

**UNIVERSITI TUNKU ABDUL RAHMAN**

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

**UCCD2303 DATABASE TECHNOLOGY /**

**UCCD2203 DATABASE SYSTEMS**

**Group Assignment Mark sheet**

February 2025 Trimester

|  |  |
| --- | --- |
| Group Number (e.g. G999): | G37 |
| Group leader to provide the OneDrive folder hyperlink (editor mode) which contains the zip file and a group presentation video in mp4 format: | [Group37.mp4](https://utarict-my.sharepoint.com/:v:/g/personal/owiwilson123_1utar_my/ETjUPmE2_LtBrmWo9mVOYRsBCS0nvsKnXhUbMQfZ_leVlA?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=6w0ypU) |
| Group leader name: | Owi Wilson |
| Group leader email address: | owiwilson123@1utar.my |
| Submission date (dd-Mon-yy): | 29-04-2025 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Member Contribution Declaration Sheet** | | | | | |
|  | | **Member 1** | **Member 2** | **Member 3** | **Member 4** |
| **Member Name** (ascending order, group leader with \* ): | | **Ng Wei Jie** | **Owi Wilson \*** | **Tan Cheng Lock** | **Yee Hao Zhe** |
| Student ID (e.g. 2299999) | | **2205566** | **2206448** | **2205118** | **2205744** |
| Program CS / IA, IB / DE: | | **CS** | **CS** | **CS** | **CS** |
| **Member’s signature** | |  |  |  |  |
|  | | Contribution of each member (Total 100%) | | | |
|  | | **%** | **%** | **%** | **%** |
| PART 1: (Group Assessment – 50 marls) | |  |  |  |  |
| **1.** | Scope of Work (5 marks). |  |  |  |  |
| **2.** | ER model (10 marks). |  |  |  |  |
| **3.** | Redesign and EER (10 marks). |  |  |  |  |
| **4.** | Data Dictionary (10 marks) |  |  |  |  |
| **5.** | Tables and records (5 marks) |  |  |  |  |
| **6.** | Script (10 marks) |  |  |  |  |

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, screenshots are not required.* | 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)** |  | **Ng Wei Jie** | **Owi Wilson** | **Tan Cheng Lock** | **Yee Hao Zhe** |
| **Student ID; (to fill)** |  | **2205566** | **2206448** | **2205118** | **2205744** |
| 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)** |  |  |  |  |  |

|  |
| --- |
| **Part 2 Individual Report within the group report.**  Include all members’ individual SQL and PL/SQL answers into the group report, screenshots are not required. |
| **Part 2 Individual Presentation videos in ONE group video**   1. Group members are required to present two SQL queries, two PL/SQL stored procedures, and two PL/SQL 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. | Ng Wei Jie | Contributed for Data Dictionary, Tables and Records, Scripts |
| 2. | Owi Wilson | Contributed for ERD, Data Dictionary, Scripts |
| 3. | Tan Cheng Lock | Contributed for ERD, EERD, Data Dictionary, Scripts |
| 4. | Yee Hao Zhe | Contributed for Data Dictionary, Tables and Records, Scripts |

**Table of Contents**

[1.0 Scope of Work 6](#_Toc196771494)

[1.1 Introduction 6](#_Toc196771495)

[1.2 Functional Scope 6](#_Toc196771496)

[1.3 Requirement Analysis & Business Rules 7](#_Toc196771497)

[1.4 Assumptions 8](#_Toc196771498)

[2.0 ERD Model 9](#_Toc196771499)

[3.0 Redesign and EERD 10](#_Toc196771500)

[4.0 Data Dictionary 11](#_Toc196771501)

[5.0 Tables And Record 24](#_Toc196771502)

[6.0 Script 36](#_Toc196771503)

[7.0 Individual SQL Script 54](#_Toc196771504)

[Member 1: Ng Wei Jie 54](#_Toc196771505)

[Member 2: Owi Wilson 59](#_Toc196771506)

[Member 3: Tan Cheng Lock 64](#_Toc196771507)

[Member 4: Yee Hao Zhe 69](#_Toc196771508)

# 1.0 Scope of Work

## 1.1 Introduction

Our company has been appointed and develop a Reward and Point Management System for The Foodie, a fast-food restaurant under the Grand Wooly conglomerate. This system will complement customer loyalty efforts by managing point-based transactions, membership tiers, and reward redemptions, providing a seamless and engaging experience across all touchpoints.

## 1.2 Functional Scope

The proposed system will support the following 10 core functionalities:

1. **Customer-Initiated Transactions**

Customers can perform various transactions, including purchasing items to earn points or redeeming available rewards using their accumulated points.

1. **Customer Payment Management**

The system supports multiple payment methods, including e-wallets, credit/debit cards, and online banking, providing flexibility for customer preferences. Additionally, customers can review the total payment amounts they have made through the system, ensuring better personal record-keeping and transparency.

1. **Point Management System**

Each customer holds a points account, with points categorized into active and expired. Points are automatically updated based on earning and expiry rules.

1. **Membership Assignment**

Customers are assigned membership tiers (e.g., Bronze, Silver, Gold) based on their accumulated points and engagement level.

1. **Tier-Based Benefits**

Membership tiers provide specific benefits. For example, ‘Gold’ members may enjoy various benefits such as promotion, voucher as well as free gift.

1. **Transaction Information Access**

Customers can easily access their transaction details, including the transaction status (Completed, Pending, Failed) and type (Earn, Redeem). They can also view a complete history of their point earnings and redemptions, helping them to track their activities and understand their tier progression within the system.

1. **Reward Redemption Using Points**

Customers can redeem rewards which are still available in stock using their active points, with the system validating point sufficiency before approval.

1. **Real-Time Redemption Tracking**

Customers can view the status of their redemptions (e.g., Pending, Completed, Cancelled) to ensure transparency and confidence in reward processing.

1. **Reward Catalog Browsing**

Customers can browse a list of available rewards, including detailed descriptions and the required number of points for each item.

1. **Reward Restocking Management**

The system manages stock levels of rewards to prevent over-redemption and ensure availability, with administrators able to monitor and restock as needed.

## 1.3 Requirement Analysis & Business Rules

1. **Transaction Rule**

**Rule: Points can only be earned or redeemed through valid and completed transactions.**

Explanation: This ensures that points are tied to legitimate customer activities. If a transaction is cancelled or fails during processing (e.g., a customer starts a redemption but cancels while it's still pending), no points will be awarded or deducted. The system will strictly enforce this rule to maintain data integrity and prevent manipulation.

1. **Reward Rule**

**Rule: If a reward is out of stock, the reward will be automatically assigned as Inactive in the reward catalog.**

Explanation: This helps customers quickly identify which rewards are currently available and prevents them from attempting to redeem unavailable items. The system’s inventory module will update stock levels in real-time to reflect availability.

1. **Point Rule**

**Rule: Customers earn 1 point for every RM1 spent. Points have a validity period and will expire if not used.**

Explanation: This encourages ongoing customer engagement by rewarding spending while also introducing an urgency to redeem points before they expire. The system will manage point balances by tracking issuance dates and automatically expiring points after a specified period.

1. **Redemption Rule**

**Rule: Each redemption request must carry a status: Pending, Completed or Cancelled.**

Explanation: Having a status system enables customers and staff to monitor the progress of redemptions. In the event of a failed or canceled redemption, points can be refunded to the customer. This promotes transparency and improves service resolution.

1. **Membership Rule**

**Rule: Customers require an additional 1000 points to upgrade to the next membership tier.**

Explanation: Membership tiers (e.g., Bronze à Silver à Gold🡪Platinum) , are based on point thresholds to incentivize repeat purchases, except ‘Diamond’ which needed 5000 points for achieving. As customers accumulate points, the system automatically upgrades them to higher tiers, unlocking greater benefits. This structure encourages long-term loyalty and continued interaction with the system.

## 1.4 Assumptions

* **Hardware Assumption:** The system will be accessible via mobile app and web portal.
* **Network Assumption:** Stable internet is required to sync real-time point transactions, payment, reward status, and tier updates.
* **User Behavior Assumption:** Majority of customers will prefer using e-wallets or cards for convenience.
* **Staff Training Assumption:** Staff will be trained to monitor inventory, approve redemptions, and manage catalog.
* **Third-Party Integration Assumption:** Loyalty and payment APIs from vendors like PayPal, GrabPay, or Touch ’n Go will be available for integration.

# 2.0 ERD Model

A diagram of a computer

AI-generated content may be incorrect.

# 3.0 Redesign and EERD

A diagram of a computer

AI-generated content may be incorrect.

# 4.0 Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Description of Entities | | | |
| Entity Name | Description | Aliases | Occurrence |
| Customer | Represents an individual registered in the system. | Guest | One customer registers an account and can have one membership, checking point balance, make redemption, and review transaction history. |
| Transaction | Records points earned or redeemed by customers when using the system. | Point Transaction | One customer can make many point transactions which could be earning or redeeming points. |
| TransactionHistory | Maintains a detailed log of all past transactions of a customer. | - | A history record corresponds to a specific transaction made by a customer. |
| Payment | Stores payment details related to a transaction (purchase item then earned points). | - | A payment is tied to a single transaction (earn) made by a customer. |
| Ewallet | Subclass of Payment as customer using an electronic wallet application to purchase item. | - | Each e-wallet record is associated with one specific payment. |
| Card | Subclass of Payment as customer using a credit or debit card to purchase item. | - | Each card record is associated with one specific payment. |
| Online Banking | Subclass of Payment as customer using an online bank transfer to purchase item. | - | Each online banking record is associated with one specific payment. |
| Redemption | Logs details of rewards redeemed by customers. | - | A customer can make multiple redemptions of a rewards. |
| Reward | An item or things that can be redeemed by using points. | - | Each reward can be redeemed multiple times by a customer. |
| RewardCatalog | The catalog is used to categorize all available rewards. | Reward List | A reward catalog contains multiple categories of rewards. |
| Restocking | Replenishing rewards into inventory when stock levels decrease. | - | One restocking record replenishes the stock of a specific reward into a specific inventory block. |
| Inventory | Track available stock quantity for each reward. | Stock Management | One inventory stores multiple rewards, tracking their stock levels and capacity. |
| Point | Loyalty points earned by customers. | Point Balance | One customer has one point account that tracks their points. |
| PointRecord | Specific detailed log of all point changes (earned or used). | - | One point record logs either points earned, or points used by a customer. |
| Active | Subclass of **Point** tracks active points and its expiration date. | Active Point | Each active record tracks points that are still valid for redemption. |
| Expiry | Subclass of **Point** tracks expired points. | Expiry Point | Each expiry record tracks points that have expired and cannot be used. |
| Membership | Records of customer membership details and status. | Member | A membership is assigned to one customer, and their points determine their membership tier. |
| MembershipTier | Define different levels of tiers of membership. | Member Level | Each membership tier defines the minimum points needed to qualify for that level. |
| MembershipBenefit | List down all available benefits for members based on their tier. | Member Benefit | Each membership benefit is assigned to a specific membership tier. |
| Promotion | A special offer or discount that members can use, usually tied to specific products or services. | Promo offer | Each voucher record offers a discount and is linked to a membership benefit. |
| Voucher | A type of benefit providing a redeemable discount or value code. | Discount Code | Each free gift is linked to a membership benefit and has eligibility rules. |
| FreeGift | A type of benefit providing a physical gift item based on eligibility. | - | Each promotion offers discounts and is linked to a membership benefit. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of Relationship | | | | |
| Entity name | Multiplicity | Relationship | Multiplicity | Entity name |
| Customer | 1..1  1..1  1..1  1..1  1..1 | Has  Check  Make  Has  Make | 1..1  1..1  1..\*  1..\*  1..\* | Membership  Point  Transaction  TransactionHistory  Redemption |
| Membership | 1..\*  1..1  1..1 | Belongs to  Belongs to  Belongs to | 1..1  1..1  1..1 | MembershipTier  Customer  Point |
| MembershipTier | 1..1  1..1 | Has  Has | 1..\*  1..\* | Membership  MembershipBenefit |
| MembershipBenefit | 1..\* | Belongs to | 1..1 | MembershipTier |
| Point | 1..1  1..1  1..1 | Included  Generate  Check by | 1..1  1..1  1..1 | Membership  PointRecord  Customer |
| PointRecord | 1..1 | Generated by | 1..1 | Point |
| Transaction | 1..\*  1..1  1..1  1..1 | Made by  Earned by  Redeemed by  Generate | 1..1  0..1  1..\*  1..1 | Customer  Payment  Redemption  TransactionHistory |
| TransactionHistory | 1..1  1..\* | Generated by  Belongs to | 1..1  1..1 | Transaction  Customer |
| Payment | 0..1 | Earn | 1..1 | Transaction |
| Redemption | 1..\*  1..\*  1..\* | Made by  Redeem  Redeem | 1..1  1..1  1..1 | Customer  Reward  Transaction |
| Reward | 1..1  1..1  1..\* | Redeemed by  Arranged by  Listed by | 1..\*  1..1  1..1 | Redemption  Restocking  RewardCatalog |
| RewardCatalog | 1..1 | List | 1..\* | Reward |
| Restocking | 1..\*  1..\* | Arranged  Stored by | 1..1  1..1 | Reward  Inventory |
| Inventory | 1..1 | Store | 1..\* | Restocking |

**Customer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| CustID | NUMBER | 5 | Primary Key | Unique ID for customer |
| CustName | VARCHAR2 | 20 | Not Null | Full name of the customer |
| PhoneNum | VARCHAR2 | 11 | Not Null | Customer's phone number |
| CustEmail | VARCHAR2 | 30 | Not Null | Customer's email address |
| MemID | VARCHAR2 | 5 | Foreign Key | References to **Membership** table |

**Transaction**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| TransactionID | VARCHAR2 | 5 | Primary Key | Unique ID for each transaction |
| TransactionType | VARCHAR2 | 10 | Check (‘Redeem’, ‘Earn’) | Specifies the type of transaction |
| PointsEarned | NUMBER | 8 | Default = 0 | Number of points earned in the transaction |
| PointsRedeemed | NUMBER | 8 | Default = 0 | Number of points redeemed in the transaction |
| TransactionDate | DATE | DD-MM-YYYY | Default SYSDATE | Date of the transaction |
| TransactionStatus | VARCHAR2 | 10 | Check (‘Pending’, ‘Failed’, ‘Completed’) | Current status of the transaction |
| CustID | NUMBER | 5 | Foreign Key | References to **Customer** table |

**TransactionHistory**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| HistoryID | VARCHAR2 | 5 | Primary Key | Unique ID for each transaction history |
| TransactionType | VARCHAR2 | 10 | Check (‘Redeem’, ‘Earn’) | Specifies the type of transaction |
| PointsEarned | NUMBER | 8 | Default = 0 | Number of points earned |
| PointsRedeemed | NUMBER | 8 | Default = 0 | Number of points redeemed |
| TransactionID | VARCHAR2 | 5 | Foreign Key | References to **Transaction** table |
| CustID | NUMBER | 5 | Foreign Key | References to **Customer** table |

**Payment (Superclass)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| PaymentID | VARCHAR2 | 8 | Primary Key | Unique ID for each payment |
| TotalAmount | NUMBER | 10, 2 | Not Null | Total amount charged in payment |
| PaymentStatus | VARCHAR2 | 10 | Check (‘Pending, Completed’, ‘Failed’) | Current status of the payment |
| PaymentMethod | VARCHAR2 | 10 | Check (‘E-Wallet’, ‘Card’, ‘Online Banking’) | Method used for payment |
| TransactionID | VARCHAR2 | 5 | Foreign Key | References to **Transaction** table |

