

Vietnam National University HCMC

International University

**School of Computer Science and Engineering**

**FINAL REPORT**

**Course: Principles of Database Management**

*Instructor: Prof. Nguyen Thi Thuy Loan*

**Group 11**

**TOPIC: FOOD DELIVERY APPLICATION**

|  |  |  |
| --- | --- | --- |
| **Group members:** | |  |
| **Name (Work contribution)** | **Student ID** | **Details** |
| Nguyễn Đăng Vũ Duy (27%) | ITITWE19021 | * Creating database (creating the remaining tables, all sample data) * Writing and editing the final report * Creating presentation slides * Assigning tasks to members |
| Trương Hùng Quân (27%) | ITITIU20068 | * In charge of front-end sources * Finishing ERD design, design RM * Connecting database with java and web application through Servlet and JDBC. * Apply CRUD with database management. |
| Vũ Hồng Tân (22%) | ITITIU18242 | * Designing ERD * Demonstrating login, register, homepage on websites * Creating customer and admin tables. |
| Nguyễn Hữu Hoàng (22%) | ITITIU19014 |  |
| Nguyễn Trần Duy Thanh (2%) | ITITIU18220 | * Graphical User Interface |

Table of Contents

* 1. Ideation4
     1. introduction4
     2. Application of dbms4
     3. Requirements of information saving within database4
  2. Establishing erd5
  3. relational model10
     1. relational model10
     2. user interface diagram13
  4. Sql instructions7
     1. creating database and tables15
     2. inserting data18
  5. querying examples8

1. **IDEATION**
   * 1. ***Introduction***

Nowadays, people are mostly surrounded by the hustle and bustle lifestyle that makes them forget to look after themselves, especially on their diets. Here, our online food system becomes convenient as customers can order online from their workplace and staff deliver fresh and hot food to customers' workplaces, households, etc. By operating 24/7 a week, not only will we deliver satisfaction to our customers’ eating demands but also efficiently provide enough energy for working.

Since the government in many countries applied the policy of self-quarantine for people infected by COVID-19, it could have never been more difficult for anyone who wants to choose what to feed their family daily but also healthily. Therefore, besides the capability of express delivery, our dedicated delivery service promises to provide fresh and hot meals within the allowed permission of distancing safeties. Through the guarantee of hygiene standards, we could confidently ensure that your social distancing will make no difference or even “tastier” than your casual days.

Main achievements of creating the app:

* Help provide an easy, free to use application that take little time to order food and manage users’ information via online sources in the most convenient way without costing any redundant paperwork.
* Reduce the amount of junk foods and the percentage of unemployment.
* Friendly interactive environment among all users.
* Acceptable costs of network, system maintenance, data management compared to total miscellaneous fees (e.g., marketing, transportation, renting facilities, etc.)

1. ***Application of DBMS:***

* Objects: restaurant owners, drivers, consumers, administrators*.*
* Controlling the behaviors and provides interfaces for all users
* Showing details of transactions, payment as well as importing and exporting them.
* Capability of providing statistic reports weekly/monthly/annually.

1. ***Requirements of information saving within database:***

The DBMS should be able to save the following data:

* Restaurants address, name, contact number.
* Food name, price, image, type.
* Drivers’ full name, license number, contact number.
* Transaction details such as its ID, payment type, customer ID, restaurant ID.
* Customers’ full name, email, username, address.

1. **ESTABLISHING ERD**

*(Related knowledge: Design Database ER, SQL)*

*Diagram

Description automatically generated*

*ERD of Food Delivery Application*

The diagram contains totally 10 entities.

**Relationship description:**

* One admin can monitor or control many restaurants and many customers/users (one to many relationship).
* One customer can place many orders (one to many relationship).
* One restaurant can has many foods (one to many relationship).
* One food can only belongs to one menu type (one to one relationship).
* OrderDetails contains many foods (one to many relationship).
* One order can have many order details (one to many relationship).
* One driver can only accpet one order once at a time. (one to one relationship).
* Many orders can have multiple transactions ( many to many relationship).
* Customer/Driver/Restaurant can have copies of the transactions (one to many relationship)

**Admin**

This is the top manager of the entire system. This entity grants the privilege of every action that happened in the system. It keeps track of all processes, such as payments, orders, transactions, and user joinings. Also, it manages all the other entities and the relationship between them. It includes the following attributes:

*Attributes:*

* **Admin\_ID:** The admin is given a unique ID to login into the system when having used their email to register an account. This ID is distinctive for all admins.
* **Username:** The admin authority is designated to more than one person to control the system in a better way. This is displayed on the application’s interface. No duplication is allowed
* **Password:** manually/automatically pre-defined secret keys when logging into the system.
* **Email:** Conversations, contacts, complaints, and other documents are sent via email which is the admin’s registered email. This email plays as login information as well.
  + *Roles:*
* Admin can ADD/EDIT/DELETE/UPDATE restaurant records.
* Admin can ADD/EDIT/DELETE/UPDATE User records.
* Admin can manage the food categories.
* Admin can manage the payment details.
* Admin can track the order records.

**User/Customer:**

The people who join and register in the system with membership access. This will record new and old users and users without verified information. The customers join the system by using their email and password to view their personal information and order food. It includes the following attributes:

*Attributes:*

* **Customer\_ID:** The customer login into the system through the distinct login credentials provided uniquely to every customer.
* **FullName:** The name of every customer is different and also the bill generated is by the name of the customer.
* **Password:** manually/automatically pre-defined secret keys when logging into the system.
* **Username:** This property is displayed on the user interface. No duplication is allowed.
* **Email:** This attribute holds the email address which customers use to create their account (login information). This email is used to inform them about the status of their order.
* **Address:** The destination where drivers can follow on the GPS and their phones to deliver food to customers.
* **Admin\_ID:** This is the id of the admin that is responsible for managing this user/customer account.

