SQL Project

Step 1. Get The Original Tables, and all of its attributes:

We looked through all the csv files given, and just copy and paste all the attributes name as the followings:

- Chefmozaaccepts
  - placeID, Rpayment
- Chefmozcuisine
  - placeID, Rcuisine
- Chefmozhouers4
  - placeID, hours, days
- Chefmozparking
  - placeID, parking_lot
- Geoplaces2
  - placeID, latitude, longitude, the_geom_meter, name, address, city, state, country, fax, zip, alcohol, smoking_area, dress_code, accessibility, price, url, Rambience, franchise, area, other_service
- Rating_final
  - userID, placeID, rating, food_rating, service_rating
- Usercuisine
  - userID, Rcuisine
- Userpayment
  - userID, Upayment
- Userprofile
  - UserID, latitude, longitude, smoker, drink_level, dress_preference,
  - Ambience,transport,marital_status,hijos,birth_year,interest,
  - Personality, religion,activity, color, weight, budget, height

Step 2: Come up with a set of Functional Dependencies.

All the relations have to be 3NF: All other attributes depend on nothing but on all candidate keys;

Relations without any functional dependency: the Chefmozaaccepts.csv, mapped to a relation called Chefmozaaccepts if we looked at its attributes: it only has placeID and rPayment. However, it is a relation with all keys: placeID and rpayment are both not unique on their own, and they form a composite key. Therefore, there are no non-key attributes that can be determined from the keys, and there is no FD.

- Chefmozaaccepts

    - No non-key attributes, no FD

- Chefmozcuisine
    - No non-key attributes, no FD

(For relation Chefmozhouers4, placeID itself cannot determine hours, but together with days they can. PlaceID and days together are the keys that functionally determined hours)

- Chefmozhouers4
    - placeID, days -> hours

- Chefmozparking
    - No non-key attributes, no FD

(create new relation Geocooridnates, because each unique pair of latitude and longitude can determine other attributes such as address, so it violate 3NF because all the non-key attributes must be depended on only keys. We will take them out)

- Geocoordinates
    - placeID -> {latitude, longitude}

(For addresses, we believe with the unique combinations of latitude and longitude given above, we can determine address, zip, city. We follow the convention that people usually fill out their address along with zip and city. Sometimes when we have address, we cannot determined the zip, or there are multiple zips with different address. For zip, there are at least one zip in a city, and there can be cities with same name. Therefore, we assume that there is no FD between the non-key attributes address, city, and zip; and they all have to be determined by both latitude nad longitude)

- Address
    - {latitude, longitude} -> {address, city, zip}

(Then, we will have the city to determined the state, and the state to determines the country)

- CityState
    - city -> state

- StateCountry
    - State -> country

- Geoplaces2
    - placeID -> {the_geom_meter, name, fax, alcohol, smoking_area, dress_code, accessibility, price, url, Rambience, franchise, area, other_service}

- Rating_final
    - userID, placeID -> {rating, food_rating, service_rating}
- Usercuisine
    - No non-key attributes, no FD
- Userpayment
    - No non-key attributes, no FD

(create new relation Usercoordinate, because each unique pair of latitude and longitude can determine other places)

- Usercoordinate
    - UserID -> {latitude, longitude}
- Userprofile
    - UserID -> {smoker, drink_level, dress_preference,
    - Ambience, transport, marital_status, hijos, birth_year, interest,
    - Personality, religion,activity, color, weight, budget, height}

Step 3. Creating the schema

Underlined attributes are all the keys, that determine all other attributes. Non-underlined attributes must be determined by all underlined attributes. If all attributes are underlined (are keys), there is no FD for that relation.

Chefmozaaccepts

placeID	Rpayment

Chefmozcuisine

placeID	Rcuisine

Chefmozhouers4

placeID	days	hours

Chefmozparking

placeID	parking_lot

Geocoordinates

placeID	latitude	longitude

Address

latitude	longitude	address	city	zip

CityState

city	state

StateCountry

state	country

Geoplaces2

placeID	the_geom_meter	name	fax	alcohol	smoking_area
dress_code	accessibility	price	url	Rambience	franchise
area	other_service				

