**FORM 1**



**FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY**

**AACS3013 DATABASE DEVELOPMENT AND APPLICATIONS**

Assignment

**Semester 202301**

|  |  |  |
| --- | --- | --- |
| Programme (Year & Group) | : | Diploma in Computer Science Y1S3 G8 |
| Tutorial Group | : | 8 |
| Date Submitted | : | 26/4/2023 |

Team members:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Name (Block Letters) | Registration No. | Signature | Marks |
| 1 | YEOH MAN TIK | 2208738 | yeoh |  |
| 2 | ALOYSIUS KHOO | 2208707 | aloy |  |
| 3 | KHOO LI XUAN | 2209619 | khoo |  |
| 4 | NICHOLAS LIM SZE WHYE | 2208641 | nick |  |
| 5 | GREGORY CHIA MING FENG | 2208692 | greg |  |



**FORM 2**



**FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY**

**Plagiarism Statement and Guideline for Late Submission of Coursework**

Read, complete, and sign this statement to be submitted with the written report.

**We confirm that we have read and shall comply with all the terms and conditions of TAR University Management and Technology’s plagiarism policy.**

**We declare that this assignment is free from all forms of plagiarism and for all intents and purposes is my own properly derived work.**

Declaration Statement Acknowledged by

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Name (Block Letters)** | **Registration No.** | **Signature** | **Date** |
| 1 | YEOH MAN TIK | 2208738 | yeoh | 24/4/2023 |
| 2 | ALOYSIUS KHOO | 2208707 | aloy | 24/4/2023 |
| 3 | KHOO LI XUAN | 2209619 | khoo | 24/4/2023 |
| 4 | NICHOLAS LIM SZE WHYE | 2208641 | nick | 24/4/2023 |
| 5 | GREGORY CHIA MING FENG | 2208692 | greg | 24/4/2023 |

**Assignment Assessment Form**

**CLO3: Produce database solutions according to the requirements and business scenarios. (P4, PLO3)**

**CLO4: Demonstrate the ability to solve problems and complete tasks in a given business scenario using a database management software. (C3, PLO6)**

Programme: \_\_\_\_\_\_( ) **Member Name**: 1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, 2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, 3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, 4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, 5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Rating (Task 1, 2, 3, 4, 5 & 7) = 1: Very Poor, 2-3: Poor, 4-5: Average, 6-7: Good, 8-10: Excellent Rating (Task 6) = 1: Very Poor, 2: Poor, 3: Average, 4: Good, 5-6: Excellent**

**Task 6**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
| **Task**  **No.** | **Task Descriptions** | **Weightage** | **Criteria** | **1** | **2** | **3** | **4** | **Comment** |
| 1  (CLO 3) | Develop Business rules | 10% | * Include the required and relevant pairs of business rules. * All business rules must be clearly defined, precise, and reflect the policies and procedures of the organization’s operational environment. |  | | | |  |
| 2  (CLO 3) | Develop ERD | 10% | * Transform business rules to a relational database model correctly. * Correct use of Crow’s Foot notations. * Include all necessary entities, attribute & relationships. |  | | | |  |
| 3  (CLO 3) | Develop DBDL | 10% | * Correct use of DBDL format as required * All required entities, attributes and relationships correctly shown * Indicate Primary key and Foreign key clearly |  | | | |  |
| 4  (CLO 4) | Database Design  20% | 10% | * Correct tables, records and fields designed according to the ERD developed. |  | | | |  |
| 10% | * Enforcement of entity integrity rule & referential integrity rule * Appropriate data types, default values and check constraints. |  | | | |  |
| 5  (CLO 4) | Records (Entries) | 10% | * Provide sufficient and quality data records * Well-designed records for adequate and logical choices of queries to be performed |  | | | |  |
| 6  (CLO 4) | Queries Design  30% | 10% | * Flexible query for variety of inputs. Clear & proper identification of information needs. * Apply Accept, Prompt and variable substitution in queries. Flexible query to cater for variety of inputs, use of multiple tables. * Apply Report Formatting features. Meaningful report handlings. Data values formatted accordingly. * **Only SELECT statements.** |  |  |  |  |  |
| 10% |  |  |  |  |  |
| 10% |  |  |  |  |  |
| 7  (CLO 4) | Assignment Report | 10% | * Comprehensive, clarity and completeness coverage * Quality of report presented * Presentation and Q & A |  |  |  |  |  |
| **Assignment Marks / 100** | | |  |  |  |  |  |  |

# 

[1.0 Background Studies……………………………………………………………………………………. 7](#_Toc133323654)

[1.1 Organisation Background 7](#_Toc133323655)

[2.0 Business Rules of the System 8](#_Toc133323663)

[2.1 Entities of System 8](#_Toc133323664)

[Diagram 1.0 Entities of Hotel System 8](#_Toc133323665)

[2.2 Business Rules of Entities 9](#_Toc133323666)

[3.0 Entity-Relationship Modelling 10](#_Toc133323677)

[3.1 Entity-Relationship Diagram 10](#_Toc133323678)

[4.0 Normalization 11](#_Toc133323686)

[4.1 Attributes of entities with keys 11](#_Toc133323687)

[5.0 Create Database in Oracle 12](#_Toc133323698)

[5.1 Country Table 12](#_Toc133323699)

[5.2 Department Table 12](#_Toc133323708)

[5.3 Employee Table 12](#_Toc133323720)

[5.4 Customer Table 12](#_Toc133323733)

[5.5 Reservation Table 13](#_Toc133323744)

[5.6 Cancellation Table 13](#_Toc133323755)

[5.7 Room Table 13](#_Toc133323766)

[5.8 Payment Table 13](#_Toc133323780)

[5.9 Refund Table 13](#_Toc133323790)

[5.10 Feedback Table 14](#_Toc133323800)

[6.0 Sample Data Records (10 Sample records for each table) 14](#_Toc133323814)

[6.1 Country Table 14](#_Toc133323815)

[6.2 Department Table 14](#_Toc133323836)

[6.3 Employee Table 16](#_Toc133323847)

[6.4 Customer Table 16](#_Toc133323858)

[6.5 Reservation Table 17](#_Toc133323869)

[6.6 Cancellation Table 17](#_Toc133323880)

[6.7 Room Table 18](#_Toc133323891)

[6.8 Payment Table 18](#_Toc133323902)

[6.9 Refund Table 19](#_Toc133323913)

[6.10 Feedback Table 19](#_Toc133323924)

[7.0 SQL Queries and Reports 20](#_Toc133323936)

[7.1 Yeoh Man Tik 20](#_Toc133323937)

[7.1.1 Query/Report 1: Total Cancellation made by Customer within 2 dates/a year. 20](#_Toc133323938)

[7.1.2 Query/Report 2: Total Reservations made by Customer within 2 dates/a year. 22](#_Toc133323974)

[7.1.3 Query/Report 3: Ranking of Employee’s Salary by Country 24](#_Toc133323978)

[7.2 Aloysius Khoo 26](#_Toc133323984)

[7.2.1 Query/Report 1: The type of feedback from customer within 2 specified dates. (Negative/Positive) 26](#_Toc133323985)

[7.2.2 Query/Report 2: Total Payment By Country 28](#_Toc133324014)

[7.2.3 Query/Report 3: Refund by country. 30](#_Toc133324048)

[7.3 Gregory Chia Ming Feng 32](#_Toc133324085)

[7.3.1 Query/Report 1: Total pending cancellations between specified dates 32](#_Toc133324086)

[7.3.2 Query/Report 2: Total customer refund between specified dates. 34](#_Toc133324118)

[7.3.3 Query/Report 3: Total customer payment between specified dates. 36](#_Toc133324122)

[7.4 Khoo Li Xuan 38](#_Toc133324152)

[7.4.1 Query/Report 1: Total reservation of each room type within a year. 38](#_Toc133324153)

[7.4.2 Query/Report 2: Amount of Customers Handled by Employees from two specified dates. 40](#_Toc133324181)

[7.4.3 Query/Report 3: Reservation with status pending 42](#_Toc133324220)

[7.5 Nicholas Lim Sze Whye 44](#_Toc133324251)

[7.5.1 Query/Report 1: Total Payment Between 2 Specified Dates 44](#_Toc133324252)

[7.5.2 Query/Report 2: Top returning Customers 46](#_Toc133324292)

[7.5.3 Query/Report 3: Room availability within specific dates 4](#_Toc133324292)8

# 1.0 Background Studies

## 1.1 Organisation Background

Grand Imperium Hotels Group (GIH Hotels and Resorts) has a 200 year history, starting as a Bass brewery in 1777, founded by William Bass. In 2010, it was known as one of the most leading hospitality companies in the world with their CEO, Keith Barr.

