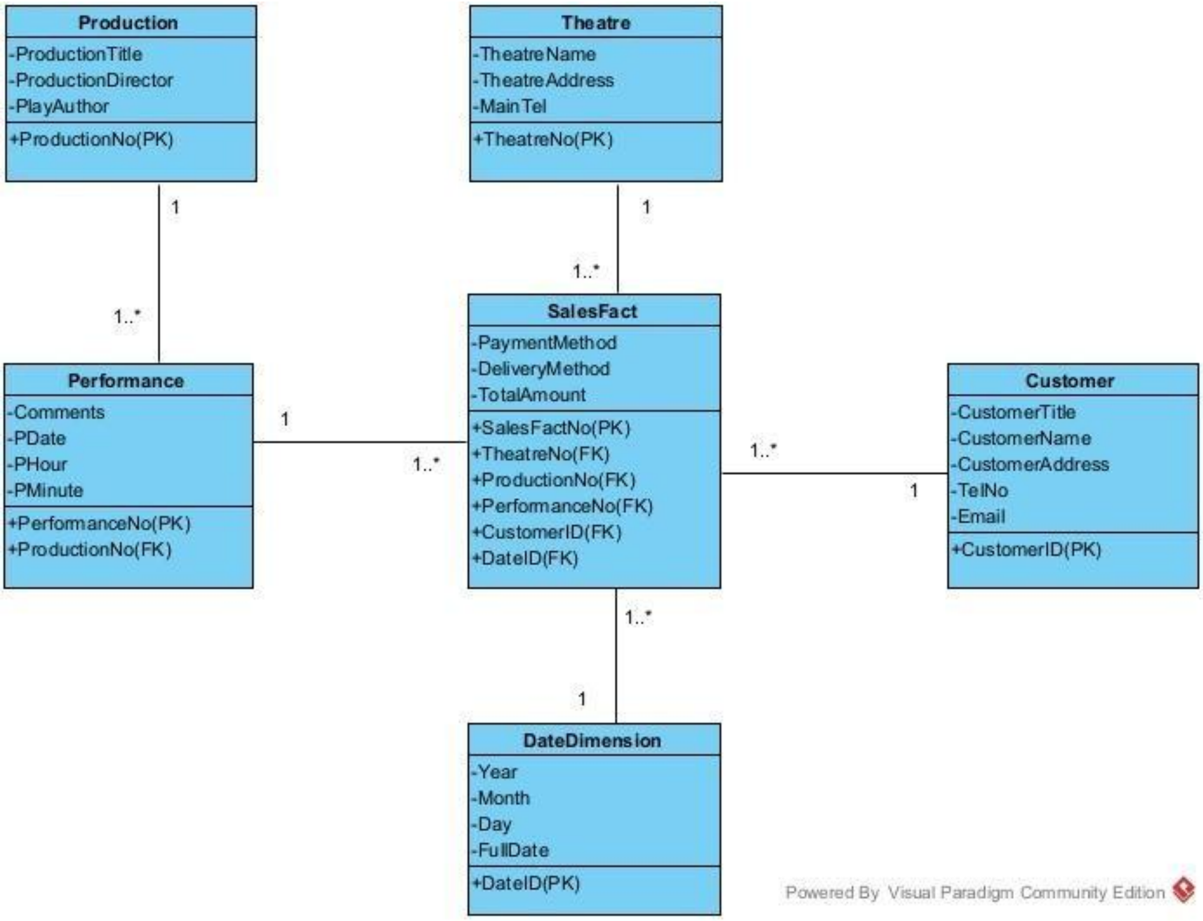
Name: Bo Kwok

Student ID:

Unit Number: 6G6Z0026_2425_9

Coursework Title: Individual Portfolio: Design and Coding

2a. Snowflake Schema for the LTC Data Mart showing all keys, relationships, attributes, and measures.



2b. Dimension selection and fact identification

The snowflake schema is created for LTC Data Mart which focuses on its ticket sales performance. The Sales Fact table is surrounded by Theatre, Customer, Date and Performance Dimension tables, while Production Dimension table does not directly link to Sales Fact table, it links to Performance Dimension table.

Compared to the ERD for London Theatres' booking system, the snowflake schema has multiple differences with it:

1. The Ticket Purchase table is transformed to a Sales Fact table. The reason is apart from ticket purchase, total value of ticket sales should be the prioritized. The Sales Fact table records all the relevant information about a ticket sale. It aims on reflecting the performance of sales. It would provide convenience for business reporting for the financial performance of LTC.
2. Staff table is excluded from the snowflake schema because its attributes does not relate to ticket sales directly. Including these information to the data mart would increase its complexity, which is not necessary.

3. Date dimension table is included in our snowflake schema, as it provides time-based analysis. It helps normalizing the date-related data, so that it would be more convenient to process sales analysis such as identifying or forecasting sales trends.
4. The Theatre table is not linked to the Performance table directly because the Sales Fact table captures all the necessary information of the performance and theatre by joining those tables by PerformanceNo and TheatreNo. It does not need any direct relationship between Theatre and Performance dimension tables.

2c. Brief design justification (max 200 words)

The design of LTC data mart aims to satisfy developing a prototype of an ORACLE data mart for ticket sales as part of a potential data warehouse for the London Theatre (LTC) Company.

A snowflake schema is created for LTC Data Mart. Sales Fact table stores key metrics and is surrounded by Theatre, Customer, Date and Performance Dimension table. Production Dimension tables does not directly linked to Sales Fact table, it links to Performance Dimension table.

A snowflake schema can store normalized data record as Production dimension table does not directly link to Sales Fact Table, it would avoid duplicating production details for each ticket sales transaction.

In this snowflake schema, surrogate keys are used for every table in order to ensure efficient table joins. The surrogate keys are primary keys that manually created, which are SalesFactNo, TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID in SalesFact, Theatre, Production, Performance, Customer and Date Dimension table respectively.

The assumptions in this Data mart are: 1. For the ticket sale value/spending, it refers to the value of money rather than number of tickets. 2. A customer purchasing several tickets for the same performance is treated as one sales transaction i.e. 1 row entry. (198 words)

3. List of Logical relations (Tables) and granularity

- Sales Fact Table:
  (PK) SalesFactNo ,
  (FK) TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID
  (Attributes) PaymentMethod, DeliveryMethod, TotalAmount

- Theatre Dimension Table:
  (PK) TheatreNo,
  (Attributes) TheatreName, TheatreAddress, MainTel

- Production Dimension Table:
  (PK) ProductionNo,
  (Attributes) ProductionTitle, ProductionDirector, PlayAuthor

- Performance Dimension Table:
  (PK) PerformanceNo,
  (FK) ProductionNo
  (Attributes) pDate, pHour, pMinute, Comments

- Customer Dimension Table:
  (PK) CustomerID,
  (Attributes) CustomerTitle, CustomerName, CustomerAddress, TelNo, Email

- Date Dimension Table:
  (PK) DateID,
  (Attributes) FullDate, Year, Month, Day

Granularity:

In date dimension table, it contains year, month and date to meet the granularity.

Expanding into a data warehouse in the future:

If the data mart aims to expand into a data warehouse, we should expand the data model with more numerical attributes and dimension tables to perform more complex data analysis.

1. Sales Fact Table

   Service Charge and Discount Amount can be added to Sales Fact Table to analyze revenue channels and measure the effectiveness of promotions. Including **Total number of tickets** into Sales Fact Table can help LTC set and allocate the optimum value of price and seating respectively.

2. Theatre table

   Seats capacity can also be added as an attribute for calculating attendance rate, in order to optimize the quantity of performance in the future. Attributes such as **Maintenance and Utilities cost** can also be added into Theatre Table to calculate operational costs.

3. Production Table

   Duration in Minutes can be added into Production Table as an attribute as it can help to schedule the performances.

4. Customer Table

   Total spendings can be added as attributes for identifying target audience, so that marketing department can organize marketing promotions to them.

5. Date Dimension Table

   Both **Fiscal Year and Fiscal Month** can be added as attributes for producing financial reports conveniently. **Quarter** can be added as attribute in order to help generate quarterly performance data analysis.

6. Adding a Staff Dimension Table

   Adding a **Staff Dimension** Table into the existing snowflake schema is feasible, with attributes of **StaffID, StaffName, StaffContactNumber, StaffEmail, StaffRole, StaffWorkingHours and StaffWagePerHour**. This help analysing individual contributions and wage expenses of staff.

