ELEVATOR MANAGEMENT SYSTEM

Group No 10:

- Akshita Barot - 002704943
- Aniruddha Tambe - 002101113
- Forum Bhatt - 002985519
- Kinjal Thakkar - 001568960
- Siddhant Kohli - 002108396

Main Contents

Introduction

Entity Relationship Diagram

Database Implementation

Reports and Visualization

Conclusion

Database Purpose

- To design & implement a database that maintains consistent records of client data
- Introduce sales pipeline tracking & accurate business KPIs computation
- Introduce call-backs, sales & maintenance jobs for tracking business entities

Business Problems Addressed

ALLOW ELEVATOR TEAM TO GENERATE DESCRIPTIVE REPORTS

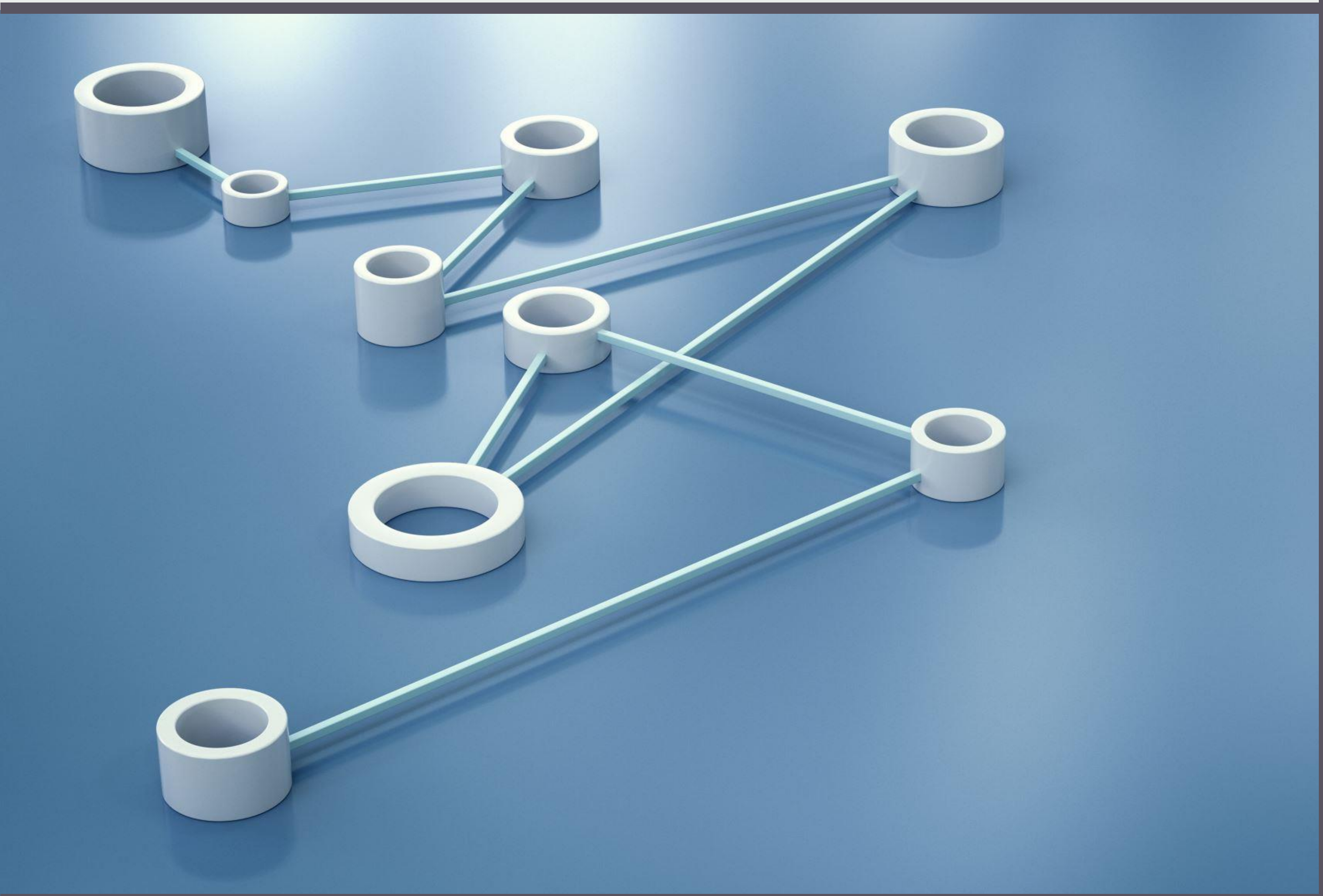
IMPLEMENT CALLBACK (COMPLAINTS) REGISTERING

IMPLEMENT MAINTENANCE JOB PIPELINE

IMPLEMENT YTD REVENUE PIPELINE

GENERATE BUSINESS INSIGHTS

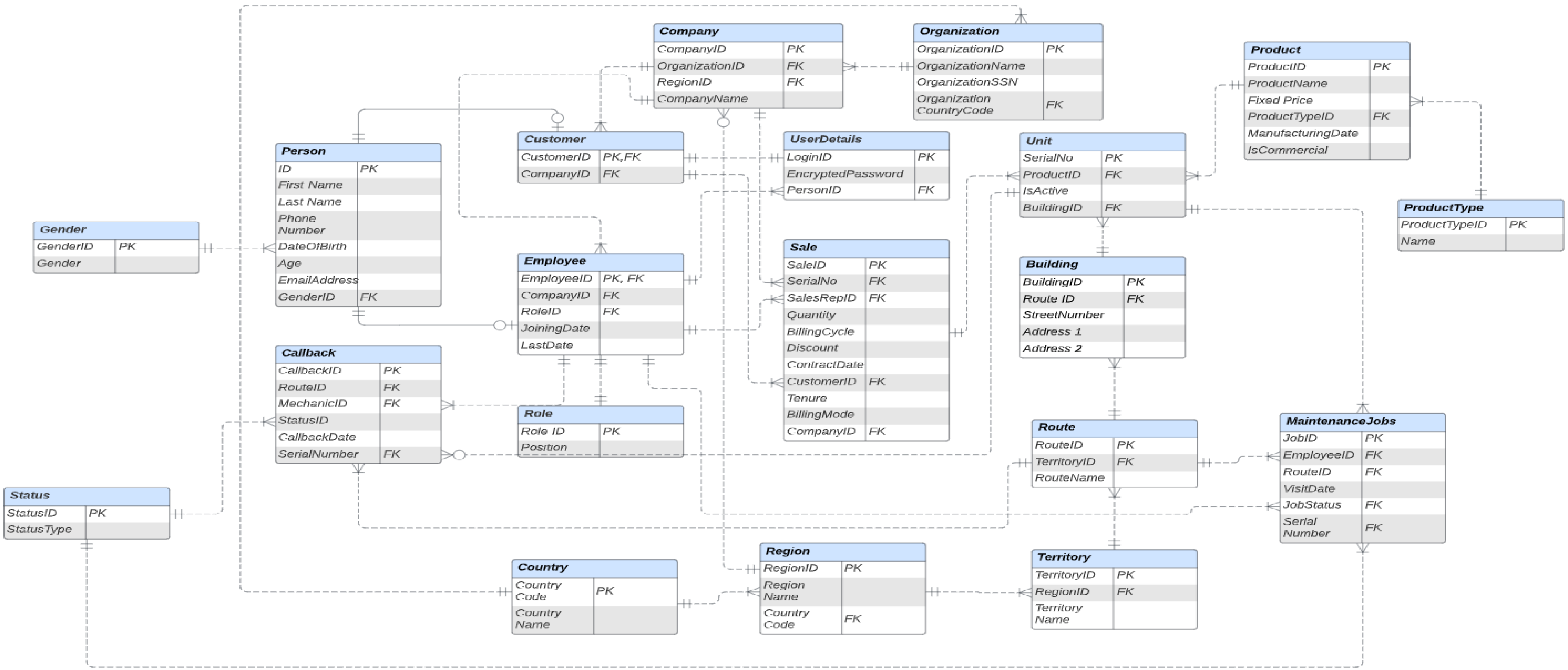
INVENTORY & MERCHANDISE MANAGEMENT



ENTITY RELATIONSHIP DIAGRAM

DAMG 6210 - Elevators Management ERD

Akshita Barot, Aniruddha Tambe, Forum
Bhatt, Kinjal Thakkar, Siddhant Kohli



Company	
CompanyID	PK
OrganizationID	FK
RegionID	FK
CompanyName	

Organization	
OrganizationID	PK
OrganizationName	
OrganizationSSN	
Organization CountryCode	FK

Product	
ProductID	PK
ProductName	
Fixed Price	
ProductTypeID	FK
ManufacturingDate	
IsCommercial	

Person	
ID	PK
First Name	
Last Name	
Phone Number	
DateOfBirth	
Age	
EmailAddress	
GenderID	FK

Customer	
CustomerID	PK,FK
CompanyID	FK

UserDetails	
LoginID	PK
EncryptedPassword	
PersonID	FK

Unit	
SerialNo	PK
ProductID	FK
IsActive	
BuildingID	FK

ProductType	
ProductTypeID	PK
Name	

Gender	
GenderID	PK
Gender	

Employee	
EmployeeID	PK, FK
CompanyID	FK
RoleID	FK
JoiningDate	
LastDate	

Sale	
SaleID	PK
SerialNo	FK
SalesRepID	FK
Quantity	
BillingCycle	
Discount	
ContractDate	
CustomerID	FK
Tenure	
BillingMode	
CompanyID	FK

Building	
BuildingID	PK
Route ID	FK
StreetNumber	
Address 1	
Address 2	

Callback	
CallbackID	PK
RouteID	FK
MechanicID	FK
StatusID	
CallbackDate	
SerialNumber	FK

Role	
Role ID	PK
Position	

Route	
RouteID	PK
TerritoryID	FK
RouteName	

MaintenanceJobs	
JobID	PK
EmployeeID	FK
RouteID	FK
VisitDate	
JobStatus	FK
Serial Number	FK