GIH Hotels and Resorts provides hotel room rental with the best services to all. Their hotels receive a large number of positive reviews on most major travel websites, including TripAdvisor. Ranking as one of the highest rated hotels in the world.

GIH Hotels and Resorts was founded by Juan Trippe in 2003. GIH Hotels and Resorts is rated one of the most famous and largest luxury hospitality brands running more than 6,000 hotels and more than 888,000 available rooms around the world. The company has over 325,000 workers to deliver true hospitality, prioritising cleanliness and the best service.

GIH Hotels and Resorts owns 17 hotel brands which includes Six Senses Hotels Resorts Spas, Regent International Hotels, Kimpton Hotels and Restaurants, Hotel Indigo, Voco, Hualuxe Hotels and Resorts, Even Hotels, Crowne Plaza, Holiday Inn Express, Holiday Inn, Holiday Inn Club Vacations, Avid Hotels, Atwell Suites, Staybridge Suites and Candlewood Suites.

GIH Hotels and Resorts provide and rent out a wide variation of hotel and resort rooms such as a collection of luxury and lifestyle hotels, premium collection rooms, essential collection hotels, and a wide range of suites.

GIH Hotels and Resorts brand hotels can be found in more than 100 countries internationally. Which includes: The United States of America, The United Kingdom, Mainland China, Canada, Mexico, Germany, France and Japan, London, Paris, The Caribbean, Dubai, Sydney, Hong Kong, Johannesburg, Amsterdam, Toronto and Shanghai. Most of their brands can be found in the US in 10 different states, being Atlanta, Chicago, Dallas, Honolulu, Houston, Los Angeles, New York, Orlando, San Francisco and Washington.

GIH Hotels and Resorts provides guests with their official website. Providing guests an easy-to-use, transparent and efficient platform for room reservations. Customers can access all information from the website such as availability of rooms, price, location and facilities of a hotel or a resort in their destination. GIH Hotels and Resorts has an average annual revenue of $1.5 billion dollars from room rentals, partnerships, sponsorships and advertising from branches worldwide. GIH Hotels and Resorts rent out 32 million rooms annually to visitors, tourists and business travellers from across the globe. Hence, the company is known as one of the highest earning hotel companies today.

# 

# 

# 2.0 Business Rules of the System

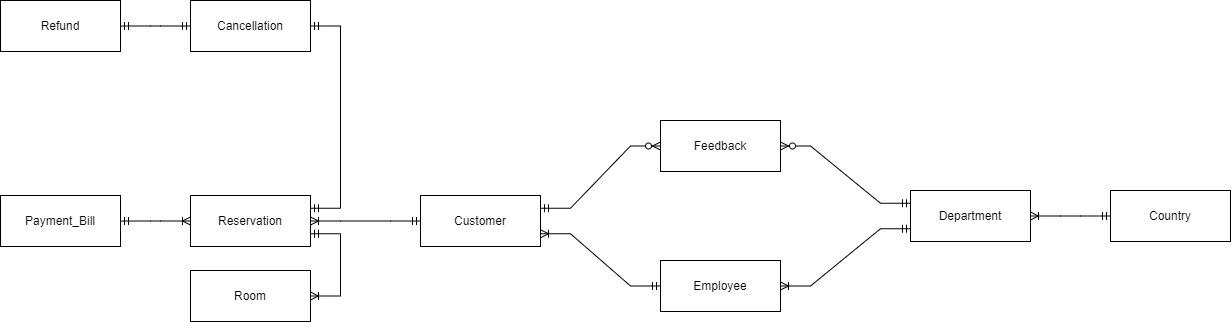
2.1 Entities of System

Diagram 1.0 Entities of Hotel System

## 2.2 Business Rules of Entities

1. Each customer is served by one and only one employee and each employee is serving one or many customers.
2. Each customer can send zero or many feedback and each feedback can be sent by one and only one customer.
3. Each feedback can be sent to one and only one department and each department can receive zero or many feedback.
4. Each employee belongs to one and only one department and each department has one or many employees.
5. Each customer can make one or many reservations and each reservation can be made by one and only one customer.
6. Each reservation can be used to book one or many rooms and each room can be booked by one and only one reservation.
7. Each refund belongs to zero or one cancellation and each cancellation belongs to one and only one refund.
8. Each payment bill is generated by one and only one reservation and each reservation generates one and only one payment bill.
9. Each cancellation is for one and only one reservation and each reservation can make one and only one cancellation.
10. Each department is located in one and only one country and one country can have one or many departments.

# 

# 3.0 Entity-Relationship Modelling

## 3.1 Entity-Relationship Diagram

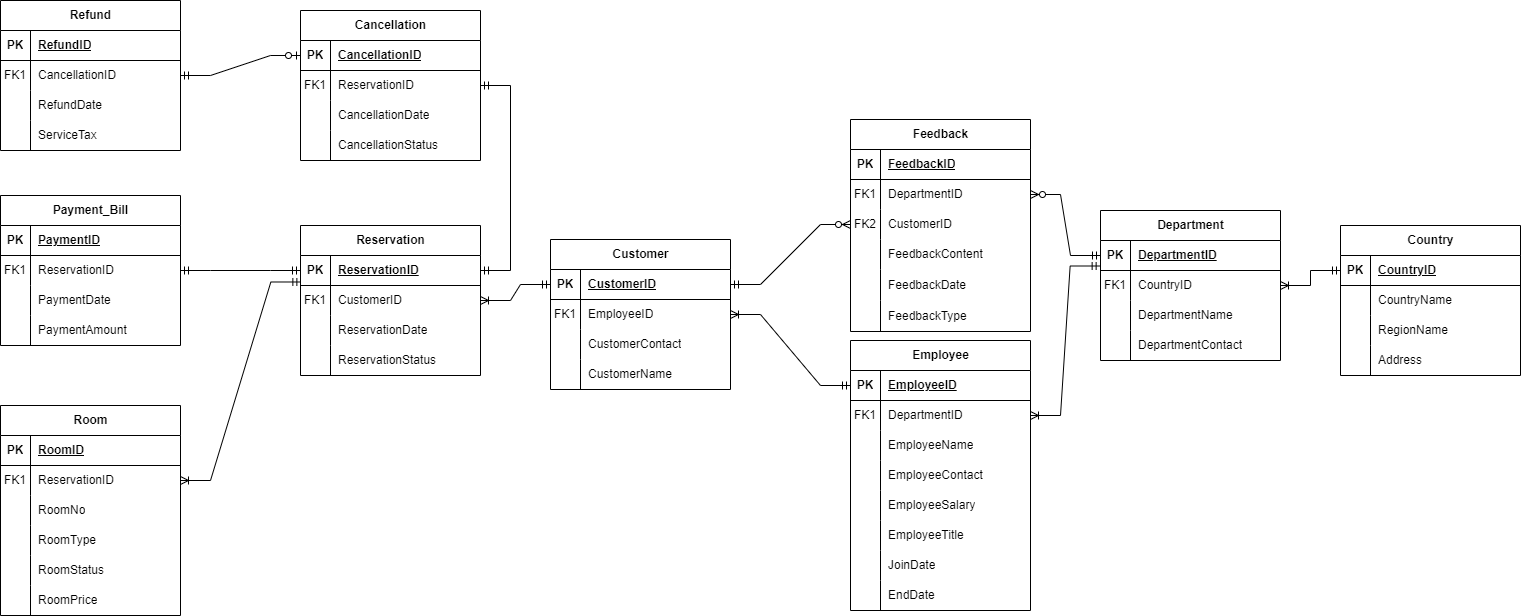


Diagram 1.1 Entity Relationship Diagram of Hotel System

## 

## 3.2 Assumptions

1. Refunds amount will always be lesser than the payment amount when a transaction report is generated.
2. Customers' reservation status will be changed from ‘booked’ to ‘cancelled’ after cancellation was made.
3. Room will be assigned after the customer has made a reservation and confirmed by the employee.
4. Room status will be changed from ‘vacant’ to ‘occupied’ after the customer is present at the hotel.
5. Room price will scale according to the room type selected.

