

**UNIVERSITI TUNKU ABDUL RAHMAN**

**Faculty of Information and Communication Technology (FICT)**

**UCCD2303 DATABASE TECHNOLOGY /**

**UCCD2203 DATABASE SYSTEMS**

**Group Assignment Mark sheet**

January 2024 Trimester

|  |  |
| --- | --- |
| Group Number (e.g. G999): | G017 |
| Group leader to provide the OneDrive folder hyperlink (editor mode) which contains the zip file and a group presentation video in mp4 format: | https://drive.google.com/drive/folders/1RT8oXvj3hb1GnTqXJXznviMfmh3yt4p4?usp=drive\_link |
| Group leader name: | Ong Yi Sheng |
| Group leader email address: | ongyisheng0309@1utar.my |
| Submission date (dd-Mon-yy): | 16-Apr-24 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Name  (in ascending order)  Group leader with \* | Student ID  (e.g. 2299999) | Programme | Signature |
| 1 | Desmond Ho Jia Shen | 2105034 | CS |  |
| 2 | \*Ong Yi Sheng | 2103887 | CS |  |
| 3 | Seow Yi Xuan | 2105524 | CS |  |
| 4 | Tharini A/P Vijesh Kumar | 2206802 | CS |  |

Note:

* Maximum 4 members per group.
* The assignment grouping can be from different tutorial groups but must be from the same programme except with approval i.e. CS student work with CS student
* Programme e.g. IA/IB/DE/CS/CN/CT
* All members should attach their individual signature confirming that the report is not plagiarized
* For assignment answer submission include the Mark sheet, Table of Contents. Members’ contributions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Assignment Mark Sheet** | |  |  |
| **PART 1: (Group Assessment – 50 marls)** | | Allocated marks | **Given Marks** |
| **1.** | **Scope of Work (5 marks)**  Analyse requirements study (briefly explain the requirements/ office / business rules in the system).  PLEASE INCLUDE ANY ASSUMPTIONS THAT YOU MAKE. | 5 |  |
| **2.** | **ER model** (**10 marks**)  You are required to design an ER diagram for the case study given, identify entities, identify relationships, identify associate attribute and determine keys.  Check your ERD with the transaction requirements stated in the case. | 10 |  |
| **3.** | **Redesign and EER** (**10 marks**)  Redesign your ER diagram with the new requirements and extending the ERD to EER model, if any. | 10 |  |
| **4.** | **Data Dictionary** (**10 marks**)  Based on EER diagram that you created in part 3, create a data dictionary for the solution. (Make sure the data types (Oracle) selected are appropriate) | 10 |  |
| **5.** | **Tables and records** (**5 marks**)  Create all relations in ERD and insert the necessary records (Minimum 5 record for each table) | 5 |  |
| **6.** | **Script** (**10 marks**)  You are required to submit the SQL schema script with proper codes. Should include Integrity and referential integrity constraints.  **Softcopy:** *Include the scripts in the submission* | 10 |  |

**PART 2: (Individual Assessment – 50 marks)**

(Fill in all your group members name and ID according to the same sequence on the cover page)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Member |  | 1 | 2 | 3 | 4 |
| **Student Name: (to fill)** |  | **Desmond Ho Jia Shen** | **Ong Yi Sheng** | **Tharini A/P Vijesh Kumar** | **Seow Yi Xuan** |
| **Student ID; (to fill)** |  | **2105034** | **2103887** | **2206802** | **2105524** |
| Two Queries (10 marks) | 10 |  |  |  |  |
| Two Stored Procedure (10 marks) | 10 |  |  |  |  |
| Two Functions (10 marks) | 10 |  |  |  |  |
| Presentation (20 marks) | 20 |  |  |  |  |
| Total Individual Assessment (50 marks) |  |  |  |  |  |
| Group Assessment (50 marls) |  |  |  |  |  |
| **Total marks (100 marks)** |  |  |  |  |  |

|  |
| --- |
| **Presentation**   1. Group members are required to present a working database system, queries, stored procedures, and functions. 2. Every group has to present their assignment work and explain each group member contribution towards the completion of the assignment. 3. Group member must present his/her work as individual for 3-5 minutes each person and compiled by the group leader as a single presentation video. 4. Each member record individually then group leader compiles as a single group video. 5. Group leader to compile one group presentation video to be uploaded into OneDrive and provide the video hyperlink. 6. For Part 2 answers**:**  * Indicate the operations performed and include an explanations on how the user can use the corresponding SQL commands. * No duplication or similar answers should be listed among the group members. |

|  |  |  |
| --- | --- | --- |
| No. | Name  (in ascending order)  Group leader with \* | A short description of members’ contributions |
| 1. | Desmond Ho Jia Shen | Data Dictionary, SQL Script |
| 2. | \*Ong Yi Sheng | ERD, SQL Script |
| 3. | Seow Yi Xuan | Scope of Work, Table of Record, SQL Script |
| 4. | Tharini A/P Vijesh Kumar | ERD, EER |

**Table of Contents**

[**Presentation** 3](#_Toc164192808)

[(i). Group members are required to present a working database system, queries, stored procedures, and functions. 3](#_Toc164192809)

[(ii). Every group has to present their assignment work and explain each group member contribution towards the completion of the assignment. 3](#_Toc164192810)

[(iii). Group member must present his/her work as individual for 3-5 minutes each person and compiled by the group leader as a single presentation video. 3](#_Toc164192811)

[(iv). Each member record individually then group leader compiles as a single group video. 3](#_Toc164192812)

[(v). Group leader to compile one group presentation video to be uploaded into OneDrive and provide the video hyperlink. 3](#_Toc164192813)

[(vi). For Part 2 answers**:** 3](#_Toc164192814)

[ No duplication or similar answers should be listed among the group members. 3](#_Toc164192815)

[1.0 Scope of Work 7](#_Toc164192816)

[1.1 Business Rules 7](#_Toc164192817)

[1.2 Assumptions 7](#_Toc164192818)

[2.0 Entity Relationship Model 8](#_Toc164192819)

[3.0 Enhanced Entity Relationship Model 9](#_Toc164192820)

[4.0 Data Dictionary 10](#_Toc164192821)

[4.1 Customer 10](#_Toc164192822)

[4.2 Inventory 10](#_Toc164192823)

[4.3 Food 11](#_Toc164192824)

[4.4 Meal 12](#_Toc164192825)

[4.5 Beverage 12](#_Toc164192826)

[4.6 Dessert 12](#_Toc164192827)

[4.7 Cart\_Food 13](#_Toc164192828)

[4.8 Cart 14](#_Toc164192829)

[4.9 Invoice 15](#_Toc164192830)

[4.10 Order\_ 16](#_Toc164192831)

[4.11 Table 17](#_Toc164192832)

[4.12 Reservation 18](#_Toc164192833)

[4.13 Staff 19](#_Toc164192834)

[4.14 Chef 20](#_Toc164192835)

[4.15 Waiter 20](#_Toc164192836)

[4.16 Admin 20](#_Toc164192837)

[5.0 Tables and Records 21](#_Toc164192838)

[5.1 Customer 21](#_Toc164192839)

[5.2 TABLE 21](#_Toc164192840)

[5.3 RESERVATION 22](#_Toc164192841)

[5.4 STAFF 22](#_Toc164192842)

[5.5 CHEF 22](#_Toc164192843)

[5.6 ADMIN 23](#_Toc164192844)

[5.7 WAITER 23](#_Toc164192845)

[5.8 INVENTORY 23](#_Toc164192846)

[5.9 FOOD 24](#_Toc164192847)

[5.10 MEAL 24](#_Toc164192848)

[5.11 BEVERAGE 24](#_Toc164192849)

[5.12 DESSERT 25](#_Toc164192850)

[5.13 ORDER 25](#_Toc164192851)

[5.14 CART 25](#_Toc164192852)

[5.15 CART\_FOOD 26](#_Toc164192853)

[5.16 INVOICE 26](#_Toc164192854)

[6.0 SQL Script Content 27](#_Toc164192855)

[7.0 Individual Assignment (Part 2) 38](#_Toc164192856)

[7.1 Ong Yi Sheng (\*) 38](#_Toc164192857)

[7.1.1 Queries 38](#_Toc164192858)

[7.1.2 PL/SQL stored procedure 38](#_Toc164192859)

[7.1.3 PS/SQL functions 41](#_Toc164192860)

[7.2 Desmond Ho Jia Shen 43](#_Toc164192861)

[7.2.1 Queries 43](#_Toc164192862)

[7.2.2 PL/SQL stored procedure 43](#_Toc164192863)

[7.2.3 PS/SQL functions 45](#_Toc164192864)

[7.3 Seow Yi Xuan 46](#_Toc164192865)

[7.3.1 Queries 46](#_Toc164192866)

[7.3.2 PL/SQL stored procedure 47](#_Toc164192867)

[7.3.3 PS/SQL functions 49](#_Toc164192868)

[7.4 Tharini Vijesh Kumar 51](#_Toc164192869)

[7.4.1 Queries 51](#_Toc164192870)

[7.4.2 PL/SQL stored procedure 51](#_Toc164192871)

[7.4.3 PS/SQL functions 53](#_Toc164192872)

# **Scope of Work**

## **1.1 Business Rules**

Each customer can have zero or many foods.

Each customer can have one or many invoices.

Each food can be consumed by only one customer.

Each food can be part of zero or many carts of food.

Each food can have one or many available in inventory.

Each food may be a meal, beverage, or dessert.

Each cart food can have only one food and each cart food can have only one cart.

Each cart can have zero or many food and each cart can have zero or one invoice.

Each invoice can be made to only one cart.

Each invoice can be made to only one customer.

Each invoice can be made for only one order.

Each order can only have one invoice.

Each order is only for one table.

Each order can be handled by only one staff.

Each table can have only one order.

Each table can have only one reservation.

Each table can have only one customer.

Each reservation can have one or many tables.

Each reservation can have only one customer.

Each staff can handle zero or many orders.

Each staff may be a chef, an admin, or a waiter.

