

**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): | G055 |
| Group leader to provide the OneDrive folder hyperlink (editor mode) which contains the zip file and a group presentation video in mp4 format: | https://utarict-my.sharepoint.com/:f:/g/personal/chengjun5487\_1utar\_my/EmdKcdY1wSxFkMREyjXCKPYBHlSiEN8-z6\_HMPKeJnwPPg?e=Aj10VI |
| Group leader name: | Lee Cheng Jun |
| Group leader email address: | chengjun5487@1utar.my |
| Submission date (dd-Mon-yy): | 29-April-2025 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Member Contribution Declaration Sheet** | | | | | |
|  | | **Member 1** | **Member 2** | **Member 3** | **Member 4** |
| **Member Name** (ascending order, group leader with \* ): | | **Brian Lee Zhen Hui** | **Lee Cheng Jun\*** | **Ng Che Te** | **Tan Jia Yi** |
| Student ID (e.g. 2299999) | | **2206413** | **2206342** | **2206349** | **2206756** |
| Program CS / IA, IB / DE: | | **CS** | **CS** | **CS** | **CS** |
| **Member’s signature** | | *brian* | *lee* | *ng* | *tan* |
|  | | Contribution of each member (Total 100%) | | | |
|  | | **%** | **%** | **%** | **%** |
| PART 1: (Group Assessment – 50 marls) | | **25** | **25** | **25** | **25** |
| **1.** | Scope of Work (5 marks). |  | **20** |  | **80** |
| **2.** | ER model (10 marks). |  |  |  | **100** |
| **3.** | Redesign and EER (10 marks). |  |  | **100** |  |
| **4.** | Data Dictionary (10 marks) |  | **60** | **40** |  |
| **5.** | Tables and records (5 marks) | **100** |  |  |  |
| **6.** | Script (10 marks) | **40** | **60** |  |  |

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)** |  | **Brian Lee Zhen Hui** | **Lee Cheng Jun** | **Ng Che Te** | **Tan Jia Yi** |
| **Student ID; (to fill)** |  | **2206413** | **2206342** | **2206349** | **2206756** |
| 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. | Brian Lee Zhen Hui | Contribute to Sql script, and Tables and Records |
| 2. | Lee Cheng Jun\* | Contribute to Data Dictionary and Sql script |
| 3. | Ng Che Te | Contribute to EER and Data Dictionary |
| 4. | Tan Jia Yi | Contribute to Scope of work and ER Model |
|  |  |  |

# **1. Scope of Work**

# **Overview of Chosen Area (Promotion)**

Promotion is an essential part of business operations. A well-executed promotion not only boosts sales but also enhances brand popularity by attracting new customers and converting them into loyal, long-term clients. Promotions can be designed in many ways depending on the target audience and the platform used. Different promotional strategies and their effects influence overall business performance.

If promotional data is not properly generated and managed, it can lead to delays in decision-making, increased operational inefficiencies, and the loss of customers who are influenced by market trends.

Therefore, the Promotion area has been selected for the database design and development project to help Grand Wooly manage their promotional activities more effectively. A complete promotion database will enable the company to generate valuable information, view up-to-date records, and perform comparisons to improve their business decisions.

The database will include **ten main functions** and **four junction tables**, designed to store different attributes based on their relationships.

# **Main Function Areas of the Promotion Database**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Main Function Area** | **Entity/Tables Involved** | **Description** |
| 1 | Manage Promotion Details | Promotion | Store event details, type, status, start and end dates for each promotion. |
| 2 | Manage Promo Items | PromoItem, Item\_Detail | Manage promoted items, their types, prices, available stock, discounts, and stock limitations. |
| 3 | Manage Discount Codes | Voucher, Voucher\_Item | Manage voucher codes, discount values, validity dates, and usage status; link vouchers to promo items. |
| 4 | Track Physical Marketing Events | MarketingEvent | Manage details of physical events linked to promotions, such as location, date, and time. |
| 5 | Manage Advertisement Campaigns | AdSchedule | Record advertisement schedule details including platform, start/end time, and content type. |
| 6 | Track Promotion Analytics | PromoAnalytics | Monitor and track sales impact and redemption rate for each promotion. |
| 7 | Gather Customer Feedback | Feedback | Collect and store customer feedback, comments, and ratings linked to promotions. |
| 8 | Customer Segmentation | CustomerSegment | Classify customers into segments based on criteria like loyalty points, behavior, or joining dates. |
| 9 | Target Segments for Promotions | Promotion\_Segment | Link promotions to specific customer segments and define the communication channel and priority. |
| 10 | Manage Customer Loyalty Program | Customer, LoyaltyProgram, CS\_Membership | Manage customer loyalty details such as points, tiers, membership in customer segments, and status. |

**The primary entity, Promotion**, will store essential information about each promotion, such as event details, current status, and validity period.

* The **PromoItem entity** captures information about the items being promoted, including the item type, stock available, and price.
* The **Item\_Detail junction entity** links Promotion and PromoItem, recording discount rates and stock limitations.
* The **Voucher entity** stores information regarding discount codes, validity periods, and usage status.
* The **Voucher\_Item junction entity** links vouchers to promoted items, allowing multiple items to be associated with a specific discount voucher.

* The **MarketingEvent entity** records physical event details associated with promotions.
* The **AdSchedule entity** captures the timeline and platform details for advertisement campaigns.

* The **PromoAnalytics entity** measures the effectiveness of each promotion by tracking sales impact and redemption rates.

* The **Feedback entity** gathers customer comments and ratings about promotions.

* The **CustomerSegment entity** classifies customers into various segments based on specific criteria.
* The **Promotion\_Segment junction entity** links promotions to customer segments while recording priority levels and communication channels.

* The **Customer entity** maintains customer information and loyalty points.
* The **CS\_Membership junction entity** manages memberships by connecting customers to their respective segments.
* Finally, the **LoyaltyProgram entity** records loyalty points, tiers, and joining dates for customers participating in the loyalty program.

·

# **Business Rules**

In terms of relationship types:

* A **Promotion** has a **one-to-many** relationship with **MarketingEvent**, **AdSchedule**, **Feedback**, and **Voucher**, meaning one promotion can link to many events, advertisements, customer feedback entries, and vouchers.
* **Promotion** also maintains **many-to-many** relationships with both **PromoItem** and **CustomerSegment**, with the connections managed through the **Item\_Detail** and **Promotion\_Segment** junction tables respectively.
* Each **Promotion** has a **one-to-one** relationship with **PromoAnalytics**, ensuring that every promotion is linked to a unique performance analysis.
* A **Voucher** maintains a **one-to-many** relationship with **PromoItem** through the **Voucher\_Item** junction table, allowing multiple items to be associated with a single voucher.

* A **Customer** has a **one-to-many** relationship with **Feedback**, allowing each customer to submit multiple feedback entries.

* Each **Customer** also has a **one-to-one** relationship with the **LoyaltyProgram**, meaning each customer can be linked to only one loyalty program record.
* Customers maintain a **many-to-many** relationship with **CustomerSegment**, managed through the **CS\_Membership** junction table, which allows customers to belong to multiple segments.

* **Promotion\_Segment** has a **many-to-one** relationship with **CustomerSegment**, meaning multiple promotions can target a single customer segment.

* Finally, **PromoAnalytics** has a **one-to-many** relationship with **Feedback**, meaning a single analytics entry can be related to multiple feedback records, helping track and analyze the effectiveness of promotions based on customer responses.

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity 1** | **Relationship** | **Entity 2** | **Relationship Type** |
| Promotion | has many | MarketingEvent | 1:N (One-to-Many) |
| Promotion | has many | AdSchedule | 1:N (One-to-Many) |
| Promotion | has many | Feedback | 1:N (One-to-Many) |
| Promotion | has many | Voucher | 1:N (One-to-Many) |
| Promotion | linked to many | PromoItem (via Item\_Detail) | M:N (Many-to-Many) |
| Promotion | linked to many | CustomerSegment (via Promotion\_Segment) | M:N (Many-to-Many) |
| Promotion | linked to one | PromoAnalytics | 1:1 (One-to-One) |
| Voucher | linked to many | PromoItem (via Voucher\_Item) | 1:N (One-to-Many) |
| Customer | submits many | Feedback | 1:N (One-to-Many) |
| Customer | belongs to one | LoyaltyProgram | 1:1 (One-to-One) |
| Customer | linked to many | CustomerSegment (via CS\_Membership) | M:N (Many-to-Many) |
| Promotion\_Segment | targets one | CustomerSegment | M:1 (Many-to-One) |
| PromoAnalytics | related to many | Feedback | 1:N (One-to-Many) |

To ensure data integrity and proper enforcement of business rules, the following **constraints** have been applied throughout the database design:

* **Primary Key (PK)** constraints have been assigned to every entity to uniquely identify each record.

For example, CustomerID is the primary key for the Customer table, and PromoID is the primary key for Promotion, ensuring that no duplicate promotions or customers can exist.

* **Foreign Key (FK)** constraints enforce referential integrity between related tables.  
   For instance, PromoID in MarketingEvent, AdSchedule, Voucher, PromoItem, and PromoAnalytics ensures that related records must correspond to an existing promotion.
* **Composite Primary Keys** have been applied for junction tables such as Item\_Detail, Voucher\_Item, Promotion\_Segment, and CS\_Membership, ensuring the correct linkage between related entities.
* **NOT NULL** constraints are placed on important mandatory fields such as Promotion.Title, PromoItem.ItemType, and Voucher.Code to prevent incomplete records from being entered.
* **CHECK constraints** are applied where needed, for example:

Feedback.Rating must be between 1 and 5 to ensure valid customer feedback scores.

Voucher.UsedFlag must be either 'Y' or 'N' to indicate whether the voucher has been redeemed.

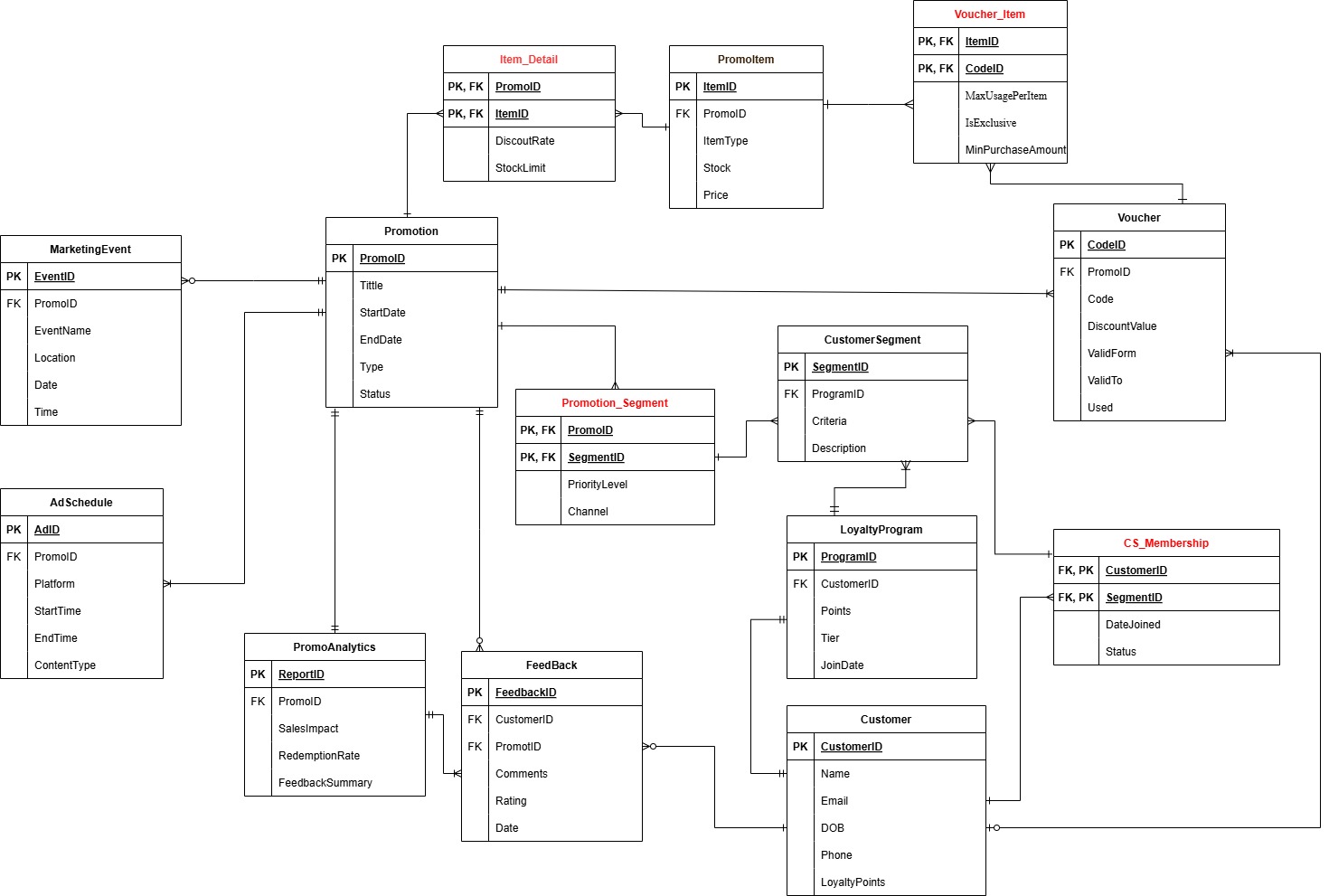
* **Default values** are set for specific fields to support business logic:

Promotion.Status defaults to 'PLANNED' to represent newly created promotions.

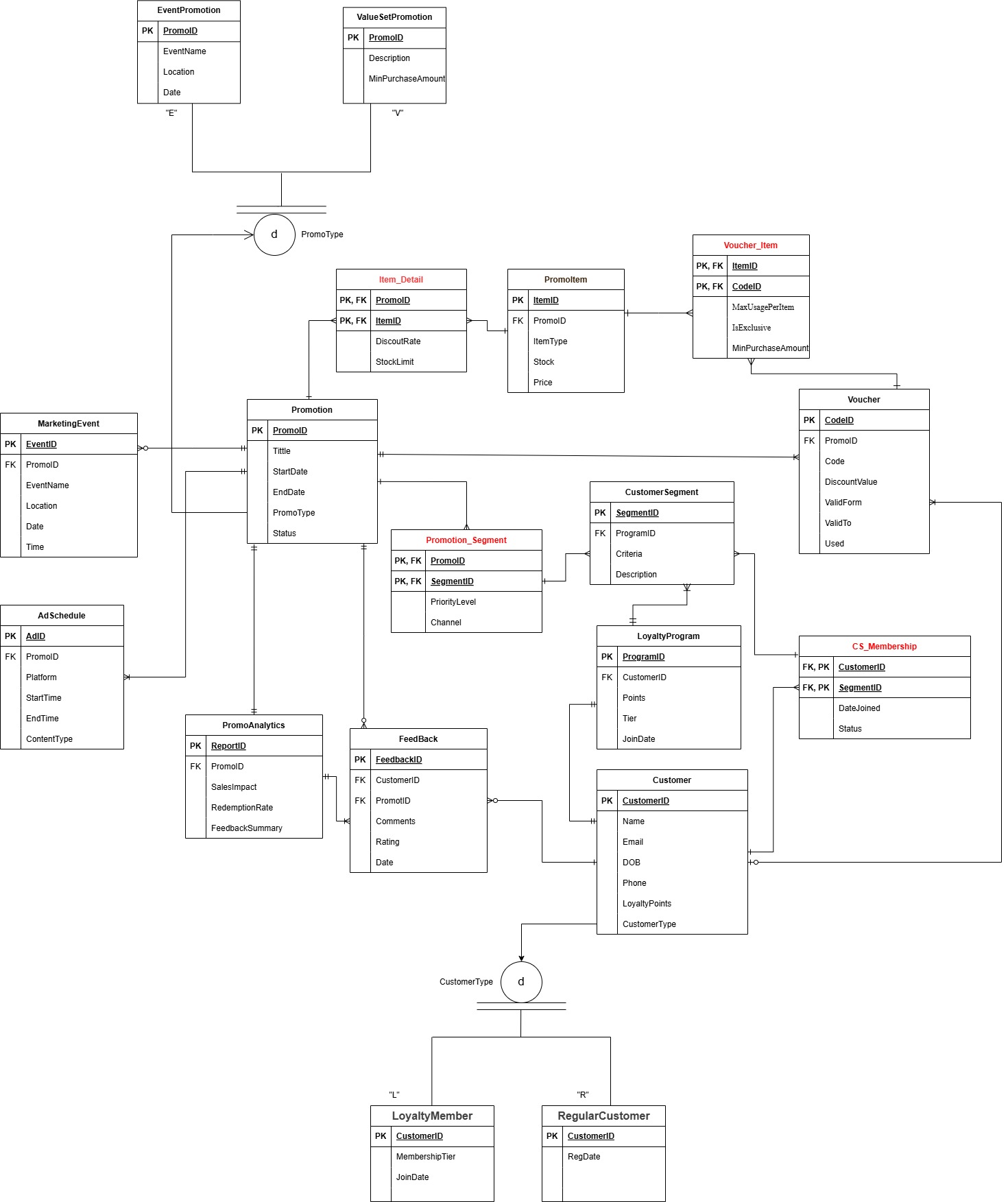
Customer.LoyaltyPoints defaults to 0 for new customers without points.

