DBMS Final Report

April 27, 2017 / CSC 675/775 (02)

Team Members

Kanakapriya Krishnakumar

Aanchal Narad

Rita Zaidan

Mehek Khan

Phase 1

Task 1

Topic: Restaurant Database

Requirement Analysis: what is going to be stored? What are we going to do with the data? Who should access the data?

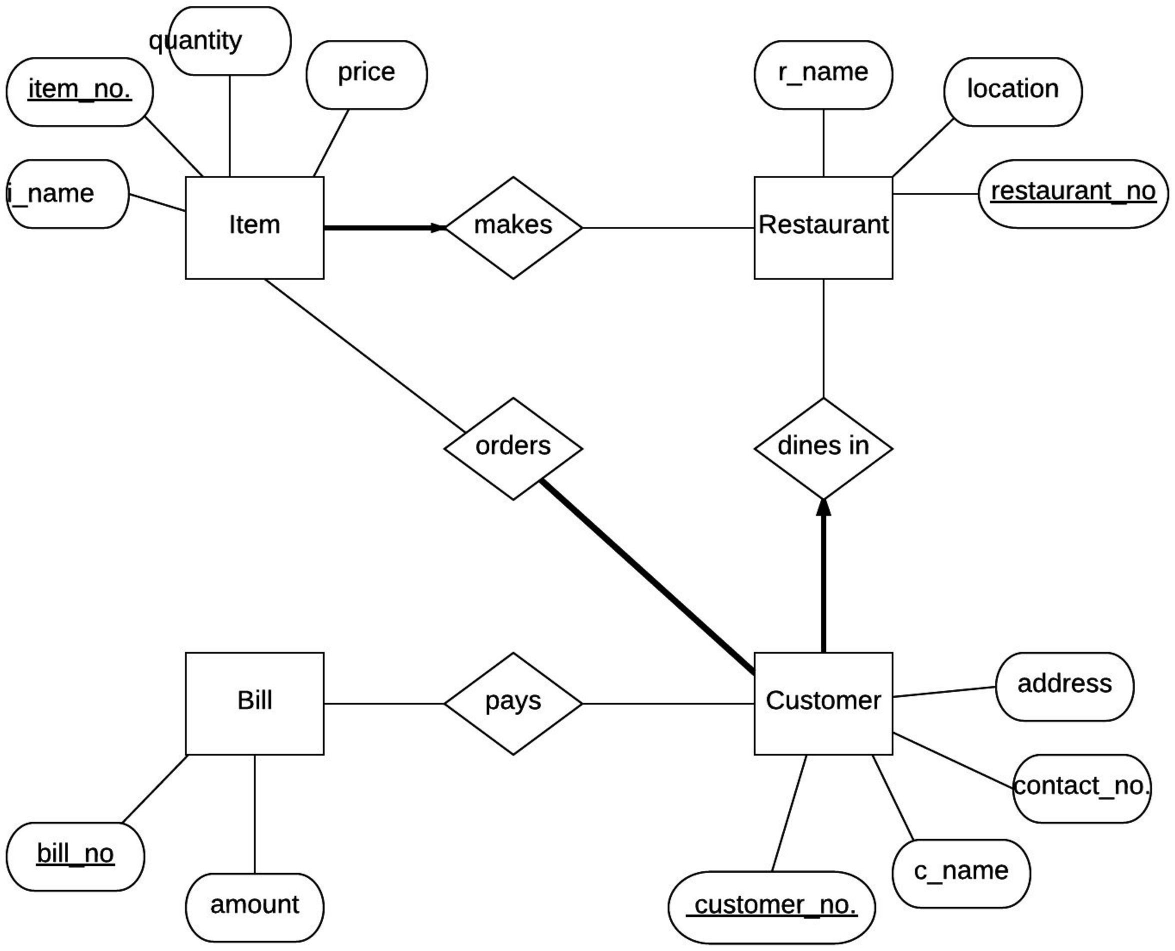
We want to store information about this restaurant (contact information, name, and location), so we created a **Restaurant** entity set, which will hold our restaurant entity. We also want to store information about customers (contact number, name, and location) who dine in the restaurant and items (with info concerning item number, name of item, quantity, and price) made in the restaurant, so we created **Item** and **Customer** entity sets. We want to store information about the bills (such as total amount, bill number, customer number, and item number) to be paid as well, so we have a **Bill** entity set. The entities and their corresponding attributes will be designed into an Entity-Relationship (ER) diagram (the conceptual design), and then converted into relational schema (the physical database design), which is represented by tables. There will be a table for each entity set: Restaurant, Item ,Customer, and Bill. Only restaurant staff should be able to access this database.

