Homework #2 **Due: turned in by Mon 01/27/2020 before class**

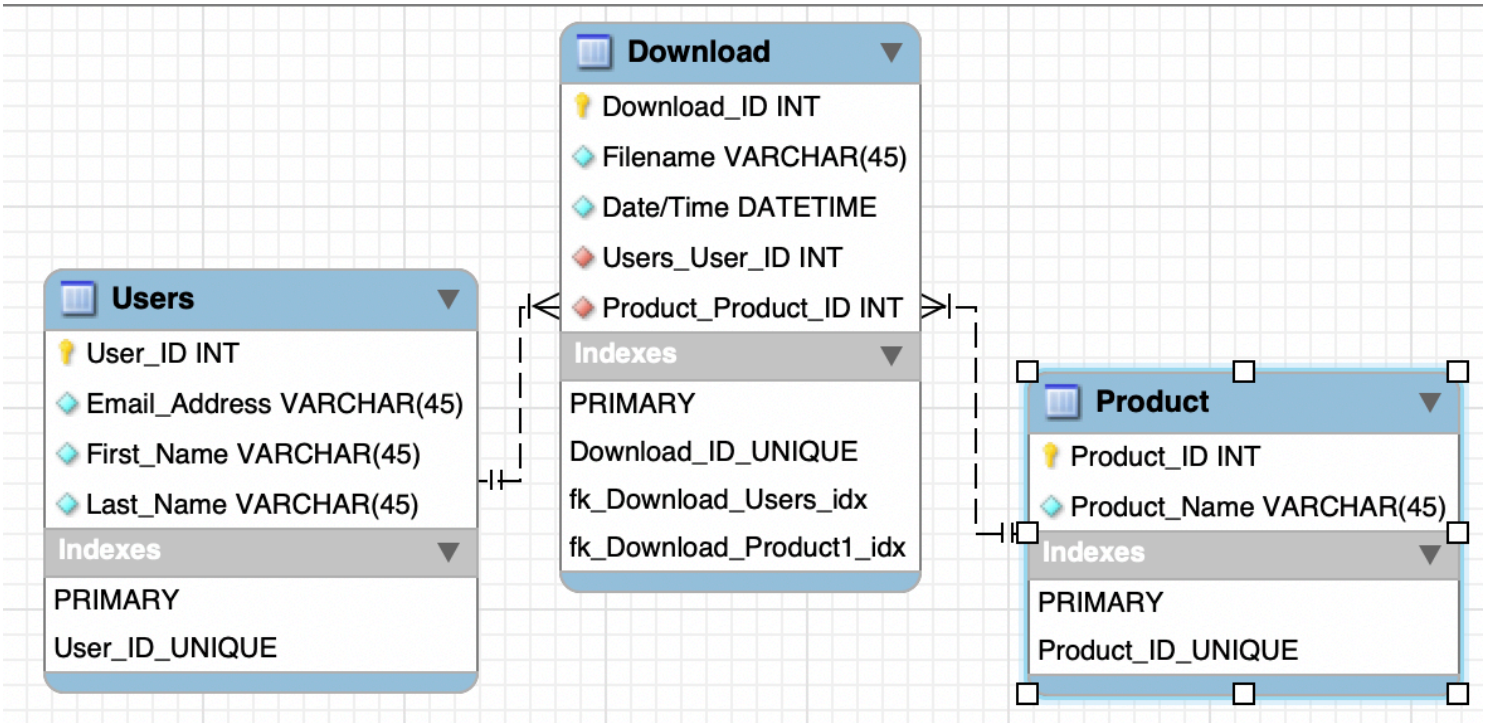
Carl Xi