* **Specialization tables** (LoyaltyMember and RegularCustomer) inherit from the Customer entity based on the CustomerType attribute, ensuring customers are correctly categorized.

**2. ER model**



**3. Redesign and EER**



**4. Data Dictionary (based on EER)**

## **CUSTOMER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| CustomerID | NUMBER | 10 | PK | PRIMARY KEY | Unique customer ID |
| Name | VARCHAR2 | 100 |  | NOT NULL | Customer full name |
| Email | VARCHAR2 | 100 |  | UNIQUE, NOT NULL | Customer email address |
| DOB | DATE | - |  |  | Date of birth |
| Phone | VARCHAR2 | 20 |  |  | Customer phone number |
| LoyaltyPoints | NUMBER | 5 |  | DEFAULT 0 | Loyalty points accumulated |
| CustomerType | VARCHAR2 | 20 |  |  | Type of customer (L/R) |

## **LOYALTYMEMBER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| CustomerID | NUMBER | 10 | PK/FK | PRIMARY KEY, FK→Customer | Link to Customer |
| MembershipTier | VARCHAR2 | 20 |  | NOT NULL | Customer tier (Gold, Silver) |
| JoinDate | DATE | - |  | DEFAULT SYSDATE | Loyalty member join date |

## **REGULARCUSTOMER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| CustomerID | NUMBER | 10 | PK/FK | PRIMARY KEY, FK→Customer | Link to Customer |
| RegDate | DATE | - |  | DEFAULT SYSDATE | Regular customer join date |

## **LOYALTYPROGRAM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| ProgramID | NUMBER | 10 | PK | PRIMARY KEY | Unique loyalty program ID |
| CustomerID | NUMBER | 10 | FK | FK→Customer | Linked customer ID |
| Points | NUMBER | 5 |  | DEFAULT 0 | Points earned |
| Tier | VARCHAR2 | 20 |  |  | Customer tier level |
| JoinDate | DATE | - |  | DEFAULT SYSDATE | Loyalty program join date |

## **PROMOTION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| PromoID | NUMBER | 10 | PK | PRIMARY KEY | Unique promotion ID |
| Title | VARCHAR2 | 150 |  | NOT NULL | Promotion title |
| StartDate | DATE | - |  | NOT NULL | Start date |
| EndDate | DATE | - |  | NOT NULL | End date |
| PromoType | VARCHAR2 | 50 |  |  | Type of promotion (Item, Voucher, Event) |
| Status | VARCHAR2 | 20 |  | DEFAULT 'PLANNED' | Current promotion status |

## **EVENT\_PROMOTION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| PromoID | NUMBER | 10 | PK/FK | PRIMARY KEY, FK→Promotion | Unique promotion ID |
| EventName | VARCHAR2 | 100 |  | NOT NULL | Name of the event |
| Location | VARCHAR2 | 100 |  |  | Location of event |
| Date | DATE | - |  |  | Date of event |

## **VALUE\_SET\_PROMOTION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| PromoID | NUMBER | 10 | PK/FK | PRIMARY KEY, FK→Promotion | Unique promotion ID |
| Description | VARCHAR2 | 150 |  |  | Description of value set |
| MinPurchaseAmount | NUMBER | 10,2 |  |  | Minimum purchase required |

## **PROMOITEM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| ItemID | NUMBER | 10 | PK | PRIMARY KEY | Unique promo item ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| ItemType | VARCHAR2 | 100 |  |  | Item type (Burger, Fries, etc.) |
| Stock | VARCHAR2 | 10 |  |  | Stock available |
| Price | NUMBER | 5,2 |  | DEFAULT 0 | Price of item |

## **ITEM\_DETAIL**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| PromoID | NUMBER | 10 | PK/FK | FK→Promotion | Promotion ID |
| ItemID | NUMBER | 10 | PK/FK | FK→PromoItem | Promo item ID |
| DiscountRate | NUMBER | 5,2 |  | DEFAULT 0 | Discount rate (%) |
| StockLimit | VARCHAR2 | 10 |  |  | Stock limit |

## **VOUCHER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| CodeID | NUMBER | 10 | PK | PRIMARY KEY | Unique voucher ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| Code | VARCHAR2 | 20 |  | UNIQUE, NOT NULL | Voucher code |
| DiscountValue | NUMBER | 5,2 |  |  | Discount percentage |
| ValidFrom | DATE | - |  |  | Voucher valid from date |
| ValidTo | DATE | - |  |  | Voucher valid to date |
| Used | CHAR | 1 |  | DEFAULT 'N' | 'Y' or 'N' (used or not used) |

## **VOUCHER\_ITEM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| ItemID | NUMBER | 10 | PK/FK | FK→PromoItem | Linked item ID |
| CodeID | NUMBER | 10 | PK/FK | FK→Voucher | Linked voucher code ID |
| MaxUsagePerItem | NUMBER | 5 |  | DEFAULT 1 | Maximum allowed per item |
| IsExclusive | CHAR | 1 |  | DEFAULT 'N' | Exclusive voucher (Y/N) |
| MinPurchaseAmount | NUMBER | 10,2 |  |  | Minimum purchase amount |

## **MARKETINGEVENT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| EventID | NUMBER | 10 | PK | PRIMARY KEY | Unique event ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| EventName | VARCHAR2 | 100 |  |  | Event name |
| Location | VARCHAR2 | 100 |  |  | Location |
| EventDate | DATE | - |  |  | Date of event |
| EventTime | TIMESTAMP | - |  |  | Time of event |

## **ADSCHEDULE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| AdID | NUMBER | 10 | PK | PRIMARY KEY | Unique ad ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| Platform | VARCHAR2 | 50 |  |  | Platform (Facebook, Insta) |
| StartTime | TIMESTAMP | - |  |  | Start time of advertisement |
| EndTime | TIMESTAMP | - |  |  | End time of advertisement |
| ContentType | VARCHAR2 | 50 |  |  | Banner, Story, Video |

## **PROMOANALYTICS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| ReportID | NUMBER | 10 | PK | PRIMARY KEY | Unique report ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| SalesImpact | VARCHAR2 | 50 |  |  | Sales impact description |
| RedemptionRate | NUMBER | 5,2 |  |  | Redemption rate (%) |
| FeedbackSummary | VARCHAR2 | 200 |  |  | Summary of feedback |

## **FEEDBACK**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| FeedbackID | NUMBER | 10 | PK | PRIMARY KEY | Unique feedback ID |
| CustomerID | NUMBER | 10 | FK | FK→Customer | Linked customer ID |
| PromoID | NUMBER | 10 | FK | FK→Promotion | Linked promotion ID |
| Comments | VARCHAR2 | 500 |  |  | Customer comments |
| Rating | NUMBER | 1 |  | CHECK (Rating BETWEEN 1 AND 5) | Rating score (1-5) |
| FeedbackDate | DATE | - |  | DEFAULT SYSDATE | Feedback date |

## **CUSTOMERSEGMENT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| SegmentID | NUMBER | 10 | PK | PRIMARY KEY | Unique segment ID |
| ProgramID | NUMBER | 10 | FK | FK→LoyaltyProgram | Linked loyalty program ID |
| Criteria | VARCHAR2 | 100 |  |  | Segmentation criteria |
| Description | VARCHAR2 | 400 |  |  | Segment description |

## **PROMOTION\_SEGMENT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| PromoID | NUMBER | 10 | PK/FK | FK→Promotion | Linked promotion ID |
| SegmentID | NUMBER | 10 | PK/FK | FK→CustomerSegment | Linked segment ID |
| PriorityLevel | NUMBER | 1 |  | DEFAULT 0 | Priority level for communication |
| Channel | VARCHAR2 | 50 |  |  | Communication channel (Email, SMS) |

## **CS\_MEMBERSHIP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Field Length** | **PK/FK** | **Constraints** | **Description** |
| CustomerID | NUMBER | 10 | PK/FK | FK→Customer | Linked customer ID |
| SegmentID | NUMBER | 10 | PK/FK | FK→CustomerSegment | Linked segment ID |
| DateJoined | DATE | - |  | DEFAULT SYSDATE | Date of joining |
| Status | VARCHAR2 | 20 |  |  | Active/Pending status |

**5. Tables and records (based on ERD)**

## **CUSTOMER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CustomerID** | **Name** | **Email** | **DOB** | **Phone** | **LoyaltyPoints** |
| 1 | Alice Tan | [alice.tan@gmail.com](mailto:alice.tan@gmail.com) | 1990-04-12 | 012-3456789 | 100 |
| 2 | Bob Lim | [bob.lim@yahoo.com](mailto:bob.lim@yahoo.com) | 1985-11-05 | 013-9876543 | 250 |
| 3 | Chen Wei | [chen.wei@hotmail.com](mailto:chen.wei@hotmail.com) | 1992-06-30 | 014-1112233 | 50 |
| 4 | Dina Rossi | [dina.rossi@gmail.com](mailto:dina.rossi@gmail.com) | 1988-12-20 | 015-4445566 | 0 |
| 5 | Evan Lee | [evan.lee@yahoo.com](mailto:evan.lee@yahoo.com) | 1995-02-14 | 016-7778899 | 300 |
| 6 | Fiona Ng | [fiona.ng@gmail.com](mailto:fiona.ng@gmail.com) | 1991-07-22 | 017-8887766 | 120 |
| 7 | Gavin Tan | [gavin.tan@hotmail.com](mailto:gavin.tan@hotmail.com) | 1987-03-15 | 018-3322110 | 80 |
| 8 | Hana Lim | [hana.lim@yahoo.com](mailto:hana.lim@yahoo.com) | 1993-09-05 | 019-9988776 | 200 |
| 9 | Ivan Choo | [ivan.choo@gmail.com](mailto:ivan.choo@gmail.com) | 1989-01-10 | 011-6677889 | 40 |
| 10 | Jasmine Ho | [jasmine.ho@hotmail.com](mailto:jasmine.ho@hotmail.com) | 1996-11-25 | 012-5566778 | 180 |
| 11 | Kelvin Teo | [kelvin.teo@gmail.com](mailto:kelvin.teo@gmail.com) | 1984-04-30 | 013-4455667 | 300 |
| 12 | Laura Goh | [laura.goh@yahoo.com](mailto:laura.goh@yahoo.com) | 1992-08-18 | 014-2233445 | 70 |
| 13 | Marcus Lee | [marcus.lee@gmail.com](mailto:marcus.lee@gmail.com) | 1990-05-01 | 015-5566778 | 90 |
| 14 | Natalie Tan | [natalie.tan@hotmail.com](mailto:natalie.tan@hotmail.com) | 1995-12-15 | 016-3344556 | 210 |
| 15 | Oscar Wong | [oscar.wong@gmail.com](mailto:oscar.wong@gmail.com) | 1988-02-20 | 017-8899001 | 60 |
| 16 | Pauline Cheng | [pauline.cheng@yahoo.com](mailto:pauline.cheng@yahoo.com) | 1991-06-08 | 018-6677885 | 130 |
| 17 | Quentin Ong | [quentin.ong@hotmail.com](mailto:quentin.ong@hotmail.com) | 1994-10-11 | 019-7766554 | 20 |
| 18 | Rachel Yap | [rachel.yap@gmail.com](mailto:rachel.yap@gmail.com) | 1993-01-17 | 011-3344557 | 110 |
| 19 | Samuel Low | [samuel.low@gmail.com](mailto:samuel.low@gmail.com) | 1987-08-22 | 012-4455667 | 240 |
| 20 | Tina Khoo | [tina.khoo@hotmail.com](mailto:tina.khoo@hotmail.com) | 1995-09-30 | 013-7788990 | 150 |

## **LOYALTYPROGRAM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ProgramID** | **CustomerID** | **Points** | **Tier** | **JoinDate** |
| 10 | 1 | 100 | Silver | 2024-01-01 |
| 11 | 2 | 250 | Gold | 2023-05-15 |
| 12 | 3 | 50 | Bronze | 2024-03-10 |
| 13 | 4 | 0 | Bronze | 2025-02-01 |
| 14 | 5 | 300 | Gold | 2023-10-20 |
| 15 | 6 | 120 | Silver | 2024-02-05 |
| 16 | 7 | 80 | Bronze | 2024-01-18 |
| 17 | 8 | 200 | Silver | 2023-11-11 |
| 18 | 9 | 40 | Bronze | 2024-04-12 |
| 19 | 10 | 180 | Silver | 2023-06-08 |
| 20 | 11 | 300 | Gold | 2022-12-30 |
| 21 | 12 | 70 | Bronze | 2024-02-20 |
| 22 | 13 | 90 | Bronze | 2023-08-15 |
| 23 | 14 | 210 | Gold | 2023-03-01 |
| 24 | 15 | 60 | Bronze | 2024-05-19 |
| 25 | 16 | 130 | Silver | 2023-09-07 |
| 26 | 17 | 20 | Bronze | 2024-06-22 |
| 27 | 18 | 110 | Silver | 2023-11-18 |
| 28 | 19 | 240 | Gold | 2023-01-29 |
| 29 | 20 | 150 | Silver | 2023-07-14 |

## **PROMOTION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Promo ID** | **Title** | **Start Date** | **End Date** | **Type** | **Status** |
| 100 | Spring Fling | 2025-03-01 | 2025-03-31 | Voucher | ACTIVE |
| 101 | Burger Week | 2025-04-01 | 2025-04-07 | Item | PLANNED |
| 102 | Kids Workshop | 2025-05-10 | 2025-05-10 | Event | PLANNED |
| 103 | Summer Combo | 2025-06-01 | 2025-06-30 | Voucher | PLANNED |
| 104 | Mother’s Day Special | 2025-05-01 | 2025-05-31 | Event | PLANNED |
| 105 | Holiday Bonanza | 2025-11-01 | 2025-11-30 | Voucher | PLANNED |
| 106 | Family Day | 2025-08-15 | 2025-08-15 | Event | PLANNED |
| 107 | Tech Gadget Week | 2025-09-05 | 2025-09-12 | Item | PLANNED |
| 108 | Back to School | 2025-07-25 | 2025-08-10 | Voucher | ACTIVE |
| 109 | Movie Mania | 2025-10-01 | 2025-10-15 | Item | PLANNED |
| 110 | Halloween Bash | 2025-10-25 | 2025-10-31 | Event | PLANNED |
| 111 | Black Friday Sale | 2025-11-28 | 2025-11-30 | Voucher | PLANNED |
| 112 | Cyber Monday | 2025-12-02 | 2025-12-02 | Voucher | PLANNED |
| 113 | New Year Gala | 2025-12-31 | 2026-01-01 | Event | PLANNED |
| 114 | Online Flash Sale | 2025-08-05 | 2025-08-05 | Item | PLANNED |
| 115 | Early Bird Discount | 2025-05-01 | 2025-05-03 | Voucher | PLANNED |
| 116 | Ramadan Special | 2025-03-15 | 2025-04-15 | Voucher | PLANNED |
| 117 | Mother’s Day Treat | 2025-05-10 | 2025-05-10 | Event | PLANNED |
| 118 | Fathers Appreciation | 2025-06-20 | 2025-06-20 | Event | PLANNED |
| 119 | Mega Online Week | 2025-07-20 | 2025-07-27 | Item | PLANNED |