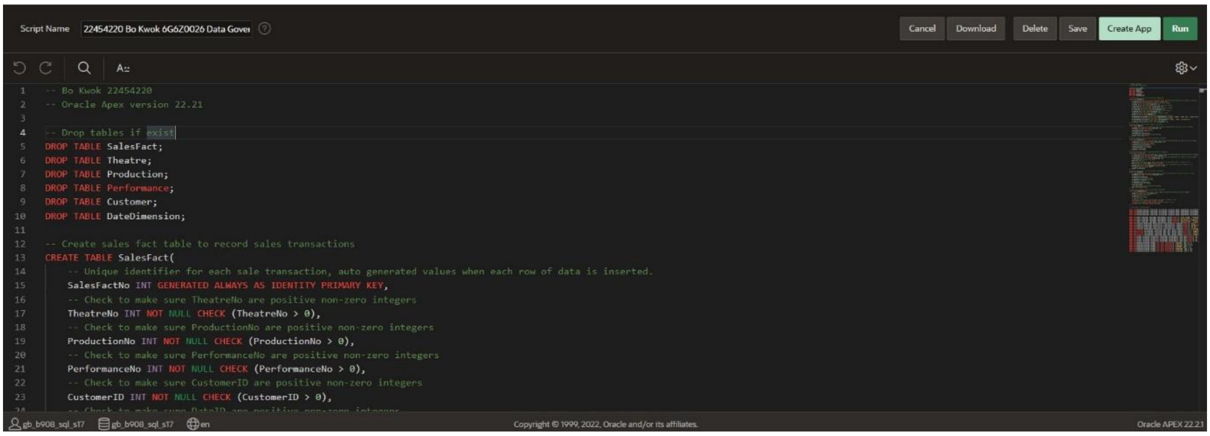
4. Screenshots of your CREATE and INSERT commands per table including the successful output of each.

SQL Code:

```
-- Bo Kwok (student ID)

-- Oracle Apex version 22.21


-- Drop tables if exist

DROP TABLE SalesFact;

DROP TABLE Theatre;

DROP TABLE Production;

DROP TABLE Performance;

DROP TABLE Customer;

DROP TABLE DateDimension;


-- Create sales fact table to record sales transactions

CREATE TABLE SalesFact(

    -- Unique identifier for each sale transaction, auto generated values when each row of data is inserted.

    SalesFactNo INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

    -- Check to make sure TheatreNo are positive non-zero integers

    TheatreNo INT NOT NULL CHECK (TheatreNo > 0),

    -- Check to make sure ProductionNo are positive non-zero integers

    ProductionNo INT NOT NULL CHECK (ProductionNo > 0),

    -- Check to make sure PerformanceNo are positive non-zero integers

    PerformanceNo INT NOT NULL CHECK (PerformanceNo > 0),

    -- Check to make sure CustomerID are positive non-zero integers

    CustomerID INT NOT NULL CHECK (CustomerID > 0),

    -- Check to make sure DateID are positive non-zero integers

    DateID INT NOT NULL CHECK (DateID > 0),
    -- Check to make sure payment method are among the listed options
    PaymentMethod VARCHAR(50) NOT NULL CHECK (PaymentMethod in ('Cash', 'PayPal', 'Debit Card', 'Credit Card')),
    -- Check to make sure delivery method are among the listed options
    DeliveryMethod VARCHAR(50) NOT NULL CHECK (DeliveryMethod IN ('Mail', 'Email', 'Box Office')),
    -- Check to make sure total amount paid is non-negative
    TotalAmount DECIMAL(10,2) NOT NULL CHECK (TotalAmount > 0)
);
-- Create theatre table to record theatre information
CREATE TABLE Theatre (
    -- Unique identifier for each theatre, check to make sure TheatreNo are positive non-zero integers
    TheatreNo INT PRIMARY KEY CHECK (TheatreNo > 0),
    -- Theatre name must not be empty
    TheatreName VARCHAR (255) NOT NULL,
    -- Theatre address
    TheatreAddress VARCHAR (255),
    -- Theatre main telephone number must be unique
    MainTel VARCHAR (20) UNIQUE
);
-- Create production table to record production information
CREATE TABLE Production (
    -- Unique identifier for each production, check to make sure ProductionNo are positive non-zero integers
    ProductionNo INT PRIMARY KEY CHECK (ProductionNo > 0),
    -- Production name must not be empty
    ProductionTitle VARCHAR(255) NOT NULL,
    -- name of production director
    ProductionDirector VARCHAR(255),
    -- name of play author
    PlayAuthor VARCHAR(255)
);
-- Create performance table to record performance information
CREATE TABLE Performance(
    -- Unique identifier for each performance, check to make sure PerformanceNo are positive non-zero integers
    PerformanceNo INT PRIMARY KEY CHECK (PerformanceNo > 0),
    -- Unique identifier for each production, check to make sure ProductionNo are positive non-zero integers
    ProductionNo INT NOT NULL CHECK (ProductionNo > 0),
    -- Performance date must not be empty
    pDate DATE NOT NULL,
    -- Performance hour must not be empty, check to make sure performance hour is between 0 to 23
    pHour INT NOT NULL CHECK (pHour between 0 and 23),
    -- Performance minute must not be empty, check to make sure performance minute is between 0 to 59
    pMinute INT NOT NULL CHECK (pMinute between 0 and 59),
    -- Comments about the performance
    Comments VARCHAR(255)
);
-- Create customer table to record customer information
CREATE TABLE Customer(
    -- Unique identifier for each customer, check to make sure CustomerID are positive non-zero integers
    CustomerID INT PRIMARY KEY CHECK (CustomerID > 0),
    -- Title of customer
    CustomerTitle VARCHAR(10),
    -- Customer name must not be empty
    CustomerName VARCHAR(255) NOT NULL,
    -- Address of customer
    CustomerAddress VARCHAR(255),
    -- Telephone number of customer
    TelNo VARCHAR(20),
    -- Customer email must be unique for validation
    Email VARCHAR(255) UNIQUE
);
-- Create date dimension table to record date information
CREATE TABLE DateDimension (
    -- Unique identifier for each date, check to make sure DateID are positive non-zero integers
    DateID INT PRIMARY KEY CHECK (DateID > 0),
    -- Full date must not be empty
    FullDate DATE NOT NULL,
    -- Year must not be empty
    Year INT NOT NULL,
    -- Month must not be empty, check to make sure month is between 1 to 12
    Month INT NOT NULL CHECK (Month Between 1 and 12),
    -- Day must not be empty, check to make sure day is between 1 to 31
    Day INT NOT NULL CHECK (Day Between 1 and 31)
);

-- Insert data to various tables

INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 1, 4, 2, 1, 'PayPal', 'Box Office', 300);
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (1, 3, 2, 1, 2, 'Debit Card', 'Mail', 200);
```

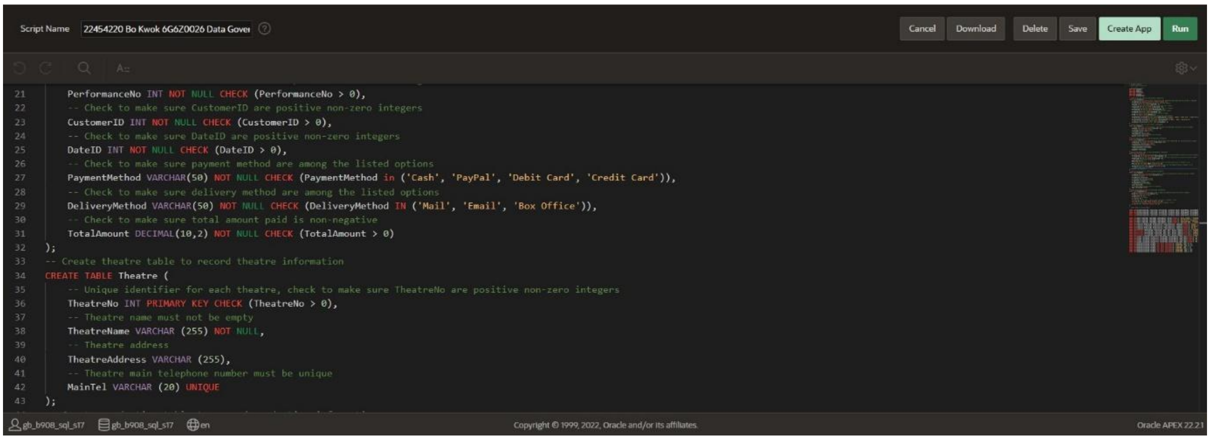
```sql
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (3, 3, 2, 2, 3, 'Credit Card', 'Email', 400);

INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 2, 3, 2, 4, 'Credit Card', 'Box Office', 400);

INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 4, 1, 2, 5, 'Debit Card', 'Email', 600);


INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (1, 'Craven Cottage', '6 Stevenage Rd', 02210252147);

INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (2, 'Emirates Stadium', '10 Emile Smith Rd', 02223584222);

INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (3, 'Old Trafford', '10 Hag Eric Street', 01614204899);

INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (4, 'Stamford Bridge', '443 Bohely Street', 01249467897);

INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (5, 'White Hart Lane', '1 Trophless Road', 01245671432);

INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (1, 'Titanic', 'James Cameron', 'James Cameron');

INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (2, 'Avengers: Endgame', 'Anthony Russo', 'Stan Lee');

INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (3, 'The Matrix', 'The Wachowskis', 'The Wachowskis');

INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (4, 'Top Gun', 'Tony Scott', 'Jim Cash');

INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (5, 'First Blood', 'Ted Kotcheff', 'David Morrell');

INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (1, 4, '24-JAN-2024', 3, 15, 'Romantic film');

INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (2, 3, '4-MAR-2024', 3, 1, 'Superhero film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (3, 2, '20-MAY-2024', 1, 49, 'Action Drama film');

INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (4, 1, '24-JUN-2024', 3, 15, 'Science fiction action film');

INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (5, 5, '24-DEC-2024', 3, 15, 'Action film');

INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (1, 'Ms', 'Erling Haaland', '6 Gabriel Road', '01152385414', 'StayHumble@gmail.com');

INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (2, 'Mr', 'Christiano Ronaldo', '7 Trafford Road', '01552001469', 'Ronaldosiu@gmail.com');

INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (3, 'Mr', 'Jose Mourinho', '1 Porto St', '01146521345', 'TheSpecialOne@hotmail.com');

INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (4, 'Ms', 'Mikel Arteta', '12 Hornsey Rd', '07998458712', 'MikelArteta@hotmail.com');

INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (5, 'Ms', 'Eric Ten Hag', '14 Manchester Rd', '03321456987', 'WeNeverLose@gmail.com');

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (1, '1-DEC-2024', 2024, 12, 1);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (2, '2-DEC-2024', 2024, 12, 2);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (3, '24-NOV-2024', 2024, 11, 24);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (4, '7-FEB-2024', 2024, 2, 7);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (5, '6-APR-2024', 2024, 4, 6);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (6, '16-MAY-2024', 2024, 5, 16);

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (7, '19-AUG-2024', 2024, 8, 19);
```

INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (8, '23-SEP-2024', 2024, 9, 23);



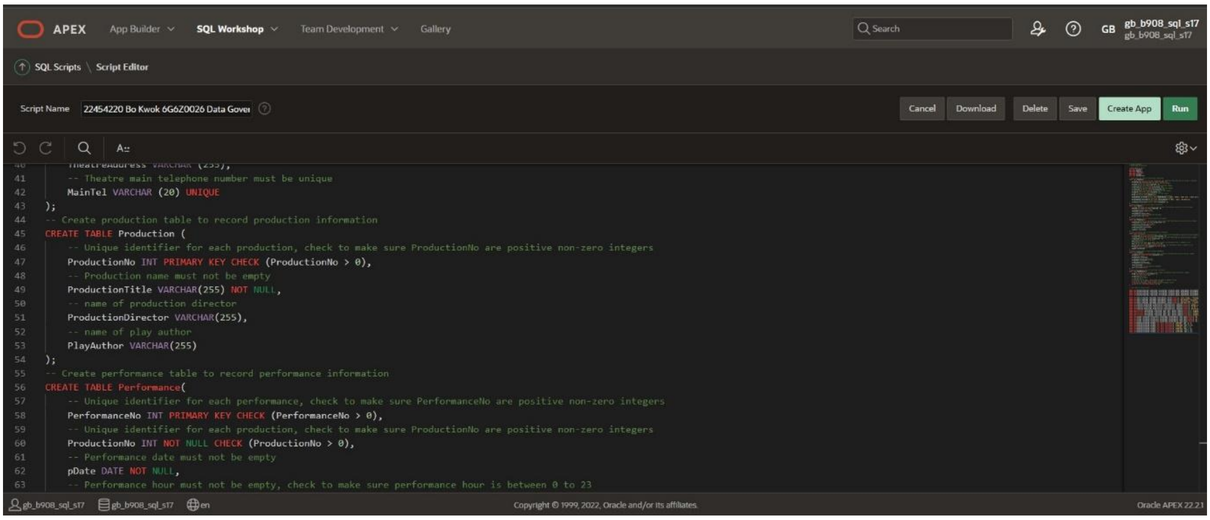
Script Name: 22454220 Bo Kwok 6G6Z0026 Data Gover

```sql
-- Bo Kwok 22454220
-- Oracle Apex version 22.21

-- Drop tables if exist
DROP TABLE SalesFact;
DROP TABLE Theatre;
DROP TABLE Production;
DROP TABLE Performance;
DROP TABLE Customer;
DROP TABLE DateDimension;

-- Create sales fact table to record sales transactions
CREATE TABLE SalesFact(
    -- Unique identifier for each sale transaction, auto generated values when each row of data is inserted.
    SalesFactNo INT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    -- Check to make sure TheatreNo are positive non-zero integers
    TheatreNo INT NOT NULL CHECK (TheatreNo > 0),
    -- Check to make sure ProductionNo are positive non-zero integers
    ProductionNo INT NOT NULL CHECK (ProductionNo > 0),
    -- Check to make sure PerformanceNo are positive non-zero integers
    PerformanceNo INT NOT NULL CHECK (PerformanceNo > 0),
    -- Check to make sure CustomerID are positive non-zero integers
    CustomerID INT NOT NULL CHECK (CustomerID > 0),
```



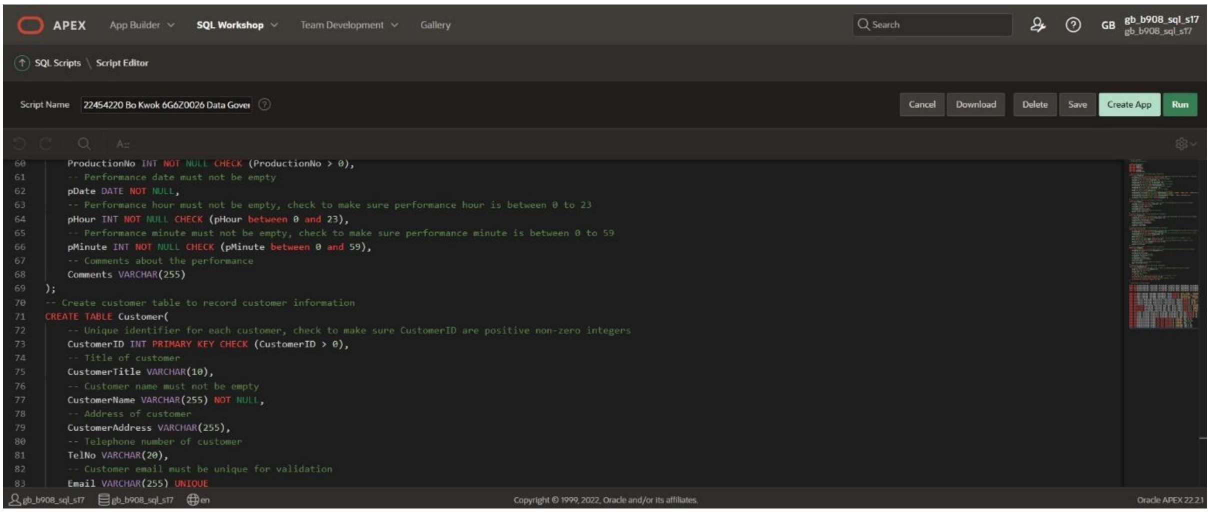
Script Name: 22454220 Bo Kwok 6G6Z0026 Data Gover

```sql
    PerformanceNo INT NOT NULL CHECK (PerformanceNo > 0),
    -- Check to make sure CustomerID are positive non-zero integers
    CustomerID INT NOT NULL CHECK (CustomerID > 0),
    -- Check to make sure DateID are positive non-zero integers
    DateID INT NOT NULL CHECK (DateID > 0),
    -- Check to make sure payment method are among the listed options
    PaymentMethod VARCHAR(50) NOT NULL CHECK (PaymentMethod in ('Cash', 'PayPal', 'Debit Card', 'Credit Card')),
    -- Check to make sure delivery method are among the listed options
    DeliveryMethod VARCHAR(50) NOT NULL CHECK (DeliveryMethod IN ('Mail', 'Email', 'Box Office')),
    -- Check to make sure total amount paid is non-negative
    TotalAmount DECIMAL(10,2) NOT NULL CHECK (TotalAmount > 0)
);
-- Create theatre table to record theatre information
CREATE TABLE Theatre (
    -- Unique identifier for each theatre, check to make sure TheatreNo are positive non-zero integers
    TheatreNo INT PRIMARY KEY CHECK (TheatreNo > 0),
    -- Theatre name must not be empty
    TheatreName VARCHAR (255) NOT NULL,
    -- Theatre address
    TheatreAddress VARCHAR (255),
    -- Theatre main telephone number must be unique
    MainTel VARCHAR (20) UNIQUE
);
```



