MARMARA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CSE3055
DATABASE SYSTEMS
PRJOJECT STEP 3

Number	First Name	Last Name
150119566	Müslim	Yılmaz
150119037	Ömer	Kibar
150119036	Serkan	Korkut

PROJECT NAME: UPS_DELIVERY

Introduction

The aim of this project is designing and creating database system related the production and selling phase of the Inform Technology. Our project supports the web-interface and implementing in MS SQL.

Entities and Definitions

- **Employee:** Contains employee information. It has 2 disjoint subtypes.
- **BlueCollarEmployee:** Subtype of employee. keeps employees working directly in production.
- **WhiteCollarEmployee:** Subtype of employee. This entity is for the employees who will work in the office.
- **Department:** An entity used for department information.
- **Factory:** An entity belonging to the factory where products are made by assembly.
- **Customer:** It contains the information of the buyer who wants to buy the manufactured product.
- **UPSProduct:** An entity that holds the manufactured UPS product and their common properties.
- **ONE_ONE_PHASE:** Subtype of UPSProduct. Includes information about 2 ac wires UPS.
- **THREE_ONE_PHASE:** Subtype of UPSProduct. Includes information about 4 ac wires UPS.
- **THREE_THREE_PHASE:** Subtype of UPSProduct. Includes information about 6 ac wires UPS.
- **SupplierCompany:** An entity for the company that sells materials used in assembly.
- **Item:** It contains the parts used in assembly and information about them.
- **SupplyRecord:** An associative entity for supplying parts from the supplier to the factory.

Business Processes and Definitions

Uninterruptible power supply (UPS shortly) is an electrical apparatus that provides emergency power to a load when the input power source or mains power fails. One of the biggest producer company of UPS in the Turkey is Inform Technology. The company has services and several products related to high electronic systems.

Products:

There are many high-level electronical products in the production part of the company. We consider three of them which is related to uninterruptible power supply types which are:

- 1 Phase in 1 Phase Out UPS
- 3 Phase in 1 Phase Out UPS
- 3 Phase in 3 Phase Out UPS

In electricity, the phase refers to the distribution of a load. Single-phase power is a two-wire alternating current (ac) power circuit. There is one power wire and one neutral wire, with current flowing between the power wire and the neutral wire.[1]

Residential homes are usually served by a single-phase in single phase out UPS, while commercial and industrial facilities usually use a three-phase in three-phase out UPS. Addition to that, three phase in one phase out is intermediate form between the other two UPS.

For each type of UPS mentioned above, there are a difference in features and prices. In each type, there are several sub-type UPS. This difference exists due to industrial features between them. Among these features, **conversion technology**, **energy saving mode**, frequency converter hertz, **environment friendly status** and **amount of voltage** it provides are the most important ones.

[1] https://www.fluke.com/en-us/learn/blog/power-quality/single-phase-vs-three-phase-power

Business Rules and Restrictions

- Before the starting design phase, requirement analysis will be done. In this analysis, we will provide information about business processes, functional, non-functional requirements, and general implementation.
- Several meetings will be done before each project phase. Also, detailed report will be submitted in each phase.
- There are many different parts in company. In project design we will focus on production and sales part.

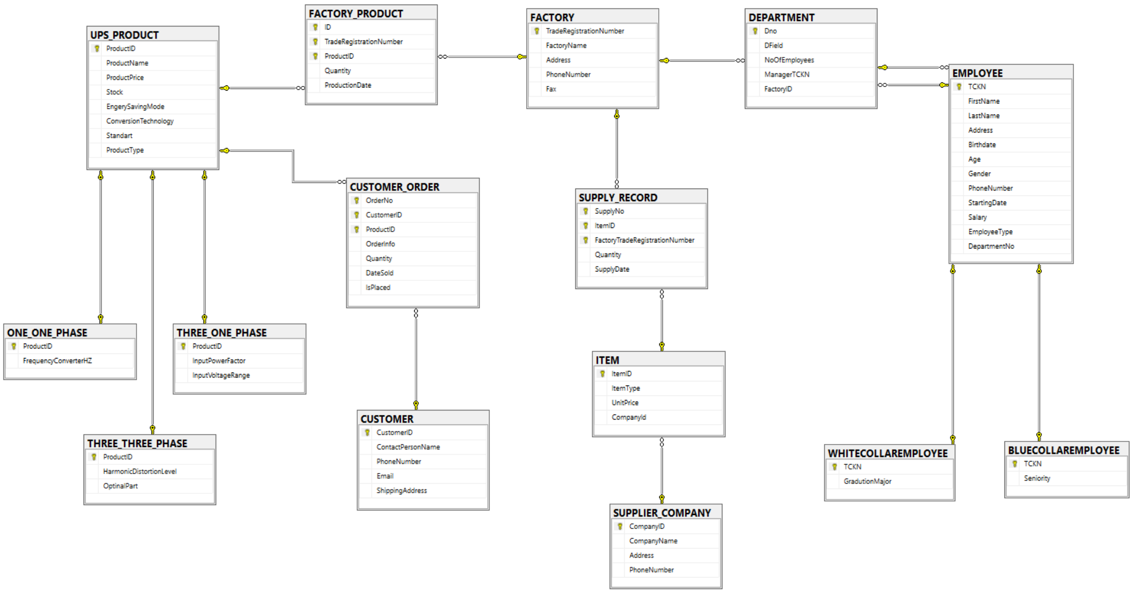
Functional Requirements

- Supplier company supplies items to factory and these supplies should be recorded. Each product needs different items to be produced in the factory.
- Customers can order product. If the product is in the stock, the order will be placed immediately otherwise order will wait until the product is produced.
- Employees should be classified as blue collar and white-collar employees. Blue collar employees have seniority attribute that is used when calculating new salary each year. White collar employees will work in different departments based on their university majors.
- Each employee should work in a different department.
- Each employee should be supervised by another employee.
- Each department should have a manager.

Non-functional Requirements

- Database system should be implemented by using Microsoft SQL Server.
- A website should be deployed and using the website customer should interact with the database and perform CRUD operations on each table.
- An entity relation diagram that clearly shows all the entities and diagrams should be provided.
- Requirement analysis document that contains entities and their definitions, business processes, business rules, functional nonfunctional requirements, should be provided.
- Database system should include views, triggers, and stored procedures

DATABASE DIAGRAM



TABLES

■ EMPLOYEE

- **Definition**
  - Employee table holds information about employees in the working factory. Employee entity is a supertype of tables called 'BLUECOLLAREMPLOYEE' and 'WHITECOLLAREMPLOYEE.' Age is computed automatically with birthdate field.
- **Columns**
  - **TCKN:** varchar (11)
  - **FirstName:** nvarchar (20)
  - **LastName:** nvarchar (20)
  - **Address:** nvarchar (60)
  - **Birthdate:** Date
  - **Age:** int
  - **Gende**r: char (1)
  - **PhoneNumber:** varchar (14) UNIQUE
  - **StartingDate:** Date
  - **Salary:** float DEFAULT 0
  - **EmployeeType:** char (1)
  - **DepartmentNo:** smallint
- **Primary key:**
  - TCKN
- **Foreign key:**
  -
- **Indexes:**
  - INDEX name_asc (LastName ASC)
  - INDEX salary (Salary DESC)
- **Others:**
  - PhoneNumber – **unique**, Age – **computed**, salary **– default**

■ BLUECOLLAREMPLOYEE

- **Definition**
  - This table holds TCKN and seniority information about the employees who called as 'blue collar.' Blue-collar worker is a working-class person who performs manual labor. Seniority column holds information about how many years he/she has been working in the factory.
- **Columns**
  - **TCKN:** varchar (11)
  - **Seniority:** smallint
- **Primary key:**
  - TCKN
- **Foreign key:**
  - TCKN
- **Indexes:**
  -
- **Others:**
  - seniority – **default**,
- **Trigger:** This table automatically filled by the trigger. **TCKN** column retrieved from **TCKN** field and **seniority** column retrieved from **StartingDate** field which both exist in the **EMPLOYEE** table.

WHITECOLLAREMPLOYEE

- **Definition**
  - This table holds TCKN and graduation information about the employees who called as 'white collar.' White-collar is a person who performs professional, desk work.
- **Columns**
  - **TCKN:** varchar (11)
  - **GraduationMajor:** nvarchar (75)
- **Primary key:**
  - TCKN
- **Foreign key:**
  - TCKN
- **Indexes:**
  -
- **Others:**
  -

DEPARTMENT

- **Definition**
  - This table holds department information in the factory. NoOfEmployees is calculated automatically by the trigger.
- **Columns**
  - **Dno:** smallint
  - **DField:** nvarchar (50)
  - **NoOfEmployees:** int
  - **ManagerTCKN:** varchar (11)
  - **FactoryID:** varchar (11)
- **Primary key:**
  - Dno
- **Foreign key:**
  - FactoryID
- **Indexes:**
  - dno_employees (Dno, NoOfEmployees)
  - manager_list(ManagerTCKN, Dfield)
- **Others:**
  - NoOfEmployees – **default**
- **Trigger:**
  - When an employee is added into the EMPLOYEE table, the number of people in the relevant department is increased by one according to the DepartmentNo in the EMPLOYEE table.
  - When a department deleted; employees in the deleted department automatically deleted from EMPLOYEE table.

■ <u>FACTORY</u>

- **Definition**
  - This table holds factory information. In this table we have just one field. It has been created due to the possible opening of new factories in the future.
- **Columns**
  - **TradeRegistrationNumber:** varchar (11)
  - **FactoryName:** nvarchar (100)
  - **Address:** nvarchar (100)
  - **PhoneNumber:** varchar (14)
  - **Fax:** varchar (12)
- **Primary key:**
  - TradeRegistrationNumber
- **Foreign key:**
  -
- **Indexes:**
  -
- **Others:**
  - PhoneNumber -- **unique**, Fax – **unique**

■ SUPPLIER_COMPANY

- **Definition**
  - This table holds information about company which supplies to item to factory.
- **Columns**
  - **CompanyID:** nvarchar (11)
  - **CompanyName:** nvarchar (35)
  - **Address:** nvarchar (60)
  - **PhoneNumber:** varchar (14)
- **Primary key:**
  - CompanyID
- **Foreign key:**
  -
- **Indexes:**
  -
- **Others:**
  - PhoneNumber - **Unique**

■ ITEM

- **Definition**
  - This table holds information about items which produces in supplier company.
- **Columns**
  - **ItemID:** varchar (11)
  - **ItemType:** nvarchar (150)
  - **UnitPrice:** float
  - **CompanyId:** nvarchar (11)
- **Primary key:**
  - ItemID
- **Foreign key:**
  - CompanyID
- **Indexes:**
  - Itemprice (ItemType, UnitPrice DESC)
- **Others:**
  - check_contraints → (UnitPrice >= 0.5)

■ SUPPLY_RECORD

- **Definition**
- This table holds records between supply company and items they supplied to factory.
- **Columns**
  - **SupplyNo:** int
  - **ItemId:** nvarchar (11)
  - **FactoryTradeRegistrationNumber:** varchar (11)
  - **Quantity:** int
  - **SupplyDate:** Date
- **Primary key:**
  - SupplyNo
  - ItemID
  - FactoryTradeRegistrationNumber
- **Foreign key:**
  - ItemID
  - FactoryTradeRegistrationNumber
- **Indexes:**

  -
- **Others:**

  -

■ CUSTOMER

- **Definition**
  - This table holds information about customer who purchase a UPS from factory. Customer might be person or a company.
- **Columns**
  - **CustomerID:** varchar (11) PRIMARY KEY
  - **ContactPersonName:** nvarchar (20)
  - **PhoneNumber**: varchar (14)
  - **Email:** nvarchar (35)
  - **ShippingAdress:** nvarchar(60)
- **Primary key:**
  - CustomerID
- **Foreign key:**
  -
- **Indexes:**
  -
- **Others:**
  - **check_contraints** → (Email LIKE ‘%@%’.com)
  - PhoneNumber - **Unique**

■ UPS_PRODUCT

- **Definition**
  - This table holds information about UPS which product by factory with several types. This table is supertype ‘**ONE_ONE_PHASE’**, **’THREE_ONE_PHASE’** and’ **THREE_THREE_PHASE’** tables. ProductType column hold information about what is the ups type.
- **Columns**
  - **ProductID:** varchar (20) PRIMARY KEY
  - **ProductName:** nvarchar (100)
  - **ProductPrice:** float
  - **Stock:** int
  - **EnergySavingMode**: nvarchar (15)
  - **ConversionTechnology:** nvarchar (30)
  - **Standard:** nvarchar (60)
  - **ProductType:** nvarchar (3)
- **Primary key:**
  - ProductID
- **Foreign key:**
  -
- **Indexes:**
  - Product_stock (ProductID, Stock)
  - Product_price (ProductName, ProductPrice DESC)
- **Others:**
  - ProductType → NOT NULL

■ ONE_ONE_PHASE

- **Definition**
  - This table holds information about UPS ‘s which type is one-in-one-out phase.
- **Columns**
  - **ProductID:** varchar (20) PRIMARY KEY
  - **FrequencyConverterHZ:** float
- **Primary key:**
  - ProductID
- **Foreign key:**
  - ProductID
- **Indexes:**
  -
- **Others:**
  - **Constraint** → CheckFrequencyConverterHZ >= 50

■ THREE_ONE_PHASE

- **Definition**
  - This table holds information about UPS ‘s which type is three-in-one-out phase.
- **Columns**
  - **ProductID:** varchar (20) PRIMARY KEY
  - **InputPowerFactor:** float
  - **InputVoltageRange:** float
- **Primary key:**
  - ProductID
- **Foreign key:**
  - ProductID
- **Indexes:**
  - Input_voltage_range (ProductID, InputVoltageRange DESC)
- **Others:**
  - **check_contraints** → CheckVoltages CHECK (InputPowerFactor >= 0.5 AND InputVoltageRange >= 35)

- **THREE_THREE_PHASE**

  - **Definition**
    - This table holds information about UPS ‘s which type is three-in-three-out phase.
  - **Columns**
    - **ProductID:** varchar (20) PRIMARY KEY
    - **HarmonicDistortionLevel:** float
    - **OptinalPart:** nvarchar (500)
  - **Primary key:**
    - ProductID
  - **Foreign key:**
    - ProductID
  - **Indexes:**
    -
  - **Others:**
    -

■ CUSTOMER_ORDER

- **Definition**
  - This table holds detailed information about UPS order. It includes information about the order both customer and factory perspective.
- **Columns**
  - **OrderNo:** int
  - **CustomerID:** varchar (11)
  - **ProductID:** varchar (20)
  - **OrderInfo:** nvarchar (100)
  - **Quantity:** int
  - **DateSold:** Date
  - **Isplaced:** bit
- **Primary key:**
  - OrderNo
  - CustomerID
  - ProductID
- **Foreign key:**
  - CustomerID
  - ProductID
- **Indexes:**
  - Order_view (OrderNo, ProductID, CustomerID)
- **Others:**
  - **Identity** → OrderNo

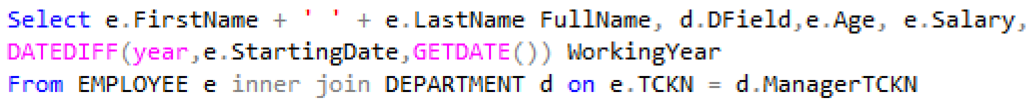
■ FACTORY_PRODUCT

- **Definition**
  - This table holds numerical (e.g., quantity) and some production related features about the UPS products.
- **Columns**
  - **ID:** int
  - **TradeRegistartionNumber:** varchar (11)
  - **ProductID:** varchar (20)
  - **Quantity:** int
  - **ProductionDate:** date
- **Primary key:**
  - ID
  - TradeRegistrationNumber
  - ProductID
- **Foreign key:**
  - TradeRegistrationNumber
  - ProductID
- **Indexes:**
  -
- **Others:**
  - **Identity** → ID

VIEWS

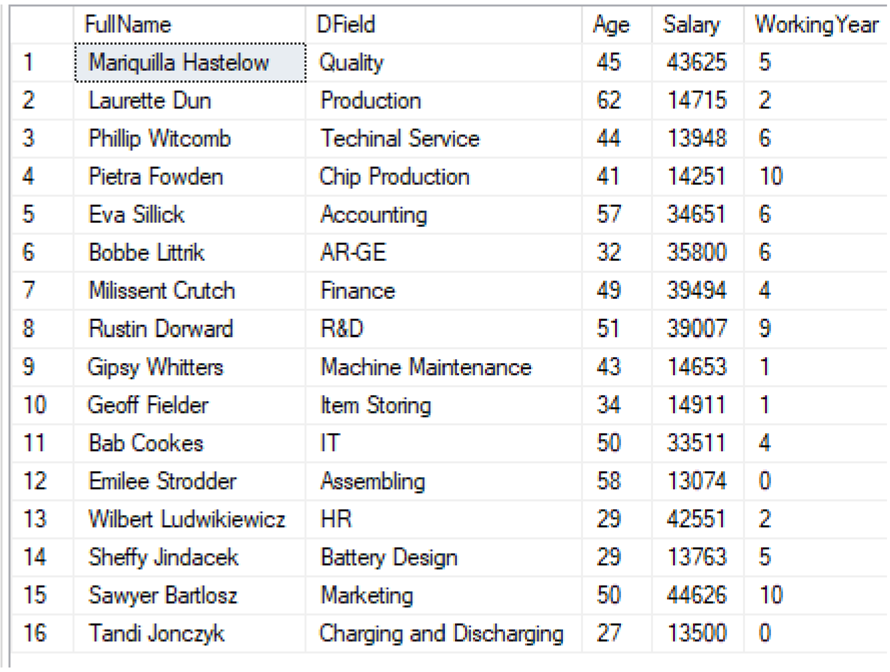
• VIEW 1

- **Name:** Manager_Details
- **Definition:** It gives detailed information about the managers.
- **View:**



```
Select e.FirstName + ' ' + e.LastName FullName, d.DField,e.Age, e.Salary,
DATEDIFF(year,e.StartingDate,GETDATE()) WorkingYear
From EMPLOYEE e inner join DEPARTMENT d on e.TCKN = d.ManagerTCKN
```

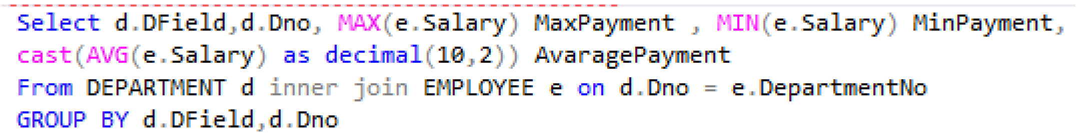
- **Output:**



	FullName	DField	Age	Salary	WorkingYear
1	Mariquilla Hastelow	Quality	45	43625	5
2	Laurette Dun	Production	62	14715	2
3	Phillip Witcomb	Techinal Service	44	13948	6
4	Pietra Fowden	Chip Production	41	14251	10
5	Eva Sillick	Accounting	57	34651	6
6	Bobbe Littrik	AR-GE	32	35800	6
7	Milissent Crutch	Finance	49	39494	4
8	Rustin Dorward	R&D	51	39007	9
9	Gipsy Whitters	Machine Maintenance	43	14653	1
10	Geoff Fielder	Item Storing	34	14911	1
11	Bab Cookes	IT	50	33511	4
12	Emilee Strodder	Assembling	58	13074	0
13	Wilbert Ludwikiewicz	HR	29	42551	2
14	Sheffy Jindacek	Battery Design	29	13763	5
15	Sawyer Bartlosz	Marketing	50	44626	10
16	Tandi Jonczyk	Charging and Discharging	27	13500	0

