MSIS 2603

DATABASE MANAGEMENT SYSTEM

DBMS SYSTEM OF WATERSHED IMPLEMENTATION FOR NGOs.

SUBMITTED BY:

BHAGYASHRI BADGUJAR

SCU ID: W1563782

References:
http://www.indiaenvironmentportal.org.in/content/349544/common-guidelines-for-watershed-development-projects-2011/

Note:
In the above link a brief description of the implementation of watershed development scheme by an Indian organisation is mentioned. But for the implementation of this DBMS project I have narrowed down the business and picked-up only the most important entities.

1. Business/Organisation description and Database description:

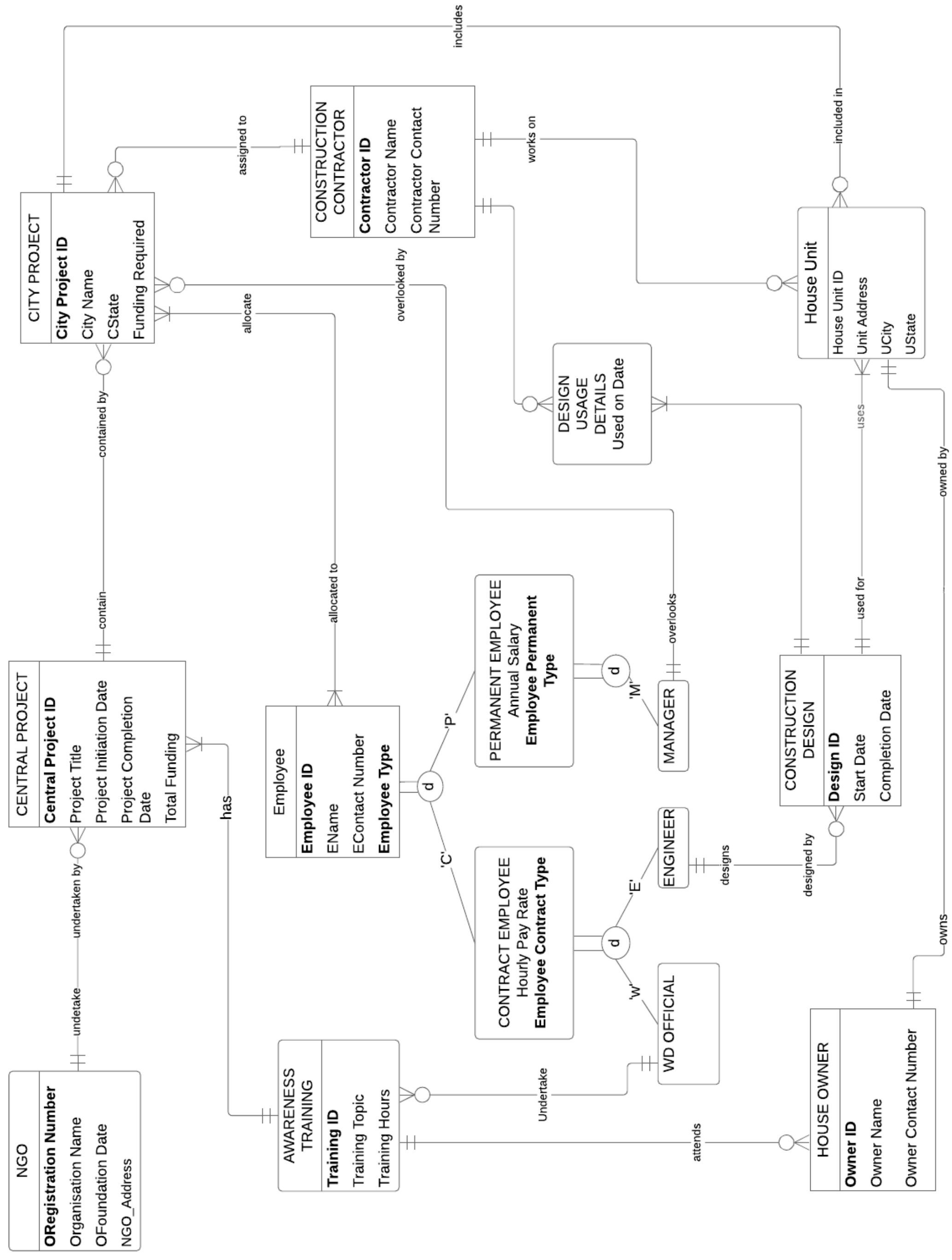
1.1 Business Description: NGOs like Sanjay Incorporation undertakes multiple central level Watershed Development projects funded by central government. These projects are an initiation towards saving the rainwater by 80% and then reusing it for household purposes. The NGO divides these Central government projects into multiple city level projects along the way with the funding for the respective city projects. One central level projects contains many city projects in depending on the focus criteria of which city needs this kind of implementation desperately. Manually, depending on the scope of the project an awareness training is assigned to a central project. Additionally, manually a manager and a contractor gets assigned to a city project. Detailed analysis is done manually too by the manager regarding on which house units the implementation of the construction for rooftop watershed harvesting construction should be done. Sanjay Incorporation ensures that many house units must be included in one city project for one central project. Once the house units are selected, the house owners owning that particular unit attends the training at the headquarters of the NGO as it has been made mandatory to them by the NGOs. This training facilitates the house owners to understand the benefits of this implementation scheme thus encouraging them to actively cooperate with the construction work and in the end attain satisfaction with the overall result getting from this implementation scheme. This training is undertaken by a watershed development (WD) official who is a type of employee of the NGO, hired on a contract basis. Manually, they assign a contractor to a city project. Contractor is a person who takes in-charge of the construction for the house units within that city. For the construction of different house units, he uses the construction designs designed by an engineer for that unit. An engineer is also a type of employee of the NGO hired on a contractual basis. A manager manages a city project for a particular city. Employees are a crucial part of this NGO who are responsible for handling the operations of these city projects. Multiple types or employees are included in city projects which includes Contractual Employees(WD Official, Engineer) and Permanent Employees (Manager) all assigned to different tasks in multiple city projects.

