/\*

1. 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 Customer.custID, 'N/A' as CustomerName, 'ANYONYMOUS CUSTOMER' AS `CUSTOMERTYPE`

From Customer

WHERE Customer.custID NOT IN (SELECT DISTINCT custID FROM PaidCustomer)

UNION

SELECT Customer.custID, customerName, 'REGISTERED CUSTOMER' AS `CUSTOMERTYPE`

FROM Customer

INNER JOIN PaidCustomer ON Customer.custID = PaidCustomer.custID

WHERE Customer.custID IN (SELECT DISTINCT custID FROM PaidCustomer) AND PaidCustomer.corpName IS NULL

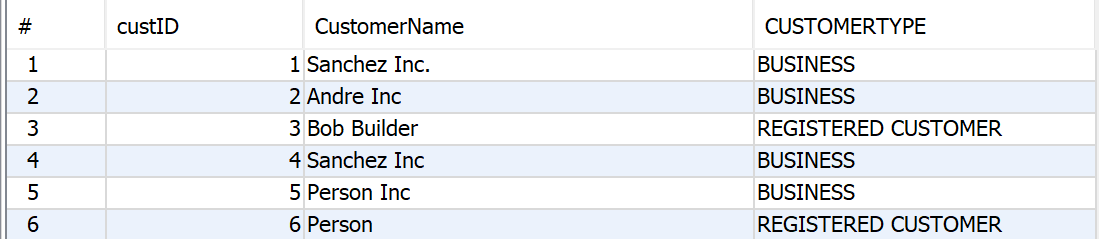
UNION

SELECT Customer.custID, corpName, 'BUSINESS' AS `CUSTOMERTYPE`

FROM Customer

INNER JOIN PaidCustomer ON Customer.custID = PaidCustomer.custID

WHERE Customer.custID IN (SELECT DISTINCT custID FROM PaidCustomer) AND PaidCustomer.corpName IS NOT NULL;



/\*

2. 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 PaidCustomer.customerName, SUM(`Check`.total - `Check`.mimingsMoney) AS "Net Spending"

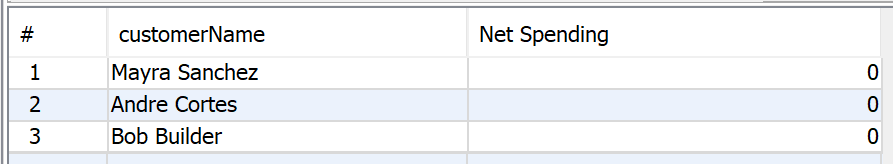
FROM `Check` INNER JOIN PaidCustomer USING (custID)

WHERE `Check`.date BETWEEN (CURDATE() - INTERVAL 2 YEAR) AND CURDATE()

GROUP BY `Check`.custID

ORDER BY "Net Spending" DESC

LIMIT 3;



/\*

3. 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 Employee.eFirstName AS "Sous Chef First Name", Employee.eLastName AS "Sous Chef Last Name", Level99Chef.item, Level99Chef.spice, Sums.eS AS "Number of Items"

FROM Level99Chef INNER JOIN Employee ON Employee.employeeID = Level99Chef.sousChef

INNER JOIN (

SELECT Counts.sousChef, SUM(Counts.eCounts) AS eS

FROM (

SELECT Level99Chef.sousChef, Level99Chef.item, COUNT(Level99Chef.spice) As eCounts