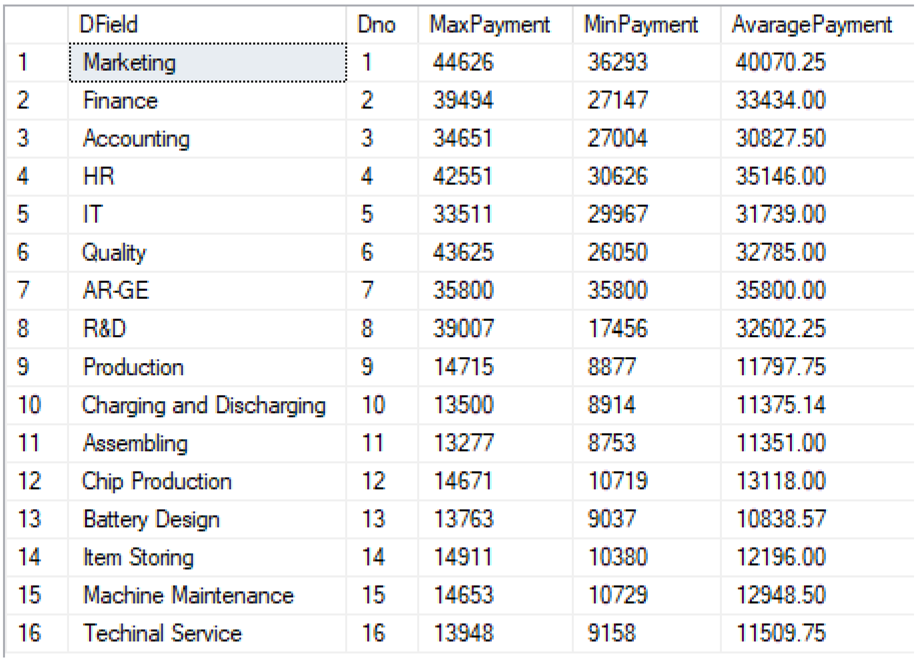
- **VIEW 2**

- **Name:** Department_Payment_Statistic
- **Definition:** It gives general salary statistic about departments. (Max payment- Min payment- Average payment)
- **View:**



```
Select d.DField,d.Dno, MAX(e.Salary) MaxPayment , MIN(e.Salary) MinPayment,
cast(AVG(e.Salary) as decimal(10,2)) AvaragePayment
From DEPARTMENT d inner join EMPLOYEE e on d.Dno = e.DepartmentNo
GROUP BY d.DField,d.Dno
```

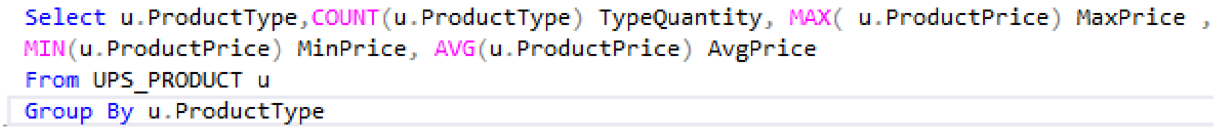
- **Output:**



	DField	Dno	MaxPayment	MinPayment	AvaragePayment
1	Marketing	1	44626	36293	40070.25
2	Finance	2	39494	27147	33434.00
3	Accounting	3	34651	27004	30827.50
4	HR	4	42551	30626	35146.00
5	IT	5	33511	29967	31739.00
6	Quality	6	43625	26050	32785.00
7	AR-GE	7	35800	35800	35800.00
8	R&D	8	39007	17456	32602.25
9	Production	9	14715	8877	11797.75
10	Charging and Discharging	10	13500	8914	11375.14
11	Assembling	11	13277	8753	11351.00
12	Chip Production	12	14671	10719	13118.00
13	Battery Design	13	13763	9037	10838.57
14	Item Storing	14	14911	10380	12196.00
15	Machine Maintenance	15	14653	10729	12948.50
16	Techinal Service	16	13948	9158	11509.75

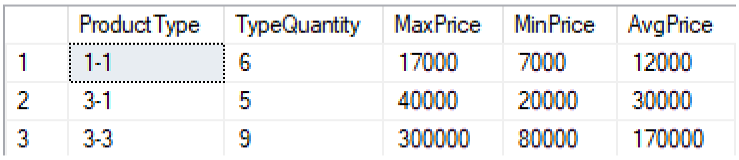
• VIEW 3

- **Name:** Ups_ProductType_Price_Statistic
- **Definition:** It gives price statistic about all type of UPS products in factory. (1-1, 3-1, 3,3)
- **View:**



```
Select u.ProductType,COUNT(u.ProductType) TypeQuantity, MAX( u.ProductPrice) MaxPrice ,
MIN(u.ProductPrice) MinPrice, AVG(u.ProductPrice) AvgPrice
From UPS_PRODUCT u
Group By u.ProductType
```

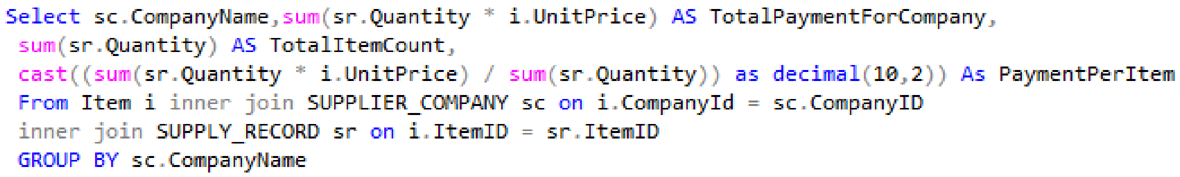
- **Output:**



	ProductType	TypeQuantity	MaxPrice	MinPrice	AvgPrice
1	1-1	6	17000	7000	12000
2	3-1	5	40000	20000	30000
3	3-3	9	300000	80000	170000

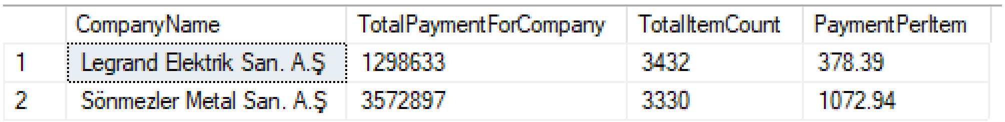
• VIEW 4

- **Name:** SupplierCompany_Payment
- **Definition:** It gives total payment for each supplier company for taken item. It also shows how many items taken from company and price for per item.
- **View:**



```
Select sc.CompanyName,sum(sr.Quantity * i.UnitPrice) AS TotalPaymentForCompany,
 sum(sr.Quantity) AS TotalItemCount,
 cast((sum(sr.Quantity * i.UnitPrice) / sum(sr.Quantity)) as decimal(10,2)) As PaymentPerItem
 From Item i inner join SUPPLIER_COMPANY sc on i.CompanyId = sc.CompanyID
 inner join SUPPLY_RECORD sr on i.ItemID = sr.ItemID
 GROUP BY sc.CompanyName
```

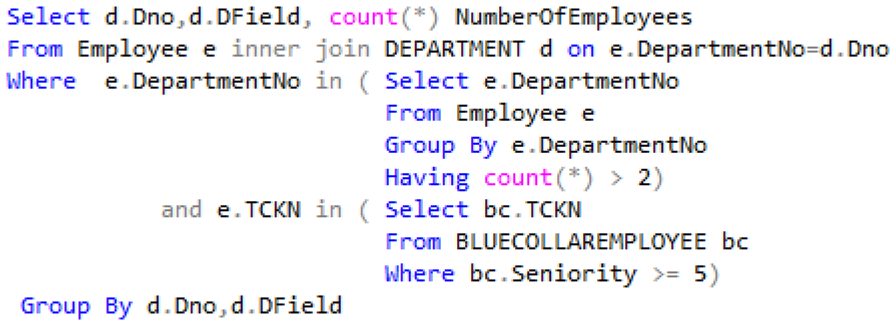
- **Output:**



	CompanyName	TotalPaymentForCompany	TotalItemCount	PaymentPerItem
1	Legrand Elektrik San. A.Ş	1298633	3432	378.39
2	Sönmezler Metal San. A.Ş	3572897	3330	1072.94

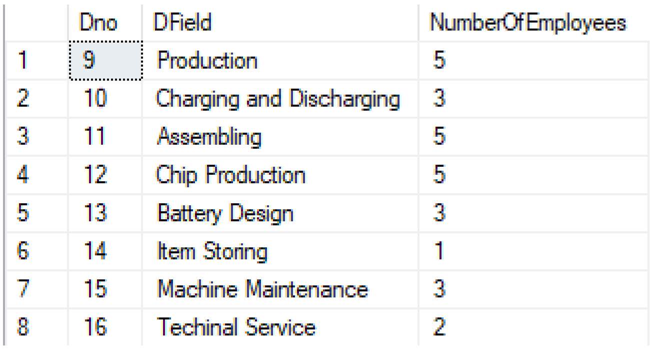
- **VIEW 5**

- **Name:** BlueCollar_Department_Seniority
- **Definition:** It shows how many blue-collar employees exist in; his/her department includes more than 2 people and at the same time he/she works for the factory at least 5 years.
- **View:**



```
Select d.Dno,d.DField, count(*) NumberOfEmployees
From Employee e inner join DEPARTMENT d on e.DepartmentNo=d.Dno
Where  e.DepartmentNo in ( Select e.DepartmentNo
                           From Employee e
                           Group By e.DepartmentNo
                           Having count(*) > 2)
           and e.TCKN in ( Select bc.TCKN
                           From BLUECOLLAREMPLOYEE bc
                           Where bc.Seniority >= 5)
 Group By d.Dno,d.DField
```

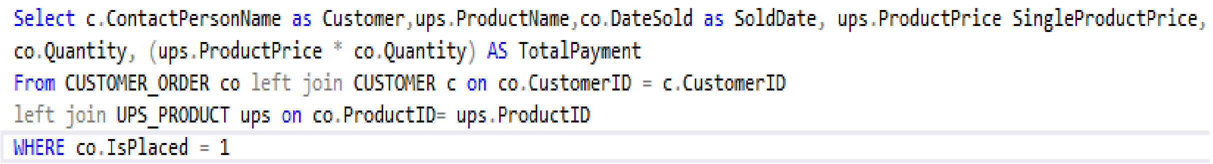
- **Output:**



	Dno	DField	NumberOfEmployees
1	9	Production	5
2	10	Charging and Discharging	3
3	11	Assembling	5
4	12	Chip Production	5
5	13	Battery Design	3
6	14	Item Storing	1
7	15	Machine Maintenance	3
8	16	Techinal Service	2

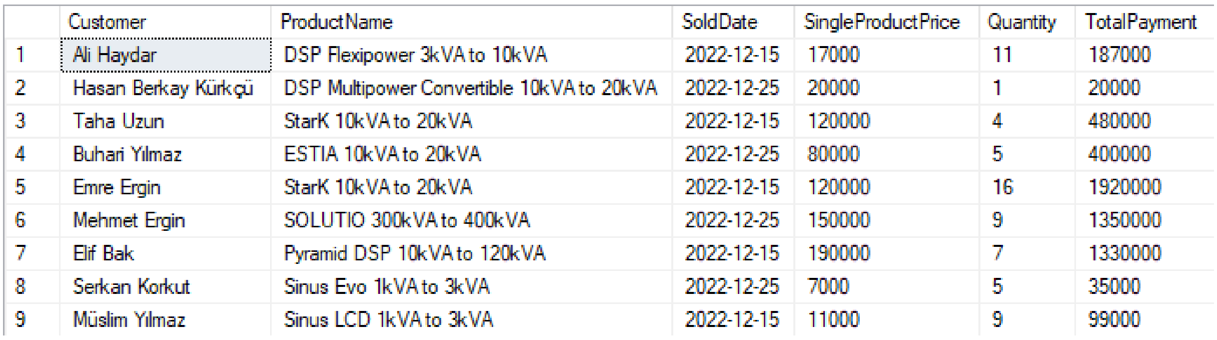
- **VIEW 6**

- **Name:** Detailed_Customer_Order
- **Definition:** It shows detailed information about accomplished successfully orders.
- **View:**



```
Select c.ContactPersonName as Customer,ups.ProductName,co.DateSold as SoldDate, ups.ProductPrice SingleProductPrice,
co.Quantity, (ups.ProductPrice * co.Quantity) AS TotalPayment
From CUSTOMER_ORDER co left join CUSTOMER c on co.CustomerID = c.CustomerID
left join UPS_PRODUCT ups on co.ProductID= ups.ProductID
WHERE co.IsPlaced = 1
```

- **Output:**



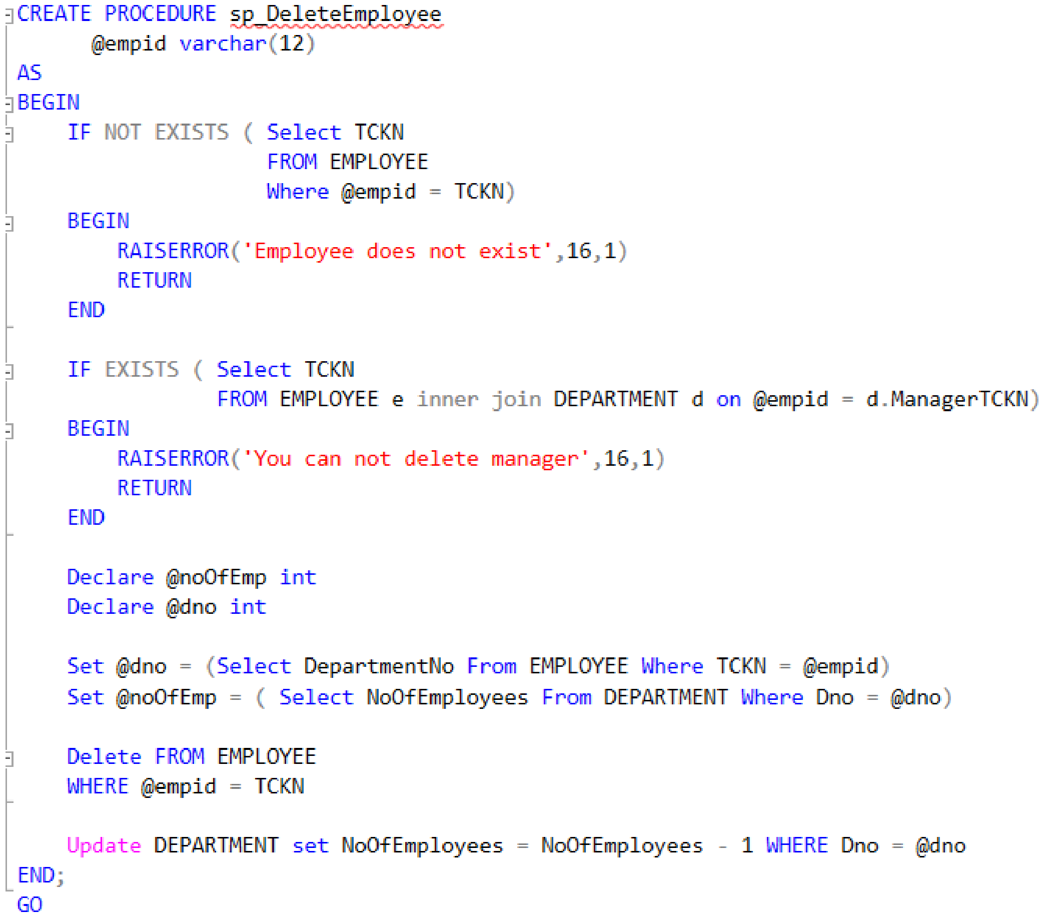
	Customer	ProductName	SoldDate	SingleProductPrice	Quantity	TotalPayment
1	Ali Haydar	DSP Flexipower 3kVA to 10kVA	2022-12-15	17000	11	187000
2	Hasan Berkay Kürkçü	DSP Multipower Convertible 10kVA to 20kVA	2022-12-25	20000	1	20000
3	Taha Uzun	StarK 10kVA to 20kVA	2022-12-15	120000	4	480000
4	Buhari Yılmaz	ESTIA 10kVA to 20kVA	2022-12-25	80000	5	400000
5	Emre Ergin	StarK 10kVA to 20kVA	2022-12-15	120000	16	1920000
6	Mehmet Ergin	SOLUTIO 300kVA to 400kVA	2022-12-25	150000	9	1350000
7	Elif Bak	Pyramid DSP 10kVA to 120kVA	2022-12-15	190000	7	1330000
8	Serkan Korkut	Sinus Evo 1kVA to 3kVA	2022-12-25	7000	5	35000
9	Müslim Yılmaz	Sinus LCD 1kVA to 3kVA	2022-12-15	11000	9	99000

PROCEDURES

- **PROCEDURE 1**

- **Name:** sp_DeleteEmployee
- **Definition:** It takes employee id as a parameter. Firstly, procedure checks if given employee exist or is it manager. If these are in case, it gives warnings. If not, it first deletes the given employee record from **EMPLOYEE** table. Then according to the collar-type, it deletes related table record. At the end it decrements the NumberOfEmployee field in the **DEPARTMENT** table according to the deleted employee’s department number.

