Procurement Management System

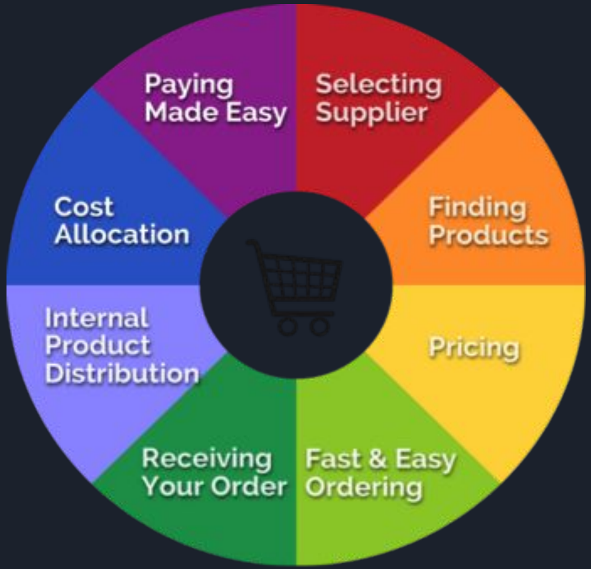
Group Number : 13

Ankur Chauhan
Aparna Shelar
Roshni Dhami
Rahul Bhiwande
Sudipta Saha
Urvi Aryamane

What is Procurement?

- Procurement is process acquiring goods by selecting vendors, establishing payment terms, strategic vetting and negotiation of contracts.

- The purpose of this database is to maintain the data used to support the procurement process of an organization by establishing and maintaining strategic thinking in all procurement efforts.



Business Problem Addressed

- Preventing unauthorized purchase
- Choosing efficient supplier based on rating/available discounts
- Analyzing the expense and budget of each department
- Create strategies to control spending of each department
- Using catalogue for well organised procurement process
- Storing purchase order and invoicing details for easy auditing
- Preventing wasteful purchasing by checking inventory

Schemas and Entities

- Organization :

    Employee, General Staff, Admin, Manager, Inventory, Department

- Product:

    Products, Category, ProductCategory, Catalogue

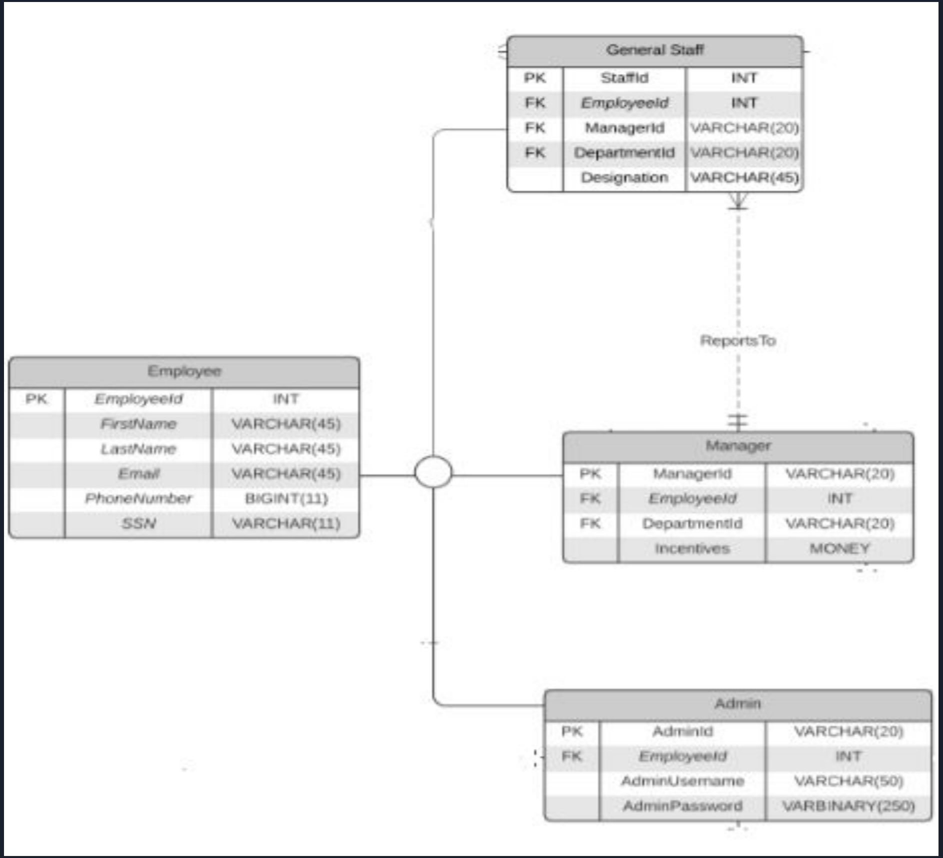
- Procurement:

    Ticket, Requisition, Order, Invoice

- Supplier:

    Supplier, Contract, ProductSupplier

Generalization and Specialization



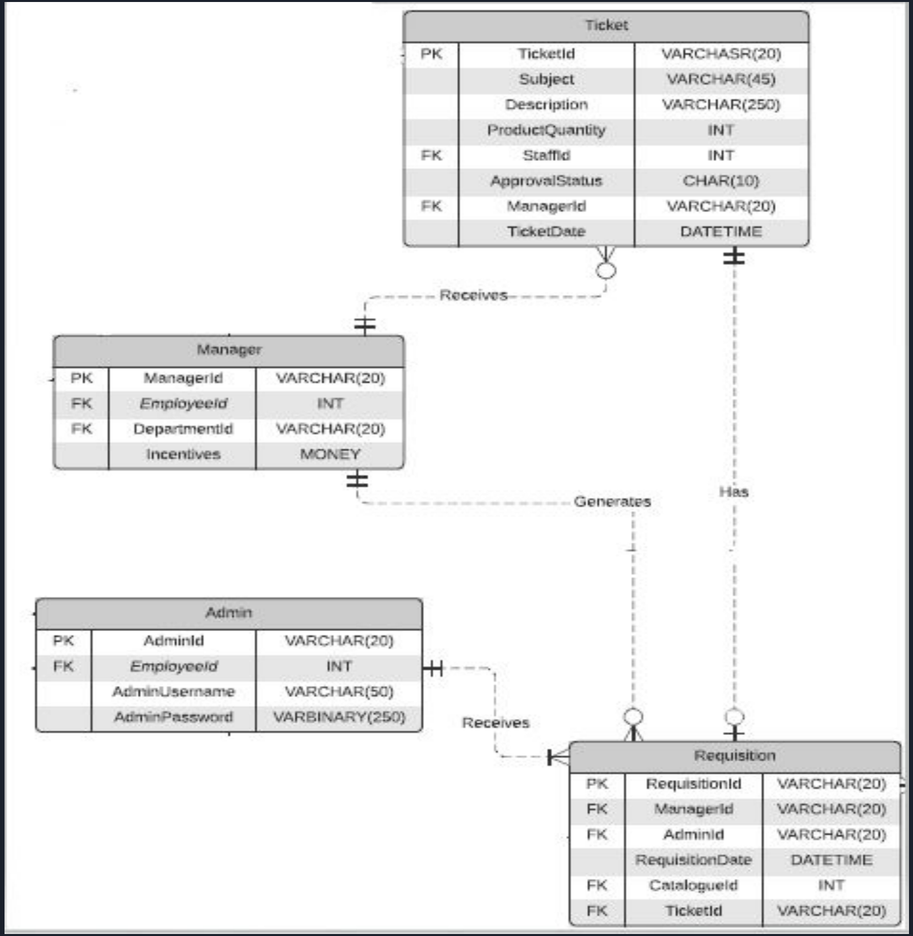
General Staff		
PK	StaffId	INT
FK	*EmployeeId*	INT
FK	ManagerId	VARCHAR(20)
FK	DepartmentId	VARCHAR(20)
	Designation	VARCHAR(45)

Employee		
PK	*EmployeeId*	INT
	FirstName	VARCHAR(45)
	LastName	VARCHAR(45)
	Email	VARCHAR(45)
	PhoneNumber	BIGINT(11)
	SSN	VARCHAR(11)

Manager		
PK	ManagerId	VARCHAR(20)
FK	*EmployeeId*	INT
FK	DepartmentId	VARCHAR(20)
	Incentives	MONEY

Admin		
PK	AdminId	VARCHAR(20)
FK	*EmployeeId*	INT
	AdminUsername	VARCHAR(50)
	AdminPassword	VARBINARY(250)