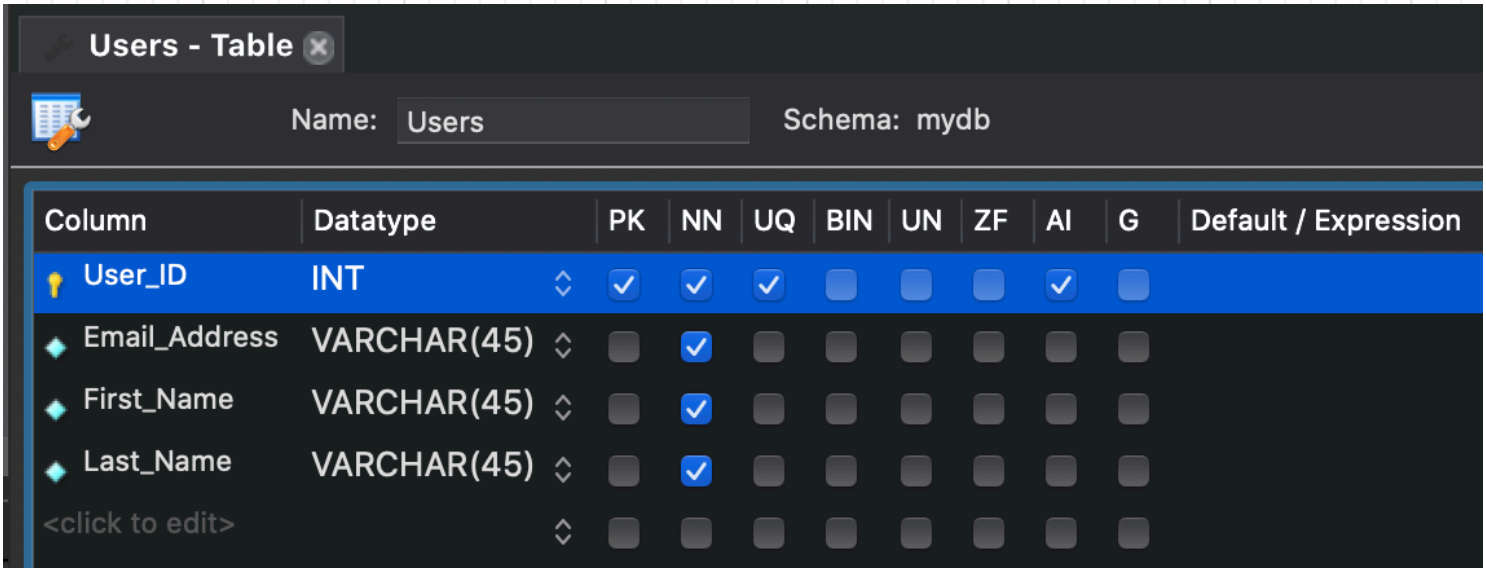
(put your name above)

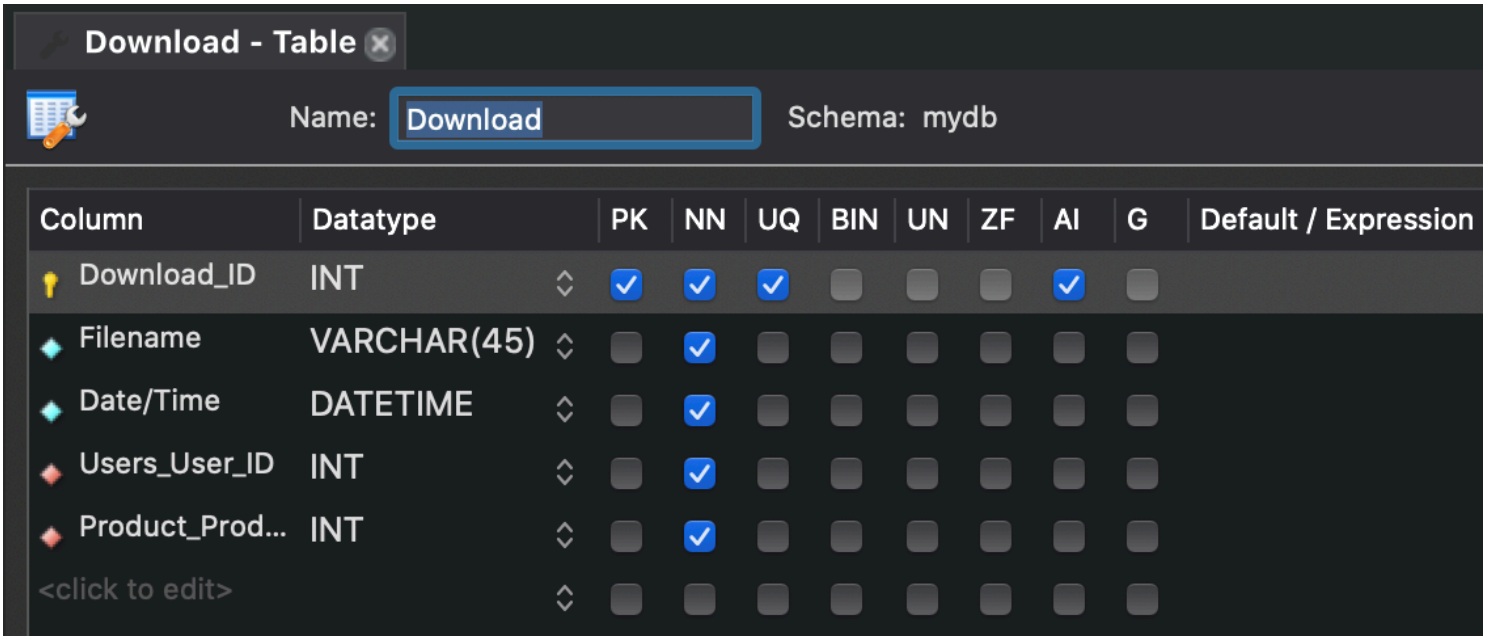
Total grade: _____ out of ___100___ points

