KXO206

Database Management Systems

Database Creation Report

Prepared by:

Name	**Student No.**
Sang Shenghao	***619309***
Song Xuanxuan	**619287**
Zhang Junwen	**619270**
Cai Yuheng	**619301**

Due date: 2023/05/25

Contents

Introduction

A store wants to build a database to handle its sales, inventory, customers, and workers. The store has multiple locations and sells a wide range of things, including apparel, gadgets, and accessories. The purpose of the store to establish a database is to retain client information, evaluate sales data, and make sensible decisions on product inventories, price, and promotion in order to increase customer service and customer satisfaction.

Based on the background information, this report will begin with the creation of an entity relationship diagram, then convert it to a relational mode, create a data dictionary, add new users to the system, provide at least 10 records, and finally realize the functions of adding, deleting, changing, and querying the database. The specifications are as follows:

• Query the transaction table to get the list of the transactions of the first 10 customers and their associated product information.

• Update the second customer's email address with this new email address: you@dzi.com

• Delete the fourth customer details from the table.

• Change the price of any two products of your choice to a new price. 1 and 2 yuan respectively

• Use secure passwords and authentication for access to the database. (Provide this password in your submission)

• Encrypt sensitive data such as price and phone number. (Provide the encryption key in your submission)

• Using SQL statement, how would you modify the schema to track customer reviews and ratings for products?

• Describe how you would implement security measures to protect customer and employee data. Do not use SQL code here, just text.

• Using SQL statement show the structure and contents of each of the table you have create in the database.

Entity-Relationship Diagram

Read the case study and develop an Entity-Relationship Diagram (ERD) using the conventions taught in this unit. This diagram needs to describe all the business rules in the

ERDs are used to model and design relational databases, in terms of logic and business rules and in terms of the specific technology to be implemented.

Using an Entity-Relationship Diagram (ERD) approach, the database developers simplified the store's business processes into the following form.

The developers outlined the procedure in depth due to the complexity of maintaining many-to-many connections in a database. Finally, the database records the complete purchase process, including CUSTOMER details (ORDER_DETAILS), customer details (TRANSACTION_DETAILS), PRODUCT details (PRODUCT), product INVENTORY (product), STORE (STORE), shift (shift) SHIFT), personnel employed, and their SALARY.

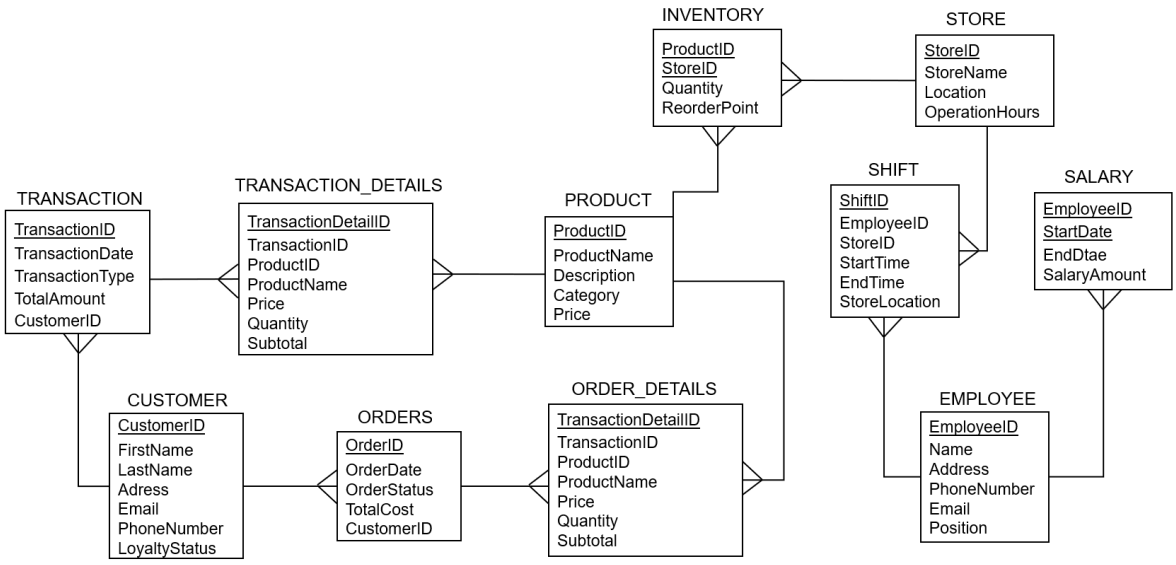

Figure 1. Entity-Relationship Diagram of Store

Relational schema and Data Dictionary

Convert your Entity-Relationship diagram into the Data dictionary table.

Create a relational database schema.

There are four steps to converting an Entity-Relationship Diagram to a relational database schema. First, each entity should become a relation used to create a table and raw properties. According to the ERD diagram above, there are eleven tables created, which are CUSTOMER, ORDERS, ORDER_DETAILS, PRODUCT, STORE, INVENTORY, TRANSACTION, TRANSACTION_DETAILS, EMPLOYEE, SALARY and SHIFT. Second, each many-to-many relationship becomes a table. However, many-to-many relationship does not exist in this case study. Third, each one-to-many relationship is converted to a foreign key. Lastly, the final relationship schemas are illustrated below.

CUSTOMER (CustomerID, [pk] FirstName, LastName, Address, Email, PhoneNumber, LoyaltyStatus)

ORDERS (OrderID, [pk] OrderDate, TotalCost, OrderStatus, CustomerID [fk1])

ORDER_DETAILS (OrderDetailID, [pk] ProductName, Price, Quantity, Subtotal, OrderID [fk1], ProductID [fk2])

PRODUCT (ProductID, [pk] ProductName, Description, Category, Price)

TRANSACTION (TransactionID, [pk] TransactionDate, TransactionType, TotalAmount, CustomerID [fk1])

TRANSACTION_DETAILS (TransactionDetailID, [pk] ProductName, Price, Subtotal, Quantity, TransactionID [fk1], ProductID [fk2])

INVENTORY (ProductID [fk1], StoreID [fk2], [pk] Quantity, ReorderPoint)

STORE (StoreID, [pk] StoreName, Location, OperationHours)

SHIFT (ShiftID, [pk] StartTime, EndTime, StoreLocation, EmployeeID [fk1], StoreID [fk2])

EMPLOYEE (EmployeeID, [pk] Name, Address, PhoneNumber, Email, Position)

SALARY (EmployeeID [fk1], StartDate [fk2], [pk] SalaryAmount, EndDate)

The data dictionary provides constraint information about the database, including definitions of entities (tables), data types of attributes, restrictions, and provides descriptions and examples of stored data to complement the ERDs, as shown below.

CUSTOMER				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
CustomerID	NUMBER(4)	PK	Customer identification number	1000
FirstName	VARCHAR2(10)	NOT NULL	Customer's first name	Kelly
LastName	VARCHAR2(10)	NOT NULL	Customer's last name	Smith
Address	VARCHAR2(20)		Customer's address	148 Main Street
PhoneNumber	VARCHAR2(15)	NOT NULL	Customer's phone number	+86147365249
Email	VARCHAR2(20)		Customer's email	15462@123.com
LoyaltyStatus	NUMBER(10)		Customer loyalty index	5

Table 1. The table of Customer

ORDERS				
Attribute	Data type	Constraint	Description	Example
OderID	NUMBER(4)	PK	Identification code of order	1001
OderDate	Date	NOT NULL	The date the order was created	02-APR-19
TotalCost	NUMBER(5,2)		Total price of the order	87.25
OrderStatus	VARCHAR2(20)		Order delivery status	Shipped
CustomerID	NUMBER(4)	FK to Customer.customerID	Customer identification number	1000

Table 2. The table of Order

ORDER_DETAILS				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
OderDetailID	NUMBER(4)	PK	Identification code of order item detail	1008
OderID	NUMBER(4)	FK to Orders.OrderID	The date identification number	1003
ProductID	NUMBER(4)	FK to Product.productID	Product identification number	1005
ProductName	VARCHAR2(20)		The name of a product in the order	T-shirt
Price	NUMBER(5,2)	NOT NULL	Unite price of this product in the order	32.95
Quantity	NUMBER(5)	NOT NULL	Quantity of this product in the order	2
Subtotal	NUMBER(5,2)		Total price of this product in the order	65.9

Table 3. The table of Order_Details

PRODUCT				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
ProductID	NUMBER(4)	PK	Identification code of product	1005
ProductName	VARCHAR2(20)		The name of product	T-shirt
Description	VARCHAR2(20)		Description of the product	Clothing
Category	VARCHAR2(20)		The category of the product	Clothing
Price	NUMBER(5,2)		Unite price of the product	32.95

Table 4. The table of Product

STORE				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
StoreID	NUMBER(4)	PK	Identification code of store	1002
StoreName	VARCHAR2(20)		The name of store	ABC store
Location	VARCHAR2(30)		Location of store	Chicago
OperationHours	VARCHAR2(20)		Operation hours of store	8a.m-10p.m

Table 5. The table of Store

INVENTORY				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
ProductID	NUMBER(4)	PK, FK to Product.productID	Identification code of product	1005
StoreID	NUMBER(4)	PK, FK to Store.storeID	Identification code of store	1002
Quantity	NUMBER(5)	NOT NULL	Quantity of product in stock	200
Reorderpoint	NUMBER(5)	NOT NULL	Reorder point of product	100

Table 6. The table of Inventory

TRANSACTION				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
TransactionID	NUMBER(4)	PK	Identification code of transaction	1001
TransactionDate	DATE	NOT NULL	The date transaction was created	12-JAN-2023
TransactionType	VARCHAR2(10)	NOT NULL	The type of transaction	Payment
TotalAmount	NUMBER(10,2)	NOT NULL	Total transaction amount	39.80
CustomerID	NUMBER(4)	FK to customer.CustomerID	Customer identification number	1002

Table 7. The table of Transaction

TRANSACTION_DETAILS				
Attribute	**Data type**	**Constraint**	**Description**	**Example**
TransactionDetailID	NUMBER(10)	PK	Identification code for each item transaction detail	2415
TransactionID	NUMBER(4)	FK to transaction.TransactionID	Identification code of transaction	1003
ProductID	NUMBER(4)	FK to production.ProductID	Identification code of product	1008
ProductName	VARCHAR2(20)		The name of product	Skirt
Price	NUMBER(5,2)	NOT NULL	Unite price of this item in transaction	78.45
Quantity	NUMBER(5)	NOT NULL	Quantity of item in transaction	2
Subtotal	NUMBER(10,2)	NOT NULL	Total price of the item in transaction	156.9

