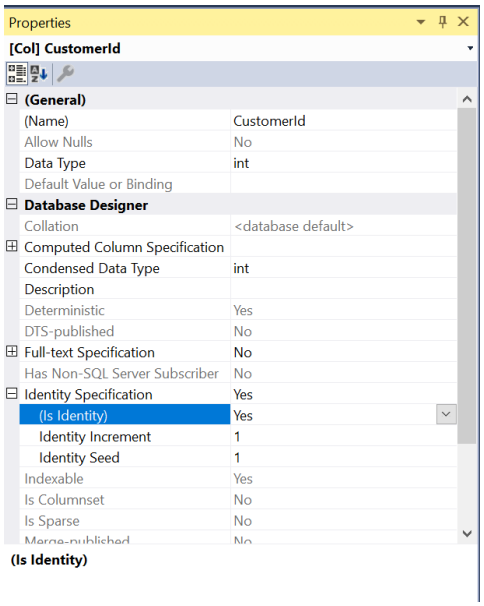
BTH3_GIẢI

BẢNG Customer:

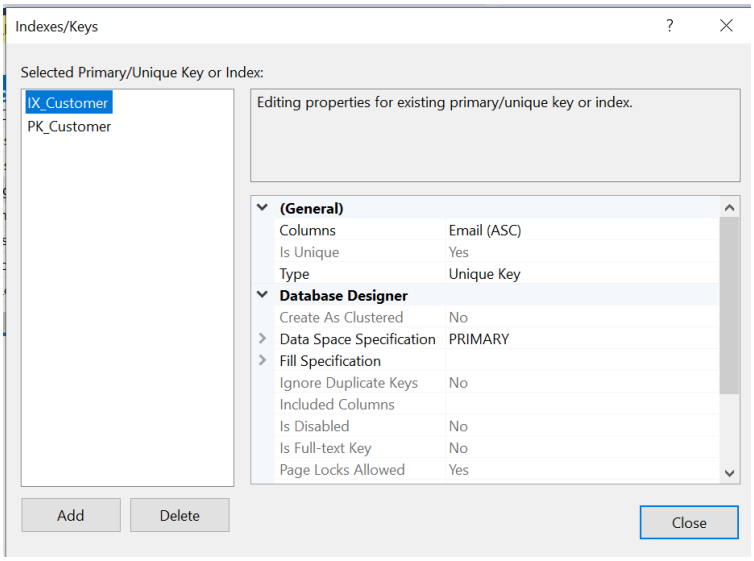
- Cột CustomerId: set tính Identity thành YES
- Nếu cần **Reset giá trị Identity trong SQL Server về mặc định (***Trong **SQL Server** khi bạn đặt giá trị ID là tự động tăng và sau khi xóa toàn bộ dữ liệu thì nó vẫn cứ tiếp tục tăng giá trị tiếp theo mà không quay lại bắt đầu từ 1. Ví dụ bảng student của mình có 10 giá trị với ID từ 1 đến 10, khi mình xóa toàn bộ dữ liệu 10 bản ghi đó đi và bắt đầu thêm mới, thì ID tiếp theo sẽ là 11. Vậy để đưa giá trị tiếp theo mà bạn muốn về bất kì số nào bạn muốn thì dùng lệnh sau.)*

> 💡 Execute lệnh: DBCC CHECKIDENT('Tên_bảng', RESEED, 0)

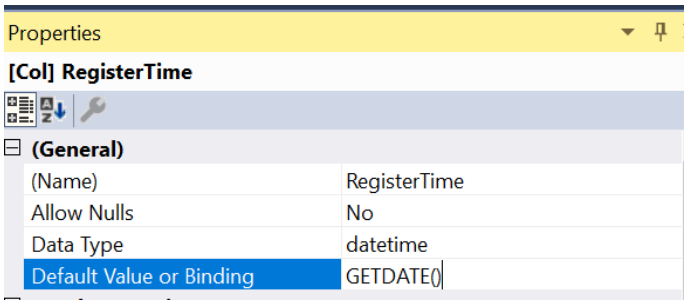


- Cột Email: set Unique Key (các dòng không được trùng email) cho nó

Chuột phải bảng —> **Indexes/Keys** —> tại **Indexes/Keys** dialog box, chọn **Add** —> tại grid dưới **General,** chọn **Type** và chọn **Unique Key** từ drop-down list box, đổi tên Key thành: UK_Tên_bảng và **Close**.



