TARUMT
TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

AACS3013 DATABASE DEVELOPMENT AND APPLICATIONS

Assignment

Semester 202301

Programme (Year & Group)	:	DCS1G3
Tutorial Group	:	3
Date Submitted	:	26/04/2022

Team members:

No	Name (Block Letters)	Registration No.	Signature	Marks
1	TAN YEN FANG	2203544	*yenfang*	
2	YAP ZHI QIAN	2203317	*zhiqian*	
3	ON SIEW LEE	2203144	*shirley*	
4	GOO YONG KANG	2203248	*yk*	
5	JEROME LU ZHENG YAO	2203589	*Jerome*	

TAN YEN FANG

YAP ZHI QIAN

ON SIEW LEE

GOO YONG KANG

JEROME LU ZHENG YAO

TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Plagiarism Statement and Guideline for Late Submission of Coursework

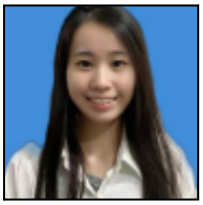
Read, complete, and sign this statement to be submitted with the written report.

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	TAN YEN FANG	2203544	*yenfang*	26/04/2022
2	YAP ZHI QIAN	2203317	*zhiqian*	26/04/2022
3	ON SIEW LEE	2203144	*shirley*	26/04/2022
4	GOO YONG KANG	2203248	*yk*	26/04/2022
5	JEROME LU ZHENG YAO	2203589	*Jerome*	26/04/2022



TAN YEN FANG

YAP ZHI QIAN

ON SIEW LEE

GOO YONG KANG

JEROME LU ZHENG YAO

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

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._____

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								

Table of Content

Task 1: Business Rules of the System

1.1 Entities of System

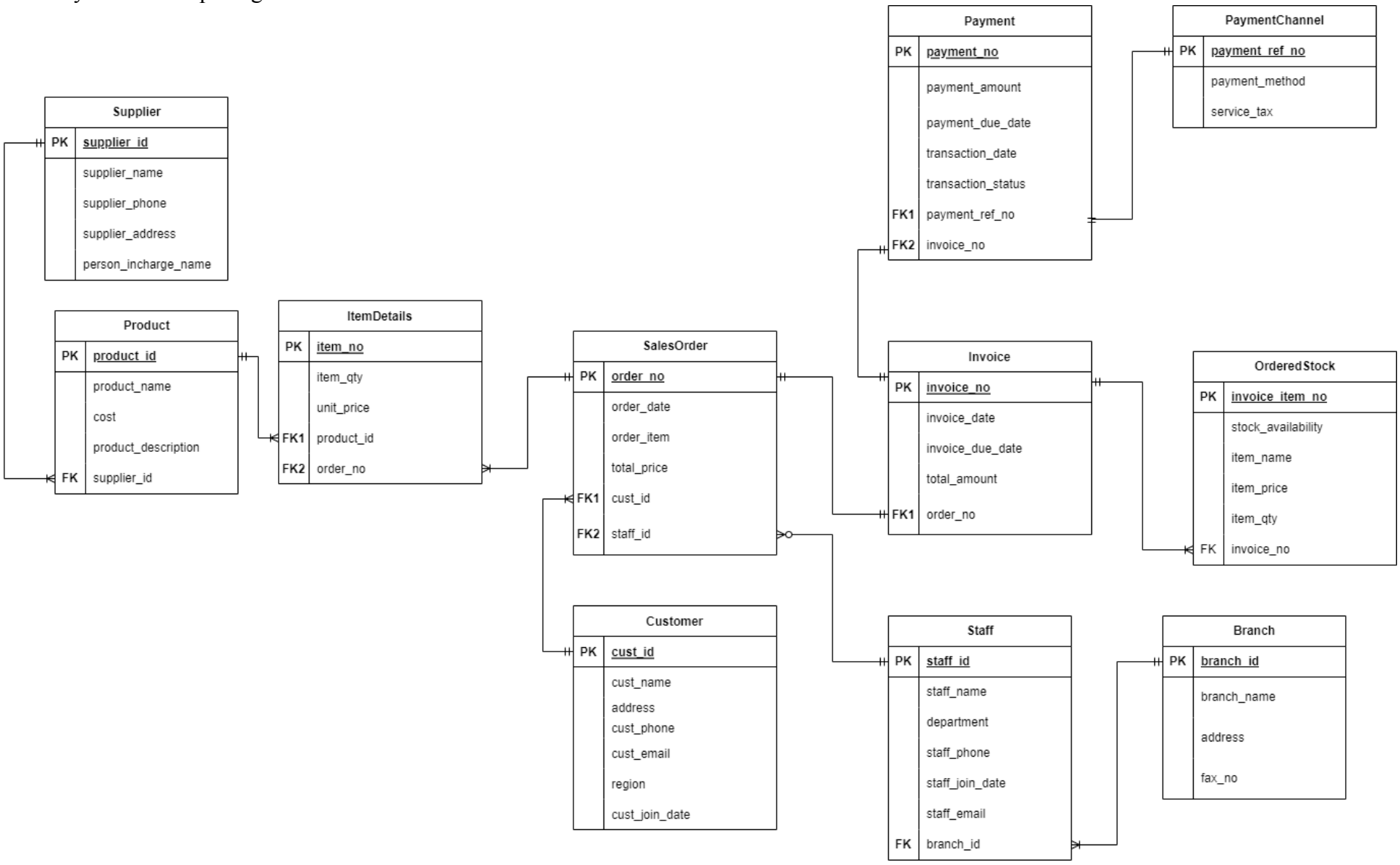
1. Customer
2. Invoice
3. SalesOrder
4. Staff
5. Product
6. Payment
7. PaymentChannel
8. ItemDetails
9. OrderedStock
10. Branch
11. Supplier

1.2 Business rules of entity A, B, C

1. One customer can have one or many sales orders. Every sales order belongs to one and only one customer.
2. One sales order can consist of one or many products. One product can be included in one or many sales orders.
3. Every product can be supplied by one and only one supplier. One supplier can supply one or many products.
4. Every sales order can be handled by one and only one staff. Every staff can handle zero or many sales orders.
5. Every staff belongs to one and only one branch. One branch can consist of one or many staffs.
6. Every sales order can only have one invoice. Every invoice belongs to one and only one sales order.
7. One invoice can consist of one or many ordered stocks. Every ordered stock can be included in one and only one invoice.
8. One invoice can consist of one and only one payment. Every payment can be paid for one and only one invoice.
9. Every payment can be made with one and only one payment channel. Every payment channel can be used to make one and only one payment.
10. Business operation hours start from 10.00 am to 6.00 pm everyday except Sunday and public holiday.
11. The system maintenance starts from 12.00 am to 1.00 am daily.

Task 2: Entity-Relationship Modelling

2.1 Entity-Relationship Diagram



2.2 Assumptions

Customer

1. Assume that one customer has only one customer ID.
2. Assume that one customer has only one phone number, address, email and region recorded.
3. Assume that the customer joined date is recorded upon the customer's registration.

Sales order

1. Assume that one sales order has only one order number.
2. Assume that one sales order can consist of one or many order items.
3. Assume that one sales order has only one order date and total price.
4. Assume that one sales order belongs to only one customer and can be handled by only one staff.

Item details

1. Assume that the item detail has only one item number.
2. Assume that the item detail has item quantity and only one unit price recorded.
3. Assume that the item details belong to only one product.
4. Assume that the item details will be included in the sales order.

Product

1. Assume that one product has only one product ID and name.
2. Assume that one product has only one cost and description recorded.
3. Assume that every product is supplied by only one supplier.

Supplier

1. Assume that one supplier has only one supplier ID and name.
2. Assume that one supplier has only one phone number and address recorded.
3. Assume that one supplier has only one person in charge.

Staff

1. Assume that one staff has only one staff ID.
2. Assume that one staff has only one phone number and email recorded.
3. Assume that the staff joined date is recorded upon the staff registration.
4. Assume that one staff only works in one department and branch.

Branch

1. Assume that one branch has only one branch ID, name and address.
2. Assume that one branch has only one fax number recorded.

Invoice

1. Assume that one invoice has only one invoice number.
2. Assume that one invoice has only one invoice date and invoice due date.
3. Assume that every invoice has a total amount and order number on it.

Order Stock

1. Assume that one ordered stock has only one invoice item number.
2. Assume that every ordered stock has its own stock availability.
3. Assume that one ordered stock has only one item price and one item quantity.
4. Assume that one ordered stock has only one invoice number.

Payment

1. Assume that one payment has only one payment number.
2. Assume that one payment has only one payment amount and payment due date.
3. Assume that one payment has only one transaction date and transaction status.
4. Assume that one payment has only one payment reference number and invoice number.

Payment Channel

1. Assume that the payment channel will generate a payment reference number upon every transaction.
2. Assume that the payment channel consists of many payment methods.
3. Assume that one payment channel may have service tax for every payment.