**E-Wallet (Subclass – inherit from Payment)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| PaymentID | VARCHAR2 | 8 | Primary Key, Foreign Key | Inherits from **Payment** table |
| Wallet\_ID | NUMBER | 15 | Primary Key | Unique ID of the e-wallet used for the transaction |
| Wallet\_Application | VARCHAR2 | 10 | Not Null | Name of the e-wallet application/ provider (e.g. PayPal, GrabPay, Touch ’n Go) |
| Currency\_Type | VARCHAR2 | 10 | Not Null | Currency used in the e-wallet payment (e.g. ‘MYR’, USD’) |

**Card (Subclass – inherit from Payment)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| PaymentID | VARCHAR2 | 8 | Primary Key, Foreign Key | Inherits from **Payment** table |
| Card\_Number | NUMBER | 20 | Primary Key | Card number used for the payment |
| Card\_Holder\_Name | VARCHAR2 | 50 | Not Null | Name of the card holder |
| Card\_Type | VARCHAR2 | 10 | Not Null | Type of card used in the payment |

**Online Banking (Subclass – inherit from Payment)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| PaymentID | VARCHAR2 | 8 | Primary Key, Foreign Key | Inherits from **Payment** table |
| Bank\_Acc\_Number | NUMBER | 10 | Primary Key | Bank account number used for the payment |
| Bank\_Name | VARCHAR2 | 20 | Not Null | Name of the bank handling the payment |

**Redemption**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| RedemptionID | VARCHAR2 | 5 | Primary Key | Unique ID for each redemption |
| CustID | NUMBER | 5 | Foreign Key | References to **Customer** table |
| RewardID | VARCHAR2 | 5 | Foreign Key | References to **Reward** table |
| RedemptionDate | DATE | DD-MM-YYYY | Default SYSDATE | Date of redemption |
| Status | VARCHAR2 | 15 | Check (‘Successful’,‘Unsuccessful’) | Status of the redemption attempt |
| TransactionID | VARCHAR2 | 5 | Foreign Key | References to **Transaction** table |
| PointsUsed | NUMBER | 5 | Not Null | Number of points used |

**Reward**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| RewardID | VARCHAR2 | 5 | Primary Key | Unique ID for each reward offered |
| RewardName | VARCHAR2 | 30 | Not Null | Name of the reward |
| PointsRequired | NUMBER | 5 | Not Null | Points needed to redeem |
| RewardType | VARCHAR2 | 20 | Not Null | Type of reward. |
| Stock | NUMBER | 3 | Default = 0, Not Null | Available stock or quantity from inventory |
| CatalogID | VARCHAR2 | 5 | Foreign Key | References to **RewardCatalog** table |

**RewardCatalog**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| CatalogID | VARCHAR2 | 5 | Primary Key | Unique ID for reward catalog |
| RewardName | VARCHAR2 | 20 | Not Null | Name of the reward |
| Description | VARCHAR2 | 100 | - | Description of the reward |
| IsActive | VARCHAR2 | 10 | Check (‘Active’, ‘Inactive’) | Use to indicate that active or inactive |

**Restocking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| RestockID | VARCHAR2 | 5 | Primary Key | Unique ID for restocking |
| RewardID | VARCHAR2 | 5 | Foreign Key | References to **Reward** table |
| InventoryID | VARCHAR2 | 4 | Foreign Key | References to **Inventory** table |
| RestockingDate | DATE | DD-MM-YYYY | Default  SYSDATE | Date when restocking occured |
| Quantity | NUMBER | - | - | Number of items restocked |
| SupplierID | VARCHAR2 | 5 | NOT NULL | ID of the supplier providing the stock |

**Inventory**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| InventoryID | VARCHAR2 | 4 | Primary Key | Unique ID for inventory |
| InventoryBlock | VARCHAR2 | 3 | - | Block / Location of inventory |
| InventoryCapacity | NUMBER | - | NOT NULL | Maximum capacity of the inventory |

**Point (Superclass)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/ Field Format | Constraint | Description |
| PointID | VARCHAR2 | 5 | Primary Key | Unique ID for point |
| PointsBalance | NUMBER | 5 | Default = 0 | Current total points available |
| PointStatus | VARCHAR2 | 20 | Check (‘Active’, ‘Expiry’) | Type of points |
| TotalPointsEarned | NUMBER | 5 | Default = 0 | The total point earned by the user from transactions |
| TotalPointsUsed | NUMBER | 5 | Default = 0 | Total points used by the user |
| CustID | NUMBER | 5 | Foreign Key | References to **Customer** table |
| MemID | VARCHAR2 | 5 | Foreign Key | References to **Membership** table |

**PointRecord**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| RecordID | VARCHAR2 | 6 | Primary Key | Unique ID for the point record |
| PointID | VARCHAR2 | 5 | Foreign Key | References to **Point** table |
| PointsChanged | NUMBER | 5 | Not Null | Number of points added (+) or deducted (-) |
| ChangedType | VARCHAR2 | 10 | CHECK  (‘earned’, ‘used’) | Type of changed |
| ChangeDate | DATE | DD/MM/YYYY | Default SYSDATE | Date when the change occurred |

**Active (Subclass – Inherit from Point)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| PointID | VARCHAR2 | 5 | Primary Key, Foreign Key | Inherits from **Point** |
| ActivePointID | VARCHAR2 | 6 | Primary Key | Unique ID for active point record |
| ExpiryDate | DATE | DD/MM/YYYY | Default SYSDATE | Date of the active points will expire |
| IsRedeemable | CHAR | 1 | Check (‘Y’, ‘N’) | Indicate whether points can be redeemed |

**Expiry (Subclass – Inherit from Point)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size/Field Format | Constraint | Description |
| PointID | VARCHAR2 | 5 | Primary Key, Foreign Key | Inherit from **Point** |
| ExpiryPointID | VARCHAR2 | 6 | Primary Key | Unique ID for expired point record |
| DateExpired | DATE | DD/MM/YYYY | Default SYSDATE | Date of the points expired |
| Reason | VARCHAR2 | 50 | NOT NULL | Reason for expiry (“Unused”, “Expired”) |

**Membership**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| MemID | VARCHAR2 | 5 | Primary Key | Unique ID for the Member |
| JoinDate | DATE | DD-MM-YYYY | Default SYSDATE | Date the member joined |
| Status | VARCHAR2 | 10 | Check (‘Active’, ‘Inactive’) | Current status of the membership |
| MemTierID | VARCHAR2 | 3 | Foreign Key | References to **MembershipTier** table |

**MembershipTier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| MemTierID | VARCHAR2 | 3 | Primary Key | Unique ID for the Membership Tier |
| TierName | VARCHAR2 | 10 | Check (‘Bronze’, ‘Silver’, ‘Gold’, ‘Platinum’, ‘Diamond’) | Name of the membership tier |
| PointsRequired | NUMBER | 5 | Default = 0 | Minimum points required to reach this tier |

**MembershipBenefit (Superclass)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| BenefitID | VARCHAR2 | 4 | Primary Key | Unique ID for the Benefit |
| MemTierID | VARCHAR2 | 3 | Foreign Key | References to **MembershipTier** table |
| BenefitName | VARCHAR2 | 50 | Not Null | Name of the specific benefit |
| BenefitDesc | VARCHAR2 | 255 | Not Null | Detailed description of the benefit |
| BenefitType | VARCHAR2 | 10 | Not Null | Type of benefit |

**Promotion** **(Subclass – Inherit from MembershipBenefit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| BenefitID | VARCHAR2 | 4 | Primary Key, Foreign Key | Inherits from **MembershipBenefit** table |
| PromotionID | VARCHAR2 | 5 | Primary Key | Unique ID for the Promotion instance |
| PromoCode | VARCHAR2 | 15 | Not Null | Code used to redeem the promotion |
| DiscountPercentage | NUMBER | 100 | Default = 0 | Percentage discount offered |
| ApplicableProducts | VARCHAR2 | 100 | Not Null | Products/Services the promotion applies to |

**Voucher (Subclass – Inherit from MembershipBenefit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| BenefitID | VARCHAR2 | 4 | Primary Key, Foreign Key | Inherits from **MembershipBenefit** table |
| VoucherID | VARCHAR2 | 5 | Primary Key | Unique ID for the Voucher instance |
| VoucherCode | VARCHAR2 | 15 | Not Null | Code representing the voucher |
| Value | NUMBER | 100 | Default = 0 | Monetary value of the voucher (in MYR) |
| RedeemableAt | VARCHAR2 | 50 | Not Null | Where the voucher can be redeemed |

**FreeGift (Subclass – Inherit from MembershipBenefit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size / Field Format | Constraint | Description |
| BenefitID | VARCHAR2 | 4 | Primary Key, Foreign Key | Inherits from **MembershipBenefit** table |
| GiftID | VARCHAR2 | 5 | Primary Key | Unique ID for the Free Gift instance |
| GiftName | VARCHAR2 | 50 | Not Null | Name of the free gift item |
| EligibilityRule | VARCHAR2 | 100 | Not Null | Conditions to receive the free gift |

# 5.0 Tables and Record

**Customer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CustID | CustName | PhoneNum | CustEmail | MemID |
| 10001 | Aiman Rahman | 0123456789 | aiman@email.com | M0001 |
| 10002 | Sarah Lim | 0198765432 | sarah.lim@gmail.com | M0002 |
| 10003 | Raj Kumar | 0171122233 | raj.kumar@yahoo.com | M0003 |
| 10004 | Lee Wei Ming | 0132233445 | lee.wei@gmail.com | M0004 |
| 10005 | Nurul Izzati | 0145566778 | izzati.nurul@hotmail.com | M0005 |
| 10006 | Ahmad Shafiq | 0189988776 | shafiq.ahmad@gmail.com | M0006 |
| 10007 | Tan Mei Ying | 0112345678 | tanmei@yahoo.com | M0007 |
| 10008 | Kavita Rani | 0167788990 | kavita.rani@gmail.com | M0008 |
| 10009 | Mohd Faizal | 0173344556 | faizal.mohd@live.com | M0009 |
| 10010 | Chloe Tan | 0154433221 | chloe.tan@gmail.com | M0010 |
| 10011 | Koo Zhe Quan | 0211209198 | kooquan@gmail.com | M0011 |
| 10012 | Nur Sabrina | 0126655443 | sabrina\_nur@gmail.com | M0012 |
| 10013 | Lim Jing Yee | 0193322114 | jingyee.lim@mail.com | M0013 |
| 10014 | Zulkifli Hassan | 0183344556 | zulkifli.has@gmail.com | M0014 |
| 10015 | Anita Das | 0115566778 | anita.das@hotmail.com | M0015 |

**Transaction**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TransactionID | TransactionType | PointsEarned | PointsRedeemed | TransactionDate | TransactionStatus | CustID |
| T0001 | Earn | 100 | 0 | 11-04-2025 | Completed | 10001 |
| T0002 | Redeem | 0 | 150 | 11-04-2025 | Completed | 10002 |
| T0003 | Earn | 200 | 0 | 12-04-2025 | Pending | 10003 |
| T0004 | Earn | 150 | 0 | 13-04-2025 | Completed | 10004 |
| T0005 | Redeem | 0 | 120 | 14-04-2025 | Completed | 10005 |
| T0006 | Earn | 300 | 0 | 15-04-2025 | Completed | 10006 |
| T0007 | Redeem | 0 | 100 | 16-04-2025 | Completed | 10007 |
| T0008 | Earn | 180 | 0 | 17-04-2025 | Completed | 10008 |
| T0009 | Earn | 250 | 0 | 17-04-2025 | Pending | 10009 |
| T0010 | Redeem | 0 | 200 | 18-04-2025 | Completed | 10010 |
| T0011 | Earn | 80 | 0 | 25-04-2025 | Completed | 10011 |
| T0012 | Redeem | 0 | 90 | 26-04-2025 | Completed | 10012 |
| T0013 | Redeem | 0 | 100 | 26-04-2025 | Pending | 10013 |
| T0014 | Redeem | 0 | 80 | 26-04-2025 | Completed | 10014 |
| T0015 | Earn | 120 | 0 | 27-04-2025 | Completed | 10015 |
| T0016 | Redeem | 0 | 50 | 27-04-2024 | Pending | 10002 |
| T0017 | Redeem | 0 | 250 | 27-04-2025 | Pending | 10004 |
| T0018 | Earn | 90 | 0 | 28-04-2025 | Failed | 10008 |
| T0019 | Redeem | 0 | 70 | 28-04-2025 | Completed | 10007 |
| T0020 | Redeem | 0 | 120 | 28-04-2025 | Failed | 10009 |
| T0021 | Redeem | 0 | 130 | 29-04-2025 | Completed | 10007 |
| T0022 | Earn | 130 | 0 | 29-04-2025 | Completed | 10008 |
| T0023 | Redeem | 0 | 110 | 29-04-2025 | Completed | 10009 |
| T0024 | Redeem | 0 | 60 | 29-04-2025 | Completed | 10012 |
| T0025 | Redeem | 0 | 160 | 29-04-2025 | Completed | 10012 |
| T0026 | Earn | 100 | 0 | 30-04-2025 | Completed | 10001 |
| T0027 | Earn | 135 | 0 | 30-04-2025 | Completed | 10002 |
| T0028 | Earn | 105 | 0 | 30-04-2025 | Completed | 10003 |
| T0029 | Earn | 160 | 0 | 30-04-2025 | Completed | 10002 |
| T0030 | Earn | 50 | 0 | 30-04-2025 | Completed | 10001 |

**TransactionHistory**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| HistoryID | TransactionType | PointsEarned | PointsRedeemed | TransactionID | CustID |
| H0001 | Earn | 100 | 0 | T0001 | 10001 |
| H0002 | Redeem | 0 | 150 | T0002 | 10002 |
| H0003 | Earn | 150 | 0 | T0004 | 10004 |
| H0004 | Redeem | 0 | 120 | T0005 | 10005 |
| H0005 | Earn | 300 | 0 | T0006 | 10006 |
| H0006 | Redeem | 0 | 100 | T0007 | 10007 |
| H0007 | Earn | 180 | 0 | T0008 | 10008 |
| H0008 | Redeem | 0 | 200 | T0010 | 10010 |
| H0009 | Earn | 80 | 0 | T0011 | 10011 |
| H0010 | Redeem | 0 | 90 | T0012 | 10012 |
| H0011 | Redeem | 0 | 80 | T0014 | 10014 |
| H0012 | Earn | 120 | 0 | T0015 | 10015 |
| H0013 | Redeem | 0 | 70 | T0019 | 10007 |
| H0014 | Redeem | 0 | 130 | T0021 | 10007 |
| H0015 | Earn | 130 | 0 | T0022 | 10008 |
| H0016 | Redeem | 0 | 110 | T0023 | 10009 |
| H0017 | Redeem | 0 | 60 | T0024 | 10012 |
| H0018 | Redeem | 0 | 160 | T0025 | 10012 |
| H0019 | Earn | 100 | 0 | T0026 | 10001 |
| H0020 | Earn | 135 | 0 | T0027 | 10002 |
| H0021 | Earn | 105 | 0 | T0028 | 10003 |
| H0022 | Earn | 160 | 0 | T0029 | 10002 |
| H0023 | Earn | 50 | 0 | T0030 | 10001 |

