Final Group Project

Group 9

Group Leader: Dimple Kuldeep Modi

Group Member: Kaixuan Han, Wanting Liu, Saloni Sharma, Ziyin Wang

Ask 1- Search for Data

Introduction

In this project, our group would like to focus on restaurants in India. More specifically, the dataset we found includes not only the name and the location of the restaurants but also the cuisine, the approximate cost per person, and the service provided by the restaurants. This dataset is available on Kaggle with a size of about 574 MB and the size might be different over time since it is updated monthly. The dataset has 51717 rows and 17 columns, and each record (row) represents a personal review of a specific restaurant with the information we need.

Dataset Source Background

This dataset was found by Kaggle, and it is produced officialy by Zomato, which is an Indian multinational restaurant aggregator and food delivery company founded in 2008. Based on the revision history, the dataset is maintained by the employees of Zomato and updated monthly. Since Zomato is known as India's largest food delivery, dining and restaurant discovery service, this dataset is authentic and representative. Dataset can be found through this link https://www.kaggle.com/datasets/rishikeshkonapure/zomato

Why this Dataset?

There are several reasons why we pick this dataset to analyze.

1. Almost all people our age have experience using websites or apps like yelp, google Maps, dianping, etc. to review and look up restaurants. It has already become a habit that people nowadays will read the reviews and check the menus before they decide the place to get the meal. Lots of people also love to rate the restaurants and write

reviews detailed based on the food they got and the amount of money they spent after they really paid visits to those restaurants. There are factors that people will consider while they are rating or voting for the restaurants, but we are not sure what these factors are and how these factors will affect the final rate.

2. People in our group are from different parts of the world. Due to the difference in our backgrounds, we may have different ideas and beliefs in many areas, but we all love food. Since it is a dataset about restaurants, we are quite interested in assessing the cuisines of the restaurants. Restaurants offer which kind of cuisine always has the highest rate. What kind of cuisine is easy to offer online ordering? Is there any relation between the kind of cuisine and the service the restaurant offers?
3. Compared with lots of other data on Kaggle, this data is very streamlined. Furthermore, unlike some data with numbers of meaningless columns, most of the columns in this data give us a brief idea about the catering industry in India and lead us to several interesting questions we may need to figure out.

Is this Dataset Suitable for Dimensional Modeling and Analytical Analysis?

This dataset is suitable for dimensional modeling and analytical analysis because this dataset has clear numerical fields for the fact tables(Rate, Vote, Approx_Cost) and several categorical attributes for dimension tables. Some of the categorical attributes are simply represented in True or False, and others can also represent under code form; therefore, it is easy for us to sort and work with this data.

Analytical Questions

- What is the best location for a restaurant? And what cuisine should this restaurant choose to be “perfect”?
- Can a restaurant remain high votes or high ratings without offering delivery options? If so, what kind of cuisine do people love to have there and what’s the average cost there? Is the approximate cost per person reasonable or not?
- List out the most popular restaurant in each cuisine. Is the restaurant popular simply because it is the only one offering this specific cuisine in the neighborhood? If not, how many competitors it has, and what’s their rate?

Concerns with the Data and Changes We Expect to Overcome

- Some columns in the dataset have values with both INTEGER and VARCHAR, so we need to recode them to INTEGER for further analysis.

- A few columns have both empty cells and NULL cells, we need to record them to the same scale. Therefore, it will not cause any confusion.

Ask 2: Data Wrangling and Dimensional Modeling

Data Downloading

We download the dataset from Kaggle and uplaod it to S3, so it is easier for us to get the dataset.

In [1]:
```
!pwd
```

```
/home/ubuntu/notebooks
```

In [2]:
```
# import command.
import os
os.listdir()
```

Out[2]:
```
['spark-warehouse',
 'Draft_FInalProject.ipynb',
 '20221101-spark.ipynb',
 'Update_FGP.ipynb',
 'ipython_cell_input.py',
 'FGP_withMarkdown.ipynb',
 'Untitled.ipynb',
 '.ipynb_checkpoints']
```

Get the data from S3

In [3]:
```
!wget https://zomatodataproject.s3.amazonaws.com/zomato.csv.zip
```

```
--2022-12-09 23:58:08--  https://zomatodataproject.s3.amazonaws.com/zomato.cs
v.zip
Resolving zomatodataproject.s3.amazonaws.com (zomatodataproject.s3.amazonaws.c
om)... 52.217.78.20, 52.217.83.52, 52.217.99.116, ...
Connecting to zomatodataproject.s3.amazonaws.com (zomatodataproject.s3.amazona
ws.com)|52.217.78.20|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 72670818 (69M) [application/zip]
Saving to: ‘zomato.csv.zip’

zomato.csv.zip      100%[===================>]  69.30M  98.2MB/s    in 0.7s

2022-12-09 23:58:08 (98.2 MB/s) - ‘zomato.csv.zip’ saved [72670818/72670818]
```

In [4]:
```
!unzip zomato.csv.zip
```

```
Archive:  zomato.csv.zip
  inflating: zomato.csv
  inflating: __MACOSX/._zomato.csv
```

Use xsv command to find the headings of csv file in order to remove the column we do not need for better analysis.

In [5]:
```
!xsv headers zomato.csv
```

```
1   ID
2   url
3   address
4   name
5   online_order
6   book_table
7   rate
8   votes
9   phone
10  location
11  rest_type
12  dish_liked
13  cuisines
14  approx_cost(for two people)
15  reviews_list
16  menu_item
17  listed_in(type)
18  listed_in(city)
```

Data Dictionary

- **ID**: Serial number of each line of data
- **Url**:URL of the restaurant
- **Address**: Complete address of the restaurant
- **Name**: Name of the restaurant
- **Online_order**: Do they accept online order (Yes/No)
- **Book_table**: Can we book table at the restaurant (Yes/No)
- **Rate**: Rating given on Zomato App
- **Votes**: Number of people gave rating
- **Phone**: Phone Number of the restaurant
- **Location**: Area of the restaurant
- **Rest_type**: Restaurant Type (Casual Dining/Cafe/Quick)
- **Dish_liked**: Type of dishes the restaurant provided
- **Cuisines**: Type of cuisines the restaurant mainly sells
- **Approx_cost(for two people)**: Approximate cost of two people
- **Reviews_list**: Reviews customer wrote on the App
- **Menu_item**: List of items in the restaurant menu
- **listed_in(type)**: The way customers eat (delivery/dine-out), and what type of food they have
- **listed_in(city)**: Street of the city the restaurant is located in

There are 18 columns in this csv files. The following columns are not needed in the further analysis.

2 url

9 phone

15 reviews_list

We will remove these 4 columns and take the rest and name it a new csv file using csvcut command and sort the new data in zomato2.csv. -z is to exapnd the maximum length of characters.

In [6]:
```
!csvcut -z 2500000 -c 1,3,4,5,6,7,8,10,11,12,13,14,16,17,18 zomato.csv > zomato
```

Chekcing the header of columns for new csv file.

In [7]:
```
!csvcut -n zomato2.csv
```

```
  1: ID
  2: address
  3: name
  4: online_order
  5: book_table
  6: rate
  7: votes
  8: location
  9: rest_type
 10: dish_liked
 11: cuisines
 12: approx_cost(for two people)
 13: menu_item
 14: listed_in(type)
 15: listed_in(city)
```

In [8]:
```
!csvstat zomato2.csv
```

1. "ID"

```
      Type of data:          Number
      Contains null values:  False
      Unique values:         51717
      Smallest value:        1
      Largest value:         51717
      Sum:                   1337349903
      Mean:                  25859
      Median:                25859
      StDev:                 14929.556
      Most common values:    1 (1x)
                             2 (1x)
                             3 (1x)
                             4 (1x)
                             5 (1x)
```

2. "address"

```
      Type of data:          Text
      Contains null values:  False
      Unique values:         11494
      Longest value:         346 characters
      Most common values:    Delivery Only (128x)
                             14th Main, 4th Sector, HSR, Bangalore (71x)
                             The Ritz-Carlton, 99, Residency Road, Bangalore
(61x)
                             Citrus Hotels, 34, Cunningham Road, Bangalore
(53x)
                             Conrad Bengaluru, Kensington Road, Ulsoor, Bang
alore (49x)
```

3. "name"

```
      Type of data:          Text
      Contains null values:  False
      Unique values:         8792
      Longest value:         159 characters
      Most common values:    Cafe Coffee Day (96x)
                             Onesta (85x)
                             Just Bake (73x)
                             Empire Restaurant (71x)
                             Five Star Chicken (70x)
```

4. "online_order"

```
      Type of data:          Boolean
      Contains null values:  False
      Unique values:         2
      Most common values:    True (30444x)
                             False (21273x)
```

5. "book_table"

```
      Type of data:          Boolean
      Contains null values:  False
      Unique values:         2
```

```
      Most common values:    False (45268x)
                             True (6449x)
```

6. "rate"

```
      Type of data:          Number
      Contains null values:  False
      Unique values:         33
      Smallest value:        -999
      Largest value:         4.9
      Sum:                   -7613045.8
      Mean:                  -147.206
      Median:                3.6
      StDev:                 358.303
      Most common values:    -999 (7775x)
                             3.9 (3972x)
                             3.8 (3873x)
                             3.7 (3821x)
                             3.6 (3316x)
```

7. "votes"

```
      Type of data:          Number
      Contains null values:  False
      Unique values:         2328
      Smallest value:        0
      Largest value:         16832
      Sum:                   14671985
      Mean:                  283.698
      Median:                41
      StDev:                 803.839
      Most common values:    0 (10027x)
                             4 (1140x)
                             6 (992x)
                             7 (872x)
                             9 (738x)
```

8. "location"

```
      Type of data:          Text
      Contains null values:  True (excluded from calculations)
      Unique values:         94
      Longest value:         29 characters
      Most common values:    BTM (5124x)
                             HSR (2523x)
                             Koramangala 5th Block (2504x)
                             JP Nagar (2235x)
                             Whitefield (2144x)
```

9. "rest_type"

```
      Type of data:          Text
      Contains null values:  True (excluded from calculations)
      Unique values:         94
      Longest value:         29 characters
      Most common values:    Quick Bites (19132x)
                             Casual Dining (10330x)
```

```
                                Cafe (3732x)
                                Delivery (2604x)
                                Dessert Parlor (2263x)
```

10. "dish_liked"

```
       Type of data:          Text
       Contains null values:  True (excluded from calculations)
       Unique values:         5272
       Longest value:         134 characters
       Most common values:    None (28078x)
                              Biryani (182x)
                              Chicken Biryani (73x)
                              Friendly Staff (69x)
                              Waffles (68x)
```

11. "cuisines"

```
       Type of data:          Text
       Contains null values:  True (excluded from calculations)
       Unique values:         2724
       Longest value:         86 characters
       Most common values:    North Indian (2913x)
                              North Indian, Chinese (2385x)
                              South Indian (1828x)
                              Biryani (918x)
                              Bakery, Desserts (911x)
```

12. "approx_cost(for two people)"

```
       Type of data:          Number
       Contains null values:  True (excluded from calculations)
       Unique values:         71
       Smallest value:        40
       Largest value:         6000
       Sum:                   28533075
       Mean:                  555.432
       Median:                400
       StDev:                 438.851
       Most common values:    300 (7576x)
                              400 (6562x)
                              500 (4980x)
                              200 (4857x)
                              600 (3714x)
```

13. "menu_item"

```
       Type of data:          Text
       Contains null values:  False
       Unique values:         9098
       Longest value:         24897 characters
       Most common values:    [] (39617x)
                              ['Butter Chicken Pizza', 'Bombay Veggie Burge
r', 'Tawa Paneer Burger', 'Oh So Cheesy Burger', 'Barbecue Chicken Burger', 'P
eri Peri Chicken Burger', 'BBQ Lamb', 'Cheesy Chicken', 'Crunchy Ferrero', 'Lo
ts of Nuts', 'Kadhai Paneer Pizza', 'Chilli Paneer and Mushroom Pizza', 'Chipo
tle Veggie Pizza', 'Creamy Cheese Pizza', 'Southern Veggie Korma Pizza', 'Pope
```

ye, The Corny Man Pizza', 'Paneer Tikka Pizza', 'Desi Margherita Pizza', 'Roas
ted Paneer with Mustard Pizza', 'Bangalore Express Pizza', 'Italian Chaska Piz
za', 'Mayo and Cheese Pizza', 'Classic Margherita', 'Cajun Hawaiian Pizza', 'B
utter Chicken Pizza', 'Lasooni Bhuna Murg Pizza', 'Chilly Chicken Pizza', 'Hot
Garlic Chicken Pizza', 'Murg Barbecue Pizza', 'Mediterranean Mutton Keema Pizz
a', 'Super Green Burger', 'Quinoa Black Bean Burger', 'Bombay Veggie Burger',
'Tawa Paneer Burger', 'Oh So Cheesy Burger', 'Barbecue Chicken Burger', 'Chipo
tle Lamb Burger', 'Crumb Fried Fish Burger', 'Peri Peri Chicken Burger', 'BBQ
Lamb', 'Cheesy Chicken', 'Picnic Chicken Burger', 'Korean Grilled Chicken with
Kimchi', 'Kiddy Kat Strawberry', 'Kiddy Kat Chocolate', 'Bubblegum', 'Oreo Sha
ke', 'Caramel Shake', 'Chocolate Shake', 'Strawberry Shake', 'Vanilla Shake',
'Cold Coffee', 'Virgin Mojito', 'Iced Tea', 'Hot Chocolate', 'Cappuccino', 'La
tte', 'Espresso Shot', 'Brownie Shake', 'Red Velvet Cheesecake Shake', 'Oreo C
heesecake Shake', 'Salted Caramel Popcorn', 'Coconut Crumble', 'Coffee Crunc
h', 'Bira', 'Thanda Paan', 'Sticky Toffee Pudding', 'Blueberry Cheesecake', 'N
utella Cheesecake', 'Caramel Custard with Strawberry or Figs', 'Red Velvet Che
esecake', 'Tiramisu', 'Zesty Vanilla and Tipsy Brownie', 'Oreo Explosion', 'Ch
ocolate Almond Crumble', 'Crunchy Ferrero', "Stoner's Chocolate Decadence", 'B
rownie Bash', 'Chocolate Lava', 'Dark Chocolate Caramel', 'Chocolate Overloa
d', 'Lots of Nuts', 'Peanut Butter Crunch', 'Nutty Fudgy', 'Lemon Pistachio Bi
scotti', 'Frutilicious Ice Cream', 'Strawberry Tease Ice Cream', 'Mango Tango
Ice Cream', 'Lychee Lovers Ice Cream', 'Monkey Business Ice Cream', 'Apple Cru
mble Ice Cream', 'Blueberry Bliss Ice Cream'] (11x)