## **PROMOITEM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item ID** | **Promo ID** | **Item Type** | **Stock** | **Price** |
| 200 | 101 | Burger | 500 | 5.50 |
| 201 | 101 | Fries | 500 | 2.00 |
| 202 | 104 | Online Burger | 999 | 4.50 |
| 203 | 104 | Online Drink | 999 | 1.50 |
| 204 | 100 | Value Set | 999 | 8.00 |
| 205 | 105 | Holiday Set | 100 | 12.00 |
| 206 | 107 | Tech Bag | 50 | 45.00 |
| 207 | 107 | Wireless Earbuds | 80 | 70.00 |
| 208 | 108 | Backpack | 200 | 40.00 |
| 209 | 109 | Movie Ticket | 300 | 8.00 |
| 210 | 109 | Popcorn Combo | 400 | 5.00 |
| 211 | 114 | Flash Drive | 150 | 10.00 |
| 212 | 114 | Keyboard | 70 | 25.00 |
| 213 | 119 | Online Headphones | 999 | 55.00 |
| 214 | 119 | Bluetooth Speaker | 999 | 75.00 |
| 215 | 100 | Coupon Deal | 999 | 3.00 |
| 216 | 103 | Meal Combo | 150 | 7.50 |
| 217 | 105 | Gift Hamper | 60 | 50.00 |
| 218 | 107 | Smartwatch | 30 | 120.00 |
| 219 | 119 | Gaming Mouse | 90 | 60.00 |

## **ITEM\_DETAIL**

|  |  |  |  |
| --- | --- | --- | --- |
| **Promo ID** | **Item ID** | **Discount Rate** | **Stock Limit** |
| 101 | 200 | 10 | 100 |
| 101 | 201 | 0 | 300 |
| 104 | 202 | 15 | 999 |
| 104 | 203 | 20 | 999 |
| 100 | 204 | 5 | 50 |
| 105 | 205 | 8 | 120 |
| 107 | 206 | 12 | 60 |
| 107 | 207 | 15 | 100 |
| 108 | 208 | 10 | 250 |
| 109 | 209 | 5 | 400 |
| 109 | 210 | 0 | 500 |
| 114 | 211 | 20 | 180 |
| 114 | 212 | 10 | 90 |
| 119 | 213 | 18 | 999 |
| 119 | 214 | 15 | 999 |
| 100 | 215 | 5 | 100 |
| 103 | 216 | 10 | 130 |
| 105 | 217 | 8 | 70 |
| 107 | 218 | 20 | 40 |
| 119 | 219 | 12 | 100 |

## **VOUCHER**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Code ID** | **Promo ID** | **Code** | **Discount Value** | **ValidFrom** | **ValidTo** | **Used** |
| 300 | 100 | SPRING25 | 25 | 2025-03-01 | 2025-03-31 | N |
| 301 | 103 | SUMMER10 | 10 | 2025-06-01 | 2025-06-30 | N |
| 302 | 100 | SPRING50 | 50 | 2025-03-01 | 2025-03-15 | N |
| 303 | 100 | SPRING5 | 5 | 2025-03-10 | 2025-03-31 | N |
| 304 | 103 | SUMMER20 | 20 | 2025-06-01 | 2025-06-30 | N |
| 305 | 105 | HOLIDAY30 | 30 | 2025-11-01 | 2025-11-30 | N |
| 306 | 108 | BACK2SCHOOL | 15 | 2025-07-25 | 2025-08-10 | N |
| 307 | 111 | BLACKFRIDAY50 | 50 | 2025-11-28 | 2025-11-30 | N |
| 308 | 112 | CYBERMONDAY20 | 20 | 2025-12-02 | 2025-12-02 | N |
| 309 | 115 | EARLY10 | 10 | 2025-05-01 | 2025-05-03 | N |
| 310 | 116 | RAMADAN15 | 15 | 2025-03-15 | 2025-04-15 | N |
| 311 | 119 | MEGAWEEK30 | 30 | 2025-07-20 | 2025-07-27 | N |
| 312 | 105 | FESTIVE40 | 40 | 2025-11-01 | 2025-11-30 | N |
| 313 | 100 | SPRING15 | 15 | 2025-03-01 | 2025-03-31 | N |
| 314 | 103 | SUMMER5 | 5 | 2025-06-01 | 2025-06-30 | N |
| 315 | 116 | RAMADAN5 | 5 | 2025-03-15 | 2025-04-15 | N |
| 316 | 119 | ONLINE10 | 10 | 2025-07-20 | 2025-07-27 | N |
| 317 | 105 | HOLIDAY10 | 10 | 2025-11-01 | 2025-11-30 | N |
| 318 | 115 | EARLY5 | 5 | 2025-05-01 | 2025-05-03 | N |
| 319 | 119 | FLASH20 | 20 | 2025-07-20 | 2025-07-27 | N |

## **VOUCHER\_ITEM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item ID** | **Code ID** | **MaxUsagePerItem** | **IsExclusive** | **MinPurchaseAmount** |
| 204 | 300 | 1 | Y | 20 |
| 204 | 302 | 1 | N | 0 |
| 200 | 303 | 2 | N | 10 |
| 202 | 301 | 1 | Y | 15 |
| 203 | 304 | 1 | N | 0 |
| 205 | 305 | 1 | Y | 25 |
| 208 | 306 | 1 | N | 5 |
| 209 | 307 | 1 | Y | 30 |
| 210 | 307 | 2 | N | 20 |
| 211 | 308 | 1 | Y | 10 |
| 212 | 308 | 1 | N | 5 |
| 213 | 311 | 1 | Y | 20 |
| 214 | 311 | 1 | N | 15 |
| 215 | 313 | 2 | N | 5 |
| 216 | 314 | 1 | Y | 5 |
| 217 | 312 | 1 | Y | 40 |
| 218 | 316 | 1 | N | 10 |
| 219 | 319 | 1 | Y | 15 |
| 206 | 317 | 1 | N | 20 |
| 207 | 318 | 1 | Y | 10 |

## **MARKETINGEVENT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event ID** | **PromoID** | **Event Name** | **Location** | **Event Date** | **Event Time** |
| 400 | 100 | Spring Launch Party | Mall A | 2025-03-01 | 2025-03-01 14:00:00 |
| 401 | 101 | Burger Festival | Mall B | 2025-04-01 | 2025-04-01 12:00:00 |
| 402 | 102 | Kids Activity Zone | Park C | 2025-05-10 | 2025-05-10 10:00:00 |
| 403 | 103 | Summer Combo Party | Beach D | 2025-06-05 | 2025-06-05 15:00:00 |
| 404 | 104 | Mother’s Day Special | Mall E | 2025-05-12 | 2025-05-12 11:00:00 |
| 405 | 100 | Spring Contest | Outlet A | 2025-03-02 | 2025-03-02 10:00:00 |
| 406 | 101 | Burger Eating Contest | Outlet B | 2025-04-02 | 2025-04-02 13:00:00 |
| 407 | 102 | Kids Coloring | Mall C | 2025-05-11 | 2025-05-11 11:00:00 |
| 408 | 103 | Summer Dance | Beach F | 2025-06-06 | 2025-06-06 16:00:00 |
| 409 | 104 | Mom and Me Special | Mall G | 2025-05-13 | 2025-05-13 10:00:00 |
| 410 | 100 | Spring Pop-up | Pop-up Store | 2025-03-03 | 2025-03-03 12:00:00 |
| 411 | 101 | Burger Combo Fiesta | Mall H | 2025-04-03 | 2025-04-03 11:00:00 |
| 412 | 102 | Kids Fun Walk | Park I | 2025-05-12 | 2025-05-12 09:00:00 |
| 413 | 103 | Beach Carnival | Beach J | 2025-06-07 | 2025-06-07 17:00:00 |
| 414 | 104 | Mother’s Day Tea | Hotel K | 2025-05-14 | 2025-05-14 14:00:00 |
| 415 | 100 | Spring Surprise | Mall L | 2025-03-04 | 2025-03-04 13:00:00 |
| 416 | 101 | Burger Launch | Restaurant M | 2025-04-04 | 2025-04-04 14:00:00 |
| 417 | 102 | Kids Party | Mall N | 2025-05-13 | 2025-05-13 15:00:00 |
| 418 | 103 | Summer BBQ | Beach O | 2025-06-08 | 2025-06-08 18:00:00 |
| 419 | 104 | Mom Gift Fair | Mall P | 2025-05-15 | 2025-05-15 12:00:00 |

## **ADSCHEDULE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AdID** | **PromoID** | **Platform** | **StartTime** | **EndTime** | **ContentType** |
| 500 | 100 | Facebook | 2025-02-25 00:00:00 | 2025-03-01 23:59:59 | Banner |
| 501 | 101 | Instagram | 2025-03-28 00:00:00 | 2025-04-07 23:59:59 | Story |
| 502 | 102 | Billboard | 2025-05-01 00:00:00 | 2025-05-10 23:59:59 | Video |
| 503 | 103 | YouTube | 2025-05-25 00:00:00 | 2025-06-10 23:59:59 | Video |
| 504 | 104 | Push Notification | 2025-05-01 00:00:00 | 2025-05-10 23:59:59 | Popup |
| 505 | 100 | Radio | 2025-02-20 00:00:00 | 2025-03-01 23:59:59 | Audio |
| 506 | 101 | In-App Banner | 2025-03-29 00:00:00 | 2025-04-06 23:59:59 | Popup |
| 507 | 102 | Kids TV | 2025-05-02 00:00:00 | 2025-05-10 23:59:59 | Video |
| 508 | 103 | Summer Billboard | 2025-06-01 00:00:00 | 2025-06-30 23:59:59 | Banner |
| 509 | 104 | Mothers App | 2025-05-05 00:00:00 | 2025-05-12 23:59:59 | Story |
| 510 | 100 | Instagram Reel | 2025-02-26 00:00:00 | 2025-03-01 23:59:59 | Video |
| 511 | 101 | Website Promo | 2025-03-30 00:00:00 | 2025-04-05 23:59:59 | Banner |
| 512 | 102 | Kids Online | 2025-05-03 00:00:00 | 2025-05-11 23:59:59 | Popup |
| 513 | 103 | YouTube Summer | 2025-05-27 00:00:00 | 2025-06-10 23:59:59 | Video |
| 514 | 104 | Gift Push | 2025-05-02 00:00:00 | 2025-05-10 23:59:59 | Popup |
| 515 | 100 | Outdoor Banners | 2025-02-27 00:00:00 | 2025-03-01 23:59:59 | Banner |
| 516 | 101 | Email Campaign | 2025-03-31 00:00:00 | 2025-04-06 23:59:59 | Text |
| 517 | 102 | Online Contest | 2025-05-04 00:00:00 | 2025-05-12 23:59:59 | Popup |
| 518 | 103 | Summer Radio | 2025-06-02 00:00:00 | 2025-06-15 23:59:59 | Audio |
| 519 | 104 | Special Ads | 2025-05-06 00:00:00 | 2025-05-12 23:59:59 | Story |

## **PROMOANALYTICS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ReportID** | **PromoID** | **SalesImpact** | **RedemptionRate** | **FeedbackSummary** |
| 600 | 100 | +15% sales | 12.5 | Mostly positive |
| 601 | 101 | +8% sales | 5.0 | Good uptake |
| 602 | 102 | N/A | 0.0 | Hands-on feedback only |
| 603 | 103 | +10% sales | 8.8 | Mixed reviews |
| 604 | 104 | +20% web orders | 15.0 | Excellent |
| 605 | 100 | +5% social engagement | 4.5 | Okay response |
| 606 | 101 | +6% customer traffic | 5.8 | Encouraging |
| 607 | 102 | +7% family signups | 6.2 | Positive |
| 608 | 103 | +12% combo meals sold | 10.0 | Great |
| 609 | 104 | +9% gifts purchased | 7.5 | Good |
| 610 | 100 | +11% app installs | 9.8 | Very good |
| 611 | 101 | +13% burger orders | 11.5 | Very happy |
| 612 | 102 | +4% event check-ins | 3.5 | Average |
| 613 | 103 | +6% ice cream sales | 5.7 | Fair |
| 614 | 104 | +15% restaurant bookings | 14.5 | Excellent |
| 615 | 100 | +3% voucher use | 2.9 | Mild |
| 616 | 101 | +8% burger combo redemptions | 7.2 | Good |
| 617 | 102 | +5% kids feedback | 4.8 | Positive |
| 618 | 103 | +10% customer returns | 8.9 | Very good |
| 619 | 104 | +12% Mother’s Day sales | 10.5 | Fantastic |

## **FEEDBACK**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FeedbackID** | **CustomerID** | **PromoID** | **FeedbackDate** | **Rating** | **Comments** |
| 700 | 1 | 100 | 2025-03-05 | 5 | Great deal! |
| 701 | 2 | 100 | 2025-03-10 | 3 | - |
| 702 | 3 | 101 | 2025-04-03 | 5 | Loved the burger |
| 703 | 4 | 103 | 2025-06-10 | 2 | Not enough items |
| 704 | 5 | 104 | 2025-07-05 | 4 | Very convenient |
| 705 | 6 | 100 | 2025-03-07 | 5 | Spring event was fun |
| 706 | 7 | 101 | 2025-04-04 | 3 | Burger could be bigger |
| 707 | 8 | 102 | 2025-05-10 | 5 | Kids loved the games |
| 708 | 9 | 102 | 2025-05-10 | 4 | Nice family event |
| 709 | 10 | 103 | 2025-06-02 | 5 | Combo was worth it |
| 710 | 11 | 103 | 2025-06-12 | 3 | More variety needed |
| 711 | 12 | 104 | 2025-05-08 | 5 | Perfect Mother's Day gift |
| 712 | 13 | 100 | 2025-03-15 | 3 | Spring promo average |
| 713 | 14 | 101 | 2025-04-06 | 4 | Burger week was tasty |
| 714 | 15 | 102 | 2025-05-11 | 5 | Enjoyed kids activities |
| 715 | 16 | 104 | 2025-05-09 | 5 | Gift fair awesome |
| 716 | 17 | 100 | 2025-03-08 | 5 | Love the vouchers |
| 717 | 18 | 103 | 2025-06-04 | 4 | Good deals overall |
| 718 | 19 | 101 | 2025-04-05 | 3 | Burger deal fair |
| 719 | 20 | 104 | 2025-05-10 | 5 | Mother loved the surprise |

## **CUSTOMERSEGMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **SegmentID** | **ProgramID** | **Criteria** | **Description** |
| 800 | 10 | Points >=100 | Silver and above |
| 801 | 11 | Points >=200 | Gold members |
| 802 | 12 | Joined last 30 days | Newbies |
| 803 | 14 | Tier = Gold | Top tier |
| 804 | 13 | Points = 0 | Inactive |
| 805 | 15 | Points between 50-100 | Moderate spenders |
| 806 | 16 | Points ≥80 | Active members |
| 807 | 17 | Points ≤20 | Low activity |
| 808 | 18 | Joined after Jan 2024 | Recent joiners |
| 809 | 19 | Points between 200-300 | High-value customers |
| 810 | 20 | Points ≥150 | Potential Gold |
| 811 | 10 | Birthday month | Special offer group |
| 812 | 11 | Top 10% Loyalty | VIP members |
| 813 | 12 | Below 100 Points | At-risk |
| 814 | 13 | No purchases | Non-active |
| 815 | 14 | Frequent feedback | Engaged users |
| 816 | 15 | Voucher users | Coupon Lovers |
| 817 | 16 | New parents | Family group |
| 818 | 17 | Students | Young adults |
| 819 | 18 | Seniors | Senior citizens |

## **PROMOTION\_SEGMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **PromoID** | **SegmentID** | **PriorityLevel** | **Channel** |
| 100 | 800 | 1 | Email |
| 100 | 801 | 2 | SMS |
| 101 | 802 | 1 | In-App |
| 102 | 803 | 1 | Email |
| 103 | 804 | 1 | Website |
| 104 | 805 | 1 | Email |
| 100 | 806 | 2 | SMS |
| 101 | 807 | 1 | In-App |
| 102 | 808 | 1 | Email |
| 103 | 809 | 1 | Website |
| 104 | 810 | 1 | Facebook |
| 100 | 811 | 1 | Instagram |
| 101 | 812 | 1 | Direct Call |
| 102 | 813 | 2 | SMS |
| 103 | 814 | 1 | Email |
| 104 | 815 | 2 | Website |
| 100 | 816 | 1 | In-App |
| 101 | 817 | 1 | Email |
| 102 | 818 | 1 | Social Media |
| 103 | 819 | 1 | Email |

## **CS\_MEMBERSHIP**

|  |  |  |  |
| --- | --- | --- | --- |
| **CustomerID** | **SegmentID** | **DateJoined** | **Status** |
| 1 | 800 | 2024-01-01 | Active |
| 2 | 801 | 2023-05-15 | Active |
| 3 | 802 | 2024-03-10 | Active |
| 5 | 803 | 2023-10-20 | Active |
| 4 | 804 | 2025-02-01 | Pending |
| 6 | 805 | 2024-02-15 | Active |
| 7 | 806 | 2024-01-12 | Active |
| 8 | 807 | 2024-04-11 | Active |
| 9 | 808 | 2024-04-25 | Active |
| 10 | 809 | 2023-06-09 | Active |
| 11 | 810 | 2022-11-20 | Active |
| 12 | 811 | 2023-12-30 | Active |
| 13 | 812 | 2024-02-21 | Active |
| 14 | 813 | 2023-07-16 | Inactive |
| 15 | 814 | 2023-08-22 | Inactive |
| 16 | 815 | 2024-04-28 | Active |
| 17 | 816 | 2023-09-15 | Active |
| 18 | 817 | 2024-05-19 | Active |
| 19 | 818 | 2024-05-05 | Active |
| 20 | 819 | 2024-06-11 | Pending |

**6. Script**  
Sql script is included in the submission.

**PART 2: Individual Assessment**

**Member 1: Brian Lee Zhen Hui**

**SQL Query 1: Select the Top 5 customers by loyalty points**

SELECT c.Name, c.Email, c.LoyaltyPoints

FROM Customer c

ORDER BY c.LoyaltyPoints DESC

FETCH FIRST 5 ROWS ONLY;

**Operation Type:** Selection

**Purpose & How User Uses It:** Retrieve the 5 customers who have earned the highest loyalty points, to reward or recognize them. Useful for loyalty program management.