## **1.2 Assumptions**

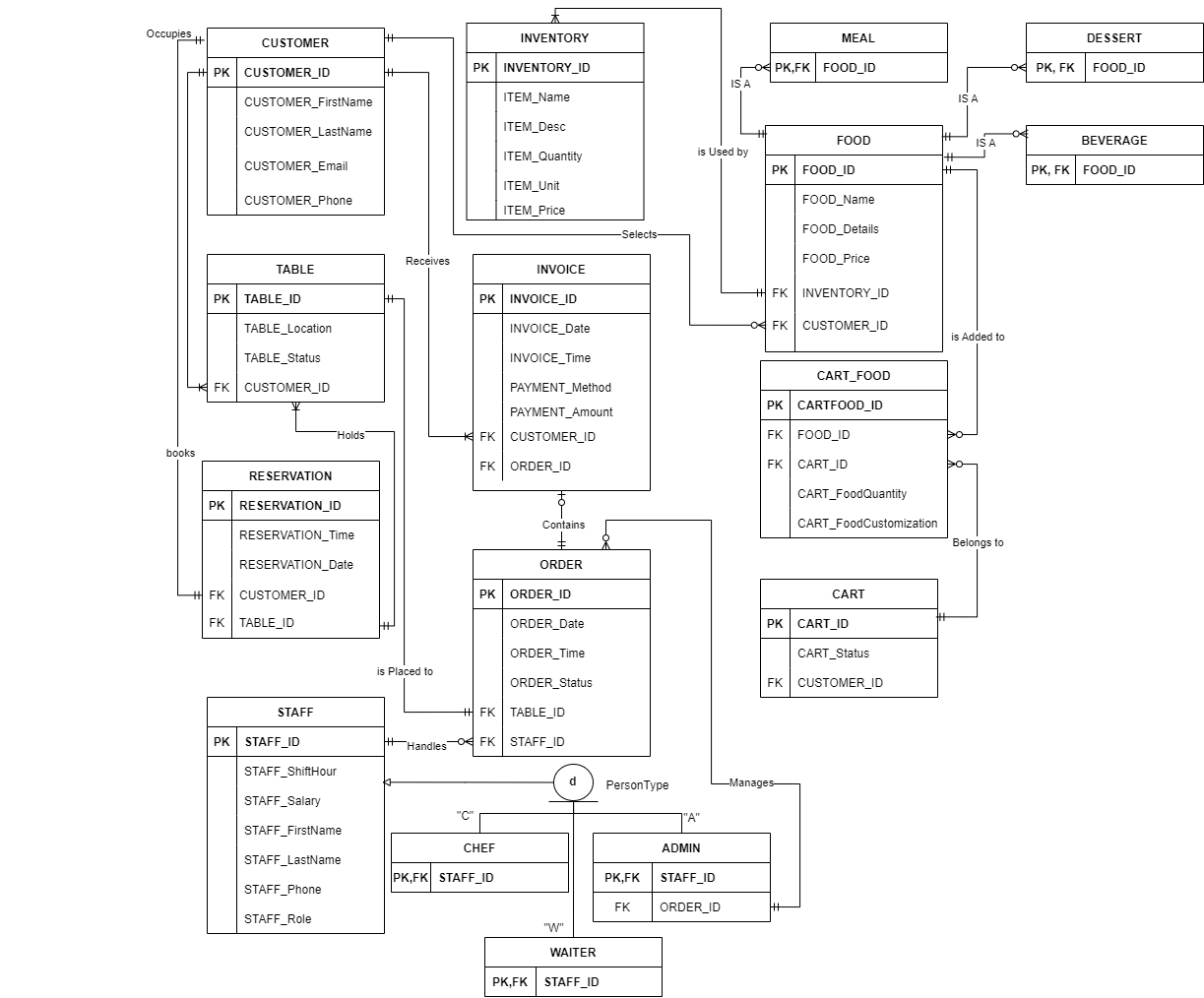
* All food items are assumed to be available.
* The prices of the food items are assumed to remain constant.
* Customers can customize their ordered food, but they are limited to modifying only the ingredients specified in the food details and cannot add any additional ingredients.
* Order can be cancelled if the ingredients needed are not sufficient.
* Item quantity in Inventory cannot be less than 20.
* Customer must receive their invoice after making their payment.

# **2.0 Entity Relationship Model**

A diagram of a food delivery service

Description automatically generated with medium confidence

## **3.0 Enhanced Entity Relationship Model**



# **4.0 Data Dictionary**

## **4.1 Customer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each customer can have zero or many foods, each customer can have one or many invoice | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| CUSTOMER\_ID | VARCHAR2 | PK | 5 | Unique ID for each customer | C002 |
| CUSTOMER\_FirstName | VARCHAR2 | - | 30 | Customer’s first name | Tan |
| CUSTOMER\_LastName | VARCHAR2 | - | 30 | Customer’s last name | Lee Hwa |
| CUSTOMER\_Email | VARCHAR2 | - | 30 | Customer’s email | tanleehwa@gmail.com |
| CUSTOMER\_Phone | NUMBER | - | 12 | Customer’s contact number | 0111110298 |

## **4.2 Inventory**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each inventory can have only one food | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| INVENTORY\_ID | VARCHAR2 | PK | 5 | Unique ID for each inventory | IT002 |
| ITEM\_Name | VARCHAR2 | - | 30 | Item name in inventory | Oil |
| ITEM\_Desc | VARCHAR2 | - | 30 | Item descriptions | Olive oil |
| ITEM\_Quantity | NUMBER | - | 4 | Item quantity in stock | 250 |
| ITEM\_Unit | NUMBER | - | 10 | Item unit number | 1.5Kg |
| ITEM\_Price | NUMBER | - | 10 | Item price | 25.00 |

## **4.3 Food**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each food can be consumed by only one customer, each food can be part of zero or many cart food, each food can have one or many available in inventory, each food may be a meal, each food may be a beverage, each food may be a dessert | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| FOOD\_ID | VARCHAR2 | PK | 5 | Unique ID for each food | F003 |
| FOOD\_Name | VARCHAR2 | - | 40 | Food’s name | Mapo Tofu with White Rice |
| FOOD\_Details | VARCHAR2 | - | 100 | Food’s ingredients and way of cooking | tofu cooked with rich spicy, and savory sauce |
| FOOD\_Price | NUMBER | - | 10 | Food’s pricing | 8.00 |
| INVENTORY\_ID | VARCHAR2 | NOT NULL CONSTRAINT  FK references  inventory(inventory\_id) | 5 | Foreign key from Inventory table | IT002 |
| CUSTOMER\_ID | VARCHAR2 | NOT NULL CONSTRAINT  FK references  customer(customer\_id) | 5 | Foreign key from Customer table to identify which customer order a food | C001 |

## **4.4 Meal**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each Meal must be a food | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| FOOD\_ID | VARCHAR2 | PK, FK | 5 | Primary & Foreign key from Food Table | F001 |

## **4.5 Beverage**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each Beverage must be a food | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| FOOD\_ID | VARCHAR2 | PK, FK | 5 | Primary & Foreign key from Food Table | F001 |

## **4.6 Dessert**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each Dessert must be a food | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| FOOD\_ID | VARCHAR2 | PK, FK | 5 | Primary & Foreign key from Food Table | F001 |

## **4.7 Cart\_Food**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each cart food can have only one food, each cart food can have only one cart | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| CARTFOOD\_ID | NUMBER | PK | 2 | Unique ID for each cart food | 1 |
| FOOD\_ID | VARCHAR2 | Not Null Constraint  FK references  Food(food\_id) | 5 | Foreign key from food table | F001 |
| CART\_ID | VARCHAR2 | Not Null Constraint  FK references  Cart(cart\_id) | 7 | Foreign key from cart table | Cart01 |
| CART\_FoodQuantity | NUMBER | - | 2 | Quantity of food in a cart | 2 |
| CART\_Food\_Customization | VARCHAR2 | - | 100 | Customer can customize their food | No sesame (Teriyaki Chicken Poke Bowl) |

## **4.8 Cart**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each cart can have zero or many cart food, each cart can have zero or one invoice | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| CART\_ID | VARCHAR2 | PK | 7 | Unique ID for each Cart | Cart02 |
| CART\_Status | VARCHAR2 | Check Constraint (cart\_status in (‘Abandoned', 'Pending’, ‘Processing’)) | 20 | Status of a cart | Pending |
| CUSTOMER\_ID | VARCHAR2 | NOT NULL CONSTRAINT  FK references  customer(customer\_id) | 5 | Foreign key from Customer table | C002 |

## **4.9 Invoice**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each invoice can be made to only one cart, each invoice can be made to only one customer, each invoice can be made to only one order | | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| INVOICE\_ID | VARCHAR2 | PK | 5 | Unique ID for each Invoice | IV002 |
| INVOICE\_Date | DATE | Default Constraint  (invoice\_date in (‘TO\_DATE()’)) | - | Date of Invoice | 3/10/2024 |
| INVOICE\_Time | VARCHAR2 | - | 8 | Time of Invoice | 16:45 |
| PAYMENT\_Method | VARCHAR2 | Check Constraint (payment\_method in ('Cash', 'QR Payment', 'Credit/Debit Card’)) | 20 | Payment method for the food ordered | Cash |
| PAYMENT\_Amount | NUMBER | - | 20 | Payment total amount | 36.80 |
| CUSTOMER\_ID | VARCHAR2 | Not Null Constraint  FK references  Customer(customer\_id) | 5 | Foreign key from Customer table | C004 |
| ORDER\_ID | VARCHAR2 | Not Null Constraint  FK references  Order(order\_id) | 7 | Foreign key from Order table | Ord003 |