Task 3: Normalization

3.1 Attributes of entities with keys

Supplier (**supplier_id**, supplier_name, supplier_phone, supplier_address, person_incharge_name)
Product (**product_id**, product_name, cost, product_description, supplier_id*)
ItemDetails (**item_no**, item_qty, unit_price, product_id*, order_no*)
SalesOrder (**order_no**, order_date, order_item, total_price, cust_id*, staff_id*)
Customer (**cust_id**, cust_name, address, cust_phone, cust_email, region, cust_join_date)
Invoice (**invoice_no**, invoice_date, invoice_due_date, total_amount, order_no*)
OrderedStock (**invoice_item_no**, stock_availability, item_name, item_price, item_qty, invoice_no*)
Payment (**payment_no**, payment_amount, payment_due_date, transaction_date, transaction_status, payment_ref_no*, invoice_no*)
PaymentChannel (**payment_ref_no,** payment_method, service_tax)
Staff (**staff_id**, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id*)
Branch (**branch_id**, branch_name, address, fax_no)

Task 4: Create Databases in Oracle

4.1 Supplier table

```
CREATE TABLE Supplier(
       supplier_id varchar(5),
       supplier_name varchar(30),
       supplier_phone varchar(11),
       supplier_address varchar(50),
       person_incharge_name varchar(30),
       primary key(supplier_id)
       );
```

4.2 Customer table

```
CREATE TABLE Customer(
       cust_id varchar(5) not null,
       cust_name varchar(30) not null,
       address varchar(50),
       cust_phone varchar(11),
       cust_email varchar(30),
       region varchar(20),
       cust_join_date date,
       primary key(cust_id),
       constraint chk_cust_email check (REGEXP_LIKE(cust_email,'^[a-zA-Z]\w+@(\S+)$'))
       );
```

4.3 Branch table

```
CREATE TABLE Branch(
            branch_id varchar(5) not null,
            branch_name varchar(51) not null,
            address varchar(500),
            fax_no varchar(11),
            primary key(branch_id)
            );
```

4.4 Staff table

```
CREATE TABLE Staff(
            staff_id varchar(5) not null,
            staff_name varchar(30) not null,
            department varchar(25),
            staff_phone varchar(11),
            staff_join_date date,
            staff_email varchar(30),
            branch_id varchar(5),
            primary key(staff_id),
            foreign key(branch_id) references branch(branch_id),
            constraint chk_staff_email check (REGEXP_LIKE(staff_email,'^[a-zA-Z]\w+@(\S+)$'))
            );
```

4.5 SalesOrder table

```
CREATE TABLE SalesOrder(
            order_no varchar(5) not null,
            order_date date,
            order_item varchar(300) not null,
            total_price number(7,2) not null,
            cust_id varchar(5),
            staff_id varchar(5),
            primary key(order_no),
            foreign key(cust_id) references customer(cust_id),
            foreign key(staff_id) references staff(staff_id)
            );
```

4.6 Invoice table

```
CREATE TABLE Invoice(
            invoice_no varchar(5) not null,
            invoice_date date,
            invoice_due_date date,
            total_amount number(7,2),
            order_no varchar(5),
            primary key(invoice_no),
            foreign key(order_no) references SalesOrder(order_no)
            );
```

4.7 OrderedStock table

```
CREATE TABLE OrderedStock(
            invoice_item_no varchar(5) not null,
            stock_availability char(1),
            item_name varchar(300),
            item_price number(7,2),
            item_qty number(8),
            invoice_no varchar(5),
            primary key(invoice_item_no),
            foreign key(invoice_no) references Invoice(invoice_no),
            constraint chk_stock_availability check (UPPER(stock_availability) in ('T','F'))
            );
```

4.8 PaymentChannel table

```
CREATE TABLE Payment_Channel(
            payment_ref_no varchar(14) not null,
            payment_method varchar(15),
            service_tax number(4,2),
            primary key(payment_ref_no)
            );
```

4.9 Payment table

```
CREATE TABLE Payment(
            payment_no varchar(5) not null,
            payment_amount number(7,2) not null,
            transaction_date date,
            transaction_status varchar(7),
            payment_ref_no varchar(14),
            invoice_no varchar(5),
            primary key(payment_no),
            foreign key(payment_ref_no) references Payment_Channel(payment_ref_no),
            foreign key(invoice_no) references Invoice(invoice_no),
            constraint chk_transaction_status check (transaction_status in ('success','fail'))
            );
```

4.10 Product table

```
CREATE TABLE Product(
            product_id varchar(5) not null,
            product_name varchar(30),
            cost number(6,2),
            product_description varchar(40),
            supplier_id varchar(5),
            primary key(product_id),
            foreign key(supplier_id) references Supplier(supplier_id)
            );
```

4.11 ItemDetails table

```
CREATE TABLE ItemDetails(
            item_no varchar(5) not null,
            item_qty number(5),
            unit_price number(6,2),
            product_id varchar(5),
            order_no varchar(5),
            primary key(item_no),
            foreign key(product_id) references Product(product_id),
            foreign key(order_no) references SalesOrder(order_no)
            );
```

Task 5: Sample Data Records (10 sample records for each table)

5.1 Supplier table

```
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP001', 'DabZ',
'015-2576898', '36130 Grim Center', 'Allen Schmeler');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP002',
'Brightdog', '017-7076971', '184 Parkside Point', 'Wally Kiehn');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP003',
'Gigashots', '019-6078170', '74 Raven Place', 'Jesse Wolff');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP004', 'Yodo',
'014-5274937', '0581 Pierstorff Street', 'Agatha Feest');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP005', 'Avamm',
'015-3686191', '56462 Gateway Way', 'Brian Bergnaum');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP006',
'Topicblab', '018-1563262', '15124 VonRueden Crossroad', 'Jong Bergstrom');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP007', 'Vinte',
'014-6285589', '2124 Kuvalis Ports', 'Suzi Hessel');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP008', 'Trilia',
'013-1354508', '050 Raquel Cape', 'Danilo Huels');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP009', 'Livetube',
'015-0571770', '894 Schiller Motorway', 'Raeann Pagac');
insert into Supplier (supplier_id, supplier_name, supplier_phone, supplier_address, person_incharge_name) values ('SP010',
'Feednation', '019-4383902', '54952 Lindgren Points', 'Levi Hammes');
```

5.2 Customer table

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';

insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C7183', 'Laure Bygreaves',
'534 Melby Lane', '0100673254', null, 'Finland', '02-05-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C1830', 'Sarge Abley', '84
Sycamore Crossing', '0179285999', null, 'AG', '02-09-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C6866', 'Vasili Collip',
'40576 Carberry Pass', '0193831590', null, 'Thailand', '17-11-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C4727', 'Glenine
Bruffell', '5 Delaware Park', '0191652826', null, 'Indonesia', '03-06-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C4495', 'Rhys Wimsett',
'8260 Shelley Court', '0108542686', null, 'Japan', '12-02-2023');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C4018', 'Francis
Boardman', '40 Rowland Pass', '0134304682', null, 'Brazil', '25-03-2023');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C2131', 'Carny Trinbey',
'681 Goodland Center', '0147533757', null, 'China', '15-01-2023');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C7509', 'Lucinda
Berryann', '20818 Bellgrove Alley', '0135809349', null, 'Malaysia', '15-10-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C1867', 'Arley Bowne', '01
Ilene Circle', '0197928187', null, 'Greece', '27-05-2022');
insert into Customer (cust_id, cust_name, address, cust_phone, cust_email, region, cust_join_date) values ('C9385', 'Nora Gaines',
'1917 Mosinee Pass', '0181662830', null, 'China', '09-08-2022');
```

5.3 Branch table

```
insert into Branch (branch_id, branch_name, address, fax_no) values ('B2788', 'Nestle (Malaysia) Berhad', 'Level 22, 1 Powerhouse
No.1, Persiaran Bandar Utama, Bandar Utama
47800 Petaling Jaya, Selangor', '03-79656767');

insert into Branch (branch_id, branch_name, address, fax_no) values ('B2571', 'Nestle Products Sdn Bhd', 'Lot 7316, Persiaran
Sijangkang Utama
Taman Industri Sijangkan Utama
42500 Teluk Panglima Garang
Selangor Darul Ehsan
', '03-31233001');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B4058', 'Nestle Manufacturing Shah Alam (Malaysia) Sdn. Bhd',
'PT 927, Jalan Playar 15/1
Seksyen 15, 40200 Shah Alam
Selangor Darul Ehsan', '03-55225999');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B5880', 'Nestle Batu Tiga Factory', 'PT 927, Jalan Playar 15/1
Seksyen 15, 40200 Shah Alam
Selangor Darul Ehsan', '03-55113149');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B6168', 'Nestle Sri Muda Factory', 'No. 2006, Kampung Maasop
Senaling, Negeri Sembilan, 72000 Kuala Pilah', '03-55206500');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B3376', 'Nestle Chembong Factory', 'Jalan Perusahaan Utama
Kawasan Perindustrian Chembong
71300 Rembau
Negeri Sembilan
', '06-6864080');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B3714', 'Nestle Chembong (Ice Cream) Factory', '4038 Monica
Plaza', '495-7481');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B6150', 'Nestle Kuching Factory', 'Lot 844, Block 7
Muara Tebas Land District
Demak Laut Industrial Park
P.O. Box 710
93714 Kuching
Sarawak
', '082-472999');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B2491', 'Nestle Nihon Canpack (Malaysia) Sdn Bhd', 'Plot 47,
Lorong Bemban
Bemban Industrial Park
P.O. Box 37
31000 Batu Gajah
Perak', '05-3651211');
insert into Branch (branch_id, branch_name, address, fax_no) values ('B5801', 'Nestle Products Sdn Bhd', '5460, Persiaran Bunga
Tanjung 1, Kawasan Perusahaan Senawang Baru, 70450 Seremban, Negeri Sembilan', '06-6517264');
```

5.4 Staff table