Table 8. The table of Transaction_Detail

SHIFT				
Attribute	**Data Type**	**Constraint**	**Description**	**Example**
ShiftID	NUMBER(4)	PK	Identification code of shift	1001
EmployeeID	NUMBER(4)	FK to employee.EmployeeID	Identification code of Employee	1001
StoreID	NUMBER(4)	FK to store.StoreID	Identification code of store	1002
StartTime	VARCHAR(20)	NOT NULL	Employee starts working time	26-Feb-2023 8:00 a.m
EndTime	VARCHAR(20)	NOT NULL	Employee ends working time	26-Feb-2023 20:00 p.m
StoreLocation	VARCHAR2(30)		The location of the store	Los Angeles

Table 9. The table of Shift

EMPLOYEE				
Attribute	**Data Type**	**Constraint**	**Description**	**Example**
EmployeeID	NUMBER(4)	PK	Identification code of Employee	1001
Name	VARCHAR2(20)	NOT NULL	Customer's first name	Kally
Address	VARCHAR2(20)		Customer's address	148 Main Street
PhoneNumber	VARCHAR2(20)		Customer's phone number	+86147365249
Email	VARCHAR2(30)		Customer's email	ja@gmail.com
Position	VARCHAR2(20)	NOT NULL	Customer's position	Manager

Table 10. The table of Employee

SALARY				
Attribute	**Data Type**	**Constraint**	**Description**	**Example**
EmployeeID	NUMBER(4)	PK, FK to Employee.employeeID	Identification code of Employee	1001
StartDate	Date	PK	Employee starts working time	26-Feb-2023
EndDate	Date		Employee ends working time	26-May-2023
SalaryAmount	NUMBER(10,2)	NOT NULL	Total salary of the employee	27-5000.34

Table 11. The table of Salary

Requirement 1

1. Using SQL statement in the oracle database create tables for all the schemas."
• Enter minimum 10 records (first record in each table should have random information) into each of the Table that you have created in your Oracle database