Business Value:
Until now all of the information storage and retrieval is paper based which is handled by the managers, as the project proceeds it gets difficult and hectic for them and to the president of the NGO to manage and store all the data manually. With the introduction of this database system in their organisation it will give them an edge of professionalism from peer NGOs as it will remove the bottleneck of offline mundane paperwork. This database will contribute towards meeting the strategic organisational goal. Better Central Project management will be possible with accurate total funding numbers which are supposed to be bounced back to the government. This system will also improve the overall database management.
Paperwork nowadays has become time consuming, costly, difficult to track information, difficult to manage, loses its value as the time passes making it difficult to store and retrieve information. However, the new system will help the NGO to save time on the additional labor work required by manager for documentation process and all his focus will now be on contributing his work towards overlooking at the multiple city projects. With the above benefits of the system, it has a great market value as there are multiple NGOs all around the globe dealing with similar situations. This system will help in smooth running, managing and retrieval of data, thus in the future will help in better organisational decisions.

Database Description:
With solid planning and analysis I figured out the most valuable entities required for these type of organisations. All the important details like which will be recorded during the implementation of the project like organisation details, central projects and its city projects, managers, contractors, design usage details, house units, house owners, awareness trainings, employees working in the NGO can be stored in this database, making this an ideal solution for the current paperwork system. The database includes tables like
NGO, Central Project (main project at state level), City Project (Sub-projects city wise), Construction Contractor, Employees (managers, engineer, WD Officials), Construction Design, House Unit, House Owners and Awareness Training. Central Project will store details like Project Title, Project Initiation Date, Project Completion Date, and Total Funding. City Project will include City Project ID, City Name, its state, and funding required for that city project. Construction contractor table will contain contractor name, and its contact details. House unit table will store details like the address, its state and the city. Construction Design table will include details like Design ID which will help us to identify that design uniquely, its start date of designing and completion date of designing. House Owner table will consist of the owner name and its contact details. Awareness Training provided to these house owners will include the training topic. Employee table will include all the details with different employee types depending on their type of permanent or contractual basis. Employees depending on their salary type will be differentiated which includes managers, engineer and WD officials.

Future Scope: In the future scope this system can be then used to track the development of the project and will help the managers to track the status of the project. Along with this with proper data analysis and surveys, the organisation will also be able to figure out ways to take better decisions which can benefit them in the future.

