**Trail\_3 Creating tables**

CREATE TABLE **CUSTOMER** (

cId INT NOT NULL CHECK (cId > 0),

Cname VARCHAR(255) NOT NULL,

Street VARCHAR(255) NOT NULL,

City VARCHAR(255) NOT NULL,

StateAb VARCHAR(2) NOT NULL,

Zipcode VARCHAR(10) NOT NULL,

CONSTRAINT PK\_CUSTOMER PRIMARY KEY (cId)

);

CREATE TABLE **STORE** (

sId INT NOT NULL CHECK (sId > 0),

Sname VARCHAR(255) NOT NULL,

Street VARCHAR(255) NOT NULL,

City VARCHAR(255) NOT NULL,

StateAb VARCHAR(2) NOT NULL,

Zipcode VARCHAR(10) NOT NULL,

Sdate DATE Not NULL,

Telno VARCHAR(15) NOT NULL,

URL VARCHAR(255) NOT NULL,

CONSTRAINT PK\_STORE PRIMARY KEY (sId)

);

CREATE TABLE **VENDOR** (

vId INT NOT NULL CHECK (vId > 0),

Vname VARCHAR(255) NOT NULL,

Street VARCHAR(255)NOT NULL,

City VARCHAR(255) NOT NULL,

StateAb VARCHAR(2) NOT NULL,

Zipcode VARCHAR(10) NOT NULL,

CONSTRAINT PK\_VENDOR PRIMARY KEY (vId)

);

CREATE TABLE **STORE\_CUSTOMER** (

sId INT NOT NULL CHECK (sId > 0),

cId INT NOT NULL CHECK (cId > 0),

CONSTRAINT PK\_STORE\_CUSTOMER PRIMARY KEY (sId, cId),

CONSTRAINT FK\_STORE\_STOREID FOREIGN KEY (sId) REFERENCES STORE(sId),

CONSTRAINT FK\_STORE\_CUSTOMERID FOREIGN KEY (cId) REFERENCES CUSTOMER(cId)

);

CREATE TABLE **ITEM** (

iId INT NOT NULL CHECK (iId > 0),

Iname VARCHAR(255) NOT NULL,

Sprice DECIMAL(10,2) NOT NULL CHECK (Sprice > 0),

CONSTRAINT PK\_ITEM PRIMARY KEY(iId)

);

CREATE TABLE **VENDOR\_ITEM** (

Vid INT NOT NULL CHECK (vId > 0),

iId INT NOT NULL CHECK (iId > 0),

Vprice DECIMAL (10, 2) NOT NULL,

CONSTRAINT PK\_VENDOR\_ITEM PRIMARY KEY (Vid, iId),

CONSTRAINT FK\_VENDOR\_ITEM FOREIGN KEY (Vid) REFERENCES VENDOR (Vid),

CONSTRAINT FK\_VENDOR\_ITEMID FOREIGN KEY (iId) REFERENCES ITEM (iId)

);

CREATE TABLE **VENDOR\_STORE** (

vId INT NOT NULL CHECK (vId > 0),

sId INT NOT NULL CHECK (sId > 0),

CONSTRAINT PK\_VENDOR\_STORE PRIMARY KEY (Vid, sId),

CONSTRAINT PK\_VENDOR\_VENDORID FOREIGN KEY (Vid) REFERENCES VENDOR (vId),

CONSTRAINT PK\_VENDOR\_VENDOR\_STOREID FOREIGN KEY (Sid) REFERENCES STORE (sId)

);

CREATE TABLE **OLDPRICE**(

iId INT NOT NULL CHECK (iId > 0),

Sprice DECIMAL (10, 2) NOT NULL CHECK (Sprice >= 0),

Sdate DATE NOT NULL,

Edate DATE CHECK (Sdate < Edate),

CONSTRAINT PK\_OLDPRICE PRIMARY KEY (iId, Sprice),

CONSTRAINT FK\_OLDPRICE FOREIGN KEY (iId) REFERENCES ITEM (iId)

);

CREATE TABLE **CONTRACT** (

vId INT NOT NULL CHECK (vId > 0),

ctId INT NOT NULL CHECK (ctId BETWEEN 1 AND 10),

SDate DATE NOT NULL,

CTime TIME NOT NULL,

Ctname VARCHAR(255) NOT NULL,

CONSTRAINT PK\_CONTRACT PRIMARY KEY (vId, ctID),

CONSTRAINT FK\_CONTRACT\_VENDOR FOREIGN KEY (vId) REFERENCES VENDOR (vId)

);