# 4.0 Normalization

## 4.1 Attributes of entities with keys

Refund (**RefundID**, CancellationID\*, RefundDate, ServiceTax)

Payment\_Bill (**PaymentID**, ReservationID\*, PaymentDate, PaymentAmount)

Room (**RoomID**, ReservationID\*, RoomNo, RoomType, RoomStatus, RoomPrice)

Cancellation (**CancellationID**, ReservationID\*, CancellationDate, CancellationStatus)

Reservation (**ReservationID**, CustomerID\*, ReservationDate, ReservationStatus)

Customer (**CustomerID**, EmployeeID\*, CustomerContact, CustomerName)

Feedback (**FeedbackID**, DepartmentID\*, CustomerID\*, FeedbackContent, FeedbackDate, FeedbackType)

Employee (**EmployeeID**, DepartmentID\*, EmployeeName, EmployeeContact, EmployeeSalary, EmployeeTitle, JoinDate, EndDate)

Department (**DepartmentID**, CountryID\*, DepartmentName, DepartmentContact)

Country (**CountryID**, CountryName, RegionName, Address)

# 

# 5.0 Create Database in Oracle

## 5.1 Country Table

create table country (

country\_id VARCHAR(5) not null,

country\_name varchar(25) not null,

region\_name varchar(25) not null,

address varchar(50),

primary key(country\_id),

constraint chk\_countryid check (country\_id like 'CO%')

);

## 5.2 Department Table

create table department (

department\_id VARCHAR(5) not null,

country\_id VARCHAR(5) not null,

department\_name varchar(40),

department\_contact varchar(12),

primary key(department\_id),

foreign key(country\_id) references country(country\_id),

constraint chk\_departmentid check (REGEXP\_LIKE(department\_id,'[DP]{2}[0-9]{3}')),

UNIQUE(department\_contact),

constraint chk\_departmentcontact check (REGEXP\_LIKE(department\_contact,'[0-9]{3}-[0-9]{3}-[0-9]{4}'))

);

## 5.3 Employee Table

CREATE TABLE employee (

emp\_id VARCHAR(5) NOT NULL,

department\_id VARCHAR(5) NOT NULL,

emp\_name VARCHAR(50) NOT NULL,

emp\_contact VARCHAR(20) NOT NULL,

emp\_salary NUMBER(20,2) NOT NULL,

emp\_title VARCHAR2(50) NOT NULL,

PRIMARY KEY(emp\_id),

FOREIGN KEY(department\_id) references department(department\_id),

CONSTRAINT chk\_emp\_id CHECK (emp\_id LIKE 'E%'),

CONSTRAINT chk\_emp\_contact CHECK (REGEXP\_LIKE(emp\_contact, '[0-9]{3}-[0-9]{3}-[0-9]{4}'))

);

## 5.4 Customer Table

CREATE TABLE customer (

cust\_id VARCHAR(6) NOT NULL,

emp\_id VARCHAR(5) NOT NULL,

cust\_name VARCHAR(25) NOT NULL,

cust\_contact VARCHAR(35),

PRIMARY KEY(cust\_id),

FOREIGN KEY(emp\_id) REFERENCES employee(emp\_id),

CONSTRAINT chk\_cust\_id CHECK (cust\_id LIKE 'CU%'),

CONSTRAINT chk\_cust\_contact CHECK (REGEXP\_LIKE(cust\_contact, '[0-9]{3}-[0-9]{3}-[0-9]{4}'))

);

## 5.5 Reservation Table

CREATE TABLE reservation (

reservation\_id VARCHAR(6) NOT NULL,

cust\_id VARCHAR(6) NOT NULL,

reservationDate DATE NOT NULL,

reservationStatus VARCHAR(10) NOT NULL,

PRIMARY KEY(reservation\_id),

FOREIGN KEY(cust\_id) REFERENCES customer(cust\_id),

CONSTRAINT chk\_reservation\_id CHECK (reservation\_id LIKE 'RE%'),

CONSTRAINT chk\_reseration\_status CHECK (reservationStatus IN ('Pending', 'Cancelled', 'Booked'))

);

## 5.6 Cancellation Table

CREATE TABLE cancellation (

cancellation\_id VARCHAR(6) NOT NULL,

reservation\_id VARCHAR(6) NOT NULL,

cancellation\_date DATE NOT NULL,

cancellation\_status VARCHAR(10) NOT NULL,

PRIMARY KEY(cancellation\_id),

FOREIGN KEY(reservation\_id) REFERENCES reservation(reservation\_id),

CONSTRAINT chk\_cancellation\_id CHECK (cancellation\_id LIKE 'CA%'),

CONSTRAINT chk\_cancellation\_status CHECK (cancellation\_status IN ('Pending', 'Approved', 'Revoked'))

);

## 5.7 Room Table

CREATE TABLE room (

room\_id VARCHAR(6) NOT NULL,

reservation\_id VARCHAR(6) NOT NULL,

roomNo NUMBER(4) NOT NULL,

roomType VARCHAR(20) NOT NULL,

roomStatus VARCHAR(10) NOT NULL,

roomPrice NUMBER(5,2) NOT NULL,

PRIMARY KEY(room\_id),

FOREIGN KEY(reservation\_id) REFERENCES reservation(reservation\_id),

CONSTRAINT chk\_room\_id CHECK (room\_id LIKE 'RO%'),

CONSTRAINT chk\_roomStatus CHECK (roomStatus IN ('Occupied','Vacant','Cleaned')),

CONSTRAINT chk\_roomtype CHECK (roomType IN ('Standard', 'Deluxe', 'Executive' ))

);

## 5.8 Payment Table

CREATE TABLE payment (

payment\_id VARCHAR(6) NOT NULL,

reservation\_id VARCHAR(6) NOT NULL,

payment\_date DATE NOT NULL,

payment\_amount DECIMAL(6, 2) NOT NULL,

PRIMARY KEY (payment\_id),

CONSTRAINT chk\_payment\_id CHECK (payment\_id LIKE 'PB%'),

CONSTRAINT chk\_payment\_amount CHECK (payment\_amount > 0)

);

## 5.9 Refund Table

CREATE TABLE refund (

refund\_id VARCHAR(6) NOT NULL,

cancellation\_id VARCHAR(6) NOT NULL,

refund\_date DATE NOT NULL,

service\_tax NUMBER(5,2) NOT NULL,

PRIMARY KEY(refund\_id),

FOREIGN KEY(cancellation\_id) REFERENCES cancellation(cancellation\_id),

CONSTRAINT chk\_refundid CHECK (refund\_id LIKE 'RU%')

);

## 5.10 Feedback Table

CREATE TABLE feedback (

feedback\_id VARCHAR(6) NOT NULL,

department\_id VARCHAR(5) NOT NULL,

cust\_id VARCHAR(6) NOT NULL,

feedback\_content VARCHAR(50) NOT NULL,

feedback\_date DATE NOT NULL,

feedback\_type VARCHAR(9),

primary key(feedback\_id),

foreign key(department\_id) references department(department\_id),

FOREIGN KEY(cust\_id) REFERENCES customer(cust\_id),

constraint chk\_feedback\_id check (feedback\_id LIKE 'FE%'),

constraint chk\_feedback\_type CHECK (feedback\_type IN ('Positive', 'Negative'))

);

# 6.0 Sample Data Records (10 Sample records for each table)

## 6.1 Country Table

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO001', 'Malaysia', 'Southeast Asia', 'Kuala Lumpur');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO002', 'Singapore', 'Southeast Asia', 'Singapore');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO003', 'Thailand', 'Southeast Asia', 'Bangkok');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO004', 'Canada', 'North America', 'Ottawa');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO005', 'Malaysia', 'Southeast Asia', 'Penang');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO006', 'Singapore', 'Southeast Asia', 'Jurong');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO007', 'Thailand', 'Southeast Asia', 'Chiang Mai');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO008', 'Canada', 'North America', 'Toronto');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO009', 'Malaysia', 'Southeast Asia', 'Johor Bahru');

INSERT INTO country (country\_id, country\_name, region\_name, address)

VALUES ('CO010', 'Singapore', 'Southeast Asia', 'Sentosa');

## 6.2 Department Table

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP001', 'CO001', 'Reservation Department', '016-123-4567');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP002', 'CO001', 'Accounting Department', '012-345-6789');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP003', 'CO001', 'Hospitality Department', '014-567-8901');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP004', 'CO001', 'Sales Department', '010-987-6543');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP005', 'CO002', 'Reservation Department', '016-111-2222');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP006', 'CO002', 'Accounting Department', '010-333-4444');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP007', 'CO002', 'Hospitality Department', '014-555-6666');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP008', 'CO002', 'Sales Department', '013-777-8888');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP009', 'CO003', 'Reservation Department', '016-237-0747');

insert into department (department\_id, country\_id, department\_name, department\_contact) values ('DP010', 'CO003', 'Accounting Department', '016-833-0390');

