**Project team 24 - <Disaster Report System>**

*Team Members:*

Anil Sadasivuni

Chaitanya Deshpande

Rithesh Shetty

Sharath Kumar Reddy Lenkala

## ***PROJECT PROPOSAL***

**Content**:

This database system keeps track of disasters, things affected by the disasters and post disaster recovery statistics.

**Scope:**

This database system allows user to understand the magnitude of particular disaster and its adverse effects on the environment, user can also track the recovery of the individuals and infrastructure.

**Objective:**

1. To track Disaster name, Disaster type, Disaster affected area
2. To track Emergency evacuation details
3. To track general issues type occurred to the public during disaster(food, transport, power)
4. To track fatality and injured people due to disaster
5. To track affected person name and details of the person
6. To track affected infrastructure, property
7. To track services provided to the affected people during disaster
8. To track funding provided by various agencies during disaster
9. To track insurance claim by the people and effected industries
10. To track funds provided by government and other agencies for recovery

# **Project: Sprint 0 - Environment setup and high level requirements & conceptual design**

## ***PROJECT ENVIRONMENT***

For project development, we will be using 2 software’s namely JAVA and AWS(RDS) + MySQL. We will be using Java to build the frontend i.e. user interface of the application and for the data storage and manipulation purposes we will be using AWS (RDS) + MySQL.

## ***HIGH LEVEL REQUIREMENTS***

### **Initial user roles**

|  |  |
| --- | --- |
| **User Role** | **Description** |
| Database Administrator | Database administrator have privileges like insert, update and delete data |
| Special Privilege User | Users who are given privileges on demand to view any kind of information in the database like President of America |
| Reporter | User with privileges to view and report any kind of information from disaster report system |
| Insurance Analyst | User who have the privileges to view insurance claims information in the database system |
| Medical Analyst | User who have the privileges to view the medical information of the effected people in the database |
| Funding Administrator | Government appointed funding administrator who tracks the funding during or after the disaster |

### **Initial user story descriptions**

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| US1 | As a Database Administrator, I want to access any kind of information in the database for inserting, updating or deleting the information |
| US2 | As a Special Privilege User, I want to have access to view any kind of information in the database so that I can analyze the disaster related information |
| US3 | As a Reporter, I should be able to present Emergency Service, adverse effects, Funding, Casualty and Insurance details of any disaster to the government on demand |
| US4 | As an Insurance analyst, I should be able to access details of any insurance related information of a disaster in the database |
| US5 | As a Medical analyst, I should be able to access details of any casualty related information of a disaster in the database |
| US6 | As a Funding administrator, I should be able to track and analyze funding related information of a disaster in the database |

## ***HIGH LEVEL CONCEPTUAL DESIGN***

Entities:

1. User

2. Role

3. Disaster

4. Emergency Service

5. Adverse effect

6. Funding

7. Casualty

8. Insurance

Relationships:

1. Disaster leading to Casualty

2. Disaster leading to adverse effect

3. Funding for Disaster

4. Emergency Service offered for adverse effect

5. Insurance claim for adverse effect

6. Insurance claim for Casualty

**Project: Sprint 1 - Database design and implementation**

## ***REQUIREMENTS***

User Stories in decreasing order of priority

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| US1 | As a Database Administrator I want to add role with read/write access of a user with privileges. |
| US2 | As a User i should be able to login and logout of the application so that i can use the application. |
| US3 | As a Database Administrator I want to add name, type, city, state, country, start date, start time, end date, end time and intensity details of a disaster so that end users can use the information on demand. |
| US4 | As a Database Administrator I want to add name, service provider and description of emergency services provided for a disaster, so that end user can use the information on demand. |
| US5 | As a Database Administrator I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US6 | As a user with the role of ‘Emergency Service Provider’, I want to add name, service provider and description of emergency services provided for a disaster, so that emergency services provided are tracked. |
| US7 | As a user with the role of ‘Adverse Effect Reporter’, I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US8 | As a reporter I should be able to give name of a disaster as input and view details of that disaster so that I can report it to the government/authorities. |
| US9 | As a reporter I should be able to give city,state and country as input and view details of the disaster affecting that place so that I can report it to the government/authorities. |
| US10 | As a reporter I should be able to enter name of disaster and view details of emergency services provided for that disaster, so that I can report to the government/authorities. |
| US11 | As a reporter I should be able to enter name of disaster and view adverse effects of that disaster, so that I can report to the government/authorities. |
| US12 | As a Database Administrator I want to update insurance claimed so that reporter can use the information. |
| US13 | As a Database Administrator I want to update casualty caused during disaster so that reporter can use the information |
| US14 | As a Database Administrator I want to update the funding for the disaster, so that reporter can use the information |
| US15 | As a reporter I want to view insurance claimed, so that I can report to the government/authorities. |
| US16 | As a reporter I want to view casualty due to disaster, so that I can report to the government/authorities. |
| US17 | As a reporter I want to view funding for disaster, so that I can report to the government/authorities. |
| US18 | As a Insurance Analyst i want to update claimed insurance details so that reporter can use the information. |
| US19 | As a Medical Analyst I want to update the medical services provides during disaster, so that reporter can report to the government |
| US20 | As a Funding Analyst i want to update details of funding for disaster so that reporter can use the information. |

## ***CONCEPTUAL DESIGN***

Entity: **Disaster**

Attributes:

name

type

place [composite]

city,

state,

country

start\_date

start\_time

intensity

end\_date

end\_time

Primary key Justification: Every disaster will have a unique name. so combination of disaster name and place can uniquely identify each and every disaster in the system. Intensity will be defined with respect to disaster type on the respective scale like (Earthquake will be measured in Richter scale, Hurricane will be measured in Saffir-Simpson scale and Volcano will be measured in Volcanic Explosivity Index).