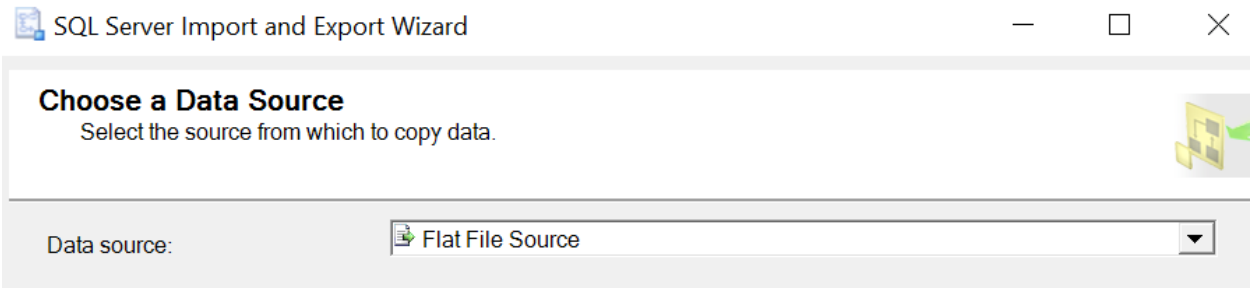
- Cột RegisterTime: set tính Default:



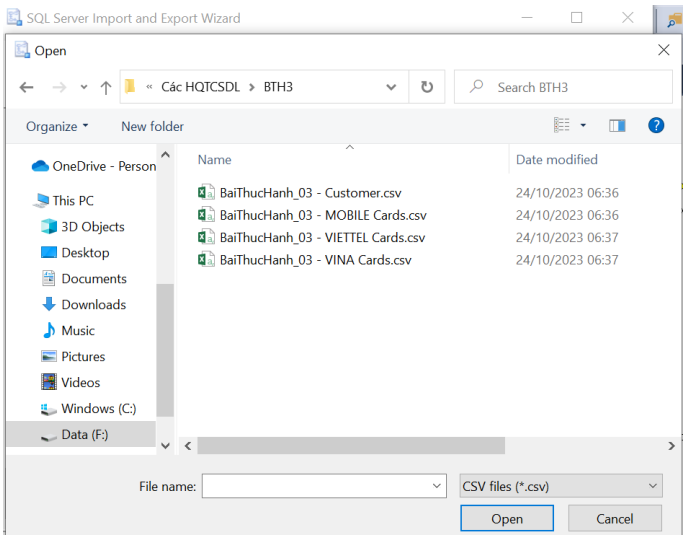
- Cột IsLocked: set tính Default là 0
- Tương tự như vậy đối với các cột khác

IMPORT DATA:

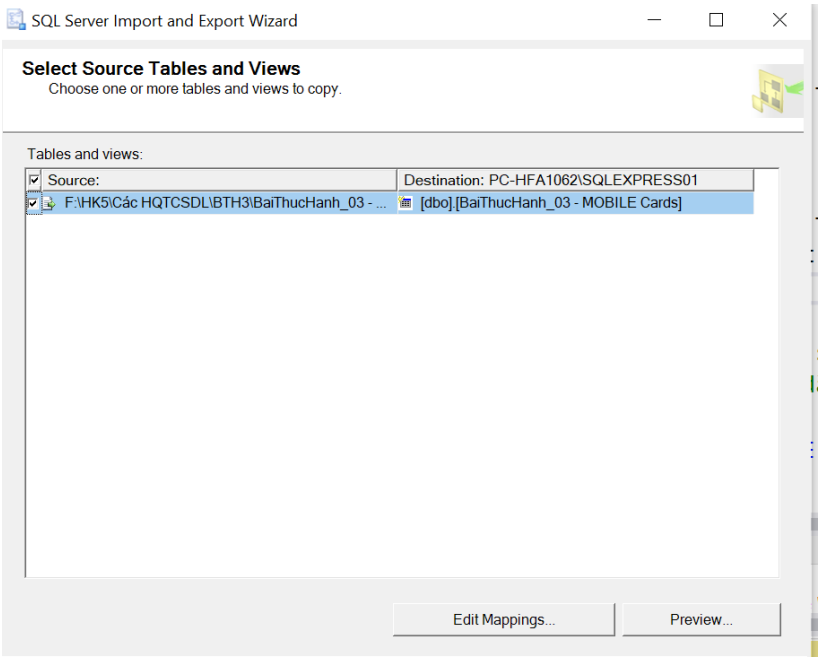
làm như bình thường, đối với đầu vào dạng CSV, ở Data soure, chọn Flat File Sourse