## 6.3 Employee Table

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E001', 'DP004', 'Sadella Eicke', '013-866-1251', 'Department Manager', '6000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E002', 'DP008', 'Trudey Gowling', '013-535-5796', 'Department Director', '12000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E003', 'DP012', 'Horatio Hoofe', '013-022-9571', 'Department Assistant Director', '9000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E004', 'DP015', 'Brooke Quaif', '013-571-5724', 'Sales Supervisor', '4000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E005', 'DP019', 'Wally Maylin', '013-310-7893', 'Sales Assistant', '2000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E006', 'DP023', 'Raff Leonida', '013-659-1254', 'Department Manager', '6000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E007', 'DP027', 'Yoshi Ingman', '013-423-5691', 'Department Director', '12000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E008', 'DP031', 'Kendricks Meininger', '013-056-0203', 'Department Assistant Director', '9000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E009', 'DP035', 'Bee Emby', '013-907-8020', 'Sales Supervisor', '4000.00');

insert into employee (emp\_id, department\_id, emp\_name, emp\_contact, emp\_title, emp\_salary) values ('E010', 'DP039', 'Coriss Uvedale', '013-251-0690', 'Sales Assistant', '2000.00');

## 6.4 Customer Table

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0001', 'E002', 'Elton Smith', '010-567-8901');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0002', 'E002', 'John Smith', '234-567-8901');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0003', 'E003', 'Sarah Johnson', '345-678-9012');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0004', 'E004', 'David Lee', '456-789-0123');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0005', 'E005', 'Karen Davis', '567-890-1234');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0006', 'E006', 'Mike Wilson', '678-901-2345');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0007', 'E007', 'Lisa Brown', '789-012-3456');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0008', 'E008', 'Tom Jackson', '890-123-4567');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0009', 'E009', 'Emily Scott', '901-234-5678');

INSERT INTO customer (cust\_id, emp\_id, cust\_name, cust\_contact) VALUES ('CU0010', 'E010', 'Ryan Lee', '012-345-6789');

## 6.5 Reservation Table

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0001', 'CU0001', '2023/09/04', 'Booked');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0002', 'CU0001', '2023/04/15', 'Pending');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0003', 'CU0001', '2023/11/19', 'Cancelled');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0004', 'CU0001', '2023/08/13', 'Booked');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0005', 'CU0001', '2023/07/31', 'Pending');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0006', 'CU0001', '2023/12/03', 'Cancelled');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0007', 'CU0001', '2023/04/03', 'Booked');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0008', 'CU0001', '2023/10/09', 'Pending');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0009', 'CU0001', '2023/05/24', 'Cancelled');

insert into reservation (reservation\_id, cust\_id, reservationDate, reservationStatus) values ('RE0010', 'CU0001', '2023/02/04', 'Booked');

## 6.6 Cancellation Table

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0001', 'RE0003', '2023/11/22', 'Approved');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0002', 'RE0006', '2023/04/15', 'Revoked');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0003', 'RE0009', '2023/08/04', 'Pending');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0004', 'RE0012', '2023/06/29', 'Approved');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0005', 'RE0015', '2023/01/09', 'Revoked');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0006', 'RE0018', '2023/10/27', 'Revoked');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0007', 'RE0021', '2023/12/04', 'Pending');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0008', 'RE0024', '2023/06/01', 'Revoked');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0009', 'RE0027', '2023/09/30', 'Approved');

insert into cancellation (cancellation\_id, reservation\_id, cancellation\_date, cancellation\_status) values ('CA0010', 'RE0030', '2023/04/22', 'Revoked');

## 6.7 Room Table

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0001', 'RE0001', 'Executive', 'Cleaned', 334, 900);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0002', 'RE0004', 'Executive', 'Occupied', 200, 900);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0003', 'RE0007', 'Executive', 'Vacant', 336, 900);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0004', 'RE0010', 'Deluxe', 'Cleaned', 342, 400);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0005', 'RE0013', 'Deluxe', 'Occupied', 209, 400);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0006', 'RE0016', 'Deluxe', 'Vacant', 394, 400);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0007', 'RE0019', 'Standard', 'Cleaned', 326, 200);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0008', 'RE0022', 'Standard', 'Occupied', 359, 200);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0009', 'RE0025', 'Standard', 'Vacant', 308, 200);

insert into room (room\_id, reservation\_id, roomType, roomStatus, roomNo, roomPrice) values ('RO0010', 'RE0028', 'Executive', 'Cleaned', 328, 900);

## 6.8 Payment Table

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0001', 'RE0001', '2023/07/31', 494.56);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0002', 'RE0004', '2023/05/28', 862.7);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0003', 'RE0007', '2023/07/04', 383.97);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0004', 'RE0010', '2023/07/29', 329.34);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0005', 'RE0013', '2023/11/07', 367.48);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0006', 'RE0016', '2023/05/30', 329.88);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0007', 'RE0019', '2023/01/19', 308.98);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0008', 'RE0022', '2023/05/05', 767.96);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0009', 'RE0025', '2023/05/08', 424.78);

insert into payment (payment\_id, reservation\_id, payment\_date, payment\_amount) values ('PB0010', 'RE0028', '2023/11/15', 312.12);

## 6.9 Refund Table

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0001', 'CA0001', '2023/09/15', 133.45);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0002', 'CA0004', '2023/11/14', 217.79);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0003', 'CA0009', '2023/01/03', 279.35);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0004', 'CA0011', '2023/07/24', 72.07);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0005', 'CA0012', '2023/03/26', 26.92);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0006', 'CA0016', '2023/01/31', 152.38);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0007', 'CA0018', '2023/02/09', 232.94);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0008', 'CA0019', '2023/07/20', 96.93);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0009', 'CA0021', '2023/07/24', 262.37);

insert into refund (refund\_id, cancellation\_id, refund\_date, service\_tax) values ('RU0010', 'CA0024', '2023/06/22', 286.51);

## 6.10 Feedback Table

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0001', 'DP001', 'CU0001', 'Great service, thank you!', TO\_DATE('01-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0002', 'DP003', 'CU0002', 'The food was amazing!', TO\_DATE('02-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0003', 'DP003', 'CU0003', 'The room was clean and comfortable.', TO\_DATE('03-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0004', 'DP005', 'CU0001', 'The room was good.', TO\_DATE('03-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0005', 'DP005', 'CU0005', 'The staff was friendly and helpful.', TO\_DATE('05-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type)

VALUES ('FE0006', 'DP001', 'CU0006', 'I had a great time and will definitely come back!', TO\_DATE('06-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0007', 'DP002', 'CU0007', 'The restaurant had a great atmosphere.', TO\_DATE('07-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0008', 'DP003', 'CU0008', 'The spa services were amazing.', TO\_DATE('08-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0009', 'DP004', 'CU0009', 'The golf course was beautiful.', TO\_DATE('09-FEB-2023', 'DD-MON-YYYY'), 'Positive');

INSERT INTO feedback (feedback\_id, department\_id, cust\_id, feedback\_content, feedback\_date, feedback\_type) VALUES ('FE0010', 'DP005', 'CU0010', 'Event was well-organized', TO\_DATE('10-FEB-2023', 'DD-MON-YYYY'), 'Positive');

# 7.0 SQL Queries and Reports

## 7.1 Yeoh Man Tik

## 7.1.1 Query/Report 1: Total Cancellation made by Customer within 2 dates/a year.

Purpose: The purpose of this report is to display and identify total amount of ‘approved’ cancellation made by customers within 2 specified date arranged from highest to lowest.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

-- 01/01/2023 & 31/12/2023 --

PROMPT 'Total amount of cancellation made by customer within a year/2 specified dates'

PROMPT

PROMPT

BREAK ON REPORT ON cust\_id NODUPLICATES

ACCEPT v\_firstdate FORMAT A10 PROMPT 'Enter First Date: '

ACCEPT v\_seconddate FORMAT A10 PROMPT 'Enter Second Date: '

ACCEPT v\_rownum FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_cancellation FORMAT 9999 HEADING "Cancellation";

COLUMN refund\_amount FORMAT 99999.99 HEADING "Refund Amount";

COLUMN average\_refund FORMAT 99999.99 HEADING "Average Refund";

COLUMN YEAR FORMAT 9999 HEADING "Year";