Approval Workflow



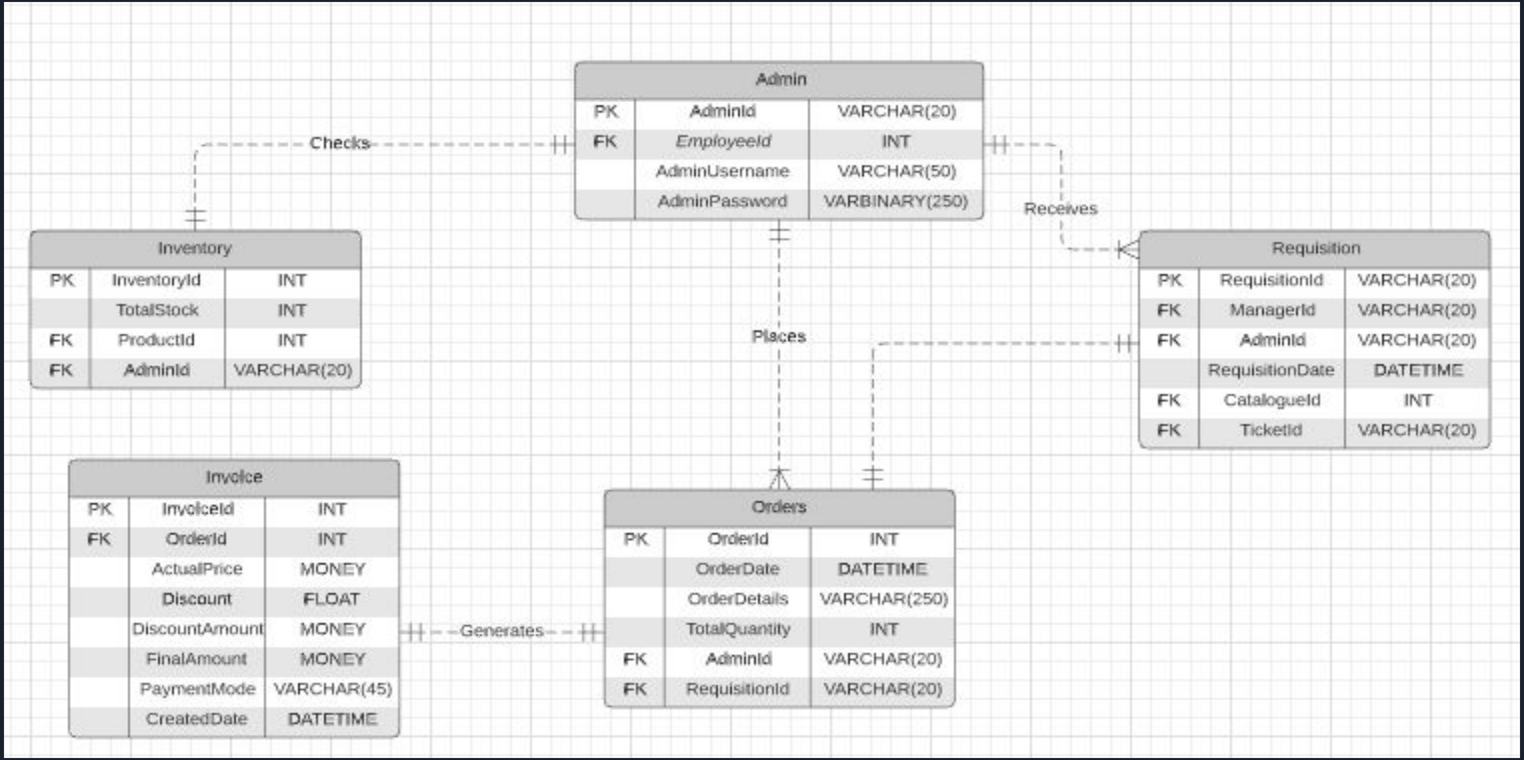
Ticket		
PK	TicketId	VARCHASR(20)
	Subject	VARCHAR(45)
	Description	VARCHAR(250)
	ProductQuantity	INT
FK	StaffId	INT
	ApprovalStatus	CHAR(10)
FK	ManagerId	VARCHAR(20)
	TicketDate	DATETIME

Manager		
PK	ManagerId	VARCHAR(20)
FK	*EmployeeId*	INT
FK	DepartmentId	VARCHAR(20)
	Incentives	MONEY

Admin		
PK	AdminId	VARCHAR(20)
FK	*EmployeeId*	INT
	AdminUsername	VARCHAR(50)
	AdminPassword	VARBINARY(250)

Requisition		
PK	RequisitionId	VARCHAR(20)
FK	ManagerId	VARCHAR(20)
FK	AdminId	VARCHAR(20)
	RequisitionDate	DATETIME
FK	CatalogueId	INT
FK	TicketId	VARCHAR(20)

Requisition Flow



Admin

PK	AdminId	VARCHAR(20)
FK	*EmployeeId*	INT
	AdminUsername	VARCHAR(50)
	AdminPassword	VARBINARY(250)

Inventory

PK	InventoryId	INT
	TotalStock	INT
FK	ProductId	INT
FK	AdminId	VARCHAR(20)

Requisition

PK	RequisitionId	VARCHAR(20)
FK	ManagerId	VARCHAR(20)
FK	AdminId	VARCHAR(20)
	RequisitionDate	DATETIME
FK	CatalogueId	INT
FK	TicketId	VARCHAR(20)

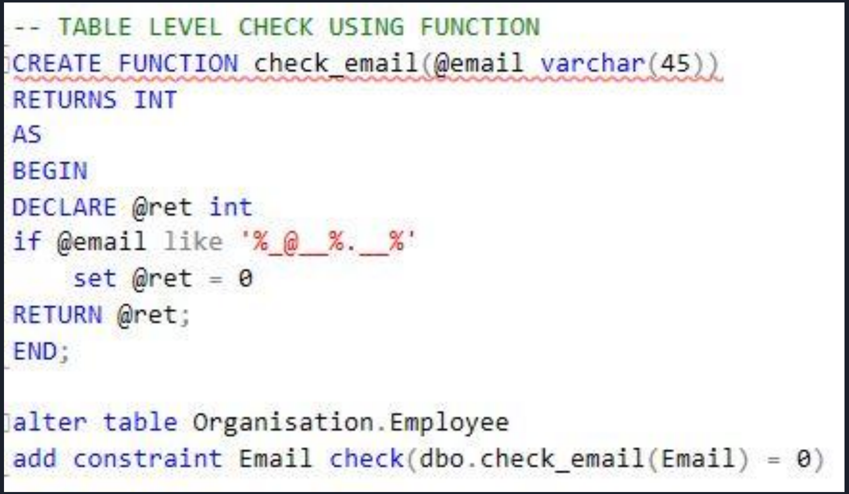
Invoice

PK	InvoiceId	INT
FK	OrderId	INT
	ActualPrice	MONEY
	Discount	FLOAT
	DiscountAmount	MONEY
	FinalAmount	MONEY
	PaymentMode	VARCHAR(45)
	CreatedDate	DATETIME

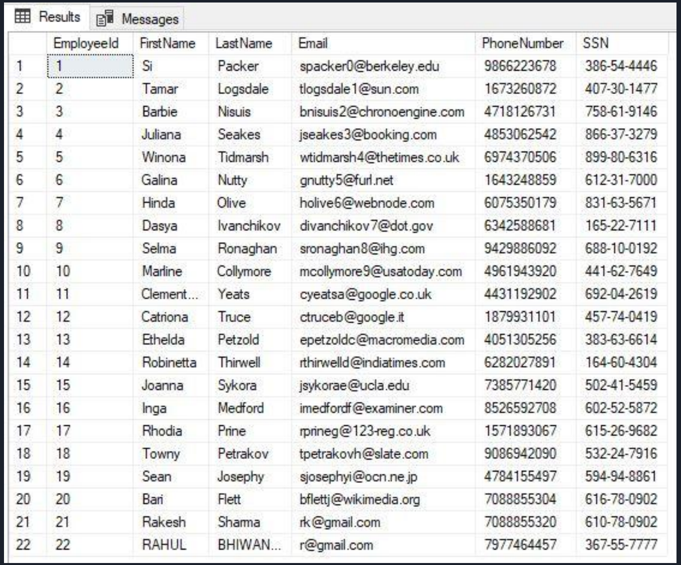
Orders

PK	OrderId	INT
	OrderDate	DATETIME
	OrderDetails	VARCHAR(250)
	TotalQuantity	INT
FK	AdminId	VARCHAR(20)
FK	RequisitionId	VARCHAR(20)