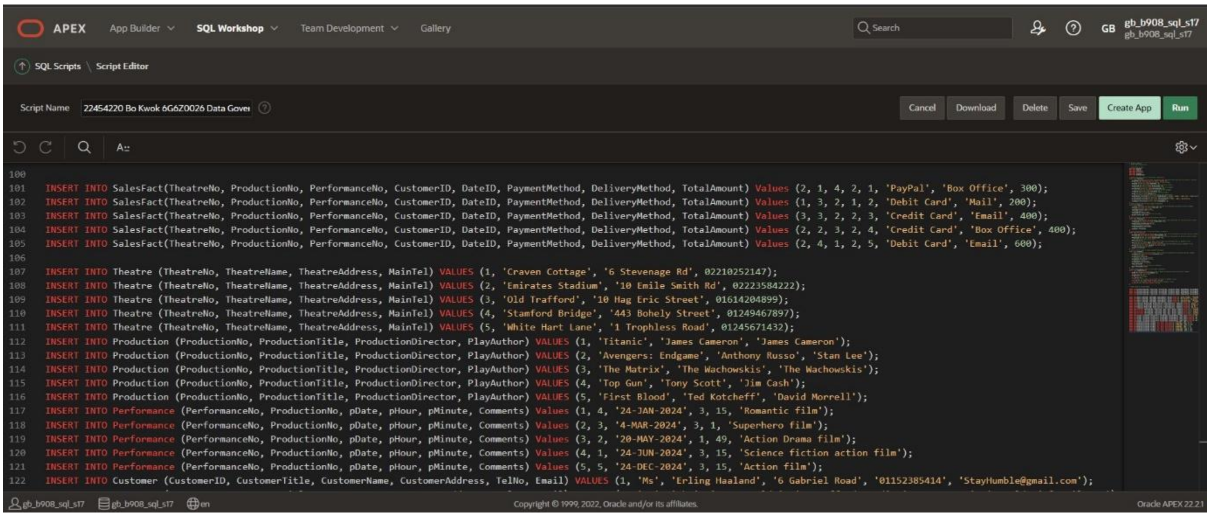
Script Name: 22454220 Bo Kwok 6G6Z0026 Data Gover

```sql
    -- Theatre main telephone number must be unique
    MainTel VARCHAR (20) UNIQUE
);
-- Create production table to record production information
CREATE TABLE Production (
    -- Unique identifier for each production, check to make sure ProductionNo are positive non-zero integers
    ProductionNo INT PRIMARY KEY CHECK (ProductionNo > 0),
    -- Production name must not be empty
    ProductionTitle VARCHAR(255) NOT NULL,
    -- name of production director
    ProductionDirector VARCHAR(255),
    -- name of play author
    PlayAuthor VARCHAR(255)
);
-- Create performance table to record performance information
CREATE TABLE Performance(
    -- Unique identifier for each performance, check to make sure PerformanceNo are positive non-zero integers
    PerformanceNo INT PRIMARY KEY CHECK (PerformanceNo > 0),
    -- Unique identifier for each production, check to make sure ProductionNo are positive non-zero integers
    ProductionNo INT NOT NULL CHECK (ProductionNo > 0),
    -- Performance date must not be empty
    pDate DATE NOT NULL,
    -- Performance hour must not be empty, check to make sure performance hour is between 0 to 23
```



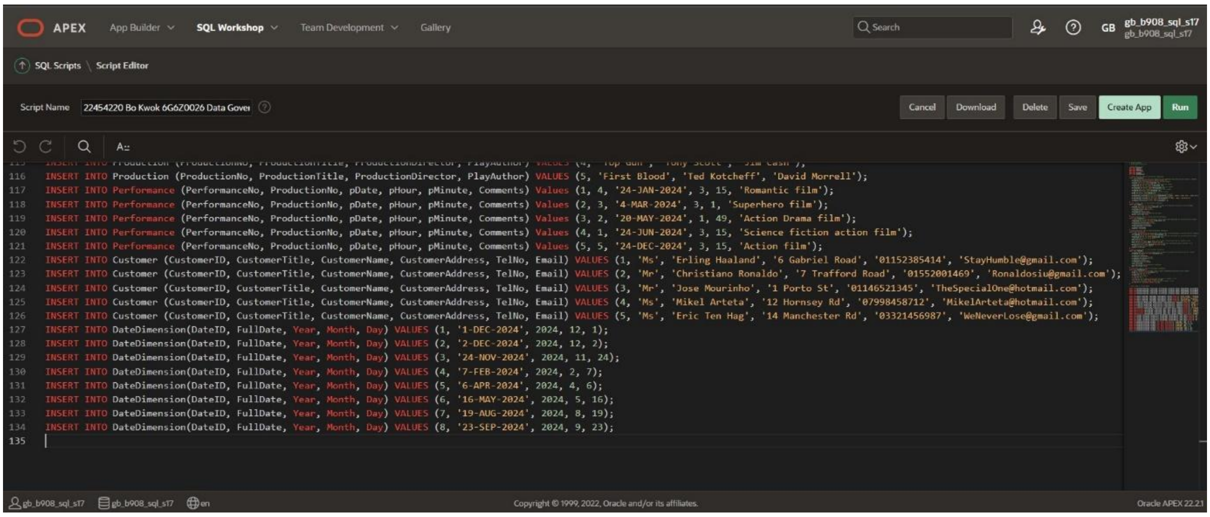
Script Name 22454220 Bo Kwok 6G6Z0026 Data Gover

```sql
    ProductionNo INT NOT NULL CHECK (ProductionNo > 0),
    -- Performance date must not be empty
    pDate DATE NOT NULL,
    -- Performance hour must not be empty, check to make sure performance hour is between 0 to 23
    pHour INT NOT NULL CHECK (pHour between 0 and 23),
    -- Performance minute must not be empty, check to make sure performance minute is between 0 to 59
    pMinute INT NOT NULL CHECK (pMinute between 0 and 59),
    -- Comments about the performance
    Comments VARCHAR(255)
);
-- Create customer table to record customer information
CREATE TABLE Customer(
    -- Unique identifier for each customer, check to make sure CustomerID are positive non-zero integers
    CustomerID INT PRIMARY KEY CHECK (CustomerID > 0),
    -- Title of customer
    CustomerTitle VARCHAR(10),
    -- Customer name must not be empty
    CustomerName VARCHAR(255) NOT NULL,
    -- Address of customer
    CustomerAddress VARCHAR(255),
    -- Telephone number of customer
    TelNo VARCHAR(20),
    -- Customer email must be unique for validation
    Email VARCHAR(255) UNIQUE
```



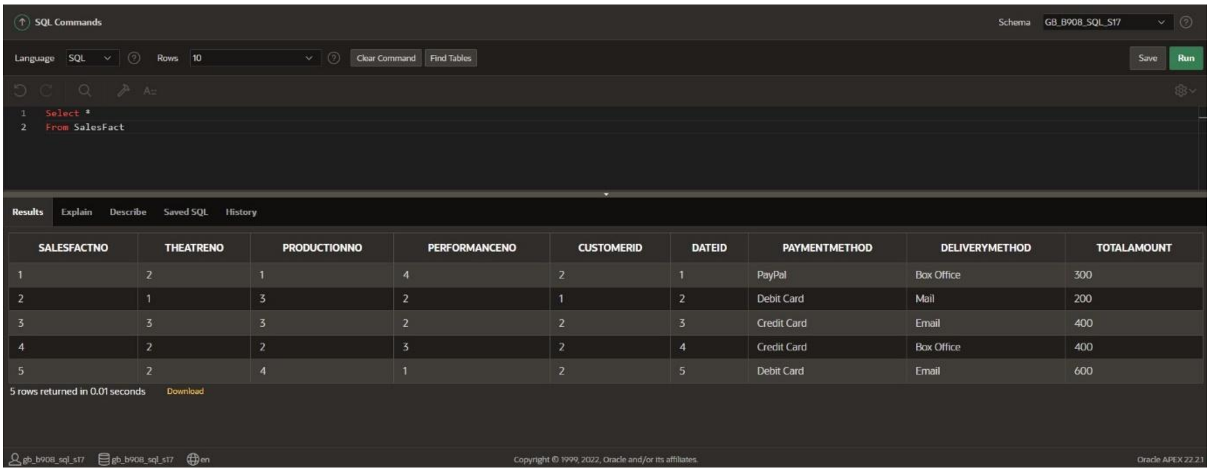
Script Name 22454220 Bo Kwok 6G6Z0026 Data Gover

```sql

INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 1, 4, 2, 1, 'PayPal', 'Box Office', 300);
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (1, 3, 2, 1, 2, 'Debit Card', 'Mail', 200);
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (3, 3, 2, 2, 3, 'Credit Card', 'Email', 400);
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 2, 3, 2, 4, 'Credit Card', 'Box Office', 400);
INSERT INTO SalesFact(TheatreNo, ProductionNo, PerformanceNo, CustomerID, DateID, PaymentMethod, DeliveryMethod, TotalAmount) Values (2, 4, 1, 2, 5, 'Debit Card', 'Email', 600);

INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (1, 'Craven Cottage', '6 Stevenage Rd', 02210252147);
INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (2, 'Emirates Stadium', '10 Emile Smith Rd', 02223584222);
INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (3, 'Old Trafford', '10 Hag Eric Street', 01614204899);
INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (4, 'Stamford Bridge', '443 Bohely Street', 01249467897);
INSERT INTO Theatre (TheatreNo, TheatreName, TheatreAddress, MainTel) VALUES (5, 'White Hart Lane', '1 Trophless Road', 01245671432);
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (1, 'Titanic', 'James Cameron', 'James Cameron');
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (2, 'Avengers: Endgame', 'Anthony Russo', 'Stan Lee');
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (3, 'The Matrix', 'The Wachowskis', 'The Wachowskis');
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (4, 'Top Gun', 'Tony Scott', 'Jim Cash');
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (5, 'First Blood', 'Ted Kotcheff', 'David Morrell');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (1, 4, '24-JAN-2024', 3, 15, 'Romantic film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (2, 3, '4-MAR-2024', 3, 1, 'Superhero film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (3, 2, '20-MAY-2024', 1, 49, 'Action Drama film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (4, 1, '24-JUN-2024', 3, 15, 'Science fiction action film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (5, 5, '24-DEC-2024', 3, 15, 'Action film');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (1, 'Ms', 'Erling Haaland', '6 Gabriel Road', '01152385414', 'StayHumble@gmail.com');
```