Entity: **EmergencyService**

Attributes:

name

services\_provider

description

Primary key justification : Every service provided by a service provider will have a unique name. Combination of service\_provider and name can uniquely identify emergency services in the system.

Entity: **AdverseEffect**

Attributes:

name

preventive

Primary key justification : From the name of adverse effect we can uniquely identify AdverseEffect.Preventive attribute will define whether the adverse effect was preventive (food outage, water outage) or non preventive (acid rain, landslide).

Entity: **User**

Attributes:

user\_name

password

Primary key justification : Each user will have unique user\_name, so from use\_name we can uniquely identify the User.

Entity: **Role**

Attributes:

role\_name

privilege

Primary key Justification : As every role will have a unique role\_name, role\_name can uniquely identify roles.

**Relationships:**

Relationship: **Disaster** leads to **AdverseEffect**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**AdversEffect** has partial participation

Note : Assuming every disaster has adverse effect.

Relationship: **EmergencyService** offered for **Disaster**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**EmergencyService** has partial participation

Note : Assuming every disaster requires emergency service.

Relationship: **User** has a **Role**

Cardinality:

Many to One

Participation:

**User** has total participation

**Role** has partial participation

Note : Every User will have only one role in the system.

## ***LOGICAL DESIGN***

Table: **Disaster**

Columns:

disaster\_id

name

type\_id[foreign key; references type\_id of **DisasterType** table]

city

country

start\_date

start\_time

end\_date

end\_time

intensity

Justification : From the disaster\_id we can uniquely identify the disaster. Foreign Key approach has been followed here as each disaster\_id is related to only one type\_id of DisasterType.

Table: **DisasterType**

Columns:

type\_id

disaster\_type

Justification : We created separate table for disaster type. We are covering 5 types of disasters which are Earthquake, Tsunami, Volcano, Nuclear-Explosion and Hurricane.

Table: **EmergencyService**

Columns:

emg\_svr\_id

name

service\_provider

description

Justification : From emg\_svr\_id we can uniquely identify the emergency service.

Table : **AdverseEffect**

Columns:

adverse\_id

name

preventive

Justification: From adverse\_id we can uniquely identify the adverse effect.

Table: **Role**

Columns:

role\_id

role\_name

privilege

Table: **User**

Columns:

user\_name

password

role\_id [foreign key; references **role\_id** of **Role** table]

Justification : As each user will have specific role\_id of role table, foreign key approach has been followed.

Table : **DisAdvEff**

Columns:

disaster\_id [foreign key; references disaster\_id of **Disaster** table]

adverse\_id [foreign key; references adverse\_id of **AdverseEffect** table]

Justification : As a disaster\_id can be linked to multiple adverse\_id and a adverse\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

Table : **DisEmgSvr**

Columns:

disaster\_id [foreign key; references disaster\_id of **Disaster** table]

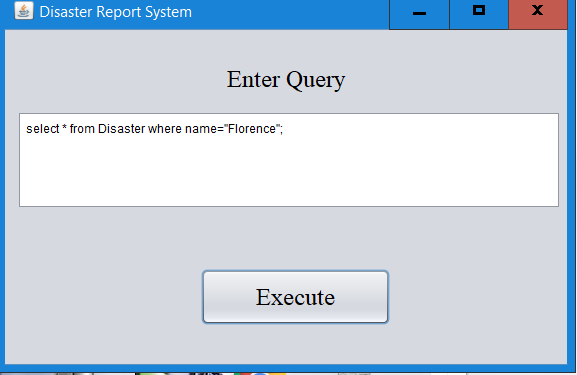
emg\_svr\_id [foreign key; references emg\_svr\_id of **EmergencyService** table]

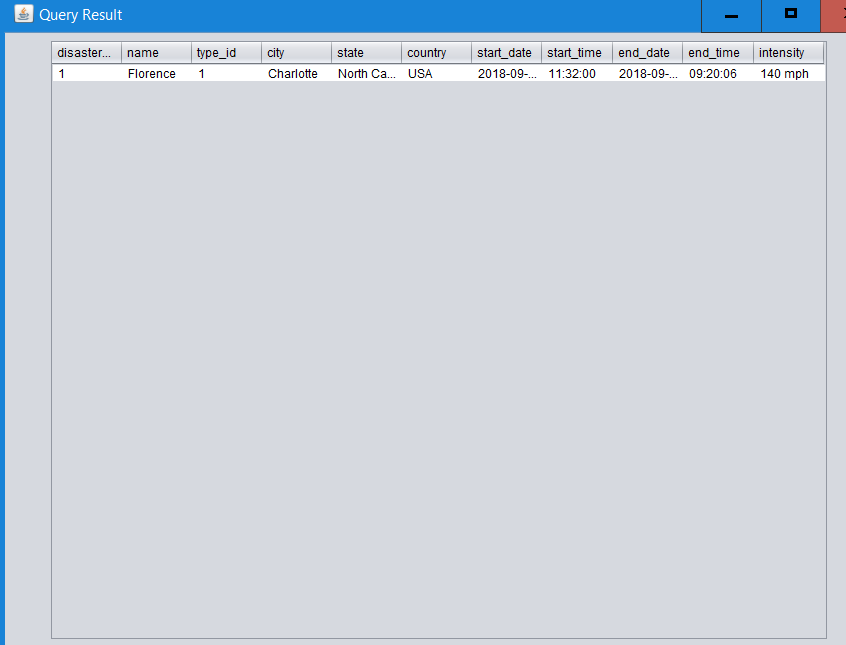
pre\_post\_type

Justification : As a disaster\_id can be linked to multiple emg\_svr\_id and a emg\_svr\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