Table Level Check for Email Address

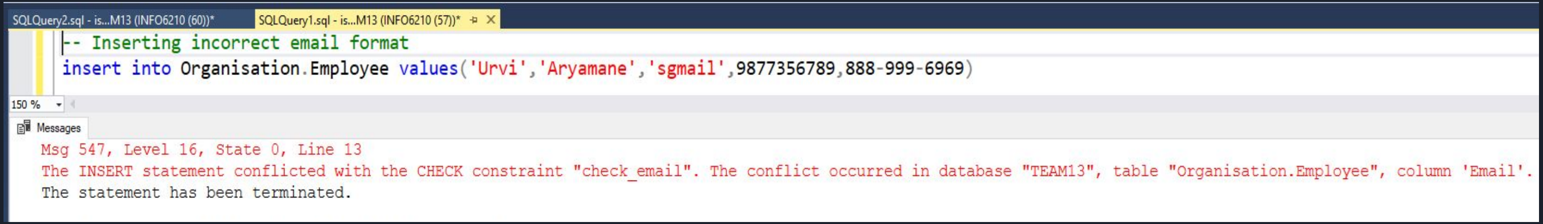


```sql
-- TABLE LEVEL CHECK USING FUNCTION
CREATE FUNCTION check_email(@email varchar(45))
RETURNS INT
AS
BEGIN
DECLARE @ret int
if @email like '%_@__%.__%'
    set @ret = 0
RETURN @ret;
END;

alter table Organisation.Employee
add constraint Email check(dbo.check_email(Email) = 0)
```



Results | Messages

	EmployeeId	FirstName	LastName	Email	PhoneNumber	SSN
1	1	Si	Packer	spacker0@berkeley.edu	9866223678	386-54-4446
2	2	Tamar	Logsdale	tlogsdale1@sun.com	1673260872	407-30-1477
3	3	Barbie	Nisuis	bnisuis2@chronoengine.com	4718126731	758-61-9146
4	4	Juliana	Seakes	jseakes3@booking.com	4853062542	866-37-3279
5	5	Winona	Tidmarsh	wtidmarsh4@thetimes.co.uk	6974370506	899-80-6316
6	6	Galina	Nutty	gnutty5@furl.net	1643248859	612-31-7000
7	7	Hinda	Olive	holive6@webnode.com	6075350179	831-63-5671
8	8	Dasya	Ivanchikov	divanchikov7@dot.gov	6342588681	165-22-7111
9	9	Selma	Ronaghan	sronaghan8@ihg.com	9429886092	688-10-0192
10	10	Marline	Collymore	mcollymore9@usatoday.com	4961943920	441-62-7649
11	11	Clement...	Yeats	cyeatsa@google.co.uk	4431192902	692-04-2619
12	12	Catriona	Truce	ctruceb@google.it	1879931101	457-74-0419
13	13	Ethelda	Petzold	epetzoldc@macromedia.com	4051305256	383-63-6614
14	14	Robinetta	Thirwell	rthirwelld@indiatimes.com	6282027891	164-60-4304
15	15	Joanna	Sykora	jsykorae@ucla.edu	7385771420	502-41-5459
16	16	Inga	Medford	imedfordf@examiner.com	8526592708	602-52-5872
17	17	Rhodia	Prine	rprineg@123-reg.co.uk	1571893067	615-26-9682
18	18	Towny	Petrakov	tpetrakovh@slate.com	9086942090	532-24-7916
19	19	Sean	Josephy	sjosephyi@ocn.ne.jp	4784155497	594-94-8861
20	20	Bari	Flett	bflettj@wikimedia.org	7088855304	616-78-0902
21	21	Rakesh	Sharma	rk@gmail.com	7088855320	610-78-0902
22	22	RAHUL	BHIWAN...	r@gmail.com	7977464457	367-55-7777



SQLQuery2.sql - is...M13 (INFO6210 (60))* | SQLQuery1.sql - is...M13 (INFO6210 (57))*

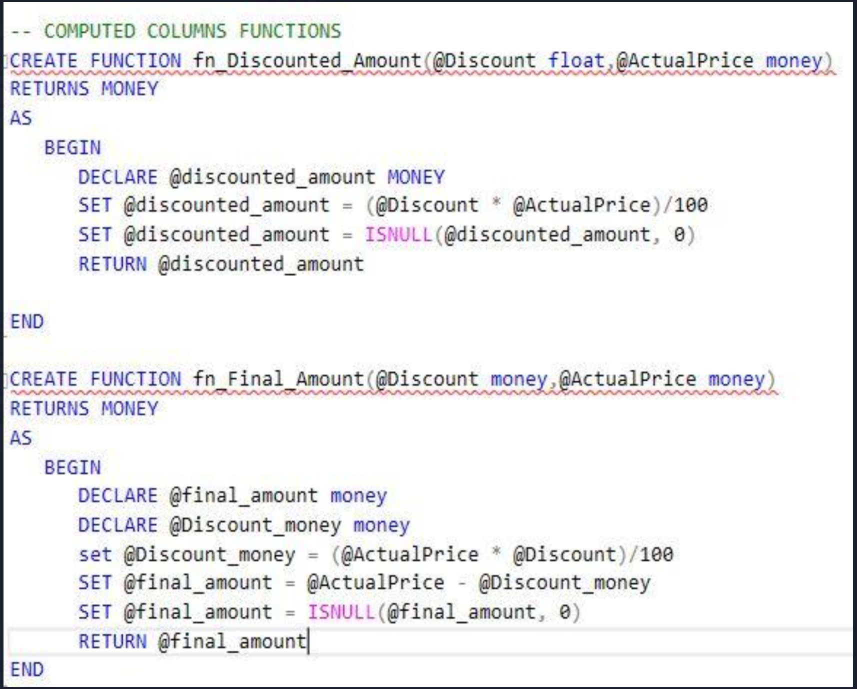
```sql
-- Inserting incorrect email format
insert into Organisation.Employee values('Urvi','Aryamane','sgmail',9877356789,888-999-6969)
```

150 %

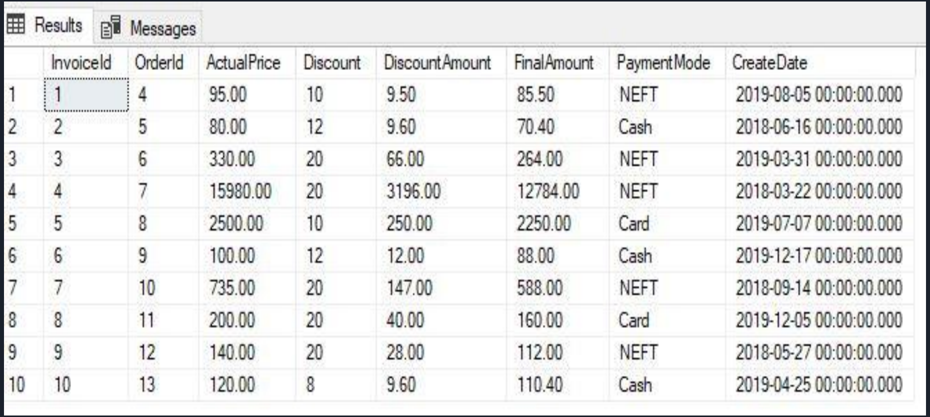
Messages

```
Msg 547, Level 16, State 0, Line 13
The INSERT statement conflicted with the CHECK constraint "check_email". The conflict occurred in database "TEAM13", table "Organisation.Employee", column 'Email'.
The statement has been terminated.
```