**Payment (Superclass)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PaymentID | TotalAmount | PaymentStatus | PaymentMethod | TransactionID |
| PAY0001 | 100.00 | Completed | E-Wallet | T0001 |
| PAY0002 | 200.00 | Pending | Card | T0003 |
| PAY0003 | 150.00 | Completed | Online Banking | T0004 |
| PAY0004 | 300.00 | Completed | E-Wallet | T0006 |
| PAY0005 | 180.14 | Completed | Card | T0008 |
| PAY0006 | 250.06 | Pending | Online Banking | T0009 |
| PAY0007 | 80.10 | Completed | E-Wallet | T0011 |
| PAY0008 | 120.65 | Completed | Card | T0015 |
| PAY0009 | 90.33 | Failed | E-Wallet | T0018 |
| PAY0010 | 130.00 | Completed | Online Banking | T0022 |
| PAY0011 | 100.20 | Completed | E-Wallet | T0026 |
| PAY0012 | 135.00 | Completed | Card | T0027 |
| PAY0013 | 105.67 | Completed | Online Banking | T0028 |
| PAY0014 | 160.66 | Completed | Card | T0029 |
| PAY0015 | 50.00 | Completed | Online Banking | T0030 |

**E-Wallet (Subclass - inherits from Payment)**

|  |  |  |  |
| --- | --- | --- | --- |
| PaymentID | Wallet\_ID | Wallet\_Application | Currency\_Type |
| PAY0001 | 912345678891 | GrabPay | MYR |
| PAY0004 | 923456789012 | Touch’n Go | MYR |
| PAY0007 | 934567890123 | PayPal | USD |
| PAY0009 | 945678901234 | Boost | MYR |
| PAY0011 | 956789012345 | Touch’n Go | MYR |

**Card (Subclass – inherits from Payment)**

|  |  |  |  |
| --- | --- | --- | --- |
| PaymentID | Card\_Number | Card\_Holder\_Name | Card\_Type |
| PAY0002 | 4098765432109871 | Sarah Lim | Credit |
| PAY0005 | 4987654321098762 | Nurul Izzati | Debit |
| PAY0008 | 4876543210987653 | Kavita Rani | Credit |
| PAY0012 | 4765432109876544 | Nur Sabrina | Credit |
| PAY0014 | 4654321098765435 | Anita Das | Debit |

**Online Banking (Subclass – inherits from Payment)**

|  |  |  |
| --- | --- | --- |
| PaymentID | Bank\_Acc\_Number | Bank\_Name |
| PAY0003 | 1234567890 | Maybank |
| PAY0006 | 2345678901 | CIMB |
| PAY0010 | 3456789012 | RHB |
| PAY0013 | 4567890123 | Public Bank |
| PAY0015 | 5678901234 | Maybank |

**Redemption**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RedemptionID | CustID | RewardID | RedemptionDate | Status | TransactionID | PointsUsed |
| RD01 | 10002 | R0001 | 11-04-2025 | Completed | T0002 | 150 |
| RD02 | 10004 | R0002 | 13-04-2025 | Completed | T0004 | 120 |
| RD03 | 10007 | R0004 | 16-04-2025 | Completed | T0007 | 100 |
| RD04 | 10009 | R0006 | 18-04-2025 | Completed | T0009 | 200 |
| RD05 | 10012 | R0008 | 26-04-2025 | Completed | T0012 | 90 |
| RD06 | 10013 | R0012 | 26-04-2025 | Pending | T0013 | 100 |
| RD07 | 10014 | R0007 | 26-04-2025 | Completed | T0014 | 80 |
| RD08 | 10002 | R0011 | 27-04-2025 | Pending | T0016 | 50 |
| RD09 | 10004 | R0013 | 27-04-2025 | Pending | T0017 | 250 |
| RD10 | 10007 | R0010 | 28-04-2025 | Completed | T0019 | 70 |
| RD11 | 10009 | R0003 | 28-04-2025 | Failed | T0020 | 120 |
| RD12 | 10007 | R0015 | 28-04-2025 | Completed | T0021 | 130 |
| RD13 | 10009 | R0014 | 29-04-2025 | Completed | T0023 | 110 |
| RD14 | 10012 | R0009 | 29-04-2025 | Completed | T0024 | 60 |
| RD15 | 10012 | R0005 | 29-04-2025 | Completed | T0025 | 160 |

**Reward**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RewardID | RewardName | PointsRequired | RewardType | Stock | CatalogID |
| R0001 | Chicken Combo Meal | 180 | Food | 25 | C1001 |
| R0002 | Coffee Mug | 120 | Merchandise | 10 | C1002 |
| R0003 | Ice Cream Sundae | 90 | Dessert | 20 | C1001 |
| R0004 | Fries Upgrade | 60 | Food Add-on | 40 | C1001 |
| R0005 | Fried Chicken Piece | 70 | Food Add-on | 50 | C1001 |
| R0006 | Cheeseburger Set | 150 | Food | 30 | C1001 |
| R0007 | Tumbler Bottle | 160 | Merchandise | 12 | C1002 |
| R0008 | Plush Toy | 100 | Merchandise | 5 | C1002 |
| R0009 | Free americano | 90 | Beverage | 15 | C1003 |
| R0010 | Free Ice Latte | 100 | Beverage | 20 | C1003 |
| R0011 | Tupperware | 300 | Merchandise | 10 | C1005 |
| R0012 | Kids Meal Set | 140 | Food | 18 | C1005 |
| R0013 | Festive Hamper | 250 | Seasonal | 5 | C1004 |
| R0014 | Free Ice mocha | 120 | Beverage | 20 | C1003 |
| R0015 | Limited Edition Toy | 300 | Collectible | 3 | C1005 |

**RewardCatalog**

|  |  |  |  |
| --- | --- | --- | --- |
| CatalogID | RewardName | Description | IsActive |
| C1001 | Combo Meals | Meal combos with drinks | Active |
| C1002 | Gift Items | Branded merchandise | Inactive |
| C1003 | Refreshments | Cold and hot beverage selections | Active |
| C1004 | Seasonal Specials | Seasonal or limited time offers | Active |
| C1005 | Kids' Rewards | Toys and items for kids | Inactive |

**Restocking**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RestockID | RewardID | InventoryID | RestockingDate | Quantity | SupplierID |
| RS001 | R0001 | I001 | 04-01-2025 | 20 | S001 |
| RS002 | R0006 | 1002 | 16-01-2025 | 20 | S002 |
| RS003 | R0005 | 1003 | 02-02-2025 | 30 | S003 |
| RS004 | R0008 | I004 | 22-02-2025 | 30 | S001 |
| RS005 | R0015 | I005 | 27-02-2025 | 3 | S005 |
| RS006 | R0013 | I006 | 11-03-2025 | 5 | S002 |
| RS007 | R0014 | I003 | 11-03-2025 | 10 | S003 |
| RS008 | R0003 | I004 | 15-03-2025 | 5 | S003 |
| RS009 | R0010 | I007 | 16-03-2025 | 10 | S004 |
| RS010 | R0011 | I001 | 20-03-2025 | 5 | S001 |
| RS011 | R0004 | I002 | 02-04-2025 | 30 | S002 |
| RS012 | R0002 | I002 | 03-04-2025 | 5 | S003 |
| RS013 | R0007 | I003 | 03-04-2025 | 5 | S001 |
| RS014 | R0009 | I006 | 10-04-2025 | 10 | S005 |
| RS015 | R0012 | I002 | 10-04-2025 | 10 | S002 |

**Inventory**

|  |  |  |
| --- | --- | --- |
| InventoryID | InventoryBlock | InventoryCapacity |
| I001 | BA1 | 100 |
| I002 | BB1 | 150 |
| I003 | BA2 | 100 |
| I004 | BA3 | 100 |
| I005 | BB3 | 150 |
| 1006 | BC1 | 50 |
| 1007 | BB2 | 150 |

**Point**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PointID | PointsBalance | PointStatus | TotalPointsEarned | TotalPointsUsed | CustID | MemID |
| P1001 | 1500 | Active | 2500 | 1000 | 10001 | M0001 |
| P1002 | 500 | Active | 1500 | 1000 | 10002 | M0002 |
| P1003 | 0 | Expiry | 500 | 500 | 10003 | M0003 |
| P1004 | 2000 | Active | 3000 | 1000 | 10004 | M0004 |
| P1005 | 0 | Expiry | 1000 | 1000 | 10005 | M0005 |
| P1006 | 750 | Active | 1250 | 500 | 10006 | M0006 |
| P1007 | 0 | Expiry | 750 | 750 | 10007 | M0007 |
| P1008 | 300 | Active | 800 | 500 | 10008 | M0008 |
| P1009 | 0 | Expiry | 300 | 300 | 10009 | M0009 |
| P1010 | 1200 | Active | 1700 | 500 | 10010 | M0010 |
| P1011 | 900 | Active | 1400 | 500 | 10011 | M0011 |
| P1012 | 0 | Expiry | 600 | 600 | 10012 | M0012 |
| P1013 | 1800 | Active | 2300 | 500 | 10013 | M0013 |
| P1014 | 400 | Active | 900 | 500 | 10014 | M0014 |
| P1015 | 0 | Expiry | 400 | 400 | 10015 | M0015 |

**PointRecord**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RecordID | PointID | PointsChanged | ChangedType | ChangeDate |
| PR2001 | P1001 | +1000 | earned | 15/07/2023 |
| PR2002 | P1001 | -500 | used | 10/11/2023 |
| PR2003 | P1001 | +1500 | earned | 20/12/2023 |
| PR2004 | P1002 | +1000 | earned | 03/01/2024 |
| PR2005 | P1002 | -500 | used | 25/04/2024 |
| PR2006 | P1003 | +500 | earned | 09/12/2023 |
| PR2007 | P1003 | -500 | used | 13/12/2023 |
| PR2008 | P1004 | +2000 | earned | 27/08/2023 |
| PR2009 | P1004 | -1000 | used | 10/10/2023 |
| PR2010 | P1005 | +1000 | earned | 04/02/2024 |
| PR2011 | P1005 | -1000 | used | 18/02/2024 |
| PR2012 | P1006 | +750 | earned | 19/03/2023 |
| PR2013 | P1006 | -500 | used | 28/03/2023 |
| PR2014 | P1007 | +750 | earned | 07/06/2023 |
| PR2015 | P1007 | -750 | used | 29/06/2023 |
| PR2016 | P1008 | +500 | earned | 16/01/2024 |
| PR2017 | P1008 | +300 | earned | 18/01/2024 |
| PR2018 | P1008 | -500 | used | 20/01/2024 |
| PR2019 | P1009 | +300 | earned | 05/01/2023 |
| PR2020 | P1009 | -300 | used | 15/01/2023 |
| PR2021 | P1010 | +1200 | earned | 10/02/2024 |
| PR2022 | P1010 | -500 | used | 25/02/2024 |
| PR2023 | P1011 | +900 | earned | 05/03/2024 |
| PR2024 | P1011 | -500 | used | 15/03/2024 |
| PR2025 | P1012 | +600 | earned | 10/01/2023 |
| PR2026 | P1012 | -600 | used | 20/01/2023 |
| PR2027 | P1013 | +1800 | earned | 15/04/2024 |
| PR2028 | P1013 | -500 | used | 25/04/2024 |
| PR2029 | P1014 | +400 | earned | 01/05/2024 |
| PR2030 | P1014 | -500 | used | 10/05/2024 |
| PR2031 | P1015 | +400 | earned | 15/06/2023 |
| PR2032 | P1015 | -400 | used | 25/06/2023 |

**Active (Subclass – Inherit from Point)**

|  |  |  |  |
| --- | --- | --- | --- |
| ActivePointID | PointID | ExpiryDate | IsRedeemable |
| ACT301 | P1001 | 31/12/2024 | Y |
| ACT302 | P1002 | 25/10/2024 | Y |
| ACT303 | P1004 | 10/04/2024 | Y |
| ACT304 | P1006 | 28/09/2023 | Y |
| ACT305 | P1008 | 20/07/2024 | Y |
| ACT306 | P1010 | 25/08/2024 | Y |
| ACT307 | P1011 | 15/09/2024 | Y |
| ACT308 | P1013 | 25/10/2024 | Y |
| ACT309 | P1014 | 10/11/2024 | Y |

**Expiry (Subclass – Inherit from Point)**

|  |  |  |  |
| --- | --- | --- | --- |
| ExpiryPointID | PointID | DateExpired | Reason |
| EXP201 | P1003 | 13/12/2023 | expired |
| EXP202 | P1005 | 18/02/2024 | expired |
| EXP203 | P1007 | 29/06/2023 | expired |
| EXP204 | P1009 | 15/01/2023 | expired |
| EXP205 | P1012 | 20/01/2023 | expired |
| EXP206 | P1015 | 25/06/2023 | expired |

**Membership**

|  |  |  |  |
| --- | --- | --- | --- |
| MemID | JoinDate | Status | MemTierID |
| M0001 | 07-07-2021 | Active | L03 |
| M0002 | 11-04-2022 | Inactive | L02 |
| M0003 | 18-09-2022 | Inactive | L01 |
| M0004 | 01-11-2022 | Active | L04 |
| M0005 | 25-12-2022 | Inactive | L02 |
| M0006 | 15-01-2023 | Active | L02 |
| M0007 | 25-03-2023 | Active | L01 |
| M0008 | 30-06-2023 | Active | L01 |
| M0009 | 01-08-2023 | Active | L01 |
| M0010 | 05-10-2023 | Active | L02 |
| M0011 | 22-01-2024 | Inactive | L02 |
| M0012 | 14-02-2024 | Active | L01 |
| M0013 | 20-05-2024 | Active | L03 |
| M0014 | 10-08-2024 | Active | L01 |
| M0015 | 01-12-2024 | Active | L01 |

**MembershipTier**

|  |  |  |
| --- | --- | --- |
| MemTierID | TierName | PointsRequired |
| L01 | Bronze | 0 |
| L02 | Silver | 1,000 |
| L03 | Gold | 2,000 |
| L04 | Platinum | 3,000 |
| L05 | Diamond | 5,000 |

**MembershipBenefit**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BenefitID | MemTierID | BenefitName | BenefitDesc | BenefitType |
| B101 | L01 | 5% App Discount | 5% discount on first App Order. | Promotion |
| B102 | L01 | 8% Monday Discount | 8% discount on Mondays (Dine-in only). | Promotion |
| B103 | L02 | 10% Lunch Discount | 10% discount on Weekday Lunch Sets. | Promotion |
| B104 | L02 | 10% Takeaway Discount | 10% discount on Takeaway Orders (Min spend RM25). | Promotion |
| B105 | L02 | 12% Combo Discount | 12% discount on Combo Meals. | Promotion |
| B106 | L03 | 15% Weekend Discount | 15% discount on Weekends (Dine-in, Min spend RM60). | Promotion |
| B107 | L03 | 20% Dine-in Discount | 20% discount on Total Dine-in Bill (Min spend RM80). | Promotion |
| B108 | L03 | 20% Family Set Discount | 20% discount on Family Set Menu. | Promotion |
| B109 | L04 | 25% Birthday Discount | 25% discount on Entire Bill during Birthday Month (Unlimited use). | Promotion |
| B110 | L04 | 15% Platinum Discount | Flat 15% discount on All Orders, always. | Promotion |
| B111 | L05 | 30% Diamond Discount | Flat 30% discount on all orders for Diamond members. | Promotion |
| B112 | L05 | 50% Anniversary Discount | 50% off entire bill during anniversary month (once a year). | Promotion |
| B201 | L01 | RM5 Drink Voucher | RM5 Off Any Drink voucher. | Voucher |
| B202 | L01 | RM5 Cashback Voucher | RM5 Cashback voucher for next visit (Min spend RM30). | Voucher |
| B203 | L01 | RM10 Dine-in Voucher | RM10 Off Dine-in voucher (Min spend RM40). | Voucher |
| B204 | L02 | RM15 In-Store Voucher | RM15 Off In-Store voucher (Min spend RM50). | Voucher |
| B205 | L02 | RM10 Delivery Voucher | RM10 Off Delivery Fee voucher. | Voucher |
| B206 | L02 | RM15 Birthday Voucher | RM15 Birthday Voucher (Redeemable on cakes/desserts). (Birthday month). | Voucher |
| B207 | L03 | RM20 Online Voucher | RM20 Off Online Order voucher (Min spend RM70). | Voucher |
| B208 | L03 | RM30 Birthday Voucher | RM30 Birthday Voucher (Any item, Dine-in). (Birthday month). | Voucher |
| B209 | L03 | RM25 Dine-in Voucher | RM25 Off Dine-in voucher (Min spend RM100). | Voucher |
| B210 | L04 | RM50 VIP Voucher | RM50 Voucher (No min spend, any use). (Issued quarterly). | Voucher |
| B211 | L05 | RM100 Diamond Voucher | RM100 voucher with no minimum spend (issued bi-annually). | Voucher |
| B212 | L05 | RM75 Dining Voucher | RM75 off dine-in voucher (Min spend RM150). | Voucher |
| B301 | L01 | Hot Drink Upgrade | Complimentary Coffee/Tea upsize. (Any dine-in visit) | Free Gift |
| B302 | L01 | Keychain Souvenir | Collectable Bronze Tier keychain. (Awarded after 3 visits) | Free Gift |
| B303 | L02 | Free Dessert | Choice of standard dessert with any main course purchase. | Free Gift |
| B304 | L02 | Birthday Cake Slice | Complimentary slice of cake during birthday week. | Free Gift |
| B305 | L02 | Loyalty Cup | Silver Tier branded reusable cup. (Welcome gift upon reaching Silver) | Free Gift |
| B306 | L03 | Signature Drink | Complimentary Signature Drink per visit. | Free Gift |
| B307 | L03 | Exclusive Mug | Gold Tier Member exclusive mug. (Welcome gift upon reaching Gold) | Free Gift |
| B308 | L04 | Free Kids Meal | Complimentary Kids Meal with any adult main course purchase. | Free Gift |
| B309 | L04 | VIP Dining Badge | Personalised Platinum Tier badge. | Free Gift |
| B310 | L04 | Priority Booking | Guaranteed table/priority seating reservations. | Free Gift |
| B311 | L05 | Complimentary Full Meal | One complimentary full meal set per quarter. | Free Gift |
| B312 | L05 | Premium Welcome Kit | Exclusive Diamond tier welcome kit upon reaching the tier. | Free Gift |