Lúc chọn file, nhớ đổi loại file cần chọn thành CSV Files



Chú ý đổi kiểu data khi import ở Edit Mappings




```
insert into Customer (CustomerName, RegisterTime, Email, Password, MobiNumber, IsLocked)
select distinct t.CustomerName, t.RegisterTime, t.Email, t.Password, t.MobiNumber, t.IsLocked
from [BaiThucHanh_03 - Customer] as t

insert into CardStore (CardTypeId, Serial, PinNumber, Amount, ExpiredTime, CardStatus, InvoiceId)
select distinct 1, t.Serial, t.PinNumber, t.Amount, t.ExpiredTime, 1, NULL
from [BaiThucHanh_03 - MOBILE Cards] as t
WHERE NOT EXISTS (SELECT * FROM CardStore WHERE Serial = t.Serial AND CardTypeId = 1)

insert into CardStore (CardTypeId, Serial, PinNumber, Amount, ExpiredTime, CardStatus, InvoiceId)
select distinct 2, t.Serial, t.PinNumber, t.Amount, t.ExpiredTime, 1, NULL
from [BaiThucHanh_03 - VINA Cards] as t
WHERE NOT EXISTS (SELECT * FROM CardStore WHERE Serial = t.Serial AND CardTypeId = 2)

insert into CardStore (CardTypeId, Serial, PinNumber, Amount, ExpiredTime, CardStatus, InvoiceId)
select distinct 3, t.Serial, t.PinNumber, t.Amount, t.ExpiredTime, 1, NULL
```

```
from [BaiThucHanh_03 - VIETTEL Cards] as t
WHERE NOT EXISTS (SELECT * FROM CardStore WHERE Serial = t.Serial AND CardTypeId = 3)
GO
```

Câu 1:

```
-- Cau 1: kiểm tra @MobiNumber có phải là số di động hợp lệ hay không. Trong đó qui định số di động
-- được xem là hợp lệ nếu là chuỗi có độ dài từ 9 đến 12 ký tự và chỉ chứa các con số (NOT LIKE
-- '%[^0-9]%').
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'fn_IsValidMobiNumber')
DROP FUNCTION fn_IsValidMobiNumber;
GO
CREATE FUNCTION fn_IsValidMobiNumber
(
@MobiNumber nvarchar(50)
)
RETURNS bit
AS
BEGIN
  DECLARE @test bit;
  IF (LEN(@MobiNumber) BETWEEN 9 AND 12) AND (@MobiNumber NOT LIKE '%[^0-9]%')
    RETURN 1;
  RETURN 0;
END
GO

-- Test
PRINT dbo.fn_IsValidMobiNumber('012366665')
```

Câu 2:

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'proc_Customer_Register')
DROP PROCEDURE proc_Customer_Register
GO
CREATE PROCEDURE proc_Customer_Register
(
  @CustomerName nvarchar(100),
  @Email nvarchar(100),
  @Password nvarchar(100),
  @MobiNumber nvarchar(50),
  @CustomerId int OUTPUT
)
AS
BEGIN
  SET NOCOUNT ON;
  IF (@Email IS NULL or @Email = N'')
    BEGIN
      SET @CustomerId = -1;
      RETURN;
    END
  IF EXISTS(SELECT * FROM Customer WHERE Email = @Email)
    BEGIN
      SET @CustomerId = -2;
      RETURN;
    END
  IF ((@Password IS NULL) OR (LEN(@Password) < 6))
    BEGIN
      SET @CustomerId = -3;
      RETURN;
    END
  IF (@CustomerName IS NULL) OR (@CustomerName = N'')
    BEGIN
      SET @CustomerId = -4;
      RETURN;
    END
  IF (dbo.fn_IsValidMobiNumber(@MobiNumber) = 0)
    BEGIN
```

```
      SET @CustomerId = -5;
      RETURN;
    END

  INSERT INTO Customer (CustomerName, Email, Password, MobiNumber)
    VALUES (@CustomerName, @Email, @Password, @MobiNumber)

  SET @CustomerId = @@IDENTITY

END
GO

DECLARE @CustomerId int;

EXECUTE proc_Customer_Register @CustomerName =  N'Nhu Y',
  @Email =  N'nhuy@gmail.com',
  @Password =  N'123456',
  @MobiNumber =  N'0123456789',
  @CustomerId = @CustomerId output;
PRINT @CustomerId
go
```

Câu 3

```
IF EXISTS (SELECT * FROM sys.objects
WHERE name = 'proc_Customer_ChangePassword')
DROP PROCEDURE proc_Customer_ChangePassword
GO

CREATE PROCEDURE proc_Customer_ChangePassword
(
  @Email nvarchar(100),
  @OldPassword nvarchar(100),
  @NewPassword nvarchar(100),
  @Result int OUTPUT
)
AS
BEGIN
  SET NOCOUNT ON;
    UPDATE Customer SET Password = @NewPassword WHERE Email = @Email AND Password = @OldPassword;