Computed Column for Discounted Amount and Final Amount in Invoice Entity



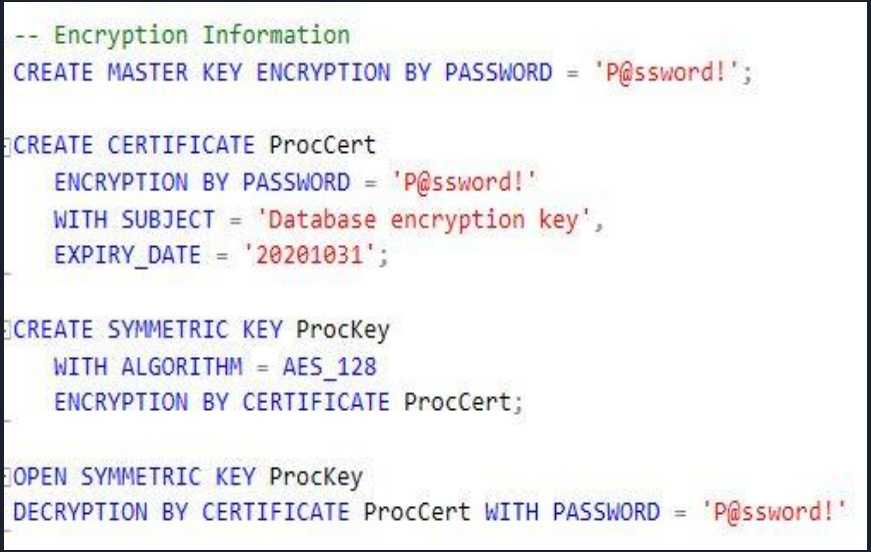
```
-- COMPUTED COLUMNS FUNCTIONS
CREATE FUNCTION fn_Discounted_Amount(@Discount float,@ActualPrice money)
RETURNS MONEY
AS
   BEGIN
      DECLARE @discounted_amount MONEY
      SET @discounted_amount = (@Discount * @ActualPrice)/100
      SET @discounted_amount = ISNULL(@discounted_amount, 0)
      RETURN @discounted_amount

END

CREATE FUNCTION fn_Final_Amount(@Discount money,@ActualPrice money)
RETURNS MONEY
AS
   BEGIN
      DECLARE @final_amount money
      DECLARE @Discount_money money
      set @Discount_money = (@ActualPrice * @Discount)/100
      SET @final_amount = @ActualPrice - @Discount_money
      SET @final_amount = ISNULL(@final_amount, 0)
      RETURN @final_amount
END
```



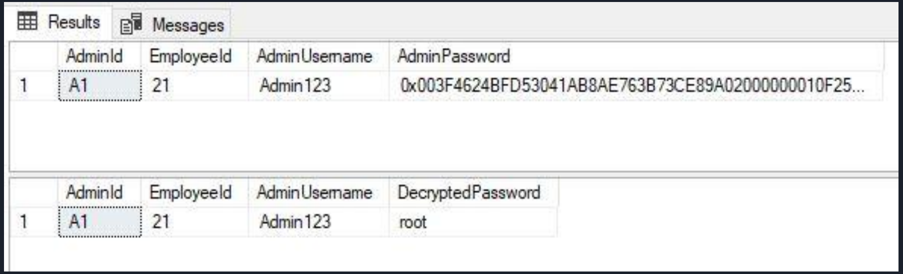
Results | Messages

	InvoiceId	OrderId	ActualPrice	Discount	DiscountAmount	FinalAmount	PaymentMode	CreateDate
1	1	4	95.00	10	9.50	85.50	NEFT	2019-08-05 00:00:00.000
2	2	5	80.00	12	9.60	70.40	Cash	2018-06-16 00:00:00.000
3	3	6	330.00	20	66.00	264.00	NEFT	2019-03-31 00:00:00.000
4	4	7	15980.00	20	3196.00	12784.00	NEFT	2018-03-22 00:00:00.000
5	5	8	2500.00	10	250.00	2250.00	Card	2019-07-07 00:00:00.000
6	6	9	100.00	12	12.00	88.00	Cash	2019-12-17 00:00:00.000
7	7	10	735.00	20	147.00	588.00	NEFT	2018-09-14 00:00:00.000
8	8	11	200.00	20	40.00	160.00	Card	2019-12-05 00:00:00.000
9	9	12	140.00	20	28.00	112.00	NEFT	2018-05-27 00:00:00.000
10	10	13	120.00	8	9.60	110.40	Cash	2019-04-25 00:00:00.000

Encryption on password



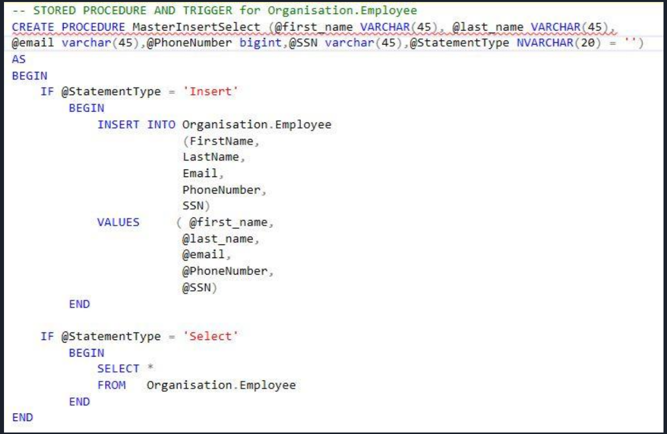
```
-- Encryption Information
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'P@ssword!';

CREATE CERTIFICATE ProcCert
   ENCRYPTION BY PASSWORD = 'P@ssword!'
   WITH SUBJECT = 'Database encryption key',
   EXPIRY_DATE = '20201031';

CREATE SYMMETRIC KEY ProcKey
   WITH ALGORITHM = AES_128
   ENCRYPTION BY CERTIFICATE ProcCert;

OPEN SYMMETRIC KEY ProcKey
DECRYPTION BY CERTIFICATE ProcCert WITH PASSWORD = 'P@ssword!'
```



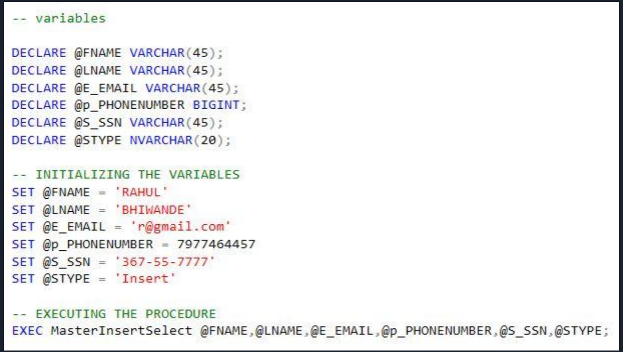
	AdminId	EmployeeId	AdminUsername	AdminPassword
1	A1	21	Admin123	0x003F4624BFD53041AB8AE763B73CE89A0200000010F25...

	AdminId	EmployeeId	AdminUsername	DecryptedPassword
1	A1	21	Admin123	root

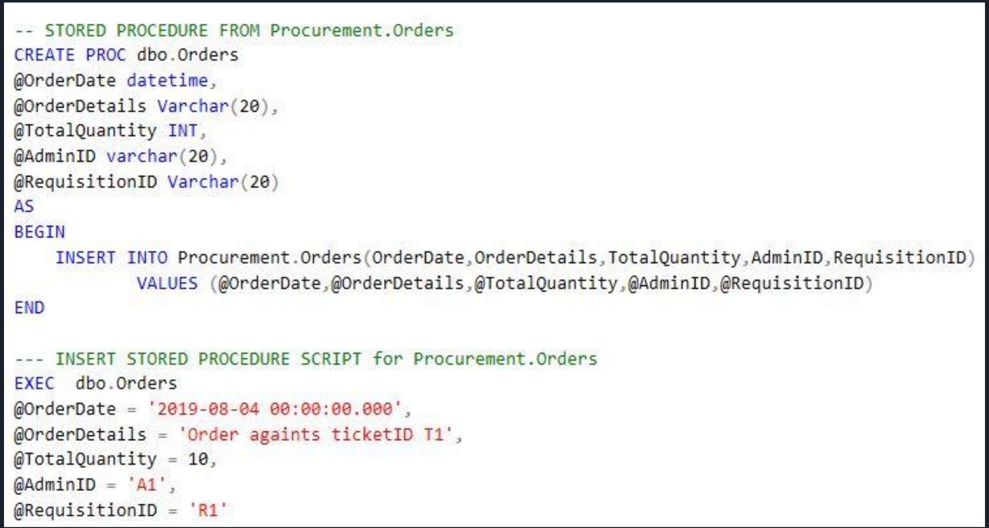
Stored Procedure for Employee Entity and Orders Entity



```sql
-- STORED PROCEDURE AND TRIGGER for Organisation.Employee
CREATE PROCEDURE MasterInsertSelect (@first_name VARCHAR(45), @last_name VARCHAR(45),
@email varchar(45),@PhoneNumber bigint,@SSN varchar(45),@StatementType NVARCHAR(20) = '')
AS
BEGIN
    IF @StatementType = 'Insert'
        BEGIN
            INSERT INTO Organisation.Employee
                        (FirstName,
                        LastName,
                        Email,
                        PhoneNumber,
                        SSN)
            VALUES     ( @first_name,
                        @last_name,
                        @email,
                        @PhoneNumber,
                        @SSN)
        END

    IF @StatementType = 'Select'
        BEGIN
            SELECT *
            FROM   Organisation.Employee
        END
END
```