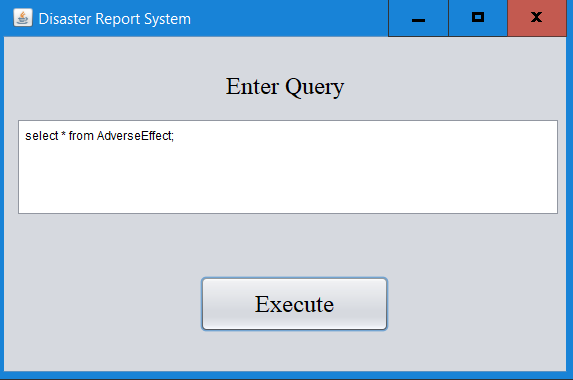
Query and Screenshots:

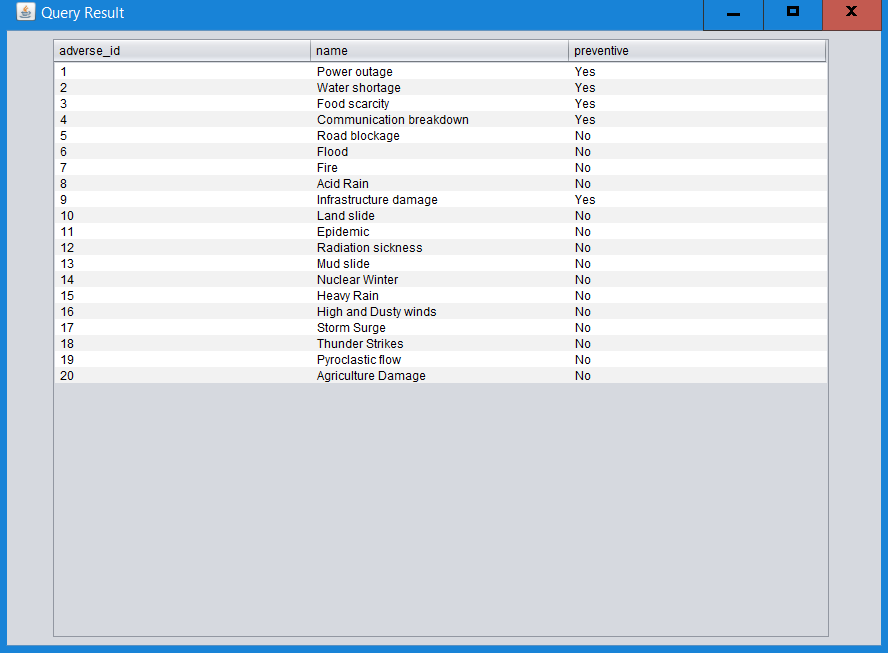
Query 1:



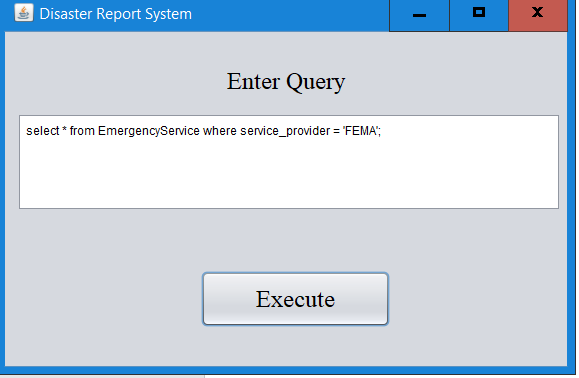


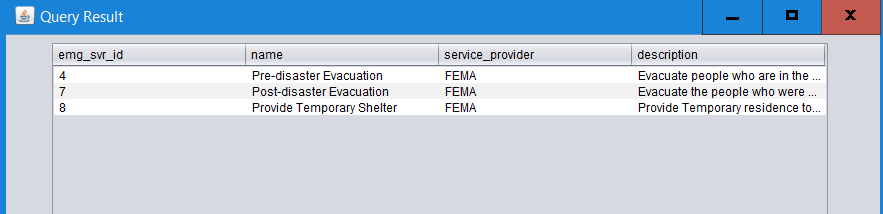
Query 2:





Query 3:





# **Project: Sprint 2 - Database design and implementation**

## ***REQUIREMENTS***

User Stories in decreasing order of priority

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| US1 | As a Database Administrator I want to add role with read/write access of a user with privileges. |
| US2 | As a User i should be able to login and logout of the application so that i can use the application. |
| US3 | As a Database Administrator I want to add name, type, city, state, country, start date, start time, end date, end time and intensity details of a disaster so that end users can use the information on demand. |
| US4 | As a Database Administrator I want to add name, service provider and description of emergency services provided for a disaster, so that end user can use the information on demand. |
| US5 | As a Database Administrator I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US6 | As a user with the role of ‘Emergency Service Provider’, I want to add name, service provider and description of emergency services provided for a disaster, so that emergency services provided are tracked. |
| US7 | As a user with the role of ‘Adverse Effect Reporter’, I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US8 | As a reporter I should be able to give name of a disaster as input and view details of that disaster so that I can report it to the government/authorities. |
| US9 | As a reporter I should be able to give city,state and country as input and view details of the disaster affecting that place so that I can report it to the government/authorities. |
| US10 | As a reporter I should be able to enter name of disaster and view details of emergency services provided for that disaster, so that I can report to the government/authorities. |
| US11 | As a reporter I should be able to enter name of disaster and view adverse effects of that disaster, so that I can report to the government/authorities |
| US12 | As a Database Administrator I want to add name, address, ssn, emergency contact and type of injury about the people who are injured during the disaster, so that end user can use the information on demand |
| US13 | As a Database Administrator I want to add name, address, ssn, emergency contact , death date and type of fatality about the people who lost their life during the disaster, so that end user can use the information on demand |
| US14 | As a user with the role of ‘Disaster Casualty Reporter’ I want to add name, address, ssn, emergency contact, type of injury about the people who are injured during the disaster, so that end user can use the information on demand |
| US15 | As a user with the role of ‘Disaster Casualty Reporter’ I want to add name, address, ssn, emergency contact, type of fatality, death date of people who lost their life during the disaster, so that end user can use the information on demand |
| US16 | As a reporter I want to enter disaster name and view details of people injured during the disaster so that I can report to the government/authorities |
| US17 | As a reporter I want to enter disaster name and view details of people died during the disaster so that I can report to the government/authorities |
| US18 | As a Database Administrator I want to add type of health-care management services provided (e.g. environmental health control, nutrition management, mental health control etc.) during the disaster, so that end user can use the information on demand |
| US19 | As a Medical Analyst I want to update type of health-care management services provided (e.g. environmental health control, nutrition management, mental health control etc.) during the disaster, so that I can use the information for analysis |
| US20 | As a Reporter, I want to enter disaster name and view health-care management services provided during a disaster so that I can report to the authorities/government. |
| US21 | As a Funding Analyst i want to update details of funding for disaster so that reporter can use the information. |
| US22 | As a Database Administrator I want to update insurance claimed so that reporter can use the information. |
| US23 | As a reporter I want to view insurance claimed, so that I can report to the government/authorities. |
| US24 | As a Insurance Analyst i want to update claimed insurance details so that reporter can use the information. |
| US25 | As a Database Administrator I want to update the funding for the disaster, so that reporter can use the information |
| US26 | As a reporter I want to view funding for disaster, so that I can report to the government/authorities. |

## 

## 

## ***CONCEPTUAL DESIGN***

Entity: **Disaster**

Attributes:

name

type

place [composite]

city,

state,

country

start\_date

start\_time

intensity

end\_date

end\_time

Primary key Justification: Every disaster will have a unique name. so combination of disaster name and place can uniquely identify each and every disaster in the system. Intensity will be defined with respect to disaster type on the respective scale like (Earthquake will be measured in Richter scale, Hurricane will be measured in Saffir-Simpson scale and Volcano will be measured in Volcanic Explosivity Index).

Entity: **EmergencyService**

Attributes:

name

services\_provider

description

Primary key justification : Every service provided by a service provider will have a unique name. Combination of service\_provider and name can uniquely identify emergency services in the system.

Entity: **AdverseEffect**

Attributes:

name

preventive

Primary key justification : From the name of adverse effect we can uniquely identify AdverseEffect.Preventive attribute will define whether the adverse effect was preventive (food outage, water outage) or non preventive (acid rain, landslide).

Entity: **User**

Attributes:

user\_name

password

Primary key justification : Each user will have unique user\_name, so from use\_name we can uniquely identify the User.

Entity: **Role**

Attributes:

role\_name

privilege

Primary key Justification : As every role will have a unique role\_name, role\_name can uniquely identify roles.

Entity: **Fatality**

Attributes:

name [composite]

ssn

address [composite]

apartment number,

street number,

country,

zip\_code

emergency\_contact

cause\_of\_death

date\_of\_death

Primary key justification: Every person will have a unique ssn so that they can be uniquely identified.

Entity: **Injured**

Attributes:

name [composite]

ssn

address [composite]

apartment number,

street number,

country,

zip\_code

emergency\_contact

type\_of\_injury

Primary key justification: Every person will have a unique ssn so that they can be uniquely identified.

Entity: **HealthCareManagementService**

Attributes:

type\_of\_service

Primary key justification: Every row will have a unique type\_of\_service which is name of the service provided.

**Relationships:**

Relationship: **Disaster** leads to **AdverseEffect**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**AdversEffect** has partial participation

Note : Assuming every disaster has adverse effect.

Relationship: **EmergencyService** offered for **Disaster**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**EmergencyService** has partial participation

Note : Assuming every disaster requires emergency service.

Relationship: **User** has a **Role**

Cardinality:

Many to One

Participation:

**User** has total participation

**Role** has partial participation

Note : Every User will have only one role in the system.

Relationship: **Fatality** due to **Disaster**

Cardinality:

Many to One

Participation:

**Fatality** has total participation

**Disaster** has partial participation

Relationship: **Injured** due to **Disaster**

Cardinality:

Many to Many

Participation:

**Injured** has total participation

**Disaster** has partial participation

Relationship: **HealthCareManagementService** provided during **Disaster**

Cardinality:

Many to Many

Participation:

**HealthCareManagementService** has partial participation

**Disaster** has partial participation

## ***LOGICAL DESIGN***

Table: **Disaster**

Columns:

disaster\_id

name

type\_id[foreign key; references **type\_id** of **DisasterType** table]

intensity\_id[foreign key; references **intensity\_id** of **DisasterIntensity** table]

city

country

start\_date

start\_time

end\_date

end\_time

Primary Key Justification: From the disaster\_id we can uniquely identify the disaster.

Foreign Key approach Justification: Foreign Key approach has been followed here as each disaster\_id is related to only one type\_id of DisasterType. Foreign key approach has been followed for adding intensity\_id column which refers to DisasterIntensitytable as each disaster will refer to any one column value of the DisasterIntensitytable.

