-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema MI

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema MI

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `MI` DEFAULT CHARACTER SET utf8 COLLATE utf8\_bin ;

USE `MI` ;

-- -----------------------------------------------------

-- Table `MI`.`Company`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `MI`.`Company` (

`CompanyID` INT NOT NULL AUTO\_INCREMENT,

`CompanyName` VARCHAR(45) NULL,

`Street` VARCHAR(45) NULL,

`city` VARCHAR(45) NULL,

`state` VARCHAR(45) NULL,

`zip` VARCHAR(2) NULL,

`Companycol` VARCHAR(10) NULL,

PRIMARY KEY (`CompanyID`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `MI`.`Contact`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `MI`.`Contact` (

`ContactID` INT NOT NULL AUTO\_INCREMENT,

`FirstName` VARCHAR(45) NULL,

`Company\_CompanyID` INT NOT NULL,

`LastName` VARCHAR(45) NULL,

`Street` VARCHAR(45) NULL,

`City` VARCHAR(45) NULL,

`State` VARCHAR(45) NULL,

`Zip` VARCHAR(45) NULL,

`IsMain` TINYINT NULL,

`Email` VARCHAR(45) NULL,

`Phone` VARCHAR(45) NULL,

PRIMARY KEY (`ContactID`),

INDEX `fk\_Contact\_Company\_idx` (`Company\_CompanyID` ASC) VISIBLE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `MI`.`ContactEmployee`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `MI`.`ContactEmployee` (

`Contact\_ContactID` INT NOT NULL,

`ContactEmployeeID` INT NULL,

`EmployeeID` INT NOT NULL AUTO\_INCREMENT,

`ContactDate` DATE NULL,

`Description` VARCHAR(100) NULL,

INDEX `fk\_table1\_Contact1\_idx` (`Contact\_ContactID` ASC) VISIBLE,

PRIMARY KEY (`EmployeeID`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `MI`.`Employee`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `MI`.`Employee` (

`ContactEmployee\_EmployeeID` INT primary key NOT NULL,

`FirstName` VARCHAR(45) NULL,

`LastName` VARCHAR(45) NULL,

`Salary` DECIMAL(10,2) NULL,

`JobTitle` VARCHAR(45) NULL,

`Email` VARCHAR(45) NULL,

`Phone` VARCHAR(12) NULL,

INDEX `fk\_table1\_ContactEmployee1\_idx` (`ContactEmployee\_EmployeeID` ASC) VISIBLE)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

INSERT INTO `MI`.`Company` (`CompanyName`, `Street`, `city`, `state`, `zip`)

VALUES

('Urban Outfitters, Inc.', '123 Main St', 'Philadelphia', 'PA', '07'),

('Toll Brothers', '456 Toll Dr', 'Huntingdon Valley', 'PA', '06'),

('Comcast Corporation', '1 Comcast Ctr', 'Philadelphia', 'PA', '03');

INSERT INTO `MI`.`Contact` (`FirstName`, `LastName`, `Street`, `City`, `State`, `Zip`, `IsMain`, `Email`, `Phone`, `Company\_CompanyID`)

VALUES

('Lesley', 'Bland', '123 Main St', 'Philadelphia', 'PA', '19107', 1, 'lesley.bland@urbanoutfitters.com', '215-555-1234', 1),

('Dianne', 'Connor', '456 Toll Dr', 'Huntingdon Valley', 'PA', '19006', 1, 'dianne.connor@tollbrothers.com', '215-555-5678', 2),

('John', 'Doe', '1 Comcast Ctr', 'Philadelphia', 'PA', '19103', 1, 'john.doe@comcast.com', '215-555-9012', 3),

('Jack', 'Lee', '123 Main St', 'Philadelphia', 'PA', '19107', 0, 'jack.lee@urbanoutfitters.com', '215-555-1111', 1);

INSERT INTO `MI`.`ContactEmployee` (`Contact\_ContactID`, `ContactEmployeeID`, `EmployeeID`, `ContactDate`, `Description`)

VALUES

(1, 1, 1, '2020-01-01', 'Initial Meeting'),

(2, 2, 2, '2020-02-01', 'Follow-up Meeting'),

(3, 3, 3, '2020-03-01', 'Final Meeting'),

(1, 4, 4, '2020-04-01', 'Additional Meeting'),

(4, 5, 5, '2020-05-01', 'New Contact');

