CREATE TRIGGER AFFECTED\_AREA\_POPULATION ON Earthquake

FOR INSERT

AS

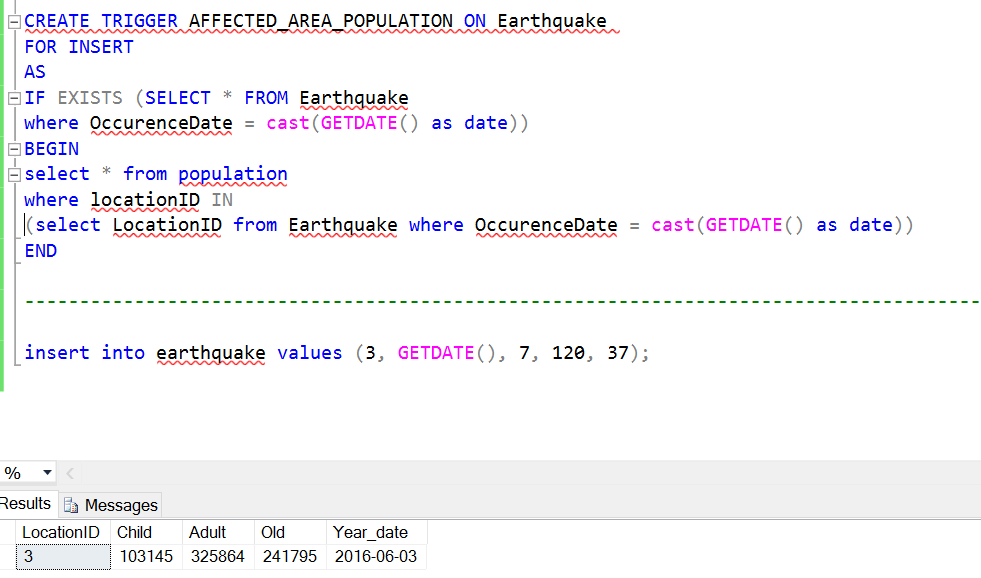
IF EXISTS (SELECT \* FROM Earthquake where OccurenceDate = cast(GETDATE() as date))

BEGIN

select \* from population where locationID IN (select LocationID from Earthquake where OccurenceDate = cast(GETDATE() as date))

END

insert into earthquake values (3, GETDATE(), 7, 120, 37);



CREATE PROCEDURE MinDistance

@location int,

@latitude float,

@longitude float,

@myCurrentLongitude float,

@myCurrentLatitude float

AS

DECLARE @locationID int

DECLARE @result int

DECLARE @radiusOfTheEarth int

SET @radiusOfTheEarth = 6371--km

BEGIN

SELECT @locationID = @location

select @result = ( @radiusOfTheEarth

\* acos( cos( radians(@myCurrentLatitude) )

\* cos( radians( @latitude ) )

\* cos( radians( @longitude ) - radians(@myCurrentLongitude) ) + sin( radians(@myCurrentLatitude) )

\* sin( radians( @latitude ) ) ) )

RETURN @result

END

create table distance (

LocationID int,

Distance int)

--Cursor Creation

DECLARE @loc int

DECLARE @lat float

DECLARE @long float

DECLARE @epix float

DECLARE @epiy float

DECLARE @locationID int

DECLARE @Result int

DECLARE curs CURSOR LOCAL FAST\_FORWARD FOR

select l.LocationID, LocationXCoordinate, LocationYCoordinate, EpicXcoordinate, EpicYcoordinate

from earthquake e, Location l

where e.OccurenceDate =(CAST(GETDATE() as DATE))

and l.LocationID NOT IN

(select LocationID from Earthquake where OccurenceDate =cast(GETDATE() as date))

OPEN curs

FETCH NEXT FROM curs INTO @loc, @lat, @long, @epix, @epiy

WHILE @@FETCH\_STATUS = 0 BEGIN

EXECUTE @Result = MinDistance @loc, @lat, @long, @epix, @epiy

INSERT into distance values(@loc, @Result)