Normalization form: The above table is in the 2nd normal form since the attribute city is functionally dependent on the country. We are leaving the table in 2nd normal since the new table for the city and country will add a new level of abstraction which is not required at this stage in our database.

Table: **DisasterIntensity**

Columns:

intensity\_id

intensity\_range

Primary Key Justification: Every DisasterIntensitycan be uniquely identified from the intensity\_id. Seperate table has been created to relate each disaster to a intensity type.

Normalization form: The above table is in 4th normal form.

Table: **DisasterType**

Columns:

type\_id

disaster\_type

Primary Key Justification : We created separate table for disaster type. We are covering 5 types of disasters which are Earthquake, Tsunami, Volcano, Nuclear-Explosion and Hurricane.

Normalization form: The above table is in 4th normal form.

Table: **Fatality**

Columns:

ssn

first\_name

last\_name

address\_line\_1

city

zip\_code

country

emergency\_contact

date\_of\_death

death\_type\_id[foreign Key:references **death\_type\_id** of **DeathType**]

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

Primary Key Justification: Each fatalitycan be uniquely identified with the help of ssn.

Foreign Key approach Justification: Foreign key approach has been followed here for death\_type\_id as each fatality corresponds to one death\_type\_id in DeathType**.** Foreign key approach has been followed here for disaster\_id as each fatality corresponds to a single disaster in Disaster.

Normalization form: The above table is in 2nd Normal Form since the address\_line\_1, city, zip\_code and country are interdependent on each other and separating this columns to a different table will not be relevant for the database need.

Table: **DeathType**

Columns:

death\_type\_id

cause\_of\_death

Primary Key Justification: EveryDeathType rowwill have a unique death\_type\_id.

Normalization form: The above table is in 4th Normal Form.

Table: **Injured**

Attributes:

ssn

first\_name

last\_name

address\_line\_1

city

country

zip\_code

emergency\_contact

Primary Key Justification: Injured row can be uniquely identified with ssn.

Normalization form: The above table is in 2nd Normal Form since the address\_line\_1, city, zip\_code and country are interdependent on each other and separating this columns to a different table will not be relevant for the database need.

Table: **InjuredType**

Columns:

injury\_type\_id

type\_of\_injury

Primary Key Justification: Every **InjuredType** row will have a unique injury\_type\_id.

Normalization: The above table is in 4th normal form.

Table: **HealthCareManagementService**

Columns:

service\_id

type\_of\_service

Primary Key Justification: Every service provided will have a unique service\_id.

Normalization: The above table is in 4th normal form.

Table: **EmergencyService**

Columns:

emg\_svr\_id

name

service\_provider

description

Primary Key Justification : From emg\_svr\_id we can uniquely identify the **EmergencyService**.

Normalization: The above table is in 4th normal form.

Table : **AdverseEffect**

Columns:

adverse\_id

name

preventive

Justification: From adverse\_id we can uniquely identify the **AdverseEffect**.

Normalization:The above table is in 4th normal form.

Table: **Role**

Columns:

role\_id

role\_name

privilege

Justification:From role\_id we can uniquely identify the **Role**.

Normalization: The above table is in 4th normal form.

Table: **User**

Columns:

user\_name

password

role\_id [foreign key; references **role\_id** of **Role** table]

Justification : As each **User** will have specific role\_id of role table, foreign key approach has been followed. Every **User** will have unique user\_name.

Normalization: The above table is in 4th normal form.

Table : **DisAdvEff**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

adverse\_id [foreign key; references **adverse\_id** of **AdverseEffect** table]

Foreign Key Justification : As a disaster\_id can be linked to multiple adverse\_id and a adverse\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

Normalization:The above table is in 4th normal form.

Table : **DisEmgSvr**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

emg\_svr\_id [foreign key; references **emg\_svr\_id** of **EmergencyService** table]

pre

post

Justification : As a disaster\_id can be linked to multiple emg\_svr\_id and a emg\_svr\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

Normalization: The above table is in 4th normal form.

Table: **DisasterInjured**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

ssn[foreign key; references **ssn** of **Injured** table]

injury\_type\_id[foreign key; references **injury\_type\_id** of **InjuryType** table]

Foreign Key Justification: Cross reference approach has been followed to establish many to many relationship betweenInjuredand DisasterInjuredrelationship. Here it is assumed that a person can be affected in multiple disasters.

Normalization: The above table is in 4th normal form.

Table: **HealthCareMngmtSvcDis**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

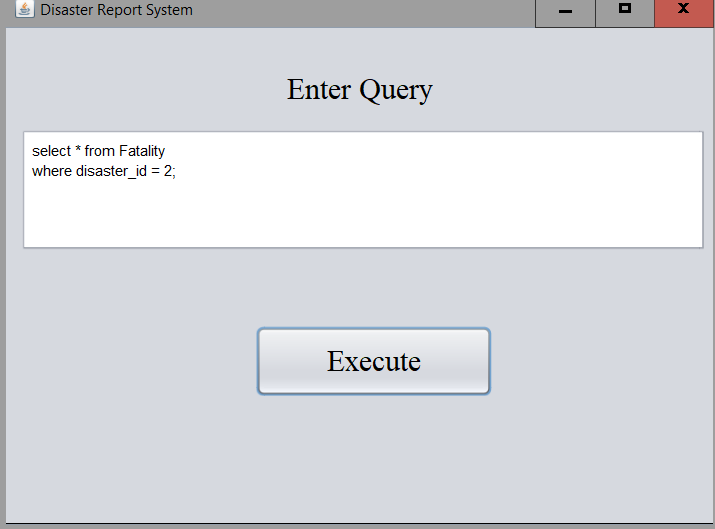
service\_id [foreign key; references **service\_id** of **HealthCareManagementService** table]