```
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0001','Ritch','Business Development','6011211293','01-JAN-1998' ,'W0001@sale.com','B4058');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0002','Archibold','Legal','6011535568','01-JAN-1998' ,'W0002@sale.com','B2788');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0003','Fardell','Support','6011361792','01-JAN-1998' ,'W0003@sale.com','B2571');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0004','Ancell','RandD','6011696726','01-JAN-1998' ,'W0004@sale.com','B5880');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0005','Holsall','Training','6011290835','01-JAN-1998' ,'W0005@sale.com','B6168');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0006','Klaff','Support','6011721414','01-JAN-1998' ,'W0006@sale.com','B3376');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0007','Camilio','Sales','6011015532','01-JAN-1998' ,'W0007@sale.com','B3714');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0008','Glavias','RandD','6011395728','01-JAN-1998' ,'W0008@sale.com','B6150');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0009','Carlozzi','Business Development','6011807304','01-JAN-1998' ,'W0009@sale.com','B2491');
insert into Staff (staff_id, staff_name, department, staff_phone, staff_join_date, staff_email, branch_id) values
('W0010','Millis','Marketing','6011946215','01-JAN-1998' ,'W0010@sale.com','B5801');
```

5.5 SalesOrder table

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S5375','25-03-2023','Nestea Iced
Tea',9.96,'C7183','W0002');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S5356','18-06-2022','Gerber', 5.98,
'C1830', 'W0002');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S4088','06-02-2023','Carnation
Breakfast',477.63,'C6866','W0003');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S3627','17-05-2022','Milkybar
Buttons,Nestle Turtles',501.58,'C4727','W0004');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S5763','18-12-2022','Island Oasis -
Raspberry,Nesquik',179.70,'C4495','W0005');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values
('S5976','10-05-2022','Crunch',11.96,'C4018','W0005');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S7678','19-11-2022','Carnation
Milk,Pumpkin of Libby,Coffee-Mate',1693.44,'C2131','W0005');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S6723','05-02-2023','Cafe
HAG',791.28,'C7509','W0005');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values ('S2673','29-09-2022','Nestle Coffee
Crisp',73.01,'C1867','W0005');
insert into SalesOrder (order_no, order_date, order_item, total_price, cust_id, staff_id) values
('S3677','18-03-2023','Crunch',645.84,'C9385','W0005');
```

5.6 Invoice table

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I6134', '27-03-2023', '29-03-2023',
9.96, 'S5375');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I4458', '20-06-2022', '22-06-2022',
5.98, 'S5356');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I3009', '08-02-2023', '10-02-2023',
477.63, 'S4088');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I1908', '19-05-2022', '21-05-2022',
501.58, 'S3627');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I6289', '20-12-2022', '22-12-2022',
179.70, 'S5763');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I4333', '12-05-2022', '14-05-2022',
11.96, 'S5976');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I6371', '21-11-2022', '23-11-2022',
1693.44, 'S7678');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I9170', '07-02-2023', '09-02-2023',
791.28, 'S6723');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I6033', '01-10-2022', '03-10-2022',
73.01, 'S2673');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I5685', '20-03-2023', '22-10-2023',
645.84, 'S3677');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I5484', '22-10-2022', '24-10-2022',
2673.84, 'S6109');
insert into Invoice (invoice_no, invoice_date, invoice_due_date, total_amount, order_no) values ('I3622', '31-10-2022', '02-11-2022',
1923.25, 'S4690');
```

5.7 OrderedStock table

```
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N8411', 'T',
'Nestea Iced Tea', 9.96, 4, 'I6134');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N6494', 'T',
'Gerber', 5.98, 2, 'I4458');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N4032', 'T',
'Carnation Breakfast', 477.63, 87, 'I3009');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N7970', 'T',
'Milkybar Buttons', 441.78, 222, 'I1908');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N7971', 'T',
'Nestle Turtles', 59.80, 20, 'I1908');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N5725', 'T',
'Nesquik', 5.99, 30, 'I6289');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N0782', 'T',
'Crunch', 11.96, 4, 'I4333');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N7942', 'T',
'Carnation Milk', 47.31, 19, 'I6371');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N7943', 'T',
'Pumpkin of Libby', 1137.15, 285, 'I6371');
insert into OrderedStock (invoice_item_no, stock_availability, item_name, item_price, item_qty, invoice_no) values ('N7944', 'T',
'Coffee-Mate', 508.98, 102, 'I6371');
```

5.8 PaymentChannel table

```
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('89808277222158', 'touch n go', '33.47');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('07902534842117', 'cash', '88.27');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('75615661750476', 'online banking', '60.64');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('99032228488540', 'online banking', '1.80');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('37767250216117', 'touch n go', '21.33');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('42272855156722', 'touch n go', '23.69');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('55773596543897', 'touch n go', '19.84');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('64778775427243', 'cash', '50.89');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('77707774352569', 'touch n go', '39.48');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('07325581386795', 'online banking', '14.49');
insert into Payment_Channel (payment_ref_no, payment_method, service_tax) values ('64298384834894', 'cash', '17.88');
```

5.9 Payment table

```
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY934',
'9.96', '28-Mar-2023','success','89808277222158','I6134');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY168',
'5.98', '21-Jun-2022', 'success','07902534842117','I4458');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY534',
'477.63', '09-Feb-2023','success','75615661750476','I3009');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY393',
'501.58', '20-May-2022', 'success','990322284885 40','I1908');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY120',
'179.70', '30-Dec-2022', 'fail','37767250216117','I6289');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY186',
'11.96', '23-May-2022', 'fail','42272855156722','I4333');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY786', '
1693.44', '23-Nov-2022', 'success','55773596543897','I6371');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY369',
'791.28', '14-Feb-2023', 'fail','64778775427243','I9170');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY949',
'73.01', '29-Oct-2022', 'fail','77707774352569','I6033');
insert into payment (payment_no, payment_amount, transaction_date, transaction_status, payment_ref_no, invoice_no) values ('PY650',
'645.84', '21-Mar-2023', 'success','07325581386795','I5685');
```

5.10 Product table

```
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD001', 'Nescafe Classic', 12.99,
'Instant coffee in a jar', 'SP001');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD002', 'KitKat', 1.99, 'Chocolate
covered wafer bar', 'SP002');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD003', 'Maggi Noodles', 2.49,
'Instant noodles in a packet', 'SP003');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD004', 'Milo', 7.99, 'Chocolate and
malt powder', 'SP005');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD005', 'Nestea Iced Tea', 1.49,
'Instant iced tea in a sachet', 'SP005');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD006', 'Nestle Fitness', 4.29,
'Cereal with whole grain', 'SP006');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD007', 'Nesquik', 3.99, 'Chocolate
powder for milk', 'SP004');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD008', 'Smarties', 1.59, 'Chocolate
candies in a tube', 'SP009');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD009', 'Bonjour', 4.99, 'Coffee
creamer in a bottle', 'SP009');
insert into Product (product_id, product_name, cost, product_description, supplier_id) values ('PD010', 'Cafe HAG', 9.49,
'Decaffeinated instant coffee in jar', 'SP010');
```

5.11 ItemDetails table

```
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID001', 22, 14.99, 'PD001', 'S7044');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID002', 8, 14.99, 'PD001', 'S0555');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID003', 46, 14.99, 'PD001', 'S9171');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID004', 302, 2.49, 'PD002', 'S7136');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID005', 171, 2.49, 'PD002', 'S9906');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID006', 9, 2.49, 'PD002', 'S4832');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID007', 38, 3.99, 'PD003', 'S7983');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID008', 363, 3.99, 'PD003', 'S2643');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID009', 4, 3.99, 'PD003', 'S5318');
insert into ItemDetails (item_no, item_qty, unit_price, product_id, order_no) values ('ID010', 15, 8.99, 'PD004', 'S3997');
```

Task 6: SQL Queries and Reports

6.1 Tan Yen Fang

6.1.1 Query/Report 1: Top N Branch which made the most Sales Order in X year

Purpose: The purpose of this query is to find the top branch that makes the most sales order in a specific year.

SQL statement:

```
SET LINESIZE 300
SET PAGESIZE 100
SET RECSEP EACH

BREAK ON YEAR ON staff_id NODUPLICATES
PROMPT 'Report: Top N Branch which made the most Sales Order in X year'
PROMPT 'Enter year and a figure to determine the top range'
ACCEPT v_year CHAR FORMAT A4 PROMPT 'Enter year: '
ACCEPT v_rowNum NUMBER FORMAT '9999' PROMPT 'Enter number of rows: '

COLUMN YEAR HEADING 'Year';
COLUMN branch_id FORMAT A20 HEADING 'Branch ID';
COLUMN branch_name FORMAT A60 HEADING 'Branch Name';
COLUMN staff_id FORMAT A20 HEADING 'Staff ID';
COLUMN staff_name FORMAT A20 HEADING 'Staff Name';
COLUMN Total_Orders HEADING 'Total Orders';
COLUMN Total_Revenue FORMAT '999,999,999.99' HEADING 'Total Revenue (RM)';