  IF(@@ROWCOUNT > 0)
    SET @Result = 1
  ELSE
    SET @Result = 0;
END
GO

DECLARE @Result int;
EXECUTE proc_Customer_ChangePassword @Email = N'nhuy@gmail.com',
               @OldPassword = N'123456',
               @NewPassword = N'1234567',
               @Result = @Result output;
PRINT @Result;
go
```

Câu 4

```
IF EXISTS (SELECT * FROM sys.objects
WHERE name = 'proc_Customer_Authenticate')
DROP PROCEDURE proc_Customer_Authenticate
GO

CREATE PROCEDURE proc_Customer_Authenticate
(
  @Email nvarchar(100),
  @Password nvarchar(100)
```

```
)
AS
BEGIN
  SET NOCOUNT ON;
  SELECT CustomerID, CustomerName, RegisterTime, Email, MobiNumber, IsLocked
  FROM Customer
  WHERE Email = @Email
END
GO

EXECUTE proc_Customer_Authenticate @Email = N'nhuy@gmail.com',
          @Password = N'1234567';
GO
```

Câu 5

```
IF EXISTS (SELECT * FROM sys.objects
WHERE name = 'proc_Customer_Update')
DROP PROCEDURE proc_Customer_Update
GO

CREATE PROCEDURE proc_Customer_Update
  @CustomerId int,
  @CustomerName nvarchar(100),
  @Email nvarchar(100),
  @MobileNumber nvarchar(50),
  @Result int OUTPUT
AS
BEGIN
  SET NOCOUNT ON;
  -- -1 : Nếu Email rỗng
   IF @Email IS NULL or @Email = N''
    BEGIN
      SET @Result = -1;
      RETURN;
    END
  -- -2 : Nếu Email trùng với Email của tài khoản khác.
    IF EXISTS (SELECT * FROM Customer WHERE Email = @Email AND CustomerID <> @CustomerId)
    BEGIN
      SET @Result = -2;
      RETURN;
    END
  -- -4 : Nếu tên của khách hàng rỗng.
    IF @CustomerName IS NULL or @CustomerName = N''
    BEGIN
      SET @Result = -4;
      RETURN;
    END
  -- -5 : Nếu số di động không hợp lệ.
    IF (dbo.fn_IsValidMobiNumber(@MobileNumber) = 0)
    BEGIN
      SET @Result = -5;
      RETURN;
    END

    UPDATE Customer
    SET CustomerName = @CustomerName, Email = @Email, MobiNumber = @MobileNumber
    WHERE CustomerID = @CustomerId

    IF @@ROWCOUNT > 0
    SET @Result = 1
    ELSE
    SET @Result = 0;
END
GO

DECLARE @Result int;
EXECUTE proc_Customer_Update @CustomerId = N'1001',
               @CustomerName = N'Nhu Y Tran',
               @Email = N'nhuytran@gmail.com',
```

```
                @MobileNumber = N'0132456789',
                @Result = @Result OUTPUT
PRINT @Result;
go

Select *
from Customer
where CustomerId = 1001
```

Câu 6

💡 “ROW_NUMBER() OVER(ORDER BY OrderDate)” nghĩa là gán một con số tăng tuần tự cho mỗi bản ghi theo thứ tự của OrderDate, bắt đầu từ 1

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'proc_Customer_Select')
DROP PROCEDURE proc_Customer_Select
GO

CREATE PROCEDURE proc_Customer_Select
(
  @Page int = 0,
  @PageSize int = 0,
  @SearchValue nvarchar(255) = N'',
  @RowCount int OUTPUT,
  @PageCount int OUTPUT
)
AS
BEGIN
  SET NOCOUNT ON;

  SELECT *, ROW_NUMBER() over(order by CustomerName) as RowNumber
  INTO #TempCustomer
  FROM Customer

  SELECT @RowCount = COUNT(*)
  FROM Customer
  where (@SearchValue = N'') or (CustomerName like @SearchValue);

  IF(@PageSize = 0)
    SET @PageCount = 1
  ELSE
    BEGIN
      SET @PageCount = @RowCount / @PageSize;
      IF (@RowCount % @PageSize > 0)
        SET @PageCount += 1;
    END;


  ;WITH cte as
  (
    SELECT * from #TempCustomer
    where (@SearchValue = N'') or (CustomerName like @SearchValue)
  )
  SELECT * FROM cte
  where (@PageSize = 0) or
      RowNumber between (@Page - 1) * @PageSize + 1 AND @Page * @PageSize
  ORDER BY RowNumber
END
GO

--TEST
DECLARE @RowCount int,
    @PageCount int
EXECUTE proc_Customer_Select
  @Page = 1,
  @PageSize = 10,
  @SearchValue = N'',
```

```
  @RowCount = @RowCount OUTPUT,
  @PageCount = @PageCount OUTPUT
GO
```