['Avil Milk', 'Oreo Shake', 'Chocolate Shake',
'Royal Falooda', 'Fruits Salad with Ice Cream', 'Fruits Salad with Ice Cream',
'Blackcurrant Ice Cream', 'Spanish Delight Ice Cream', 'Chillout Special Ice C
ream', 'Chocolate Ice Cream', 'Butterscotch Ice Cream', 'Pista Ice Cream', 'St
rawberry Ice Cream', 'Vanilla Ice Cream', 'Royal Falooda', 'Strawberry Falood
a', 'Chillout Special Falooda', 'Lemon Juice', 'Mint Lemon Juice', 'Ginger Lem
on Juice', 'Lemon Soda', 'Mint Lemon Soda', 'CAP Juice', 'Kiwi Juice', 'Carrot
Juice', 'Pomegranate Juice', 'Watermelon Juice', 'Muskmelon Juice', 'Anjeer Ju
ice', 'Pure Strawberry Juice', 'Papaya Juice', 'Orange Juice', 'Mosambi Juic
e', 'Grape Juice', 'Apple Juice', 'Pineapple Juice', 'Banana Juice', 'Rose Mil
k', 'Chikoo Shake', 'Sharja Shake', 'Kiwi Shake', 'Oreo Shake', 'Raspberry Sha
ke', 'Pomegranate Shake', 'Mango Shake', 'Muskmelon Shake', 'Papaya Shake', 'A
pple Shake', 'Berries Shake', 'Spanish Delight Shake', 'Blackcurrant Shake',
'Chillout Special Shake', 'Chocolate Shake', 'Butterscotch Shake', 'Strawberry
Shake', 'Vanilla Shake', 'Pista Shake', 'Cold Coffee Shake', 'Avil Milk', 'Spe
cial Milk', 'Kulki Sarbath', 'Sweet Lassi', 'Chocolate Lassi', 'Mango Lassi',
'Banana Lassi', 'Strawberry Lassi'] (10x)

['Chicken Cheese Burger', 'Chicken Billys BIg B
urger', 'French Fries', 'Alfredo Veg Pasta', 'Brownie Waffles', 'Veg Burger',
'Veg Cheese Burger', 'Veg Caramelized onion Burger', 'Mushroom Classic Burge
r', 'Veg English Cheddar Cheese Burger', 'Veg Chipotle Corn Burger', 'Veg Bill
ys BIg Burger', 'Chicken Burger', 'Chicken Cheese Burger', 'Chicken Caramelize
d Onion Burger', 'Chicken Mushroom Burger', 'Chicken English Cheddar Cheese Bu
rger', 'Chicken Chipotle Corn Burger', 'Chicken Billys BIg Burger', 'Chicken N
uggets Burger', 'Veg Grilled Sandwich', 'Paneer Burji Sandwich', 'Toasted Pane
er Sandwich', 'Grilled Mushroom Sandwich', 'Corn and Cheese Sandwich', 'Egg Sa
ndwich', 'Egg Chutney Sandwich', 'Chicken Mayo Sandwich', 'Chicken Cheese Sand
wich', 'Hotdog Sandwich', 'Tuna Sandwich', 'Tangy Tomato Pasta', 'Spicy Penne
Pasta', 'Alfredo Veg Pasta', 'Smoked Chicken Pasta', 'Alfredo Chicken Pasta',
'French Fries', 'Veg', 'Chicken', 'Fish Finger', 'Vegetable Omelette', 'Masala
Omelette', 'Spanish Omelette', 'Cheese Omelette', 'Paneer Omelette', 'Ultimate
Vegan Shakes', 'Bittersweet Symphony Shakes', 'The White Paradise Shakes', 'De
ep Dark Abyss Shakes', 'Irene Adlers Mystery Shakes', 'Red Weather Parody Shak
es', 'Mind palace Shakes', 'Tower of Baskerville Shakes', 'Nights Demon Shake

s', 'Black and White Romance Shakes', 'Day Dream Shakes', 'Sherlocks Addiction Shakes', 'Mrs Hudsons Secret Shakes', 'Classic Waffles', 'Caramel Waffles', 'Choco Treat Waffles', 'Brownie Waffles', 'Nutella Crunch Waffles', 'Snowflakes Waffles', 'Rocky Road Waffles', 'Lights Out Waffles'] (9x)
['Students Veg Combo', 'Office Veg Combo', 'Students Non Veg Combo', 'Office Non Veg Combo', 'Mughlai Non Veg Combo', 'Aloo Paratha Combo', 'Poori Aloo Jeera Combo', 'Poori Aloo Tomato Combo', 'Paneer Paratha Combo', 'Students Veg Combo', 'Office Veg Combo', 'Students Non Veg Combo', 'Office Non Veg Combo', 'Mughlai Non Veg Combo', 'Shami Kebab Roll', 'Mughlai Biryani Combo [Veg]', 'Nawabi Thali Combo', 'The Lajawab Gravy Combo [Veg]', 'Mughlai Meal For Two [Veg]', 'The After Party Starters', 'Royal Feast For Four [Veg]', 'The Galawati Roll', 'Mughlai Biryani Combo [Non Veg]', 'The Nihari Combo [Chicken]', 'The Nihari Combo (Mutton)', 'The Lajawab Gravy Combo [Non Veg]', 'Mughlai Meal For Two [Non-Veg]', 'Biryani Blast', 'Royal Feast For Four [Non-Veg]', 'Baby Corn 65', 'Chilli Baby Corn', 'Mushroom 65', 'Chilli Mushroom', 'Veg Seekh Kebab', 'Kasturi Paneer Tikka', 'Paneer 65', 'Paneer Angara', 'Chilli Paneer', 'Hara Bhara Kebab', 'Paneer Tikka', 'Paneer Malai Tikka', 'Tandoori Veg Platter', 'Tandoori Chicken', 'Mutton Shami Kebab [Signature]', 'Mutton Galouti Kebab [Signature]', 'Chicken 65', 'Chicken Seekh Kebab', 'Chicken Tikka', 'Chilli Chicken', 'Chicken Angara', 'Chicken Kasturi Kebab', 'Chicken Malai Tikka', 'Mutton Sultani Seekh Kebab', 'Non Veg Tandoori Platter', 'Aloo Tomato Gravy', 'Aloo Capsicum Dry', 'Aloo Jeera Dry', 'Dal Fry', 'Dal Tadka', 'Rajma', 'Chole', 'Dal Makhani [Signature]', 'Lipta Mushroom Masala', 'Veg Kolhapuri', 'Chole Curry', 'Paneer Bhurji', 'Miloni Tarkari [Signature]', 'Mixed Veg', 'Palak Paneer', 'Diwani Handi', 'Mattar Paneer', 'Paneer Do Pyaza', 'Butter Paneer', 'Kadai Paneer [Signature]', 'Paneer Tikka Masala [Signature]', 'Methi Malai Paneer', 'Paneer Lababdar [Signature]', 'Murg Musallam', 'Egg Bhurji', 'Egg Curry', 'Chicken Nihari [Signature]', 'Chicken Curry', 'Chicken Do Pyaza', 'Chicken Methi Khas', 'Hyderabadi Chicken', 'Chicken Hara Pyaza', 'Chicken Kali Mirch', 'Chicken Kohlapuri', 'Kadai Chicken', 'Chicken Lababdar', 'Chicken Lajawab [Signature]', 'Chicken Tikka Masala', 'Chicken Boti Kebab Masala', 'Butter Chicken', 'Mughlai Chicken Masala [Signature]', 'Mutton Nihari [Signature]', 'Mutton Do Pyaza', 'Mutton Korma', 'Mutton Rogan Josh', 'Mughlai Mutton Masala', 'Mutton Boti Kebab Masala [Signature]', 'Murg Musallam', 'Steamed Rice', 'Jeera Rice', 'Ghee Rice', 'Khushka', 'Pulao', 'Veg Biryani', 'Mughlai Chicken Dum Biryani [Signature]', 'Chicken Tikka Biryani', 'Mughlai Mutton Dum Biryani [Signature]', 'Tandoori Roti', 'Lachha Paratha', 'Naan', 'Amritsari Kulcha', 'Garlic Naan', 'Missi Roti', 'Mughlai Paratha', 'Poori', 'Mughlai Bread Basket', 'Papad', 'Curd', 'Boondi Raita', 'Mixed Raita', 'Masala Papad', 'Pineapple Raita', 'Paneer Roll', 'Hawker Style Boiled Egg', 'Masala Omelette', 'Chicken Roll', 'Chicken Boti Roll', 'Shami Roll', 'The Galouti Roll [Signature]', 'Mutton Boti Roll', 'Gulab Jamun [2 Pieces]', 'Ice Cream Scoop [single]', 'Rice Kheer', 'Awadhi Kheer [Signature]', 'Gajar Ka Halwa', 'Lachhedar Rabri', 'Shahi Tukda [Signature]', 'Thums Up [750 Ml]', 'Mineral Water [1 Litre]', 'Buttermilk', 'Lime Water', 'Thums Up (600 Ml)', 'Jal Jeera', 'Lassi', 'Lime Water Salt', 'Lime Water Sweet', 'Chocolate Lassi', 'Strawberry Lassi', 'Alphonso Mango Lassi', 'Thandai', 'Mango Lassi'] (9x)

14. "listed_in(type)"

Type of data:	Text
Contains null values:	False
Unique values:	7
Longest value:	18 characters
Most common values:	Delivery (25942x) Dine-out (17779x) Desserts (3593x)

```
                                Cafes (1723x)
                                Drinks & nightlife (1101x)

 15. "listed_in(city)"

        Type of data:          Text
        Contains null values:  False
        Unique values:         30
        Longest value:         21 characters
        Most common values:    BTM (3279x)
                               Koramangala 7th Block (2938x)
                               Koramangala 5th Block (2836x)
                               Koramangala 4th Block (2779x)
                               Koramangala 6th Block (2623x)

Row count: 51717
```

As we can see from the result of csvstats above, 'location', 'rest_type', 'dish_liked', 'cuisines', 'approx_cost(for two people)' has null values. We will further recode them after inputting data into the table.

Now we want to check if the new csv file has common syntax errors

In [9]:
```
!csvclean zomato2.csv
```

```
No errors.
```

Our data has done the simple cleaning and we can make other changes after creating table.

Creating Table

In [10]:
```
!pip freeze | grep -E 'ipython-sql|psycopg2'
```

```
ipython-sql==0.4.1
psycopg2==2.9.5
psycopg2-binary==2.9.5
```

In [11]:
```
%load_ext sql
```

In [12]:
```
!dropdb -U student GP9
```

In [13]:
```
!createdb -U student GP9
```

In [14]:
```
%sql postgresql://student@/GP9
```

In [15]:
```
!psql --version
```

```
psql (PostgreSQL) 12.12 (Ubuntu 12.12-0ubuntu0.20.04.1)
```

In [16]:
```
%%sql
DROP TABLE IF EXISTS ZOMATO Cascade;
CREATE TABLE ZOMATO (
    ID INT NOT NULL,
    address VARCHAR(10000),
```

```
    name VARCHAR(10000),
    online_order VARCHAR(100),
    book_table VARCHAR(100),
    rate FLOAT,
    votes INTEGER,
    location VARCHAR(100),
    rest_type VARCHAR(100),
    dish_liked VARCHAR(1000),
    cuisines VARCHAR(100),
    approx_cost_two_people INTEGER,
    menu_item VARCHAR(100000),
    listed_in_type VARCHAR(100),
    listed_in_city VARCHAR(100)
);
```

```
 * postgresql://student@/GP9
Done.
Done.
```

Out[16]: []

In [17]:
```
%%sql
COPY ZOMATO FROM '/home/ubuntu/notebooks/zomato2.csv'
CSV
HEADER;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[17]: []

Table Cleaning

As we can mentioned above, there are columns with null values, so we now recode these empty cells into value 'NULL'.

In [18]:
```
%%sql
UPDATE ZOMATO
SET dish_liked=NULL
WHERE dish_liked is NULL
```

```
 * postgresql://student@/GP9
28078 rows affected.
```

Out[18]: []

In [19]:
```
%%sql
UPDATE ZOMATO
SET location=NULL
WHERE location is NULL
```

```
 * postgresql://student@/GP9
21 rows affected.
```

Out[19]: []

In [20]:
```
%%sql
UPDATE ZOMATO
```

```
SET rest_type=NULL
WHERE rest_type is NULL
```

```
 * postgresql://student@/GP9
227 rows affected.
```

Out[20]: []

In [21]:
```
%%sql
UPDATE ZOMATO
SET cuisines=NULL
WHERE cuisines is NULL
```

```
 * postgresql://student@/GP9
45 rows affected.
```

Out[21]: []

In [22]:
```
%%sql
UPDATE ZOMATO
SET approx_cost_two_people=0
WHERE approx_cost_two_people is NULL
```

