# Database Management


Team Members

- Jiawei Zhu
- Marium Sarah
- Rahuldeep Shenoy
- Victor Ehizonomen

# Table of Contents

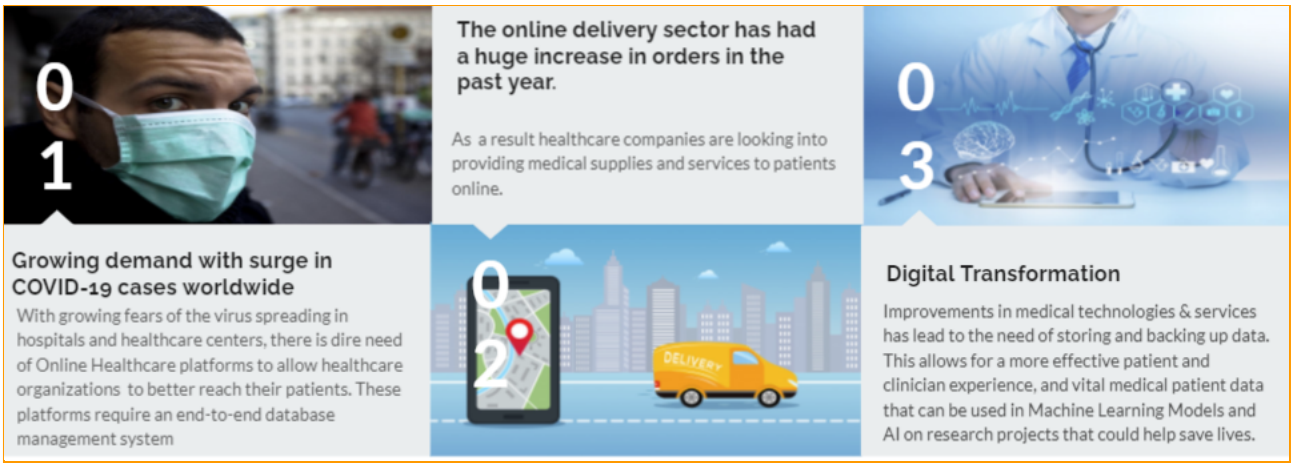
# 1 | Introduction

## I. Topic Overview

The subject of our real-word database is “Healthcare”.

Healthcare is important when it comes to anyone in general. Every day, there are injuries that happen from people, and diseases, viruses, and vaccines being developed. In addition, health issues continue to happen. In this case, both medical services and medical demands will continue to always grow. Compared to the centuries in the past, there were many changes that have happened, and will continue to, the society has placed all hopes in this industry to come through. This includes improvement on technologies, medical services, and medical demands. If the industry is not managed properly, it could cost life, batter the health systems, and damage livelihoods.

### Why Healthcare?



## Company Introduction

The company we focused on to build our real-world database is...McKesson.

McKesson is one of the largest providers of medicines, pharmaceutical supplies, and health information technology (IT) products and services in the United States and rated as the oldest and largest healthcare company in the nation, serving more than **50% of United States hospitals** and **20% of physicians**. The company was founded in **1833** by John McKesson and Charles Olcott in New York with a focus on importing and wholesaling pharmaceutical products, branded as the central nervous system of healthcare in the United States.

**Their goal is to**

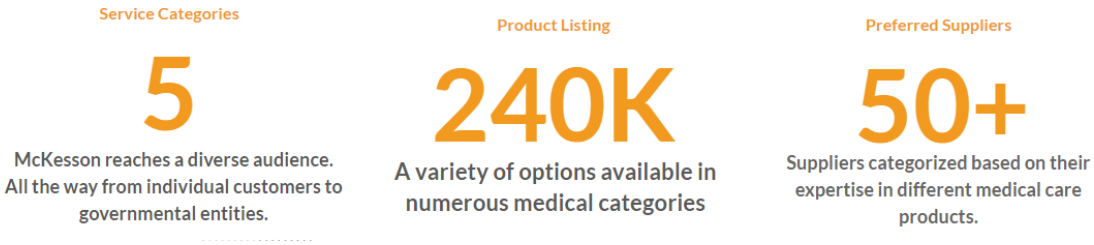
- Improving healthcare in every setting - one product, one partner, one patient at a time.
- Making medications available to customers and their patients during this critical time. McKesson is currently supporting the U.S Government as a centralized distributor of COVID-19 Vaccines and ancillary supply kits needed to administer vaccinations.
- Provide Business and Clinical resources to healthcare providers aiding growth in terms of business profit and ensure customers health and safety needs are met.

**Why did we choose McKesson?**

- Diverse customer base and complexity of services that can be transformed into a well-structured comprehensible database connected to a database application.
- Readily available data on their website will be critical in our SQL implementation.
- Frequent scalability of service offering. This can be handled well with a database management system

**The following key concepts of Big Data clearly state the organization's need of a database management system.**

- **Volume:** They have over 240k products and 80k employees and various different sectors in the organization.
- **Variety:** Unstructured and structured data that can be stored well in a database. (X-ray imagery, dental video recordings, mechanical ventilator records etc.)
- **Velocity** The need for a fast retrieval of vital data that could potentially save lives.



# II. General Description

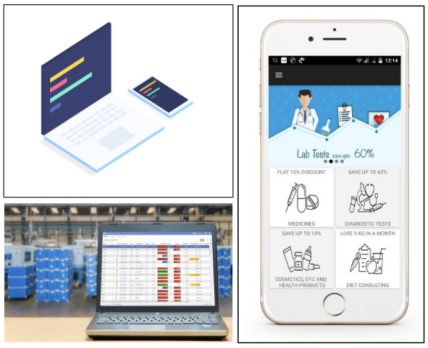
McKesson has reached out to our group concerning an application in development that requires a database management system.

**Organizational Requirements**

With the vital need of an interactive healthcare platform during this critical time. McKesson is working to launch inter-connected interfaces by the end of July'2021. The following are the applications their Product team is working to build and require our assistance to create a full fledged database management system.

- **Mobile-based Application:** Allowing customers to search up medical supplies and services; make purchases, track orders, organized calendar for medicine usage tracking, communicate with healthcare providers directly, book appointments through a single centralized application.

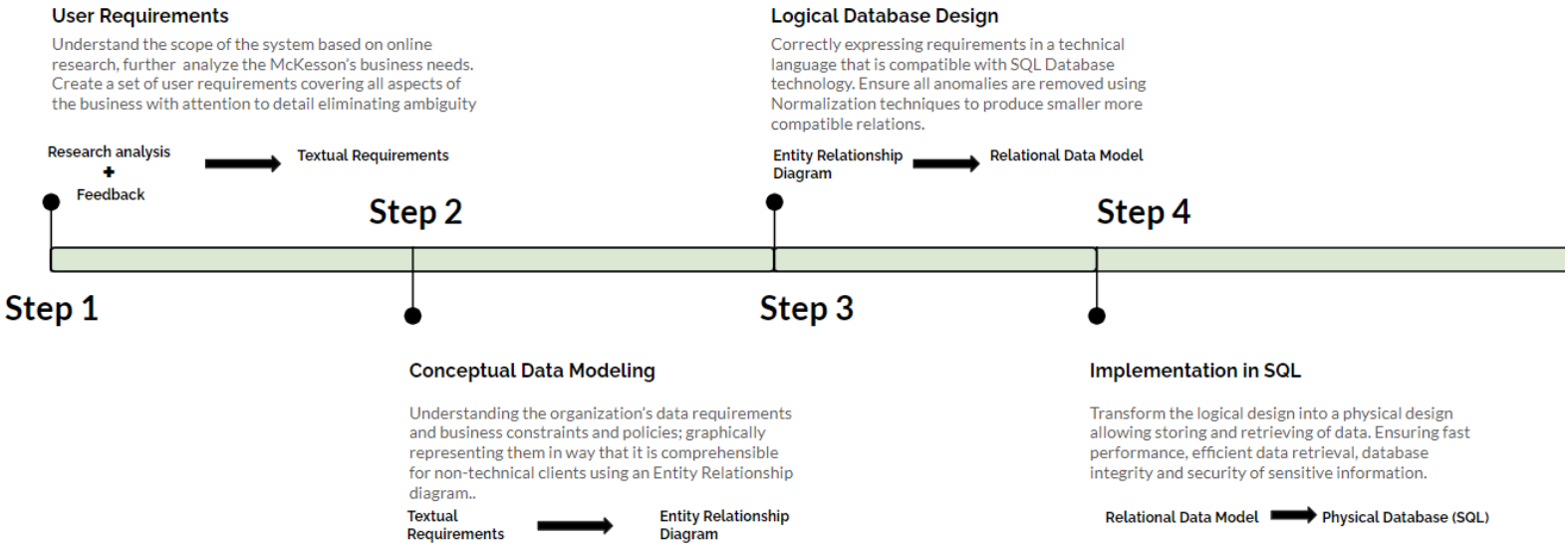
- **Healthcare Center System:** Providing a tool for healthcare workers to review patient tests and medicine usage history, prescribe medicines, schedule diagnostic tests, and communicate with patients with regard to their symptoms and diagnosis.



## Project Goal

- To provide an end-to-end database management system inter-connecting the 3 different applications that McKenson is working on.
- Taking into account possible scalability of application in the future.
- Provide a comprehensive and easily understandable solution to a complex user-requirements
- Provide an optimized data storage solution to ensure fast retrieval, easy search and privacy of sensitive data.

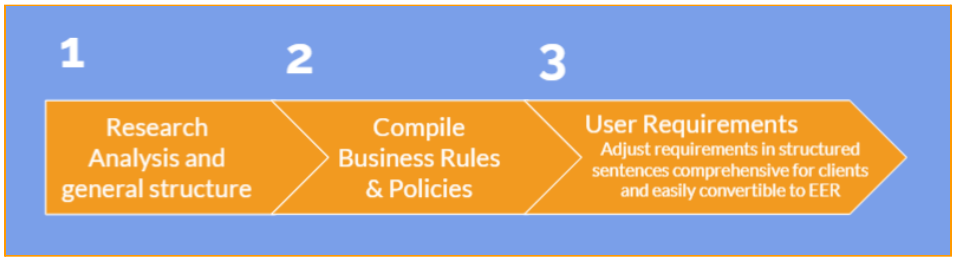
# III. Report Organization



# 2 | User Requirements

Gathering Business rules is the first step of the database development process. This involves researching the website and defining precise rules to answer questions that fit all circumstances and scenarios.

## I. Process Overview




## II. Requirements Outline

McKesson has hired our group to create a database management system to support their new user interface platform (database application. The following is a draft overview of the requirements to build this entity diagram:

- The entity diagram must include a flow showcasing the procedure to make orders for medical supplies and services for customers. This includes
  Customers, Orders, Items, Suppliers, Brand Supplier and its Branches.

- The entity diagram must support the entities, relations and attributes involved in booking an appoint for customers. Viewing their diagnostic test requirement schedules and doctors appointed etc. This includes
  Healthcare provider, Appointment Booking

- The entity diagram should include the information needed to allow health care providers to assign tests to customers, provide prescriptions and report on their appoints with customers. This includes
  Usage schedule for prescription, Assigned tests and information about the doctors themselves and nurses assigned to take tests.