Conceptual Schema - ER model:



Logical Schema - relational table design:

NGO	**ORegistration Number**	Organisation Name	Foundation Date	NGO_Address

CENTRAL PROJECT	**Central Project ID**	***ORegistration Number***	***Training ID***	Project Title
		Project Initiation Date	Project Completion Date	Total Funding

CITY_PROJECT	**City Project ID**	***Central Project ID***	***Contractor ID***	***MPEmployeeID***
		City Name	CState	Funding Required

CITY_PROJECT_EMPLOYEE_DETAILS	***City Project ID***	***Employee ID***

EMPLOYEE	**Employee ID**	EName	EContact Number	**Employee Type**

EMPLOYEE_PERMANENT	***PEmployeeID***	Annual Salary	**Employee Permanent Type**

EMPLOYEE_PERMANENT_MANAGER	***MPEmployee ID***

EMPLOYEE_CONTRACT	***CEmployeeID***	Hourly Pay Rate	**Employee Contract Type**

EMPLOYEE_CONTRACT_ENGINEER	***ECEmployee ID***

EMPLOYEE_CONTRACT_WD_OFFICIAL	**WCEmployee ID**

AWARENESS_TRAINING	**Training ID**	***WCEmployee ID***	Training Topic	Training Hours

HOUSE_OWNER	**Owner ID**	***Training ID***	***House Unit ID***	Owner Name
				Owner Contact Number

HOUSE UNIT	**House Unit ID**	***City Project ID***	***Contractor ID***	***Design ID***
		Unit Address	UCity	UState

CONSTRUCTION_DESIGN	**Design ID**	***ECEmployee ID***	Start Date	Completion Date

DESIGN_USAGE_DETAILS	**Design Contractor ID**	***Design ID***	***Contractor ID***	Used on Date

CONSTRUCTION CONTRACTOR	**Contractor ID**	Contractor Name	Contractor Contact Number

Data Dictionary:

NGO

Name	Data Type	Constraints	Key	Description	Example Value
ORegistration Number	int	>0	PK	Unique identifier of an organisation	5555
Organisation Name	nvarchar(20)	>0		Name of an organisation	Sanjay Incorporation
OFoundation Date	date			Foundation Date of an organisation	05/19/1995
NGO_Address	nvarchar(255)			Address of the office of NGO	532 Illinois Street Allison Park, PA 15101

Central Project

Name	Data Type	Constraint	Key	Description	Example Value
Central Project ID	int	>0	PK	Unique identifier of a project	1111
ORegistration Number	int	>0	FK	Unique identifier of an organisation	5555
Training ID	int	>0	FK	Unique identifier of an awareness training	109
Project Title	nvarchar(max)			Project Title of a project	Rural WCD Implementation
Project Initiation Date	date			Initiation Date of the project	1/12/2019
Project Completion Date	date			Completion Date of a project	1/12/2022
Total Funding	decimal(10,2)	>0.0		Total funding received for a project	110000

City Project

Name	Data Type	Constraint	Key	Description	Example Value
City Project ID	int	>0	PK	Unique identifier of a city project	11111
Central Project ID	int	>0	FK	Unique identifier of a project	1111
Contractor ID	int		FK	Unique identifier of a construction contractor	6005
MPEmployee ID	int		FK	Unique identifier of a manager	502
City Name	nvarchar(20)			Name of the city	Santa Clara
CState	char(2)			State of the city	CA
Funding Required	decimal(8,2)	>0.0		funding required for the city project	33456.03

City_Project_Employee_Detail

Name	Data Type	Constraint	Key	Description	Example Value
City Project ID	int	>0	PK,FK	City project assigned to employees; unique identifier for a city project; composite identifier for an employee assigned in a city project	11111
Employee ID	int	>0	PK,FK	Employees assigned to a city project; unique identifier for an employee; composite identifier for an city project to an employee	501

Employee

Name	Data Type	Constraint	Key	Description	Example Value
Employee ID	int	>0	PK	Unique identifier of an employee	501
EName	nvarchar(25)			Name of the employee	Nile Neale
EContact Number	char(10)			Contact number of the employee	6622031317
Employee Type	char(2)	('C', 'P')		Discriminator for employee type, permanent employee (P) or contract (C) employee type	C