**Promotion (Subclass – inherits from MembershipBenefit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BenefitID | PromotionID | PromoCode | DiscountPercentage | ApplicableProducts |
| B101 | PN001 | APP5 | 5 | First app order |
| B102 | PN002 | MONDAY8 | 8 | All Monday dine-in orders |
| B103 | PN003 | Lunch10 | 10 | Weekday lunch |
| B104 | PN004 | TAKEOUT10 | 10 | Takeaway Orders |
| B105 | PN005 | COMBO12 | 12 | Combo meals |
| B106 | PN006 | WEEKEND15 | 15 | Weekend dine-in order only |
| B107 | PN007 | DINE20 | 20 | Dine-in only |
| B108 | PN008 | FAMILY20 | 20 | Family set only |
| B109 | PN009 | BDAY25 | 25 | Birthday month only |
| B110 | PN010 | PLATINUM15 | 15 | All orders |
| B111 | PN011 | DIAMOND30 | 30 | All orders |
| B112 | PN012 | ANNIVERSARY50 | 50 | Entire bill |

**Voucher (Subclass – inherits from MembershipBenefit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BenefitID | VoucherID | VoucherCode | Value | RedeemableAt |
| B201 | VR001 | DRINKONME | 5 | Dine-in |
| B202 | VR002 | CASHBACK5 | 5 | Online delivery |
| B203 | VR003 | TREAT10 | 10 | Dine-in |
| B204 | VR004 | VOUCHER15 | 15 | Dine-in |
| B205 | VR005 | DELIVERY10 | 10 | Online delivery |
| B206 | VR006 | BDAYCAKE | 15 | Any |
| B207 | VR007 | ONLINE20 | 20 | Online delivery |
| B208 | VR008 | VCH30BIRTH | 30 | Dine-in |
| B209 | VR009 | DINE25OFF | 25 | Dine-in |
| B210 | VR010 | VIPRM50 | 50 | Any |
| B211 | VR011 | DIAMOND100 | 100 | Any |
| B212 | VR012 | DINE75OFF | 75 | Dine-in |

**FreeGift (Subclass – inherits from MembershipBenefit)**

|  |  |  |  |
| --- | --- | --- | --- |
| BenefitID | GiftID | GiftName | EligibilityRule |
| B301 | GF001 | Hot Drink Upgrade | Any dine-in visit |
| B302 | GF002 | Keychain Souvenir | Awarded after 3 visits |
| B303 | GF003 | Free Dessert | With purchase of any main course |
| B304 | GF004 | Birthday Cake Slice | During birthday week |
| B305 | GF005 | Loyalty Cup | Welcome gift upon reaching Silver |
| B306 | GF006 | Signature Drink | Complementary signature drink per visit |
| B307 | GF007 | Exclusive Mug | Welcome gift upon reaching Gold |
| B308 | GF008 | Free Kids Meal | With purchase any main course |
| B309 | GF009 | VIP Dining Badge | Personalised platinum tier badge |
| B310 | GF010 | Priority Booking | Priority seating reservation |
| B311 | GF011 | Complimentary Full Meal | One per quarter |
| B312 | GF012 | Premium Welcome Kit | Welcome gift upon reaching Diamond tier |

# 6.0 Script

/\*

-- Group Information

COURSE CODE: UCCD2303

PROGRAMME: CS

GROUP NUMBER: G37

GROUP LEADER NAME AND EMAIL: Owi Wilson, owiwilson123@1utar.my

MEMBER 2 NAME: Ng Wei Jie

MEMBER 3 NAME: Tan Cheng Lock

MEMBER 4 NAME: Yee Hao Zhe

Submission date and time: 29 Apr 2025, 5 pm

\*/

--drop table in case there is existing table inside the sql

DROP TABLE MembershipTier CASCADE CONSTRAINTS;

DROP TABLE MembershipBenefit CASCADE CONSTRAINTS;

DROP TABLE Promotion CASCADE CONSTRAINTS;

DROP TABLE Voucher CASCADE CONSTRAINTS;

DROP TABLE FreeGift CASCADE CONSTRAINTS;

DROP TABLE RewardCatalog CASCADE CONSTRAINTS;

DROP TABLE Reward CASCADE CONSTRAINTS;

DROP TABLE Restocking CASCADE CONSTRAINTS;

DROP TABLE Inventory CASCADE CONSTRAINTS;

DROP TABLE Customer CASCADE CONSTRAINTS;

DROP TABLE Membership CASCADE CONSTRAINTS;

DROP TABLE Point CASCADE CONSTRAINTS;

DROP TABLE PointRecord CASCADE CONSTRAINTS;

DROP TABLE Active CASCADE CONSTRAINTS;

DROP TABLE Expiry CASCADE CONSTRAINTS;

DROP TABLE Transaction CASCADE CONSTRAINTS;

DROP TABLE TransactionHistory CASCADE CONSTRAINTS;

DROP TABLE Payment CASCADE CONSTRAINTS;

DROP TABLE Ewallet CASCADE CONSTRAINTS;

DROP TABLE Card CASCADE CONSTRAINTS;

DROP TABLE OnlineBanking CASCADE CONSTRAINTS;

DROP TABLE Redemption CASCADE CONSTRAINTS;

--create table for database

CREATE TABLE MembershipTier

(MemTierID VARCHAR2(3) PRIMARY KEY,

TierName VARCHAR2(10) CHECK (TierName IN ('Bronze', 'Silver', 'Gold', 'Platinum', 'Diamond')),

PointsRequired NUMBER(5) DEFAULT 0);

CREATE TABLE Membership

(MemID VARCHAR2(5) PRIMARY KEY,

JoinDate DATE DEFAULT SYSDATE,

Status VARCHAR2(10) CHECK (Status IN ('Active', 'Inactive')),

MemTierID VARCHAR2(3),

CONSTRAINT Membership\_MemTierID\_fk FOREIGN KEY (MemTierID) REFERENCES MembershipTier(MemTierID));

CREATE TABLE Customer

(CustID NUMBER(5) PRIMARY KEY,

CustName VARCHAR2(20) NOT NULL,

PhoneNum VARCHAR2(11) NOT NULL,

CustEmail VARCHAR2(30) UNIQUE NOT NULL,

MemID VARCHAR2(5),

CONSTRAINT Customer\_MemID\_fk FOREIGN KEY (MemID) REFERENCES Membership(MemID));

CREATE TABLE Point

(PointID VARCHAR2(5) PRIMARY KEY,

PointsBalance NUMBER(5) DEFAULT 0,

PointStatus VARCHAR2(20) CHECK (PointStatus IN ('Active', 'Expiry')),

TotalPointsEarned NUMBER(5) DEFAULT 0,

TotalPointsUsed NUMBER(5) DEFAULT 0,

CustID NUMBER(5),

MemID VARCHAR(5),

CONSTRAINT Point\_CustID\_fk FOREIGN KEY (CustID) REFERENCES Customer(CustID),

CONSTRAINT Point\_MemID\_fk FOREIGN KEY (MemID) REFERENCES Membership(MemID));

CREATE TABLE MembershipBenefit

(BenefitID VARCHAR2(4) PRIMARY KEY,

MemTierID VARCHAR2(3),

BenefitName VARCHAR2(50) NOT NULL,

BenefitDesc VARCHAR2(255) NOT NULL,

BenefitType VARCHAR2(10) NOT NULL,

CONSTRAINT MembershipBenefit\_MemTierID\_fk FOREIGN KEY (MemTierID) REFERENCES MembershipTier(MemTierID));

CREATE TABLE Promotion

(BenefitID VARCHAR2(4),

PromotionID VARCHAR2(5) PRIMARY KEY,

PromoCode VARCHAR2(15) NOT NULL,

DiscountPercentage NUMBER(3) DEFAULT 0,

ApplicableProducts VARCHAR2(100) NOT NULL,

CONSTRAINT Promotion\_BenefitID\_fk FOREIGN KEY (BenefitID) REFERENCES MembershipBenefit(BenefitID));

CREATE TABLE Voucher

(BenefitID VARCHAR2(4),

VoucherID VARCHAR2(5) PRIMARY KEY,

VoucherCode VARCHAR2(15) NOT NULL,

Value NUMBER(5, 2) DEFAULT 0,

RedeemableAt VARCHAR2(50) NOT NULL,

CONSTRAINT Voucher\_BenefitID\_fk FOREIGN KEY (BenefitID) REFERENCES MembershipBenefit(BenefitID));

CREATE TABLE FreeGift

(BenefitID VARCHAR2(4),

GiftID VARCHAR2(5) PRIMARY KEY,

GiftName VARCHAR2(50) NOT NULL,

EligibilityRule VARCHAR2(100) NOT NULL,

CONSTRAINT FreeGift\_BenefitID\_fk FOREIGN KEY (BenefitID) REFERENCES MembershipBenefit(BenefitID));

CREATE TABLE RewardCatalog

(CatalogID VARCHAR2(5) PRIMARY KEY,

RewardName VARCHAR2(20) NOT NULL,

Description VARCHAR2(100),

IsActive VARCHAR2(10) CHECK (IsActive IN ('Active', 'Inactive')));

CREATE TABLE Reward

(RewardID VARCHAR2(5) PRIMARY KEY,

RewardName VARCHAR2(30) NOT NULL,

PointsRequired NUMBER(5) NOT NULL,

RewardType VARCHAR2(20),

Stock NUMBER(3) DEFAULT 0 NOT NULL,

CatalogID VARCHAR2(5),

CONSTRAINT Reward\_CatalogID\_fk FOREIGN KEY (CatalogID) REFERENCES RewardCatalog(CatalogID));

CREATE TABLE Inventory

(InventoryID VARCHAR2(4) PRIMARY KEY,

InventoryBlock VARCHAR2(3),

InventoryCapacity NUMBER NOT NULL);

CREATE TABLE Restocking

(RestockID VARCHAR2(5) PRIMARY KEY,

RewardID VARCHAR2(5),

InventoryID VARCHAR2(4),

RestockingDate DATE DEFAULT SYSDATE,

Quantity NUMBER,

SupplierID VARCHAR2(5) NOT NULL,

CONSTRAINT Restocking\_RewardID\_fk FOREIGN KEY (RewardID) REFERENCES Reward(RewardID),

CONSTRAINT Restocking\_InventoryID\_fk FOREIGN KEY (InventoryID) REFERENCES Inventory(InventoryID));

CREATE TABLE PointRecord

(RecordID VARCHAR2(6) PRIMARY KEY,

PointID VARCHAR2(5),

PointsChanged NUMBER(5) NOT NULL,

ChangedType VARCHAR2(10) CHECK(ChangedType IN ('earned', 'used')),

ChangeDate DATE DEFAULT SYSDATE,

CONSTRAINT PointRecord\_PointID\_fk FOREIGN KEY (PointID) REFERENCES Point(PointID));

CREATE TABLE Active

(ActivePointID VARCHAR2(6) PRIMARY KEY,

PointID VARCHAR2(5),

ExpiryDate DATE DEFAULT SYSDATE,

IsRedeemable CHAR(1) CHECK (IsRedeemable IN ('Y', 'N')),

CONSTRAINT Active\_PointID\_fk FOREIGN KEY (PointID) REFERENCES Point(PointID));

CREATE TABLE Expiry

(ExpiryPointID VARCHAR2(6) PRIMARY KEY,

PointID VARCHAR2(5),

DateExpired DATE DEFAULT SYSDATE,

Reason VARCHAR2(50) NOT NULL,

CONSTRAINT Expiry\_PointID\_fk FOREIGN KEY (PointID) REFERENCES Point(PointID));

CREATE TABLE Transaction

(TransactionID VARCHAR2(5) PRIMARY KEY,

TransactionType VARCHAR2(10) CHECK (TransactionType IN ('Redeem', 'Earn')),

PointsEarned NUMBER(8) DEFAULT 0,

PointsRedeemed NUMBER(8) DEFAULT 0,

TransactionDate DATE DEFAULT SYSDATE,

TransactionStatus VARCHAR2(10) CHECK (TransactionStatus IN ('Pending', 'Failed', 'Completed')),

CustID NUMBER(5),

CONSTRAINT Transaction\_CustID\_fk FOREIGN KEY (CustID) REFERENCES Customer(CustID));

CREATE TABLE TransactionHistory

(HistoryID VARCHAR2(5) PRIMARY KEY,

TransactionType VARCHAR2(10) CHECK (TransactionType IN ('Redeem', 'Earn')),

PointsEarned NUMBER(8) DEFAULT 0,

PointsRedeemed NUMBER(8) DEFAULT 0,

TransactionID VARCHAR2(5),

CustID NUMBER(5),

CONSTRAINT TransactionHistory\_TransactionID\_fk FOREIGN KEY (TransactionID) REFERENCES Transaction(TransactionID),

CONSTRAINT TransactionHistory\_CustID\_fk FOREIGN KEY (CustID) REFERENCES Customer(CustID));

CREATE TABLE Payment

(PaymentID VARCHAR2(8) PRIMARY KEY,

TotalAmount NUMBER(10, 2) NOT NULL,

PaymentStatus VARCHAR2(10) CHECK (PaymentStatus IN ('Pending', 'Completed', 'Failed')),

PaymentMethod VARCHAR2(20) CHECK (PaymentMethod IN ('E-Wallet', 'Card', 'Online Banking')),

TransactionID VARCHAR2(5),

CONSTRAINT Payment\_TransactionID\_fk FOREIGN KEY (TransactionID) REFERENCES Transaction(TransactionID));