Project Description

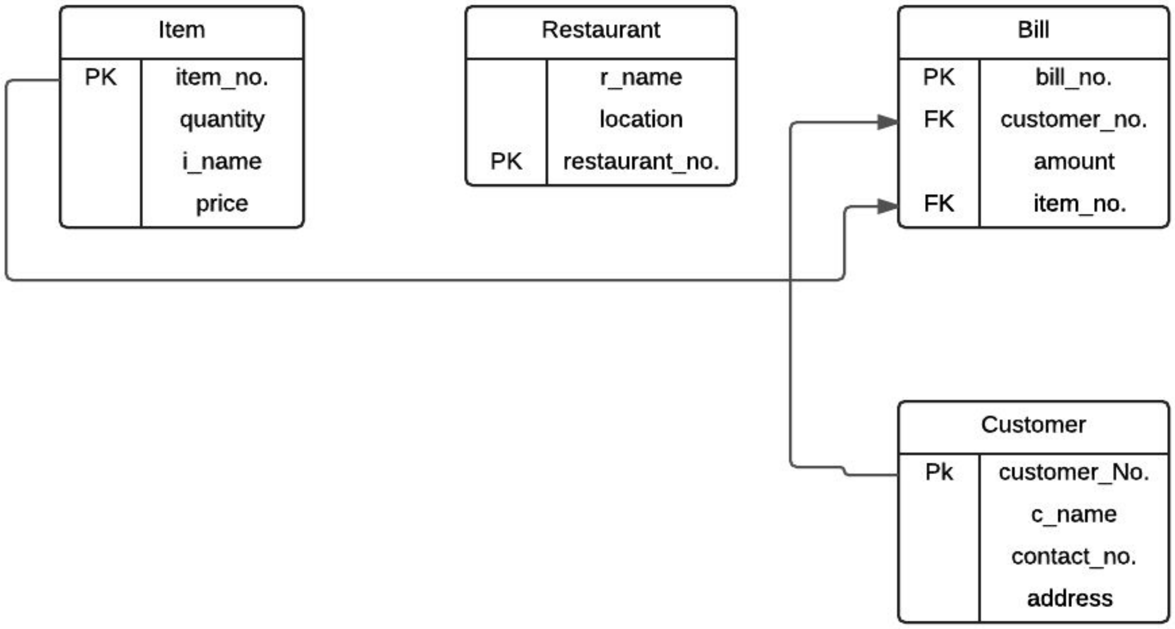
Our project is the database of a restaurant. We have a restaurant entity with its attributes, its contact number (the primary key, since it's a unique identifier for the restaurant), name, and location. The restaurant makes (indicated by "makes" relationship) items for customers. The Item entity set has item number (primary key), name, quantity, and price as attributes that apply to all its entities. Customers dine in (indicated by the "dines_in" relationship) this restaurant. Customers have their own contact number (primary key), name, and address. Every customer orders at least one item, so there is total participation (indicated by a bold line) for customers ordering item(s) from the restaurant. However, not every customer will necessarily pay the bill (indicated by the "pays" relationship). For example, there may be a group of customers where one customer pays the total bill; therefore, there is not total participation of customers paying bills. The attributes of entities in the Bill entity set are the unique bill number for each bill (primary key), amount of bill, and item numbers (a foreign key, since it is a primary key for the Item entity set), and customer number, another foreign key since it is also the primary key for the Customer entity set (this is an attribute for the Bill entity set because it tells which customer(s) pays the bill).

Task 2

ER Diagram: .