*Role:*

* Users can log in to the system with their email and password
* Users can ADD/EDIT/DELETE personal orders.
* Users can give the payment information for a particular order.
* Users are managed by the Admin.

**Restaurant:**

Owners of restaurants who want to start a business in delivering service. They are able to upload their menu, images, and prices related to their business. Also, their UI is designed for easily managing the order details of every customer. It includes the following attributes:

*Attributes:*

* **Name:** This attribute saves the name of every restaurant attached to the system along with other details.
* **Restaurant\_ID:** The restaurant login into the system through the distinct login credentials provided uniquely to every restaurant.
* **Address:** The location of every restaurant is saved in this attribute so that customers could choose the nearby ones to their place.
* **Contact no:** Every restaurant provides their contact number while delivering for giving the support needed by the customer.
* **Admin\_ID:** This is the id of the admin that is responsible for managing this restaurant account.

*Role:*

* Restaurant owners can ADD/EDIT/DELETE the menu
* Restaurant owners are managed by Admin.
* Restaurant owners can ACCEPT/DECLINE/VIEW Orders.

**Driver**

The people who are in search of a job as a food delivery man. They must be aging over 18, and have already received a license number, and this number must not be expired. They play an intermediary role between customers and restaurants in the system. It includes the following attributes:

*Attributes:*

* **Full Name:** name of drivers who signs up to the system as an intermediate connection between customers and restaurants.
* **Driver\_ID:** This is the civil document ID the driver offers to the system
* **License number:** Registered license number offered to the system by the driver and only the driver and the admin can see this information. This number must has proof of verification by the authority.
* **Contact number:** Mobile phone numbers that are published on their profile so that customers and restaurants could be able to make contacts.

*Roles:*

* The driver can ACCEPT/DECLINE the delivery.
* The driver can view the details of orders.
* The driver can CHOOSE the payment type.

**Transaction**

This entity record all information from other entities which will be the billing details if the food order is confirmed. This is the core of the entire paying process. Unpaid, processing orders will be highlighted among approved ones and saved for further tracking log. It includes the following attributes:

*Attributes:*

* **Transaction\_ID:** The system generates a unique number which is the order ID mentioned on the order receipt to track and provide information about this transaction.
* **Driver\_ID:** The ID of the driver who receives the order.
* **Customer\_ID:** The ID of the customer who books the order.
* **Restaurant\_ID:** The ID of the restaurant which has the food booked.
* **Order\_ID:** The ID of the order of this transaction.
* **Payment\_ID:** The type of payment that this transaction will be done through.

**Payment:**

Customers, restaurants owners, and drivers get to decide which payment types they use for the food orders. All types of payment will be available on the system when billing but this depends on the functioning area whether it is suitable or not.

*Attributes:*

* **Payment\_ID:** The index of each payment type
* **Payment\_type:** available choices of paying like cash, credits, debits, cheques.

**Order:**

This entity holds the details of every order booked by the customer through the system. The customer chooses the food from the menu and adds it to the online cart then proceeds to the payment option if he wants to pay online or at the time of delivery.

*Attributes*

* **Order\_ID**: The system generates a unique number which is the order ID mentioned on the order receipt to track and provide information about this order.
* **Customer\_ID**: ID The order is given by the customer, so his/her id is attached with the order to safeguard the delivery details.
* **Status:** The customer can track the food he/she ordered through the system by the tracking ID which is stuck in this attribute.
* **Date**: Store the date when the customer order.
* **Driver\_ID:** The id of the driver that accepted the order and deliver it to the customer.
* **Amount:** The amount for each food that the customers ordered.

**Food:**

The variety of foods is enormous, however, they will be classified by placing ID rules according to meals of the day. The price of each dish is recorded and displayed on the system. This attribute can be sorted and filtered depending on the customers’ budgets. Images of dishes can be viewed and uploaded by customers, however, they must be verified by the admin due to appropriateness. It includes the following attributes:

*Attributes:*

* **Food\_ID:** The system generates a unique number which is the order ID mentioned on each food in each restaurant.
* **Name:** The name of every food in a restaurant is different and also the bill generated is by the name of the food.
* **Price:** This attribute holds the amount which the customer needs to pay for this food.
* **Image:** The image is provided by the restaurant and each image of food in the system is different.
* **Type\_ID:** Each type of food is given a specific ID in order to easily manage and classify among the vast variety of them.

**Menu Type:**

The details of every food are also present here. This entity is based on the cuisine of the area, for instance, there will be appetizers, mains, and deserts or meals of the day. The attributes of this entity are

*Attributes:*

* **Type\_ID**: The system generates a unique number which is the order ID mentioned on each menu in each restaurant.
* **Food Type**: It includes the dishes which are in each type of food like seafood, vegetable,...