## **4.10 Order\_**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each order can only have one invoice, each order is only for one table, each order can be handled by only one staff | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| ORDER\_ID | VARCHAR2 | PK | 7 | Unique ID for each Order | Ord002 |
| ORDER\_Date | DATE | Default Constraint  (order\_date in (‘TO\_DATE()’)) | - | Date of order | 15/11/2024 |
| ORDER\_Time | VARCHAR2 | - | 8 | Time of order | 08:32 |
| ORDER\_Status | VARCHAR2 | Check Constraint (order\_status in ('Completed', 'Processing', Cancelled’)) | 20 | Status of order | Processing |
| TABLE\_ID | VARCHAR2 | Not Null Constraint  FK references  Table(table\_id) | 4 | Foreign key from Table table | T001 |
| STAFF\_ID | VARCHAR2 | Not Null Constraint  FK references  Staff(staff\_id) | 4 | Foreign key from Staff table | S001 |

## **4.11 Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each table can have only one order, each table can have only one reservation, each table can have only one customer | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| Table\_ID | VARCHAR2 | PK | 5 | Unique ID for each table | T004 |
| Table\_Location | VARCHAR2 | Check Constraint  (table\_location in (‘Indoor’, ‘Outdoor’)) | 10 | Table location for customer | Indoor |
| Table\_Status | VARCHAR2 | Check Constraint  (table\_status in (‘Available’, ‘Occupied’, ‘Reserved’)) | 15 | Table availability | Occupied |
| CUSTOMER\_ID | VARCHAR2 | Not Null Constraint  FK references  Customer(customer\_id) | 5 | Foreign key from Customer table | C001 |

## **4.12 Reservation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each reservation can have one or many table, each reservation can have only one customer | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| RESERVATION\_ID | VARCHAR2 | PK | 5 | Unique ID for each reservation | R001 |
| RESERVATION\_Time | VARCHAR2 | - | 8 | A composite primary key for prescription. A foreign key from appointment table | 10:04 |
| RESERVATION\_Date | DATE | Default Constraint  (reservation\_date in (‘TO\_DATE()’)) | - | Date of reservation | 26/04/2024 |
| CUSTOMER\_ID | VARCHAR2 | Not Null Constraint  FK references  Customer(customer\_id) | 5 | Foreign key from Customer table | C004 |
| Table\_ID | VARCHAR2 | Not Null Constraint  FK references  Table(table\_id) | 5 | Foreign key from Table table | T005 |

## **4.13 Staff**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each staff can handle zero or many order, each staff may be a chef or not a chef, each staff may be an admin or not an admin, each staff may be a waiter or not a waiter | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| STAFF\_ID | VARCHAR2 | PK | 5 | Unique ID for each staff | S001 |
| STAFF\_ShiftHour | VARCHAR2 | Check Constraint  (staff\_shifthour in (‘Morning’, ‘Evening’, ‘Off’)) | 15 | Staff working shift status | Evening |
| STAFF\_Salary | NUMBER | - | 20 | Staff salary | 2500.00 |
| STAFF\_FirstName | VARCHAR2 | - | 30 | Staff first name | Ong |
| STAFF\_LastName | VARCHAR2 | - | 30 | Staff last name | Yi Sheng |
| STAFF\_Phone | NUMBER | - | 12 | Staff contact number | 0126435156 |
| STAFF\_Role | VARCHAR2 | Check Constraint  (staff\_role in (‘Chef’, ‘Waiter’, ‘Admin’)) | 10 | Staff role | Waiter |

## **4.14 Chef**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each chef must be a staff | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| STAFF\_ID | VARCHAR2 | PK | 6 | Unique ID for each chef | S001 |

## **4.15 Waiter**

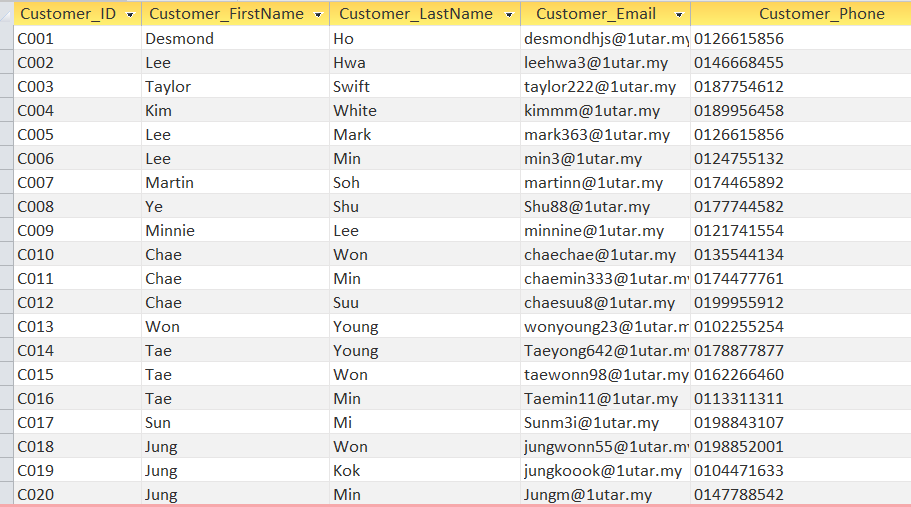
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each waiter must be a staff | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| STAFF\_ID | VARCHAR2 | PK | 6 | Unique ID for each waiter | S002 |

## **4.16 Admin**

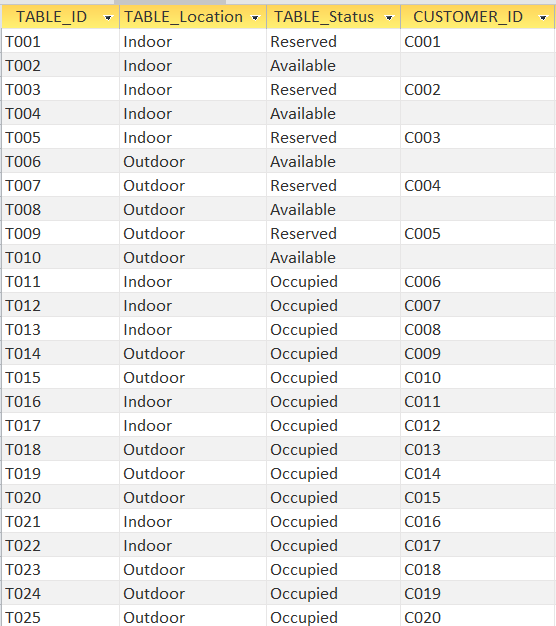
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Occurrence:** Each admin must be a staff | | | | | |
| **Entity Name** | **Data Type** | **Constraint Type** | **Field Size** | **Description** | **Example** |
| STAFF\_ID | VARCHAR2 | PK | 6 | Unique ID for each admin | S003 |

# **5.0 Tables and Records**

## **5.1 Customer**



## **5.2 TABLE**

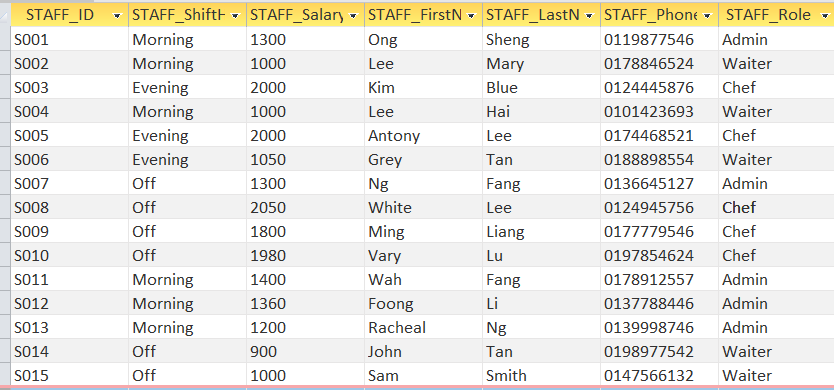


## **5.3 RESERVATION**

A screenshot of a table

Description automatically generated

## **5.4 STAFF**

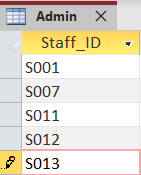


## **5.5 CHEF**

A screenshot of a computer

Description automatically generated

## **5.6 ADMIN**

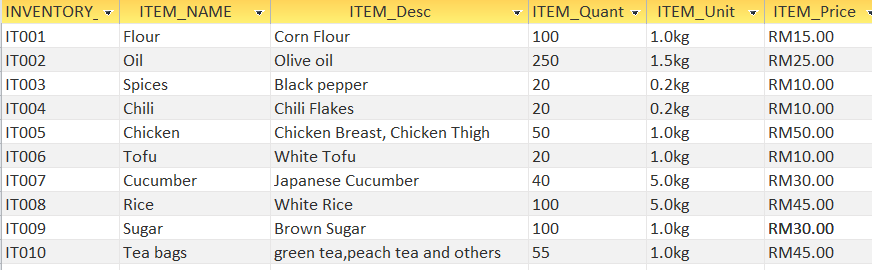


## **5.7 WAITER**

A screenshot of a computer

Description automatically generated

## **5.8 INVENTORY**



## **5.9 FOOD**



## **5.10 MEAL**

A screenshot of a computer

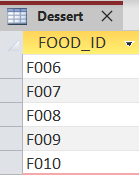
Description automatically generated

## **5.11 BEVERAGE**

A screenshot of a computer

Description automatically generated

## **5.12 DESSERT**

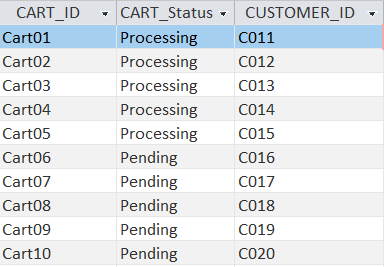


## **5.13 ORDER**

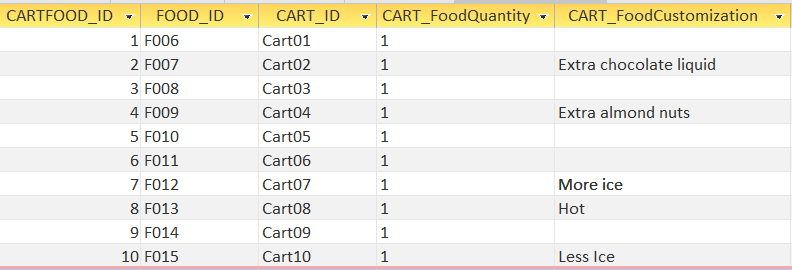
A screenshot of a computer

Description automatically generated

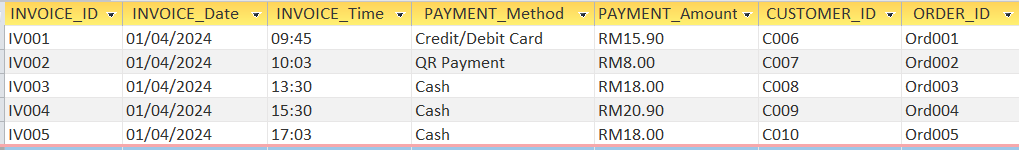
## **5.14 CART**



## **5.15 CART\_FOOD**



## **5.16 INVOICE**



# **6.0 SQL Script Content**

-- Dropping tables with CASCADE CONSTRAINTS to remove all related constraints

DROP TABLE Invoice CASCADE CONSTRAINTS;

