Sample Output Queries

AKZ06DAU

CNOSM3E8

МЈВРЈСМТ

OM2HY2Z8

HH8ZHX6G

Sage Xzavier

Sergio

a. List the customers. For each customer, indicate which category he or she fall into, and his or her contact information. If you have more than one independent categorization of customers, please indicate which category the customer falls into for all of the categorizations.

select customerID, c.cFirstName, c.cLastName, m.cPhoneNum, "Member" as membership, "CoporationAccount" as Category from corporation inner join member m using(customerID) inner join customer c using(customerID) union select customerID, c.cFirstName, c.cLastName, m.cPhoneNum, "Member" as membership, "IndividualAccount" as indiv from individual inner join member m using(customerID) inner join customer c using(customerID) union select customerID, cFirstName, cLastName, NULL,"Non-Member" as membership, "Anon" as indiv from customer where customerID NOT IN (select customerID from member); ■ ■ ■ ■ ● * Max. rows: 100 Fetched Rows: 10 Matching Rows: cFirstName cLastName cPhoneNum customerID membership Category JSJAT87R (918) 910-0657 CoporationAccount Jamison Lee Member Gonzalez CoporationAccount QPTIDI3C (988) 735-9270 Member XLN1LOX4 Makhi Archer (758) 808-8112 Member CoporationAccount (227) 760-0982 IndividualAccount 3F58QIGC Lance Castillo 5TW95YKO Ryker Solomon (367) 310-1921 Member IndividualAccount

b. List the top three customers in terms of their net spending for the past **two** years, and the total that they have spent in that period.

Member

Member

Member

Member

Non-Member

(456) 960-4482

(951) 720-4059

(482) 440-6009

IndividualAccount

IndividualAccount

IndividualAccount

IndividualAccount

Anon

Mullen

Watts

Orr

Peck

Ramsey

select c.cFirstName, c.cLastName, SUM(totalAmount) as "Total Spent" from orders inner join customer c using(customerID)
where orderDate < CURDATE() and (YEAR(CURRENT_TIMESTAMP) - 2)
GROUP BY customerID
order by totalAmount DESC LiMIT 3;

c. Find all of the sous chefs who have three or more menu items that they can prepare. For each sous chef, list their name, the number of menu items that they can prepare, and each of

the menu items. You can use group_concat to get all of a given sous chef's data on one row, or print out one row per sous chef per menu item.

- d. Find all of the sous chefs who have three or more menu items in common.
- i. Please give the name of each of the two sous chefs sharing three or more menu items
 - ii. Please make sure that any given pair of sous chefs only shows up once.

Select specialty, e1.eFirstName, e1.eLastName, e2.eFirstName, e2. eLastName, count(specialty) as "In common" from specialty inner join employee e1 using (eID) inner join (select specialty, e1.eFirstName, e1.eLastName from specialty inner join employee e1 using (eID)) e2 using (specialty) group by specialty having count(specialty) >= 3; eFirstName eLastName eFirstName eLastName Amount In commor Sauteed Bok Choy Tessa Sauteed Mustard Greens with Shredde... Tessa Duncan Duncan Duncan Duncan Sauteed String Green Beans with Garlic Tessa

e. Find the three menu items most often ordered from the Children's menu and order them from most frequently ordered to least frequently ordered.

select menuID, itemName, SUM(itemPrice*quantity) as "Revenue"

from orders
natural join orderDetail
natural join menuMenuItem
natural join menuItem
where menuID = "003"
GROUP BY itemName
order by SUM(itemPrice*quantity) DESC LIMIT 3;



f. List all the menu items, the shift in which the menu item was ordered, and the sous chef on duty at the time, when the sous chef was not an expert in that menu item.

