MULTIMEDIA UNIVERSITY

CIT6234 - ADVANCED DATABASE

ASSIGNMENT 1 (30%)
GROUP 4

CS3: Data warehouse for AirAsia flight ticket booking system

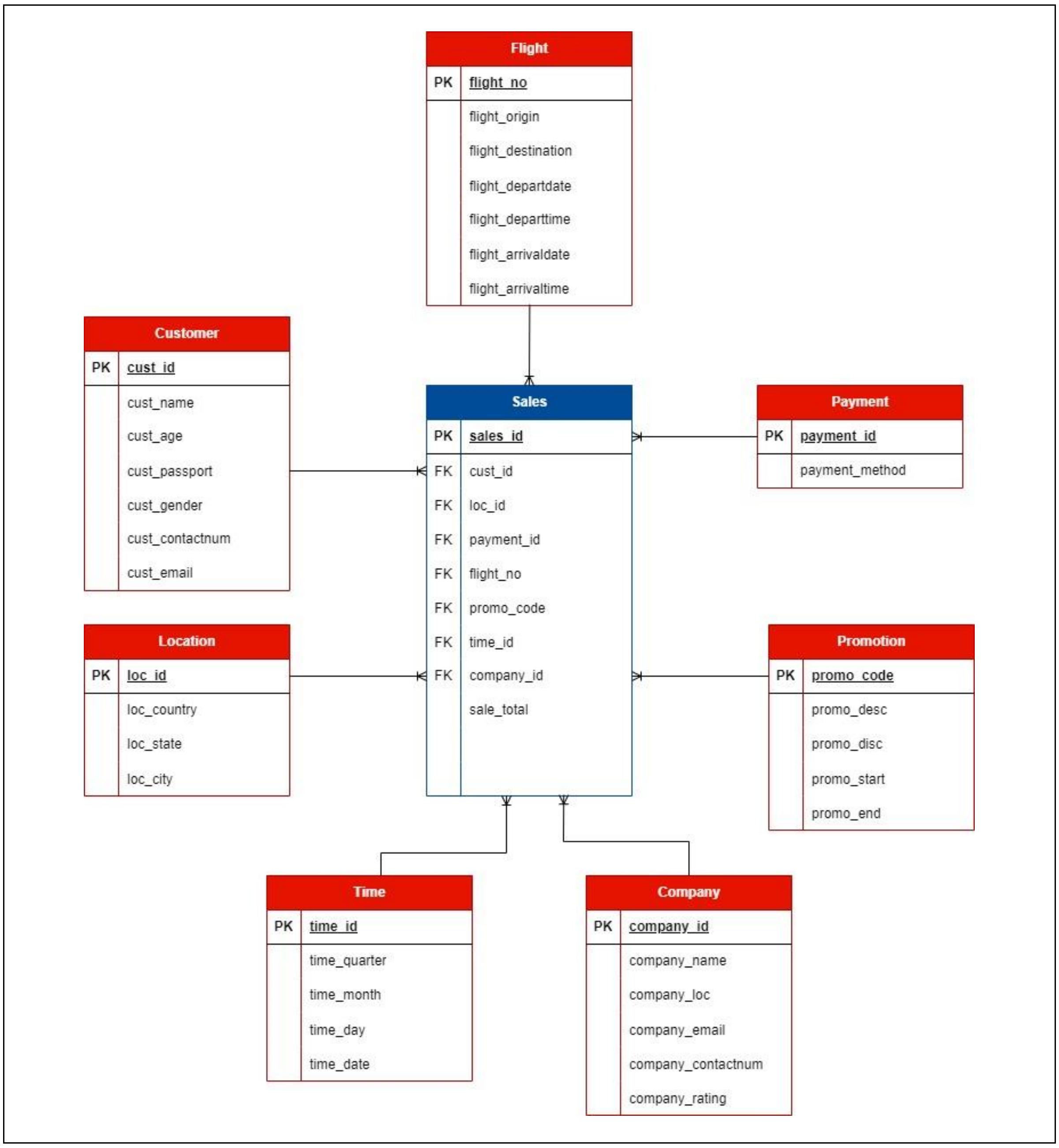
Design a data warehouse to keep track of flight ticket booking system for AirAsia (https://www.airasia.com/flight/). This include information such as destination from, destination to, date, time, promotion, and so on.

Prepared by:

Name	Student ID	Student Email
Nur Fatin Nabilah Binti Md.Irzan	1221302587	1221302587@student.mmu.edu.my
Nur Alia Shazwani Binti Mohd Nazri	1231302985	1231302985@student.mmu.edu.my
Haizatul Nazirah Nizam Binti Hairunizam	1231303504	1231303504@student.mmu.edu.my
Wong Jin Yin	1211100975	1211100975@student.mmu.edu.my

Table of Contents

1.0 Star Schema



2.0 Data Dictionary

Table Name	Attribute Name	Contents	Type	Format	Range	Request	PK/ FK	FK Referenced Table
Sales	sales_id	Sales Identify Number	Varchar(20)	Xxxxxxxx	1-9999999999	Y	PK	
	cust_id	Customer Identify Number	Int			Y	FK	Customer
	loc_id	Location Identify Number	Char(10)	9999999999		Y	FK	Location
	payment_id	Payment Identify Number	BigInt			Y	FK	Payment
	flight_no	Flight Identify Number	Varchar(20)	Xxxxxxxx		Y	FK	Flight
	promo_code	Promo Code	Vachar(20)	Xxxxxxxx		Y	FK	Promotion
	time_id	Time Identify Number	BigInt			Y	FK	Time
	company_id	Company Identify Number	Char(10)	9999999999		Y	FK	Company
	sale_total	Total of sales	Decimal(7,2)	99999.99		Y		
Flight	flight_no	Flight Number	Varchar(20)	Xxxxxxxx	1-9999999999	Y	PK	
	flight_origin	Flight Origin	Char(50)	9999999999		Y		
	flight_destination	Flight Destination	Char(50)	9999999999		Y		
	flight_departdate	Flight Depart Date	Date	YYYY-MM-DD		Y		
	flight_departtime	Flight Depart Time	Time	99.99.99		Y		
	flight_arrivaldate	Flight Arrival Date	Date	YYYY-MM-DD		Y		
	flight_arrivaltime	Flight Arrival Time	Time	99.99.99		Y		
Company	company_id	Company Identify Number	Char(10)	9999999999	1-9999999999	Y	PK	
	company_name	Name of Company	Char(100)	Xxxxxxxx		Y		
	company_loc	Location of Company	Char(50)	Xxxxxxxx		Y		
	company_email	Email of Company	Varchar(50)	Xxxxxxxx		Y		
	company_contactnum	Contact Number of Company	Varchar(50)	999-999999999		Y		
	company_rating	Rating of Company	Decimal (2,1)	9.9		Y		
Customer	cust_id	Customer Identify Number	Int		1-9999999999	Y	PK	
	cust_name	Name of Customer	Varchar(100)	9999999999		Y		
	cust_age	Age of Customer	Int			Y		
	cust_passport	Passport No of Customer	Varchar(25)	Xxxxxxxx		Y		
	cust_gender	Gender of Customer	Char(20)	9999999999		Y		
	cust_contactnum	Contact Number of Customer	Char(20)	9999999999		Y		
	cust_email	Email of Customer	Varchar(30)	Xxxxxxxx				
Location	loc_id	Location Identify Number	Char(10)	9999999999	1-9999999999	Y	PK	
	loc_country	Country of location	Char(50)	9999999999		Y		
	loc_state	State of location	Char(50)	9999999999		Y		
	loc_city	City of location	Char(50)	9999999999		Y		
Time	time_id	Time Identify Number	BigInt		1-9999999999	Y	PK	
	time_quater	Quarter of the time	Int			Y		
	time_month	Month of the time	Char(20)	9999999999		Y		
	time_day	Day of the time	Char(20)	9999999999		Y		
	time_date	Date of the date	Date	YYYY-MM-DD		Y		
Payment	payment_id	Payment Identify Number	BigInt		1-9999999999	Y	PK	
	payment_method	Payment Method	Varchar(10)	Xxxxxxxx		Y		
Promotion	promo_code	Promo code	Varchar(20)	Xxxxxxxx	1-9999999999	Y	PK	
	promo_desc	Description about Promotion	Varchar(100)	Xxxxxxxx		Y		
	promo_disc	Discount percentage	Decimal(3,1)	99.9		Y		
	promo_start	Promo start date	Date	YYYY-MM-DD		Y		
	promo_end	Promo end date	Date	YYYY-MM-DD		Y		

3.0 Storage Size

Each table have 10 rows of data,

Size of fact table by rows = 10*10*10*10*10*10*10

= 10,000,000 rows

= 10^7 rows

To find the average bytes of each table,

Each row consists of the following columns:

- `Sales_id` (varchar (20)): Approximately 20 bytes.
- `Cust_id` (int): 4 bytes.
- `Loc_id` (char (10)): A fixed-length char(10) takes 10 bytes.
- `Time_id` (bigint): 8 bytes.
- `Company_id` (char (10)): A fixed-length char(10) takes 10 bytes.
- `Promo_code` (varchar (20)): Approximately 20 bytes.
- `Payment_id` (bigint): 8 bytes.
- `Flight_no` (varchar (20)): Approximately 20 bytes.
- `Sale_total` (decimal (7,2)): For decimal, the total of round down by dividing the length with 2 then add 1, it will be 4 bytes.

