**SQL case study 3**

**1. Display the count of customers in each region who have done the transaction in the year 2020.**

**SELECT c.region\_id, COUNT(DISTINCT c.customer\_id) AS CustomerCount**

**FROM Customers c I**

**NNER JOIN Transaction t ON c.customer\_id = t.customer\_id**

**WHERE YEAR(t.txn\_date) = 2020**

**GROUP BY c.region\_id;**

**2. Display the maximum and minimum transaction amount of each transaction type.**

**Select txn\_type , max(txn\_amount) as max\_amt, min(txn\_amount) as min\_amount**

**From transaction group by txn\_type ;**

**3. Display the customer id, region name and transaction amount where transaction type is deposit and transaction amount > 2000.**

**Select c.customer\_id, co.region\_name, t.txn\_amount**

**From customer c inner join transaction**

**On c.customer\_id = t.customer\_id**

**Inner join continent co**

**On c.region\_id = co.region\_id**

**Where t.txn\_type ='deposit’ and t.txn\_amount >2000;**

**4. Find duplicate records in the Customer table.**

**Select customer\_id, count (customer\_id) as duplicate\_count**

**From customers**

**Group by customer\_id having**

**count(customer\_id)>1;**

**5. Display the customer id, region name, transaction type and transaction amount for the minimum transaction amount in deposit.**

**Select c.customer\_id, co.region\_name, t.txn\_type, t.txn\_amount**

**From customers c inner join transaction t**

**On c.customer\_id =t.customer\_id inner join continent co**

**On c.region\_id = co.region\_id**

**Where t.txn\_type =’deposit’ and t.txn\_amount = (select min(txn\_ampunt) from transaction where txn\_type =’deposit’);**

**6. Create a stored procedure to display details of customers in the Transaction table where the transaction date is greater than Jun 2020.**

**Create procedure cust\_transactiondate**

**As**

**Begin**

**set Nocount on ;**

**Select c.customer\_id, c.region\_id, c.start\_date, c.end\_date**

**From customers c**

**Inner join transaction t**

**On c.customer\_id =t.customer\_id**

**Where t.txn\_date >’2020-06-30’;**

**End;**

**Exec cust\_transactiondate ;**

**7. Create a stored procedure to insert a record in the Continent table.**

**Create procedure continent\_record @region\_id int , @region\_name nvarchar(255)**

**As**

**Begin**

**Set nocount on ;**

**Insert into continent (region\_id, region\_name) values (@region\_id,@region\_name);**

**End;**

**Exec continent\_record @region\_id = 1, @region\_name =’new continent’**

**8. Create a stored procedure to display the details of transactions that happened on a specific day.**

**Create procedure transact\_date @targetdate date**

**As**

**Begin**

**Set oncount on ;**

**Select \* from transaction whereconvert(date , txn\_date) = @targetsate;**

**End;**

**Exec transact\_date @targetdate =’2023-09-25’ ;**

**9. Create a user defined function to add 10% of the transaction amount in a table.**

**Create function add10percent (amount decimal(18,2))**

**Returns**

**decimal(18,2)**

**As begin declare @newamount decimal(18,2);**

**Set @newamount =@amount + (@amount \*0.10 );**

**Return @newamount;**

**End;**

**Select dbo.add10percent (100.00);**

**10. Create a user defined function to find the total transaction amount for a given transaction type.**

**Create function total\_transaction\_amount (@txntype nvarchar(255))**

**Returns decimal (18,2)**

**As begin declare @totalamount decimal (18,2);**

**Select @totalamount= sum(txn\_amount)**

**From transaction where txn\_type =@txntype;**

**Return @totalamount;**

**End;**

**Select dbo.total\_transaction\_amount(‘deposit’);**

**11. Create a table value function which comprises the columns customer\_id, region\_id ,txn\_date , txn\_type , txn\_amount which will retrieve data from the above table.**