Employee_Permanent

Name	Data Type	Constraint	Key	Description	Example Value
PEmployeeID	int	>0	PK, FK	Unique identifier of a permanent employee	502
Annual Salary	decimal(7,2)	>0		Annual salary of a permanent employee	55000
Employee Permanent Type	char(2)	('M')		Discriminator for permanent employee type, manager(M)	M

Employee_Permanent_Manager

Name	Data Type	Constraint	Key	Description	Example Value
MPEmployee ID	int	>0	PK, FK	Manager of a city project;Unique identifier of a permanent manager employee	502

Employee_Contract

Name	Data Type	Constraint	Key	Description	Example Value
CEmployeeID	int	>0	PK, FK	Unique identifier of a contractor employee	501
Hourly Pay Rate	decimal(5,2)	>0.0		Hourly pay rate of a contractor	11
Employee Contract Type	char(2)	('W','E')		Discriminator for contract employee type, WD official(W) or Engineer(E)	W

Employee_Contract_Engineer

Name	Data Type	Constraint	Key	Description	Example Value
ECEmployee ID	int	>0	PK, FK	Engineer for a construction design;Unique identifier of a contractor engineer	504

Employee_Contract_WD_Official

Name	Data Type	Constraint	Key	Description	Example Value
WCEmployee ID	int	>0	PK, FK	Watershed development official conducting the training;Unique identifier of a contractor watershed development official	501

Awareness_Training

Name	Data Type	Constraint	Key	Description	Example Value
Training ID	int	>0	PK	Unique identifier of an awareness training	101
WCEmployee ID	int	>0	FK	Unique identifier of a contractor watershed development official	501
Training Topic	nvarchar(100)			Training topic included in a project	Watershed Awareness Training
Training Hours	int	>0		number of hours a training is conducted	3

House_Owner

Name	Data Type	Constraint	Key	Description	Example Value
Owner ID	int	>0	PK	Unique identifier of an owner	1001
Training ID	int	>0	FK	Unique identifier of an Awareness Training	101
Owner ID	int	>0	FK	Unique identifier of an owner	1003
Owner Name	nvarchar(50)			Name of the owner	Rex Hancock
Owner Contact Number	int	>0		Contact number of the owner	2722221494

House Unit

Name	Data Type	Constraint	Key	Description	Example Value
House Unit ID	int	>0	PK	Unique identifier of a house unit	1
City Project ID	int	>0	FK	Unique identifier of a city project	11111
Contruction Contractor ID	int	>0	FK	Unique identifier of a construction contractor	6008
Design ID	int	>0	FK	Unique identifier of a design	2010
Unit Address	nvarchar(50)			Address of the house unit	8135 Sage Ave. Westford
UCity	nvarchar(20)			City of the house unit	Charlotte
UState	char(2)			State of the house unit	NC

Construction_Design

Name	Data Type	Constraint	Key	Description	Example Value
Design ID	int	>0	PK	Unique identifier of a design	2001
ECEmployee ID	int	>0	FK	Unique identifier of a	504
Start Date	date			Start date of the construction design	3/2/2016
Completion Date	date			Completion Date of the construction design	3/17/2016

Design_Usage_Details

Name	Data Type	Constraint	Key	Description	Example Value
Design Contractor ID	int	>0	PK	Unique identifier of which design is made by which constructor	901
Design ID	int	>0	FK	Unique identifier of a design	2001
Construction Contractor ID	int	>0	FK	Unique identifier of a construction contractor	6005
Used on Date	datetime			Date on which the design is used	3/29/2016

Contractor

Name	Data Type	Constraint	Key	Description	Example Value
Contractor ID	int	>0	PK	Unique identifier of a construction contractor	6001
Contractor Name	nvarchar(50)			Name of the contractor	Donald Watson
Contractor Contact Number	int			Contact number of the contractor	1234567891

SQL Statements to create each table:

1)City Project

```
CREATE TABLE CityProject_T
(CityProjectID int not null CHECK (CityProjectID > 0),
CentralProjectID int not null CHECK (CentralProjectID > 0),
ORegistrationNumber int not null CHECK (ORegistrationNumber > 0),
TrainingID int not null CHECK (TrainingID > 0),
ProjectTitle nvarchar,
ProjectInitiationDate date,
ProjectCompletionDate date,
TotalFunding decimal(10,2) Check (TotalFunding >0)
CONSTRAINT CityProject_PK PRIMARY KEY (CityProject),
CONSTRAINT CityProject_FK1 FOREIGN KEY (CentralProjectID) REFERENCES CentralProject_T
(CentralProjectID),
CONSTRAINT CityProject_FK2 FOREIGN KEY (TrainingID) REFERENCES AwarenessTraining_T
(TrainingID))
```

2)CityProject_Employee_T

```
CREATE TABLE CityProject_Employee_T
(CityProjectID int not null CHECK (CityProjectID > 0),
EmployeeID int not null CHECK (EmployeeID > 0),
CONSTRAINT CityProject_PK PRIMARY KEY (CityProjectID),
CONSTRAINT CityProject_FK1 FOREIGN KEY (CentralProjectID) REFERENCES CentralProject_T
(CentralProjectID),
CONSTRAINT CityProject_FK2 FOREIGN KEY (ContractorID) REFERENCES Contractor_T
(ContractorID),
CONSTRAINT CityProject_FK3 FOREIGN KEY (MPEmployeeID) REFERENCES
Employee_Permanent_Manager_T (MPEmployeeID))
```

3)AwarenessTraining_T

```
create table AwarenessTraining_T
(TrainingID int not null check (TrainingID>0),
TrainingTopic nvarchar(20) not null,
TrainingHours int (check TrainingHours>0),
Constraint AwarenessTraining_PK Primary Key (TrainingID))
```

4)NGO_T

```
create table NGO_T
(ORegistrationNumber int not null check (ORegistrationNumber>0),
OrganisationName nvarchar(20) not null,
OFoundationDate date,
CONSTRAINT NGO_PK primary key (ORegistrationNumber))
```

5) CityProject_EmployeeDetails_T

```
CREATE TABLE CityProject_EmployeeDetails_T
(CityProjectID int not null CHECK (CityProjectID > 0),
EmployeeID int not null CHECK (EmployeeID > 0),
CONSTRAINT CityProject_EmployeeDetails_PK PRIMARY KEY (CityProjectID,EmployeeID),
CONSTRAINT CityProject_EmployeeDetails_FK1 FOREIGN KEY (CityProjectID) REFERENCES
CityProject_T (CityProjectID),
CONSTRAINT CityProject_EmployeeDetails_FK2 FOREIGN KEY (EmployeeID) REFERENCES
Employee_T (EmployeeID))
```

6) Contractor_T

```
CREATE TABLE Contractor_T
(ContractorID int not null CHECK (ContractorID > 0),
ContractorName nvarchar(50),
ContractorContactNumber bigint CHECK (ContractorContactNumber > 0),
CONSTRAINT ConstructionContractor_PK PRIMARY KEY (ContractorID))
```

7) HouseOwner_T

```
CREATE TABLE HouseOwner_T
(OwnerID int not null CHECK (OwnerID > 0),
TrainingID int not null CHECK (TrainingID > 0),
OwnerName nvarchar(50),
OwnerContactNumber bigint CHECK (OwnerContactNumber > 0),
CONSTRAINT ConstructionContractor_PK PRIMARY KEY (ContractorID),
CONSTRAINT Owner_FK FOREIGN KEY (TrainingID) REFERENCES Training_T (TrainingID))
```

8) ConstructionDesign_T

```
CREATE TABLE ConstructionDesign_T
(DesignID int not null CHECK (DesignID > 0),
ECEmployeeID int not null CHECK (ECEmployeeID > 0),
StartDate datetime,
CompletionDate datetime
CONSTRAINT ConstructionDesign_PK PRIMARY KEY (DesignID),
CONSTRAINT ConstructionDesign_FK1 FOREIGN KEY (ECEmployeeID) REFERENCES
Employee_Contract_Engineer_T (ECEmployeeID))
```