Average bytes of fact table = ((20+4+10+8+10+20+8+20+(7/2+1=4))/9

= 104/9

= 11.55 bytes

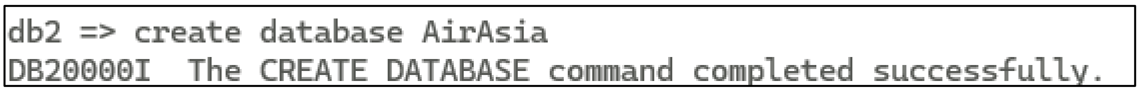
Total storage for fact table = 10,000,000 * 11.55 * 9

= 110,302,500,000 bytes = 110.3025 GB

4.0 Data Definition Language (DDL)

Implement the data warehouse on IBM DB2 using SQL commands (Data Definition Language (DDL)).

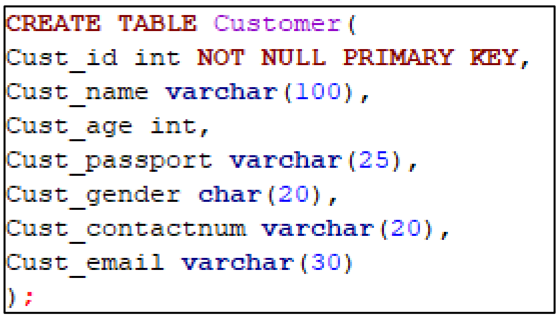
i. Creating database using db2 Command Line Processor.



```
db2 => create database AirAsia
DB20000I  The CREATE DATABASE command completed successfully.
```

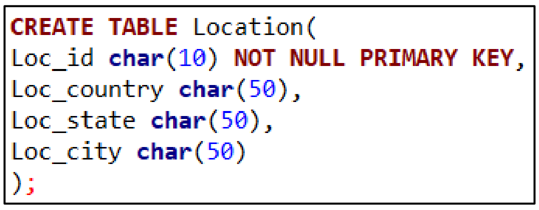
ii. SQL Scripts for tables creation.

- Table Customer



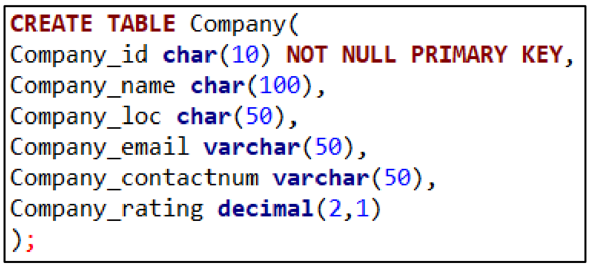
```sql
CREATE TABLE Customer(
Cust_id int NOT NULL PRIMARY KEY,
Cust_name varchar(100),
Cust_age int,
Cust_passport varchar(25),
Cust_gender char(20),
Cust_contactnum varchar(20),
Cust_email varchar(30)
);
```

- Table Location



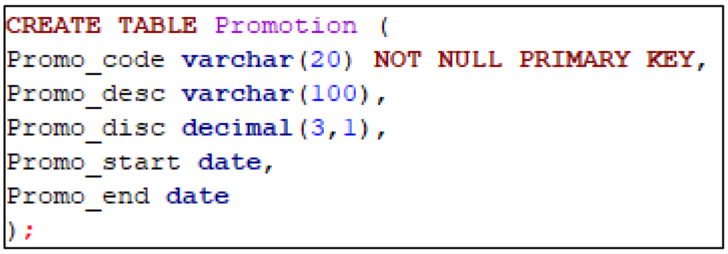
```sql
CREATE TABLE Location(
Loc_id char(10) NOT NULL PRIMARY KEY,
Loc_country char(50),
Loc_state char(50),
Loc_city char(50)
);
```

- Table Company



```sql
CREATE TABLE Company(
Company_id char(10) NOT NULL PRIMARY KEY,
Company_name char(100),
Company_loc char(50),
Company_email varchar(50),
Company_contactnum varchar(50),
Company_rating decimal(2,1)
);
```

- Table Promotion



```sql
CREATE TABLE Promotion (
Promo_code varchar(20) NOT NULL PRIMARY KEY,
Promo_desc varchar(100),
Promo_disc decimal(3,1),
Promo_start date,
Promo_end date
);
```

- Table Payment

```
CREATE TABLE Payment (
Payment_id bigint NOT NULL PRIMARY KEY,
Payment_method varchar(50)
);
```

- Table Flight

```
CREATE TABLE Flight (
Flight_no varchar(20) NOT NULL PRIMARY KEY,
Flight_origin char(50),
Flight_destination char(50),
Flight_departdate date,
Flight_departtime time,
Flight_arrivaldate date,
Flight_arrivaltime time
);
```

- Table Time

```
CREATE TABLE Time (
Time_id bigint NOT NULL PRIMARY KEY,
Time_quarter int,
Time_month char(20),
Time_day char(20),
Time_date date
);
```

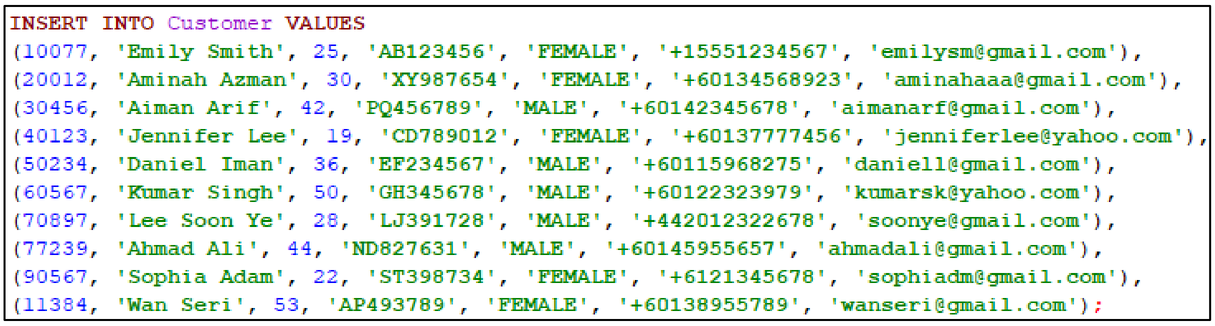
- Table Sales

```
CREATE TABLE Sales (
Sales_id varchar(20) NOT NULL PRIMARY KEY,
Cust_id int,
Loc_id char(10),
Time_id bigint,
Company_id char(10),
Promo_code varchar(20),
Payment_id bigint,
Flight_no varchar(20),
Sale_total decimal (7,2),

FOREIGN KEY (Cust_id) REFERENCES Customer,
FOREIGN KEY (Loc_id) REFERENCES Location,
FOREIGN KEY (Time_id) REFERENCES Time,
FOREIGN KEY (Company_id) REFERENCES Company,
FOREIGN KEY (Promo_code) REFERENCES Promotion,
FOREIGN KEY (Payment_id) REFERENCES Payment,
FOREIGN KEY (Flight_no) REFERENCES Flight
);
```

5.0 Data Manipulation Language (DML)

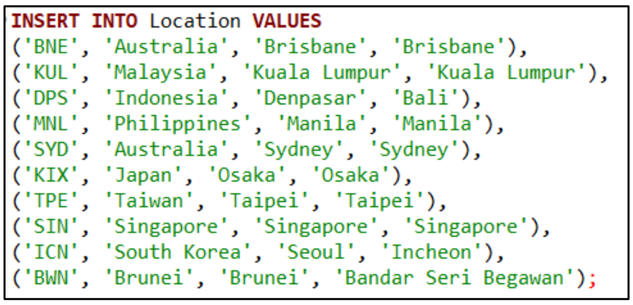
Enter sample data into the data warehouse using SQL commands (Data Manipulation Language (DML))

- Customer



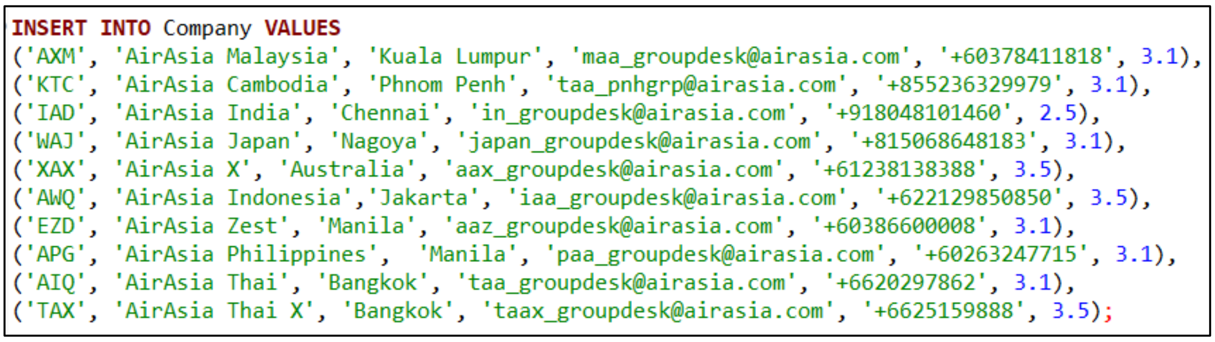
```
INSERT INTO Customer VALUES
(10077, 'Emily Smith', 25, 'AB123456', 'FEMALE', '+15551234567', 'emilysm@gmail.com'),
(20012, 'Aminah Azman', 30, 'XY987654', 'FEMALE', '+60134568923', 'aminahaaa@gmail.com'),
(30456, 'Aiman Arif', 42, 'PQ456789', 'MALE', '+60142345678', 'aimanarf@gmail.com'),
(40123, 'Jennifer Lee', 19, 'CD789012', 'FEMALE', '+60137777456', 'jenniferlee@yahoo.com'),
(50234, 'Daniel Iman', 36, 'EF234567', 'MALE', '+60115968275', 'daniell@gmail.com'),
(60567, 'Kumar Singh', 50, 'GH345678', 'MALE', '+60122323979', 'kumarsk@yahoo.com'),
(70897, 'Lee Soon Ye', 28, 'LJ391728', 'MALE', '+442012322678', 'soonye@gmail.com'),
(77239, 'Ahmad Ali', 44, 'ND827631', 'MALE', '+60145955657', 'ahmadali@gmail.com'),
(90567, 'Sophia Adam', 22, 'ST398734', 'FEMALE', '+6121345678', 'sophiadm@gmail.com'),
(11384, 'Wan Seri', 53, 'AP493789', 'FEMALE', '+60138955789', 'wanseri@gmail.com');
```