# III. User Requirements

**The following are the list of entities, attributes, and business rules we have come up with:**

- A **customer** can submit orders for various **products**.
  - A **customer** can submit any number of **orders**, but may not submit any orders at all.
  - Each **order** is submitted by exactly one customer.
  - The identifier for an **Order** is Order ID, and other attributes are concerned with delivery information including City, Area, State, ZIP code and Street. Order Date, Delivery Phone Number and order price are also attributes part of the Order.
  - A **customer** has the attributes Customer ID, Customer Name which includes FirstName and Last Name, Health care insurance ID, Gender, Birthday and a composition of Weight and Height called the BMI.
  - A **customer** can have none or one Emergency Contact. The identifier that distinguishes an emergency contact is their Emergency Contact Name (First Name and Last Name) combined. An emergency contact cannot be a stand-alone entity since it depends on the customer to co-exist. An emergency contact has further information including Date of Birth, Phone Number and address.
- A customer **order** must include one or more **offerings**.
  - An **offering** may not be added on an **order** but may also be added to more than one **order** at a time.
  - An **offering on an order** must be supplied by one and only one supplier branch. But the Supplier brand may be linked to 0 or many offerings on orders.
  - Attributes associated with an **offering** and **order** include Quantity, Scheduled Delivery Date and Actual Delivery Date
- An offering can be a **service** or an **item**. But an offering must be one of them and cannot be both at once.
  - An **offering** includes the attributes Offering ID, Offering Name, Created Date, Price, Description and Offering Type
  - When an **offering** is an Item, it includes the attributes Quantity, Item Type, Unit of Measure.
  - A **supplier branch** has one or many items that it sells. And an item can be sold in none or many **supplier branches**.
  - Attributes associated with an **item** and **supplier branch** include Availability Status, Restocking date, Available Quantity, Next Shipping Date and Is Active Status identifying if the item is available in the **supplier branch** based on the Available Quantity Attribute.
  - When an offering is a **service** it has the attributes Time Taken, Given Time, Service Rating, Schedule Type and Duration.
- A **supplier brand** has one or many different **supplier branches** around the city. (Does not cover the storage information on a branch level - this is considered to be automated in the front end)
  - A **supplier brand** includes the attributes Supplier ID which is the identifier, Brand Name, Brand Description and Supplier Type.
  - Every supplier branch is linked to one and only one supplier brand
  - The one or many **supplier branches** under a **brand** has the attributes Supplier Branch ID, Branch Name, Branch Location, Branch Activation Date, Branch Closing Date and Is Online which is a derived tag from the Branch Closing Date.
- A **customer** can submit 0 or many **appointment bookings**.
  - An **Appointment booking** includes the attributes Appointment ID, Appointment Date, Appointment Type, Requested Doctor Type, Reason Description and Appointment Review.
  - An appointment booking must be made by one and only one customer at a time.
  - One and only one **healthcare provider** must be assigned to every **appointment booking**. However, a healthcare provider may be part of 0 or many **appointment bookings**.
  - The Attributes associated with an **appointment booking** and **healthcare provider** include Result Status, Appointment Return Date

- An **appointment booking** may involve none or many **usage schedules** as prescribed by the **doctor**.
  - A **usage schedule** must be linked to one and only one **appointment booking** at a given time.
  - A **usage schedule** provides information about the schedule to follow for a specific **offering**. This information includes Schedule ID, Until Date (which is the offering usage end date), the periodic setting attribute which is the times of the day the offering needs to be taken and the day of the week it needs to be taken. There are multiple periodic settings for a single usage schedule.
  - A usage schedule must be associated with one offering at a given time.
  - An offering may be part of 0 or many usage schedules at a given time.
- A **health care provider** can be an **independent doctor** or a **health care center doctor**. A **healthcare provider** can be both of the mentioned at once. But a health care provider must be at least one of them.
  - A health care provider has the attributes: Doctor ID, Full Name, Multiple Doctor Specializations and Phone Number
  - If a **healthcare provider** is an **independent doctor** he or she has additional attributes including Hourly Fee and Doctor Address.
  - If a **healthcare provider** is a doctor at a **health care center**(healthcare doctor), the attribute Health-Care Employee ID is included.
  - A health care doctor can be employed at one and only one Health care center. Every health care doctor is managed by one and only one healthcare doctor. A health care doctor manages 0 or many health care doctors.
- A **Health care center** consists of one or many health care **doctors** and **nurses**.
  - A **health care center** has the attributes Health Care Center ID, Health care center Name and Partnership Start Date
  - A nurse must be part of one and only one **health care center**. A **nurse** has the attributes Nurse ID, Nurse Name, Nurse Specialization, Work Status and Salary.
  - A **nurse** is assigned to perform zero or many customer tests that need to be done. Every assigned test needs to be performed by one and only one nurse.
- A **test** can be scheduled for one or no **appointment booking** for a single **customer** by a one assigned **healthcare provider**.
  - However, a **healthcare provider** may assign many tests or may not assign a test at all. A **customer** may attend none or many **assigned tests.** And an **assigned test** may or may not be associated with one **appointment booking**.
  - An assigned test includes the attributes Assigned Test ID, Expected Date, Taken Date, Test Result Description and Test Rating.
- An **assigned test** is associated with one and only one **diagnostic test**.
  - A **diagnostic test** can be associated with none or many **assigned tests**.
  - **Diagnostic test** has the attributes Test ID, Test Name, Test Description, Approximate Time, Test Price and Test Type.

# 3 | Conceptual Data Modeling

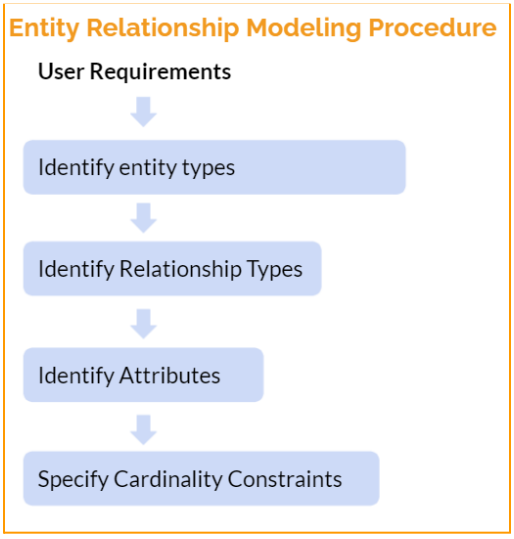
Conceptual Modeling is done when we understand the data requirements and business rules given in the User requirements and represent them visually in the form of an ER Diagram.

## I. Concepts Outline

The following are the basic constructs learned through lectures and included as part of our ER diagram..

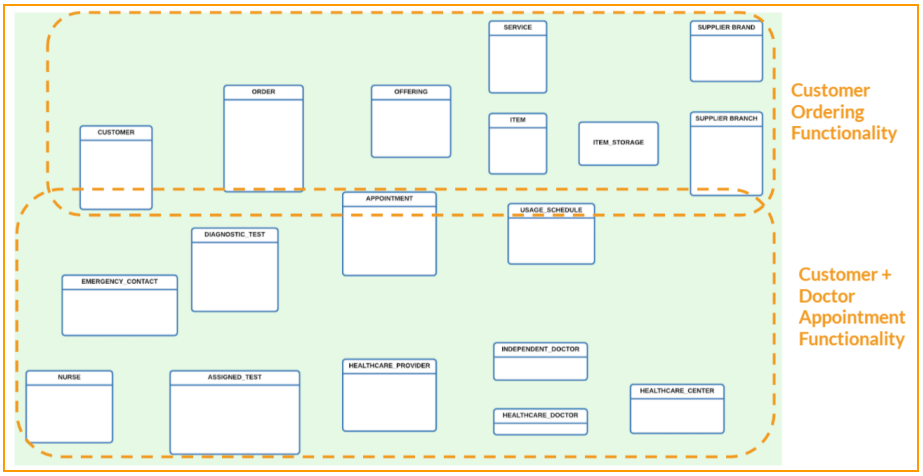
1. **Entity Types**
   - ➔ Strong/Weak Entity Types
   - ➔ Associative Entity Types
   - ➔ Subtype and Supertype
2. **Relationship Type**
   - ➔ Degree of Relationship Type: Unary, Binary, Ternary
   - ➔ Cardinality Constraints: One-to-One, One-to-Many, Many-to-Many (at least 1 required/optional)
   - ➔ Subtype/Supertype constraints:
     - ◆ Completeness Constraint(Total/Partial)
     - ◆ Disjoint Constraint (Disjoint/Overlap)
3. **Attributes**
   - ➔ Required/Optional
   - ➔ Simple/Composite
   - ➔ Single-Valued/ Multi-valued
   - ➔ Composite
   - ➔ Derived
   - ➔ Identifier
   - ➔ Discriminator

# II. Conceptual Modeling Process



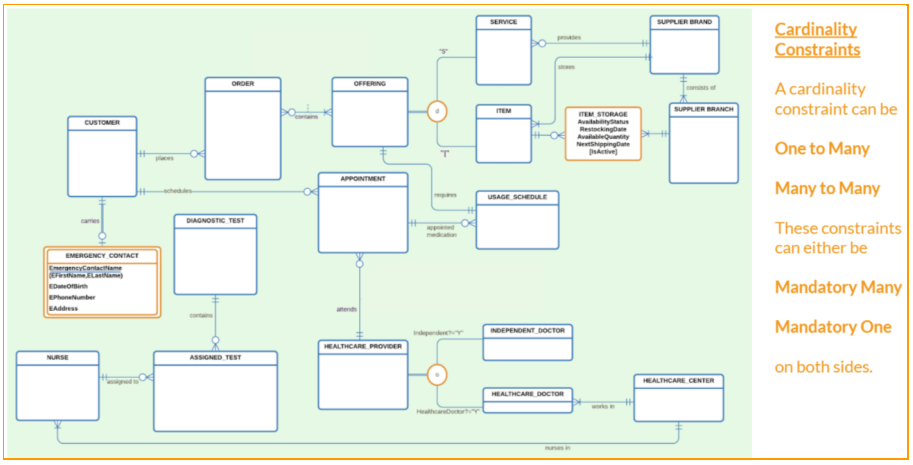

**Step 1**
**Identifying Entity Types**

The ER diagram expresses two main functionalities, the ordering of medical supplies and the booking of doctor appointments. The following are all the entity types organized from the user requirements provided.



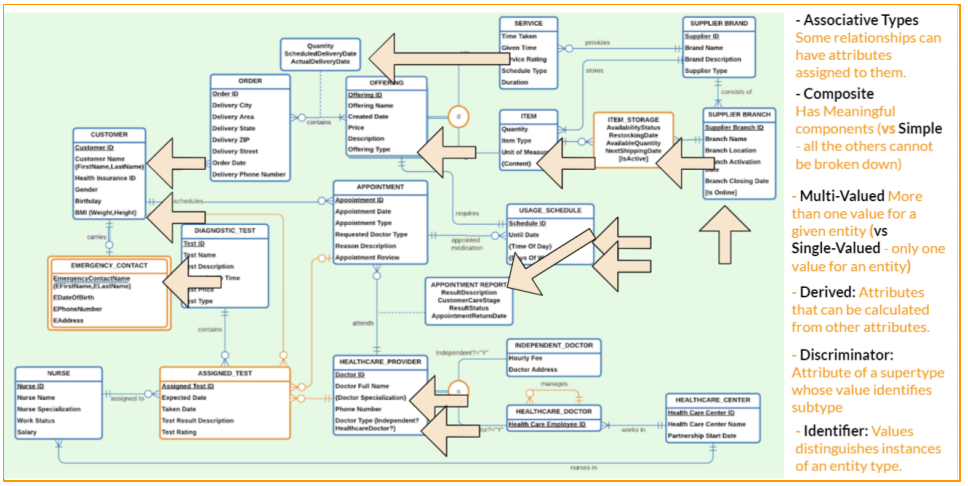
## Step 2
## Identify Relationship Types & Cardinality Constraints

The following diagram consists of the relationships between entity types as well as the cardinality constraints that were identified from the User requirements.



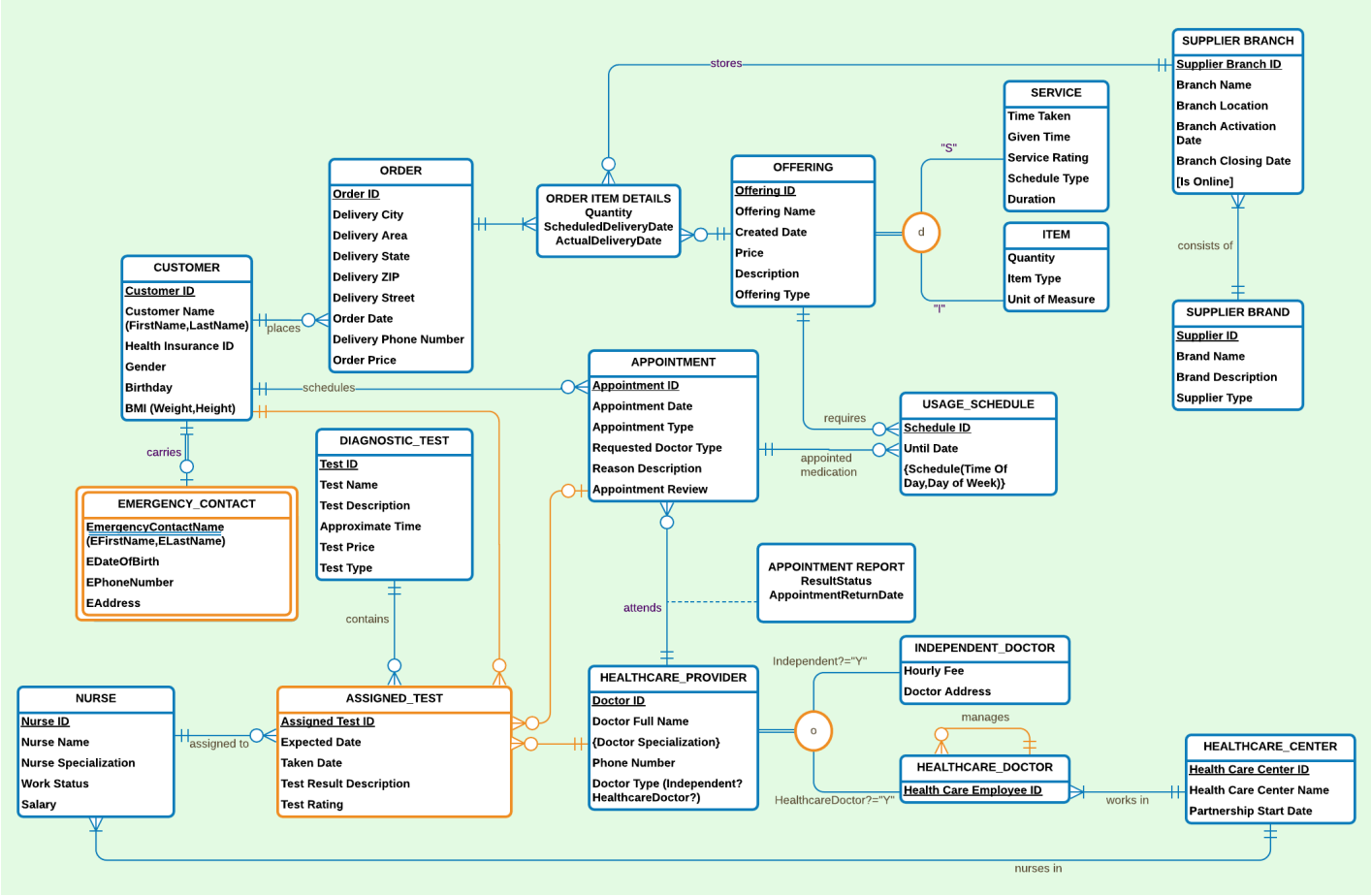

## Step 3
## Identify Attributes

The following diagram provides all the different types of Attributes that were used in our ER Diagram based on the User requirements provided.