SQL Script

```
SPOOL C:\KXO206\Q1.txt

@@ Requirement 1


DROP TABLE CUSTOMER CASCADE CONSTRAINTS;

DROP TABLE ORDERS CASCADE CONSTRAINTS;

DROP TABLE ORDER_DETAILS CASCADE CONSTRAINTS;

DROP TABLE PRODUCT CASCADE CONSTRAINTS;

DROP TABLE INVENTORY CASCADE CONSTRAINTS;

DROP TABLE STORE CASCADE CONSTRAINTS;

DROP TABLE TRANSACTION CASCADE CONSTRAINTS;

DROP TABLE TRANSACTION_DETAILS CASCADE CONSTRAINTS;

DROP TABLE SHIFT CASCADE CONSTRAINTS;

DROP TABLE EMPLOYEE CASCADE CONSTRAINTS;

DROP TABLE SALARY CASCADE CONSTRAINTS;


Create table CUSTOMER

(CustomerID NUMBER(4),

FirstName VARCHAR2(10) NOT NULL,

LastName VARCHAR2(10) NOT NULL,

Address VARCHAR2(20),

PhoneNumber VARCHAR2(15) NOT NULL,

Email VARCHAR2(20),

LoyaltyStatus NUMBER(10),

CONSTRAINT customer_CustomerID_pk PRIMARY KEY(CustomerID));
```

```
INSERT INTO CUSTOMER
VALUES(1000, 'Kally', 'Smith', '148 Main Street', '+86147365249', '15426@123.com', 5);
INSERT INTO CUSTOMER
VALUES(1001, 'Alice', 'Leila', '82 Dirt Road', '+86137954943', 'alice@example.com', 6);
INSERT INTO CUSTOMER
VALUES(1002, 'Lucas', 'Jake', '114 East Savannah', '+86134567324', 'Lucas@123.com', 8);
INSERT INTO CUSTOMER
VALUES(1003, 'Schell', 'Steve', '1008 Grand Avenue', '+86139019237', 'Schell@123.com', 4);
INSERT INTO CUSTOMER
VALUES(1004, 'Daum', 'Michell', '9851231 Long Road', '+86139013478', 'Daum@123.com', 10);
INSERT INTO CUSTOMER
VALUES(1005, 'Falah', 'Kenneth', '456 Elm Street', '+86137827464', 'Falah@123.com', 8);
INSERT INTO CUSTOMER
VALUES(1006, 'Perez', 'Jorge', '123 Main Steet', '+86287426346', 'Perez@123.com', 6);
INSERT INTO CUSTOMER
VALUES(1007, 'Cruz', 'Meshia', '69821 South Avenue', '+86247268374', 'Cruz@123.com', 7);
INSERT INTO CUSTOMER
VALUES(1008, 'Smith', 'Reese', '123 Main Steet', '+86348276467', 'Smith@123.com', 6);
INSERT INTO CUSTOMER
VALUES(1009, 'Tom', 'Thomas', '123 Main Steet', '+86847563846', 'Tom@123.com', 9);

CREATE table product
(ProductID NUMBER(4),
ProductName VARCHAR2(20),
Description VARCHAR2(20),
Category VARCHAR2(20),
Price NUMBER(5,2),
CONSTRAINT product_ProductID_pk PRIMARY KEY(ProductID));

INSERT INTO product
```

```
VALUES(1001, 'Shorts', 'Clothing', 'Clothing', 20.65);

INSERT INTO product

VALUES(1002, 'Skirt', 'Clothing', 'Clothing', 33.75);

INSERT INTO product

VALUES(1003, 'Keyboard', 'Electronic', 'Electronic', 89.90);

INSERT INTO product

VALUES(1004, 'Watch', 'accessories', 'accessories', 40.00);

INSERT INTO product

VALUES(1005, 'T-shirt', 'Clothing', 'Clothing', 32.95);

INSERT INTO product

VALUES(1006, 'Mouse', 'Electronic', 'Electronic', 42.50);

INSERT INTO product

VALUES(1007, 'Mouse Pad', 'accessories', 'accessories', 22.50);

INSERT INTO product

VALUES(1008, 'Battery', 'accessories', 'accessories', 2.50);

INSERT INTO product

VALUES(1009, 'Hat', 'Clothing', 'Clothing', 12.50);

INSERT INTO product

VALUES(1010, 'Phone', 'Electronic', 'Electronic', 250.00);


Create table ORDERS(

OrderID NUMBER(4),

OrderDate Date NOT NULL,

OrderStatus VARCHAR2(20),

TotalCost NUMBER (5,2),

CustomerID NUMBER(4),

CONSTRAINT orders_OrderID_pk PRIMARY KEY(OrderID),

CONSTRAINT orders_CustomerID_fk FOREIGN KEY (CustomerID)

   REFERENCES customer (CustomerID));
```

```
INSERT INTO ORDERS
VALUES(1000, '01-JAN-2023', 'shipped', 12.50, 1003);
INSERT INTO ORDERS
VALUES(1001, '09-MAY-2023', 'shipped', 50.00, 1007);
INSERT INTO ORDERS
VALUES(1002, '15-JUNE-2023', 'unshipped', 32.95, 1003);
INSERT INTO ORDERS
VALUES(1003, '15-JUNE-2023', 'unshipped', 88.40, 1000);
INSERT INTO ORDERS
VALUES(1004, '29-JUNE-2023', 'shipped', 45.00, 1001);
INSERT INTO ORDERS
VALUES(1005, '12-JAN-2023', 'unshipped', 67.50, 1002);
INSERT INTO ORDERS
VALUES(1006, '22-FEB-2023', 'shipped', 121.30, 1004);
INSERT INTO ORDERS
VALUES(1007, '07-MAR-2023', 'unshipped', 59.40, 1004);
INSERT INTO ORDERS
VALUES(1008, '09-MAY-2023', 'shipped', 250.00, 1008);
INSERT INTO ORDERS
VALUES(1009, '16-MAR-2023', 'shipped', 85.00, 1005);
INSERT INTO ORDERS
VALUES(1010, '27-MAY-2023', 'shipped', 250.00, 1006);
INSERT INTO ORDERS
VALUES(1011, '21-AUG-2023', 'unshipped', 89.90, 1009);

CREATE table order_details
(OrderDetailID NUMBER(4),
OrderID NUMBER(4),
ProductID NUMBER(4),
ProductName VARCHAR2(20),
```

```
Price NUMBER(5,2) NOT NULL,
Quantity NUMBER(5) NOT NULL,
Subtotal NUMBER(5,2),
CONSTRAINT order_details_OrderDetailID_pk PRIMARY KEY(OrderDetailID),
CONSTRAINT order_details_OrderID_fk FOREIGN KEY (OrderID)
   REFERENCES ORDERS (OrderID),
CONSTRAINT order_details_ProductID_fk FOREIGN KEY (ProductID)
   REFERENCES Product (ProductID));

INSERT INTO order_details
VALUES(1001, 1003, 1005, 'Tshirt', 32.95, 2, 65.90);
INSERT INTO order_details
VALUES(1002, 1006, 1004, 'Watch', 40.00, 2, 80.00);
INSERT INTO order_details
VALUES(1003, 1006, 1001, 'Shorts', 20.65, 2, 41.30);
INSERT INTO order_details
VALUES(1004, 1002, 1005, 'T-shirt', 32.95, 1, 32.95);
INSERT INTO order_details
VALUES(1005, 1005, 1002, 'Skirt', 33.75, 2, 67.50);
INSERT INTO order_details
VALUES(1006, 1003, 1007, 'Mouse Pat', 22.50, 1, 22.50);
INSERT INTO order_details
VALUES(1007, 1004, 1007, 'Mouse Pat', 22.50, 2, 45.00);
INSERT INTO order_details
VALUES(1008, 1000, 1008, 'Battery', 2.50, 5, 12.50);
INSERT INTO order_details
VALUES(1009, 1001, 1009, 'Hat', 12.50, 4, 50.00);
INSERT INTO order_details
VALUES(1010, 1007, 1008, 'Battery', 19.80, 3, 59.40);
INSERT INTO order_details
```

```
VALUES(1011, 1008, 1010, 'Phone', 250.00, 1, 250.00);

INSERT INTO order_details

VALUES(1012, 1009, 1006, 'Mouse', 42.50, 2, 85.00);

INSERT INTO order_details

VALUES(1013, 1010, 1001, 'Phone', 250.00, 1, 250.00);

INSERT INTO order_details

VALUES(1014, 1011, 1003, 'keyboard', 89.90, 1, 89.90);


CREATE TABLE store (

   StoreID NUMBER(4),

   StoreName VARCHAR2(20),

   Location VARCHAR2(30),

   OperationHours VARCHAR2(20),

   PRIMARY KEY (StoreID));


INSERT INTO store

VALUES (1001, 'ABC store', 'Los Angeles','8a.m-16p.m');

INSERT INTO store

VALUES(1002, 'AC store', 'New York City','12a.m-20p.m' );

INSERT INTO store

VALUES(1003, 'AB store', 'Miami','8a.m-16p.m');

INSERT INTO store

VALUES(1004, 'BC store', 'Chicago','12a.m-20p.m');

INSERT INTO store

VALUES(1005, 'C store', 'New Orleans','12a.m-20p.m');

INSERT INTO store

VALUES(1006, 'A store', 'Seattle','12a.m-20p.m');

INSERT INTO store

VALUES(1007, 'B store', 'San Francisco','8a.m-16p.m');

INSERT INTO store
```

```
VALUES(1008, 'BC store', 'Boston','12a.m-20p.m');

INSERT INTO store

VALUES(1009, 'ABC store','Nashville','8a.m-16p.m');

INSERT INTO store

VALUES(1010, 'AC store', 'Austin','12a.m-20p.m');



CREATE TABLE inventory (

 ProductID NUMBER(4),

 StoreID NUMBER(4),

 Quantity NUMBER(5) NOT NULL,

 ReorderPoint NUMBER(5) NOT NULL,

 CONSTRAINT inventory_ProductID_StoreID_pk PRIMARY KEY (ProductID, StoreID),

 CONSTRAINT inventory_ProductID_fk FOREIGN KEY (ProductID)

   REFERENCES product(ProductID),

 CONSTRAINT inventory_StoreID_fk FOREIGN KEY (StoreID)

   REFERENCES store(StoreID));


INSERT INTO INVENTORY

VALUES (1001, 1001, 100, 60);

INSERT INTO INVENTORY

VALUES (1003, 1001, 50, 30);

INSERT INTO INVENTORY

VALUES (1005, 1001, 80, 40);

INSERT INTO INVENTORY

VALUES (1002, 1002, 78, 35);

INSERT INTO INVENTORY

VALUES (1005, 1002, 45, 40);

INSERT INTO INVENTORY

VALUES (1007, 1002, 112, 80);
```

```
INSERT INTO INVENTORY
VALUES (1001, 1003, 102, 70);
INSERT INTO INVENTORY
VALUES (1002, 1003, 48, 35);
INSERT INTO INVENTORY
VALUES (1004, 1003, 43, 35);
INSERT INTO INVENTORY
VALUES (1005, 1004, 65, 60);
INSERT INTO INVENTORY
VALUES (1006, 1004, 72, 60);
INSERT INTO INVENTORY
VALUES (1008, 1005, 65, 40);
INSERT INTO INVENTORY
VALUES (1009, 1005, 52, 35);
INSERT INTO INVENTORY
VALUES (1003, 1006, 68, 60);
INSERT INTO INVENTORY
VALUES (1006, 1006, 62, 75);
INSERT INTO INVENTORY
VALUES (1002, 1007, 168, 80);
INSERT INTO INVENTORY
VALUES (1010, 1007, 97, 80);
INSERT INTO INVENTORY
VALUES (1001, 1008, 58, 40);
INSERT INTO INVENTORY
VALUES (1009, 1008, 67, 40);
INSERT INTO INVENTORY
VALUES (1002, 1009, 68, 35);
INSERT INTO INVENTORY
VALUES (1004, 1009, 47, 35);
```

```
INSERT INTO INVENTORY
VALUES (1006, 1010, 77, 50);
INSERT INTO INVENTORY
VALUES (1007, 1010, 113, 85);

CREATE TABLE transaction (
 TransactionID NUMBER(4),
 TransactionDate DATE NOT NULL,
 TransactionType VARCHAR2(10) NOT NULL,
 TotalAmount NUMBER(10,2) NOT NULL,
 CustomerID NUMBER(4),
 CONSTRAINT transaction_TransactionID_pk PRIMARY KEY (TransactionID),
 CONSTRAINT transaction_CustomerID_fk FOREIGN KEY (CustomerID)
   REFERENCES customer(CustomerID));

INSERT INTO transaction
VALUES(1000, '02-JAN-2023', 'payment', 12.50, 1003);
INSERT INTO transaction
VALUES(1001, '9-MAY-2023', 'payment', 50.00, 1007);
INSERT INTO transaction
VALUES(1002, '18-JUNE-2023', 'credit', 32.95, 1003);
INSERT INTO transaction
VALUES(1003, '15-JUNE-2023', 'payment', 88.40, 1000);
INSERT INTO transaction
VALUES(1004, '30-JUNE-2023', 'credit', 45.00, 1001);
INSERT INTO transaction
VALUES(1005, '12-JAN-2023', 'credit', 67.50, 1002);
INSERT INTO transaction
VALUES(1006, '24-FEB-2023', 'credit', 121.30, 1004);
INSERT INTO transaction
```

```
VALUES(1007, '07-MAR-2023', 'payment', 59.40, 1004);

INSERT INTO transaction

VALUES(1008, '09-MAY-2023', 'payment', 250.00, 1008);

INSERT INTO transaction

VALUES(1009, '16-MAR-2023', 'payment', 85.00, 1005);

INSERT INTO transaction

VALUES(1010, '28-MAY-2023', 'credit', 250.00, 1006);

INSERT INTO transaction

VALUES(1011, '21-AUG-2023', 'payment', 89.90, 1009);


CREATE TABLE transaction_details (

 TransactionDetailID NUMBER(10),

 TransactionID NUMBER(4),

 ProductID NUMBER(4),

 ProductName VARCHAR2(20),

 Price NUMBER(5,2) NOT NULL,

 Quantity NUMBER(5) NOT NULL,

 Subtotal NUMBER(10,2) NOT NULL,

 CONSTRAINT transaction_details_TransactionDetailID_pk PRIMARY KEY (TransactionDetailID),

 CONSTRAINT transaction_details_TransactionID_fk FOREIGN KEY (TransactionID)

   REFERENCES transaction(TransactionID),

 CONSTRAINT transaction_details_ProductID_fk FOREIGN KEY (ProductID)

   REFERENCES product(ProductID));


INSERT INTO transaction_details

VALUES(1001, 1003, 1005, 'Tshirt', 32.95, 2, 65.90);

INSERT INTO transaction_details

VALUES(1002, 1006, 1004, 'Watch', 40.00, 3, 80.00);

INSERT INTO transaction_details

VALUES(1003, 1006, 1001, 'Shorts', 20.65, 2, 41.30);
```

```
INSERT INTO transaction_details
VALUES(1004, 1002, 1005, 'T-shirt', 32.95, 1, 32.95);
INSERT INTO transaction_details
VALUES(1005, 1005, 1002, 'Skirt', 33.75, 2, 67.50);
INSERT INTO transaction_details
VALUES(1006, 1003, 1007, 'Mouse Pat', 22.50, 1, 22.50);
INSERT INTO transaction_details
VALUES(1007, 1004, 1007, 'Mouse Pat', 22.50, 2, 45.00);
INSERT INTO transaction_details
VALUES(1008, 1000, 1008, 'Battery', 2.50, 5, 12.50);
INSERT INTO transaction_details
VALUES(1009, 1001, 1009, 'Hat', 12.50, 4, 50.00);
INSERT INTO transaction_details
VALUES(1010, 1007, 1008, 'Battery', 19.80, 3, 59.40);
INSERT INTO transaction_details
VALUES(1011, 1008, 1010, 'Phone', 250.00, 1, 250.00);
INSERT INTO transaction_details
VALUES(1012, 1009, 1006, 'Mouse', 42.50, 2, 85.00);
INSERT INTO transaction_details
VALUES(1013, 1010, 1001, 'Phone', 250.00, 1, 250.00);
INSERT INTO transaction_details
VALUES(1014, 1011, 1003, 'keyboard', 89.90, 1, 89.90);

CREATE TABLE employee (
  EmployeeID NUMBER(4),
  Name VARCHAR2(20) NOT NULL,
  Address VARCHAR2(20),
  PhoneNumber VARCHAR2(20),
  Email VARCHAR2(30),
  Position VARCHAR2(20) NOT NULL,
```

```
 Constraint Employee_EmployeeID_pk PRIMARY KEY (employeeID));


INSERT INTO EMPLOYEE

VALUES (1001, 'John Doe', '123 Main St', '+8618721567890', 'johndoe@example.com', 'Manager');

INSERT INTO EMPLOYEE

VALUES (1002, 'Jane Smith', '456 Oak Ave', '+8613862457893', 'janesmith@example.com', 'Sales');

INSERT INTO EMPLOYEE

VALUES (1003, 'Bob Johnson', '789 Elm St', '+8615824768905', 'bobjohnson@example.com','Cashier');

INSERT INTO EMPLOYEE

VALUES (1004, 'Sara Lee', '321 Pine St', '+8617896542310', 'saralee@example.com', 'Sales');

INSERT INTO EMPLOYEE

VALUES (1005, 'Tom Smith', '654 Maple Ave', '+8615698745234', 'tomsmith@example.com', 'Stock
Clerk');

INSERT INTO EMPLOYEE

VALUES (1006, 'Amy Nguyen', '987 Cedar St', '+8618356978456', 'amynguyen@example.com', 'Sales');

INSERT INTO EMPLOYEE

VALUES (1007, 'Mike Johnson', '246 Birch Rd', '+8614725869312', 'mikejohnson@example.com',
'Cashier');

INSERT INTO EMPLOYEE

VALUES (1008, 'Emily Chen', '135 Cherry St', '+8619387562147', 'emilychen@example.com',
'Manager');

INSERT INTO EMPLOYEE

VALUES (1009, 'David Lee', '864 Pine Ave', '+8617283490876', 'davidlee@example.com', 'Sales');

INSERT INTO EMPLOYEE

VALUES (1010, 'Lisa Kim', '279 Oak Rd', '+8615642390785', 'lisakim@example.com', 'Cashier');



CREATE TABLE salary (

 EmployeeID NUMBER(4),

 StartDate Date,

 EndDate Date,

 SalaryAmount NUMBER(10,2) NOT NULL,
```

```
 Constraint Salary_EmployeeID_StartDate_pk PRIMARY KEY (employeeID, startDate),
 Constraint Salary_EmployeeID_fk FOREIGN KEY (employeeID) REFERENCES employee(employeeID));

INSERT INTO SALARY
VALUES (1001, '01-JAN-2023','04-JAN-2023' , 9000.00);
INSERT INTO SALARY
VALUES (1002, '04-JAN-2023', '07-JAN-2023', 5000.00);
INSERT INTO SALARY
VALUES (1003, '07-JAN-2023', '12-JAN-2023', 4000.00);
INSERT INTO SALARY
VALUES (1004, '02-JAN-2023', '11-JAN-2023', 5500.00);
INSERT INTO SALARY
VALUES (1005, '11-JAN-2023', '26-Feb-2023', 7500.00);
INSERT INTO SALARY
VALUES (1006, '26-Feb-2023','26-Mar-2023' , 5000.00);
INSERT INTO SALARY
VALUES (1007, '20-Mar-2023','25-Mar-2023' , 4000.00);
INSERT INTO SALARY
VALUES (1008, '6-Apr-2023','18-Apr-2023', 10000.00);
INSERT INTO SALARY
VALUES (1009, '23-May-2023','23-May-2023' ,8500.00);
INSERT INTO SALARY
VALUES (1010, '15-May-2023','20-May-2023', 7000.00);

CREATE TABLE SHIFT (
   ShiftID NUMBER(4),
   EmployeeID NUMBER(4),
   StoreID NUMBER(4),
   StartTime VARCHAR2(20) NOT NULL,
   EndTime VARCHAR2(20) NOT NULL,
```

```
    StoreLocation VARCHAR2(30),
    Constraint Shift_ShiftID_pk PRIMARY KEY (ShiftID),
    Constraint Shift_EmployeeID_fk FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),
    Constraint Shift_StoreID_fk FOREIGN KEY (StoreID) REFERENCES STORE(StoreID));

INSERT INTO shift
VALUES (1001, 1005, 1001,'2023-01-01 08:00:00', '2023-01-01 16:00:00', 'Los Angeles');
INSERT INTO shift
VALUES(1002, 1004, 1002,'2023-01-11 12:00:00', '2023-01-11 20:00:00' , 'New York City');
INSERT INTO shift
VALUES(1003, 1001, 1003,'2023-01-22 08:00:00', '2023-01-32 16:00:00', 'Miami');
INSERT INTO shift
VALUES(1004, 1003, 1004,'2023-02-02 12:00:00', '2023-02-02 20:00:00', 'Chicago' );
INSERT INTO shift
VALUES(1005, 1002, 1005,'2023-02-12 12:00:00', '2023-02-12 20:00:00' , 'New Orleans');
INSERT INTO shift
VALUES(1006, 1006, 1006,'2023-03-22 12:00:00', '2023-03-22 20:00:00' , 'Seattle');
INSERT INTO shift
VALUES(1007, 1007, 1007,'2023-04-04 08:00:00', '2023-04-04 16:00:00' , 'San Francisco');
INSERT INTO shift
VALUES(1008, 1010, 1008,'2023-04-24 12:00:00', '2023-04-24 20:00:00', 'Boston' );
INSERT INTO shift
VALUES(1009, 1009, 1009,'2023-05-05 08:00:00', '2023-05-15 16:00:00','Nashville' );
INSERT INTO shift
VALUES(1010, 1008, 1010,'2023-05-15 12:00:00', '2023-05-15 20:00:00' , 'Austin');
SPOOL OFF
```