FROM Level99Chef

GROUP BY Level99Chef.sousChef, Level99Chef.item) AS Counts

GROUP BY Counts.sousChef

HAVING eS > 2) AS Sums ON Employee.employeeID = Sums.sousChef

WHERE Employee.employeeID IN (

SELECT Counts.sousChef

FROM (

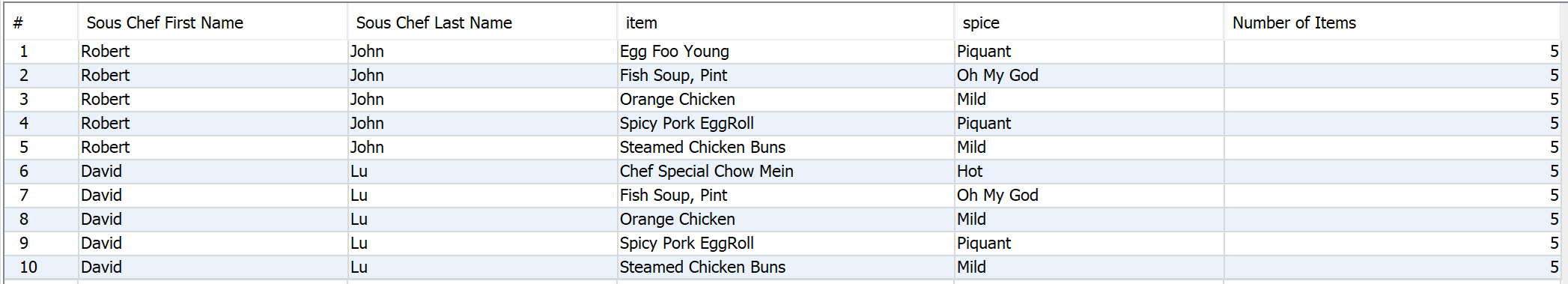
SELECT Level99Chef.sousChef, Level99Chef.item, COUNT(Level99Chef.spice) As eCounts

FROM Level99Chef

GROUP BY Level99Chef.sousChef, Level99Chef.item) AS Counts

GROUP BY Counts.sousChef

HAVING SUM(Counts.eCounts) > 2);



/\*

4. 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 Employee1.eFirstName AS "Sous Chef 1 First Name", Employee1.eLastName AS "Sous Chef 1 Last Name",

Employee2.eFirstName AS "Sous Chef 2 First Name", Employee2.eLastName AS "Sous Chef 2 Last Name",

SUM(spiceCount) AS "Shared Expertise"

FROM (

SELECT Level99Chef1.sousChef AS SC1, Level99Chef2.sousChef AS SC2, Level99Chef1.item, COUNT(Level99Chef1.spice) AS spiceCount

FROM Level99Chef AS Level99Chef1 INNER JOIN Level99Chef AS Level99Chef2

ON Level99Chef1.item = Level99Chef2.item AND Level99Chef1.spice = Level99Chef2.spice

WHERE Level99Chef1.sousChef < Level99Chef2.sousChef

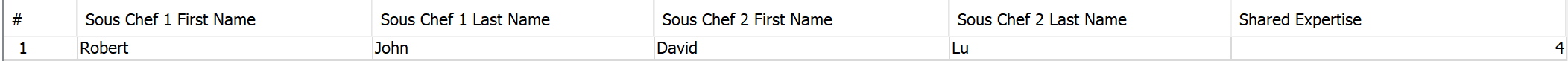
GROUP BY Level99Chef1.sousChef, Level99Chef2.sousChef, Level99Chef1.item) AS Counts

INNER JOIN Employee AS Employee1 ON Employee1.employeeID = Counts.SC1

INNER JOIN Employee AS Employee2 ON Employee2.employeeID = Counts.SC2

GROUP BY "Sous Chef 1 First Name", "Sous Chef 1 Last Name", "Sous Chef 2 First Name", "Sous Chef 2 Last Name"

HAVING SUM(spiceCount) > 2;



/\*

5. 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 menuType, item, spice, SUM(quantity) AS "Number of times ordered"

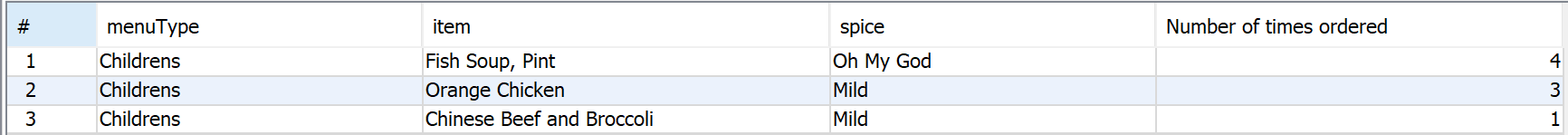
FROM OrderDetail

WHERE menuType = 'Childrens'

GROUP BY item, spice

ORDER BY SUM(quantity) DESC

LIMIT 3;