**SQL Query 2: Upcoming promotions next month**

SELECT p.Title, p.StartDate, p.EndDate

FROM Promotion p

WHERE p.StartDate BETWEEN

TRUNC(ADD\_MONTHS(SYSDATE, 1), 'MM')

AND LAST\_DAY(ADD\_MONTHS(SYSDATE, 1))

AND p.Status = 'PLANNED';

**Operation Type:** Selection

**Purpose & How User Uses It:** To find promotions that will start during next month and are in ‘Planned’ status and easy for users to display automatically.

**Procedure 1: Award bonus points to all members of a segment**

CREATE OR REPLACE PROCEDURE Award\_Segment\_Bonus(

p\_SegmentID IN NUMBER,

p\_Bonus IN NUMBER

) AS

BEGIN

UPDATE Customer c

SET c.LoyaltyPoints = c.LoyaltyPoints + p\_Bonus

WHERE c.CustomerID IN (

SELECT cs.CustomerID

FROM CS\_Membership cs

WHERE cs.SegmentID = p\_SegmentID

);

COMMIT;

END Award\_Segment\_Bonus;

/

EXEC Award\_Segment\_Bonus(801, 50);

SELECT c.CustomerID, c.LoyaltyPoints, cs.SegmentID

FROM Customer c ,CS\_Membership cs

Where c.CustomerID = cs.CustomerID;

**Operation Type:** Update

**Purpose & How User Uses It:** Award bonus loyalty points to all customers belonging to a specific segment, useful for targeted reward programs.

**Procedure 2: Reset voucher usage for a given promotion**

CREATE OR REPLACE PROCEDURE Reset\_Vouchers\_For\_Promo( p\_PromoID IN NUMBER ) AS

BEGIN

UPDATE Voucher v

SET v.Used = 'N'

WHERE v.PromoID = p\_PromoID;

COMMIT;

END Reset\_Vouchers\_For\_Promo;

/

EXEC Reset\_Vouchers\_For\_Promo(100);

SELECT \* FROM Voucher WHERE PromoID = 100;

**Operation Type:** Update

**Purpose & How User Uses It:** Reset voucher usage status for all vouchers tied to a specific promotion, allowing the re-use or correction of vouchers.

**Function 1: Calculate days until promotion starts**

CREATE OR REPLACE FUNCTION Days\_Until\_Promo( p\_PromoID IN NUMBER )

RETURN NUMBER

IS

v\_Days NUMBER;

BEGIN

SELECT ABS(p.StartDate - TRUNC(SYSDATE))

INTO v\_Days

FROM Promotion p

WHERE p.PromoID = p\_PromoID;

RETURN v\_Days;

END Days\_Until\_Promo;

/

SELECT Days\_Until\_Promo(100) As “Days Until Promo” FROM DUAL;

**Operation Type:** Calculation (Return)

**Purpose & How User Uses It:** Calculate how many days remain before a promotion starts to inform customers or staff for preparation.

**Function 2: Check if a promotion is currently active (Y/N)**

CREATE OR REPLACE FUNCTION Is\_Promo\_Active( p\_PromoID IN NUMBER ) RETURN VARCHAR2 IS

v\_Status VARCHAR2(1);

v\_Start DATE;

v\_End DATE;

BEGIN

SELECT p.StartDate, p.EndDate

INTO v\_Start, v\_End

FROM Promotion p

WHERE p.PromoID = p\_PromoID;

IF TRUNC(SYSDATE) BETWEEN v\_Start AND v\_End THEN

v\_Status := 'Y';

ELSE

v\_Status := 'N';

END IF;

RETURN v\_Status;

END Is\_Promo\_Active;

/

SELECT Is\_Promo\_Active(100) As “Is Promo Active? Y-Yes N-No” FROM DUAL;

**Operation Type:** Verification (Return)

**Purpose & How User Uses It:** Verify whether a promotion is currently active based on the current date, used for enabling special promotions in systems.

**Member 2: Lee Cheng Jun\***

**SQL Query 1: Top 5 promotions by redemption rate**

SELECT p.Title,

TO\_NUMBER(REPLACE(a.RedemptionRate, '%', '')) AS Rate

FROM Promotion p

JOIN PromoAnalytics a ON p.PromoID = a.PromoID

ORDER BY Rate DESC

FETCH FIRST 5 ROWS ONLY;

**Operation Type:** Selection

**Purpose & How User Uses It:** Retrieve the top 5 promotions with the highest redemption rates to identify successful campaigns.

**SQL Query 2: Number of customers in each segment**

SELECT seg.Description, COUNT(cs.CustomerID) AS MemberCount

FROM CustomerSegment seg

LEFT JOIN CS\_Membership cs ON seg.SegmentID = cs.SegmentID

GROUP BY seg.Description;

**Operation Type:** Aggregation

**Purpose & How User Uses It:** Count the number of customers in each segment to assist in customer segmentation and targeted marketing.

**Procedure 1: Mark all vouchers expired for a given promotion past a date**

CREATE OR REPLACE PROCEDURE Expire\_Vouchers(

p\_PromoID IN NUMBER,

p\_ExpiryDate IN DATE

) AS

BEGIN

UPDATE Voucher

SET Used = 'Y'

WHERE PromoID = p\_PromoID

AND ValidTo < p\_ExpiryDate;

COMMIT;

END Expire\_Vouchers;

/

-- Example call for Expire\_Vouchers:

-- This will mark as used all vouchers for promo 101 that expired before April 1, 2025

BEGIN Expire\_Vouchers(

103,

TO\_DATE('2025-04-01','YYYY-MM-DD')

);

END;

/

-- Verify the update with a SELECT:

SELECT \* FROM Voucher

WHERE PromoID = 103

AND ValidTo < DATE '2025-04-01';

**Operation Type:** Update

**Purpose & How User Uses It:** Mark vouchers as used if they expired before a specified date, ensuring data is clean and up-to-date.

**Procedure 2: Schedule a new event for a promotion**

-- Create the sequence

CREATE SEQUENCE MarketingEvent\_SEQ

START WITH 400

INCREMENT BY 1

NOCACHE

NOCYCLE;

CREATE OR REPLACE PROCEDURE Add\_Marketing\_Event(

p\_PromoID IN NUMBER,

p\_Name IN VARCHAR2,

p\_Location IN VARCHAR2,

p\_Date IN DATE,

p\_Time IN TIMESTAMP

) AS

v\_event\_id NUMBER;

v\_max\_event\_id NUMBER;

BEGIN

-- Get the latest EventID used in MarketingEvent

SELECT MAX(EventID) INTO v\_max\_event\_id FROM MarketingEvent;

-- Get the next value from sequence

SELECT MarketingEvent\_SEQ.NEXTVAL INTO v\_event\_id FROM DUAL;

-- If sequence value is less than or equal to MAX(EventID), loop until it's greater

WHILE v\_event\_id <= v\_max\_event\_id LOOP

SELECT MarketingEvent\_SEQ.NEXTVAL INTO v\_event\_id FROM DUAL;

END LOOP;

-- Insert the new marketing event with the correct EventID

INSERT INTO MarketingEvent (EventID, PromoID, EventName, Location, EventDate, EventTime)

VALUES (v\_event\_id, p\_PromoID, p\_Name, p\_Location, p\_Date, p\_Time);

COMMIT;

END Add\_Marketing\_Event;

/

-- Example call for Add\_Marketing\_Event:

BEGIN

Add\_Marketing\_Event(

109,

'Glorious Event',

'Hall D',

TO\_DATE('2025-06-01','YYYY-MM-DD'),

TO\_TIMESTAMP('2025-06-01 10:00:00','YYYY-MM-DD HH24:MI:SS') );

END;

/

-- Verify the insertion with a SELECT:

SELECT \*

FROM MarketingEvent

WHERE PromoID = 109;

**Operation Type:** Insertion  
**Purpose & How User Uses It:**  
This procedure helps users easily add a new marketing event without worrying about setting the “EventID” themselves. When scheduling a new event, users simply provide details like the promotion, event name, location, date, and time. The system will automatically find the next available event ID, making the process smooth, accurate, and hassle-free.

**Function 1: Get total loyalty points for a customer**

CREATE OR REPLACE FUNCTION Get\_Customer\_Points(p\_CustID IN NUMBER)

RETURN NUMBER

IS

v\_points NUMBER;

BEGIN

SELECT LoyaltyPoints

INTO v\_points

FROM Customer

WHERE CustomerID = p\_CustID;

RETURN v\_points;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_points := 0; -- If customer not found, return 0

RETURN v\_points;

END Get\_Customer\_Points;

/

-- Example call for Get\_Customer\_Points:

-- Calculate total loyalty points for customer 1

SELECT Get\_Customer\_Points(1) AS total\_points

FROM dual;

**Operation Type:** Retrieval (Return)

**Purpose & How User Uses It:** Retrieve the total loyalty points for a customer, useful for loyalty rewards and verification.

**Function 2: Get top feedback (max rating) for a promotion**

CREATE OR REPLACE FUNCTION Get\_Top\_Feedback( p\_PromoID IN NUMBER )

RETURN VARCHAR2

IS

v\_comment VARCHAR2(400);

BEGIN

-- Use a subquery to get the top comment for the promo with the highest rating

SELECT Comments

INTO v\_comment

FROM (

SELECT Comments

FROM Feedback

WHERE PromoID = p\_PromoID

AND Rating = (SELECT MAX(Rating)

FROM Feedback

WHERE PromoID = p\_PromoID)

ORDER BY RATING DESC

)

WHERE ROWNUM = 1; -- Ensures we only get the first row (top feedback)

RETURN v\_comment;

END Get\_Top\_Feedback;

/

-- Example call for Get\_Top\_Feedback:

-- Retrieve the best feedback comment for promotion 101

SELECT Get\_Top\_Feedback(101) AS top\_comment

FROM dual;

**Operation Type:** Retrieval (Return)

**Purpose & How User Uses It:** This function retrieves the best feedback for a promotion based on the highest rating. Users input a “PromoID” to get the top-rated comment, such as:  
“SELECT Get\_Top\_Feedback(101) FROM dual;”

**Member 3: Ng Che Te**

**SQL Query 1: Customers without any feedback**

SELECT c.Name, c.Email, f.Comments, f.Rating

FROM Customer c, Feedback f

WHERE c.CustomerID = f.CustomerID

AND f.Comments IS NULL;

**Operation Type:** Selection

**Purpose & How User Uses It:** Identify customers who have never provided feedback, assisting in planning follow-up or feedback campaigns.

**SQL Query 2: Promotions without analytics records**

SELECT p.PromoID, p.Title

FROM Promotion p

WHERE NOT EXISTS ( SELECT 1 FROM PromoAnalytics a WHERE a.PromoID = p.PromoID );

**Operation Type:** Selection

**Purpose & How User Uses It:** Identify promotions that lack analytics reports to improve tracking and future campaign evaluations.

**Procedure 1: Close old promotions before a given date**

CREATE OR REPLACE PROCEDURE Close\_Old\_Promos(p\_BeforeDate IN DATE) AS v\_count NUMBER; BEGIN -- Count how many promotions will be updated SELECT COUNT(\*) INTO v\_count FROM Promotion WHERE EndDate < p\_BeforeDate AND Status != 'CLOSED';

UPDATE Promotion   
SET Status = 'CLOSED'   
WHERE EndDate < p\_BeforeDate AND Status != 'CLOSED';   
  
   
COMMIT;   
  
  
DBMS\_OUTPUT.PUT\_LINE(v\_count || ' promotion(s) closed before ' || TO\_CHAR(p\_BeforeDate, 'YYYY-MM-DD'));

END Close\_Old\_Promos;

/

SET SERVEROUTPUT ON;

EXEC Close\_Old\_Promos(TO\_DATE('2025-06-01', 'YYYY-MM-DD'));

SELECT PromoID, Title, EndDate, Status

FROM Promotion

WHERE EndDate < TO\_DATE('2025-06-01', 'YYYY-MM-DD');

**Operation Type:** Update

**Purpose & How User Uses It:** Close promotions that ended before a given date to keep the promotion list updated and organized.

**Procedure 2: Sends a reminder email to customers with low loyalty points**

CREATE OR REPLACE PROCEDURE Send\_Low\_Points\_Reminder AS

BEGIN

FOR low IN (

SELECT c.Name, c.Email, c.LoyaltyPoints FROM Customer c JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID WHERE c.LoyaltyPoints < 50 AND cs.Status = 'active'

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Sending reminder to: ' || low.Name || ' (' || low.Email || ')'); DBMS\_OUTPUT.PUT\_LINE('Loyalty Points: ' || low.LoyaltyPoints || ' - Reminder: Earn more points for rewards!'); DBMS\_OUTPUT.PUT\_LINE('--------------------------------------------------------');

END LOOP;

END Send\_Low\_Points\_Reminder;

/

SET SERVEROUTPUT ON;

EXEC Send\_Low\_Points\_Reminder;

SELECT c.CustomerID, c.Name, c.Email, c.LoyaltyPoints, cs.Status

FROM Customer c

JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID

WHERE c.LoyaltyPoints < 50 AND cs.Status = 'Active';

**Operation Type:** Processing (Display)

**Purpose & How User Uses It:** This procedure sends reminders to customers with low loyalty points to encourage them to earn more points and stay engaged in the loyalty program. It’s useful for customer retention and boosting engagement.

**Function 1: Average redemption rate across all promotions**

CREATE OR REPLACE FUNCTION Avg\_Redemption\_Rate

RETURN NUMBER

IS

v\_avg NUMBER;

BEGIN

SELECT AVG(TO\_NUMBER(REPLACE(RedemptionRate, '%', '')))

INTO v\_avg

FROM PromoAnalytics;

RETURN v\_avg;

END Avg\_Redemption\_Rate;

/

SELECT Avg\_Redemption\_Rate() AS avg\_redemption\_rate FROM dual;

**Operation Type:** Aggregation (Return)

**Purpose & How User Uses It:** Calculate the average redemption rate across all promotions to evaluate marketing performance.