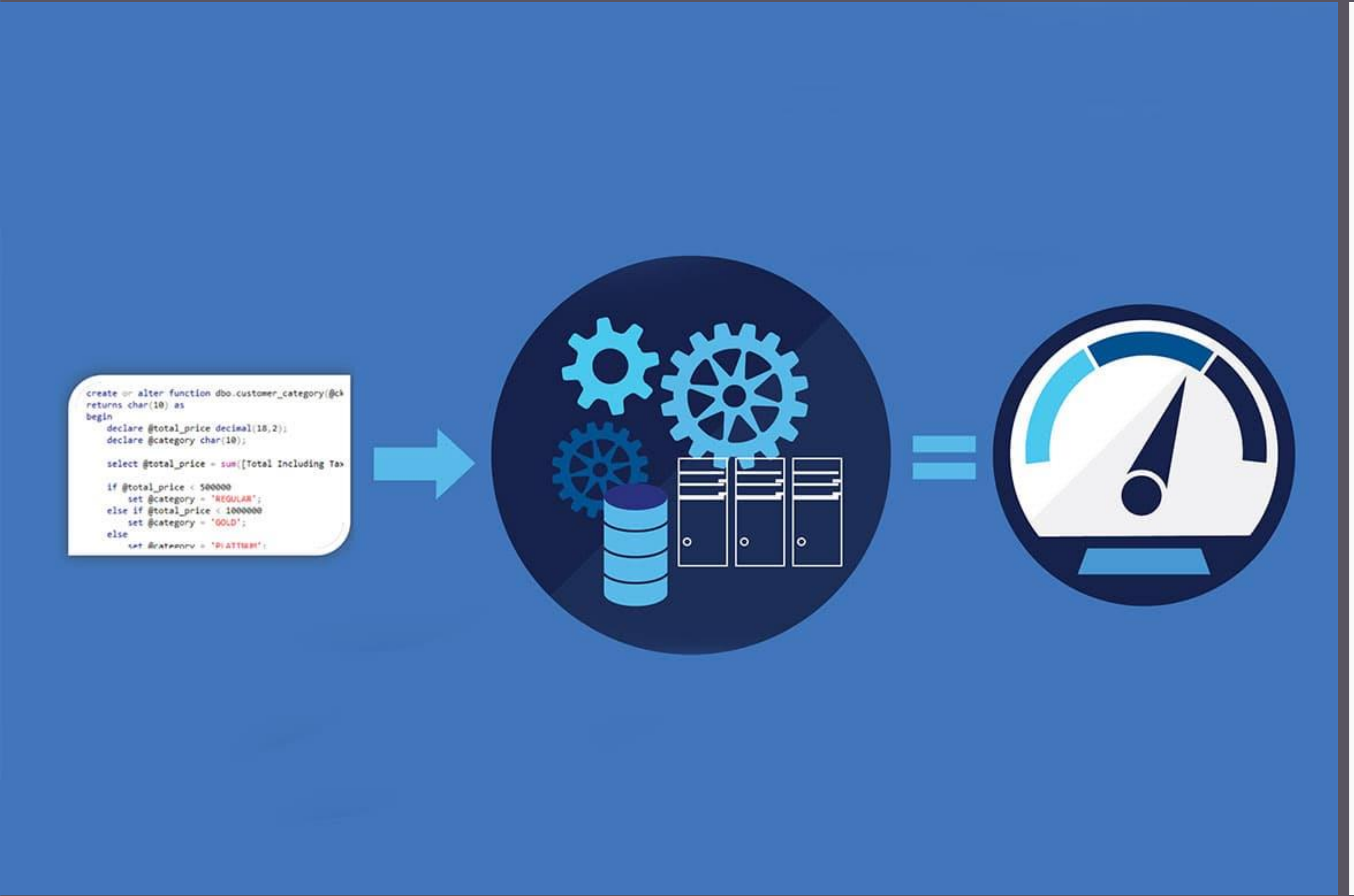
Status	
StatusID	PK
StatusType	

Country	
Country Code	PK
Country Name	

Region	
RegionID	PK
Region Name	
Country Code	FK

Territory	
TerritoryID	PK
RegionID	FK
Territory Name	

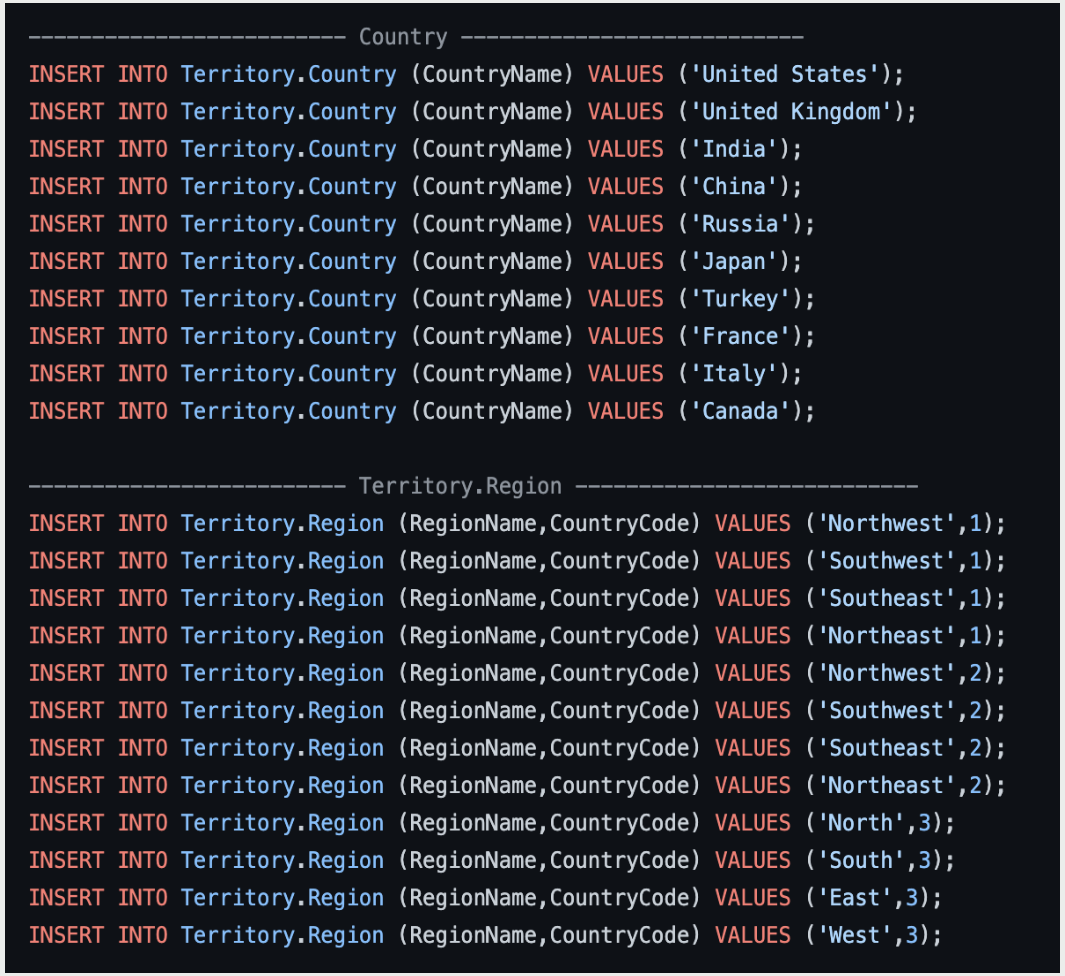
DATABASE IMPLEMENTATIONS



```
create or alter function dbo.customer_category(@ck
returns char(10) as
begin
    declare @total_price decimal(18,2);
    declare @category char(10);

    select @total_price = sum([Total Including Tax

    if @total_price < 500000
        set @category = 'REGULAR';
    else if @total_price < 1000000
        set @category = 'GOLD';
    else
        set @category = 'PLATINUM';
```