Results

Script output:

USER SERVER SESSION_ID

--- --- ---

SYSTEM XE 470012

TERMINAL USED_ON USED_AT

--- --- ---

DESKTOP-A3EL3T9 24-MAY-2023 02:52 P.M.

Requirement: 1

Table dropped.

…

Table dropped.

Table created.

1 row created.

…

1 row created.

Table created.

1 row created.

…

1 row created.

Table created.

1 row created.

…

1 row created.

Table created.

1 row created.

…

1 row created.

Table created.

1 row created.

…

1 row created.

Comments

1. The create table command is used to create database tables:

```
CREATE TABLE table_name (
    column_1 data_type column_constraint,
    column_2 data_type column_constraint,
    ...
    table_constraint
);
```

2.

- Character data types：These are used to store alphanumeric values or strings.

  CHAR: Fixed-length string; if no length is supplied, default value is 1.

  VARCHAR(2): Variable length string, a VARCHAR2 column's maximum length for a value is governed by the size supplied when the column is defined2.

- Numeric data types: These are used to hold numerical values, including decimal and full numbers.

  NUMBER (p,s): Negative numbers can be stored in a number with a decimal length of s and a total length of p digits. The maximum number of digits is 38. Insufficient digits will result in the number being rounded off.

- Date/Time data types: Date and time values are kept in these.

  DATE: Date and time type.

3. INSERT INTO table_name (column_list)
   VALUES (value_list);
   Column names can be omitted, and the data is inserted in the order of the columns.

4. Create constraints at table level when creating table.

[CONSTRAINT constraintname] constrainttype (columnname, …),

- Primary Key

  A primary key is a constraint that is used to enforce uniqueness and identify each row uniquely in a table. A primary key is a column or a group of columns in a table that uniquely identifies each row in the table. a primary key is a constraint that is used to enforce uniqueness and identify each row uniquely in a table. A primary key is a column or a group of columns in a table that uniquely identifies each row in the table.

  A primary key constraint can be defined on one or more columns in a table.

5. Foreign key is a constraint that is used to enforce referential integrity between two tables in a relational database.

   A foreign key is a column or a group of columns in a table that refers to the primary key of another table. It ensures that the values in the foreign key column(s) of the referencing table must match the values in the primary key column(s) of the referenced table.

Requirement 2

Add a new customer with the following particulars.
- First name: Wǒài
- Last Name: xuéxiào
- Email: woaix@data.com
- Phone:+86346578910

SQL Script

```
SPOOL C:\KXO206\Q2.txt
@@ requirement 2

INSERT INTO customer(CustomerID, FirstName, LastName, PhoneNumber, Email)
VALUES(1010, 'Woai', 'xuexiao', '+86346578910', 'woaix@data.com');

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         470012


TERMINAL                  USED_ON                       USED_AT
------------------------------ ----------------------------------------- --------------------
DESKTOP-A3EL3T9             24-MAY-2023                   03:04 P.M.


Requirement: 2


1 row created.
```

Comments

This specific command is an INSERT INTO statement:

INSERT INTO table_name (column_list)

VALUES (value_list);

The CustomerID, FirstName, LastName, PhoneNumber, and Email are the names of the columns in the customer table where the values will be inserted. The values in the VALUES clause correspond to these columns in the order they are listed. Importantly, since the CustomerID is the primary key, it must be populated with value.

The CustomerID value is an integer (1010), so it does not need to be enclosed in single quotes. However, the other values (FirstName, LastName, PhoneNumber, and Email) are all strings, so they are enclosed in single quotes.

Requirement 3

Add a new product
- Product name: Qúnzi
- Product description: Clothing
- Product price: ¥10

SQL Script

```
SPOOL C:\KXO206\Q3.txt
@@ requirement 3

INSERT INTO product
VALUES(1011, 'Qunzi', 'Clothing', 'Clothing', 10);

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                         SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                              470012


TERMINAL                       USED_ON                                  USED_AT
------------------------------ ---------------------------------------- --------------------
DESKTOP-A3EL3T9                24-MAY-2023                              03:11 P.M.


Requirement: 3


1 row created.
```

Comments

```
INSERT INTO table_name (column_list)
VALUES (value_list);
```

"INSERT INTO" statement is used to insert a new row into the "product" table.

The "VALUES" clause is used to specify the values to be inserted into the columns of the new row.

Column names can be omitted, and the data is inserted in the order of the columns.

String values (such as 'Qunzi' and 'Clothing') must be enclosed in single quotes. Numeric values (such as 1011 and 10) do not require quotes.

Requirement 4

Query the transaction table to get the list of the transactions of the first 10 customers and their associated product information.

SQL Script

```
SPOOL C:\KXO206\Q4.txt
@@ requirement 4

SET LINESIZE 250
SET PAGESIZE 200

SELECT c.CustomerID, t.TransactionID, t.TransactionDate, t.TransactionType, t.TotalAmount,
p.ProductID, p.ProductName, p.Description, p.Category, p.Price as unitPrice
FROM CUSTOMER c
JOIN TRANSACTION t ON c.CustomerID = t.CustomerID
JOIN TRANSACTION_DETAILS td ON t.TransactionID = td.TransactionID
JOIN PRODUCT p ON td.ProductID = p.ProductID
WHERE c.CustomerID IN (SELECT CustomerID FROM CUSTOMER WHERE ROWNUM <= 10)
ORDER BY c.CustomerID;

SPOOL OFF
```

Results

Script output:

```
USER         SERVER                         SESSION_ID
---------- ------------------------------ ----------
SYSTEM       XE                              170608


TERMINAL                       USED_ON                                  USED_AT
------------------------------ ---------------------------------------- --------------------
LAPTOP-G0QGGGE8                24-MAY-2023                              11:24 A.M.


Requirement: 4


CUSTOMERID TRANSACTIONID TRANSACTIOND TRANSACTIONTYPE      TOTALAMOUNT  PRODUCTID PRODUCTNAME                DESCRIPTION
CATEGORY                       UNITPRICE
```

1000	1003	15-JUN-23	payment	88.4	1007	Mouse Pad	accessories	accessories	22.5
1000	1003	15-JUN-23	payment	88.4	1005	T-shirt	Clothing	Clothing	32.95
1001	1004	30-JUN-23	credit	45	1007	Mouse Pad	accessories	accessories	22.5
1002	1005	12-JAN-23	credit	67.5	1002	Skirt	Clothing	Clothing	33.75
1003	1002	18-JUN-23	credit	32.95	1005	T-shirt	Clothing	Clothing	32.95
1003	1000	02-JAN-23	payment	12.5	1008	Battery	accessories	accessories	2.5
1004	1006	24-FEB-23	credit	121.3	1001	Shorts	Clothing	Clothing	20.65
1004	1007	07-MAR-23	payment	59.4	1008	Battery	accessories	accessories	2.5
1004	1006	24-FEB-23	credit	121.3	1004	Watch	accessories	accessories	40
1005	1009	16-MAR-23	payment	85	1006	Mouse	Electronic	Electronic	42.5
1006	1010	28-MAY-23	credit	250	1001	Shorts	Clothing	Clothing	20.65
1007	1001	09-MAY-23	payment	50	1009	Hat	Clothing	Clothing	12.5
1008	1008	09-MAY-23	payment	250	1010	Phone	Electronic	Electronic	250
1009	1011	21-AUG-23	payment	89.9	1003	Keyboard	Electronic	Electronic	89.9

14 rows selected.