CREATE TABLE **STORE\_ITEM** (

sId INT NOT NULL CHECK (sId > 0),

iId INT NOT NULL CHECK (iId > 0),

Scount INT CHECK (Scount >= 0),

CONSTRAINT PK\_STORE\_ITEM PRIMARY KEY(sId, iID),

CONSTRAINT FK\_STOREID FOREIGN KEY (sId) REFERENCES STORE (sId),

CONSTRAINT FK\_ITEMID FOREIGN KEY (iId) REFERENCES ITEM (iId)

);

CREATE TABLE EMPLOYEE (

sId INT NOT NULL CHECK (sId > 0),

SSN CHAR(9) NOT NULL,

Ename VARCHAR(255) NOT NULL,

Street VARCHAR (255) NOT NULL,

City VARCHAR (255) NOT NULL,

StateAb CHAR (2) NOT NULL,

ZipCode INT NOT NULL,

EType ENUM('F', 'T', 'I') NOT NULL,

Bdate DATE NOT NULL,

Sdate DATE NOT NULL,

Edate DATE CHECK (EDate IS NULL OR (EDate >= SDate)),

Levels VARCHAR(255),

Asalary DECIMAL(10, 2),

Agency VARCHAR (255),

Hsalary DECIMAL (10, 2),

Institute VARCHAR (255),

Itype VARCHAR(10),

CONSTRAINT PK\_EMPLOYEE\_SSN PRIMARY KEY(SSN),

CONSTRAINT FK\_STORE\_ID FOREIGN KEY (sId) REFERENCES STORE(sId)

);

DELIMITER $$

CREATE TRIGGER NUM\_EMPLOYEES\_STORE

BEFORE INSERT ON EMPLOYEE

FOR EACH ROW BEGIN

IF (SELECT COUNT(\*) FROM EMPLOYEE WHERE sId = New.sId) >= 25 THEN

SIGNAL SQLSTATE '02000' SET MESSAGE\_TEXT = 'More than 25 employees cannot work in a store';

END IF;

END$$

DELIMITER ;

CREATE TABLE **ORDERS** (

oId INT NOT NULL CHECK (oId > 0),

sId INT NOT NULL CHECK (sId > 0),

cId INT NOT NULL CHECK (cId > 0),

Odate DATE NOT NULL,

Ddate DATE NOT NULL CHECK (Ddate >= Odate),

Amount INT NOT NULL CHECK (Amount > 0),

PRIMARY KEY (oId),

CONSTRAINT FK\_sId FOREIGN KEY (sId) REFERENCES STORE (sId),

CONSTRAINT FK\_cId FOREIGN KEY (cId) REFERENCES CUSTOMER (cId)

);

CREATE TABLE **ORDER\_ITEM**(

oId INT NOT NULL CHECK (oId > 0),

sId INT NOT NULL CHECK (sId > 0),

iId INT NOT NULL CHECK (iId > 0),

Icount INT,

CONSTRAINT PK\_ORDER PRIMARY KEY (oId, sId, iId),

CONSTRAINT FK\_ORDER FOREIGN Key (oId) REFERENCES ORDERS (oId),

CONSTRAINT FK\_STORE FOREIGN Key (sId) REFERENCES STORE (sId),

CONSTRAINT FK\_ITEM FOREIGN Key (iId) REFERENCES ITEM (iId)

);

**PART 2**

Q9 Retrieve a list of records where each record is comprised of the order date, customer name, and the minimum or the  
max order amount. If there are multiple such orders, list the information for the minimum amount orders followed by the  
information for the maximum amount.  
The dollar amount should be stated with $, appropriate commas, and up to 2 decimal places. (Example: $12,000.00)

SELECT

o.Odate AS 'Order date',

c.Cname AS 'Customer Name',

CONCAT('$', FORMAT(CASE WHEN o.Amount = min\_max.min\_amount THEN min\_max.min\_amount ELSE min\_max.max\_amount END, 2)) AS Amount

FROM

orders o

JOIN customer c ON o.cId = c.cId

JOIN (

SELECT

MIN(orders.Amount) AS min\_amount,

MAX(orders.Amount) AS max\_amount

FROM

orders

) min\_max ON o.Amount IN (min\_max.min\_amount, min\_max.max\_amount)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q10 Retrieve a record which is comprised of information about the minimum and maximum amount of order placed. The  
record will need to list the order date, customer name, and the order amount for the minimum and maximum orders. If  
there are multiple customers with the minimum and/or maximum order, list only the first name in the alphabetical order of  
such a list.  
The dollar amount should be stated with $, appropriate commas, and up to 2 decimal places. (Example: $12,000.00)