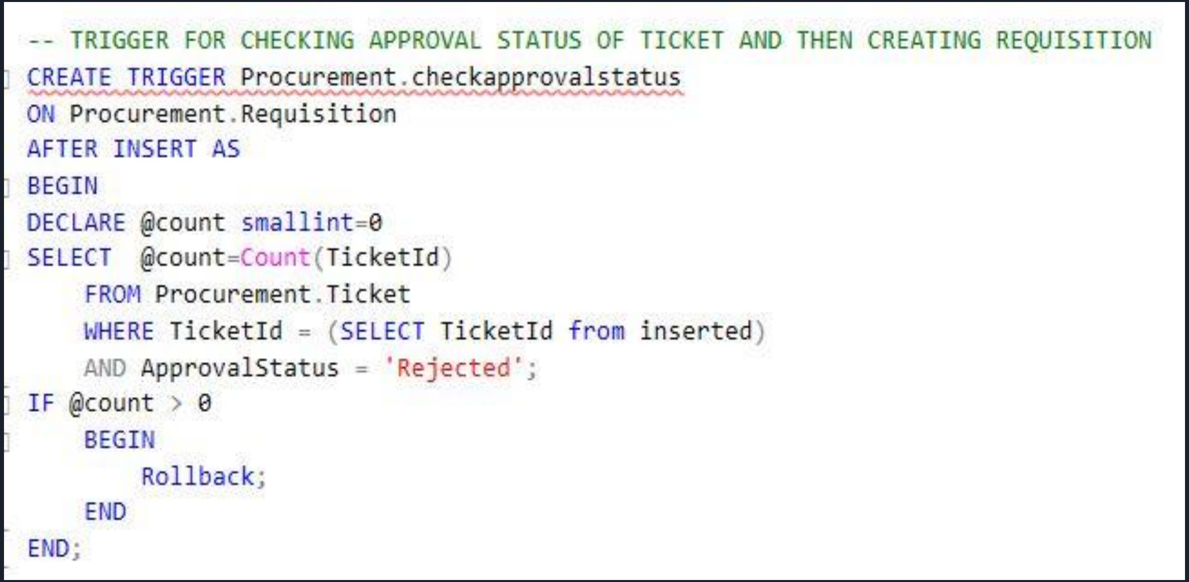


```sql
-- variables

DECLARE @FNAME VARCHAR(45);
DECLARE @LNAME VARCHAR(45);
DECLARE @E_EMAIL VARCHAR(45);
DECLARE @p_PHONENUMBER BIGINT;
DECLARE @S_SSN VARCHAR(45);
DECLARE @STYPE NVARCHAR(20);

-- INITIALIZING THE VARIABLES
SET @FNAME = 'RAHUL'
SET @LNAME = 'BHIWANDE'
SET @E_EMAIL = 'r@gmail.com'
SET @p_PHONENUMBER = 7977464457
SET @S_SSN = '367-55-7777'
SET @STYPE = 'Insert'

-- EXECUTING THE PROCEDURE
EXEC MasterInsertSelect @FNAME,@LNAME,@E_EMAIL,@p_PHONENUMBER,@S_SSN,@STYPE;
```

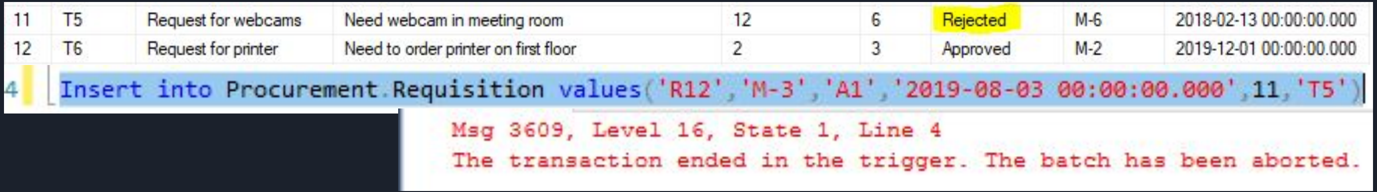


```sql
-- STORED PROCEDURE FROM Procurement.Orders
CREATE PROC dbo.Orders
@OrderDate datetime,
@OrderDetails Varchar(20),
@TotalQuantity INT,
@AdminID varchar(20),
@RequisitionID Varchar(20)
AS
BEGIN
    INSERT INTO Procurement.Orders(OrderDate,OrderDetails,TotalQuantity,AdminID,RequisitionID)
            VALUES (@OrderDate,@OrderDetails,@TotalQuantity,@AdminID,@RequisitionID)
END

--- INSERT STORED PROCEDURE SCRIPT for Procurement.Orders
EXEC  dbo.Orders
@OrderDate = '2019-08-04 00:00:00.000',
@OrderDetails = 'Order againts ticketID T1',
@TotalQuantity = 10,
@AdminID = 'A1',
@RequisitionID = 'R1'
```

Trigger for checking the ticket approval status



```sql
-- TRIGGER FOR CHECKING APPROVAL STATUS OF TICKET AND THEN CREATING REQUISITION
CREATE TRIGGER Procurement.checkapprovalstatus
ON Procurement.Requisition
AFTER INSERT AS
BEGIN
DECLARE @count smallint=0
SELECT  @count=Count(TicketId)
    FROM Procurement.Ticket
    WHERE TicketId = (SELECT TicketId from inserted)
    AND ApprovalStatus = 'Rejected';
IF @count > 0
    BEGIN
        Rollback;
    END
END;
```

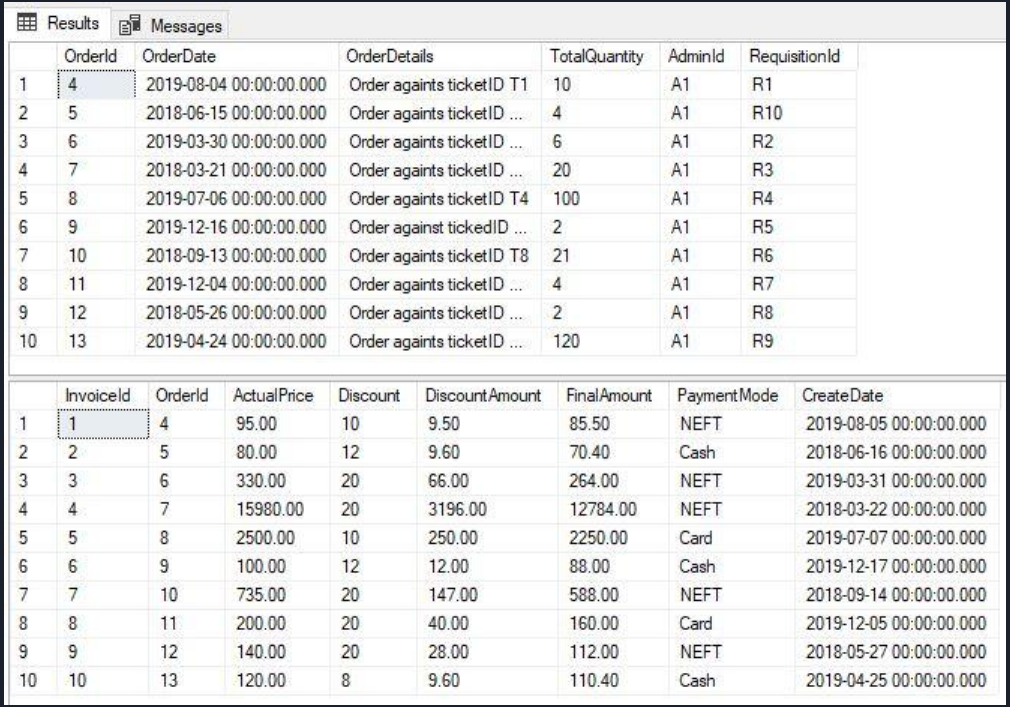


11	T5	Request for webcams	Need webcam in meeting room	12	6	Rejected	M-6	2018-02-13 00:00:00.000
12	T6	Request for printer	Need to order printer on first floor	2	3	Approved	M-2	2019-12-01 00:00:00.000

```sql
Insert into Procurement.Requisition values('R12','M-3','A1','2019-08-03 00:00:00.000',11,'T5')
```

```
Msg 3609, Level 16, State 1, Line 4
The transaction ended in the trigger. The batch has been aborted.
```

