```
-- Ziyan Yuan homework day4
use northwind
-- problem1
go
CREATE VIEW view_product_order_yuan AS
SELECT p.ProductName, SUM(od.Quantity) AS TotalOrderedQuantity
FROM
   Products p
   JOIN [Order Details] od ON p.ProductID = od.ProductID
GROUP BY
   p.ProductName;
go
-- problem2
go
CREATE PROCEDURE sp_product_order_quantity_yuan
   @ProductID INT,
   @TotalQuantity INT OUT
AS
BEGIN
   SELECT @TotalQuantity = SUM(od.Quantity)
   FROM [Order Details] od
   WHERE od.ProductID = @ProductID
END
go
-- search quantity when productid is 11
DECLARE @TotalQuantity int
EXEC sp_product_order_quantity_yuan @ProductID = 11, @TotalQuantity = @TotalQuantity
OUT
SELECT @TotalQuantity as [total quantity]
-- problem3
-- Here I try to use a table as a output but it comes out some syntax errors.
CREATE PROCEDURE sp_product_order_city_yuan
   @productName NVARCHAR(50)
AS
BEGIN
   SELECT TOP 5 o.ShipCity AS City, SUM(od.Quantity) AS TotalQuantity
```

```
FROM Orders o
   JOIN [Order Details] od ON o.OrderID = od.OrderID
   JOIN Products p ON od.ProductID = p.ProductID
   WHERE p.ProductName = @productName
   GROUP BY o.ShipCity
   ORDER BY TotalQuantity DESC
END
go
-- search quantity when productName is 'Chang'
EXEC sp_product_order_city_yuan @productName = 'Chang'
-- problem4
Begin tran
   CREATE TABLE city_yuan (
       Id INT PRIMARY KEY,
       City VARCHAR(50) NOT NULL
   );
    CREATE TABLE people_yuan (
       Id INT PRIMARY KEY,
       Name VARCHAR(50) NOT NULL,
       CityId INT NOT NULL,
       FOREIGN KEY (CityId) REFERENCES city_yuan(Id)
   );
   -- Insert records to these two tables
   INSERT INTO city_yuan (Id, City) VALUES
   (1, 'Seattle'),
   (2, 'Green Bay');
   INSERT INTO people_yuan (Id, Name, CityId) VALUES
   (1, 'Aaron Rodgers', 2),
   (2, 'Russell Wilson', 1),
   (3, 'Jody Nelson', 2);
    -- check if anyone lives in seattle
    SELECT * FROM people_yuan WHERE CityId = 1
    BEGIN
        -- move them from seattle to madison with a new Cityid = 3
```

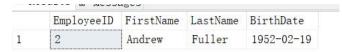
```
INSERT INTO city_yuan (Id, City) VALUES (3, 'Madison');
       UPDATE people_yuan SET CityId = 3 WHERE CityId = 1;
       DELETE FROM city_yuan WHERE Id = 1;
   END
    go
   CREATE VIEW Packers yuan AS
   SELECT Name FROM people_yuan p
   JOIN city_yuan c ON p.CityId = c.Id
   WHERE c.City = 'Green Bay';
    go
    Select * From people_yuan
    Select * From city_yuan
    Select * From Packers_yuan
rollback
end
-- drop these table and view
Drop VIEW [dbo].[Packers yuan];
Drop TABLE [dbo].[people_yuan];
Drop TABLE [dbo].[city_yuan];
-- problem5
go
CREATE PROCEDURE sp birthday employees yuan
AS
BEGIN
   -- Create a new table
   CREATE TABLE birthday_employees_yuan (
       EmployeeID INT,
       FirstName VARCHAR(50),
       LastName VARCHAR(50),
       BirthDate DATE
   );
   -- Insert employees with a birthday on Feb into the new table
   INSERT INTO birthday_employees_yuan (EmployeeID, FirstName, LastName, BirthDate)
   SELECT EmployeeID, FirstName, LastName, BirthDate
   FROM Employees
```

```
WHERE MONTH(BirthDate) = 2;
END;

EXEC sp_birthday_employees_yuan;
SELECT * FROM birthday_employees_yuan;
DROP TABLE birthday_employees_yuan;
```

Go

Screenshot:



```
-- problem6
```

So here we use qureies and except.

```
Eg: select * From table 1
    Except
    select * From table 2
```

Here we can get values from table1 except table 2, so if the result table is empty, table 1 is a subset of table 2.

Then we do:

select * From table 2
Except

select * From table 1
Likely, here we can get values from tab

Likely, here we can get values from table2 except table 1, so if the result table is empty, table 2 is a subset of table 1.

- 1. if these two table are empty, table 1&2 are the same
- 2. if the first tables is empty but second table not, table 1 is a subset of table 2.
- 3. if the second table is empty but first table not, table 2 is a subset of table 1.
- 4. if these two tables are not empty, table 1&2 has different values respectively. We can use union and except to look these same values.