FETCH NEXT FROM curs INTO @loc, @lat, @long, @epix, @epiy

PRINT @Result

--PRINT @Result

END

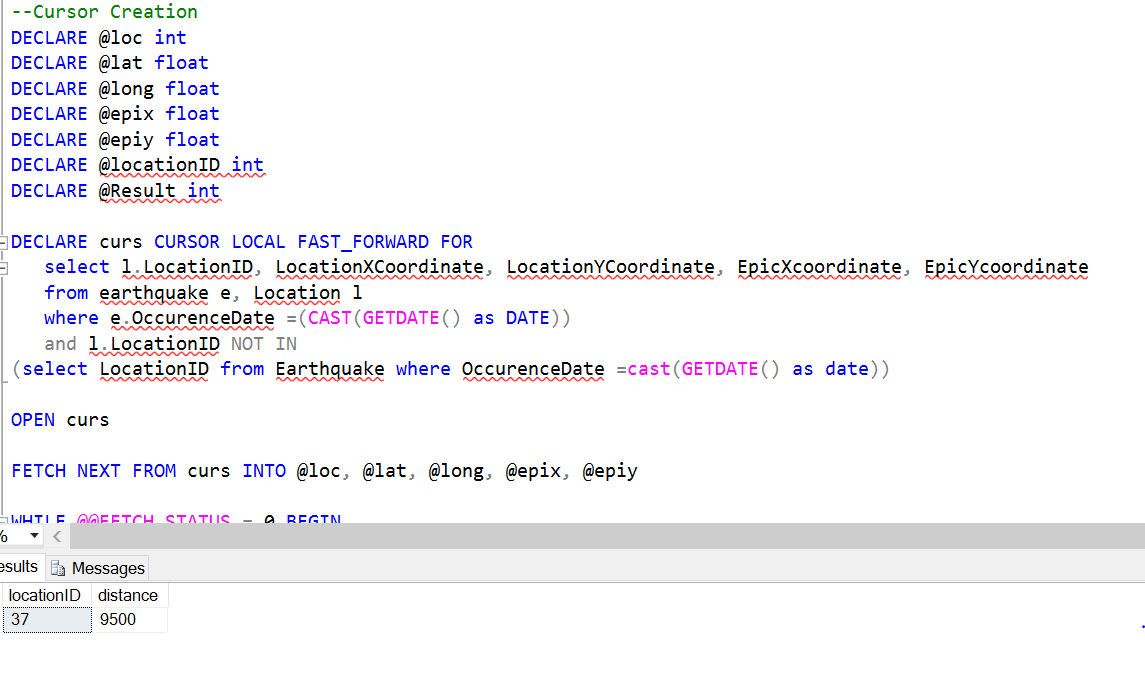
select locationID, distance from distance

where distance = (select min(distance) from distance)

--truncate table distance

CLOSE curs

DEALLOCATE curs



View to get the allocated volunteers and to which organization(NGO, COMPANY) they belong

create view VolunteerList as

(

select row\_number() over ( order by PERSON\_NAME) as AllocatedID, PERSON\_NAME, Type

from (

select Name as PERSON\_NAME, 'NGO' as Type

from NGOVOLUNTEER n

where NGOId IN (select NGOID from NGO where LocationID = 10)