```
 * postgresql://student@/GP9
346 rows affected.
```

Out[22]: []

Star Schema

We need to have a star scheme first to create fact table and dimension table.

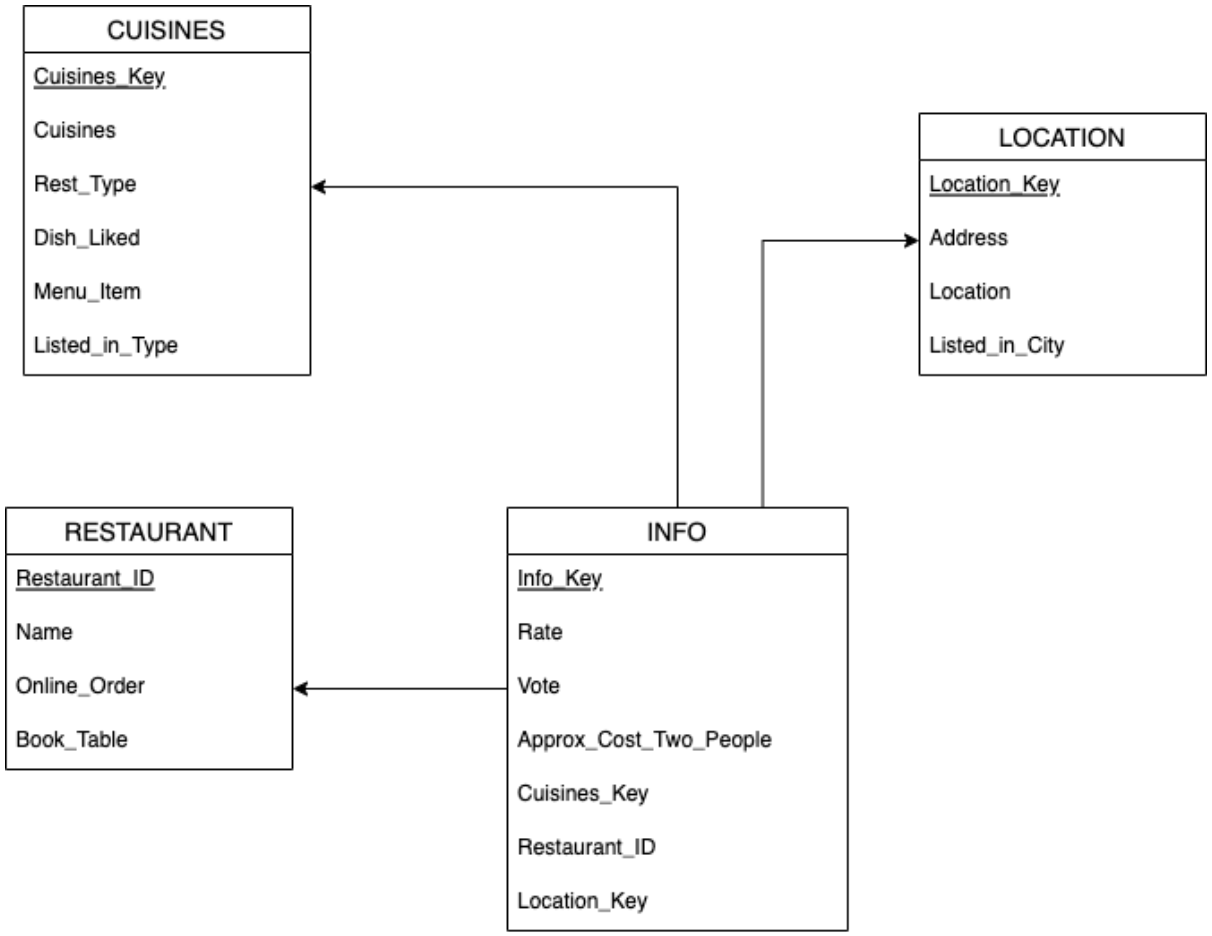
For Fact Table, we name it INFO, contains INFO_Key (Primary Key we create), Rate, Votes, and Approx_Cost_Two_People

For Dimension Table, we create 3 tables, CUISINES, RESTAURANT, LOCATION. CUISINES table contains column Cuisines_Ley (Primary Key we create), Cuisines, Rest_Type, Dish_Likes, Menu_Item, Listed_in_Type RESTAURANT table contains column Restaurant_ID (Primary Key we create), Name, Online_Order, Book_Table LOCATION table contains column Location_Key (Primary Key we create), Address, Location, Listed_in_City

The star scheme shows as the picture below.

In [23]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206814105-30cd7
```

Out[23]:



Fact Table

Create Fact table INFO

In [24]:
```
%%sql
DROP TABLE IF EXISTS INFO Cascade;
CREATE TABLE INFO(
    Info_Key SERIAL PRIMARY KEY,
    Rate FLOAT,
    Votes INTEGER,
    Approx_Cost_Two_People INTEGER
);
```

```
 * postgresql://student@/GP9
Done.
Done.
```

Out[24]: []

Populate data in ZOMATO into the table INFO:

In [25]:
```
%%sql
INSERT INTO INFO (Rate, Votes, Approx_Cost_Two_people)
SELECT rate, votes, approx_cost_two_people FROM
ZOMATO;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[25]: []

Add Foreign Key Info_Key to the ZOMATO table:

In [26]:
```
%%sql
ALTER TABLE ZOMATO
ADD COLUMN Info_Key INTEGER,
ADD CONSTRAINT fk_Info_Key
FOREIGN KEY (Info_Key)
REFERENCES INFO (Info_Key);
```

```
 * postgresql://student@/GP9
Done.
```

Out[26]: []

Dimension Table

Add first Dimension table LOCATION

In [27]:
```
%%sql
DROP TABLE IF EXISTS LOCATION Cascade;
CREATE TABLE LOCATION(
    Location_Key SERIAL PRIMARY KEY,
    Address VARCHAR(1000),
    Location VARCHAR(100),
    Listed_in_City VARCHAR(100)
);
```

```
 * postgresql://student@/GP9
Done.
Done.
```

Out[27]: []

Populate data in ZOMATO into the table LOCATION:

In [28]:
```
%%sql
INSERT INTO LOCATION (Address,Location, Listed_in_City)
SELECT address, location, listed_in_city FROM
ZOMATO;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[28]: []

Add Foreign Key Location_Key to the ZOMATO table:

In [29]:
```
%%sql
ALTER TABLE ZOMATO
ADD COLUMN Location_Key INTEGER,
ADD CONSTRAINT fk_Location_Key
FOREIGN KEY (Location_Key)
REFERENCES LOCATION (Location_Key);
```

```
 * postgresql://student@/GP9
Done.
```

Out[29]: []

Add Second Dimension table CUISINES

In [30]:
```
%%sql
DROP TABLE IF EXISTS CUISINES Cascade;
CREATE TABLE CUISINES(
    Cuisines_key SERIAL PRIMARY KEY,
    Rest_Type VARCHAR(100),
    Dish_Liked VARCHAR(1000),
    Cuisines VARCHAR(100),
    Menu_Item VARCHAR(100000),
    Listed_in_Type VARCHAR(100)
);
```

```
 * postgresql://student@/GP9
Done.
Done.
```

Out[30]: []

Populate data in ZOMATO into the table CUISINES:

In [31]:
```
%%sql
INSERT INTO CUISINES (Rest_Type, Dish_Liked, Cuisines, Menu_Item, Listed_in_Typ
SELECT rest_type, dish_liked, cuisines, menu_item, listed_in_city FROM
ZOMATO;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[31]: []

Add Foreign Key Cuisines_Key to the ZOMATO table:

In [32]:
```
%%sql
ALTER TABLE ZOMATO
ADD COLUMN Cuisines_key INTEGER,
ADD CONSTRAINT fk_Cuisines_key
FOREIGN KEY (Cuisines_key)
REFERENCES CUISINES (Cuisines_key);
```

```
 * postgresql://student@/GP9
Done.
```

Out[32]: []

Add last Dimension table RESTAURANT

In [33]:
```
%%sql
DROP TABLE IF EXISTS RESTAURANT Cascade;
CREATE TABLE RESTAURANT(
    Restaurant_ID SERIAL PRIMARY KEY,
    Name VARCHAR(10000),
    Online_Order VARCHAR(100),
```

```
    Book_Table VARCHAR(100)
);
```

```
 * postgresql://student@/GP9
Done.
Done.
```

Out[33]: []

Populate data in ZOMATO into the table RESTAURANT:

In [34]:
```
%%sql
INSERT INTO RESTAURANT (Name, Online_Order, Book_Table)
SELECT name, online_order, book_table FROM
ZOMATO;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[34]: []

Add Foreign Key Restaurant_ID to the ZOMATO table:

In [35]:
```
%%sql
ALTER TABLE ZOMATO
ADD COLUMN Restaurant_ID INTEGER,
ADD CONSTRAINT fk_Restaurant_ID
FOREIGN KEY (Restaurant_ID)
REFERENCES RESTAURANT (Restaurant_ID);
```

```
 * postgresql://student@/GP9
Done.
```

Out[35]: []

After creating all tables, we need to connect each table with the main ZOMATO table using the foreign key we created before.

Connect INFO with ZOMATO

In [36]:
```
%%sql
UPDATE ZOMATO AS a
SET Info_Key = b.Info_Key
FROM INFO AS b
WHERE a.ID = b.Info_Key;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[36]: []

Connect LOCATION with ZOMATO

In [37]:
```
%%sql
UPDATE ZOMATO AS a
SET Location_Key = b.Location_Key
FROM LOCATION AS b
WHERE a.ID = b.Location_Key;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[37]: []

Connect CUISINES with ZOMATO

In [38]:
```
%%sql
UPDATE ZOMATO AS a
SET Cuisines_key = b.Cuisines_key
FROM CUISINES AS b
WHERE a.ID = b.Cuisines_key;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[38]: []

Connect RESTAURANT with ZOMATO

In [39]:
```
%%sql
UPDATE ZOMATO AS a
SET Restaurant_ID = b.Restaurant_ID
FROM RESTAURANT AS b
WHERE a.ID = b.Restaurant_ID;
```

```
 * postgresql://student@/GP9
51717 rows affected.
```

Out[39]: []

Data Analysis

Question 1: What is the best location for a restaurant? And what cuisine should this restaurant choose to be “perfect”?

Firstly, we select the top 5 popular location for consumers. The votes are controlled over 50 because the median of votes is 41 and a small number of votes with extreme rate might generate higher bias.

In [40]:
```
%%sql
select count (l.location) as count,l.location from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where i.votes > 50
group by l.location
order by count DESC
limit 5
```

```
 * postgresql://student@/GP9
5 rows affected.
```

Out[40]:

count	location
1735	Koramangala 5th Block
1640	BTM
1366	Indiranagar
1226	HSR
1082	Jayanagar

From the results, we notice that the location selected are places with over 1000 rows of data while each row of data represents to an average of 283 consumers' rating.

In [41]:
```
%%sql
select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Koramangala 5th Block' and i.votes >50 and i.rate not in (-9
group by l.address,rate
order by rate DESC
```

```
 * postgresql://student@/GP9
17 rows affected.
```

Out[41]:

rate	address
4.900000000000001	136, Ground Floor, 1st Cross, 5th Block, Jyoti Niwas College Road, Koramangala 5th Block, Bangalore
4.9	13 KHB Colony, 17th Main, M.I.G, Koramangala 5th Block, Bangalore
4.8	122/B, Jyothi Nivas Road, 5th Block, Koramangala 5th Block, Bangalore
4.799999999999999	4th B Cross, Koramangala 5th Block, Bangalore
4.799999999999999	13 KHB Colony, 17th Main, M.I.G, Koramangala 5th Block, Bangalore
4.700000000000001	105, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore
4.700000000000001	28, 4th 'B' Cross, Koramangala 5th Block, Bangalore
4.700000000000001	122/B, Jyothi Nivas Road, 5th Block, Koramangala 5th Block, Bangalore
4.7	Shop 44, 4th B Cross Road, Koramangala 5th Block, Bangalore
4.7	12, 17th Main, 1st Cross, 5th A Block, Koramangala 5th Block, Bangalore
4.7	130, 17th H Main Road, Koramangala 5th Block, Bangalore
4.6000000000000005	146, Near William Penn, Koramangala 5th Block, Bangalore
4.6000000000000005	2nd Floor, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore
4.6	17th Main Road, JNC Road, Koramangala 5th Block, Bangalore
4.6	130, 17th H Main Road, Koramangala 5th Block, Bangalore
4.6	93/A 4th 'B' Cross, Koramangala 5th Block, Bangalore
4.6	105, 1st A Cross, Koramangala 5th Block, Bangalore

Secondly, we select the addresses in Koramangala 5th Block and present it ordered by rate. In this case we are only showing restaurants that is over 4.5 due to consumers' common sense of a good restaurant.

In [42]:
```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Koramangala 5th Block' and i.votes >50 and i.rate not in (-9
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

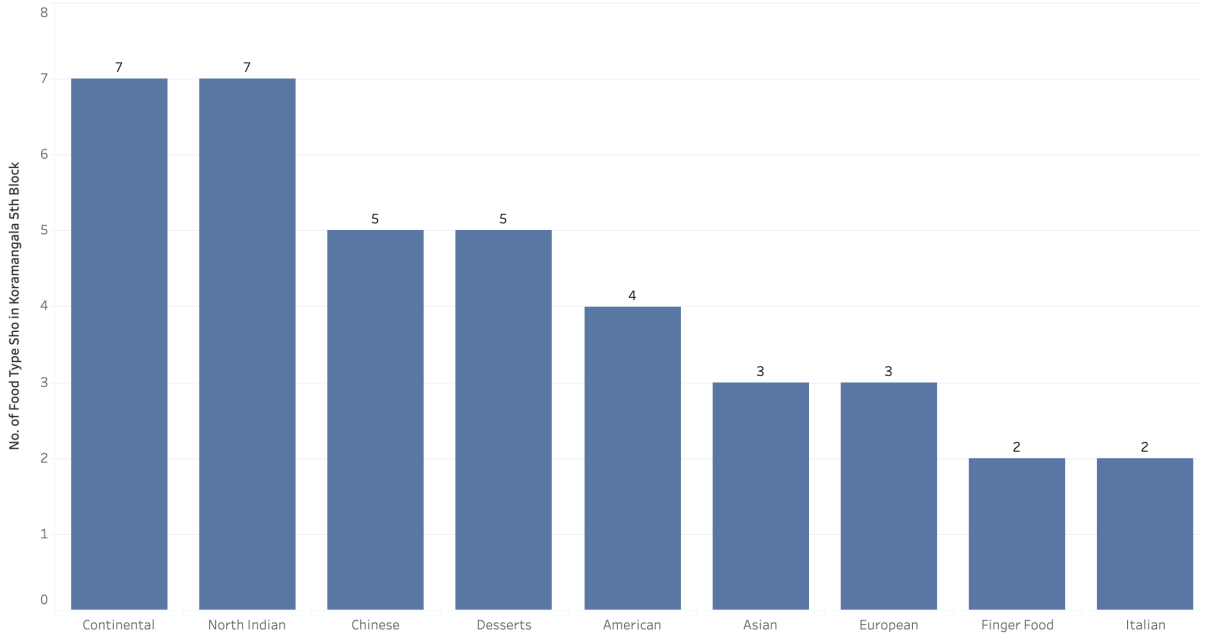
```
 * postgresql://student@/GP9
15 rows affected.
```