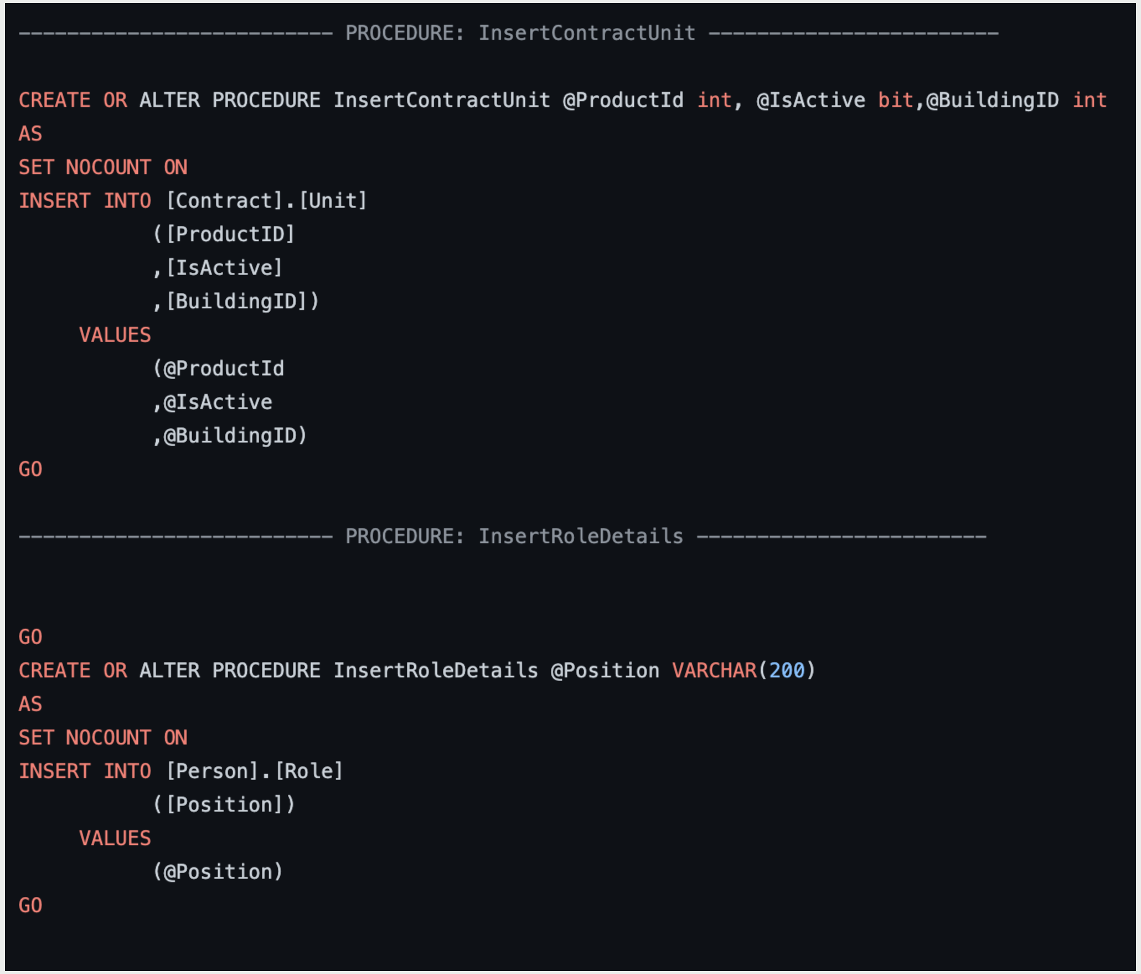
Create Database Framework

Insert Queries

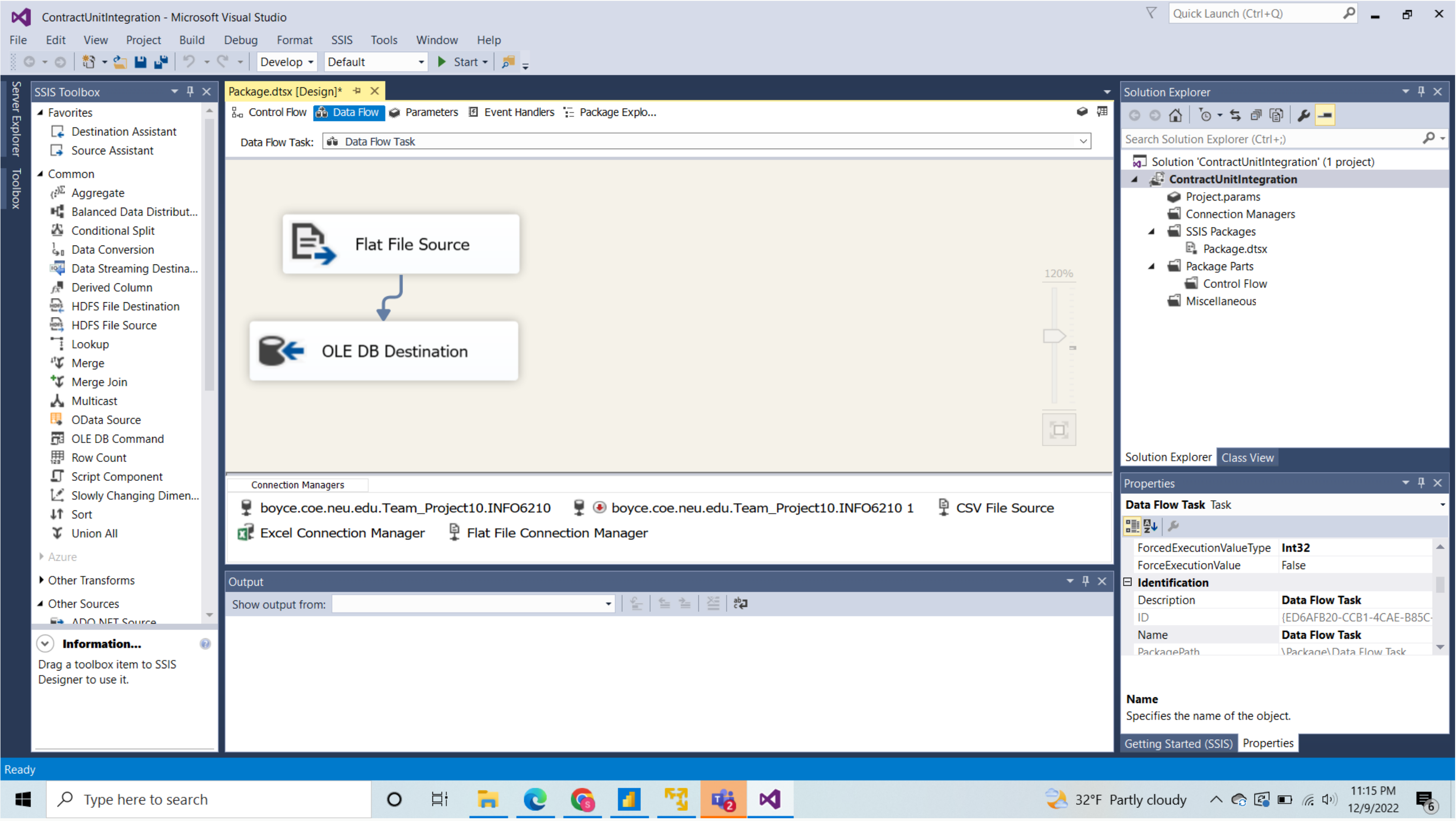


```sql
-------------------------- Country ----------------------------
INSERT INTO Territory.Country (CountryName) VALUES ('United States');
INSERT INTO Territory.Country (CountryName) VALUES ('United Kingdom');
INSERT INTO Territory.Country (CountryName) VALUES ('India');
INSERT INTO Territory.Country (CountryName) VALUES ('China');
INSERT INTO Territory.Country (CountryName) VALUES ('Russia');
INSERT INTO Territory.Country (CountryName) VALUES ('Japan');
INSERT INTO Territory.Country (CountryName) VALUES ('Turkey');
INSERT INTO Territory.Country (CountryName) VALUES ('France');
INSERT INTO Territory.Country (CountryName) VALUES ('Italy');
INSERT INTO Territory.Country (CountryName) VALUES ('Canada');

-------------------------- Territory.Region ---------------------------
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Northwest',1);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Southwest',1);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Southeast',1);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Northeast',1);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Northwest',2);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Southwest',2);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Southeast',2);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('Northeast',2);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('North',3);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('South',3);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('East',3);
INSERT INTO Territory.Region (RegionName,CountryCode) VALUES ('West',3);
```

Insert Queries by Procedures

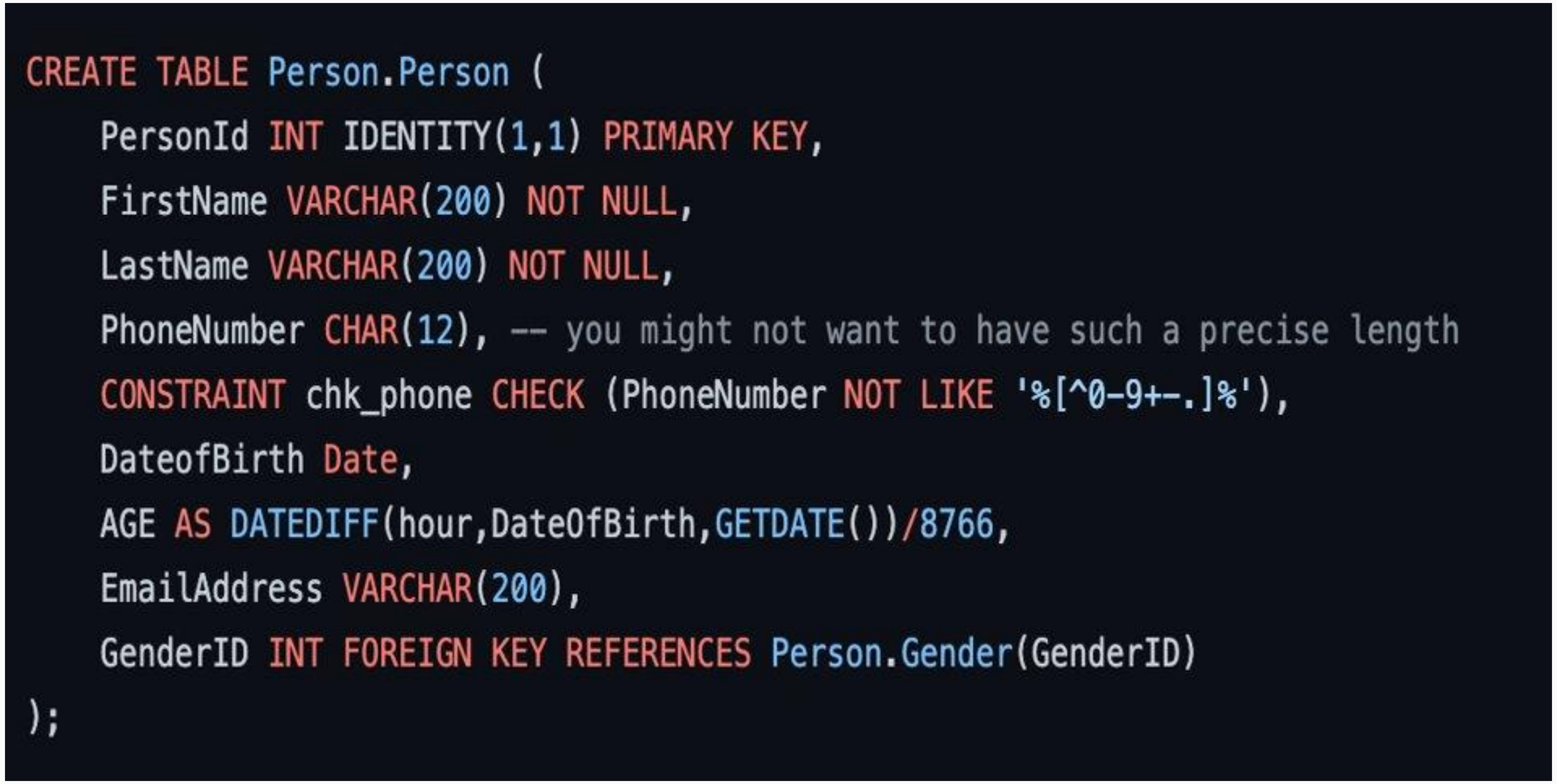


```sql
-------------------------- PROCEDURE: InsertContractUnit ------------------------

CREATE OR ALTER PROCEDURE InsertContractUnit @ProductId int, @IsActive bit,@BuildingID int
AS
SET NOCOUNT ON
INSERT INTO [Contract].[Unit]
           ([ProductID]
           ,[IsActive]
           ,[BuildingID])
     VALUES
           (@ProductId
           ,@IsActive
           ,@BuildingID)
GO

-------------------------- PROCEDURE: InsertRoleDetails ------------------------



GO
CREATE OR ALTER PROCEDURE InsertRoleDetails @Position VARCHAR(200)
AS
SET NOCOUNT ON
INSERT INTO [Person].[Role]
           ([Position])
     VALUES
           (@Position)
GO
```