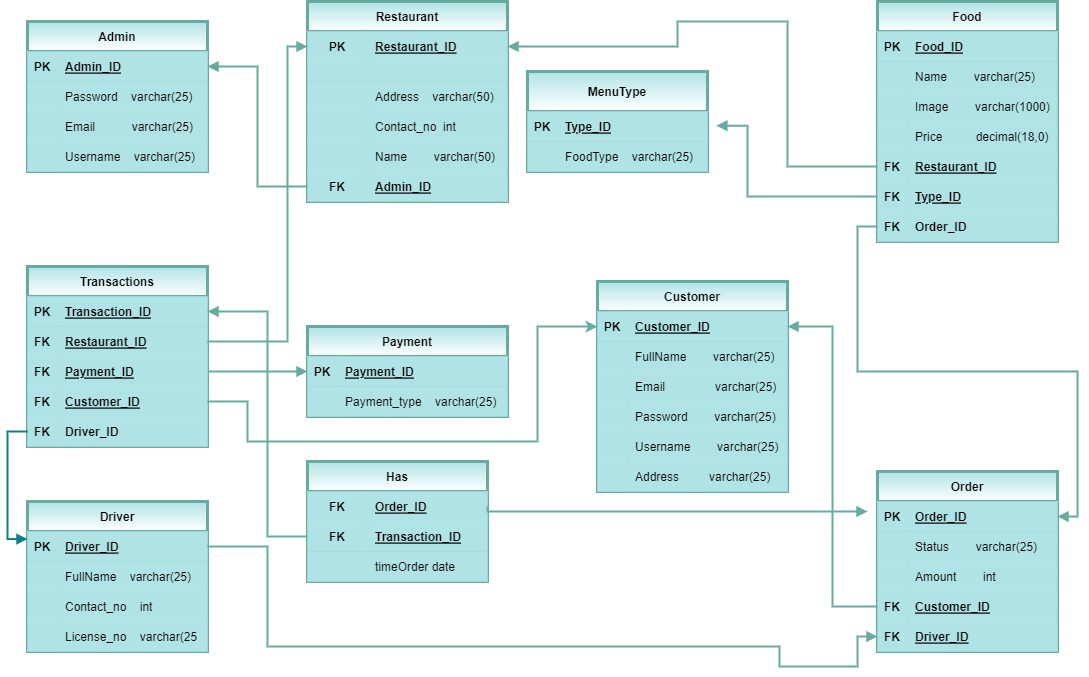
1. **Relational Model:**

*(Related knowledge: Design database ER, ERD to Relational)*

* + 1. ***Relational Model***

*(Related knowledge: ERD to Relational, Keys and FDs, Normalization)*

From 11 entities of the ERD, we convert equivalently into RM

****

*(III – i ). Relational Model*

* **Admin** (Admin\_ID, Password, Email, Username)
  + **F1=** {Admin\_ID 🡪 Username; Admin\_ID 🡪 Password; Admin\_ID 🡪 Email}
    - Admin\_ID is the key
    - Username, Password, Email can be derived fully from Admin\_ID.
    - The table is under 3NF since each attribute contains only atomic units and fully FD on Admin\_ID.
* **Restaurant** (Restaurant\_ID, Address, Contact\_no, Name, Admin\_ID)
  + **F2**= {Restaurant\_ID 🡪 Address; Restaurant\_ID 🡪 Contact\_no; Restaurant\_ID 🡪 Name}
    - From Restaurant\_ID (Primary Key), we can derive other keys except for Admin\_ID since (Restaurant\_ID, Admin\_ID) is super key and other keys are functionally dependent on them.
    - Thus, **Restaurants** is under 3NF.
* **MenuType**  (Type\_ID, FoodType)
  + **F3 =** {Type\_ID 🡪 FoodType)
    - FoodType can be derived from Type\_ID which is the Primary Key.
    - The entity contains only atomic values. (1NF)
    - Thus, it is under 3NF since Type\_ID is the super key in the above FD.
* **Food** (Food\_ID, Name, Image, Price, Restaurant\_ID, Type\_ID, OrderDetails\_ID)
  + **F4 =** {Food\_ID 🡪 Name; Food\_ID 🡪 Image; Food\_ID 🡪 Price}
    - Every row contains only single (atomic) values. (1NF)
    - Name, Price, Image can be derived from Food\_ID and functionally dependent on it. (2NF)
    - Thus, the entity is under 3NF since Food\_ID is the super key of the table.
* **Transactions** (Transaction\_ID, Restaurant\_ID, Payment\_ID, Customer\_ID, Driver\_ID)
  + **F5 =** {Transaction\_ID 🡪 Restaurant\_ID; Transaction\_ID 🡪 Payment\_ID; Transaction\_ID 🡪 Customer\_ID; Transaction\_ID 🡪 Driver\_ID}
    - Certainly, there is no duplicated rows but only atomic values (1NF)
    - Restaurant\_ID, Payment\_ID, Customer\_ID, Driver\_ID are fully functional dependent on Transaction\_ID. (2NF)
    - Thus, the entity is under 3NF since Transaction\_ID is the super key of the table.
* **Payment** (Payment\_ID, Payment\_type)
  + **F6 =** {Payment\_ID 🡪 Payment\_type}
    - Every row contains only single (atomic) values. (1NF)
    - Payment\_type is functionally dependent on Payment\_ID. (2NF)
    - Payment\_ID is the super key of the table. Hence it is under 3NF.
* **Customer (**Customer\_ID, FullName, Email, Password, Username)
  + **F7 =** {Customer\_ID 🡪 FullName; Customer\_ID 🡪 Email; Customer\_ID 🡪 Password; Customer\_ID 🡪 Username}
    - Every row contains only single (atomic) values. (1NF)
    - FullName, Email, Password, Username are functionally dependent on Customer\_ID. (2NF)
    - Customer\_ID is the super key of the table.
* **Driver** (Driver\_ID, FullName, Contact\_no, License\_no, Order\_ID)
  + **F9 =** {Driver\_ID 🡪 FullName; Driver\_ID 🡪 Contact\_no; Driver\_ID 🡪 License\_no}
    - Every row contains only single (atomic) values. (1NF)
    - FullName, Contact\_no, License\_no is functionally dependent on (Driver\_ID, Order\_ID) which is the candidate key. (2NF)
    - Driver\_ID is the super key/primary key of the table. (3NF)
* **Has** (Order\_ID, Transaction\_ID, Date, totalPrice)
  + **F10 =** {Order\_ID, Transaction\_ID 🡪 Date, Order\_ID, Transaction\_ID 🡪 totalPrice}
    - Every row contains only single (atomic) values. (1NF)
    - Date is fully functional dependent on (Order\_ID, Transaction\_ID) (2NF)
    - The table has no PK so it stays under 2NF.
* **Order** (Order\_ID, Status, TotalPrice, Customer\_ID)
  + **F10 =** {Order\_ID, Customer\_ID 🡪 Status, Order\_ID, Customer\_ID 🡪 TotalPrice}
    - Every row contains only single (atomic) values. (1NF)
    - Status, TotalPrice are fully functional dependent on (Order\_ID, Customer\_ID) which is the super key (2NF)
    - Order\_ID is the key of the table. (3NF)