9) HouseUnit_T

```
CREATE TABLE HouseUnit_T
(HouseUnitID int not null CHECK (HouseUnitID > 0),
CityProjectID int not null CHECK (CityProjectID > 0),
ContractorID int not null CHECK (ContractorID > 0),
DesignID int not null CHECK (DesignID > 0),
HouseOwnerID int not null CHECK (HouseOwnerID > 0),
UnitAddress nvarchar(50),
UCity nvarchar(20),
UState char(2),
CONSTRAINT HouseUnit_PK PRIMARY KEY (HouseUnitID),
CONSTRAINT HouseUnit_FK1 FOREIGN KEY (CityProjectID) REFERENCES CityProject_T
(CityProjectID),
CONSTRAINT HouseUnit_FK2 FOREIGN KEY (ContractorID) REFERENCES Contractor_T
(ContractorID),
CONSTRAINT HouseUnit_FK3 FOREIGN KEY (DesignID) REFERENCES ConstructionDesign_T
(DesignID),
CONSTRAINT HouseUnit_FK4 FOREIGN KEY (HouseOwnerID) REFERENCES HouseOwner_T
(HouseOwnerID))
```

10) Design_Usage_Details_T

```
CREATE TABLE Design_Usage_Details_T
(DesignContractorID int not null CHECK (DesignContractorID > 0),
DesignID int not null CHECK (DesignID > 0),
ContractorID int not null CHECK (ContractorID > 0),
UsedOnDate date,
CONSTRAINT HouseUnit_PK PRIMARY KEY (DesignContractorID),
CONSTRAINT HouseUnit_FK1 FOREIGN KEY (DesignID) REFERENCES ConstructionDesign_T
(DesignID),
CONSTRAINT HouseUnit_FK2 FOREIGN KEY (ContractorID) REFERENCES Contractor_T
(ContractorID))
```

11) Employee_T

```
Create table Employee_T
(EmployeeID int not null CHECK (EmployeeID >0),
Ename nvarchar(25),
EContactNumber char(10),
EmployeeType char(2) not null CHECK  (EmployeeType IN ('P' , 'C')),
CONSTRAINT Employee_PK PRIMARY KEY (EmployeeID))
```

12) Employee_Permanent_T

```
CREATE TABLE Employee_Permanent_T
(PEmployeeID int not null CHECK (PEmployeeID > 0),
AnnualSalary decimal(7,2) CHECK (AnnualSalary > 0),
EmployeePermanentType char(2) not null CHECK (EmployeePermanentType IN ('M'),
UsedOnDate date,
CONSTRAINT HouseUnit_PK PRIMARY KEY (DesignContractorID),
CONSTRAINT HouseUnit_FK1 FOREIGN KEY (DesignID) REFERENCES ConstructionDesign_T
(DesignID),
CONSTRAINT HouseUnit_FK2 FOREIGN KEY (ContractorID) REFERENCES Contractor_T
(ContractorID))
```

13) Employee_Permanent_Manager_T

```
CREATE TABLE Employee_Permanent_Manager_T
(MPEmployeeID int not null CHECK (MPEmployeeID>0),
CONSTRAINT Employee_Permanent_Manager_PK PRIMARY KEY (MPEmployeeID)
CONSTRAINT Employee_Permanent_Manager_FK FOREIGN KEY (MPEmployeeID)
REFERENCES Employee_Permanent_T (PEmployeeID)
```

14) Employee_Contract_T

```
CREATE TABLE Employee_Contract_T
(CEmployeeID int not null CHECK(CEmployeeID>0),
HourlyPayRate decimal(5, 2) CHECK(HourlyPayRate >0),
EmployeeContractType char(2) not null CHECK  (EmployeeContractType IN ('W' , 'E')),
CONSTRAINT Employee_Contract_PK PRIMARY KEY (CEmployeeID),
CONSTRAINT CONSTRAINT Employee_Contract_PK PRIMARY KEY (CEmployeeID),
CONSTRAINT Employee_Contract_FK1 FOREIGN KEY (CEmployeeID)
REFERENCES Employee_T (EmployeeID))
```

15) Employee_Contract_Engineer_T

```
CREATE TABLE Employee_Contract_Engineer_T
(ECEmployeeID int not null CHECK (ECEmployeeID>0),
CONSTRAINT Employee_Contract_Engineer_PK PRIMARY KEY (ECEmployeeID),
CONSTRAINT Employee_Contract_Engineer_FK FOREIGN KEY (ECEmployeeID)
REFERENCES Employee_Contract_T (CEmployeeID))
```

16) Employee_Contract_WDOfficial_T

```
CREATE TABLE Employee_Contract_WDOFFICIAL_T
(WCEmployeeID int not null CHECK (WCEmployeeID>0),
CONSTRAINT Employee_Contract_WDOFFICIAL_PK PRIMARY KEY (WCEmployeeID),
CONSTRAINT Employee_Contract_WDOFFICIAL_FK FOREIGN KEY (WCEmployeeID)
REFERENCES Employee_Contract_T (CEmployeeID)
```

SQL statements for materialized views and procedures with justification:

For keeping a track of what percentage of the total funding is bounced back to the government of a particular central project.