CREATE TABLE Ewallet

(PaymentID VARCHAR2(8),

Wallet\_ID NUMBER(15) PRIMARY KEY,

Wallet\_Application VARCHAR2(10) NOT NULL,

Currency\_Type VARCHAR2(10) NOT NULL,

CONSTRAINT Ewallet\_PaymentID\_fk FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID));

CREATE TABLE Card

(PaymentID VARCHAR2(8),

Card\_Number NUMBER(20) PRIMARY KEY,

Card\_Holder\_Name VARCHAR2(50) NOT NULL,

Card\_Type VARCHAR2(10) NOT NULL,

CONSTRAINT Card\_PaymentID\_fk FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID));

CREATE TABLE OnlineBanking

(PaymentID VARCHAR2(8),

Bank\_Acc\_Number NUMBER(10) PRIMARY KEY,

Bank\_Name VARCHAR2(20) NOT NULL,

CONSTRAINT OnlineBanking\_PaymentID\_fk FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID));

CREATE TABLE Redemption

(RedemptionID VARCHAR2(5) PRIMARY KEY,

CustID NUMBER(5),

RewardID VARCHAR2(5),

RedemptionDate DATE DEFAULT SYSDATE,

Status VARCHAR2(15) CHECK (Status IN ('Pending', 'Completed', 'Failed')),

TransactionID VARCHAR2(5),

PointsUsed NUMBER(5) NOT NULL,

CONSTRAINT Redemption\_CustID\_fk FOREIGN KEY (CustID) REFERENCES Customer(CustID),

CONSTRAINT Redemption\_RewardID\_fk FOREIGN KEY (RewardID) REFERENCES Reward(RewardID),

CONSTRAINT Redemption\_TransactionID\_fk FOREIGN KEY (TransactionID) REFERENCES Transaction(TransactionID));

--insert data for each table

INSERT INTO MembershipTier VALUES ('L01', 'Bronze', 0);

INSERT INTO MembershipTier VALUES ('L02', 'Silver', 1000);

INSERT INTO MembershipTier VALUES ('L03', 'Gold', 2000);

INSERT INTO MembershipTier VALUES ('L04', 'Platinum', 3000);

INSERT INTO MembershipTier VALUES ('L05', 'Diamond', 5000);

INSERT INTO Membership VALUES ('M0001', TO\_DATE('07-07-2021', 'DD-MM-YYYY'), 'Active', 'L03');

INSERT INTO Membership VALUES ('M0002', TO\_DATE('11-04-2022', 'DD-MM-YYYY'), 'Inactive', 'L02');

INSERT INTO Membership VALUES ('M0003', TO\_DATE('18-09-2022', 'DD-MM-YYYY'), 'Inactive', 'L01');

INSERT INTO Membership VALUES ('M0004', TO\_DATE('01-11-2022', 'DD-MM-YYYY'), 'Active', 'L04');

INSERT INTO Membership VALUES ('M0005', TO\_DATE('25-12-2022', 'DD-MM-YYYY'), 'Inactive', 'L02');

INSERT INTO Membership VALUES ('M0006', TO\_DATE('15-01-2023', 'DD-MM-YYYY'), 'Active', 'L02');

INSERT INTO Membership VALUES ('M0007', TO\_DATE('25-03-2023', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Membership VALUES ('M0008', TO\_DATE('30-06-2023', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Membership VALUES ('M0009', TO\_DATE('01-08-2023', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Membership VALUES ('M0010', TO\_DATE('05-10-2023', 'DD-MM-YYYY'), 'Active', 'L02');

INSERT INTO Membership VALUES ('M0011', TO\_DATE('22-01-2024', 'DD-MM-YYYY'), 'Inactive', 'L02');

INSERT INTO Membership VALUES ('M0012', TO\_DATE('14-02-2024', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Membership VALUES ('M0013', TO\_DATE('20-05-2024', 'DD-MM-YYYY'), 'Active', 'L03');

INSERT INTO Membership VALUES ('M0014', TO\_DATE('10-08-2024', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Membership VALUES ('M0015', TO\_DATE('01-12-2024', 'DD-MM-YYYY'), 'Active', 'L01');

INSERT INTO Customer VALUES (10001, 'Ali Bin Ahmad', '01234567890', 'ali.ahmad@gmail.com', 'M0001');

INSERT INTO Customer VALUES (10002, 'Tan Mei Ling', '01987654321', 'tan.mei@outlook.com','M0002');

INSERT INTO Customer VALUES (10003, 'Ravi Kumar', '01311223344', 'ravi.kumar@yahoo.com', 'M0003');

INSERT INTO Customer VALUES (10004, 'Aisha Zainal', '01455667788', 'aisha.zainal@gmail.com', 'M0004');

INSERT INTO Customer VALUES (10005, 'Lim Kok Leong', '01233445566', 'kokleong.lim@mail.com', 'M0005');

INSERT INTO Customer VALUES (10006, 'Siti Nurhaliza', '01199887766', 'siti.nur@gmail.com', 'M0006');

INSERT INTO Customer VALUES (10007, 'James Wong', '01766778899', 'james.wong@live.com', 'M0007');

INSERT INTO Customer VALUES (10008, 'Aminah Hussein', '01811223344', 'aminah.hussein@ymail.com', 'M0008');

INSERT INTO Customer VALUES (10009, 'Chong Wei', '01055667788', 'chong.wei@gmail.com', 'M0009');

INSERT INTO Customer VALUES (10010, 'Maria Tan', '01912345678', 'maria.tan@hotmail.com', 'M0010');

INSERT INTO Customer VALUES (10011, 'Koo Zhe Quan', '0211209198', 'kooquan@gmail.com', 'M0011');

INSERT INTO Customer VALUES (10012, 'Nur Sabrina', '01266554433', 'sabrina\_nur@outlook.com', 'M0012');

INSERT INTO Customer VALUES (10013, 'Lim Jing Yee', '01933221144', 'jingyee.lim@mail.com', 'M0013');

INSERT INTO Customer VALUES (10014, 'Zulkifli Hassan', '01833445566', 'zulkifli.has@gmail.com', 'M0014');

INSERT INTO Customer VALUES (10015, 'Anita Das', '01155667788', 'anita.das@hotmail.com', 'M0015');

INSERT INTO Point VALUES ('P1001', 1500, 'Active', 2500, 1000, 10001, 'M0001');

INSERT INTO Point VALUES ('P1002', 500, 'Active', 1500, 1000, 10002, 'M0002');

INSERT INTO Point VALUES ('P1003', 0, 'Expiry', 500, 500, 10003, 'M0003');

INSERT INTO Point VALUES ('P1004', 2000, 'Active', 3000, 1000, 10004, 'M0004');

INSERT INTO Point VALUES ('P1005', 0, 'Expiry', 1000, 1000, 10005, 'M0005');

INSERT INTO Point VALUES ('P1006', 750, 'Active', 1250, 500, 10006, 'M0006');

INSERT INTO Point VALUES ('P1007', 0, 'Expiry', 750, 750, 10007, 'M0007');

INSERT INTO Point VALUES ('P1008', 300, 'Active', 800, 500, 10008, 'M0008');

INSERT INTO Point VALUES ('P1009', 0, 'Expiry', 300, 300, 10009, 'M0009');

INSERT INTO Point VALUES ('P1010', 1200, 'Active', 1700, 500, 10010, 'M0010');

INSERT INTO Point VALUES ('P1011', 900, 'Active', 1400, 500, 10011, 'M0011');

INSERT INTO Point VALUES ('P1012', 0, 'Expiry', 600, 600, 10012, 'M0012');

INSERT INTO Point VALUES ('P1013', 1800, 'Active', 2300, 500, 10013, 'M0013');

INSERT INTO Point VALUES ('P1014', 400, 'Active', 900, 500, 10014, 'M0014');

INSERT INTO Point VALUES ('P1015', 0, 'Expiry', 400, 400, 10015, 'M0015');

INSERT INTO MembershipBenefit VALUES ('B101', 'L01', '5% App Discount', '5% discount on first App Order.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B102', 'L01', '8% Monday Discount', '8% discount on Mondays (Dine-in only).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B103', 'L02', '10% Lunch Discount', '10% discount on Weekday Lunch Sets.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B104', 'L02', '10% Takeaway Discount', '10% discount on Takeaway Orders (Min spend RM25).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B105', 'L02', '12% Combo Discount', '12% discount on Combo Meals.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B106', 'L03', '15% Weekend Discount', '15% discount on Weekends (Dine-in, Min spend RM60).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B107', 'L03', '20% Dine-in Discount', '20% discount on Total Dine-in Bill (Min spend RM80).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B108', 'L03', '20% Family Set Discount', '20% discount on Family Set Menu.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B109', 'L04', '25% Birthday Discount', '25% discount on Entire Bill during Birthday Month (Unlimited use).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B110', 'L04', '15% Platinum Discount', 'Flat 15% discount on All Orders, always.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B111', 'L05', '30% Diamond Discount', 'Flat 30% discount on all orders for Diamond members.', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B112', 'L05', '50% Anniversary Discount', '50% off entire bill during anniversary month (once a year).', 'Promotion');

INSERT INTO MembershipBenefit VALUES ('B201', 'L01', 'RM5 Drink Voucher', 'RM5 Off Any Drink voucher.', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B202', 'L01', 'RM5 Cashback Voucher', 'RM5 Cashback voucher for next visit (Min spend RM30).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B203', 'L01', 'RM10 Dine-in Voucher', 'RM10 Off Dine-in voucher (Min spend RM40).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B204', 'L02', 'RM15 In-Store Voucher', 'RM15 Off In-Store voucher (Min spend RM50).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B205', 'L02', 'RM10 Delivery Voucher', 'RM10 Off Delivery Fee voucher.', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B206', 'L02', 'RM15 Birthday Voucher', 'RM15 Birthday Voucher (Redeemable on cakes/desserts). (Birthday month).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B207', 'L03', 'RM20 Online Voucher', 'RM20 Off Online Order voucher (Min spend RM70).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B208', 'L03', 'RM30 Birthday Voucher', 'RM30 Birthday Voucher (Any item, Dine-in). (Birthday month).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B209', 'L03', 'RM25 Dine-in Voucher', 'RM25 Off Dine-in voucher (Min spend RM100).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B210', 'L04', 'RM50 VIP Voucher', 'RM50 Voucher (No min spend, any use). (Issued quarterly).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B211', 'L05', ' RM100 Diamond Voucher', ' RM100 voucher with no minimum spend (issued bi-annually).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B212', 'L05', 'RM75 Dining Voucher', 'RM75 off dine-in voucher (Min spend RM150).', 'Voucher');

INSERT INTO MembershipBenefit VALUES ('B301', 'L01', 'Hot Drink Upgrade', 'Complimentary Coffee/Tea upsize. (Any dine-in visit)', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B302', 'L01', 'Keychain Souvenir', 'Collectable Bronze Tier keychain. (Awarded after 3 visits)', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B303', 'L02', 'Free Dessert', 'Choice of standard dessert with any main course purchase.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B304', 'L02', 'Birthday Cake Slice', 'Complimentary slice of cake during birthday week.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B305', 'L02', 'Loyalty Cup', 'Silver Tier branded reusable cup. (Welcome gift upon reaching Silver)', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B306', 'L03', 'Signature Drink', 'Complimentary Signature Drink per visit.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B307', 'L03', 'Exclusive Mug', 'Gold Tier Member exclusive mug. (Welcome gift upon reaching Gold)', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B308', 'L04', 'Free Kids Meal', 'Complimentary Kids Meal with any adult main course purchase.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B309', 'L04', 'VIP Dining Badge', 'Personalised Platinum Tier badge.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B310', 'L04', 'Priority Booking', 'Guaranteed table/priority seating reservations.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B311', 'L05', 'Complimentary Full Meal', 'One complimentary full meal set per quarter.', 'Free Gift');

INSERT INTO MembershipBenefit VALUES ('B312', 'L05', 'Premium Welcome Kit', 'Exclusive Diamond tier welcome kit upon reaching the tier.', 'Free Gift');

INSERT INTO Promotion VALUES ('B101', 'PN001', 'APP5', 5, 'First app order');

INSERT INTO Promotion VALUES ('B102', 'PN002', 'MONDAY8', 8, 'All Monday dine-in orders');

INSERT INTO Promotion VALUES ('B103', 'PN003', 'Lunch10', 10, 'Weekday lunch');

INSERT INTO Promotion VALUES ('B104', 'PN004', 'TAKEOUT10', 10, 'Takeaway Orders');

INSERT INTO Promotion VALUES ('B105', 'PN005', 'COMBO12', 12, 'Combo meals');

INSERT INTO Promotion VALUES ('B106', 'PN006', 'WEEKEND15', 15, 'Weekend dine-in order only');

INSERT INTO Promotion VALUES ('B107', 'PN007', 'DINE20', 20, 'Dine-in only');

INSERT INTO Promotion VALUES ('B108', 'PN008', 'FAMILY20', 20, 'Family set only');

INSERT INTO Promotion VALUES ('B109', 'PN009', 'BDAY25', 25, 'Birthday month only');

INSERT INTO Promotion VALUES ('B110', 'PN010', 'PLATINUM15', 15, 'All orders');

INSERT INTO Promotion VALUES ('B111', 'PN011', 'DIAMOND30', 30, 'All orders');

INSERT INTO Promotion VALUES ('B112', 'PN012', 'ANNIVERSARY50', 50, 'Entire bill');

INSERT INTO Voucher VALUES ('B201', 'VR001', 'DRINKONME', 5, 'Dine-in');

INSERT INTO Voucher VALUES ('B202', 'VR002', 'CASHBACK5', 5, 'Online delivery');

INSERT INTO Voucher VALUES ('B203', 'VR003', 'TREAT10', 10, 'Dine-in');

INSERT INTO Voucher VALUES ('B204', 'VR004', 'VOUCHER15', 15, 'Dine-in');

INSERT INTO Voucher VALUES ('B205', 'VR005', 'DELIVERY10', 10, 'Online delivery');

INSERT INTO Voucher VALUES ('B206', 'VR006', 'BDAYCAKE', 15, 'Any');

INSERT INTO Voucher VALUES ('B207', 'VR007', 'ONLINE20', 20, 'Online delivery');

INSERT INTO Voucher VALUES ('B208', 'VR008', 'VCH30BIRTH', 30, 'Dine-in');

INSERT INTO Voucher VALUES ('B209', 'VR009', 'DINE25OFF', 25, 'Dine-in');

INSERT INTO Voucher VALUES ('B210', 'VR010', 'VIPRM50', 50, 'Any');

INSERT INTO Voucher VALUES ('B211', 'VR011', 'DIAMOND100', 100, 'Any');

INSERT INTO Voucher VALUES ('B212', 'VR012', 'DINE75OFF', 75, 'Dine-in');

INSERT INTO FreeGift VALUES ('B301', 'GF001', 'Hot Drink Upgrade', 'Any dine-in visit');

INSERT INTO FreeGift VALUES ('B302', 'GF002', 'Keychain Souvenir', 'Awarded after 3 visits');

INSERT INTO FreeGift VALUES ('B303', 'GF003', 'Free Dessert', 'With purchase of any main course');

INSERT INTO FreeGift VALUES ('B304', 'GF004', 'Birthday Cake Slice', 'During birthday week');

INSERT INTO FreeGift VALUES ('B305', 'GF005', 'Loyalty Cup', 'Welcome gift upon reaching Silver');

INSERT INTO FreeGift VALUES ('B306', 'GF006', 'Signature Drink', 'Complementary signature drink per visit');

INSERT INTO FreeGift VALUES ('B307', 'GF007', 'Exclusive Mug', 'Welcome gift upon reaching Gold');

INSERT INTO FreeGift VALUES ('B308', 'GF008', 'Free Kids Meal', 'With purchase any main course');