UNION

select Name, 'COMPANY' as Type

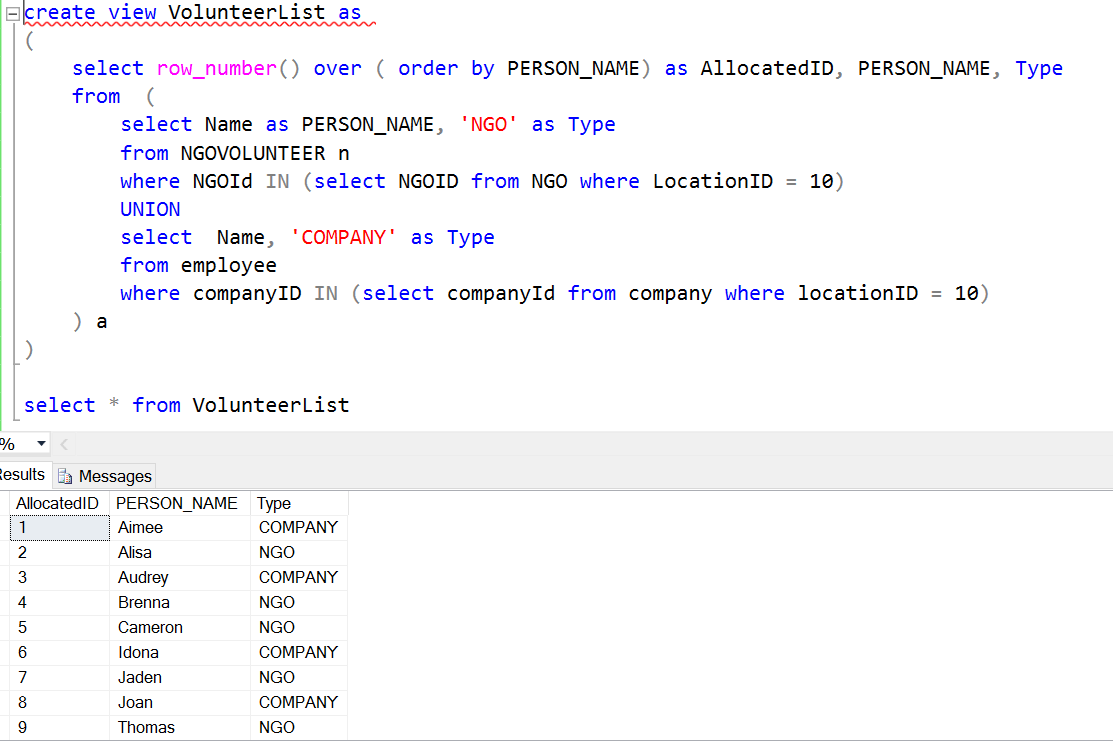
from employee

where companyID IN (select companyId from company where locationID = 10)

) a

)