TTITLE COL 40 'Top '&v_rowNum' Branch Made the Most sales in year '&v_year'' SKIP 1 -
COL 40 '-----------------------------------------' SKIP 2
REPFOOTER COL 40 '--END OF REPORT--'
COMPUTE SUM LABEL 'Total(RM):' OF "Total_Revenue" on year
SELECT *
FROM (SELECT EXTRACT(YEAR FROM SO.order_date) AS YEAR, B.branch_id,
B.branch_name, COUNT (SO.order_no) AS Total_Orders, SUM(SO.total_price) AS
Total_Revenue
FROM Branch B
JOIN STAFF S
ON B.branch_id=S.branch_id
JOIN SalesOrder SO
```

```
ON S.staff_id=SO.staff_id
AND EXTRACT(YEAR FROM SO.order_date) =&v_year
GROUP BY EXTRACT(YEAR FROM SO.order_date), B.branch_id, B.branch_name
ORDER BY SUM (SO.Total_Price) DESC)
WHERE ROWNUM<=&v_rowNum;
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
REPFOOTER OFF
```

```
'Report: Top N Branch which made the most Sales Order in X year'
'Enter year and a figure to determine the top range'
Enter year: 2022
Enter number of rows: 10
old  10: AND EXTRACT(YEAR FROM SO.order_date) =&v_year
new  10: AND EXTRACT(YEAR FROM SO.order_date) =2022
old  13: WHERE ROWNUM<=&v_rowNum
new  13: WHERE ROWNUM<=        10

                                Top 10 Branch Made the Most sales in year 2022
                                ----------------------------------------------

      Year Branch ID           Branch Name                                                    Total Orders Total Revenue (RM)
---------- -------------------- -------------------------------------------------------------- ------------ ------------------
      2022 B5801                Nestles Products Sdn Bhd                                                7          33,530.78

           B5880                Nestle Batu Tiga Factory                                                6          28,159.44

           B2571                Nestle Products Sdn Bhd                                                 8          28,075.83

           B3714                Nestle Chembong (Ice Cream) Factory                                     8          27,005.09

           B6168                Nestle Sri Muda Factory                                                13          26,390.15

           B4058                Nestle Manufacturing Shah Alam (Malaysia) Sdn. Bhd                     16          20,100.63

           B3376                Nestle Chembong Factory                                                 6          17,675.51

           B2491                Nestle Nihon Canpack (Malaysia) Sdn Bhd                                 4          12,946.71

           B2788                Nestle (Malaysia) Berhad                                                5          10,166.16

           B6150                Nestle Kuching Factory                                                  5           5,040.37

**********                                                                                                    ------------------
Total(RM):                                                                                                        209,090.67
                                --END OF REPORT--

10 rows selected.
```

6.1.2 Query/Report 2: Total Number of Orders for each product and Total Revenue in X year
Purpose: The purpose of this report is to find the top product that has the highest number of orders and calculate the total revenue in specific year.

SQL statement:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';
SET linesize 150
SET pagesize 150
SET RECSEP EACH
BREAK ON YEAR ON staff_id NODUPLICATES

PROMPT 'Total Number of Orders for each product and Total Revenue in X year'
ACCEPT v_year CHAR FORMAT A4 PROMPT 'Enter year:'

COLUMN YEAR HEADING 'Year';
COLUMN product_id FORMAT A10 HEADING 'Product ID';
COLUMN product_name FORMAT A30 HEADING 'Product Name';
COLUMN Total_Orders HEADING 'Total Orders';
COLUMN Total_Revenue FORMAT '9999999.99' HEADING 'Total Revenue (RM)';

TTITLE COL15 'Total Number of Orders for each product and Total Revenue in '&v_year'' SKIP 1 -
COL 15 '-------------------------------------------------------------' SKIP 2

COMPUTE SUM LABEL 'Total(RM):' OF Total_Revenue on Year
CREATE OR REPLACE VIEW Total_Order_Each_Product AS
SELECT EXTRACT(YEAR FROM SO.order_date) AS Year, I.product_id,P.product_name,
COUNT(SO.order_no) AS Total_Orders,I.unit_price,SUM(SO.total_price) AS Total_Revenue
FROM SalesOrder SO
JOIN ItemDetails I
ON SO.order_no=I.order_no
JOIN Product P
ON I.product_id=P.product_id
AND EXTRACT(YEAR FROM SO.order_date)=&v_year
GROUP BY EXTRACT(YEAR FROM SO.order_date),I.product_id,P.product_name,I.unit_price
ORDER BY I.product_id,Total_Orders;
SELECT*
FROM Total_Order_Each_Product;
```

```
Session altered.

'Total Number of Orders for each product and Total Revenue in X year'
'Enter year to determine the total of orders for each product in that year'
Enter year:2023
old   9: AND EXTRACT(YEAR FROM SO.order_date)=&v_year
new   9: AND EXTRACT(YEAR FROM SO.order_date)=2023

View created.


            Total Number of Orders for each product and Total Revenue in 2023
            ----------------------------------------------------------------

     Year Product ID Product Name                Total Orders UNIT_PRICE Total Revenue (RM)
--------- ---------- ---------------------------- ------------ ---------- -----------------
     2023 PD002      KitKat                                  1       2.49            261.45
          PD003      Maggi Noodles                           1       3.99            151.62
          PD004      Milo                                    2       8.99          10277.04
          PD005      Nestea Iced Tea                         2       2.49            271.41
          PD009      Bonjour                                 1       5.49           3098.14
          PD010      Cafe HAG                                2      10.99           8007.81
          PD012      Nestle Cheerios                         1       6.49           7216.53
          PD013      Nestle Crunch                           1       1.99           5138.52
          PD014      Nestle Dibs                             1       4.99           3098.14
          PD016      Nestle Lion                             1       2.99           3098.14
          PD017      Nestle After Eight                      3       5.99          14341.62
          PD018      Buitoni Pasta                           1       3.99           3098.14
          PD019      Perrier                                 1       2.99           5138.52
          PD021      Vittel                                  2       3.99           5425.80
          PD023      Pumpkin of Libby                        1       3.99           1986.57
          PD025      Toll House                              1       3.99            199.50
          PD026      Gerber                                  1       2.99           3098.14
          PD029      Coffee-Mate                             1       4.99           5259.46
          PD033      Lean Cuisine                            1       4.99           1986.57
          PD035      Nestle Resource                         1       4.99           5259.46
          PD036      Stouffer                                1       6.99            817.83
          PD039      Carnation Breakfast                     1       5.49            477.63
          PD040      Herta                                   1       5.49           1986.57
          PD044      Crunch                                  1       2.99            645.84
          PD047      Milkybar                                2       1.99           1997.56
          PD051      Aero Bubbles                            1       2.99           6926.55
          PD056      Nestle Peppermint                       1       3.99             19.95
          PD066      Toll House                              2       4.99           8767.86
          PD074      Baby Ruth Crispety Crunchety            1       4.99           6926.55
          PD079      Chocapic Cereal                         1       5.99             83.86
          PD083      Nestle Cerelac                          1       9.99           6926.55
          PD091      Maggi Vegetable Noodles                 1       3.99           3018.38
          PD092      Maggi Masala Noodles                    1       3.99           6926.55
          PD093      Maggi Hot and Sweet Sauce               1       4.99           6926.55
          PD094      Maggi Tomato Ketchup                    1       3.99           6926.55
          PD095      Maggi Bhuna Masala                      1       3.99           3018.38
          PD096      Maggi Magic Cubes                       2       3.99           2501.85
          PD099      Maggi Rich Tomato Sauce                 2       4.99           5504.27
**********                                                                    -----------------
Total(RM):                                                                        156811.86

              --END OF REPORT--
```

6.1.3 Query/Report 3: Top N Staff Made the Most sales in X year

Purpose: The purpose of this report is to find the top staff that makes the most number of orders in a specific year and calculate the total revenue done by the staff.