## III. ER Diagram

The following is the final ER Diagram that was developed on LucidChart. This ER Diagram is rectified based on our discussion with the professor in Discussion 1 as well as caveats that were identified while building the Logical Database design in the next step.




[Link to ER Diagram on LucidChart]

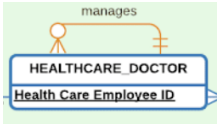
**Entity Types:** Collection of entities that share common properties and characteristics. *CUSTOMER, ORDER*

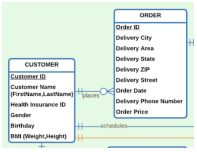
1. **Weak Entity type: *Emergency Contact*** is an example of Weak Entity Type. Weak entity types have a double outline and the partial identifier for weak entity types have a double underline. The existence of the Emergency Contact entity type depends on the Customer entity type. The Customer entity type in this case is considered the identifying owner.

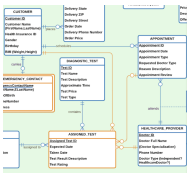
2. **Associative Entity Type:** ***Order Item Details*** is an example of an Associative Entity Type that represents a relationship between an Order and the offerings associated with it. Both the offering and order entity have a many relationship and there is a presence of more than one attribute in this relationship type.
3. **Supertype/Subtype:** Supertypes are connected to each of the subtypes.
    a. **Completeness Constraint:** addresses whether the supertype must be a member of at least one subtype or not. Total Specialization rule means a member of the supertype must be a member of a subtype. Partial Specialization rule means a member of a supertype must not belong to a subtype. **Example of Total Specialization Rule:** An offering must be a service or an item, A health care provider must be an independent doctor or a health care doctor
    b. **Disjoint Constraint:** addresses if an instance of supertype can be a member of both subtypes (overlap) or can only be a member of one subtype (disjoint). **Example of Overlap Rule:** A health care provider can be an independent doctor, health care provider or even both. **Example of Disjoint Rule:** An offering must be either an item or service, it cannot be both.

**Relationship Types:** Meaningful association between entity types

1. **Degree of Relationship Type:**
    a. **Unary Relationship:** A relationship between the instances of a single entity type - recursive relationship. In our diagram we have a Health care doctor managing one or many health care doctors.
    b. **Binary Relationship:** Relationship between two entity types. Most Commonly used relationship degree. For example, The relationship between the Customer and order entity type.
    c. **Ternary Relationship:** Simultaneous relationship among instances of three entity types. In our diagram, An Assigned test is given to a customer by a health care provider on a specific appointment. This is converted into an associative entity type for simplicity.



2. **Cardinality Constraints:**
    a. **Mandatory One:** An order must be linked to one and only one customer at a given time.
    b. **Mandatory Many:** A supplier brand has one or many supplier branches under it.
    c. **Optional One:** An assigned test for a customer may or may not be associated with a certain appointment booking.
    d. **Optional Many:** A customer can make 0 or many orders

**Type of Relationships:**

a. **One to One:** A single prescription **usage schedule** is assigned to a single **Offering**
b. **One to Many:** A single **Customer** can make many **orders**

**Attributes:** Property or characteristic of an entity type or relationship type that is of interest to the organization

1. **Required vs Optional:**
    a. **Required:** An attribute that has to be present for each entity instance. Example: Customer Name in Customer Entity type.
    b. **Optional:** An attribute that does not always have to have a value. Example: Doctor Address in Doctor Entity type.
2. **Composite vs Simple**
    a. **Composite:** An example of a composite attribute is Customer Name in the Customer Entity type. The customer name has two meaningful components First Name and Last Name.
    b. **Simple:** An example of a simple attribute is Delivery City in Orders Entity type. This attribute cannot be broken down into smaller components.
3. **Single-Valued vs Multi-Valued**
    a. **Single-valued:** Attributes that can only have one value. For example: Gender in Customer Entity Type
    b. **Multi-valued:** Attributes that can have more than one value. For example Doctor Specialization in Health Care Provider entity type.
4. **Derived:** Is Online is a derived attribute from Supplier Branch Closing date attribute in the Supplier Branch Entity Type. Is Online is considered derived since it can be calculated from another attribute.
5. **Subtype Discriminator:** Doctor type in Health care provider entity type and offering type in Offering entity type helps identify which subtypes the supertypes are associated with.
6. **Identifier:** An attribute or a combination of attributes that distinguish each entity instance of an entity type. For example Customer ID in Customer entity type.
7. **Partial Identifier:** The Weak Entity type Emergency Contact has the unique identifier EmergencyContactName which serves as the partial identifier for the entity.

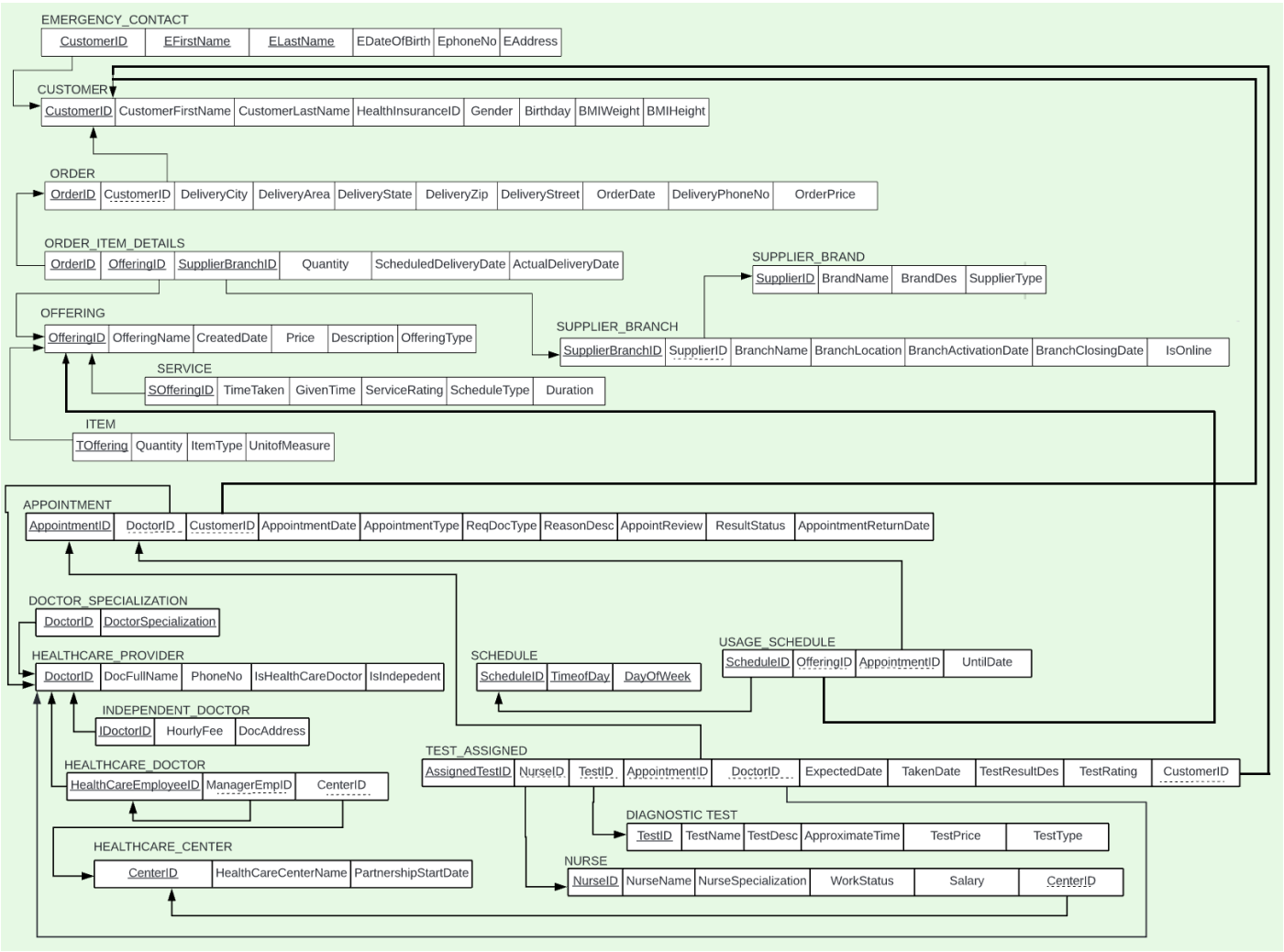
# 4 | Logical Database Design

The logical database design helps model the ER diagram in a form that is consistent and compatible with a database technology. It helps create a stable database structure to correctly express requirements in a technical language.

## I. Concepts Outline

1. Transforming  ER Diagram into Relational Model.
    - Mapping Regular Entities
    - Mapping Weak Entities
    - Mapping Binary Relationships
    - Mapping Associative Entities with no identifier
    - Mapping Unary & Ternary Relationships
    - Mapping Supertype/Subtype Relationships.
2. Converting Attributes: Multivalued Attributes , Composite Attributes
3. Relational Keys: Primary Key, Foreign Key, Composite Key

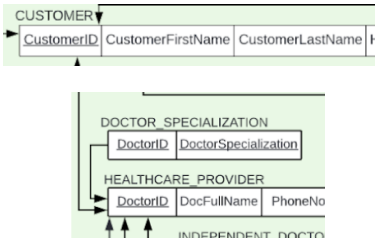
## II. Relational Model




Link to Relational Data Model - Part 1, Link to Relational Data Model - Part 2

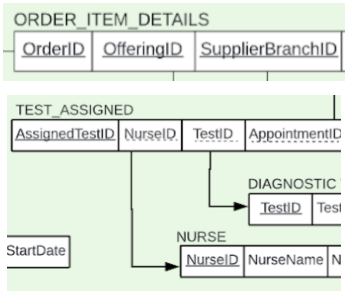
**Attributes:**

a. **Composite Attributes:** All multivalued attributes are converted to simple attributes. For example Customer First Name and Last Name are in two separate columns.
b. **Multivalued Attributes:** Two new relations are created to suffice a multi valued attribute.




**Keys:**

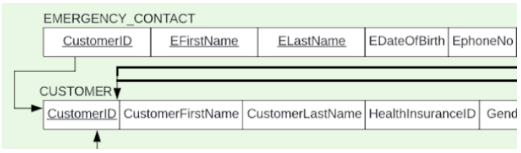
a. **Primary Key:** Attributes or a combination of them (composite) that uniquely identify relation rows. Order Item Details consists of a Composite Primary key including Order ID, OfferingID and SupplierBrandID
b. **Foreign Key (non-unique key):** Represent relationships between two tables or relations. Denoted with a dashed underline with an arrow pointing to the associated relation. For example Test Assigned has Nurse ID foreign key assigned to the Nurse ID in the Nurse relation




**Steps:**

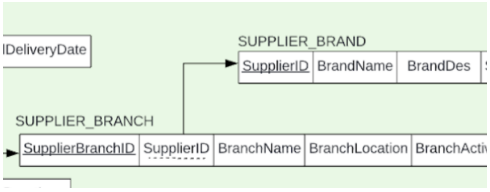
**Step 1 - Map Regular Entities:** Entities with independent existence - real world objects are transformed into relations. This includes: Customers, Health Care Provider, Offering, Supplier Brand, Health Care Center

**Step 2 - Map Weak Entities:** Each weak entity becomes a relation where the primary key of the identifying relation (Customer) is used as a foreign key attribute in the new relation (Emergency Contact).