INSERT INTO FreeGift VALUES ('B309', 'GF009', 'VIP Dining Badge', 'Personalised platinum tier badge');

INSERT INTO FreeGift VALUES ('B310', 'GF010', 'Priority Booking', 'Priority seating reservation');

INSERT INTO FreeGift VALUES ('B311', 'GF011', 'Complimentary Full Meal', 'One per quarter');

INSERT INTO FreeGift VALUES ('B312', 'GF012', 'Premium Welcome Kit', 'Welcome gift upon reaching Diamond tier');

INSERT INTO RewardCatalog VALUES ('C1001', 'Combo Meals', 'Meal combos with drinks', 'Active');

INSERT INTO RewardCatalog VALUES ('C1002', 'Gift Items', 'Branded merchandise', 'Inactive');

INSERT INTO RewardCatalog VALUES ('C1003', 'Refreshments', 'Cold and beverage selections', 'Active');

INSERT INTO RewardCatalog VALUES ('C1004', 'Seasonal Specials', 'Seasonal or limited time offers', 'Active');

INSERT INTO RewardCatalog VALUES ('C1005', 'Kids'' Rewards', 'Toys and items for kids', 'Inactive');

INSERT INTO Reward VALUES ('R0001', 'Chicken Combo Meal', 180, 'Food', 25, 'C1001');

INSERT INTO Reward VALUES ('R0002', 'Coffee Mug', 120, 'Merchandise', 10, 'C1002');

INSERT INTO Reward VALUES ('R0003', 'Ice Cream Sundae', 90, 'Dessert', 20, 'C1001');

INSERT INTO Reward VALUES ('R0004', 'Fries Upgrade', 60, 'Food Add-on', 40, 'C1001');

INSERT INTO Reward VALUES ('R0005', 'Fried Chicken Piece', 70, 'Food Add-on', 50, 'C1001');

INSERT INTO Reward VALUES ('R0006', 'Cheeseburger Set', 150, 'Food', 30, 'C1001');

INSERT INTO Reward VALUES ('R0007', 'Tumbler Bottle', 160, 'Merchandise', 12, 'C1002');

INSERT INTO Reward VALUES ('R0008', 'Plush Toy', 100, 'Merchandise', 5, 'C1002');

INSERT INTO Reward VALUES ('R0009', 'Free americano', 90, 'Beverage', 15, 'C1003');

INSERT INTO Reward VALUES ('R0010', 'Free Ice Latte', 100, 'Beverage', 20, 'C1003');

INSERT INTO Reward VALUES ('R0011', 'Tupperware', 300, 'Merchandise', 10, 'C1005');

INSERT INTO Reward VALUES ('R0012', 'Kids Meal Set', 140, 'Food', 18, 'C1005');

INSERT INTO Reward VALUES ('R0013', 'Festive Hamper', 250, 'Seasonal', 5, 'C1004');

INSERT INTO Reward VALUES ('R0014', 'Free Ice mocha', 120, 'Beverage', 20, 'C1003');

INSERT INTO Reward VALUES ('R0015', 'Limited Edition Toy', 300, 'Collectible', 3, 'C1005');

INSERT INTO Inventory VALUES ('I001', 'BA1', 100);

INSERT INTO Inventory VALUES ('I002', 'BB1', 150);

INSERT INTO Inventory VALUES ('I003', 'BA2', 100);

INSERT INTO Inventory VALUES ('I004', 'BA3', 100);

INSERT INTO Inventory VALUES ('I005', 'BB3', 150);

INSERT INTO Inventory VALUES ('I006', 'BC1', 50);

INSERT INTO Inventory VALUES ('I007', 'BB2', 150);

INSERT INTO Restocking VALUES ('RS001', 'R0001', 'I001', TO\_DATE('04-01-2025', 'DD-MM-YYYY'), 20, 'S001');

INSERT INTO Restocking VALUES ('RS002', 'R0006', 'I002', TO\_DATE('16-01-2025', 'DD-MM-YYYY'), 20, 'S002');

INSERT INTO Restocking VALUES ('RS003', 'R0005', 'I003', TO\_DATE('02-02-2025', 'DD-MM-YYYY'), 30, 'S003');

INSERT INTO Restocking VALUES ('RS004', 'R0008', 'I004', TO\_DATE('22-02-2025', 'DD-MM-YYYY'), 30, 'S001');

INSERT INTO Restocking VALUES ('RS005', 'R0015', 'I005', TO\_DATE('27-02-2025', 'DD-MM-YYYY'), 3, 'S005');

INSERT INTO Restocking VALUES ('RS006', 'R0013', 'I006', TO\_DATE('11-03-2025', 'DD-MM-YYYY'), 5, 'S002');

INSERT INTO Restocking VALUES ('RS007', 'R0014', 'I003', TO\_DATE('11-03-2025', 'DD-MM-YYYY'), 10, 'S003');

INSERT INTO Restocking VALUES ('RS008', 'R0003', 'I004', TO\_DATE('15-03-2025', 'DD-MM-YYYY'), 5, 'S003');

INSERT INTO Restocking VALUES ('RS009', 'R0010', 'I007', TO\_DATE('16-03-2025', 'DD-MM-YYYY'), 10, 'S004');

INSERT INTO Restocking VALUES ('RS010', 'R0011', 'I001', TO\_DATE('20-03-2025', 'DD-MM-YYYY'), 5, 'S001');

INSERT INTO Restocking VALUES ('RS011', 'R0004', 'I002', TO\_DATE('02-04-2025', 'DD-MM-YYYY'), 30, 'S002');

INSERT INTO Restocking VALUES ('RS012', 'R0002', 'I002', TO\_DATE('03-04-2025', 'DD-MM-YYYY'), 5, 'S003');

INSERT INTO Restocking VALUES ('RS013', 'R0007', 'I003', TO\_DATE('03-04-2025', 'DD-MM-YYYY'), 5, 'S001');

INSERT INTO Restocking VALUES ('RS014', 'R0009', 'I006', TO\_DATE('10-04-2025', 'DD-MM-YYYY'), 10, 'S005');

INSERT INTO Restocking VALUES ('RS015', 'R0012', 'I002', TO\_DATE('10-04-2025', 'DD-MM-YYYY'), 10, 'S002');

INSERT INTO PointRecord VALUES ('PR2001', 'P1001', 1000, 'earned', TO\_DATE('15/07/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2002', 'P1001', -500, 'used', TO\_DATE('10/11/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2003', 'P1001', 1500, 'earned', TO\_DATE('20/12/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2004', 'P1002', 1000, 'earned', TO\_DATE('03/01/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2005', 'P1002', -500, 'used', TO\_DATE('25/04/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2006', 'P1003', 500, 'earned', TO\_DATE('09/12/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2007', 'P1003', -500, 'used', TO\_DATE('13/12/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2008', 'P1004', 2000, 'earned', TO\_DATE('27/08/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2009', 'P1004', -1000, 'used', TO\_DATE('10/10/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2010', 'P1005', 1000, 'earned', TO\_DATE('04/02/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2011', 'P1005', -1000, 'used', TO\_DATE('18/02/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2012', 'P1006', 750, 'earned', TO\_DATE('19/03/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2013', 'P1006', -500, 'used', TO\_DATE('28/03/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2014', 'P1007', 750, 'earned', TO\_DATE('07/06/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2015', 'P1007', -750, 'used', TO\_DATE('29/06/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2016', 'P1008', 500, 'earned', TO\_DATE('16/01/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2017', 'P1008', 300, 'earned', TO\_DATE('18/01/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2018', 'P1008', -500, 'used', TO\_DATE('20/01/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2019', 'P1009', 300, 'earned', TO\_DATE('05/01/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2020', 'P1009', -300, 'used', TO\_DATE('15/01/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2021', 'P1010', 1200, 'earned', TO\_DATE('10/02/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2022', 'P1010', -500, 'used', TO\_DATE('25/02/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2023', 'P1011', 900, 'earned', TO\_DATE('05/03/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2024', 'P1011', -500, 'used', TO\_DATE('15/03/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2025', 'P1012', 600, 'earned', TO\_DATE('10/01/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2026', 'P1012', -600, 'used', TO\_DATE('20/01/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2027', 'P1013', 1800, 'earned', TO\_DATE('15/04/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2028', 'P1013', -500, 'used', TO\_DATE('25/04/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2029', 'P1014', 400, 'earned', TO\_DATE('01/05/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2030', 'P1014', -500, 'used', TO\_DATE('10/05/2024', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2031', 'P1015', 400, 'earned', TO\_DATE('15/06/2023', 'DD/MM/YYYY'));

INSERT INTO PointRecord VALUES ('PR2032', 'P1015', -400, 'used', TO\_DATE('25/06/2023', 'DD/MM/YYYY'));

INSERT INTO Active VALUES ('ACT301', 'P1001', TO\_DATE('31/12/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT302', 'P1002', TO\_DATE('25/10/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT303', 'P1004', TO\_DATE('10/04/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT304', 'P1006', TO\_DATE('28/09/2023', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT305', 'P1008', TO\_DATE('20/07/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT306', 'P1010', TO\_DATE('25/08/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT307', 'P1011', TO\_DATE('15/09/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT308', 'P1013', TO\_DATE('25/10/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Active VALUES ('ACT309', 'P1014', TO\_DATE('10/11/2024', 'DD/MM/YYYY'), 'Y');

INSERT INTO Expiry VALUES ('EXP201', 'P1003', TO\_DATE('13/12/2023', 'DD/MM/YYYY'), 'expired');

INSERT INTO Expiry VALUES ('EXP202', 'P1005', TO\_DATE('18/02/2024', 'DD/MM/YYYY'), 'expired');

INSERT INTO Expiry VALUES ('EXP203', 'P1007', TO\_DATE('29/06/2023', 'DD/MM/YYYY'), 'expired');

INSERT INTO Expiry VALUES ('EXP204', 'P1009', TO\_DATE('15/01/2023', 'DD/MM/YYYY'), 'expired');

INSERT INTO Expiry VALUES ('EXP205', 'P1012', TO\_DATE('20/01/2023', 'DD/MM/YYYY'), 'expired');

INSERT INTO Expiry VALUES ('EXP206', 'P1015', TO\_DATE('25/06/2023', 'DD/MM/YYYY'), 'expired');

INSERT INTO Transaction VALUES ('T0001', 'Earn', 100, 0, TO\_DATE('11-04-2025', 'DD-MM-YYYY'), 'Completed', 10001);

INSERT INTO Transaction VALUES ('T0002', 'Redeem', 0, 150, TO\_DATE('11-04-2025', 'DD-MM-YYYY'), 'Completed', 10002);

INSERT INTO Transaction VALUES ('T0003', 'Earn', 200, 0, TO\_DATE('12-04-2025', 'DD-MM-YYYY'), 'Pending', 10003);

INSERT INTO Transaction VALUES ('T0004', 'Earn', 150, 0, TO\_DATE('13-04-2025', 'DD-MM-YYYY'), 'Completed', 10004);

INSERT INTO Transaction VALUES ('T0005', 'Redeem', 0, 120, TO\_DATE('14-04-2025', 'DD-MM-YYYY'), 'Completed', 10005);

INSERT INTO Transaction VALUES ('T0006', 'Earn', 300, 0, TO\_DATE('15-04-2025', 'DD-MM-YYYY'), 'Completed', 10006);

INSERT INTO Transaction VALUES ('T0007', 'Redeem', 0, 100, TO\_DATE('16-04-2025', 'DD-MM-YYYY'), 'Completed', 10007);

INSERT INTO Transaction VALUES ('T0008', 'Earn', 180, 0, TO\_DATE('17-04-2025', 'DD-MM-YYYY'), 'Completed', 10008);

INSERT INTO Transaction VALUES ('T0009', 'Earn', 250, 0, TO\_DATE('17-04-2025', 'DD-MM-YYYY'), 'Pending', 10009);

INSERT INTO Transaction VALUES ('T0010', 'Redeem', 0, 200, TO\_DATE('18-04-2025', 'DD-MM-YYYY'), 'Completed', 10010);

INSERT INTO Transaction VALUES ('T0011', 'Earn', 80, 0, TO\_DATE('25-04-2025', 'DD-MM-YYYY'), 'Completed', 10011);

INSERT INTO Transaction VALUES ('T0012', 'Redeem', 0, 90, TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Completed', 10012);

INSERT INTO Transaction VALUES ('T0013', 'Redeem', 0, 100, TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Pending', 10013);

INSERT INTO Transaction VALUES ('T0014', 'Redeem', 0, 80, TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Completed', 10014);

INSERT INTO Transaction VALUES ('T0015', 'Earn', 120, 0, TO\_DATE('27-04-2025', 'DD-MM-YYYY'), 'Completed', 10015);

INSERT INTO Transaction VALUES ('T0016', 'Redeem', 0, 50, TO\_DATE('27-04-2024', 'DD-MM-YYYY'), 'Pending', 10002);

INSERT INTO Transaction VALUES ('T0017', 'Redeem', 0, 250, TO\_DATE('27-04-2025', 'DD-MM-YYYY'), 'Pending', 10004);

INSERT INTO Transaction VALUES ('T0018', 'Earn', 90, 0, TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Failed', 10008);

INSERT INTO Transaction VALUES ('T0019', 'Redeem', 0, 70, TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Completed', 10007);

INSERT INTO Transaction VALUES ('T0020', 'Redeem', 0, 120, TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Failed', 10009);

INSERT INTO Transaction VALUES ('T0021', 'Redeem', 0, 130, TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 10007);

INSERT INTO Transaction VALUES ('T0022', 'Earn', 130, 0, TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 10008);

INSERT INTO Transaction VALUES ('T0023', 'Redeem', 0, 110, TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 10009);

INSERT INTO Transaction VALUES ('T0024', 'Redeem', 0, 60, TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 10012);

INSERT INTO Transaction VALUES ('T0025', 'Redeem', 0, 160, TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 10012);

INSERT INTO Transaction VALUES ('T0026', 'Earn', 100, 0, TO\_DATE('30-04-2025', 'DD-MM-YYYY'), 'Completed', 10001);

INSERT INTO Transaction VALUES ('T0027', 'Earn', 135, 0, TO\_DATE('30-04-2025', 'DD-MM-YYYY'), 'Completed', 10002);

INSERT INTO Transaction VALUES ('T0028', 'Earn', 105, 0, TO\_DATE('30-04-2025', 'DD-MM-YYYY'), 'Completed', 10003);

INSERT INTO Transaction VALUES ('T0029', 'Earn', 160, 0, TO\_DATE('30-04-2025', 'DD-MM-YYYY'), 'Completed', 10002);

INSERT INTO Transaction VALUES ('T0030', 'Earn', 50, 0, TO\_DATE('30-04-2025', 'DD-MM-YYYY'), 'Completed', 10001);

INSERT INTO TransactionHistory VALUES ('H0001', 'Earn', 100, 0, 'T0001', 10001);

INSERT INTO TransactionHistory VALUES ('H0002', 'Redeem', 0, 150, 'T0002', 10002);

INSERT INTO TransactionHistory VALUES ('H0003', 'Earn', 150, 0, 'T0004', 10004);

INSERT INTO TransactionHistory VALUES ('H0004', 'Redeem', 0, 120, 'T0005', 10005);

INSERT INTO TransactionHistory VALUES ('H0005', 'Earn', 300, 0, 'T0006', 10006);

INSERT INTO TransactionHistory VALUES ('H0006', 'Redeem', 0, 100, 'T0007', 10007);

INSERT INTO TransactionHistory VALUES ('H0007', 'Earn', 180, 0, 'T0008', 10008);

INSERT INTO TransactionHistory VALUES ('H0008', 'Redeem', 0, 200, 'T0010', 10010);

INSERT INTO TransactionHistory VALUES ('H0009', 'Earn', 80, 0, 'T0011', 10011);