**Function 2: Most popular customer segment by membership count**

CREATE OR REPLACE FUNCTION Top\_Customer\_Segment RETURN VARCHAR2 IS

v\_segment VARCHAR2(400);

BEGIN

SELECT seg.Description

INTO v\_segment

FROM (

SELECT SegmentID

FROM CS\_Membership

GROUP BY SegmentID

ORDER BY COUNT(CustomerID) DESC

) best

JOIN CustomerSegment seg ON best.SegmentID = seg.SegmentID

WHERE ROWNUM = 1;

RETURN v\_segment;

END Top\_Customer\_Segment;

/

SELECT Top\_Customer\_Segment() AS top\_segment FROM dual;

**Operation Type:** Aggregation (Return)

**Purpose & How User Uses It:** Find the most popular customer segment based on membership count to focus marketing efforts effectively.

**Member 4: Tan Jia Yi**

**SQL Query 1: List the details of Customer name and comments are in high tier.**

SELECT c.Name, Tier, Comments

FROM Customer c

JOIN FeedBack f

ON c.CustomerID = f.CustomerID

JOIN LoyaltyProgram l

ON c.CustomerID = l.CustomerID

WHERE Tier = 'Gold';

**Operation Type:** Selection

**Purpose & How User Uses It:** Retrieve customer names and feedback comments for customers in a specific high loyalty tier to evaluate valuable customer feedback.

**SQL Query 2: List the promotion type according to the sum of rating, from highest to lower.**

SELECT p.Type, COUNT(f.Rating) AS NumberOfRatings,

SUM(f.Rating) AS TotalRating

FROM Promotion p JOIN Feedback f

ON p.PromoID = f.PromoID

GROUP BY p.Type

ORDER BY TotalRating DESC;

**Operation Type:** Aggregation

**Purpose & How User Uses It:** Display promotion types ranked by total feedback ratings to understand which types of promotions are most successful.

**Procedure 1: List the Customer Feedback base on tier input**

CREATE OR REPLACE PROCEDURE Tier\_Customer\_Feedback(

P\_tier IN VARCHAR2)

IS

BEGIN

FOR cusF IN (

SELECT c.Name, f.Comments, l.Tier

FROM Customer c

JOIN FeedBack f ON c.CustomerID = f.CustomerID

JOIN LoyaltyProgram l ON c.CustomerID = l.CustomerID

WHERE l.Tier = P\_tier

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Name: ' || cusF.Name); DBMS\_OUTPUT.PUT\_LINE('Comments: ' || cusF.Comments); DBMS\_OUTPUT.PUT\_LINE('Tier: ' || cusF.Tier); DBMS\_OUTPUT.PUT\_LINE('-------------------------');

END LOOP;

END;

/

**Operation Type:** Selection and Display

**Purpose & How User Uses It:** List customer feedback based on loyalty tier input to analyze customer satisfaction for different tiers.

**Procedure 2: List the detail of efficient promo base on Redemption Rate.**

CREATE OR REPLACE PROCEDURE Efficient\_Promo

IS

BEGIN

FOR ePro IN (

SELECT p.Title, p.Type, p.Status, a.RedemptionRate

FROM Promotion p

JOIN PromoAnalytics a ON p.PromoID = a.PromoID

WHERE a.RedemptionRate > 10

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Title: ' || ePro.Title); DBMS\_OUTPUT.PUT\_LINE('Type: ' || ePro.Type); DBMS\_OUTPUT.PUT\_LINE('Status: ' || ePro.Status); DBMS\_OUTPUT.PUT\_LINE('Redemption Rate: ' || ePro.RedemptionRate || '%');

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

END LOOP;

END;

/

**Operation Type:** Selection and Display

**Purpose & How User Uses It:** List promotions that show strong redemption rate for management to review effective promotional strategies.

**Function 1: Check voucher by promo tittle that are not used**

CREATE OR REPLACE FUNCTION Check\_Voucher (

p\_Title IN VARCHAR2

)

RETURN VARCHAR2

IS

v\_output VARCHAR2(4000) := '';

BEGIN

FOR vou IN (

SELECT v.CodeID, v.DiscountValue, v.ValidFrom, v.ValidTo, v.Used

FROM Voucher v

JOIN Promotion p ON p.PromoID = v.PromoID

WHERE p.Title = p\_Title

AND v.Used = 'N'

) LOOP

v\_output := v\_output ||

'Code: ' || vou.CodeID || CHR(10) ||

'Discount Value: ' || vou.DiscountValue || CHR(10) ||

'Valid From: ' || TO\_CHAR(vou.ValidFrom, 'DD-MON-YYYY') || CHR(10) ||

'Valid To: ' || TO\_CHAR(vou.ValidTo, 'DD-MON-YYYY') || CHR(10) ||

'Used: ' || vou.Used || CHR(10) || CHR(10);

END LOOP;

IF v\_output IS NULL THEN

v\_output := 'No unused vouchers found for the given title.';

END IF;

RETURN v\_output;

END; /

**Operation Type:** Retrieval (Return)

**Purpose & How User Uses It:** Check for vouchers that are still unused for a particular promotion, helping manage inventory of active vouchers.

**Function 2: List the customer name and email which are membership but loyaltyPoints is low then 200**

CREATE OR REPLACE FUNCTION Low\_Point\_Members

RETURN CLOB

IS

v\_result CLOB := EMPTY\_CLOB();

BEGIN

FOR low IN (

SELECT c.Name, c.Email, c.LoyaltyPoints

FROM Customer c

JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID

WHERE c.LoyaltyPoints < 200

AND cs.Status = 'Active'

) LOOP

v\_result := v\_result || 'Name: ' || low.Name || CHR(10) ||

'Email: ' || low.Email || CHR(10) ||

'Loyalty Points: ' || low.LoyaltyPoints || CHR(10) || CHR(10);

END LOOP;

RETURN v\_result;

END;

/

**Operation Type:** Selection and Display (Return)

**Purpose & How User Uses It:** List members with loyalty points below 50 who are active members, useful for special retention campaigns.

**Command for users to use it:**

**Procedure 1:**

BEGIN

Tier\_Customer\_Feedback('Gold');

END;

/

**Procedure 2:**

BEGIN

Efficient\_Promo();

END;

/

**Function 1:**

BEGIN

DBMS\_OUTPUT.PUT\_LINE(Check\_Voucher('Spring Fling'));

END;

/

**Function 2:**

BEGIN

DBMS\_OUTPUT.PUT\_LINE(Low\_Point\_Members);

END;

/

**Appendix**

Member 1:

SELECT c.Name, c.Email, c.LoyaltyPoints

FROM Customer c

ORDER BY c.LoyaltyPoints DESC

FETCH FIRST 5 ROWS ONLY;

SELECT p.Title, p.StartDate, p.EndDate

FROM Promotion p

WHERE p.StartDate BETWEEN

TRUNC(ADD\_MONTHS(SYSDATE, 1), 'MM')

AND LAST\_DAY(ADD\_MONTHS(SYSDATE, 1))

AND p.Status = 'PLANNED';

CREATE OR REPLACE PROCEDURE Award\_Segment\_Bonus(

p\_SegmentID IN NUMBER,   
  
p\_Bonus IN NUMBER

) AS

BEGIN

UPDATE Customer c   
  
SET c.LoyaltyPoints = c.LoyaltyPoints + p\_Bonus   
  
WHERE c.CustomerID IN (   
  
 SELECT cs.CustomerID   
  
 FROM CS\_Membership cs   
  
 WHERE cs.SegmentID = p\_SegmentID   
  
);   
  
COMMIT;

END Award\_Segment\_Bonus;

/

EXEC Award\_Segment\_Bonus(801, 50);

SELECT c.CustomerID, c.LoyaltyPoints, cs.SegmentID

FROM Customer c ,CS\_Membership cs

Where c.CustomerID = cs.CustomerID;

CREATE OR REPLACE PROCEDURE Reset\_Vouchers\_For\_Promo( p\_PromoID IN NUMBER ) AS

BEGIN

UPDATE Voucher v   
  
SET v.Used = 'N'   
  
WHERE v.PromoID = p\_PromoID;   
  
COMMIT;

END Reset\_Vouchers\_For\_Promo;

/

EXEC Reset\_Vouchers\_For\_Promo(100);

SELECT \* FROM Voucher WHERE PromoID = 100;

CREATE OR REPLACE FUNCTION Days\_Until\_Promo( p\_PromoID IN NUMBER )

RETURN NUMBER

IS

v\_Days NUMBER;

BEGIN

SELECT ABS(p.StartDate - TRUNC(SYSDATE))   
  
INTO v\_Days   
  
FROM Promotion p   
  
WHERE p.PromoID = p\_PromoID;   
  
   
  
RETURN v\_Days;

END Days\_Until\_Promo;

/

SELECT Days\_Until\_Promo(100) As â€œDays Until Promoâ€ FROM DUAL;

CREATE OR REPLACE FUNCTION Is\_Promo\_Active( p\_PromoID IN NUMBER ) RETURN VARCHAR2 IS

v\_Status VARCHAR2(1);   
  
v\_Start DATE;   
  
v\_End DATE;

BEGIN

SELECT p.StartDate, p.EndDate   
  
INTO v\_Start, v\_End   
  
FROM Promotion p   
  
WHERE p.PromoID = p\_PromoID;   
  
  
  
IF TRUNC(SYSDATE) BETWEEN v\_Start AND v\_End THEN   
  
 v\_Status := 'Y';   
  
ELSE   
  
 v\_Status := 'N';   
  
END IF;   
  
  
  
RETURN v\_Status;

END Is\_Promo\_Active;

/

SELECT Is\_Promo\_Active(100) As â€œIs Promo Active? Y-Yes N-Noâ€ FROM DUAL;

Member 2:

SELECT p.Title,

TO\_NUMBER(REPLACE(a.RedemptionRate, '%', '')) AS Rate

FROM Promotion p

JOIN PromoAnalytics a ON p.PromoID = a.PromoID

ORDER BY Rate DESC

FETCH FIRST 5 ROWS ONLY;

SELECT seg.Description, COUNT(cs.CustomerID) AS MemberCount

FROM CustomerSegment seg

LEFT JOIN CS\_Membership cs ON seg.SegmentID = cs.SegmentID

GROUP BY seg.Description;

CREATE OR REPLACE PROCEDURE Expire\_Vouchers(

p\_PromoID IN NUMBER,

p\_ExpiryDate IN DATE

) AS

BEGIN

UPDATE Voucher

SET Used = 'Y'

WHERE PromoID = p\_PromoID

AND ValidTo < p\_ExpiryDate;

COMMIT;

END Expire\_Vouchers;

/

-- Example call for Expire\_Vouchers:

-- This will mark as used all vouchers for promo 101 that expired before April 1, 2025

BEGIN Expire\_Vouchers(

103,

TO\_DATE('2025-04-01','YYYY-MM-DD')

);

END;

/

-- Verify the update with a SELECT:

SELECT \* FROM Voucher

WHERE PromoID = 103

AND ValidTo < DATE '2025-04-01';

-- Create the sequence

CREATE SEQUENCE MarketingEvent\_SEQ

START WITH 400   
  
INCREMENT BY 1   
  
NOCACHE   
  
NOCYCLE;

CREATE OR REPLACE PROCEDURE Add\_Marketing\_Event(

p\_PromoID IN NUMBER,   
  
p\_Name IN VARCHAR2,   
  
p\_Location IN VARCHAR2,   
  
p\_Date IN DATE,   
  
p\_Time IN TIMESTAMP

) AS

v\_event\_id NUMBER;   
  
v\_max\_event\_id NUMBER;

BEGIN

-- Get the latest EventID used in MarketingEvent   
  
SELECT MAX(EventID) INTO v\_max\_event\_id FROM MarketingEvent;   
  
  
  
-- Get the next value from sequence   
  
SELECT MarketingEvent\_SEQ.NEXTVAL INTO v\_event\_id FROM DUAL;   
  
  
  
-- If sequence value is less than or equal to MAX(EventID), loop until it's greater   
  
WHILE v\_event\_id <= v\_max\_event\_id LOOP   
  
 SELECT MarketingEvent\_SEQ.NEXTVAL INTO v\_event\_id FROM DUAL;   
  
END LOOP;   
  
  
  
-- Insert the new marketing event with the correct EventID   
  
INSERT INTO MarketingEvent (EventID, PromoID, EventName, Location, EventDate, EventTime)   
  
VALUES (v\_event\_id, p\_PromoID, p\_Name, p\_Location, p\_Date, p\_Time);   
  
  
  
COMMIT;

END Add\_Marketing\_Event;

/

-- Example call for Add\_Marketing\_Event:

BEGIN

Add\_Marketing\_Event(

109,

'Glorious Event',

'Hall D',

TO\_DATE('2025-06-01','YYYY-MM-DD'),

TO\_TIMESTAMP('2025-06-01 10:00:00','YYYY-MM-DD HH24:MI:SS') );

END;

/

-- Verify the insertion with a SELECT:

SELECT \*

FROM MarketingEvent

WHERE PromoID = 109;

CREATE OR REPLACE FUNCTION Get\_Customer\_Points(p\_CustID IN NUMBER)

RETURN NUMBER

IS

v\_points NUMBER;

BEGIN

SELECT LoyaltyPoints   
  
INTO v\_points   
  
FROM Customer   
  
WHERE CustomerID = p\_CustID;   
  
  
  
RETURN v\_points;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN   
  
 v\_points := 0; -- If customer not found, return 0   
  
 RETURN v\_points;

END Get\_Customer\_Points;

/

-- Example call for Get\_Customer\_Points:

-- Calculate total loyalty points for customer 1

SELECT Get\_Customer\_Points(1) AS total\_points

FROM dual;

CREATE OR REPLACE FUNCTION Get\_Top\_Feedback( p\_PromoID IN NUMBER )

RETURN VARCHAR2

IS

v\_comment VARCHAR2(400);

BEGIN

-- Use a subquery to get the top comment for the promo with the highest rating   
  
SELECT Comments   
  
INTO v\_comment   
  
FROM (   
  
 SELECT Comments   
  
 FROM Feedback   
  
 WHERE PromoID = p\_PromoID   
  
 AND Rating = (SELECT MAX(Rating)   
  
 FROM Feedback   
  
 WHERE PromoID = p\_PromoID)   
  
 ORDER BY RATING DESC   
  
)   
  
WHERE ROWNUM = 1; -- Ensures we only get the first row (top feedback)   
  
  
  
RETURN v\_comment;

END Get\_Top\_Feedback;

/

-- Example call for Get\_Top\_Feedback:

-- Retrieve the best feedback comment for promotion 101

SELECT Get\_Top\_Feedback(101) AS top\_comment

FROM dual;

Member 3:

SELECT c.Name, c.Email, f.Comments, f.Rating

FROM Customer c, Feedback f

WHERE c.CustomerID = f.CustomerID

AND f.Comments IS NULL;

SELECT p.PromoID, p.Title

FROM Promotion p

WHERE NOT EXISTS ( SELECT 1 FROM PromoAnalytics a WHERE a.PromoID = p.PromoID );

CREATE OR REPLACE PROCEDURE Close\_Old\_Promos(p\_BeforeDate IN DATE) AS v\_count NUMBER; BEGIN -- Count how many promotions will be updated SELECT COUNT(\*) INTO v\_count FROM Promotion WHERE EndDate < p\_BeforeDate AND Status != 'CLOSED';

UPDATE Promotion

SET Status = 'CLOSED'

WHERE EndDate < p\_BeforeDate AND Status != 'CLOSED';

COMMIT;

DBMS\_OUTPUT.PUT\_LINE(v\_count || ' promotion(s) closed before ' || TO\_CHAR(p\_BeforeDate, 'YYYY-MM-DD'));

END Close\_Old\_Promos;

/

SET SERVEROUTPUT ON;

EXEC Close\_Old\_Promos(TO\_DATE('2025-06-01', 'YYYY-MM-DD'));

SELECT PromoID, Title, EndDate, Status

FROM Promotion

WHERE EndDate < TO\_DATE('2025-06-01', 'YYYY-MM-DD');

CREATE OR REPLACE PROCEDURE Send\_Low\_Points\_Reminder AS

BEGIN

FOR low IN (

SELECT c.Name, c.Email, c.LoyaltyPoints FROM Customer c JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID WHERE c.LoyaltyPoints < 50 AND cs.Status = 'active'