- Location



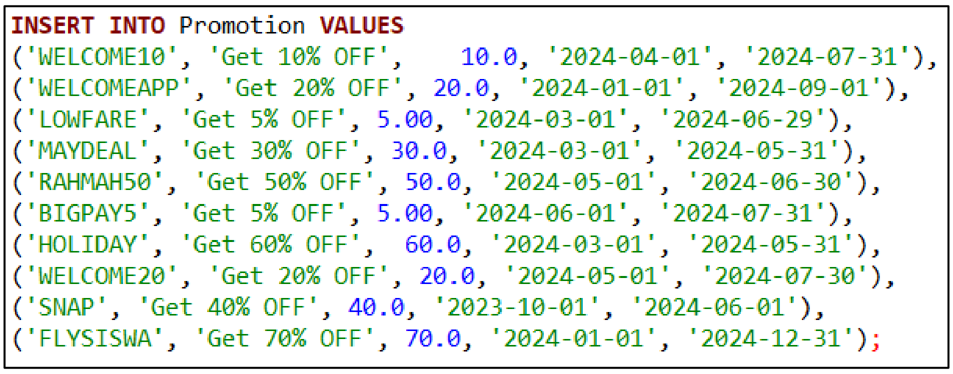
```
INSERT INTO Location VALUES
('BNE', 'Australia', 'Brisbane', 'Brisbane'),
('KUL', 'Malaysia', 'Kuala Lumpur', 'Kuala Lumpur'),
('DPS', 'Indonesia', 'Denpasar', 'Bali'),
('MNL', 'Philippines', 'Manila', 'Manila'),
('SYD', 'Australia', 'Sydney', 'Sydney'),
('KIX', 'Japan', 'Osaka', 'Osaka'),
('TPE', 'Taiwan', 'Taipei', 'Taipei'),
('SIN', 'Singapore', 'Singapore', 'Singapore'),
('ICN', 'South Korea', 'Seoul', 'Incheon'),
('BWN', 'Brunei', 'Brunei', 'Bandar Seri Begawan');
```

- Company



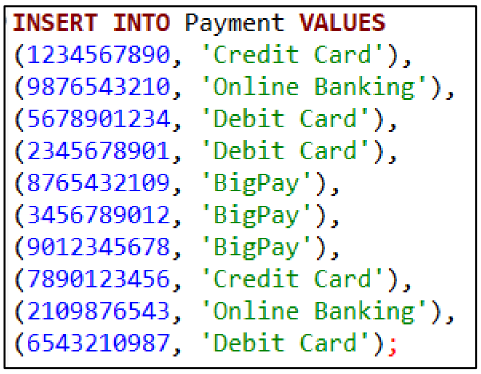
```
INSERT INTO Company VALUES
('AXM', 'AirAsia Malaysia', 'Kuala Lumpur', 'maa_groupdesk@airasia.com', '+60378411818', 3.1),
('KTC', 'AirAsia Cambodia', 'Phnom Penh', 'taa_pnhgrp@airasia.com', '+855236329979', 3.1),
('IAD', 'AirAsia India', 'Chennai', 'in_groupdesk@airasia.com', '+918048101460', 2.5),
('WAJ', 'AirAsia Japan', 'Nagoya', 'japan_groupdesk@airasia.com', '+815068648183', 3.1),
('XAX', 'AirAsia X', 'Australia', 'aax_groupdesk@airasia.com', '+61238138388', 3.5),
('AWQ', 'AirAsia Indonesia','Jakarta', 'iaa_groupdesk@airasia.com', '+622129850850', 3.5),
('EZD', 'AirAsia Zest', 'Manila', 'aaz_groupdesk@airasia.com', '+60386600008', 3.1),
('APG', 'AirAsia Philippines',  'Manila', 'paa_groupdesk@airasia.com', '+60263247715', 3.1),
('AIQ', 'AirAsia Thai', 'Bangkok', 'taa_groupdesk@airasia.com', '+6620297862', 3.1),
('TAX', 'AirAsia Thai X', 'Bangkok', 'taax_groupdesk@airasia.com', '+6625159888', 3.5);
```

- Promotion



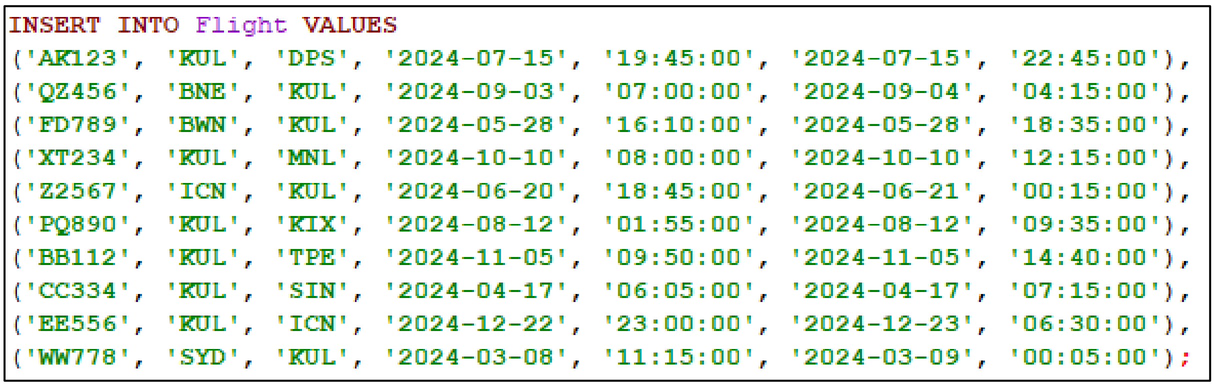
```
INSERT INTO Promotion VALUES
('WELCOME10', 'Get 10% OFF',    10.0, '2024-04-01', '2024-07-31'),
('WELCOMEAPP', 'Get 20% OFF', 20.0, '2024-01-01', '2024-09-01'),
('LOWFARE', 'Get 5% OFF', 5.00, '2024-03-01', '2024-06-29'),
('MAYDEAL', 'Get 30% OFF', 30.0, '2024-03-01', '2024-05-31'),
('RAHMAH50', 'Get 50% OFF', 50.0, '2024-05-01', '2024-06-30'),
('BIGPAY5', 'Get 5% OFF', 5.00, '2024-06-01', '2024-07-31'),
('HOLIDAY', 'Get 60% OFF',  60.0, '2024-03-01', '2024-05-31'),
('WELCOME20', 'Get 20% OFF', 20.0, '2024-05-01', '2024-07-30'),
('SNAP', 'Get 40% OFF', 40.0, '2023-10-01', '2024-06-01'),
('FLYSISWA', 'Get 70% OFF', 70.0, '2024-01-01', '2024-12-31');
```

- Payment



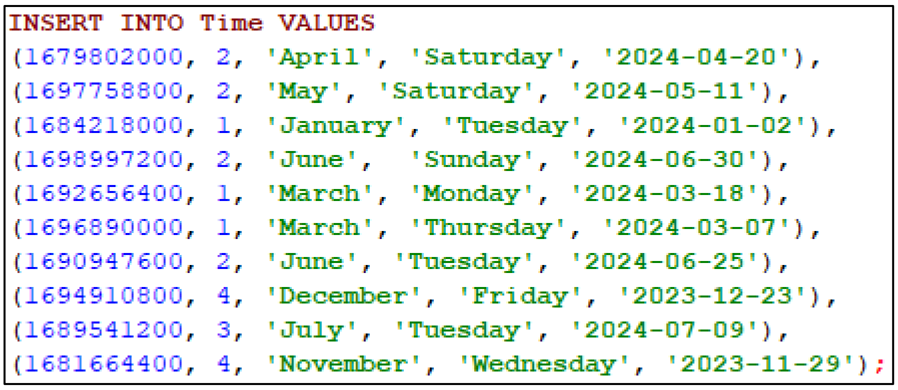
```sql
INSERT INTO Payment VALUES
(1234567890, 'Credit Card'),
(9876543210, 'Online Banking'),
(5678901234, 'Debit Card'),
(2345678901, 'Debit Card'),
(8765432109, 'BigPay'),
(3456789012, 'BigPay'),
(9012345678, 'BigPay'),
(7890123456, 'Credit Card'),
(2109876543, 'Online Banking'),
(6543210987, 'Debit Card');
```