Rating_final

userID	placeID	rating	food_rating	service_rating

Usercuisine

userID	Rcuisine

Userpayment

userID	Upayment

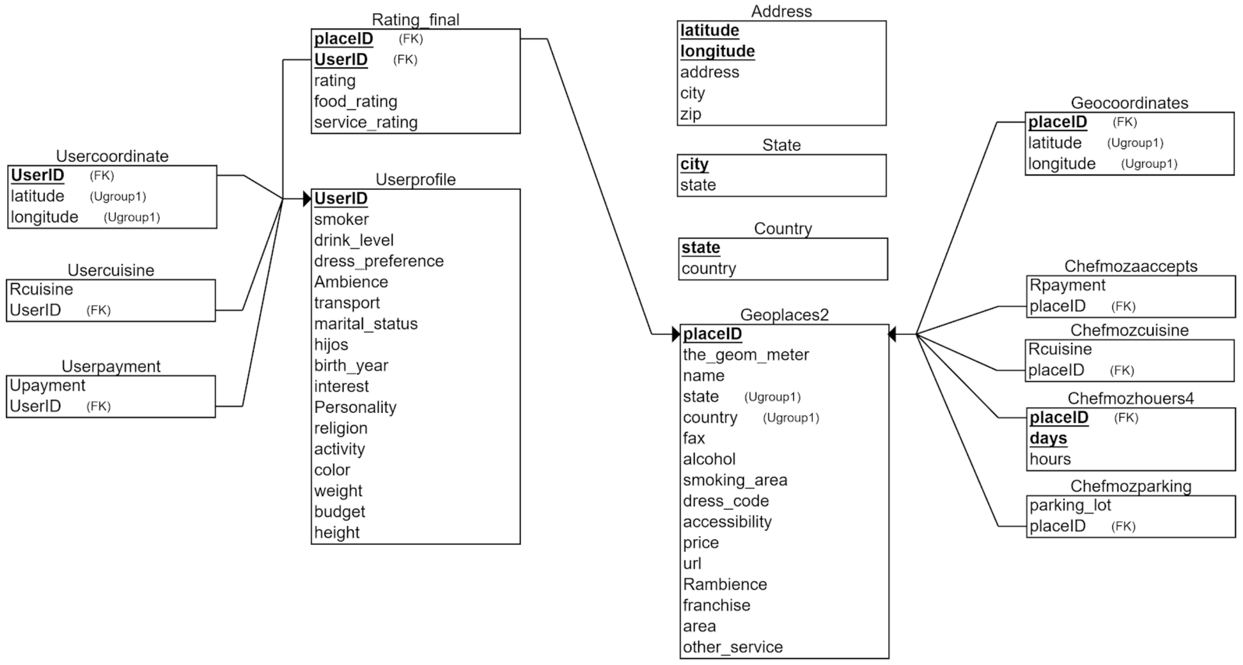
Usercoordinate

userID	latitude	longitude

Userprofile

UserID	smoker	drink_level	dress_preference
Ambience	transport	marital_status	hijos
birth_year	interest	personality	religion
activity	color	weight	Budget
Height			

(optional)Creating diagram using ERDplus:



```
CREATE TABLE Geoplaces2
(
  placeID CHAR(6) NOT NULL,
  name VARCHAR NOT NULL,
  state VARCHAR NOT NULL,
  country VARCHAR NOT NULL,
  fax VARCHAR NOT NULL,
  alcohol VARCHAR NOT NULL,
  smoking_area VARCHAR NOT NULL,
  dress_code VARCHAR NOT NULL,
  accessibility VARCHAR NOT NULL,
  price VARCHAR NOT NULL,
  url VARCHAR NOT NULL,
  Rambience VARCHAR NOT NULL,
  franchise CHAR(1) NOT NULL,
  area VARCHAR NOT NULL,
  other_service VARCHAR NOT NULL,
  the_geom_meter VARCHAR NOT NULL,
  PRIMARY KEY (placeID),
  UNIQUE (state, country)
);

CREATE TABLE Geocoordinates
(
  latitude FLOAT NOT NULL,
  longitude FLOAT NOT NULL,
  placeID CHAR(6) NOT NULL,
  PRIMARY KEY (placeID),
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID),
  UNIQUE (latitude, longitude)
);

CREATE TABLE Chefmozaaccepts
(
  Rpayment INT NOT NULL,
  placeID CHAR(6) NOT NULL,
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID)
);

CREATE TABLE Chefmozcuisine
(
  Rcuisine VARCHAR NOT NULL,
  placeID CHAR(6) NOT NULL,
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID)
);
```

```
CREATE TABLE Chefmozhouers4
(
  hours VARCHAR NOT NULL,
  days VARCHAR NOT NULL,
  placeID CHAR(6) NOT NULL,
  PRIMARY KEY (days, placeID),
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID)
);
CREATE TABLE Chefmozparking
(
  parking_lot VARCHAR NOT NULL,
  placeID CHAR(6) NOT NULL,
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID)
);
CREATE TABLE Userprofile
(
  UserID CHAR(5) NOT NULL,
  smoker VARCHAR NOT NULL,
  drink_level VARCHAR NOT NULL,
  dress_preference VARCHAR NOT NULL,
  Ambience VARCHAR NOT NULL,
  transport VARCHAR NOT NULL,
  marital_status VARCHAR NOT NULL,
  hijos VARCHAR NOT NULL,
  birth_year INT NOT NULL,
  interest VARCHAR NOT NULL,
  Personality VARCHAR NOT NULL,
  religion VARCHAR NOT NULL,
  activity VARCHAR NOT NULL,
  color VARCHAR NOT NULL,
  weight INT NOT NULL,
  budget VARCHAR NOT NULL,
  height FLOAT NOT NULL,
  PRIMARY KEY (UserID)
);

CREATE TABLE Usercoordinate
(
  latitude FLOAT NOT NULL,
  longitude FLOAT NOT NULL,
  UserID CHAR(5) NOT NULL,
  PRIMARY KEY (UserID),
  FOREIGN KEY (UserID) REFERENCES Userprofile(UserID),
  UNIQUE (latitude, longitude)
);
```

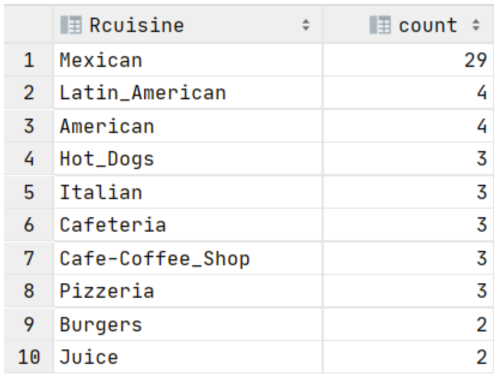
```
CREATE TABLE Userpayment
(
  Upayment VARCHAR NOT NULL,
  UserID CHAR(5) NOT NULL,
  FOREIGN KEY (UserID) REFERENCES Userprofile(UserID)
);
CREATE TABLE Rating_final
(
  rating INT NOT NULL,
  food_rating INT NOT NULL,
  service_rating INT NOT NULL,
  placeID CHAR(6) NOT NULL,
  UserID CHAR(5) NOT NULL,
  PRIMARY KEY (placeID, UserID),
  FOREIGN KEY (placeID) REFERENCES Geoplaces2(placeID),
  FOREIGN KEY (UserID) REFERENCES Userprofile(UserID)
);
CREATE TABLE Address
(
  latitude FLOAT NOT NULL,
  longitude FLOAT NOT NULL,
  city VARCHAR NOT NULL,
  address VARCHAR NOT NULL,
  zip VARCHAR NOT NULL,
  PRIMARY KEY (latitude, longitude)
);
CREATE TABLE State
(
  city VARCHAR NOT NULL,
  state VARCHAR NOT NULL,
  PRIMARY KEY (city)
);
CREATE TABLE Country
(
  state VARCHAR NOT NULL,
  country VARCHAR NOT NULL,
  PRIMARY KEY (state)
);
CREATE TABLE Usercuisine
(
  Rcuisine VARCHAR NOT NULL,
  UserID CHAR(5) NOT NULL,
  FOREIGN KEY (UserID) REFERENCES Userprofile(UserID)
);
```

Step 4. Questions

answer the following questions using one or more SQL queries. You should submit your SQL queries along with the first 10 rows of each result.