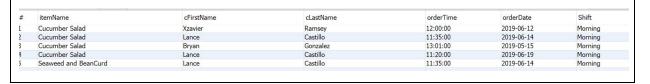
SELECT DISTINCT itemName, cFirstName, cLastName, orderTime, orderDate, IF(orderTime >= '10:00:00' AND orderTime < '16:00:00', "Morning", "Evening") AS "Shift" FROM (orders INNER JOIN orderDetail USING(orderID))

NATURAL JOIN menuItem

NATURAL JOIN customer

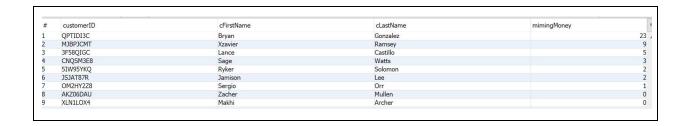
NATURAL JOIN ((employee INNER JOIN sousChef USING (eID)) INNER JOIN mentorShip using(eID))

WHERE orders.orderDate < mentorShip.startDate AND shiftType = IF(orderTime >= '10:00:00' and orderTime < '16:00:00' , "Morning", "Evening");



g. List the customers, sorted by the amount of Miming's Money that they have, from largest to smallest.

select customerID, c.cFirstName, c.cLastName, a.mimingMoney from customer c inner join account a using(customerID) order by mimingMoney DESC;



h. List the customers and the total that they have spent at Miming's ever, in descending order by the amount that they have spent.

SELECT cFirstName, cLastName, SUM(totalAmount) as "Amount Spent Last Year"
FROM customer
INNER JOIN orders USING (customerID)
GROUP BY cFirstName, cLastName
ORDER BY SUM(totalAmount) DESC;

cFirstName | cLastName | clast



i. Report on the customers at Miming's by the number of **times** that they come in by month and order the report from most frequent to the least frequent.

SELECT cFirstName, cLastName, MONTH(orderDate) AS "Month", YEAR(orderDate) AS "YEAR", COUNT(MONTH(orderDate)) AS "Visits"

FROM customer

NATURAL JOIN order

GROUP BY cFirstName, cLastName, MONTH(orderDate), YEAR(orderDate)

ORDER BY COUNT(MONTH(orderDate)) DESC



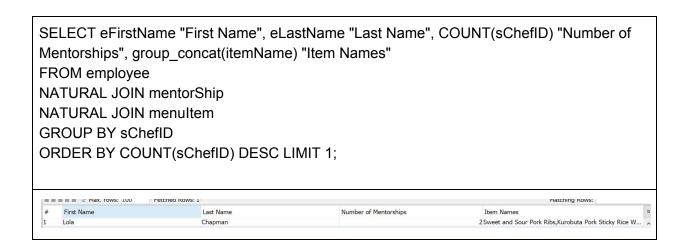
j. List the three customers who have spent the most at Miming's over the past year. Order by the amount that they spent, from largest to smallest.

SELECT cFirstName, cLastName, SUM(totalAmount) as "Amount Spent Last Year"
FROM customer
INNER JOIN orders USING (customerID)
WHERE orderDate < CURDATE() and (YEAR(CURRENT_TIMESTAMP) - 1)
GROUP BY cFirstName, cLastName
ORDER BY SUM(totalAmount) DESC LIMIT 3;

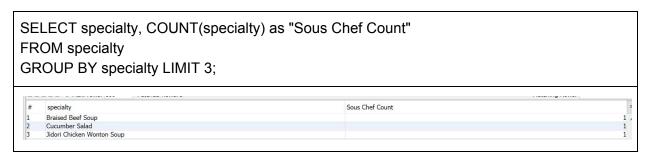
k. List the five menu items that have generated the most revenue for Miming's over the past year.

select itemName, SUM(itemPrice*quantity) as "Revenue", YEAR(orderDate) as "Year" from orders
natural join orderDetail
natural join menuMenuItem
natural join menuItem
where orderDate < CURDATE() and (YEAR(CURRENT_TIMESTAMP) - 1)
GROUP BY itemName
order by SUM(itemPrice*quantity) DESC LIMIT 5;

