are no transitive functional dependencies within the relations above
  - BCNF
    - There are no other FDs, Therefore already in BCNF
- Appointment (AppointmentID, Date, Time, Room, **DentistID**)
  - 1NF
    - Does not have a composite attribute
    - Does not have a multivalued attribute
    - Does not have nested relations
  - 2NF
    - Since every non-prime attribute is fully functionally dependent on the primary key, this table is already in 2NF.
  - 3NF: Is there any transitive functional dependencies?
    - No there is not since it has been transferred into a new
  - BCNF
    - There are no other FDs, which means all relations are in BCNF
  - Decomposed already before checking with normalization
    - Appointment\_PurposeOfVisit(**AppointmentID**,PurposeOfVisit)
    - AppointmentPatientID(**AppointmentID**,**PatientID**)
    - AppointmentPaymentHistory(PaymentID,AmountPaid,AmountPaidByPatient,AmountPaidByInsurance,AppointmentID)
    - Patients\_PaidFor\_PaidByPatient(**PatientID**,**PaymentID**)
    - InsuranceProvider\_PaysFor\_PaidByInsurance(**PaymentID**,**InsuranceID**)

- Patients (PatientID, HealthServiceNumber, DateOfBirth, Address, Name, PhoneNumber)
  - 1NF
    - Does not have a composite attribute
    - Does not have a multivalued attribute
    - Does not have nested relations
  - 2NF
    - Since every non-prime attribute is fully functionally dependent on the primary key, this table is already in 2NF.
  - 3NF: Is there any transitive functional dependencies?
    - No there is not since it has been transferred into a new
  - BCNF
    - There are no other FDs, which means all relations are in BCNF
  - Decomposed already before checking with normalization:
    - PatientRelationship(PatientID, Relationship)
    - PatientRelationshipContact(PatientID, PatientRelationName, PatientRelationPhone)
- InsuranceProvider (InsuranceID, InsuranceCompanyName, Location, Policies, **PatientID**)
  - 1NF
    - Does not have a composite attribute
    - Does not have a multivalued attribute
    - Does not have nested relations
  - 2NF
    - Since every non-prime attribute is fully functionally dependent on the primary key, this table is already in 2NF.
  - 3NF: Is there any transitive functional dependencies?
    - No there is not since it has been transferred into a new
  - BCNF
    - There are no other FDs, which means all relations are in BCNF

## 9. Determining Data Types (Domain) and Constraints

- Dentist Table
  - DentistID has a Int because it isn't known how many digits a DentistID has and that it can be fully customized by the dentist office themselves
  - DentistName has a Varchar length of 50 because there is a possibility that a Dentist may have a long first name or last name
- Patient Table
  - Healthservicenumber is Int because it's length is 9 and it can be entered with no spaces
  - PatientID is int since it can be fully customized by dentist office
  - Char(10) is used for phone number because it will always be 10 digits.
  - Since the address can be long sometimes, Variable character of length 100 is used.
  - PatientName has a Varchar length of 50 because there is a possibility that a Dentist may have a long first name or last name
- Patient Relationship Table
  - A varchar of length 20 is used to store the relationship between the patient. The relation names are not really long.
- Patient Relationship Contact Table
  - PatientRelationName has a Varchar length of 50 because there is a possibility that a Dentist may have a long first name or last name
  - PatientRelationPhone: Char(10) is used for phone number because it will always be 10 digits.
- Insurance Provider Table
  - InsuranceIDNumber is a varchar as it can be as long as 20 characters and is a primary key
  - InsuranceCompanyName can be long so a varchar of 40 is applied
  - For the InsuranceProviderLocation, Variable character of length 100 is used, Since the address can be long sometimes
  - InsurancePolicies have a long description usually of what it covers so it is a 100 character long

- Appointment Table
  - o AppointmentID is a primary key for this table with data type INT because various appointments are booked for a single day and it can go in thousands for months, so the number is not fixed
  - o Date and Time are used for date-time of booked appointments.
- AppointmentPurposeOfVisit
  - o The PurposeOfVisit is 15 character variable as it just states the cause of patient's appointment
- AppointmentPaymentHistory
  - o The primary key for this table is a composite key that consists of PaymentID (that is an INT because the length for a paymentID is unknown) and appointmentID which is a foreign key from appointment table.
  - o INT is used for total amount paid and the amount paid by patient and insurance respectively. Since the amount paid is dependent on what treatment a patient takes, it cannot be limited.

## 10. Creating Database and Tables - SQL DDL

SmileDatabase\_CreateTablesQuery.sql

SmileDatabase\_DropTablesQuery.sql

## 11. Inserting Values in Tables

SmileDatabase\_InsertionQuery.sql

## 12. SQL Queries

RetrivalQueries.txt or SmileDatabase\_RetrivalQueries.sql

## 13. 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.