(a) Based on the statistics, which cuisine will a person who is interested in technology most likely to choose? Describe your ranking criteria.

```
SELECT UC."Rcuisine", count(*) FROM userprofile UP, usercuisine UC
WHERE UP."userID" = UC."userID" AND UP.interest = 'technology'
GROUP BY UC."Rcuisine"
ORDER BY count(*) DESC;
```



	Rcuisine	count
1	Mexican	29
2	Latin_American	4
3	American	4
4	Hot_Dogs	3
5	Italian	3
6	Cafeteria	3
7	Cafe-Coffee_Shop	3
8	Pizzeria	3
9	Burgers	2
10	Juice	2

Based on this ranking criteria of the sorted count of all the times when a user put a cuisine type as its preference, out of all the cuisines liked by people who are interested in technology, “Mexican” is the most popular choice, liked by 29 people. Furthermore, the number is much greater than the second popular cuisine. Therefore, we believe he/she will choose Mexican cuisine.

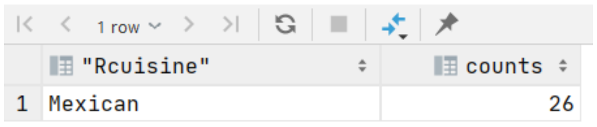
(b) Based on the data provided, which cuisine is most popular (they serve people with very different characteristics), which is most exclusive (they serve people with very similar characteristics)?

To find which cuisine is most popular, and which is the most exclusive, we will need to group the tables by different categories of cuisine. Given these groups, we will select some attributes from the tables that can be considered “characteristics”. In fact, we have tried adding almost all attributes, and the result is similar to having 6 attributes. We think “interest”, “personality”, “religion”, “activity” are the most important personal traits for a customer, and furthermore, the restaurant may be interested in the customer’s drinking level and budget, because these are pretty related to ordering cuisine.

Our SQL query generates a table, with the first column being the unique cuisine name, and the second column indicate how popular the cuisine is among people with very different characteristic. The large the number on the right, the more popular the cuisine is. We used two different sorting order to find the max and min. (Note: for min, there are more than 1 type of cuisine that is exclusive, and we decide to use what our query picked: Africa)

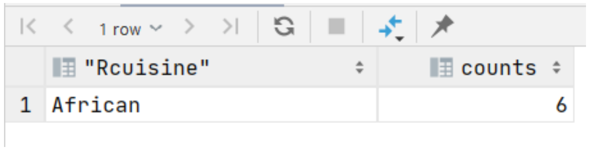
[Notes about codes: there is drink_level <> ‘?’ because it was a missing data]

```
SELECT UC."Rcuisine", (count(distinct UP.drink_level) + count(distinct UP.interest)
  + count(distinct UP.personality) + count(distinct UP.religion)
  + count(distinct UP.activity) + count(distinct UP.budget)
  ) as counts
FROM userprofile UP, usercuisine UC
WHERE UP."userID" = UC."userID" AND UP.drink_level <> '?'
GROUP BY UC."Rcuisine"
ORDER BY counts DESC
LIMIT 1
```



```
  "Rcuisine"    counts
1 Mexican           26
```

```
SELECT UC."Rcuisine", (count(distinct UP.drink_level) + count(distinct UP.interest)
  + count(distinct UP.personality) + count(distinct UP.religion)
  + count(distinct UP.activity) + count(distinct UP.budget)
  ) as counts
FROM userprofile UP, usercuisine UC
WHERE UP."userID" = UC."userID" AND UP.drink_level <> '?'
GROUP BY UC."Rcuisine"
ORDER BY counts
LIMIT 1
```



```
  "Rcuisine"    counts
1 African            6
```

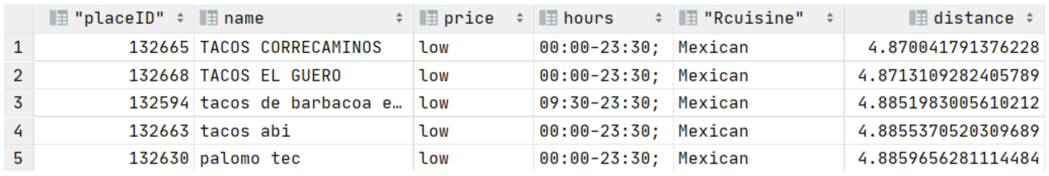
(c) Based on the customer information provided, list 3 ranked choices of restaurants for two customers below.

Customer	Time preferred	Cuisine	Preference	Time allowance
U1004	Mon 18:00	Mexican	Price > rating	Fast
U1124	Sat 22:00	FastFood	Rating > Price	Whatever

For customer U1004:

```
-- get the latitutde and longitude of that customer U1004
SELECT *
FROM usercoordinate
WHERE "userID" = 'U1004';
-- It turns out to be lat 18.867, long -99.183

SELECT GC."placeID",
    PLACE.name,
    PLACE.price,
    PLACE.hours,
    PLACE."Rcuisine",
    -- find the euclidean distance to that user's home address
    ((GC.latitude - 18.867) ^ 2 + (GC.longitude - (-99.183)) ^ 2) ^ (0.5) AS distance
FROM geocoordinates GC,
  (SELECT GP."placeID", GP.name, GP.price, PLACE.hours, PLACE."Rcuisine"
   FROM geoplaces2 GP,
      -- get all the places that offered mexican cuisine
      (
        SELECT distinct CC."placeID", CH.hours, CC."Rcuisine"
        FROM chefmozcuisine CC,
          (SELECT *
           FROM (
               -- Get all the places that is opened at 18:00, and closed till later,
               -- which will a time interval that covers the guest's preferred time
               SELECT *,
                   TO_TIMESTAMP(SPLIT_PART(hours, '-', 1), 'HH24:MI')::TIME       as open,
                   TO_TIMESTAMP(TRIM(SPLIT_PART(hours, '-', 2), ';'), 'HH24:MI')::TIME as close
               FROM (SELECT *
                   FROM (SELECT "placeID", "days", unnest(string_to_array(hours, ';')) as hours
                       FROM chefmozhours4) as CH
                   WHERE CH.hours <> '') CH
               WHERE lower(CH.days) like '%mon%') CH
           WHERE open <= '18:00'::TIME
            AND ('18:00'::TIME < close OR close <= '8:00'::TIME)) CH
        WHERE CC."placeID" = CH."placeID"
         AND lower(CC."Rcuisine") like '%mexican%') as PLACE
    WHERE GP."placeID" = PLACE."placeID"
      AND price = 'low') AS PLACE
WHERE GC."placeID" = PLACE."placeID"
ORDER BY distance;
```