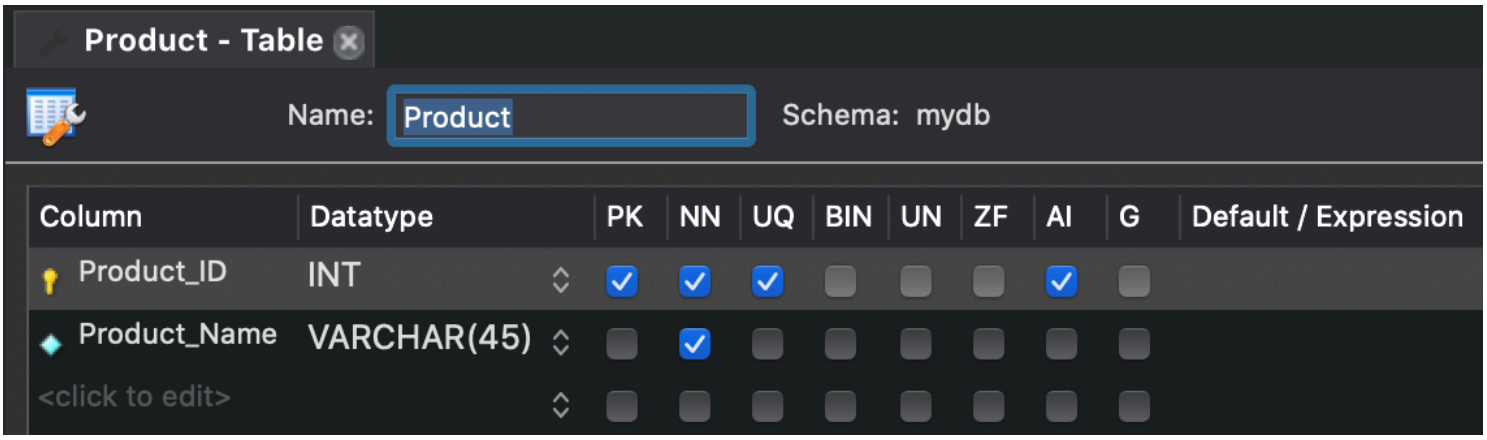
There are 5 numbered questions. Please answer them all and submit your assignment as a single PDF or Word file by uploading it to the HW2 drop-box on the course website. You should provide: SQL statements, results of the SQL statement (typically copy first 10 rows), and answers to questions, if any.