INSERT INTO TransactionHistory VALUES ('H0010', 'Redeem', 0, 90, 'T0012', 10012);

INSERT INTO TransactionHistory VALUES ('H0011', 'Redeem', 0, 80, 'T0014', 10014);

INSERT INTO TransactionHistory VALUES ('H0012', 'Earn', 120, 0, 'T0015', 10015);

INSERT INTO TransactionHistory VALUES ('H0013', 'Redeem', 0, 70, 'T0019', 10007);

INSERT INTO TransactionHistory VALUES ('H0014', 'Redeem', 0, 130, 'T0021', 10007);

INSERT INTO TransactionHistory VALUES ('H0015', 'Earn', 130, 0, 'T0022', 10008);

INSERT INTO TransactionHistory VALUES ('H0016', 'Redeem', 0, 110, 'T0023', 10009);

INSERT INTO TransactionHistory VALUES ('H0017', 'Redeem', 0, 60, 'T0024', 10012);

INSERT INTO TransactionHistory VALUES ('H0018', 'Redeem', 0, 160, 'T0025', 10012);

INSERT INTO TransactionHistory VALUES ('H0019', 'Earn', 100, 0, 'T0026', 10001);

INSERT INTO TransactionHistory VALUES ('H0020', 'Earn', 135, 0, 'T0027', 10002);

INSERT INTO TransactionHistory VALUES ('H0021', 'Earn', 105, 0, 'T0028', 10003);

INSERT INTO TransactionHistory VALUES ('H0022', 'Earn', 160, 0, 'T0029', 10002);

INSERT INTO TransactionHistory VALUES ('H0023', 'Earn', 50, 0, 'T0030', 10001);

INSERT INTO Payment VALUES ('PAY0001', 100.00, 'Completed', 'E-Wallet', 'T0001');

INSERT INTO Payment VALUES ('PAY0002', 200.00, 'Pending', 'Card', 'T0003');

INSERT INTO Payment VALUES ('PAY0003', 150.00, 'Completed', 'Online Banking', 'T0004');

INSERT INTO Payment VALUES ('PAY0004', 300.00, 'Completed', 'E-Wallet', 'T0006');

INSERT INTO Payment VALUES ('PAY0005', 180.14, 'Completed', 'Card', 'T0008');

INSERT INTO Payment VALUES ('PAY0006', 250.06, 'Pending', 'Online Banking', 'T0009');

INSERT INTO Payment VALUES ('PAY0007', 80.10, 'Completed', 'E-Wallet', 'T0011');

INSERT INTO Payment VALUES ('PAY0008', 120.65, 'Completed', 'Card', 'T0015');

INSERT INTO Payment VALUES ('PAY0009', 90.33, 'Failed', 'E-Wallet', 'T0018');

INSERT INTO Payment VALUES ('PAY0010', 130.00, 'Completed', 'Online Banking', 'T0022');

INSERT INTO Payment VALUES ('PAY0011', 100.20, 'Completed', 'E-Wallet', 'T0026');

INSERT INTO Payment VALUES ('PAY0012', 135.00, 'Completed', 'Card', 'T0027');

INSERT INTO Payment VALUES ('PAY0013', 105.67, 'Completed', 'Online Banking', 'T0028');

INSERT INTO Payment VALUES ('PAY0014', 160.66, 'Completed', 'Card', 'T0029');

INSERT INTO Payment VALUES ('PAY0015', 50.00, 'Completed', 'Online Banking', 'T0030');

INSERT INTO EWALLET VALUES ('PAY0001', 912345678891, 'GrabPay', 'MYR');

INSERT INTO EWALLET VALUES ('PAY0004', 923456789012, 'Touch''n Go', 'MYR');

INSERT INTO EWALLET VALUES ('PAY0007', 934567890123, 'PayPal', 'USD');

INSERT INTO EWALLET VALUES ('PAY0009', 945678901234, 'Boost', 'MYR');

INSERT INTO EWALLET VALUES ('PAY0011', 956789012345, 'Touch''n Go', 'MYR');

INSERT INTO Card VALUES ('PAY0002', 4098765432109871, 'Sarah Lim', 'Credit');

INSERT INTO Card VALUES ('PAY0005', 4987654321098762, 'Nurul Izzati', 'Debit');

INSERT INTO Card VALUES ('PAY0008', 4876543210987653, 'Kavita Rani', 'Credit');

INSERT INTO Card VALUES ('PAY0012', 4765432109876544, 'Nur Sabrina', 'Credit');

INSERT INTO Card VALUES ('PAY0014', 4654321098765435, 'Anita Das', 'Debit');

INSERT INTO OnlineBanking VALUES ('PAY0003', 1234567890, 'Maybank');

INSERT INTO OnlineBanking VALUES ('PAY0006', 2345678901, 'CIMB');

INSERT INTO OnlineBanking VALUES ('PAY0010', 3456789012, 'RHB');

INSERT INTO OnlineBanking VALUES ('PAY0013', 4567890123, 'Public Bank');

INSERT INTO OnlineBanking VALUES ('PAY0015', 5678901234, 'Maybank');

INSERT INTO Redemption VALUES ('RD01', 10002, 'R0001', TO\_DATE('11-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0002', 150);

INSERT INTO Redemption VALUES ('RD02', 10004, 'R0002', TO\_DATE('13-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0004', 120);

INSERT INTO Redemption VALUES ('RD03', 10007, 'R0004', TO\_DATE('16-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0007', 100);

INSERT INTO Redemption VALUES ('RD04', 10009, 'R0006', TO\_DATE('18-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0009', 200);

INSERT INTO Redemption VALUES ('RD05', 10012, 'R0008', TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0012', 90);

INSERT INTO Redemption VALUES ('RD06', 10013, 'R0012', TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Pending', 'T0013', 100);

INSERT INTO Redemption VALUES ('RD07', 10014, 'R0007', TO\_DATE('26-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0014', 80);

INSERT INTO Redemption VALUES ('RD08', 10002, 'R0011', TO\_DATE('27-04-2025', 'DD-MM-YYYY'), 'Pending', 'T0016', 50);

INSERT INTO Redemption VALUES ('RD09', 10004, 'R0013', TO\_DATE('27-04-2025', 'DD-MM-YYYY'), 'Pending', 'T0017', 250);

INSERT INTO Redemption VALUES ('RD10', 10007, 'R0010', TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0019', 70);

INSERT INTO Redemption VALUES ('RD11', 10009, 'R0003', TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Failed', 'T0020', 120);

INSERT INTO Redemption VALUES ('RD12', 10007, 'R0015', TO\_DATE('28-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0021', 130);

INSERT INTO Redemption VALUES ('RD13', 10009, 'R0014', TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0023', 110);

INSERT INTO Redemption VALUES ('RD14', 10012, 'R0009', TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0024', 60);

INSERT INTO Redemption VALUES ('RD15', 10012, 'R0005', TO\_DATE('29-04-2025', 'DD-MM-YYYY'), 'Completed', 'T0025', 160);

# 7.0 PART2 INDIVIDUAL SQL SCRIPT

## Member 1: Ng Wei Jie

**Query 1**

**Display active customer with their membership and Points Balance.**

SELECT c.CustID, c.CustName, mt.TierName, p.TotalPointsEarned

FROM Customer c, Membership m, MembershipTier mt, Point p

WHERE c.MemID = m.MemID

AND m.MemTierID = mt.MemTierID

AND p.CustID = c.CustID

AND m.Status = 'Active'

ORDER BY mt.PointsRequired DESC, p.TotalPointsEarned DESC;

**Query 2**

**Display expired points and reasons from "Expiry".**

SELECT c.CustName, e.ExpiryPointID, e.DateExpired, e.Reason

FROM Customer c, Point p, Expiry e

WHERE c.CustID = p.CustID

AND p.PointID = e.PointID

ORDER BY e.expiryPointID , e.DateExpired;

**Stored Procedure 1**

**Customer could earn points after transaction.**

CREATE OR REPLACE PROCEDURE Earn\_Points\_After\_Transaction (

p\_CustID IN NUMBER,

p\_TransactionAmount IN NUMBER

)

IS

v\_EarnedPoints NUMBER;

v\_PointID VARCHAR2(5);

v\_RecordID VARCHAR2(6);

BEGIN

-- Calculate points earned (1 point per RM1 spent)

v\_EarnedPoints := FLOOR(p\_TransactionAmount);

-- Get PointID for this customer (assume only 1 active point record)

SELECT PointID

INTO v\_PointID

FROM Point

WHERE CustID = p\_CustID

AND PointStatus = 'Active'

AND ROWNUM = 1;

-- Generate a new RecordID manually (e.g., 'PR' || random number)

v\_RecordID := 'PR' || TO\_CHAR(TRUNC(DBMS\_RANDOM.VALUE(1000,9999)));

-- Update PointsBalance and TotalPointsEarned

UPDATE Point

SET PointsBalance = PointsBalance + v\_EarnedPoints,

TotalPointsEarned = TotalPointsEarned + v\_EarnedPoints

WHERE PointID = v\_PointID;

-- Insert into PointRecord

INSERT INTO PointRecord (RecordID, PointID, PointsChanged, ChangedType, ChangeDate)

VALUES (

v\_RecordID,

v\_PointID,

v\_EarnedPoints,

'earned',

SYSDATE

);

COMMIT;

END;

/

Call procedure:

EXEC Earn\_Points\_After\_Transaction(10001, 120);

To check:

SELECT \* FROM Point WHERE CustID = 10001;

**Procedure 2**

**Expire customer points manually.**

CREATE OR REPLACE PROCEDURE Expire\_Customer\_Points (

p\_CustID IN NUMBER

)

IS

v\_PointID VARCHAR2(5);

v\_ExpiredPoints NUMBER;

v\_RecordID VARCHAR2(6);

BEGIN

-- Find the active point record for the customer

SELECT PointID

INTO v\_PointID

FROM Point

WHERE CustID = p\_CustID

AND PointStatus = 'Active'

AND ROWNUM = 1;

-- Get current PointsBalance

SELECT PointsBalance

INTO v\_ExpiredPoints

FROM Point

WHERE PointID = v\_PointID;

-- Set PointsBalance to 0 and mark as 'Expiry'

UPDATE Point

SET PointsBalance = 0,

TotalPointsUsed = TotalPointsUsed + v\_ExpiredPoints,

PointStatus = 'Expiry'

WHERE PointID = v\_PointID;

-- Generate new RecordID manually

v\_RecordID := 'PR' || TO\_CHAR(TRUNC(DBMS\_RANDOM.VALUE(1000,9999)));

-- Insert into PointRecord as "used"

INSERT INTO PointRecord (RecordID, PointID, PointsChanged, ChangedType, ChangeDate)

VALUES (

v\_RecordID,

v\_PointID,

-v\_ExpiredPoints,

'used',

SYSDATE

);

COMMIT;

END;

/

Call procedure:

EXECUTE Expire\_Customer\_Points(10002);

To check:

SELECT PointID, CustID, PointsBalance, TotalPointsUsed, PointStatus

FROM Point

WHERE CustID = 10002;

**Function 1**

**Calculate available points for a customer.**

CREATE OR REPLACE FUNCTION Get\_Available\_Points (

p\_CustID IN NUMBER

) RETURN NUMBER

IS

v\_Points NUMBER;

BEGIN

SELECT PointsBalance

INTO v\_Points

FROM Point

WHERE CustID = p\_CustID

AND PointStatus = 'Active'

AND ROWNUM = 1;

RETURN v\_Points;

END;

/

Call function:

SET SERVEROUTPUT ON;

DECLARE

v\_custID NUMBER := 10001; -- Can change to test other customers

v\_points NUMBER;

BEGIN

v\_points := Get\_Available\_Points(v\_custID);

DBMS\_OUTPUT.PUT\_LINE('Customer ' || v\_custID || ' has ' || v\_points || ' available points.');

END;

/

**Function 2**

**Predict Customer's need how many points to upgrade to next Tier.**

CREATE OR REPLACE FUNCTION Predict\_Tier\_Upgrade(

p\_CustID IN NUMBER

) RETURN VARCHAR2

IS

v\_Points NUMBER;

BEGIN

SELECT NVL(SUM(PointsBalance), 0)

INTO v\_Points

FROM Point

WHERE CustID = p\_CustID

AND PointStatus = 'Active';

IF v\_Points >= 5000 THEN

RETURN 'Already Top Tier';

ELSIF v\_Points >= 3000 THEN

RETURN 'Needs ' || (5000 - v\_Points) || ' points to Top Tier';

ELSIF v\_Points >= 1000 THEN

RETURN 'Needs ' || (3000 - v\_Points) || ' points to Mid Tier';

ELSE

RETURN 'Needs ' || (1000 - v\_Points) || ' points to Basic Tier';

END IF;

EXCEPTION

WHEN OTHERS THEN

RETURN 'ERROR';

END;

/

Call function:

SET SERVEROUTPUT ON;

DECLARE

v\_result VARCHAR2(200);

BEGIN

v\_result := Predict\_Tier\_Upgrade(10001);

DBMS\_OUTPUT.PUT\_LINE('Tier Prediction: ' || v\_result);

END;

/

## Member 2: Owi Wilson

**Query 1**

**Display all pending reward redemption with customers names and reward redemption details. This query helps to ease the staffs to check on which redemption is pending to be processed.**

SELECT

    r.RedemptionID,

    c.CustName AS "NAME",

    rw.RewardName,

    r.RedemptionDate AS "DATE",

    r.Status

FROM

    Redemption r, Customer c, Reward rw

WHERE

    r.CustID = c.CustID AND r.RewardID = rw.RewardID AND r.Status = 'Pending'

ORDER BY

    r.RedemptionDate ASC;

**Query 2**

**Display the rewards that are in active status based on their reward catalog. This query ease customers in viewing all available rewards that can be redeemed.**

SELECT

    rc.CatalogID AS "CID",

    rc.RewardName AS "Catalog",

    r.RewardID AS "RID",

    r.RewardName AS Reward,

    r.PointsRequired

FROM

    RewardCatalog rc

JOIN

    Reward r ON rc.CatalogID = r.CatalogID

WHERE

    rc.IsActive = 'Active'

ORDER BY

    rc.CatalogID, r.RewardName;

**Stored Procedure 1**

**Processes the reward redemption after the request is approved for a customer by updating the customer’s point balance and reward stock. The procedure checks the status, stock and points balance before processing the redemption.**

Initially:

SELECT \* FROM Reward;

CREATE OR REPLACE PROCEDURE RedeemRewards (

    cID IN Customer.CustID%TYPE,

    rID IN Reward.RewardID%TYPE

)

IS

    curPoints Point.PointsBalance%TYPE;

    rewardPoints Reward.PointsRequired%TYPE;

    curStock Reward.Stock%TYPE;

    status RewardCatalog.IsActive%TYPE;

BEGIN

    SELECT PointsBalance INTO curPoints

    FROM Point

    WHERE CustID = cID;

    SELECT r.PointsRequired, r.Stock, rc.IsActive

    INTO rewardPoints, curStock, status

    FROM Reward r, RewardCatalog rc

    WHERE r.CatalogID = rc.CatalogID AND r.RewardID = rID;

    IF status != 'Active' THEN

        DBMS\_OUTPUT.PUT\_LINE('Reward is Inactive. Cannot Redeem.');

    ELSIF curPoints < rewardPoints THEN

        DBMS\_OUTPUT.PUT\_LINE('Insufficient points to redeem this reward.');

    ELSIF curStock <= 0 THEN

        DBMS\_OUTPUT.PUT\_LINE('Reward is out of stock.');

    ELSE

        UPDATE Reward

        SET Stock = Stock - 1

        WHERE RewardID = rID;

        COMMIT;