Script Name 22454220 Bo Kwok 6G6Z0026 Data Gover

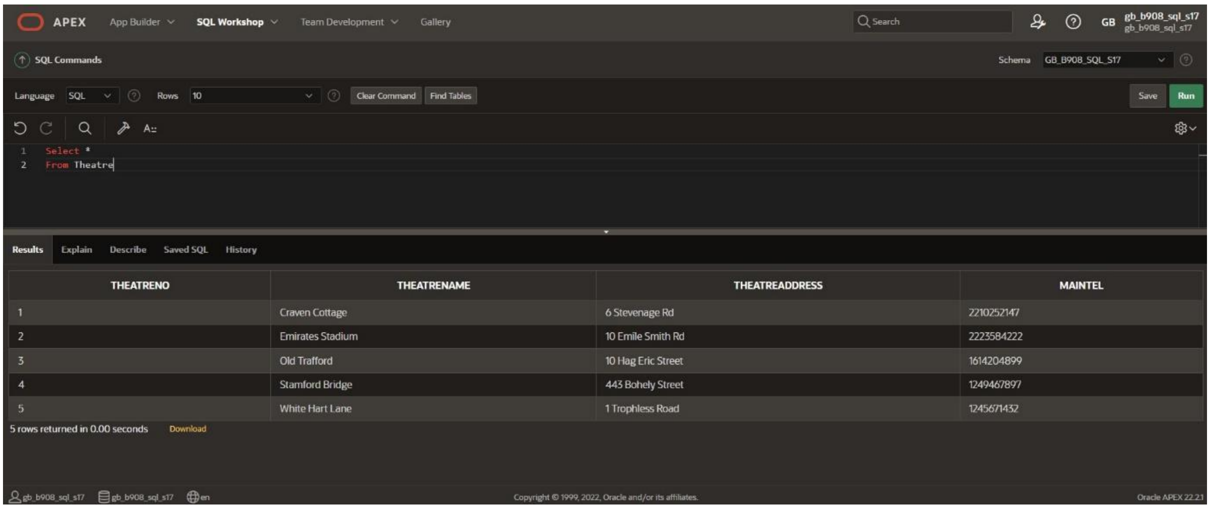
```sql
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (4, 'Top Gun', 'Tony Scott', 'Jim Cash');
INSERT INTO Production (ProductionNo, ProductionTitle, ProductionDirector, PlayAuthor) VALUES (5, 'First Blood', 'Ted Kotcheff', 'David Morrell');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (1, 4, '24-JAN-2024', 3, 15, 'Romantic film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (2, 3, '4-MAR-2024', 3, 1, 'Superhero film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (3, 2, '20-MAY-2024', 1, 49, 'Action Drama film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (4, 1, '24-JUN-2024', 3, 15, 'Science fiction action film');
INSERT INTO Performance (PerformanceNo, ProductionNo, pDate, pHour, pMinute, Comments) Values (5, 5, '24-DEC-2024', 3, 15, 'Action film');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (1, 'Ms', 'Erling Haaland', '6 Gabriel Road', '01152385414', 'StayHumble@gmail.com');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (2, 'Mr', 'Christiano Ronaldo', '7 Trafford Road', '01552001469', 'Ronaldosiu@gmail.com');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (3, 'Mr', 'Jose Mourinho', '1 Porto St', '01146521345', 'TheSpecialOne@hotmail.com');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (4, 'Ms', 'Mikel Arteta', '12 Hornsey Rd', '07998458712', 'MikelArteta@hotmail.com');
INSERT INTO Customer (CustomerID, CustomerTitle, CustomerName, CustomerAddress, TelNo, Email) VALUES (5, 'Ms', 'Eric Ten Hag', '14 Manchester Rd', '03321456987', 'WeNeverLose@gmail.com');
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (1, '1-DEC-2024', 2024, 12, 1);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (2, '2-DEC-2024', 2024, 12, 2);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (3, '24-NOV-2024', 2024, 11, 24);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (4, '7-FEB-2024', 2024, 2, 7);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (5, '6-APR-2024', 2024, 4, 6);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (6, '16-MAY-2024', 2024, 5, 16);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (7, '19-AUG-2024', 2024, 8, 19);
INSERT INTO DateDimension(DateID, FullDate, Year, Month, Day) VALUES (8, '23-SEP-2024', 2024, 9, 23);

```



```
Select *
From SalesFact
```

SALESFACTNO	THEATRENO	PRODUCTIONNO	PERFORMANCENO	CUSTOMERID	DATEID	PAYMENTMETHOD	DELIVERYMETHOD	TOTALAMOUNT
1	2	1	4	2	1	PayPal	Box Office	300
2	1	3	2	1	2	Debit Card	Mail	200
3	3	3	2	2	3	Credit Card	Email	400
4	2	2	3	2	4	Credit Card	Box Office	400
5	2	4	1	2	5	Debit Card	Email	600

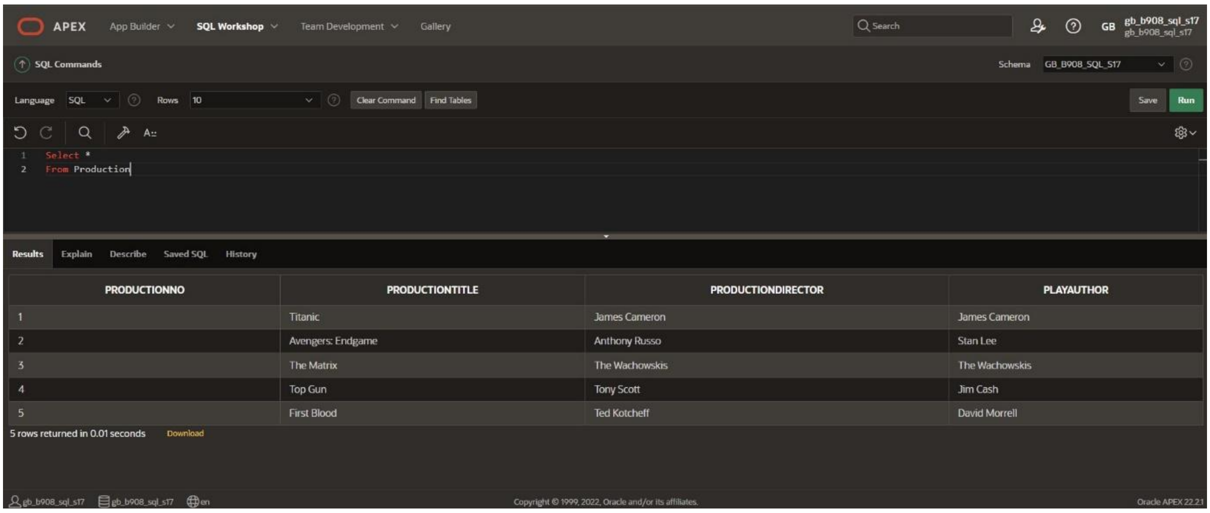
5 rows returned in 0.01 seconds



```
Select *
From Theatre
```

THEATRENO	THEATRENAME	THEATREADDRESS	MAINTEL
1	Craven Cottage	6 Stevenage Rd	2210252147
2	Emirates Stadium	10 Emile Smith Rd	2223584222
3	Old Trafford	10 Hag Eric Street	1614204899
4	Stamford Bridge	443 Bohely Street	1249467897
5	White Hart Lane	1 Trophless Road	1245671432