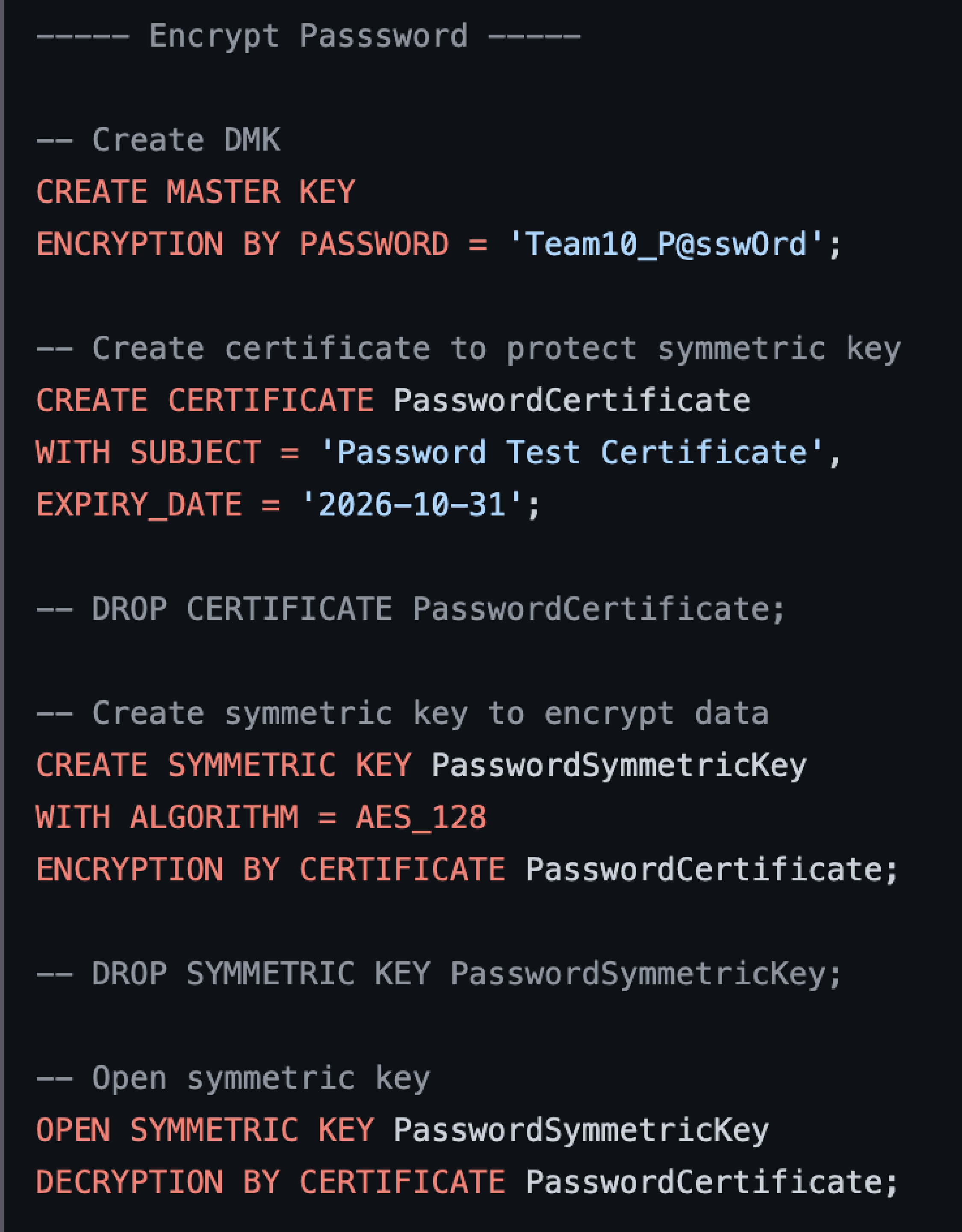
DATA IMPORT WIZARD

COMPUTED COLUMNS



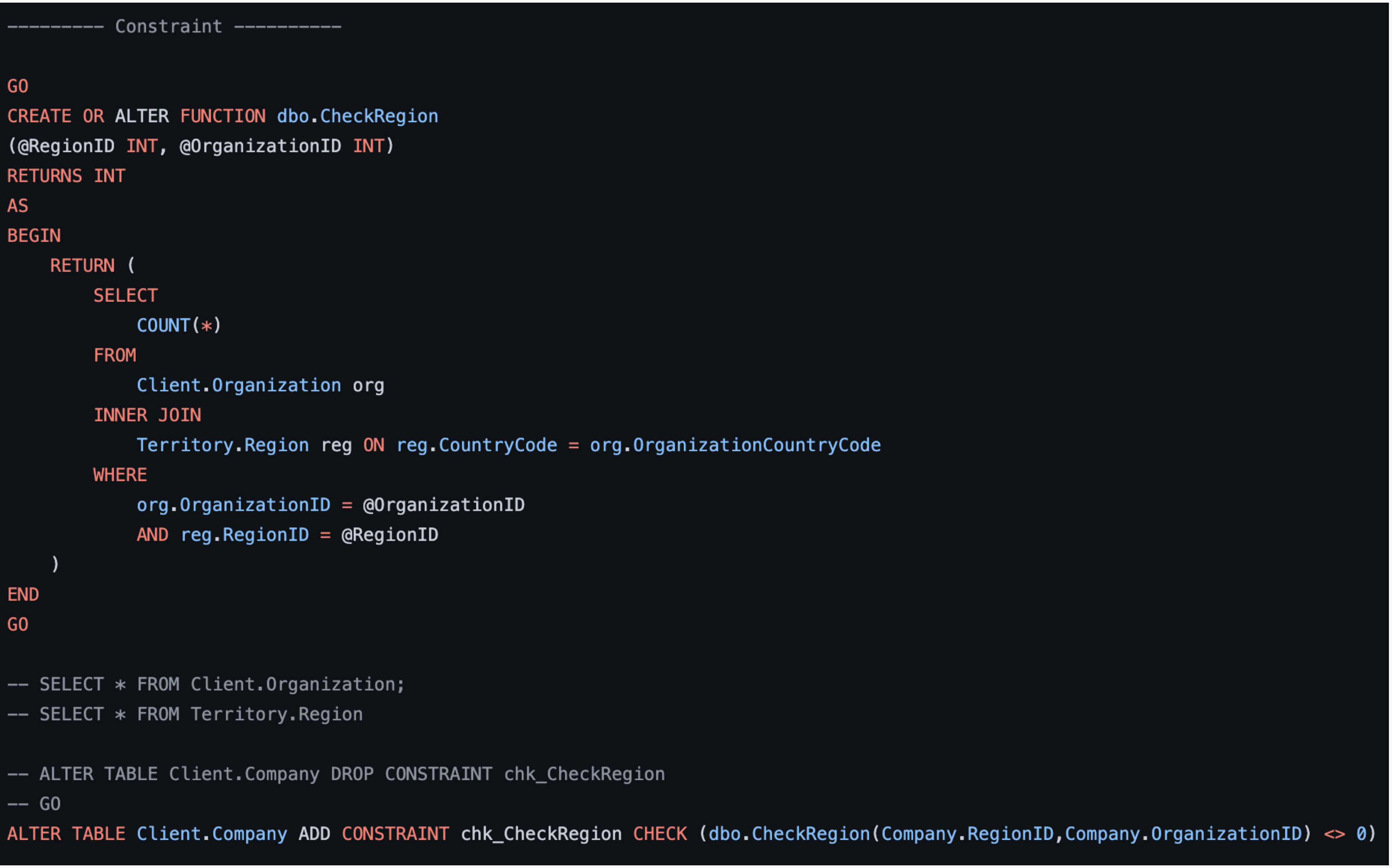
```
CREATE TABLE Person.Person (
    PersonId INT IDENTITY(1,1) PRIMARY KEY,
    FirstName VARCHAR(200) NOT NULL,
    LastName VARCHAR(200) NOT NULL,
    PhoneNumber CHAR(12), -- you might not want to have such a precise length
    CONSTRAINT chk_phone CHECK (PhoneNumber NOT LIKE '%[^0-9+-.]%'),
    DateofBirth Date,
    AGE AS DATEDIFF(hour,DateOfBirth,GETDATE())/8766,
    EmailAddress VARCHAR(200),
    GenderID INT FOREIGN KEY REFERENCES Person.Gender(GenderID)
);
```

Column Data Encryption:
On User Password



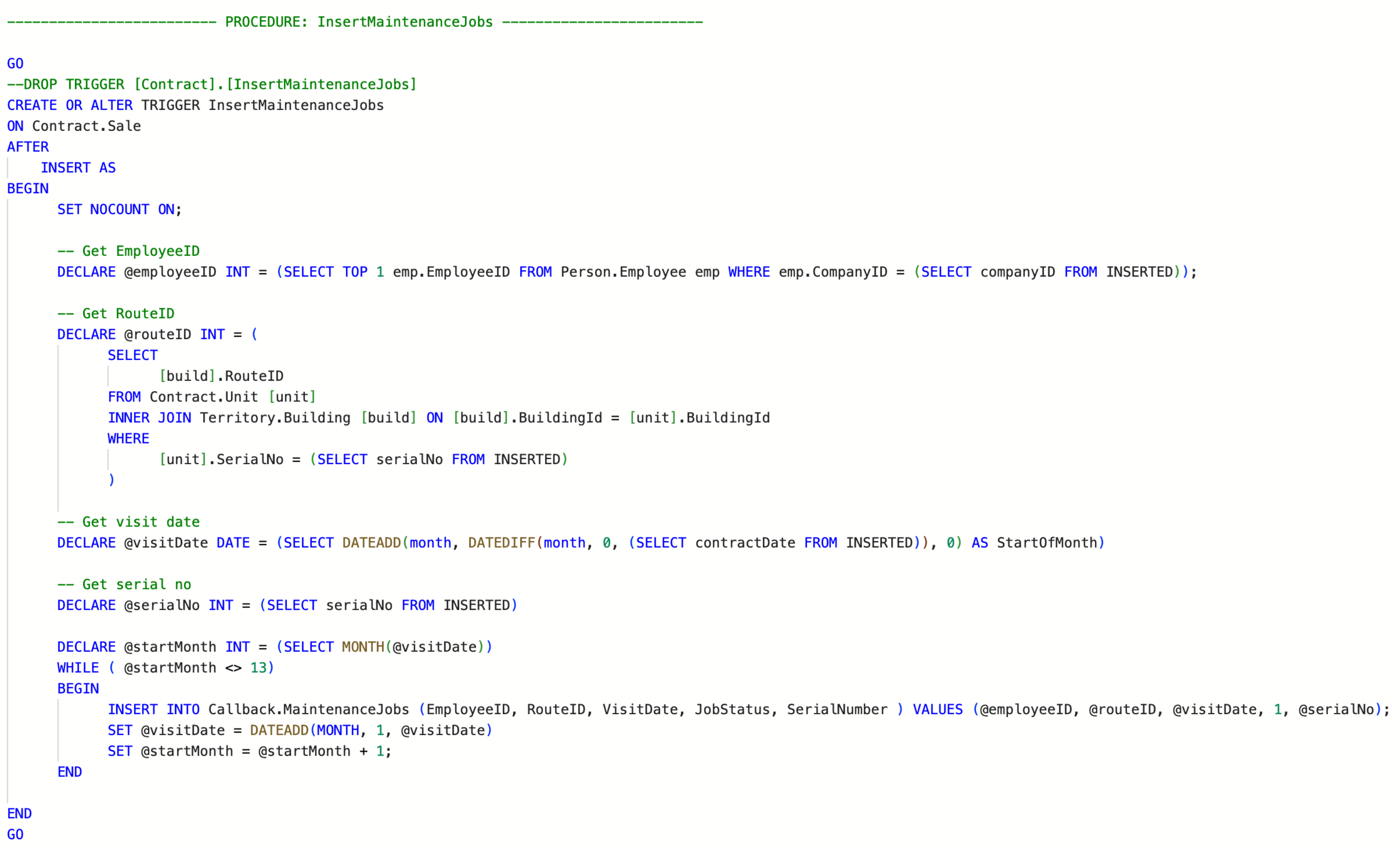
```
----- Encrypt Passsword -----

-- Create DMK
CREATE MASTER KEY
ENCRYPTION BY PASSWORD = 'Team10_P@sswOrd';

-- Create certificate to protect symmetric key
CREATE CERTIFICATE PasswordCertificate
WITH SUBJECT = 'Password Test Certificate',
EXPIRY_DATE = '2026-10-31';

-- DROP CERTIFICATE PasswordCertificate;

-- Create symmetric key to encrypt data
CREATE SYMMETRIC KEY PasswordSymmetricKey
WITH ALGORITHM = AES_128
ENCRYPTION BY CERTIFICATE PasswordCertificate;

-- DROP SYMMETRIC KEY PasswordSymmetricKey;

-- Open symmetric key
OPEN SYMMETRIC KEY PasswordSymmetricKey
DECRYPTION BY CERTIFICATE PasswordCertificate;
```

TABLE LEVEL CHECK CONSTRAINTS

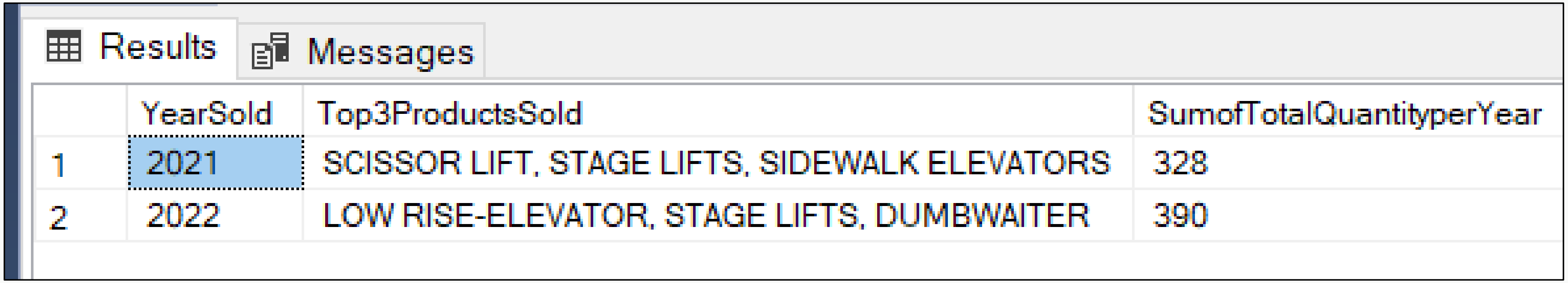


```
--------- Constraint ----------

GO
CREATE OR ALTER FUNCTION dbo.CheckRegion
(@RegionID INT, @OrganizationID INT)
RETURNS INT
AS
BEGIN
    RETURN (
        SELECT
            COUNT(*)
        FROM
            Client.Organization org
        INNER JOIN
            Territory.Region reg ON reg.CountryCode = org.OrganizationCountryCode
        WHERE
            org.OrganizationID = @OrganizationID
            AND reg.RegionID = @RegionID
    )
END
GO

-- SELECT * FROM Client.Organization;
-- SELECT * FROM Territory.Region

-- ALTER TABLE Client.Company DROP CONSTRAINT chk_CheckRegion
-- GO
ALTER TABLE Client.Company ADD CONSTRAINT chk_CheckRegion CHECK (dbo.CheckRegion(Company.RegionID,Company.OrganizationID) <> 0)
```