**CREATE FUNCTION GetTransactionData()**

**RETURNS TABLE**

**AS**

**RETURN (**

**SELECT**

**c.customer\_id,**

**c.region\_id,**

**t.txn\_date,**

**t.txn\_type,**

**t.txn\_amount**

**FROM**

**Customers c**

**INNER JOIN**

**Transaction t ON c.customer\_id = t.customer\_id**

**);**

**Select \* from dbo.GetTransactionData();**

**12. Create a TRY...CATCH block to print a region id and region name in a single column.**

**Begin try**

**Declare @regionid int;**

**Declare @regionname nvarchar(255);**

**Select concat(cast (@regionid as nvarchar(10)),’-’,@regionname)**

**As regionidandname;**

**End try**

**Begin catch**

**Print 'an error occurred’ + error\_message();**

**End**

**Catch;**

**13. Create a TRY...CATCH block to insert a value in the Continent table.**

**Begin try**

**Insert into continent(region\_id,region\_name) values(1,’asia’);**

**Print ‘value inserted successfully’ ;**

**End**

**Try begin catch**

**Print ‘an error occurred ’ + error\_message ();**

**End catch;**

**14. Create a trigger to prevent deleting a table in a database.**

**Create trigger preventtabledeletion on**

**Datbase for drop\_table**

**As begin set**

**Nocount on ;**

**Declar @eventdata xml;**

**Set @eventdata = eventdata();**

**If @eventdat.value (‘(/event\_instance/objecttype )[1]’,’nvarchar(max)’) =’table’**

**Begin raiseerror**

**(‘table delection is not allowed’,16,1);**

**Rollback;**

**End**

**;**

**15. Create a trigger to audit the data in a table.**

**-- Create a table to store audit data**

**CREATE TABLE AuditLog (**

**AuditLogID INT IDENTITY(1,1) PRIMARY KEY,**

**TableName NVARCHAR(255),**

**AuditDate DATETIME,**

**ActionType NVARCHAR(10),**

**OldData NVARCHAR(MAX),**

**NewData NVARCHAR(MAX)**

**);**

**-- Create a trigger to capture data changes**

**CREATE TRIGGER AuditYourTable**

**ON YourTable**

**AFTER INSERT, UPDATE, DELETE**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**DECLARE @AuditDate DATETIME = GETDATE();**

**DECLARE @ActionType NVARCHAR(10);**

**DECLARE @OldData NVARCHAR(MAX);**

**DECLARE @NewData NVARCHAR(MAX);**

**IF EXISTS (SELECT \* FROM INSERTED) AND EXISTS (SELECT \* FROM DELETED)**

**BEGIN**

**SET @ActionType = 'UPDATE';**

**-- Capture old and new data for updates**

**SELECT @OldData = CONVERT(NVARCHAR(MAX), DELETED.\*) FROM DELETED;**

**SELECT @NewData = CONVERT(NVARCHAR(MAX), INSERTED.\*) FROM INSERTED;**

**END**

**ELSE IF EXISTS (SELECT \* FROM INSERTED)**

**BEGIN**

**SET @ActionType = 'INSERT';**

**-- Capture new data for inserts**

**SELECT @NewData = CONVERT(NVARCHAR(MAX), INSERTED.\*) FROM INSERTED;**

**END**

**ELSE IF EXISTS (SELECT \* FROM DELETED)**

**BEGIN**

**SET @ActionType = 'DELETE';**

**-- Capture old data for deletes**

**SELECT @OldData = CONVERT(NVARCHAR(MAX), DELETED.\*) FROM DELETED;**

**END**

**-- Insert audit record into AuditLog table**

**INSERT INTO AuditLog (TableName, AuditDate, ActionType, OldData, NewData)**

**VALUES ('YourTable', @AuditDate, @ActionType, @OldData, @NewData);**

**END;**

**17. Display top n customers on the basis of transaction type.**

**WITH CustomerTransactionCounts AS (**

**SELECT c.customer\_id, c.customer\_name, COUNT(t.transaction\_id) AS TransactionCount FROM Customers c**

**INNER JOIN Transactions t**

**ON c.customer\_id = t.customer\_id**

**WHERE t.transaction\_type = 'Purchase' c.customer\_id, c.customer\_name )**

**SELECT customer\_id, customer\_name, TransactionCount**

**FROM CustomerTransactionCounts**

**ORDER BY TransactionCount DESC LIMIT N;**

**18. Create a pivot table to display the total purchase, withdrawal and deposit for all the customers.**

**SELECT \* FROM (**

**SELECT customer\_id, transaction\_type, transaction\_amount FROM Transactions )**

**AS SourceTable PIVOT**

**( SUM(transaction\_amount) FOR transaction\_type IN ([Purchase], [Withdrawal], [Deposit]) )**

**AS PivotTable;**