- Flight



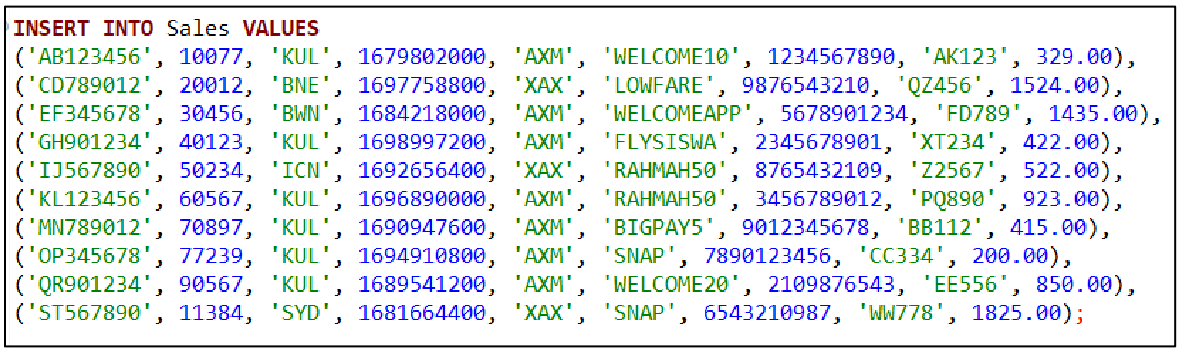
```sql
INSERT INTO Flight VALUES
('AK123', 'KUL', 'DPS', '2024-07-15', '19:45:00', '2024-07-15', '22:45:00'),
('QZ456', 'BNE', 'KUL', '2024-09-03', '07:00:00', '2024-09-04', '04:15:00'),
('FD789', 'BWN', 'KUL', '2024-05-28', '16:10:00', '2024-05-28', '18:35:00'),
('XT234', 'KUL', 'MNL', '2024-10-10', '08:00:00', '2024-10-10', '12:15:00'),
('Z2567', 'ICN', 'KUL', '2024-06-20', '18:45:00', '2024-06-21', '00:15:00'),
('PQ890', 'KUL', 'KIX', '2024-08-12', '01:55:00', '2024-08-12', '09:35:00'),
('BB112', 'KUL', 'TPE', '2024-11-05', '09:50:00', '2024-11-05', '14:40:00'),
('CC334', 'KUL', 'SIN', '2024-04-17', '06:05:00', '2024-04-17', '07:15:00'),
('EE556', 'KUL', 'ICN', '2024-12-22', '23:00:00', '2024-12-23', '06:30:00'),
('WW778', 'SYD', 'KUL', '2024-03-08', '11:15:00', '2024-03-09', '00:05:00');
```

- Time



```sql
INSERT INTO Time VALUES
(1679802000, 2, 'April', 'Saturday', '2024-04-20'),
(1697758800, 2, 'May', 'Saturday', '2024-05-11'),
(1684218000, 1, 'January', 'Tuesday', '2024-01-02'),
(1698997200, 2, 'June',  'Sunday', '2024-06-30'),
(1692656400, 1, 'March', 'Monday', '2024-03-18'),
(1696890000, 1, 'March', 'Thursday', '2024-03-07'),
(1690947600, 2, 'June', 'Tuesday', '2024-06-25'),
(1694910800, 4, 'December', 'Friday', '2023-12-23'),
(1689541200, 3, 'July', 'Tuesday', '2024-07-09'),
(1681664400, 4, 'November', 'Wednesday', '2023-11-29');
```

- Sales



```sql
INSERT INTO Sales VALUES
('AB123456', 10077, 'KUL', 1679802000, 'AXM', 'WELCOME10', 1234567890, 'AK123', 329.00),
('CD789012', 20012, 'BNE', 1697758800, 'XAX', 'LOWFARE', 9876543210, 'QZ456', 1524.00),
('EF345678', 30456, 'BWN', 1684218000, 'AXM', 'WELCOMEAPP', 5678901234, 'FD789', 1435.00),
('GH901234', 40123, 'KUL', 1698997200, 'AXM', 'FLYSISWA', 2345678901, 'XT234', 422.00),
('IJ567890', 50234, 'ICN', 1692656400, 'XAX', 'RAHMAH50', 8765432109, 'Z2567', 522.00),
('KL123456', 60567, 'KUL', 1696890000, 'AXM', 'RAHMAH50', 3456789012, 'PQ890', 923.00),
('MN789012', 70897, 'KUL', 1690947600, 'AXM', 'BIGPAY5', 9012345678, 'BB112', 415.00),
('OP345678', 77239, 'KUL', 1694910800, 'AXM', 'SNAP', 7890123456, 'CC334', 200.00),
('QR901234', 90567, 'KUL', 1689541200, 'AXM', 'WELCOME20', 2109876543, 'EE556', 850.00),
('ST567890', 11384, 'SYD', 1681664400, 'XAX', 'SNAP', 6543210987, 'WW778', 1825.00);
```

6.0 Procedural SQL

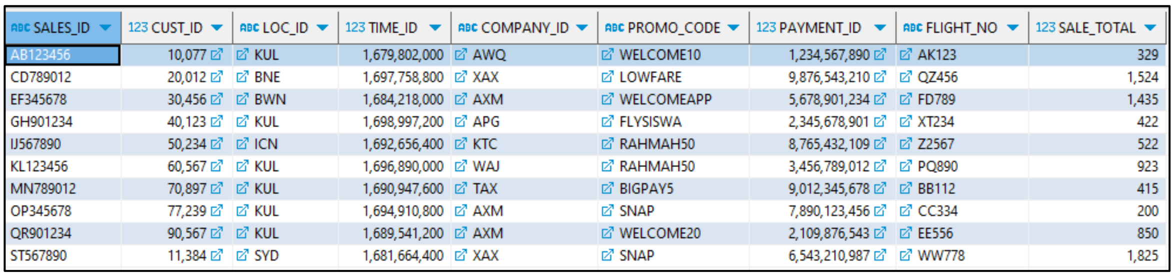
6.1 Stored Procedure

The Stored Procedure updates each sale totals in the ‘Sales’ table based on promotional offers specified in the ‘Promotion’ table. It loops through each distinct promo code, calculates the discount for sales that used the promo code and updates the sale total accordingly in the ‘Sales’ table.

- SQL Command

```sql
CREATE PROCEDURE afterDiscount()
LANGUAGE SQL
BEGIN
    DECLARE done BOOLEAN DEFAULT FALSE;
    DECLARE PromoCode VARCHAR(20);
    DECLARE num_rows_affected INT;
    DECLARE PromoCursor CURSOR FOR
        SELECT DISTINCT Promo_code
        FROM Promotion;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

    OPEN PromoCursor;

    read_loop: LOOP
        FETCH PromoCursor INTO PromoCode;
        IF done THEN
            LEAVE read_loop;
        END IF;

       SELECT COUNT(*) INTO num_rows_affected
        FROM Sales
        WHERE Promo_code = PromoCode;

        IF num_rows_affected > 0 THEN
            UPDATE Sales s
            SET Sale_total = Sale_total - (
                SELECT SUM(s.Sale_total * pd.Promo_disc / 100)
                FROM Promotion pd
                WHERE pd.Promo_code = s.Promo_code
                AND pd.Promo_code = PromoCode
            )
            WHERE s.Promo_code = PromoCode;
        ELSE

        END IF;
    END LOOP;

    CLOSE PromoCursor;
END
```

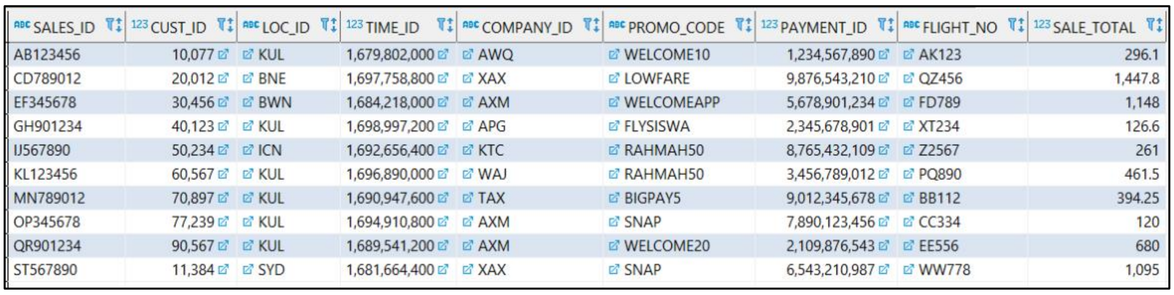
- Before Store Procedure



SALES_ID	CUST_ID	LOC_ID	TIME_ID	COMPANY_ID	PROMO_CODE	PAYMENT_ID	FLIGHT_NO	SALE_TOTAL
AB123456	10,077	KUL	1,679,802,000	AWQ	WELCOME10	1,234,567,890	AK123	329
CD789012	20,012	BNE	1,697,758,800	XAX	LOWFARE	9,876,543,210	QZ456	1,524
EF345678	30,456	BWN	1,684,218,000	AXM	WELCOMEAPP	5,678,901,234	FD789	1,435
GH901234	40,123	KUL	1,698,997,200	APG	FLYSISWA	2,345,678,901	XT234	422
IJ567890	50,234	ICN	1,692,656,400	KTC	RAHMAH50	8,765,432,109	Z2567	522
KL123456	60,567	KUL	1,696,890,000	WAJ	RAHMAH50	3,456,789,012	PQ890	923
MN789012	70,897	KUL	1,690,947,600	TAX	BIGPAY5	9,012,345,678	BB112	415
OP345678	77,239	KUL	1,694,910,800	AXM	SNAP	7,890,123,456	CC334	200
QR901234	90,567	KUL	1,689,541,200	AXM	WELCOME20	2,109,876,543	EE556	850
ST567890	11,384	SYD	1,681,664,400	XAX	SNAP	6,543,210,987	WW778	1,825