Comments

The script retrieves transaction and product information for the first ten customers in the CUSTOMER table based on join multiple tables and operations such as ROWNUM.

The SELECT statement specifies the columns to retrieve from the joined tables, including the columns in CUSTOMER, TRANSACTION and PRODUCT tables.

The FROM clause specifies the tables to join: CUSTOMER, TRANSACTION, TRANSACTION_DETAILS, and PRODUCT. The JOIN ON clause joins multiple tables by specifying columns in different tables for an exact match. The JOIN ON clause connects the four tables according to their CustomerID, TransactionID, and ProductID keys. For example, PRODUCT table joins TRANSACTION_DETAILS table by matching ProductID column in each table. Among the statement, aliases simplify the task of specifying a table name.

The WHERE clause limits the results of customers with the subquery statement, which uses the ROWNUM pseudo-column returns first ten records in the CUSTOMER table to select first ten customers' CUSTOMERID. The 'IN' operator is used to match CustomerID at least one condition inside the parentheses.

The Order by clause sort the query results according to the default ascending order of CustomerID.

Requirement 5

Update the second customer's email address with this new email address: you@dzi.com

SQL Script

```
SPOOL C:\KXO206\Q5.txt
@@ requirement 5

UPDATE CUSTOMER
SET Email='you@dzi.com'
WHERE CustomerID = (SELECT CustomerID
                    FROM (SELECT ROWNUM no, CustomerID FROM CUSTOMER)
                    WHERE no=2);

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608


TERMINAL                  USED_ON                          USED_AT
------------------------------- ---------------------------------------- --------------------
LAPTOP-G0QGGGE8             24-MAY-2023                    11:48 A.M.


Requirement: 5


1 row updated.
```

Comments

This statement uses the UPDATE statement to update the E-mail address of a specific CUSTOMER in the Customer table.

The UPDATE statement first specifies the name of the table to be updated, CUSTOMER. The SET clause specifies that the column 'Email' be updated with a value of 'you@dzi.com'.

The WHERE clause specifies the conditions under which the update is performed. In this case, the constraint implemented through the subquery is that the CustomerID for the row to be updated is the second row in the CUSTOMER table.

The subquery in the WHERE clause uses the ROWNUM pseudo-column to create a derived table with assigned sequential number column to each row returned by the inner subquery (SELECT ROWNUM no, CustomerID FROM CUSTOMER). The outer subquery then using WHERE clause filters the results to only return the CustomerID of the row with no=2, which corresponds to the second customer in the CUSTOMER table.

Requirement 6

Delete the fourth customer details from the table.

SQL Script

```
SPOOL C:\KXO206\Q6.txt
@@ requirement 6
DELETE FROM order_details
WHERE OrderID IN (SELECT OrderID
                  FROM orders
                  WHERE CustomerID =
                              (SELECT CustomerID
                               FROM (SELECT ROWNUM no, CustomerID from customer)
                               WHERE no=4));

DELETE FROM orders
WHERE CustomerID = (SELECT CustomerID
                    FROM (SELECT ROWNUM no, CustomerID from customer)
                    WHERE no=4);

DELETE FROM transaction_details
WHERE TransactionID IN (SELECT TransactionID
                        FROM transaction
                        WHERE CustomerID =
                                (SELECT CustomerID
                                 FROM (SELECT ROWNUM no, CustomerID from customer)
                                 WHERE no=4));
DELETE FROM transaction
WHERE CustomerID = (SELECT CustomerID
                    FROM (SELECT ROWNUM no, CustomerID from customer)
                    WHERE no=4);

DELETE FROM customer
WHERE CustomerID = (SELECT CustomerID
                    FROM (SELECT ROWNUM no, CustomerID from customer)
                    WHERE no=4);

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608

TERMINAL                  USED_ON                          USED_AT
------------------------------ ---------------------------------------- --------------------
LAPTOP-G0QGGGE8           24-MAY-2023                      12:13 P.M.
```

Requirement: 6

2 rows deleted.

2 rows deleted.

2 rows deleted.

2 rows deleted.

1 row deleted.

Comments

When a FOREIGN KEY constraint exists between two tables, by default, a record can't be deleted from the parent table if matching entries exist in the child table.

Since the CUSTOMER table is the parent table of the ORDERS and TRANSACTION tables, which are in turn the parent table of the ORDER_DETAILS and TRANSACTION_DETAILS, Therefore, details about the fourth customer in the CUSTOMER table cannot be deleted directly, while it needs to start deleting records from the child tables.

The first two DELETE statements is used to delete the records of ORDER_DETAILS first and then ORDERS table associated with the customer. The WHERE clause in each DELETE statement uses a subquery to get the OrderID values or CustomerID values associated with the customer, respectively, using the ROWNUM pseudo-column to assign a sequential number to each row returned by the innermost subquery (SELECT ROWNUM no, CustomerID FROM CUSTOMER). Then, the subquery limits the record of sequence number 4 with a WHERE clause

(no=4) to filter the CustomerID in the fourth row of the CUSTOMER table. Especially, when deleting the data in the ORDER_DETAILS table, the nested query uses the ORDER table to match the filtered CustomerID to get the corresponding OrderID. The 'IN' operator is used to match OrderID at least one condition inside the parentheses.

The following two DELETE statements for TRANSACTION_DETAILS and TRANSACTION tables is similar to the ORDER_DETAILS and ORDERS tables.

After deleting the records in the child table, records in the parent table can be deleted.

The last DELETE statement deletes the record in CUSTOMER table, using a subquery in the WHERE clause to limit the CustomerID of outer query which equal to the selected CustomerID from a derived table that using ROWNUM pseudo-column assigns a row number to each record in the "customer" table. In a derived table, the WHERE clause restricts records with row number equal to 4 to select the CustomerID of the fourth customer.

Overall, this SQL script is designed to delete all data associated with a specific customer in a cascading manner. It starts by deleting the child records (order details and transaction details), then deletes the parent records (orders and transactions), and finally deletes the customer record itself.

Requirement 7

Change the price of any two products of your choice to a new price. 1 and 2 yuan respectively

SQL Script

```
SPOOL C:\KXO206\Q7.txt
@@ requirement 7

UPDATE product
SET price =
    CASE
       WHEN ProductName = 'Battery' THEN 1
       WHEN ProductName = 'Hat' THEN 2
    END
 WHERE ProductName IN ('Battery', 'Hat');

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608


TERMINAL                  USED_ON                          USED_AT
------------------------------- ---------------------------------------- --------------------
LAPTOP-G0QGGGE8           24-MAY-2023                      13:13 P.M.


Requirement: 7


2 rows updated.
```

Comments

"UPDATE" statement, which is used to modify existing data in a table.

The "SET" clause is used to specify the new value for the "price" column. In this statement, it is using a "CASE" statement to set the price based on the product name. The "CASE" statement is a conditional statement that allows you to perform different actions based on different conditions.

The "WHEN" clause is used to evaluate the condition, and if the condition is true, then the "THEN" clause is executed. In this case, if the "ProductName" is 'Battery', then the "price" column is set to 1, and if the "ProductName" is 'Hat', then the "price" column is

The "WHERE" clause is used to specify the condition that must be met in order for the update to occur. In this case, the condition is that the "ProductName" must be either 'Battery' or 'Hat'. This ensures that only the rows with the specified product names are updated.

Requirement 8

Use secure passwords and authentication for access to the database.
(User:Group21 Password:Group21YYDS)

SQL Script

```
SPOOL C:\KXO206\Q8.txt
@@ requirement 8

create user c##Group21 identified by Group21YYDS;
grant connect,resource to c##Group21;
grant all on customer to c##Group21;
grant all on orders to c##Group21;
grant all on order_details to c##Group21;
grant all on product to c##Group21;
grant all on inventory to c##Group21;
grant all on transaction to c##Group21;
grant all on transaction_details to c##Group21;
grant all on transaction_details to c##Group21;
grant all on salary to c##Group21;
grant all on SHIFT to c##Group21;
grant all on STORE to c##Group21;


SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608

TERMINAL                  USED_ON                          USED_AT
------------------------------ ---------------------------------------- --------------------
LAPTOP-G0QGGGE8           24-MAY-2023                      9:41 P.M.

Requirement:8
```

```
SQL> create user c##Group21 identified by Group21YYDS;

User created.

SQL> grant connect,resource to c##Group21;

Grant succeeded.

SQL> grant all on customer to c##Group21;

Grant succeeded.

SQL> grant all on orders to c##Group21;

Grant succeeded.

SQL> grant all on order_details to c##Group21;

Grant succeeded.

SQL> grant all on product to c##Group21;

Grant succeeded.

SQL> grant all on inventory to c##Group21;

Grant succeeded.

SQL> grant all on transaction to c##Group21;

Grant succeeded.

SQL> grant all on transaction_details to c##Group21;

Grant succeeded.

SQL> grant all on transaction_details to c##Group21;

Grant succeeded.

SQL> grant all on salary to c##Group21;

Grant succeeded.

SQL> grant all on SHIFT to c##Group21;

Grant succeeded.

SQL> grant all on STORE to c##Group21;
```

Grant succeeded.