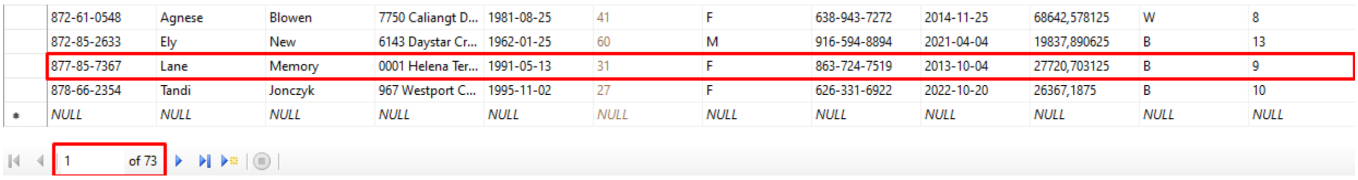
- **Procedure:**



```
CREATE PROCEDURE sp_DeleteEmployee
      @empid varchar(12)
AS
BEGIN
    IF NOT EXISTS ( Select TCKN
                    FROM EMPLOYEE
                    Where @empid = TCKN)
    BEGIN
        RAISERROR('Employee does not exist',16,1)
        RETURN
    END

    IF EXISTS ( Select TCKN
                FROM EMPLOYEE e inner join DEPARTMENT d on @empid = d.ManagerTCKN)
    BEGIN
        RAISERROR('You can not delete manager',16,1)
        RETURN
    END

    Declare @noOfEmp int
    Declare @dno int

    Set @dno = (Select DepartmentNo From EMPLOYEE Where TCKN = @empid)
    Set @noOfEmp = ( Select NoOfEmployees From DEPARTMENT Where Dno = @dno)

    Delete FROM EMPLOYEE
    WHERE @empid = TCKN

    Update DEPARTMENT set NoOfEmployees = NoOfEmployees - 1 WHERE Dno = @dno
END;
GO
```

Before exec sp_DeleteEmployee ‘877-85-7367’:

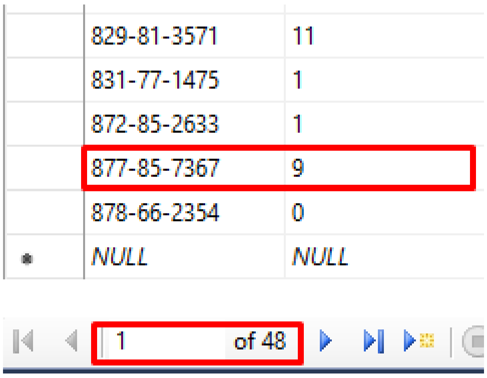
→ EMPLOYEE TABLE



	872-61-0548	Agnese	Blowen	7750 Caliangt D...	1981-08-25	41	F	638-943-7272	2014-11-25	68642,578125	W	8
	872-85-2633	Ely	New	6143 Daystar Cr...	1962-01-25	60	M	916-594-8894	2021-04-04	19837,890625	B	13
	877-85-7367	Lane	Memory	0001 Helena Ter...	1991-05-13	31	F	863-724-7519	2013-10-04	27720,703125	B	9
	878-66-2354	Tandi	Jonczyk	967 Westport C...	1995-11-02	27	F	626-331-6922	2022-10-20	26367,1875	B	10
●	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

1 of 73

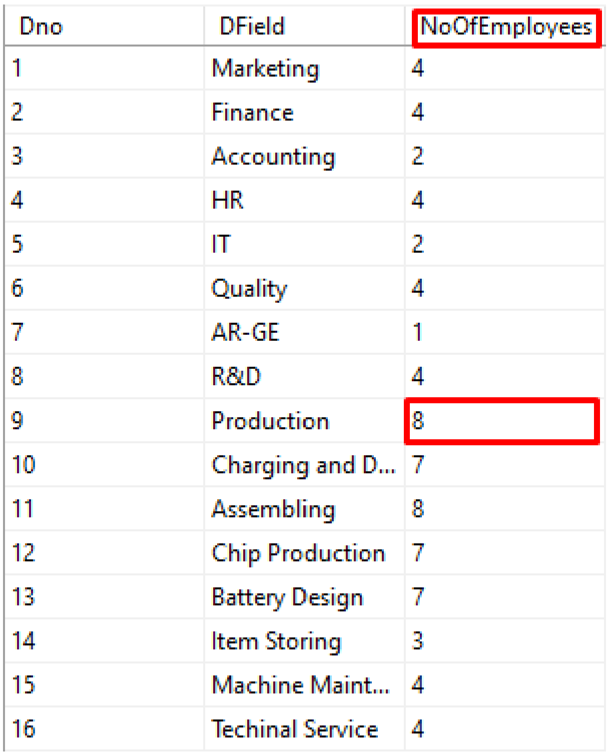
→ BLUECOLLAR TABLE



	829-81-3571	11
	831-77-1475	1
	872-85-2633	1
	877-85-7367	9
	878-66-2354	0
●	*NULL*	*NULL*

1 of 48

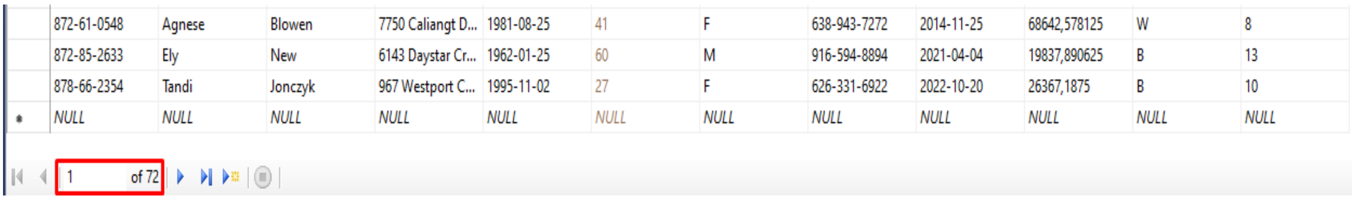
→ **DEPARTMENT TABLE**



Dno	DField	NoOfEmployees
1	Marketing	4
2	Finance	4
3	Accounting	2
4	HR	4
5	IT	2
6	Quality	4
7	AR-GE	1
8	R&D	4
9	Production	8
10	Charging and D...	7
11	Assembling	8
12	Chip Production	7
13	Battery Design	7
14	Item Storing	3
15	Machine Maint...	4
16	Techinal Service	4

- ## After exec sp_DeleteEmployee ‘877-85-7367’:

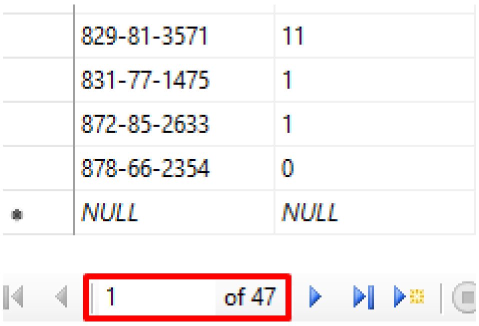
→ **EMPLOYEE TABLE**



	872-61-0548	Agnese	Blowen	7750 Caliangt D...	1981-08-25	41	F	638-943-7272	2014-11-25	68642,578125	W	8
	872-85-2633	Ely	New	6143 Daystar Cr...	1962-01-25	60	M	916-594-8894	2021-04-04	19837,890625	B	13
	878-66-2354	Tandi	Jonczyk	967 Westport C...	1995-11-02	27	F	626-331-6922	2022-10-20	26367,1875	B	10
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

1 of 72

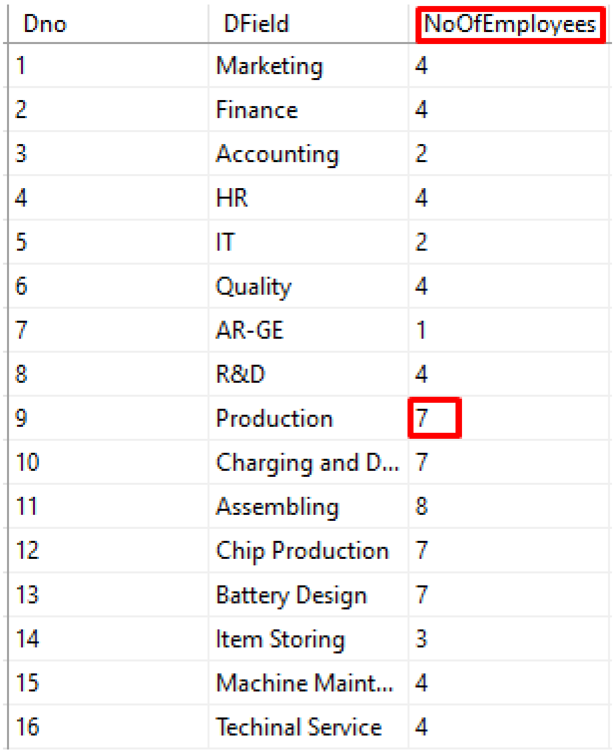
→ **BLUECOLLAR TABLE**



	829-81-3571	11
	831-77-1475	1
	872-85-2633	1
	878-66-2354	0
*	NULL	NULL

1 of 47

→ **DEPARTMENT TABLE**

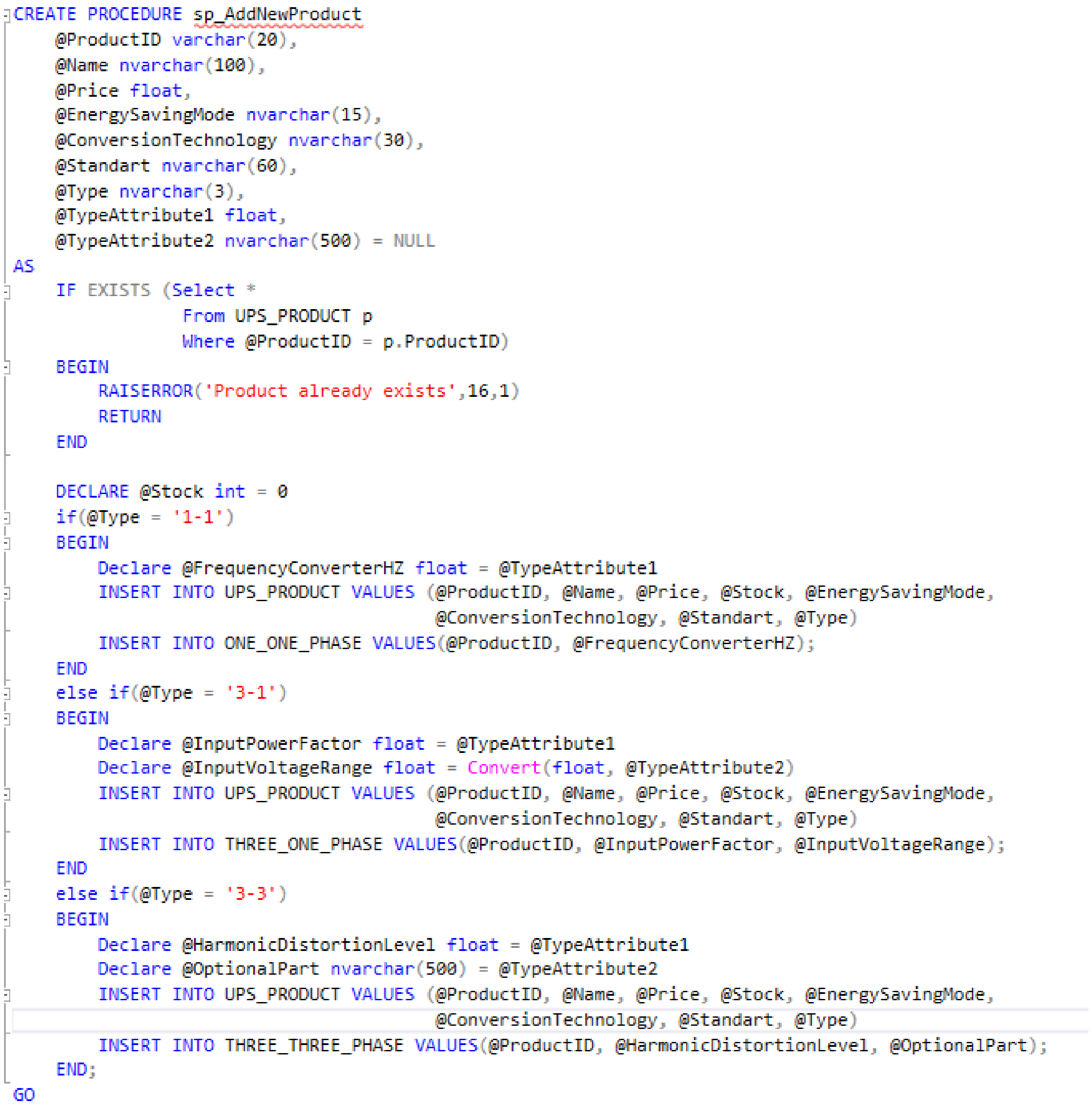


Dno	DField	NoOfEmployees
1	Marketing	4
2	Finance	4
3	Accounting	2
4	HR	4
5	IT	2
6	Quality	4
7	AR-GE	1
8	R&D	4
9	Production	7
10	Charging and D...	7
11	Assembling	8
12	Chip Production	7
13	Battery Design	7
14	Item Storing	3
15	Machine Maint...	4
16	Techinal Service	4

• PROCEDURE 2

- **Name:** sp_AddNewProduct
- **Definition:** It adds the given UPS to the UPS_PRODUCT table. In addition to that according to the UPS type it also create new record for related table.

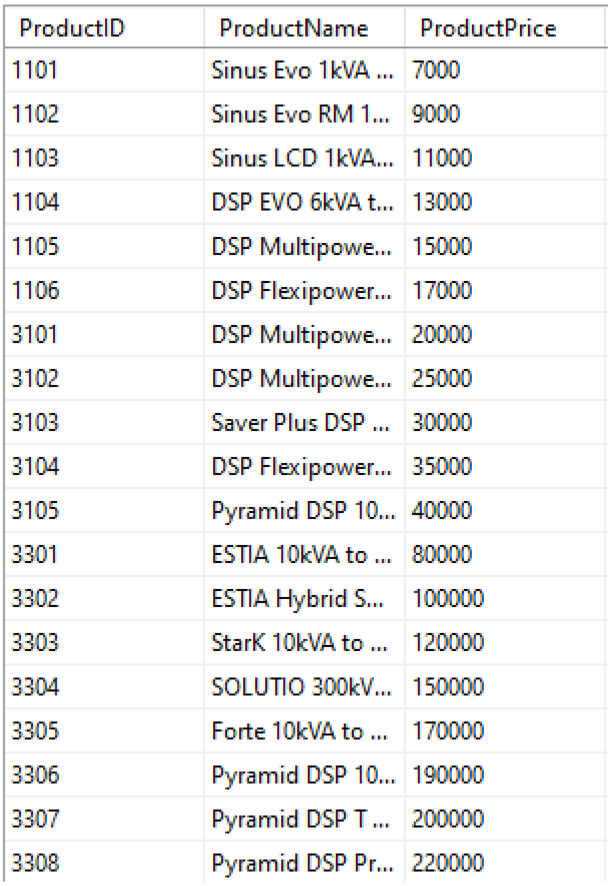
- **Procedure:**



```
CREATE PROCEDURE sp_AddNewProduct
    @ProductID varchar(20),
    @Name nvarchar(100),
    @Price float,
    @EnergySavingMode nvarchar(15),
    @ConversionTechnology nvarchar(30),
    @Standart nvarchar(60),
    @Type nvarchar(3),
    @TypeAttribute1 float,
    @TypeAttribute2 nvarchar(500) = NULL
AS
    IF EXISTS (Select *
                From UPS_PRODUCT p
                Where @ProductID = p.ProductID)
    BEGIN
        RAISERROR('Product already exists',16,1)
        RETURN
    END

    DECLARE @Stock int = 0
    if(@Type = '1-1')
    BEGIN
        Declare @FrequencyConverterHZ float = @TypeAttribute1
        INSERT INTO UPS_PRODUCT VALUES (@ProductID, @Name, @Price, @Stock, @EnergySavingMode,
                                        @ConversionTechnology, @Standart, @Type)
        INSERT INTO ONE_ONE_PHASE VALUES(@ProductID, @FrequencyConverterHZ);
    END
    else if(@Type = '3-1')
    BEGIN
        Declare @InputPowerFactor float = @TypeAttribute1
        Declare @InputVoltageRange float = Convert(float, @TypeAttribute2)
        INSERT INTO UPS_PRODUCT VALUES (@ProductID, @Name, @Price, @Stock, @EnergySavingMode,
                                        @ConversionTechnology, @Standart, @Type)
        INSERT INTO THREE_ONE_PHASE VALUES(@ProductID, @InputPowerFactor, @InputVoltageRange);
    END
    else if(@Type = '3-3')
    BEGIN
        Declare @HarmonicDistortionLevel float = @TypeAttribute1
        Declare @OptionalPart nvarchar(500) = @TypeAttribute2
        INSERT INTO UPS_PRODUCT VALUES (@ProductID, @Name, @Price, @Stock, @EnergySavingMode,
                                        @ConversionTechnology, @Standart, @Type)
        INSERT INTO THREE_THREE_PHASE VALUES(@ProductID, @HarmonicDistortionLevel, @OptionalPart);
    END;
GO
```

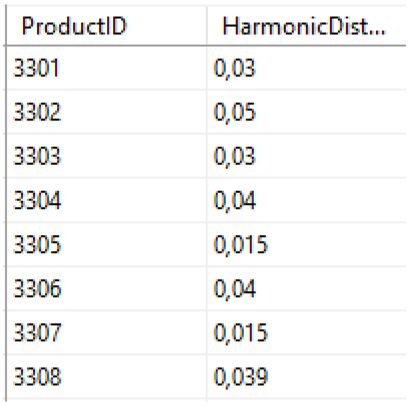
- **Before exec ‘3309’,’Pyramid DSP Premium-T 160kVA to 300kVA’,’300000,’GREEN MODE’, ‘TGG WP16 Double Conversion’, ‘SNMP’, ‘3-3’,0.04,’SNMP, Batter Block’**

→ UPS_PRODUCT TABLE



ProductID	ProductName	ProductPrice
1101	Sinus Evo 1kVA ...	7000
1102	Sinus Evo RM 1...	9000
1103	Sinus LCD 1kVA...	11000
1104	DSP EVO 6kVA t...	13000
1105	DSP Multipowe...	15000
1106	DSP Flexipower...	17000
3101	DSP Multipowe...	20000
3102	DSP Multipowe...	25000
3103	Saver Plus DSP ...	30000
3104	DSP Flexipower...	35000
3105	Pyramid DSP 10...	40000
3301	ESTIA 10kVA to ...	80000
3302	ESTIA Hybrid S...	100000
3303	StarK 10kVA to ...	120000
3304	SOLUTIO 300kV...	150000
3305	Forte 10kVA to ...	170000
3306	Pyramid DSP 10...	190000
3307	Pyramid DSP T ...	200000
3308	Pyramid DSP Pr...	220000

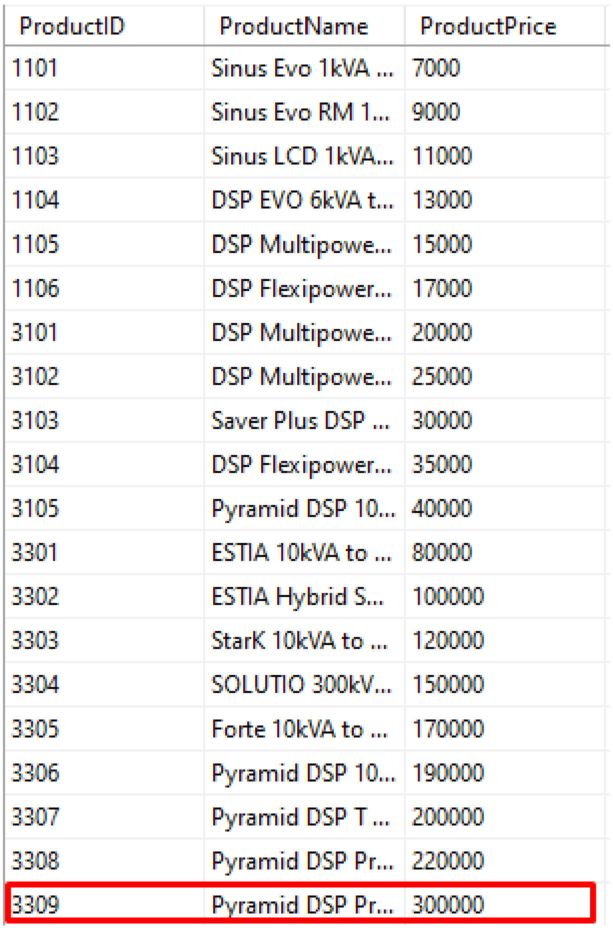
→ **THREE_THREE_PHASE TABLE**



ProductID	HarmonicDist...
3301	0,03
3302	0,05
3303	0,03
3304	0,04
3305	0,015
3306	0,04
3307	0,015
3308	0,039