- After Store Procedure

```sql
CALL afterDiscount();

SELECT * FROM Sales;
```



SALES_ID	CUST_ID	LOC_ID	TIME_ID	COMPANY_ID	PROMO_CODE	PAYMENT_ID	FLIGHT_NO	SALE_TOTAL
AB123456	10,077	KUL	1,679,802,000	AWQ	WELCOME10	1,234,567,890	AK123	296.1
CD789012	20,012	BNE	1,697,758,800	XAX	LOWFARE	9,876,543,210	QZ456	1,447.8
EF345678	30,456	BWN	1,684,218,000	AXM	WELCOMEAPP	5,678,901,234	FD789	1,148
GH901234	40,123	KUL	1,698,997,200	APG	FLYSISWA	2,345,678,901	XT234	126.6
IJ567890	50,234	ICN	1,692,656,400	KTC	RAHMAH50	8,765,432,109	Z2567	261
KL123456	60,567	KUL	1,696,890,000	WAJ	RAHMAH50	3,456,789,012	PQ890	461.5
MN789012	70,897	KUL	1,690,947,600	TAX	BIGPAY5	9,012,345,678	BB112	394.25
OP345678	77,239	KUL	1,694,910,800	AXM	SNAP	7,890,123,456	CC334	120
QR901234	90,567	KUL	1,689,541,200	AXM	WELCOME20	2,109,876,543	EE556	680
ST567890	11,384	SYD	1,681,664,400	XAX	SNAP	6,543,210,987	WW778	1,095

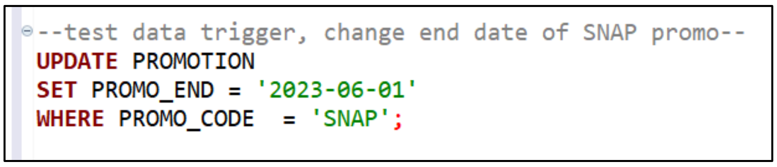
6.2 Trigger

To create a trigger and check if the data inserted in Sales are invalid such as expired promo code or cust_id is not existed, then display error message. If all data are valid, then update the company rating by 0.1 for each of the new sale made.

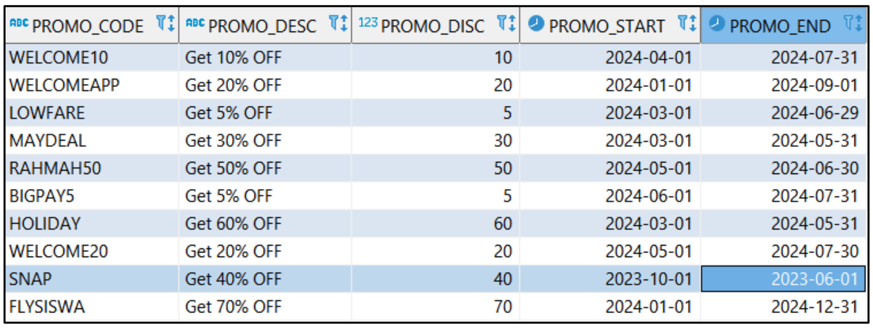
- Trigger SQL Command

```sql
CREATE TRIGGER CheckPromoCode
BEFORE INSERT ON SALES
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL

BEGIN
DECLARE promo_start DATE;
DECLARE promo_end DATE;
DECLARE current_company_rating DECIMAL(2, 1);
DECLARE new_company_rating DECIMAL(2, 1);

    -- get the promo start and end dates
    SELECT promo_start, promo_end INTO promo_start, promo_end
    FROM PROMOTION pd
    WHERE pd.promo_code = NEW.promo_code;

    -- check if promo code is valid
    IF CURRENT DATE NOT BETWEEN promo_start AND promo_end THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Promo code is not valid.';
    END IF;

    --check if cust exist
    IF NOT EXISTS (SELECT 1 FROM CUSTOMER cu WHERE cu.cust_id = NEW.cust_id) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Customer does not exist.';
    END IF;

    -- select current company rating
    SELECT company_rating INTO current_company_rating
    FROM COMPANY c
    WHERE c.company_id = NEW.company_id;

    --Calculate new company rating
    SET new_company_rating = current_company_rating + 0.1;

    -- Update company rating
    UPDATE COMPANY
    SET company_rating = new_company_rating
    WHERE company_id = NEW.company_id;
END
```

- Update the end date of the SNAP promo code to 2023-06-01.



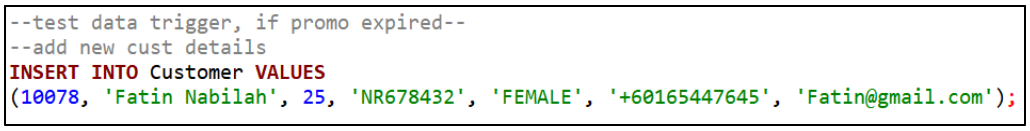
```
--test data trigger, change end date of SNAP promo--
UPDATE PROMOTION
SET PROMO_END = '2023-06-01'
WHERE PROMO_CODE  = 'SNAP';
```



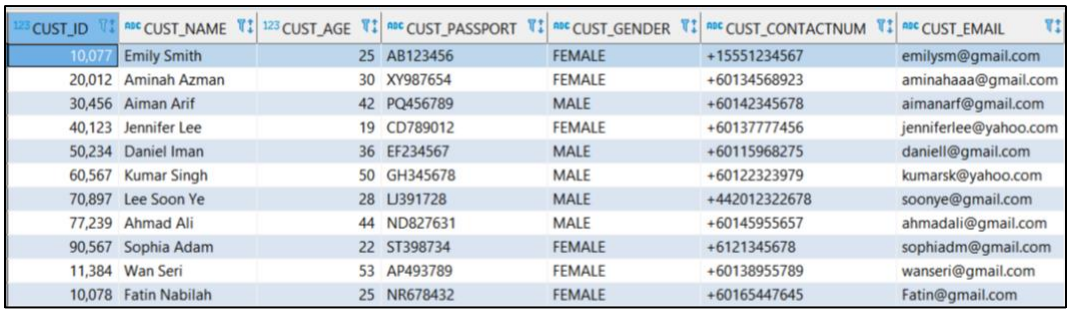
PROMO_CODE	PROMO_DESC	PROMO_DISC	PROMO_START	PROMO_END
WELCOME10	Get 10% OFF	10	2024-04-01	2024-07-31
WELCOMEAPP	Get 20% OFF	20	2024-01-01	2024-09-01
LOWFARE	Get 5% OFF	5	2024-03-01	2024-06-29
MAYDEAL	Get 30% OFF	30	2024-03-01	2024-05-31
RAHMAH50	Get 50% OFF	50	2024-05-01	2024-06-30
BIGPAY5	Get 5% OFF	5	2024-06-01	2024-07-31
HOLIDAY	Get 60% OFF	60	2024-03-01	2024-05-31
WELCOME20	Get 20% OFF	20	2024-05-01	2024-07-30
SNAP	Get 40% OFF	40	2023-10-01	2023-06-01
FLYSISWA	Get 70% OFF	70	2024-01-01	2024-12-31

- Test Data – Insert a new set of data in table Customer, Time, Payment, and Sale

i. Customer

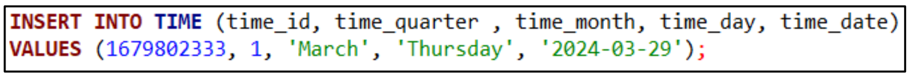


```
--test data trigger, if promo expired--
--add new cust details
INSERT INTO Customer VALUES
(10078, 'Fatin Nabilah', 25, 'NR678432', 'FEMALE', '+60165447645', 'Fatin@gmail.com');
```

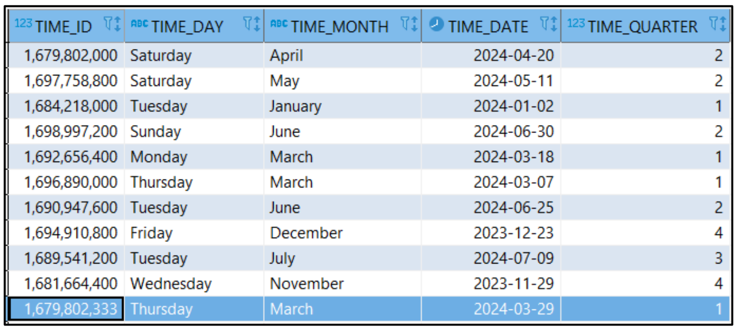


CUST_ID	CUST_NAME	CUST_AGE	CUST_PASSPORT	CUST_GENDER	CUST_CONTACTNUM	CUST_EMAIL
10,077	Emily Smith	25	AB123456	FEMALE	+15551234567	emilysm@gmail.com
20,012	Aminah Azman	30	XY987654	FEMALE	+60134568923	aminahaaa@gmail.com
30,456	Aiman Arif	42	PQ456789	MALE	+60142345678	aimanarf@gmail.com
40,123	Jennifer Lee	19	CD789012	FEMALE	+60137777456	jenniferlee@yahoo.com
50,234	Daniel Iman	36	EF234567	MALE	+60115968275	daniell@gmail.com
60,567	Kumar Singh	50	GH345678	MALE	+60122323979	kumarsk@yahoo.com
70,897	Lee Soon Ye	28	LJ391728	MALE	+442012322678	soonye@gmail.com
77,239	Ahmad Ali	44	ND827631	MALE	+60145955657	ahmadali@gmail.com
90,567	Sophia Adam	22	ST398734	FEMALE	+6121345678	sophiadm@gmail.com
11,384	Wan Seri	53	AP493789	FEMALE	+60138955789	wanseri@gmail.com
10,078	Fatin Nabilah	25	NR678432	FEMALE	+60165447645	Fatin@gmail.com

ii. Time

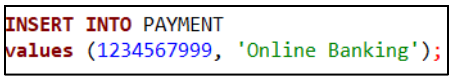


```
INSERT INTO TIME (time_id, time_quarter , time_month, time_day, time_date)
VALUES (1679802333, 1, 'March', 'Thursday', '2024-03-29');
```