Logical Schemas:



Phase 2

Task 1:

Create Tables, indexes and constraints.

```
CREATE TABLE Item (
  item_no INTEGER NOT NULL PRIMARY KEY,
  quantity INTEGER NOT NULL,
  i_name CHAR(20)  NOT NULL,
  price integer  NOT NULL );

CREATE TABLE Customer(
  customer_no CHAR(20)   NOT NULL PRIMARY KEY,
  c_name CHAR(20)  NOT NULL,
  contact_no CHAR(20) NOT NULL ,
  address VARCHAR(100)
);

CREATE TABLE Restaurant (
  restaurant_no CHAR(20) NOT NULL PRIMARY KEY,
  r_name CHAR (30) NOT NULL,
  location VARCHAR(100) NOT NULL
);

CREATE TABLE Bill(
  bill_no INTEGER NOT NULL PRIMARY KEY,
  amount DOUBLE NOT NULL,
  customer_no NOT NULL CHAR(20),
  item_no NOT NULL  INTEGER,

  FOREIGN KEY (item_no) REFERENCES Item(item_no) ON DELETE NO ACTION,
  FOREIGN KEY (customer_no) REFERENCES Customer(customer_no) ON DELETE NO
ACTION
);

CREATE TABLE Makes(
restaurant_no NOT NULL CHAR(20),
Item_no NOT NULL INTEGER,
PRIMARY KEY (restaurant_no, item_no),
FOREIGN KEY (item_no) REFERENCES Item(item_no) ON DELETE NO ACTION,
FOREIGN KEY (restaurant_no) REFERENCES Re staurant(restaurant_no) ON DELETE
NO ACTION
);

CREATE TABLE Orders(
item_no  NOT NULL INTEGER,
customer_no NOT NULL CHAR(20),
PRIMARY KEY (customer_no,item_no),
FOREIGN KEY (customer_no) REFERENCES Customer(customer_no) ON DELETE NO
ACTION,
FOREIGN KEY (item_no) REFERENCES Item(item_no) ON DELETE NO ACTION
);

CREATE TABLE Pays(
bill_no INTEGER NOT NULL,
customer_no CHAR(20) NOT NULL,
PRIMARY KEY (bill_no, customer_no),
FOREIGN KEY (bill_no) REFERENCES Bill(bill_no) ON DELETE CASCADE,
FOREIGN KEY(customer_no) REFERENCES Customer(customer_no) ON DELETE NO ACTION
);

CREATE TABLE Dines_In(
customer_no CHAR(20) NOT NULL,
restaurant_no CHAR(20) NOT NULL,
PRIMARY KEY (customer_no, restaurant_no),
FOREIGN KEY (customer_no) REFERENCES Customer(customer_no) ON DELETE NO ACTION,
FOREIGN KEY (restaurant_no) REFERENCES Restaurant(restaurant_no) ON DELETE NO ACTION
);

ALTER TABLE Orders
ADD INDEX IX_tbleOrder_itemno UNIQUE
HASH (item_no) WITH (BUCKET_COUNT = 3)

ALTER TABLE Dines_In
ADD INDEX IX_tbleDinesIn_customer_no UNIQUE
HASH (customer_no) WITH (BUCKET_COUNT = 3)
```

Task 2:

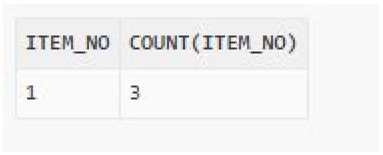
Collect and import data. You can collect data manually or import data from any available data repository.

```
INSERT INTO Item VALUES(1, 500,'Krabby Patty', 8.50);
INSERT INTO Item VALUES(2, 400,'French  Fries', 3.75);
INSERT INTO Item VALUES(3, 300,'Krusty Milkshake', 4.20);
INSERT INTO Item VALUES(4, 200,'Cakes', 3.50);
INSERT INTO Item VALUES(5, 100,'Juice', 6.50);
INSERT INTO Item VALUES(6, 600,'eggs', 7.50);
INSERT INTO Item VALUES(7, 700,'french toast', 9.50);
INSERT INTO Item VALUES(8, 800,'milk', 2.50);
INSERT INTO Item VALUES(9, 900,'soda', 1.50);
INSERT INTO Item VALUES(10, 90,'pancakes', 8.50);
INSERT INTO Item VALUES(11, 80,'crepes', 7.50);
INSERT INTO Item VALUES(12, 90,'chicken tenders', 7.50);
INSERT INTO Item VALUES(13, 20,'salad', 6.50);
INSERT INTO Item VALUES(14, 750,'rice', 8.50);
INSERT INTO Item VALUES(15, 90,'onion rings', 5.50);


INSERT INTO Customer VALUES(1, 'Angel', '415-001-8976','001 Hot Street');
INSERT INTO Customer VALUES(2, 'Mary', '650-556-8760','657 Sweet Street');
INSERT INTO Customer VALUES(3, 'John', '415-654-8776','234 Chilly Street');
INSERT INTO Customer VALUES(4, 'Edwin', '650-765-7689','333 Spicy Street');
INSERT INTO Customer VALUES(5, 'Jim', '415-654-1234','111 Sear Street');
INSERT INTO Customer VALUES(6, 'Teddy', '510-767-8769','222 Gear Street');
INSERT INTO Customer VALUES(7, 'Joel', '650-675-6453','333 Hello Street');
INSERT INTO Customer VALUES(8, 'Bill', '415-123-5678','121 Fella Street');
INSERT INTO Customer VALUES(9, 'Sandy', '510-222-3333','220 Della Street');
INSERT INTO Customer VALUES(10, 'Mady', '210-444-4444','999 Hayne Street');
INSERT INTO Customer VALUES(11, 'Stela', '210-222-4444','222 Diana Street');
INSERT INTO Customer VALUES(12, 'Cary', '310-234-2345','289 Fiscal Street');
INSERT INTO Customer VALUES(13, 'Pady', '210-234-2345','121 Fear Street');
INSERT INTO Customer VALUES(14, 'Hail', '210-234-1111','234 Ghost Street');
INSERT INTO Customer VALUES(15, 'Rainy', '230-444-4444','112 Monster Street');

INSERT INTO Bill VALUES (17, 100.50, 1, 13);
INSERT INTO Bill VALUES (18, 50.50, 2, 12);
INSERT INTO Bill VALUES (19, 51.50, 3, 11);
INSERT INTO Bill VALUES (20, 52.50, 4, 10);
INSERT INTO Bill VALUES (21, 53.99, 5, 9);
INSERT INTO Bill VALUES (22, 54.99, 6, 8);
INSERT INTO Bill VALUES (23, 55.99, 7, 7);
INSERT INTO Bill VALUES (24, 56.99, 8, 6);
INSERT INTO Bill VALUES (25, 57.99, 9, 5);
INSERT INTO Bill VALUES (26, 58.99, 10, 4);
INSERT INTO Bill VALUES (27, 59.99, 11, 3);
INSERT INTO Bill VALUES (28, 60.99, 12, 2);
INSERT INTO Bill VALUES (29, 76.99, 13, 1);
INSERT INTO Bill VALUES (30, 78.99, 14, 14);