SELECT

MIN(o.Odate) AS 'MinOrder Date',

SUBSTRING\_INDEX(MIN(c.Cname), ' ', 1) AS 'MinCustomer Name',

CONCAT('$', FORMAT(MIN(o.Amount), 2)) AS 'MinAmount',

MAX(o.Odate) AS 'MaxOrder Date',

SUBSTRING\_INDEX(MIN(c.Cname), ' ', 1) AS 'MaxCustomer Name',

CONCAT('$', FORMAT(MAX(o.Amount), 2)) AS 'Max Amount'

FROM

orders o

LEFT OUTER JOIN customer c ON o.cId = c.cId

WHERE

o.Amount = (SELECT MIN(Amount) FROM orders) OR

o.Amount = (SELECT MAX(Amount) FROM orders)  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q11 Retrieve a list of records that state the information pertaining to the dates, number of min/max orders, and the total  
amounts of such orders on the pertinent dates on which the minimum and maximum orders were placed. Each record in  
this list should be comprised of the date (when min or max amount order was placed), number of minimum or maximum  
orders separately, and the total of such min or max amount orders individually.  
If multiple minimum and maximum orders are placed on same dates, add the min amount and max amount orders  
separately and list them individually.  
The dollar amount should be stated with $, appropriate commas, and up to 2 decimal places. (Example: $12,000.00)  
Column headers:  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q12 Retrieve a list of records that lists the name of customers and the sum of all the orders that they have ever placed at  
any of the Arlington Sprouts store. List the records in an ascending order by the total amount spent by customers at the  
store.  
The dollar amount should be stated with $, appropriate commas, and up to 2 decimal places. (Example: $12,000.00)  
Column headers  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q13 Retrieve a list of records that lists the name of customers, the total number of items (boxes of sprouts) they ordered at  
any time, and the sum of all the order amounts that they have ever placed at any of the Arlington Sprouts store . List the  
records in an ascending order by the total number of items(boxes of sprouts) ordered at any time , followed by the total  
amount spent by customers at the store.  
The dollar amount should be stated with $, appropriate commas, and up to 2 decimal places. (Example: $12,000.00)  
Column headers:  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q14. Execute a command to delete a record that violates a referential integrity constraint. State the message produced  
by the DBMS.  
  
DELETE FROM customer WHERE cId = 2;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q15 Execute 3 insert commands for any table that attempt to insert records, such that the records violate the explicit  
schema-based constraints (Key, Entity Integrity, Referential Integrity constraints). Make each of the 3 records  
violate a different types of integrity constraint. Include the insert statements and the error messages produced.

**Key Constraint violation**

INSERT INTO EMPLOYEE VALUES ('1', '123456781', 'Shubash Muniyappa', '10 Ray Street', 'Arlington', 'TX', '76019-1111', 'F', '1996-08-19', '2021-01-20', NULL, 'Associate 2', '30000', NULL, NULL, NULL, NULL);

**Entity Integrity**

INSERT INTO EMPLOYEE VALUES ('1', NULL , 'Shubash Muniyappa', '10 Ray Street', 'Arlington', 'TX', '76019-1111', 'F', '1996-08-19', '2021-01-20', NULL, 'Associate 2', '30000', NULL, NULL, NULL, NULL);

**Referential Integrity constraint violation**

INSERT INTO `vendor\_store` VALUES ('40', '20');  
\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Q16 Execute an update command for store\_customer table that attempts to update records in child rows, such that the  
record violate the foreign key constraint. Include the update statement and the error message.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UPDATE STORE\_CUSTOMER SET sId = 2 WHERE cId = 1;

Q6:

SELECT VENDOR.Vname AS 'Vendor Name', ITEM.Iname AS 'Item Name', VENDOR\_ITEM.Vprice AS 'Vendor Price', ITEM.Sprice AS 'Store Price'

FROM (vendor\_item

INNER JOIN vendor ON VENDOR\_ITEM.Vid =VENDOR.vId)

INNER JOIN ITEM ON vendor\_item.iId = ITEM.iId

WHERE (ITEM.Sprice - VENDOR\_ITEM.Vprice) = 2

ORDER BY VENDOR.Vname;

Q8:

SELECT Odate, CONCAT('$',COUNT(Amount)) AS 'Sum AMT', CONCAT('$',AVG(Amount)) AS 'Avg AMT', CONCAT('$',MAX(AMOUNT)) AS 'Max AMT'

FROM ORDERS GROUP BY Odate;