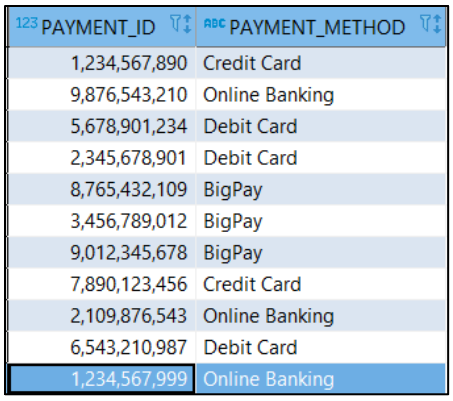


TIME_ID	TIME_DAY	TIME_MONTH	TIME_DATE	TIME_QUARTER
1,679,802,000	Saturday	April	2024-04-20	2
1,697,758,800	Saturday	May	2024-05-11	2
1,684,218,000	Tuesday	January	2024-01-02	1
1,698,997,200	Sunday	June	2024-06-30	2
1,692,656,400	Monday	March	2024-03-18	1
1,696,890,000	Thursday	March	2024-03-07	1
1,690,947,600	Tuesday	June	2024-06-25	2
1,694,910,800	Friday	December	2023-12-23	4
1,689,541,200	Tuesday	July	2024-07-09	3
1,681,664,400	Wednesday	November	2023-11-29	4
1,679,802,333	Thursday	March	2024-03-29	1

iii. Payment

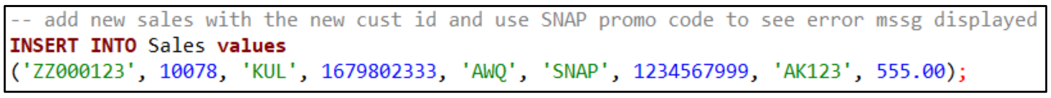


```
INSERT INTO PAYMENT
values (1234567999, 'Online Banking');
```



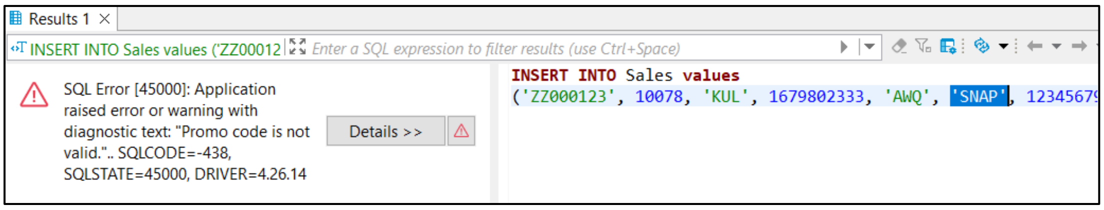
PAYMENT_ID	PAYMENT_METHOD
1,234,567,890	Credit Card
9,876,543,210	Online Banking
5,678,901,234	Debit Card
2,345,678,901	Debit Card
8,765,432,109	BigPay
3,456,789,012	BigPay
9,012,345,678	BigPay
7,890,123,456	Credit Card
2,109,876,543	Online Banking
6,543,210,987	Debit Card
1,234,567,999	Online Banking

iv. Sales



```
-- add new sales with the new cust id and use SNAP promo code to see error mssg displayed
INSERT INTO Sales values
('ZZ000123', 10078, 'KUL', 1679802333, 'AWQ', 'SNAP', 1234567999, 'AK123', 555.00);
```

- After running the command – the trigger will validate the data. Data with invalid promo code like ('SNAP') will not be added in the table. Data remain the same.



Results 1

INSERT INTO Sales values ('ZZ00012 | Enter a SQL expression to filter results (use Ctrl+Space)

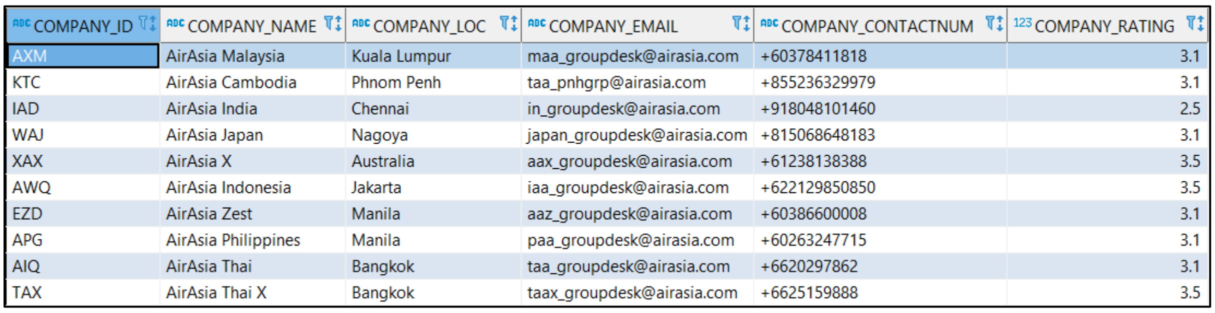
```
SQL Error [45000]: Application
raised error or warning with
diagnostic text: "Promo code is not
valid.". SQLCODE=-438,
SQLSTATE=45000, DRIVER=4.26.14
```

Details >>

```
INSERT INTO Sales values
('ZZ000123', 10078, 'KUL', 1679802333, 'AWQ', 'SNAP', 1234567
```

- New sales data with valid customer and promo code will be added and the company rating in Company table will raise by 0.1

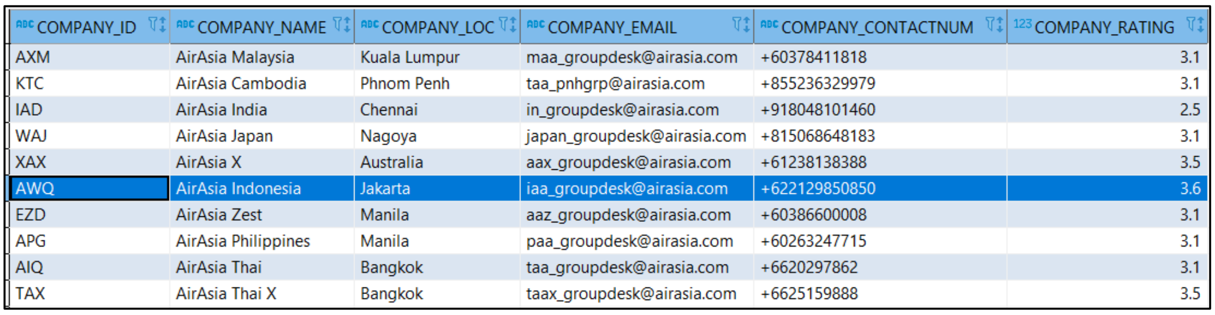
i. Company table before trigger



COMPANY_ID	COMPANY_NAME	COMPANY_LOC	COMPANY_EMAIL	COMPANY_CONTACTNUM	COMPANY_RATING
AXM	AirAsia Malaysia	Kuala Lumpur	maa_groupdesk@airasia.com	+60378411818	3.1
KTC	AirAsia Cambodia	Phnom Penh	taa_pnhgrp@airasia.com	+855236329979	3.1
IAD	AirAsia India	Chennai	in_groupdesk@airasia.com	+918048101460	2.5
WAJ	AirAsia Japan	Nagoya	japan_groupdesk@airasia.com	+815068648183	3.1
XAX	AirAsia X	Australia	aax_groupdesk@airasia.com	+61238138388	3.5
AWQ	AirAsia Indonesia	Jakarta	iaa_groupdesk@airasia.com	+622129850850	3.5
EZD	AirAsia Zest	Manila	aaz_groupdesk@airasia.com	+60386600008	3.1
APG	AirAsia Philippines	Manila	paa_groupdesk@airasia.com	+60263247715	3.1
AIQ	AirAsia Thai	Bangkok	taa_groupdesk@airasia.com	+6620297862	3.1
TAX	AirAsia Thai X	Bangkok	taax_groupdesk@airasia.com	+6625159888	3.5

ii. After trigger

```
-- add new sales data with valid customer and valid promocode,
--the company rating will raise 0.1 when receive a new sale from customer
INSERT INTO Sales values
('ZZ000123', 10078, 'KUL', 1679802333, 'AWQ', 'WELCOME10', 1234567999, 'AK123', 555.00);
```



COMPANY_ID	COMPANY_NAME	COMPANY_LOC	COMPANY_EMAIL	COMPANY_CONTACTNUM	COMPANY_RATING
AXM	AirAsia Malaysia	Kuala Lumpur	maa_groupdesk@airasia.com	+60378411818	3.1
KTC	AirAsia Cambodia	Phnom Penh	taa_pnhgrp@airasia.com	+855236329979	3.1
IAD	AirAsia India	Chennai	in_groupdesk@airasia.com	+918048101460	2.5
WAJ	AirAsia Japan	Nagoya	japan_groupdesk@airasia.com	+815068648183	3.1
XAX	AirAsia X	Australia	aax_groupdesk@airasia.com	+61238138388	3.5
AWQ	AirAsia Indonesia	Jakarta	iaa_groupdesk@airasia.com	+622129850850	3.6
EZD	AirAsia Zest	Manila	aaz_groupdesk@airasia.com	+60386600008	3.1
APG	AirAsia Philippines	Manila	paa_groupdesk@airasia.com	+60263247715	3.1
AIQ	AirAsia Thai	Bangkok	taa_groupdesk@airasia.com	+6620297862	3.1
TAX	AirAsia Thai X	Bangkok	taax_groupdesk@airasia.com	+6625159888	3.5

6.3 User-Defined Function

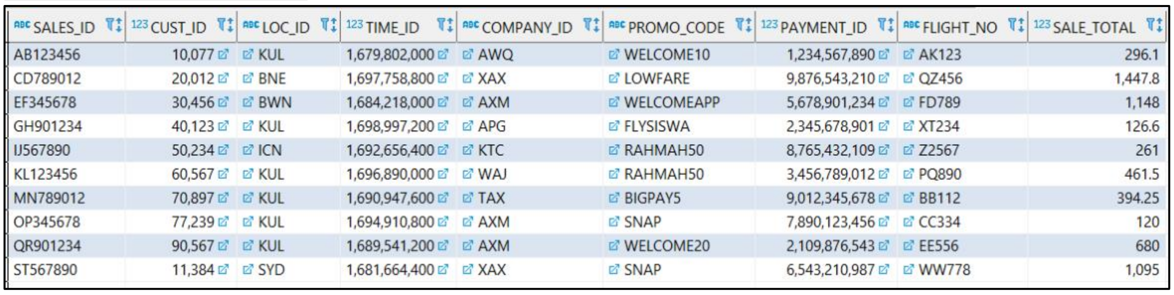
To view all sales under specific airline company (*AXM*) with promo code that less than 25% discount.

