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 c.customer\_name, c.customer\_email, c.street\_name\_number, c.zipcode,

CASE

c.customer\_name

WHEN

c.customer\_name IN (SELECT m.customer\_name FROM mimi\_business m

NATURAL

JOIN customer c WHERE m.customer\_name = c.customer\_name) THEN "Individual"

ELSE

"Corporate"

END AS

customer\_type

FROM

customer AS c;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **customer\_name** | **customer\_email** | | **street\_name\_number** | **zipcode** | **customer\_type** |
| Big Spender | | abcdef@gmail.com | 283 Elm St. | 16519 | Individual |
| Clayton Kershaw | | abcdef@gmail.com | 5161 Garfield Ave | 51574 | Individual |
| James Harden | | abcdef@gmail.com | 763 1st St. | 54374 | Individual |
| John Smith | | john\_smith@gmail.com | 123 Main St. | 12345 | Corporate |
| Pedro Biaz | | abcdef@gmail.com | 65161 Paramount Blvd | 54344 | Individual |
| Peggy Sue | | abcdef@gmail.com | 21 Jump Street | 44374 | Individual |
| Sum Dude | | abcdef@gmail.com | 11601 Some Street | 23456 | Individual |
|  |  |  |  |  |  |

2) List the top three customers in terms of their net spending for the past two years (last 730 days), and the total that they have spent in that period.

SELECT NAME, TOTAL from

(SELECT

customer.customer\_name AS "NAME", SUM(`order`.total) AS "TOTAL"

FROM

customer INNER JOIN `order` on customer.customer\_name = `order`.customer\_name

GROUP BY

customer.customer\_name

order by

TOTAL desc) descTot

limit 3;

|  |  |
| --- | --- |
| **NAME** | **TOTAL** |
| Big Spender | 1433.07 |
| Clayton Kershaw | 1314.14 |
| John Smith | 445.2 |

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 fname, lname, "Mastered Dishes Count", dish\_name from EMPLOYEE

INNER JOIN

(SELECT

empid, count(dish\_name) AS "Mastered Dishes Count" from

sous\_chef\_dish

group by

empid

HAVING

count(dish\_name) >= 3

) dishMaster

USING(EMPID)

INNER JOIN

SOUS\_CHEF\_DISH

USING(empid);

|  |  |  |  |
| --- | --- | --- | --- |
| **FNAME** | **LNAME** | **Mastered Dishes Count** | **DISH\_NAME** |
| Abraham | Lincoln | 3 | Burger |
| Abraham | Lincoln | 3 | Spagetti |
| Abraham | Lincoln | 3 | Tacos |

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.

iii. Please list the items that the two Sous Chefs have in common. Again, you can use group\_concat to get all of those items into one value in the output.

select employee1.fname, employee1.lname, employee2.fname, employee2.lname, dish\_name from

employee

employee1

inner join

(select

`EMP1 ID` AS `Employee 1`, `EMP2 ID` AS `Employee 2`, count(dish\_name) as Dish

from

(select

emp1.empid AS `EMP1 ID`, emp2.empid AS `EMP2 ID`, emp1.dish\_name from

(select

distinct empid, dish\_name

from

sous\_chef\_dish

inner

join employee

using(empid))

emp1

inner join

(select

distinct empid, dish\_name

from

sous\_chef\_dish

inner

join employee

using(empid))

emp2

USING(dish\_name)

where

emp1.empid < emp2.empid) common

group by

`EMP1 ID`, `EMP2 ID`

having

count(dish\_name) >= 3) someName

on

employee1.EMPID = someName.`Employee 1`

inner

join employee employee2

on

employee2.EMPID = someName.`Employee 2`

inner

join

sous\_chef\_dish

scd

on

employee1.EMPID = scd.EMPID;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FNAME** | **LNAME** | **FNAME** | **LNAME** | **DISH\_NAME** |
| Abraham | Lincoln | Barack | Obama | Burger |
| Abraham | Lincoln | Barack | Obama | Spagetti |
| Abraham | Lincoln | Barack | Obama | Tacos |