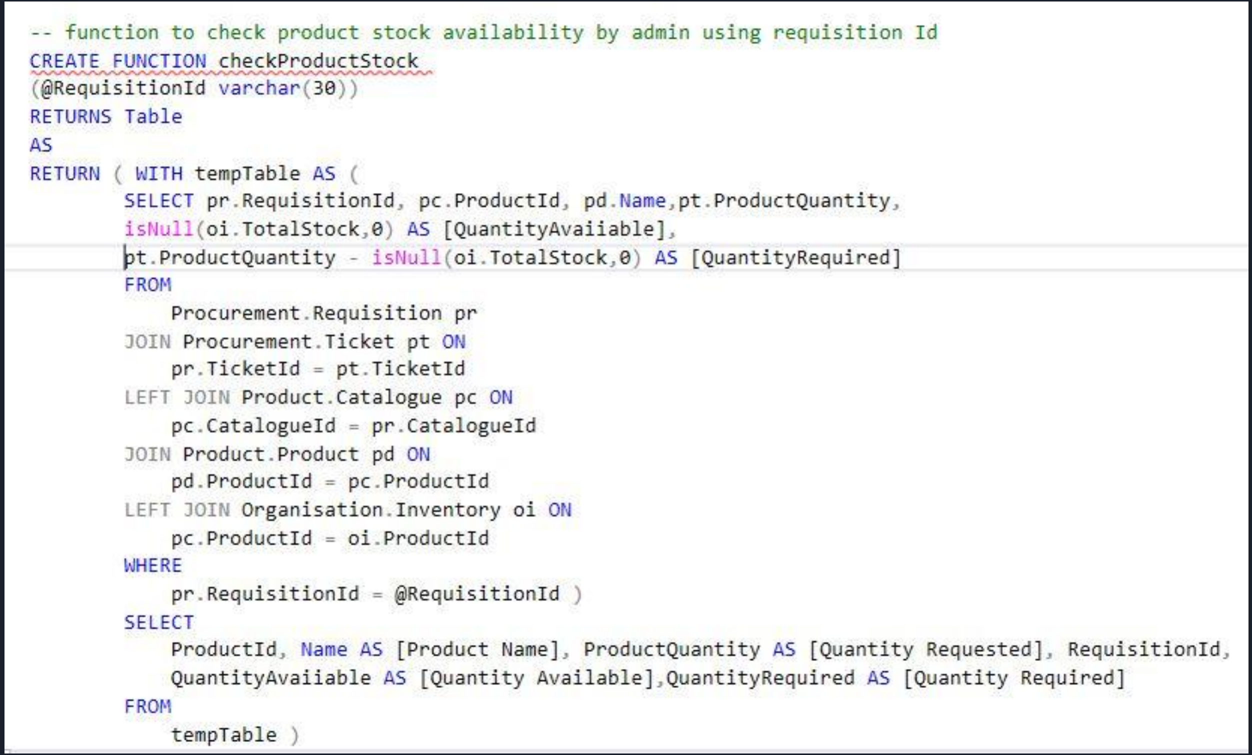
Storing purchase order and invoicing details for easy auditing



	OrderId	OrderDate	OrderDetails	TotalQuantity	AdminId	RequisitionId
1	4	2019-08-04 00:00:00.000	Order againts ticketID T1	10	A1	R1
2	5	2018-06-15 00:00:00.000	Order againts ticketID ...	4	A1	R10
3	6	2019-03-30 00:00:00.000	Order againts ticketID ...	6	A1	R2
4	7	2018-03-21 00:00:00.000	Order againts ticketID ...	20	A1	R3
5	8	2019-07-06 00:00:00.000	Order againts ticketID T4	100	A1	R4
6	9	2019-12-16 00:00:00.000	Order against tickedID ...	2	A1	R5
7	10	2018-09-13 00:00:00.000	Order againts ticketID T8	21	A1	R6
8	11	2019-12-04 00:00:00.000	Order againts ticketID ...	4	A1	R7
9	12	2018-05-26 00:00:00.000	Order againts ticketID ...	2	A1	R8
10	13	2019-04-24 00:00:00.000	Order againts ticketID ...	120	A1	R9

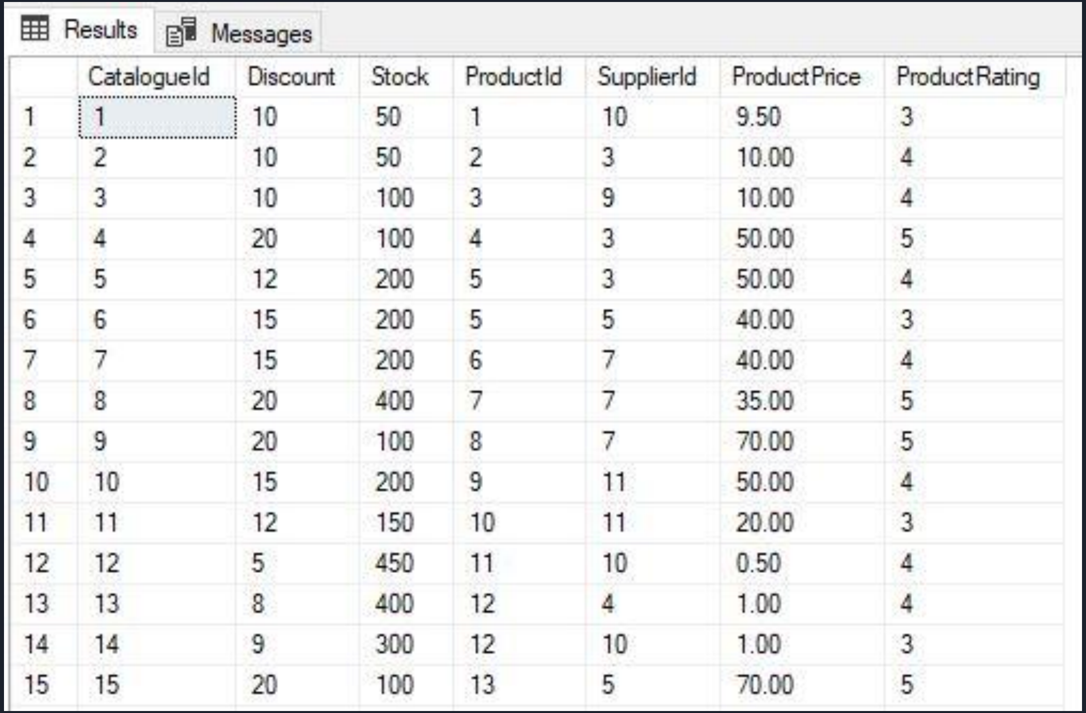
	InvoiceId	OrderId	ActualPrice	Discount	DiscountAmount	FinalAmount	PaymentMode	CreateDate
1	1	4	95.00	10	9.50	85.50	NEFT	2019-08-05 00:00:00.000
2	2	5	80.00	12	9.60	70.40	Cash	2018-06-16 00:00:00.000
3	3	6	330.00	20	66.00	264.00	NEFT	2019-03-31 00:00:00.000
4	4	7	15980.00	20	3196.00	12784.00	NEFT	2018-03-22 00:00:00.000
5	5	8	2500.00	10	250.00	2250.00	Card	2019-07-07 00:00:00.000
6	6	9	100.00	12	12.00	88.00	Cash	2019-12-17 00:00:00.000
7	7	10	735.00	20	147.00	588.00	NEFT	2018-09-14 00:00:00.000
8	8	11	200.00	20	40.00	160.00	Card	2019-12-05 00:00:00.000
9	9	12	140.00	20	28.00	112.00	NEFT	2018-05-27 00:00:00.000
10	10	13	120.00	8	9.60	110.40	Cash	2019-04-25 00:00:00.000