SQL statement:

```
SET LINESIZE 150
SET PAGESIZE 60
SET RECSEP EACH

BREAK ON YEAR ON staff_id NODUPLICATES
PROMPT 'Top N Staff Made the Most sales in X year'
PROMPT 'Enter year and a figure to determine the top range'
ACCEPT v_year CHAR FORMAT A4 PROMPT 'Enter year: '
ACCEPT v_rowNum NUMBER FORMAT '9999' PROMPT 'Enter number of rows: '

COLUMN YEAR HEADING 'Year';
COLUMN branch_id FORMAT A9 HEADING 'Branch ID';
COLUMN staff_id FORMAT A8 HEADING 'Staff ID';
COLUMN staff_name FORMAT A10 HEADING 'Staff Name';
COLUMN Total_Orders HEADING 'Total Orders';
COLUMN Total_Revenue FORMAT '9999999.99' HEADING 'Total Revenue (RM)';

TTITLE COL 20 'Top '&v_rowNum' Staff Made the Most sales in year '&v_year'' SKIP 1 -
COL 20 '-----------------------------------------' SKIP 2
REPFOOTER COL 20 '--END OF REPORT--'
SELECT *
FROM (SELECT EXTRACT(YEAR FROM SO.order_date) AS YEAR,S.staff_id,S.staff_name, B.branch_id, COUNT
(SO.order_no) AS Total_Orders, SUM(SO.total_price) AS
Total_Revenue
FROM Branch B
JOIN STAFF S
ON B.branch_id=S.branch_id
JOIN SalesOrder SO
ON S.staff_id=SO.staff_id
AND EXTRACT(YEAR FROM SO.order_date) =&v_year
```

```
GROUP BY EXTRACT(YEAR FROM SO.order_date),S.staff_id,S.staff_name, B.branch_id
ORDER BY SUM (SO.Total_Price) DESC)
WHERE ROWNUM<=&v_rowNum;
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
REPFOOTER OFF
```

```
'Top N Staff Made the Most sales in X year'
'Enter year and a figure to determine the top range'
Enter year: 2022
Enter number of rows: 10
old   9: AND EXTRACT(YEAR FROM SO.order_date) =&v_year
new   9: AND EXTRACT(YEAR FROM SO.order_date) =2022
old  12: WHERE ROWNUM<=&v_rowNum
new  12: WHERE ROWNUM<=        10

                  Top 10 Staff Made the Most sales in year 2022
                  ---------------------------------------------

      Year Staff ID Staff Name Branch ID Total Orders Total Revenue (RM)
---------- -------- ---------- --------- ------------ ------------------
      2022 W0100    Tillman    B5801                3           20038.44

           W0093    Bernardoni B2571                3           19041.13

           W0084    Gilks      B5880                1           12354.27

           W0079    Isaacson   B2491                1           11049.08

           W0064    Wiggans    B5880                1           10377.70

           W0051    Manuaud    B4058                3            9665.31

           W0075    Ruseworth  B6168                1            9613.53

           W0076    Pengilly   B3376                1            6985.91

           W0057    Sapseed    B3714                1            6793.42

           W0097    Bellard    B3714                1            6322.57

********** ********                                      ------------------
Total(RM):                                                      112241.36
                  --END OF REPORT--

10 rows selected.
```

6.2 Yap Zhi Qian

6.2.1 Query/Report 1: Total number of sales order made by each staff

Purpose: The purpose of this query is to find the total number of sales orders made by each staff member in a specific year that will be input by the user.

SQL statement:

```
SET linesize 120
SET pagesize 100
SET RECSEP EACH
BREAK ON YEAR NODUPLICATES ON staff_id NODUPLICATES

PROMPT 'Total number of sales order made by each staff In A Specific Year'
ACCEPT v_year NUMBER PROMPT 'Enter Year: '

COLUMN year FORMAT A15 HEADING "Month and Year"
COLUMN staff_id FORMAT A10 HEADING "Staff ID";
COLUMN staff_name FORMAT A30 HEADING "Staff Name";
COLUMN total_orders FORMAT 999999 HEADING "Number Of Sales Order";

TTITLE 'Total Number of Sales Order made by Each Staff in Year '&v_year'' SKIP 1 -
'------------------------------------------------------' SKIP 2

BREAK ON "Staff ID" SKIP 1 ON "Staff Name" SKIP 1

SELECT EXTRACT(MONTH FROM SO.order_date) || '-' || EXTRACT(YEAR FROM SO.order_date) AS year, SO.staff_id,
S.staff_name, COUNT(SO.order_no) AS total_orders
FROM SalesOrder SO
JOIN Staff S ON S.staff_id = SO.staff_id
WHERE S.staff_id = SO.staff_id
AND EXTRACT(YEAR FROM SO.order_date) = &v_year
GROUP BY EXTRACT(MONTH FROM SO.order_date), EXTRACT(YEAR FROM SO.order_date), SO.staff_id, S.staff_name
ORDER BY total_orders DESC;

CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF;
```

```
SQL> start c:\\query.txt
'Total number of sales order made by each staff In A Specific Year'
Enter Year: 2023
old   5: AND EXTRACT(YEAR FROM SO.order_date) = &v_year
new   5: AND EXTRACT(YEAR FROM SO.order_date) =       2023

Total Number of Sales Order made by Each Staff in Year 2023
-----------------------------------------------------

Month and Year  Staff ID   Staff Name                     Number Of Sales Order
--------------- ---------- ------------------------------ ---------------------
2-2023          W0082      Arnli                                              2

3-2023          W0071      Blackledge                                         2

1-2023          W0038      Drescher                                           1

3-2023          W0046      Birtonshaw                                         1

3-2023          W0051      Manuaud                                            1

3-2023          W0069      Grimmett                                           1

3-2023          W0037      Mussettini                                         1

2-2023          W0059      Elstob                                             1

2-2023          W0003      Fardell                                            1

3-2023          W0005      Holsall                                            1

2-2023          W0005      Holsall                                            1

3-2023          W0044      D Hooghe                                           1

2-2023          W0051      Manuaud                                            1

2-2023          W0095      Arling                                             1

2-2023          W0016      Greensite                                          1

2-2023          W0039      Keyho                                              1

1-2023          W0047      Tiptaft                                            1

1-2023          W0037      Mussettini                                         1

3-2023          W0002      Archibold                                          1

1-2023          W0051      Manuaud                                            1

20 rows selected.
```

6.2.2 Query/Report 2: Total number and amount of sales order made by each customer
Purpose: The purpose of this query is to find the total number and amount of sales orders made by each customer in a specific year that will be input by the user.
SQL statement:

```
SET linesize 120
SET pagesize 100
SET RECSEP EACH
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY';

PROMPT 'Total Amount of Sales Order made by Each Customer In A Specific Year'
ACCEPT v_year NUMBER PROMPT 'Enter Year: '

COLUMN year FORMAT A15 HEADING "Month and Year"
COLUMN cust_id FORMAT A20 HEADING "Customer ID"
COLUMN cust_name FORMAT A20 HEADING "Customer Name"
COLUMN total_order FORMAT 9999 HEADING "Number of Order"
COLUMN total_price FORMAT $99999,999.99 HEADING "Total Price"

TTITLE 'Total amount of sales order made by each customer In '&v_year'' SKIP 1 -
'-----------------------------------------------------' SKIP 2
BREAK ON "Customer ID" SKIP 1 ON "Customer Name" SKIP 1

SELECT EXTRACT(MONTH FROM SO.order_date) || '-' || EXTRACT(YEAR FROM SO.order_date) AS year, C.cust_id,
C.cust_name, COUNT(SO.order_no) AS total_order, SUM(SO.total_price) AS total_price
FROM Customer C
JOIN SalesOrder SO ON C.cust_id = SO.cust_id
WHERE C.cust_id = SO.cust_id
AND EXTRACT(YEAR FROM SO.order_date) = &v_year
GROUP BY EXTRACT(MONTH FROM SO.order_date), EXTRACT(YEAR FROM SO.order_date), C.cust_id, C.cust_name
ORDER BY EXTRACT(MONTH FROM SO.order_date);

CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF;
```

```
Session altered.

'Total Amount of Sales Order made by Each Customer In A Specific Year'
Enter Year: 2023
old   5: AND EXTRACT(YEAR FROM SO.order_date) = &v_year
new   5: AND EXTRACT(YEAR FROM SO.order_date) =       2023

Total amount of sales order made by each customer In 2023
---------------------------------------------------------

Month and Year  Customer ID          Customer Name        Number of Order    Total Price
--------------- -------------------- -------------------- --------------- --------------
1-2023          C1783                Joline Walcar                      1         $83.86

1-2023          C3387                Tarah Jerdein                      1        $287.28

1-2023          C5085                Beverie Fleckness                  1        $199.50

1-2023          C8754                Gerardo Hobbema                    1      $5,259.46

2-2023          C0152                Mina Biggam                        1         $19.95

2-2023          C0934                Arin Mathewes                      1      $5,138.52

2-2023          C0972                Ginevra Skillicorn                 1      $7,216.53

2-2023          C1374                Osborne Sowley                     1      $6,926.55

2-2023          C1710                Harrietta Brabin                   1      $3,098.14

2-2023          C4972                Yurik Duffer                       1        $151.62

2-2023          C6035                Suzann Eede                        1      $2,485.89

2-2023          C6866                Vasili Collip                      1        $477.63

2-2023          C7509                Lucinda Berryann                   1        $791.28

3-2023          C0236                Arch Disdel                        1      $3,018.38

3-2023          C4561                Vanny Collihole                    1      $1,986.57

3-2023          C4904                Ulrikaumeko Dossit                 1      $1,841.31

3-2023          C4945                Klarrisa Pountney                  1         $15.96

3-2023          C6626                Bobbye Ferres                      1         $10.99

3-2023          C6735                Sophia Pickle                      1        $261.45

3-2023          C7183                Laure Bygreaves                    1          $9.96

3-2023          C9165                Charles Idwal Evans                1        $817.83

3-2023          C9385                Nora Gaines                        1        $645.84

22 rows selected.
```

6.2.3 Query/Report 3: Top N Customers Spent the Most in Orders

Purpose: The purpose of this query is to find the top customers spent the most in orders in a specific year and top rows that will be input by the user.