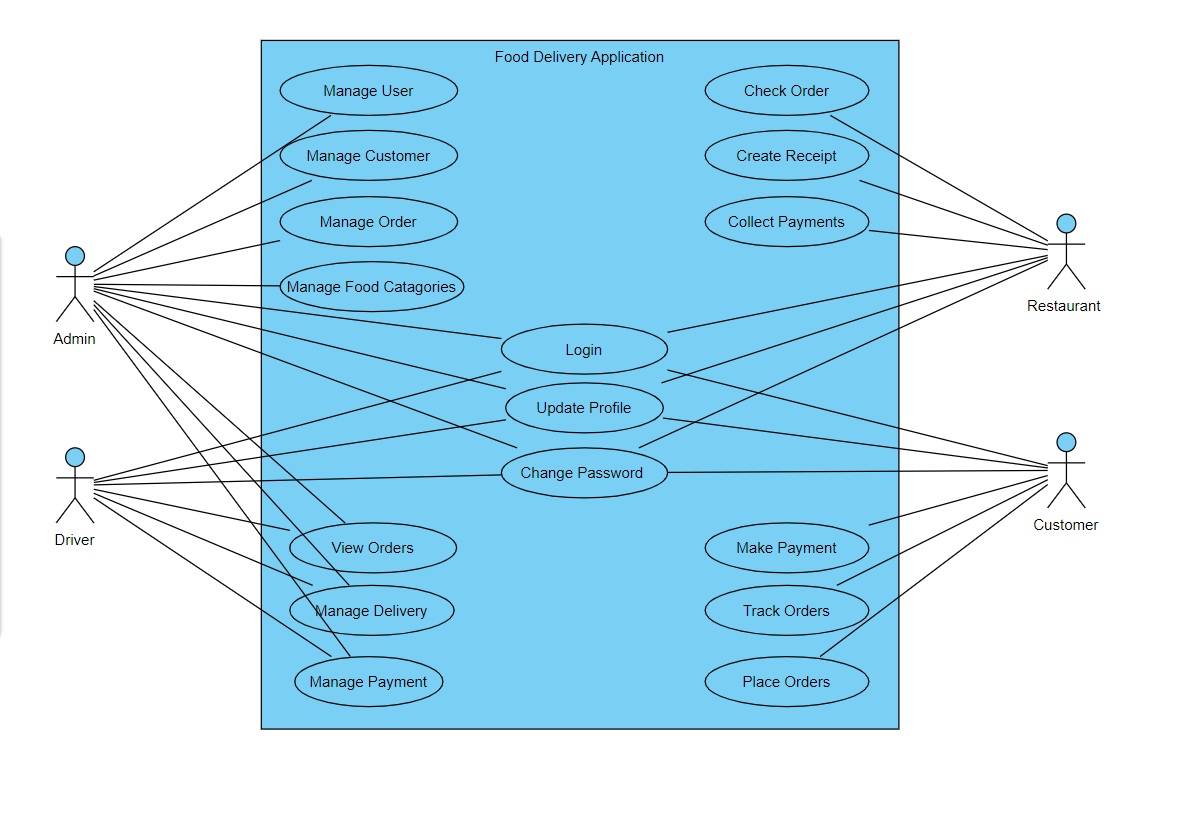
***ii. User Interface Diagram***

**Login**

In general, all users including admins, drivers, restaurant owners, and customers have the same interface of login by using their registered email and password. Their role will decide which interface afterward.

**Update profile & Change Password (Optional)**

Unless they have finished this step during registration, it is obligatory for them to update important information such as phone number, delivery address, and civil ID as proof that they are old enough to use the application and financial capability.



*(III – ii) Interactive diagram among users*

**Customer**

Customers will be shown promotions, discounts, and lists of restaurants that they see as their cup of tea.

**Admin**  
After the customers have placed their orders, the admin will verify their information by sending OTP via their email or phone number to make sure that these are conducted by actual beings and true actions.

**Restaurant**

The restaurant owners receive the verified orders from the admin via the notification center. and begin to make dishes. If there are any cancellations of customers, restaurant owners will directly make contact with them and cancel the order. Else, after having prepared the dishes and ready for delivery, available drivers will be there in advance for 15 minutes to take the order.

**Driver**

Simultaneously with the notification being given to the restaurant owners by the admin, drivers will see available orders depending on their radius of 5 kilometers.

1. **SQL Instructions:**

*(Related knowledge: SQL part 1 & 2)*

1. ***Create tables:***
2. **Customer:**

use FoodDeli;

CREATE TABLE Customer (

customerID int NOT NULL AUTO\_INCREMENT,

fullName varchar(25),

passwrd varchar(25),

email varchar(25),

username varchar(25),

address varchar(25),

PRIMARY KEY (customerID)

);

1. **Driver:**

CREATE TABLE Driver (

driverID int NOT NULL AUTO\_INCREMENT,

fullName varchar(25),

fontact\_no int,

license\_no varchar(25),

PRIMARY KEY (driverID)

);

ALTER TABLE Driver

MODIFY fontact\_no varchar(25);

1. **Orders:**

CREATE TABLE Orders (

orderID int NOT NULL AUTO\_INCREMENT,

ordStatus varchar(25),

totalPrice decimal(18,0),

amount int NOT NULL,

customerID int,

driverID int,

PRIMARY KEY (orderID),

FOREIGN KEY (customerID) REFERENCES Customer(customerID),

FOREIGN KEY(driverID) REFERENCES Driver(driverID)

);

ALTER TABLE Orders

DROP COLUMN totalPrice;

1. **Menu type:**

CREATE TABLE MenuType (

typeID int NOT NULL,

foodType varchar(25),

PRIMARY KEY (typeID)

);