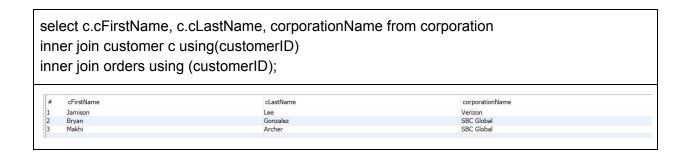
I. Find the sous chef who is mentoring the most other sous chef. List the menu items that the sous chef is passing along to the other sous chefs.



m. Find the three menu items that have the fewest sous chefs skilled in those menu items.



n. List all of the customers who eat at Miming's on their own as well as ordering for their corporation.



o. List the contents and prices of each of the menus.

SELECT menuType, itemName, itemPrice FROM cecs323sec03og6.menuMenuItem inner join aLaCarte using(menuID) inner join menuItem ORDER BY menuID;

#	menuType	itemName	itemPrice
1	Evening	Braised Beef	8.5
2	Evening	Pork Chop Suey	10.7
3	Evening	Sweet and Sour Pork Ribs	11.5
4	Evening	Braised Beef	11.1
5	Evening	Sweet and Sour Pork Ribs	11.1
6	Evening	Sauteed Bok Choy	3.5
7	Evening	Red Hot Chow Mein	7.5
8	Evening	Sauteed Bok Choy	10.9
9	Evening	Red Hot Chow Mein	10.7
10	Evening	Chicken Chop Suey	4.5
11	Evening	Chicken Chop Suey	10.7
12	Evening	Kurobuta Pork Sticky Rice Wrap	7.5
13	Evening	Jidori Chicken Wontons with Spicy Sauce	11.25
14	Evening	Kurobuta Pork Sticky Rice Wrap	10.7
15	Evening	Jidori Chicken Wontons with Spicy Sauce	10.7
16	Evening	Sauteed Mustard Greens with Shredded Ginger	10.25
17	Evening	Pork Egg Foo Young	11.25
18	Evening	Sauteed Mustard Greens with Shredded Ginger	10.7
19	Evening	Pork Egg Foo Young	10.7
20	Evening	Seafood Chop Suey	10.25
21	Evening	Stuffed Eggplant in Black Bean Sauce	8.5
22	Evening	Seafood Chop Suey	11.1
77	Consina	Duniand Dank Cours	11 75

- p. Three additional queries that demonstrate the five additional business rules. Feel free to create additional views to support these queries if you so desire.
- p1 : List out the reserved table and the party size of the table

select e.orderID, e.tableNum, partySize, reservationStat from eatIn e inner join tableInfo using (orderID) where reservationStat = "Reserved";

orderID | tableNum | partySize | reservationStat | reservationStat | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 1202190100 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 120219000 | 12021900 | 12021900 | 12021900 | 12021900 | 12021900 | 1202

p2 : Show the orders that are split by card where the minimum is \$15

SELECT o.orderID, o.orderDate AS "Date", o.totalAmount AS "Total", o.gratuityGiven AS "Tip"

FROM orders o INNER JOIN payment p ON p.orderID = o.orderID INNER JOIN card c ON c.paymentNum = p.paymentNum WHERE c.minCharge = '15' AND c.paymentSplit > 1;

orderID	Date	Total	Tip	
1202190100	2019-06-12		92.3	18.4
1202190102	2019-06-14		39.98	6.

p3: List out the parties being charged gratuity for parties 6 or more during the sunday brunch.

 ${\tt SELECT~c.customerID,partySize,~gratuityGiven,~od.menuID,~group_concat(fixedRate)~as~"Charge~Rate"}$

, group_concat(ageRange) as "Age Range"

FROM customer c INNER JOIN orders o ON c.customerID = o.customerID

INNER JOIN orderDetail od ON od.orderID = o.orderID

INNER JOIN eatln e on od.orderID = e.orderID

INNER JOIN buffet

WHERE od.menuID = '004' AND partySize >= 6 AND o.gratuityGiven >0;

