COL362: Application Project Milestone 1

Vishwas Kalani 2020CS10411
Aman Bansal 2020CS10319
Nischay Diwan 2020CS50433

March 2023

Contents

1 ER Diagram

1.1 Assumptions concerned with ER diagram

Some assumptions while making ER diagram are as follows:

1. Whenever we want to show a relation between two entities and there exists a corresponding relational table in our schema then we have shown the connecting attributes for the two entities.
2. If the entities are related in the underlying schema but there is no table which acts as a link between the entities then we have simple shown the meaning of the relation between two entities without mentioning the attributes.
3. Apart from these all the rules for the ER diagram are followed in the diagram below :

1.2 Listing of Entities and Relations in ER diagram

There are nine entities in our ER Diagram namely:

1. Restaurant
2. City
3. Currency
4. Rating
5. Average_cost_for_two
6. Food_table
7. Meal_type_details
8. Meal_table
9. User

There are eight relations (not all are tables in final schema) :

1. restaurant_cuisine (separate table in final schema)
2. Rest_in_city
3. Country_currency (separate table in final schema)
4. Restaurant_price_range
5. Restaurant_has_rating
6. Food_of_type
7. Meal_of_type
8. Meal_taken_by

1.3 ER diagram

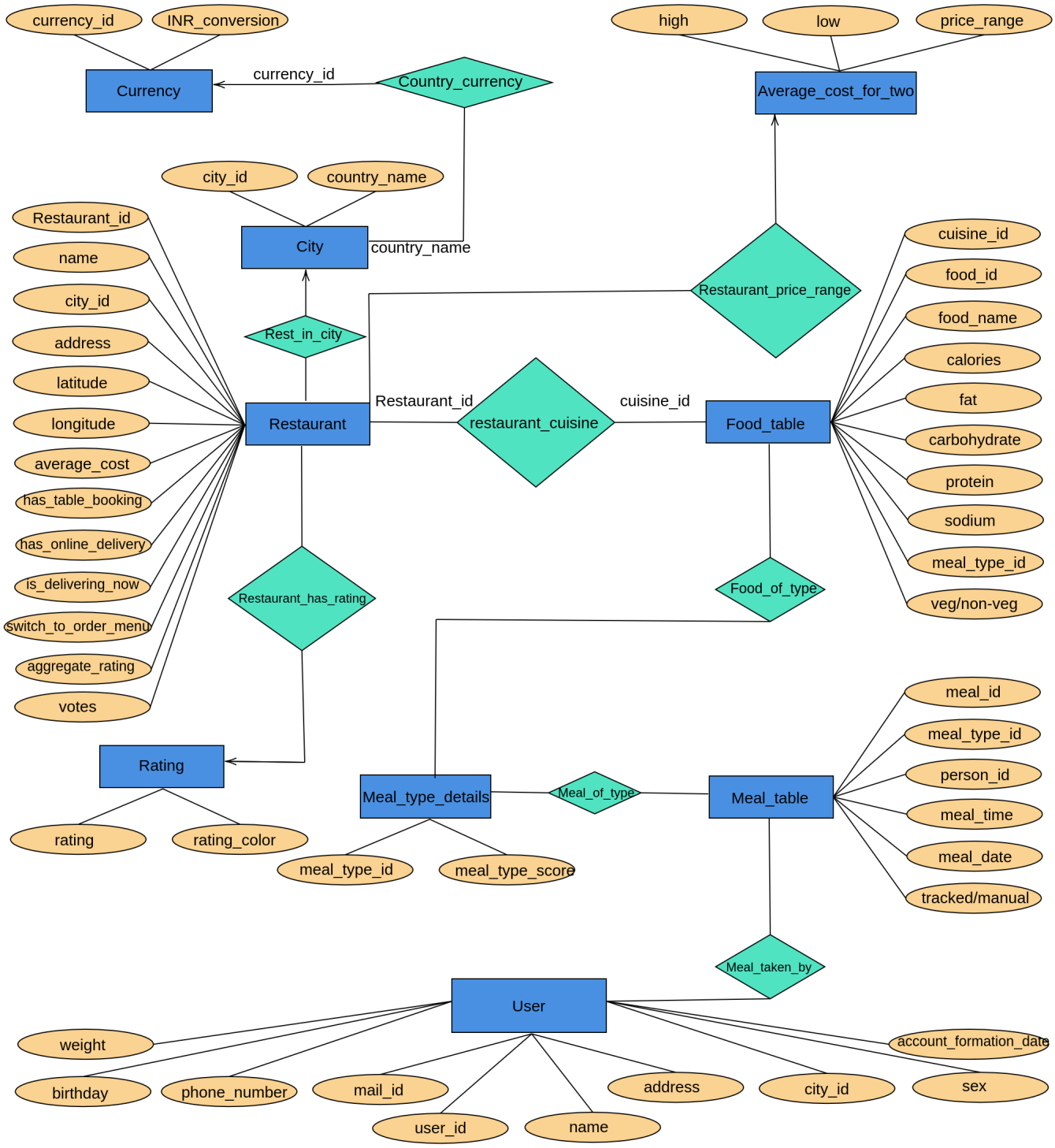

Figure 1: ER Diagram

2 Functional dependencies

We chose to have our relations in *BCNF* to ensure data integrity and avoid redundancy. We will decompose our original relation into smaller relations.