/\*

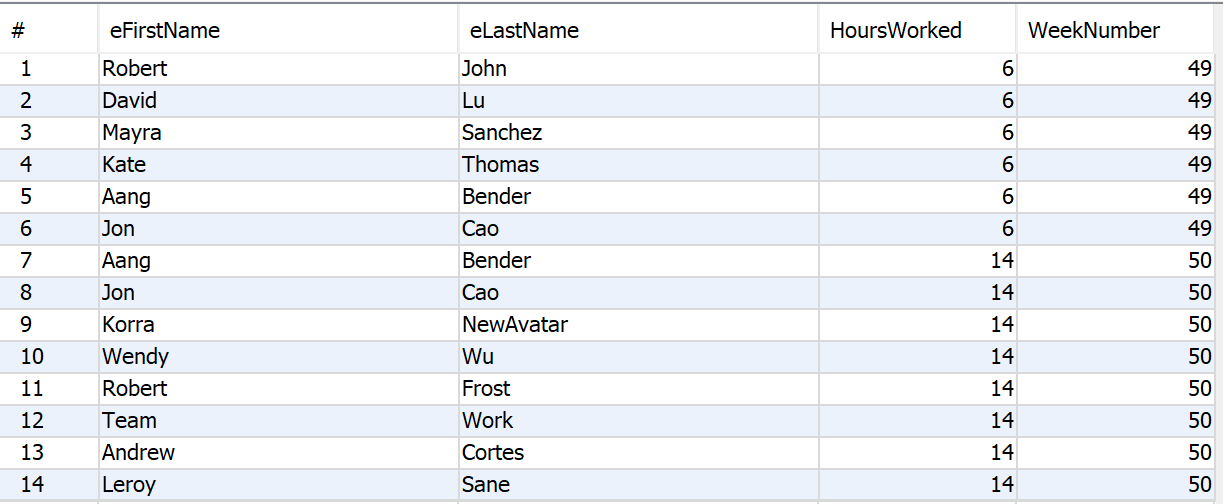
6) Show by week, how many hours each employee works

\*/

select eFirstName,eLastName,(hour(endTime)-hour(startTIme)) as HoursWorked, weekNum as WeekNumber

from Shift natural join EmployeeShift natural join Employee

order by weekNum;



/\*

7. List the customers, sorted by the amount of Miming’s Money that they have, from largest

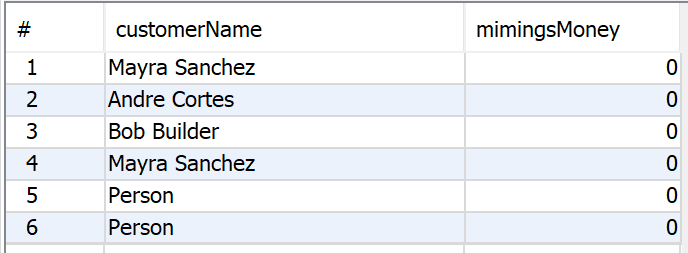
to smallest.

\*/

SELECT customerName, mimingsMoney

FROM PaidCustomer

ORDER BY mimingsMoney DESC;



/\*

8. List the customers and the total that they have spent at Miming’s ever, in descending

order by the amount that they have spent.

\*/

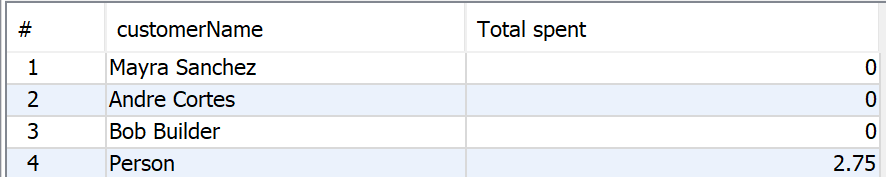
SELECT PaidCustomer.customerName, SUM(`Check`.total) AS "Total spent"

FROM PaidCustomer

INNER JOIN `Check` USING(custID)

GROUP BY PaidCustomer.customerName

ORDER BY "Total spent" DESC;



/\*

9. 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 MonthlyVisits.custID, MonthlyVisits.customerName, AVG(Visits) AS "AVG monthly visits"

FROM (

SELECT PaidCustomer.custID, PaidCustomer.customerName, YEAR(date) AS "Year", MONTH(date) AS "Month", COUNT(date) AS "Visits"