DROP TABLE Cart\_food CASCADE CONSTRAINTS;

DROP TABLE Cart CASCADE CONSTRAINTS;

DROP TABLE Food CASCADE CONSTRAINTS;

DROP TABLE Inventory CASCADE CONSTRAINTS;

DROP TABLE Order\_ CASCADE CONSTRAINTS;

DROP TABLE Staff CASCADE CONSTRAINTS;

DROP TABLE Reservation CASCADE CONSTRAINTS;

DROP TABLE "Table" CASCADE CONSTRAINTS;

DROP TABLE Customer CASCADE CONSTRAINTS;

DROP TABLE Chef CASCADE CONSTRAINTS;

DROP TABLE Admin CASCADE CONSTRAINTS;

DROP TABLE Waiter CASCADE CONSTRAINTS;

DROP TABLE Meal CASCADE CONSTRAINTS;

DROP TABLE Dessert CASCADE CONSTRAINTS;

DROP TABLE Beverage CASCADE CONSTRAINTS;

-- Now, creating tables with the corrected definitions

-- Customer Table

CREATE TABLE Customer (

CUSTOMER\_ID VARCHAR2(5) PRIMARY KEY,

CUSTOMER\_FirstName VARCHAR2(30),

CUSTOMER\_LastName VARCHAR2(30),

CUSTOMER\_Email VARCHAR2(30),

CUSTOMER\_Phone VARCHAR2(12)

);

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C001','Desmond','Ho','desmondhjs@1utar.my','0126615856');

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C002','Lee','Hwa','leehwa3@1utar.my','0146668455');

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C003','Taylor','Swift','taylor222@1utar.my','0187754612');

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C004','Kim','White','kimmm@1utar.my','0189956458');

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C005','Lee','Mark','mark363@1utar.my','0126615856');

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES

('C006','Lee','Min','min3@1utar.my','0124755132');

INSERT INTO Customer VALUES ('C007','Martin','Soh','martinn@1utar.my','0174465892');

INSERT INTO Customer VALUES ('C008','Ye','Shu','Shu88@1utar.my','0177744582');

INSERT INTO Customer VALUES ('C009','Minnie','Lee','minnine@1utar.my','0121741554');

INSERT INTO Customer VALUES ('C010','Chae','Won','chaechae@1utar.my','0135544134');

INSERT INTO Customer VALUES ('C011','Chae','Min','chaemin333@1utar.my','0174477761');

INSERT INTO Customer VALUES ('C012','Chae','Suu','chaesuu8@1utar.my','0199955912');

INSERT INTO Customer VALUES ('C013','Won','Young','wonyoung23@1utar.my','0102255254');

INSERT INTO Customer VALUES ('C014','Tae','Young','Taeyong642@1utar.my','0178877877');

INSERT INTO Customer VALUES ('C015','Tae','Won','taewonn98@1utar.my','0162266460');

INSERT INTO Customer VALUES ('C016','Tae','Min','Taemin11@1utar.my','0113311311');

INSERT INTO Customer VALUES ('C017','Sun','Mi','Sunm3i@1utar.my','0198843107');

INSERT INTO Customer VALUES ('C018','Jung','Won','jungwonn55@1utar.my','0198852001');

INSERT INTO Customer VALUES ('C019','Jung','Kok','jungkoook@1utar.my','0104471633');

INSERT INTO Customer VALUES ('C020','Jung','Min','Jungm@1utar.my','0147788542');

-- Table Table

CREATE TABLE "Table" (

Table\_ID VARCHAR2(5) PRIMARY KEY,

Table\_Location VARCHAR2(10),

Table\_Status VARCHAR2(15),

CUSTOMER\_ID VARCHAR2(5),

CONSTRAINT fk\_customer\_table FOREIGN KEY (CUSTOMER\_ID) REFERENCES Customer(CUSTOMER\_ID)

);

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T001','Indoor','Reserved','C001');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T002','Indoor','Available','');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T003','Indoor','Reserved','C002');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T004','Indoor','Available','');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T005','Indoor','Reserved','C003');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T006','Outdoor','Available','');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T007','Outdoor','Reserved','C004');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T008','Outdoor','Available','');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T009','Outdoor','Reserved','C005');

INSERT INTO "Table" (Table\_ID, Table\_Location, Table\_Status, CUSTOMER\_ID)

VALUES

('T010','Outdoor','Available','');

INSERT INTO "Table" VALUES ('T011','Indoor','Occupied','C006');

INSERT INTO "Table" VALUES ('T012','Indoor','Occupied','C007');

INSERT INTO "Table" VALUES ('T013','Indoor','Occupied','C008');

INSERT INTO "Table" VALUES ('T014','Outdoor','Occupied','C009');

INSERT INTO "Table" VALUES ('T015','Outdoor','Occupied','C010');

INSERT INTO "Table" VALUES ('T016','Indoor','Occupied','C011');

INSERT INTO "Table" VALUES ('T017','Indoor','Occupied','C012');

INSERT INTO "Table" VALUES ('T018','Outdoor','Occupied','C013');

INSERT INTO "Table" VALUES ('T019','Outdoor','Occupied','C014');

INSERT INTO "Table" VALUES ('T020','Outdoor','Occupied','C015');

INSERT INTO "Table" VALUES ('T021','Indoor','Occupied','C016');

INSERT INTO "Table" VALUES ('T022','Indoor','Occupied','C017');

INSERT INTO "Table" VALUES ('T023','Outdoor','Occupied','C018');

INSERT INTO "Table" VALUES ('T024','Outdoor','Occupied','C019');

INSERT INTO "Table" VALUES ('T025','Outdoor','Occupied','C020');

-- Reservation Table

CREATE TABLE Reservation (

RESERVATION\_ID VARCHAR2(5) PRIMARY KEY,

RESERVATION\_Time VARCHAR2(8),

RESERVATION\_Date DATE,

CUSTOMER\_ID VARCHAR2(5),

Table\_ID VARCHAR2(5),

CONSTRAINT fk\_customer\_reservation FOREIGN KEY (CUSTOMER\_ID) REFERENCES Customer(CUSTOMER\_ID),

CONSTRAINT fk\_table\_reservation FOREIGN KEY (Table\_ID) REFERENCES "Table"(Table\_ID)

);

INSERT INTO Reservation (RESERVATION\_ID, RESERVATION\_Time, RESERVATION\_Date, CUSTOMER\_ID,Table\_ID)

VALUES

('R001','14:34',TO\_DATE('03-04-2024', 'DD-MM-YYYY '), 'C001','T001');

INSERT INTO Reservation (RESERVATION\_ID, RESERVATION\_Time, RESERVATION\_Date, CUSTOMER\_ID,Table\_ID)

VALUES

('R002','09:26',TO\_DATE('03-04-2024', 'DD-MM-YYYY '), 'C002','T003');

INSERT INTO Reservation (RESERVATION\_ID, RESERVATION\_Time, RESERVATION\_Date, CUSTOMER\_ID,Table\_ID)

VALUES

('R003','12:30',TO\_DATE('04-04-2024', 'DD-MM-YYYY '), 'C003','T005');

INSERT INTO Reservation (RESERVATION\_ID, RESERVATION\_Time, RESERVATION\_Date, CUSTOMER\_ID,Table\_ID)

VALUES

('R004','12:00',TO\_DATE('05-04-2024', 'DD-MM-YYYY '), 'C004','T007');

INSERT INTO Reservation (RESERVATION\_ID, RESERVATION\_Time, RESERVATION\_Date, CUSTOMER\_ID,Table\_ID)

VALUES

('R005','17:37',TO\_DATE('10-04-2024', 'DD-MM-YYYY '), 'C005','T009');

-- Staff Table

CREATE TABLE Staff (

STAFF\_ID VARCHAR2(5) PRIMARY KEY,

STAFF\_ShiftHour VARCHAR2(20),

STAFF\_Salary VARCHAR2(30),

STAFF\_FirstName VARCHAR2(30),

STAFF\_LastName VARCHAR2(30),

STAFF\_Phone VARCHAR2(12),

STAFF\_Role VARCHAR2(10)

);

INSERT INTO Staff VALUES ('S001', 'Morning','1300', 'Ong','Sheng', '0119877546', 'Admin');

INSERT INTO Staff VALUES ('S002', 'Morning','1000', 'Lee','Mary', '0178846524', 'Waiter');

INSERT INTO Staff VALUES ('S003', 'Evening','2000', 'Kim','Blue', '0124445876', 'Chef');

INSERT INTO Staff VALUES ('S004', 'Morning','1000', 'Lee','Hai', '0101423693', 'Waiter');

INSERT INTO Staff VALUES ('S005', 'Evening','2000', 'Anthony','Lee', '0174468521', 'Chef');

INSERT INTO Staff VALUES ('S006', 'Evening','1050', 'Grey','Tan', '0188898554', 'Waiter');

INSERT INTO Staff VALUES ('S007', 'Off','1300', 'Ng','Fang', '0136645127', 'Admin');