2.1 Original relations and functional dependencies

- **Restaurant** (Restaurant_id, Name, City_id, Country_name, Currency_id, Inr_conversion, Address, Latitude, Longitude, Cuisine_id, Avg_cost_for_two, Has_table_booking, Has_online_delivery, Is_delivering_now, Switch_to_order_menu, Aggregate_rating, Votes, Price_range, Rating_colour)

  Functional dependencies:

  1. Restaurant_id, Cuisine_id → Name, City_id, Country_name, Currency_id, Inr_conversion, Address, Latitude, Longitude, Avg_cost_for_two, Has_table_booking, Has_online_delivery, Is_delivering_now, Switch_to_order_menu, Aggregate_rating, Votes, Price_range, Rating_colour

     *(**Note** : This functional dependency can be used to create many functional dependencies, since (Restaurant_id, Cuisine_id) is a **primary key** and thus leads to formation of many super keys to form many functional dependencies. Similar functional dependencies can be formed using (Latitude,Longitude,Cuisine_id) as well.These don't violate BCNF)*
  2. Restaurant_id → Name, City_id, Country_name, Currency_id, Inr_conversion, Address, Latitude, Longitude, Avg_cost_for_two, Has_table_booking, Has_online_delivery, Is_delivering_now, Switch_to_order_menu, Aggregate_rating, Votes, Price_range, Rating_colour (Violate BCNF)

     *(**Note** : As multiple cuisine ids are possible for a restaurant, we can see due to this functional dependency, there is a lot of redundant data in table thus we need to decompose the table while preserving this functional dependency. Following are more dependencies which violate BCNF.)*
  3. City_id → Country_name (Violates BCNF)
  4. Country_name → Currency_id (Violates BCNF)
  5. Currency_id → Inr_conversion (Violates BCNF)
  6. Avg_cost_for_two → Price_range (Violates BCNF)
  7. Aggregate_rating → Rating_colour (Violates BCNF)

- **Food**(Food_id, Food_name, Cuisine_id, Calories, Fat, Carbohydrates, Protein, Sodium, Meal_type_id, Veg_non_veg)

  Functional dependencies:

  1. Food_id → Food_name, Cuisine_id, Calories, Fat, Carbohydrates, Protein, Sodium, Meal_type_id, Veg_non_veg
     *(**Note** : This functional dependency can be used to create many functional dependencies, since (Food_id) is a **primary key** and thus leads to formation of many super keys to form many functional dependencies. Since each food item has a unique entry in table and there are no redundant information in this table, it does't violate BCNF)*

- **Meal**(Meal_id, Meal_type_id, Person_id, Meal_time, Meal_date, Entry_type,Meal_score)

  Functional dependencies:

  1. Meal_id → Meal_type_id, Person_id, Meal_time, Meal_date, Entry_type,Meal_score
     *(**Note** : This functional dependency can be used to create many functional dependencies, since*

*(Meal_id) is a **primary key** and thus leads to formation of many super keys to form many functional dependencies. This does't violate BCNF)*

2. Meal_type_id → Meal_score
   *(**Note** : Each meal has a meal type. The meal type uniquely determines the score of each meal and thus it is not needed to store multiple same values of score corresponding to the same values of meal type This violates BCNF)*

- **User Table** (User_id,Name,Address,City_id,Mail_id,Phone_number,Birthday,Sex,Weight, Account_creation_date)

  Functional dependencies:

  1. User_id → Name,Address,City_id,Mail_id,Phone_number,Birthday,Sex,Weight, Account_creation_date
     *(**Note** : This functional dependency can be used to create many functional dependencies, since (User_id) is a **primary key** and thus leads to formation of many super keys to form many functional dependencies. Simialar functional dependencies can be formed using Mail_id and Phone_number as well which are **candidate keys**. Each entry in the table is unique for every user and there is no redundant data thus table doesn't require further decomposition. This does't violate BCNF)*

2.2 Decomposition into smaller relations while preserving functional dependencies

Since there were several functional dependencies due to which 2 of our tables are not in *BCNF* normal form thus we will decompose the table so that we are able to preserve the functional dependency as well as get smaller relational tables which are in appropriate normal form. Procedure for decomposition of relations in BCNF form is that for functional dependency $\alpha \rightarrow \beta$, we include $\alpha \cup \beta$ and $R - (\beta - \alpha)$ and we continue this process until we have all the relations in normal form. Finally we have following 11 tables which also satisfy the functional dependencies we have mentioned above:

- **Restaurant**(Restaurant_id, Name, City_id, Address, Latitude, Longitude, Avg_cost_for_two, Has_table_booking, Has_online_delivery, Is_delivering_now, Switch_to_order_menu, Aggregate_rating, Votes)
- **Avg_Cost_for_Two**(High, Low, Price_range)
- **Rating**(Rating, Rating_colour)
- **Restaurant_Cuisine**(Restaurant_id, Cuisine_id)
- **City**(City_id, Country_name)
- **Country_Currency**(Country_name, Currency_id)
- **Food**(Food_id, Food_name, Cuisine_id, Calories, Fat, Carbohydrates, Protein, Sodium, Meal_type_id, Veg_non_veg)
- **Meal**(Meal_id, Meal_type_id, Person_id, Meal_time, Meal_date, Entry_type)
- **Meal_Type_Details**(Meal_type_id, Meal_type_score)
- **Currency**(Currency_id, Inr_conversion)
- **User**(User_id, Name, Address, City_id, Mail_id, Phone_number, Birthday, Sex, Weight, Account_creation_date)

3 Relational Schema

Figure 2: Schema Representation

4 Description of schema and their attrbutes

Restaurant Table:

- Restaurant_id: a unique identifier for each restaurant in the database
- Name: the name of the restaurant
- City_id: a foreign key linking to the city table, indicating which city the restaurant is located in
- Address: the physical address of the restaurant
- Latitude: the latitude of the restaurant's location
- Longitude: the longitude of the restaurant's location
- Avg_cost_for_two: the average cost for two people to dine at the restaurant
- Has_table_booking: a binary attribute indicating whether the restaurant takes table bookings or not
- Has_online_delivery: a binary attribute indicating whether the restaurant offers online delivery or not
- Is_delivering_now: a binary attribute indicating whether the restaurant is currently delivering food or not
- Switch_to_order_menu: a link to the restaurant's online ordering system
- Aggregate_rating: the average rating of the restaurant, based on customer reviews
- Votes: the number of customer reviews that have been left for the restaurant

Avg Cost for Two Table:

- High: The highest price for which the restaurant lies in this range
- Low: The lowest price for which the restaurant lies in this range
- Price_range: a description of the price range, e.g. "low budget","budget", "mid-range","fine dining", "royal"

Rating Table:

- Rating: the numerical rating assigned to a restaurant, e.g. 4.5 out of 5 stars
- Rating_colour: a visual representation of the rating, e.g. a color-coded system that uses green for high ratings and red for low ratings

Restaurant Cuisine Table:

- Restaurant_id: a foreign key linking to the restaurant table, indicating which restaurant the cuisine is associated with
- Cuisine_id: a foreign key linking to the cuisine table, indicating which type of cuisine the restaurant specializes in

City Table:

- City_id: a unique identifier for each city in the database
- Country_name: the name of the country the city is located in

Country Currency Table:

- Country_name: a unique identifier for each country in the database
- Currency_id: the code for the currency used in the country

Food Table:

- Food_id: a unique identifier for each food item in the database
- Food_name: the name of the food item
- Cuisine_id: a foreign key linking to the cuisine table, indicating which type of cuisine the food item belongs to
- Calories: the number of calories in the food item per 100 g of food item
- Fat: the amount of fat in the food item per 100 g of food item
- Carbohydrates: the amount of carbohydrates in the food item per 100 g of food item
- Protein: the amount of protein in the food item per 100 g of food item
- Sodium: the amount of sodium in the food item per 100 g of food item
- Meal_type_id: a foreign key linking to the meal type details table, indicating which meal type the food item is associated
- Veg_non_veg: a binary attribute indicating whether the food item is vegetarian or non-vegetarian

Meal Table:

- Meal_id: a unique identifier for each meal in the database
- Meal_type_id: a foreign key linking to the meal type details table, indicating which meal type the meal is associated with (we have following 8 types : Fruits, Wheat based, Rice based, Meat based, Sea food based, Salad based, Milk based, Junk)
- Person_id: a foreign key linking to the user table, indicating which person the meal is associated with
- Meal_time: the time of day the meal was eaten
- Meal_date: the date the meal was eaten
- Entry_type: a binary attribute indicating whether the meal was automatically tracked from site or added manually

Meal Type Details Table

- Meal_type_id: unique identifier for the meal type
- Meal_type_score: a score assigned to the meal type based on its healthiness or other criteria

Currency Table

- Currency_id: unique identifier for the currency
- Inr_conversion: the conversion rate from the local currency to Indian Rupees

User Table

- User_id: unique identifier for the user
- Name: the name of the user

- Address: the address of the user
- City_id: the identifier of the city where the user lives
- Mail_id: the email address of the user
- Phone_number: the phone number of the user
- Birthday: the date of birth of the user
- Sex: the gender of the user
- Weight: the weight of the user
- Account_creation_date: the date when the user's account was created