SQL statement:

```
SET linesize 120
SET pagesize 100
SET RECSEP EACH
BREAK ON YEAR ON cust_id NODUPLICATES
PROMPT 'Top N Customers Spent the Most in Orders In A Specific Year'
ACCEPT v_year NUMBER PROMPT 'Enter Year: '
ACCEPT v_rowNum NUMBER PROMPT 'Enter Row Number: '
COLUMN year FORMAT A18 HEADING 'Month - Year'
COLUMN cust_id FORMAT A20 HEADING "Customer ID"
COLUMN cust_name FORMAT A20 HEADING "Customer Name"
COLUMN total_order FORMAT 9999 HEADING "Total Order"
COLUMN total_price FORMAT $99999,999.99 HEADING "Total Price"
TTITLE 'Top '&v_rowNum' Customer Spent the Most in Orders in '&v_year'' SKIP 1 -
'-----------------------------------------------------' SKIP 2
BREAK ON "Customer ID" SKIP 1 ON "Customer Name" SKIP 1
SELECT *
FROM (
    SELECT EXTRACT(MONTH FROM SO.order_date) || '-' || EXTRACT(YEAR FROM SO.order_date) AS year,
    C.cust_ID,
    C.cust_name,
    COUNT (SO.order_no) AS total_order,
    SUM(SO.total_price) AS total_price
    FROM Customer C
    JOIN SalesOrder SO ON C.cust_id = SO.cust_id
    WHERE C.cust_id = SO.cust_id
    AND EXTRACT(YEAR FROM SO.order_date) = &v_year
    GROUP BY EXTRACT(MONTH FROM SO.order_date), EXTRACT(YEAR FROM SO.order_date), C.cust_ID, C.cust_name
    ORDER BY SUM(SO.total_price) DESC
)
WHERE ROWNUM <= &v_rowNum;
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF;
```

```
'Top N Customers Spent the Most in Orders In A Specific Year'
Enter Year: 2023
Enter Row Number: 10
old  11:     AND EXTRACT(YEAR FROM SO.order_date) = &v_year
new  11:     AND EXTRACT(YEAR FROM SO.order_date) =       2023
old  15: WHERE ROWNUM <= &v_rowNum
new  15: WHERE ROWNUM <=         10

Top 10 Customer Spent the Most in Orders in 2023
-----------------------------------------------------

Month - Year       Customer ID          Customer Name        Total Order    Total Price
------------------ -------------------- -------------------- ----------- --------------
2-2023             C0972                Ginevra Skillicorn             1      $7,216.53

2-2023             C1374                Osborne Sowley                 1      $6,926.55

1-2023             C8754                Gerardo Hobbema                1      $5,259.46

2-2023             C0934                Arin Mathewes                  1      $5,138.52

2-2023             C1710                Harrietta Brabin               1      $3,098.14

3-2023             C0236                Arch Disdel                    1      $3,018.38

2-2023             C6035                Suzann Eede                    1      $2,485.89

3-2023             C4561                Vanny Collihole                1      $1,986.57

3-2023             C4904                Ulrikaumeko Dossit             1      $1,841.31

3-2023             C9165                Charles Idwal Evans            1        $817.83


10 rows selected.
```

6.3 On Siew Lee
6.3.1 Query/Report 1: Total invoice made by customer
Purpose: The purpose of this query is to find the total invoice made by the customer that is more than a certain amount.

SQL statement:

```
SET linesize 150
SET pagesize 30

PROMPT Total invoice made by customer;
PROMPT What is the minimum amount of invoice you want to check?;
ACCEPT v_amount num(7,2) PROMPT 'Enter amount:';

COLUMN cust_id FORMAT A12 HEADING "Customer Id";

TTITLE COL20 'Total invoice made by customer ' SKIP 2

SELECT C.cust_id,c.cust_name, COUNT(i.invoice_no) AS Total_invoice,SUM(i.total_amount) AS Total_Price
FROM customer C
JOIN salesorder SO
ON C.cust_id = SO.cust_id
JOIN invoice I
ON so.order_no = i.order_no
GROUP BY EXTRACT(YEAR FROM SO.order_date),C.cust_id,c.cust_name
Having SUM(i.total_amount)>=&v_amount
ORDER BY Total_invoice;


CLEAR BREAKS;
CLEAR COLUMNS;
TTITLE OFF;
```

```
Total invoice made by customer
What is the minimum amount of invoice you want to check?
SP2-0003: Ill-formed ACCEPT command starting as (7,2) PROMPT 'Enter amount:'
Enter value for v_amount: 3000
old   8: Having SUM(i.total_amount)>=&v_amount
new   8: Having SUM(i.total_amount)>=3000

                  Total invoice made by customer

Customer Id  CUST_NAME                     TOTAL_INVOICE TOTAL_PRICE
------------ ----------------------------- ------------- -----------
C7299        Cordie Saffell                            1     3332.85
C8732        Jannelle Dimelow                          1     7716.86
C1579        Kaitlynn Asche                            1     6322.57
C0771        Mab Rubery                                1     3636.54
C1710        Harrietta Brabin                          1     3098.14
C6646        Zacharie Matuszak                         1     6392.11
C9705        Holt Housaman                             1     4759.21
C6051        Sophia Hymers                             1      6049.5
C9281        Marleah Bullent                           1      5272.8
C0236        Arch Disdel                               1     3018.38
C4798        Austen Edinburough                        1    11802.96
C4423        Peggi Kopfer                              1     4328.25
C6359        Noach Cantua                              1     3287.18
C0972        Ginevra Skillicorn                        1     7216.53
C1255        Jacquelynn Van Merwe                      1     7562.37

                  Total invoice made by customer

Customer Id  CUST_NAME                     TOTAL_INVOICE TOTAL_PRICE
------------ ----------------------------- ------------- -----------
C8885        Shelli Housley                            1     4973.67
C8754        Gerardo Hobbema                           1      3742.5
C0934        Arin Mathewes                             1     5138.52
C2557        Pete Mulvin                               1    10345.78
C2584        Ceil Ayllett                              1     5629.08
C1290        Trumann Nelissen                          1     9775.51
C4664        Mady Euplate                              1     5195.17
C4015        Reena Stickford                           1     4430.52
C4707        Isac Charon                               1     6985.91
C0747        Rufus Bartozzi                            1     3005.43
C3373        Benedetto Farman                          1     5561.29
C3685        Tabbi Bretherton                          1     9613.53
C5879        Tedda Lark                                1    10100.38
C1374        Osborne Sowley                            1     6926.55
C8285        Adeline Widdop                            1     3642.04
```

6.3.2 Query/Report 2: Total Number of Orders for each branch in a year
Purpose: The purpose of this query is to find the total number of orders for every branch in a specific year and calculate the total revenue of the orders.

SQL statement:

```
SET linesize 150
SET pagesize 80
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';
BREAK ON YEAR on branch_id NODUPLICATES;

PROMPT Total Number of Orders for each branch;
PROMPT Which year do you want to check?;
ACCEPT v_year CHAR FORMAT A4 PROMPT 'Enter year:';
COLUMN YEAR HEADING 'Year'
COLUMN branch_id FORMAT A9 HEADING 'Branch Id'
COLUMN branch_name FORMAT A30 HEADING 'Branch Name'
COLUMN Total_Orders HEADING 'Total Orders'
COLUMN Total_Revenue HEADING 'Total Revenue(RM)'
TTITLE COL20 'Total Number of Orders for each branch in '&v_year'';
COMPUTE SUM LABEL 'Total(RM):'OF Total_Revenue ON Year
CREATE OR REPLACE VIEW Total_Order_Each_Branch AS

SELECT EXTRACT(YEAR FROM SO.order_date) AS Year,B.branch_id,B.branch_name,COUNT(SO.order_no) AS
Total_Orders,SUM(SO.total_price) AS Total_Revenue
FROM Branch B
JOIN Staff S
ON B.branch_id=S.branch_id
JOIN SalesOrder SO
ON S.staff_id=SO.staff_id
AND EXTRACT(YEAR FROM SO.order_date)=&v_year
GROUP BY EXTRACT(YEAR FROM SO.order_date),B.branch_id,B.branch_name
ORDER BY B.branch_id,Total_Orders;
SELECT*
FROM Total_Order_Each_Branch;
```

```
Session altered.

Total Number of Orders for each branch
Which year do you want to check?
Enter year:2023
old   7: AND EXTRACT(YEAR FROM SO.order_date)=&v_year
new   7: AND EXTRACT(YEAR FROM SO.order_date)=2023

             Total Number of Orders for each branch in 2023
      Year Branch Id Branch Name                    Total Orders Total Revenue(RM)
---------- --------- ------------------------------ ------------ -----------------
      2023 B2491     Nestle Nihon Canpack (Malaysia            3           2767.29
                     ) Sdn Bhd

           B2571     Nestle Products Sdn Bhd                   1            477.63
           B2788     Nestle (Malaysia) Berhad                  3          14153.04
           B3376     Nestle Chembong Factory                   2            167.58
           B3714     Nestle Chembong (Ice Cream) Fa            3           2212.45
                     ctory

           B4058     Nestle Manufacturing Shah Alam            5          15413.92
                      (Malaysia) Sdn. Bhd

           B5880     Nestle Batu Tiga Factory                  1            817.83
           B6150     Nestle Kuching Factory                    1             199.5
           B6168     Nestle Sri Muda Factory                   3           4535.26
********** ********** ------------------------------------------ -----------------
Total(RM):                                                                40744.5

9 rows selected.
```

```
             Total Number of Orders for each branch in 2023
      Year Branch Id Branch Name                   Total Orders Total Revenue(RM)
---------- --------- ------------------------------ ------------ -----------------
      2022 B2491     Nestle Nihon Canpack (Malaysia            3           1897.63
                     ) Sdn Bhd

           B2571     Nestle Products Sdn Bhd                   7          26415.94
           B2788     Nestle (Malaysia) Berhad                  7          30549.83
           B3376     Nestle Chembong Factory                   5          14477.59
           B3714     Nestle Chembong (Ice Cream) Fa            7          17032.28
                     ctory

           B4058     Nestle Manufacturing Shah Alam           24          60459.65
                      (Malaysia) Sdn. Bhd

           B5801     Nestle Products Sdn Bhd                   6          28257.98
           B5880     Nestle Batu Tiga Factory                  3           3360.15
           B6150     Nestle Kuching Factory                    4           1398.33
           B6168     Nestle Sri Muda Factory                  12          25241.29
********** ********** ------------------------------------------ -----------------
Total(RM):                                                               209090.67

10 rows selected.
```

6.3.3 Query/Report 3: Query 3: Benefit of each product in Year
Purpose: The purpose of this query is to find the total benefit of each product in a specific year.