1. **Administrator:**

CREATE TABLE adminis (

adminID int NOT NULL AUTO\_INCREMENT,

passwrd varchar(25),

email varchar(25),

username varchar(25),

PRIMARY KEY (adminID)

);

1. **Restaurant:**

CREATE TABLE Restaurant (

restaurantID int NOT NULL AUTO\_INCREMENT,

address varchar(50),

contact\_no int,

restName varchar(50),

adminID int,

PRIMARY KEY (restaurantID),

FOREIGN KEY (adminID) REFERENCES adminis(adminID)

);

ALTER TABLE Restaurant

MODIFY contact\_no varchar(50);

1. **Food:**

CREATE TABLE Food (

foodID int NOT NULL AUTO\_INCREMENT,

foodName varchar(25),

image varchar(1000),

price decimal(18,0),

orderID int,

typeID int,

restaurantID int,

PRIMARY KEY (foodID),

FOREIGN KEY (orderID) REFERENCES Orders(orderID),

FOREIGN KEY (typeID) REFERENCES MenuType(typeID),

FOREIGN KEY (restaurantID) REFERENCES Restaurant(restaurantID)

);

ALTER TABLE Food

MODIFY price varchar(10);

1. **Payment:**

CREATE TABLE Payment (

paymentID int NOT NULL,

payment\_type varchar(25),

PRIMARY KEY (paymentID)

);

1. **Transactions:**

CREATE TABLE Transactions (

transactionsID int NOT NULL AUTO\_INCREMENT,

restaurantID int,

paymentID int,

customerID int,

driverID int,

PRIMARY KEY (transactionsID),

FOREIGN KEY (restaurantID) REFERENCES Restaurant(restaurantID),

FOREIGN KEY (paymentID) REFERENCES Payment(paymentID),

FOREIGN KEY (customerID) REFERENCES Customer(customerID),

FOREIGN KEY (driverID) REFERENCES Driver(driverID)

);

1. **Has:**

CREATE TABLE Has (

timeOrder date,

orderID int,

transactionsID int ,

FOREIGN KEY (orderID) REFERENCES Orders(orderID),

FOREIGN KEY (transactionsID) REFERENCES Transactions(transactionsID)

);

1. ***Inserting data:***

use fooddeli;

INSERT INTO adminis VALUES

(1,'sdf41','rohan.lucy@yahoo.com','cschowalter'),

(2,'sdf34','therese71@yahoo.com','batz.adah'),

(3,'fg13r','treutel.andreane@hotmail.','waino.raynor'),

(4,'jdj38','mark.romaguera@yahoo.com','leopoldo.hammes'),

(5,'33sd3','maybelle78@gmail.com','edyth.stiedemann'),

(6,'43s3d','krodriguez@hotmail.com','fahey.hipolito');

Graphical user interface, text, application, email

Description automatically generated

*(IV-ii-1). Admin’s table*

INSERT INTO restaurant (`restaurantID`,`address`,`contact\_no`,`restName`,`adminID`)

VALUES

(1,"864-4077 Magna. Avenue","(522) 363-0751","Sophia",5),

(2,"Ap #625-1289 Elit. St.","(473) 711-4456","Addison",4),

(3,"969-2557 Aliquet Street","(488) 255-3153","Regina",6),

(4,"Ap #549-2262 Sed St.","(498) 414-0752","Patricia",4),

(5,"2641 Facilisi. Avenue","(573) 678-3270","Jennifer",2),

(6,"600-3148 Mi. Avenue","(883) 644-7573","Wylie",4),

(7,"6246 Ornare Avenue","(157) 364-7274","Wayne",2),

(8,"Ap #421-9303 Sagittis. Ave","(354) 285-1679","Sybil",5),

(9,"377-1635 At, Road","(727) 836-1878","Susan",5),

(10,"P.O. Box 454, 3652 Consectetuer Av.","(774) 335-8297","Drake",4);

Graphical user interface, text, application

Description automatically generated

*(IV-ii-2). Restaurant’s table*

INSERT INTO driver (`driverID`,`fullName`,`contact\_no`,`license\_no`)

VALUES

(1,"Laith Mercado","(600) 345-6438","699-766-4641"),

(2,"Jonas Vasquez","(197) 672-3784","239-331-5438"),

(3,"Austin Clarke","(878) 639-4424","954-196-4133"),

(4,"Mira Moss","(415) 697-7383","471-120-0295"),

(5,"Kaitlin Crawford","(217) 331-4628","878-358-7963"),

(6,"Natalie Mcclure","(618) 412-0438","520-251-5232"),

(7,"Hillary Henson","(441) 493-1688","436-141-9286"),

(8,"Tatiana Hall","(587) 917-7516","986-188-4570"),

(9,"Austin Bradford","(840) 753-6681","878-584-1083"),

(10,"Rhoda Bentley","(579) 686-7679","386-466-7205");

Graphical user interface, text, application, table, Excel

Description automatically generated

*(IV-ii-3). Driver’s table*

INSERT INTO customer (`customerID`,`fullName`,`email`,`passwrd`,`username`,`address`)

VALUES

(1,"Gloria Aguilar","amet@outlook.com","Mw18LK2","nfwm95","332-2350 Dui Rd."),

(2,"Guinevere Rasmussen","sit@google.com","Nk63XH8","gngr23","8382 Vitae Av."),

(3,"Barry Perez","dui.aliquam@google.net","Mv25JV2","iyty73","P.O. Box 826, 243 Vel St."),

(4,"Desiree Estes","aliquam.eros@outlook.edu","Lq91SI0","icyj28","Ap #649-5752 Ipsum Rd."),

(5,"Todd Powers","tempus.lorem@outlook.com","Xd22LI2","uhgd64","Ap #298-7216 Blandit Ave"),