President(of NGO) View:

```
Create Table PercentBounceBack_View
(CentralProjectID int not null,
ProjectTitle nvarchar(max),
BalanceFunding decimal(7,2),
PercentOfTotalFunding decimal(7,2))


Create procedure RefreshPercentBounceBack_View as
delete from PercentBounceBack_View
insert into PercentBounceBack_View
select CityProject_T.CentralProjectID, CentralProject_T.ProjectTitle,
(CentralProject_T.TotalFunding-derivedtable.totalCityFunding) as balancefunding,
(((CentralProject_T.TotalFunding-
derivedtable.totalCityFunding)/CentralProject_T.TotalFunding)*100) as
PercentOfTotalFunding
from cityproject_t, CentralProject_T,
(select CityProject_T.CentralProjectID, sum(CityProject_T.fundingrequired) as
totalCityFunding
 from CityProject_T, centralproject_t
where CentralProject_T.CentralProjectID=CityProject_T.CentralProjectID
group by CityProject_T.CentralProjectID)as derivedtable
where CentralProject_T.CentralProjectID=CityProject_T.CentralProjectID
and CityProject_T.CentralProjectID=derivedtable.CentralProjectID
group by CityProject_T.CentralProjectID,  CentralProject_T.ProjectTitle,
(CentralProject_T.TotalFunding-derivedtable.totalCityFunding),
(((CentralProject_T.TotalFunding-
derivedtable.totalCityFunding)/CentralProject_T.TotalFunding)*100)


execute RefreshPercentBounceBack_View

select * from PercentBounceBack_View
```

Justification:

This view will help the President to know the president of the NGO about the difference between how much of the total funding has been used in distributing in the city projects as funding required minus the total funding given for a central project as total funding. A percentage of how much of the funding is remaining will also be shown. This is a very important view in term of finance as the remaining funding needs to be returned back to the central government. This amount of funding returned is termed as bounce back funding. It also helps in doing financial analysis on an NGO level about what percentage of the total funding received is used and how much is not.

President(of NGO) View: To keep a track of which design is used for which Central Project and on which date.

```
CREATE TABLE ProjectUsedDesign_View
(ProjectTitle nvarchar(max),
UsedOnDate date,
DesignID int not null)

create procedure RefreshProjectUsedDesign_View as
delete from ProjectUsedDesign_View
insert into ProjectUsedDesign_View
select CentralProject_T.ProjectTitle, Design_Usage_Details_T.UsedOnDate,
ConstructionDesign_T.DesignID
from CentralProject_T, Design_Usage_Details_T, CityProject_T,
Employee_Permanent_Manager_T,
ConstructionDesign_T
where CentralProject_T.CentralProjectID=CityProject_T.CentralProjectID
and CityProject_T.MPEmployeeID=Employee_Permanent_Manager_T.MPEmployeeID
and ConstructionDesign_T.DesignID=Design_Usage_Details_T.DesignID


execute RefreshProjectUsedDesign_View
select * from ProjectUsedDesign_View
```

Justification:

This view is precisely made for a president keeping in mind the importance of design and its importance in a construction which will be undertaken by the contractor. If the designs used are incorrect the whole construction can go in vain. This view will help in enlightening the president of the organisation that which design is used for which project and on which date. This view will also help the president keep an eye and on record the designs used with respect to its used on date, so if in the future somethings doesn’t add up, the president can immediately look up at the records and check which design was used on which date under which central project.