Checking the stock availability in Inventory



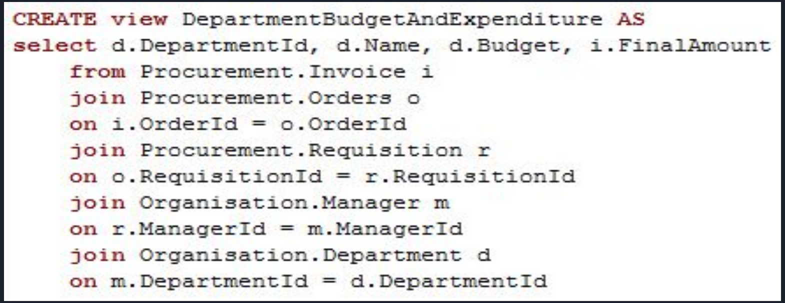
```sql
-- function to check product stock availability by admin using requisition Id
CREATE FUNCTION checkProductStock
(@RequisitionId varchar(30))
RETURNS Table
AS
RETURN ( WITH tempTable AS (
        SELECT pr.RequisitionId, pc.ProductId, pd.Name,pt.ProductQuantity,
        isNull(oi.TotalStock,0) AS [QuantityAvaiiable],
        pt.ProductQuantity - isNull(oi.TotalStock,0) AS [QuantityRequired]
        FROM
            Procurement.Requisition pr
        JOIN Procurement.Ticket pt ON
            pr.TicketId = pt.TicketId
        LEFT JOIN Product.Catalogue pc ON
            pc.CatalogueId = pr.CatalogueId
        JOIN Product.Product pd ON
            pd.ProductId = pc.ProductId
        LEFT JOIN Organisation.Inventory oi ON
            pc.ProductId = oi.ProductId
        WHERE
            pr.RequisitionId = @RequisitionId )
        SELECT
            ProductId, Name AS [Product Name], ProductQuantity AS [Quantity Requested], RequisitionId,
            QuantityAvaiiable AS [Quantity Available],QuantityRequired AS [Quantity Required]
        FROM
            tempTable )
```

Using catalogue for well-organized procurement process

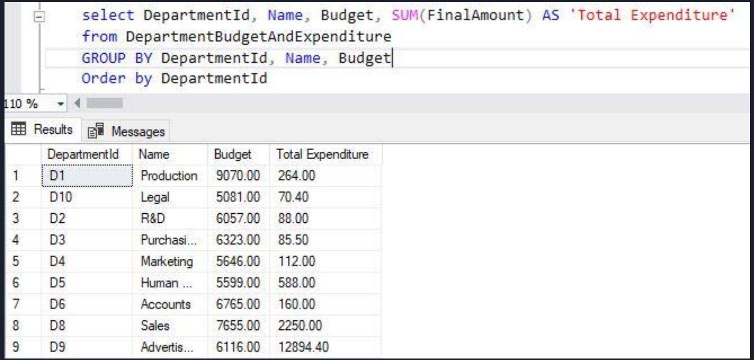


	CatalogueId	Discount	Stock	ProductId	SupplierId	ProductPrice	ProductRating
1	1	10	50	1	10	9.50	3
2	2	10	50	2	3	10.00	4
3	3	10	100	3	9	10.00	4
4	4	20	100	4	3	50.00	5
5	5	12	200	5	3	50.00	4
6	6	15	200	5	5	40.00	3
7	7	15	200	6	7	40.00	4
8	8	20	400	7	7	35.00	5
9	9	20	100	8	7	70.00	5
10	10	15	200	9	11	50.00	4
11	11	12	150	10	11	20.00	3
12	12	5	450	11	10	0.50	4
13	13	8	400	12	4	1.00	4
14	14	9	300	12	10	1.00	3
15	15	20	100	13	5	70.00	5

View for analysing the expense and budget of each department

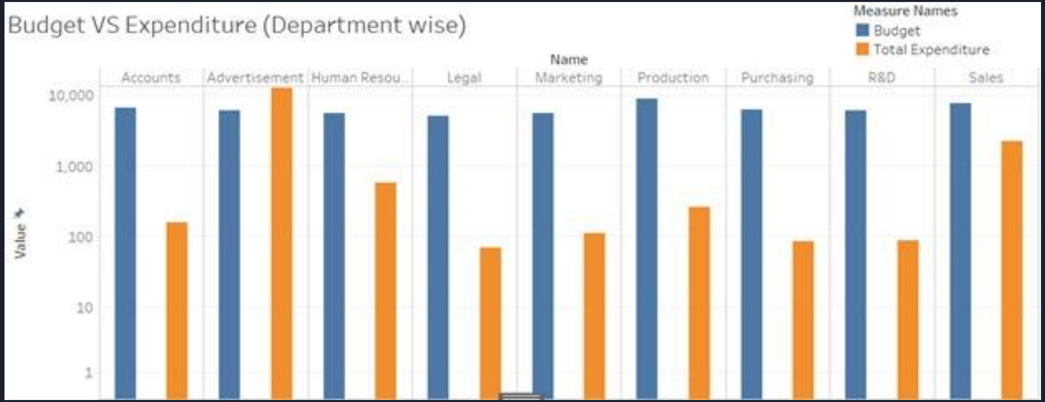


```
CREATE view DepartmentBudgetAndExpenditure AS
select d.DepartmentId, d.Name, d.Budget, i.FinalAmount
    from Procurement.Invoice i
    join Procurement.Orders o
    on i.OrderId = o.OrderId
    join Procurement.Requisition r
    on o.RequisitionId = r.RequisitionId
    join Organisation.Manager m
    on r.ManagerId = m.ManagerId
    join Organisation.Department d
    on m.DepartmentId = d.DepartmentId
```

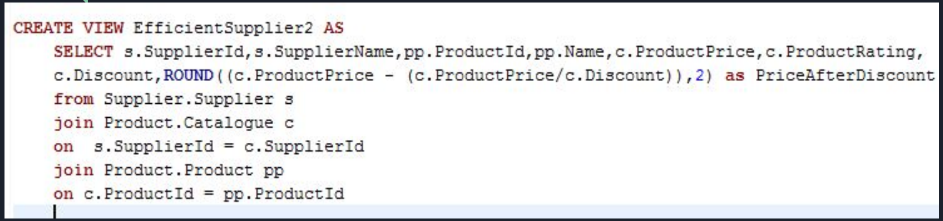


```
select DepartmentId, Name, Budget, SUM(FinalAmount) AS 'Total Expenditure'
from DepartmentBudgetAndExpenditure
GROUP BY DepartmentId, Name, Budget
Order by DepartmentId
```

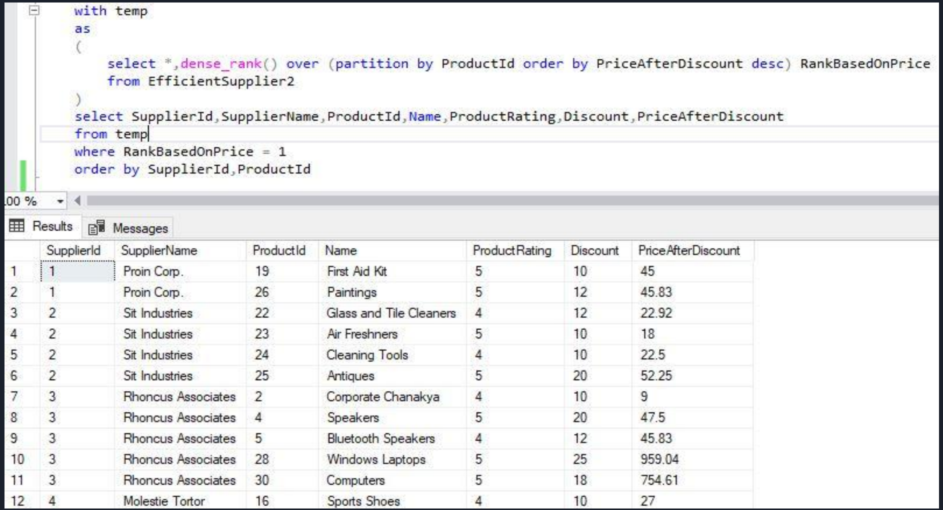
	DepartmentId	Name	Budget	Total Expenditure
1	D1	Production	9070.00	264.00
2	D10	Legal	5081.00	70.40
3	D2	R&D	6057.00	88.00
4	D3	Purchasi...	6323.00	85.50
5	D4	Marketing	5646.00	112.00
6	D5	Human ...	5599.00	588.00
7	D6	Accounts	6765.00	160.00
8	D8	Sales	7655.00	2250.00
9	D9	Advertis...	6116.00	12894.40