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Sending reminder to: ' || low.Name || ' (' || low.Email || ')'); DBMS\_OUTPUT.PUT\_LINE('Loyalty Points: ' || low.LoyaltyPoints || ' - Reminder: Earn more points for rewards!'); DBMS\_OUTPUT.PUT\_LINE('--------------------------------------------------------');

END LOOP;

END Send\_Low\_Points\_Reminder;

/

SET SERVEROUTPUT ON;

EXEC Send\_Low\_Points\_Reminder;

SELECT c.CustomerID, c.Name, c.Email, c.LoyaltyPoints, cs.Status

FROM Customer c

JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID

WHERE c.LoyaltyPoints < 50 AND cs.Status = 'Active';

CREATE OR REPLACE FUNCTION Avg\_Redemption\_Rate

RETURN NUMBER

IS

v\_avg NUMBER;

BEGIN

SELECT AVG(TO\_NUMBER(REPLACE(RedemptionRate, '%', '')))

INTO v\_avg

FROM PromoAnalytics;

RETURN v\_avg;

END Avg\_Redemption\_Rate;

/

SELECT Avg\_Redemption\_Rate() AS avg\_redemption\_rate FROM dual;

CREATE OR REPLACE FUNCTION Top\_Customer\_Segment RETURN VARCHAR2 IS

v\_segment VARCHAR2(400);

BEGIN

SELECT seg.Description   
  
INTO v\_segment   
  
FROM (   
  
 SELECT SegmentID   
  
 FROM CS\_Membership   
  
 GROUP BY SegmentID   
  
 ORDER BY COUNT(CustomerID) DESC   
  
) best   
  
JOIN CustomerSegment seg ON best.SegmentID = seg.SegmentID   
  
WHERE ROWNUM = 1;   
  
  
  
RETURN v\_segment;

END Top\_Customer\_Segment;

/

SELECT Top\_Customer\_Segment() AS top\_segment FROM dual;

Member 4:

SELECT c.Name, Tier, Comments

FROM Customer c

JOIN FeedBack f

ON c.CustomerID = f.CustomerID

JOIN LoyaltyProgram l

ON c.CustomerID = l.CustomerID

WHERE Tier = 'Gold';

SELECT p.Type, COUNT(f.Rating) AS NumberOfRatings,

SUM(f.Rating) AS TotalRating

FROM Promotion p JOIN Feedback f

ON p.PromoID = f.PromoID

GROUP BY p.Type

ORDER BY TotalRating DESC;

CREATE OR REPLACE PROCEDURE Tier\_Customer\_Feedback(

P\_tier IN VARCHAR2)

IS

BEGIN

FOR cusF IN (

SELECT c.Name, f.Comments, l.Tier

FROM Customer c

JOIN FeedBack f ON c.CustomerID = f.CustomerID

JOIN LoyaltyProgram l ON c.CustomerID = l.CustomerID

WHERE l.Tier = P\_tier

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Name: ' || cusF.Name); DBMS\_OUTPUT.PUT\_LINE('Comments: ' || cusF.Comments); DBMS\_OUTPUT.PUT\_LINE('Tier: ' || cusF.Tier); DBMS\_OUTPUT.PUT\_LINE('-------------------------');

END LOOP;

END;

/

CREATE OR REPLACE PROCEDURE Efficient\_Promo

IS

BEGIN

FOR ePro IN (

SELECT p.Title, p.Type, p.Status, a.RedemptionRate

FROM Promotion p

JOIN PromoAnalytics a ON p.PromoID = a.PromoID

WHERE a.RedemptionRate > 10

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Title: ' || ePro.Title); DBMS\_OUTPUT.PUT\_LINE('Type: ' || ePro.Type); DBMS\_OUTPUT.PUT\_LINE('Status: ' || ePro.Status); DBMS\_OUTPUT.PUT\_LINE('Redemption Rate: ' || ePro.RedemptionRate || '%');

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

END LOOP;

END;

/

CREATE OR REPLACE FUNCTION Check\_Voucher (

p\_Title IN VARCHAR2

)

RETURN VARCHAR2

IS

v\_output VARCHAR2(4000) := '';

BEGIN

FOR vou IN (

SELECT v.CodeID, v.DiscountValue, v.ValidFrom, v.ValidTo, v.Used

FROM Voucher v

JOIN Promotion p ON p.PromoID = v.PromoID

WHERE p.Title = p\_Title

AND v.Used = 'N'

) LOOP

v\_output := v\_output ||

'Code: ' || vou.CodeID || CHR(10) ||

'Discount Value: ' || vou.DiscountValue || CHR(10) ||

'Valid From: ' || TO\_CHAR(vou.ValidFrom, 'DD-MON-YYYY') || CHR(10) ||

'Valid To: ' || TO\_CHAR(vou.ValidTo, 'DD-MON-YYYY') || CHR(10) ||

'Used: ' || vou.Used || CHR(10) || CHR(10);

END LOOP;

IF v\_output IS NULL THEN

v\_output := 'No unused vouchers found for the given title.';

END IF;

RETURN v\_output;

END; /

CREATE OR REPLACE FUNCTION Low\_Point\_Members

RETURN CLOB

IS

v\_result CLOB := EMPTY\_CLOB();

BEGIN

FOR low IN (

SELECT c.Name, c.Email, c.LoyaltyPoints

FROM Customer c

JOIN CS\_Membership cs ON c.CustomerID = cs.CustomerID

WHERE c.LoyaltyPoints < 200

AND cs.Status = 'Active'

) LOOP

v\_result := v\_result || 'Name: ' || low.Name || CHR(10) ||

'Email: ' || low.Email || CHR(10) ||   
  
'Loyalty Points: ' || low.LoyaltyPoints || CHR(10) || CHR(10);

END LOOP;

RETURN v\_result;

END;

/

--Command for users to use it:

--Procedure 1:

BEGIN

Tier\_Customer\_Feedback('Gold');

END;

/

--Procedure 2:

BEGIN

Efficient\_Promo();

END;

/

--Function 1:

BEGIN

DBMS\_OUTPUT.PUT\_LINE(Check\_Voucher('Spring Fling'));

END;

/

--Function 2:

BEGIN

DBMS\_OUTPUT.PUT\_LINE(Low\_Point\_Members);

END;

/

Group Script:

/\* COURSE CODE: UCCD2303 PROGRAMME: CS GROUP NUMBER: G055 GROUP LEADER NAME & EMAIL: Lee Cheng Jun & [chengjun5487@1utar.my](mailto:chengjun5487@1utar.my) MEMBER 2 NAME: Brian Lee Zhen Hui MEMBER 3 NAME: Ng Che Te MEMBER 4 NAME: Tan Jia Yi Submission date and time (DD-MON-YY): 29-April-2025, 5pm

Part 1 group work: Template save as "G055.sql"

Refer to the format of Northwoods.sql as an example for group sql script submission \*/

-- Drop tables DROP TABLE Feedback CASCADE CONSTRAINTS; DROP TABLE PromoAnalytics CASCADE CONSTRAINTS; DROP TABLE Voucher\_Item CASCADE CONSTRAINTS; DROP TABLE Voucher CASCADE CONSTRAINTS; DROP TABLE Item\_Detail CASCADE CONSTRAINTS; DROP TABLE PromoItem CASCADE CONSTRAINTS; DROP TABLE MarketingEvent CASCADE CONSTRAINTS; DROP TABLE AdSchedule CASCADE CONSTRAINTS; DROP TABLE Promotion\_Segment CASCADE CONSTRAINTS; DROP TABLE Promotion CASCADE CONSTRAINTS; DROP TABLE CS\_Membership CASCADE CONSTRAINTS; DROP TABLE CustomerSegment CASCADE CONSTRAINTS; DROP TABLE LoyaltyProgram CASCADE CONSTRAINTS; DROP TABLE Customer CASCADE CONSTRAINTS;

-- 1. CUSTOMER CREATE TABLE Customer ( CustomerID NUMBER PRIMARY KEY, Name VARCHAR2(100) NOT NULL, Email VARCHAR2(100) UNIQUE NOT NULL, DOB DATE, Phone VARCHAR2(20), LoyaltyPoints NUMBER DEFAULT 0 );

-- 2. LOYALTYPROGRAM CREATE TABLE LoyaltyProgram ( ProgramID NUMBER PRIMARY KEY, CustomerID NUMBER NOT NULL, Points NUMBER DEFAULT 0, Tier VARCHAR2(20), JoinDate DATE DEFAULT SYSDATE, CONSTRAINT FK\_LP\_Customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID) );

-- 3. PROMOTION CREATE TABLE Promotion ( PromoID NUMBER PRIMARY KEY, Title VARCHAR2(150) NOT NULL, StartDate DATE NOT NULL, EndDate DATE NOT NULL, Type VARCHAR2(50), Status VARCHAR2(20) DEFAULT 'PLANNED' );

-- 4. PROMOITEM CREATE TABLE PromoItem ( ItemID NUMBER PRIMARY KEY, PromoID NUMBER NOT NULL, ItemType VARCHAR2(50), Stock NUMBER DEFAULT 0, Price NUMBER(10,2) DEFAULT 0, CONSTRAINT FK\_PI\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

-- 5. ITEM\_DETAIL CREATE TABLE Item\_Detail ( PromoID NUMBER NOT NULL, ItemID NUMBER NOT NULL, DiscountRate NUMBER(5,2) DEFAULT 0, StockLimit NUMBER, CONSTRAINT PK\_Item\_Detail PRIMARY KEY (PromoID, ItemID), CONSTRAINT FK\_ID\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID), CONSTRAINT FK\_ID\_PromoItem FOREIGN KEY (ItemID) REFERENCES PromoItem(ItemID) );

-- 6. VOUCHER CREATE TABLE Voucher ( CodeID NUMBER PRIMARY KEY, PromoID NUMBER NOT NULL, Code VARCHAR2(50) UNIQUE NOT NULL, DiscountValue NUMBER(5,2) DEFAULT 0, ValidFrom DATE, ValidTo DATE, Used CHAR(1) DEFAULT 'N', CONSTRAINT FK\_Voucher\_Promo FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

-- 7. VOUCHER\_ITEM CREATE TABLE Voucher\_Item ( ItemID NUMBER NOT NULL, CodeID NUMBER NOT NULL, MaxUsagePerItem NUMBER DEFAULT 1, IsExclusive CHAR(1) DEFAULT 'N', MinPurchaseAmount NUMBER(10,2), CONSTRAINT PK\_Voucher\_Item PRIMARY KEY (ItemID, CodeID), CONSTRAINT FK\_VI\_PromoItem FOREIGN KEY (ItemID) REFERENCES PromoItem(ItemID), CONSTRAINT FK\_VI\_Voucher FOREIGN KEY (CodeID) REFERENCES Voucher(CodeID) );

-- 8. MARKETINGEVENT CREATE TABLE MarketingEvent ( EventID NUMBER PRIMARY KEY, PromoID NUMBER NOT NULL, EventName VARCHAR2(100) NOT NULL, Location VARCHAR2(100), EventDate DATE, EventTime TIMESTAMP, CONSTRAINT FK\_ME\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

-- 9. ADSCHEDULE CREATE TABLE AdSchedule ( AdID NUMBER PRIMARY KEY, PromoID NUMBER NOT NULL, Platform VARCHAR2(50), StartTime TIMESTAMP, EndTime TIMESTAMP, ContentType VARCHAR2(50), CONSTRAINT FK\_AS\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

--10. PROMOANALYTICS CREATE TABLE PromoAnalytics ( ReportID NUMBER PRIMARY KEY, PromoID NUMBER NOT NULL, SalesImpact VARCHAR2(200), RedemptionRate NUMBER(5,2), FeedbackSummary VARCHAR2(400), CONSTRAINT FK\_PA\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

--11. FEEDBACK CREATE TABLE Feedback ( FeedbackID NUMBER PRIMARY KEY, CustomerID NUMBER NOT NULL, PromoID NUMBER NOT NULL, Comments VARCHAR2(400), Rating NUMBER(1) CHECK (Rating BETWEEN 1 AND 5), FeedbackDate DATE DEFAULT SYSDATE, CONSTRAINT FK\_FB\_Customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID), CONSTRAINT FK\_FB\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID) );

--12. CUSTOMERSEGMENT CREATE TABLE CustomerSegment ( SegmentID NUMBER PRIMARY KEY, ProgramID NUMBER NOT NULL, Criteria VARCHAR2(200), Description VARCHAR2(400), CONSTRAINT FK\_CS\_LoyaltyProgram FOREIGN KEY (ProgramID) REFERENCES LoyaltyProgram(ProgramID) );

--13. PROMOTION\_SEGMENT CREATE TABLE Promotion\_Segment ( PromoID NUMBER NOT NULL, SegmentID NUMBER NOT NULL, PriorityLevel NUMBER DEFAULT 0, Channel VARCHAR2(50), CONSTRAINT PK\_Promo\_Segment PRIMARY KEY (PromoID, SegmentID), CONSTRAINT FK\_PS\_Promotion FOREIGN KEY (PromoID) REFERENCES Promotion(PromoID), CONSTRAINT FK\_PS\_Segment FOREIGN KEY (SegmentID) REFERENCES CustomerSegment(SegmentID) );

--14. CS\_MEMBERSHIP CREATE TABLE CS\_Membership ( CustomerID NUMBER NOT NULL, SegmentID NUMBER NOT NULL, DateJoined DATE DEFAULT SYSDATE, Status VARCHAR2(20), CONSTRAINT PK\_CS\_Membership PRIMARY KEY (CustomerID, SegmentID), CONSTRAINT FK\_CSM\_Customer FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID), CONSTRAINT FK\_CSM\_Segment FOREIGN KEY (SegmentID) REFERENCES CustomerSegment(SegmentID) );

-- CUSTOMER INSERT INTO Customer VALUES (1, 'Alice Tan', '[alice.tan@gmail.com](mailto:alice.tan@gmail.com)', DATE '1990-04-12', '012-3456789', 100); INSERT INTO Customer VALUES (2, 'Bob Lim', '[bob.lim@yahoo.com](mailto:bob.lim@yahoo.com)', DATE '1985-11-05', '013-9876543', 250); INSERT INTO Customer VALUES (3, 'Chen Wei', '[chen.wei@hotmail.com](mailto:chen.wei@hotmail.com)', DATE '1992-06-30', '014-1112233', 50); INSERT INTO Customer VALUES (4, 'Dina Rossi', '[dina.rossi@gmail.com](mailto:dina.rossi@gmail.com)', DATE '1988-12-20', '015-4445566', 0); INSERT INTO Customer VALUES (5, 'Evan Lee', '[evan.lee@yahoo.com](mailto:evan.lee@yahoo.com)', DATE '1995-02-14', '016-7778899', 300); INSERT INTO Customer VALUES (6, 'Fiona Ng', '[fiona.ng@gmail.com](mailto:fiona.ng@gmail.com)', DATE '1991-07-22', '017-8887766', 120); INSERT INTO Customer VALUES (7, 'Gavin Tan', '[gavin.tan@hotmail.com](mailto:gavin.tan@hotmail.com)', DATE '1987-03-15', '018-3322110', 80); INSERT INTO Customer VALUES (8, 'Hana Lim', '[hana.lim@yahoo.com](mailto:hana.lim@yahoo.com)', DATE '1993-09-05', '019-9988776', 200); INSERT INTO Customer VALUES (9, 'Ivan Choo', '[ivan.choo@gmail.com](mailto:ivan.choo@gmail.com)', DATE '1989-01-10', '011-6677889', 40); INSERT INTO Customer VALUES (10, 'Jasmine Ho', '[jasmine.ho@hotmail.com](mailto:jasmine.ho@hotmail.com)', DATE '1996-11-25', '012-5566778', 180); INSERT INTO Customer VALUES (11, 'Kelvin Teo', '[kelvin.teo@gmail.com](mailto:kelvin.teo@gmail.com)', DATE '1984-04-30', '013-4455667', 300); INSERT INTO Customer VALUES (12, 'Laura Goh', '[laura.goh@yahoo.com](mailto:laura.goh@yahoo.com)', DATE '1992-08-18', '014-2233445', 70); INSERT INTO Customer VALUES (13, 'Marcus Lee', '[marcus.lee@gmail.com](mailto:marcus.lee@gmail.com)', DATE '1990-05-01', '015-5566778', 90); INSERT INTO Customer VALUES (14, 'Natalie Tan', '[natalie.tan@hotmail.com](mailto:natalie.tan@hotmail.com)', DATE '1995-12-15', '016-3344556', 210); INSERT INTO Customer VALUES (15, 'Oscar Wong', '[oscar.wong@gmail.com](mailto:oscar.wong@gmail.com)', DATE '1988-02-20', '017-8899001', 60); INSERT INTO Customer VALUES (16, 'Pauline Cheng', '[pauline.cheng@yahoo.com](mailto:pauline.cheng@yahoo.com)', DATE '1991-06-08', '018-6677885', 130); INSERT INTO Customer VALUES (17, 'Quentin Ong', '[quentin.ong@hotmail.com](mailto:quentin.ong@hotmail.com)', DATE '1994-10-11', '019-7766554', 20); INSERT INTO Customer VALUES (18, 'Rachel Yap', '[rachel.yap@gmail.com](mailto:rachel.yap@gmail.com)', DATE '1993-01-17', '011-3344557', 110); INSERT INTO Customer VALUES (19, 'Samuel Low', '[samuel.low@gmail.com](mailto:samuel.low@gmail.com)', DATE '1987-08-22', '012-4455667', 240); INSERT INTO Customer VALUES (20, 'Tina Khoo', '[tina.khoo@hotmail.com](mailto:tina.khoo@hotmail.com)', DATE '1995-09-30', '013-7788990', 150);