INSERT INTO Staff VALUES ('S008', 'Off','2050', 'White','Lee', '0124945756', 'Chef');

INSERT INTO Staff VALUES ('S009', 'Off','1800', 'Ming','Liang', '0177779546', 'Chef');

INSERT INTO Staff VALUES ('S010', 'Off','1980', 'Vary','Lu', '0197854624', 'Chef');

INSERT INTO Staff VALUES ('S011', 'Morning','1400', 'Wah','Fang', '0178912557', 'Admin');

INSERT INTO Staff VALUES ('S012', 'Morning','1360', 'Foong','Li', '0137788446', 'Admin');

INSERT INTO Staff VALUES ('S013', 'Morning','1200', 'Racheal','Ng', '0139998746', 'Admin');

INSERT INTO Staff VALUES ('S014', 'Off','900', 'John','Tan', '0198977542', 'Waiter');

INSERT INTO Staff VALUES ('S015', 'Off','1000', 'Sam','Smith', '0147566132', 'Waiter');

--Chef Table

CREATE TABLE Chef (

STAFF\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_staff\_chef FOREIGN KEY (STAFF\_ID) REFERENCES Staff(STAFF\_ID)

);

INSERT INTO Chef(STAFF\_ID) VALUES ('S003');

INSERT INTO Chef(STAFF\_ID) VALUES ('S005');

INSERT INTO Chef(STAFF\_ID) VALUES ('S008');

INSERT INTO Chef(STAFF\_ID) VALUES ('S009');

INSERT INTO Chef(STAFF\_ID) VALUES ('S010');

--Admin Table

CREATE TABLE Admin (

STAFF\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_staff\_admin FOREIGN KEY (STAFF\_ID) REFERENCES Staff(STAFF\_ID));

INSERT INTO Admin (STAFF\_ID) VALUES ('S001');

INSERT INTO Admin (STAFF\_ID) VALUES ('S007');

INSERT INTO Admin (STAFF\_ID) VALUES ('S011');

INSERT INTO Admin (STAFF\_ID) VALUES ('S012');

INSERT INTO Admin (STAFF\_ID) VALUES ('S013');

--Table Waiter

CREATE TABLE Waiter (

STAFF\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_staff\_waiter FOREIGN KEY (STAFF\_ID) REFERENCES Staff(STAFF\_ID)

);

INSERT INTO Waiter (STAFF\_ID) VALUES ('S002');

INSERT INTO Waiter (STAFF\_ID) VALUES ('S004');

INSERT INTO Waiter (STAFF\_ID) VALUES ('S006');

INSERT INTO Waiter (STAFF\_ID) VALUES ('S014');

INSERT INTO Waiter (STAFF\_ID) VALUES ('S015');

-- Inventory Table

CREATE TABLE Inventory (

INVENTORY\_ID VARCHAR2(5) PRIMARY KEY,

ITEM\_Name VARCHAR2(30),

ITEM\_Desc VARCHAR2(30),

ITEM\_Quantity NUMBER(4),

ITEM\_Unit NUMBER(10,1),

ITEM\_Price NUMBER(10,2)

);

INSERT INTO Inventory VALUES ('IT001', 'Flour','Corn Flour', '100', '1.0', '15.00');

INSERT INTO Inventory VALUES ('IT002', 'Oil','Olive oil', '250', '1.5', '25.00');

INSERT INTO Inventory VALUES ('IT003', 'Spices','Black pepper', '20', '0.2', '10.00');

INSERT INTO Inventory VALUES ('IT004', 'Chili','Chili Flakes', '20', '0.2', '10.00');

INSERT INTO Inventory VALUES ('IT005', 'Chicken','Chicken Breast, Chicken Thigh', '50', '1.0', '50.00');

INSERT INTO Inventory VALUES ('IT006', 'Tofu','White Tofu', '20', '1.0', '10.00');

INSERT INTO Inventory VALUES ('IT007', 'Cucumber','Japanese cucumber', '40', '5.0', '30.00');

INSERT INTO Inventory VALUES ('IT008', 'Rice','White Rice', '100', '5.0', '45.00');

INSERT INTO Inventory VALUES ('IT009', 'Sugar','Brown sugar', '100', '1.0', '30.00');

INSERT INTO Inventory VALUES ('IT010', 'Tea bags','green tea,peach tea and others', '55', '1.0', '45.00');

-- Food Table

CREATE TABLE Food (

FOOD\_ID VARCHAR2(5) PRIMARY KEY,

FOOD\_Name VARCHAR2(40),

FOOD\_Details VARCHAR2(100),

FOOD\_Price NUMBER(10,2),

INVENTORY\_ID VARCHAR2(5),

CUSTOMER\_ID VARCHAR2(5),

CONSTRAINT fk\_inventory\_food FOREIGN KEY (INVENTORY\_ID) REFERENCES Inventory(INVENTORY\_ID),

CONSTRAINT fk\_customer\_food FOREIGN KEY (CUSTOMER\_ID) REFERENCES Customer(CUSTOMER\_ID)

);

INSERT INTO Food VALUES ('F001', 'Nasi Lemak with Fried Chicken','served with roasted nuts,

egg, anchovies, and slices of cucumber', '15.90', 'IT008', 'C006');

INSERT INTO Food VALUES ('F002', 'Mapo Tofu with White Rice','tofu cooked with rich spicy, and savory sauce',

'8.00','IT006', 'C007');

INSERT INTO Food VALUES ('F003', 'Hainanese Chicken Rice','succulent poached white chicken served on fragrant rice with light soy sauce',

'18.00','IT005', 'C008');

INSERT INTO Food VALUES ('F004', 'Grilled Chicken Chop','marinated chicken covered in a rich, bold black pepper sauce',

'20.90', 'IT005', 'C009');

INSERT INTO Food VALUES ('F005', 'Teriyaki Chicken Poke Bowl','sushi rice and teriyaki chicken, topped with vegetables and roasted sesame',

'18.00', 'IT007', 'C010');

INSERT INTO Food VALUES ('F006', 'Cheese cake','serves with graham cracker crumb crust',

'15.00', 'IT001', 'C011');

INSERT INTO Food VALUES ('F007', 'Chocolate Lava cake','chocolate cake with a liquid chocolate core',

'11.00', 'IT001', 'C012');

INSERT INTO Food VALUES ('F008', 'Caramel Pudding','plain custard with sugar syrup',

'8.00', 'IT001', 'C013');

INSERT INTO Food VALUES ('F009', 'Almond Cupcake','with almond nuts and frosting',

'5.00', 'IT001', 'C014');

INSERT INTO Food VALUES ('F010', 'Brownies','a chocolate baked confection',

'10.00', 'IT001', 'C015');

INSERT INTO Food VALUES ('F011', 'Green Tea','green tea aroma',

'3.00', 'IT010', 'C016');

INSERT INTO Food VALUES ('F012', 'Iced Chocolate','served with chocolate chips',

'13.00', 'IT010', 'C017');

INSERT INTO Food VALUES ('F013', 'Peach Tea','fruity aroma',

'4.00', 'IT010', 'C018');

INSERT INTO Food VALUES ('F014', 'Ice lemon tea','fresh and tangy taste of lemon',

'4.00', 'IT010', 'C019');

INSERT INTO Food VALUES ('F015', 'Milk tea','combination of tea and milk',

'6.00', 'IT010', 'C020');

--Meal Table

CREATE TABLE Meal (

FOOD\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_food\_meal FOREIGN KEY (FOOD\_ID) REFERENCES Food(FOOD\_ID)

);

INSERT INTO Meal(FOOD\_ID) VALUES ('F001');

INSERT INTO Meal(FOOD\_ID) VALUES ('F002');

INSERT INTO Meal(FOOD\_ID) VALUES ('F003');

INSERT INTO Meal(FOOD\_ID) VALUES ('F004');

INSERT INTO Meal(FOOD\_ID) VALUES ('F005');

--Dessert Table

CREATE TABLE Dessert (

FOOD\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_food\_dessert FOREIGN KEY (FOOD\_ID) REFERENCES Food(FOOD\_ID)

);

INSERT INTO Dessert(FOOD\_ID) VALUES ('F006');

INSERT INTO Dessert(FOOD\_ID) VALUES ('F007');

INSERT INTO Dessert(FOOD\_ID) VALUES ('F008');

INSERT INTO Dessert(FOOD\_ID) VALUES ('F009');

INSERT INTO Dessert(FOOD\_ID) VALUES ('F010');

--Beverage Table

CREATE TABLE Beverage (

FOOD\_ID VARCHAR2(5) PRIMARY KEY,

CONSTRAINT fk\_food\_beverage FOREIGN KEY (FOOD\_ID) REFERENCES Food(FOOD\_ID)

);

INSERT INTO Beverage(FOOD\_ID) VALUES ('F011');

INSERT INTO Beverage(FOOD\_ID) VALUES ('F012');

INSERT INTO Beverage(FOOD\_ID) VALUES ('F013');

INSERT INTO Beverage(FOOD\_ID) VALUES ('F014');

INSERT INTO Beverage(FOOD\_ID) VALUES ('F015');

-- Order Table

CREATE TABLE Order\_ (

ORDER\_ID VARCHAR2(7) PRIMARY KEY,

ORDER\_Date DATE,

ORDER\_Time VARCHAR2(8),

ORDER\_Status VARCHAR2(20),

Table\_ID VARCHAR2(5),

STAFF\_ID VARCHAR2(5),

CONSTRAINT fk\_table\_order FOREIGN KEY (Table\_ID) REFERENCES "Table"(Table\_ID),

CONSTRAINT fk\_staff\_order FOREIGN KEY (STAFF\_ID) REFERENCES Staff(STAFF\_ID)

);

INSERT INTO Order\_ VALUES ('Ord001', TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'08:32',

'Completed','T011','S001');

INSERT INTO Order\_ VALUES ('Ord002',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'09:00',

'Completed','T012','S007');

INSERT INTO Order\_ VALUES ('Ord003',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'09:13',