Trigger on Maintenance Jobs



```
------------------------- PROCEDURE: InsertMaintenanceJobs ------------------------

GO
--DROP TRIGGER [Contract].[InsertMaintenanceJobs]
CREATE OR ALTER TRIGGER InsertMaintenanceJobs
ON Contract.Sale
AFTER
    INSERT AS
BEGIN
      SET NOCOUNT ON;

      -- Get EmployeeID
      DECLARE @employeeID INT = (SELECT TOP 1 emp.EmployeeID FROM Person.Employee emp WHERE emp.CompanyID = (SELECT companyID FROM INSERTED));

      -- Get RouteID
      DECLARE @routeID INT = (
            SELECT
                  [build].RouteID
            FROM Contract.Unit [unit]
            INNER JOIN Territory.Building [build] ON [build].BuildingId = [unit].BuildingId
            WHERE
                  [unit].SerialNo = (SELECT serialNo FROM INSERTED)
            )

      -- Get visit date
      DECLARE @visitDate DATE = (SELECT DATEADD(month, DATEDIFF(month, 0, (SELECT contractDate FROM INSERTED)), 0) AS StartOfMonth)

      -- Get serial no
      DECLARE @serialNo INT = (SELECT serialNo FROM INSERTED)

      DECLARE @startMonth INT = (SELECT MONTH(@visitDate))
      WHILE ( @startMonth <> 13)
      BEGIN
            INSERT INTO Callback.MaintenanceJobs (EmployeeID, RouteID, VisitDate, JobStatus, SerialNumber ) VALUES (@employeeID, @routeID, @visitDate, 1, @serialNo);
            SET @visitDate = DATEADD(MONTH, 1, @visitDate)
            SET @startMonth = @startMonth + 1;
      END

END
GO
```

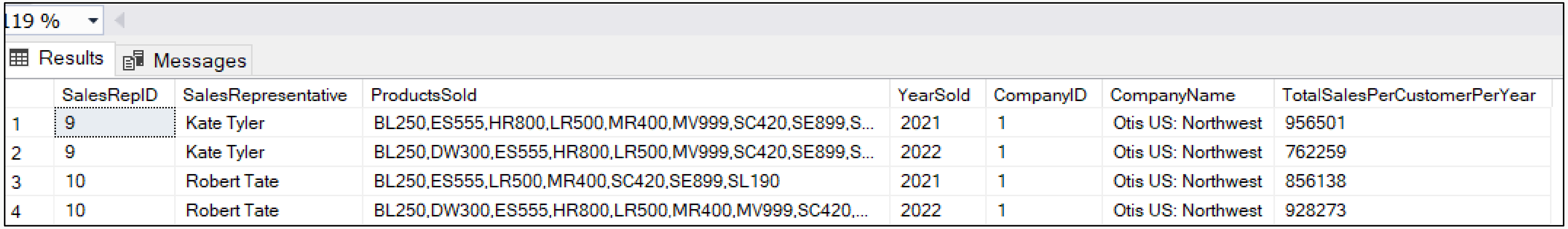


	YearSold	Top3ProductsSold	SumofTotalQuantityperYear
1	2021	SCISSOR LIFT, STAGE LIFTS, SIDEWALK ELEVATORS	328
2	2022	LOW RISE-ELEVATOR, STAGE LIFTS, DUMBWAITER	390

VIEW 1: TOP 3 PRODUCTS PER YEAR

VIEW 1: IMPLEMENTATION

```
GO
        CREATE VIEW Top3ProductsperYear as
        With SumProductsperyear as(SELECT Year(sale.ContractDate) as YearSold,SUM(Quantity) as TotalQuantitySoldperyear
        From Contract.Sale sale
        INNER JOIN Contract.Unit unit ON unit.SerialNo=sale.SerialNo
        INNER JOIN Product.Product product ON product.ProductID=unit.ProductID
        INNER JOIN Person.Person person ON person.PersonId=sale.SalesRepID
        INNER JOIN Product.ProductType producttype ON producttype.ProductTypeID=product.ProductTypeID
        GROUP BY Year(sale.ContractDate)
        ),
        ProductTypeSales as(SELECT Year(sale.ContractDate) as YearSold,product.ProductTypeID,producttype.Name,SUM(Quantity) as
        TotalQuantitySoldperyear
        From Contract.Sale sale
        INNER JOIN Contract.Unit unit ON unit.SerialNo=sale.SerialNo
        INNER JOIN Product.Product product ON product.ProductID=unit.ProductID
        INNER JOIN Person.Person person ON person.PersonId=sale.SalesRepID
        INNER JOIN Product.ProductType producttype ON producttype.ProductTypeID=product.ProductTypeID
        GROUP BY Year(sale.ContractDate),product.ProductTypeID,producttype.Name)


        SELECT temp.YearSold,STRING_AGG(temp.Name,', ') as Top3ProductsSold,sumperyear.TotalQuantitySoldperyear as SumofTotalQuantityperYear
        FROM(
        SELECT YearSold,Name,TotalQuantitySoldperyear,RANK() OVER (Partition By YearSold Order By TotalQuantitySoldperyear DESC) AS
        Top3ProductsperYear
        FROM ProductTypeSales)  as temp
        INNER JOIN SumProductsperyear as sumperyear ON sumperyear.YearSold = temp.YearSold
        WHERE Top3ProductsperYear<= 3
        GROUP BY temp.YearSold,sumperyear.TotalQuantitySoldperyear
        GO
```



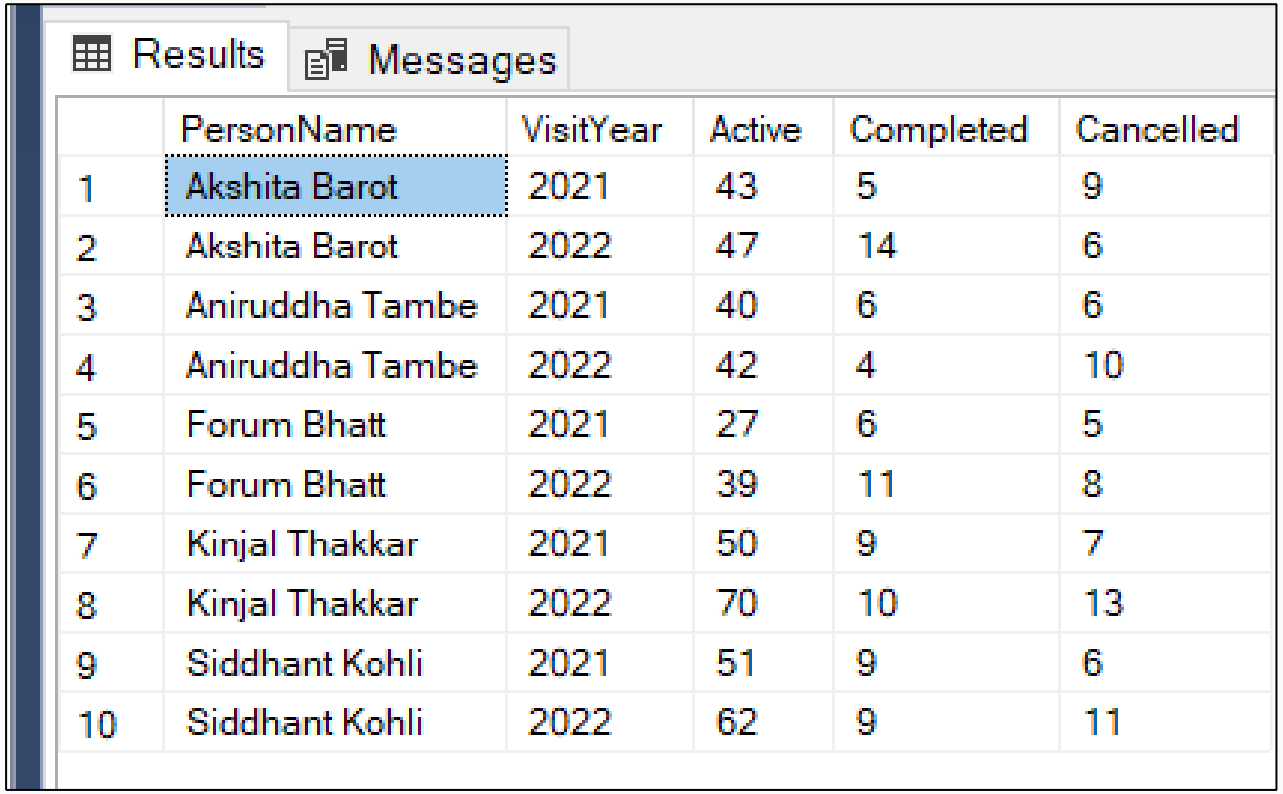
	SalesRepID	SalesRepresentative	ProductsSold	YearSold	CompanyID	CompanyName	TotalSalesPerCustomerPerYear
1	9	Kate Tyler	BL250,ES555,HR800,LR500,MR400,MV999,SC420,SE899,S...	2021	1	Otis US: Northwest	956501
2	9	Kate Tyler	BL250,DW300,ES555,HR800,LR500,MV999,SC420,SE899,S...	2022	1	Otis US: Northwest	762259
3	10	Robert Tate	BL250,ES555,LR500,MR400,SC420,SE899,SL190	2021	1	Otis US: Northwest	856138
4	10	Robert Tate	BL250,DW300,ES555,HR800,LR500,MR400,MV999,SC420,...	2022	1	Otis US: Northwest	928273

VIEW 2: YEARLY COMPANY SALES PER REPRESENTATIVE

VIEW 2: IMPLEMENTATION

```
GO

        CREATE VIEW YearlycompanysalesperRep as

        WITH temp as (SELECT sale.SalesRepID,person.FirstName+ ' '+person.LastName as
        SalesRepresentative,sale.CompanyID,company.CompanyName,product.ProductName,Year(sale.ContractDate
        ) as YearSold,SUM((100-sale.Discount)*sale.Quantity*product.FixedPrice)/100 as SalesPerProduct


        From Contract.Sale sale

        INNER JOIN Contract.Unit unit ON unit.SerialNo=sale.SerialNo

        INNER JOIN Product.Product product ON product.ProductID=unit.ProductID

        INNER JOIN Person.Person person ON person.PersonId=sale.SalesRepID

        INNER JOIN Client.Company company on company.CompanyID=sale.CompanyID

        GROUP BY
        sale.SalesRepID,Year(sale.ContractDate),product.ProductName,sale.CompanyID,person.FirstName,person.La
        stName,company.CompanyName)




        SELECT t1.SalesRepID,SalesRepresentative,STRING_AGG(ProductName,',') as
        ProductsSold,YearSold,CompanyID,CompanyName,CAST(SUM(SalesPerProduct)as int) as
        TotalSalesPerCustomerPerYear

        FROM temp t1

        GROUP BY SalesRepID,YearSold,CompanyID,SalesRepresentative,CompanyName

        GO
```