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 kids\_menu.pic\_to\_color, COUNT(bill\_item.dish\_name)

FROM kids\_menu NATURAL JOIN menu NATURAL JOIN menu\_to\_item NATURAL JOIN menu\_item

NATURAL JOIN bill\_item

GROUP BY kids\_menu.pic\_to\_color

ORDER BY COUNT(bill\_item.dish\_name) DESC;

|  |  |
| --- | --- |
| **pic\_to\_color** | **COUNT(bill\_item.dish\_name)** |
| Juicy Cow | 1 |
| Fluffy Bunny | 1 |

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

select fname, lname, week(shift\_date) as week, sum(hour(subtime(end, start))) as hours\_worked

from scheduled\_shift natural join employee

group by fname, lname;

|  |  |  |  |
| --- | --- | --- | --- |
| **fname** | **lname** | **week** | **hours\_worked** |
| George | Washington | 9 | 4 |
| Kevin | Durant | 9 | 13 |
| John | Kennedy | 9 | 5 |
| Abraham | Lincoln | 9 | 4 |
| Barack | Obama | 9 | 4 |

7) List the customers, sorted by the amount of Miming`s Money that they have, from largest to smallest.

Select customer\_name, current\_balance from customer

order by

current\_balance desc;

|  |  |
| --- | --- |
| **CUSTOMER\_NAME** | **CURRENT\_BALANCE** |
| James Harden | 23 |
| Peggy Sue | 1 |
| Pedro Biaz | 0 |
| Clayton Kershaw | 0 |
| Big Spender | 0 |
| Sum Dude | 0 |
| John Smith | 0 |

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 customer\_name, total from `order`

where

payment\_name = "Miming Money";

|  |  |
| --- | --- |
| **CUSTOMER\_NAME** | **TOTAL** |
| Big Spender | 100 |

9) Report on the customers at Miming`s by the number of timesthat they come in by month and order the report from most frequent to the least frequent. Each row in the output should have the Customer name, the month, the year, and the number of times that customer came in during that month of that year.

SELECT `order`.customer\_name, MONTH(`order`.date\_ordered), YEAR(`order`.date\_ordered), COUNT(\*)

FROM `order`

GROUP BY `order`.customer\_name;

|  |  |  |  |
| --- | --- | --- | --- |
| customer\_name | MONTH(`order`.date\_ordered) | YEAR(`order`.date\_ordered) | COUNT(\*) |
| Big Spender | 6 | 2019 | 3 |
| Clayton Kershaw | 7 | 2019 | 1 |
| James Harden | 10 | 2019 | 1 |
| John Smith | 9 | 2019 | 2 |
| Pedro Biaz | 1 | 2020 | 1 |
| Peggy Sue | 2 | 2020 | 1 |
| Sum Dude | 3 | 2020 | 1 |

10) List the three customers who have spent the most at Miming`s over the past year (365 days). Order by the amount that they spent, from largest to smallest.

SELECT customer.customer\_name AS "NAME", SUM(`order`.total) AS "TOTAL", date\_ordered as "order date"

FROM

customer INNER JOIN `order` on customer.customer\_name = `order`.customer\_name

where

`order`.date\_ordered > year(now()) -1 and `order`.date\_ordered

< NOW()

GROUP BY

customer.customer\_name

order by

TOTAL desc

limit 3;

|  |  |  |
| --- | --- | --- |
| **NAME** | **TOTAL** | **order date** |
| Big Spender | 1433.07 | 6/10/2019 |
| Clayton Kershaw | 1314.14 | 7/30/2019 |
| John Smith | 395.19 | 9/11/2019 |

11) List the five menu items that have generated the most revenue for Miming`s over the past year (365 days).