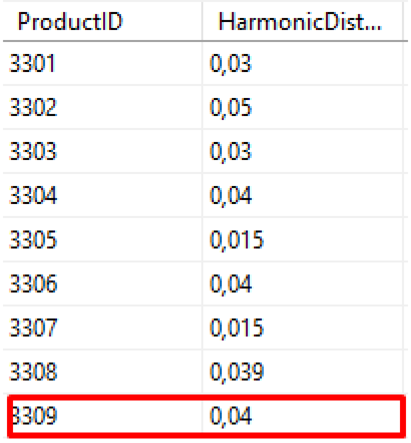
- **After exec ‘3309’,’Pyramid DSP Premium-T 160kVA to 300kVA’,’300000,’GREEN MODE’, ‘TGG WP16 Double Conversion’, ‘SNMP’, ‘3-3’,0.04,’SNMP, Batter Block’**

→ **UPS_PRODUCT TABLE**



ProductID	ProductName	ProductPrice
1101	Sinus Evo 1kVA ...	7000
1102	Sinus Evo RM 1...	9000
1103	Sinus LCD 1kVA...	11000
1104	DSP EVO 6kVA t...	13000
1105	DSP Multipowe...	15000
1106	DSP Flexipower...	17000
3101	DSP Multipowe...	20000
3102	DSP Multipowe...	25000
3103	Saver Plus DSP ...	30000
3104	DSP Flexipower...	35000
3105	Pyramid DSP 10...	40000
3301	ESTIA 10kVA to ...	80000
3302	ESTIA Hybrid S...	100000
3303	StarK 10kVA to ...	120000
3304	SOLUTIO 300kV...	150000
3305	Forte 10kVA to ...	170000
3306	Pyramid DSP 10...	190000
3307	Pyramid DSP T ...	200000
3308	Pyramid DSP Pr...	220000
3309	Pyramid DSP Pr...	300000

→ **THREE_THREE_PHASE TABLE**



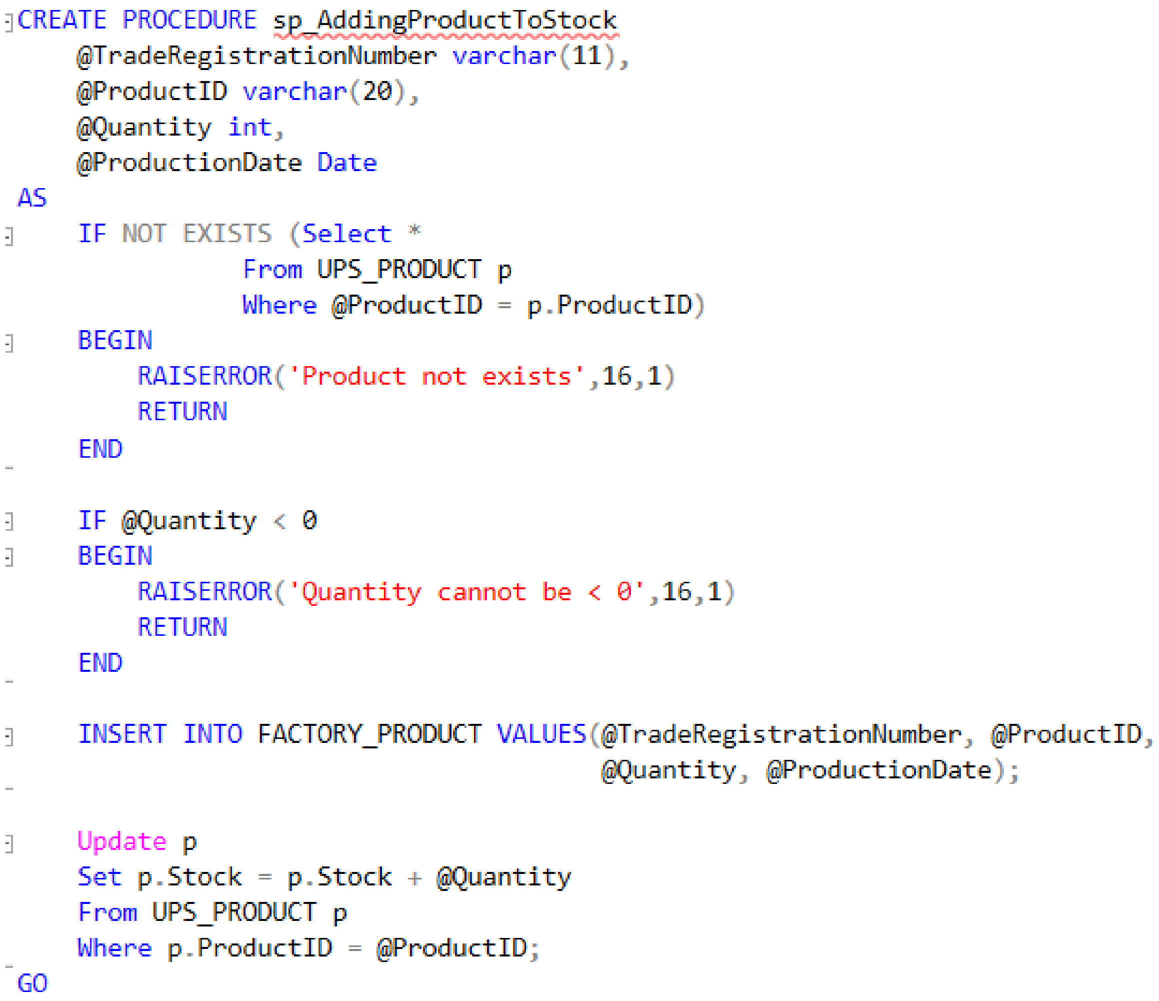
ProductID	HarmonicDist...
3301	0,03
3302	0,05
3303	0,03
3304	0,04
3305	0,015
3306	0,04
3307	0,015
3308	0,039
3309	0,04

• PROCEDURE 3

- **Name:** sp_AddingProductToStock
- **Definition:** This procedure first creates record for the **FACTORY_PRODUCT** table with the given input values.

Then it updates the stock field in the **UPS_PRODUCT** table with the given quantity value.

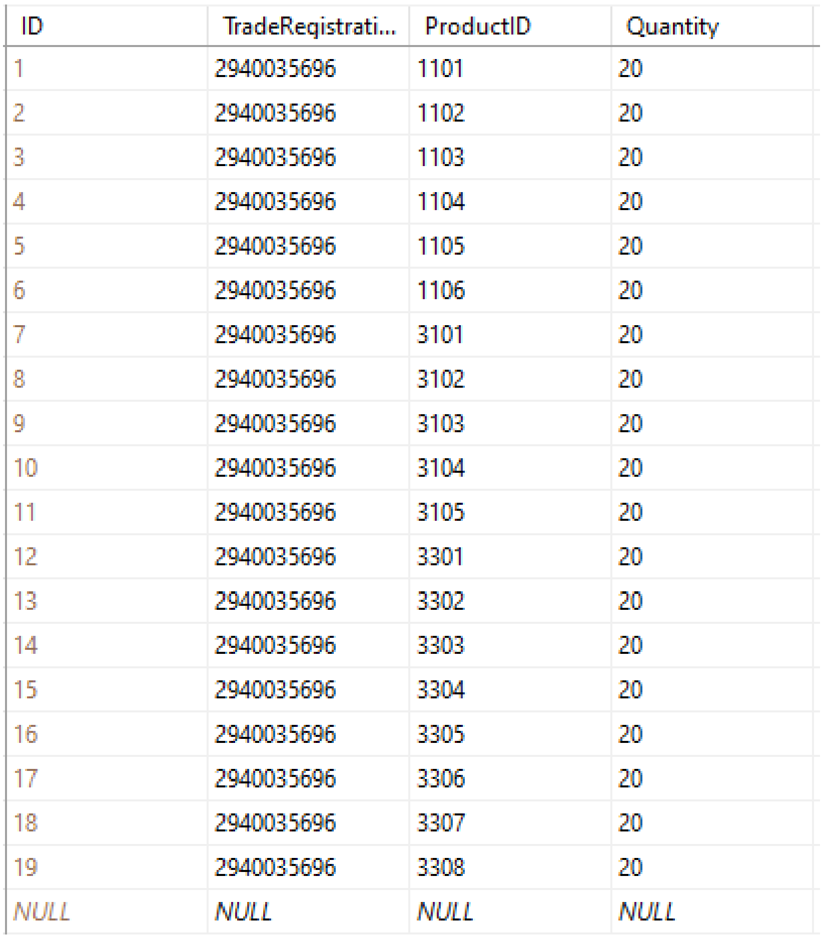
- **Procedure:**



```sql
CREATE PROCEDURE sp_AddingProductToStock
    @TradeRegistrationNumber varchar(11),
    @ProductID varchar(20),
    @Quantity int,
    @ProductionDate Date
AS
    IF NOT EXISTS (Select *
                From UPS_PRODUCT p
                Where @ProductID = p.ProductID)
    BEGIN
        RAISERROR('Product not exists',16,1)
        RETURN
    END

    IF @Quantity < 0
    BEGIN
        RAISERROR('Quantity cannot be < 0',16,1)
        RETURN
    END

    INSERT INTO FACTORY_PRODUCT VALUES(@TradeRegistrationNumber, @ProductID,
                                       @Quantity, @ProductionDate);

    Update p
    Set p.Stock = p.Stock + @Quantity
    From UPS_PRODUCT p
    Where p.ProductID = @ProductID;
GO
```

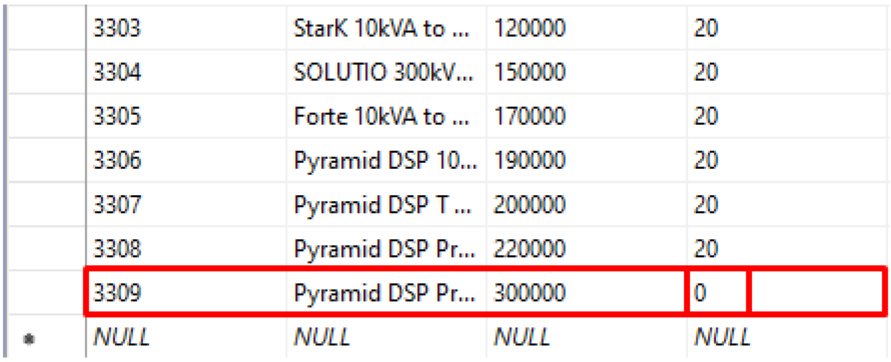
- **Before exec sp_AddingProductToStock ‘2940035696’,’3309’,20,’2022-12-26’**

→ FACTORY_PRODUCT TABLE



ID	TradeRegistrati...	ProductID	Quantity
1	2940035696	1101	20
2	2940035696	1102	20
3	2940035696	1103	20
4	2940035696	1104	20
5	2940035696	1105	20
6	2940035696	1106	20
7	2940035696	3101	20
8	2940035696	3102	20
9	2940035696	3103	20
10	2940035696	3104	20
11	2940035696	3105	20
12	2940035696	3301	20
13	2940035696	3302	20
14	2940035696	3303	20
15	2940035696	3304	20
16	2940035696	3305	20
17	2940035696	3306	20
18	2940035696	3307	20
19	2940035696	3308	20
NULL	*NULL*	*NULL*	*NULL*

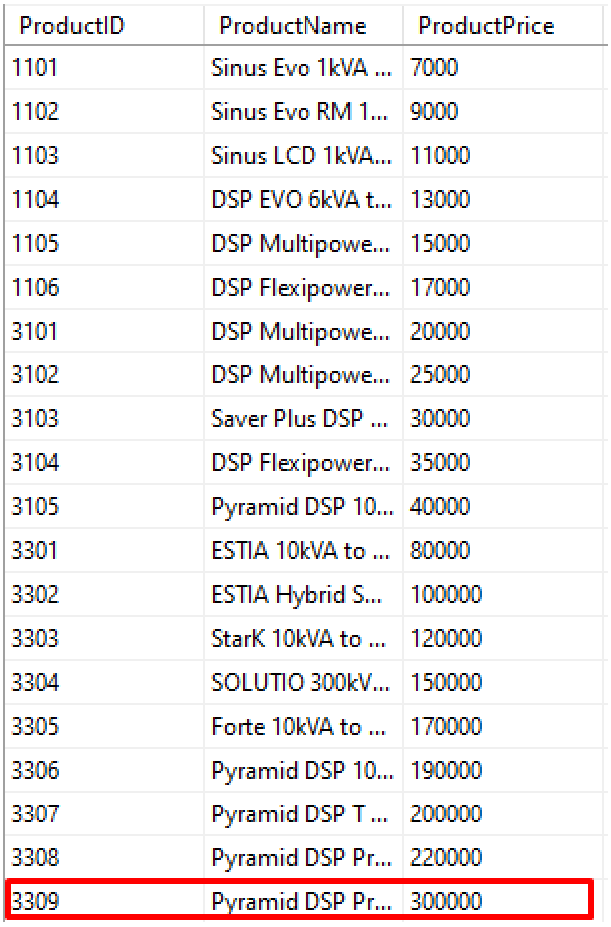
→ **UPS_PRODUCT TABLE**



	3303	StarK 10kVA to ...	120000	20
	3304	SOLUTIO 300kV...	150000	20
	3305	Forte 10kVA to ...	170000	20
	3306	Pyramid DSP 10...	190000	20
	3307	Pyramid DSP T ...	200000	20
	3308	Pyramid DSP Pr...	220000	20
	3309	Pyramid DSP Pr...	300000	0
*	*NULL*	*NULL*	*NULL*	*NULL*

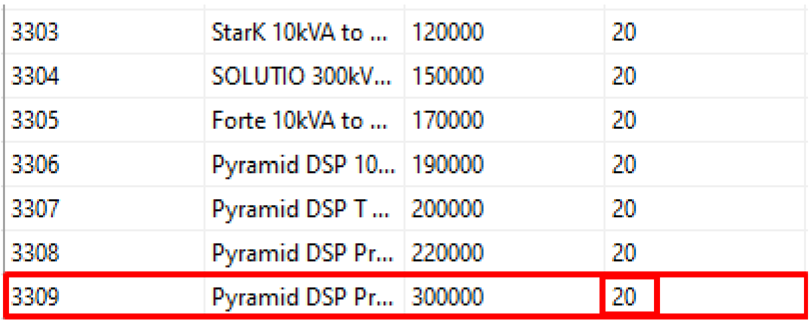
- **After exec sp_AddingProductToStock ‘2940035696’,’3309’,20,’2022-12-26’**

→ **FACTORY_PRODUCT TABLE**



ProductID	ProductName	ProductPrice
1101	Sinus Evo 1kVA ...	7000
1102	Sinus Evo RM 1...	9000
1103	Sinus LCD 1kVA...	11000
1104	DSP EVO 6kVA t...	13000
1105	DSP Multipowe...	15000
1106	DSP Flexipower...	17000
3101	DSP Multipowe...	20000
3102	DSP Multipowe...	25000
3103	Saver Plus DSP ...	30000
3104	DSP Flexipower...	35000
3105	Pyramid DSP 10...	40000
3301	ESTIA 10kVA to ...	80000
3302	ESTIA Hybrid S...	100000
3303	StarK 10kVA to ...	120000
3304	SOLUTIO 300kV...	150000
3305	Forte 10kVA to ...	170000
3306	Pyramid DSP 10...	190000
3307	Pyramid DSP T ...	200000
3308	Pyramid DSP Pr...	220000
3309	Pyramid DSP Pr...	300000

→ **UPS_PRODUCT TABLE**

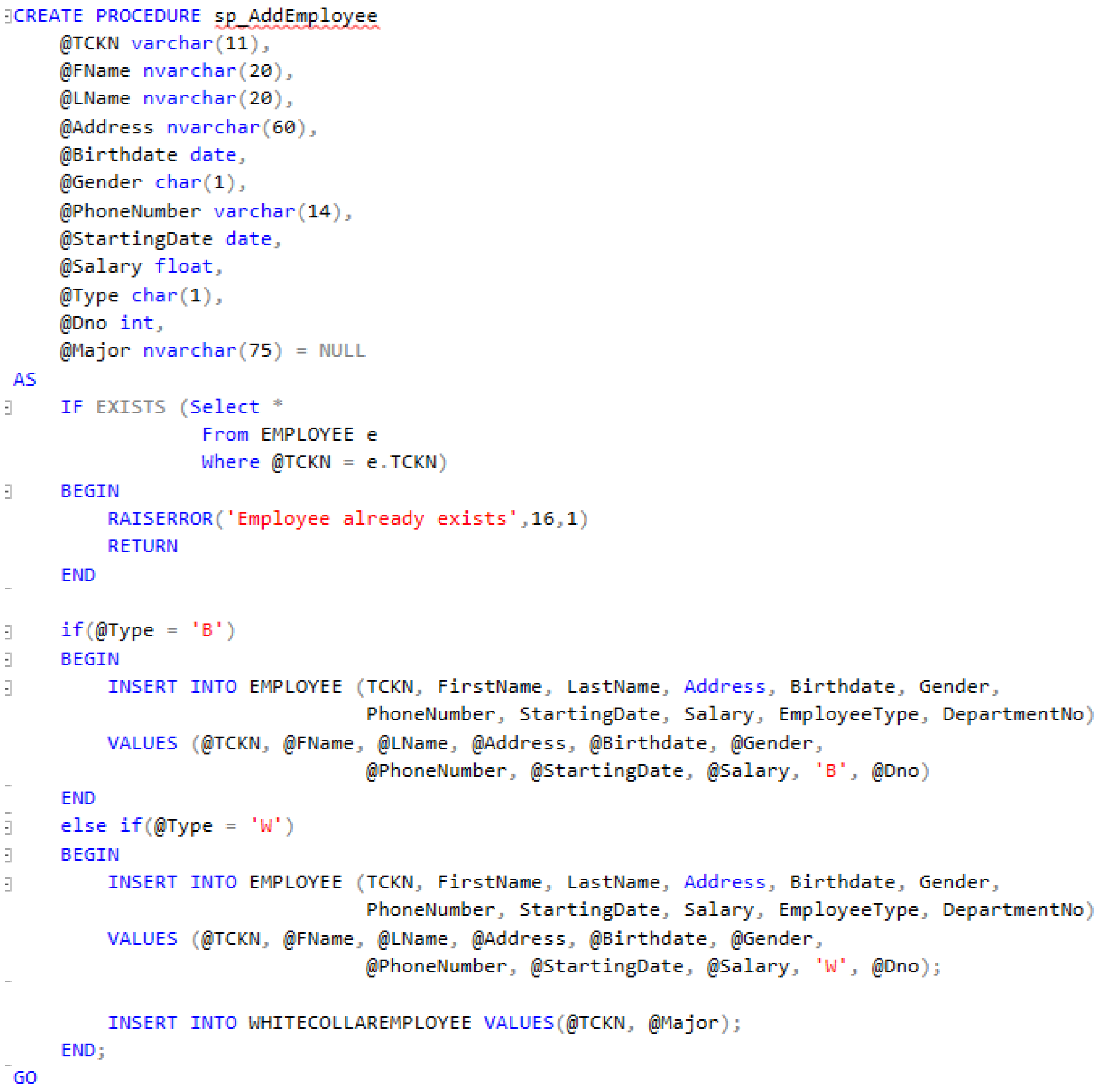


3303	StarK 10kVA to ...	120000	20
3304	SOLUTIO 300kV...	150000	20
3305	Forte 10kVA to ...	170000	20
3306	Pyramid DSP 10...	190000	20
3307	Pyramid DSP T ...	200000	20
3308	Pyramid DSP Pr...	220000	20
3309	Pyramid DSP Pr...	300000	20