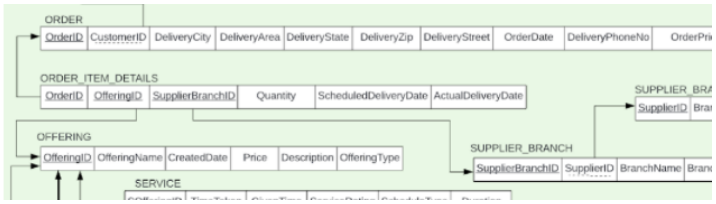



**Step 3 - Map Binary Relationships:**

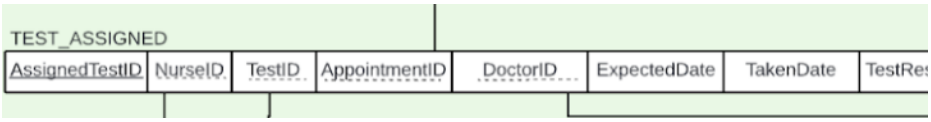
**Example of One to Many:** Every supplier branch is connected to one supplier brand. Hence the supplier ID in Supplier Branch Relation is a foreign key linked to the Supplier Brand Relation.



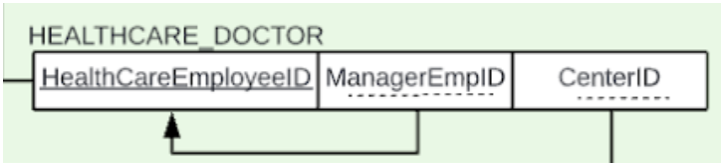
**Step 4 - Map Associate Entities with no identifiers:** Creating three relations one for each participating entity and the third is the associate entity. In this case the participating relations are Order,Offering and Supplier Branch relations and the associate relation is Order Item Details. The primary key is a combination of OrderID, OfferingID and SupplierBranchID which are all connected to the participating relations using a foreign key.



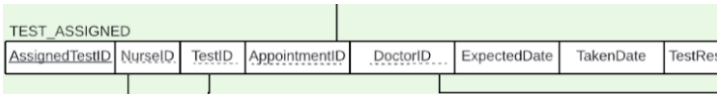

**Mapping associative entities with identifier:** Test Assigned is an example of an associate relationship turned into an associative entity. Test Assigned has the identifier AssignedTestID and has columns that connect to Nurse, Test, Appointment and Doctor Relations using a foreign key.



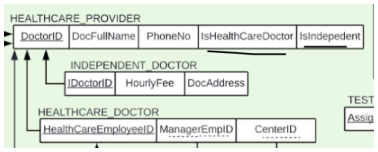

**Step 5 - Map Unary One to Many relationships:** Healthcare doctor managing health care employees this includes a foreign key in a relation that is linked to a primary key in the same relation.



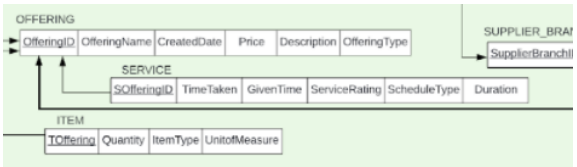

**Step 6 - Map Ternary relationships :** Test Assigned is an example of a ternary entity type that links 4 other regular entity types.




**Step 7 - Map Supertype and subtype relationships:** Health care provider has two subtypes Independent Doctor and Healthcare doctor. These two subtypes have a primary key that is directly connected to the DoctorID in the healthcare provider supertype table. IsHealthcaredoctor and isIndependent are boolean values used to identify which subtype the healthcare doctor belongs to.




Service and Item have a primary key that is directly connected to the Offering Table. The discriminator in the Offering supertype is the Offering Type.



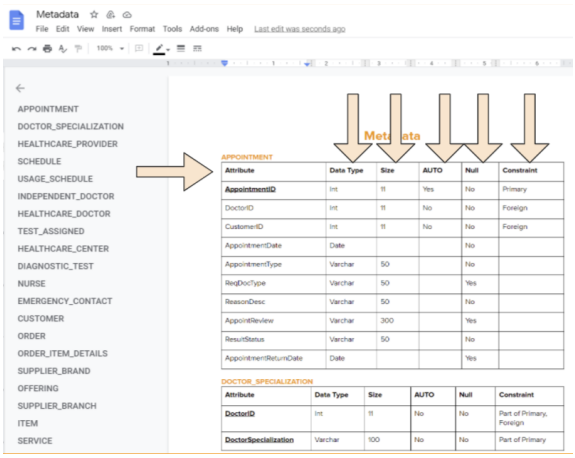
# 5 | mySQL Implementation

## I. Metadata

Metadata summarizes basic information about the data. Metadata makes it easy for developers to find information about columns. The process of creating Metadata helps compile and decide on the structure of the SQL Data Definition

The entire Metadata is visible in the Appendix Section

With the Metadata we can answer the following questions about our database:




- **Data type**
  What type of data is stored in this column?
- **Size**
  How much storage space is required for this column ?
- **AUTO**
  Does this field automatically increment during insertion?
- **Null**
  Is Input to this column required on insertion?
- **Constraint**
  Is this a unique column in the table? (Primary key)
  Is this column referring to a value in another relation? (Foreign Key)

# II. Data Definition Language

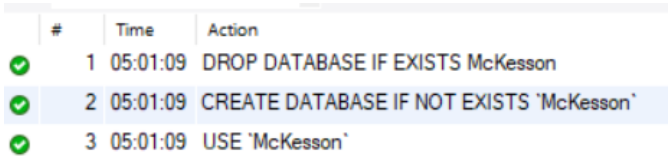
[Link to SQL Table Structure DDL]

Commands that define the structure of the database including creating altering and dropping tables and establishing constraints.

**Step 1: Create a database**

The “DROP DATABASE IF EXISTS” helps to automatically run the entire script without having to drop manually.

```
DROP DATABASE IF EXISTS McKesson;
-- Database: `healthcare_mckesson`
--
CREATE DATABASE IF NOT EXISTS `McKesson`;
USE `McKesson`;
```



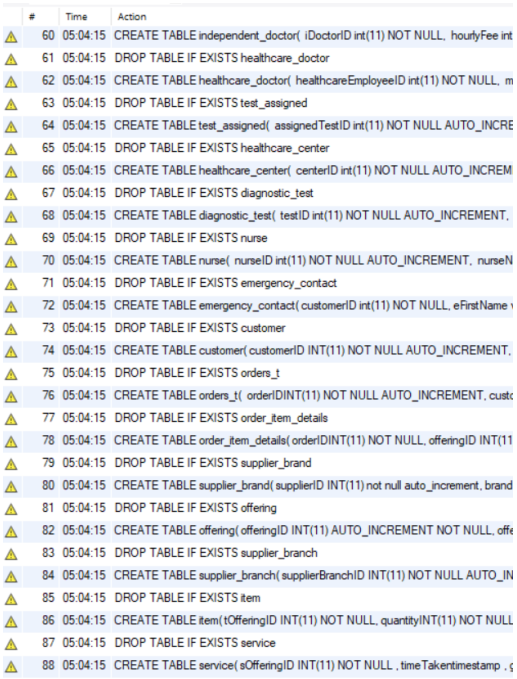
| # | Time | Action |
|---|---|---|
| 1 | 05:01:09 | DROP DATABASE IF EXISTS McKesson |
| 2 | 05:01:09 | CREATE DATABASE IF NOT EXISTS `McKesson` |
| 3 | 05:01:09 | USE `McKesson` |

**Step 2: Create all 20 tables with primary keys**

The foreign keys were set later to avoid any issues with dependencies on uncreated relations.

```
DROP TABLE IF EXISTS customer;
CREATE TABLE customer(
customerID INT(11) NOT NULL AUTO_INCREMENT,
customerFirstName VARCHAR(100) NOT NULL,
customerLastName VARCHAR(100) NOT NULL ,
healthInsuranceID Varchar(100),
gender VARCHAR(11),
birthday DATE,
bmiWeight FLOAT NOT NULL,
bmiHeight FLOAT NOT NULL,
CONSTRAINT customer_pk PRIMARY KEY (customerID)
)ENGINE = INNODB;

-- ----------------------------------------------
```

```
DROP TABLE IF EXISTS orders_t;
CREATE TABLE orders_t(
orderID INT(11) NOT NULL AUTO_INCREMENT,
customerID INT(11) NOT NULL ,
deliveryCity VARCHAR (100) NOT NULL ,
deliveryArea VARCHAR (100) NOT NULL ,
deliveryState VARCHAR (11) NOT NULL ,
deliveryZIP INT(5) ,
deliveryStreet VARCHAR (100),
orderDate DATETIME NOT NULL ,
deliveryPhoneNo varchar(10) NOT NULL ,
deliveryPrice   FLOAT NOT NULL,
CONSTRAINT orders_pk PRIMARY KEY (orderID)
)ENGINE = INNODB;
```



| # | Time | Action |
|---|---|---|
| 60 | 05:04:15 | CREATE TABLE independent_doctor( iDoctorID int(11) NOT NULL, hourlyFee int |
| 61 | 05:04:15 | DROP TABLE IF EXISTS healthcare_doctor |
| 62 | 05:04:15 | CREATE TABLE healthcare_doctor( healthcareEmployeeID int(11) NOT NULL, m |
| 63 | 05:04:15 | DROP TABLE IF EXISTS test_assigned |
| 64 | 05:04:15 | CREATE TABLE test_assigned( assignedTestID int(11) NOT NULL AUTO_INCRE |
| 65 | 05:04:15 | DROP TABLE IF EXISTS healthcare_center |
| 66 | 05:04:15 | CREATE TABLE healthcare_center( centerID int(11) NOT NULL AUTO_INCREM |
| 67 | 05:04:15 | DROP TABLE IF EXISTS diagnostic_test |
| 68 | 05:04:15 | CREATE TABLE diagnostic_test( testID int(11) NOT NULL AUTO_INCREMENT, |
| 69 | 05:04:15 | DROP TABLE IF EXISTS nurse |
| 70 | 05:04:15 | CREATE TABLE nurse( nurseID int(11) NOT NULL AUTO_INCREMENT, nurseN |
| 71 | 05:04:15 | DROP TABLE IF EXISTS emergency_contact |
| 72 | 05:04:15 | CREATE TABLE emergency_contact( customerID int(11) NOT NULL, eFirstName |
| 73 | 05:04:15 | DROP TABLE IF EXISTS customer |
| 74 | 05:04:15 | CREATE TABLE customer( customerID INT(11) NOT NULL AUTO_INCREMENT, |
| 75 | 05:04:15 | DROP TABLE IF EXISTS orders_t |
| 76 | 05:04:15 | CREATE TABLE orders_t( orderIDINT(11) NOT NULL AUTO_INCREMENT, custo |
| 77 | 05:04:15 | DROP TABLE IF EXISTS order_item_details |
| 78 | 05:04:15 | CREATE TABLE order_item_details( orderIDINT(11) NOT NULL, offeringID INT(11 |
| 79 | 05:04:15 | DROP TABLE IF EXISTS supplier_brand |
| 80 | 05:04:15 | CREATE TABLE supplier_brand( supplierID INT(11) not null auto_increment, brand |
| 81 | 05:04:15 | DROP TABLE IF EXISTS offering |
| 82 | 05:04:15 | CREATE TABLE offering( offeringID INT(11) AUTO_INCREMENT NOT NULL, offe |
| 83 | 05:04:15 | DROP TABLE IF EXISTS supplier_branch |
| 84 | 05:04:15 | CREATE TABLE supplier_branch( supplierBranchID INT(11) NOT NULL AUTO_IN |
| 85 | 05:04:15 | DROP TABLE IF EXISTS item |
| 86 | 05:04:15 | CREATE TABLE item( tOfferingID INT(11) NOT NULL, quantityINT(11) NOT NULL |
| 87 | 05:04:15 | DROP TABLE IF EXISTS service |
| 88 | 05:04:15 | CREATE TABLE service( sOfferingID INT(11) NOT NULL , timeTakentimestamp , g |