	"placeID"	name	price	hours	"Rcuisine"	distance
1	132665	TACOS CORRECAMINOS	low	00:00-23:30;	Mexican	4.870041791376228
2	132668	TACOS EL GUERO	low	00:00-23:30;	Mexican	4.8713109282405789
3	132594	tacos de barbacoa e…	low	09:30-23:30;	Mexican	4.8851983005610212
4	132663	tacos abi	low	00:00-23:30;	Mexican	4.8855370520309689
5	132630	palomo tec	low	00:00-23:30;	Mexican	4.8859656281114484

Explanation:

In the very inside of the query, we unnested the places with multiple hours values into different rows.

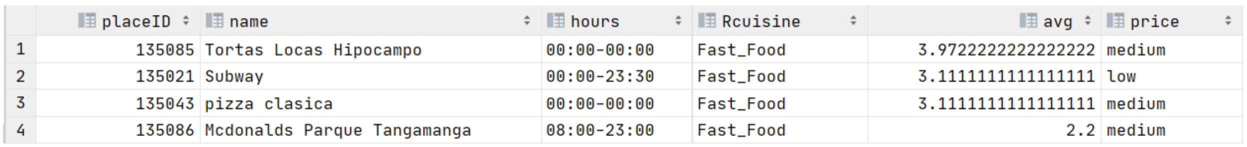
Since the customer preferred Mon 18:00, we want all the places that is open at 18:00 Monday. This includes places that just opened at 18:00, but not places that closed at 18:00. There are several places that closed during 3am, and we will consider them too by adding more conditions in the closing time.

Along with that condition, we can also find all places that offered Mexican cuisine. Then, we will consider the price preference, and the distance from the user's place to the restaurant. We decide that since the user prioritize price over rating, we will simply select all the low price places. [We could select some medium price places, but the distance does not seem to get significantly smaller, therefore we will not consider them]. Finally, we sort the places by the euclidean distances from the user's coordinate to the restaurant's coordinate, and get our list of recommendations.

The top three on the list are our recommendations: Tacos Correcaminos, Tacos El Guero, Tacos De Barbacoa Enfrente Del Tec

For customer U1124:

```sql
SELECT PLACE."placeID", PLACE.name, PLACE.hours, PLACE."Rcuisine", avg(PLACE.final_rating), PLACE.price
FROM (SELECT PLACE.name,
          PLACE.hours,
          PLACE."Rcuisine",
          PLACE.price,
          RT."placeID",
          (RT.rating + RT.food_rating + RT.service_rating) as final_rating
   FROM rating_final RT,
       (SELECT GP."placeID", GP.name, GP.price, PLACE.hours, PLACE."Rcuisine"
        FROM geoplaces2 GP,
           -- get all the places that offered fast food
           (
             SELECT distinct CC."placeID", CH.hours, CC."Rcuisine"
             FROM chefmozcuisine CC,
                 (SELECT *
                  FROM (
                       -- Get all the places that is opened at 22:00, and closed till later,
                       -- which will a time interval that covers the guest's preferred time
                       SELECT *,
                           TO_TIMESTAMP(SPLIT_PART(hours, '-', 1), 'HH24:MI')::TIME          as open,
                           TO_TIMESTAMP(TRIM(SPLIT_PART(hours, '-', 2), ';'), 'HH24:MI')::TIME as close
                       FROM (SELECT *
                           FROM (SELECT "placeID", "days", unnest(string_to_array(hours, ';')) as hours
                               FROM chefmozhours4) as CH
                           WHERE CH.hours <> '') CH
                       WHERE lower(CH.days) like '%sat%') CH
                  WHERE open <= '22:00'::TIME
                    AND ('22:00'::TIME < close OR close <= '8:00'::TIME)) CH
             WHERE CC."placeID" = CH."placeID"
               AND lower(CC."Rcuisine") like '%fast%') as PLACE
        WHERE GP."placeID" = PLACE."placeID") as PLACE
   WHERE PLACE."placeID" = RT."placeID") as PLACE
GROUP BY PLACE."placeID", PLACE."Rcuisine", PLACE.hours, PLACE.name, PLACE.price
ORDER BY avg(PLACE.final_rating) DESC
```



	placeID	name	hours	Rcuisine	avg	price
1	135085	Tortas Locas Hipocampo	00:00-00:00	Fast_Food	3.9722222222222222	medium
2	135021	Subway	00:00-23:30	Fast_Food	3.1111111111111111	low
3	135043	pizza clasica	00:00-00:00	Fast_Food	3.1111111111111111	medium
4	135086	Mcdonalds Parque Tangamanga	08:00-23:00	Fast_Food	2.2	medium

On calculating average: From our SQL code, our way to calculate the average of each restaurant is:

For each rating of each restaurant, find the average of one single rating instance's 'rating', 'food rating' and 'service rating', and then we calculate the average of all of the average ratings among all the rating rows of each restaurant.