1. **Use MySQL Workbench to create an EER diagram for a database that stores information about the downloads that users make.**
   - **Each user must have an email address, first name, and last name.**
   - **Each user can have one or more downloads.**
   - **Each download must have a filename and download date/time.**
   - **Each product can be related to one or more downloads.**
   - **Each product must have a name.**









2. Use MySQL Workbench to open the EER diagram that you created in exercise 1. Then, export a script that creates the database and save this script in a file named ex3-2.sql. Next, use MySQL Workbench to open this file and review it. Report the script here.

```
-- MySQL Script generated by MySQL Workbench
-- Mon Jan 27 01:28:52 2020
-- Model: New Model    Version: 1.0
-- 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,ER
ROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
```

```
-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`Users`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Users` (
  `User_ID` INT NOT NULL AUTO_INCREMENT,
  `Email_Address` VARCHAR(45) NOT NULL,
  `First_Name` VARCHAR(45) NOT NULL,
  `Last_Name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`User_ID`),
  UNIQUE INDEX `User_ID_UNIQUE` (`User_ID` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Product`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Product` (
  `Product_ID` INT NOT NULL AUTO_INCREMENT,
  `Product_Name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`Product_ID`),
  UNIQUE INDEX `Product_ID_UNIQUE` (`Product_ID` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`Download`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Download` (
  `Download_ID` INT NOT NULL AUTO_INCREMENT,
  `Filename` VARCHAR(45) NOT NULL,
  `Date/Time` DATETIME NOT NULL,
  `Users_User_ID` INT NOT NULL,
  `Product_Product_ID` INT NOT NULL,
  PRIMARY KEY (`Download_ID`),
  UNIQUE INDEX `Download_ID_UNIQUE` (`Download_ID` ASC) VISIBLE,
  INDEX `fk_Download_Users_idx` (`Users_User_ID` ASC) VISIBLE,
  INDEX `fk_Download_Product1_idx` (`Product_Product_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Download_Users`
    FOREIGN KEY (`Users_User_ID`)
    REFERENCES `mydb`.`Users` (`User_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Download_Product1`
    FOREIGN KEY (`Product_Product_ID`)
```

REFERENCES `mydb`.`Product` (`Product_ID`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

3. **Run the script you created in exercise 2 to create the database under the name my_web_db. Write a script that adds rows to the database. In particular,**
   - **Add two rows to the Users and Products tables.**

INSERT INTO Users(User_ID, Email_Address, First_Name, Last_Name)
Values (1, 'johnsmith@gmail.com', 'John', 'Smith');
INSERT INTO Users(User_ID, Email_Address, First_Name, Last_Name)
Values (2, 'janedoe@yahoo.com', 'Jane', 'Doe');
INSERT INTO Product(Product_ID, Product_Name)
Values(1, 'Local Music Vol 1');
INSERT INTO Product(Product_ID, Product_Name)
Values(2, 'Local Music Vol 2');

SQL Code

```
INSERT INTO Users(User_ID, Email_Address, First_Name, Last_Name)
Values (1, 'johnsmith@gmail.com', 'John', 'Smith');
INSERT INTO Users(User_ID, Email_Address, First_Name, Last_Name)
Values (2, 'janedoe@yahoo.com', 'Jane', 'Doe');
INSERT INTO Product(Product_ID, Product_Name)
Values(1, 'Local Music Vol 1');
INSERT INTO Product(Product_ID, Product_Name)
Values(2, 'Local Music Vol 2');
```

Product Table

Product_ID	Product_Name
1	Local Music Vol 1
2	Local Music Vol 2

Users Table

User_ID	Email_Address	First_Name	Last_Name
1	johnsmith@gmail.com	John	Smith
2	janedoe@yahoo.com	Jane	Doe

- **Add three rows to the Downloads table:**
    - **one row for user 1 and product 2;**
    - **one row for user 2 and product 1;**
    - **and one row for user 2 and product 2.**
  - **Use the NOW function to insert the current date and time into the download_date column.**

INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (1, 'pedals_are_falling.mp3', NOW(), 1, 2);
INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (2, 'turn_signal.mp3', NOW(), 2, 1);
INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (3, 'one_horse_town.mp3', NOW(), 2, 2);

SQL Code

```
INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (1, 'pedals_are_falling.mp3', NOW(), 1, 2);
INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (2, 'turn_signal.mp3', NOW(), 2, 1);
INSERT INTO Download(Download_ID, FIlename, `Date/Time`, Users_User_ID, Product_Product_ID)
VALUE (3, 'one_horse_town.mp3', NOW(), 2, 2);
```

Download Table

Download_ID	Filename	Date/Time	Users_User_ID	Product_Product_ID
1	pedals_are_falling.mp3	2020-01-27 01:53:21	1	2
2	turn_signal.mp3	2020-01-27 01:53:21	2	1
3	one_horse_town.mp3	2020-01-27 01:53:21	2	2

