ONLINE FOOD DELIVERY – BOTTOM-UP APPROACH

Submitted to: Prof. Ashok Harnal

Submitted by: Group 15
Yog Raj Singh (025040)
Yogesh Sachdeva (025041)
Manyata Manocha (025053)

INTRODUCTION

In this project, we have picked up a complex form data of online food delivery database to extract pieces of information from this form and have Organized the extracted pieces of information (attributes) into a table. We have followed these steps:

i) Normalize the resulting table following rules of normalization up to 3NF. In this process, we have made certain assumptions about some attributes or about relationships among them, which have been clearly stated in the project. ii) Translate the tables to MySQL (SQL) code using Data Definition language. Select suitable data types. iii) Apply proper database constraints to attributes in the coded tables. iv) Create various users of the database. Define their roles and assign them proper authority. Translate these roles to MySQL-code using SQL Data Control language. v) Run the code in the MySQL server to create all objects and users with proper role definitions. vi) Use MySQL server Workbench to create an ER diagram. vii) Finally prepared the report.

This repository contains
a) The starting form
b) Normalized tables--step-by-step
c) An SQL file with SQL code
d) ER diagram
e) Report containing your case, assumptions, etc.

Order ID	Date	Customer ID	Customer Name	Customer Phone	Restaurant_ID	Restaurant location	Food Price	Item Id	Item Ordered	Time for delivering	Delivery Partner	Partner ID
101	01-30-2022				721	Vasant Kunj	300	7214	Chicken Tikka	36	Brijesh Singh	Z334
112	01-31-2022	AA100	Harish Kumar	9878967232	721	Vasant Kunj	100	7216	Butter Naan	36	Brijesh Singh	Z334
102	01-30-2022				771	Malviya Nagar	250	7715	Paneer Tikka	40	Manoj Pal	Z322
114	01-31-2022	AA101	Harvinder Pal	9345678900	771	Malviya Nagar	100	7716	Chole Bathure	40	Manoj Pal	Z322
103	01-31-2022				321	Saket	80	3216	Pav Bhaji	44	Kushwant Singh	Z336
209	02-07-2022				721	Vasant Kunj	350	7217	Fish Fry	36	Brijesh Singh	Z334
299	02-18-2022	AA102	Kush Mehra	9876512345	112	Rohini	100	1128	Veg Noodles	40	Manohar Dubey	Z337
104	02-01-2022				112	Rohini	150	1125	Manchurian	40	Manohar Dubey	Z337
199	02-02-2022	AA103	Akhilesh Pal	8156789043	112	Rohini	350	1127	Chicken Tikka	40	Manohar Dubey	Z337
105	02-01-2022				543	Greater Kailash	550	5432	Butter Chicken	40	Brijesh Singh	Z334
188	02-02-2022				654	Chhatarpur	700	6541	Mutten Rogan Josh	44	Zoravar Singh	Z889
192	02-03-2022	AA104	Babjeet Singh	9711546789	654	Chhatarpur	150	6542	Dal Makhani	44	Zoravar Singh	Z889
106	02-02-2022				721	Vasant Kunj	100	7219	Chilly Potato	36	Brijesh Singh	Z334
109	02-03-2022	AA105	Ayush Srivastava	9876541234	721	Vasant Kunj	80	7214	Chicken Tikka	36	Brijesh Singh	Z334

FIRST NORMAL FORM:

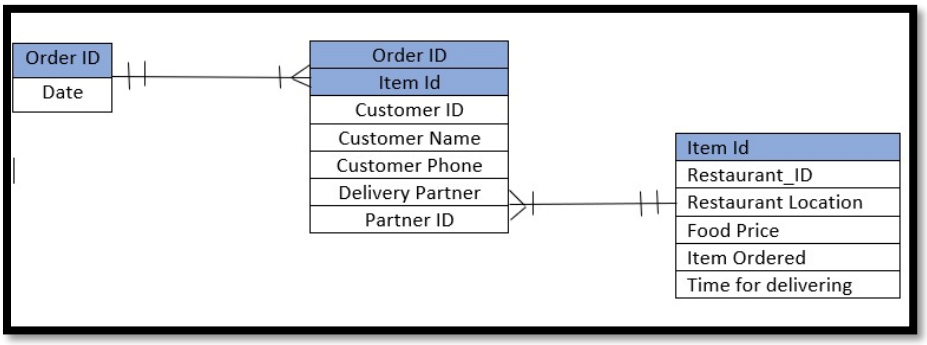
According to the first normal form, there should not be any nested tables for all groups of elements. For the table to satisfy the first normal form, the group was removed by giving the sets the values for their missing attributes. Therefore, after the first normalization the following table was obtained:

Order ID	Date	Customer ID	Customer Name	Customer Phone	Restaurant_ID	Restaurant location	Food Price	Item Id	Item Ordered	Time for delivering	Delivery Partner	Partner ID
101	01-30-2022	AA100	Harish Kumar	9878967232	721	Vasant Kunj	300	7214	Chicken Tikka	36	Brijesh Singh	Z334
101	01-30-2022	AA100	Harish Kumar	9878967232	721	Vasant Kunj	100	7216	Butter Naan	36	Brijesh Singh	Z334
102	01-31-2022	AA101	Harvinder Pal	9345678900	771	Malviya Nagar	250	7715	Paneer Tikka	40	Manoj Pal	Z322
102	01-31-2022	AA101	Harvinder Pal	9345678900	771	Malviya Nagar	100	7716	Chole Bathure	40	Manoj Pal	Z322
103	01-31-2022	AA102	Kush Mehra	9876512345	321	Saket	80	3216	Pav Bhaji	44	Kushwant Singh	Z336
209	02-07-2022	AA102	Kush Mehra	9876512345	721	Vasant Kunj	350	7217	Fish Fry	36	Brijesh Singh	Z334
299	02-18-2022	AA102	Kush Mehra	9876512345	112	Rohini	100	1128	Veg Noodles	40	Manohar Dubey	Z337
199	02-02-2022	AA103	Akhilesh Pal	8156789043	112	Rohini	150	1125	Manchurian	40	Manohar Dubey	Z337
199	02-02-2022	AA103	Akhilesh Pal	8156789043	112	Rohini	350	1127	Chicken Tikka	40	Manohar Dubey	Z337
105	02-01-2022	AA104	Babjeet Singh	9711546789	543	Greater Kailash	550	5432	Butter Chicken	40	Brijesh Singh	Z334
188	02-02-2022	AA104	Babjeet Singh	9711546789	654	Chhatarpur	700	6541	Mutten Rogan Josh	44	Zoravar Singh	Z889
188	02-03-2022	AA104	Babjeet Singh	9711546789	654	Chhatarpur	150	6542	Dal Makhani	44	Zoravar Singh	Z889
106	02-02-2022	AA105	Ayush Srivastava	9876541234	721	Vasant Kunj	100	7219	Chilly Potato	36	Brijesh Singh	Z334
106	02-03-2022	AA105	Ayush Srivastava	9876541234	721	Vasant Kunj	80	7214	Chicken Tikka	36	Brijesh Singh	Z334

Primary keys**: Order ID and Item ID**

SECOND NORMAL FORM:

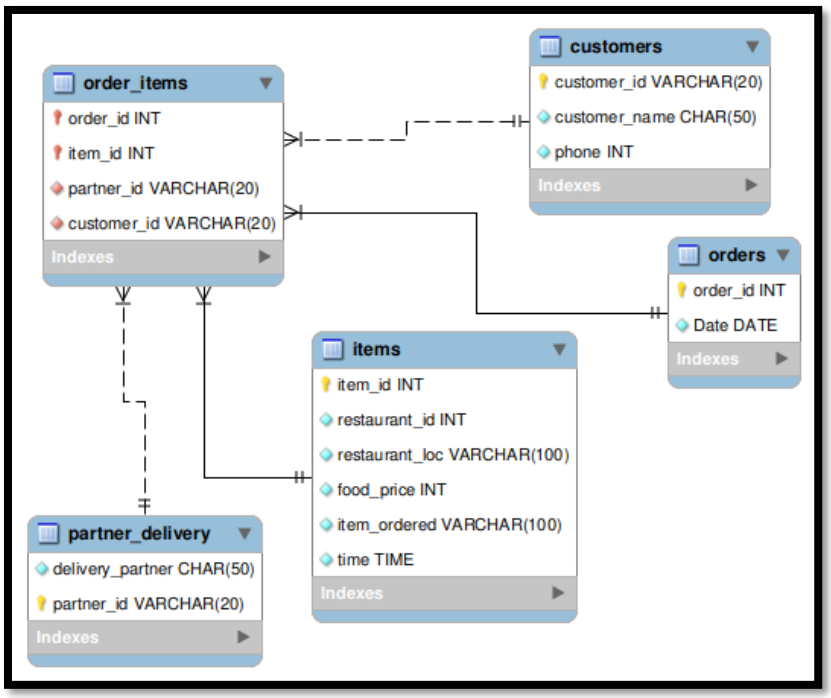
The second normal form says there should be no dependencies on a part of the composite key. And if any such exists in the table, it needs to be separated along with that primary key. After applying 2 NF, the tables generated are named as, orders, order_items and items.