INSERT INTO Bill VALUES (31, 771.99, 15, 15);

INSERT INTO Restaurant values ('415-123-4567', 'Krusty Krab', '123_underdog_Street');
INSERT INTO Restaurant VALUES('415-222-4467', 'Krusty Love' ,
'323_Streetdog_Street');
INSERT INTO Restaurant VALUES('415-143-6667', 'Krusty Fun' , '363_Crapdog_Street');
INSERT INTO Pays VALUES (17, 1);
INSERT INTO Pays VALUES (18, 2);
INSERT INTO Pays VALUES (19, 3);
INSERT INTO Pays VALUES (20, 4);
INSERT INTO Pays VALUES (20, 5);
INSERT INTO Pays VALUES (22, 6);
INSERT INTO Pays VALUES (22, 7);
INSERT INTO Pays VALUES (22, 8);
INSERT INTO Pays VALUES (25, 9);
INSERT INTO Pays VALUES (26, 10);
INSERT INTO Pays VALUES (27, 11);
INSERT INTO Pays VALUES (27, 13);
INSERT INTO Pays VALUES (27, 12);
INSERT INTO Pays VALUES (28, 11);
INSERT INTO Pays VALUES (28, 14);

INSERT INTO Orders VALUES( 1, 15 );
INSERT INTO Orders VALUES( 2, 14 );
INSERT INTO Orders VALUES( 3, 13 );
INSERT INTO Orders VALUES( 4, 12 );
INSERT INTO Orders VALUES( 5, 11 );
INSERT INTO Orders VALUES( 6, 10 );
INSERT INTO Orders VALUES( 7, 9 );
INSERT INTO Orders VALUES( 8, 8 );
INSERT INTO Orders VALUES( 9, 7 );
INSERT INTO Orders VALUES( 10, 6 );
INSERT INTO Orders VALUES( 11, 5 );
INSERT INTO Orders VALUES( 12, 4 );
INSERT INTO Orders VALUES( 13, 3 );
INSERT INTO Orders VALUES( 14, 2 );
INSERT INTO Orders VALUES( 15, 1 );

INSERT INTO Makes VALUES( '415-123-4567', 1 );
INSERT INTO Makes VALUES( '415-143-6667', 2 );
INSERT INTO Makes VALUES( '415-143-6667', 3 );
INSERT INTO Makes VALUES( '415-143-6667', 4 );
INSERT INTO Makes VALUES( '415-123-4567', 5 );
INSERT INTO Makes VALUES( '415-222-4467', 6 );
INSERT INTO Makes VALUES( '415-222-4467', 7 );
INSERT INTO Makes VALUES( '415-123-4567', 8 );
INSERT INTO Makes VALUES( '415-143-6667', 9 );
INSERT INTO Makes VALUES( '415-143-6667', 10 );
INSERT INTO Makes VALUES( '415-123-4567', 10 );
INSERT INTO Makes VALUES( '415-143-6667', 13);
INSERT INTO Makes VALUES( '415-143-6667', 14 );

INSERT INTO Dines_In VALUES(1, '415-123-4567' );
INSERT INTO Dines_In VALUES(2 ,'415-222-4467');
INSERT INTO Dines_In VALUES(3, '415-143-6667' );
INSERT INTO Dines_In VALUES(4, '415-123-4567');
INSERT INTO Dines_In VALUES(5,'415-222-4467');
INSERT INTO Dines_In VALUES(6, '415-143-6667' );
INSERT INTO Dines_In VALUES(7, '415-123-4567');
INSERT INTO Dines_In VALUES(8, '415-222-4467' );
INSERT INTO Dines_In VALUES(9, '415-143-6667');
INSERT INTO Dines_In VALUES(10, '415-123-4567' );
INSERT INTO Dines_In VALUES(11, '415-222-4467' );
INSERT INTO Dines_In VALUES(12, '415-143-6667');
INSERT INTO Dines_In VALUES(13, '415-123-4567');
INSERT INTO Dines_In VALUES(14, '415-222-4467');
INSERT INTO Dines_In VALUES(15, '415-143-6667');
```