• PROCEDURE 4

- **Name:** sp_AddEmployee

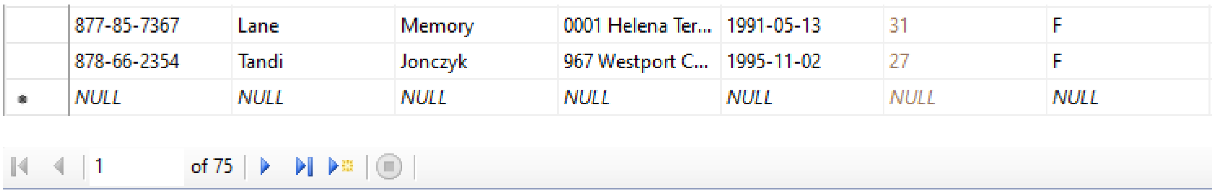
- **Definition:** This procedure creates an employee record using the employee information provided as input. It first creates record for EMPLOYEE table and according to the employee type it also creates record for relevant type table.
- **Procedure:**



```
CREATE PROCEDURE sp_AddEmployee
    @TCKN varchar(11),
    @FName nvarchar(20),
    @LName nvarchar(20),
    @Address nvarchar(60),
    @Birthdate date,
    @Gender char(1),
    @PhoneNumber varchar(14),
    @StartingDate date,
    @Salary float,
    @Type char(1),
    @Dno int,
    @Major nvarchar(75) = NULL
AS
    IF EXISTS (Select *
               From EMPLOYEE e
               Where @TCKN = e.TCKN)
    BEGIN
        RAISERROR('Employee already exists',16,1)
        RETURN
    END

    if(@Type = 'B')
    BEGIN
        INSERT INTO EMPLOYEE (TCKN, FirstName, LastName, Address, Birthdate, Gender,
                              PhoneNumber, StartingDate, Salary, EmployeeType, DepartmentNo)
        VALUES (@TCKN, @FName, @LName, @Address, @Birthdate, @Gender,
                              @PhoneNumber, @StartingDate, @Salary, 'B', @Dno)
    END
    else if(@Type = 'W')
    BEGIN
        INSERT INTO EMPLOYEE (TCKN, FirstName, LastName, Address, Birthdate, Gender,
                              PhoneNumber, StartingDate, Salary, EmployeeType, DepartmentNo)
        VALUES (@TCKN, @FName, @LName, @Address, @Birthdate, @Gender,
                              @PhoneNumber, @StartingDate, @Salary, 'W', @Dno);

        INSERT INTO WHITECOLLAREMPLOYEE VALUES(@TCKN, @Major);
    END;
GO
```

- **Before exec sp_AddEmployee ‘999-99-999’,’John’,’Doe’,’somewhere,’**

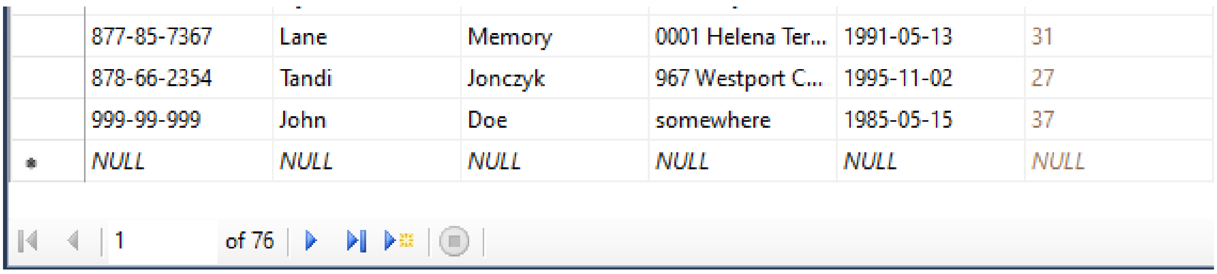
→ EMPLOYEE TABLE



	877-85-7367	Lane	Memory	0001 Helena Ter...	1991-05-13	31	F
	878-66-2354	Tandi	Jonczyk	967 Westport C...	1995-11-02	27	F
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- **After exec sp_AddEmployee ‘999-99-999’,’John’,’Doe’,’somewhere,’**

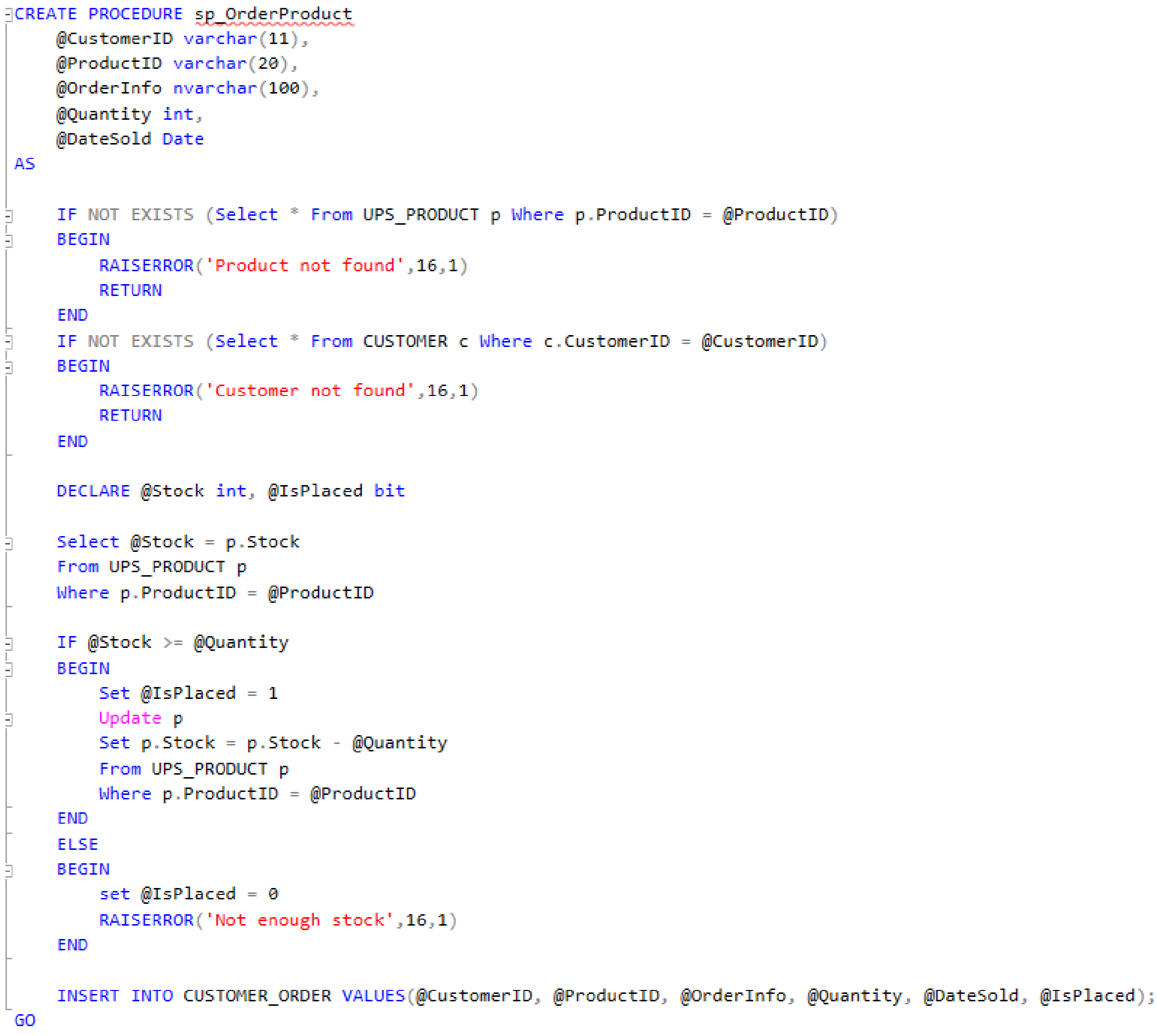
→ EMPLOYEE TABLE



	877-85-7367	Lane	Memory	0001 Helena Ter...	1991-05-13	31
	878-66-2354	Tandi	Jonczyk	967 Westport C...	1995-11-02	27
	999-99-999	John	Doe	somewhere	1985-05-15	37
*	NULL	NULL	NULL	NULL	NULL	NULL

- **PROCEDURE 5**

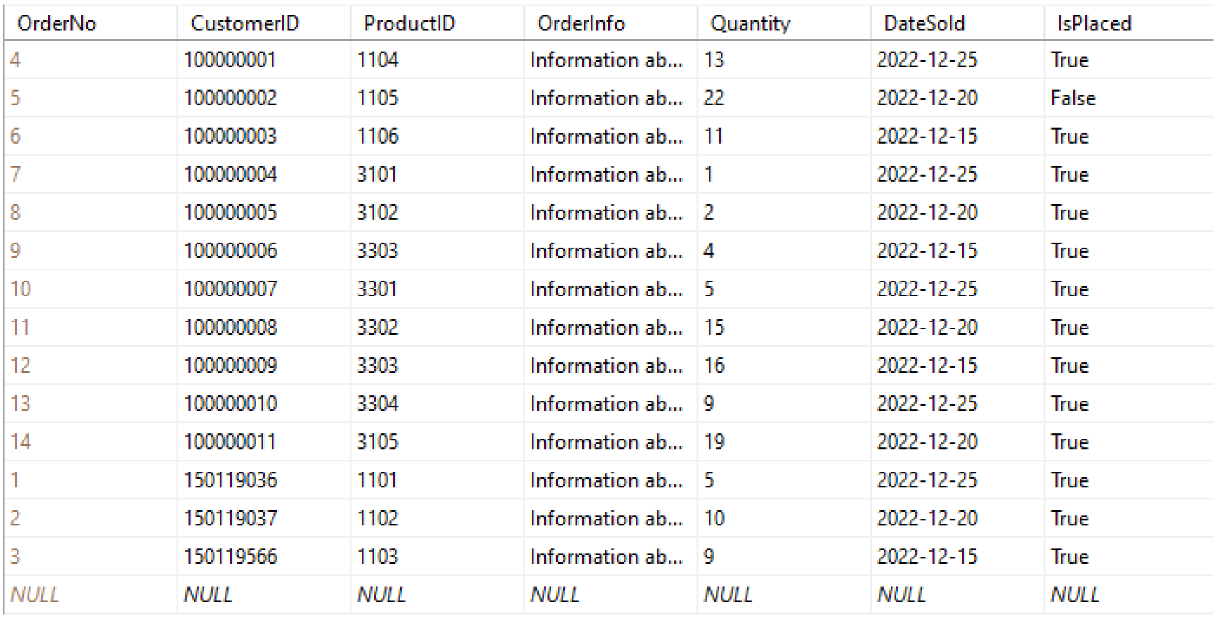
- **Name:** sp_OrderProduct
- **Definition:** This procedure updates the transactions related to order. First, it created record for the CUSTOMER_ORDER table with using the given inputs. Then, it updates the remaining stock amount by deducting the ordered amount from the stock amount. Finally, it checks whether there is the desired stock in the order and updates IsPlaced field true or false accordingly.
- **Procedure:**



```
CREATE PROCEDURE sp_OrderProduct
    @CustomerID varchar(11),
    @ProductID varchar(20),
    @OrderInfo nvarchar(100),
    @Quantity int,
    @DateSold Date
AS

    IF NOT EXISTS (Select * From UPS_PRODUCT p Where p.ProductID = @ProductID)
    BEGIN
        RAISERROR('Product not found',16,1)
        RETURN
    END
    IF NOT EXISTS (Select * From CUSTOMER c Where c.CustomerID = @CustomerID)
    BEGIN
        RAISERROR('Customer not found',16,1)
        RETURN
    END

    DECLARE @Stock int, @IsPlaced bit

    Select @Stock = p.Stock
    From UPS_PRODUCT p
    Where p.ProductID = @ProductID

    IF @Stock >= @Quantity
    BEGIN
        Set @IsPlaced = 1
        Update p
        Set p.Stock = p.Stock - @Quantity
        From UPS_PRODUCT p
        Where p.ProductID = @ProductID
    END
    ELSE
    BEGIN
        set @IsPlaced = 0
        RAISERROR('Not enough stock',16,1)
    END

    INSERT INTO CUSTOMER_ORDER VALUES(@CustomerID, @ProductID, @OrderInfo, @Quantity, @DateSold, @IsPlaced);
GO
```

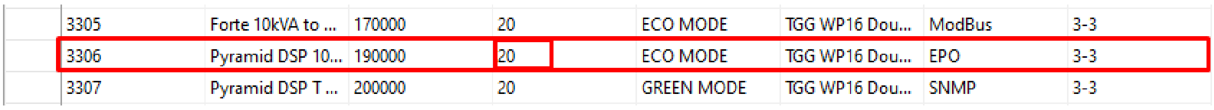
- **Before exec sp_OrderProduct ‘100000012’,3306,’…’,7,’2012-12-15’**

→ CUSTOMER_ORDER TABLE



OrderNo	CustomerID	ProductID	OrderInfo	Quantity	DateSold	IsPlaced
4	100000001	1104	Information ab...	13	2022-12-25	True
5	100000002	1105	Information ab...	22	2022-12-20	False
6	100000003	1106	Information ab...	11	2022-12-15	True
7	100000004	3101	Information ab...	1	2022-12-25	True
8	100000005	3102	Information ab...	2	2022-12-20	True
9	100000006	3303	Information ab...	4	2022-12-15	True
10	100000007	3301	Information ab...	5	2022-12-25	True
11	100000008	3302	Information ab...	15	2022-12-20	True
12	100000009	3303	Information ab...	16	2022-12-15	True
13	100000010	3304	Information ab...	9	2022-12-25	True
14	100000011	3105	Information ab...	19	2022-12-20	True
1	150119036	1101	Information ab...	5	2022-12-25	True
2	150119037	1102	Information ab...	10	2022-12-20	True
3	150119566	1103	Information ab...	9	2022-12-15	True
NULL	NULL	NULL	NULL	NULL	NULL	NULL

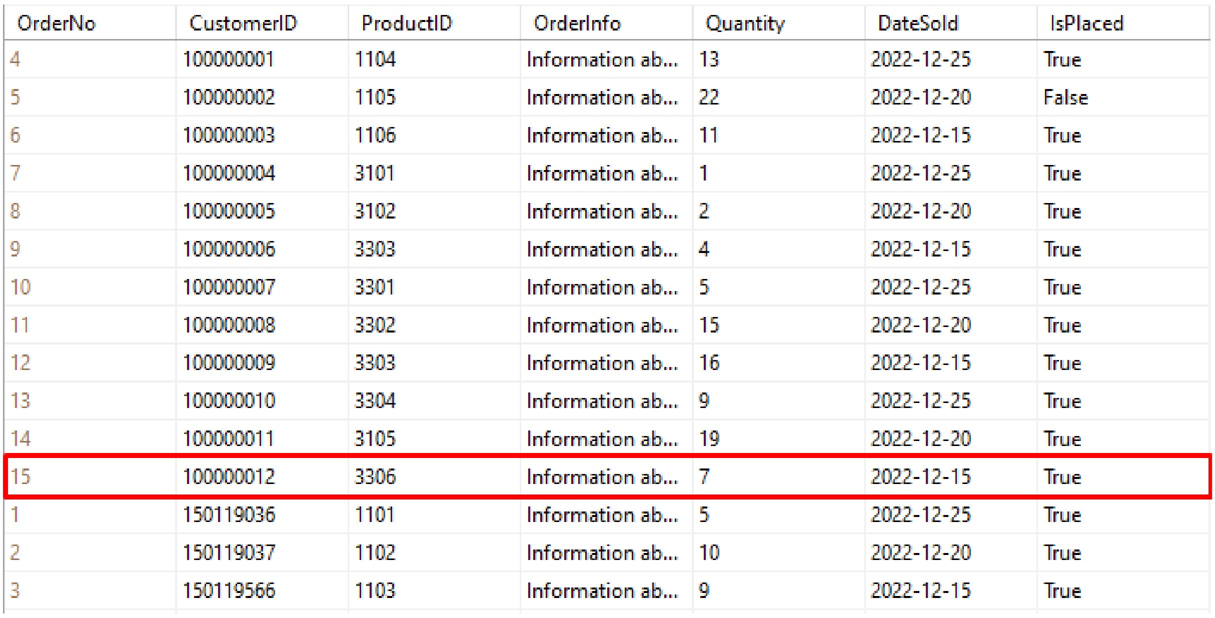
→ CUSTOMER_ORDER TABLE



	3305	Forte 10kVA to ...	170000	20	ECO MODE	TGG WP16 Dou...	ModBus	3-3
	3306	Pyramid DSP 10...	190000	20	ECO MODE	TGG WP16 Dou...	EPO	3-3
	3307	Pyramid DSP T ...	200000	20	GREEN MODE	TGG WP16 Dou...	SNMP	3-3

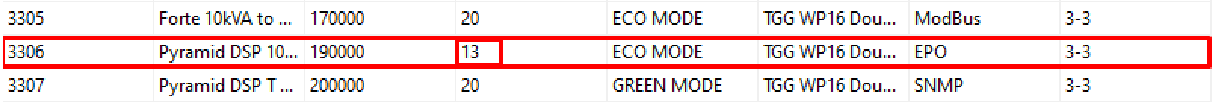
- **After exec sp_OrderProduct ‘100000012’,3306,’…’,7,’2012-12-15’**

→ **CUSTOMER_ORDER TABLE**



OrderNo	CustomerID	ProductID	OrderInfo	Quantity	DateSold	IsPlaced
4	100000001	1104	Information ab...	13	2022-12-25	True
5	100000002	1105	Information ab...	22	2022-12-20	False
6	100000003	1106	Information ab...	11	2022-12-15	True
7	100000004	3101	Information ab...	1	2022-12-25	True
8	100000005	3102	Information ab...	2	2022-12-20	True
9	100000006	3303	Information ab...	4	2022-12-15	True
10	100000007	3301	Information ab...	5	2022-12-25	True
11	100000008	3302	Information ab...	15	2022-12-20	True
12	100000009	3303	Information ab...	16	2022-12-15	True
13	100000010	3304	Information ab...	9	2022-12-25	True
14	100000011	3105	Information ab...	19	2022-12-20	True
15	100000012	3306	Information ab...	7	2022-12-15	True
1	150119036	1101	Information ab...	5	2022-12-25	True
2	150119037	1102	Information ab...	10	2022-12-20	True
3	150119566	1103	Information ab...	9	2022-12-15	True