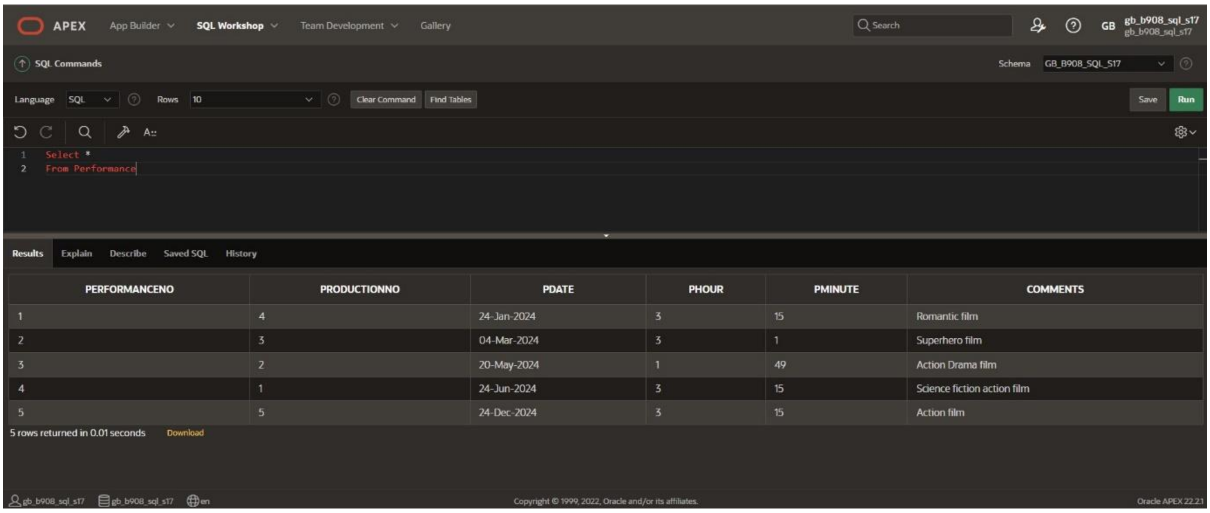
5 rows returned in 0.00 seconds



```
Select *
From Production
```

PRODUCTIONNO	PRODUCTIONTITLE	PRODUCTIONDIRECTOR	PLAYAUTHOR
1	Titanic	James Cameron	James Cameron
2	Avengers: Endgame	Anthony Russo	Stan Lee
3	The Matrix	The Wachowskis	The Wachowskis
4	Top Gun	Tony Scott	Jim Cash
5	First Blood	Ted Kotcheff	David Morrell

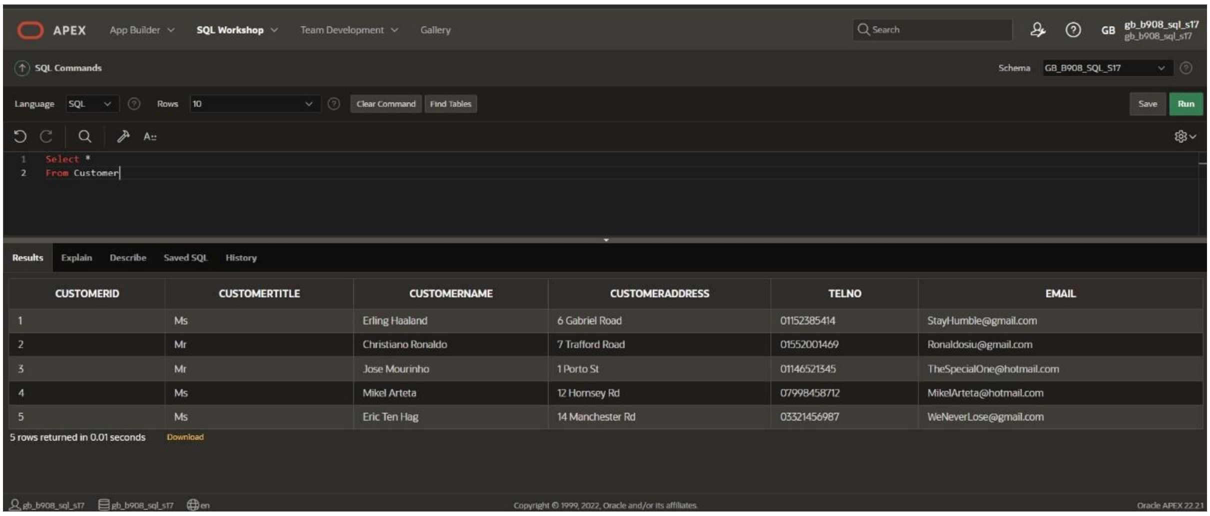
5 rows returned in 0.01 seconds



```
Select *
From Performance
```

PERFORMANCENO	PRODUCTIONNO	PDATE	PHOUR	PMINUTE	COMMENTS
1	4	24-Jan-2024	3	15	Romantic film
2	3	04-Mar-2024	3	1	Superhero film
3	2	20-May-2024	1	49	Action Drama film
4	1	24-Jun-2024	3	15	Science fiction action film
5	5	24-Dec-2024	3	15	Action film

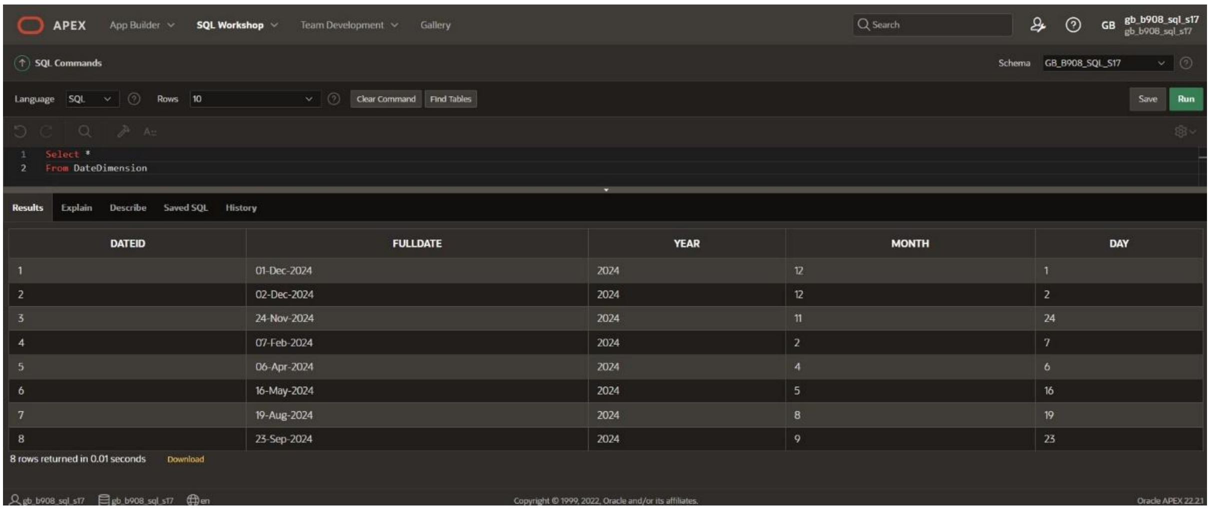
5 rows returned in 0.01 seconds



```
Select *
From Customer
```

CUSTOMERID	CUSTOMERTITLE	CUSTOMERNAME	CUSTOMERADDRESS	TELNO	EMAIL
1	Ms	Erling Haaland	6 Gabriel Road	01152385414	StayHumble@gmail.com
2	Mr	Christiano Ronaldo	7 Trafford Road	01552001469	Ronaldosiu@gmail.com
3	Mr	Jose Mourinho	1 Porto St	01146521345	TheSpecialOne@hotmail.com
4	Ms	Mikel Arteta	12 Hornsey Rd	07998458712	MikelArteta@hotmail.com
5	Ms	Eric Ten Hag	14 Manchester Rd	03321456987	WeNeverLose@gmail.com

5 rows returned in 0.01 seconds



```
Select *
From DateDimension
```

DATEID	FULLDATE	YEAR	MONTH	DAY
1	01-Dec-2024	2024	12	1
2	02-Dec-2024	2024	12	2
3	24-Nov-2024	2024	11	24
4	07-Feb-2024	2024	2	7
5	06-Apr-2024	2024	4	6
6	16-May-2024	2024	5	16
7	19-Aug-2024	2024	8	19
8	23-Sep-2024	2024	9	23

8 rows returned in 0.01 seconds

5. Screenshots of the SQL code of your **all four** analytical queries, a brief description of what the aim of each query is, a justification and screen shots showing successful execution of your queries.

