**Food App Database Design Documentation**

This template provides a structure for documenting the design of a relational database.

**Food App Analysis**

***Customer:***

* Add items: User can add items to his cart
* Favorite items: User can add items to favorite list
* Place order: User can checkout his cart and place order
* Rating: After order status becomes *delivered* user can rate that item

***Food Seller:***

* Sell items: A seller can place at least one or many dishes
* Completed order-list: A seller would have a list of completed orders
* Add discount: A seller can add discount to particular dishes

**Entities & Attributes:**

* **Customer:**
  + *customer\_id, name, address, email, favorite\_items, cart\_items, order\_history, phone\_no.*
* **Food Seller:**
  + *seller\_id, name, address, email, food\_items, completed\_orders*
* **Dish (Food-item):**
  + *dish\_id, name, price, discount, rating, comments, seller\_id, description*

**Tables:**

* *Users, Sellers, Orders*

**Relationships**

1. ***Customer – Dish (One-to-Many):***
   * **Customer:** One customer can place many orders.
   * **Order:** Each order is placed by one customer.
   * **Relationship:** Each customer can have multiple orders, but each order belongs to only one customer.
2. ***Food Seller – Dish (one-to-many):***
   * **Food Seller:** One Seller can have many orders.
   * **Order:** Each order is associated with one seller.
   * **Relationship:** Each seller can have multiple orders, but each order would be tied to only one seller.
3. ***Order******– Food Item (One-to-Many):***
   * **Order:**  An order can contain one or more food items.
   * **Quantity:**It will keep the quantity of item ordered.
   * **Food Item:**Each food item would belong to one order only.
   * **Relationship:** Each order can include multiple food items, but each food item belongs to only one order.