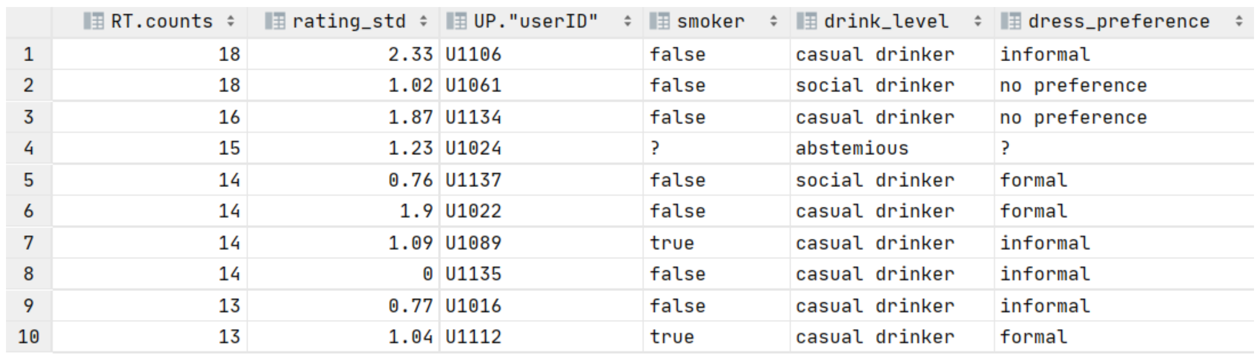
Explanation: like the last customer, we found all the places that is open for the customer's preferred time. Then we found all the fast food places. We do not need to consider distance because the customer said "whatever" for time allowance. Finally, he preferred a good rating than price. We can look at the table, sorted by the average of total rating for each place that fulfills the previous requirement, and we will find that the top three have average ratings above 3. [prices are all medium or low, and since the customer cares more about the avg rating, we will not move the second choice up, because it has a worse rating than the first one, and it also have everything else same with choice 3, but lower price, therefore it will be the second choice, not third.

Therefore, the top three on the list from the above criteria are our recommendations: Tortas Locas Hipocampo, Subway, pizza clasica.

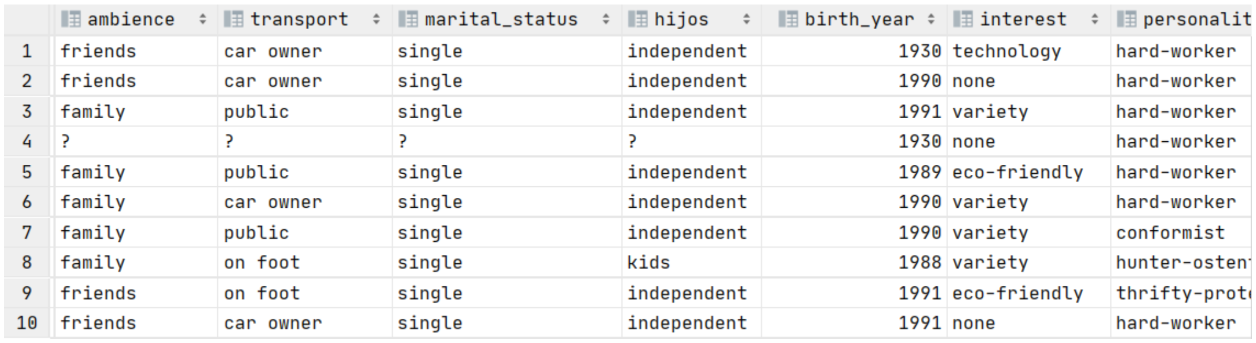
(d) Find the top 10 people who give the most # of rating and find all the information about these people and their rating diversity (e.g., how "evenly" do they rate restaurants?).

To find the rating diversity of a user, we decide to find all the ratings for that unique user, calculate the total rating of each place that he/she rated by adding rating, food rating, and service rating together, and then find the standard deviation (we think standard deviation is a good way to measure how “evenly” do they rate the restaurants, if the standard deviation of the ratings is high, that means there is a greater spread in the ratings: sometimes the user gives very high score, and sometimes low scores, but in general not same score consistently) for all the final ratings of places that the customer rated.
The standard deviation method used the default postgres stddev, and we can get a slight different result if we used stddev_pop because of the difference of denominator in the formula;

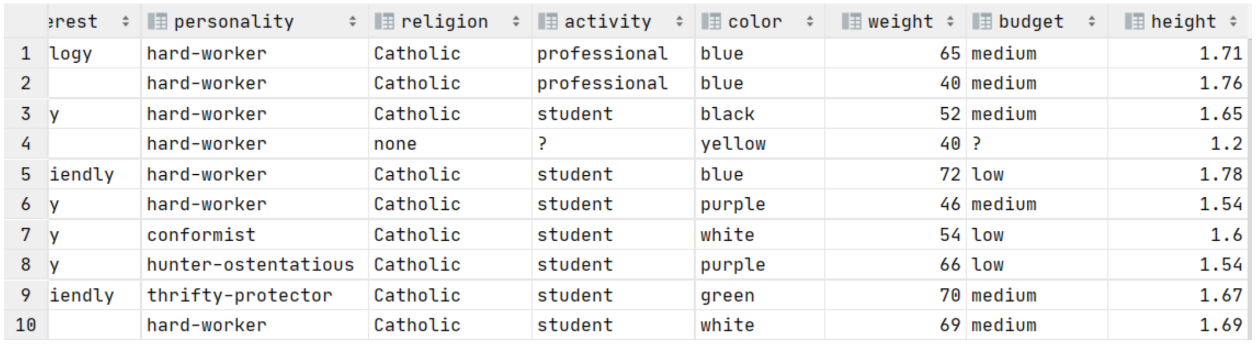
```
SELECT RT.counts, ROUND(RT.rating_std, 2) as rating_std, * FROM userprofile UP, (
  SELECT "userID", count(*) as counts,
    stddev(RT.rating + RT.food_rating + RT.service_rating) as rating_std FROM rating_final RT
  GROUP BY "userID"
  ORDER BY count(*) DESC
  LIMIT 10
) RT
WHERE RT."userID" = UP."userID"
ORDER BY RT.counts DESC;
```