-- LOYALTYPROGRAM INSERT INTO LoyaltyProgram VALUES (10, 1, 100, 'Silver', DATE '2024-01-01'); INSERT INTO LoyaltyProgram VALUES (11, 2, 250, 'Gold', DATE '2023-05-15'); INSERT INTO LoyaltyProgram VALUES (12, 3, 50, 'Bronze', DATE '2024-03-10'); INSERT INTO LoyaltyProgram VALUES (13, 4, 0, 'Bronze', DATE '2025-02-01'); INSERT INTO LoyaltyProgram VALUES (14, 5, 300, 'Gold', DATE '2023-10-20'); INSERT INTO LoyaltyProgram VALUES (15, 6, 120, 'Silver', DATE '2024-02-05'); INSERT INTO LoyaltyProgram VALUES (16, 7, 80, 'Bronze', DATE '2024-01-18'); INSERT INTO LoyaltyProgram VALUES (17, 8, 200, 'Silver', DATE '2023-11-11'); INSERT INTO LoyaltyProgram VALUES (18, 9, 40, 'Bronze', DATE '2024-04-12'); INSERT INTO LoyaltyProgram VALUES (19, 10, 180, 'Silver', DATE '2023-06-08'); INSERT INTO LoyaltyProgram VALUES (20, 11, 300, 'Gold', DATE '2022-12-30'); INSERT INTO LoyaltyProgram VALUES (21, 12, 70, 'Bronze', DATE '2024-02-20'); INSERT INTO LoyaltyProgram VALUES (22, 13, 90, 'Bronze', DATE '2023-08-15'); INSERT INTO LoyaltyProgram VALUES (23, 14, 210, 'Gold', DATE '2023-03-01'); INSERT INTO LoyaltyProgram VALUES (24, 15, 60, 'Bronze', DATE '2024-05-19'); INSERT INTO LoyaltyProgram VALUES (25, 16, 130, 'Silver', DATE '2023-09-07'); INSERT INTO LoyaltyProgram VALUES (26, 17, 20, 'Bronze', DATE '2024-06-22'); INSERT INTO LoyaltyProgram VALUES (27, 18, 110, 'Silver', DATE '2023-11-18'); INSERT INTO LoyaltyProgram VALUES (28, 19, 240, 'Gold', DATE '2023-01-29'); INSERT INTO LoyaltyProgram VALUES (29, 20, 150, 'Silver', DATE '2023-07-14');

-- PROMOTION INSERT INTO Promotion VALUES (100, 'Spring Fling', DATE '2025-03-01', DATE '2025-03-31', 'Voucher', 'ACTIVE'); INSERT INTO Promotion VALUES (101, 'Burger Week', DATE '2025-04-01', DATE '2025-04-07', 'Item', 'PLANNED'); INSERT INTO Promotion VALUES (102, 'Kids Workshop', DATE '2025-05-10', DATE '2025-05-10', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (103, 'Summer Combo', DATE '2025-06-01', DATE '2025-06-30', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (104, 'Mother''s Day Special', DATE '2025-05-01', DATE '2025-05-31', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (105, 'Holiday Bonanza', DATE '2025-11-01', DATE '2025-11-30', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (106, 'Family Day', DATE '2025-08-15', DATE '2025-08-15', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (107, 'Tech Gadget Week', DATE '2025-09-05', DATE '2025-09-12', 'Item', 'PLANNED'); INSERT INTO Promotion VALUES (108, 'Back to School', DATE '2025-07-25', DATE '2025-08-10', 'Voucher', 'ACTIVE'); INSERT INTO Promotion VALUES (109, 'Movie Mania', DATE '2025-10-01', DATE '2025-10-15', 'Item', 'PLANNED'); INSERT INTO Promotion VALUES (110, 'Halloween Bash', DATE '2025-10-25', DATE '2025-10-31', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (111, 'Black Friday Sale', DATE '2025-11-28', DATE '2025-11-30', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (112, 'Cyber Monday', DATE '2025-12-02', DATE '2025-12-02', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (113, 'New Year Gala', DATE '2025-12-31', DATE '2026-01-01', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (114, 'Online Flash Sale', DATE '2025-08-05', DATE '2025-08-05', 'Item', 'PLANNED'); INSERT INTO Promotion VALUES (115, 'Early Bird Discount', DATE '2025-05-01', DATE '2025-05-03', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (116, 'Ramadan Special', DATE '2025-03-15', DATE '2025-04-15', 'Voucher', 'PLANNED'); INSERT INTO Promotion VALUES (117, 'Mother''s Day Treat', DATE '2025-05-10', DATE '2025-05-10', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (118, 'Fathers Appreciation', DATE '2025-06-20', DATE '2025-06-20', 'Event', 'PLANNED'); INSERT INTO Promotion VALUES (119, 'Mega Online Week', DATE '2025-07-20', DATE '2025-07-27', 'Item', 'PLANNED');

-- PROMOITEM INSERT INTO PromoItem VALUES (200, 101, 'Burger', 500, 5.50); INSERT INTO PromoItem VALUES (201, 101, 'Fries', 500, 2.00); INSERT INTO PromoItem VALUES (202, 104, 'Online Burger', 999, 4.50); INSERT INTO PromoItem VALUES (203, 104, 'Online Drink', 999, 1.50); INSERT INTO PromoItem VALUES (204, 100, 'Value Set', 999, 8.00); INSERT INTO PromoItem VALUES (205, 105, 'Holiday Set', 100, 12.00); INSERT INTO PromoItem VALUES (206, 107, 'Tech Bag', 50, 45.00); INSERT INTO PromoItem VALUES (207, 107, 'Wireless Earbuds', 80, 70.00); INSERT INTO PromoItem VALUES (208, 108, 'Backpack', 200, 40.00); INSERT INTO PromoItem VALUES (209, 109, 'Movie Ticket', 300, 8.00); INSERT INTO PromoItem VALUES (210, 109, 'Popcorn Combo', 400, 5.00); INSERT INTO PromoItem VALUES (211, 114, 'Flash Drive', 150, 10.00); INSERT INTO PromoItem VALUES (212, 114, 'Keyboard', 70, 25.00); INSERT INTO PromoItem VALUES (213, 119, 'Online Headphones', 999, 55.00); INSERT INTO PromoItem VALUES (214, 119, 'Bluetooth Speaker', 999, 75.00); INSERT INTO PromoItem VALUES (215, 100, 'Coupon Deal', 999, 3.00); INSERT INTO PromoItem VALUES (216, 103, 'Meal Combo', 150, 7.50); INSERT INTO PromoItem VALUES (217, 105, 'Gift Hamper', 60, 50.00); INSERT INTO PromoItem VALUES (218, 107, 'Smartwatch', 30, 120.00); INSERT INTO PromoItem VALUES (219, 119, 'Gaming Mouse', 90, 60.00);

-- ITEM\_DETAIL INSERT INTO Item\_Detail VALUES (101, 200, 10, 100); INSERT INTO Item\_Detail VALUES (101, 201, 0, 300); INSERT INTO Item\_Detail VALUES (104, 202, 15, 999); INSERT INTO Item\_Detail VALUES (104, 203, 20, 999); INSERT INTO Item\_Detail VALUES (100, 204, 5, 50); INSERT INTO Item\_Detail VALUES (105, 205, 8, 120); INSERT INTO Item\_Detail VALUES (107, 206, 12, 60); INSERT INTO Item\_Detail VALUES (107, 207, 15, 100); INSERT INTO Item\_Detail VALUES (108, 208, 10, 250); INSERT INTO Item\_Detail VALUES (109, 209, 5, 400); INSERT INTO Item\_Detail VALUES (109, 210, 0, 500); INSERT INTO Item\_Detail VALUES (114, 211, 20, 180); INSERT INTO Item\_Detail VALUES (114, 212, 10, 90); INSERT INTO Item\_Detail VALUES (119, 213, 18, 999); INSERT INTO Item\_Detail VALUES (119, 214, 15, 999); INSERT INTO Item\_Detail VALUES (100, 215, 5, 100); INSERT INTO Item\_Detail VALUES (103, 216, 10, 130); INSERT INTO Item\_Detail VALUES (105, 217, 8, 70); INSERT INTO Item\_Detail VALUES (107, 218, 20, 40); INSERT INTO Item\_Detail VALUES (119, 219, 12, 100);

-- VOUCHER INSERT INTO Voucher VALUES (300, 100, 'SPRING25', 25, DATE '2025-03-01', DATE '2025-03-31', 'N'); INSERT INTO Voucher VALUES (301, 103, 'SUMMER10', 10, DATE '2025-06-01', DATE '2025-06-30', 'N'); INSERT INTO Voucher VALUES (302, 100, 'SPRING50', 50, DATE '2025-03-01', DATE '2025-03-15', 'N'); INSERT INTO Voucher VALUES (303, 100, 'SPRING5', 5, DATE '2025-03-10', DATE '2025-03-31', 'N'); INSERT INTO Voucher VALUES (304, 103, 'SUMMER20', 20, DATE '2025-06-01', DATE '2025-06-30', 'N'); INSERT INTO Voucher VALUES (305, 105, 'HOLIDAY30', 30, DATE '2025-11-01', DATE '2025-11-30', 'N'); INSERT INTO Voucher VALUES (306, 108, 'BACK2SCHOOL', 15, DATE '2025-07-25', DATE '2025-08-10', 'N'); INSERT INTO Voucher VALUES (307, 111, 'BLACKFRIDAY50', 50, DATE '2025-11-28', DATE '2025-11-30', 'N'); INSERT INTO Voucher VALUES (308, 112, 'CYBERMONDAY20', 20, DATE '2025-12-02', DATE '2025-12-02', 'N'); INSERT INTO Voucher VALUES (309, 115, 'EARLY10', 10, DATE '2025-05-01', DATE '2025-05-03', 'N'); INSERT INTO Voucher VALUES (310, 116, 'RAMADAN15', 15, DATE '2025-03-15', DATE '2025-04-15', 'N'); INSERT INTO Voucher VALUES (311, 119, 'MEGAWEEK30', 30, DATE '2025-07-20', DATE '2025-07-27', 'N'); INSERT INTO Voucher VALUES (312, 105, 'FESTIVE40', 40, DATE '2025-11-01', DATE '2025-11-30', 'N'); INSERT INTO Voucher VALUES (313, 100, 'SPRING15', 15, DATE '2025-03-01', DATE '2025-03-31', 'N'); INSERT INTO Voucher VALUES (314, 103, 'SUMMER5', 5, DATE '2025-06-01', DATE '2025-06-30', 'N'); INSERT INTO Voucher VALUES (315, 116, 'RAMADAN5', 5, DATE '2025-03-15', DATE '2025-04-15', 'N'); INSERT INTO Voucher VALUES (316, 119, 'ONLINE10', 10, DATE '2025-07-20', DATE '2025-07-27', 'N'); INSERT INTO Voucher VALUES (317, 105, 'HOLIDAY10', 10, DATE '2025-11-01', DATE '2025-11-30', 'N'); INSERT INTO Voucher VALUES (318, 115, 'EARLY5', 5, DATE '2025-05-01', DATE '2025-05-03', 'N'); INSERT INTO Voucher VALUES (319, 119, 'FLASH20', 20, DATE '2025-07-20', DATE '2025-07-27', 'N');

-- VOUCHER\_ITEM INSERT INTO Voucher\_Item VALUES (204, 300, 1, 'Y', 20); INSERT INTO Voucher\_Item VALUES (204, 302, 1, 'N', 0); INSERT INTO Voucher\_Item VALUES (200, 303, 2, 'N', 10); INSERT INTO Voucher\_Item VALUES (202, 301, 1, 'Y', 15); INSERT INTO Voucher\_Item VALUES (203, 304, 1, 'N', 0); INSERT INTO Voucher\_Item VALUES (205, 305, 1, 'Y', 25); INSERT INTO Voucher\_Item VALUES (208, 306, 1, 'N', 5); INSERT INTO Voucher\_Item VALUES (209, 307, 1, 'Y', 30); INSERT INTO Voucher\_Item VALUES (210, 307, 2, 'N', 20); INSERT INTO Voucher\_Item VALUES (211, 308, 1, 'Y', 10); INSERT INTO Voucher\_Item VALUES (212, 308, 1, 'N', 5); INSERT INTO Voucher\_Item VALUES (213, 311, 1, 'Y', 20); INSERT INTO Voucher\_Item VALUES (214, 311, 1, 'N', 15); INSERT INTO Voucher\_Item VALUES (215, 313, 2, 'N', 5); INSERT INTO Voucher\_Item VALUES (216, 314, 1, 'Y', 5); INSERT INTO Voucher\_Item VALUES (217, 312, 1, 'Y', 40); INSERT INTO Voucher\_Item VALUES (218, 316, 1, 'N', 10); INSERT INTO Voucher\_Item VALUES (219, 319, 1, 'Y', 15); INSERT INTO Voucher\_Item VALUES (206, 317, 1, 'N', 20); INSERT INTO Voucher\_Item VALUES (207, 318, 1, 'Y', 10);