- SQL Command

```
CREATE FUNCTION ViewSaleTotal(input_id CHAR(10))
RETURNS TABLE (
    Company_id_in CHAR(10),
    Cust_id INT,
    Loc_id CHAR(10),
    Time_id BIGINT,
    Company_id CHAR(10),
    Promo_code VARCHAR(20),
    Payment_id BIGINT,
    Flight_no VARCHAR(20),
    Sale_total decimal (7,2)
)
LANGUAGE SQL
READS SQL DATA
NO EXTERNAL ACTION
DETERMINISTIC
RETURN
    SELECT input_id, s.Cust_id, s.Loc_id,
           s.Time_id, s.Company_id, s.Promo_code,
           s.Payment_id, s.Flight_no,s.Sale_total
    FROM Sales s
    JOIN Promotion p ON s.Promo_code = p.Promo_code
    WHERE s.Company_id = input_id
      AND p.Promo_disc < 25.0;
```

- Before User-Defined Function

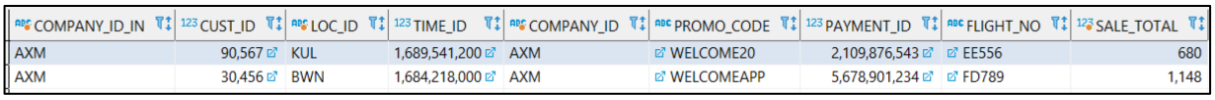
```
SELECT * FROM SALES
```



SALES_ID	CUST_ID	LOC_ID	TIME_ID	COMPANY_ID	PROMO_CODE	PAYMENT_ID	FLIGHT_NO	SALE_TOTAL
AB123456	10,077	KUL	1,679,802,000	AWQ	WELCOME10	1,234,567,890	AK123	296.1
CD789012	20,012	BNE	1,697,758,800	XAX	LOWFARE	9,876,543,210	QZ456	1,447.8
EF345678	30,456	BWN	1,684,218,000	AXM	WELCOMEAPP	5,678,901,234	FD789	1,148
GH901234	40,123	KUL	1,698,997,200	APG	FLYSISWA	2,345,678,901	XT234	126.6
IJ567890	50,234	ICN	1,692,656,400	KTC	RAHMAH50	8,765,432,109	Z2567	261
KL123456	60,567	KUL	1,696,890,000	WAJ	RAHMAH50	3,456,789,012	PQ890	461.5
MN789012	70,897	KUL	1,690,947,600	TAX	BIGPAY5	9,012,345,678	BB112	394.25
OP345678	77,239	KUL	1,694,910,800	AXM	SNAP	7,890,123,456	CC334	120
QR901234	90,567	KUL	1,689,541,200	AXM	WELCOME20	2,109,876,543	EE556	680
ST567890	11,384	SYD	1,681,664,400	XAX	SNAP	6,543,210,987	WW778	1,095

- After User-Defined Function

```
SELECT * FROM TABLE (ViewSaleTotal('AXM'))
```



COMPANY_ID_IN	CUST_ID	LOC_ID	TIME_ID	COMPANY_ID	PROMO_CODE	PAYMENT_ID	FLIGHT_NO	SALE_TOTAL
AXM	90,567	KUL	1,689,541,200	AXM	WELCOME20	2,109,876,543	EE556	680
AXM	30,456	BWN	1,684,218,000	AXM	WELCOMEAPP	5,678,901,234	FD789	1,148

7.0 Complex Query

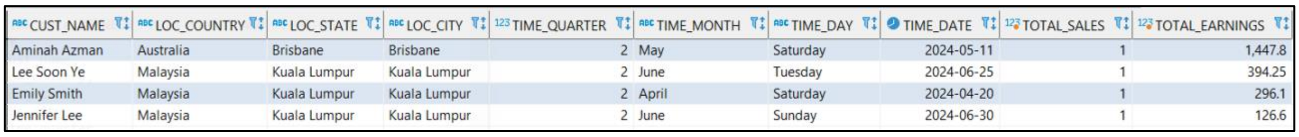
7.1 Complex Query with Joins

This query returns the customer's name, location details (country, state, city), time details (quarter, month, day, date), total number of sales, and total earnings for each customer, location, and time. The data are filtered to include only sales made between April 1, 2024, and June 30, 2024, and ordered in descending order of total earnings.

- SQL Command

```
SQL
SELECT
    c.Cust_name,
    l.Loc_country,
    l.Loc_state,
    l.Loc_city,
    t.Time_quarter,
    t.Time_month,
    t.Time_day,
    t.Time_date,
    COUNT(s.Sales_id) AS total_sales,
    SUM(s.Sale_total) AS total_earnings
FROM
    Sales s
    JOIN Customer c ON s.Cust_id = c.Cust_id
    JOIN Location l ON s.Loc_id = l.Loc_id
    JOIN Time t ON s.Time_id = t.Time_id
WHERE
    t.Time_date BETWEEN '2024-04-01' AND '2024-06-30'
GROUP BY
    c.Cust_name,
    l.Loc_country,
    l.Loc_state,
    l.Loc_city,
    t.Time_quarter,
    t.Time_month,
    t.Time_day,
    t.Time_date
ORDER BY
    total_earnings DESC;
```

- Output

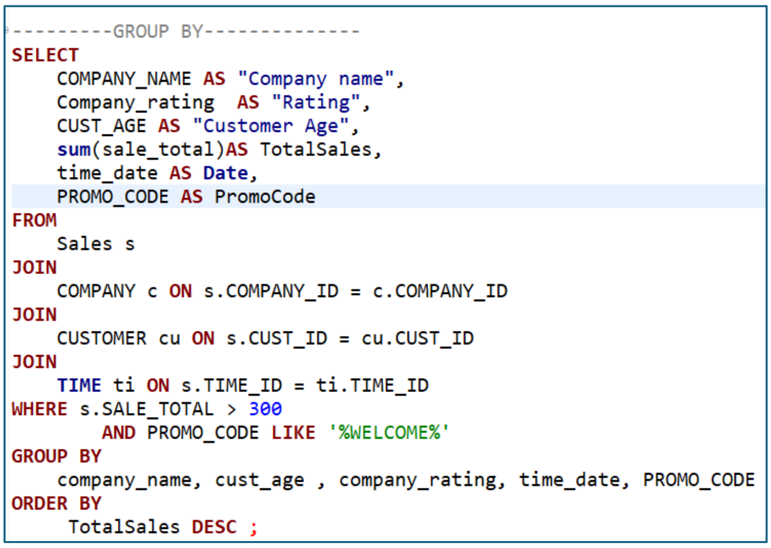


CUST_NAME	LOC_COUNTRY	LOC_STATE	LOC_CITY	TIME_QUARTER	TIME_MONTH	TIME_DAY	TIME_DATE	TOTAL_SALES	TOTAL_EARNINGS
Aminah Azman	Australia	Brisbane	Brisbane	2	May	Saturday	2024-05-11	1	1,447.8
Lee Soon Ye	Malaysia	Kuala Lumpur	Kuala Lumpur	2	June	Tuesday	2024-06-25	1	394.25
Emily Smith	Malaysia	Kuala Lumpur	Kuala Lumpur	2	April	Saturday	2024-04-20	1	296.1
Jennifer Lee	Malaysia	Kuala Lumpur	Kuala Lumpur	2	June	Sunday	2024-06-30	1	126.6

7.2 Group by/Group by Rollup/Group by Cube and having clause

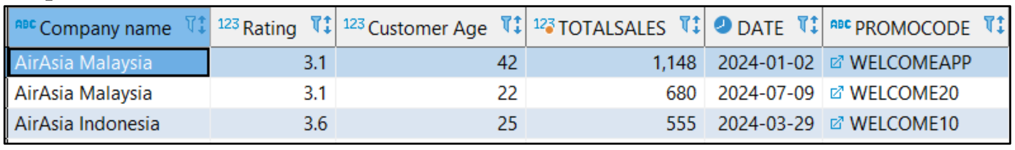
To select data of Company name, Company rating, Customer Age, Total sales, Date and Promotion Code where the sales total is more than 300 and the promo code used has word like 'Welcome' and data is group by Company Name, Cust Age, Company Rating, Date, Promotion Code, sort by the highest total sales first.