select \* from VolunteerList



--Procedure to get the message from a specific person

create procedure person\_message

@personId int

AS

DECLARE @message int

BEGIN

select \* from INFORMATION where AllocatedID = @personId

END

--DECLARE @Message

exec person\_message @personID = 2

--Cursor Creation

DECLARE @personID int

DECLARE @Result int

DECLARE curs CURSOR LOCAL FAST\_FORWARD FOR

select AllocatedID from VolunteerList v

OPEN curs

FETCH NEXT FROM curs INTO @personID

WHILE @@FETCH\_STATUS = 0 BEGIN

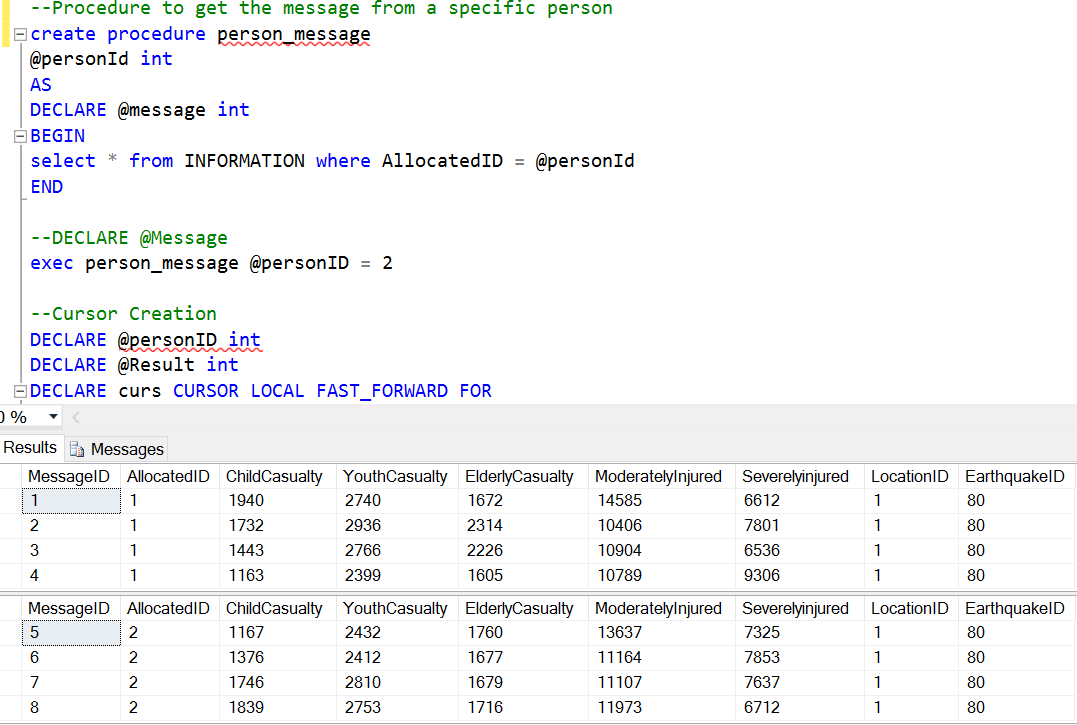
EXECUTE @Result = person\_message @personID

FETCH NEXT FROM curs INTO @personID

PRINT @Result

--PRINT @Result

END



--Assign message to analyst and create report

create procedure RetriveMessage

@allocatedID varchar,

@empID varchar(200)

as

DECLARE @SQL nvarchar(max)

begin

SET @SQL = 'create view AssignedMessage AS

select '+ @empID +' as EmployeeID,MessageID, LocationID, EarthQuakeID, ChildCasualty, YouthCasualty,

ElderlyCasualty, ModeratelyInjured, SeverelyInjured from information where AllocatedID = ' + @allocatedID

EXEC (@SQL)

INSERT INTO Report (EmployeeID, LocationID, EarthquakeID, ChildCasualty, YouthCasualty, ElderlyCasualty, ModeratelyInjured, SeverelyInjured)

(

select EmployeeID, LocationID, EarthquakeID, avg(ChildCasualty), avg(YouthCasualty), avg(ElderlyCasualty), avg(ModeratelyInjured), avg(SeverelyInjured)

from AssignedMessage m

where m.LocationID = 1

group by m.EmployeeID, m.LocationID, m.EarthQuakeID

)