Câu 7:

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'proc_Invoice_Select')
DROP PROCEDURE proc_Invoice_Select
GO

CREATE PROCEDURE proc_Invoice_Select
(
  @Page int = 1,
  @PageSize int = 0,
  @CustomerId int = 0,
  @FromTime datetime = null,
  @ToTime datetime = null,
  @RowCount int OUTPUT,
  @PageCount int OUTPUT
)
AS
BEGIN
  SET NOCOUNT ON;
  SELECT Invoice.InvoiceId, Customer.CustomerId, CustomerName, CreatedTime, InvoiceStatus,
    FinishedTime, SUM(CardStore.Amount) AS TotalAmount, ROW_NUMBER() over(order by Invoice.InvoiceId) as RowNumber
  INTO #TempInvoices
  FROM Invoice JOIN CardStore ON Invoice.InvoiceId = CardStore.InvoiceId
  JOIN Customer ON Invoice.CustomerId = Customer.CustomerId
  GROUP BY Invoice.InvoiceId, Customer.CustomerId, CustomerName, CreatedTime, InvoiceStatus, FinishedTime

  -- @RowCount : Tham số đầu ra cho biết tổng số dòng dữ liệu tìm được
  SELECT @RowCount = COUNT(*)
  FROM Customer
  where (@CustomerId = N'') or (CustomerId = @CustomerId);

  IF(@PageSize = 0)
    SET @PageCount = 1
  ELSE
    BEGIN
    -- @PageCount : Tham số đầu ra cho biết tổng số trang.
      SET @PageCount = @RowCount / @PageSize;
      IF (@RowCount % @PageSize > 0)
        SET @PageCount += 1;
    END;
  ;WITH cte
  as
  (
    SELECT * from #TempInvoices
    where (@CustomerId = N'') or (CustomerId = @CustomerId)
  )
  SELECT * FROM cte
  -- @PageSize : Số dòng trên mỗi trang (nếu @PageSize = 0 thì hiển thị toàn bộ dữ liệu tìm được)
  where (@PageSize = 0) or
      RowNumber between (@Page - 1) * @PageSize + 1 AND @Page * @PageSize
  ORDER BY RowNumber
END
GO

--TEST
DECLARE @RowCount int,
    @PageCount int
EXECUTE proc_Invoice_Select
  @Page = 1,
  @PageSize = 10,
  @CustomerId = 3,
  @FromTime = null,
  @ToTime = null,
  @RowCount = @RowCount OUTPUT,
```

```
  @PageCount = @PageCount OUTPUT
GO
```

Câu 8

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'fn_GetInventories')
  DROP FUNCTION fn_GetInventories;
GO
CREATE FUNCTION fn_GetInventories
(
  @CardTypeId int,
  @Amount money
)
RETURNS int
AS
BEGIN
     DECLARE @slthe int;
         SELECT @slthe = COUNT(*)
     FROM CardStore
     WHERE CardTypeId = @CardTypeId AND Amount = @Amount AND CardStatus = 1;
     RETURN @slthe
END
GO

-- Test
PRINT dbo.fn_GetInventories(1, 50000)
```

Câu 9

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'fn_GetCardStatistics')
  DROP FUNCTION fn_GetCardStatistics;
GO
CREATE FUNCTION fn_GetCardStatistics ()
RETURNS @tbl TABLE
( IDThe int,
  LoaiThe nvarchar(50),
  MenhGia money,
  TongSlThe int,
  SlDaBan int,
  SlTon int,
  SlBiKhoa int
)
AS
BEGIN
  INSERT INTO @tbl (IDThe, LoaiThe, MenhGia, TongSlThe, SlDaBan, SlTon, SlBiKhoa)
        SELECT cs.CardTypeId, ct.CardTypeName as 'Loại thẻ', Amount as 'Mệnh giá',  COUNT(cs.CardTypeId) as 'Tổng số lượng thẻ',
                              SUM(CASE WHEN CardStatus = 0 THEN 1 ELSE 0 END) AS 'Số lượng thẻ đã bán',
                              SUM(CASE WHEN CardStatus = 1 THEN 1 ELSE 0 END) AS 'Số lượng thẻ còn tồn',
                              SUM(CASE WHEN CardStatus = -1 THEN 1 ELSE 0 END) AS 'Số lượng thẻ bị khóa hoặc hết hạn'
    FROM CardStore as cs JOIN CardType as ct ON cs.CardTypeId = ct.CardTypeId
        GROUP BY cs.CardTypeId, CardTypeName, Amount
    ORDER BY cs.CardTypeId;
        RETURN;
END
GO
SELECT * FROM dbo.fn_GetCardStatistics()
ORDER BY IDThe;
```

Câu 10