Contractor View: For keeping a track for contractor on which contractor has worked on how many house units and used which design on which date.
Contractor View

```
Create Table ContractorDetails_View
(ContractorName nvarchar(50),
DesignID int,
UsedOnDate date,
NumberOfHouseUnits int,
ProjectTitle nvarchar(max))

create procedure RefreshContractorDetails_View as
delete from ContractorDetails_View
insert into ContractorDetails_View
select Contractor_T.ContractorName,Design_Usage_Details_T.DesignID,
Design_Usage_Details_T.UsedOnDate,
count(HouseUnit_T.HouseUnitID) as NumberOfHouseUnits,
CentralProject_T.ProjectTitle
from Contractor_T, Design_Usage_Details_T, HouseUnit_T, CentralProject_T, CityProject_T,
ConstructionDesign_T
where Contractor_T.ContractorID=Design_Usage_Details_T.ContractorID
and HouseUnit_T.ContractorID=Contractor_T.ContractorID
and CityProject_T.CentralProjectID=CentralProject_T.CentralProjectID
and HouseUnit_T.DesignID=ConstructionDesign_T.DesignID
group by Contractor_T.ContractorName,Design_Usage_Details_T.DesignID,
Design_Usage_Details_T.UsedOnDate, CentralProject_T.ProjectTitle

execute RefreshContractorDetails_View
select * from ContractorDetails_View
```

Justification:

This view is important for the contractor as it will help him to have a look at how many units he has totally worked on under which project. Additionally, this view will allow the contractor to check under which central project he has used which design. Contractor needs the track of all his work, as it will help him to generate a bill which will be handled manually. The bill generated will be based on the number of house units he has worked on.

Contractor View: Which contractor has worked in which state on how many city proposals having city funding greater than $10000

```
Create Table ContractorWorkDone_View
(ContractorName nvarchar(50),
Cstate char(2),
FundingRequired decimal(8,2),
TotalCityProjects int)

create procedure RefreshContractorWorkDone_View as
delete from ContractorWorkDone_View
insert into ContractorWorkDone_View
select Contractor_T.ContractorName, CityProject_T.Cstate,CityProject_T.FundingRequired,
count(CityProject_T.CityProjectID) as TotalCityProjects
from Contractor_T,CityProject_T, HouseUnit_T
where Contractor_T.ContractorID=CityProject_T.ContractorID
and HouseUnit_T.ContractorID=Contractor_T.ContractorID
and CityProject_T.CentralProjectID=CityProject_T.CentralProjectID
group by Contractor_T.ContractorName, CityProject_T.Cstate,
CityProject_T.FundingRequired
having CityProject_T.FundingRequired>10000

execute RefreshContractorWorkDone_View
select * from ContractorWorkDone_View
```

Justification:

This view is important and comes in use for the contractors when there is a manual bidding at the beginning of every new big project (in terms of funding and states)

The work done in different states in a country along with the total number of city projects worked on will be some of the deciding. For big projects only the contractors who have previously worked on more than $10000 of funding per city proposal can participate in this bid. The reason for this is only the contractors who have experience in working for bigger scale projects can handle such incoming big projects.

SQL statements for database triggers with justifications:

Trigger created for the log of change in funding required for a particular city project. Intentionally/Unintentionally.

```
CREATE TABLE FundingUpdates_Log
(CityProjectID int,
CityName nvarchar(20),
OldFundingRequired decimal(7,2),
NewFundingRequired decimal(7,2),
UpdateDate datetime)


create trigger FundingUpdates on CityProject_T
for update
as
if update(FundingRequired)begin
insert into FundingUpdates_Log (CityProjectid, CityName, oldfundingrequired,
newfundingrequired, updatedate)
select inserted.CityProjectID, inserted.CityName,
deleted.FundingRequired, inserted.FundingRequired, GETDATE()
from inserted,
deleted where inserted.CityProjectID = deleted.CityProjectID
end


update CityProject_t set FundingRequired = 36000 where CityProjectid = 11111
```

Justification:

This trigger can be considered as the most important trigger in the whole system. City Project's total funding can change over time if required. The funding required is always the estimated funding for a city project. If there is any change in the funding required or someone tries to intentionally change this sensitive data it should be stored in the database. This trigger will help in knowing for the CityProjectID of 11111 what was the old funding required and what will be the new funding required. Any intentional illegal changes done to this attribute will also be stored in the database making the DBA aware of intrusive behaviour of a person in the database.