Out[42]:

address	cuisines
105, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore	North Indian, European, Mediterranean
105, 1st A Cross, Koramangala 5th Block, Bangalore	Desserts, Beverages
12, 17th Main, 1st Cross, 5th A Block, Koramangala 5th Block, Bangalore	Italian
122/B, Jyothi Nivas Road, 5th Block, Koramangala 5th Block, Bangalore	Continental, Asian, North Indian
13 KHB Colony, 17th Main, M.I.G, Koramangala 5th Block, Bangalore	Desserts
130, 17th H Main Road, Koramangala 5th Block, Bangalore	Desserts, Fast Food
136, Ground Floor, 1st Cross, 5th Block, Jyoti Niwas College Road, Koramangala 5th Block, Bangalore	Asian, Chinese, Thai, Momos
146, Near William Penn, Koramangala 5th Block, Bangalore	Ice Cream, Desserts
17th Main Road, JNC Road, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, American
17th Main Road, JNC Road, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, American, Pizza, Finger Food
28, 4th 'B' Cross, Koramangala 5th Block, Bangalore	Cafe, American, Burger, Steak
2nd Floor, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore	European, Continental
4th B Cross, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, European, BBQ, Finger Food, Asian
93/A 4th 'B' Cross, Koramangala 5th Block, Bangalore	Healthy Food, North Indian, Biryani, Continental, Sandwich, Desserts
Shop 44, 4th B Cross Road, Koramangala 5th Block, Bangalore	Chinese, American, Continental, Italian, North Indian

In [43]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206810647-e5e4b8
```

Out[43]:



After that, we select the cuisines of each restaurant and repeat the same steps to deal with the rest four locations.

In [44]:
```
%%sql
select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='BTM' and i.votes >50 and i.rate not in (-999,0) and rate > 4
group by l.address,rate
order by rate DESC
```

```
 * postgresql://student@/GP9
2 rows affected.
```

Out[44]:

rate	address
4.8999999999999995	100 Feet Road, 1st Phase, Near Jayadeva Flyover, 2nd Stage, BTM, Bangalore
4.6	96, 29th Main, 23rd Cross, 2nd Stage, BTM, Bangalore

In [45]:
```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='BTM' and i.votes >50 and i.rate not in (-999,0) and rate > 4
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

```
 * postgresql://student@/GP9
3 rows affected.
```

Out[45]:

address	cuisines
100 Feet Road, 1st Phase, Near Jayadeva Flyover, 2nd Stage, BTM, Bangalore	European, Mediterranean, North Indian, BBQ
96, 29th Main, 23rd Cross, 2nd Stage, BTM, Bangalore	Healthy Food, North Indian, Biryani, Continental, Desserts
96, 29th Main, 23rd Cross, 2nd Stage, BTM, Bangalore	Healthy Food, North Indian, Biryani, Continental, Sandwich, Desserts

In [46]:

```
%%sql
select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Indiranagar' and i.votes >50 and i.rate not in (-999,0) and
group by l.address,rate
order by rate DESC
```

```
 * postgresql://student@/GP9
17 rows affected.
```

Out[46]:

rate	address
4.9	151, 2nd Cross, Domlur 2nd Stage, Indiranagar, Bangalore
4.9	460, 2nd Cross, Krishna Temple Road, Indiranagar, Bangalore
4.8	2985, 12th Main, HAL 2nd Stage, Indiranagar, Bangalore
4.7	3283, 2nd Stage, Off Double Road, Indiranagar, Bangalore
4.7	607, Ground Floor, 12th Main, Hal 2nd Stage, Indiranagar, Bangalore
4.7	960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore
4.7	298, Namma Metro Pillar 62, 100 Feet Road, Indiranagar, Bangalore
4.7	4005, HAL 2nd Stage, 100 Feet Road, Indiranagar, Bangalore
4.6000000000000005	1131, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore
4.6000000000000005	610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore
4.6	Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore
4.6	100/1B, 10th Cross, 2nd Main, Indiranagar, Bangalore
4.6	1209, 100 Feet Road, Opposite Apollo Clinic, Indiranagar, Bangalore
4.6	1st Stage, 100 feet Road, Indiranagar, Bangalore
4.6	2008, 2nd Floor, 100 Feet Road, Indiranagar, Bangalore
4.6	960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore
4.6	100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore

In [47]:

```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
```

```
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Indiranagar' and i.votes >50 and i.rate not in (-999,0) and
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

```
 * postgresql://student@/GP9
18 rows affected.
```

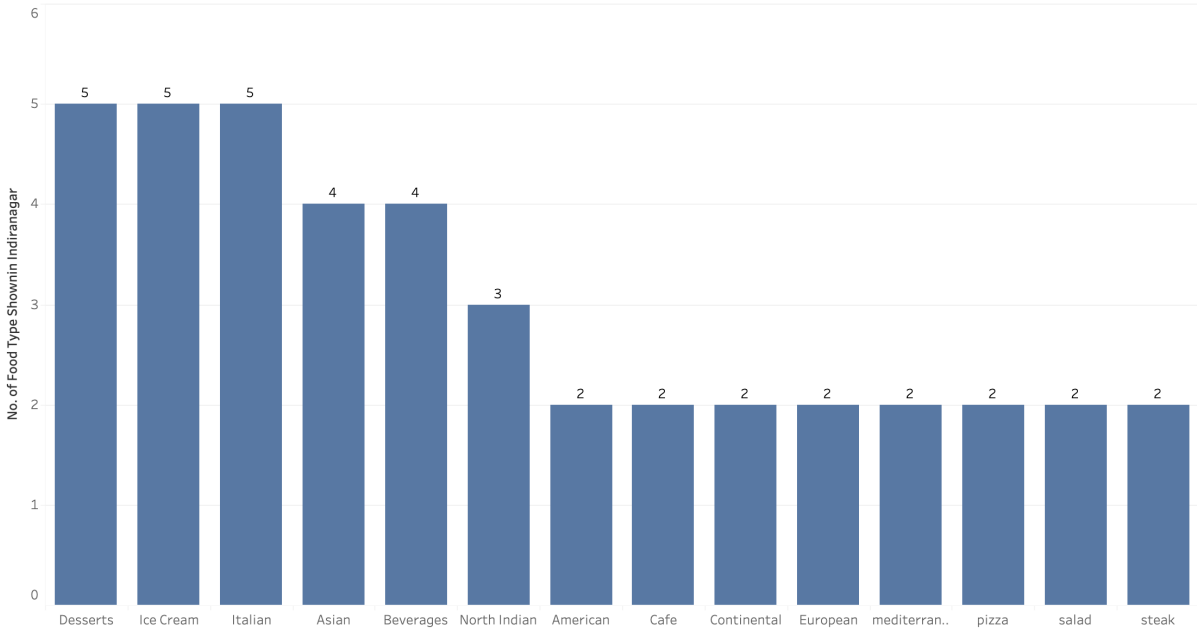
Out[47]:

address	cuisines
100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore	Continental, Asian, Italian, North Indian
100/1B, 10th Cross, 2nd Main, Indiranagar, Bangalore	Ice Cream, Desserts
1131, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore	Desserts, Cafe, Beverages
1209, 100 Feet Road, Opposite Apollo Clinic, Indiranagar, Bangalore	European, Italian, Desserts, Salad, Juices, Steak
151, 2nd Cross, Domlur 2nd Stage, Indiranagar, Bangalore	Healthy Food, Salad, Mediterranean
1st Stage, 100 feet Road, Indiranagar, Bangalore	Cafe, American, Burger, Steak
2008, 2nd Floor, 100 Feet Road, Indiranagar, Bangalore	Chinese, Continental, North Indian, Mexican
298, Namma Metro Pillar 62, 100 Feet Road, Indiranagar, Bangalore	Italian, American, Pizza
2985, 12th Main, HAL 2nd Stage, Indiranagar, Bangalore	Italian, Pizza, Beverages
3283, 2nd Stage, Off Double Road, Indiranagar, Bangalore	Ice Cream, Desserts
4005, HAL 2nd Stage, 100 Feet Road, Indiranagar, Bangalore	North Indian, European, Mediterranean, BBQ, Kebab
460, 2nd Cross, Krishna Temple Road, Indiranagar, Bangalore	Ice Cream, Desserts
607, Ground Floor, 12th Main, Hal 2nd Stage, Indiranagar, Bangalore	Asian, Burmese
610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore	Asian, Thai, Vietnamese, Malaysian, Beverages
610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore	Japanese, Sushi, Asian
960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore	Italian
Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore	Ice Cream, Beverages
Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore	Ice Cream, Desserts

In [48]:

```
from IPython.display import Image
Image(url='https://user-images.githubusercontent.com/111759300/206810604-e88cc
```

Out[48]:



In [49]:

```
%%sql
select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Indiranagar' and i.votes >50 and i.rate not in (-999,0) and
group by l.address,rate
order by rate DESC
```

```
 * postgresql://student@/GP9
17 rows affected.
```

Out[49]:

rate	address
4.9	151, 2nd Cross, Domlur 2nd Stage, Indiranagar, Bangalore
4.9	460, 2nd Cross, Krishna Temple Road, Indiranagar, Bangalore
4.8	2985, 12th Main, HAL 2nd Stage, Indiranagar, Bangalore
4.7	3283, 2nd Stage, Off Double Road, Indiranagar, Bangalore
4.7	607, Ground Floor, 12th Main, Hal 2nd Stage, Indiranagar, Bangalore
4.7	960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore
4.7	298, Namma Metro Pillar 62, 100 Feet Road, Indiranagar, Bangalore
4.7	4005, HAL 2nd Stage, 100 Feet Road, Indiranagar, Bangalore
4.6000000000000005	1131, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore
4.6000000000000005	610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore
4.6	Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore
4.6	100/1B, 10th Cross, 2nd Main, Indiranagar, Bangalore
4.6	1209, 100 Feet Road, Opposite Apollo Clinic, Indiranagar, Bangalore
4.6	1st Stage, 100 feet Road, Indiranagar, Bangalore
4.6	2008, 2nd Floor, 100 Feet Road, Indiranagar, Bangalore
4.6	960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore
4.6	100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore

In [50]:

```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='HSR' and i.votes >50 and i.rate not in (-999,0) and rate > 4
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

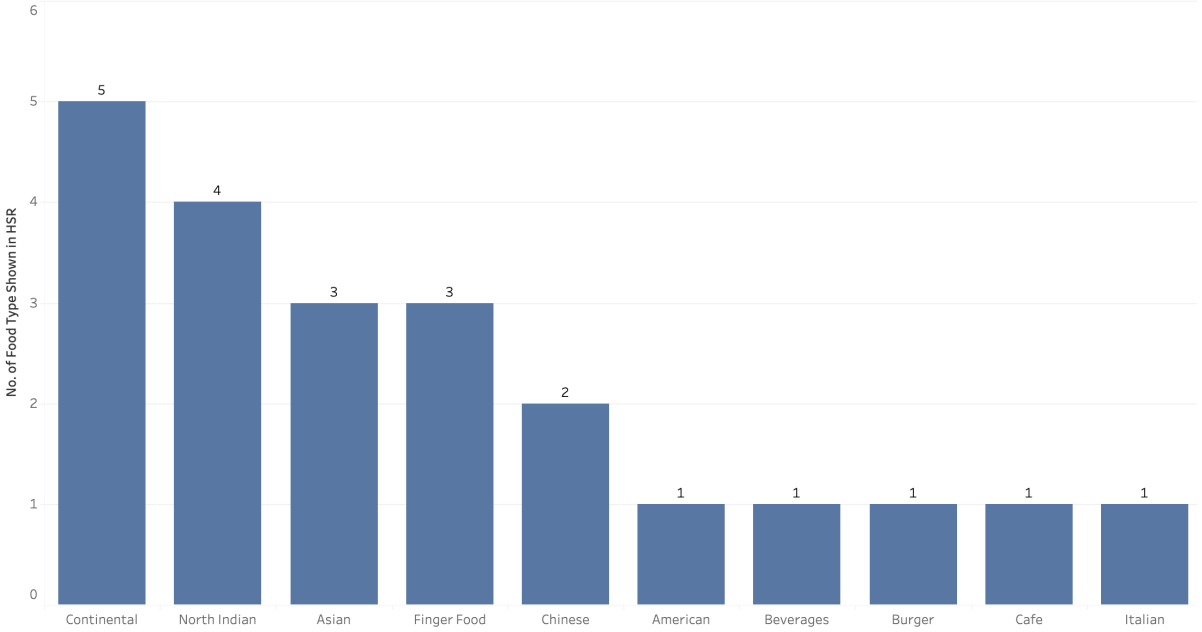
```
 * postgresql://student@/GP9
6 rows affected.
```

Out[50]:

address	cuisines
1085, 14th Main, 18th Cross, Sector 3, HSR, Bangalore	Continental, Asian, North Indian
1085, 14th Main, 18th Cross, Sector 3, HSR, Bangalore	Finger Food, Continental, North Indian
253, 1st Floor, 5th Main, 17th Cross, Sector 6, HSR Layout, HSR, Bangalore	Continental, Finger Food, Asian, Chinese
253, 2nd Floor, 5th Main, 17th Cross, Sector 6, HSR, Bangalore	Chinese, Continental, North Indian, Finger Food
Astra Hotel, 2795, 27th Main, Sector 1, HSR, Bangalore	Continental, Asian, Italian, North Indian
Shop 28, Opposite BDA Complex, 14th Main Road, Sector 2, HSR Layout, Bangalore	Cafe, American, Burger, Beverages

In [51]:
```
from IPython.display import Image
Image(url='https://user-images.githubusercontent.com/111759300/206810579-d4df6
```

Out[51]:



In [52]:
```
%%sql
select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Jayanagar' and i.votes >50 and i.rate not in (-999,0) and ra
group by l.address,rate
order by rate DESC
```

```
 * postgresql://student@/GP9
2 rows affected.
```