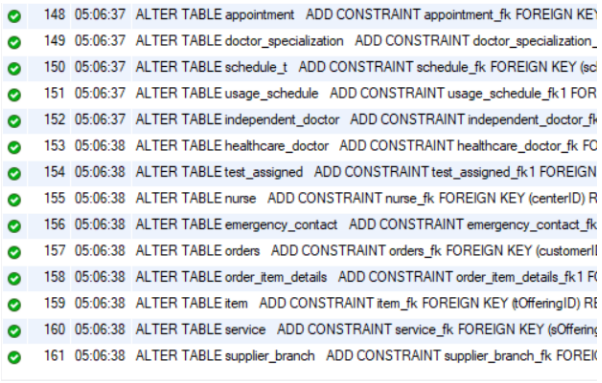
**Syntax used:**

AUTO_INCREMENT, NOT NULL, CREATE TABLE, PRIMARY KEY, CONSTRAINT

**Data Types examples:** Varchar (100), Int(11), Float, Datetime, Date

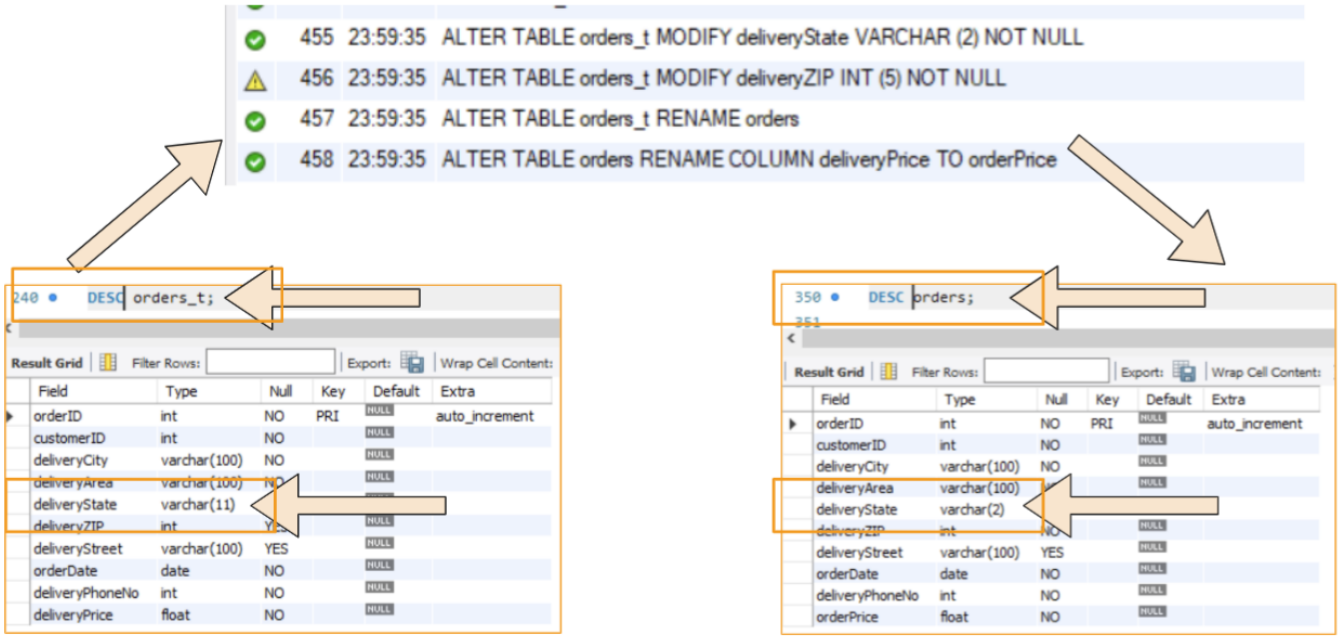
### Step 3 - Create all the connecting Foreign Key Constraints

**ON UPDATE CASCADE ON DELETE CASCADE** was used to avoid cases were deletions in one table are not effective in other connected tables. This ensures that when one row with a unique identifier is deleted from the primary table all other tables connected to this table are also deleted. Same goes for Update.

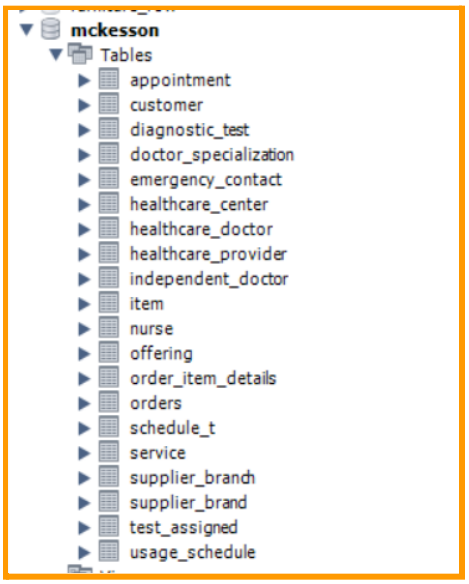



```
ALTER TABLE appointment
  ADD CONSTRAINT appointment_fk FOREIGN KEY (doctorID) REFERENCES healthcare_provider (doctorID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT appointment_fk1 FOREIGN KEY (customerID) REFERENCES customer (customerID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE doctor_specialization
  ADD CONSTRAINT doctor_specialization_fk FOREIGN KEY (doctorID) REFERENCES healthcare_provider (doctorID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE schedule_t
  ADD CONSTRAINT schedule_fk FOREIGN KEY (scheduleID) REFERENCES usage_schedule (scheduleID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE usage_schedule
  ADD CONSTRAINT usage_schedule_fk1 FOREIGN KEY (offeringID) REFERENCES offering(offeringID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT usage_schedule_fk2 FOREIGN KEY (appointmentID) REFERENCES appointment(appointmentID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE independent_doctor
  ADD CONSTRAINT independent_doctor_fk FOREIGN KEY (iDoctorID) REFERENCES healthcare_provider (doctorID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE healthcare_doctor
  ADD CONSTRAINT healthcare_doctor_fk FOREIGN KEY (healthcareEmployeeID) REFERENCES healthcare_provider (doctorID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE test_assigned
  ADD CONSTRAINT test_assigned_fk1 FOREIGN KEY (nurseID) REFERENCES nurse (nurseID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT test_assigned_fk2 FOREIGN KEY (testID) REFERENCES diagnostic_test (testID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT test_assigned_fk3 FOREIGN KEY (appointmentID) REFERENCES appointment (appointmentID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT test_assigned_fk4 FOREIGN KEY (doctorID) REFERENCES healthcare_provider (doctorID) ON UPDATE CASCADE ON DELETE CASCADE,
  ADD CONSTRAINT test_assigned_fk5 FOREIGN KEY (customerID) REFERENCES customer (customerID) ON UPDATE CASCADE ON DELETE CASCADE;

ALTER TABLE nurse
```

### Step 4: Alter table example to rename a table and fix a Data Type Delivery State



## Final Database with all 20 Tables



# III. Data Manipulation Language

Commands that are used to update, insert, modify and query the data in the database. This is the core of **SQL**.

## 1.1. Insertion of Data

Link to SQL Table Insertions

**Examples of Insertion Queries:**

```
INSERT INTO CUSTOMER(CustomerID,CustomerFirstName,CustomerLastName,HealthInsuranceID,Gender,Birthday, BMIWeight, BMIHeight)
VALUES
(666501, 'Filmore', 'Blunn', '0536-0355', 'Male', '1999-07-25', 419.29, 187.17),
(666502, 'Hercules', 'Mifflin', '50367-100', 'Female', '1999-08-14', 188.94, 119.54),
(666503, 'Griff', 'Pinel', '63629-1328', 'Male', '1994-06-11', 480.94, 164.08),
(666504, 'Dori', 'Snoding', '0004-0086', 'Female', '1993-02-02', 459.59, 229.93),
(666505, 'Billy', 'Youd', '21695-110', 'Male', '1992-10-09', 248.94, 103.51),
(666506, 'Arther', 'Adriaens', '48951-1150', 'Male', '1999-10-10', 319.24, 124.99),
(666507, 'Claudian', 'Woan', '0615-7752', 'Female', '1998-03-14', 392.13, 147.5),
(666508, 'Maren', 'Tenniswood', NULL, 'Male', '1994-07-02', 270.85, 241.51),
(666509, 'Zerk', 'Witnall', null, 'Female', '1993-12-04', 431.52, 232.81),
(666510, 'Erny', 'Surcomb', null , 'Male', '1998-04-01', 90.52, 212.38),
(666511, 'Gayel', 'Twinbrow', null, 'Female', '1992-04-16', 260.22, 127.18),
```

```
INSERT INTO offering(offeringID,offeringName,price,descriptions,offeringType)

VALUES ( 1,'Masks',11.0,'Procedure Mask Cypress Pleated Earloops One Size Fits Most Blue NonSterile ASTM Level 1','Item'),
(2, 'Sanitizer',20.0,'Hand Sanitizer McCord 8 oz. Ethyl Alcohol Gel Pump Bottle','Item'),
(3, 'Gown',11.0,'Protective Procedure Gown Large Blue NonSterile AAMI Level 1 Disposable','Item'),
(4, 'Sanitizer Ethyl Alcohol',19.9,'Hand Sanitizer HandClean 8.45 oz. Ethyl Alcohol Gel Bottle','Item'),
(5, 'Earloops',30.0,'Procedure Mask McKesson Pleated Earloops One Size Fits Most Blue NonSterile ASTM Level 1','Item'),
(6, 'Procedure Gown',5.0,'Protective Procedure Gown One Size Fits Most Yellow NonSterile Disposable','Item'),
```

**Successful Insertions in all tables:**

| | | | | | | |
|---|---|---|---|---|---|---|
| 1993 | 07:30:58 | INSERT INTO CUSTOMER(CustomerID,CustomerFirstName,CustomerLastName,HealthInsura... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |
| 1994 | 07:30:58 | insert into EMERGENCY_CONTACT (customerID, eFirstName, eLastName, eDateofBirth, eph... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |
| 1995 | 07:30:58 | INSERT INTO HEALTHCARE_CENTER(CenterID, HealthCareCenterName, PartnershipStart... | 22 row(s) affected | Records: 22 | Duplicates: 0 | Warnings: 0 |
| 1996 | 07:30:58 | INSERT INTO NURSE (nurseID, nurseName, nurseSpecialization, workStatus, salary, centerI... | 31 row(s) affected | Records: 31 | Duplicates: 0 | Warnings: 0 |
| 1997 | 07:30:58 | INSERT INTO SUPPLIER_BRAND(supplierID, brandName, brandDes, supplierType) values (... | 26 row(s) affected | Records: 26 | Duplicates: 0 | Warnings: 0 |
| 1998 | 07:30:58 | insert into SUPPLIER_BRANCH (supplierBranchID, supplierID, branchName, branchLocation, ... | 22 row(s) affected | Records: 22 | Duplicates: 0 | Warnings: 0 |
| 1999 | 07:30:58 | INSERT INTO DIAGNOSTIC_TEST(TestID, TestName, TestDesc,ApproximateTime, TestPric... | 25 row(s) affected | Records: 25 | Duplicates: 0 | Warnings: 0 |
| 2000 | 07:30:58 | INSERT INTO offering(offeringID,offeringName,price,descriptions,offeringType) VALUES( 1,'M... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |
| 2001 | 07:30:58 | INSERT INTO service(sOfferingID,serviceRating,duration) VALUES (12,null,20), (13,null,15), (... | 9 row(s) affected | Records: 9 | Duplicates: 0 | Warnings: 0 |
| 2002 | 07:30:58 | INSERT INTO item(tOfferingID,quantity,itemType,unitOfMeasure) VALUES (1,20,'Protection','... | 11 row(s) affected | Records: 11 | Duplicates: 0 | Warnings: 0 |
| 2003 | 07:30:58 | INSERT INTO HEALTHCARE_PROVIDER(DoctorID, DocFullName, PhoneNo, IsHealthCare... | 25 row(s) affected | Records: 25 | Duplicates: 0 | Warnings: 0 |
| 2004 | 07:30:58 | INSERT INTO HEALTHCARE_DOCTOR(HealthcareEmployeeID, ManagerEmpID, CenterID) ... | 15 row(s) affected | Records: 15 | Duplicates: 0 | Warnings: 0 |
| 2005 | 07:30:58 | insert into INDEPENDENT_DOCTOR (iDoctorID, hourlyFee, docAddress) values (909051, 74... | 11 row(s) affected | Records: 11 | Duplicates: 0 | Warnings: 0 |
| 2006 | 07:30:58 | insert into DOCTOR_SPECIALIZATION (doctorID, doctorSpecialization) values (201, 'Emerge... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |
| 2007 | 07:30:58 | INSERT INTO ORDERS (OrderID, CustomerID,DeliveryCity,DeliveryArea,DeliveryState, Delive... | 30 row(s) affected | Records: 30 | Duplicates: 0 | Warnings: 0 |
| 2008 | 07:30:58 | INSERT INTO ORDER_ITEM_DETAILS( OrderID, OfferingID, SupplierBranchID, Quantity, Sc... | 57 row(s) affected | Records: 57 | Duplicates: 0 | Warnings: 0 |
| 2009 | 07:30:58 | INSERT INTO Appointment(AppointmentID, DoctorID, CustomerID, AppointmentDate,Appoint... | 30 row(s) affected | Records: 30 | Duplicates: 0 | Warnings: 0 |
| 2010 | 07:30:58 | INSERT INTO TEST_ASSIGNED(AssignedTestID, NurseID, TestID, AppointmentID, DoctorI... | 30 row(s) affected | Records: 30 | Duplicates: 0 | Warnings: 0 |
| 2011 | 07:30:58 | insert into usage_schedule (scheduleID, offeringID, appointmentID, untilDate) values (1001, 1... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |
| 2012 | 07:30:58 | insert into schedule_t (scheduleID, timeOfDay, dayOfWeek) values (1001, '2000-01-01 08:00:... | 20 row(s) affected | Records: 20 | Duplicates: 0 | Warnings: 0 |

* The tables with 9-15 rows (underlined) are Subtypes of the supertype classes and hence do not have 20 rows since the supertype has all the 20 rows.