select dish\_name, round(sum(price), 2) as revenue

from bill\_item

natural join menu\_item natural join menu\_to\_item

where date\_ordered between

DATE\_SUB(now(), INTERVAL 1 year)

and

now()

group by dish\_name

order by sum(price) desc

limit 5;

|  |  |
| --- | --- |
| **dish\_name** | **revenue** |
| Burger | 35.96 |
| Spagetti | 14.99 |
| Tacos | 12.99 |

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 fname, lname from employee

inner join

(select sous\_teacher from mentorship

group by sous\_teacher

having count(sous\_teacher) =

(select max(`Teaching`) from

(select sous\_teacher, count(sous\_teacher) AS `Teaching` from mentorship

group by sous\_teacher

order by count(sous\_teacher)) teach)) teach2

on sous\_teacher = empid;

|  |  |
| --- | --- |
| **FNAME** | **LNAME** |
| Barack | Obama |
| David | Brown |
| LeBron | James |

(Three tied for first, all teaching one)

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

select dish\_name, count(dish\_name) AS "Dish Masters" from sous\_chef\_dish

group by dish\_name

order by count(dish\_name) asc

limit 3;

|  |  |
| --- | --- |
| **DISH\_NAME** | **Dish Masters** |
| Tacos | 2 |
| Burger | 2 |
| Spagetti | 4 |

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

select customer\_name from customer

natural join mimi\_business;

|  |
| --- |
| **CUSTOMER\_NAME** |
| John Smith |

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

select menu\_type, price, dish\_name

from menu\_to\_item

order by menu\_type;

|  |  |  |
| --- | --- | --- |
| **MENU\_TYPE** | **PRICE** | **DISH\_NAME** |
| Breakfast | 8.99 | Burger |
| Dinner | 14.99 | Spagetti |
| Lunch | 12.99 | Tacos |

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.

/\*Demonstrates business rule 1 and 2 showing that each waitstaff employee has only 1 section

and each section has at most 5 tables\*/

select fname, lname, count(sectionID) as waitstaff\_section\_count, sectionID, count(tableID)

from employee natural join part\_time natural join wait\_staff natural join section natural join `Table`

where employee.empID = wait\_staff.empID

group by fname, lname;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **fname** | **lname** | **waitstaff\_section\_count** | **sectionID** | **count(tableID)** |
| Kobe | Bryant | 2 | 1 | 2 |
| John | Kennedy | 2 | 2 | 2 |

/\*Demonstrates business rule 3 showing that each station has at most 2 line cooks\*/

select station\_name, count(empID)

from station\_assignment

group by station\_name;

|  |  |
| --- | --- |
| **station\_name** | **count(empID)** |
| Entree Station | 1 |
| Salad Station | 1 |
| Soup Station | 1 |

/\*Demonstrates business rule 4 showing that each line cook can have at most 3 stations assigned to them\*/

select fname, lname, ifnull(count(station\_name), 0) as station\_count

from station\_assignment

left outer join line\_cook on station\_assignment.empID = line\_cook.empID

left outer join full\_time on line\_cook.empID = full\_time.empID

left outer join employee on full\_time.empID = employee.empID

where line\_cook.empID = employee.empID

group by fname, lname;

|  |  |  |
| --- | --- | --- |
| **fname** | **lname** | **station\_count** |
| Jorge | Bush | 3 |

/\*Demonstrates business rule 5 showing that each shift has at least 1 maitre d\*/

select count(scheduled\_shift.empID) as maitre\_d\_count, shift\_date, start, end

from scheduled\_shift natural join employee natural join part\_time natural join maitre\_d

where scheduled\_shift.empID = maitre\_d.empID

group by shift\_date, start, end;

|  |  |  |  |
| --- | --- | --- | --- |
| **maitre\_d\_count** | **shift\_date** | **start** | **end** |
| 1 | 3/4/2020 | 8:00:00 | 12:00:00 |
| 1 | 3/4/2020 | 12:00:00 | 17:00:00 |
| 1 | 3/4/2020 | 17:00:00 | 21:00:00 |