Out[52]:

rate	address
4.6000000000000005	615/1, Ground Floor, Janardhan Mansion, 10th C Main, 32nd D Cross, 4th Block, Jayanagar, Bangalore
4.6	24, 46th Cross, 5th Block Jayanagar, Bangalore

In [53]:
```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location ='Jayanagar' and i.votes >50 and i.rate not in (-999,0) and ra
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

```
 * postgresql://student@/GP9
2 rows affected.
```

Out[53]:

address	cuisines
24, 46th Cross, 5th Block Jayanagar, Bangalore	North Indian, Chinese, Mughlai, Mexican, BBQ
615/1, Ground Floor, Janardhan Mansion, 10th C Main, 32nd D Cross, 4th Block, Jayanagar, Bangalore	Cafe, Beverages

From the results, we notice that in Koramangala 5th Block, Indiranagar and HSR, there exists variety among highly-rated restaurants while in BTM and Jayanagar, only two restaurants in each location are performing well.

In terms of cuisines, the top 3 popular cuisines in Koramangala 5th Block are North Indian, continental and Chinese/dessert; the top 3 popular cuisines in Indiranagar are desserts, ice cream and italian; the top 3 popular cuisines in HSR are north India, continental and Asian/Finger food. For BTM and Jayanagar, samples are limited for analysis of popular cuisines.

In terms of rating, we notice that in Koramangala 5th Block, Indiranagar and HSR, the rates of restaurants vary from 4.6 to 4.9. In BTM the rates are 4.6 and 4.9 while in Jayanagar the rates are both 4.6.

The restaurants in Koramangala 5th Block, Indiranagar and HSR are well developed with high rates and variety of cuisines. In BTM, the popular cusisines of 4.9-rated restaurant are European, Mediterranean, North Indian, BBQ. Therefore, continental, Asian, Chinese of high quality is scarce. In Jayanagar, there are no highly-rated restaurants but there exists high volume of consumption. The only two restaurants listed here are not serving Continental and Asian which are popular in other locations.

In conlusion, it is recommended to open a restaurant in BTM, features in continental, Asian or Chinese with high quality. Moreover, it is also suggested to open a restaurant in Jayanagar features in continental and Asian.

Question 2: Can a restaurant remain high votes or high ratings without offering delivery options? If so, what

kind of cuisine do people love to have there and what's the average cost there? Is the approximate cost per person reasonable or not?

In [54]:
```
%%sql
SELECT COUNT(DISTINCT(Name)) FROM RESTAURANT
WHERE online_order = 'No'
```

```
 * postgresql://student@/GP9
1 rows affected.
```

Out[54]:

count
5180

- **Background:** 5180 resturants do not offer online-order.
- **Votes:** There may be customers given malicious feedbacks. In order to minimize the impact of those malicious feedbacks, we assume reviews with Votes above 50 as valid Votes and only include those valid Votes in our analysis.
- **Rates:** Since rate is not a mandatory part to fill out in the review, there are reivews with empty rates. Also, some of the new opening restaurants do not have rates. Therefore, we decide to set a limitation to the rate (Rate > 0).

In [55]:
```
%%sql
SELECT LEFT(RESTAURANT.Name,30) as Restaurant, avg(INFO.Rate) as Rate FROM REST
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
WHERE online_order = 'No' AND Rate > 0 AND Votes > 50
GROUP BY LEFT(RESTAURANT.Name,30)
ORDER BY 2 DESC
LIMIT 10
```

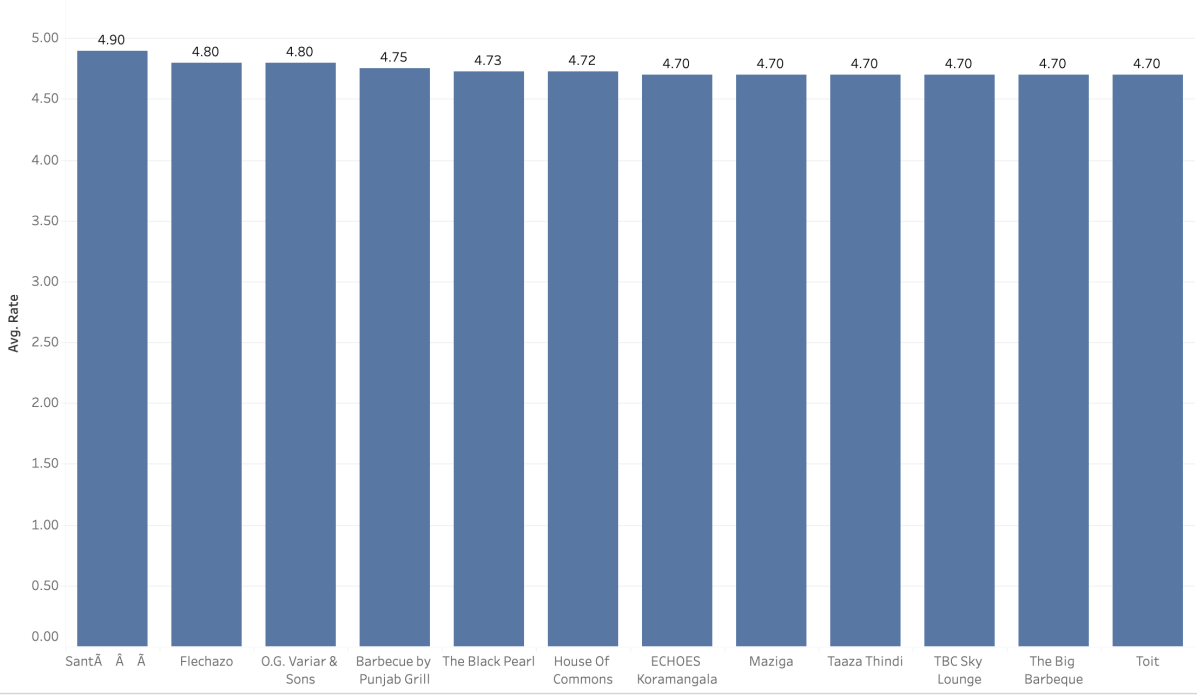
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[55]:

restaurant	rate
SantÃ Â Ã Â Ã Â Ã Â Ã Â Ã Â Ã	4.9
Flechazo	4.8
O.G. Variar & Sons	4.8
AB's - Absolute Barbecues	4.7894736842105265
Biergarten	4.766666666666666
Barbecue by Punjab Grill	4.75
The Black Pearl	4.727777777777778
House Of Commons	4.723809523809526
The Big Barbeque	4.7
TBC Sky Lounge	4.7

In [56]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206805934-364114
```

Out[56]:



In [57]:
```
%%sql
SELECT LEFT(RESTAURANT.Name,30) as Restaurant, avg(INFO.Rate) as Rate FROM REST
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
WHERE online_order = 'Yes' AND Rate > 0 AND Votes > 50
GROUP BY LEFT(RESTAURANT.Name,30)
ORDER BY 2 DESC
LIMIT 10
```

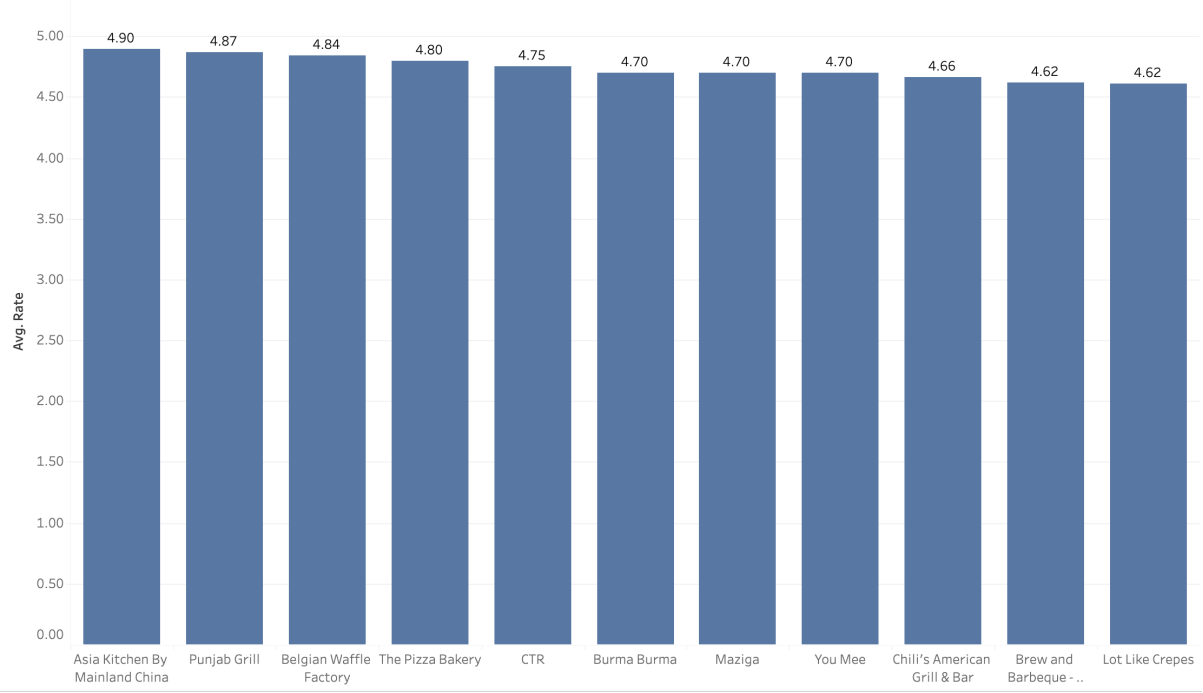
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[57]:

restaurant	rate
Asia Kitchen By Mainland China	4.900000000000001
Byg Brewski Brewing Company	4.8999999999999995
Punjab Grill	4.871428571428572
Belgian Waffle Factory	4.844827586206896
The Pizza Bakery	4.8
CTR	4.75
Maziga	4.7
Burma Burma	4.7
You Mee	4.7
Chili's American Grill & Bar	4.664285714285714

In [58]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206806014-a54295
```

Out[58]:



- *SantA* has the highest rate, 4.9, among the restaurants without online ordering and *Asia Kitchen By Mainland China* has the highest rate, 4.9, among the restaurants with online ordering. Both resturants have the exact same rate.
- Plotted TOP 10 highest rate restaurants with and without online ordering, we find that there is not much different between the rate of them. All of the 20 restaurants have the overall rates above 4.65. All TOP 10 rate restaurants without online ordering have the rate around 4.7.
- In this way, we can say that restaurants can remain high rate without offering online ordering.

In [59]:

```
%%sql
SELECT RESTAURANT.Name, avg(INFO.Votes) as Votes FROM RESTAURANT
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
WHERE online_order = 'No' AND Rate > 0 AND Votes > 50
GROUP BY RESTAURANT.Name
ORDER BY 2 DESC
LIMIT 10
```

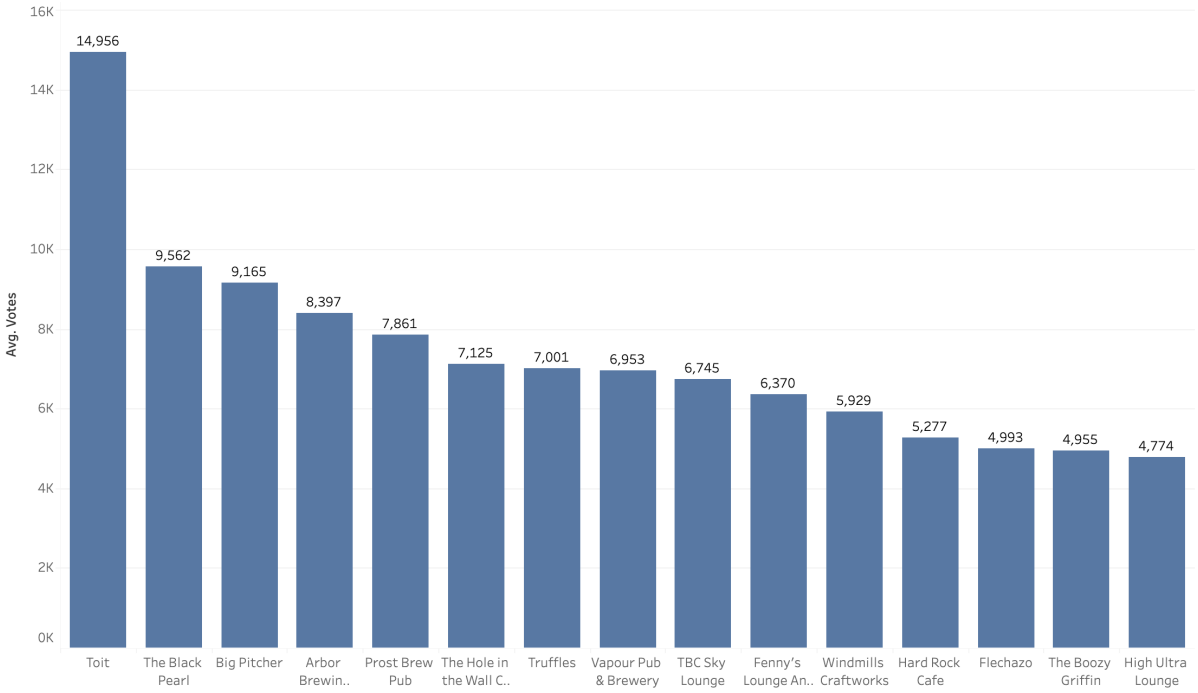
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[59]:

name	votes
Toit	14956.0000000000000000
The Black Pearl	9562.3333333333333333
Big Pitcher	9164.5000000000000000
Arbor Brewing Company	8396.5454545454545455
Prost Brew Pub	7860.9000000000000000
The Hole in the Wall Cafe	7124.8750000000000000
Truffles	7001.3720930232558140
Vapour Pub & Brewery	6952.5000000000000000
TBC Sky Lounge	6745.0000000000000000
Fenny's Lounge And Kitchen	6370.2142857142857143

In [60]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206805554-bfbf70
```

Out[60]:



In [61]:
```
%%sql
SELECT RESTAURANT.Name, avg(INFO.Votes) as Votes FROM RESTAURANT
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
WHERE online_order = 'Yes' AND Rate > 0 AND Votes > 50
GROUP BY RESTAURANT.Name
ORDER BY 2 DESC
LIMIT 10
```