VIEW 3: MAINTENANCE VISITS PER YEAR

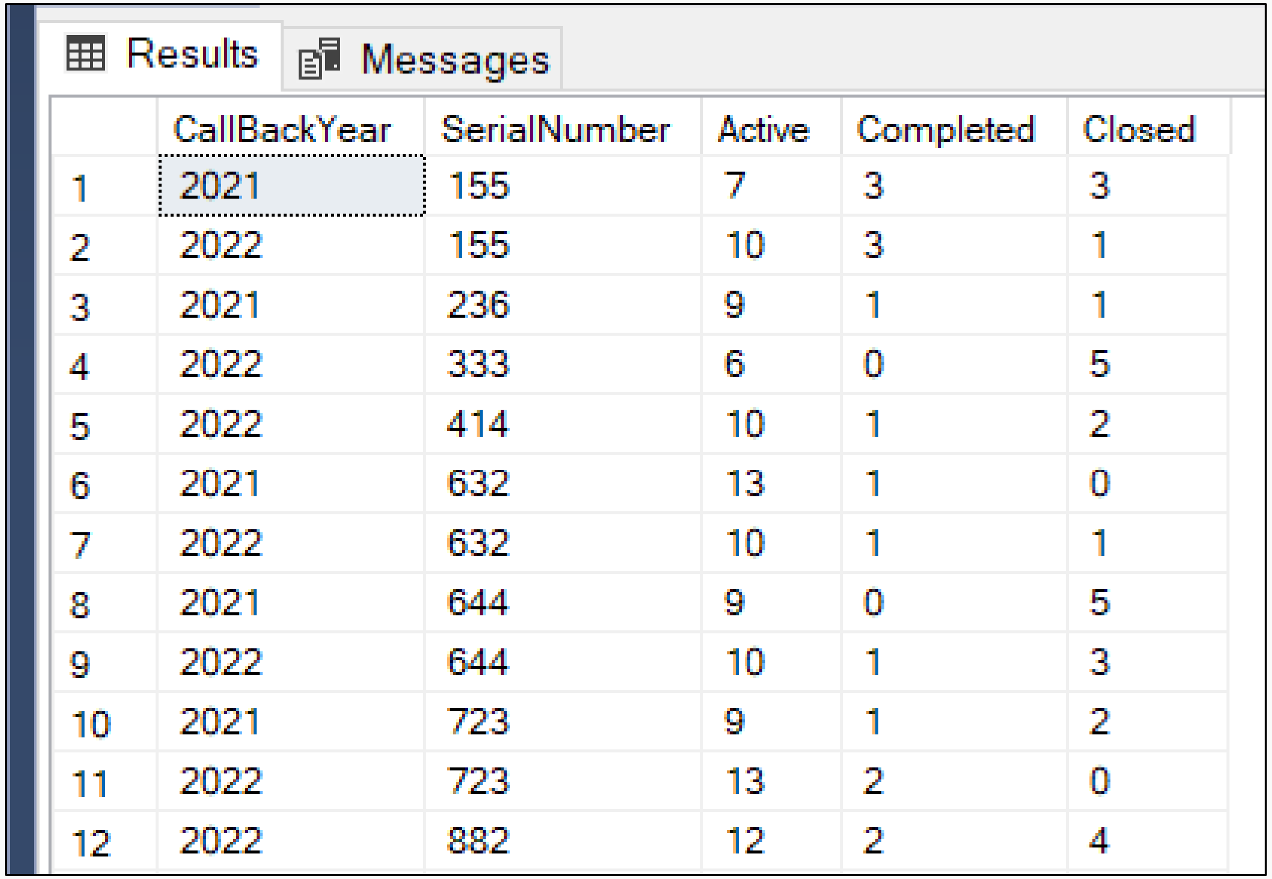


	PersonName	VisitYear	Active	Completed	Cancelled
1	Akshita Barot	2021	43	5	9
2	Akshita Barot	2022	47	14	6
3	Aniruddha Tambe	2021	40	6	6
4	Aniruddha Tambe	2022	42	4	10
5	Forum Bhatt	2021	27	6	5
6	Forum Bhatt	2022	39	11	8
7	Kinjal Thakkar	2021	50	9	7
8	Kinjal Thakkar	2022	70	10	13
9	Siddhant Kohli	2021	51	9	6
10	Siddhant Kohli	2022	62	9	11

VIEW3: IMPLEMENTATION

```
CREATE VIEW
MaintenanceVisitsperYear as

SELECT maintenance.EmployeeID,person.FirstName+' ' +
person.LastName AS 'PersonName',YEAR(maintenance.VisitDate)
VisitYear,callstatus.StatusType,Count(JobID) Visits

FROM Callback.MaintenanceJobs maintenance

INNER JOIN Callback.Status callstatus ON
callstatus.StatusID=maintenance.JobStatus

INNER JOIN Person.Employee emp ON
emp.EmployeeId=maintenance.EmployeeID

INNER JOIN Person.Person person ON
person.PersonId=emp.EmployeeId

GROUP BY
maintenance.EmployeeID,callstatus.StatusType,YEAR(maintenance.VisitDate),person.FirstName,person.LastName
```

VIEW 4: YEARLY CALLBACKS PER SERIAL

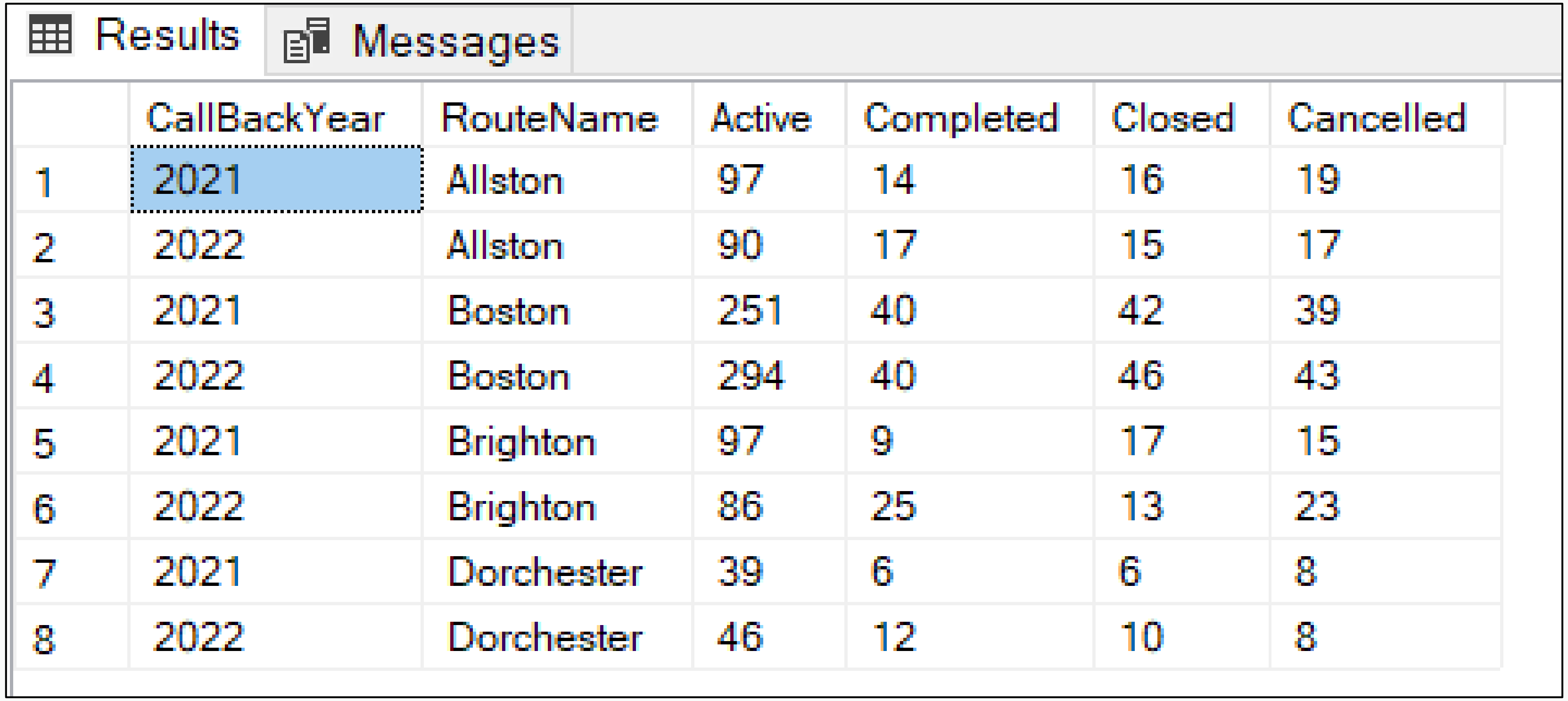


	CallBackYear	SerialNumber	Active	Completed	Closed
1	2021	155	7	3	3
2	2022	155	10	3	1
3	2021	236	9	1	1
4	2022	333	6	0	5
5	2022	414	10	1	2
6	2021	632	13	1	0
7	2022	632	10	1	1
8	2021	644	9	0	5
9	2022	644	10	1	3
10	2021	723	9	1	2
11	2022	723	13	2	0
12	2022	882	12	2	4

```
CREATE VIEW YearlyCallBackPerSerial10 as
SELECT * FROM (
SELECT Year(callback.CallbackDate)
CallBackYear,callback.SerialNumber,callback.CallbackID,callstatus.StatusType
FROM Callback.Callback callback
  INNER JOIN Callback.Status callstatus ON callstatus.StatusID=callback.StatusID
) src
PIVOT (
Count(CallbackID)
FOR src.StatusType IN (Active,Completed,Closed)
)piv
WHERE Active+Closed>=10
```