end

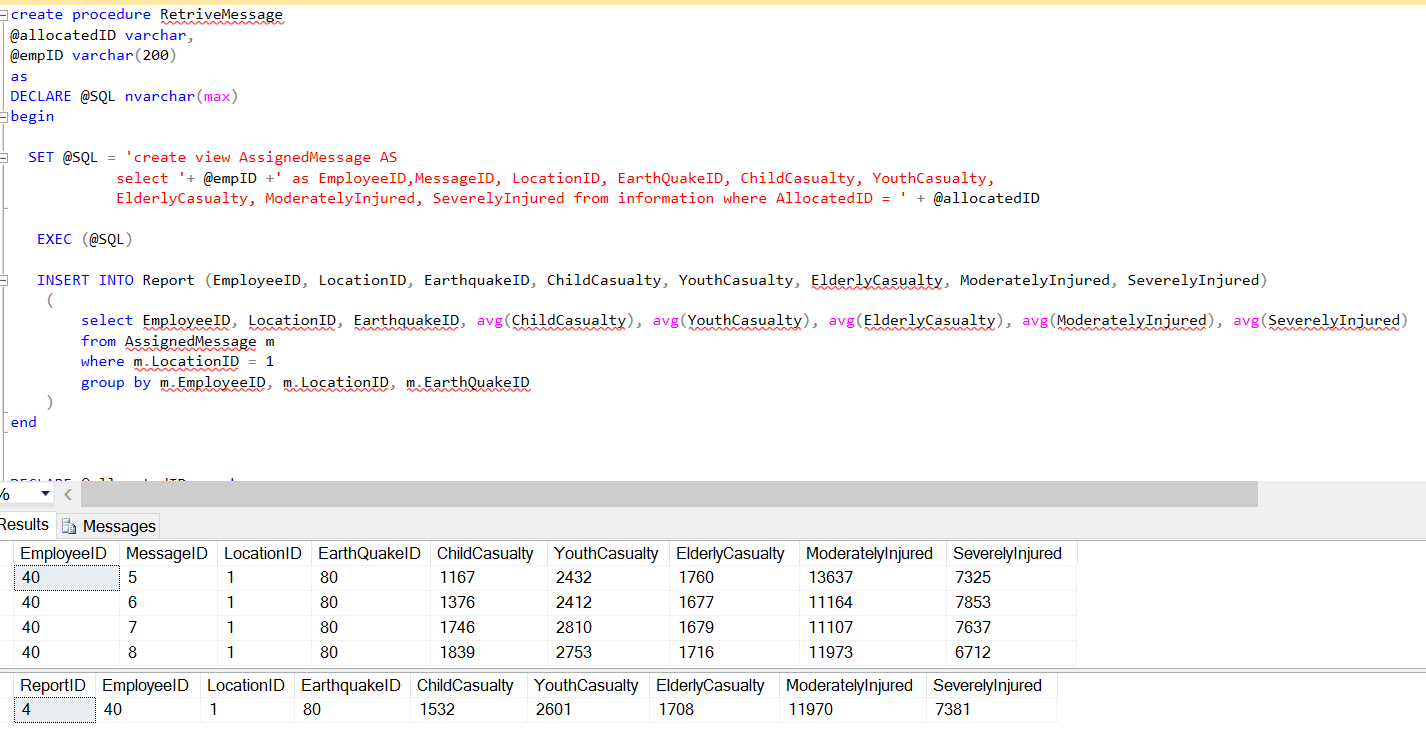
DECLARE @allocatedID varchar

DECLARE @empID varchar

execute RetriveMessage @allocatedID = 2, @empID = 40

select \* from AssignedMessage

select \* from Report



**Analysis Procedure**

--Analysis Procedure

create procedure AllocateResource

@location int

AS

BEGIN

insert into Analysis (GenerateDate, LocationID) values (cast(getdate() as Date), @location)

Update Analysis set

--LocationID = @location,

Rice = (select (CEILING(((p.Child - r.ChildCasualty) /1000) \* 250) + CEILING(((p.Adult - r.YouthCasualty) /1000) \*400) + CEILING(((p.Old - r.ElderlyCasualty) / 1000) \*350))

from Report r

INNER JOIN Population p

ON r.LocationID = p.LocationID

where p.LocationID = @location),

Water = (select (CEILING(((p.Child - r.ChildCasualty) \* 1.5)) + CEILING((p.Adult - r.YouthCasualty) \* 3) + CEILING((p.Old - r.ElderlyCasualty) \*2.5))

from Report r

INNER JOIN Population p

ON r.LocationID = p.LocationID

where p.LocationID = @location),

Tent = (select (CEILING(((p.Child - r.ChildCasualty) + CEILING(p.Adult - r.YouthCasualty) + CEILING(p.Old - r.ElderlyCasualty)) / 1000))

from Report r

INNER JOIN Population p

ON r.LocationID = p.LocationID

where p.LocationID = @location),

HospitalBed = (select (CEILING(r.severelyinjured))

from Report r

INNER JOIN Population p

ON r.LocationID = p.LocationID

where p.LocationID = @location),

FirstAid = (select (CEILING((r.ModeratelyInjured \* 3/ 4)))

from Report r

INNER JOIN Population p

ON r.LocationID = p.LocationID

where p.LocationID = @location)