SQL statement:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';

PROMPT Benefit of each product;
PROMPT Which year do you want to check?;

ACCEPT v_year NUMBER PROMPT 'Enter Year > '

TTITLE COL20 'Benefit of each product in '&v_year'';

SELECT P.product_id, P.product_name, SUM(I.item_qty*I.unit_price) as total_benefit, SO.order_date
FROM Product P
JOIN ItemDetails I ON I.product_id = P.product_id
JOIN SalesOrder SO ON SO.order_no = I.order_no
WHERE P.product_id = I.product_id
AND EXTRACT(YEAR FROM SO.order_date) = &v_year
GROUP BY P.product_id, P.product_name, SO.order_date
ORDER BY total_benefit DESC;
```

```
Session altered.

Benefit of each product
Which year do you want to check?
Enter Year > 2023
old   6: AND EXTRACT(YEAR FROM SO.order_date) = &v_year
new   6: AND EXTRACT(YEAR FROM SO.order_date) =       2023

              Benefit of each product in 2023
PRODU PRODUCT_NAME                   TOTAL_BENEFIT ORDER_DATE
----- ------------------------------ ------------- ----------
PD010 Cafe HAG                              4703.72 20-02-2023
PD029 Coffee-Mate                            3742.5 08-01-2023
PD093 Maggi Hot and Sweet Sauce                2994 09-02-2023
PD017 Nestle After Eight                    2899.16 16-02-2023
PD017 Nestle After Eight                    2473.87 20-02-2023
PD099 Maggi Rich Tomato Sauce               2230.53 10-02-2023
PD099 Maggi Rich Tomato Sauce               1881.23 13-03-2023
PD066 Toll House                            1841.31 24-03-2023
PD094 Maggi Tomato Ketchup                  1568.07 09-02-2023
PD035 Nestle Resource                       1516.96 08-01-2023
PD014 Nestle Dibs                           1332.33 12-02-2023
PD009 Bonjour                                  1098 12-02-2023
PD091 Maggi Vegetable Noodles                889.77 13-03-2023
PD066 Toll House                             833.33 09-02-2023
PD036 Stouffer                               817.83 31-03-2023
PD092 Maggi Masala Noodles                   813.96 09-02-2023
PD010 Cafe HAG                               791.28 05-02-2023
PD021 Vittel                                 782.04 16-02-2023
PD013 Nestle Crunch                          754.21 16-02-2023
PD047 Milkybar                               728.34 16-03-2023
PD044 Crunch                                 645.84 18-03-2023
PD040 Herta                                  565.47 16-03-2023
PD019 Perrier                                496.34 16-02-2023
PD016 Nestle Lion                            484.38 12-02-2023
PD039 Carnation Breakfast                    477.63 06-02-2023
PD017 Nestle After Eight                     473.21 16-03-2023
PD074 Baby Ruth Crispety Crunchety           404.19 09-02-2023
PD051 Aero Bubbles                           293.02 09-02-2023
PD021 Vittel                                 287.28 08-01-2023
PD096 Maggi Magic Cubes                      255.36 10-02-2023
PD095 Maggi Bhuna Masala                     247.38 13-03-2023
PD005 Nestea Iced Tea                        239.04 12-03-2023
PD004 Milo                                   206.77 16-02-2023
PD033 Lean Cuisine                            199.6 16-03-2023
PD025 Toll House                              199.5 31-01-2023
PD003 Maggi Noodles                          151.62 14-02-2023
PD026 Gerber                                 131.56 12-02-2023
```

```
PD010 Cafe HAG                          791.28 05-02-2023
PD021 Vittel                            782.04 16-02-2023
PD013 Nestle Crunch                     754.21 16-02-2023
PD047 Milkybar                          728.34 16-03-2023
PD044 Crunch                            645.84 18-03-2023
PD040 Herta                             565.47 16-03-2023
PD019 Perrier                           496.34 16-02-2023
PD016 Nestle Lion                       484.38 12-02-2023
PD039 Carnation Breakfast               477.63 06-02-2023
PD017 Nestle After Eight                473.21 16-03-2023
PD074 Baby Ruth Crispety Crunchety      404.19 09-02-2023
PD051 Aero Bubbles                      293.02 09-02-2023
PD021 Vittel                            287.28 08-01-2023
PD096 Maggi Magic Cubes                 255.36 10-02-2023
PD095 Maggi Bhuna Masala                247.38 13-03-2023
PD005 Nestea Iced Tea                   239.04 12-03-2023
PD004 Milo                              206.77 16-02-2023
PD033 Lean Cuisine                       199.6 16-03-2023
PD025 Toll House                         199.5 31-01-2023
PD003 Maggi Noodles                     151.62 14-02-2023
PD026 Gerber                            131.56 12-02-2023
PD079 Chocapic Cereal                    83.86 21-01-2023
PD018 Buitoni Pasta                      51.87 12-02-2023
PD012 Nestle Cheerios                    38.94 20-02-2023
PD002 KitKat                             22.41 12-03-2023
PD083 Nestle Cerelac                     19.98 09-02-2023
PD056 Nestle Peppermint                  19.95 22-02-2023
PD023 Pumpkin of Libby                   19.95 16-03-2023
PD047 Milkybar                            19.9 23-03-2023
PD096 Maggi Magic Cubes                  15.96 25-03-2023
PD005 Nestea Iced Tea                     9.96 25-03-2023

47 rows selected.

SQL>
```

6.4 Goo Yong Kang

6.4.1 Query/Report 1: Top 10 worst sales by number of products sold

Purpose: The purpose of this query is to find the top 10 worst sales made by the total number of products sold in a specific year that will be input by the user.

SQL statement:

```
SET linesize 120
SET pagesize 100

PROMPT Report: Top 10 Worst Sales by Number of Products Sold
PROMPT -----------------------------------------------------

ACCEPT year_prompt NUMBER PROMPT 'Enter a year: '

COLUMN product_name FORMAT A30 HEADING 'PRODUCT NAME'
COLUMN num_products_sold FORMAT 999,999,999 HEADING 'NUMBER OF PRODUCTS SOLD'
COLUMN total_sales FORMAT $99,999,999.99 HEADING 'TOTAL SALES'

TTITLE 'Top 10 Worst Sales by Number of Products Sold in Year &year_prompt' SKIP 1 -
'-----------------------------------------------------------------' SKIP 2

SELECT *
FROM (
SELECT Product.product_name, SUM(ItemDetails.item_qty) AS num_products_sold, SUM(ItemDetails.unit_price * ItemDetails.item_qty) AS
total_sales
FROM Product
INNER JOIN ItemDetails ON Product.product_id = ItemDetails.product_id
INNER JOIN SalesOrder ON ItemDetails.order_no = SalesOrder.order_no
WHERE Product.product_name LIKE '%' || '%'
AND EXTRACT(YEAR FROM SalesOrder.order_date) = &year_prompt
GROUP BY Product.product_name
ORDER BY num_products_sold ASC
)
WHERE ROWNUM <= 10;

CLEAR COLUMNS
TTITLE OFF
```

Sample Output:

```
Report: Top 10 Worst Sales by Number of Products Sold
--------------------------------------------------
Enter a year: 2022
old   8: AND EXTRACT(YEAR FROM SalesOrder.order_date) = &year_prompt
new   8: AND EXTRACT(YEAR FROM SalesOrder.order_date) =       2022

Top 10 Worst Sales by Number of Products Sold in Year       2022
-----------------------------------------------------------------

PRODUCT NAME                   NUMBER OF PRODUCTS SOLD     TOTAL SALES
-------------------------- ----------------------- ---------------
Maggi Rich Tomato Sauce                        13           $64.87
Nestea Iced Tea                                18           $44.82
Buitoni Pasta                                  24           $95.76
Vittel                                         29          $115.71
Maggi Magic Cubes                              41          $163.59
Lean Cuisine                                   41          $204.59
Crunch                                         46          $137.54
Fitness Cereal                                 48          $383.52
Milkybar                                       55          $109.45
Nescafe Dolce Gusto                            56        $6,775.44

10 rows selected.
```

6.4.2 Query/Report 2: Top 10 product with the most profit

Purpose: The purpose of this query is to find the top 10 products with the most profit in a specific year that will be input by the user.

SQL statement:

```
SET linesize 120
SET pagesize 100

PROMPT Report: Top 10 product with the most profit
PROMPT ------------------------------------------