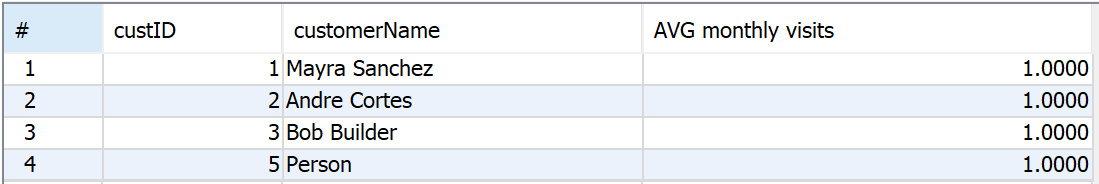
FROM PaidCustomer

INNER JOIN `Check` USING(custID)

GROUP BY PaidCustomer.custID, PaidCustomer.customerName, YEAR(date), MONTH(date)) AS MonthlyVisits

GROUP BY MonthlyVisits.customerName

ORDER BY AVG(Visits) DESC;



/\*

10. 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 PaidCustomer.customerName, SUM(`Check`.total) AS "Spent over last year"

FROM PaidCustomer

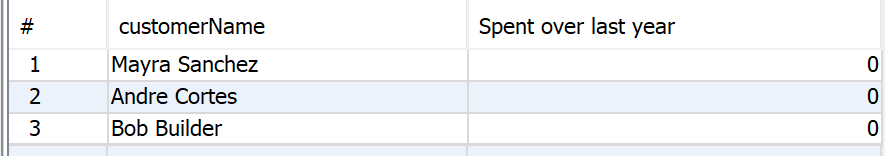
INNER JOIN `Check` USING(custID)

WHERE `Check`.date BETWEEN (CURDATE() - INTERVAL 1 YEAR) AND CURDATE()

GROUP BY PaidCustomer.customerName

ORDER BY "Spent over last year" DESC

LIMIT 3;



/\*

11. List the five menu items that have generated

the most revenue for Miming’s over the past year.\*/

Select Menu.item, sum(price\*OrderDetail.quantity) `Total Revenue`

from Menu

inner join

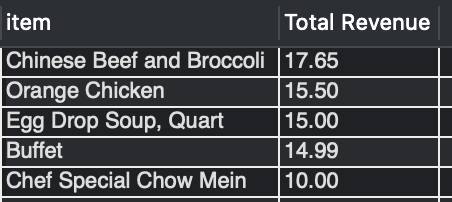
OrderDetail on OrderDetail.item = Menu.item

Where OrderDetail.menuType = `Menu`.menuType and OrderDetail.spice = `Menu`.spice

group by Menu.item

ORDER BY `Total Revenue` DESC

LIMIT 5;



/\*

12. 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 Mentors.eFirstName AS "Mentor First Name", Mentors.eLastName AS "Mentor Last Name",

Mentees.eFirstName AS "Mentee First Name", Mentees.eLastName AS "Mentee Last Name",

Mentorship.item, Mentorship.spice

FROM Mentorship INNER JOIN Employee AS Mentors ON Mentorship.sousChefMentor = Mentors.employeeID

INNER JOIN Employee AS Mentees ON Mentorship.sousChefMentee = Mentees.employeeID

WHERE Mentorship.sousChefMentor = (

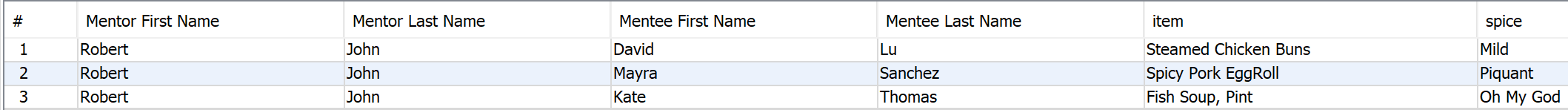
SELECT Mentorship.sousChefMentor

FROM Mentorship

GROUP BY Mentorship.sousChefMentor

ORDER BY COUNT(Mentorship.sousChefMentee) DESC

LIMIT 1);