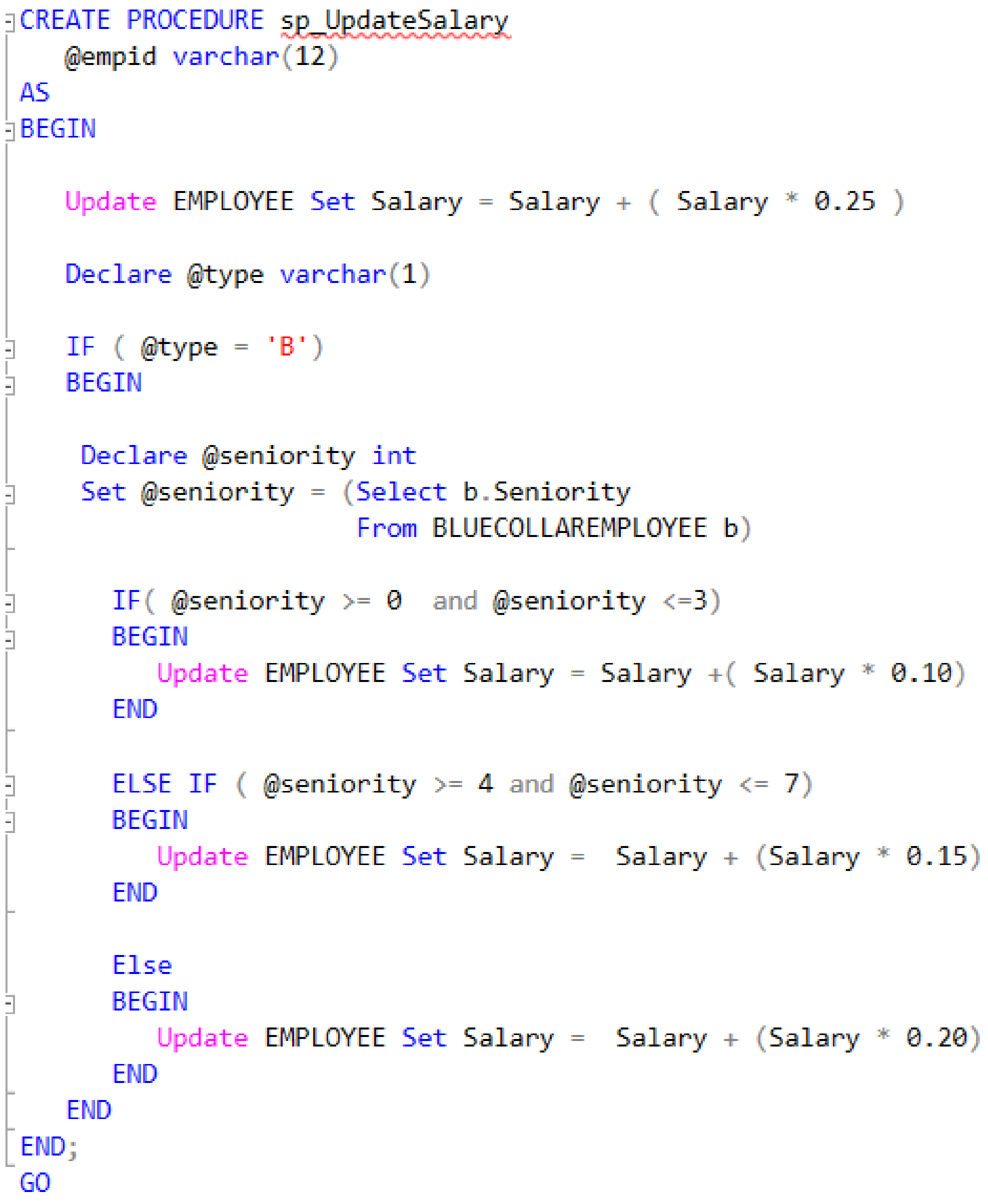
→ **CUSTOMER_ORDER TABLE**



3305	Forte 10kVA to ...	170000	20		ECO MODE	TGG WP16 Dou...	ModBus	3-3
3306	Pyramid DSP 10...	190000	13		ECO MODE	TGG WP16 Dou...	EPO	3-3
3307	Pyramid DSP T ...	200000	20		GREEN MODE	TGG WP16 Dou...	SNMP	3-3

• PROCEDURE 6

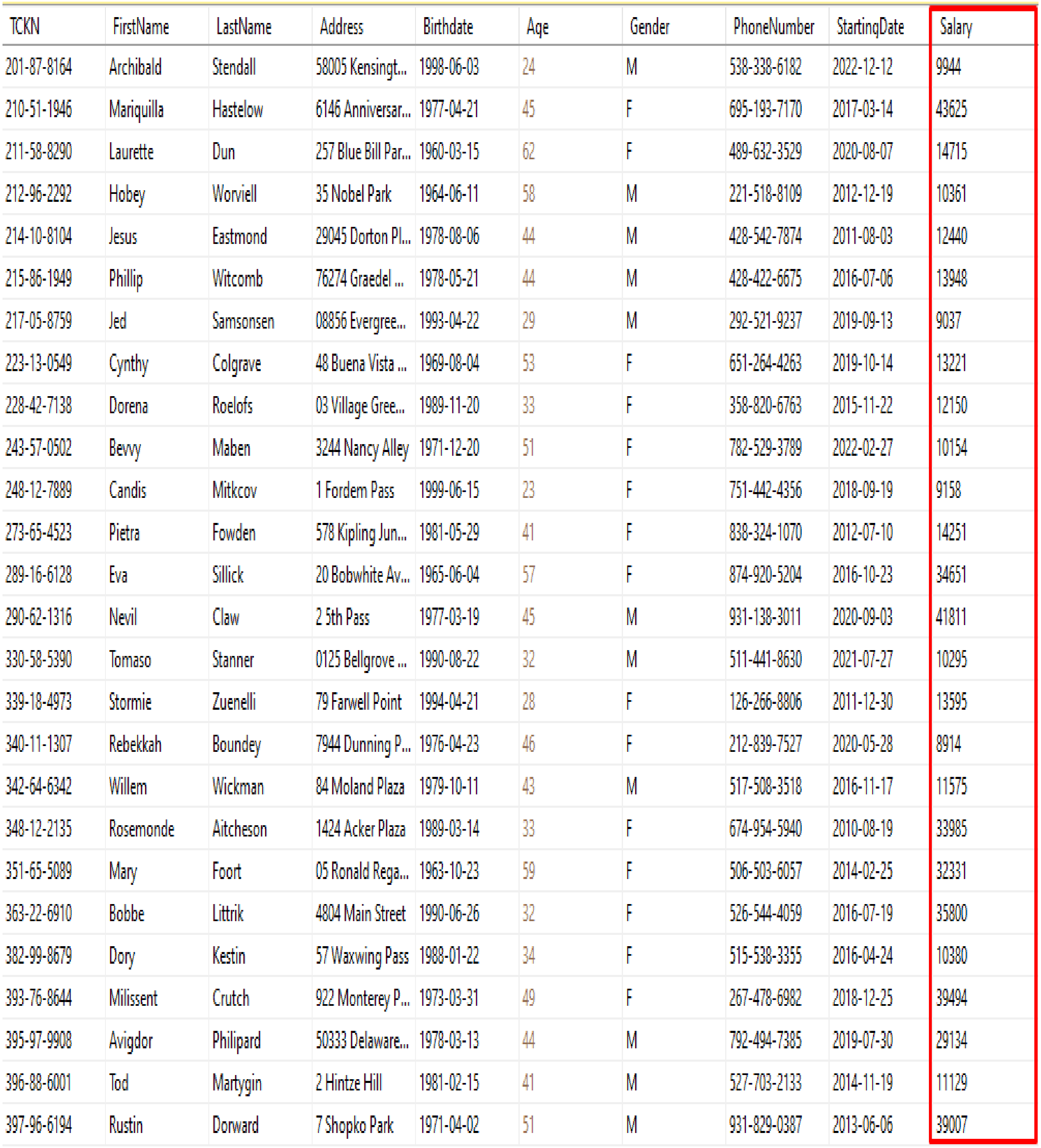
- **Name:** sp_UpdateSalary
- **Definition:** This procedure updates the salary for the employees in the **EMPLOYEE** table. A fixed 25% raise is made for two types of employees working in the factory. Afterwards, an extra raise is made for blue-collar employees according to their seniority levels.
- **Procedure:**



```
CREATE PROCEDURE sp_UpdateSalary
   @empid varchar(12)
AS
BEGIN

   Update EMPLOYEE Set Salary = Salary + ( Salary * 0.25 )

   Declare @type varchar(1)

   IF ( @type = 'B')
   BEGIN

    Declare @seniority int
    Set @seniority = (Select b.Seniority
                      From BLUECOLLAREMPLOYEE b)

      IF( @seniority >= 0  and @seniority <=3)
      BEGIN
         Update EMPLOYEE Set Salary = Salary +( Salary * 0.10)
      END

      ELSE IF ( @seniority >= 4 and @seniority <= 7)
      BEGIN
         Update EMPLOYEE Set Salary =  Salary + (Salary * 0.15)
      END

      Else
      BEGIN
         Update EMPLOYEE Set Salary =  Salary + (Salary * 0.20)
      END
   END
END;
GO
```

- **Before exec sp_UpdateSalary**

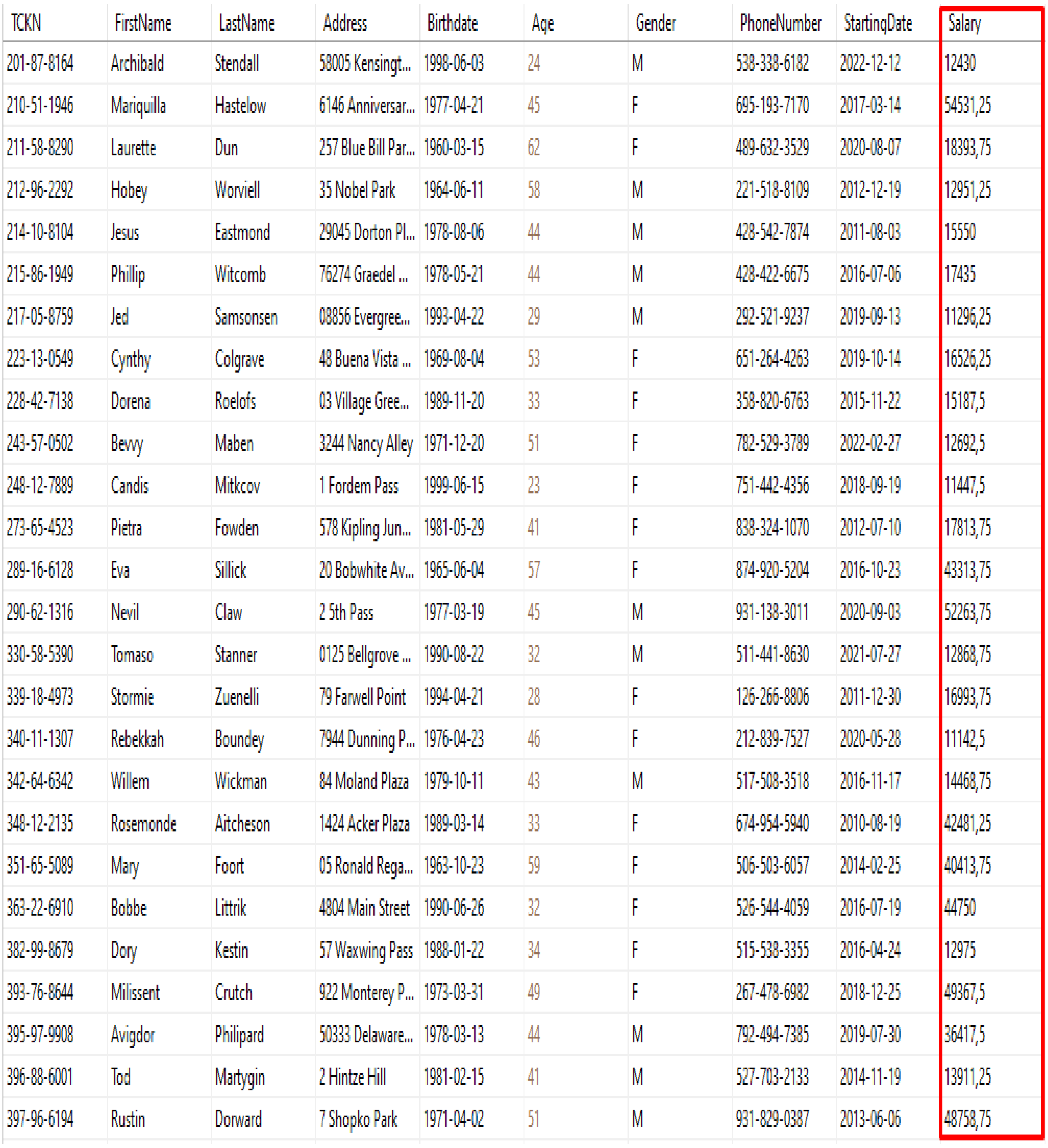
→ **EMPLOYEE TABLE**



TCKN	FirstName	LastName	Address	Birthdate	Age	Gender	PhoneNumber	StartingDate	Salary
201-87-8164	Archibald	Stendall	58005 Kensingt...	1998-06-03	24	M	538-338-6182	2022-12-12	9944
210-51-1946	Mariquilla	Hastelow	6146 Anniversar...	1977-04-21	45	F	695-193-7170	2017-03-14	43625
211-58-8290	Laurette	Dun	257 Blue Bill Par...	1960-03-15	62	F	489-632-3529	2020-08-07	14715
212-96-2292	Hobey	Worviell	35 Nobel Park	1964-06-11	58	M	221-518-8109	2012-12-19	10361
214-10-8104	Jesus	Eastmond	29045 Dorton Pl...	1978-08-06	44	M	428-542-7874	2011-08-03	12440
215-86-1949	Phillip	Witcomb	76274 Graedel ...	1978-05-21	44	M	428-422-6675	2016-07-06	13948
217-05-8759	Jed	Samsonsen	08856 Evergree...	1993-04-22	29	M	292-521-9237	2019-09-13	9037
223-13-0549	Cynthy	Colgrave	48 Buena Vista ...	1969-08-04	53	F	651-264-4263	2019-10-14	13221
228-42-7138	Dorena	Roelofs	03 Village Gree...	1989-11-20	33	F	358-820-6763	2015-11-22	12150
243-57-0502	Bevvy	Maben	3244 Nancy Alley	1971-12-20	51	F	782-529-3789	2022-02-27	10154
248-12-7889	Candis	Mitkcov	1 Fordem Pass	1999-06-15	23	F	751-442-4356	2018-09-19	9158
273-65-4523	Pietra	Fowden	578 Kipling Jun...	1981-05-29	41	F	838-324-1070	2012-07-10	14251
289-16-6128	Eva	Sillick	20 Bobwhite Av...	1965-06-04	57	F	874-920-5204	2016-10-23	34651
290-62-1316	Nevil	Claw	2 5th Pass	1977-03-19	45	M	931-138-3011	2020-09-03	41811
330-58-5390	Tomaso	Stanner	0125 Bellgrove ...	1990-08-22	32	M	511-441-8630	2021-07-27	10295
339-18-4973	Stormie	Zuenelli	79 Farwell Point	1994-04-21	28	F	126-266-8806	2011-12-30	13595
340-11-1307	Rebekkah	Boundey	7944 Dunning P...	1976-04-23	46	F	212-839-7527	2020-05-28	8914
342-64-6342	Willem	Wickman	84 Moland Plaza	1979-10-11	43	M	517-508-3518	2016-11-17	11575
348-12-2135	Rosemonde	Aitcheson	1424 Acker Plaza	1989-03-14	33	F	674-954-5940	2010-08-19	33985
351-65-5089	Mary	Foort	05 Ronald Rega...	1963-10-23	59	F	506-503-6057	2014-02-25	32331
363-22-6910	Bobbe	Littrik	4804 Main Street	1990-06-26	32	F	526-544-4059	2016-07-19	35800
382-99-8679	Dory	Kestin	57 Waxwing Pass	1988-01-22	34	F	515-538-3355	2016-04-24	10380
393-76-8644	Milissent	Crutch	922 Monterey P...	1973-03-31	49	F	267-478-6982	2018-12-25	39494
395-97-9908	Avigdor	Philipard	50333 Delaware...	1978-03-13	44	M	792-494-7385	2019-07-30	29134
396-88-6001	Tod	Martygin	2 Hintze Hill	1981-02-15	41	M	527-703-2133	2014-11-19	11129
397-96-6194	Rustin	Dorward	7 Shopko Park	1971-04-02	51	M	931-829-0387	2013-06-06	39007