```
/* Thêm dữ liệu để Test:
INSERT INTO Invoice (CustomerID, CreatedTime, InvoiceStatus)
values (1, '2023-10-28', 1),
  (2, '2023-10-28', 1),
```

```
  (3, '2023-10-23', 1),
  (4, '2023-10-30', 1)

  UPDATE CardStore
  SET CardStatus = 0, InvoiceId = 1
  WHERE Serial = 'MB00000006' --50000

  UPDATE CardStore
  SET CardStatus = 0, InvoiceId = 1
  WHERE Serial = 'MB00000007' --50000

  UPDATE CardStore
  SET CardStatus = 0, InvoiceId = 2
  WHERE Serial = 'VN00000001' --50000

  UPDATE CardStore
  SET CardStatus = 0, InvoiceId = 3 --10000
  WHERE Serial = 'VN00000707'

  UPDATE CardStore
  SET CardStatus = 0, InvoiceId = 4 --50000
  WHERE Serial = 'VT00000096'
 GO*/

IF EXISTS (SELECT * FROM sys.objects WHERE name = 'fn_GetRevenueByDate')
  DROP FUNCTION fn_GetRevenueByDate;
GO
CREATE FUNCTION fn_GetRevenueByDate
(
  @CardTypeId int = 0,
  @FromTime datetime,
  @ToTime datetime
)
RETURNS @tblDoanhThu TABLE
( Ngay datetime,
  DoanhThu money
)
AS
BEGIN
  DECLARE @tblNgay Table
  (
     ngay date
  )
  DECLARE @tmp date = DATEFROMPARTS(YEAR(@FromTime), MONTH(@FromTime), DAY(@ToTime))
  WHILE (DATEDIFF(dd, @tmp, @ToTime) != 0)
  BEGIN
    insert into @tblNgay values (@tmp);
    set @tmp = DATEADD(day, 1, @tmp);
  END
  -- Nếu tham số @CardTypeId = 0 thì thống kê doanh thu của tất cả loại thẻ
  IF (@CardTypeId = 0)
    BEGIN
      INSERT @tblDoanhThu (Ngay, DoanhThu)
        SELECT t1.ngay, ISNULL(t2.DoanhThu, 0) as Revenue
        FROM @tblNgay AS t1
        LEFT JOIN
        (
          SELECT CreatedTime, SUM(Amount) as DoanhThu
          FROM CardStore JOIN Invoice ON CardStore.InvoiceId = Invoice.InvoiceId
          WHERE CreatedTime >= @FromTime  AND CreatedTime <= @ToTime
          GROUP BY CreatedTime, CardStore.CardTypeId
        ) AS t2
        ON t1.ngay = t2.CreatedTime
    END
  ELSE
    BEGIN
      INSERT @tblDoanhThu (Ngay, DoanhThu)
        SELECT t1.ngay, ISNULL(t2.DoanhThu, 0) as Revenue
        FROM @tblNgay AS t1
        LEFT JOIN
        (
          SELECT CreatedTime, SUM(Amount) as DoanhThu
          FROM CardStore JOIN Invoice ON CardStore.InvoiceId = Invoice.InvoiceId
```

```
          WHERE CardStore.CardTypeId = @CardTypeId AND CreatedTime >= @FromTime  AND CreatedTime <= @ToTime
          GROUP BY CreatedTime, CardStore.CardTypeId
        ) AS t2
        ON t1.ngay = t2.CreatedTime
    END
  RETURN;
END
GO