	RT.counts	rating_std	UP."userID"	smoker	drink_level	dress_preference
1	18	2.33	U1106	false	casual drinker	informal
2	18	1.02	U1061	false	social drinker	no preference
3	16	1.87	U1134	false	casual drinker	no preference
4	15	1.23	U1024	?	abstemious	?
5	14	0.76	U1137	false	social drinker	formal
6	14	1.9	U1022	false	casual drinker	formal
7	14	1.09	U1089	true	casual drinker	informal
8	14	0	U1135	false	casual drinker	informal
9	13	0.77	U1016	false	casual drinker	informal
10	13	1.04	U1112	true	casual drinker	formal



	ambience	transport	marital_status	hijos	birth_year	interest	personalit
1	friends	car owner	single	independent	1930	technology	hard-worker
2	friends	car owner	single	independent	1990	none	hard-worker
3	family	public	single	independent	1991	variety	hard-worker
4	?	?	?	?	1930	none	hard-worker
5	family	public	single	independent	1989	eco-friendly	hard-worker
6	family	car owner	single	independent	1990	variety	hard-worker
7	family	public	single	independent	1990	variety	conformist
8	family	on foot	single	kids	1988	variety	hunter-osten
9	friends	on foot	single	independent	1991	eco-friendly	thrifty-prot
10	friends	car owner	single	independent	1991	none	hard-worker



	erest	personality	religion	activity	color	weight	budget	height
1	logy	hard-worker	Catholic	professional	blue	65	medium	1.71
2		hard-worker	Catholic	professional	blue	40	medium	1.76
3	y	hard-worker	Catholic	student	black	52	medium	1.65
4		hard-worker	none	?	yellow	40	?	1.2
5	iendly	hard-worker	Catholic	student	blue	72	low	1.78
6	y	hard-worker	Catholic	student	purple	46	medium	1.54
7	y	conformist	Catholic	student	white	54	low	1.6
8	y	hunter-ostentatious	Catholic	student	purple	66	low	1.54
9	iendly	thrifty-protector	Catholic	student	green	70	medium	1.67
10		hard-worker	Catholic	student	white	69	medium	1.69

(e) Suppose you are a rich investor with 200 million dollars to invest, what type of restaurant will you invest and where, why? This is a data science question and your answer should be based on queries you have come up with to make your decision evidence-based.
The criteria could be like:
(i) I want to build a restaurant in cities that don't have many restaurants currently,but they are perspective (their restaurantsreceivehigher ratings and suppose there are no geographic bias in rating)
(ii) I want to invest in "little pizza"and make it franchise because I think franchising can increase sales and pizza restaurants seem to be very popular in that city.
(iii)...... (come up with your own!!)

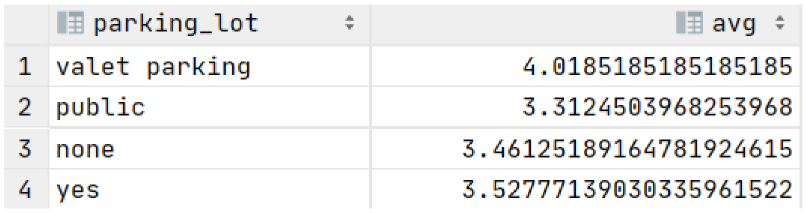
We can come up with a few criteria like the followings:

In order to invest in restaurants that maximize our sales, it is reasonable to infer that restaurants with higher user ratings are generally more popular, which makes it profitable to invest. There are many attributes associated with each known unique restaurant. We can explore these features.

1. In the given sample, does the type of parking lot offer associate with the rate?

```
SELECT PK.parking_lot, AVG(avg_final_rating) FROM chefmozparking PK,
        (SELECT "placeID", AVG(rating + food_rating + service_rating) as avg_final_rating FROM
rating_final RT
         GROUP BY "placeID") as RT
```

```
WHERE PK."placeID" = RT."placeID"
GROUP BY parking_lot
```



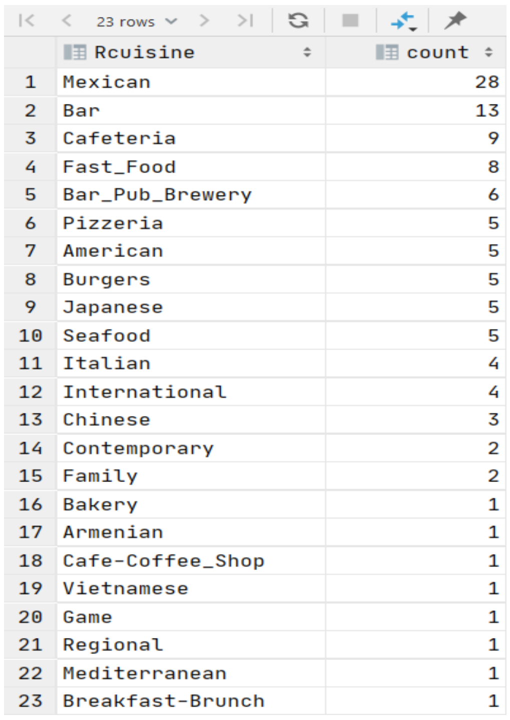
	parking_lot	avg
1	valet parking	4.0185185185185185
2	public	3.3124503968253968
3	none	3.46125189164781924615
4	yes	3.52777139030335961522

(rating for places with different types of parking lot)

As we can see from the table, places with valet parking have the highest rating, then we have none, yes, and no parking. Places with no parking are the lowest among all of them. However, we do need to claim that this does not mean whether having a parking lot causes the rating to be high, or potentially making the places more popular or having more sales. It is just usually an indicator for people to associate a high rating with good parking.