INSERT INTO `MI`.`Employee` (`ContactEmployee\_EmployeeID`, `FirstName`, `LastName`, `Salary`, `JobTitle`, `Email`, `Phone`)

VALUES

(1, 'Jane', 'Smith', 50000.00, 'Sales Manager', 'jane.smith@urbanoutfitters.com', '215-555-2222'),

(2, 'Bob', 'Johnson', 60000.00, 'Marketing Manager', 'bob.johnson@tollbrothers.com', '215-555-3333'),

(3, 'Alice', 'Williams', 70000.00, 'IT Manager', 'alice.williams@comcast.com', '215-555-4444'),

(4, 'Jack', 'Lee', 40000.00, 'Sales Representative', 'jack.lee@urbanoutfitters.com', '215-555-1111'),

(5, 'Lesley', 'Bland', 45000.00, 'HR Manager', 'lesley.bland@urbanoutfitters.com', '215-555-1234');

select \* from company ;

**1) Statement to create the Contact table**

select \* from contact ;

**3) Statement to create the ContactEmployee table**

select \* from contactemployee ;

**2) Statement to create the Employee table**

select \* from employee ;

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

UPDATE Employee SET Phone = '215-555-8800' WHERE FirstName = 'Lesley';

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

UPDATE Company SET CompanyName = 'Urban Outfitters'

WHERE CompanyName = 'Urban Outfitters, Inc.';

**6) In ContactEmployee table, the statement that removes Dianne Connor’s contact event with Jack Lee (one statement).**

DELETE FROM ContactEmployee WHERE Contact\_ContactID = 2 AND EmployeeID = 4;

**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.**

SELECT e.FirstName, e.LastName

FROM Employee e

JOIN ContactEmployee ce ON e.ContactEmployee\_EmployeeID = ce.EmployeeID

JOIN Contact c ON ce.Contact\_ContactID = c.ContactID

JOIN Company co ON c.Company\_CompanyID = co.CompanyID

WHERE co.CompanyName = 'Toll' ;

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

In the SQL the LIKE statement where “%” or “\_” will be use.

* “%” is for matching the name alphabet. For example, “A%” means where A is first character that is include in result.
* “\_” is use for given the space location. For example, “\_A” means that where A is a second character result will be print.

**9) Explain normalization in the context of databases.**

It involves dividing a database into two or more tables and defining relationships between the tables.

* **First Normal Form (1NF)**: Ensures that the table has a primary key and each column contains indivisible values.
* **Second Normal Form (2NF)**: Achieved when a table is in 1NF and all non-key columns are fully dependent on the primary key.
* **Third Normal Form (3NF)**: Achieved when a table is in 2NF and all the columns are only dependent on the primary key, not on other non-key columns.

**10) What does a join in MySQL mean?**

Join method is basically use foe the connect the table using Inner join, Left join and, Right join.

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

DDL: Data Definition Language mainly use for the commend the table like Create, Alter, Drop.

DCL: Data Control Language this use to give control all over the table commend like Grant and, Revoke.

DML: Data Manipulation Language use for the manipulate the table data using commend like Update, Delete, Select and, Insert.

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

The JOIN clause in MySQL is used to combine rows from two or more tables based on a related column between them.

**INNER JOIN**: Returns records that have matching values in both tables.

* Syntax: SELECT columns FROM table1 INNER JOIN table2 ON table1.column = table2.column;

**LEFT JOIN:** Returns all records from the left table, and the matched records from the right table. If no match is found, NULL values are returned for columns from the right table.

* Syntax: SELECT columns FROM table1 LEFT JOIN table2 ON table1.column = table2.column;

**RIGHT JOIN**: Returns all records from the right table, and the matched records from the left table. If no match is found, NULL values are returned for columns from the left table.

* Syntax: SELECT columns FROM table1 RIGHT JOIN table2 ON table1.column = table2.column;

**CROSS JOIN**: Returns the Cartesian product of the two tables, i.e., each row from the first table is combined with all rows from the second table.

* Syntax: SELECT columns FROM table1 CROSS JOIN table2;