'Completed','T013','S011');

INSERT INTO Order\_ VALUES ('Ord004',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'10:24',

'Completed','T014','S012');

INSERT INTO Order\_ VALUES ('Ord005',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'10:28',

'Completed','T015','S013');

INSERT INTO Order\_ VALUES ('Ord006',TO\_DATE('02-04-2024', 'DD-MM-YYYY'),'11:08',

'Processing','T016','S002');

INSERT INTO Order\_ VALUES ('Ord007',TO\_DATE('02-04-2024', 'DD-MM-YYYY'),'11:58',

'Processing','T017','S004');

INSERT INTO Order\_ VALUES ('Ord008',TO\_DATE('02-04-2024', 'DD-MM-YYYY'),'14:20',

'Processing','T018','S006');

INSERT INTO Order\_ VALUES ('Ord009',TO\_DATE('02-04-2024', 'DD-MM-YYYY'),'14:28',

'Processing','T019','S003');

INSERT INTO Order\_ VALUES ('Ord010',TO\_DATE('02-04-2024', 'DD-MM-YYYY'),'14:57',

'Processing','T020','S005');

-- Cart Table

CREATE TABLE Cart (

CART\_ID VARCHAR2(7) PRIMARY KEY,

CART\_Status VARCHAR2(20),

CUSTOMER\_ID VARCHAR2(5),

CONSTRAINT fk\_customer\_cart FOREIGN KEY (CUSTOMER\_ID) REFERENCES Customer(CUSTOMER\_ID)

);

INSERT INTO Cart VALUES ('Cart01', 'Processing','C011');

INSERT INTO Cart VALUES ('Cart02', 'Processing','C012');

INSERT INTO Cart VALUES ('Cart03', 'Processing','C013');

INSERT INTO Cart VALUES ('Cart04', 'Processing','C014');

INSERT INTO Cart VALUES ('Cart05', 'Processing','C015');

INSERT INTO Cart VALUES ('Cart06', 'Pending','C016');

INSERT INTO Cart VALUES ('Cart07', 'Pending','C017');

INSERT INTO Cart VALUES ('Cart08', 'Pending','C018');

INSERT INTO Cart VALUES ('Cart09', 'Pending','C019');

INSERT INTO Cart VALUES ('Cart10', 'Pending','C020');

-- Cart\_Food Table

CREATE TABLE Cart\_food (

CARTFOOD\_ID NUMBER(2) PRIMARY KEY,

FOOD\_ID VARCHAR2(5),

CART\_ID VARCHAR2(7),

CART\_FoodQuantity NUMBER(2),

CART\_FoodCustomization VARCHAR2(100),

CONSTRAINT fk\_food\_cartfood FOREIGN KEY (FOOD\_ID) REFERENCES Food(FOOD\_ID),

CONSTRAINT fk\_cart\_cartfood FOREIGN KEY (CART\_ID) REFERENCES Cart(CART\_ID)

);

INSERT INTO Cart\_Food VALUES ('1','F006','Cart01','1',NULL);

INSERT INTO Cart\_Food VALUES ('2','F007','Cart02','1','Extra chocolate liquid');

INSERT INTO Cart\_Food VALUES ('3','F008','Cart03','1',NULL);

INSERT INTO Cart\_Food VALUES ('4','F009','Cart04','1','Extra almond nuts');

INSERT INTO Cart\_Food VALUES ('5','F010','Cart05','1',NULL);

INSERT INTO Cart\_Food VALUES ('6','F011','Cart06','1',NULL);

INSERT INTO Cart\_Food VALUES ('7','F012','Cart07','1','More ice');

INSERT INTO Cart\_Food VALUES ('8','F013','Cart08','1','Hot');

INSERT INTO Cart\_Food VALUES ('9','F014','Cart09','1',NULL);

INSERT INTO Cart\_Food VALUES ('10','F015','Cart10','1','Less Ice');

-- Invoice Table

CREATE TABLE Invoice (

INVOICE\_ID VARCHAR2(5) PRIMARY KEY,

INVOICE\_Date DATE,

INVOICE\_Time VARCHAR2(8),

PAYMENT\_Method VARCHAR2(20),

PAYMENT\_Amount NUMBER(20,2),

CUSTOMER\_ID VARCHAR2(5),

ORDER\_ID VARCHAR2(7),

CONSTRAINT fk\_customer\_invoice FOREIGN KEY (CUSTOMER\_ID) REFERENCES Customer(CUSTOMER\_ID),

CONSTRAINT fk\_order\_invoice FOREIGN KEY (ORDER\_ID) REFERENCES Order\_(ORDER\_ID)

);

INSERT INTO Invoice VALUES ('IV001',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'09:45',

'Credit/Debit Card','15.90','C006','Ord001');

INSERT INTO Invoice VALUES ('IV002',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'10:03',

'QR Payment','8.00','C007','Ord002');

INSERT INTO Invoice VALUES ('IV003',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'13:30',

'Cash','18.00','C008','Ord003');

INSERT INTO Invoice VALUES ('IV004',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'15:30',

'Cash','20.90','C009','Ord004');

INSERT INTO Invoice VALUES ('IV005',TO\_DATE('01-04-2024', 'DD-MM-YYYY'),'17:03',

'Cash','18.00','C010','Ord005');

# **7.0 Individual Assignment (Part 2)**

## **7.1 Ong Yi Sheng (\*)**

### **7.1.1 Queries**

1. Shows the Staff's full name, order date and the table id, where the table location is indoor and arranged accordingly with the table id.

SELECT s.staff\_firstname || s.staff\_lastname AS "Full Name" , o.order\_date AS "Order Time", t.table\_id AS " Table ID", t.table\_location

FROM staff s, order\_ o, "Table" t

WHERE s.staff\_id = o.staff\_id

AND o.table\_id = t.table\_id

AND table\_location = 'Indoor'

ORDER BY t.table\_id;

2. Retrieve the Customer's full name, phone number, invoice ID, and payment Amount using the payment method as Cash and the payment amount is between 15 to 20 and arranged accordingly with the payment amount.

SELECT c.customer\_firstname || c.customer\_lastname AS "Full Name" , c.customer\_phone AS "Phone Number" , i.payment\_amount AS "Payment Amount"

FROM customer c , invoice i

WHERE c.customer\_id = i.customer\_id

AND payment\_method LIKE 'Cash'

AND payment\_amount BETWEEN 15.00 AND 20.00

ORDER BY payment\_amount;

### **7.1.2 PL/SQL stored procedure**

1) update a staff information by giving the staff ID.

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE UpdateStaff(

p\_Staff\_ID IN VARCHAR2,

p\_ShiftHour IN VARCHAR2,

p\_Salary IN VARCHAR2,

p\_FirstName IN VARCHAR2,

p\_LastName IN VARCHAR2,

p\_Phone IN VARCHAR2,

p\_Role IN VARCHAR2

)

AS

BEGIN

UPDATE Staff

SET staff\_shifthour = p\_ShiftHour,

staff\_salary = p\_Salary,

staff\_firstname = p\_FirstName,

staff\_lastname = p\_LastName,

staff\_phone = p\_Phone,

staff\_role = p\_Role

WHERE staff\_id = p\_Staff\_ID;

IF SQL%ROWCOUNT = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Staff member updated successfully.');

ELSIF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Staff ID not found.');

END IF;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

ROLLBACK;

END;

/

**Execution Code:**

EXECUTE UpdateStaff('S001','Off','5000','Ace','Smie','01122323','Chef');

**Explanation:**

This procedure is named as UpdateStaff and this procedure is to change the staff info by giving the staff ID. At the procedure definition part, there are 7 input parameters which is p\_staff\_ID, p \_shifthour, p\_Salary, p\_FirstName, p\_LastName, p\_Phone, and p\_Role. These parameters gonna carry the input and insert into the real data that already exists in the database inside the block of UPDATE. At SQL%ROWCOUNT = 1 this means that the sql selected one row which to the staffID, if rowcount = 1 prompt the info has been changed successfully. Hence, SQL%ROWCOUNT = 0 handle the error when the staff ID not found.

2) Insert a reservation to the table when a customer wants to make reservation and add the new reservation ID, reservation time, reservation date, customer ID and Table ID. The procedure shows error when the tableID has been reserved and occupied, and the procedure handle error when the customerID not exist. The table status will update to reserved after the customer make the reservation.

CREATE OR REPLACE PROCEDURE Add\_Reservation(

p\_reservation\_id IN VARCHAR2,

p\_reservation\_time IN VARCHAR2,

p\_reservation\_date IN DATE,

p\_customer\_id IN VARCHAR2,

p\_table\_id IN VARCHAR2

)

IS

BEGIN

DECLARE

customer\_exists NUMBER;

BEGIN

SELECT COUNT(\*)

INTO customer\_exists

FROM Customer

WHERE CUSTOMER\_ID = p\_customer\_id;

IF customer\_exists = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Customer ID does not exist.');

END IF;

END;

DECLARE

table\_status VARCHAR2(15);

BEGIN

SELECT Table\_Status

INTO table\_status

FROM "Table"

WHERE Table\_ID = p\_table\_id;

IF table\_status != 'Available' THEN

DBMS\_OUTPUT.PUT\_LINE('Table is not available for reservation.');

END IF;

END;

INSERT INTO Reservation (

RESERVATION\_ID,

RESERVATION\_Time,

RESERVATION\_Date,

CUSTOMER\_ID,

Table\_ID

) VALUES (

p\_reservation\_id,

p\_reservation\_time,

p\_reservation\_date,

p\_customer\_id,

p\_table\_id

);

UPDATE "Table"

SET Table\_Status = 'Reserved', CUSTOMER\_ID = p\_customer\_id

WHERE Table\_ID = p\_table\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

RAISE;

END Add\_Reservation;

/

Execution Code:

EXECUTE Add\_Reservation('R006', '18:00', TO\_DATE('12-04-2024', 'DD-MM-YYYY'), 'C002', 'T002');

**Explanation**:

This procedure name as Add\_Reservation which insert a reservation to the table when a customer wants to make reservation and add the new reservation ID, reservation time, reservation date, customer ID and Table ID.first of all, customers\_exist is a variable that test does the customerID exist or not. If not exist it will end and prompt the system error. Hence, table\_status is a variable that stored the data in the Table\_Status from the “Table” table. If the table status is not available it will prompt the table is not available for reservation. If the reservation is going successfully, the table\_status in the “Table” table will be updated to reserved and the customer ID in the “Table” table will be updated to the customer ID that input.