2. <u>What is the type of cuisine with the highest rating?</u>

In order to get a legtimimate answer to this questions, we may consider that some type of cuisine has only one appearance in the entire database. Let's see what are the different types of cuisines (that are in Geoplaces2)



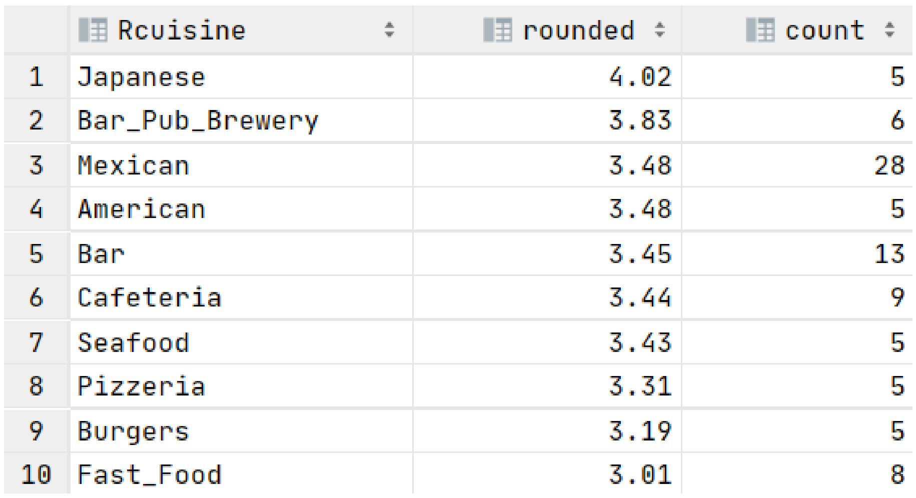
	Rcuisine	count
1	Mexican	28
2	Bar	13
3	Cafeteria	9
4	Fast_Food	8
5	Bar_Pub_Brewery	6
6	Pizzeria	5
7	American	5
8	Burgers	5
9	Japanese	5
10	Seafood	5
11	Italian	4
12	International	4
13	Chinese	3
14	Contemporary	2
15	Family	2
16	Bakery	1
17	Armenian	1
18	Cafe-Coffee_Shop	1
19	Vietnamese	1
20	Game	1
21	Regional	1
22	Mediterranean	1
23	Breakfast-Brunch	1

```
SELECT CC."Rcuisine", count(*) FROM chefmozcuisine CC,
geoplaces2 GP
WHERE GP."placeID" = CC."placeID"
group by CC."Rcuisine"
ORDER by count(*) DESC
```

From the descending order of counts of each cuisine, we can see that there are some cuisine types that do not make much appearance. These would not provide us with enough rating information. We will only keep the cuisine types that have appeared 5 times or more.

After filtering out these cuisines, applying the same strategy we had from the previous question, and find all the average ratings for each type of cuisine.

```
SELECT CC."Rcuisine", round(avg(RT.avg_final_rating), 2) as rounded FROM chefmozcuisine CC,
geoplaces2 GP,
  (SELECT "placeID", AVG(rating + food_rating + service_rating) as avg_final_rating FROM rating_final
RT
     GROUP BY "placeID") as RT
WHERE GP."placeID" = CC."placeID" AND CC."placeID" = RT."placeID"
group by CC."Rcuisine"
HAVING count(*) >= 5
ORDER by rounded DESC
```



	Rcuisine	rounded	count
1	Japanese	4.02	5
2	Bar_Pub_Brewery	3.83	6
3	Mexican	3.48	28
4	American	3.48	5
5	Bar	3.45	13
6	Cafeteria	3.44	9
7	Seafood	3.43	5
8	Pizzeria	3.31	5
9	Burgers	3.19	5
10	Fast_Food	3.01	8

(column rounded shows the rounded rating)

The result shown: The places that serve Japanese cuisine have higher ratings overall from our sample.

The previous findings from the sample will inform us that places with good ratings are generally associated with having parking lots, especially valet parking, and serve cuisine such as the Japanese, or Bar_pub_brewery, not like fast food.

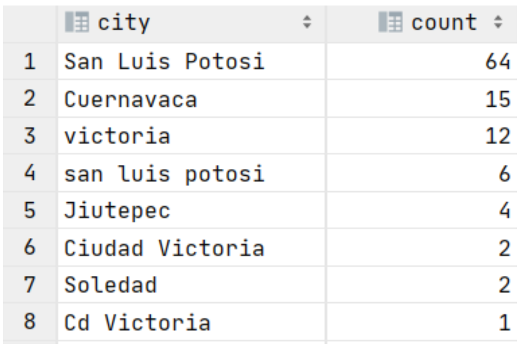
Taking into account that we have millions to invest, it is not challenging to set up a fancy restaurant that serves popular cuisines, and have a valet parking.

3. Looking for locations to invest:
   If we want to invest in a restaurant, it is important to consider the location so that we maximize the number of customers. Since our budget must be limited, we cannot directly open thousands of restaurants across North America. However, we can start the business by trying out in one area first.

   We will select a city to start our dining services. We can look into the address table and find the place that most frequently appears.

```
SELECT city, count(*) FROM address
-- get rid of the missing values
WHERE city <> '?'
```

```
GROUP BY city
ORDER by count(*) DESC
```

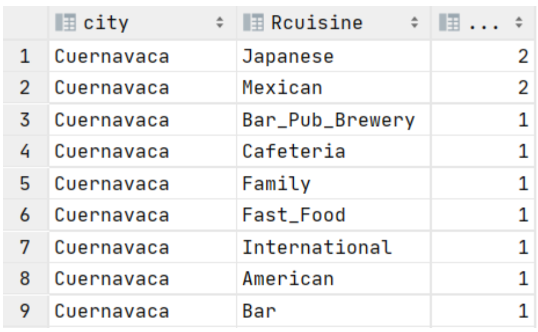


	city	count
1	San Luis Potosi	64
2	Cuernavaca	15
3	victoria	12
4	san luis potosi	6
5	Jiutepec	4
6	Ciudad Victoria	2
7	Soledad	2
8	Cd Victoria	1

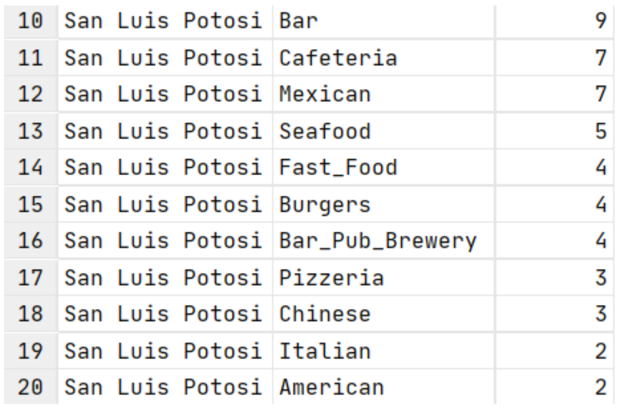
(number of appearance for cities in our sample)

At least from the sample, we could see that the city of San Luis Potosi, Cuernavaca, and victoria has a leading number of places to eat. Other than how the selection worked to generate this data, there must be reasons why these places have more places. We assume that these places are popular enough to be worthy of investment.