ACCEPT year_prompt NUMBER PROMPT 'Enter a year: '

COLUMN product_name FORMAT A30 HEADING 'PRODUCT NAME'
COLUMN num_products_sold FORMAT 999,999,999 HEADING 'NUMBER OF PRODUCTS SOLD'
COLUMN total_profit FORMAT $99,999,999.99 HEADING 'TOTAL PROFIT'

TTITLE 'Top 10 Product with the Most Profit in Year &year_prompt' SKIP 1 -
'------------------------------------------------------' SKIP 2

SELECT *
FROM (
SELECT Product.product_name, SUM(ItemDetails.item_qty) AS num_products_sold, SUM(ItemDetails.unit_price * ItemDetails.item_qty -
Product.cost * ItemDetails.item_qty) AS total_profit
FROM Product
INNER JOIN ItemDetails ON Product.product_id = ItemDetails.product_id
INNER JOIN SalesOrder ON ItemDetails.order_no = SalesOrder.order_no
WHERE Product.product_name LIKE '%' || '%'
AND EXTRACT(YEAR FROM SalesOrder.order_date) = &year_prompt
GROUP BY Product.product_name
ORDER BY total_profit DESC
)
WHERE ROWNUM <= 10;

CLEAR COLUMNS
TTITLE OFF
```

Sample output:

```
Report: Top 10 product with the most profit
----------------------------------------
Enter a year: 2023
old   8: AND EXTRACT(YEAR FROM SalesOrder.order_date) = &year_prompt
new   8: AND EXTRACT(YEAR FROM SalesOrder.order_date) =       2023

Top 10 Product with the Most Profit in Year       2023
------------------------------------------------

PRODUCT NAME                   NUMBER OF PRODUCTS SOLD    TOTAL PROFIT
------------------------------ ----------------------- ---------------
Maggi Rich Tomato Sauce                            824       $2,060.00
Maggi Hot and Sweet Sauce                          600       $1,500.00
Nestle After Eight                                 976         $976.00
Maggi Tomato Ketchup                               393         $786.00
Coffee-Mate                                        750         $750.00
Cafe HAG                                           500         $750.00
Nestle Resource                                    304         $608.00
Maggi Vegetable Noodles                            223         $602.10
Toll House                                         586         $586.00
Maggi Masala Noodles                               204         $550.80

10 rows selected.
```

6.4.3 Query/Report 3: Numbers of product supplied by supplier

Purpose: The purpose of this query is to display the numbers of products that are supplied by every supplier.

SQL statement:

```
SET linesize 120
SET pagesize 100

COLUMN supplier_name FORMAT A30 HEADING 'SUPPLIER NAME'
COLUMN num_products FORMAT 99999 HEADING 'NUMBER OF PRODUCTS SUPPLIED'

TTITLE 'Numbers of Product supplied by Supplier' SKIP 1 -
'-------------------------------------' SKIP 2
COMPUTE SUM LABEL 'Total:' OF num_products ON num_products

SELECT *
FROM (
SELECT supplier_name, COUNT(product_id) AS num_products
FROM Supplier
INNER JOIN Product ON Supplier.supplier_id = Product.supplier_id
GROUP BY supplier_name
ORDER BY num_products DESC
);

CLEAR COLUMNS
TTITLE OFF
```

Sample output:

```
Numbers of Product supplied by Supplier
---------------------------------------

SUPPLIER NAME                  NUMBER OF PRODUCTS SUPPLIED
------------------------------ ---------------------------
Avamm                                                   13
Feednation                                              12
Brightdog                                               11
DabZ                                                    10
Livetube                                                10
Yodo                                                     9
Topicblab                                                9
Trilia                                                   9
Gigashots                                                9
Vinte                                                    8

10 rows selected.
```

6.5 Jerome Lu Zheng Yao

6.5.1 Query/Report 1:How many people are using each different payment method at each payment?

Purpose: The purpose of this query is to find how many people are using each payment method in their payment.

SQL statement:

```
-- 6.5.1 How many people using each different payment method at each payment? --

SET linesize 120
SET pagesize 100

COLUMN payment_method FORMAT A30 HEADING 'Payment method'
COLUMN num_payments HEADING 'Total people'

TTITLE LEFT 'How many people using each different payment method at each payment?'SKIP 2

SELECT pc.payment_method, COUNT(*) AS num_payments
 FROM payment_channel pc
 JOIN payment p ON pc.payment_ref_no = p.payment_ref_no
 GROUP BY pc.payment_method
 ORDER BY num_payments DESC;

CLEAR COLUMNS
TTITLE OFF
```

```
How many people using each diifent payment method at each payment?

Payment method                Total people
----------------------------- ------------
cash                                    44
touch n go                              28
online banking                          28
```

6.5.2 Query/Report 2: Total amount of ordered stock by each invoice

Purpose: The purpose of this query is to find the total amount of ordered by each invoice.

SQL statement:

```
-- 6.5.2 Total amount of ordered stock by each invoice --

SET linesize 120
SET pagesize 100

COLUMN invoice_no HEADING 'Code of invoice' FORMAT A20
COLUMN total_ordered_stock HEADING 'Total quantity'

TTITLE LEFT 'Total amount of ordered stock by each invoice' SKIP 2


 SELECT i.invoice_no, SUM(os.item_qty) as total_ordered_stock
    FROM Invoice i
    JOIN OrderedStock os
    ON i.invoice_no = os.invoice_no
    GROUP BY i.invoice_no
    ORDER BY total_ordered_stock DESC;

CLEAR COLUMNS
TTITLE OFF
```

```
Total amount of ordered stock by each inovice

Code of inovice      Total quantity
-------------------- --------------
I3804                          2492
I9334                          2480
I8635                          1545
I6620                          1509
I8859                          1397
I0140                          1349
I8192                          1158
I0949                          1133
I8574                          1093
I6861                          1082
I6072                          1054
I9467                           973
I8408                           963
I3715                           948
I4389                           914
I7839                           892
I9270                           847
I0737                           837
I5757                           820
I1028                           753
I0074                           746
I9117                           719
I8003                           711
I1819                           686
I1016                           679
I5484                           676
I3622                           673
I0415                           662
I3480                           651
I9643                           630
I2563                           593
I1638                           590
I4754                           584
I2516                           582
I4866                           568
I5835                           557
I3577                           533
```

I0053	511
I7895	511
I3479	498
I1712	495
I9320	486
I6084	451
I3263	420
I6371	406
I2701	396
I3598	390
I0683	371
I8010	369
I5876	369
I4690	296
I4082	281
I4010	273
I2275	249
I1908	242
I5630	221
I0667	217
I5685	216
I7785	127
I1593	118
I4413	117
I3486	114
I1463	105
I2932	100
I4293	97
I4284	90
I0626	90
I3009	87
I3554	76
I9170	72
I1210	72
I5230	70
I6710	70
I5052	67
I8904	50
I0779	50
I6033	49
I8661	47

```
I3031                            38
I6289                            30
I2777                            22
I5546                            18
I3689                            16
I0109                            14
I8011                            13
I0098                            13
I9613                            12
I7320                            10
I3902                            10
I3264                            10
I8847                             8
I2729                             5
I3797                             5
I1107                             4

Total amount of ordered stock by each inovice

Code of inovice      Total quantity
-------------------- --------------
I6780                             4
I4333                             4
I6134                             4
I4458                             2
I9206                             1

100 rows selected.
```

6.5.3 Query/Report 3: Top 10 best selling products

Purpose: The purpose of this query is to find the top 10 products that having sold the most product.

SQL statement:

```
-- 6.4.3 Top 10 best selling products --



SET linesize 120
SET pagesize 100


TTITLE 'Top 10 best selling products

 SELECT *
    FROM (
    SELECT Product.product_name, SUM(ItemDetails.item_qty) AS num_products_sold, SUM(ItemDetails.unit_price * ItemDetails.item_qty)
AS total_sales
    FROM Product
    INNER JOIN ItemDetails ON Product.product_id = ItemDetails.product_id
    INNER JOIN SalesOrder ON ItemDetails.order_no = SalesOrder.order_no
    WHERE Product.product_name LIKE '%' || '%'
    AND EXTRACT(YEAR FROM SalesOrder.order_date) = &year_prompt
    GROUP BY Product.product_name
   ORDER BY num_products_sold DESC
  )
WHERE ROWNUM <= 10;


CLEAR COLUMNS
TTITLE OFF
```

```
Top 10 best selling products

PRODUCT_NAME                   NUM_PRODUCTS_SOLD TOTAL_SALES
------------------------------ ----------------- -----------
S. Pellegrino                               1267     6322.33
Nestle Lion                                 1224     3659.76
Milkybar Buttons                            1205     2397.95
Nestea                                      1118     5578.82
Nesquik                                      913     4840.87
Nestle Coffee Crisp                          883     1315.67
Nestle Bear Brand                            855     7686.45
Nestle Nips                                  802     2397.98
Nestle Peptamen                              802    12823.98
Nestle Crunch                                774     2508.26

10 rows selected.
```

Individual References

Tan Yen Fang, On Siew Lee, Yap Zhi Qian, Goo Yong Kang, Jerome Lu Zheng Yao

1. Oracle Tutorial, (2019), Oracle Tutorial, https://www.oracletutorial.com/.