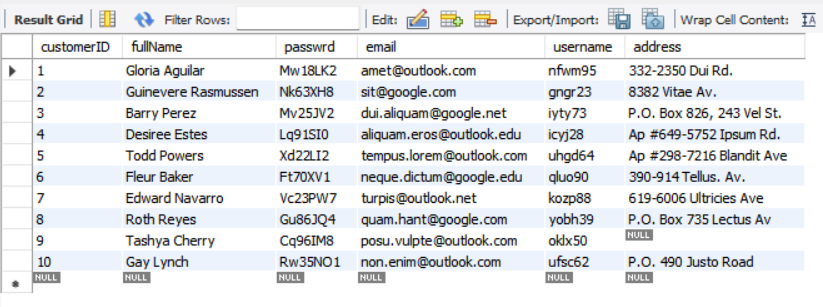
(6,"Fleur Baker","neque.dictum@google.edu","Ft70XV1","qluo90","390-914 Tellus. Av."),

(7,"Edward Navarro","turpis@outlook.net","Vc23PW7","kozp88","619-6006 Ultricies Ave"),

(8,"Roth Reyes","quam.hant@google.com","Gu86JQ4","yobh39","P.O. Box 735 Lectus Av"),

(9,"Tashya Cherry","posu.vulpte@outlook.com","Cq96IM8","oklx50","null”, Rd."),

(10,"Gay Lynch","non.enim@outlook.com","Rw35NO1","ufsc62","P.O. 490 Justo Road");



*(IV-ii-4). Customer’s table*

INSERT INTO orders (`orderID`,`ordStatus`,`amount`,`customerID`,`driverID`)

VALUES

(1,"Undone",5,2,10),

(2,"Done",1,4,3),

(3,"Undone",17,3,5),

Table

Description automatically generated (4,"Done",18,6,10),

(5,"Done",14,7,3),

(6,"Undone",4,9,2),

(7,"Done",20,3,1),

(8,"Done",3,5,4),

(9,"Undone",2,4,4),

(10,"Done",13,6,1);

*(IV-ii-5). Order’s table*

Graphical user interface, application

Description automatically generated

INSERT INTO MenuType (`typeID`,`foodType`)

VALUES

(1,"dinner"),

(2,"breakfast"),

(3,"brunch"),

(4,"fast-food"),

(5,"lunch");

*(IV-ii-6). Menu type’s table*

use fooddeli;

INSERT INTO food (`foodID`,`foodName`,`image`,`price`,`orderID`,`typeID`,`restaurantID`)

VALUES

(1,"fried-chicken","Kng82DYY63i43","$4.90",1,1,3),

(2,"spaghetti","Ere84ZCB14u41","$6.95",2,4,6),

(3,"rice","Sue13RQG18w77","$8.29",3,4,5),

(4,"noodles","Bcf34GBJ55d34","$5.68",4,4,6),

(5,"pasta","Okl16EMR40e96","$8.75",5,3,2),

(6,"springrolls","Vsd42JIA32i45","$7.89",6,2,2),

(7,"pizza","Qsi16TOF78d26","$6.49",7,4,5),

(8,"pho","Tut73NET65c63","$3.59",8,4,6),

(9,"sandwich","Dhb25EOI15q79","$4.82",9,3,3),

(10,"burger","Alk94ORL41w34","$1.99",10,2,2);

Graphical user interface, application

Description automatically generated

*(IV-ii-7). Food’s table*

Graphical user interface, application, table

Description automatically generatedINSERT INTO payment (`paymentID`,`payment\_type`)

VALUES

(1,"credits"),

(2,"crash"),

(3,"debits");

*(IV-ii-8). Paymentr’s table*

INSERT INTO transactions (`transactionsID`,`restaurantID`,`paymentID`,`customerID`,`driverID`)

VALUES

Table

Description automatically generated (1,7,3,7,1),

(2,3,1,8,2),

(3,8,2,5,3),

(4,5,1,4,4),

(5,10,1,3,5),

(6,9,3,9,6),

(7,4,1,6,7),

(8,1,2,10,8),

(9,2,2,1,9),

(10,6,1,2,10);

*(IV-ii-9). Transactions’ table*

INSERT INTO has (`timeOrder`,`orderID`,`transactionsID`)