Task 3:

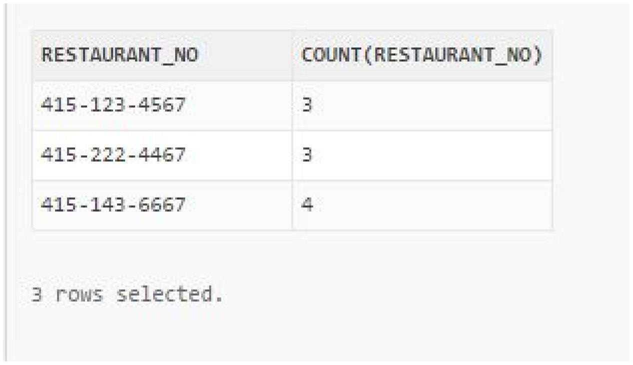
Write SQL Queries • At least 2 queries involving GROUP BY, HAVING, and aggregate operators. • At least 2 nested queries involving IN, EXIST, op ANY, op ALL…

```
SELECT item_no,count(item_no)
FROM Orders
GROUP BY item_no
HAVING count(item_no) > 2
```



ITEM_NO	COUNT(ITEM_NO)
1	3

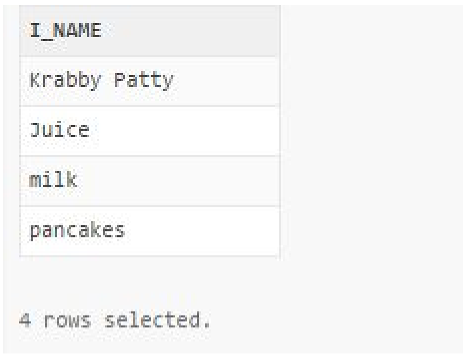
```
SELECT restaurant_no, count(restaurant_no)
FROM Dines_In
WHERE customer_no = ANY(SELECT customer_no
                    FROM Customer
                    WHERE customer_no > 5)
GROUP BY restaurant_no
Order by count(restaurant_no)
```



RESTAURANT_NO	COUNT(RESTAURANT_NO)
415-123-4567	3
415-222-4467	3
415-143-6667	4

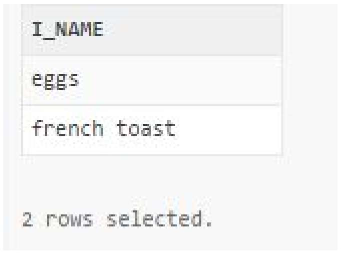
```
3 rows selected.
```

```
SELECT I1.i_name
FROM Item I1
WHERE I1.item_no IN  (SELECT M1.item_no
                FROM Makes M1
                WHERE M1.restaurant_no = '415-123-4567' and M1.item_no=I1.item_no)
```



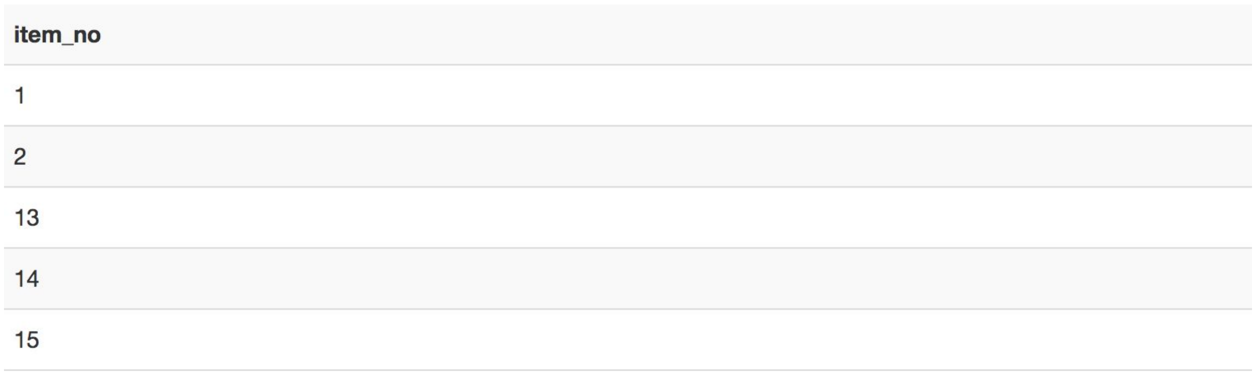
```
I_NAME
Krabby Patty
Juice
milk
pancakes

4 rows selected.
```

```
SELECT I1.i_name
FROM Item I1
WHERE Exists (select *
            FROM Makes M1
            WHERE M1.restaurant_no = '415-222-4467' and M1.item_no=I1.item_no)
```



```
I_NAME
eggs
french toast

2 rows selected.
```

```
SELECT I1.item_no
FROM Item I1
WHERE Exists(SELECT *
            FROM Bill B1
            WHERE B1.amount>60 and B1.item_no=I1.item_no)
```



item_no
1
2
13
14
15