        DBMS\_OUTPUT.PUT\_LINE('Redemption successful.');

    END IF;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: Customer or Reward not found.');

    WHEN OTHERS THEN

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

END;

/

Call procedure:

EXECUTE RedeemRewards('10001','R0001');

To Check:

SELECT \* FROM Reward;

**Stored procedure 2**

**Insert new value into Restocking table after an reward item being restocked by the staff. The procedures also updates the stock value of each reward item in Reward Table**

Initially:

SELECT \* FROM Restocking;

CREATE OR REPLACE PROCEDURE RestockRewardItem(

    rID VARCHAR2,

    rewID VARCHAR2,

    invID VARCHAR2,

    qty NUMBER,

    suppID VARCHAR2

)

IS

BEGIN

    UPDATE Reward

    SET Stock = Stock + qty

    WHERE RewardID = rewID;

    INSERT INTO Restocking(RestockID, RewardID, InventoryID, RestockingDate, Quantity, SupplierID)

    VALUES (rID, rewID, invID, SYSDATE, qty, suppID);

    COMMIT;

EXCEPTION

    WHEN OTHERS THEN

        ROLLBACK;

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

END;

/

Call procedure:

EXECUTE RestockRewardItem('RS017','R0001','I001',50,'S0001');

To Check:

SELECT \* FROM Restocking;

**Function 1**

**This function helps to check the stock condition of a specific reward item to evaluate whether it needs restocking. The function returns a message indicating whether it requires restocking or not.**

CREATE OR REPLACE FUNCTION CheckStock(

    p\_RewardID VARCHAR2

)

RETURN VARCHAR2

IS

    lastRestockDate DATE;

    curStock NUMBER(3);

    str VARCHAR2(100);

BEGIN

    SELECT MAX(RestockingDate) INTO lastRestockDate

    FROM Restocking

    WHERE RewardID = p\_RewardID;

    SELECT Stock INTO curStock

    FROM Reward

    WHERE RewardID = p\_RewardID;

    IF (SYSDATE - lastRestockDate > 60) AND (curStock <= 10) THEN

        str := 'Low Stock and Overdue Restocking! Need to Restock!';

    ELSIF (SYSDATE - lastRestockDate > 60) THEN

        str := 'Restocking Overdue. Need to Restock!';

    ELSIF (curStock <= 10) THEN

        str := 'Insufficient Stock. Need to Restock!';

    ELSE

        str := 'Sufficient Stock';

    END IF;

    RETURN str;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        RETURN 'Reward Item Not Found';

    WHEN OTHERS THEN

        RETURN 'Error: Failed To Check Stock';

END;

/

Call Function:

BEGIN

    DBMS\_OUTPUT.PUT\_LINE(CheckStock('R0001'));

END;

/

**Function 2**

**Check whether a customer can redeem a specific reward based on his point balance. The function returns a message for customers to help them indicate whether they can redeem this reward.**

CREATE OR REPLACE FUNCTION CanRedeemReward (

    cID IN Customer.CustID%TYPE,

    rID IN Reward.RewardID%TYPE

) RETURN VARCHAR2

IS

    custPoints Point.PointsBalance%TYPE;

    pointsNeeded Reward.PointsRequired%TYPE;

BEGIN

    SELECT PointsBalance INTO custPoints

    FROM Point

    WHERE CustID = cID;

    SELECT PointsRequired INTO pointsNeeded

    FROM Reward

    WHERE RewardID = rID;

    IF custPoints >= pointsNeeded THEN

        RETURN 'YES, U CAN REDEEM';

    ELSE

        RETURN 'NO, U CANNOT REDEEM';

    END IF;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        RETURN 'CUSTOMER DOES NOT EXISTS';

    WHEN OTHERS THEN

        RETURN 'NO, U CANNOT REDEEM';

END;

/

Call Function:

BEGIN

    DBMS\_OUTPUT.PUT\_LINE(CanRedeemReward(1001, 'R0001'));

END;

/

## Member 3: Tan Cheng Lock

**Query 1**

**This query ensures only *completed* transactions are recorded in the TransactionHistory table, maintaining data consistency by excluding pending and failed transactions.**

SELECT

t.TransactionID, t.TransactionStatus, th.HistoryID,

th.TransactionType,

th.PointsEarned,

th.PointsRedeemed

FROM transaction t, transactionHistory th

WHERE t.transactionID = th.transactionID

AND t.TransactionType = th.TransactionType

AND t.PointsEarned = th.PointsEarned

AND t.PointsRedeemed = th.PointsRedeemed

AND t.transactionStatus = 'Completed';

**Query 2**

**This query shows how points are awarded based on the payment amount, following the rule that RM1 spent = 1 point earned, for completed 'Earn' transactions.**

SELECT p.PaymentID, p.TotalAmount, p.PaymentMethod,

t.transactionType, t.pointsEarned, t.transactionStatus

FROM payment p, transaction t

WHERE p.TransactionID = t.TransactionID

AND t.transactionType = 'Earn'

AND t.transactionStatus = 'Completed';

**Stored Procedure 1**

**This procedure updates a transaction’s status from 'Pending' to 'Completed' to reflect successful transaction completion.**

Initially:

SELECT \* FROM transaction WHERE transactionID = 'T0003';

SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE Update\_TransactionStatus\_To\_Completed

(p\_TransactionID IN VARCHAR2)

IS

BEGIN

-- Update the transaction status to 'Completed'

UPDATE Transaction

SET TransactionStatus = 'Completed'

WHERE TransactionID = p\_TransactionID

AND TransactionStatus = 'Pending';

DBMS\_OUTPUT.PUT\_LINE('Transaction ' || p\_TransactionID || ' status updated to Completed.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Transaction ID ' || p\_TransactionID || ' not found or not in Pending status.');

WHEN OTHERS THEN

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

END;

/

Call Procedure:

EXECUTE Update\_TransactionStatus\_To\_Completed('T0003');

To Check:

SELECT \* FROM transaction WHERE transactionID = 'T0003';

**Stored Procedure 2**

**This procedure automatically finds and deletes all failed transactions and their related records.**

Initially:

SELECT \* FROM transaction WHERE transactionStatus = 'Failed';

SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE Delete\_Failed\_Transactions IS

CURSOR failed\_transactions IS

SELECT TransactionID, TransactionType

FROM Transaction

WHERE TransactionStatus = 'Failed';

v\_payment\_id Payment.PaymentID%TYPE;

BEGIN

FOR txn IN failed\_transactions LOOP

-- If it is a failed redemption, delete from Redemption table first (child table)

IF txn.TransactionType = 'Redeem' THEN

DELETE FROM Redemption WHERE TransactionID = txn.TransactionID;

END IF;

-- Find the PaymentID associated with Transaction

BEGIN

SELECT PaymentID INTO v\_payment\_id

FROM Payment

WHERE TransactionID = txn.TransactionID;

-- Delete from subclasses

DELETE FROM Ewallet WHERE PaymentID = v\_payment\_id;

DELETE FROM Card WHERE PaymentID = v\_payment\_id;

DELETE FROM OnlineBanking WHERE PaymentID = v\_payment\_id;

-- Delete Payment (superclass)

DELETE FROM Payment WHERE PaymentID = v\_payment\_id;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

-- Finally, delete the Transaction table (parent table)

DELETE FROM Transaction WHERE TransactionID = txn.TransactionID;

DBMS\_OUTPUT.PUT\_LINE('Successfully deleted failed transaction: ' || txn.TransactionID);

END LOOP;

EXCEPTION

WHEN OTHERS THEN

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

END;

/

Call Procedure:

EXECUTE Delete\_Failed\_Transactions;

To Check

SELECT \* FROM transaction WHERE transactionStatus = 'Failed';

**Function 1**

**This function allows customers and staff to check the status of a transaction using its ID.**

SET SERVEROUTPUT ON

CREATE OR REPLACE FUNCTION Get\_TransactionStatus

(p\_TransactionID IN VARCHAR2)

RETURN VARCHAR2

IS

v\_status VARCHAR2(20);

BEGIN

SELECT TransactionStatus INTO v\_status

FROM Transaction

WHERE TransactionID = p\_TransactionID;

RETURN v\_status;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Transaction not found';

WHEN OTHERS THEN

RETURN 'Error: ' || SQLERRM;

END;

/

Call function:

SET SERVEROUTPUT ON;

DECLARE

v\_transactionID VARCHAR2(20) := 'T0001';

v\_status VARCHAR2(20);

BEGIN

v\_status := Get\_TransactionStatus(v\_transactionID);

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

END;

/

**Function 2**

**This function summarizes a customer's total transactions, total payment amount, and transaction history.**

SET SERVEROUTPUT ON

CREATE OR REPLACE FUNCTION Get\_Customer\_Total\_Payment\_Amount

(p\_CustID IN NUMBER)

RETURN VARCHAR2

IS

v\_total\_transactions NUMBER := 0;

v\_total\_payments NUMBER := 0;

v\_total\_history NUMBER := 0;

v\_summary VARCHAR2(1000);

BEGIN

-- Count total transactions

SELECT COUNT(\*)

INTO v\_total\_transactions

FROM Transaction

WHERE CustID = p\_CustID;

SELECT NVL(SUM(p.TotalAmount), 0)

INTO v\_total\_payments

FROM Payment p, Transaction t

WHERE p.TransactionID = t.TransactionID

AND t.CustID = p\_CustID;

-- Count transaction history records

SELECT COUNT(\*)

INTO v\_total\_history

FROM TransactionHistory

WHERE CustID = p\_CustID;

-- Display message

v\_summary := 'Customer ' || p\_CustID || ': ' || v\_total\_transactions || ' transaction(s), ' || 'Total Paid: RM' || v\_total\_payments || ', ' || v\_total\_history || ' history record(s).';

RETURN v\_summary;

EXCEPTION

WHEN OTHERS THEN

RETURN 'Error: ' || SQLERRM;

END;

/

Call function:

SET SERVEROUTPUT ON;

DECLARE

v\_custID NUMBER := 10001;

v\_summary VARCHAR2(1000);

BEGIN

v\_summary := Get\_Customer\_Total\_Payment\_Amount(v\_custID);

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

END;

/

## Member 4: Yee Hao Zhe

**Query 1**

**Retrieves the Member ID, Tier ID, Tier Name and Total Points Earned for all members.**

SELECT

m.MemID, m.MemTierID,

mt.TierName,

p.TotalPointsEarned

FROM Membership m

JOIN MembershipTier mt ON m.MemTierID = mt.MemTierID

JOIN Point p ON m.MemID = p.MemID

ORDER BY p.TotalPointsEarned;

**Query 2**

**Lists vouchers with a value greater than RM 30 and their associated benefit names.**

SELECT

v.VoucherCode, v.Value,

mb.BenefitName

FROM Voucher v

JOIN MembershipBenefit mb ON v.BenefitID = mb.BenefitID

WHERE v.Value > 30;

**Stored Procedure 1**

**Updates the status of a specific membership record.**

CREATE OR REPLACE PROCEDURE UpdateMembershipStatus(

p\_MemID IN Membership.MemID%TYPE,

p\_NewStatus IN Membership.Status%TYPE

)

IS

BEGIN

IF p\_NewStatus NOT IN ('Active', 'Inactive') THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Invalid status provided. Use ''Active'' or ''Inactive''.');

RETURN;

END IF;

UPDATE Membership

SET Status = p\_NewStatus

WHERE MemID = p\_MemID;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Warning: Member ID ' || p\_MemID || ' not found. No status updated.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Success: Status for Member ID ' || p\_MemID || ' updated to ' || p\_NewStatus || '.');

COMMIT;

END IF;

END;

/

Call Procedure:

SET SERVEROUTPUT ON;

EXECUTE UpdateMembershipStatus('M0002', 'Active');

To Check:

SELECT \* FROM membership;

**Stored Procedure 2**

**Displays the details (discount percentage, applicable products) for a specific promotion code.**

CREATE OR REPLACE PROCEDURE ViewPromotionDetails(

p\_PromoCode IN Promotion.PromoCode%TYPE

)

IS

v\_Discount Promotion.DiscountPercentage%TYPE;

v\_ApplicableProducts Promotion.ApplicableProducts%TYPE;

BEGIN

SELECT DiscountPercentage, ApplicableProducts

INTO v\_Discount, v\_ApplicableProducts

FROM Promotion

WHERE PromoCode = p\_PromoCode;

DBMS\_OUTPUT.PUT\_LINE('Promo Code: ' || p\_PromoCode);

DBMS\_OUTPUT.PUT\_LINE('Discount Percentage: ' || v\_Discount || '%');

DBMS\_OUTPUT.PUT\_LINE('Applicable Products: ' || v\_ApplicableProducts);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Promotion not found for promo code: ' || p\_PromoCode);

END;

/

Call Procedure:

SET SERVEROUTPUT ON;

EXECUTE ViewPromotionDetails('APP5');

To Check:

SELECT \* FROM promotion;

**Function 1**

**Retrieves the membership tier name for a given member ID.**

CREATE OR REPLACE FUNCTION GetMemberTier(

p\_MemID IN Membership.MemID%TYPE

) RETURN VARCHAR2 IS

v\_TierName MembershipTier.TierName%TYPE;

BEGIN

SELECT mt.TierName

INTO v\_TierName

FROM Membership m

JOIN MembershipTier mt ON m.MemTierID = mt.MemTierID

WHERE m.MemID = p\_MemID;

RETURN v\_TierName;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Not Found';

END;

/

Call Function:

SET SERVEROUTPUT ON;

DECLARE

v\_tier VARCHAR2(20);

BEGIN

v\_tier := GetMemberTier('M0001');

DBMS\_OUTPUT.PUT\_LINE('Tier for M0001: ' || v\_tier);

END;

/

**Function 2**

**Retrieves all the benefit names for a specific membership tier.**

CREATE OR REPLACE FUNCTION Get\_Tier\_Benefit\_Names (

p\_MemTierID IN MembershipTier.MemTierID%TYPE

) RETURN VARCHAR2

IS

v\_benefit\_list VARCHAR2(4000) := NULL;

v\_benefit\_name MembershipBenefit.BenefitName%TYPE;

CURSOR benefit\_cursor IS

SELECT BenefitName

FROM MembershipBenefit

WHERE MemTierID = p\_MemTierID

ORDER BY BenefitName;

BEGIN

OPEN benefit\_cursor;

LOOP

FETCH benefit\_cursor INTO v\_benefit\_name;

EXIT WHEN benefit\_cursor%NOTFOUND;

IF v\_benefit\_list IS NOT NULL THEN

v\_benefit\_list := v\_benefit\_list || ', ';

END IF;

IF LENGTH(v\_benefit\_list || v\_benefit\_name) < 4000 THEN

v\_benefit\_list := v\_benefit\_list || v\_benefit\_name;

ELSE

v\_benefit\_list := v\_benefit\_list || '...';

EXIT;

END IF;

END LOOP;

CLOSE benefit\_cursor;

IF v\_benefit\_list IS NULL THEN

RETURN 'No benefits found';

ELSE

RETURN v\_benefit\_list;

END IF;

EXCEPTION

WHEN OTHERS THEN

IF benefit\_cursor%ISOPEN THEN

CLOSE benefit\_cursor;

END IF;

RETURN 'Error retrieving benefits: ' || SQLERRM;

END;

/

Call Function:

SET SERVEROUTPUT ON;

DECLARE

v\_benefits VARCHAR2(4000);

v\_tier\_id MembershipTier.MemTierID%TYPE := 'L02';

BEGIN

v\_benefits := Get\_Tier\_Benefit\_Names(v\_tier\_id);

DBMS\_OUTPUT.PUT\_LINE('Benefit names for Tier ' || v\_tier\_id || ': ' || v\_benefits);

END;

/