Write a SELECT statement that joins the three tables and retrieves the data from these tables like this:

email_address	first_name	last_name	download_date	filename	product_name
johnsmith@gmail.com	John	Smith	2015-04-24 16:15:38	pedals_are_falling.mp3	Local Music Vol 1
janedoe@yahoo.com	Jane	Doe	2015-04-24 16:15:38	turn_signal.mp3	Local Music Vol 1
janedoe@yahoo.com	Jane	Doe	2015-04-24 16:15:38	one_horse_town.mp3	Local Music Vol 2

Sort the results by the email address in descending sequence and the product name in ascending sequence.

SELECT Email_Address, First_Name, Last_Name, `Date/Time`, Filename, Product_Name
FROM Users AS U
INNER JOIN Download AS D ON U.User_ID = D.Users_User_ID
INNER JOIN Product AS P ON D.Product_Product_ID = P.Product_ID
ORDER BY Email_Address DESC, P.Product_Name;

SQL Code

```
SELECT Email_Address, First_Name, Last_Name, `Date/Time`, Filename, Product_Name
FROM Users AS U
INNER JOIN Download AS D ON U.User_ID = D.Users_User_ID
INNER JOIN Product AS P ON D.Product_Product_ID = P.Product_ID
ORDER BY Email_Address DESC, P.Product_Name;
```

SQL Query Result

Email_Address	First_Name	Last_Name	Date/Time	Filename	Product_Name
johnsmith@gmail.com	John	Smith	2020-01-27 01:53:21	pedals_are_falling.mp3	Local Music VOl 2
janedoe@yahoo.com	Jane	Doe	2020-01-27 01:53:21	turn_signal.mp3	Local Music VOl 1
janedoe@yahoo.com	Jane	Doe	2020-01-27 01:53:21	one_horse_town.mp3	Local Music VOl 2

4. **Create a view named customer_addresses that shows the shipping and billing addresses for each customer in the *my_guitar_shop* database. This view should return these columns from the Customers table: customer_id, email_address, last_name, and first_name. This view should also return these columns from the Addresses table: bill_line1, bill_line2, bill_city, bill_state, bill_zip, ship_line1, ship_line2, ship_city, ship_state, and ship_zip. The rows in this view should be sorted by the last_name and then first_name columns.**

CREATE VIEW Customer_Addresses AS
SELECT C.customer_id, C.email_address, C.last_name, C.first_name,
A.line1 AS bill_line1, A.line2 AS bill_line2, A.city AS bill_city, A.state AS bill_state, A.zip_code AS bill_zip,
A2.line1 AS ship_line1, A2.line2 AS ship_line2, A2.city AS ship_city, A2.state AS ship_state, A2.zip_code AS ship_zip
FROM customers AS C
JOIN addresses AS A on A.address_id = C.billing_address_id
JOIN addresses AS A2 on C.shipping_address_id = A2.address_id
ORDER BY last_name, first_name;

SQL Code to Generate the View

```
CREATE VIEW Customer_Addresses AS
SELECT C.customer_id, C.email_address, C.last_name, C.first_name,
A.line1 AS bill_line1, A.line2 AS bill_line2, A.city AS bill_city, A.state AS bill_state, A.zip_code AS bill_zip,
A2.line1 AS ship_line1, A2.line2 AS ship_line2, A2.city AS ship_city, A2.state AS ship_state, A2.zip_code AS ship_zip
FROM customers AS C
JOIN addresses AS A on A.address_id = C.billing_address_id
JOIN addresses AS A2 on C.shipping_address_id = A2.address_id
ORDER BY last_name, first_name;
```

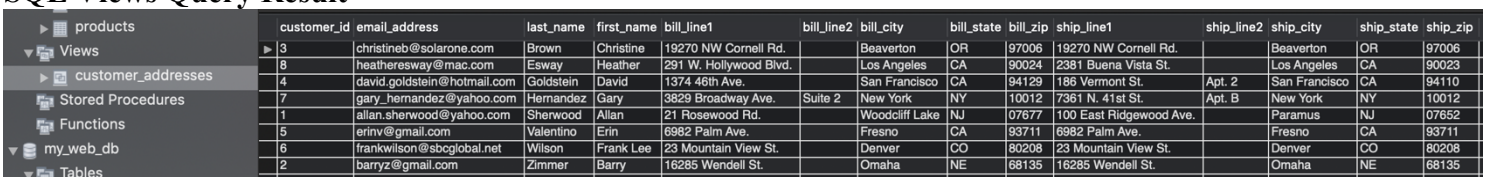
---

*SELECT * FROM my_guitar_shop.customer_addresses;*

SQL Code to View the View

```
SELECT * FROM my_guitar_shop.customer_addresses;
```

