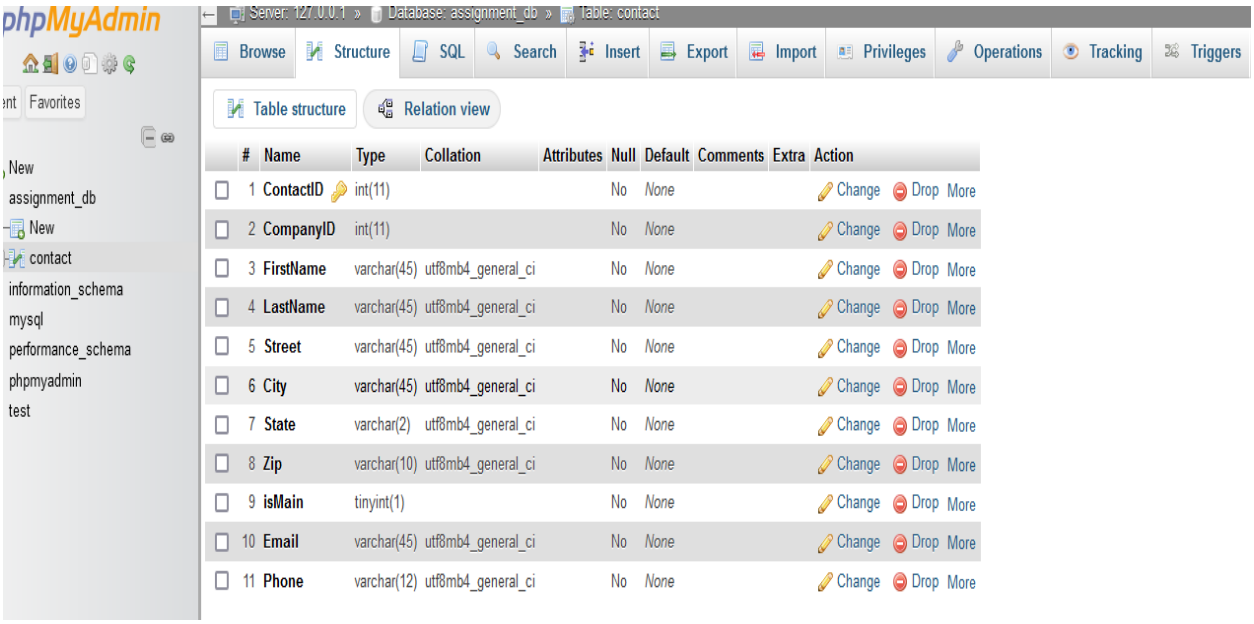
Megh Bhatt
TOPS SG Highway

Assignment – SQL

1) Statement to create the Contact table.