SQL> conn
Enter user-name: c##group21
Enter password:
Connected.

Comments

This code creates a user for the database named c##Group21 with the password "Group21YYDS". Then there is the empowerment of the user.

RESOURCE: Users with Resource rights can only create entities but cannot create database structures.

CONNECT: Users with the Connect permission can only log in to the Oracle database and cannot create entities or database structures.

DBA: has all privileges and is the highest permission of the system. Only the DBA can create the database structure.

For common users, only connect and resource permissions can be granted.

Next is the table that authorized users can view. In the project requirements, we set up eleven tables, using statement ‘grant all on tablename to c##Group21;’ to grant permissions on each of the eleven tables.

Requirement 9

Encrypt sensitive data such as price and phone number.

SQL Script

```
SPOOL C:\KXO206\Q9.txt
@@ requirement 9

ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "Group21YYDS";

ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "Group21YYDS";
ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "Group21YYDS";

ALTER TABLE CUSTOMER MODIFY (PHONENUMBER ENCRYPT);
ALTER TABLE CUSTOMER MODIFY (EMAIL ENCRYPT);
ALTER TABLE EMPLOYEE MODIFY (PHONENUMBER ENCRYPT);
ALTER TABLE EMPLOYEE MODIFY (EMAIL ENCRYPT);
ALTER TABLE SALARY MODIFY (SALARYAMOUNT ENCRYPT);
ALTER TABLE TRANSACTION MODIFY (TOTALAMOUNT ENCRYPT);
ALTER TABLE TRANSACTION_DETAILS MODIFY (PRICE ENCRYPT);
ALTER TABLE TRANSACTION_DETAILS MODIFY (SUBTOTAL ENCRYPT);
ALTER TABLE ORDERS MODIFY (TotalCost ENCRYPT);
ALTER TABLE ORDER_DETAILS MODIFY (PRICE ENCRYPT);
ALTER TABLE ORDER_DETAILS MODIFY (SUBTOTAL ENCRYPT);
ALTER TABLE PRODUCT MODIFY (PRICE ENCRYPT);
ALTER TABLE INVENTORY MODIFY (Quantity ENCRYPT);
ALTER TABLE INVENTORY MODIFY (ReorderPoint ENCRYPT);

ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "Group21YYDS";


SPOOL OFF
```

Results

Script output:

```
SQL> @@C:\KXO206\requirement.sql

USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608
```

```
TERMINAL                       USED_ON                              USED_AT
------------------------------ ------------------------------------ --------------------
LAPTOP-G0QGGGE8                25-MAY-2023                          5:13 A.M.
```

Requirement: 9

```
SQL> ALTER SYSTEM SET ENCRYPTION KEY IDENTIFIED BY "Group21YYDS";

System altered.

SQL> ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "Group21YYDS";

System altered.

SQL> ALTER SYSTEM SET ENCRYPTION WALLET OPEN IDENTIFIED BY "Group21YYDS";

System altered.

SQL> ALTER TABLE CUSTOMER MODIFY (PHONENUMBER ENCRYPT);

Table altered.

SQL> ALTER TABLE CUSTOMER MODIFY (EMAIL ENCRYPT);

Table altered.

SQL> ALTER TABLE EMPLOYEE MODIFY (PHONENUMBER ENCRYPT);

Table altered.

SQL> ALTER TABLE EMPLOYEE MODIFY (EMAIL ENCRYPT);

Table altered.

SQL>ALTER TABLE SALARY MODIFY (SALARYAMOUNT ENCRYPT);

Table altered.

SQL> ALTER TABLE TRANSACTION MODIFY (TOTALAMOUNT ENCRYPT);

Table altered.

SQL> ALTER TABLE TRANSACTION_DETAILS MODIFY (PRICE ENCRYPT);
```

```
Table altered.

SQL> ALTER TABLE TRANSACTION_DETAILS MODIFY (SUBTOTAL ENCRYPT);

Table altered.

SQL> ALTER TABLE ORDERS MODIFY (TotalCost ENCRYPT);

Table altered.

SQL> ALTER TABLE ORDER_DETAILS MODIFY (PRICE ENCRYPT);

Table altered.

SQL> ALTER TABLE ORDER_DETAILS MODIFY (SUBTOTAL ENCRYPT);

Table altered.

SQL> ALTER TABLE PRODUCT MODIFY (PRICE ENCRYPT);

Table altered.

SQL> ALTER TABLE INVENTORY MODIFY (Quantity ENCRYPT);

Table altered.

SQL> ALTER TABLE INVENTORY MODIFY (ReorderPoint ENCRYPT);

Table altered.

SQL> ALTER SYSTEM SET ENCRYPTION WALLET CLOSE IDENTIFIED BY "Group21YYDS";

System altered.


SQL> SPOOL OFF
```

Comments

The detailed purpose of this code is to encrypt the privacy related part of the table. First, we set the system encryption password, the password is "Group21YYDS". The wallet is then closed by the key to test whether the wallet is open, and the wallet is opened during subsequent encryption.

Next, we encrypt the private data in the different tables. Encrypt the Email and PhoneNumber columns in the customer table; Encrypt the Price and Subtotal columns in the

TRANSACTION_DETAILS table; Encrypts the TotalAmount column in the TRANSACTION table; Encrypt the Price and Subtotal columns in the ORDER_DETAILS table; Encrypt the TotalCost column in the ORDERS table; Encrypt the Quantity and RecorderPoint columns in the INVENTORY table.

Requirement 10

Using SQL statement, how would you modify the schema to track customer reviews and ratings for products?

SQL Script

```
SPOOL C:\KXO206\Q10.txt
@@ requirement 10

CREATE table customerFeedback(
CustomerFeedbackID NUMBER(4),
CustomerID NUMBER(4),
ProductID NUMBER(4),
Reviews VARCHAR2(20),
Ratings NUMBER(2),
CONSTRAINT customerFeedback_CustomerFeedbackID_pk PRIMARY KEY
(CustomerFeedbackID),
CONSTRAINT customerFeedback_CustomerID_fk FOREIGN KEY (CustomerID)
   REFERENCES CUSTOMER (CustomerID),
CONSTRAINT customerFeedback_ProductID_fk FOREIGN KEY (ProductID)
   REFERENCES PRODUCT (ProductID));

SPOOL OFF
```

Results

Script output:

```
USER          SERVER                    SESSION_ID
--------------- ------------------------------ ----------
SYSTEM        XE                         170608


TERMINAL                  USED_ON                          USED_AT
-------------------------------- ---------------------------------------- --------------------
LAPTOP-G0QGGGE8             24-MAY-2023                      12:37 P.M.


Requirement: 10


Table created.
```

Comments

To track customer reviews and ratings for products, a many-to-many relationship occurs between the Customer table and the product table. Thus, the many-to-many relationship needs to be converted into two one-to-many relationships by creating a table.

The table should have a relation with CUSTOMER and PRODUCT tables. Thus, this table need to include a primary key of CustomerFeedbackID and foreign keys of CustomerID to the CUSTOMER and ProductID of PRODUCT tables. Meanwhile, the table should contains information of reviews and ratings.

```
CREATE TABLE [ schema. ] tablename
 ( columnname datatype [ DEFAULT value ]
 [ , columnname datatype [ DEFAULT value ]] );
```

```
[CONSTRAINT constraintname] constrainttype (columnname, …),
```

According to the above rules, the script should create a table named "customerFeedback" with five columns: CustomerFeedbackID, CustomerID, ProductID, Reviews, and Ratings. It also includes three CONSTRAINT clauses to define the primary key and foreign key relationships for the table.

The CustomerFeedbackID, CustomerID and ProductID columns are defined as a NUMBER data type with a maximum of four digits. The Reviews column is defined as a VARCHAR2 data type with a maximum length of 20 characters. The Ratings column is defined as a NUMBER data type with a maximum of two digits.

In the create statement, constraints are identified in the table level by using the above statement. Primary Key constraint makes certain the columns identified as the table's primary key are unique. The script specifies the CustomerFeedbackID column is the primary key for the table. Foreign Key constraint are also created to enforce referential integrity with the CUSTOMER and PRODUCT tables.

Requirement 11

Describe how you would implement security measures to protect customer and employee data. Do not use SQL code here, just text.

1.Start with access control: Limiting access to sensitive data is a crucial aspect of data security. Implementing access control measures such as authentication, authorization, and accounting (AAA) can help ensure that only authorized users can access sensitive data.

2.Encrypt sensitive data: Encrypting sensitive information while storing it in the database or in transit can prevent unauthorized access to the data. Implementing encryption standards such as Advanced Encryption Standard (AES) can help protect the data from hackers.

3.Implement firewalls and intrusion detection systems (IDS): Firewalls and IDS help keep an eye on any potential intrusion into the database. Configure firewalls to allow only necessary traffic to the database, and IDS can identify any suspicious activity on the network.

4.Regularly update software: Make sure database software and frameworks are up to date to avoid potential security vulnerabilities that can be exploited by malicious actors.

5.Implement data backup and recovery procedures: Having a backup plan ensures availability of data in case of any data loss, corruption, or disasters. Implementing a disaster recovery plan can help the organization prevent data loss.

6.Train employees: Educate employees on data security best practices and how to follow security protocols when accessing, storing, and transmitting sensitive information.

Requirement 12

Using SQL statement show the structure and contents of each of the table you have create in the database.