TTITLE LEFT 'Total cancellation made by customer from ' &v\_firstdate ' TO ' &v\_seconddate' ' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT EXTRACT(YEAR FROM C.cancellation\_date) AS year, N.cust\_id, N.cust\_name, COUNT(C.cancellation\_id) AS total\_cancellation, SUM(RU.service\_tax) AS refund\_amount, AVG(RU.service\_tax) AS average\_refund

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN cancellation C ON R.reservation\_id = C.reservation\_id

JOIN refund RU ON C.cancellation\_id = RU.cancellation\_id

WHERE C.cancellation\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate' AND C.cancellation\_status = 'Approved'

GROUP BY EXTRACT(YEAR FROM C.cancellation\_date), N.cust\_id, N.cust\_name

ORDER BY total\_cancellation DESC;

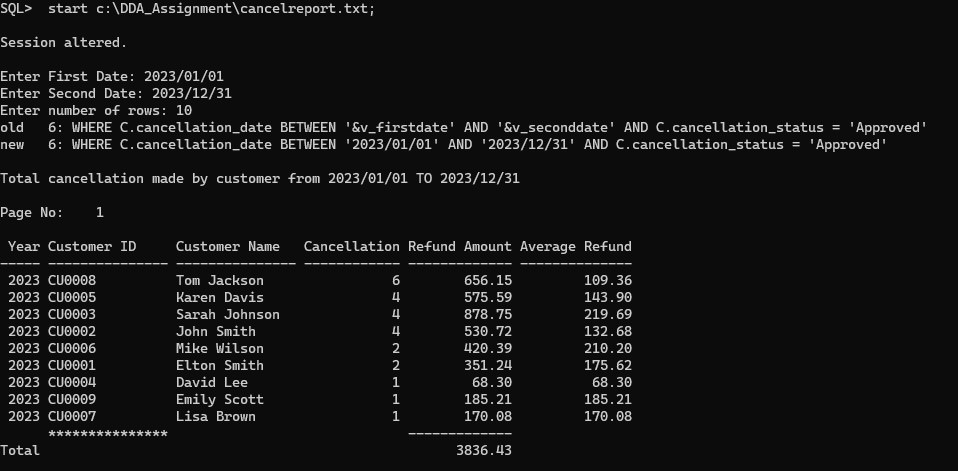
COMPUTE LABEL 'Total (RM): 'OF refund\_amount ON REPORT

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:



## 7.1.2 Query/Report 2: Total Reservations made by Customer within 2 dates/a year.

Purpose: The purpose of this report is to display and identify total amount of ‘booked’ reservation made by customers within 2 specified date arranged from highest to lowest.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

-- 01/01/2023 & 31/12/2023 --

PROMPT 'Total amount of reservation made by customer within a year/2 specified dates'

PROMPT

PROMPT

BREAK ON REPORT ON cust\_id NODUPLICATES

ACCEPT v\_firstdate CHAR FORMAT 'A11' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate CHAR FORMAT 'A11' PROMPT ' Enter the second date: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT ' Enter number of rows: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_reservation FORMAT 9999 HEADING "Total Reservation";

COLUMN paymentAmt FORMAT 99999.99 HEADING "Payment Amount";

COLUMN avg\_payment FORMAT 99999.99 HEADING "Average Payment";

TTITLE LEFT 'Total amount of reservation made by customer from &v\_firstdate TO &v\_seconddate' SKIP 2 -

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 1

SELECT EXTRACT(YEAR FROM R.reservationDate) AS year, C.cust\_id, C.cust\_name, COUNT(R.reservation\_id) AS total\_reservation, SUM(P.payment\_amount) AS paymentAmt, AVG(P.payment\_amount) AS avg\_payment

FROM Customer C

JOIN Reservation R ON C.cust\_id = R.cust\_id

JOIN payment P ON R.reservation\_id = P.reservation\_id

WHERE R.reservationDate BETWEEN '&v\_firstdate' AND '&v\_seconddate' AND R.reservationStatus = 'Booked'

GROUP BY EXTRACT(YEAR FROM R.reservationDate), C.cust\_id, C.cust\_name

ORDER BY total\_reservation DESC;

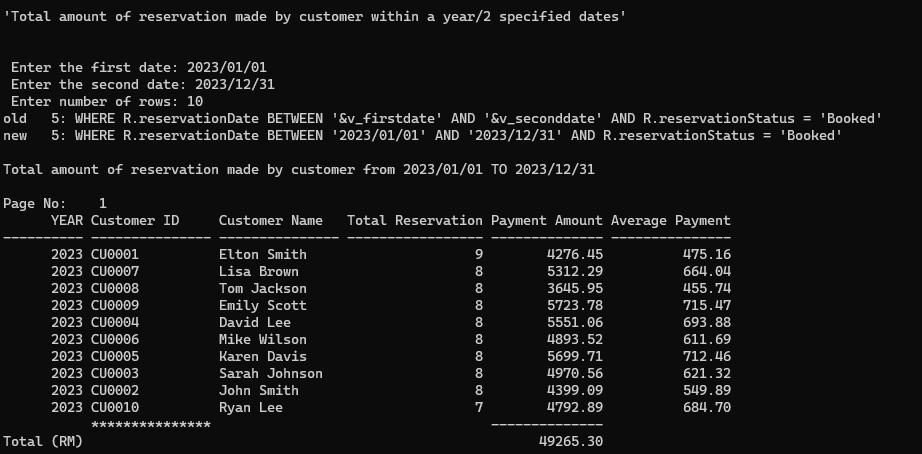
COMPUTE SUM LABEL 'Total (RM): ' OF paymentAmt ON REPORT

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:



## 7.1.3 Query/Report 3: Ranking of Employee’s Salary by Country

Purpose: To determine the employee with highest salary from different countries.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

-- 01/01/2023 & 31/12/2023 --

BREAK ON emp\_id ON REPORT

PROMPT 'Highest Employee Salary with their respective Country and Department'

PROMPT

PROMPT

ACCEPT v\_country CHAR FORMAT 'A20' PROMPT 'Enter country name: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN emp\_id FORMAT A15 HEADING "Employee ID"

COLUMN emp\_name FORMAT A30 HEADING "Employee Name"

COLUMN emp\_title FORMAT A30 HEADING "Title"

COLUMN country\_name FORMAT A11 HEADING "Country"

COLUMN department\_name FORMAT A25 HEADING "Department"

COLUMN emp\_salary FORMAT 99999.99 HEADING 'Salary(RM)'

COLUMN totalsalary FORMAT 999999.99 HEADING 'Total Salary(RM)'

TTITLE LEFT 'Employee Salary Ranking from ' &v\_country ' ' SKIP 1-

LEFT ' ' SKIP 1-

RIGHT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT E.emp\_id, E.emp\_name, E.emp\_title, E.emp\_salary, D.department\_name, C.country\_name

FROM country C

JOIN department D ON C.country\_id = D.country\_id

JOIN employee E ON D.department\_id = E.department\_id

WHERE C.country\_name = '&v\_country' AND ROWNUM <= &v\_numberRows

ORDER BY E.emp\_salary DESC;

TTITLE OFF

COMPUTE SUM LABEL 'TOTAL (RM) ' OF emp\_salary ON REPORT

CLEAR COLUMNS

CLEAR BREAKS

Sample Output 1:

Text

Description automatically generated

Sample Output 2:

Text

Description automatically generated

## 7.2 Aloysius Khoo

## 7.2.1 Query/Report 1: The type of feedback from customer within 2 specified dates. (Negative/Positive)

Purpose: To display and identify the type of feedback received from customer within 2 specified dates such as negative feedback and positive feedback.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'FEEDBACK RECEIVE BETWEEN SPECIFIC DATES'

PROMPT '======================================='

PROMPT

PROMPT

ACCEPT v\_firstDate char FORMAT 'A11' PROMPT'Enter first date :'

ACCEPT v\_secondDate char FORMAT 'A11' PROMPT'Enter second date:'

ACCEPT v\_feedbackType char FORMAT 'A10' PROMPT'Enter feedback type (Positive/Negative):'

COLUMN feedback\_id FORMAT A11 HEADING "FEEDBACK ID"

COLUMN cust\_id FORMAT A11 HEADING "CUSTOMER ID"

COLUMN cust\_name FORMAT A25 HEADING "CUSTOMER NAME"

COLUMN feedback\_date FORMAT A13 HEADING "FEEDBACK DATE"

COLUMN feedback\_content FORMAT A60 HEADING "Feedback Content"