-- MARKETINGEVENT INSERT INTO MarketingEvent VALUES (400, 100, 'Spring Launch Party', 'Mall A', DATE '2025-03-01', TIMESTAMP '2025-03-01 14:00:00'); INSERT INTO MarketingEvent VALUES (401, 101, 'Burger Festival', 'Mall B', DATE '2025-04-01', TIMESTAMP '2025-04-01 12:00:00'); INSERT INTO MarketingEvent VALUES (402, 102, 'Kids Activity Zone', 'Park C', DATE '2025-05-10', TIMESTAMP '2025-05-10 10:00:00'); INSERT INTO MarketingEvent VALUES (403, 103, 'Summer Combo Party', 'Beach D', DATE '2025-06-05', TIMESTAMP '2025-06-05 15:00:00'); INSERT INTO MarketingEvent VALUES (404, 104, 'Mother''s Day Special', 'Mall E', DATE '2025-05-12', TIMESTAMP '2025-05-12 11:00:00'); INSERT INTO MarketingEvent VALUES (405, 100, 'Spring Contest', 'Outlet A', DATE '2025-03-02', TIMESTAMP '2025-03-02 10:00:00'); INSERT INTO MarketingEvent VALUES (406, 101, 'Burger Eating Contest', 'Outlet B', DATE '2025-04-02', TIMESTAMP '2025-04-02 13:00:00'); INSERT INTO MarketingEvent VALUES (407, 102, 'Kids Coloring', 'Mall C', DATE '2025-05-11', TIMESTAMP '2025-05-11 11:00:00'); INSERT INTO MarketingEvent VALUES (408, 103, 'Summer Dance', 'Beach F', DATE '2025-06-06', TIMESTAMP '2025-06-06 16:00:00'); INSERT INTO MarketingEvent VALUES (409, 104, 'Mom and Me Special', 'Mall G', DATE '2025-05-13', TIMESTAMP '2025-05-13 10:00:00'); INSERT INTO MarketingEvent VALUES (410, 100, 'Spring Pop-up', 'Pop-up Store', DATE '2025-03-03', TIMESTAMP '2025-03-03 12:00:00'); INSERT INTO MarketingEvent VALUES (411, 101, 'Burger Combo Fiesta', 'Mall H', DATE '2025-04-03', TIMESTAMP '2025-04-03 11:00:00'); INSERT INTO MarketingEvent VALUES (412, 102, 'Kids Fun Walk', 'Park I', DATE '2025-05-12', TIMESTAMP '2025-05-12 09:00:00'); INSERT INTO MarketingEvent VALUES (413, 103, 'Beach Carnival', 'Beach J', DATE '2025-06-07', TIMESTAMP '2025-06-07 17:00:00'); INSERT INTO MarketingEvent VALUES (414, 104, 'Mother''s Day Tea', 'Hotel K', DATE '2025-05-14', TIMESTAMP '2025-05-14 14:00:00'); INSERT INTO MarketingEvent VALUES (415, 100, 'Spring Surprise', 'Mall L', DATE '2025-03-04', TIMESTAMP '2025-03-04 13:00:00'); INSERT INTO MarketingEvent VALUES (416, 101, 'Burger Launch', 'Restaurant M', DATE '2025-04-04', TIMESTAMP '2025-04-04 14:00:00'); INSERT INTO MarketingEvent VALUES (417, 102, 'Kids Party', 'Mall N', DATE '2025-05-13', TIMESTAMP '2025-05-13 15:00:00'); INSERT INTO MarketingEvent VALUES (418, 103, 'Summer BBQ', 'Beach O', DATE '2025-06-08', TIMESTAMP '2025-06-08 18:00:00'); INSERT INTO MarketingEvent VALUES (419, 104, 'Mom Gift Fair', 'Mall P', DATE '2025-05-15', TIMESTAMP '2025-05-15 12:00:00');

-- ADSCHEDULE INSERT INTO AdSchedule VALUES (500, 100, 'Facebook', TIMESTAMP '2025-02-25 00:00:00', TIMESTAMP '2025-03-01 23:59:59', 'Banner'); INSERT INTO AdSchedule VALUES (501, 101, 'Instagram', TIMESTAMP '2025-03-28 00:00:00', TIMESTAMP '2025-04-07 23:59:59', 'Story'); INSERT INTO AdSchedule VALUES (502, 102, 'Billboard', TIMESTAMP '2025-05-01 00:00:00', TIMESTAMP '2025-05-10 23:59:59', 'Video'); INSERT INTO AdSchedule VALUES (503, 103, 'YouTube', TIMESTAMP '2025-05-25 00:00:00', TIMESTAMP '2025-06-10 23:59:59', 'Video'); INSERT INTO AdSchedule VALUES (504, 104, 'Push Notification', TIMESTAMP '2025-05-01 00:00:00', TIMESTAMP '2025-05-10 23:59:59', 'Popup'); INSERT INTO AdSchedule VALUES (505, 100, 'Radio', TIMESTAMP '2025-02-20 00:00:00', TIMESTAMP '2025-03-01 23:59:59', 'Audio'); INSERT INTO AdSchedule VALUES (506, 101, 'In-App Banner', TIMESTAMP '2025-03-29 00:00:00', TIMESTAMP '2025-04-06 23:59:59', 'Popup'); INSERT INTO AdSchedule VALUES (507, 102, 'Kids TV', TIMESTAMP '2025-05-02 00:00:00', TIMESTAMP '2025-05-10 23:59:59', 'Video'); INSERT INTO AdSchedule VALUES (508, 103, 'Summer Billboard', TIMESTAMP '2025-06-01 00:00:00', TIMESTAMP '2025-06-30 23:59:59', 'Banner'); INSERT INTO AdSchedule VALUES (509, 104, 'Mothers App', TIMESTAMP '2025-05-05 00:00:00', TIMESTAMP '2025-05-12 23:59:59', 'Story'); INSERT INTO AdSchedule VALUES (510, 100, 'Instagram Reel', TIMESTAMP '2025-02-26 00:00:00', TIMESTAMP '2025-03-01 23:59:59', 'Video'); INSERT INTO AdSchedule VALUES (511, 101, 'Website Promo', TIMESTAMP '2025-03-30 00:00:00', TIMESTAMP '2025-04-05 23:59:59', 'Banner'); INSERT INTO AdSchedule VALUES (512, 102, 'Kids Online', TIMESTAMP '2025-05-03 00:00:00', TIMESTAMP '2025-05-11 23:59:59', 'Popup'); INSERT INTO AdSchedule VALUES (513, 103, 'YouTube Summer', TIMESTAMP '2025-05-27 00:00:00', TIMESTAMP '2025-06-10 23:59:59', 'Video'); INSERT INTO AdSchedule VALUES (514, 104, 'Gift Push', TIMESTAMP '2025-05-02 00:00:00', TIMESTAMP '2025-05-10 23:59:59', 'Popup'); INSERT INTO AdSchedule VALUES (515, 100, 'Outdoor Banners', TIMESTAMP '2025-02-27 00:00:00', TIMESTAMP '2025-03-01 23:59:59', 'Banner'); INSERT INTO AdSchedule VALUES (516, 101, 'Email Campaign', TIMESTAMP '2025-03-31 00:00:00', TIMESTAMP '2025-04-06 23:59:59', 'Text'); INSERT INTO AdSchedule VALUES (517, 102, 'Online Contest', TIMESTAMP '2025-05-04 00:00:00', TIMESTAMP '2025-05-12 23:59:59', 'Popup'); INSERT INTO AdSchedule VALUES (518, 103, 'Summer Radio', TIMESTAMP '2025-06-02 00:00:00', TIMESTAMP '2025-06-15 23:59:59', 'Audio'); INSERT INTO AdSchedule VALUES (519, 104, 'Special Ads', TIMESTAMP '2025-05-06 00:00:00', TIMESTAMP '2025-05-12 23:59:59', 'Story');

-- PROMOANALYTICS INSERT INTO PromoAnalytics VALUES (600, 100, '+15% sales', 12.5, 'Mostly positive'); INSERT INTO PromoAnalytics VALUES (601, 101, '+8% sales', 5.0, 'Good uptake'); INSERT INTO PromoAnalytics VALUES (602, 102, 'N/A', 0.0, 'Hands-on feedback only'); INSERT INTO PromoAnalytics VALUES (603, 103, '+10% sales', 8.8, 'Mixed reviews'); INSERT INTO PromoAnalytics VALUES (604, 104, '+20% web orders', 15.0, 'Excellent'); INSERT INTO PromoAnalytics VALUES (605, 100, '+5% social engagement', 4.5, 'Okay response'); INSERT INTO PromoAnalytics VALUES (606, 101, '+6% customer traffic', 5.8, 'Encouraging'); INSERT INTO PromoAnalytics VALUES (607, 102, '+7% family signups', 6.2, 'Positive'); INSERT INTO PromoAnalytics VALUES (608, 103, '+12% combo meals sold', 10.0, 'Great'); INSERT INTO PromoAnalytics VALUES (609, 104, '+9% gifts purchased', 7.5, 'Good'); INSERT INTO PromoAnalytics VALUES (610, 100, '+11% app installs', 9.8, 'Very good'); INSERT INTO PromoAnalytics VALUES (611, 101, '+13% burger orders', 11.5, 'Very happy'); INSERT INTO PromoAnalytics VALUES (612, 102, '+4% event check-ins', 3.5, 'Average'); INSERT INTO PromoAnalytics VALUES (613, 103, '+6% ice cream sales', 5.7, 'Fair'); INSERT INTO PromoAnalytics VALUES (614, 104, '+15% restaurant bookings', 14.5, 'Excellent'); INSERT INTO PromoAnalytics VALUES (615, 100, '+3% voucher use', 2.9, 'Mild'); INSERT INTO PromoAnalytics VALUES (616, 101, '+8% burger combo redemptions', 7.2, 'Good'); INSERT INTO PromoAnalytics VALUES (617, 102, '+5% kids feedback', 4.8, 'Positive'); INSERT INTO PromoAnalytics VALUES (618, 103, '+10% customer returns', 8.9, 'Very good'); INSERT INTO PromoAnalytics VALUES (619, 104, '+12% Mother''s Day sales', 10.5, 'Fantastic');

-- FEEDBACK INSERT INTO Feedback VALUES (700, 1, 100, 'Great deal!', 5, DATE '2025-03-05'); INSERT INTO Feedback VALUES (701, 2, 100, NULL, 3, DATE '2025-03-10'); INSERT INTO Feedback VALUES (702, 3, 101, 'Loved the burger', 5, DATE '2025-04-03'); INSERT INTO Feedback VALUES (703, 4, 103, 'Not enough items', 2, DATE '2025-06-10'); INSERT INTO Feedback VALUES (704, 5, 104, 'Very convenient', 4, DATE '2025-07-05'); INSERT INTO Feedback VALUES (705, 6, 100, 'Spring event was fun', 5, DATE '2025-03-07'); INSERT INTO Feedback VALUES (706, 7, 101, 'Burger could be bigger', 3, DATE '2025-04-04'); INSERT INTO Feedback VALUES (707, 8, 102, 'Kids loved the games', 5, DATE '2025-05-10'); INSERT INTO Feedback VALUES (708, 9, 102, 'Nice family event', 4, DATE '2025-05-10'); INSERT INTO Feedback VALUES (709, 10, 103, 'Combo was worth it', 5, DATE '2025-06-02'); INSERT INTO Feedback VALUES (710, 11, 103, 'More variety needed', 3, DATE '2025-06-12'); INSERT INTO Feedback VALUES (711, 12, 104, 'Perfect Mother''s Day gift', 5, DATE '2025-05-08'); INSERT INTO Feedback VALUES (712, 13, 100, 'Spring promo average', 3, DATE '2025-03-15'); INSERT INTO Feedback VALUES (713, 14, 101, 'Burger week was tasty', 4, DATE '2025-04-06'); INSERT INTO Feedback VALUES (714, 15, 102, 'Enjoyed kids activities', 5, DATE '2025-05-11'); INSERT INTO Feedback VALUES (715, 16, 104, 'Gift fair awesome', 5, DATE '2025-05-09'); INSERT INTO Feedback VALUES (716, 17, 100, 'Love the vouchers', 5, DATE '2025-03-08'); INSERT INTO Feedback VALUES (717, 18, 103, 'Good deals overall', 4, DATE '2025-06-04'); INSERT INTO Feedback VALUES (718, 19, 101, 'Burger deal fair', 3, DATE '2025-04-05'); INSERT INTO Feedback VALUES (719, 20, 104, 'Mother loved the surprise', 5, DATE '2025-05-10');

-- CUSTOMERSEGMENT INSERT INTO CustomerSegment VALUES (800, 10, 'Points >=100', 'Silver and above'); INSERT INTO CustomerSegment VALUES (801, 11, 'Points >=200', 'Gold members'); INSERT INTO CustomerSegment VALUES (802, 12, 'Joined last 30 days', 'Newbies'); INSERT INTO CustomerSegment VALUES (803, 14, 'Tier = Gold', 'Top tier'); INSERT INTO CustomerSegment VALUES (804, 13, 'Points = 0', 'Inactive'); INSERT INTO CustomerSegment VALUES (805, 15, 'Points between 50-100', 'Moderate spenders'); INSERT INTO CustomerSegment VALUES (806, 16, 'Points >=80', 'Active members'); INSERT INTO CustomerSegment VALUES (807, 17, 'Points <=20', 'Low activity'); INSERT INTO CustomerSegment VALUES (808, 18, 'Joined after Jan 2024', 'Recent joiners'); INSERT INTO CustomerSegment VALUES (809, 19, 'Points between 200-300', 'High-value customers'); INSERT INTO CustomerSegment VALUES (810, 20, 'Points >=150', 'Potential Gold'); INSERT INTO CustomerSegment VALUES (811, 10, 'Birthday month', 'Special offer group'); INSERT INTO CustomerSegment VALUES (812, 11, 'Top 10% Loyalty', 'VIP members'); INSERT INTO CustomerSegment VALUES (813, 12, 'Below 100 Points', 'At-risk'); INSERT INTO CustomerSegment VALUES (814, 13, 'No purchases', 'Non-active'); INSERT INTO CustomerSegment VALUES (815, 14, 'Frequent feedback', 'Engaged users'); INSERT INTO CustomerSegment VALUES (816, 15, 'Voucher users', 'Coupon Lovers'); INSERT INTO CustomerSegment VALUES (817, 16, 'New parents', 'Family group'); INSERT INTO CustomerSegment VALUES (818, 17, 'Students', 'Young adults'); INSERT INTO CustomerSegment VALUES (819, 18, 'Seniors', 'Senior citizens');

-- PROMOTION\_SEGMENT INSERT INTO Promotion\_Segment VALUES (100, 800, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (100, 801, 2, 'SMS'); INSERT INTO Promotion\_Segment VALUES (101, 802, 1, 'In-App'); INSERT INTO Promotion\_Segment VALUES (102, 803, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (103, 804, 1, 'Website'); INSERT INTO Promotion\_Segment VALUES (104, 805, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (100, 806, 2, 'SMS'); INSERT INTO Promotion\_Segment VALUES (101, 807, 1, 'In-App'); INSERT INTO Promotion\_Segment VALUES (102, 808, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (103, 809, 1, 'Website'); INSERT INTO Promotion\_Segment VALUES (104, 810, 1, 'Facebook'); INSERT INTO Promotion\_Segment VALUES (100, 811, 1, 'Instagram'); INSERT INTO Promotion\_Segment VALUES (101, 812, 1, 'Direct Call'); INSERT INTO Promotion\_Segment VALUES (102, 813, 2, 'SMS'); INSERT INTO Promotion\_Segment VALUES (103, 814, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (104, 815, 2, 'Website'); INSERT INTO Promotion\_Segment VALUES (100, 816, 1, 'In-App'); INSERT INTO Promotion\_Segment VALUES (101, 817, 1, 'Email'); INSERT INTO Promotion\_Segment VALUES (102, 818, 1, 'Social Media'); INSERT INTO Promotion\_Segment VALUES (103, 819, 1, 'Email');

-- CS\_MEMBERSHIP INSERT INTO CS\_Membership VALUES (1, 800, DATE '2024-01-01', 'Active'); INSERT INTO CS\_Membership VALUES (2, 801, DATE '2023-05-15', 'Active'); INSERT INTO CS\_Membership VALUES (3, 802, DATE '2024-03-10', 'Active'); INSERT INTO CS\_Membership VALUES (5, 803, DATE '2023-10-20', 'Active'); INSERT INTO CS\_Membership VALUES (4, 804, DATE '2025-02-01', 'Pending'); INSERT INTO CS\_Membership VALUES (6, 805, DATE '2024-02-15', 'Active'); INSERT INTO CS\_Membership VALUES (7, 806, DATE '2024-01-12', 'Active'); INSERT INTO CS\_Membership VALUES (8, 807, DATE '2024-04-11', 'Active'); INSERT INTO CS\_Membership VALUES (9, 808, DATE '2024-04-25', 'Active'); INSERT INTO CS\_Membership VALUES (10, 809, DATE '2023-06-09', 'Active'); INSERT INTO CS\_Membership VALUES (11, 810, DATE '2022-11-20', 'Active'); INSERT INTO CS\_Membership VALUES (12, 811, DATE '2023-12-30', 'Active'); INSERT INTO CS\_Membership VALUES (13, 812, DATE '2024-02-21', 'Active'); INSERT INTO CS\_Membership VALUES (14, 813, DATE '2023-07-16', 'Inactive'); INSERT INTO CS\_Membership VALUES (15, 814, DATE '2023-08-22', 'Inactive'); INSERT INTO CS\_Membership VALUES (16, 815, DATE '2024-04-28', 'Active'); INSERT INTO CS\_Membership VALUES (17, 816, DATE '2023-09-15', 'Active'); INSERT INTO CS\_Membership VALUES (18, 817, DATE '2024-05-19', 'Active'); INSERT INTO CS\_Membership VALUES (19, 818, DATE '2024-05-05', 'Active'); INSERT INTO CS\_Membership VALUES (20, 819, DATE '2024-06-11', 'Pending');

COMMIT;