1. SELECT \* FROM`CUSTOMER` where Cname LIKE '% J%' OR Cname LIKE '% M%';

2. SELECT `VENDOR`.`Vname` as 'Vendor Name', `CONTRACT`.`Cname` as 'Course Name', `CONTRACT`.`Sdate` AS 'Course Date', TIME\_FORMAT(`CONTRACT`.`Ctime`, '%r') As 'Course Time' FROM `VENDOR` LEFT JOIN `CONTRACT` ON `CONTRACT`.`vId` = `VENDOR`.`vId` WHERE `CONTRACT`.`Ctime` > '14:00';

3. SELECT `VENDOR`.`Vname`, `ITEM`.`Iname`, concat('$',`VENDOR\_ITEM`.`Vprice`) as 'Vendor Price', concat('$',`ITEM`.`Sprice`) as 'Store Price'

FROM `VENDOR`

LEFT JOIN `VENDOR\_ITEM` ON `VENDOR\_ITEM`.`vId` = `VENDOR`.`vId`

LEFT JOIN `ITEM` ON `VENDOR\_ITEM`.`iId` = `ITEM`.`iId`;

4. SELECT `ITEM`.`Iname` as ‘Item Name’ , concat('$',`ITEM`.`Sprice`) as 'Store Price' FROM `ITEM` ORDER by Sprice DESC, Iname ASC LIMIT 10;

5. SELECT `VENDOR`.`Vname`,`ITEM`.`Iname`, concat('$',`VENDOR\_ITEM`.`Vprice`) as 'Vendor Price', concat('$',`ITEM`.`Sprice`) as 'Store Price' FROM `VENDOR` LEFT JOIN `VENDOR\_ITEM` ON `VENDOR\_ITEM`.`vId` = `VENDOR`.`vId` LEFT JOIN `ITEM` ON `VENDOR\_ITEM`.`iId` = `ITEM`.`iId` WHERE `ITEM`.`Sprice` > `VENDOR\_ITEM`.`Vprice`\*2;

7. SELECT `ORDERS`.`Odate`, `ORDERS`.`Amount`, `CUSTOMER`.`Cname`, `ORDER\_ITEM`.`Icount`

FROM `ORDERS`

LEFT JOIN `CUSTOMER` ON `ORDERS`.`cId` = `CUSTOMER`.`cId`

LEFT JOIN `ORDER\_ITEM` ON `ORDER\_ITEM`.`oId` = `ORDERS`.`oId` ORDER BY `ORDERS`.`Odate` ASC, `CUSTOMER`.`Cname` ASC;

OR

7. SELECT `ORDERS`.`Odate` as ‘Order date’ , `CUSTOMER`.`Cname` as ’Customer Name’ , CONCAT('$', FORMAT(SUM(`ORDERS`.`Amount`), 2)) AS 'Amount', SUM(`ORDER\_ITEM`.`Icount`) AS 'Items Ordered' FROM `ORDERS` LEFT JOIN `CUSTOMER` ON `ORDERS`.`cId` = `CUSTOMER`.`cId` LEFT JOIN `ORDER\_ITEM` ON `ORDER\_ITEM`.`oId` = `ORDERS`.`oId` GROUP BY `ORDERS`.`Odate`, `CUSTOMER`.`Cname` ORDER BY `ORDERS`.`Odate` ASC, `CUSTOMER`.`Cname` ASC;

12. SELECT `CUSTOMER`.`Cname` as 'Customer Name',CONCAT('$', FORMAT(SUM( `ORDERS`.`Amount`), 2)) AS 'Total Amount Spent' FROM `CUSTOMER` LEFT JOIN `ORDERS` ON `ORDERS`.`cId` = `CUSTOMER`.`cId` GROUP BY `CUSTOMER`.`Cname` ORDER BY SUM(`ORDERS`.`Amount`) ASC;

13. SELECT `CUSTOMER`.`Cname`, SUM(`ORDER\_ITEM`.`Icount`) AS 'Total Boxes',CONCAT('$', FORMAT(SUM(`ORDERS`.`Amount`), 2)) AS 'Total Amount' FROM `ORDERS` LEFT JOIN `CUSTOMER` ON `ORDERS`.`cId` = `CUSTOMER`.`cId` LEFT JOIN `ORDER\_ITEM` ON `ORDER\_ITEM`.`oId` = `ORDERS`.`oId` GROUP BY `CUSTOMER`.`Cname` ORDER BY SUM(`ORDER\_ITEM`.`Icount`) ASC, SUM(`ORDERS`.`Amount`) ASC;