### **7.1.3 PS/SQL functions**

1. The function that calculate the numbers of order in a specfic date.

CREATE OR REPLACE FUNCTION CountOrdersOnDate(p\_date IN DATE)

RETURN NUMBER IS

order\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO order\_count

FROM Order\_

WHERE TRUNC(ORDER\_Date) = TRUNC(p\_date);

RETURN order\_count;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

WHEN OTHERS THEN

RAISE;

END;

/

Execution Code:

SELECT CountOrdersOnDate(TO\_DATE('01-04-2024', 'DD-MM-YYYY')) AS Total\_Order\_Amount FROM dual;

Explanation:

This function is named as CountOrdersOnDate. This function is function as to count the order count at a specified date. Hence, the function will return the order count by referring the date given in execution code.

2. Return the numbers of staff who's salary exceed and equal 1300 by giving the specfic staff role.

CREATE OR REPLACE FUNCTION CountSalaryByrole(p\_role IN VARCHAR2)

RETURN NUMBER IS

count\_staff NUMBER;

BEGIN

SELECT COUNT(\*)

INTO count\_staff

FROM Staff

WHERE STAFF\_Role = p\_role

AND TO\_NUMBER(STAFF\_Salary) >= 1300;

RETURN count\_staff;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

WHEN OTHERS THEN

RAISE;

END;

/

**Execution Code:**

SELECT CountSalaryByrole('Chef') AS "Staff's salary that exceed and equal 1300 " FROM dual;

**Explanation:**

This function is named as CountSalaryByrole. This function is function as to count the number of the staff which exceed and equal 1300. Hence, the function will return the number count of the staff who exceed and equal 1300 by referring the staff\_role that given in execution code.

## **7.2 Desmond Ho Jia Shen**

### **7.2.1 Queries**

**1) List Customer with reservation on 03-04-2024**

This query will select all the customers and their information if they made a reservation on 3 April 2024. It will show their first name, last name, contact number and the reservation date.

SELECT c.CUSTOMER\_FirstName, c.CUSTOMER\_LastName, c.CUSTOMER\_Phone, r.RESERVATION\_Date

from Customer c

join Reservation r ON c.CUSTOMER\_ID = r.CUSTOMER\_ID

where r.RESERVATION\_Date = '03-APR-24';

**2) Detailed Order Information with Customer and Table Data**

This query will select the order and table information of a customer. It will show their order ID, order date, order time, order status, customer id, customer name, table ID, table location, table status and the staff that is serving the customer.

SELECT

o.ORDER\_ID,

o.ORDER\_Date,

o.ORDER\_Time,

o.ORDER\_Status,

cust.CUSTOMER\_ID,

cust.CUSTOMER\_FirstName || ' ' || cust.CUSTOMER\_LastName AS Customer\_Name,

t.Table\_ID,

t.Table\_Location,

t.Table\_Status,

s.STAFF\_FirstName || ' ' || s.STAFF\_LastName AS Staff\_Name

FROM

Order\_ o

JOIN "Table" t ON o.Table\_ID = t.Table\_ID

JOIN Customer cust ON cust.CUSTOMER\_ID = t.CUSTOMER\_ID

JOIN Staff s ON o.STAFF\_ID = s.STAFF\_ID

WHERE

o.ORDER\_Date = TO\_DATE('2024-04-01', 'YYYY-MM-DD');

### **7.2.2 PL/SQL stored procedure**

**1) Add new Customer into the table**

This stored procedure will add a new customer when they want to have a meal in the restaurant. It will add the necessary data into the Customer table which will include their customer ID, customer first name, customer last name, customer email and customer contact number. This procedure will also prompt an output line (“'Error: Duplicate Value for a Unique Constraint”) if there is any duplicated value added into the table that is already existed.

CREATE OR REPLACE PROCEDURE AddNewCustomer(

p\_customer\_id VARCHAR2,

p\_first\_name VARCHAR2,

p\_last\_name VARCHAR2,

p\_email VARCHAR2,

p\_phone VARCHAR2)

IS

BEGIN

INSERT INTO Customer (CUSTOMER\_ID, CUSTOMER\_FirstName, CUSTOMER\_LastName, CUSTOMER\_Email, CUSTOMER\_Phone)

VALUES (p\_customer\_id, p\_first\_name, p\_last\_name, p\_email, p\_phone);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Duplicate Value for a Unique Constraint.');

END;

/

================================

**Example:**

EXECUTE AddNewCustomer ('C001','Dreamy','Bull','dreamy@1utar.my','0126615856');

**2) Update Order Status**

This stored procedure will update the order status of a customer’s order to either ‘Completed’, ‘Processing’, ‘Cancelled’. If the status updated is the same as the current value, a message will prompt (“No difference as current”), If the order status is updated successfully, it will prompt a message (“Order status updated successfully”), and if there is an unavailable order ID used, it will prompt (“Error: Order ID not found”). Then if there is an error for the procedure, it will run a rollback to ensure that the data remains unchanged.

CREATE OR REPLACE PROCEDURE UpdateOrderStatus(

p\_order\_id VARCHAR2,

p\_new\_status VARCHAR2)

IS

v\_current\_status VARCHAR2(20);

BEGIN

-- Retrieve the current status for the given order ID

SELECT ORDER\_Status INTO v\_current\_status

FROM Order\_

WHERE ORDER\_ID = p\_order\_id;

-- Check if the new status is the same as the current status

IF v\_current\_status = p\_new\_status THEN

DBMS\_OUTPUT.PUT\_LINE('No difference as current.');

ELSE

-- Update the order status if it's different

UPDATE Order\_

SET ORDER\_Status = p\_new\_status

WHERE ORDER\_ID = p\_order\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Order status updated successfully.');

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Order ID not found.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

-- Optional: Rollback changes if something goes wrong

ROLLBACK;

END;

/

======================================

**Example:**

EXECUTE UpdateOrderStatus ('Ord001','Processing');

### **7.2.3 PS/SQL functions**

**1) Calculate Total Orders for Customer**

This function will calculate the total orders for a customer by using the v\_order\_count variable. It will first count all the orders of a customer using their Customer ID and total the numbers. Then, we can select this function with a Customer ID to prompt the total orders in dual.

CREATE OR REPLACE FUNCTION TotalOrdersForCustomer(p\_customer\_id VARCHAR2)

RETURN NUMBER

IS

v\_order\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_order\_count

FROM Order\_ o

JOIN "Table" t ON o.Table\_ID = t.Table\_ID

WHERE t.CUSTOMER\_ID = p\_customer\_id;

RETURN v\_order\_count;

END;

/

===========================

**Example**:

select TotalOrdersForCustomer ('C006') from dual;

**2) Get Total Revenue on a Date**

This function will calculate the total revenue on a certain date. It will first create a total\_revenue variable to store the amount of revenue earned. Then it will select and sum all the payment\_amount from the invoice table using a certain date. Finally, it will show the total revenue in dual.

CREATE OR REPLACE FUNCTION GetTotalRevenue(p\_date DATE)

RETURN NUMBER

IS

v\_total\_revenue NUMBER(10,2);

BEGIN

SELECT SUM(PAYMENT\_Amount)

INTO v\_total\_revenue

FROM Invoice

WHERE INVOICE\_Date = p\_date;

RETURN v\_total\_revenue;

END;

/

===========================

**Example:**

select GetTotalRevenue ('01-Apr-2024') from dual;

## **7.3 Seow Yi Xuan**

### **7.3.1 Queries**

1.Retrieve the table number and food name for completed orders and display the invoice id and the time of invoice generated for each food

SELECT t.TABLE\_ID AS "Table", f.FOOD\_NAME, i.INVOICE\_ID AS "Inv"

, i.INVOICE\_Time AS "Time"

FROM "Table" t, Food f, Order\_ o, Invoice i,Customer c

WHERE c.CUSTOMER\_ID = t.CUSTOMER\_ID

AND t.CUSTOMER\_ID = i.CUSTOMER\_ID

AND f.CUSTOMER\_ID = i.CUSTOMER\_ID

AND o.TABLE\_ID = t.TABLE\_ID

AND TABLE\_STATUS = 'Occupied'

AND ORDER\_STATUS = 'Completed';

2.Select the Customer whose cart contains food which uses "Tea bags" or "Flour" as ingredient

SELECT c.CUSTOMER\_ID, f.FOOD\_NAME, i.ITEM\_NAME AS "Ingredients", i.ITEM\_DESC AS "Description"

FROM CART c, FOOD f, Inventory i, CART\_FOOD t

WHERE c.CART\_ID = t.CART\_ID

AND t.FOOD\_ID = f.FOOD\_ID

AND f.INVENTORY\_ID = i.INVENTORY\_ID

AND (ITEM\_NAME = 'Tea bags' OR ITEM\_NAME = 'Flour');

### **7.3.2 PL/SQL stored procedure**

1.Insert new Table

SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE insertTable (

new\_Table\_ID IN VARCHAR2,

new\_Table\_Location IN VARCHAR2,

new\_Table\_Status IN VARCHAR2,

new\_CUSTOMER\_ID IN VARCHAR2

)

IS

BEGIN

INSERT INTO "Table" (Table\_ID, TABLE\_LOCATION, TABLE\_STATUS, CUSTOMER\_ID)

VALUES (new\_Table\_ID, new\_Table\_Location, new\_Table\_Status, new\_CUSTOMER\_ID);

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Duplicate Value for a Unique Constraint.');

END;

/

Execution Code :

EXECUTE insertTable ('T026','Indoor','Available','');

2.Update item quantity in Inventory where users cannot enter quantity less than 20

CREATE OR REPLACE PROCEDURE updateInventory (

new\_INVENTORY\_ID IN VARCHAR2,

new\_ITEM\_Quantity IN NUMBER

)

IS

-- Current item quantity

current\_item\_stock NUMBER;

BEGIN

-- Retrieve current item quantity

SELECT ITEM\_Quantity INTO current\_item\_stock

FROM Inventory

WHERE INVENTORY\_ID = new\_INVENTORY\_ID;

-- Check if new quantity meets the constraint

IF (new\_ITEM\_Quantity < 20) THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Quantity cannot be less than 20.');

RETURN;

END IF;

-- Update inventory quantity and provide feedback

IF (new\_ITEM\_Quantity < current\_item\_stock) THEN

DBMS\_OUTPUT.PUT\_LINE('Item quantity decreases');

ELSIF (new\_ITEM\_Quantity > current\_item\_stock) THEN

DBMS\_OUTPUT.PUT\_LINE('Item restocked');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Item quantity unchanged');

END IF;

-- Update the inventory with the new quantity

UPDATE Inventory

SET ITEM\_Quantity = new\_ITEM\_Quantity

WHERE INVENTORY\_ID = new\_INVENTORY\_ID;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Inventory ID not in list.');

END;

/

Execution Code :

EXECUTE updateInventory ('IT004',22);

### **7.3.3 PS/SQL functions**

1.Calculate total salary of staff who is a Chef.