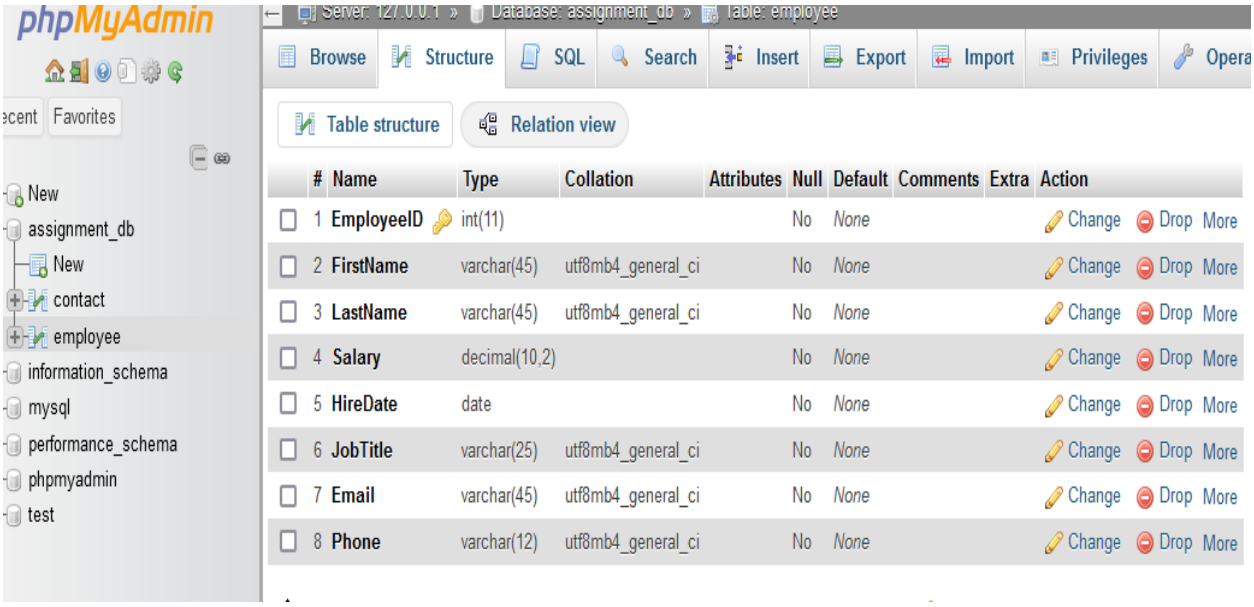
A- CREATE TABLE `assignment_db`.
`contact` (`ContactID` INT NOT NULL , `CompanyID` INT NOT NULL ,
`FirstName` VARCHAR(45) NOT NULL ,
`LastName` VARCHAR(45) NOT NULL ,
`Street` VARCHAR(45) NOT NULL ,
`City` VARCHAR(45) NOT NULL ,
`State` VARCHAR(2) NOT NULL ,
`Zip` VARCHAR(10) NOT NULL ,
`isMain` BOOLEAN NOT NULL ,
`Email` VARCHAR(45) NOT NULL ,
`Phone` VARCHAR(12) NOT NULL ,
PRIMARY KEY (`ContactID`))



Megh Bhatt
TOPS SG Highway

2) Statement to create the Employee table.

A- CREATE TABLE `assignment_db`.
`employee` (`EmployeeID` INT NOT NULL , `FirstName` VARCHAR(45) NOT NULL ,
`LastName` VARCHAR(45) NOT NULL , `
Salary` DECIMAL(10,2) NOT NULL ,
`HireDate` DATE NOT NULL ,
`JobTitle` VARCHAR(25) NOT NULL ,
`Email` VARCHAR(45) NOT NULL , `
Phone` VARCHAR(12) NOT NULL ,
PRIMARY KEY (`EmployeeID`))



3) Statement to create the ContactEmployee table.
(HINT: Use DATE as the datatype for ContactDate. It allows you to store the
date in this format: YYYY-MM-DD (i.e., '2014-03-12' for March 12, 2014).

A- CREATE TABLE `assignment_db`.`contactemployee` (`ContactEmployeeID` INT NOT NULL ,
`ContactID` INT NOT NULL ,
`EmployeeID` INT NOT NULL ,
`ContactDate` DATE NOT NULL ,
`Description` VARCHAR(100) NOT NULL)

4) In the Employee table, the statement that changes Lesley Bland's phone number to 215-555-8800.

A-

Before Updating:

	EmployeeID	FirstName	LastName	Salary	HireDate	JobTitle	Email	Phone
Edit Copy Delete	3	Lesley	Bland	50000.00	2022-05-30	Data Analyst	lesley.bland@example.com	992-576-2316

After Updating:

```
UPDATE Employee
SET phone_number = '215-555-8800'
WHERE first_name = 'Lesley' AND last_name = 'Bland';
```

	EmployeeID	FirstName	LastName	Salary	HireDate	JobTitle	Email	Phone
Edit Copy Delete	3	Lesley	Bland	50000.00	2022-05-30	Data Analyst	lesley.bland@example.com	215-555-8800

5) In the Company table, the statement that changes the name of “Urban Outfitters, Inc.” to “Urban Outfitters”.

A- **Before Updating:**

CompanyID	CompanyName	Street	City	State	Zip
8	Urban Outfitters, Inc.	C V Raman Nagar	Bengaluru	MA	02101

After Updating:

```
UPDATE Company
SET CompanyName = 'Urban Outfitters'
WHERE CompanyName = 'Urban Outfitters, Inc.'
```

CompanyID	CompanyName	Street	City	State	Zip
8	Urban Outfitters	C V Raman Nagar	Bengaluru	MA	02101

6) In ContactEmployee table, the statement that removes Dianne Connor's contact event with Jack Lee (one statement). (HINT: Use the primary key of the ContactEmployee table to specify the correct record to remove.)