View for choosing an efficient supplier based on rating/available discounts

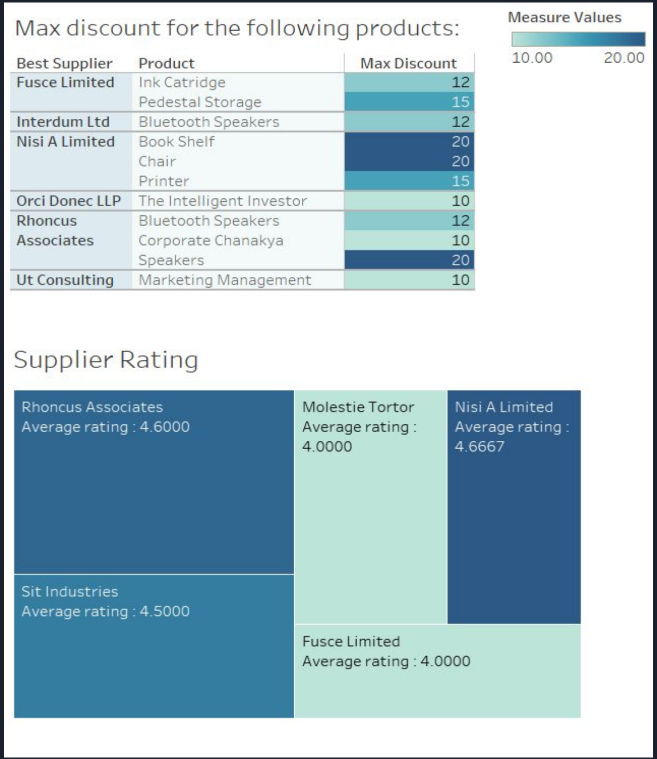


```sql
CREATE VIEW EfficientSupplier2 AS
    SELECT s.SupplierId,s.SupplierName,pp.ProductId,pp.Name,c.ProductPrice,c.ProductRating,
    c.Discount,ROUND((c.ProductPrice - (c.ProductPrice/c.Discount)),2) as PriceAfterDiscount
    from Supplier.Supplier s
    join Product.Catalogue c
    on  s.SupplierId = c.SupplierId
    join Product.Product pp
    on c.ProductId = pp.ProductId
```



```sql
with temp
as
(
    select *,dense_rank() over (partition by ProductId order by PriceAfterDiscount desc) RankBasedOnPrice
    from EfficientSupplier2
)
select SupplierId,SupplierName,ProductId,Name,ProductRating,Discount,PriceAfterDiscount
from temp
where RankBasedOnPrice = 1
order by SupplierId,ProductId
```

	SupplierId	SupplierName	ProductId	Name	ProductRating	Discount	PriceAfterDiscount
1	1	Proin Corp.	19	First Aid Kit	5	10	45
2	1	Proin Corp.	26	Paintings	5	12	45.83
3	2	Sit Industries	22	Glass and Tile Cleaners	4	12	22.92
4	2	Sit Industries	23	Air Freshners	5	10	18
5	2	Sit Industries	24	Cleaning Tools	4	10	22.5
6	2	Sit Industries	25	Antiques	5	20	52.25
7	3	Rhoncus Associates	2	Corporate Chanakya	4	10	9
8	3	Rhoncus Associates	4	Speakers	5	20	47.5
9	3	Rhoncus Associates	5	Bluetooth Speakers	4	12	45.83
10	3	Rhoncus Associates	28	Windows Laptops	5	25	959.04
11	3	Rhoncus Associates	30	Computers	5	18	754.61
12	4	Molestie Tortor	16	Sports Shoes	4	10	27



Max discount for the following products:

Measure Values: 10.00 – 20.00

Best Supplier	Product	Max Discount
Fusce Limited	Ink Catridge	12
	Pedestal Storage	15
Interdum Ltd	Bluetooth Speakers	12
Nisi A Limited	Book Shelf	20
	Chair	20
	Printer	15
Orci Donec LLP	The Intelligent Investor	10
Rhoncus Associates	Bluetooth Speakers	12
	Corporate Chanakya	10
	Speakers	20
Ut Consulting	Marketing Management	10

Supplier Rating

Rhoncus Associates
Average rating : 4.6000

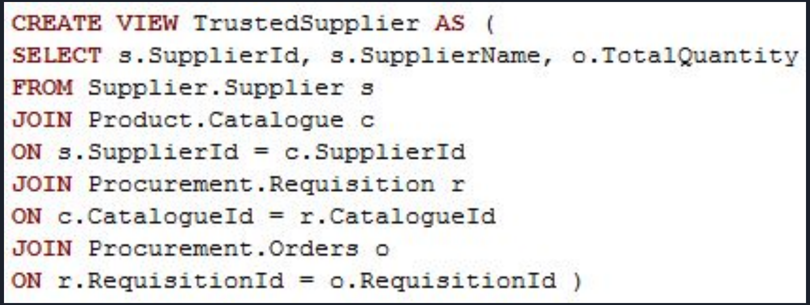
Molestie Tortor
Average rating : 4.0000

Nisi A Limited
Average rating : 4.6667

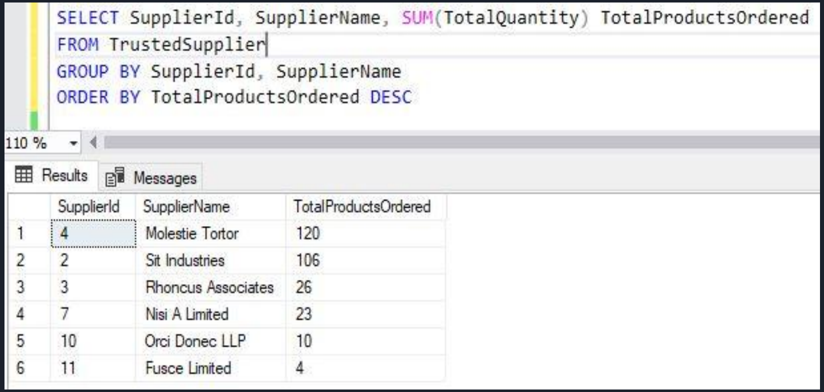
Sit Industries
Average rating : 4.5000

Fusce Limited
Average rating : 4.0000

Viewing Trusted Supplier for the Organisation

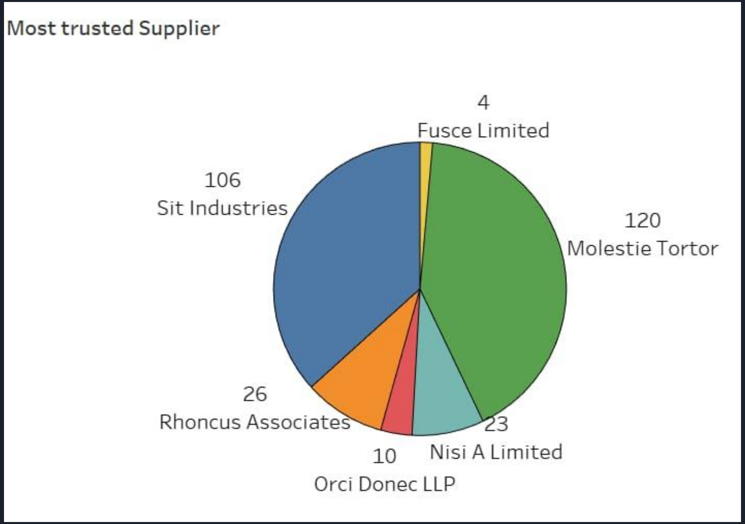


```
CREATE VIEW TrustedSupplier AS (
SELECT s.SupplierId, s.SupplierName, o.TotalQuantity
FROM Supplier.Supplier s
JOIN Product.Catalogue c
ON s.SupplierId = c.SupplierId
JOIN Procurement.Requisition r
ON c.CatalogueId = r.CatalogueId
JOIN Procurement.Orders o
ON r.RequisitionId = o.RequisitionId )
```



```
SELECT SupplierId, SupplierName, SUM(TotalQuantity) TotalProductsOrdered
FROM TrustedSupplier
GROUP BY SupplierId, SupplierName
ORDER BY TotalProductsOrdered DESC
```

	SupplierId	SupplierName	TotalProductsOrdered
1	4	Molestie Tortor	120
2	2	Sit Industries	106
3	3	Rhoncus Associates	26
4	7	Nisi A Limited	23
5	10	Orci Donec LLP	10
6	11	Fusce Limited	4



Thank You..!!

Any Questions?