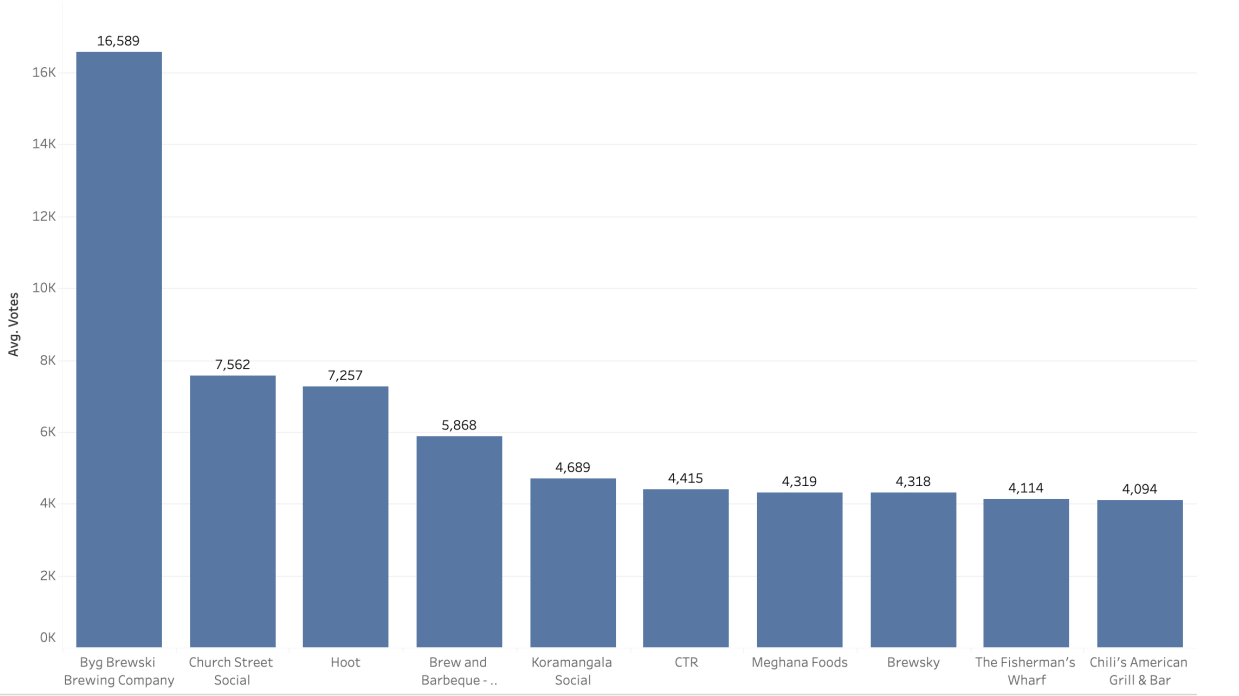
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[61]:

name	votes
Byg Brewski Brewing Company	16588.500000000000
Church Street Social	7561.7272727272727273
Hoot	7257.0000000000000000
Brew and Barbeque - A Microbrewery Pub	5868.2000000000000000
Koramangala Social	4688.8125000000000000
Hammered	4472.8750000000000000
CTR	4414.5000000000000000
Meghana Foods	4318.5666666666666667
Brewsky	4317.5000000000000000
The Fisherman's Wharf	4114.0769230769230769

In [62]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206805465-382360
```

Out[62]:



- *Toit* is the restaurant with the largest average votes among non-online-order restaurants. It has an average of 14956 in its votes. And *Byg Brewski Brewing Company* is the restaurant with the largest average votes among online-order restaurants. It has an average of 16588.5 in its votes.
- Even though *Byg Brewski Brewing Company's* average vote is larger than Toit's, the non-online-order restaurants' second largest average votes is 9562, which is almost 2000 larger than the online-order restaurants' second largest average votes.
- The TOP 10 non-online-order restaurants all have average votes over 6000, but only TOP 3 online-order restaurants' average votes is over 6000.

- Based on the plots, we can see that the TOP 1 non-online-order restaurant's average vote is way higher than the online-order restaurant with the second highest average vote. Same as the TOP 1 online-order restaurant, its average vote is 2 times higher than the restaurant in the second place.
- As a result, we can conclude that, there is not much difference between non-online-order restaurants and online-order restaurants. Restaurants can still get a high average vote if they choose not to offer online ordering.

In [63]:
```
%%sql
SELECT LEFT(RESTAURANT.Name,30) as Restaurant, avg(INFO.Rate) as Rate, avg(INFO
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
JOIN CUISINES ON
    CUISINES.Cuisines_Key = INFO.info_key
WHERE online_order = 'No' AND Rate > 0 AND Votes > 50
GROUP BY RESTAURANT.Name, CUISINES.Cuisines
ORDER BY 2 DESC
LIMIT 10
```

```
 * postgresql://student@/GP9
10 rows affected.
```

Out[63]:

restaurant	rate	cost	
SantÃ Â Ã Â Ã Â Ã Â Ã Â Ã Â Ã	4.9	1000.0000000000000000	H M
Flechazo	4.8	1400.0000000000000000	M
The Black Pearl	4.8	1500.0000000000000000	M
O.G. Variar & Sons	4.8	200.0000000000000000	
Biergarten	4.799999999999999	2100.0000000000000000	
AB's - Absolute Barbecues	4.7894736842105265	1568.4210526315789474	M
Barbecue by Punjab Grill	4.75	1300.0000000000000000	
House Of Commons	4.723809523809526	1000.0000000000000000	
The Black Pearl	4.700000000000001	1400.0000000000000000	M
Toit	4.7	1500.0000000000000000	

In [64]:

```
%%sql
SELECT LEFT(RESTAURANT.Name,30) as Restaurant, avg(INFO.Votes) as Vote, avg(INF
JOIN INFO ON
    RESTAURANT.Restaurant_ID = INFO.info_key
JOIN CUISINES ON
    CUISINES.Cuisines_Key = INFO.info_key
WHERE online_order = 'No' AND Rate > 0 AND Votes > 50
GROUP BY RESTAURANT.Name, CUISINES.Cuisines
ORDER BY 2 DESC
LIMIT 10
```

```
 * postgresql://student@/GP9
10 rows affected.
```

Out[64]:

restaurant	vote	cost	cuisines
Toit	14956.0000000000000000	1500.0000000000000000	Italian, American, Pizza
The Black Pearl	10498.8461538461538462	1400.0000000000000000	North Indian, European, Mediterranean
Big Pitcher	9164.5000000000000000	1800.0000000000000000	American, Continental, North Indian, Mediterranean
Arbor Brewing Company	8411.0000000000000000	2000.0000000000000000	American, Continental
Arbor Brewing Company	8379.2000000000000000	2000.0000000000000000	American, Continental, Salad
Prost Brew Pub	7860.9000000000000000	1800.0000000000000000	American, Continental, North Indian, Salad
The Black Pearl	7127.4000000000000000	1500.0000000000000000	North Indian, European, Mediterranean, BBQ
The Hole in the Wall Cafe	7124.8750000000000000	600.0000000000000000	Cafe, American, Burger
Truffles	7001.3720930232558140	900.0000000000000000	Cafe, American, Burger, Steak
Biergarten	6986.6000000000000000	2400.0000000000000000	Continental, European, BBQ, Chinese, Asian

- There is some difference between the cuisines offered by the TOP 10 highest rate non-online-order restaurants and the cuisines offered by the TOP 10 largest average vote non-online-order restaurants.
- Most of the restaurants with the largest average votes offered American food, but most of the restaurants with the highest rate offered Mediterranean and North Indian food.

In [65]:
```
%%sql
SELECT INFO.Rate as Rate, avg(INFO.approx_cost_two_people) as Cost FROM INFO
JOIN CUISINES ON
    CUISINES.Cuisines_Key = INFO.info_key
WHERE Rate > 0 AND Votes > 50 AND Cuisines LIKE '%Mediterranean%'
GROUP BY INFO.Rate
ORDER BY 1 DESC
LIMIT 10
```

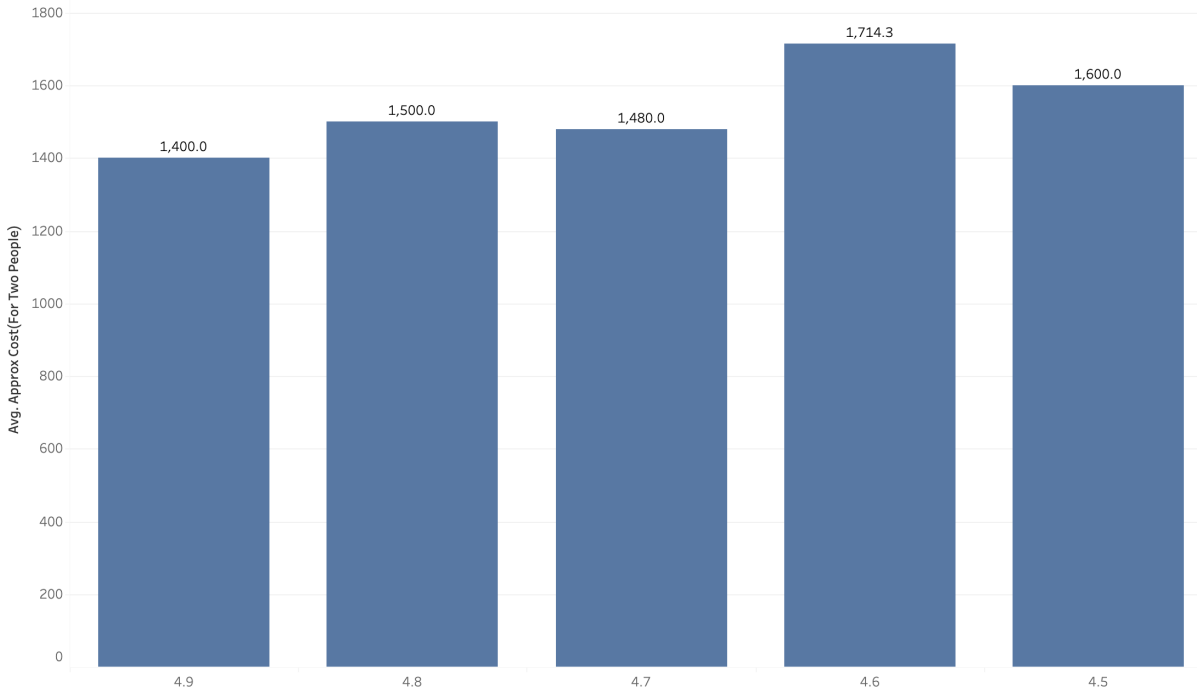
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[65]:

rate	cost
4.9	1490.9090909090909091
4.8	1554.5454545454545455
4.7	1539.5348837209302326
4.6	1837.5000000000000000
4.5	1407.4074074074074074
4.4	1415.9420289855072464
4.3	1657.6271186440677966
4.2	1415.2941176470588235
4.1	1288.8888888888888889
4.0	1722.2222222222222222

In [69]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206806116-1118fb
```

Out[69]:



In [70]:
```
%%sql
SELECT INFO.Rate as Rate, avg(INFO.approx_cost_two_people) as Cost FROM INFO
JOIN CUISINES ON
    CUISINES.Cuisines_Key = INFO.info_key
WHERE Rate > 0 AND Votes > 50 AND Cuisines LIKE '%Indian%'
GROUP BY INFO.Rate
ORDER BY 1 DESC
LIMIT 10
```

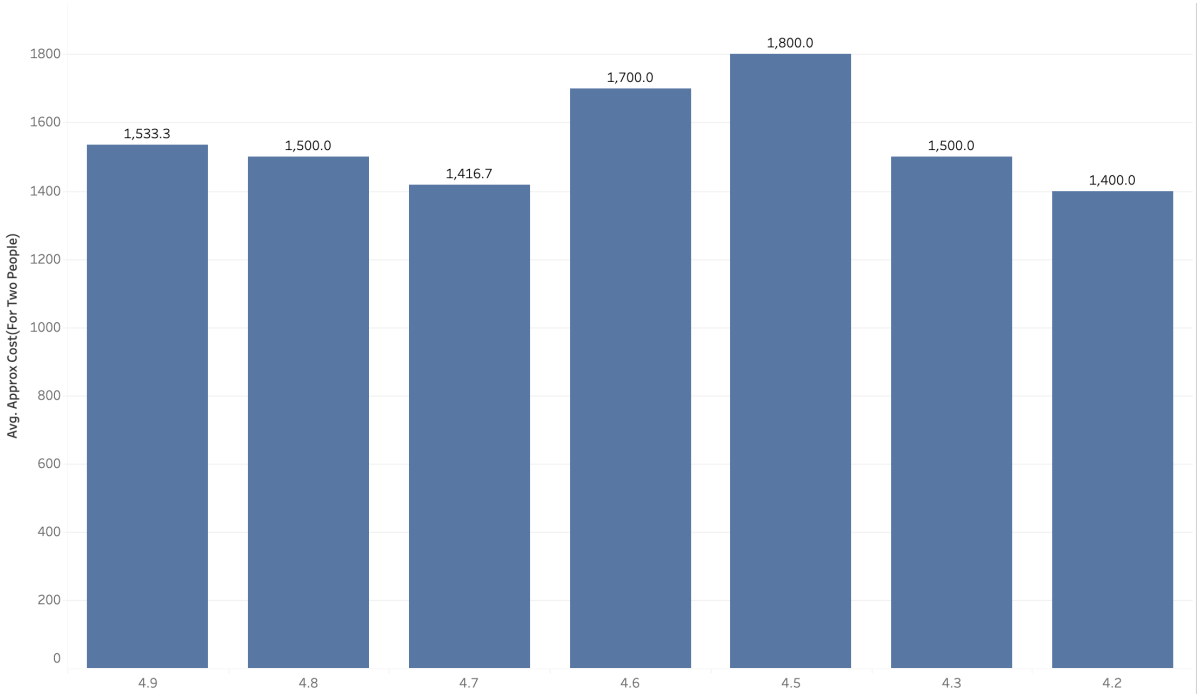
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[70]:

rate	cost
4.9	1666.6666666666666667
4.8	1518.1818181818181818
4.7	1297.7064220183486239
4.6	1393.4782608695652174
4.5	1205.6250000000000000
4.4	1212.1920668058455115
4.3	1207.5376884422110553
4.2	1017.8571428571428571
4.1	866.7255075022065313
4.0	788.5479688850475367