SELECT Ngay AS N'Ngày', DoanhThu as N'Tổng doanh thu'
FROM dbo.fn_GetRevenueByDate(0, '2023-10-28', '2023-11-01')
```

Câu 11

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'fn_GetRevenueByDateAndAmount')
  DROP FUNCTION fn_GetRevenueByDateAndAmount;
GO
CREATE FUNCTION fn_GetRevenueByDateAndAmount
(
  @CardTypeId int = 0,
  @FromTime datetime,
  @ToTime datetime
)
RETURNS @tblDoanhThu TABLE
( Ngay datetime,
  a50k money,
  a100k money,
  a200k money,
  a500k money,
  Tong money
)
AS
BEGIN
  DECLARE @tblNgay Table
  (
     ngay date
  )
  DECLARE @tmp date = DATEFROMPARTS(YEAR(@FromTime), MONTH(@FromTime), DAY(@ToTime))
  WHILE (DATEDIFF(dd, @tmp, @ToTime) != 0)
  BEGIN
    insert into @tblNgay values (@tmp);
    set @tmp = DATEADD(day, 1, @tmp);
  END

  IF (@CardTypeId = 0)
  BEGIN
    INSERT INTO @tblDoanhThu (Ngay, a50k, a100k, a200k, a500k, Tong)
      SELECT t1.ngay, ISNULL(t2.b50k, 0), ISNULL(t2.b100k, 0), ISNULL(t2.b200k, 0), ISNULL(t2.b500k, 0), ISNULL(t2.TongDoanhThu, 0)
      FROM @tblNgay AS t1
    LEFT JOIN
      (SELECT
      CreatedTime AS Ngay,
      SUM(CASE WHEN Amount = 50000 THEN Amount ELSE 0 END) AS b50k,
      SUM(CASE WHEN Amount = 100000 THEN Amount ELSE 0 END) AS b100k,
      SUM(CASE WHEN Amount = 200000 THEN Amount ELSE 0 END) AS b200k,
      SUM(CASE WHEN Amount = 500000 THEN Amount ELSE 0 END) AS b500k,
      SUM(Amount) AS TongDoanhThu
    FROM CardStore JOIN Invoice ON CardStore.InvoiceId = Invoice.InvoiceId
    WHERE CreatedTime >= @FromTime  AND CreatedTime <= @ToTime
    GROUP BY CreatedTime, CardStore.CardTypeId) as t2
    ON t1.ngay = t2.Ngay
  END
  ELSE
  BEGIN
    INSERT INTO @tblDoanhThu (Ngay, a50k, a100k, a200k, a500k, Tong)
      SELECT t1.ngay, ISNULL(t2.b50k, 0), ISNULL(t2.b100k, 0), ISNULL(t2.b200k, 0), ISNULL(t2.b500k, 0), ISNULL(t2.TongDoanhThu, 0)
      FROM @tblNgay AS t1
    LEFT JOIN
      (SELECT
      CreatedTime AS Ngay,
```

```
      SUM(CASE WHEN Amount = 50000 THEN Amount ELSE 0 END) AS b50k,
      SUM(CASE WHEN Amount = 100000 THEN Amount ELSE 0 END) AS b100k,
      SUM(CASE WHEN Amount = 200000 THEN Amount ELSE 0 END) AS b200k,
      SUM(CASE WHEN Amount = 500000 THEN Amount ELSE 0 END) AS b500k,
      SUM(Amount) AS TongDoanhThu
    FROM CardStore JOIN Invoice ON CardStore.InvoiceId = Invoice.InvoiceId
    WHERE CardStore.CardTypeId = @CardTypeId AND CreatedTime >= @FromTime  AND CreatedTime <= @ToTime
    GROUP BY CreatedTime, CardStore.CardTypeId) as t2
    ON t1.ngay = t2.Ngay
  END
  RETURN;
END
GO
SELECT Ngay as N'Ngày', t.a50k as '50k', t.a100k as '100k', t.a200k as '200k', t.a500k as '500k', t.Tong as N'Tổng doanh thu'
FROM dbo.fn_GetRevenueByDateAndAmount(0, '2023-10-28', '2023-11-01') as t
```