COLUMN feedback\_type FORMAT A15 HEADING "Feedback Type"

COLUMN feedback\_date FORMAT A15 HEADING "Feedback Date"

TTITLE 'The '&v\_feedbackType' Feedback by customer from ' &v\_firstdate ' To ' &v\_seconddate' ' SKIP 2

SELECT F.feedback\_id, F.cust\_id, D.department\_name, C.cust\_name, F.feedback\_content, F.feedback\_type, F.feedback\_date

FROM department D

JOIN feedback F ON D.department\_id = F.department\_id

JOIN customer C ON C.cust\_id = F.cust\_id

WHERE feedback\_date BETWEEN '&v\_firstDate' AND '&v\_secondDate' AND feedback\_type='&v\_feedbackType'

ORDER BY feedback\_date DESC;

Sample Output (Negative):

Graphical user interface, text

Description automatically generated

Sample Output (Positive): Text

Description automatically generated

## 7.2.2 Query/Report 2: Total Payment By Country

## Purpose: To identify the total payment of customers from different countries selected by user.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'Total Payment in Country'

PROMPT '========================'

PROMPT

PROMPT

ACCEPT v\_country char FORMAT A25 PROMPT 'Enter Country Name :'

COLUMN country\_id format A11;

COLUMN country\_id heading "COUNTRY ID";

COLUMN cust\_id format A11;

COLUMN cust\_id heading "CUSTOMER ID";

COLUMN country\_name heading "COUNTRY NAME"

COLUMN country\_name format A25;

COLUMN cust\_name heading "CUSTOMER NAME";

COLUMN cust\_name format A25;

COLUMN department\_name heading "DEPARTMENT NAME";

COLUMN department\_name format A40;

TTITLE LEFT 'The Sales in '&v\_country'' ' SKIP 1-

LEFT ' ' SKIP 1-

RIGHT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT C.country\_id, C.country\_name, D.department\_name, Q.cust\_id, Q.cust\_name, COUNT(P.payment\_id) AS Total\_Payment\_Done, SUM(P.payment\_amount) AS Total\_Payment\_Amount

FROM country C

JOIN department D ON C.country\_id = D.country\_id

JOIN employee E ON D.department\_id = E.department\_id

JOIN customer Q ON E.emp\_id = Q.emp\_id

JOIN reservation R ON Q.cust\_id = R.cust\_id

JOIN payment P ON R.reservation\_id = P.reservation\_id

WHERE C.country\_name = '&v\_country'

GROUP BY C.country\_id, C.country\_name, D.department\_name, Q.cust\_id, Q.cust\_name;

Sample Output:

Text

Description automatically generated

## 7.2.3 Query/Report 3: Refund by country.

## Purpose: To determine the total amount of refund from different countries selected by user.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'Total Refund in Country'

PROMPT '========================'

PROMPT

PROMPT

ACCEPT v\_country char FORMAT 'A25' PROMPT 'Enter Country Name :'

COLUMN country\_id format A11;

COLUMN country\_id heading "COUNTRY ID";

COLUMN cust\_id format A11;

COLUMN cust\_id heading "CUSTOMER ID";

COLUMN country\_name heading "COUNTRY NAME"

COLUMN country\_name format A25;

COLUMN cust\_name heading "CUSTOMER NAME";

COLUMN cust\_name format A25;

COLUMN department\_name heading "DEPARTMENT NAME";

COLUMN department\_name format A40;

COLUMN total\_refund HEADING "TOTAL REFUND"

COLUMN total\_tax\_charges HEADING "TOTAL TAX"

TTITLE LEFT 'The Cancelled in '&v\_country''

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT C.country\_name, D.department\_name, Q.cust\_id, Q.cust\_name, COUNT(K.cancellation\_id) AS Total\_REFUND, SUM(K.service\_tax) AS Total\_Tax\_ChargeS

FROM country C

JOIN department D ON C.country\_id = D.country\_id

JOIN employee E ON D.department\_id = E.department\_id

JOIN customer Q ON E.emp\_id = Q.emp\_id

JOIN reservation R ON Q.cust\_id = R.cust\_id

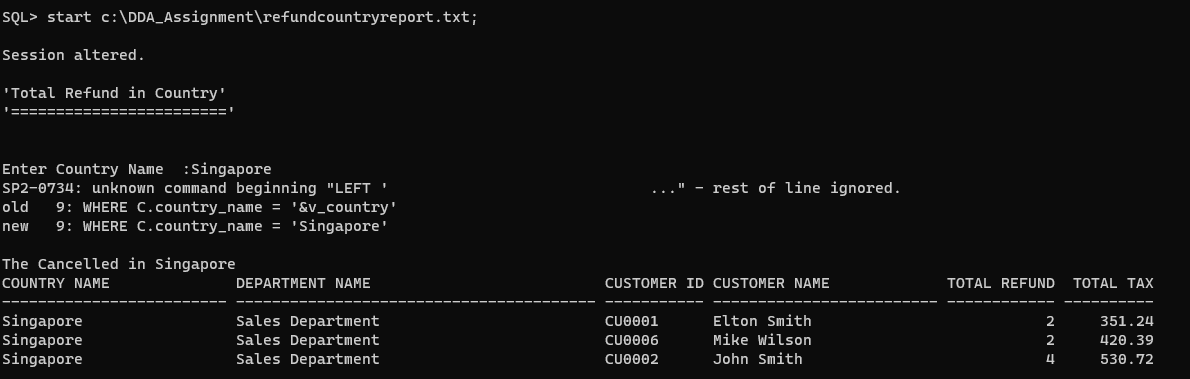
JOIN cancellation H ON R.reservation\_id = H.reservation\_id

JOIN refund K ON H.cancellation\_id = K.cancellation\_id

WHERE C.country\_name = '&v\_country'

GROUP BY C.country\_id, C.country\_name, D.department\_name, Q.cust\_id, Q.cust\_name;

Sample Output:



## 7.3 Gregory Chia Ming Feng

## 7.3.1 Query/Report 1: Total pending cancellations between specified dates

Purpose: To identify the total amount of pending cancellations between 2 specified dates input by the user.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

-- 01/01/2023 & 31/12/2023 --

PROMPT 'Total pending cancellations between specified dates'

PROMPT '---------------------------------------------------'

PROMPT

ACCEPT v\_firstdate CHAR FORMAT 'A11' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate CHAR FORMAT 'A11' PROMPT ' Enter the second date: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_cancellation FORMAT 9999 HEADING "Total Cancellation";

COLUMN cancellation\_status FORMAT A10 HEADING "Status";

TTITLE LEFT 'Total cancellation made by customer from ' &v\_firstdate ' TO ' &v\_seconddate' with Pending status' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT N.cust\_id, N.cust\_name, COUNT(C.cancellation\_id) AS total\_cancellation, C.cancellation\_status

FROM customer N

JOIN reservation R

ON N.cust\_id = R.cust\_id

JOIN cancellation C

ON R.reservation\_id = C.reservation\_id

WHERE C.cancellation\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate' AND C.cancellation\_status = 'Pending'

GROUP BY N.cust\_id, N.cust\_name, C.cancellation\_status

ORDER BY total\_cancellation DESC;

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:

Text

Description automatically generated

## 7.3.2 Query/Report 2: Total customer refund between specified dates.

Purpose: To determine the total refund from each customer between 2 specified dates and sorted from the highest refund to the lowest.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'Total refund made by customer within specified dates'

PROMPT '----------------------------------------------------'

PROMPT

ACCEPT v\_firstdate CHAR FORMAT 'A11' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate CHAR FORMAT 'A11' PROMPT ' Enter the second date: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_refund FORMAT 9999 HEADING "Total Refund(s)";

COLUMN refund\_amount FORMAT $9999.99 HEADING "Refund Amount";

COLUMN avg\_refund\_amount FORMAT $9999.99 HEADING "Average Refund Amount";

TTITLE LEFT 'Total refund from ' &v\_firstdate ' TO ' &v\_seconddate' ' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT \* FROM(

SELECT N.cust\_id, N.cust\_name, COUNT(RU.refund\_id) AS total\_refund, SUM(RU.service\_tax) AS refund\_amount

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN cancellation C ON R.reservation\_id = C.reservation\_id

JOIN refund RU ON C.cancellation\_id = RU.cancellation\_id

WHERE RU.refund\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY N.cust\_id, N.cust\_name

ORDER BY refund\_amount DESC

) WHERE ROWNUM <= '&v\_numberRows';

PROMPT 'Average Refund Amount: '