## 1.2. Querying of Data

Link to SQL Table Queries

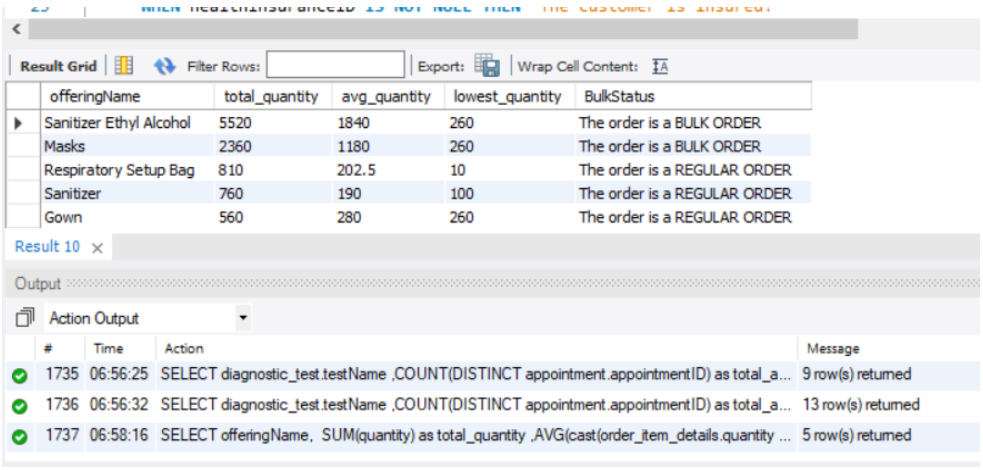
---

### 1 | A query to check the bulk status of items with orders.

**Query Used:**

```
SELECT offeringName,
SUM(quantity) as total_quantity
,AVG(cast(order_item_details.quantity as real)) as avg_quantity
,MIN(order_item_details.quantity) as lowest_quantity
,CASE
   WHEN SUM(quantity) > 1000  THEN 'The order is a BULK ORDER'
   ELSE 'The order is a REGULAR ORDER'
END AS BulkStatus
FROM offering
INNER JOIN order_item_details ON offering.offeringID = order_item_details.offeringID
INNER JOIN orders ON order_item_details.orderid = orders.orderID
WHERE offeringType = 'Item'
GROUP BY 1 ORDER BY 2 DESC LIMIT 5;
```

**Query Result:**



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| offeringName | total_quantity | avg_quantity | lowest_quantity | BulkStatus |
|---|---|---|---|---|
| Sanitizer Ethyl Alcohol | 5520 | 1840 | 260 | The order is a BULK ORDER |
| Masks | 2360 | 1180 | 260 | The order is a BULK ORDER |
| Respiratory Setup Bag | 810 | 202.5 | 10 | The order is a REGULAR ORDER |
| Sanitizer | 760 | 190 | 100 | The order is a REGULAR ORDER |
| Gown | 560 | 280 | 260 | The order is a REGULAR ORDER |

Result 10

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 1735 | 06:56:25 | SELECT diagnostic_test.testName ,COUNT(DISTINCT appointment.appointmentID) as total_a... | 9 row(s) returned |
| 1736 | 06:56:32 | SELECT diagnostic_test.testName ,COUNT(DISTINCT appointment.appointmentID) as total_a... | 13 row(s) returned |
| 1737 | 06:58:16 | SELECT offeringName,  SUM(quantity) as total_quantity ,AVG(cast(order_item_details.quantity ... | 5 row(s) returned |

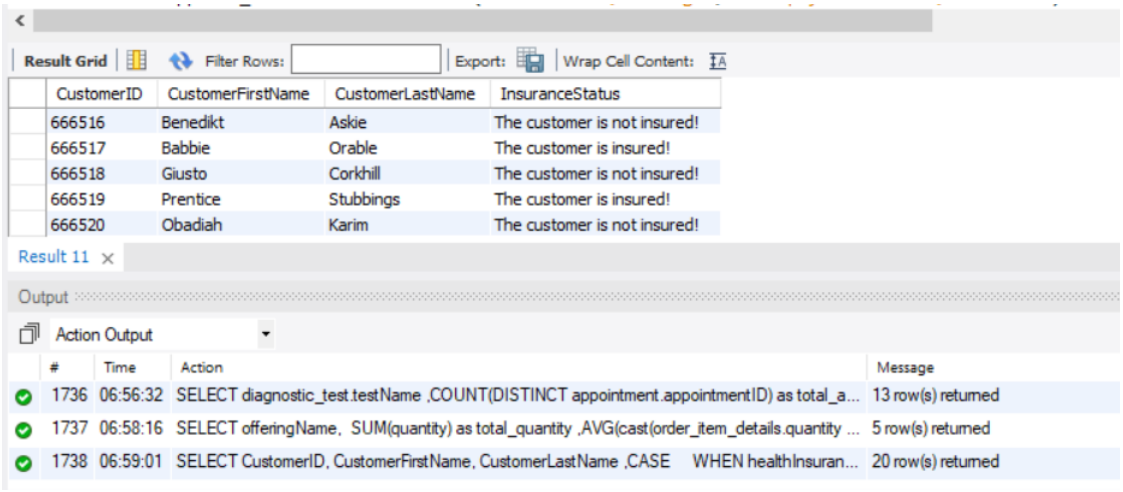
**Functionalities Used: LIKE , CASE WHEN**

## 2 | A query to check the status of insurance of a customer

### Query Used:

```
SELECT CustomerID, CustomerFirstName, CustomerLastName
,CASE
   WHEN healthInsuranceID IS NOT NULL THEN 'The customer is insured!'
   ELSE 'The customer is not insured!'
END AS InsuranceStatus
FROM customer;
```

### Results:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| CustomerID | CustomerFirstName | CustomerLastName | InsuranceStatus |
|---|---|---|---|
| 666516 | Benedikt | Askie | The customer is not insured! |
| 666517 | Babbie | Orable | The customer is insured! |
| 666518 | Giusto | Corkhill | The customer is not insured! |
| 666519 | Prentice | Stubbings | The customer is insured! |
| 666520 | Obadiah | Karim | The customer is not insured! |

Result 11

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 1736 | 06:56:32 | SELECT diagnostic_test.testName ,COUNT(DISTINCT appointment.appointmentID) as total_a... | 13 row(s) returned |
| 1737 | 06:58:16 | SELECT offeringName, SUM(quantity) as total_quantity ,AVG(cast(order_item_details.quantity ... | 5 row(s) returned |
| 1738 | 06:59:01 | SELECT CustomerID, CustomerFirstName, CustomerLastName ,CASE WHEN healthInsuran... | 20 row(s) returned |

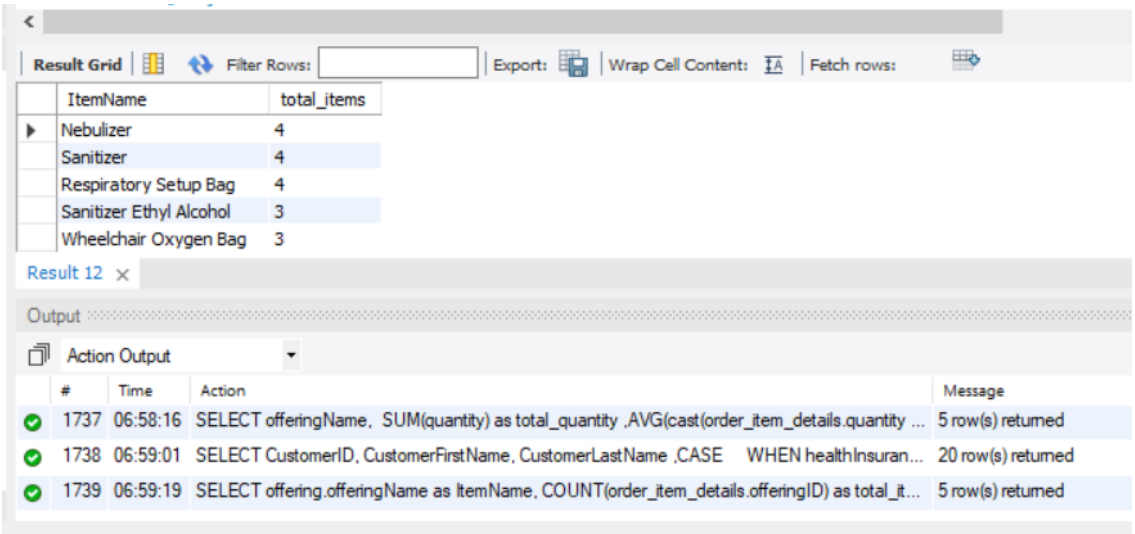
### Functionalities Used: CASE WHEN

**3 | What are the top 5 most popular items that were purchased considering only orders during midnight hours** *12am-4am* **and items only ordered from** *Branchs Ethicon Inc, Chicago* **and** *Dentsply Sirona Inc, New York*

**Query Used:**

```
SELECT offering.offeringName as ItemName,
        COUNT(order_item_details.offeringID) as total_items
 FROM
orders,order_item_details,supplier_branch,offering
WHERE orders.orderID = order_item_details.orderID
AND order_item_details.supplierBranchID = supplier_branch.supplierBranchID
AND order_item_details.offeringID = offering.offeringID
AND offering.offeringType = 'Item'
AND supplier_branch.branchName IN ('Ethicon Inc, Chicago','Dentsply Sirona Inc, New York')
AND CAST(date_format(orders.orderdate, '%H') AS SIGNED) BETWEEN 0 AND 4
GROUP BY 1 ORDER BY 2 DESC LIMIT 5;
```

**Results:**




**Functionalities Used:**

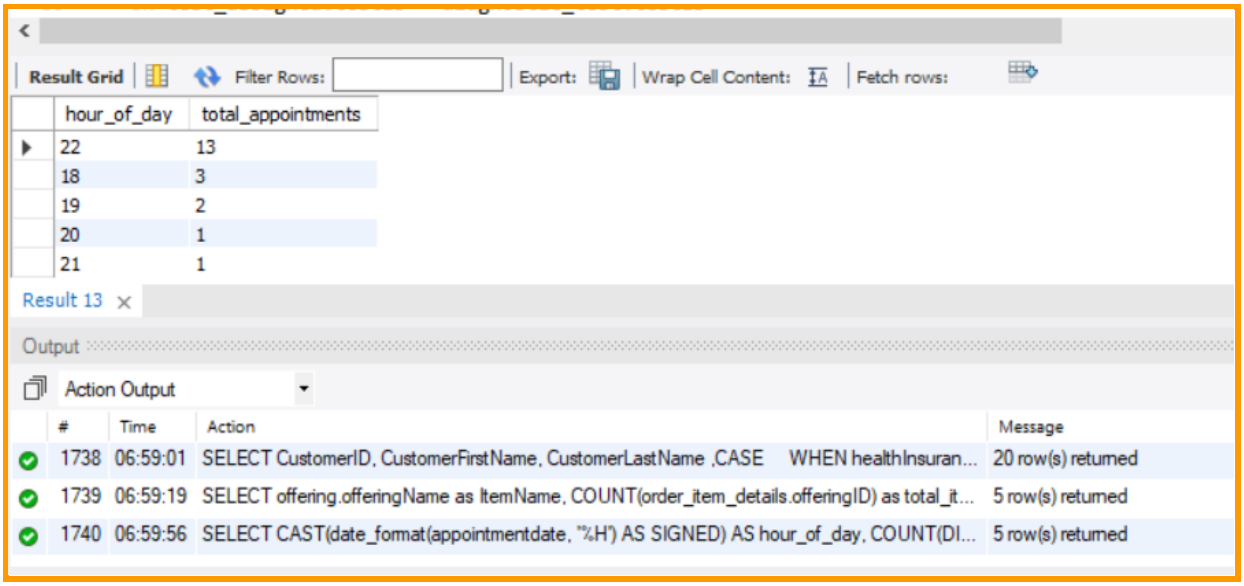
**BETWEEN, IN, EQUI JOIN**

**4 | What are the top 5 hours of the day that healthcare doctors (who have been appointed to HealthCenters in NY) busy with appointments?**

**Query Used:**

```
SELECT CAST(date_format(appointmentdate, '%H') AS SIGNED) AS hour_of_day,
COUNT(DISTINCT appointmentID) AS total_appointments
FROM appointment NATURAL JOIN healthcare_provider
WHERE isHealthCareDoctor = 1 AND doctorID IN (SELECT doctorID FROM healthcare_doctor NATURAL JOIN healthcare_center WHERE healthcarecentername LIKE '%NY')
GROUP BY 1 ORDER BY 2 DESC LIMIT 5;
```