VIEW 4: IMPLEMENTATION

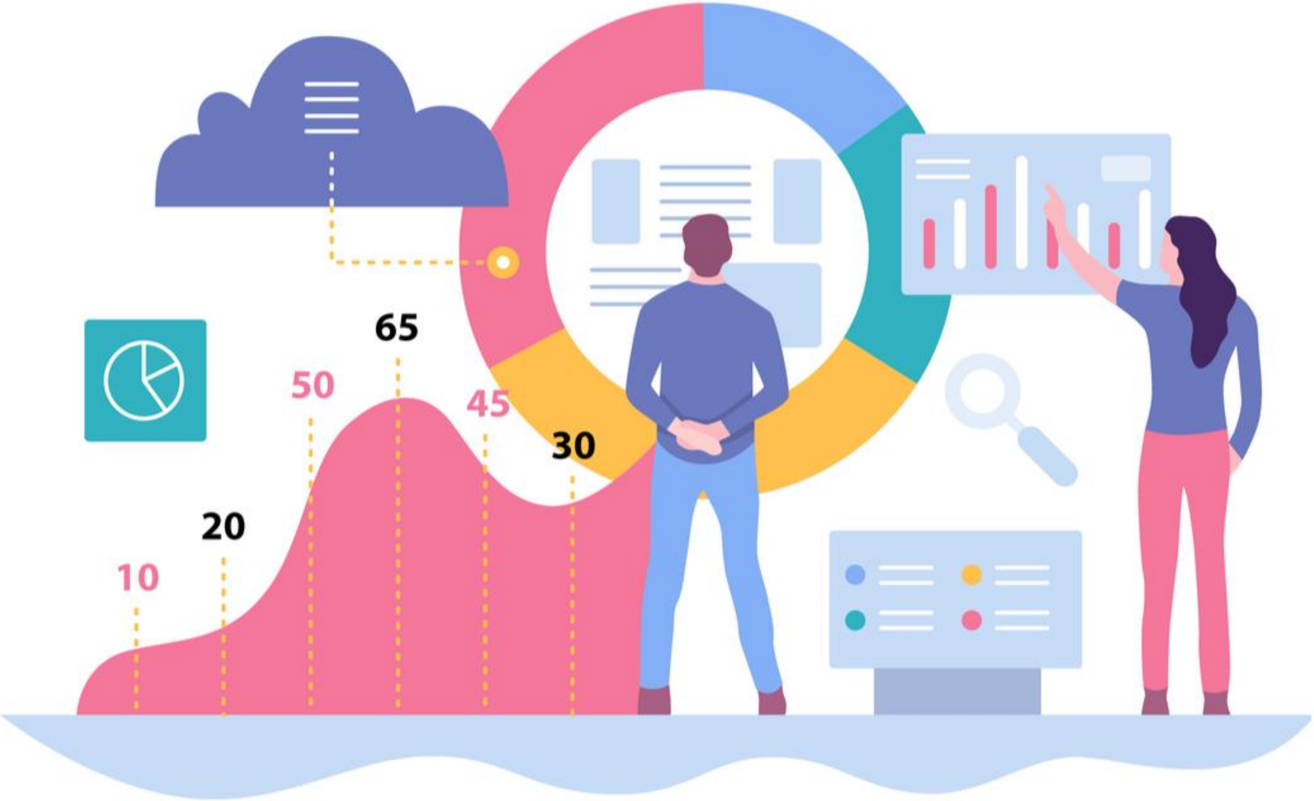
View 5: Yearly Callbacks per Route



	CallBackYear	RouteName	Active	Completed	Closed	Cancelled
1	2021	Allston	97	14	16	19
2	2022	Allston	90	17	15	17
3	2021	Boston	251	40	42	39
4	2022	Boston	294	40	46	43
5	2021	Brighton	97	9	17	15
6	2022	Brighton	86	25	13	23
7	2021	Dorchester	39	6	6	8
8	2022	Dorchester	46	12	10	8

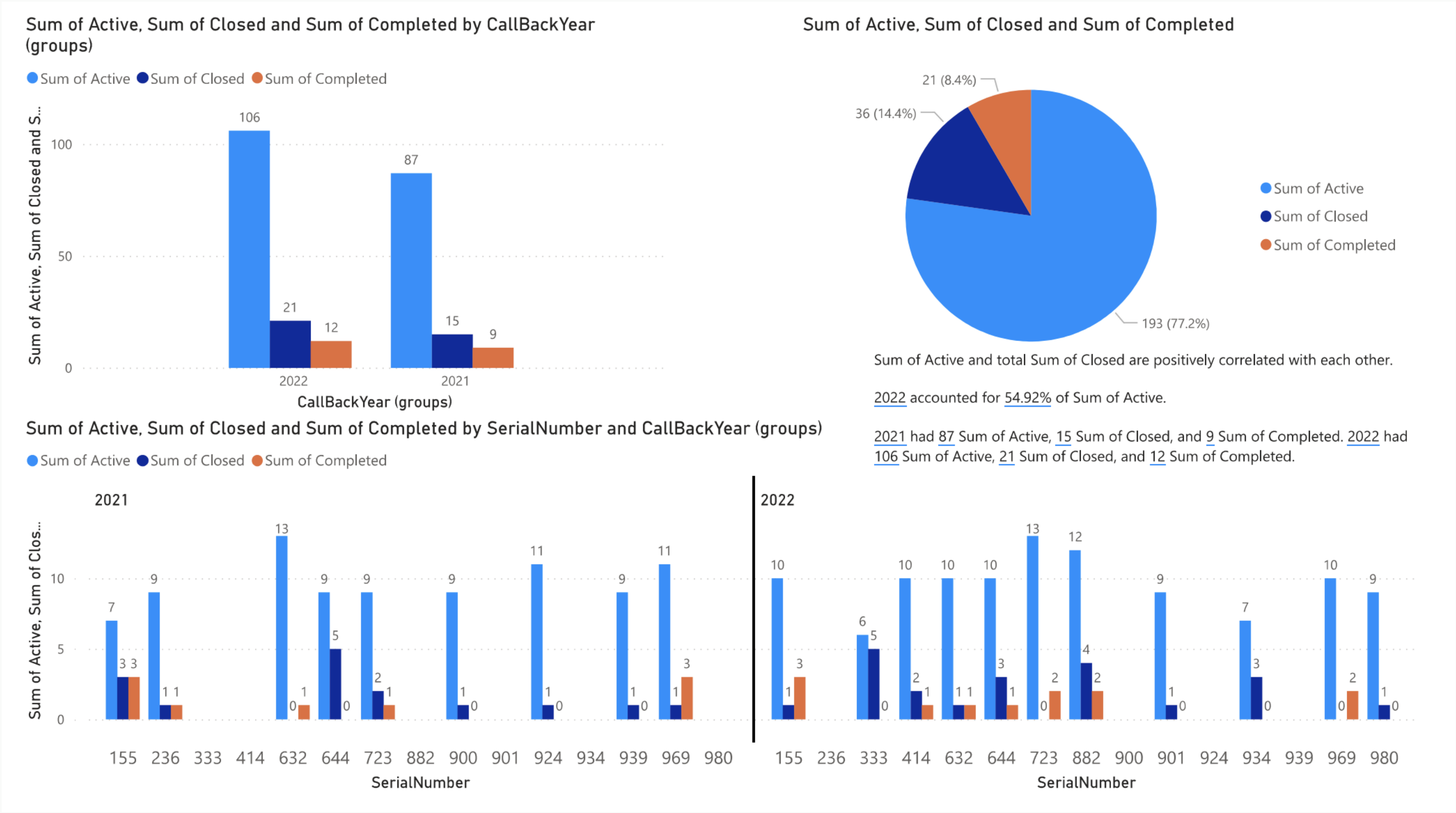
VIEW 5: IMPLEMENTATION

```
CREATE VIEW YearlyCallbacksPerRoute AS
SELECT *
FROM(
SELECT Year(callback.CallbackDate) CallBackYear,terroute.RouteName,callstatus.StatusType,callback.CallbackID
FROM Callback.Callback callback
INNER JOIN Callback.Status callstatus ON callstatus.StatusID=callback.StatusID
INNER JOIN Territory.Route terroute ON terroute.RouteID=callback.RouteID
)src
PIVOT(
COUNT(CallbackID)
FOR src.StatusType IN (Active,Completed,Closed,Cancelled)
)piv
```



REPORTS AND VISUALIZATION

Callbacks per Serial



Sum of Active, Sum of Closed and Sum of Completed by CallBackYear (groups)

● Sum of Active ● Sum of Closed ● Sum of Completed
Sum of Active, Sum of Closed and S...
100
50
0
106
21
12
87
15
9
2022
2021
CallBackYear (groups)

Sum of Active, Sum of Closed and Sum of Completed

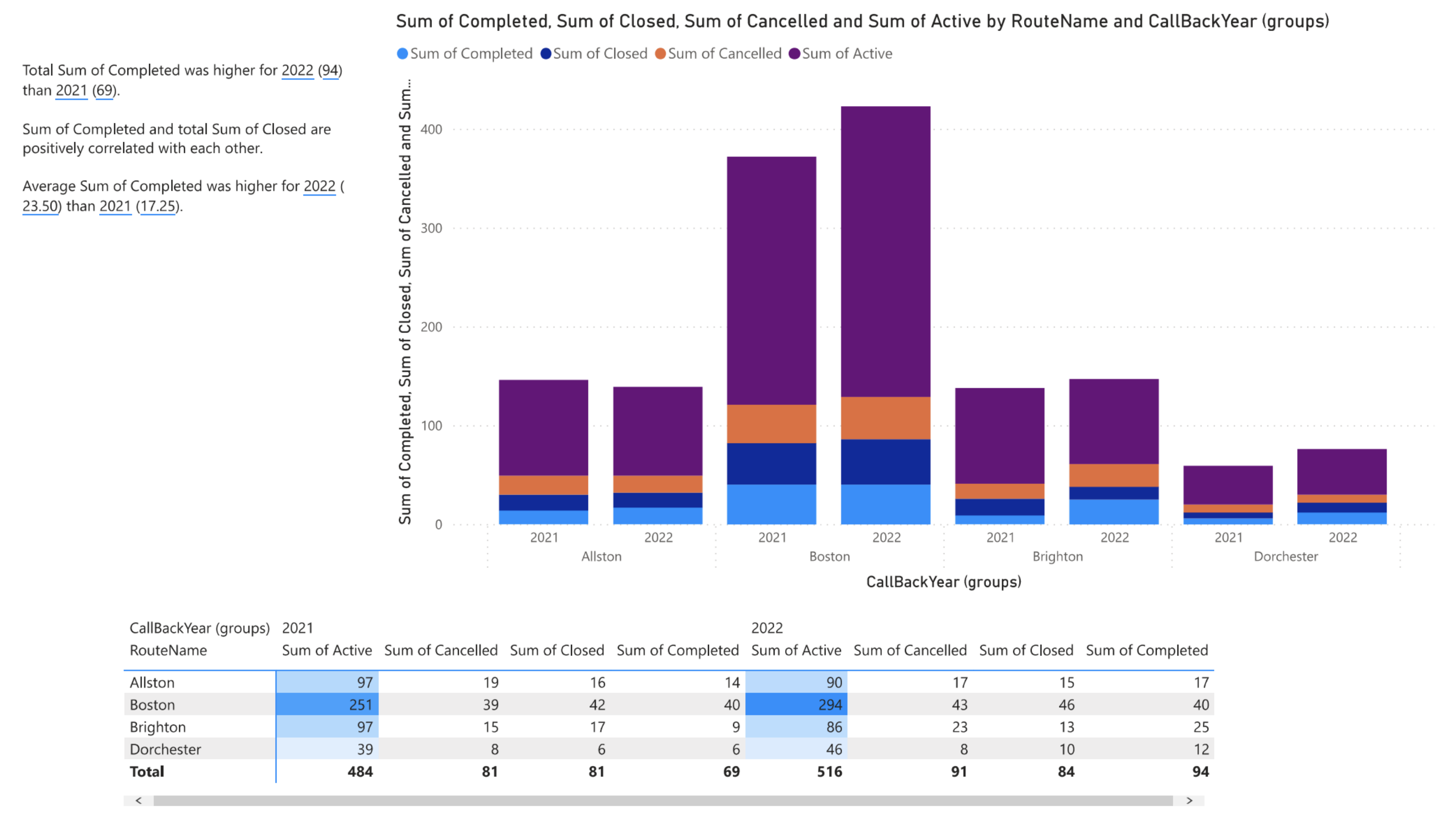
Sum of Active and total Sum of Closed are positively correlated with each other.

2022 accounted for 54.92% of Sum of Active.

2021 had 87 Sum of Active, 15 Sum of Closed, and 9 Sum of Completed. 2022 had 106 Sum of Active, 21 Sum of Closed, and 12 Sum of Completed.

Sum of Active, Sum of Closed and Sum of Completed by SerialNumber and CallBackYear (groups)

Yearly Callbacks per Route



Total Sum of Completed was higher for 2022 (94) than 2021 (69).

Sum of Completed and total Sum of Closed are positively correlated with each other.