SQL Script

```
SPOOL C:\KXO206\Q12.txt
@@ Requirement 12

DESC customer
SELECT * FROM customer;
DESC orders
SELECT * FROM orders;
DESC order_details
SELECT * FROM order_details;
DESC product
SELECT * FROM product;
DESC store
SELECT * FROM store;
DESC inventory
SELECT * FROM inventory;
DESC transaction
SELECT * FROM transaction;
DESC transaction_details
SELECT * FROM transaction_details;
DESC employee
SELECT * FROM employee;
DESC salary
SELECT * FROM salary;
DESC shift
SELECT * FROM shift;
DESC customerFeedback
SELECT * FROM customerFeedback;
DESC customerFeedback
SELECT * FROM customerFeedback;

SPOOL OFF
```

Results

Script output:

```
USER            SERVER                         SESSION_ID
--------------- ------------------------------ ----------
SYSTEM          XE                             470012

TERMINAL                       USED_ON                                 USED_AT
------------------------------ --------------------------------------- --------------------
DESKTOP-A3EL3T9                24-MAY-2023                             06:00 P.M.
```

Requirement: 12

```
 Name                                                                                    Null?    Type
 --------------------------------------------------------------------------------------- -------- ------------------------------
-------------------------
 CUSTOMERID                                                                              NOT NULL NUMBER(4)
 FIRSTNAME                                                                               NOT NULL VARCHAR2(10)
 LASTNAME                                                                                NOT NULL VARCHAR2(10)
 ADDRESS                                                                                          VARCHAR2(20)
 PHONENUMBER                                                                             NOT NULL VARCHAR2(15) ENCRYPT
 EMAIL                                                                                            VARCHAR2(20) ENCRYPT
 LOYALTYSTATUS                                                                                    NUMBER(10)
```

```
CUSTOMERID FIRSTNAME          LASTNAME           ADDRESS                                 PHONENUMBER
---------- -------------------- ------------------ --------------------------------------- ------------------------------
EMAIL                                    LOYALTYSTATUS
---------------------------------------- -------------
      1000 Kally              Smith              148 Main Street                         +86147365249
15426@123.com                                           5

      1001 Alice              Leila              82 Dirt Road                            +86137954943
alice@example.com                                       6

      1002 Lucas              Jake               114 East Savannah                       +86134567324
Lucas@123.com                                           8

      1003 Schell             Steve              1008 Grand Avenue                       +86139019237
Schell@123.com                                          4

      1004 Daum               Michell            9851231 Long Road                       +86139013478
Daum@123.com                                            10

      1005 Falah              Kenneth            456 Elm Street                          +86137827464
Falah@123.com                                           8

      1006 Perez              Jorge              123 Main Steet                          +86287426346
Perez@123.com                                           6
```

1007 Cruz Meshia 69821 South Avenue +86247268374 Cruz@123.com 7

1008 Smith Reese 123 Main Steet +86348276467 Smith@123.com 6

1009 Tom Thomas 123 Main Steet +86847563846 Tom@123.com 9

1010 Woai xuexiao +86346578910 woaix@data.com

11 rows selected.

Name	Null?	Type
ORDERID	NOT NULL	NUMBER(4)
ORDERDATE	NOT NULL	DATE
ORDERSTATUS		VARCHAR2(20)
TOTALCOST		NUMBER(5,2) ENCRYPT
CUSTOMERID		NUMBER(4)

ORDERID	ORDERDATE	ORDERSTATUS	TOTALCOST	CUSTOMERID
1000	01-JAN-23	shipped	12.5	1003
1001	09-MAY-23	shipped	50	1007
1002	15-JUN-23	unshipped	32.95	1003
1003	15-JUN-23	unshipped	88.4	1000
1004	29-JUN-23	shipped	45	1001
1005	12-JAN-23	unshipped	67.5	1002
1006	22-FEB-23	shipped	121.3	1004
1007	07-MAR-23	unshipped	59.4	1004
1008	09-MAY-23	shipped	250	1008
1009	16-MAR-23	shipped	85	1005
1010	27-MAY-23	shipped	250	1006
1011	21-AUG-23	unshipped	89.9	1009

12 rows selected.

Name	Null?	Type
ORDERDETAILID	NOT NULL	NUMBER(4)
ORDERID		NUMBER(4)
PRODUCTID		NUMBER(4)
PRODUCTNAME		VARCHAR2(20)

PRICE	NOT NULL NUMBER(5,2) ENCRYPT
QUANTITY	NOT NULL NUMBER(5)
SUBTOTAL	NUMBER(5,2) ENCRYPT

ORDERDETAILID	ORDERID	PRODUCTID	PRODUCTNAME	PRICE	QUANTITY	SUBTOTAL
1001	1003	1005	Tshirt	32.95	2	65.9
1002	1006	1004	Watch	40	2	80
1003	1006	1001	Shorts	20.65	2	41.3
1004	1002	1005	T-shirt	32.95	1	32.95
1005	1005	1002	Skirt	33.75	2	67.5
1006	1003	1007	Mouse Pat	22.5	1	22.5
1007	1004	1007	Mouse Pat	22.5	2	45
1008	1000	1008	Battery	2.5	5	12.5
1009	1001	1009	Hat	12.5	4	50
1010	1007	1008	Battery	19.8	3	59.4
1011	1008	1010	Phone	250	1	250
1012	1009	1006	Mouse	42.5	2	85
1013	1010	1001	Phone	250	1	250
1014	1011	1003	keyboard	89.9	1	89.9

14 rows selected.

Name	Null?	Type
PRODUCTID	NOT NULL	NUMBER(4)
PRODUCTNAME		VARCHAR2(20)
DESCRIPTION		VARCHAR2(20)
CATEGORY		VARCHAR2(20)
PRICE		NUMBER(5,2) ENCRYPT

PRODUCTID	PRODUCTNAME	DESCRIPTION	CATEGORY	PRICE
1001	Shorts	Clothing	Clothing	20.65
1002	Skirt	Clothing	Clothing	33.75
1003	Keyboard	Electronic	Electronic	89.9
1004	Watch	accessories	accessories	40
1005	T-shirt	Clothing	Clothing	32.95
1006	Mouse	Electronic	Electronic	42.5
1007	Mouse Pad	accessories	accessories	22.5
1008	Battery	accessories	accessories	1
1009	Hat	Clothing	Clothing	2
1010	Phone	Electronic	Electronic	250
1011	Qunzi	Clothing	Clothing	10

11 rows selected.

```
Name                                                                                Null?    Type
----------------------------------------------------------------------------------- -------- ----------------------------
-------------------------
STOREID                                                                             NOT NULL NUMBER(4)
STORENAME                                                                           VARCHAR2(20)
LOCATION                                                                            VARCHAR2(30)
OPERATIONHOURS                                                                      VARCHAR2(20)
```

STOREID	STORENAME	LOCATION	OPERATIONHOURS
1001	ABC store	Los Angeles	8a.m-16p.m
1002	AC store	New York City	12a.m-20p.m
1003	AB store	Miami	8a.m-16p.m
1004	BC store	Chicago	12a.m-20p.m
1005	C store	New Orleans	12a.m-20p.m
1006	A store	Seattle	12a.m-20p.m
1007	B store	San Francisco	8a.m-16p.m
1008	BC store	Boston	12a.m-20p.m
1009	ABC store	Nashville	8a.m-16p.m
1010	AC store	Austin	12a.m-20p.m

10 rows selected.

```
Name                                                                                Null?    Type
----------------------------------------------------------------------------------- -------- ----------------------------
-------------------------
PRODUCTID                                                                           NOT NULL NUMBER(4)
STOREID                                                                             NOT NULL NUMBER(4)
QUANTITY                                                                            NOT NULL NUMBER(5) ENCRYPT
REORDERPOINT                                                                        NOT NULL NUMBER(5) ENCRYPT
```

PRODUCTID	STOREID	QUANTITY	REORDERPOINT
1001	1001	100	60

1003	1001	50	30
1005	1001	80	40
1002	1002	78	35
1005	1002	45	40
1007	1002	112	80
1001	1003	102	70
1002	1003	48	35
1004	1003	43	35
1005	1004	65	60
1006	1004	72	60
1008	1005	65	40
1009	1005	52	35
1003	1006	68	60
1006	1006	62	75
1002	1007	168	80
1010	1007	97	80
1001	1008	58	40
1009	1008	67	40
1002	1009	68	35
1004	1009	47	35
1006	1010	77	50
1007	1010	113	85

23 rows selected.

Name	Null?	Type
TRANSACTIONID	NOT NULL	NUMBER(4)
TRANSACTIONDATE	NOT NULL	DATE
TRANSACTIONTYPE	NOT NULL	VARCHAR2(10)
TOTALAMOUNT	NOT NULL	NUMBER(10,2) ENCRYPT
CUSTOMERID		NUMBER(4)

TRANSACTIONID	TRANSACTIOND	TRANSACTIONTYPE	TOTALAMOUNT	CUSTOMERID
1000	02-JAN-23	payment	12.5	1003
1001	09-MAY-23	payment	50	1007
1002	18-JUN-23	credit	32.95	1003
1003	15-JUN-23	payment	88.4	1000
1004	30-JUN-23	credit	45	1001
1005	12-JAN-23	credit	67.5	1002
1006	24-FEB-23	credit	121.3	1004
1007	07-MAR-23	payment	59.4	1004
1008	09-MAY-23	payment	250	1008
1009	16-MAR-23	payment	85	1005
1010	28-MAY-23	credit	250	1006
1011	21-AUG-23	payment	89.9	1009

12 rows selected.

Name	Null?	Type
TRANSACTIONDETAILID	NOT NULL	NUMBER(10)
TRANSACTIONID		NUMBER(4)
PRODUCTID		NUMBER(4)
PRODUCTNAME		VARCHAR2(20)
PRICE	NOT NULL	NUMBER(5,2) ENCRYPT
QUANTITY	NOT NULL	NUMBER(5)
SUBTOTAL	NOT NULL	NUMBER(10,2) ENCRYPT

TRANSACTIONDETAILID	TRANSACTIONID	PRODUCTID	PRODUCTNAME	PRICE	QUANTITY	SUBTOTAL
1001	1003	1005	Tshirt	32.95	2	65.9
1002	1006	1004	Watch	40	3	80
1003	1006	1001	Shorts	20.65	2	41.3
1004	1002	1005	T-shirt	32.95	1	32.95
1005	1005	1002	Skirt	33.75	2	67.5
1006	1003	1007	Mouse Pat	22.5	1	22.5
1007	1004	1007	Mouse Pat	22.5	2	45
1008	1000	1008	Battery	2.5	5	12.5
1009	1001	1009	Hat	12.5	4	50
1010	1007	1008	Battery	19.8	3	59.4
1011	1008	1010	Phone	250	1	250
1012	1009	1006	Mouse	42.5	2	85
1013	1010	1001	Phone	250	1	250
1014	1011	1003	keyboard	89.9	1	89.9

14 rows selected.