**Result:**




**Functionalities Used: NATURAL JOIN, SUB QUERY, BOOLEAN Filter, REG EXP Filter**

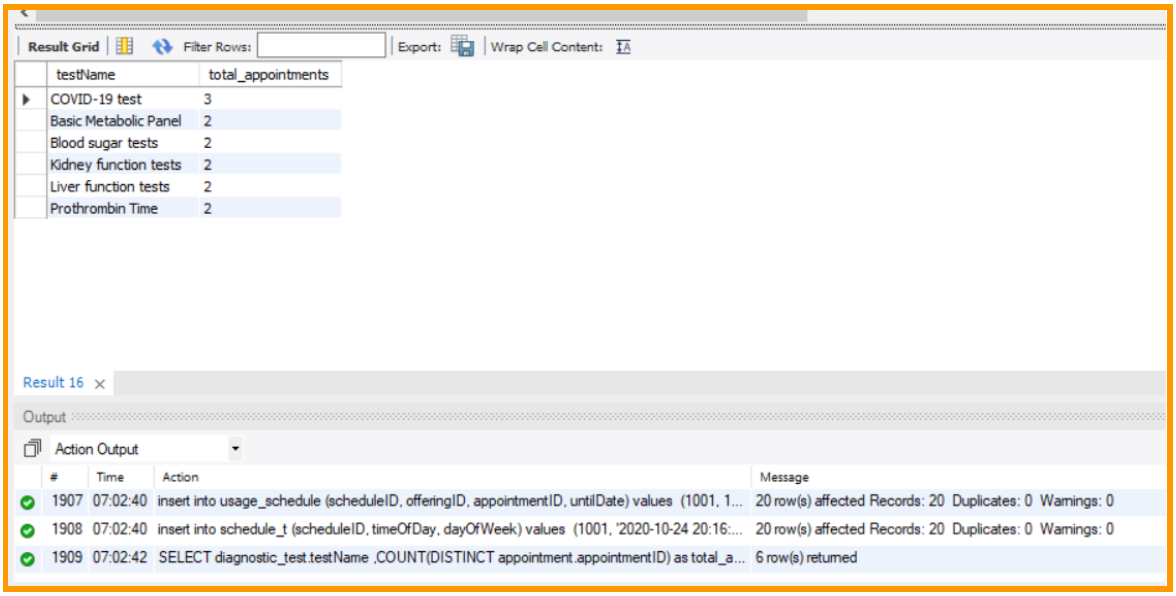
## COVID-19 Relief Fund

**5 | List the tests having more than one customer taking them. Present the total appointments of these tests.**

**Query Used:**

```
SELECT diagnostic_test.testName
,COUNT(DISTINCT appointment.appointmentID) as total_appointments
FROM test_assigned
INNER JOIN diagnostic_test
ON test_assigned.testID = diagnostic_test.testID
INNER JOIN appointment
ON test_assigned.appointmentID = appointment.appointmentID
GROUP BY 1 HAVING COUNT(DISTINCT appointment.customerID) > 1
ORDER BY 2 DESC;
```

**Result:**



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| testName | total_appointments |
|---|---|
| COVID-19 test | 3 |
| Basic Metabolic Panel | 2 |
| Blood sugar tests | 2 |
| Kidney function tests | 2 |
| Liver function tests | 2 |
| Prothrombin Time | 2 |

Result 16

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 1907 | 07:02:40 | insert into usage_schedule (scheduleID, offeringID, appointmentID, untilDate) values (1001, 1... | 20 row(s) affected Records: 20 Duplicates: 0 Warnings: 0 |
| 1908 | 07:02:40 | insert into schedule_t (scheduleID, timeOfDay, dayOfWeek) values (1001, '2020-10-24 20:16:... | 20 row(s) affected Records: 20 Duplicates: 0 Warnings: 0 |
| 1909 | 07:02:42 | SELECT diagnostic_test.testName ,COUNT(DISTINCT appointment.appointmentID) as total_a... | 6 row(s) returned |

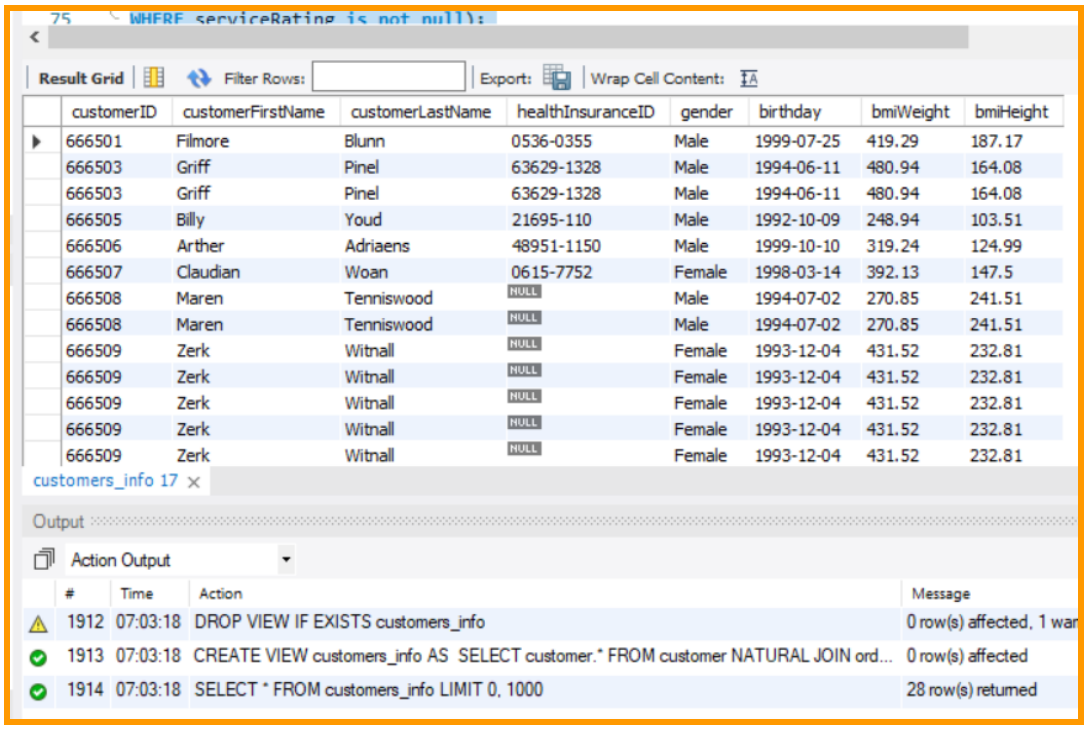
**Functionalities Used: GROUP BY, HAVING, INNER JOIN**

**View 1 | Dynamic view of all customers ordering items(only item type = 'Respiratory') and services that are rated.**

**Query Used:**

```
CREATE VIEW customers_info AS
SELECT customer.* FROM customer NATURAL JOIN orders NATURAL JOIN order_item_details WHERE
offeringID IN
(SELECT tOfferingID FROM item
        WHERE itemtype = 'Respiratory'
UNION
SELECT sOfferingID FROM service
        WHERE serviceRating is not null);
SELECT * FROM customers_info;
```

**Results:**



75 WHERE serviceRating is not null);

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| customerID | customerFirstName | customerLastName | healthInsuranceID | gender | birthday | bmiWeight | bmiHeight |
|---|---|---|---|---|---|---|---|
| 666501 | Filmore | Blunn | 0536-0355 | Male | 1999-07-25 | 419.29 | 187.17 |
| 666503 | Griff | Pinel | 63629-1328 | Male | 1994-06-11 | 480.94 | 164.08 |
| 666503 | Griff | Pinel | 63629-1328 | Male | 1994-06-11 | 480.94 | 164.08 |
| 666505 | Billy | Youd | 21695-110 | Male | 1992-10-09 | 248.94 | 103.51 |
| 666506 | Arther | Adriaens | 48951-1150 | Male | 1999-10-10 | 319.24 | 124.99 |
| 666507 | Claudian | Woan | 0615-7752 | Female | 1998-03-14 | 392.13 | 147.5 |
| 666508 | Maren | Tenniswood | NULL | Male | 1994-07-02 | 270.85 | 241.51 |
| 666508 | Maren | Tenniswood | NULL | Male | 1994-07-02 | 270.85 | 241.51 |
| 666509 | Zerk | Witnall | NULL | Female | 1993-12-04 | 431.52 | 232.81 |
| 666509 | Zerk | Witnall | NULL | Female | 1993-12-04 | 431.52 | 232.81 |
| 666509 | Zerk | Witnall | NULL | Female | 1993-12-04 | 431.52 | 232.81 |
| 666509 | Zerk | Witnall | NULL | Female | 1993-12-04 | 431.52 | 232.81 |
| 666509 | Zerk | Witnall | NULL | Female | 1993-12-04 | 431.52 | 232.81 |

customers_info 17

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 1912 | 07:03:18 | DROP VIEW IF EXISTS customers_info | 0 row(s) affected, 1 war |
| 1913 | 07:03:18 | CREATE VIEW customers_info AS SELECT customer.* FROM customer NATURAL JOIN ord... | 0 row(s) affected |
| 1914 | 07:03:18 | SELECT * FROM customers_info LIMIT 0, 1000 | 28 row(s) returned |

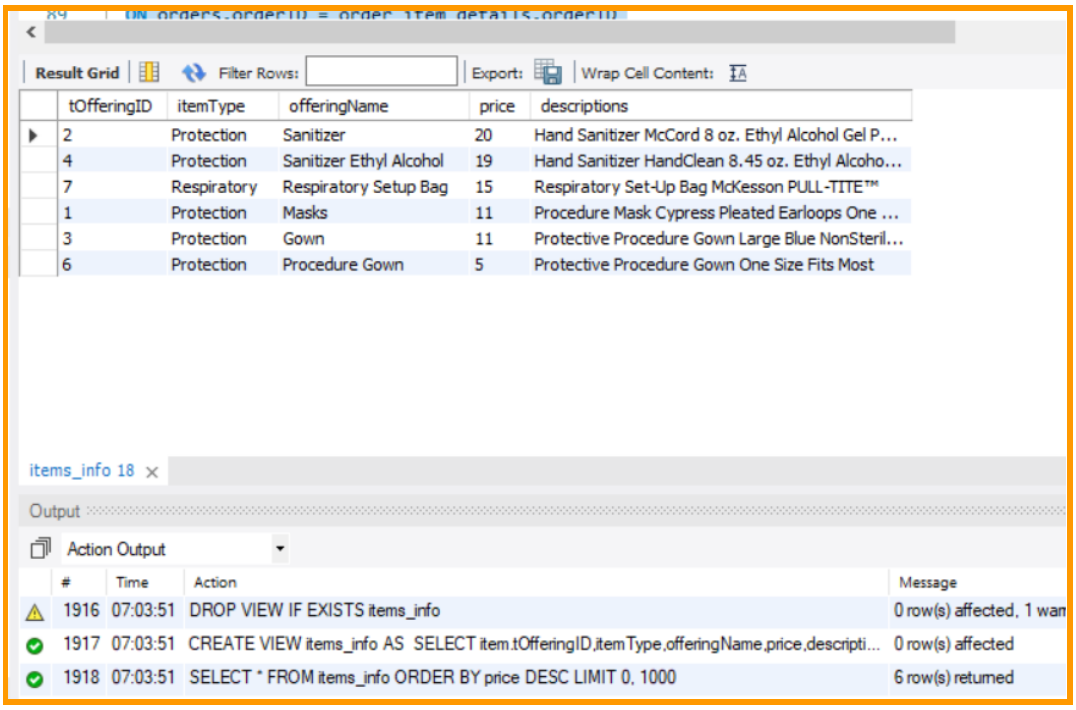
**Functionalities Used: UNION , OPERATORS = ,IS NOT NULL, NATURAL JOIN, IN**

## View 2 | Dynamic view of all items that have total quantity ata least 100 and only ordered from East Bay (area consisting of the word 'East Bay')

### Query Used:

```
DROP VIEW IF EXISTS items_info;
CREATE VIEW items_info AS
SELECT item.tOfferingID,itemType,offeringName,price,descriptions FROM item INNER JOIN offering ON
item.tOfferingID = offering.offeringID
WHERE
EXISTS
(SELECT order_item_details.offeringID
FROM order_item_details
INNER JOIN orders
ON orders.orderID = order_item_details.orderID
WHERE deliveryArea LIKE '%EastBay%'
AND item.tOfferingID = order_item_details.offeringID
GROUP BY 1 HAVING SUM(quantity) >= 100);
SELECT * FROM items_info ORDER BY price DESC;
```