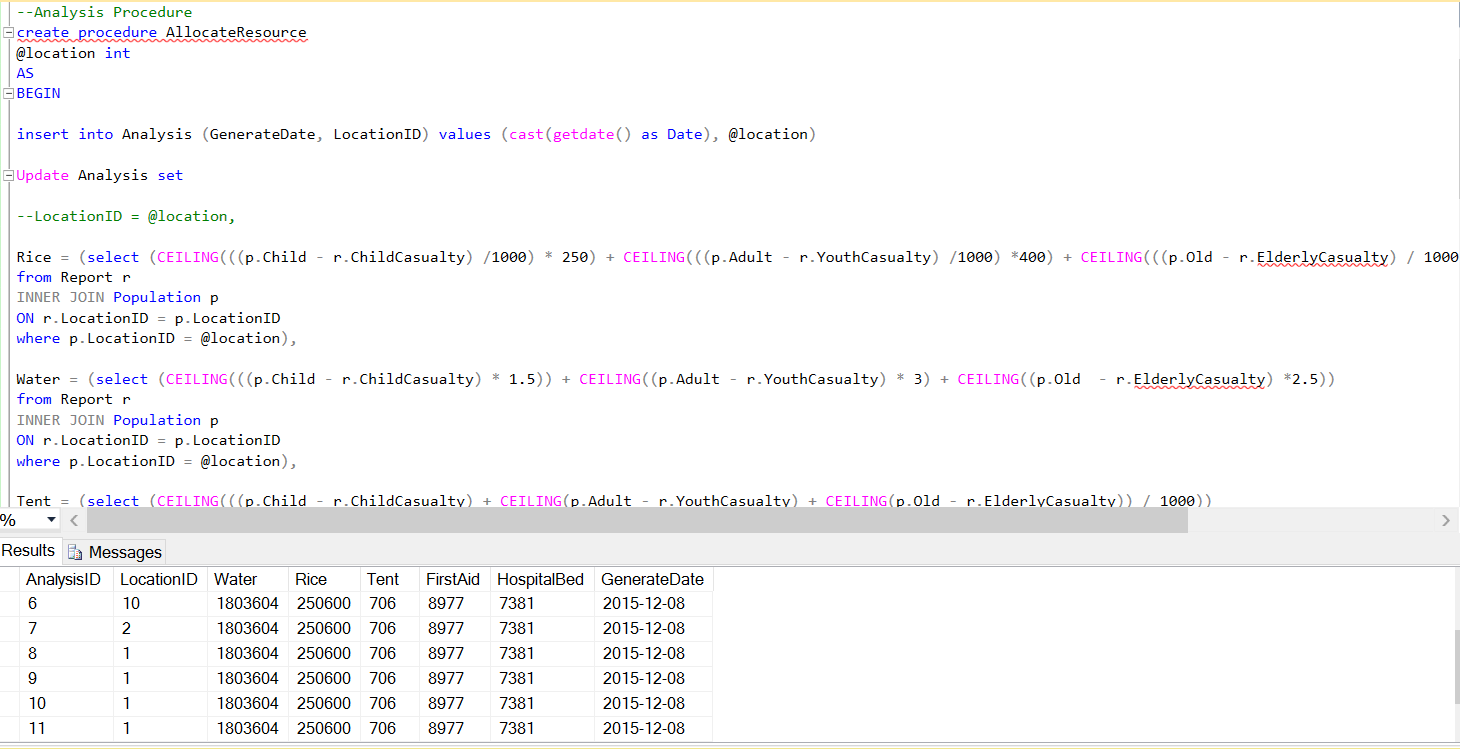
where GenerateDate = (CAST(GETDATE() as date))