Name	Null?	Type
EMPLOYEEID	NOT NULL	NUMBER(4)
NAME	NOT NULL	VARCHAR2(20)
ADDRESS		VARCHAR2(20)
PHONENUMBER		VARCHAR2(20) ENCRYPT
EMAIL		VARCHAR2(30) ENCRYPT
POSITION	NOT NULL	VARCHAR2(20)

EMPLOYEEID	NAME	ADDRESS	PHONENUMBER

```
EMAIL                                                    POSITION
-------------------------------------------------------- ----------------------------------------
     1001 John Doe                    123 Main St                    +8618721567890
johndoe@example.com                                  Manager

     1002 Jane Smith                  456 Oak Ave                    +8613862457893
janesmith@example.com                                Sales

     1003 Bob Johnson                 789 Elm St                     +8615824768905
bobjohnson@example.com                               Cashier

     1004 Sara Lee                    321 Pine St                    +8617896542310
saralee@example.com                                  Sales

     1005 Tom Smith                   654 Maple Ave                  +8615698745234
tomsmith@example.com                                 Stock Clerk

     1006 Amy Nguyen                  987 Cedar St                   +8618356978456
amynguyen@example.com                                Sales

     1007 Mike Johnson                246 Birch Rd                   +8614725869312
mikejohnson@example.com                              Cashier

     1008 Emily Chen                  135 Cherry St                  +8619387562147
emilychen@example.com                                Manager

     1009 David Lee                   864 Pine Ave                   +8617283490876
davidlee@example.com                                 Sales

     1010 Lisa Kim                    279 Oak Rd                     +8615642390785
lisakim@example.com                                  Cashier


10 rows selected.

 Name                                                                                  Null?    Type
 ----------------------------------------------------------------------------------------- -------- ----------------------------
----------------------------
 EMPLOYEEID                                                                            NOT NULL NUMBER(4)
 STARTDATE                                                                             NOT NULL DATE
 ENDDATE                                                                               DATE
 SALARYAMOUNT                                                                          NOT NULL NUMBER(10,2) ENCRYPT


EMPLOYEEID STARTDATE    ENDDATE     SALARYAMOUNT
---------- ------------ ------------ ------------
     1001 01-JAN-23    04-JAN-23          9000
     1002 04-JAN-23    07-JAN-23          5000
     1003 07-JAN-23    12-JAN-23          4000
```

```
      1004 02-JAN-23   11-JAN-23         5500
      1005 11-JAN-23   26-FEB-23         7500
      1006 26-FEB-23   26-MAR-23         5000
      1007 20-MAR-23   25-MAR-23         4000
      1008 06-APR-23   18-APR-23         10000
      1009 23-MAY-23   23-MAY-23         8500
      1010 15-MAY-23   20-MAY-23         7000

10 rows selected.

 Name                                                                                                Null?    Type
 ----------------------------------------------------------------------------------------------------- -------- ------------------------------
-------------------------
 SHIFTID                                                                                             NOT NULL NUMBER(4)
 EMPLOYEEID                                                                                          NUMBER(4)
 STOREID                                                                                             NUMBER(4)
 STARTTIME                                                                                           NOT NULL VARCHAR2(20)
 ENDTIME                                                                                             NOT NULL VARCHAR2(20)
 STORELOCATION                                                                                       VARCHAR2(30)


   SHIFTID EMPLOYEEID    STOREID STARTTIME                            ENDTIME
---------- ---------- ---------- ---------------------------------------- ----------------------------------------
STORELOCATION
------------------------------------------------------------
      1001       1005       1001 2023-01-01 08:00:00                  2023-01-01 16:00:00
Los Angeles

      1002       1004       1002 2023-01-11 12:00:00                  2023-01-11 20:00:00
New York City

      1003       1001       1003 2023-01-22 08:00:00                  2023-01-32 16:00:00
Miami

      1004       1003       1004 2023-02-02 12:00:00                  2023-02-02 20:00:00
Chicago

      1005       1002       1005 2023-02-12 12:00:00                  2023-02-12 20:00:00
New Orleans

      1006       1006       1006 2023-03-22 12:00:00                  2023-03-22 20:00:00
Seattle

      1007       1007       1007 2023-04-04 08:00:00                  2023-04-04 16:00:00
San Francisco

      1008       1010       1008 2023-04-24 12:00:00                  2023-04-24 20:00:00
Boston
```

```
    1009      1009      1009 2023-05-05 08:00:00              2023-05-15 16:00:00
Nashville

    1010      1008      1010 2023-05-15 12:00:00              2023-05-15 20:00:00
Austin


10 rows selected.

Name                                                          Null?    Type
 ----------------------------------------------------------------------------- -------- ------------------------------
-------------------------
 CUSTOMERFEEDBACKID                                           NOT NULL NUMBER(4)
 CUSTOMERID                                                   NUMBER(4)
 PRODUCTID                                                    NUMBER(4)
 REVIEWS                                                      VARCHAR2(20)
 RATINGS                                                      NUMBER(2)

no rows selected.
```

Comments

The statement "DESC tablename" is used to describe the structure of the table in a database. "DESC" is short for "describe", and when followed by the name of a table, it returns information about the table's columns, data types, and constraints. This statement can be useful when you want to get a quick overview of the structure of a table, especially if it has a large number of columns.

The statement "SELECT * FROM tablename" is used to retrieve all data from the table in a database. "SELECT" is used to retrieve data from one or more tables in a database, and the "*" symbol is used as a wildcard to select all columns from a table. This query will return a result set containing all the data from the table.

General Note: The output to the screen for all requirements should be formatted so that it is professional in appearance, easy to read, and should include meaningful column names.

Documentation

You are required to document all work done in a formal business report prepared in accordance with normal professional business practice

The report is to have a one-page introduction identifying what has been achieved, and what (if anything) remains to be done. Each specific requirement is to be documented on a new page, and should have the requirement number as the major heading, with sub-headings for (1) the SQL script, (2) the query results, and (3) comments (as appropriate).

Each requirement is to be constructed through an SQL script with the following typical structure:

Sample Structure :

```
SPOOL C:\206\Q1.txt
@@requirement 1
SELECT * FROM TableName;
SPOOL OFF
```

This letter corresponds to the drive letter where your scripts are located, and the output text file will be stored. Substitute this

Substitute this number with the number of the requirement n you are running.

The script requirement.sql accepts the number as a parameter

The file requirement.sql can be downloaded from the Assessment page on the unit's MyLO site. The file should be stored on the same drive, and same directory, as should your answer scripts to each requirement. It (requirement.sql) ***must*** be run at the start of each requirement answer (and provided with the appropriate requirement number) in order to identify you as the user, plus the date/time and location of your test run, plus identify which requirement you are attempting to meet. Your output for each requirement should be spooled to a file (here assumed to be on 'C: drive', but you can use whatever drive you wish). The content of these spool files should be listed as part of your report. The execution of the requirement.sql script results in the following typical introduction to each spool file:

```
   USER            SERVER
SESSION_ID
   =============== ==============================
==========
   A4              XE                                330349


   TERMINAL         USED_ON     USED_AT
   ================ =========== ==========
   SANDERSON        11-MAY-2017 11:17 P.M.


   Requirement: 1
```

Some of the requirements might prove difficult, so if you encounter a problem, provide the scripts that are partial solutions, on the basis that something is better than nothing. Your report's introduction should highlight any such problems.

Note: Each requirement will be accessed from the point of view: "Does the solution meet the requirements, and how well does it do it?" Where there is some inconsistency in that the solution and documentation does not completely match every sub-criterion within a particular criterion, then the grade reflects the value of the work 'on average'.

Students are reminded of the fundamental principle of Information Systems - that **quality is meeting the customer's requirements**.

Submission of Assignments

In submitting your assignment, you are agreeing that you have read the "Plagiarism and Academic Integrity" section below, and that your assignment submission complies with the assignment requirement that it is your own work.

Your completed solution must be submitted with an Assignment Cover Sheet by the deadline shown in the unit outline. Assignments must be submitted electronically via MyLO (http://mylo.utas.edu.au) as a Microsoft Word document or as an unrestricted pdf file. Details of the actual submission procedure are available through a notice on the unit's MyLO pages.

Submission :

Your assignment must be submitted through MyLO. No other form of submission is acceptable. No eMail or hard copy submissions are acceptable.

Students must take responsibility for the correct submission of their assignments. Students are expected to adhere to the following procedure for submission:

- Once submitted to MyLO, submitted file/s MUST be checked by the student to ensure that correct submission of the file has been undertaken.
- Students are expected to notify the Lecturer WITHIN 12 HOURS of submission if their files have not been submitted correctly.

Students must take responsibility for safely backing up of their own files during the academic year to ensure that no files are permanently lost.

Extensions will only be granted under exceptional conditions and must be requested with adequate notice on the School's official Request for Extension form.

Penalties

Please refer to the unit outline.

Plagiarism and Academic Integrity

While students are encouraged to discuss the assignments in this unit and to engage in active learning from each other, it is important that they are also aware of the University's policy on plagiarism.

Plagiarism is taking and using someone else's thoughts, writings or inventions and representing them as your own; for example downloading an essay wholly or in part from the internet, copying another student's work or using an author's words or ideas without citing the source.

> Plagiarism is a form of cheating. It is taking and using someone else's thoughts, writings or inventions and representing them as your own; for example, using an author's words without putting them in quotation marks and citing the source, using an author's ideas without proper acknowledgment and citation or copying another student's work.
>
> If you have any doubts about how to refer to the work of others in your assignments, please consult your lecturer or tutor for relevant referencing guidelines, and the academic integrity resources on the web at: http://www.academicintegrity.utas.edu.au/
>
> The intentional copying of someone else's work as one's own is a serious offence punishable by penalties that may range from a fine or deduction/cancellation of marks and, in the most serious of cases, to exclusion from a unit, a course or the University. Details of penalties that can be imposed are available in the Ordinance of Student Discipline – Part 3 Academic Misconduct, see: http://www.utas.edu.au/universitycouncil/legislation/
>
> **The University reserves the right to submit assignments to plagiarism detection software, and might then retain a copy of the assignment on its database for the purpose of future plagiarism checking.**

It is important that you understand this statement on plagiarism. Should you require clarification please see your unit coordinator or lecturer. Useful resources on academic integrity, including what it is and how to maintain it, are also available at:
http://www.academicintegrity.utas.edu.au/