SELECT AVG(refund\_amount) AS avg\_refund\_amount

FROM (

SELECT SUM(RU.service\_tax) AS refund\_amount

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN cancellation C ON R.reservation\_id = C.reservation\_id

JOIN refund RU ON C.cancellation\_id = RU.cancellation\_id

WHERE RU.refund\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY N.cust\_id, N.cust\_name

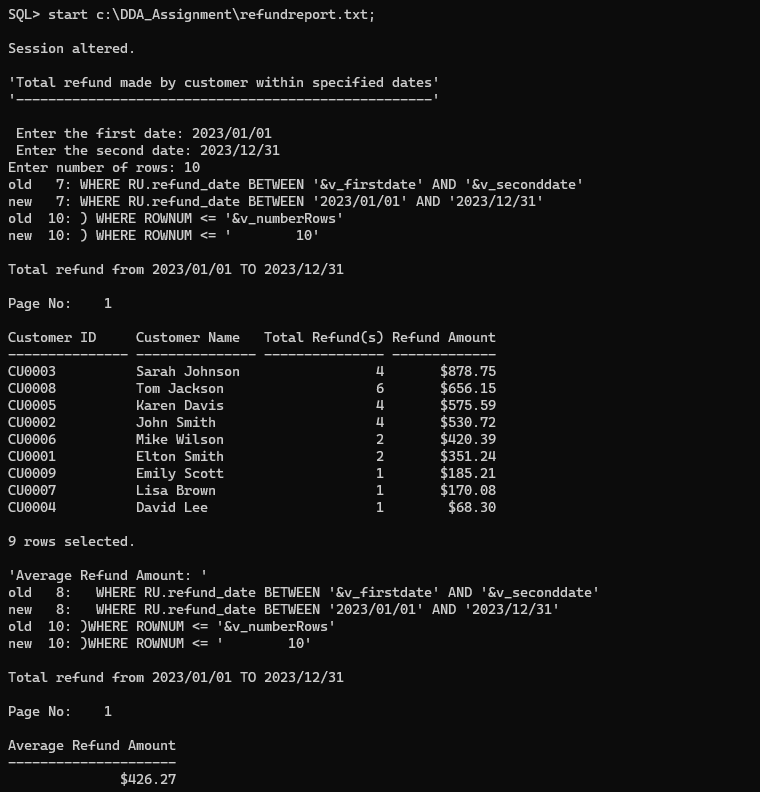
)WHERE ROWNUM <= '&v\_numberRows';

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:



## 7.3.3 Query/Report 3: Total customer payment between specified dates.

Purpose: To determine the total payment from customer between 2 specified dates and sorted from the highest payment to the lowest.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

-- 01/01/2023 & 31/12/2023 --

PROMPT 'Highest payment from customers between specified dates'

PROMPT '---------------------------------------------------'

PROMPT

ACCEPT v\_firstdate CHAR FORMAT 'A11' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate CHAR FORMAT 'A11' PROMPT ' Enter the second date: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_payment FORMAT 9999 HEADING "Total Payments";

COLUMN payment\_amount FORMAT $9999.99 HEADING "Payment Amount";

TTITLE LEFT 'Highest Payment made by customer from ' &v\_firstdate ' TO ' &v\_seconddate' with Pending status' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT N.cust\_id, N.cust\_name, COUNT(P.payment\_id) AS total\_payments, SUM(P.payment\_amount) AS payment\_amount

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN payment P ON R.reservation\_id = P.reservation\_id

WHERE P.payment\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY N.cust\_id, N.cust\_name

ORDER BY payment\_amount DESC;

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:

Text

Description automatically generated

## 

## 7.4 Khoo Li Xuan

## 7.4.1 Query/Report 1: Total reservation of each room type within a year.

Purpose: To identify the total amount of reservation from each room type within a year.

SET LINESIZE 120

SET PAGESIZE 30

BREAK ON YEAR ON room\_id NODUPLICATES

ACCEPT v\_year CHAR FORMAT A5 PROMPT 'Enter a year: '

ACCEPT v\_rowNum NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN YEAR HEADING 'Year'

COLUMN roomType FORMAT A20 HEADING 'Room Type'

COLUMN total\_reservation HEADING 'Total Reservations'

COLUMN total\_payment HEADING 'Total Payment'

TTITLE 'Total Reservation of each room type in ' &v\_year SKIP 1 -

LEFT '-------------------------------------------------' SKIP 2 -

LEFT 'Page: ' FORMAT 999 SQL.PNO SKIP 2

COMPUTE SUM LABEL 'Total (RM): 'OF total\_payment ON YEAR

SELECT \*

FROM (

SELECT EXTRACT(YEAR FROM E.reservationDate) AS Year, R.roomType, COUNT(\*) AS total\_reservation, SUM(P.payment\_amount) AS total\_payment

FROM reservation E

JOIN room R ON E.reservation\_id = R.reservation\_id

JOIN payment P ON E.reservation\_id = P.reservation\_id

WHERE E.reservationStatus = 'Booked'

GROUP BY EXTRACT(YEAR FROM E.reservationDate), R.roomType

ORDER BY R.roomType DESC

)

WHERE ROWNUM <= &v\_rowNum;

Sample Output:

Text

Description automatically generated

## 7.4.2 Query/Report 2: Amount of Customers Handled by Employees from two specified dates.

Purpose: To identify the amount of customers handled by which employees between two specified dates.

SET LINESIZE 200

SET PAGESIZE 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'DD/MM/YYYY';

-- 01/01/2023 & 31/12/2023 --

PROMPT '---------------------------------------------------'

PROMPT

ACCEPT v\_firstdate DATE FORMAT 'dd/mm/yyyy' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate DATE FORMAT 'dd/mm/yyyy' PROMPT ' Enter the second date: '

COLUMN emp\_id FORMAT A20 HEADING 'Employee ID'

COLUMN department\_id FORMAT A20 HEADING 'Department ID'

COLUMN emp\_name FORMAT A20 HEADING 'Employee Name'

COLUMN emp\_contact FORMAT A20 HEADING 'Employee Contact'

COLUMN emp\_salary FORMAT 9999999.99 HEADING 'Employee Salary'

COLUMN emp\_title FORMAT A20 HEADING 'Employee Title'

COLUMN CUSTOMER\_ID HEADING 'Customers Handeled'

COLUMN QUANTITY HEADING 'Employee\_handle\_Cust'

TTITLE LEFT'Amount of Customers Handled By Employees from ' &v\_firstdate ' TO ' &v\_seconddate' with status 'Pending''

SKIP 1-

CENTER COL 30 '\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_' SKIP 2 -

CENTER 'Page : ' FORMAT 999 SQL.PNO SKIP 2

CREATE OR REPLACE VIEW Employee\_handle\_Cust AS

SELECT EXTRACT(YEAR FROM R.reservationDate) AS year, E.emp\_id, D.department\_id, E.emp\_name, E.emp\_contact, E.emp\_salary, E.emp\_title,

COUNT(C.cust\_id) as CUSTOMER\_ID

FROM department D

JOIN employee E ON D.department\_id = E.department\_id

JOIN customer C ON E.emp\_id = C.emp\_id

JOIN reservation R ON C.cust\_id = R.cust\_id

WHERE R.reservationDate BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY EXTRACT(YEAR FROM R.reservationDate), E.emp\_id, D.department\_id, E.emp\_name, E.emp\_contact, E.emp\_salary, E.emp\_title

ORDER BY CUSTOMER\_ID DESC;

SELECT \*

FROM Employee\_handle\_Cust;

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:

Text

Description automatically generated

## 7.4.3 Query/Report 3: Reservation with status pending

Purpose: To determine the amount of reservation with the status “pending”.