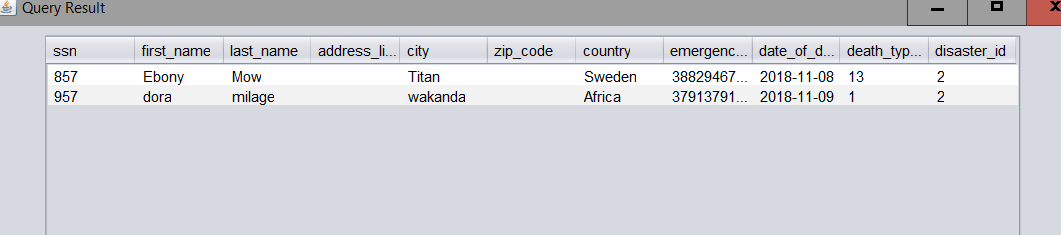
Foreign key Justification: cross reference approach has been followed to establish many to many relationship between Disasterand HealthCareManagementService

Normalization: The above table is in 4th normal form.

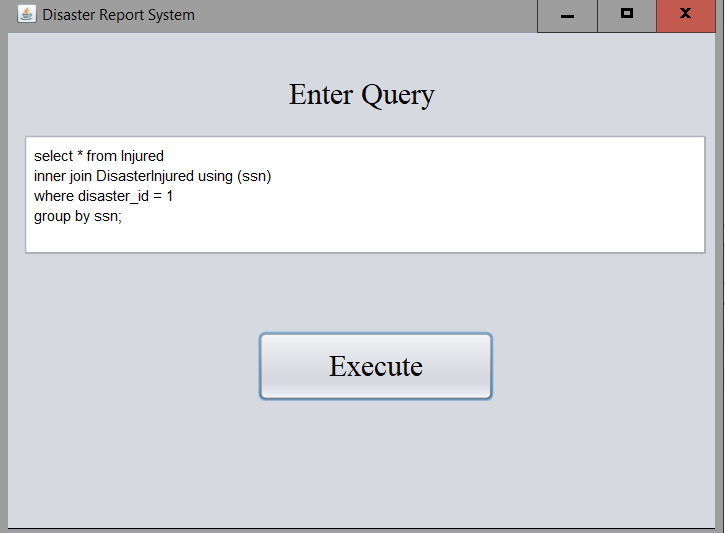
**Query and Screenshots:**

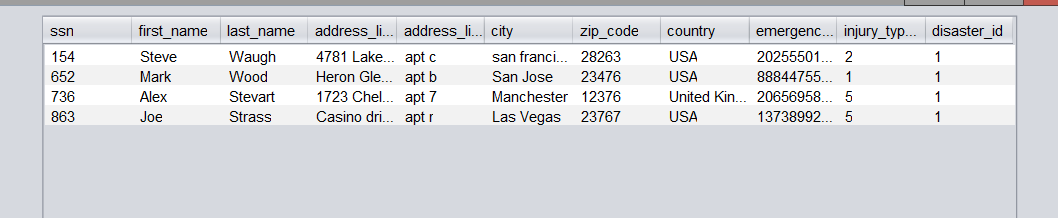
Query 1



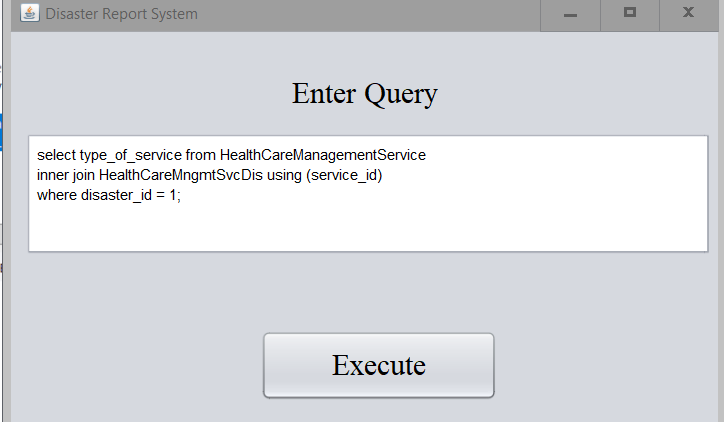


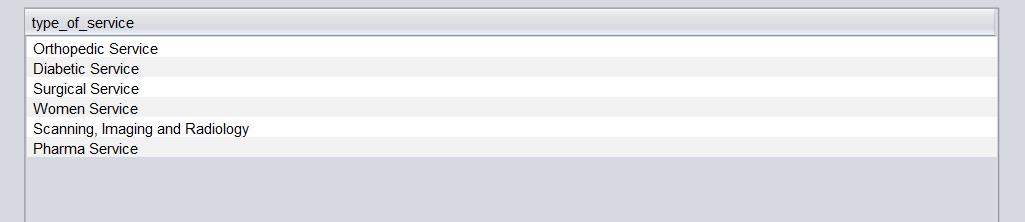
Query 2:





Query 3:





# **Project: Sprint 3 - Database design and implementation**

## ***REQUIREMENTS***

User Stories in decreasing order of priority

|  |  |
| --- | --- |
| **Story ID** | **Story description** |
| US1 | As a Database Administrator I want to add role with read/write access of a user with privileges. |
| US2 | As a User i should be able to login and logout of the application so that i can use the application. |
| US3 | As a Database Administrator I want to add name, type, city, state, country, start date, start time, end date, end time and intensity details of a disaster so that end users can use the information on demand. |
| US4 | As a Database Administrator I want to add name, service provider and description of emergency services provided for a disaster, so that end user can use the information on demand. |
| US5 | As a Database Administrator I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US6 | As a user with the role of ‘Emergency Service Provider’, I want to add name, service provider and description of emergency services provided for a disaster, so that emergency services provided are tracked. |
| US7 | As a user with the role of ‘Adverse Effect Reporter’, I want to add name and prevention possibility of adverse effects caused by a disaster, so that end user can use the information on demand. |
| US8 | As a reporter I should be able to give name of a disaster as input and view details of that disaster so that I can report it to the government/authorities. |
| US9 | As a reporter I should be able to give city,state and country as input and view details of the disaster affecting that place so that I can report it to the government/authorities. |
| US10 | As a reporter I should be able to enter name of disaster and view details of emergency services provided for that disaster, so that I can report to the government/authorities. |
| US11 | As a reporter I should be able to enter name of disaster and view adverse effects of that disaster, so that I can report to the government/authorities |
| US12 | As a Database Administrator I want to add name, address, ssn, emergency contact and type of injury about the people who are injured during the disaster, so that end user can use the information on demand |
| US13 | As a Database Administrator I want to add name, address, ssn, emergency contact , death date and type of fatality about the people who lost their life during the disaster, so that end user can use the information on demand |
| US14 | As a user with the role of ‘Disaster Casualty Reporter’ I want to add name, address, ssn, emergency contact, type of injury about the people who are injured during the disaster, so that end user can use the information on demand |
| US15 | As a user with the role of ‘Disaster Casualty Reporter’ I want to add name, address, ssn, emergency contact, type of fatality, death date of people who lost their life during the disaster, so that end user can use the information on demand |
| US16 | As a reporter I want to enter disaster name and view details of people injured during the disaster so that I can report to the government/authorities |
| US17 | As a reporter I want to enter disaster name and view details of people died during the disaster so that I can report to the government/authorities |
| US18 | As a Database Administrator I want to add type of health-care management services provided (e.g. environmental health control, nutrition management, mental health control etc.) during the disaster, so that end user can use the information on demand |
| US19 | As a Medical Analyst I want to update type of health-care management services provided (e.g. environmental health control, nutrition management, mental health control etc.) during the disaster, so that I can use the information for analysis |
| US20 | As a Reporter, I want to enter disaster name and view health-care management services provided during a disaster so that I can report to the authorities/government. |
| US21 | As a Funding Analyst, I want to update details of funding like amount and funder name for disaster so that information can be used for analysis. |
| US22 | As a Database Administrator, I want to update details of funding like amount and funder name for disaster, so that end user can use the information |
| US23 | As a reporter I want to view funding amount for disaster, so that I can report to the government/authorities. |

## 

## ***CONCEPTUAL DESIGN***

Entity: **Disaster**

Attributes:

name

type

place [composite]

city,

state,

country

start\_date

start\_time

intensity

end\_date

end\_time

Primary key Justification: Every disaster will have a unique name. so combination of disaster name and place can uniquely identify each and every disaster in the system. Intensity will be defined with respect to disaster type on the respective scale like (Earthquake will be measured in Richter scale, Hurricane will be measured in Saffir-Simpson scale and Volcano will be measured in Volcanic Explosivity Index).

Entity: **EmergencyService**

Attributes:

name

services\_provider

description

Primary key justification : Every service provided by a service provider will have a unique name. Combination of service\_provider and name can uniquely identify emergency services in the system.

Entity: **AdverseEffect**

Attributes:

name

preventive

Primary key justification : From the name of adverse effect we can uniquely identify AdverseEffect.Preventive attribute will define whether the adverse effect was preventive (food outage, water outage) or non preventive (acid rain, landslide).

Entity: **User**

Attributes:

user\_name

password

Primary key justification : Each user will have unique user\_name, so from use\_name we can uniquely identify the User.

Entity: **Role**

Attributes:

role\_name

privilege

Primary key Justification : As every role will have a unique role\_name, role\_name can uniquely identify roles.

Entity: **Fatality**

Attributes:

name [composite]

ssn

address [composite]

apartment number,

street number,

country,

zip\_code

emergency\_contact

cause\_of\_death

date\_of\_death

Primary key justification: Every person will have a unique ssn so that they can be uniquely identified.

Entity: **Injured**

Attributes:

name [composite]

ssn

address [composite]

apartment number,

street number,

country,

zip\_code

emergency\_contact

type\_of\_injury

Primary key justification: Every person will have a unique ssn so that they can be uniquely identified.

Entity: **HealthCareManagementService**

Attributes:

type\_of\_service

Primary key justification: Every row will have a unique type\_of\_service which is name of the service provided.

Entity: **Funding**

Attributes:

funder\_name

amount(in USD)

Primary key justification: Every row will have a unique funder\_name

**Relationships:**

Relationship: **Disaster** leads to **AdverseEffect**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**AdversEffect** has partial participation

Note : Assuming every disaster has adverse effect.

Relationship: **EmergencyService** offered for **Disaster**

Cardinality:

Many to many

Participation:

**Disaster** has total participation

**EmergencyService** has partial participation

Note : Assuming every disaster requires emergency service.

Relationship: **User** has a **Role**

Cardinality:

Many to One

Participation:

**User** has total participation

**Role** has partial participation

Note : Every User will have only one role in the system.