- **After exec sp_UpdateSalary**

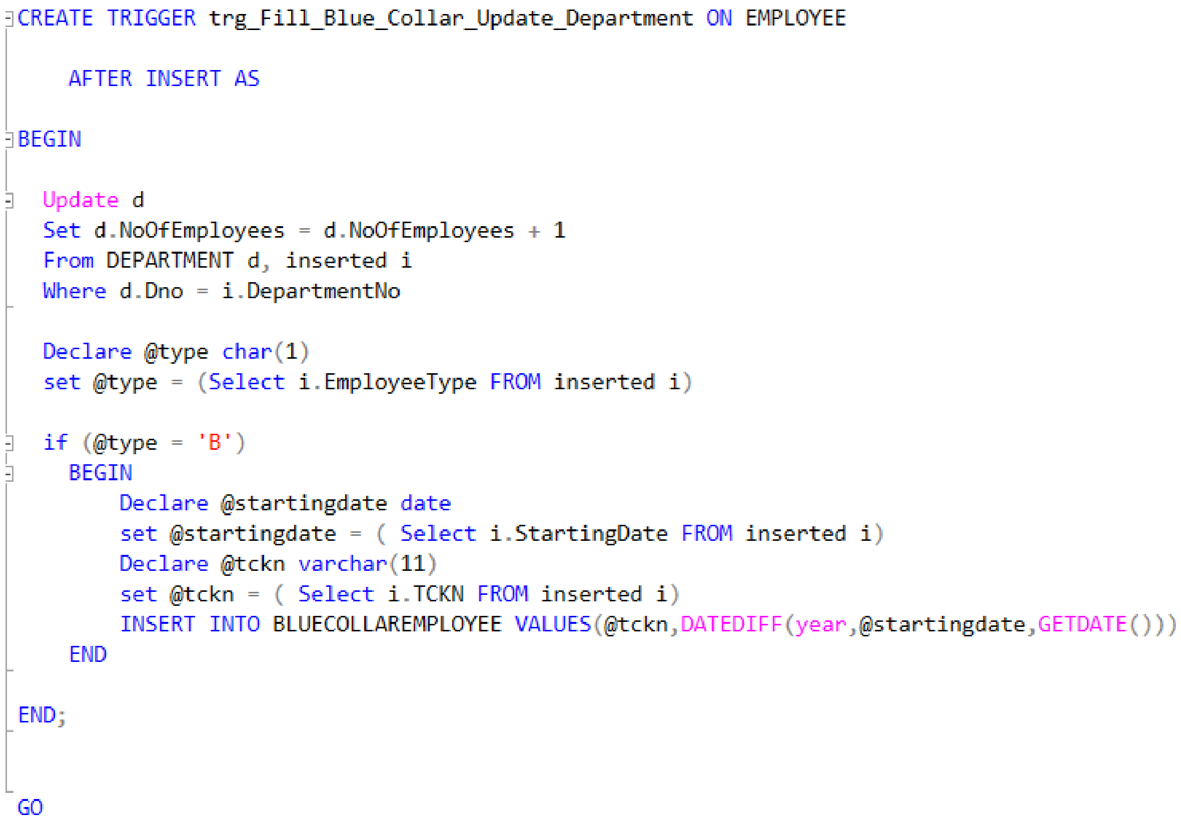
→ **EMPLOYEE TABLE**



TCKN	FirstName	LastName	Address	Birthdate	Age	Gender	PhoneNumber	StartingDate	Salary
201-87-8164	Archibald	Stendall	58005 Kensingt...	1998-06-03	24	M	538-338-6182	2022-12-12	12430
210-51-1946	Mariquilla	Hastelow	6146 Anniversar...	1977-04-21	45	F	695-193-7170	2017-03-14	54531,25
211-58-8290	Laurette	Dun	257 Blue Bill Par...	1960-03-15	62	F	489-632-3529	2020-08-07	18393,75
212-96-2292	Hobey	Worviell	35 Nobel Park	1964-06-11	58	M	221-518-8109	2012-12-19	12951,25
214-10-8104	Jesus	Eastmond	29045 Dorton Pl...	1978-08-06	44	M	428-542-7874	2011-08-03	15550
215-86-1949	Phillip	Witcomb	76274 Graedel ...	1978-05-21	44	M	428-422-6675	2016-07-06	17435
217-05-8759	Jed	Samsonsen	08856 Evergree...	1993-04-22	29	M	292-521-9237	2019-09-13	11296,25
223-13-0549	Cynthy	Colgrave	48 Buena Vista ...	1969-08-04	53	F	651-264-4263	2019-10-14	16526,25
228-42-7138	Dorena	Roelofs	03 Village Gree...	1989-11-20	33	F	358-820-6763	2015-11-22	15187,5
243-57-0502	Bevvy	Maben	3244 Nancy Alley	1971-12-20	51	F	782-529-3789	2022-02-27	12692,5
248-12-7889	Candis	Mitkcov	1 Fordem Pass	1999-06-15	23	F	751-442-4356	2018-09-19	11447,5
273-65-4523	Pietra	Fowden	578 Kipling Jun...	1981-05-29	41	F	838-324-1070	2012-07-10	17813,75
289-16-6128	Eva	Sillick	20 Bobwhite Av...	1965-06-04	57	F	874-920-5204	2016-10-23	43313,75
290-62-1316	Nevil	Claw	2 5th Pass	1977-03-19	45	M	931-138-3011	2020-09-03	52263,75
330-58-5390	Tomaso	Stanner	0125 Bellgrove ...	1990-08-22	32	M	511-441-8630	2021-07-27	12868,75
339-18-4973	Stormie	Zuenelli	79 Farwell Point	1994-04-21	28	F	126-266-8806	2011-12-30	16993,75
340-11-1307	Rebekkah	Boundey	7944 Dunning P...	1976-04-23	46	F	212-839-7527	2020-05-28	11142,5
342-64-6342	Willem	Wickman	84 Moland Plaza	1979-10-11	43	M	517-508-3518	2016-11-17	14468,75
348-12-2135	Rosemonde	Aitcheson	1424 Acker Plaza	1989-03-14	33	F	674-954-5940	2010-08-19	42481,25
351-65-5089	Mary	Foort	05 Ronald Rega...	1963-10-23	59	F	506-503-6057	2014-02-25	40413,75
363-22-6910	Bobbe	Littrik	4804 Main Street	1990-06-26	32	F	526-544-4059	2016-07-19	44750
382-99-8679	Dory	Kestin	57 Waxwing Pass	1988-01-22	34	F	515-538-3355	2016-04-24	12975
393-76-8644	Milissent	Crutch	922 Monterey P...	1973-03-31	49	F	267-478-6982	2018-12-25	49367,5
395-97-9908	Avigdor	Philipard	50333 Delaware...	1978-03-13	44	M	792-494-7385	2019-07-30	36417,5
396-88-6001	Tod	Martygin	2 Hintze Hill	1981-02-15	41	M	527-703-2133	2014-11-19	13911,25
397-96-6194	Rustin	Dorward	7 Shopko Park	1971-04-02	51	M	931-829-0387	2013-06-06	48758,75

TRIGGERS

- **Name:** Fill_BlueColar_Update_Department
- **Definition:** Type of this trigger is ‘After insert’. When new employee added to the **EMPLOYEE** table, the NoOfEmployees field in the **DEPARTMENT** table is automatically increased by one according to the given employee’s department number. Then, if the added worker type is blue-collar, the columns in the **BLUECOLLAREMPLOYEE** table are automatically filled.
- **Trigger:**

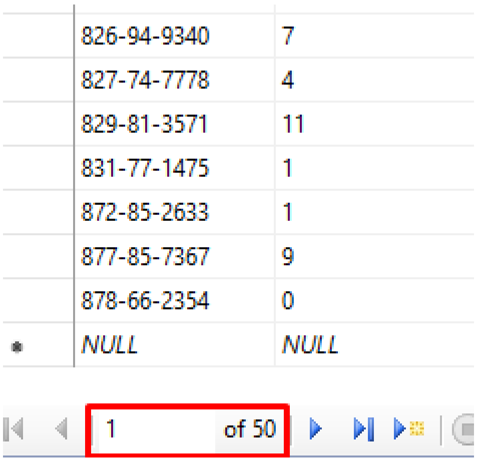


```
CREATE TRIGGER trg_Fill_Blue_Collar_Update_Department ON EMPLOYEE

    AFTER INSERT AS

BEGIN

  Update d
  Set d.NoOfEmployees = d.NoOfEmployees + 1
  From DEPARTMENT d, inserted i
  Where d.Dno = i.DepartmentNo

  Declare @type char(1)
  set @type = (Select i.EmployeeType FROM inserted i)

  if (@type = 'B')
    BEGIN
        Declare @startingdate date
        set @startingdate = ( Select i.StartingDate FROM inserted i)
        Declare @tckn varchar(11)
        set @tckn = ( Select i.TCKN FROM inserted i)
        INSERT INTO BLUECOLLAREMPLOYEE VALUES(@tckn,DATEDIFF(year,@startingdate,GETDATE()))
    END

END;


GO
```

```
insert into EMPLOYEE (TCKN, FirstName, LastName, Address,
Birthdate, Gender, PhoneNumber, StartingDate, Salary,
EmployeeType, DepartmentNo) values ('111-11-1111', 'Fatih',
'Terim', 'Florya', '1953-03-15', 'M', '111-111-1905',
'8/7/2015', 19050, 'B', 10);
```

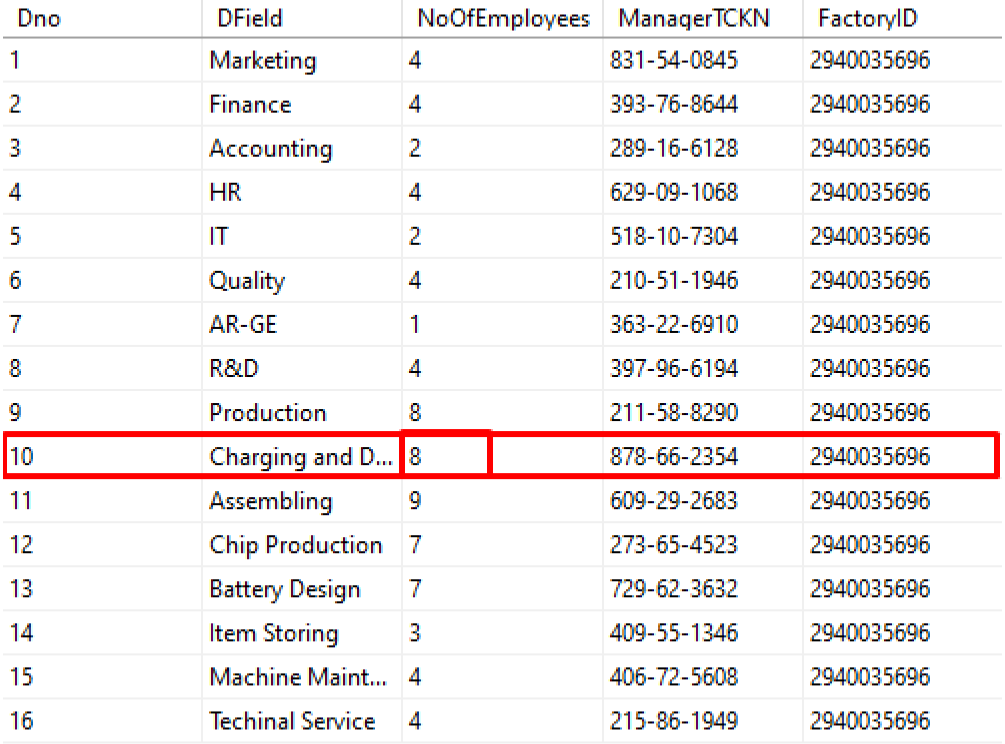
- **Before insert:**

→ BLUECOLLAR_TABLE



826-94-9340	7
827-74-7778	4
829-81-3571	11
831-77-1475	1
872-85-2633	1
877-85-7367	9
878-66-2354	0
NULL	*NULL*

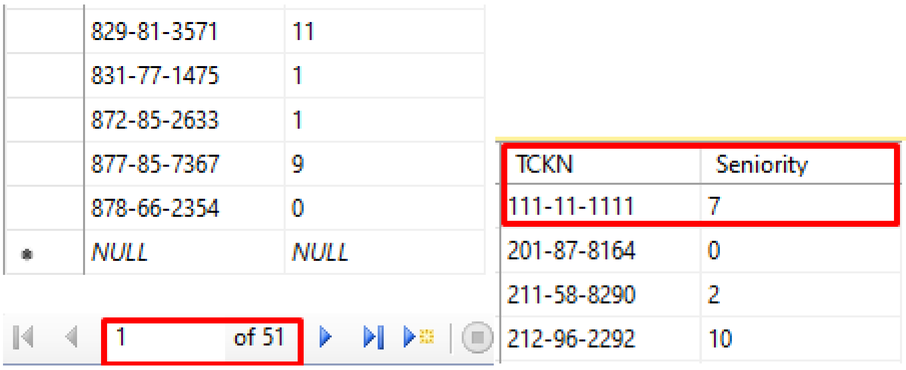
→ DEPARTMENT_TABLE



Dno	DField	NoOfEmployees	ManagerTCKN	FactoryID
1	Marketing	4	831-54-0845	2940035696
2	Finance	4	393-76-8644	2940035696
3	Accounting	2	289-16-6128	2940035696
4	HR	4	629-09-1068	2940035696
5	IT	2	518-10-7304	2940035696
6	Quality	4	210-51-1946	2940035696
7	AR-GE	1	363-22-6910	2940035696
8	R&D	4	397-96-6194	2940035696
9	Production	8	211-58-8290	2940035696
10	Charging and D...	8	878-66-2354	2940035696
11	Assembling	9	609-29-2683	2940035696
12	Chip Production	7	273-65-4523	2940035696
13	Battery Design	7	729-62-3632	2940035696
14	Item Storing	3	409-55-1346	2940035696
15	Machine Maint...	4	406-72-5608	2940035696
16	Techinal Service	4	215-86-1949	2940035696

- **After trigger process:**

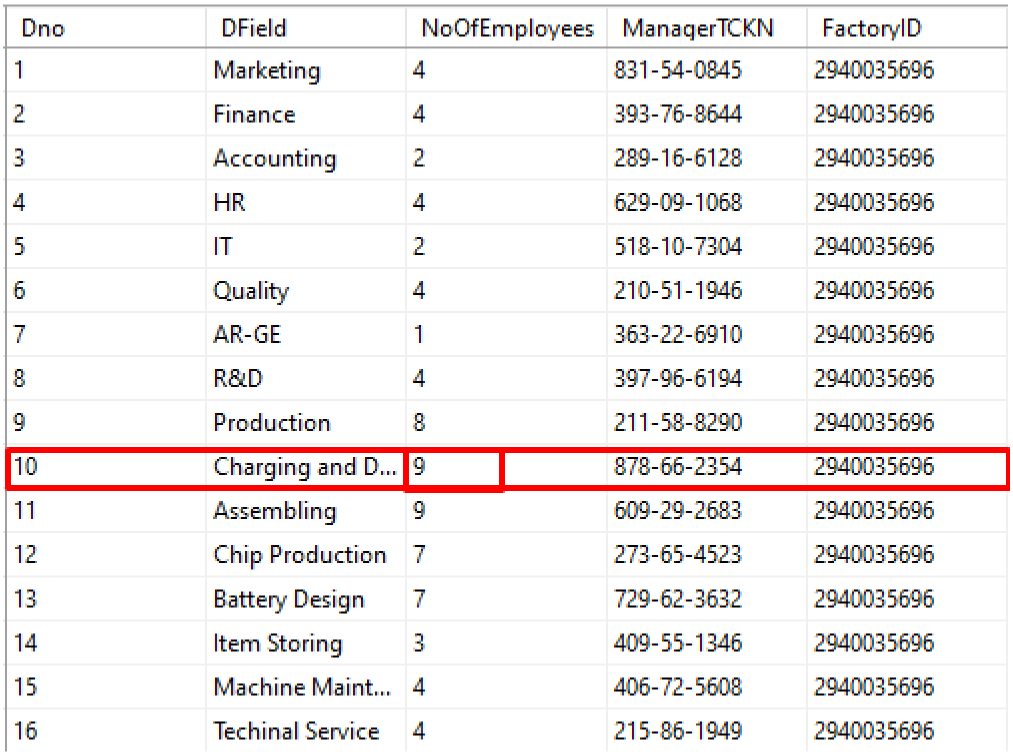
→ **BLUECOLLAR_TABLE**



	829-81-3571	11
	831-77-1475	1
	872-85-2633	1
	877-85-7367	9
	878-66-2354	0
*	*NULL*	*NULL*

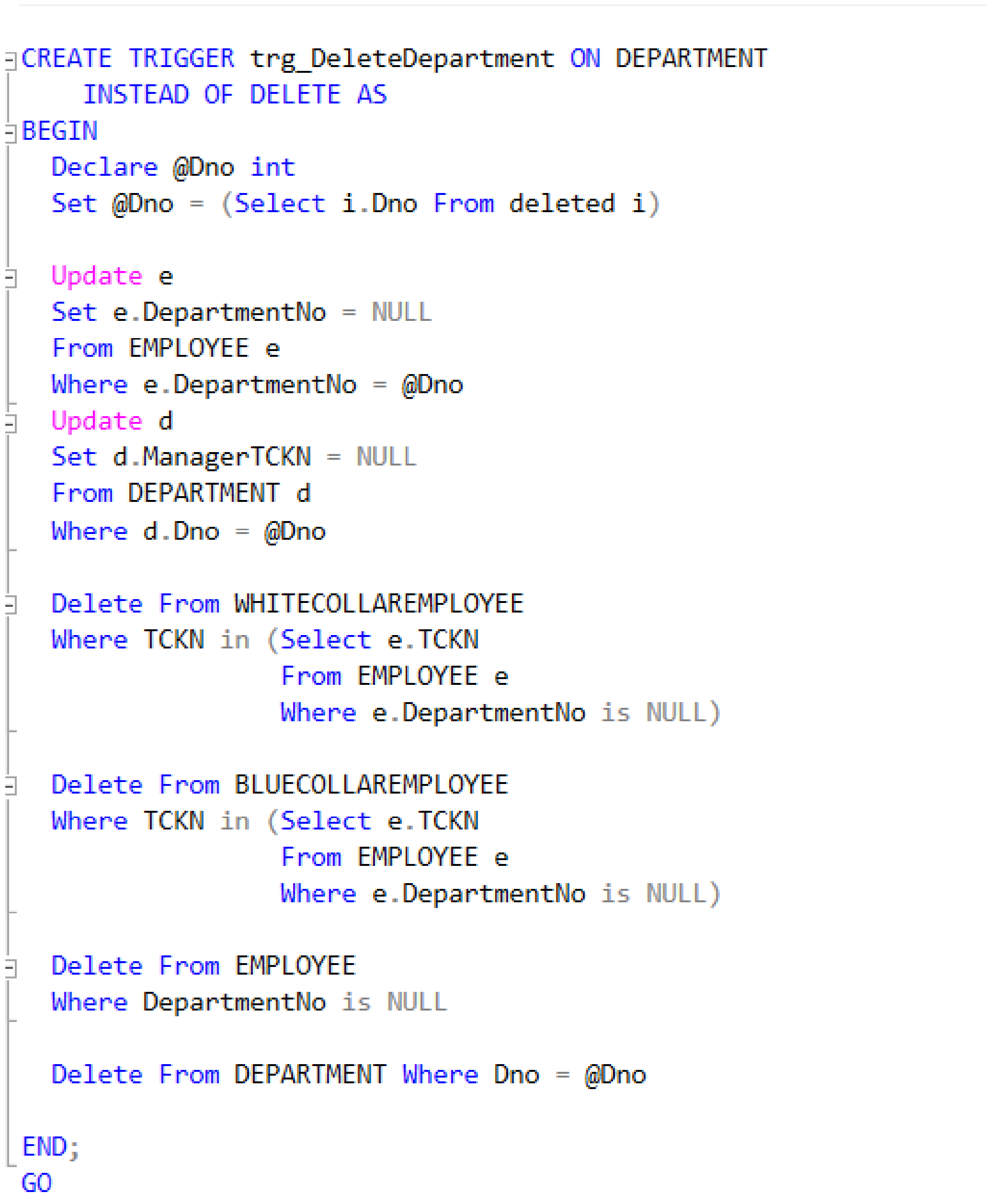
TCKN	Seniority
111-11-1111	7
201-87-8164	0
211-58-8290	2
212-96-2292	10

→ **DEPARTMENT_TABLE**



Dno	DField	NoOfEmployees	ManagerTCKN	FactoryID
1	Marketing	4	831-54-0845	2940035696
2	Finance	4	393-76-8644	2940035696
3	Accounting	2	289-16-6128	2940035696
4	HR	4	629-09-1068	2940035696
5	IT	2	518-10-7304	2940035696
6	Quality	4	210-51-1946	2940035696
7	AR-GE	1	363-22-6910	2940035696
8	R&D	4	397-96-6194	2940035696
9	Production	8	211-58-8290	2940035696
10	Charging and D...	9	878-66-2354	2940035696
11	Assembling	9	609-29-2683	2940035696
12	Chip Production	7	273-65-4523	2940035696
13	Battery Design	7	729-62-3632	2940035696
14	Item Storing	3	409-55-1346	2940035696
15	Machine Maint...	4	406-72-5608	2940035696
16	Techinal Service	4	215-86-1949	2940035696

- **Name:** trg_DeleteDepartment
- **Definition:** Type of this trigger is ‘Instead of delete’. When a department is deleted, firstly all employees working there are deleted from **EMPLOYEE** and its subtype tables. After that the department is deleted from the **DEPARTMENT** table.
- **Trigger:**