1. Yearly total sale for each theatre.

SQL Code:

-- Select theatre name, extract the year from performance date in performance table and name it as year, sum of sales total amount and name it as TotalSales

SELECT the.TheatreName,EXTRACT(Year FROM per.pDate) AS Year, SUM(sf.TotalAmount) AS TotalSales

-- from SalesFact table with sf in short form

FROM SalesFact sf

-- Join Performance table by PerformanceNo

JOIN Performance per on sf.PerformanceNo = per.PerformanceNo

-- Join Theatre table by TheatreNo

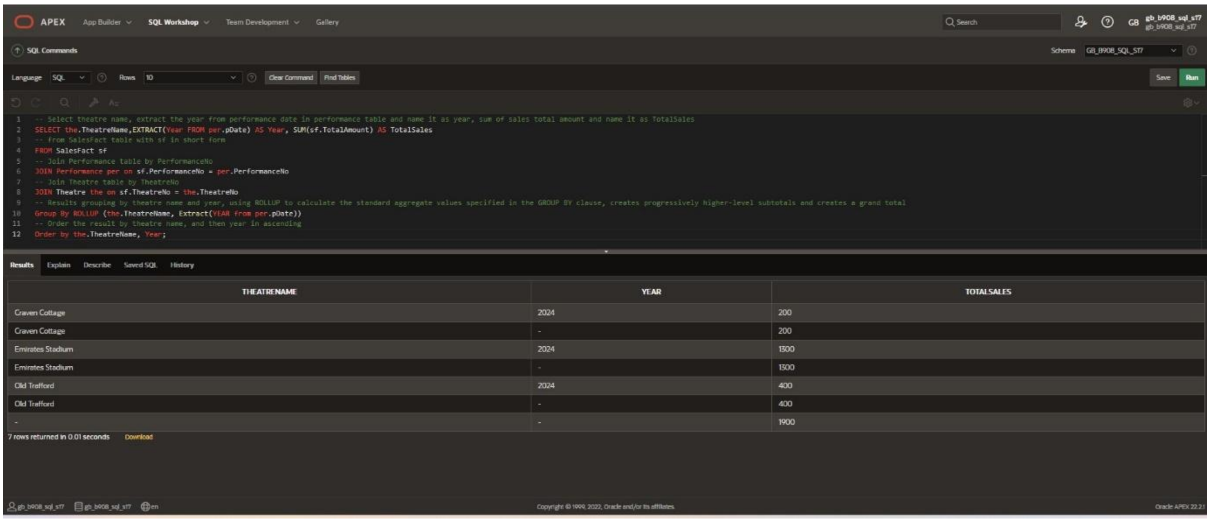
JOIN Theatre the on sf.TheatreNo = the.TheatreNo

-- Results grouping by theatre name and year, using ROLLUP to calculate the standard aggregate values specified in the GROUP BY clause, creates progressively higher-level subtotals and creates a grand total

Group By ROLLUP (the.TheatreName, Extract(YEAR from per.pDate))

-- Order the result by theatre name, and then year in ascending

Order by the.TheatreName, Year;



Schema: GB_B908_SQL_S17

Language: SQL   Rows: 10

```
-- Select theatre name, extract the year from performance date in performance table and name it as year, sum of sales total amount and name it as TotalSales
SELECT the.TheatreName,EXTRACT(Year FROM per.pDate) AS Year, SUM(sf.TotalAmount) AS TotalSales
-- from SalesFact table with sf in short form
FROM SalesFact sf
-- Join Performance table by PerformanceNo
JOIN Performance per on sf.PerformanceNo = per.PerformanceNo
-- Join Theatre table by TheatreNo
JOIN Theatre the on sf.TheatreNo = the.TheatreNo
-- Results grouping by theatre name and year, using ROLLUP to calculate the standard aggregate values specified in the GROUP BY clause, creates progressively higher-level subtotals and creates a grand total
Group By ROLLUP (the.TheatreName, Extract(YEAR from per.pDate))
-- Order the result by theatre name, and then year in ascending
Order by the.TheatreName, Year;
```

THEATRENAME	YEAR	TOTALSALES
Craven Cottage	2024	200
Craven Cottage	-	200
Emirates Stadium	2024	1300
Emirates Stadium	-	1300
Old Trafford	2024	400
Old Trafford	-	400
-	-	1900

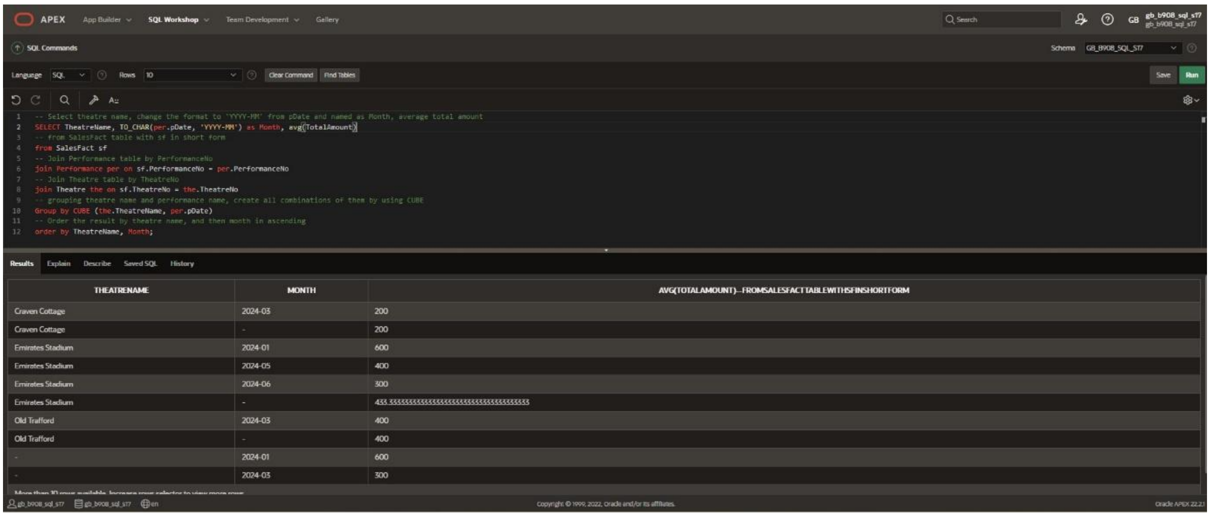
7 rows returned in 0.01 seconds

This query used ROLLUP function to calculate the yearly total sale for each theatre, it helps evaluating the performance of theatres so that it can identify which theatre generates high revenue. It could also help allocating resources and comparing performances between theatres.

2. Average sales across each month for each theatre.

SQL Code:

```
-- select theatre name, change the format to ‘YYYY-MM’ from pDate and named as month, average total amount

SELECT TheatreName, TO_CHAR(per.pDate, 'YYYY-MM') as Month, avg(TotalAmount)

-- from SalesFact table with sf in short form

from SalesFact sf

-- Join Performance table by PerformanceNo

join Performance per on sf.PerformanceNo = per.PerformanceNo

-- Join Theatre table by TheatreNo

join Theatre the on sf.TheatreNo = the.TheatreNo

-- grouping theatre name and performance name, create all combinations of them by using CUBE

Group by CUBE (the.TheatreName, per.pDate)

-- Order the result by theatre name, and then month in ascending

order by TheatreName, Month;
```



```sql
-- Select theatre name, change the format to 'YYYY-MM' from pDate and named as Month, average total amount
SELECT TheatreName, TO_CHAR(per.pDate, 'YYYY-MM') as Month, avg(TotalAmount)
-- from SalesFact table with sf in short form
from SalesFact sf
-- Join Performance table by PerformanceNo
join Performance per on sf.PerformanceNo = per.PerformanceNo
-- Join Theatre table by TheatreNo
join Theatre the on sf.TheatreNo = the.TheatreNo
-- grouping theatre name and performance name, create all combinations of them by using CUBE
Group by CUBE (the.TheatreName, per.pDate)
-- Order the result by theatre name, and then month in ascending
order by TheatreName, Month;
```

THEATRENAME	MONTH	AVG(TOTALAMOUNT)--FROMSALESFACTTABLEWITHSFINSHORTFORM
Craven Cottage	2024-03	200
Craven Cottage	-	200
Emirates Stadium	2024-01	600
Emirates Stadium	2024-05	400
Emirates Stadium	2024-06	300
Emirates Stadium	-	433.33333333333333333333333333333333333
Old Trafford	2024-03	400
Old Trafford	-	400
-	2024-01	600
-	2024-03	300

More than 10 rows available. Increase rows selector to view more rows.

gb_b908_sql_s17 gb_b908_sql_s17 en Copyright © 1999, 2022, Oracle and/or its affiliates. Oracle APEX 22.2.1

This query used CUBE function to display average sales across each month for each theatre. It helps identifying which months are high demand so that LTC could plan to allocate resources well for each month. It can also schedule productions with high demand in peak months.