SQL Views Query Result



products
Views
customer_addresses
Stored Procedures
Functions
my_web_db
Tables

customer_id	email_address	last_name	first_name	bill_line1	bill_line2	bill_city	bill_state	bill_zip	ship_line1	ship_line2	ship_city	ship_state	ship_zip
3	christineb@solarone.com	Brown	Christine	19270 NW Cornell Rd.		Beaverton	OR	97006	19270 NW Cornell Rd.		Beaverton	OR	97006
8	heatheresway@mac.com	Esway	Heather	291 W. Hollywood Blvd.		Los Angeles	CA	90024	2381 Buena Vista St.		Los Angeles	CA	90023
4	david.goldstein@hotmail.com	Goldstein	David	1374 46th Ave.		San Francisco	CA	94129	186 Vermont St.	Apt. 2	San Francisco	CA	94110
7	gary_hernandez@yahoo.com	Hernandez	Gary	3829 Broadway Ave.	Suite 2	New York	NY	10012	7361 N. 41st St.	Apt. B	New York	NY	10012
1	allan.sherwood@yahoo.com	Sherwood	Allan	21 Rosewood Rd.		Woodcliff Lake	NJ	07677	100 East Ridgewood Ave.		Paramus	NJ	07652
5	erinv@gmail.com	Valentino	Erin	6982 Palm Ave.		Fresno	CA	93711	6982 Palm Ave.		Fresno	CA	93711
6	frankwilson@sbcglobal.net	Wilson	Frank Lee	23 Mountain View St.		Denver	CO	80208	23 Mountain View St.		Denver	CO	80208
2	barryz@gmail.com	Zimmer	Barry	16285 Wendell St.		Omaha	NE	68135	16285 Wendell St.		Omaha	NE	68135

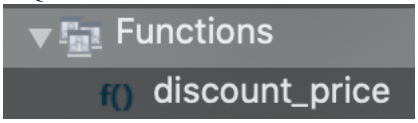
5. Write a script that creates and calls a stored function named discount_price that calculates the discount price of an item in the Order_Items table of the *my_guitar_shop* database (discount amount subtracted from item price). To do that, this function should accept one parameter for the item ID, and it should return the value of the discount price for that item.

```
DELIMITER |
CREATE FUNCTION discount_price (itemid INT)
        RETURNS DECIMAL(10,2)
   DETERMINISTIC
   BEGIN
   DECLARE price DECIMAL(10,2);
   DECLARE discount DECIMAL(10,2);
   SELECT item_price FROM order_items WHERE itemid = item_id INTO price;
   SELECT discount_amount FROM order_items WHERE itemid = item_id INTO discount;
   RETURN price - discount;
   END |
```

SQL Code to Generate Function

```
DELIMITER |
CREATE FUNCTION discount_price (itemid INT)
    RETURNS DECIMAL(10,2)
    DETERMINISTIC
    BEGIN
    DECLARE price DECIMAL(10,2);
    DECLARE discount DECIMAL(10,2);
    SELECT item_price FROM order_items WHERE itemid = item_id INTO price;
    SELECT discount_amount FROM order_items WHERE itemid = item_id INTO discount;
    RETURN price - discount;
    END |
```

SQL Function Stored



Functions
f() discount_price

SELECT item_id, discount_price(item_id) AS `discount_price` FROM order_items;

SQL Code for implementing the Function on item_id

```
SELECT item_id, discount_price(item_id) AS `discount_price` FROM order_items;
```

SQL Function Run Results

item_id	discount_price
1	839.30
2	303.79
3	1208.16
4	253.15
5	839.30
6	299.00
7	299.00
8	489.30
9	559.99
10	489.99
11	679.99
12	489.30