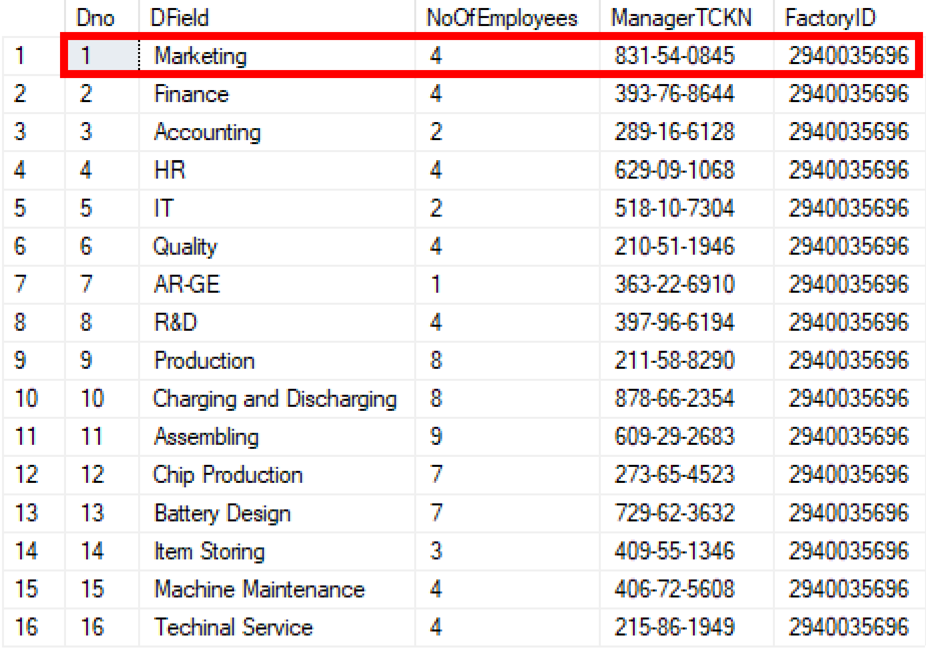


```
CREATE TRIGGER trg_DeleteDepartment ON DEPARTMENT
    INSTEAD OF DELETE AS
BEGIN
  Declare @Dno int
  Set @Dno = (Select i.Dno From deleted i)

  Update e
  Set e.DepartmentNo = NULL
  From EMPLOYEE e
  Where e.DepartmentNo = @Dno
  Update d
  Set d.ManagerTCKN = NULL
  From DEPARTMENT d
  Where d.Dno = @Dno

  Delete From WHITECOLLAREMPLOYEE
  Where TCKN in (Select e.TCKN
                 From EMPLOYEE e
                 Where e.DepartmentNo is NULL)

  Delete From BLUECOLLAREMPLOYEE
  Where TCKN in (Select e.TCKN
                 From EMPLOYEE e
                 Where e.DepartmentNo is NULL)

  Delete From EMPLOYEE
  Where DepartmentNo is NULL

  Delete From DEPARTMENT Where Dno = @Dno

END;
GO
```

```
Delete From DEPARTMENT Where Dno = 1
```

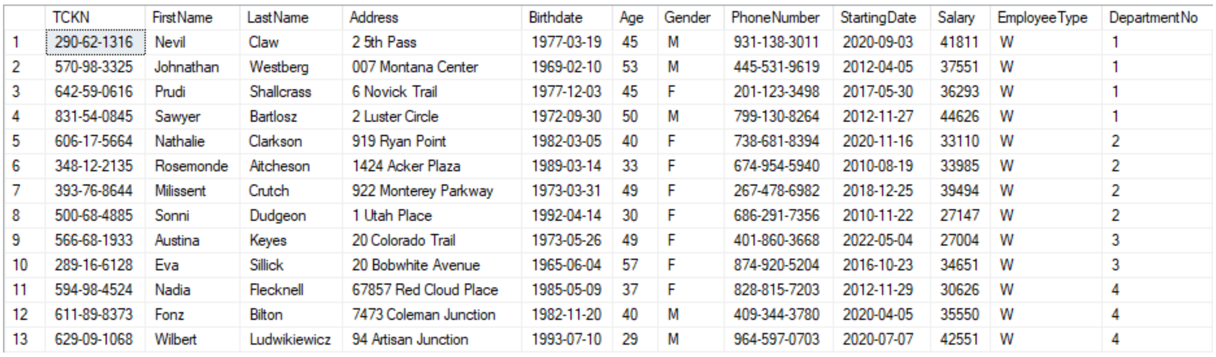
- **Before delete:**

→ **DEPARTMENT_TABLE**



	Dno	DField	NoOfEmployees	ManagerTCKN	FactoryID
1	1	Marketing	4	831-54-0845	2940035696
2	2	Finance	4	393-76-8644	2940035696
3	3	Accounting	2	289-16-6128	2940035696
4	4	HR	4	629-09-1068	2940035696
5	5	IT	2	518-10-7304	2940035696
6	6	Quality	4	210-51-1946	2940035696
7	7	AR-GE	1	363-22-6910	2940035696
8	8	R&D	4	397-96-6194	2940035696
9	9	Production	8	211-58-8290	2940035696
10	10	Charging and Discharging	8	878-66-2354	2940035696
11	11	Assembling	9	609-29-2683	2940035696
12	12	Chip Production	7	273-65-4523	2940035696
13	13	Battery Design	7	729-62-3632	2940035696
14	14	Item Storing	3	409-55-1346	2940035696
15	15	Machine Maintenance	4	406-72-5608	2940035696
16	16	Techinal Service	4	215-86-1949	2940035696

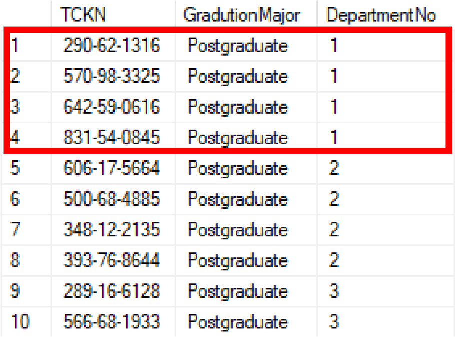
→ **EMPLOYEE_TABLE**

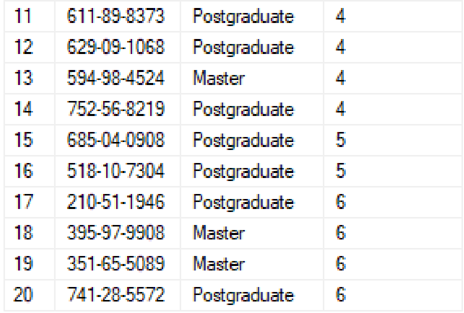


	TCKN	FirstName	LastName	Address	Birthdate	Age	Gender	PhoneNumber	StartingDate	Salary	EmployeeType	DepartmentNo
1	290-62-1316	Nevil	Claw	2 5th Pass	1977-03-19	45	M	931-138-3011	2020-09-03	41811	W	1
2	570-98-3325	Johnathan	Westberg	007 Montana Center	1969-02-10	53	M	445-531-9619	2012-04-05	37551	W	1
3	642-59-0616	Prudi	Shallcrass	6 Novick Trail	1977-12-03	45	F	201-123-3498	2017-05-30	36293	W	1
4	831-54-0845	Sawyer	Bartlosz	2 Luster Circle	1972-09-30	50	M	799-130-8264	2012-11-27	44626	W	1
5	606-17-5664	Nathalie	Clarkson	919 Ryan Point	1982-03-05	40	F	738-681-8394	2020-11-16	33110	W	2
6	348-12-2135	Rosemonde	Aitcheson	1424 Acker Plaza	1989-03-14	33	F	674-954-5940	2010-08-19	33985	W	2
7	393-76-8644	Milissent	Crutch	922 Monterey Parkway	1973-03-31	49	F	267-478-6982	2018-12-25	39494	W	2
8	500-68-4885	Sonni	Dudgeon	1 Utah Place	1992-04-14	30	F	686-291-7356	2010-11-22	27147	W	2
9	566-68-1933	Austina	Keyes	20 Colorado Trail	1973-05-26	49	F	401-860-3668	2022-05-04	27004	W	3
10	289-16-6128	Eva	Sillick	20 Bobwhite Avenue	1965-06-04	57	F	874-920-5204	2016-10-23	34651	W	3
11	594-98-4524	Nadia	Flecknell	67857 Red Cloud Place	1985-05-09	37	F	828-815-7203	2012-11-29	30626	W	4
12	611-89-8373	Fonz	Bilton	7473 Coleman Junction	1982-11-20	40	M	409-344-3780	2020-04-05	35550	W	4
13	629-09-1068	Wilbert	Ludwikiewicz	94 Artisan Junction	1993-07-10	29	M	964-597-0703	2020-07-07	42551	W	4

...

→ **WHITECOLLAREMPLOYEE_TABLE**

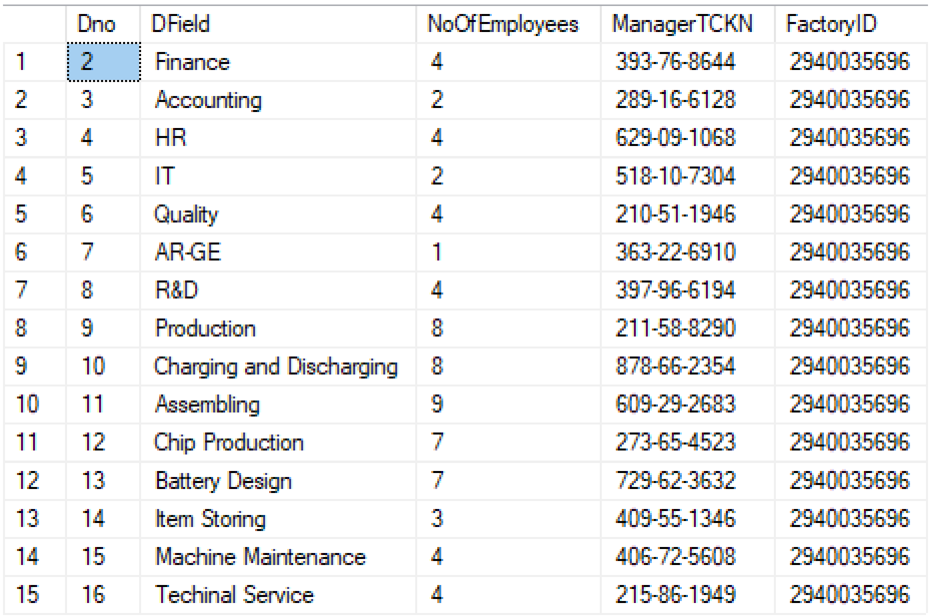




	TCKN	GradutionMajor	DepartmentNo
1	290-62-1316	Postgraduate	1
2	570-98-3325	Postgraduate	1
3	642-59-0616	Postgraduate	1
4	831-54-0845	Postgraduate	1
5	606-17-5664	Postgraduate	2
6	500-68-4885	Postgraduate	2
7	348-12-2135	Postgraduate	2
8	393-76-8644	Postgraduate	2
9	289-16-6128	Postgraduate	3
10	566-68-1933	Postgraduate	3
11	611-89-8373	Postgraduate	4
12	629-09-1068	Postgraduate	4
13	594-98-4524	Master	4
14	752-56-8219	Postgraduate	4
15	685-04-0908	Postgraduate	5
16	518-10-7304	Postgraduate	5
17	210-51-1946	Postgraduate	6
18	395-97-9908	Master	6
19	351-65-5089	Master	6
20	741-28-5572	Postgraduate	6

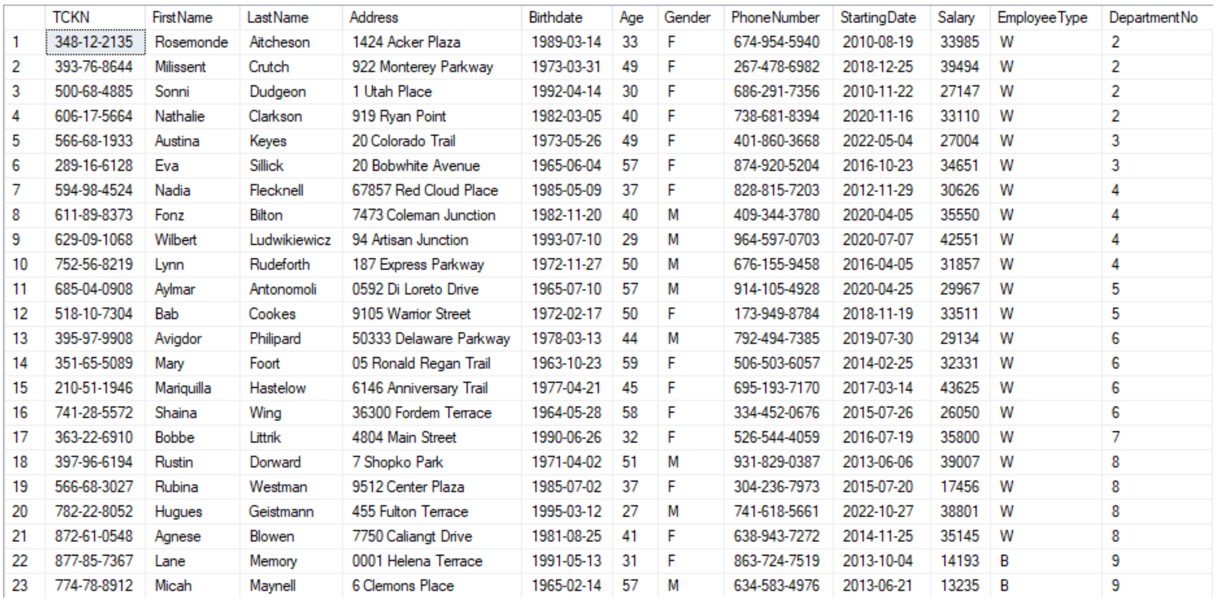
- **After trigger process:**

→ **DEPARTMENT_TABLE**



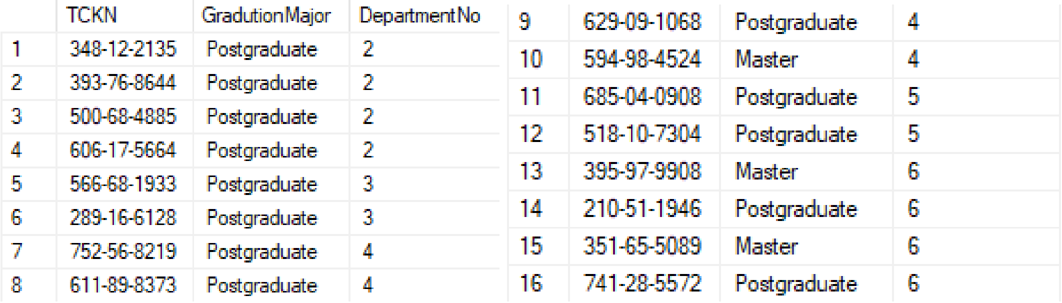
	Dno	DField	NoOfEmployees	ManagerTCKN	FactoryID
1	2	Finance	4	393-76-8644	2940035696
2	3	Accounting	2	289-16-6128	2940035696
3	4	HR	4	629-09-1068	2940035696
4	5	IT	2	518-10-7304	2940035696
5	6	Quality	4	210-51-1946	2940035696
6	7	AR-GE	1	363-22-6910	2940035696
7	8	R&D	4	397-96-6194	2940035696
8	9	Production	8	211-58-8290	2940035696
9	10	Charging and Discharging	8	878-66-2354	2940035696
10	11	Assembling	9	609-29-2683	2940035696
11	12	Chip Production	7	273-65-4523	2940035696
12	13	Battery Design	7	729-62-3632	2940035696
13	14	Item Storing	3	409-55-1346	2940035696
14	15	Machine Maintenance	4	406-72-5608	2940035696
15	16	Techinal Service	4	215-86-1949	2940035696

→ **EMPLOYEE_TABLE**



	TCKN	FirstName	LastName	Address	Birthdate	Age	Gender	PhoneNumber	StartingDate	Salary	EmployeeType	DepartmentNo
1	348-12-2135	Rosemonde	Aitcheson	1424 Acker Plaza	1989-03-14	33	F	674-954-5940	2010-08-19	33985	W	2
2	393-76-8644	Milissent	Crutch	922 Monterey Parkway	1973-03-31	49	F	267-478-6982	2018-12-25	39494	W	2
3	500-68-4885	Sonni	Dudgeon	1 Utah Place	1992-04-14	30	F	686-291-7356	2010-11-22	27147	W	2
4	606-17-5664	Nathalie	Clarkson	919 Ryan Point	1982-03-05	40	F	738-681-8394	2020-11-16	33110	W	2
5	566-68-1933	Austina	Keyes	20 Colorado Trail	1973-05-26	49	F	401-860-3668	2022-05-04	27004	W	3
6	289-16-6128	Eva	Sillick	20 Bobwhite Avenue	1965-06-04	57	F	874-920-5204	2016-10-23	34651	W	3
7	594-98-4524	Nadia	Flecknell	67857 Red Cloud Place	1985-05-09	37	F	828-815-7203	2012-11-29	30626	W	4
8	611-89-8373	Fonz	Bilton	7473 Coleman Junction	1982-11-20	40	M	409-344-3780	2020-04-05	35550	W	4
9	629-09-1068	Wilbert	Ludwikiewicz	94 Artisan Junction	1993-07-10	29	M	964-597-0703	2020-07-07	42551	W	4
10	752-56-8219	Lynn	Rudeforth	187 Express Parkway	1972-11-27	50	M	676-155-9458	2016-04-05	31857	W	4
11	685-04-0908	Aylmar	Antonomoli	0592 Di Loreto Drive	1965-07-10	57	M	914-105-4928	2020-04-25	29967	W	5
12	518-10-7304	Bab	Cookes	9105 Warrior Street	1972-02-17	50	F	173-949-8784	2018-11-19	33511	W	5
13	395-97-9908	Avigdor	Philipard	50333 Delaware Parkway	1978-03-13	44	M	792-494-7385	2019-07-30	29134	W	6
14	351-65-5089	Mary	Foort	05 Ronald Regan Trail	1963-10-23	59	F	506-503-6057	2014-02-25	32331	W	6
15	210-51-1946	Mariquilla	Hastelow	6146 Anniversary Trail	1977-04-21	45	F	695-193-7170	2017-03-14	43625	W	6
16	741-28-5572	Shaina	Wing	36300 Fordem Terrace	1964-05-28	58	F	334-452-0676	2015-07-26	26050	W	6
17	363-22-6910	Bobbe	Littrik	4804 Main Street	1990-06-26	32	F	526-544-4059	2016-07-19	35800	W	7
18	397-96-6194	Rustin	Dorward	7 Shopko Park	1971-04-02	51	M	931-829-0387	2013-06-06	39007	W	8
19	566-68-3027	Rubina	Westman	9512 Center Plaza	1985-07-02	37	F	304-236-7973	2015-07-20	17456	W	8
20	782-22-8052	Hugues	Geistmann	455 Fulton Terrace	1995-03-12	27	M	741-618-5661	2022-10-27	38801	W	8
21	872-61-0548	Agnese	Blowen	7750 Caliangt Drive	1981-08-25	41	F	638-943-7272	2014-11-25	35145	W	8
22	877-85-7367	Lane	Memory	0001 Helena Terrace	1991-05-13	31	F	863-724-7519	2013-10-04	14193	B	9
23	774-78-8912	Micah	Maynell	6 Clemons Place	1965-02-14	57	M	634-583-4976	2013-06-21	13235	B	9

→ **WHITECOLLAREMPLOYEE_TABLE**



	TCKN	GradutionMajor	DepartmentNo
1	348-12-2135	Postgraduate	2
2	393-76-8644	Postgraduate	2
3	500-68-4885	Postgraduate	2
4	606-17-5664	Postgraduate	2
5	566-68-1933	Postgraduate	3
6	289-16-6128	Postgraduate	3
7	752-56-8219	Postgraduate	4
8	611-89-8373	Postgraduate	4
9	629-09-1068	Postgraduate	4
10	594-98-4524	Master	4
11	685-04-0908	Postgraduate	5
12	518-10-7304	Postgraduate	5
13	395-97-9908	Master	6
14	210-51-1946	Postgraduate	6
15	351-65-5089	Master	6
16	741-28-5572	Postgraduate	6