Graphical user interface, table

Description automatically generatedVALUES

("2022-05-06",1,9),

("2022-05-20",2,7),

("2022-06-09",3,6),

("2022-05-02",4,10),

("2022-06-03",5,4),

("2022-05-24",6,3),

("2022-05-13",7,8),

("2022-06-02",8,1),

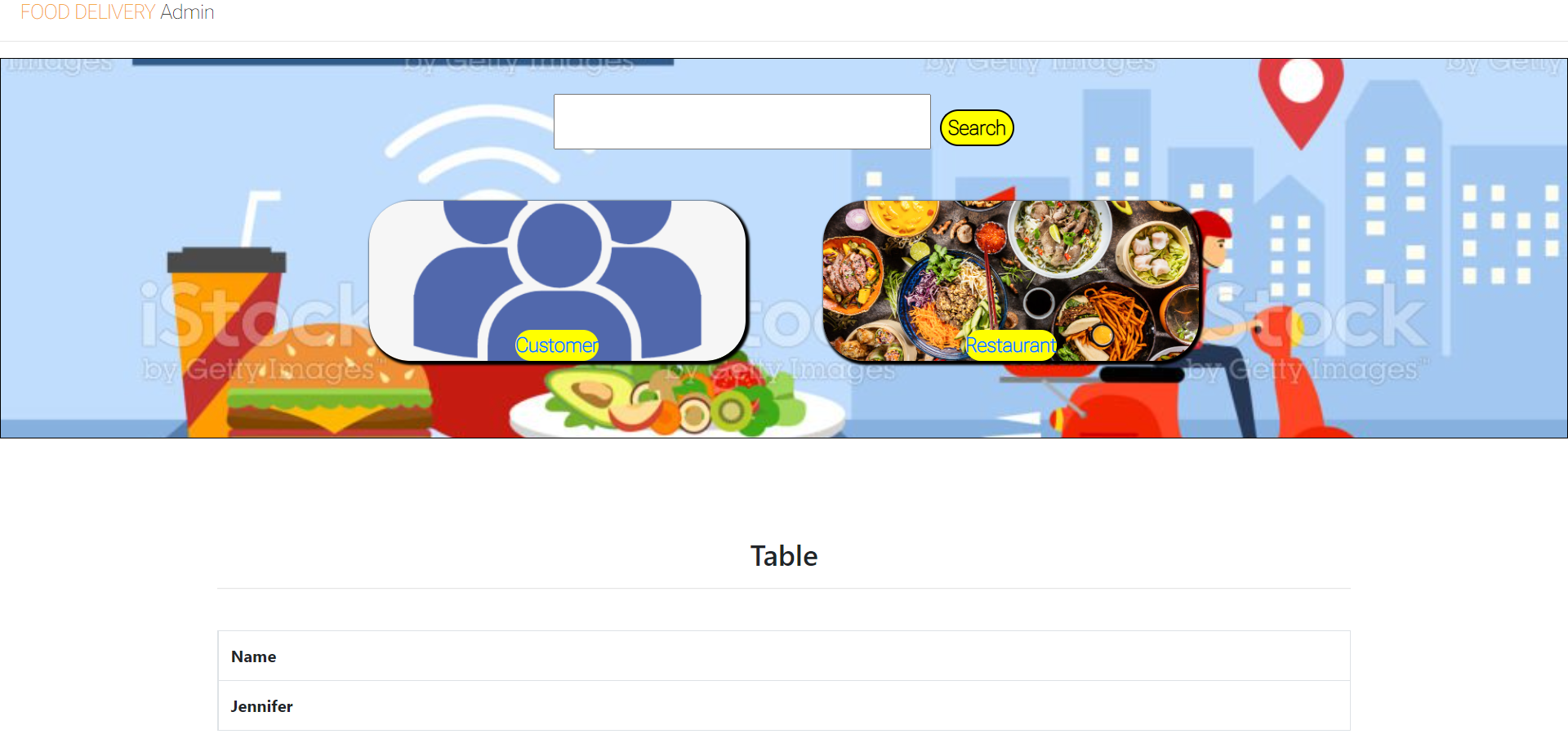
("2022-05-10",9,2),

("2022-05-08",10,5);

*(IV-ii-10). Has’s table*

1. **Querying examples**
2. Find all the restaurants that have a specific food type name “…”. For example: “Pizza”

**Query**: select r.restName from restaurant as r,food as f where r.restaurantID = f.restaurantID AND f.foodName LIKE '%Pizza%';



1. Find the food that are available in the area Z. For example: “Sed St.”

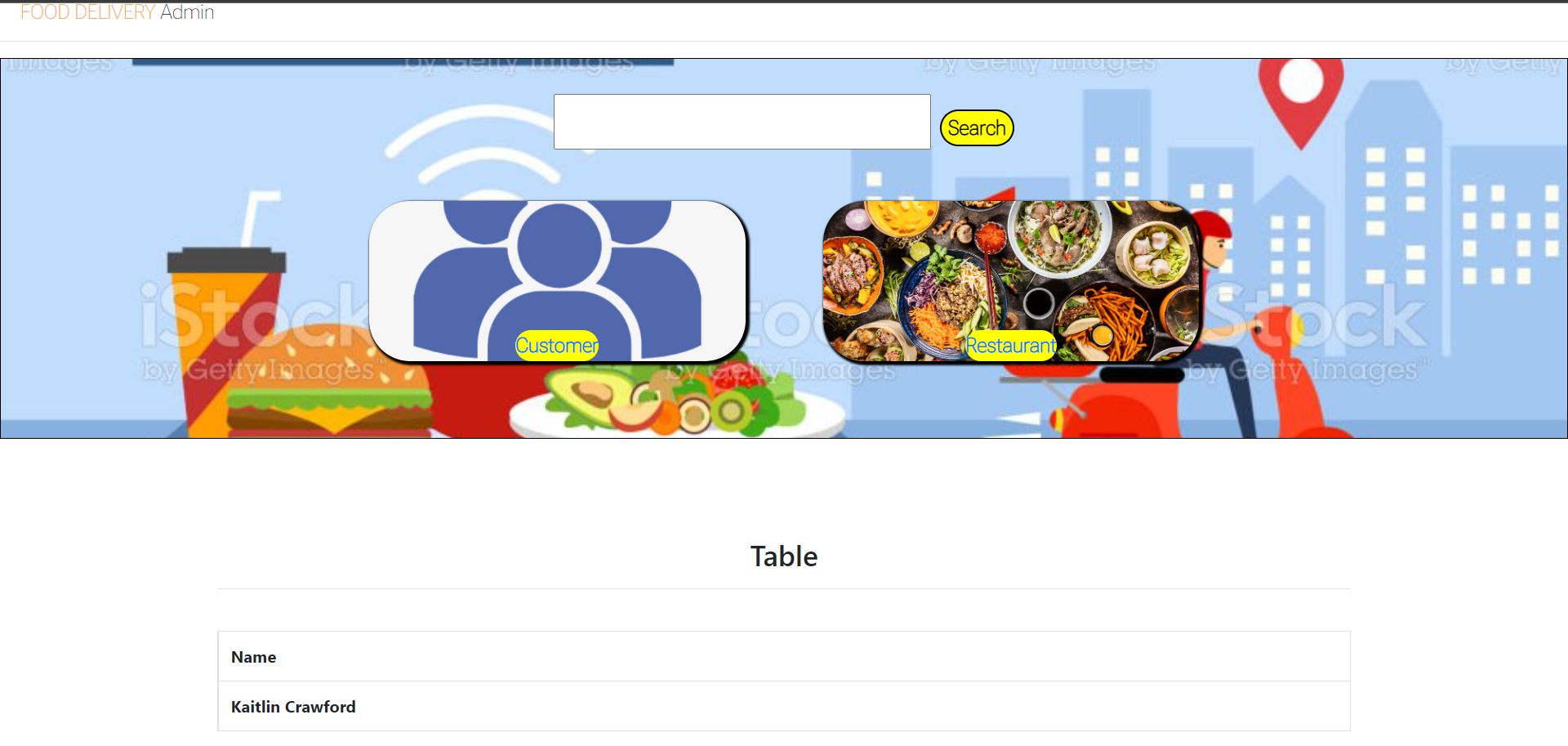
**Query:** select food.foodName from food where food.restaurantID = (select r.restaurantID from restaurant as r where r.address LIKE '%Mi. Avenue%');



