1. TRIGGER: This trigger is activated as soon as earthquake entry is made in the database.

CREATE TRIGGER EFFECTED\_AREA\_POPULATION ON Earthquake

FOR INSERT

AS

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

BEGIN

select \* from population where location\_id IN (select LocationID from Earthquake where Occurance\_Date = cast(GETDATE() as date))

END

---------------------------------------------------------------------------------------

delete from Earthquake where LocationID = 3

insert into earthquake values (5,3, GETDATE(), 7, 120, 37);

--Minimum Distance

CREATE TRIGGER EFFECTED\_AREA\_POPULATION ON Earthquake

FOR INSERT

AS

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

BEGIN

select \* from population where location\_id IN (select LocationID from Earthquake where Occurance\_Date = cast(GETDATE() as date))

END

insert into earthquake values (5,3, GETDATE(), 7, 120, 37);

drop procedure MinDistance

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 ) ) ) )

--select @locationID as 'Location ID', @result as 'DISTANCE'

--select @result = @lat1 + @lat2

--SELECT @result = LocationXCordinate

--FROM location

--where Location\_ID = 1

--RETURN @locationID

RETURN @result

END

GO

DECLARE @locID int

DECLARE @result int

DECLARE @loc int

DECLARE @lat float

DECLARE @long float

DECLARE @epix float

DECLARE @epiy float

select @locID = LocationID from earthquake where EarthquakeID = 5

select @lat=LocationXCordinate from location where Location\_ID = 2

select @long=LocationYCordinate from location where Location\_ID = 2

select @epix=epicenterxcoordinate from Earthquake where locationid = 2

select @epiy=epicenterycoordinate from Earthquake where locationid = 2

EXECUTE @result = MinDistance @locID, @lat, @long,@epix,@epiy

PRINT @loc

PRINT @result

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 @distant TABLE (LocationID int, Dist int)

DECLARE curs CURSOR LOCAL FAST\_FORWARD FOR

select Location\_ID, LocationXCordinate, LocationYCordinate, Epicenterxcoordinate, Epicenterycoordinate from earthquake e, Location where e.Occurance\_Date=(CAST(GETDATE() as DATE)) and Location\_ID NOT IN

(select LocationID from Earthquake where Occurance\_Date =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 FirstName as PERSON\_NAME, VOLUNTEER\_ID as PERSON\_ID, 'NGO' as Type from NGOVOLUNTEERS where NGO\_Id

IN (select NGO\_ID from NGO where Location\_ID = 2)

UNION

(select firstname, ID, 'COMPANY' as Type

from employee

where ID IN (select companyId from company where location\_ID = 2))

select \* from VolunteerList

-----------------------------------------------------------------------------------------

Procedure to get the message from a specific person

create procedure person\_message

@person\_Id int

AS

DECLARE @message int

BEGIN

select \* from message where PersonID = @person\_Id

END

----------------------------------------------------------------------------------------

Assigned Message to Employee

create table AssignedMessage (

EmpId int,

MessageId int,

LocationId int,

EarthquakeId int,

ChildCasualtyNo int,

YouthCasualtyNo int,

ElderlyCasualtyNo int,

ModeratelyInjuredNo int,

SeverlyInjuredNo int);

drop procedure person\_message

create procedure person\_message

@person\_Id int

AS

DECLARE @message int

BEGIN

INSERT into AssignedMessage (EmpId, MessageId, LocationId, EarthquakeId, ChildCasualtyNo, YouthCasualtyNo,

ElderlyCasualtyNo ,

ModeratelyInjuredNo ,SeverlyInjuredNo) select 1, MessageID, LocationID, EarthQuakeID, ChildCasualtyNo, YouthCasualtyNo,

ElderlyCasulatyNo ,

ModeratelyInjuredNo ,SeverelyInjuredNo from message where PersonID = @person\_Id

END

GO

EXECUTE person\_message @person\_id =2

select \* from AssignedMessage

----------------------------------------------------------------------------------------

Analysis Procedure

create procedure AllocateResource

@location int

AS

BEGIN

insert into Analysis (Generated\_Date) values (cast(getdate() as Date))

Update Analysis set

Location\_ID = @location,

Rice = (select (CEILING(((p.PeopleUnder14 - m.ChildCasualtyNo) /1000) \* 250) + CEILING(((p.People14to51 - m.YouthCasualtyNo) /1000) \*400) + CEILING(((p.PeopleGreaterThan51 - m.ElderlyCasualtyNo) / 1000) \*350))

from AssignedMessage m

INNER JOIN PASTPOPULATION p

ON m.LocationID = p.Location\_ID

where p.Location\_ID = @location),

Water = (select (CEILING(((p.PeopleUnder14 - m.ChildCasualtyNo) \* 1.5)) + CEILING((p.People14to51 - m.YouthCasualtyNo) \* 3) + CEILING((p.PeopleGreaterThan51 - m.ElderlyCasualtyNo) \*2.5))

from AssignedMessage m

INNER JOIN PASTPOPULATION p

ON m.LocationID = p.Location\_ID

where p.Location\_ID = @location),

Tent = (select (CEILING(((p.PeopleUnder14 - m.ChildCasualtyNo) + CEILING(p.People14to51 - m.YouthCasualtyNo) + CEILING(p.PeopleGreaterThan51 - m.ElderlyCasualtyNo)) / 1000))

from AssignedMessage m

INNER JOIN PASTPOPULATION p

ON m.LocationID = p.Location\_ID

where p.Location\_ID = @location),

HospitalBed = (select (CEILING(m.SeverlyInjuredNo))

from AssignedMessage m

INNER JOIN PASTPOPULATION p

ON m.LocationID = p.Location\_ID

where p.Location\_ID = @location),

FirstAid = (select (CEILING((m.ModeratelyInjuredNo \* 3/ 4)))

from AssignedMessage m

INNER JOIN PASTPOPULATION p

ON m.LocationID = p.Location\_ID

where p.Location\_ID = @location)

where Generated\_Date = (CAST(GETDATE() as date))

END

execute AllocateResource @location = 2

-----------------------------------------------------------------------------------------------------------------------------------------

create Trigger resourceManagement on Analysis

after update

as

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

BEGIN

DECLARE @warehouseID int

SET @warehouseID = (select 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(Resource\_ID,Warehouse\_ID,Water,Rice,Tent,FirstAid,HospitalBed,generated\_date)

Select ResourceID, @warehouseID, Water, Rice, Tent, FirstAid, HospitalBed, (CAST(GETDATE() as Date)) from Analysis where Generated\_Date = (CAST(GETDATE() as Date))

END

Generate tracking as soon as data inserted in resource allocated table

create Trigger generateTracking on ResourceAllocated

after Insert

as

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

BEGIN

Insert into TRACKING(Tracking\_ID,Resource\_ID,LocationID) select 1, resource\_ID, LocationID from ResourceAllocated where Generated\_Date = (CAST(GETDATE() as Date))

END

insert into ResourceAllocated values(1, 8, 1089, 333,444,32,122,11,(CAST(GETDATE() as DATE)),2)

---------------------------------------------------------------------------------