END

execute AllocateResource @location = 1

select \* from analysis

select \* from report where locationId = 1



TRIGGER Resource Allocated

create Trigger resourceManagement on Analysis

after update

as

if exists (Select \* from Analysis where GenerateDate = (CAST(GETDATE() as Date)))

BEGIN

DECLARE @warehouseID int

SET @warehouseID = (select top 1 WarehouseID from Warehouse

where locationID IN

(select e.locationID from distance e

where e.distance = (select min(d.distance) from distance d)));

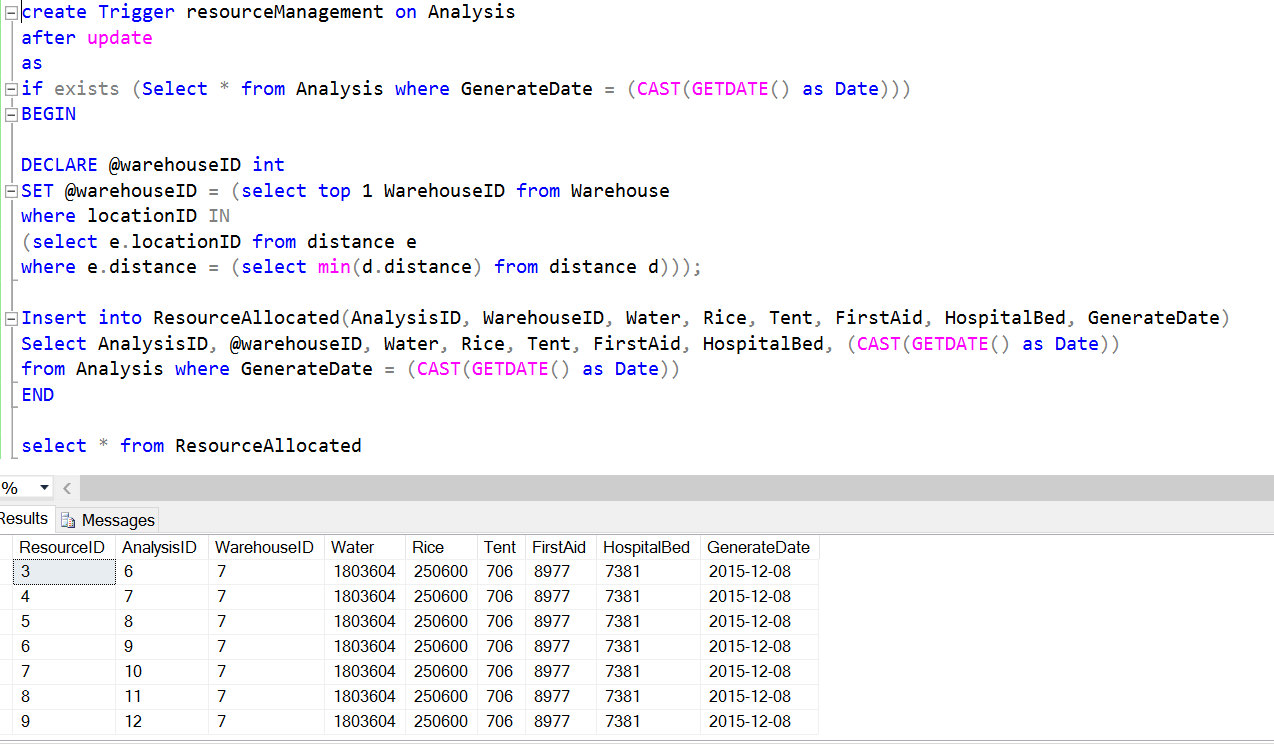
Insert into ResourceAllocated(AnalysisID, WarehouseID, Water, Rice, Tent, FirstAid, HospitalBed, GenerateDate)