CREATE OR REPLACE FUNCTION ChefSalary(STAFF\_SALARY IN VARCHAR2)

RETURN VARCHAR

IS

total\_salary NUMBER(10,2);

BEGIN

SELECT SUM(STAFF\_SALARY) INTO total\_salary

FROM STAFF

WHERE STAFF\_ROLE = 'Chef';

RETURN TO\_CHAR(total\_salary);

END;

/

DECLARE

--Declare variable as NUMBER to hold 2 decimal place

chef\_total\_salary NUMBER(10,2);

BEGIN

--Convert result

chef\_total\_salary := TO\_NUMBER(ChefSalary('Chef'));

DBMS\_OUTPUT.PUT\_LINE('Total salary of all chefs: RM' || TO\_CHAR(chef\_total\_salary, '9999.99'));

END;

/

2.Return total number of tables which are located at outdoor and occupied.

CREATE OR REPLACE FUNCTION OccupiedOutdoorTables(TABLE\_LOCATION IN VARCHAR2)

RETURN NUMBER

IS

occupied\_tables NUMBER;

BEGIN

SELECT COUNT(TABLE\_LOCATION) INTO occupied\_tables

FROM "Table"

WHERE TABLE\_LOCATION = 'Outdoor'

AND TABLE\_STATUS = 'Occupied';

RETURN occupied\_tables;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

END;

/

DECLARE

num\_occupied\_tables NUMBER;

BEGIN

num\_occupied\_tables := OccupiedOutdoorTables('Outdoor');

DBMS\_OUTPUT.PUT\_LINE('Number of occupied outdoor tables: ' || num\_occupied\_tables);

END;

/

## **7.4 Tharini Vijesh Kumar**

### **7.4.1 Queries**

* 1. List waiters with earnings from completed orders on 01-04-2024 and 02-04-2024.

SELECT

i.INVOICE\_Date,

(s.STAFF\_FirstName || ' ' || s.STAFF\_LastName) AS "FULL NAME",

s.STAFF\_Phone AS "Contact. No",

SUM(i.PAYMENT\_Amount) AS "Total Amount"

FROM

Staff s

JOIN

Order\_ o ON s.STAFF\_ID = o.STAFF\_ID

JOIN

Invoice i ON o.ORDER\_ID = i.ORDER\_ID

WHERE

i.INVOICE\_Date BETWEEN TO\_DATE('01-04-2024', 'DD-MM-YYYY') AND TO\_DATE('02-04-2024', 'DD-MM-YYYY')

AND o.ORDER\_Status = 'Completed'

GROUP BY

i.INVOICE\_Date, s.STAFF\_ID, (s.STAFF\_FirstName || ' ' || s.STAFF\_LastName), s.STAFF\_Phone

ORDER BY

i.INVOICE\_Date, s.STAFF\_ID;

* 1. list all ingredients, their unit left, price and counts the number of food items that are cooked with each ingredient

SELECT

i.ITEM\_Name AS "Ingredient Name",

i.ITEM\_Unit AS "Unit",

i.ITEM\_Price AS "Price",

COUNT(f.FOOD\_ID) AS "Number of Foods"

FROM

Inventory i

LEFT JOIN

Food f ON i.INVENTORY\_ID = f.INVENTORY\_ID

GROUP BY

i.ITEM\_Name, i.ITEM\_Desc, i.ITEM\_Quantity, i.ITEM\_Unit, i.ITEM\_Price

ORDER BY

"Number of Foods" DESC, i.ITEM\_Name;

### **PL/SQL stored procedure**

1. add ingredients into the inventory for a new menu

CREATE OR REPLACE PROCEDURE AddNewIngredients AS

BEGIN

INSERT INTO Inventory (INVENTORY\_ID, ITEM\_Name, ITEM\_Desc, ITEM\_Quantity, ITEM\_Unit, ITEM\_Price)

VALUES ('IT011', 'Sweetcorn', 'Frozen sweetcorn', '50', '1', '1.00');

INSERT INTO Inventory (INVENTORY\_ID, ITEM\_Name, ITEM\_Desc, ITEM\_Quantity, ITEM\_Unit, ITEM\_Price)

VALUES ('IT012', 'Peas', 'Frozen peas', '50', '1', '1.00');

INSERT INTO Inventory (INVENTORY\_ID, ITEM\_Name, ITEM\_Desc, ITEM\_Quantity, ITEM\_Unit, ITEM\_Price)

VALUES ('IT013', 'Red Onion', 'Fresh red onion', '1', '2', '0.50');

INSERT INTO Inventory (INVENTORY\_ID, ITEM\_Name, ITEM\_Desc, ITEM\_Quantity, ITEM\_Unit, ITEM\_Price)

VALUES ('IT014', 'Lemon', 'Fresh lemon', '1', '2', '0.75');

INSERT INTO Inventory (INVENTORY\_ID, ITEM\_Name, ITEM\_Desc, ITEM\_Quantity, ITEM\_Unit, ITEM\_Price)

VALUES ('IT015', 'Cajun Seasoning', 'Schwartz Cajun Seasoning', '1', '3', '2.50');

COMMIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Duplicate Value for a Unique Constraint.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END AddNewIngredients;

/

EXEC AddNewIngredients;

1. Displays alert message when inventory item runs low

CREATE OR REPLACE PROCEDURE InventoryReorderAlert AS

v\_message VARCHAR2(1000);

BEGIN

FOR item IN (SELECT INVENTORY\_ID, ITEM\_Name, ITEM\_Quantity

FROM Inventory

WHERE ITEM\_Quantity < 5)

LOOP

v\_message := 'Inventory Alert: Item ' || item.ITEM\_Name ||

' (ID: ' || item.INVENTORY\_ID || ')' ||

' has a quantity of ' || item.ITEM\_Quantity ||

'. Please reorder as soon as possible.';

DBMS\_OUTPUT.PUT\_LINE(v\_message);

END LOOP;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END InventoryReorderAlert;

/

EXEC InventoryReorderAlert;

### **PS/SQL functions**

1. Cancels an existing order.

CREATE OR REPLACE FUNCTION CancelOrder(

p\_order\_id VARCHAR2

) RETURN VARCHAR2

IS

v\_status VARCHAR2(20);

v\_result VARCHAR2(100);

BEGIN

SELECT ORDER\_Status INTO v\_status

FROM Order\_

WHERE ORDER\_ID = p\_order\_id;

IF v\_status = 'Cancelled' THEN

v\_result := 'Order ' || p\_order\_id || ' has already been cancelled.';

RETURN v\_result;

END IF;

UPDATE Order\_

SET ORDER\_Status = 'Cancelled'

WHERE ORDER\_ID = p\_order\_id;

IF SQL%ROWCOUNT > 0 THEN

v\_result := 'Order ' || p\_order\_id || ' has been cancelled successfully.';

ELSE

v\_result := 'No order found with ID: ' || p\_order\_id;

END IF;

RETURN v\_result;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'No order found with ID: ' || p\_order\_id;

WHEN OTHERS THEN

RETURN 'Error occurred while cancelling the order: ' || SQLERRM;

END CancelOrder;

/

DECLARE

v\_result VARCHAR2(100);

BEGIN

v\_result := CancelOrder('Ord009');

DBMS\_OUTPUT.PUT\_LINE(v\_result);

END;

/

1. Displays all carts with the pending status along with the customer\_id

CREATE OR REPLACE FUNCTION GetUnclaimedCarts

RETURN SYS\_REFCURSOR

IS

v\_cart\_cursor SYS\_REFCURSOR;

BEGIN

OPEN v\_cart\_cursor FOR

SELECT

CART\_ID,

CART\_STATUS,

CUSTOMER\_ID

FROM

cart

WHERE

CART\_STATUS = 'Pending';

RETURN v\_cart\_cursor;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

RETURN NULL;

END GetUnclaimedCarts;

/

DECLARE

v\_cart\_cursor SYS\_REFCURSOR;

v\_cart\_id VARCHAR2(7);

v\_cart\_status VARCHAR2(20);

v\_customer\_id VARCHAR2(5);

BEGIN

v\_cart\_cursor := GetUnclaimedCarts();

LOOP

FETCH v\_cart\_cursor INTO v\_cart\_id, v\_cart\_status, v\_customer\_id;

EXIT WHEN v\_cart\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Cart ID: ' || v\_cart\_id);

DBMS\_OUTPUT.PUT\_LINE('Status: ' || v\_cart\_status);

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_customer\_id);

DBMS\_OUTPUT.PUT\_LINE('--------------------------');

END LOOP;

CLOSE v\_cart\_cursor;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

END;

/