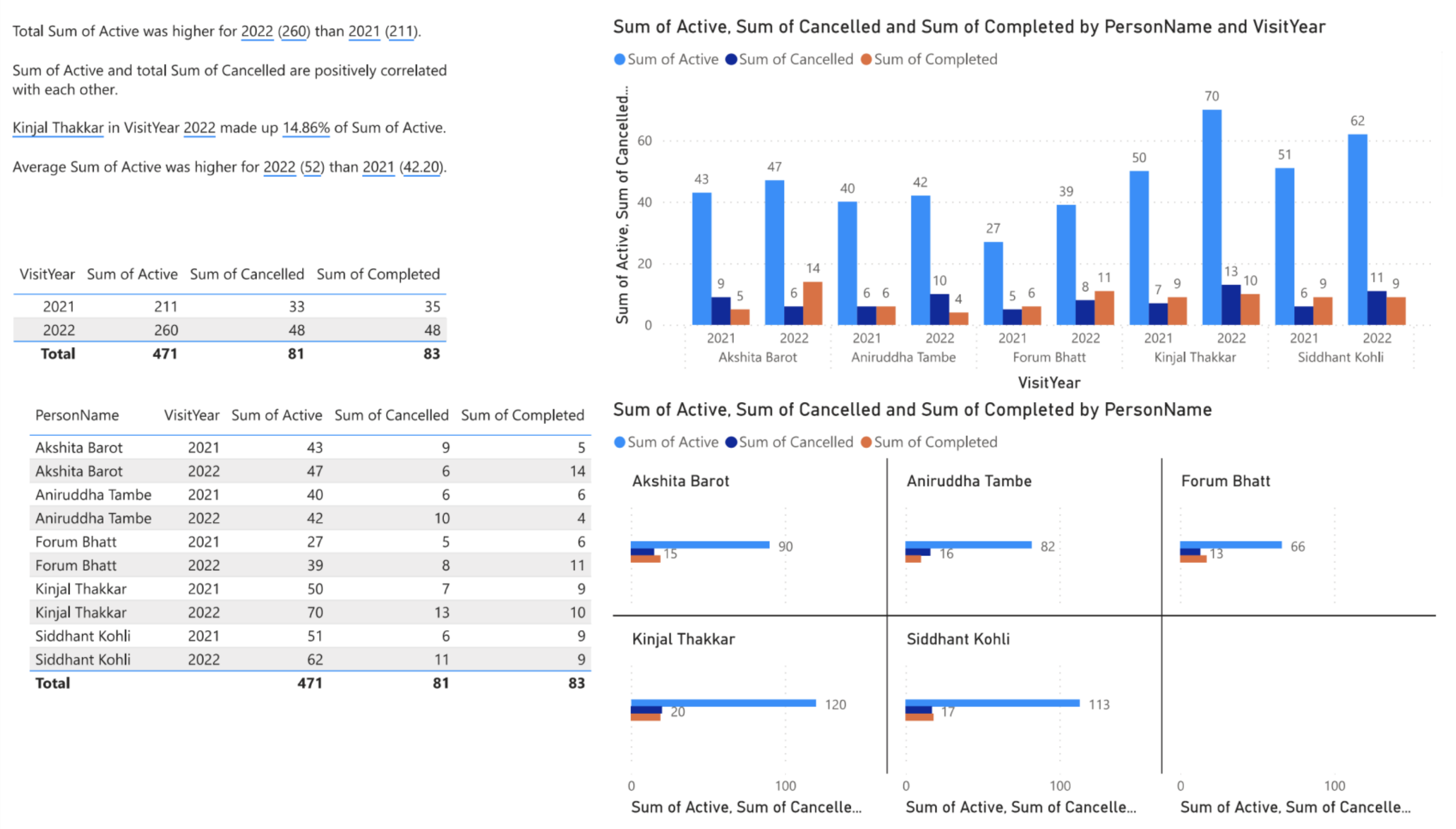
Average Sum of Completed was higher for 2022 (23.50) than 2021 (17.25).

Sum of Completed, Sum of Closed, Sum of Cancelled and Sum of Active by RouteName and CallBackYear (groups)

● Sum of Completed ● Sum of Closed ● Sum of Cancelled ● Sum of Active
Sum of Completed, Sum of Closed, Sum of Cancelled and Sum...
400
300
200
100
0
2021
2022
Allston
2021
2022
Boston
2021
2022
Brighton
2021
2022
Dorchester
CallBackYear (groups)

CallBackYear (groups)	2021				2022			
RouteName	Sum of Active	Sum of Cancelled	Sum of Closed	Sum of Completed	Sum of Active	Sum of Cancelled	Sum of Closed	Sum of Completed
Allston	97	19	16	14	90	17	15	17
Boston	251	39	42	40	294	43	46	40
Brighton	97	15	17	9	86	23	13	25
Dorchester	39	8	6	6	46	8	10	12
Total	**484**	**81**	**81**	**69**	**516**	**91**	**84**	**94**

Yearly Callbacks per Mechanic



Total Sum of Active was higher for 2022 (260) than 2021 (211).

Sum of Active and total Sum of Cancelled are positively correlated with each other.

Kinjal Thakkar in VisitYear 2022 made up 14.86% of Sum of Active.

Average Sum of Active was higher for 2022 (52) than 2021 (42.20).

VisitYear	Sum of Active	Sum of Cancelled	Sum of Completed
2021	211	33	35
2022	260	48	48
Total	**471**	**81**	**83**

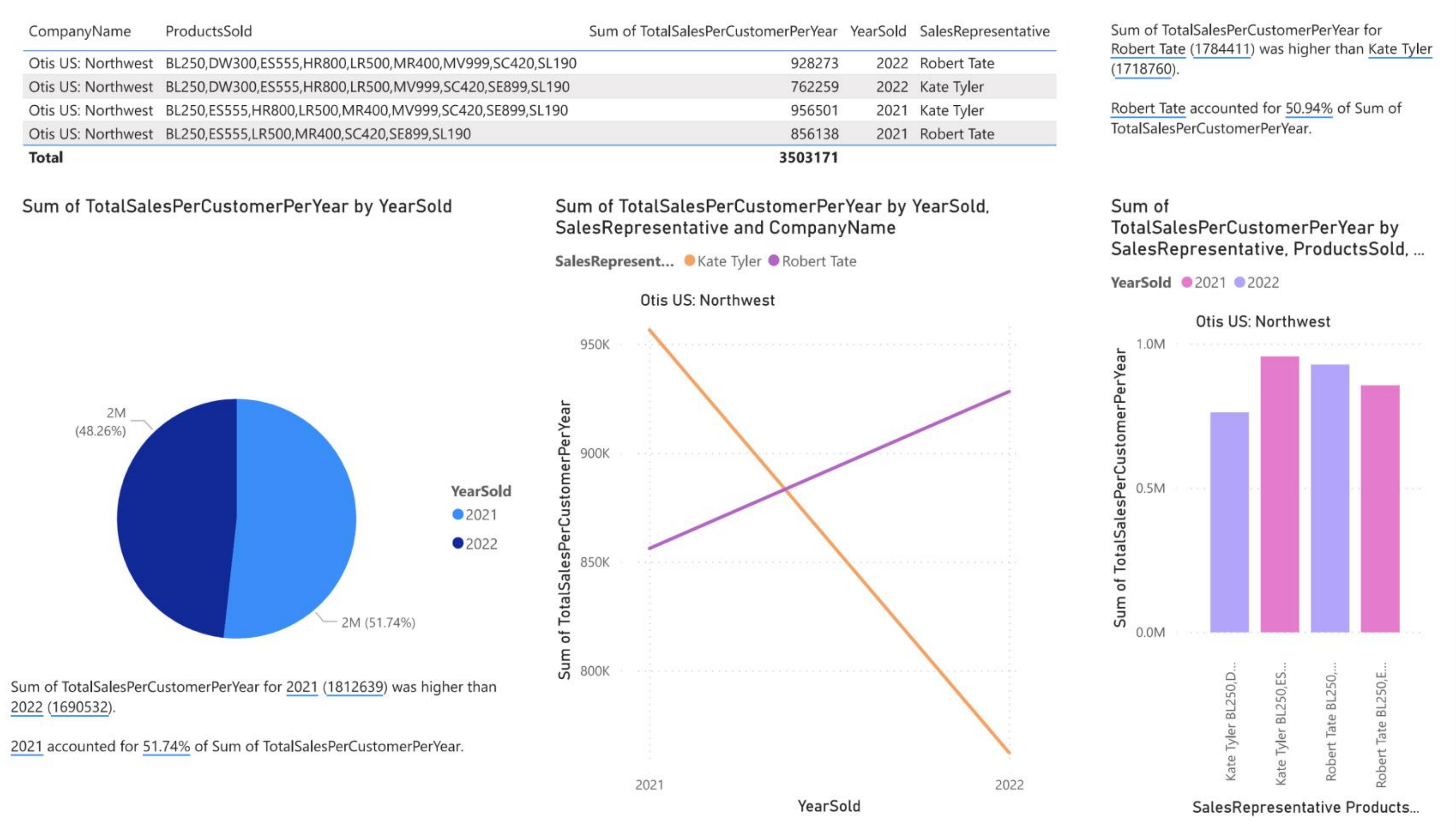
PersonName	VisitYear	Sum of Active	Sum of Cancelled	Sum of Completed
Akshita Barot	2021	43	9	5
Akshita Barot	2022	47	6	14
Aniruddha Tambe	2021	40	6	6
Aniruddha Tambe	2022	42	10	4
Forum Bhatt	2021	27	5	6
Forum Bhatt	2022	39	8	11
Kinjal Thakkar	2021	50	7	9
Kinjal Thakkar	2022	70	13	10
Siddhant Kohli	2021	51	6	9
Siddhant Kohli	2022	62	11	9
Total		**471**	**81**	**83**

Sum of Active, Sum of Cancelled and Sum of Completed by PersonName and VisitYear

Sum of Active, Sum of Cancelled and Sum of Completed by PersonName

● Sum of Active ● Sum of Cancelled ● Sum of Completed
Akshita Barot
90
15
Aniruddha Tambe
82
16
Forum Bhatt
66
13
Kinjal Thakkar
120
20
Siddhant Kohli
113
17
0
100
Sum of Active, Sum of Cancelle...
0
100
Sum of Active, Sum of Cancelle...
0
100
Sum of Active, Sum of Cancelle...

Yearly Sales Pipeline



CompanyName	ProductsSold	Sum of TotalSalesPerCustomerPerYear	YearSold	SalesRepresentative
Otis US: Northwest	BL250,DW300,ES555,HR800,LR500,MR400,MV999,SC420,SL190	928273	2022	Robert Tate
Otis US: Northwest	BL250,DW300,ES555,HR800,LR500,MV999,SC420,SE899,SL190	762259	2022	Kate Tyler
Otis US: Northwest	BL250,ES555,HR800,LR500,MR400,MV999,SC420,SE899,SL190	956501	2021	Kate Tyler
Otis US: Northwest	BL250,ES555,LR500,MR400,SC420,SE899,SL190	856138	2021	Robert Tate
Total		**3503171**		

Sum of TotalSalesPerCustomerPerYear for Robert Tate (1784411) was higher than Kate Tyler (1718760).

Robert Tate accounted for 50.94% of Sum of TotalSalesPerCustomerPerYear.

Sum of TotalSalesPerCustomerPerYear by YearSold

Sum of TotalSalesPerCustomerPerYear for 2021 (1812639) was higher than 2022 (1690532).

2021 accounted for 51.74% of Sum of TotalSalesPerCustomerPerYear.

Sum of TotalSalesPerCustomerPerYear by YearSold, SalesRepresentative and CompanyName

Sum of TotalSalesPerCustomerPerYear by SalesRepresentative, ProductsSold, ...

CONCLUSION

ANY QUESTIONS?