In [71]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206806183-8bcdec
```

Out[71]:



In [72]:
```
%%sql
SELECT INFO.Rate as Rate, avg(INFO.approx_cost_two_people) as Cost FROM INFO
JOIN CUISINES ON
    CUISINES.Cuisines_Key = INFO.info_key
WHERE Rate > 0 AND Votes > 50 AND Cuisines LIKE '%BBQ%'
GROUP BY INFO.Rate
ORDER BY 1 DESC
LIMIT 10
```

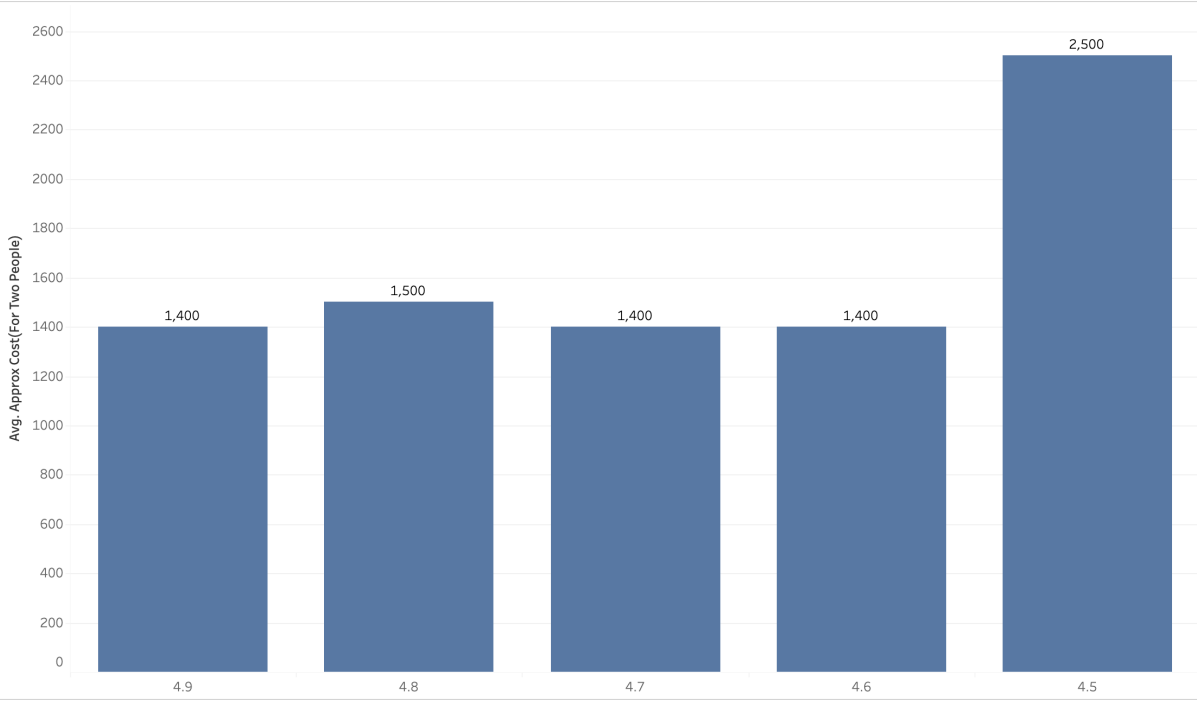
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[72]:

rate	cost
4.9	1540.0000000000000000
4.8	1707.1428571428571429
4.7	1653.8461538461538462
4.6	1425.0000000000000000
4.5	1888.4615384615384615
4.4	1336.5853658536585366
4.3	1577.1084337349397590
4.2	1300.0000000000000000
4.1	947.4137931034482759
4.0	856.5573770491803279

In [73]:
```
from IPython.display import Image
Image(url="https://user-images.githubusercontent.com/111759300/206806222-782b5c
```

Out[73]:



- The overall average cost per two people among the non-online-order resturants with the TOP 10 highest rate is around 1500.
- In order to make the result more representative, we break down the TOP 10 non-online-order highest rate resturants' average cost per two people based on cuisine and compared the cost with the average cost of all the resturants sharing the same rate and offering the same cuisine.
- 5 out of 10 non-online-order highest rate resturants offered BBQ, but after comparing the average cost per two people with all the restaurants offered BBQ and had a rate over 4, we believe that the cost in the highest rate resturants is a little bit higher than

the overall average price. Thus, we can say that the price is not that reasonable for BBQ cuisine in the TOP 10 highest rate resturants without offering online order.

- Besides BBQ, we compared restaurant offered Mediterranean food and North Indian food, and find out that the price is quite reasonable.

Question 3: List out the most popular restaurant for certain cuisine. Is the restaurant popular simply because it is the only one offering this specific cuisine in the neighborhood? If not, how many competitors it has, and what's their rate?

In [74]:
```
%%sql
select a.address, CUISINES.cuisines from LOCATION
inner join INFO on LOCATION.location_key = INFO.info_key
inner join CUISINES on INFO.info_key = CUISINES.cuisines_key
inner join
(select AVG(i.rate) as rate,l.address from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where l.location in('Koramangala 5th Block','BTM','Indiranagar','HSR','Jayanaga
group by l.address,rate) a on LOCATION.address = a.address
group by a.address, CUISINES.cuisines
```

```
 * postgresql://student@/GP9
44 rows affected.
```

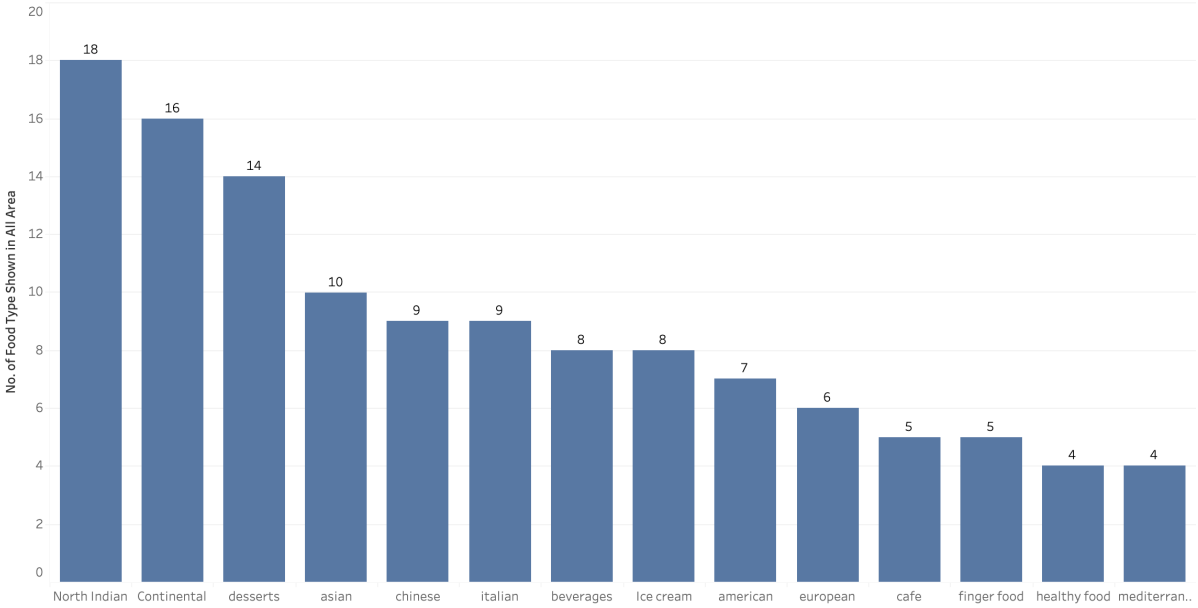
Out[74]:

address	cuisines
100 Feet Road, 1st Phase, Near Jayadeva Flyover, 2nd Stage, BTM, Bangalore	European, Mediterranean, North Indian, BBQ
100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore	Continental, Asian, Italian, North Indian
100/1B, 10th Cross, 2nd Main, Indiranagar, Bangalore	Ice Cream, Desserts
105, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore	North Indian, European, Mediterranean
105, 1st A Cross, Koramangala 5th Block, Bangalore	Desserts, Beverages
1085, 14th Main, 18th Cross, Sector 3, HSR, Bangalore	Continental, Asian, North Indian
1085, 14th Main, 18th Cross, Sector 3, HSR, Bangalore	Finger Food, Continental, North Indian
1131, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore	Desserts, Cafe, Beverages
12, 17th Main, 1st Cross, 5th A Block, Koramangala 5th Block, Bangalore	Italian
1209, 100 Feet Road, Opposite Apollo Clinic, Indiranagar, Bangalore	European, Italian, Desserts, Salad, Juices, Steak
122/B, Jyothi Nivas Road, 5th Block, Koramangala 5th Block, Bangalore	Continental, Asian, North Indian
13 KHB Colony, 17th Main, M.I.G, Koramangala 5th Block, Bangalore	Desserts
130, 17th H Main Road, Koramangala 5th Block, Bangalore	Desserts, Fast Food
136, Ground Floor, 1st Cross, 5th Block, Jyoti Niwas College Road, Koramangala 5th Block, Bangalore	Asian, Chinese, Thai, Momos
146, Near William Penn, Koramangala 5th Block, Bangalore	Ice Cream, Desserts
151, 2nd Cross, Domlur 2nd Stage, Indiranagar, Bangalore	Healthy Food, Salad, Mediterranean
17th Main Road, JNC Road, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, American
17th Main Road, JNC Road, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, American, Pizza, Finger Food
1st Stage, 100 feet Road, Indiranagar, Bangalore	Cafe, American, Burger, Steak
2008, 2nd Floor, 100 Feet Road, Indiranagar, Bangalore	Chinese, Continental, North Indian, Mexican
24, 46th Cross, 5th Block Jayanagar, Bangalore	North Indian, Chinese, Mughlai, Mexican, BBQ
253, 1st Floor, 5th Main, 17th Cross, Sector 6, HSR Layout, HSR, Bangalore	Continental, Finger Food, Asian, Chinese
253, 2nd Floor, 5th Main, 17th Cross, Sector 6, HSR, Bangalore	Chinese, Continental, North Indian, Finger Food
28, 4th 'B' Cross, Koramangala 5th Block, Bangalore	Cafe, American, Burger, Steak
298, Namma Metro Pillar 62, 100 Feet Road, Indiranagar, Bangalore	Italian, American, Pizza
2985, 12th Main, HAL 2nd Stage, Indiranagar, Bangalore	Italian, Pizza, Beverages

	address	cuisines
	2nd Floor, 1st A Cross Road, Jyothi Nivas College Road, Koramangala 5th Block, Bangalore	European, Continental
	3283, 2nd Stage, Off Double Road, Indiranagar, Bangalore	Ice Cream, Desserts
	4005, HAL 2nd Stage, 100 Feet Road, Indiranagar, Bangalore	North Indian, European, Mediterranean, BBQ, Kebab
	460, 2nd Cross, Krishna Temple Road, Indiranagar, Bangalore	Ice Cream, Desserts
	4th B Cross, Koramangala 5th Block, Bangalore	Continental, North Indian, Chinese, European, BBQ, Finger Food, Asian
	607, Ground Floor, 12th Main, Hal 2nd Stage, Indiranagar, Bangalore	Asian, Burmese
	610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore	Asian, Thai, Vietnamese, Malaysian, Beverages
	610, 3rd Floor, 12th Main, Off 80 Feet Road, Indiranagar, Bangalore	Japanese, Sushi, Asian
	615/1, Ground Floor, Janardhan Mansion, 10th C Main, 32nd D Cross, 4th Block, Jayanagar, Bangalore	Cafe, Beverages
	93/A 4th 'B' Cross, Koramangala 5th Block, Bangalore	Healthy Food, North Indian, Biryani, Continental, Sandwich, Desserts
	96, 29th Main, 23rd Cross, 2nd Stage, BTM, Bangalore	Healthy Food, North Indian, Biryani, Continental, Desserts
	96, 29th Main, 23rd Cross, 2nd Stage, BTM, Bangalore	Healthy Food, North Indian, Biryani, Continental, Sandwich, Desserts
	960, Next To Gold Gym,12thMain, HAL 2nd Stage, Indiranagar, Bangalore	Italian
	Astra Hotel, 2795, 27th Main, Sector 1, HSR, Bangalore	Continental, Asian, Italian, North Indian
	Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore	Ice Cream, Beverages
	Next to Apple Of My Eye, 12th Main, 2nd Stage, HAL, Off 100 Feet Road, Indiranagar, Bangalore	Ice Cream, Desserts
	Shop 28, Opposite BDA Complex, 14th Main Road, Sector 2, HSR Layout, Bangalore	Cafe, American, Burger, Beverages
	Shop 44, 4th B Cross Road, Koramangala 5th Block, Bangalore	Chinese, American, Continental, Italian, North Indian

In [75]:
```
from IPython.display import Image
Image(url='https://user-images.githubusercontent.com/111759300/206810685-1171db
```

Out[75]:



Based on Q1, we select the cuisines from the top 5 locations and count the frequency of each cuisine. From the chart, we pick the top 3 popular cuisines as the object of our analysis, which are North Indian, continental and desserts.

In [76]:
```
%%sql
select r.name, l.location,AVG(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where  c.cuisines like '%North Indian%'and i.votes >50 and i.rate not in (-999,
group by r.name,l.location
order by rate DESC
limit 5
```

```
 * postgresql://student@/GP9
5 rows affected.
```

Out[76]:

name	location	rate
Flechazo	Whitefield	4.9
Punjab Grill	Malleshwaram	4.9
AB's - Absolute Barbecues	BTM	4.8999999999999995
Byg Brewski Brewing Company	Sarjapur Road	4.8999999999999995
Punjab Grill	Whitefield	4.833333333333333

In [77]:
```
%%sql
select count(distinct(r.name)) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%North Indian%'and i.votes >50 and i.rate not in (-999,(
and l.location in ('Whitefield')
```

```
 * postgresql://student@/GP9
1 rows affected.
```

Out[77]:

rate
162

In [78]:
```
%%sql
select count(distinct(r.name)) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%North Indian%'and i.votes >50 and i.rate not in (-999,0
and l.location in ('Malleshwaram')
```