/\*

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

\*/

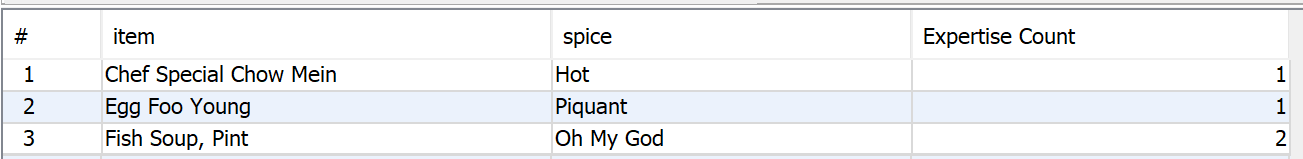
SELECT Level99Chef.item, Level99Chef.spice, COUNT(sousChef) AS "Expertise Count"

FROM Level99Chef

GROUP BY Level99Chef.item, Level99Chef.spice

ORDER BY COUNT(sousChef)

LIMIT 3;



/\*

14. List all of the customers who eat at Miming’s on their own as well as ordering for their

corporation. \*/

SELECT customerName, email, custAddress

FROM PaidCustomer

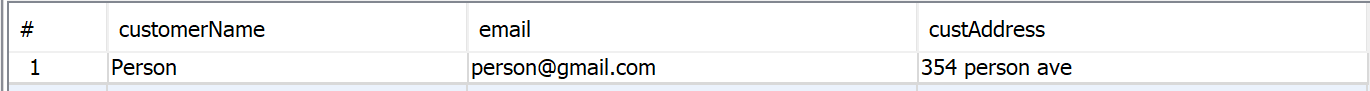
WHERE corpName IS NOT NULL

AND PaidCustomer.customerName IN

(SELECT PaidCustomer.customerName FROM PaidCustomer WHERE corpName IS NULL)

AND PaidCustomer.email IN

(SELECT PaidCustomer.email FROM PaidCustomer WHERE corpName IS NULL);



/\*

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

\*/

SELECT menuType, item, spice, price

FROM Menu

ORDER BY menuType, item, spice;



/\*

16. Three additional queries that demonstrate the five additional business rules. Feel free to

create additional views to support these queries if you so desire.

\*/

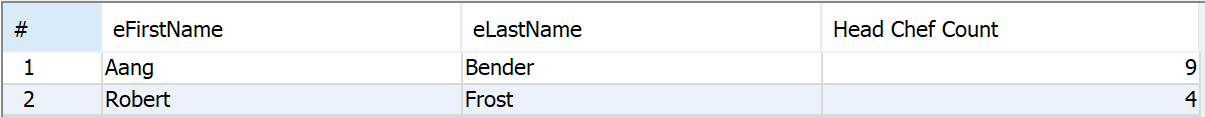
-- A head chef can have only make 9 recipes

SELECT Employee.eFirstName, Employee.eLastName, COUNT(headChef) AS 'Head Chef Count'

FROM Item

INNER JOIN Employee ON Item.headChef = Employee.employeeID

GROUP BY headChef;



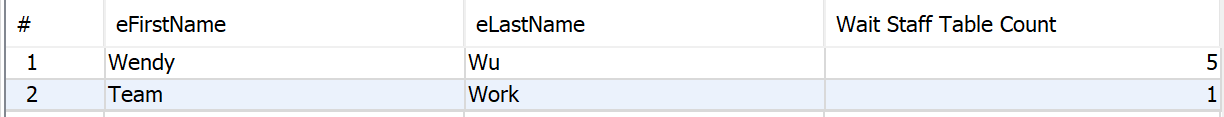
-- A given wait staff member can only wait on up to 5 tables at a time

SELECT Employee.eFirstName, Employee.eLastName, COUNT(TableShift.employeeID) AS 'Wait Staff Table Count'

FROM TableShift

INNER JOIN Employee ON TableShift.employeeID = Employee.employeeID

GROUP BY TableShift.employeeID;



-- A sous chef cannot mentor more than 3 people at a time

SELECT Employee.eFirstName, Employee.eLastName, COUNT(Mentorship.sousChefMentor) AS 'Sous Chef Mentor Count'

FROM Mentorship

INNER JOIN Employee ON Mentorship.sousChefMentor = Employee.employeeID

GROUP BY Mentorship.sousChefMentor;