3. All clients who visited LTC theatres in at least 4 different months in a year.

SQL Code:

-- select distinct customerID and their name

select distinct CustomerID, CustomerName

-- subquery to calculate visit year, month and rankings for customers

from (

-- select customerID and name, extract year from performance date and name as visited Year, format the performance date and extract Month as string,

-- Using DENSE_RANK to rank over each unique month in a year, partition by customerID and visit year from performancce date, ordered by month

SELECT cu.CustomerID, cu.CustomerName, extract(YEAR from per.pDate) as VisitedYear, TO_Char(per.pDate, 'MM') AS Month, DENSE_RANK() over (Partition by cu.CustomerID, extract (year from per.pDate) order by to_char (per.pDate, 'MM')) as MonthRanking

-- from SalesFact table with sf in short form

From SalesFact sf

-- Join Performance table by PerformanceNo

join Performance per on sf.PerformanceNo = per.PerformanceNo

-- Join Theatre table by TheatreNo

join Theatre the on sf.TheatreNo = the.TheatreNo

-- Join Customer table by CustomerID

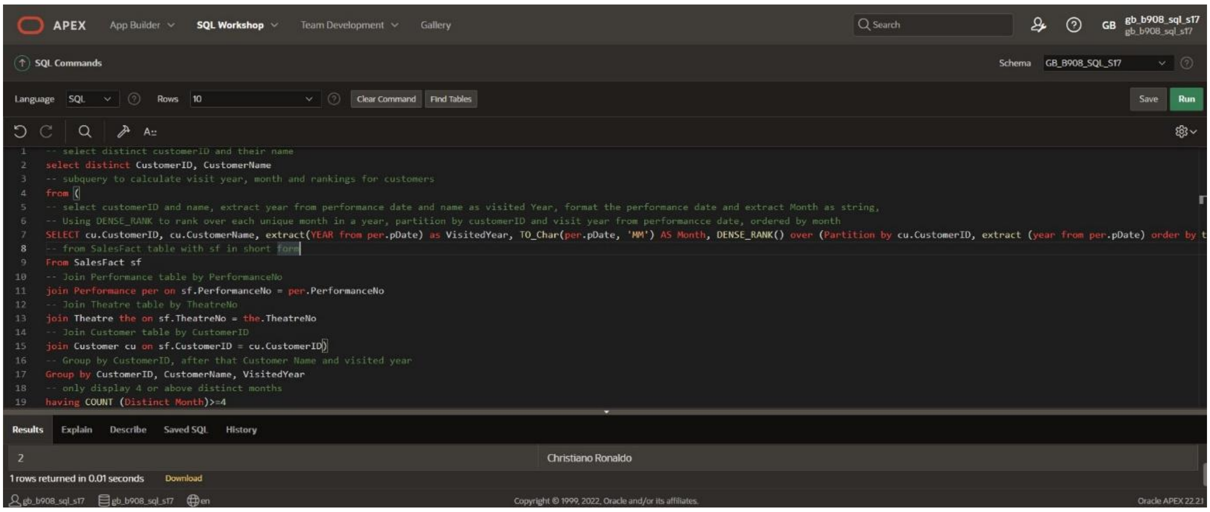
join Customer cu on sf.CustomerID = cu.CustomerID)

-- Group by CustomerID, after that Customer Name and visited year

Group by CustomerID, CustomerName, VisitedYear

-- only display 4 or above distinct months

having COUNT (Distinct Month)>=4



Schema: GB_B908_SQL_S17

Language: SQL

Rows: 10

```sql
-- select distinct customerID and their name
select distinct CustomerID, CustomerName
-- subquery to calculate visit year, month and rankings for customers
from (
-- select customerID and name, extract year from performance date and name as visited Year, format the performance date and extract Month as string,
-- Using DENSE_RANK to rank over each unique month in a year, partition by customerID and visit year from performancce date, ordered by month
SELECT cu.CustomerID, cu.CustomerName, extract(YEAR from per.pDate) as VisitedYear, TO_Char(per.pDate, 'MM') AS Month, DENSE_RANK() over (Partition by cu.CustomerID, extract (year from per.pDate) order by t
-- from SalesFact table with sf in short form
From SalesFact sf
-- Join Performance table by PerformanceNo
join Performance per on sf.PerformanceNo = per.PerformanceNo
-- Join Theatre table by TheatreNo
join Theatre the on sf.TheatreNo = the.TheatreNo
-- Join Customer table by CustomerID
join Customer cu on sf.CustomerID = cu.CustomerID)
-- Group by CustomerID, after that Customer Name and visited year
Group by CustomerID, CustomerName, VisitedYear
-- only display 4 or above distinct months
having COUNT (Distinct Month)>=4
```

```
2                                    Christiano Ronaldo
1 rows returned in 0.01 seconds
```

This query used RANK function and PARTITIONING method to display all clients who visited LTC theatres in at least 4 different months in a year. This query helps to find loyal customers, attracting them to visit LTC Theatres by providing benefits and loyalty programs.

4. List of the titles, production directors and play authors of all products with the highest total sale.

SQL Code:

-- Select production title, director and play author

SELECT ProductionTitle, ProductionDirector, PlayAuthor

-- subquery to calculate the sum of total sales, and to rank productions by total sales

from (

-- Select production title, director and play author, calculate total sale of each production, rank over productions by total sales and named as SalesRanking

select pro.ProductionTitle, pro.ProductionDirector, pro.PlayAuthor, SUM(sf.TotalAmount) as TotalSale, RANK() OVER (Order by SUM(sf.TotalAmount)DESC) AS SalesRanking

-- from SalesFact table with sf in short form

From SalesFact sf

-- Join Performance table by PerformanceNo

Join Performance per on sf.PerformanceNo = per.PerformanceNo

-- Join Production table by ProductionNo

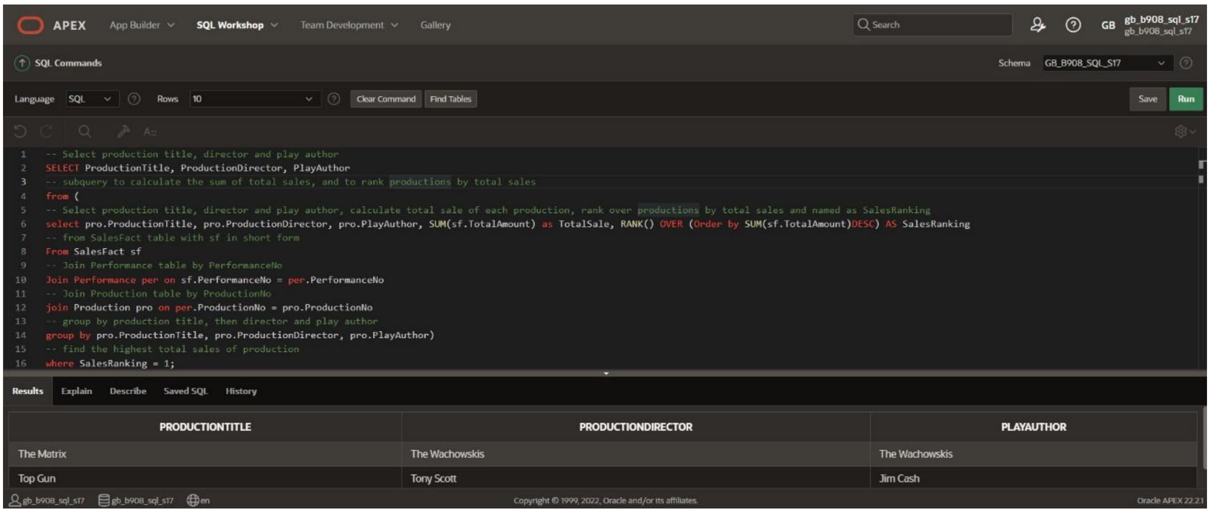
join Production pro on per.ProductionNo = pro.ProductionNo

-- group by production title, then director and play author

group by pro.ProductionTitle, pro.ProductionDirector, pro.PlayAuthor)

-- find the highest total sales of production

where SalesRanking = 1;



```
-- Select production title, director and play author
SELECT ProductionTitle, ProductionDirector, PlayAuthor
-- subquery to calculate the sum of total sales, and to rank productions by total sales
from (
-- Select production title, director and play author, calculate total sale of each production, rank over productions by total sales and named as SalesRanking
select pro.ProductionTitle, pro.ProductionDirector, pro.PlayAuthor, SUM(sf.TotalAmount) as TotalSale, RANK() OVER (Order by SUM(sf.TotalAmount)DESC) AS SalesRanking
-- from SalesFact table with sf in short form
From SalesFact sf
-- Join Performance table by PerformanceNo
Join Performance per on sf.PerformanceNo = per.PerformanceNo
-- Join Production table by ProductionNo
join Production pro on per.ProductionNo = pro.ProductionNo
-- group by production title, then director and play author
group by pro.ProductionTitle, pro.ProductionDirector, pro.PlayAuthor)
-- find the highest total sales of production
where SalesRanking = 1;
```

PRODUCTIONTITLE	PRODUCTIONDIRECTOR	PLAYAUTHOR
The Matrix	The Wachowskis	The Wachowskis
Top Gun	Tony Scott	Jim Cash

This query used RANK function to display the list of the titles, production directors and play authors of all products with the highest total sale. LTC may forecast which future movies that produced or written by directors and authors could have a high total sale, they could allocate more performances for these movies in order to generate a high profit.