```
 * postgresql://student@/GP9
1 rows affected.
```

Out[78]:

rate
43

In [79]:
```
%%sql
select r.name, avg(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%North Indian%'and i.votes >50 and i.rate not in (-999,0
and l.location in ('Whitefield','Malleshwaram')
group by r.name
order by rate DESC
limit 10
```

```
 * postgresql://student@/GP9
10 rows affected.
```

Out[79]:

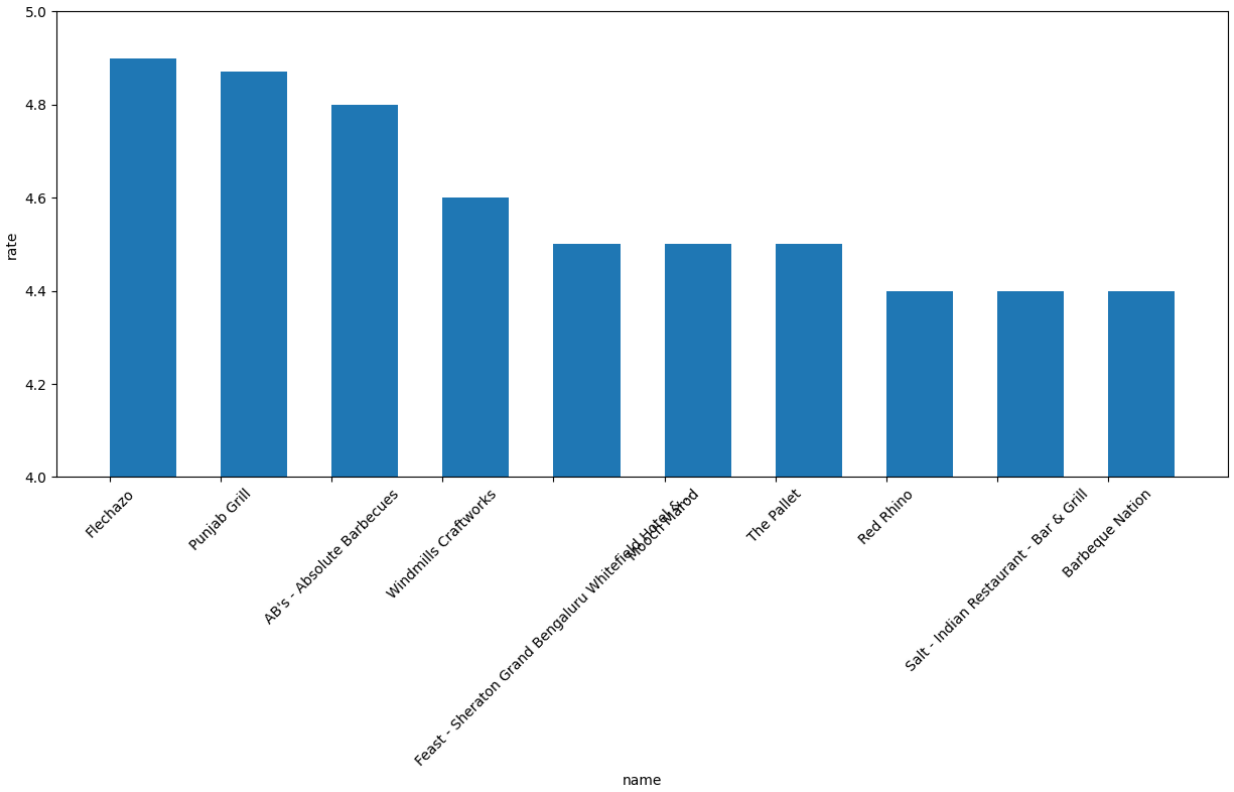
name	rate
Flechazo	4.9
Punjab Grill	4.871428571428571
AB's - Absolute Barbecues	4.8
Windmills Craftworks	4.6
Feast - Sheraton Grand Bengaluru Whitefield Hotel &...	4.5
Mooch Marod	4.5
The Pallet	4.5
Red Rhino	4.4
Salt - Indian Restaurant - Bar & Grill	4.4
Barbeque Nation	4.4

In [80]:
```
import matplotlib
from matplotlib import pyplot as plt
%matplotlib inline
```

In [81]:
```
plt.figure(figsize=(15, 6))
_.bar(width = 0.6,align = 'edge')
```

```
plt.ylim(4, 5)
```

Out[81]: (4.0, 5.0)



The results show that Flechazo and Punjab Grill are best restaurants for North Indian and they are located in Whitefield and Malleshwaram respectively with 161 competitors in Whitefield and 42 competitors in Malleshwaram. The table shows the top 10 rated restaurants in these two areas that serve North Indian.

In [82]:
```
%%sql
select r.name, l.location,AVG(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where  c.cuisines like '%Continental%'and i.votes >50 and i.rate not in (-999,0
group by r.name,l.location
order by rate DESC
limit 5
```

```
 * postgresql://student@/GP9
5 rows affected.
```

Out[82]:

name	location	rate
Byg Brewski Brewing Company	Sarjapur Road	4.8999999999999995
The Globe Grub	Marathahalli	4.8
The Boozy Griffin	Marathahalli	4.8
Biergarten	Koramangala 5th Block	4.799999999999999
House Of Commons	Koramangala 5th Block	4.727777777777779

In [83]:
```
%%sql
select count(distinct(r.name)) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%Continental%'and i.votes >50 and i.rate not in (-999,0)
and l.location in ('Sarjapur Road')
```

```
 * postgresql://student@/GP9
1 rows affected.
```

Out[83]:

rate
19

In [84]:
```
%%sql
select r.name, avg(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%Continental%'and i.votes >50 and i.rate not in (-999,0)
and l.location in ('Sarjapur Road')
group by r.name
order by rate DESC
limit 10
```

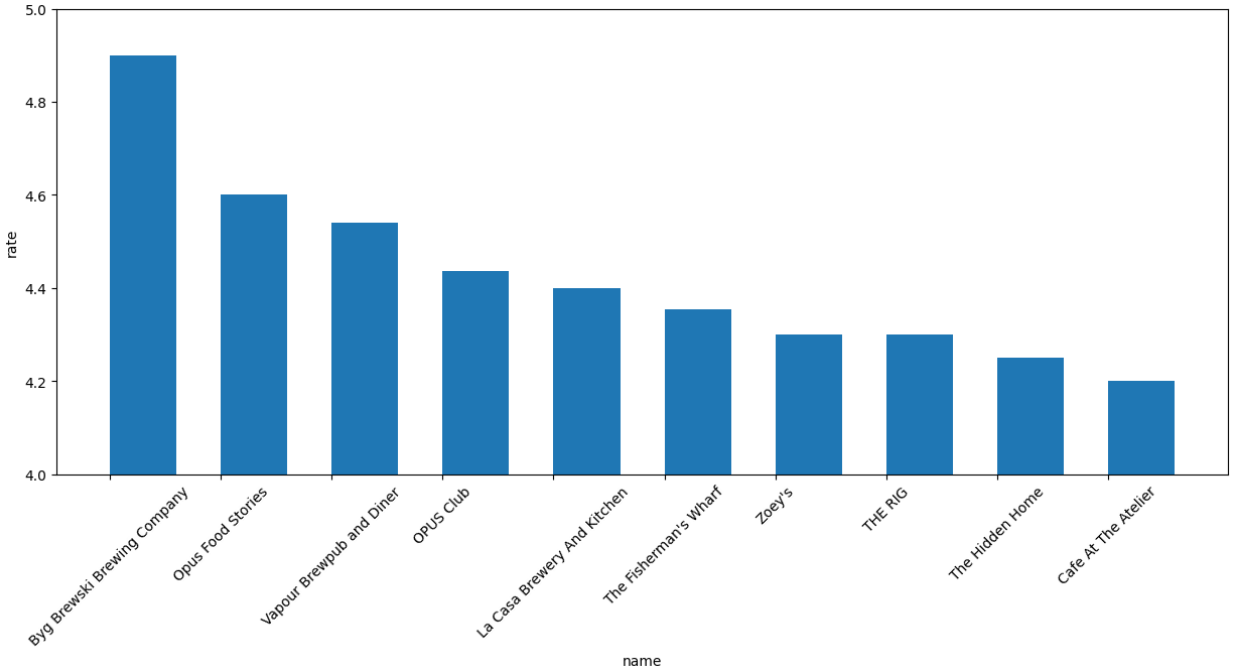
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[84]:

name	rate
Byg Brewski Brewing Company	4.8999999999999995
Opus Food Stories	4.6
Vapour Brewpub and Diner	4.54
OPUS Club	4.4363636363636365
La Casa Brewery And Kitchen	4.3999999999999995
The Fisherman's Wharf	4.353846153846153
Zoey's	4.3
THE RIG	4.3
The Hidden Home	4.25
Cafe At The Atelier	4.2

In [85]:
```
plt.figure(figsize=(15, 6))
_.bar(width = 0.6,align = 'edge')
plt.ylim(4, 5)
```

Out[85]:
```
(4.0, 5.0)
```



The results show that Byg Brewski Brewing Company is the best restaurant for Continental and it is located in Sarjapur Road with 18 competitors. The table shows the top 10 rated restaurants in that area that serve Continental.

In [86]:
```
%%sql
select r.name, l.location,AVG(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where  c.cuisines like '%Desserts%'and i.votes >50 and i.rate not in (-999,0)
group by r.name,l.location
order by rate DESC
limit 5
```

```
 * postgresql://student@/GP9
5 rows affected.
```

Out[86]:

name	location	rate
Milano Ice Cream	Indiranagar	4.9
Belgian Waffle Factory	Brigade Road	4.8999999999999995
Belgian Waffle Factory	Koramangala 5th Block	4.809090909090909
Belgian Waffle Factory	Kalyan Nagar	4.8
O.G. Variar & Sons	Rajajinagar	4.8

In [87]:
```
%%sql
select count(distinct(r.name)) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%Desserts%'and i.votes >50 and i.rate not in (-999,0)
and l.location in ('Indiranagar')
```

```
 * postgresql://student@/GP9
1 rows affected.
```

Out[87]:

rate
49

In [88]:
```
%%sql
select r.name, avg(i.rate) as rate from INFO i
inner join LOCATION l on i.info_key = l.location_key
inner join RESTAURANT r on i.info_key = r. restaurant_id
inner join CUISINES c on i.info_key = c.cuisines_key
where c.cuisines like '%Desserts%'and i.votes >50 and i.rate not in (-999,0)
and l.location in ('Indiranagar')
group by r.name
order by rate DESC
limit 10
```

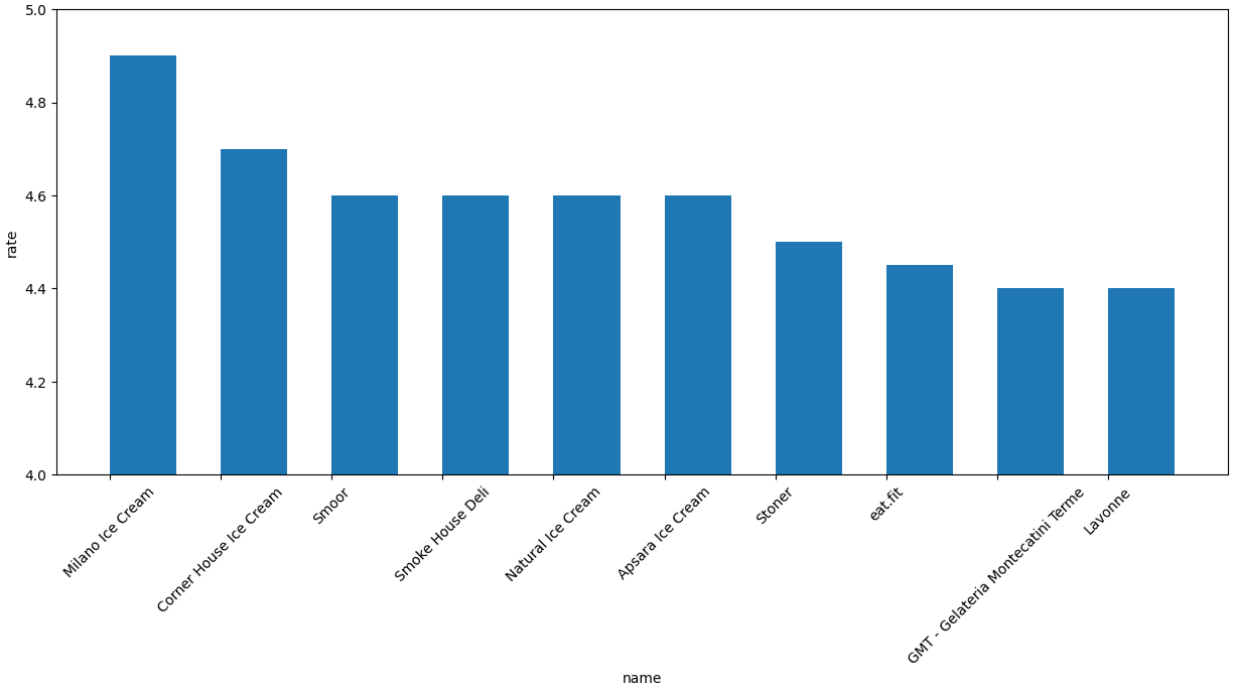
```
 * postgresql://student@/GP9
10 rows affected.
```

Out[88]:

name	rate
Milano Ice Cream	4.9
Corner House Ice Cream	4.7
Smoor	4.6000000000000005
Smoke House Deli	4.6
Natural Ice Cream	4.6
Apsara Ice Cream	4.6
Stoner	4.5
eat.fit	4.45
GMT - Gelateria Montecatini Terme	4.4
Lavonne	4.4

In [89]:
```
plt.figure(figsize=(15, 6))
_.bar(width = 0.6,align = 'edge')
plt.ylim(4, 5)
```

Out[89]:
```
(4.0, 5.0)
```



The results show that Milano Ice Cream is the best restaurant for Desserts and it is located in Indiranagar with 48 competitors. The most competiable competitor it has is COrner House Ice Cream since their rating is close. The table shows the top 10 rated restaurants in that area that serve Desserts.