

Sample Output Queries

- 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, cFirstNme, 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:
#	customerID	cFirstName	cLastName	cPhoneNum	membership	Category
1	JSJAT87R	Jamison	Lee	(918) 910-0657	Member	CoporationAccount
2	QPTIDI3C	Bryan	Gonzalez	(988) 735-9270	Member	CoporationAccount
3	XLN1LOX4	Makhi	Archer	(758) 808-8112	Member	CoporationAccount
4	3F58QIGC	Lance	Castillo	(227) 760-0982	Member	IndividualAccount
5	5IW95YKQ	Ryker	Solomon	(367) 310-1921	Member	IndividualAccount
6	AKZ06DAU	Zacher	Mullen	(577) 991-9765	Member	IndividualAccount
7	CNQSM3E8	Sage	Watts	(456) 960-4482	Member	IndividualAccount
8	MJBPPJCMT	Xzavier	Ramsey	(951) 720-4059	Member	IndividualAccount
9	OM2HY2Z8	Sergio	Orr	(482) 440-6009	Member	IndividualAccount
10	HH8ZHX6G	Elvis	Peck	<None>	Non-Member	Anon

- 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.

```
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;
```

Max. rows: 100 Fetched Rows: 3				Matching Rows:
#	cFirstName	cLastName	Total Spent	
1	Bryan	Gonzalez	234.05	
2	Xzavier	Ramsey	184.6	
3	Lance	Castillo	61.63	

- 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.

```
SELECT eLastName, eFirstName, Group_Concat(Specialty) as "Specialties",
COUNT(eID) AS "# of Menu Items"
FROM specialty
NATURAL JOIN employee
GROUP BY eID
HAVING COUNT(Specialty) >= 3;
```

#	eLastName	eFirstName	"Specialties"	# of Menu Items
1	Kennedy	Verity	Braised Beef Soup,Jidori Chicken Wonton Soup,Shrimp & Ku...	4

- d. Find all of the sous chefs who have three or more menu items in common.
- Please give the name of each of the two sous chefs sharing three or more menu items
 - 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;
```

#	specialty	eFirstName	eLastName	eFirstName	eLastName	Amount In common
1	Sauteed Bok Choy	Tessa	Duncan	Tessa	Duncan	4
2	Sauteed Mustard Greens with Shredde...	Tessa	Duncan	Tessa	Duncan	4
3	Sauteed String Green Beans with Garlic	Tessa	Duncan	Tessa	Duncan	4

- 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;

```

#	menuID	itemName	Revenue
1	003	Braised Beef Soup	30.0
2	003	Chicken & Mushroom Bun	7.25
3	003	Beef Chow Mein	7.1

- 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");

```

#	itemName	cFirstName	cLastName	orderTime	orderDate	Shift
1	Cucumber Salad	Xavier	Ramsey	12:00:00	2019-06-12	Morning
2	Cucumber Salad	Lance	Castillo	11:35:00	2019-06-14	Morning
3	Cucumber Salad	Bryan	Gonzalez	13:01:00	2019-05-15	Morning
4	Cucumber Salad	Lance	Castillo	11:20:00	2019-06-19	Morning
5	Seaweed and BeanCurd	Lance	Castillo	11:35:00	2019-06-14	Morning

- 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;

```

#	customerID	cFirstName	cLastName	mimingMoney
1	QPTIDI3C	Bryan	Gonzalez	23
2	MJBPJCMT	Xzavier	Ramsey	9
3	3F58QJGC	Lance	Castillo	5
4	CNQSM3E8	Sage	Watts	3
5	5IW95YKQ	Ryker	Solomon	2
6	JSJAT87R	Jamison	Lee	2
7	OM2HY2Z8	Sergio	Orr	1
8	AKZ06DAU	Zacher	Mullen	0
9	XLN1LOX4	Makhi	Archer	0

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	Amount Spent Last Year
1	Bryan	Gonzalez	234.05
2	Xzavier	Ramsey	184.6
3	Lance	Castillo	61.63
4	Sage	Watts	33.98
5	Ryker	Solomon	29.95
6	Jamison	Lee	27.75
7	Elvis	Peck	22.9
8	Sergio	Orr	15.75
9	Makhi	Archer	9.25
10	Zacher	Mullen	9.1

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
```

#	cFirstName	cLastName	Month	YEAR	Visits	10
1	Lance	Castillo	6	2019	2	^
2	Sage	Watts	6	2019	1	
3	Xzavier	Ramsey	2	2019	1	
4	Elvis	Peck	6	2019	1	
5	Sergio	Orr	6	2019	1	
6	Ryker	Solomon	5	2019	1	
7	Jamison	Lee	5	2019	1	
8	Bryan	Gonzalez	5	2019	1	
9	Zacher	Mullen	7	2019	1	
10	Xzavier	Ramsey	6	2019	1	
11	Makhi	Archer	7	2019	1	

- 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;

```

#	cFirstName	cLastName	Amount Spent Last Year	10
1	Bryan	Gonzalez	234.05	^
2	Xzavier	Ramsey	184.6	
3	Lance	Castillo	61.63	

- 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;

```

#	itemName	Revenue	Year	10
1	Vegetarian Noodle Soup	65.0	2019	^
2	Cucumber Salad	58.5	2019	
3	Braised Beef Soup	52.5	2019	
4	Shrimp Fried Rice	41.0	2019	
5	Kurobuta Pork Xiao Long Bao	37.6	2019	

l. 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.

```
SELECT eFirstName "First Name", eLastName "Last Name", COUNT(sChefID) "Number of
Mentorships", group_concat(itemName) "Item Names"
FROM employee
NATURAL JOIN mentorShip
NATURAL JOIN menuItem
GROUP BY sChefID
ORDER BY COUNT(sChefID) DESC LIMIT 1;
```

#	First Name	Last Name	Number of Mentorships	Item Names
1	Lola	Chapman		2 Sweet and Sour Pork Ribs, Kurobuta Pork Sticky Rice W...

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

```
SELECT specialty, COUNT(specialty) as "Sous Chef Count"
FROM specialty
GROUP BY specialty LIMIT 3;
```

#	specialty	Sous Chef Count
1	Braised Beef Soup	1
2	Cucumber Salad	1
3	Jidori Chicken Wonton Soup	1

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

```
select c.cFirstName, c.cLastName, corporationName from corporation
inner join customer c using(customerID)
inner join orders using (customerID);
```

#	cFirstName	cLastName	corporationName
1	Jamison	Lee	Verizon
2	Bryan	Gonzalez	SBC Global
3	Makhi	Archer	SBC Global

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
23	Evening	Braised Beef	11.25

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
1	1202190100		64	8 Reserved

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.gratuuityGiven 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
1	1202190100	2019-06-12	92.3	18.46
2	1202190102	2019-06-14	39.98	6.0

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

```

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 eatIn e on od.orderID = e.orderID
INNER JOIN buffet
WHERE od.menuID = '004' AND partySize >= 6 AND o.gratuityGiven >0;

```

#	customerID	partySize	gratuityGiven	menuID	Charge Rate	Age Range
1	MJBPCMT	6	50.46	004	0.00,19.99,13.99	0-2,14+,3-13