Select AnalysisID, @warehouseID, Water, Rice, Tent, FirstAid, HospitalBed, (CAST(GETDATE() as Date))

from Analysis where GenerateDate = (CAST(GETDATE() as Date))

END

select \* from ResourceAllocated



create Trigger generateTracking on ResourceAllocated

after Insert

as

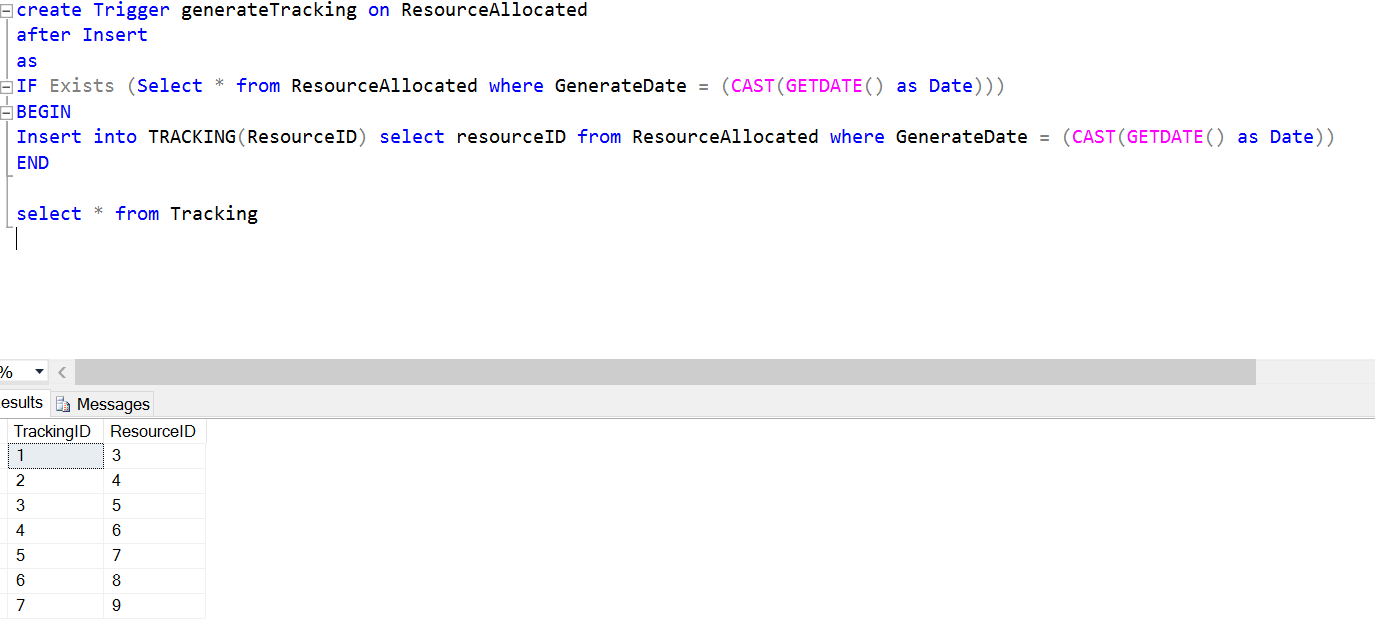
IF Exists (Select \* from ResourceAllocated where GenerateDate = (CAST(GETDATE() as Date)))

BEGIN

Insert into TRACKING(ResourceID) select resourceID from ResourceAllocated where GenerateDate = (CAST(GETDATE() as Date))

END

select \* from Tracking



Find out the total employees with Role at the field

select count(n.roleID), r.RoleName

from NgoVolunteer n

INNER JOIN Role r

ON n.roleID = r.RoleID

where n.NgoID IN (select ngoID from NGO where locationID = 10)

group by r.RoleName

select count(e.roleID), r.RoleName

from Employee e

INNER JOIN Role r

ON e.roleID = r.RoleID

where e.companyID IN (select companyId from company where locationID = 10)

group by r.RoleName