A-
```
DELETE FROM ContactEmployee
WHERE EmployeeName = 'Dianne Connor'
AND ContactName = 'Jack Lee';
```

7) Write the SQL SELECT query that displays the names of the employees that have contacted Toll Brothers (one statement). Run the SQL SELECT query in MySQL Workbench. Copy the results below as well.

A-
```
SELECT EmployeeName
FROM ContactEmployee
WHERE ContactName = 'Toll Brothers';
```

Sample Input : -

id	EmployeeName	ContactName
1	John Smith	Toll Brothers
2	Dianne Connor	Toll Brothers
3	Mark Thompson	Acme Corp
4	Sarah Johnson	Toll Brothers

Sample Output : -

EmployeeName
John Smith
Dianne Connor
Sarah Johnson

8) What is the significance of “%” and “_” operators in the LIKE statement?

A- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.
There are two wildcards often used in conjunction with the LIKE operator:

- The percent sign (%) represents zero, one, or multiple characters.

- The underscore sign (_) represents one, single character.

9) Explain normalization in the context of databases.

A- *Normalization* is the process of organizing a database to reduce redundancy and improve data integrity.

Normalization also simplifies the database design so that it achieves the optimal structure composed of atomic elements (i.e. elements that cannot be broken down into smaller parts).

Also referred to as *database normalization* or *data normalization*, normalization is an important part of relational database design, as it helps with the speed, accuracy, and efficiency of the database.

By normalizing a database, you arrange the data into tables and columns. You ensure that each table contains only related data. If data is not directly related, you create a new table for that data.

For example:

if you have a "Customers" table, you'd normally create a separate table for the products they can order (you could call this table "Products"). You'd create another table for customers' orders (perhaps called "Orders"). And if each order could contain multiple items, you'd typically create yet another table to store each order item (perhaps called "OrderItems"). All these tables would be linked by their primary key, which allows you to find related data across all these tables (such as all orders by a given customer).

10) What does a join in MySQL mean?

A- A join is a method of linking data between one (self-join) or more tables based on the values of the common column between the tables.

MySQL supports the following types of joins:

- Inner join
- Left join
- Right join
- Cross join

To join tables, you use the cross join, inner join, left join, or right join clause. The join clause is used in the SELECT statement appeared after the FROM clause.

(Note: MySQL hasn't supported the FULL OUTER JOIN yet.)

11) What do you understand about DDL, DCL, and DML in MySQL?

A- SQL can do many different things: create database tables, insert or change records, add indexes, retrieve information, and so on. So, it can be useful to divide SQL into several **sublanguages**; this helps us wrap our heads around all the different operations that can be performed on an SQL database. These sublanguages are:

- **Data Query Language (DQL) -** The Data Query Language is the sublanguage responsible for reading, or querying, data from a database. In SQL, this corresponds to the SELECT

- **Data Manipulation Language (DML)** - The Data Manipulation Language is the sublanguage responsible for adding, editing or deleting data from a database. In SQL, this corresponds to the INSERT, UPDATE, and DELETE

- **Data Definition Language (DDL)** - The Data Definition Language is the sublanguage responsible for defining the way data is structured in a database. In SQL, this corresponds to manipulating tables through the CREATE TABLE, ALTER TABLE, and DROP TABLE

- **Data Control Language (DCL)** - The Data Control Language is the sublanguage responsible for the administrative tasks of controlling the database itself, most notably granting and revoking database permissions for users. In SQL, this corresponds to the GRANT, REVOKE, and DENY commands, among others.

12) What is the role of the MySQL JOIN clause in a query, and what are some common types of joins?

A- A join is a method of linking data between one (self-join) or more tables based on the values of the common column between the tables.

MySQL supports the following types of joins:

- Inner join
- Left join
- Right join
- Cross join

To join tables, you use the cross join, inner join, left join, or right join clause. The join clause is used in the SELECT statement appeared after the FROM clause.

(Note: MySQL hasn't supported the FULL OUTER JOIN yet.)