Câu 12

```
IF NOT EXISTS (SELECT * FROM sys.types WHERE name = 'TypeInvoiceDetails')
  BEGIN
    create type TypeInvoiceDetails as table
    (
      CardTypeId int,
      Amount money,
      Quantity int,
      primary key (CardTypeId, Amount)
    )
  END
GO

IF EXISTS (SELECT * FROM sys.objects WHERE name = 'proc_Invoice_Init')
DROP PROCEDURE proc_Invoice_Init
GO
CREATE PROCEDURE proc_Invoice_Init
    @CustomerId int,
    @InvoiceDetails TypeInvoiceDetails readonly,
    @InvoiceId bigint output
AS
BEGIN
  SET NOCOUNT ON;
  -- Kiểm soát đầu vào
  DECLARE @IsLocked bit;
  SELECT @IsLocked = IsLocked FROM Customer WHERE CustomerId = @CustomerId;
    -- -101 : Khách hàng không tồn tại
  IF(@IsLocked IS NULL)
    BEGIN
      SET @InvoiceId = -101;
      RETURN;
    END
    -- -102 : Tài khoản khách hàng bị khóa
  IF(@IsLocked = 1)
    BEGIN
      SET @InvoiceId = -102;
      RETURN;
    END

    -- 103 : Loại thẻ cần mua không tồn tại
    IF EXISTS (SELECT * FROM @InvoiceDetails AS inv
        WHERE CardTypeID NOT IN (SELECT CardTypeId FROM CardType))
    BEGIN
        SET @InvoiceId = -103;
        RETURN;
    END

    -- Kiểm tra số lượng thẻ cần mua
  -- -104 : Số lượng thẻ cần mua không hợp lệ
    IF EXISTS (SELECT * FROM @InvoiceDetails inv WHERE inv.Quantity <= 0)
    BEGIN
        SET @InvoiceId = -104;
        RETURN;
```

```
    END

  -- Lập đơn hàng
  -- Insert dữ liệu cho bảng Invoice
  INSERT INTO Invoice(CustomerId, InvoiceStatus, CreatedTime, FinishedTime)
  values (@CustomerId, 0, GETDATE(), NULL)

  SET @InvoiceId = @@IDENTITY --SCOPE_IDENTITY();

  -- Update dữ liệu của bảng CardStore: CardStatus và InvoiceID
  -- CardTypeID     Amount     Quantity
  ----------------------------------------
  --     1          100000        5
  --     1          200000        10
  --     2          100000        5
  DECLARE contro cursor for
    select CardTypeId, Amount, Quantity from @InvoiceDetails;
  OPEN contro

  DECLARE @CardTypeId int,
      @Amount money,
      @Quantity int,
      @isFailed bit = 0;
  FETCH NEXT FROM contro into @CardTypeId, @Amount, @Quantity;
  WHILE @@FETCH_STATUS = 0
    BEGIN
      -- Giữ thẻ cho đơn hàng
      UPDATE TOP(@Quantity) CardStore
      SET CardStatus = 0,
        InvoiceId = @InvoiceId
      WHERE CardTypeId = @CardTypeId
        AND Amount = @Amount
        AND CardStatus = 1
        AND ExpiredTime > DATEADD(HOUR, 1, GETDATE()); -- Chú ý điều kiện này phải theo chính sách

      IF(@@ROWCOUNT <> @Quantity)
        BEGIN
          SET @isFailed = 1;
          BREAK;
        END
      FETCH NEXT FROM contro into @CardTypeId, @Amount, @Quantity;
    END

  CLOSE contro;
  deallocate contro;

  IF(@isFailed = 1)
    BEGIN
    -- Trả lại thẻ đã tạm giữ
      UPDATE CardStore
      SET CardStatus = 1,
        InvoiceId = NULL
      WHERE InvoiceId = @InvoiceId;

      SET @InvoiceId = -105;
      RETURN;
    END

END
GO

DECLARE @InvoiceId bigint;
DECLARE @InvoiceDetails TypeInvoiceDetails;

INSERT INTO @InvoiceDetails (CardTypeId, Amount, Quantity)
VALUES (1, 50000, 5), (2, 100000, 3), (3, 20000, 90000)

EXECUTE proc_Invoice_Init @CustomerId = 15,
            @InvoiceDetails = @InvoiceDetails,
            @InvoiceId = @InvoiceId output;
SELECT @InvoiceId
IF(@InvoiceId > 0)
```

```
  SELECT * FROM CardStore WHERE InvoiceId = @InvoiceId
GO
```

```
IF EXISTS (SELECT * FROM sys.objects WHERE name = 'proc_Invoice_Finish')
DROP PROCEDURE proc_Invoice_Finish
GO
CREATE PROCEDURE proc_Invoice_Finish
        @InvoiceId bigint,
    @ResponseCode int
AS
BEGIN
        SET NOCOUNT ON;
    UPDATE Invoice
    SET InvoiceStatus = @ResponseCode
    WHERE InvoiceId = @InvoiceId;

    /*ResponseCode          Ý nghĩa
        1         Khách hàng thanh toán thành công*/
    IF(@ResponseCode = 1)
      BEGIN
        SELECT ct.CardTypeName, cs.Serial, cs.PinNumber, cs.Amount, cs.ExpiredTime
        FROM CardStore as cs join CardType as ct on cs.CardTypeId = ct.CardTypeId
        WHERE cs.InvoiceId = @InvoiceId
        ORDER BY ct.CardTypeName, cs.Amount
        RETURN;
      END
    ELSE
      BEGIN
        UPDATE CardStore
        SET CardStatus = 1,
          InvoiceId = NULL
        WHERE InvoiceId = @InvoiceId
      END
END
GO

EXECUTE proc_Invoice_Finish @InvoiceId = 1,
             @ResponseCode = 1
GO
```