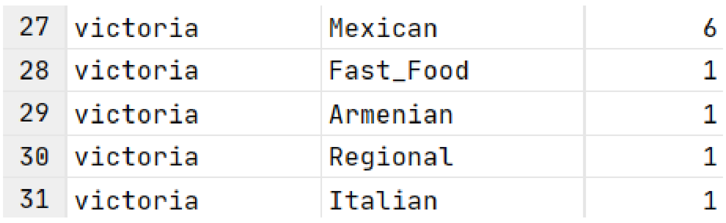
```
SELECT A.city, CC."Rcuisine", count(*) FROM address A, geocoordinates GC, geoplaces2 G,
chefmozcuisine CC
WHERE A.longitude = GC.longitude AND A.latitude = GC.latitude
AND GC."placeID" = G."placeID"
AND (A.city = 'San Luis Potosi' or A.city = 'victoria' or A.city = 'Cuernavaca')
AND CC."placeID" = G."placeID"
GROUP BY A.city, CC."Rcuisine"
ORDER BY A.city, count(*) DESC;
```



	city	Rcuisine	...
1	Cuernavaca	Japanese	2
2	Cuernavaca	Mexican	2
3	Cuernavaca	Bar_Pub_Brewery	1
4	Cuernavaca	Cafeteria	1
5	Cuernavaca	Family	1
6	Cuernavaca	Fast_Food	1
7	Cuernavaca	International	1
8	Cuernavaca	American	1
9	Cuernavaca	Bar	1



10	San Luis Potosi	Bar	9
11	San Luis Potosi	Cafeteria	7
12	San Luis Potosi	Mexican	7
13	San Luis Potosi	Seafood	5
14	San Luis Potosi	Fast_Food	4
15	San Luis Potosi	Burgers	4
16	San Luis Potosi	Bar_Pub_Brewery	4
17	San Luis Potosi	Pizzeria	3
18	San Luis Potosi	Chinese	3
19	San Luis Potosi	Italian	2
20	San Luis Potosi	American	2



27	victoria	Mexican	6
28	victoria	Fast_Food	1
29	victoria	Armenian	1
30	victoria	Regional	1
31	victoria	Italian	1

Above (same table) is the result of grouping this table by city, cuisine, and find all the number of places for each cuisine for the three cities we chose.

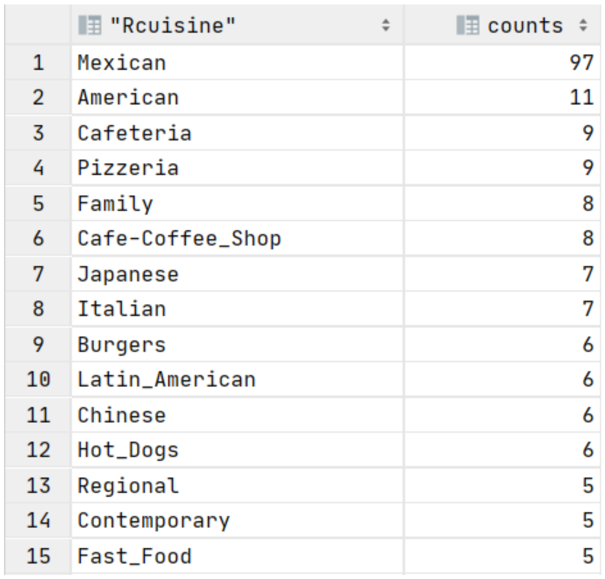
We can investigate the result:

At Cuernavaca, the top of the number of cuisine list is Japanese and Mexican. This is pretty consistent with what we found previously, as Japanese and Mexican cuisine has higher ratings.

However, in the city of San Luis Potosi and victoria, there seems to be no places that offer Japanese cuisine. This shows that there is a chance to profit more from these two popular city as starting Japnaese restaurant may attracts people to try.

4. <u>More exploration on cuisine</u>

Is rating the only measurable value we can get about how we should choose the type of cuisine from the sample? Apparently not, from the Usercuisine relation, each of the 330 user gives their favouriate cuisines. We can certainly group each cuisine and figure out how many people choose each cuisine.



	"Rcuisine"	counts
1	Mexican	97
2	American	11
3	Cafeteria	9
4	Pizzeria	9
5	Family	8
6	Cafe-Coffee_Shop	8
7	Japanese	7
8	Italian	7
9	Burgers	6
10	Latin_American	6
11	Chinese	6
12	Hot_Dogs	6
13	Regional	5
14	Contemporary	5
15	Fast_Food	5

(user's preferred cuisines)

We might think that even if Japanese cuisine has higher rating, it might not be the best choice for business when we consider the potential income. People may not go there as often as the American restaurants. The table above can serve as the information about demand of different cuisines. We assume that the majority demand is Mexican cuisine (also due to the fact that this dataset is collecte in Mexico, as the only unique non-missing country). <u>Meixican cuisine seems to be very (overwhelmingly) popular</u>.

With that popularity, is having 7 Mexican places in the city of San Luis Potosi enough? That city has 9 bars in generally. Therefore it is reasonable to asusme that it is safe to open a lot more Mexican cuisines/places.

Conclusion: Based on information about ratings on different types of places, we found that high rated places often has valet parking/non-public parking, and serving cuisines such as Japanese food. With that in mind, we found in some cities that we assumed has larger population from our sample, we can start some Japanese restaurants that people may like. In the end, we can always open more Mexican places because of its popularity among users, given the current number of Mexican places.

(An example of way to spend that money: spend 150M in San Luis, and split that into Mexican, Japanese, and others. (50M each), and spend 25M in Victoria and Cuernavaca.

We can start with spending less proportionally as an experiment in each city, so we do not spend all of it at once, from a business