1. Find the driver who has accepted the order of restaurant “…” For example Drake

**Query:** select driver.fullName from driver, transactions where transactions.restaurantID = (select restaurantID from restaurant where restName LIKE ‘%Drake%’) and

driver.driverID = transactions.driverID;



1. Find the name of the restaurant whose name starts with the letter “…” Example: “S”

**Query:** select restName from restaurant where restName LIKE 'S%'



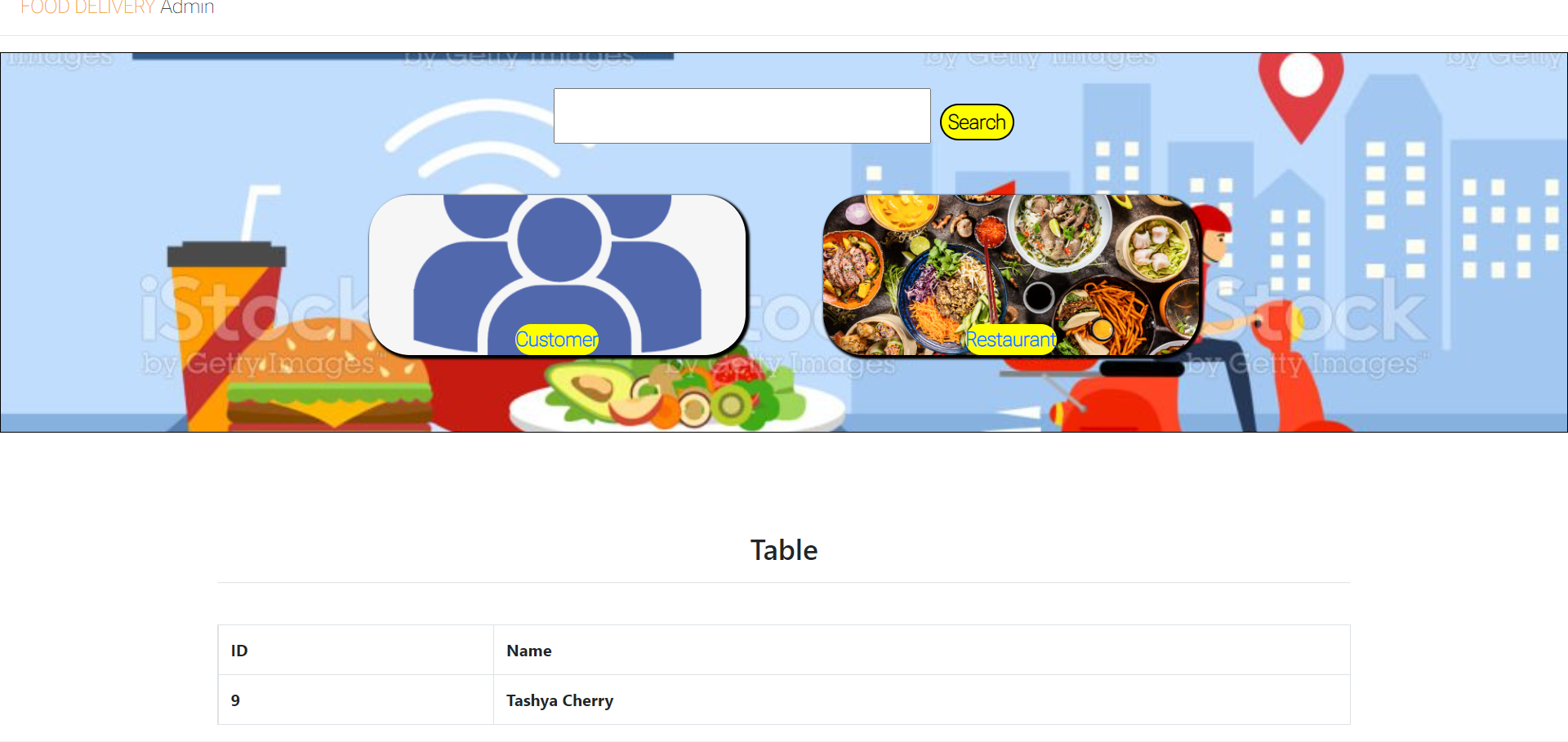
1. Show all the id orders that are paid via credits.

**Query** : select has.orderID from has, transactions where has.transactionsID = transactions.transactionsID and transactions.paymentID = '1'



1. Find the name of customer who have not register their address on the system.

**Query:**



References

* Báo cáo bài tập lớn môn cơ sở dữ liệu phân tán: hệ thống quản lý vật tư. [*https://tailieu.vn/doc/bao-cao-bai-tap-lon-mon-co-so-du-lieu-phan-tan-he-thong-quan-ly-vat-tu-2381878.html]*
* Báo cáo bài tập lớn môn Cơ sở dữ liệu - Học viện công nghệ bưu chính viễn thông. [*https://www.slideshare.net/HuyenPham31/bo-co-bi-tp-ln-mn-c-s-d-liu-hc-vin-cng-ngh-bu-chnh-vin-thng*]
* Working with related tables. [*https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP\_Help%2Frelated-tables-files.html%23*]
* Generate Data.

[*generatedata.com*]

* Servlet and JSPC.

[https://o7planning.org/10979/servlet-jsp.com]