- Group By SQL Command



```
---------GROUP BY-------------
SELECT
    COMPANY_NAME AS "Company name",
    Company_rating  AS "Rating",
    CUST_AGE AS "Customer Age",
    sum(sale_total)AS TotalSales,
    time_date AS Date,
    PROMO_CODE AS PromoCode
FROM
    Sales s
JOIN
    COMPANY c ON s.COMPANY_ID = c.COMPANY_ID
JOIN
    CUSTOMER cu ON s.CUST_ID = cu.CUST_ID
JOIN
    TIME ti ON s.TIME_ID = ti.TIME_ID
WHERE s.SALE_TOTAL > 300
        AND PROMO_CODE LIKE '%WELCOME%'
GROUP BY
    company_name, cust_age , company_rating, time_date, PROMO_CODE
ORDER BY
     TotalSales DESC ;
```

- Output

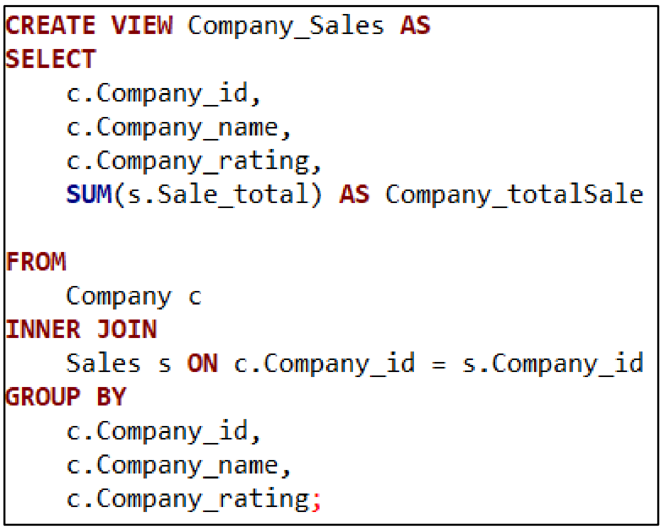


Company name	Rating	Customer Age	TOTALSALES	DATE	PROMOCODE
AirAsia Malaysia	3.1	42	1,148	2024-01-02	WELCOMEAPP
AirAsia Malaysia	3.1	22	680	2024-07-09	WELCOME20
AirAsia Indonesia	3.6	25	555	2024-03-29	WELCOME10

7.3 View

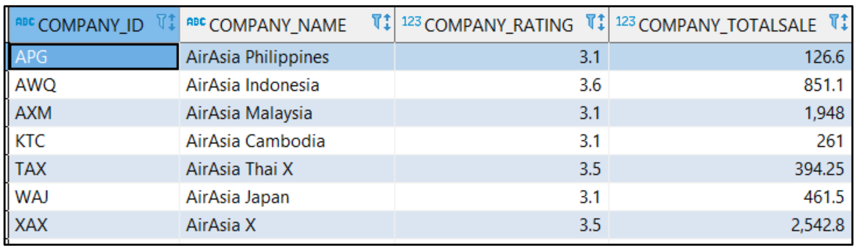
The View command create virtual table name “Company_Sales”, which combines information from two tables: Company and Sales. It includes the company ID, name, and rating from the Company table, with the total sale calculated from the Sale_total in the Sales table. The data is grouped by company ID, name, and rating to provide these aggregated values for each company.

- View SQL Command



```
CREATE VIEW Company_Sales AS
SELECT
    c.Company_id,
    c.Company_name,
    c.Company_rating,
    SUM(s.Sale_total) AS Company_totalSale

FROM
    Company c
INNER JOIN
    Sales s ON c.Company_id = s.Company_id
GROUP BY
    c.Company_id,
    c.Company_name,
    c.Company_rating;
```

- Output



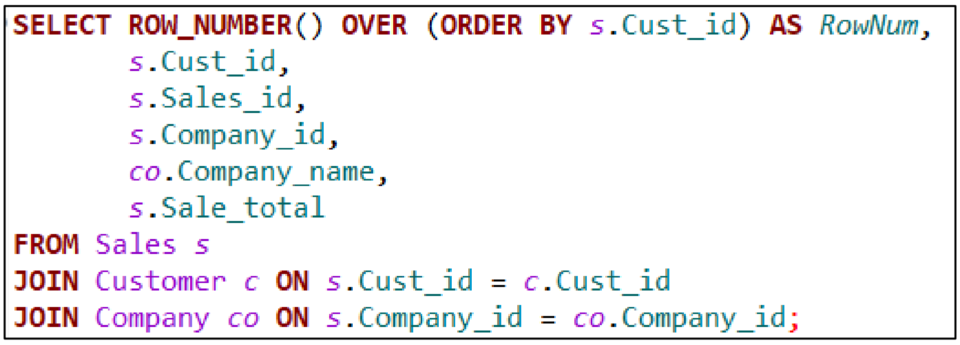
COMPANY_ID	COMPANY_NAME	COMPANY_RATING	COMPANY_TOTALSALE
APG	AirAsia Philippines	3.1	126.6
AWQ	AirAsia Indonesia	3.6	851.1
AXM	AirAsia Malaysia	3.1	1,948
KTC	AirAsia Cambodia	3.1	261
TAX	AirAsia Thai X	3.5	394.25
WAJ	AirAsia Japan	3.1	461.5
XAX	AirAsia X	3.5	2,542.8

7.4 TWO SQL not covered in lecture

(A) ROW_NUMBER Function

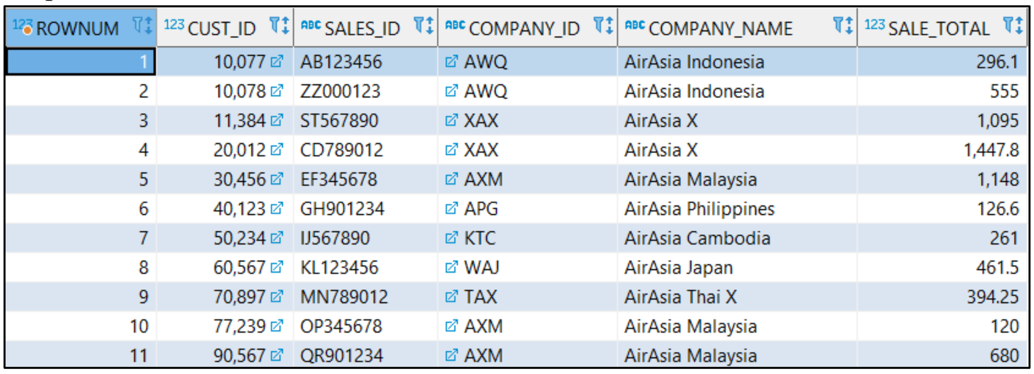
It assigns row numbers based on the ascending order of *Cust_id* from the *Sales* table and retrieves corresponding *Sales_id, Company_id, Company_Name* and *Sales_total* by joining with the *Customer* and *Company* tables.

- SQL Command



```
SELECT ROW_NUMBER() OVER (ORDER BY s.Cust_id) AS RowNum,
       s.Cust_id,
       s.Sales_id,
       s.Company_id,
       co.Company_name,
       s.Sale_total
FROM Sales s
JOIN Customer c ON s.Cust_id = c.Cust_id
JOIN Company co ON s.Company_id = co.Company_id;
```

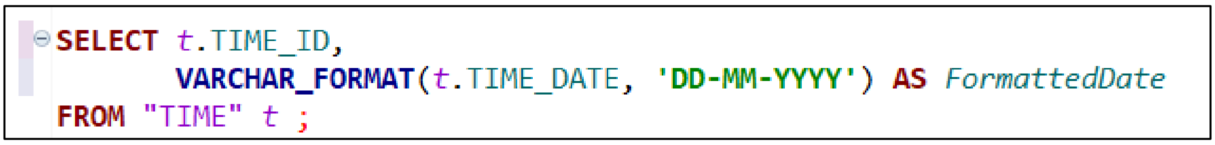
- Output



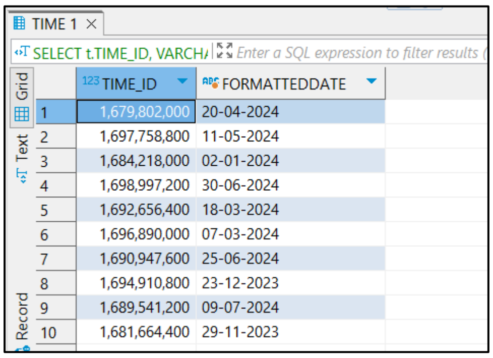
ROWNUM	CUST_ID	SALES_ID	COMPANY_ID	COMPANY_NAME	SALE_TOTAL
1	10,077	AB123456	AWQ	AirAsia Indonesia	296.1
2	10,078	ZZ000123	AWQ	AirAsia Indonesia	555
3	11,384	ST567890	XAX	AirAsia X	1,095
4	20,012	CD789012	XAX	AirAsia X	1,447.8
5	30,456	EF345678	AXM	AirAsia Malaysia	1,148
6	40,123	GH901234	APG	AirAsia Philippines	126.6
7	50,234	IJ567890	KTC	AirAsia Cambodia	261
8	60,567	KL123456	WAJ	AirAsia Japan	461.5
9	70,897	MN789012	TAX	AirAsia Thai X	394.25
10	77,239	OP345678	AXM	AirAsia Malaysia	120
11	90,567	QR901234	AXM	AirAsia Malaysia	680

(B) VARCHAR_FORMAT

The VARCHAR_FORMAT function converts the TIME_DATE column to a string in 'DD-MM-YYYY' format in SQL queries.



```
SELECT t.TIME_ID,
       VARCHAR_FORMAT(t.TIME_DATE, 'DD-MM-YYYY') AS FormattedDate
FROM "TIME" t ;
```



	TIME_ID	FORMATTEDDATE
1	1,679,802,000	20-04-2024
2	1,697,758,800	11-05-2024
3	1,684,218,000	02-01-2024
4	1,698,997,200	30-06-2024
5	1,692,656,400	18-03-2024
6	1,696,890,000	07-03-2024
7	1,690,947,600	25-06-2024
8	1,694,910,800	23-12-2023
9	1,689,541,200	09-07-2024
10	1,681,664,400	29-11-2023