SET LINESIZE 300

SET PAGESIZE 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'DD/MM/YYYY';

-- 01/01/2023 & 31/12/2023 --

PROMPT 'Reservation with Status Pending'

PROMPT '---------------------------------------------------'

PROMPT

ACCEPT v\_firstdate DATE FORMAT 'dd/mm/yyyy' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate DATE FORMAT 'dd/mm/yyyy' PROMPT ' Enter the second date: '

COLUMN reservationDate FORMAT A20 HEADING 'Reservation Date'

COLUMN reservation\_id FORMAT A20 HEADING 'Room ID'

COLUMN roomType FORMAT 20 HEADING 'Room Type'

COLUMN reservationStatus FORMAT A30 HEADING 'Reservation Status'

TTITLE LEFT 'Reservation with status pending Between ' &v\_firstdate & ' and ' &v\_seconddate ' '

SKIP 1 -

LEFT '\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_'

SKIP 2 -

LEFT 'Page : ' FORMAT 999 SQL.PNO SKIP 2

SELECT

E.reservationDate, E.reservation\_id, R.roomType, E.reservationStatus

FROM reservation E, room R

WHERE E.reservationStatus = 'Pending' AND E.reservationDate BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY E.reservationDate, E.reservation\_id, R.roomType, E.reservationStatus

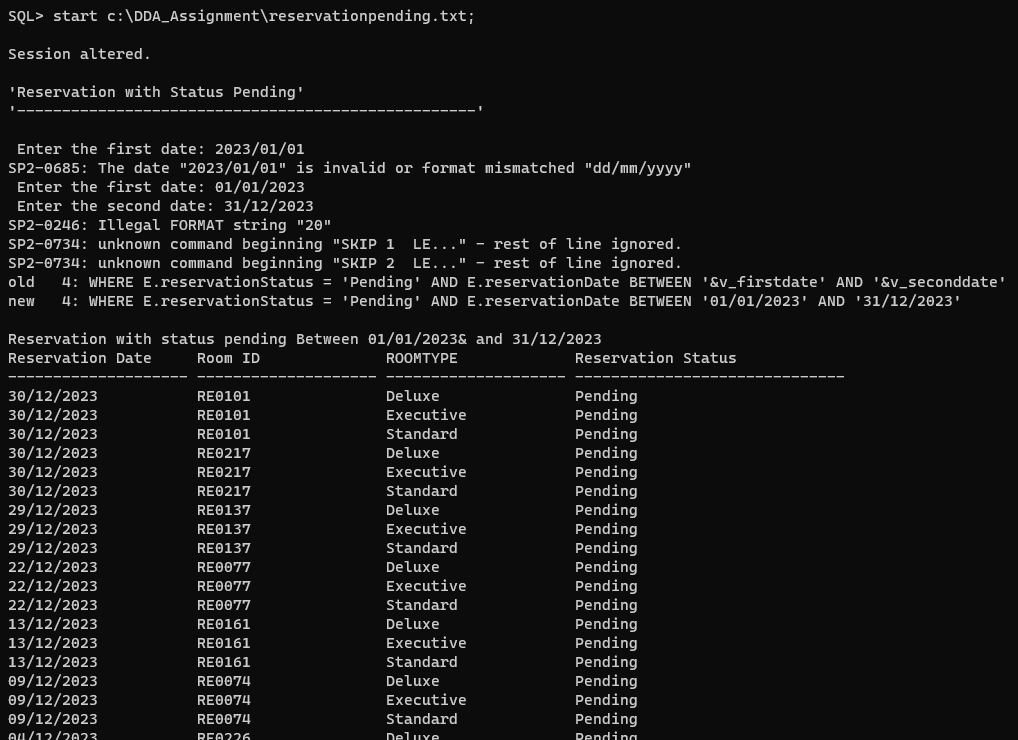
ORDER BY E.reservationDate DESC;

CLEAR BREAK

CLEAR COLUMNS

TTITLE OFF

Sample Output:



## 7.5 Nicholas Lim Sze Whye

## 7.5.1 Query/Report 1: Total Payment Between 2 Specified Dates

Purpose: To display total payment of customers between 2 specified date from user input as well as number of rows from user input.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'PAYMENT REPORT BETWEEN A SPECIFIC DATES'

PROMPT

PROMPT

ACCEPT v\_firstDate char FORMAT 'A11' PROMPT 'Enter first date: '

ACCEPT v\_secondDate char FORMAT 'A11' PROMPT 'Enter second date: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN payment\_id FORMAT A15 HEADING "Payment ID";

COLUMN payment\_date FORMAT A15 HEADING "Payment Date";

COLUMN payment\_amount FORMAT $99999.99 HEADING "Payment Amount";

TTITLE LEFT 'Total payment amount from ' &v\_firstdate ' TO ' &v\_seconddate' ' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT \* FROM (

SELECT P.payment\_id, P.payment\_date, SUM(P.payment\_amount) AS payment\_amount

FROM payment P

WHERE P.payment\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY P.payment\_id, P.payment\_date

ORDER BY P.payment\_date

)

WHERE ROWNUM <= '&v\_numberRows';

PROMPT 'Sum of Payments : '

SELECT SUM(payment\_amount) AS payment\_amount

FROM (

SELECT P.payment\_id, P.payment\_date, SUM(P.payment\_amount) AS payment\_amount

FROM payment P

WHERE P.payment\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY P.payment\_id, P.payment\_date

ORDER BY P.payment\_date

)

WHERE ROWNUM <= '&v\_numberRows';

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:

Text

Description automatically generated

## 7.5.2 Query/Report 2: Top returning Customers

Purpose: To determine the top return returning customers within a year.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'Total refund made by customer within specified dates'

PROMPT '----------------------------------------------------'

PROMPT

ACCEPT v\_firstdate CHAR FORMAT 'A11' PROMPT ' Enter the first date: '

ACCEPT v\_seconddate CHAR FORMAT 'A11' PROMPT ' Enter the second date: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN cust\_id FORMAT A15 HEADING "Customer ID";

COLUMN cust\_name FORMAT A15 HEADING "Customer Name";

COLUMN total\_refund FORMAT 9999 HEADING "Total Refund(s)";

COLUMN refund\_amount FORMAT $9999.99 HEADING "Refund Amount";

COLUMN avg\_refund\_amount FORMAT $9999.99 HEADING "Average Refund Amount";

TTITLE LEFT 'Total refund from ' &v\_firstdate ' TO ' &v\_seconddate' ' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT \* FROM(

SELECT N.cust\_id, N.cust\_name, COUNT(RU.refund\_id) AS total\_refund, SUM(RU.service\_tax) AS refund\_amount

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN cancellation C ON R.reservation\_id = C.reservation\_id

JOIN refund RU ON C.cancellation\_id = RU.cancellation\_id

WHERE RU.refund\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY N.cust\_id, N.cust\_name

ORDER BY refund\_amount DESC

) WHERE ROWNUM <= '&v\_numberRows';

PROMPT 'Average Refund Amount: '

SELECT AVG(refund\_amount) AS avg\_refund\_amount

FROM (

SELECT SUM(RU.service\_tax) AS refund\_amount

FROM customer N

JOIN reservation R ON N.cust\_id = R.cust\_id

JOIN cancellation C ON R.reservation\_id = C.reservation\_id

JOIN refund RU ON C.cancellation\_id = RU.cancellation\_id

WHERE RU.refund\_date BETWEEN '&v\_firstdate' AND '&v\_seconddate'

GROUP BY N.cust\_id, N.cust\_name

)WHERE ROWNUM <= '&v\_numberRows';

CLEAR COLUMNS

CLEAR BREAKS

TTITLE OFF

Sample Output:

Text

Description automatically generated

## 7.5.3 Query/Report 3: Room availability within specified dates.

Purpose: To identify the amount of available room within 2 specific dates.

SET linesize 200

SET pagesize 100

ALTER SESSION SET NLS\_DATE\_FORMAT = 'YYYY/MM/DD';

PROMPT 'Room Availability within specified dates'

PROMPT '----------------------------------------------------'

PROMPT

ACCEPT v\_firstDate char FORMAT 'A11' PROMPT 'Enter first date: '

ACCEPT v\_secondDate char FORMAT 'A11' PROMPT 'Enter second date: '

ACCEPT v\_numberRows NUMBER FORMAT 999 PROMPT 'Enter number of rows: '

COLUMN room\_id FORMAT A15 HEADING "Room ID";

COLUMN reservation\_id FORMAT A15 HEADING "Reservation ID";

COLUMN room\_type FORMAT A15 HEADING "Room Type";

COLUMN room\_status FORMAT A15 HEADING "Room Status";

TTITLE LEFT 'Room availability from ' &v\_firstdate ' TO ' &v\_seconddate' ' SKIP 1-

LEFT ' ' SKIP 1-

LEFT 'Page No: ' FORMAT 999 SQL.PNO SKIP 2

SELECT R.room\_id, E.reservation\_id, R.roomType, R.roomStatus

FROM room R

JOIN reservation E ON R.reservation\_id=E.reservation\_id

WHERE reservationDate BETWEEN '&v\_firstDate' AND '&v\_secondDate' AND R.roomStatus='Vacant' AND ROWNUM <= '&v\_numberRows';

Sample Output:

Text

Description automatically generated