THIRD NORMAL FORM:

The third normal form is about no dependencies on non-key attributes. And in our table, there are no dependencies on non-key attributes.

ER DIAGRAM:



ASSUMPTIONS:

1. Order ID is unique.
2. Customer ID is unique.
3. We are covering restaurants in Delhi.
4. Franchise restaurant chains have different restaurant IDs for different locations.

DDL COMMANDS FOR TABLES:

```
#0.
drop database if exists online_food ;
create database if not exists online_food ;
use online_food ;

#1.
CREATE TABLE IF NOT EXISTS orders (
order_id int PRIMARY KEY,
Date DATE NOT NULL);



#2.
CREATE TABLE IF NOT EXISTS items (
item_id int PRIMARY KEY,
restaurant_id int NOT NULL,
restaurant_loc VARCHAR(100) NOT NULL,
food_price int NOT NULL,
item_ordered VARCHAR(100) NOT NULL,
time TIME NOT NULL );



#3.
CREATE TABLE IF NOT EXISTS partner_delivery (
delivery_partner CHAR(50) NOT NULL,
partner_id VARCHAR(20) PRIMARY KEY);
#FOREIGN KEY (partner_id) REFERENCES order_items (partner_id));



#4.
CREATE TABLE IF NOT EXISTS order_items (
order_id int NOT NULL,
item_id int NOT NULL,
partner_id VARCHAR(20) NOT NULL,
customer_id VARCHAR(20) NOT NULL,
FOREIGN KEY (customer_id) REFERENCES customers (customer_id),
FOREIGN KEY (partner_id) REFERENCES partner_delivery (partner_id),
FOREIGN KEY (item_id) REFERENCES items (item_id),
```

```
FOREIGN KEY (order_id) REFERENCES orders (order_id),
PRIMARY KEY (order_id, item_id) );



#5.
CREATE TABLE IF NOT EXISTS customers (
customer_id VARCHAR(20) PRIMARY KEY,
customer_name CHAR(50) NOT NULL,
phone int NOT NULL );



show tables ;
##########################################
#Roles:
# dba: Yog Raj Singh
# admin: Yogesh Sachdeva                                                c
# restaurant: Akhil Khosla
# customer: Manyata Manocha
# delivery_partner: Dhruv Sharma

#Users Creation
CREATE USER 'yograj'@'localhost' IDENTIFIED BY 'yograj' ;
CREATE USER 'yogesh'@'localhost' IDENTIFIED BY 'yogesh' ;
CREATE USER 'manyata'@'localhost' IDENTIFIED BY 'manyata' ;
CREATE USER 'dhruv'@'localhost' IDENTIFIED BY 'dhruv' ;
CREATE USER 'akhil'@'localhost' IDENTIFIED BY 'akhil' ;

# delivery_partner
grant  select  on  online_food.partner_delivery  to
'dhruv'@'localhost'  ;
grant  select  on  online_food.items to 'dhruv'@'localhost'  ;


#dba
GRANT create,drop, select ON online_food.*  TO 'yograj'@'localhost'
with grant option ;

#admin
GRANT select, insert, update, delete ON online_food.*  TO
'yogesh'@'localhost' ;

#restaurant
GRANT select ON online_food.order_items TO 'akhil'@'localhost' ;
GRANT insert, update, delete ON online_food.items TO
'akhil'@'localhost' ;

#customer
GRANT select ON online_food.customers TO 'manyata'@'localhost' ;
GRANT select ON online_food.orders TO 'manyata'@'localhost' ;
GRANT select ON online_food.order_items TO 'manyata'@'localhost' ;
```

```
GRANT select ON online_food.items TO 'manyata'@'localhost' ;
```

SNAPSHOT OF THE FORM:
Below is the form which was used to collect the data for the project.