### Results:



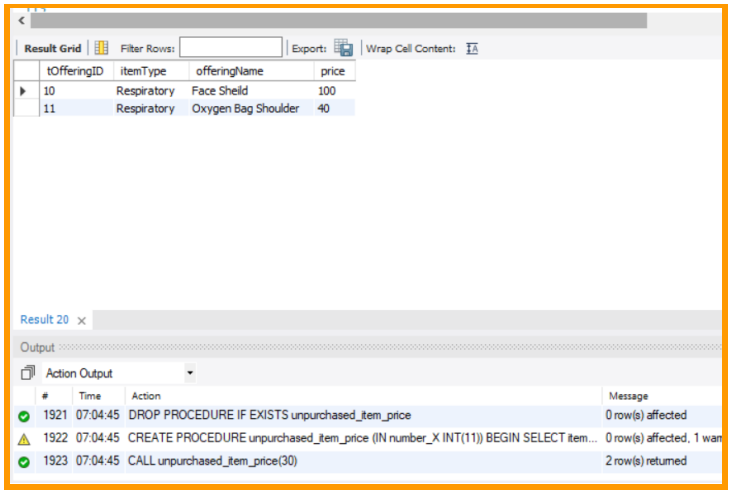

### Functionalities Used: EXISTS , GROUP BY, HAVING, LIKE REG EXP '%%'

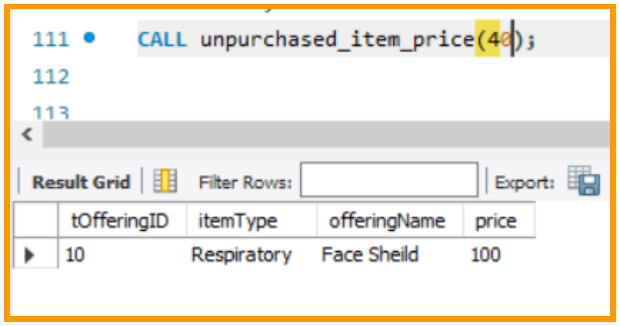
**Procedure 1 | List all items that have never been ordered and have the price more than X.**

**Query Used:**

```
DROP PROCEDURE IF EXISTS unpurchased_item_price;
DELIMITER //
CREATE PROCEDURE unpurchased_item_price (IN number_X INT(11))
BEGIN
SELECT item.tOfferingID,itemType,offering.offeringName,offering.price
FROM item INNER JOIN offering ON item.tOfferingID = offering.offeringID
WHERE tOfferingID
NOT IN (SELECT DISTINCT offeringID FROM order_item_details)
AND offering.price > number_X;
END //
DELIMITER ;
CALL unpurchased_item_price(30);
```

**Result:**



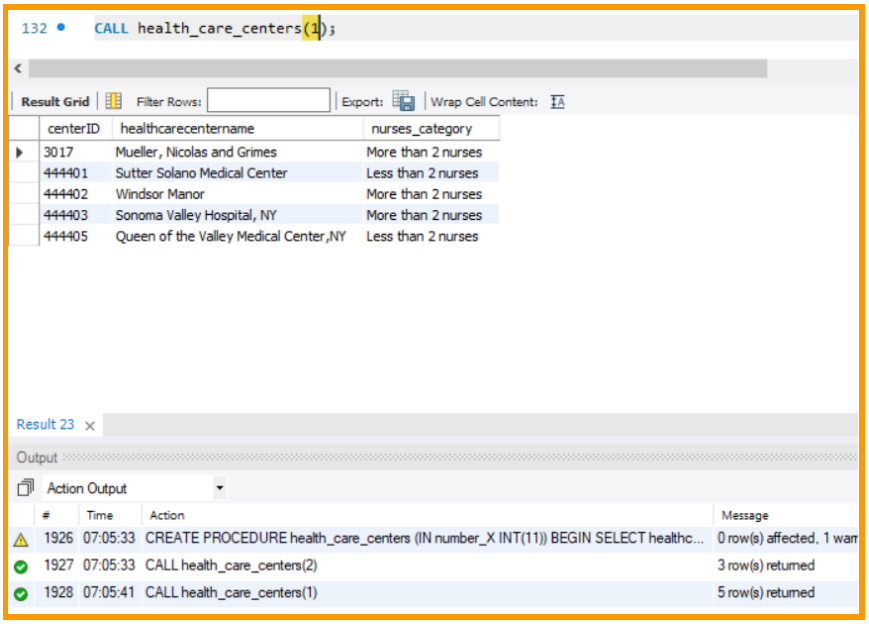

**Functionalities Used: GROUP BY, WHERE, NOT IN**

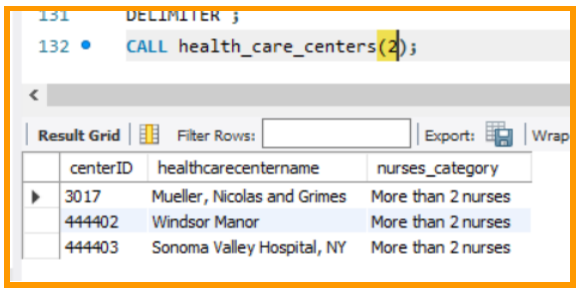
**Procedure 2 | List all the healthcare centers that have more than X nurse. Distinguish health care centers with more than 10 nurses vs those with less than 10 nurses.**

**Query Used:**

```
DROP PROCEDURE IF EXISTS health_care_centers;
DELIMITER //
CREATE PROCEDURE health_care_centers (IN number_X INT(11))
BEGIN
SELECT healthcare_center.centerID,healthcarecentername,
CASE WHEN COUNT(DISTINCT nurseID) > 2 THEN 'More than 2 nurses' ELSE 'Less than 2 nurses' END AS
nurses_category
FROM healthcare_center NATURAL JOIN nurse
GROUP BY 1,2 HAVING COUNT(DISTINCT nurseID) > number_X ;
END //
DELIMITER ;
CALL health_care_centers(1);
```

**Results:**




**Functionalities Used: NATURAL JOIN, HAVING, CASE WHEN**

# 6 | Appendix

1. Link to ER Diagram on LucidChart
2. Relational Data Model
    a. Link to Relational Data Model - Part 1
    b. Link to Relational Data Model - Part 2
3. SQL
    a. Link to SQL Table Structure DDL
    b. Link to SQL Table Insertions
    c. Link to SQL Table Queries
4. **METADATA Below**

## Metadata

**APPOINTMENT**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **AppointmentID** | Int | 11 | Yes | No | Primary |
| DoctorID | Int | 11 | No | No | Foreign |
| CustomerID | Int | 11 | No | No | Foreign |
| AppointmentDate | DateTime | | | No | |
| AppointmentType | Varchar | 50 | | Yes | |
| ReqDocType | Varchar | 50 | | Yes | |
| ReasonDesc | Varchar | 50 | | No | |
| AppointReview | Varchar | 300 | | Yes | |
| ResultStatus | Varchar | 50 | | No | |

| | | | | | |
|---|---|---|---|---|---|
| AppointmentReturnDate | Date | | | Yes | |

**DOCTOR_SPECIALIZATION**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **DoctorID** | Int | 11 | No | No | Part of Primary, Foreign |
| **DoctorSpecialization** | Varchar | 100 | No | No | Part of Primary |

**HEALTHCARE_PROVIDER**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **DoctorID** | Int | 11 | Yes | No | Primary |
| DocFullName | Varchar | 100 | | No | |
| PhoneNo | varchar | 10 | | No | |
| IsHealthCareDoctor | Boolean | | | No | |
| IsIndependent | Boolean | | | No | |

**SCHEDULE**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **ScheduleID** | Int | 11 | No | No | Part of Primary,Foreign |
| **TimeofDay** | Timestamp | | No | No | Part of Primary |
| **DayOfWeek** | Int | 11 | No | No | Part of Primary |

**USAGE_SCHEDULE**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **ScheduleID** | Int | 11 | Yes | No | Primary |
| OfferingID | Int | 11 | No | No | Foreign |
| AppointmentID | Int | 11 | No | No | Foreign |

| | | | | | |
|---|---|---|---|---|---|
| UntilDate | Date | | No | No | |

**INDEPENDENT_DOCTOR**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **IDoctorID** | Int | 11 | No | No | Primary, Foreign |
| HourlyFee | Int | 11 | No | No | |
| DocAddress | Varchar | 100 | No | Yes | |

**HEALTHCARE_DOCTOR**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **HealthCareEmployeeID** | Int | 11 | No | No | Primary, Foreign |
| ManagerEmpID | Int | 11 | No | No | Foreign |
| CenterID | Int | 11 | No | No | Foreign |

**TEST_ASSIGNED**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **AssignedTestID** | Int | 11 | Yes | No | Primary |
| NurseID | Int | 11 | No | No | Foreign |
| TestID | Int | 11 | No | No | Foreign |
| AppointmentID | Int | 11 | No | Yes | Foreign |
| DoctorID | Int | 11 | No | No | Foreign |
| CustomerID | Int | 11 | No | No | Foreign |
| ExpectedDate | Date | | No | Yes | |
| TakenDate | Date | | No | No | |

| | | | | | |
|---|---|---|---|---|---|
| TestResultDes | Varchar | 100 | No | Yes | |
| TestRating | Varchar | 100 | No | Yes | |

**HEALTHCARE_CENTER**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **CenterID** | Int | 11 | Yes | No | Primary |
| HealthCareCenterName | Varchar | 50 | No | No | |
| PartnershipStartDate | Date | | No | Yes | |

**DIAGNOSTIC_TEST**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **TestID** | Int | 11 | Yes | No | Primary |
| TestName | Varchar | 100 | No | No | |
| TestDesc | Varchar | 100 | No | No | |
| ApproximateTime | Timestamp | | No | Yes | |
| TestPrice | Int | 11 | No | No | |
| TestType | Varchar | 100 | No | No | |

**NURSE**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **NurseID** | Int | 11 | Yes | No | Primary |
| NurseName | Varchar | 100 | No | No | |
| NurseSpecialization | Varchar | 100 | No | Yes | |
| WorkStatus | Varchar | 100 | No | No | |
| Salary | Int | 11 | No | No | |

| | | | | | |
|---|---|---|---|---|---|
| CenterID | Int | 11 | No | No | Foreign |

**EMERGENCY_CONTACT**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **CustomerID** | Int | 11 | No | No | Part of Primary |
| **EFirstName** | Varchar | 100 | No | No | Part of Primary |
| **ELastName** | Varchar | 100 | No | No | Part of Primary |
| EDateofBirth | Date | | No | Yes | |
| EphoneNo | varchar | 11 | No | No | |
| EAddress | Varchar | 100 | No | Yes | |

**CUSTOMER**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **CustomerID** | Int | 11 | Yes | No | Primary |
| CustomerFirstName | Varchar | 100 | No | No | |
| CustomerLastName | Varchar | 100 | No | No | |
| HealthInsuranceID | Varchar | 100 | No | Yes | |
| Gender | Varchar | 11 | No | Yes | |
| Birthday | Date | | No | Yes | |
| BMIWeight | Float | | No | No | |
| BMIHeight | Float | | No | No | |

**ORDER**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **OrderID** | Int | 11 | Yes | No | Primary |
| CustomerID | Int | 11 | No | No | Foreign |
| DeliveryCity | Varchar | 100 | No | No | |

| | | | | | |
|---|---|---|---|---|---|
| DeliveryArea | Varchar | 100 | No | No | |
| DeliveryState | Varchar | 2 | No | No | |
| DeliveryZIP | Int | 5 | No | Yes | |
| DeliveryStreet | Varchar | 100 | No | Yes | |
| OrderDate | Datetime | | No | No | |
| DeliveryPhoneNo | varchar | 10 | No | No | |
| DeliveryPrice | Float | | No | No | |

**ORDER_ITEM_DETAILS**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **OrderID** | Int | 11 | No | No | Part of Primary |
| **OfferingID** | Int | 11 | No | No | Part of Primary |
| **SupplierBranchID** | Int | 11 | No | No | Part of Primary |
| Quantity | Int | 11 | No | No | |
| ScheduledDeliveryDate | Date | | No | Yes | |
| ActualDeliveryDate | Date | | No | No | |

**SUPPLIER_BRAND**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **SupplierID** | Int | 11 | Yes | No | Primary |
| BrandName | Varchar | 100 | No | No | |
| BrandDes | Varchar | 100 | No | No | |
| SupplierType | Varchar | 100 | No | No | |

**OFFERING**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **OfferingID** | Int | 11 | Yes | No | Primary |
| OfferingName | Varchar | 100 | No | No | |
| CreatedDate | Date | | No | Yes | |
| Price | Decimal | 11 | No | No | |
| Description | Varchar | 100 | No | No | |
| OfferingType | Varchar | 100 | No | No | |

**SUPPLIER_BRANCH**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **SupplierBranchID** | Int | 11 | Yes | No | Primary |
| SupplierID | Int | 11 | No | No | Foreign |
| BranchName | Varchar | 100 | No | No | |
| BranchLocation | Varchar | 100 | No | No | |
| BranchActivationDate | Date | | No | No | |
| BranchClosingDate | Date | | No | Yes | |
| IsOnline | Boolean | | No | No | |

**ITEM**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|
| **TOfferingID** | Int | 11 | No | No | Primary, Foreign |
| Quantity | Int | 11 | No | No | |
| ItemType | Varchar | 100 | No | No | |
| UnitofMeasure | Varchar | 100 | No | No | |

**SERVICE**

| Attribute | Data Type | Size | AUTO | Null | Constraint |
|---|---|---|---|---|---|

| **SOfferingID** | Int | 11 | No | No | Primary, Foreign |
|---|---|---|---|---|---|
| TimeTaken | Timestamp | | No | No | |
| GivenTime | Timestamp | | No | Yes | |
| ServiceRating | Varchar | 100 | No | Yes | |
| ScheduleType | Varchar | 100 | No | Yes | |
| Duration | Decimal | 11 | No | No | |