Relationship: **Fatality** due to **Disaster**

Cardinality:

Many to One

Participation:

**Fatality** has total participation

**Disaster** has partial participation

Relationship: **Injured** due to **Disaster**

Cardinality:

Many to Many

Participation:

**Injured** has total participation

**Disaster** has partial participation

Relationship: **HealthCareManagementService** provided during **Disaster**

Cardinality:

Many to Many

Participation:

**HealthCareManagementService** has partial participation

**Disaster** has partial participation

Relationship: **Funding** provided for **Disaster**

Cardinality:

Many to Many

Participation:

**Funding** has total participation

**Disaster** has partial participation

## ***LOGICAL DESIGN***

Table: **Disaster**

Columns:

disaster\_id

name

type\_id[foreign key; references **type\_id** of **DisasterType** table]

intensity\_id[foreign key; references **intensity\_id** of **DisasterIntensity** table]

city

country

start\_date

start\_time

end\_date

end\_time

Primary Key Justification: From the disaster\_id we can uniquely identify the disaster.

Foreign Key approach Justification: Foreign Key approach has been followed here as each disaster\_id is related to only one type\_id of DisasterType. Foreign key approach has been followed for adding intensity\_id column which refers to DisasterIntensitytable as each disaster will refer to any one column value of the DisasterIntensitytable.

Normalization form: The above table is in the 2nd normal form since the attribute city is functionally dependent on the country. We are leaving the table in 2nd normal since the new table for the city and country will add a new level of abstraction which is not required at this stage in our database.

Indexes:

Index: clustered

Columns: disaster\_id

Justification: As disaster\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: type\_id

Justification: As type\_id is a foreign key, database by default considers it as a non clustered index.

Index: non clustered

Columns: intensity\_id

Justification: As intensity\_id is a foreign key, database by default considers it as a non clustered index.

Index: non clustered

Columns: name

Justification: In most of the queries the search will be based on the name of the disaster, so name is considered as non clustered index.

Table: **DisasterIntensity**

Columns:

intensity\_id

intensity\_range

Primary Key Justification: Every DisasterIntensitycan be uniquely identified from the intensity\_id. Seperate table has been created to relate each disaster to a intensity type.

Indexes:

Index: clustered

Columns: intensity\_id

Justification: As intensity\_id is a primary key, database by default considers it as a clustered index.

Normalization form: The above table is in 4th normal form.

Table: **DisasterType**

Columns:

type\_id

disaster\_type

Primary Key Justification : We created separate table for disaster type. We are covering 5 types of disasters which are Earthquake, Tsunami, Volcano, Nuclear-Explosion and Hurricane.

Normalization form: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: type\_id

Justification: As type\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: disaster\_type

Justification: In most of the queries the search will be based on the disaster\_type of the disaster, so disaster\_type is considered as non clustered index.

Table: **Fatality**

Columns:

ssn

first\_name

last\_name

address\_line\_1

city

zip\_code

country

emergency\_contact

date\_of\_death

death\_type\_id[foreign Key:references **death\_type\_id** of **DeathType**]

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

Primary Key Justification: Each fatalitycan be uniquely identified with the help of ssn.

Foreign Key approach Justification: Foreign key approach has been followed here for death\_type\_id as each fatality corresponds to one death\_type\_id in DeathType**.** Foreign key approach has been followed here for disaster\_id as each fatality corresponds to a single disaster in Disaster.

Normalization form: The above table is in 2nd Normal Form since the address\_line\_1, city, zip\_code and country are interdependent on each other and separating this columns to a different table will not be relevant for the database need.

Indexes:

Index: clustered

Columns: ssn

Justification: As ssn is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: death\_type\_id

Justification: As death\_type\_id is a foreign key, database by default considers it as a non clustered index.

Index: non clustered

Columns: disaster\_id

Justification: As disaster\_id is a foreign key, database by default considers it as a non clustered index.

Table: **DeathType**

Columns:

death\_type\_id

cause\_of\_death

Primary Key Justification: EveryDeathType rowwill have a unique death\_type\_id.

Normalization form: The above table is in 4th Normal Form.

Indexes:

Index: clustered

Columns: death\_type\_id

Justification: As death\_type\_id is a primary key, database by default considers it as a clustered index.

Table: **Injured**

Attributes:

ssn

first\_name

last\_name

address\_line\_1

city

country

zip\_code

emergency\_contact

Primary Key Justification: Injured row can be uniquely identified with ssn.

Normalization form: The above table is in 2nd Normal Form since the address\_line\_1, city, zip\_code and country are interdependent on each other and separating this columns to a different table will not be relevant for the database need.

Indexes:

Index: clustered

Columns: ssn

Justification: As ssn is a primary key, database by default considers it as a clustered index.

Table: **InjuredType**

Columns:

injury\_type\_id

type\_of\_injury

Primary Key Justification: Every **InjuredType** row will have a unique injury\_type\_id.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: injury\_type\_id

Justification: As injury\_type\_idis a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: type\_of\_injury

Justification: In most of the queries the search will be based on the type\_of\_injury , so type\_of\_injury is considered as non clustered index.

Table: **HealthCareManagementService**

Columns:

service\_id

type\_of\_service

Primary Key Justification: Every service provided will have a unique service\_id.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: service\_id

Justification: As service\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: type\_of\_service

Justification: In most of the queries the search will be based on the type\_of\_service, so type\_of\_service is considered as non clustered index.

Table: **EmergencyService**

Columns:

emg\_svr\_id

name

service\_provider

description

Primary Key Justification : From emg\_svr\_id we can uniquely identify the **EmergencyService**.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: emg\_svr\_id

Justification: As emg\_svr\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: name

Justification: In most of the queries the search will be based on the name of the EmergencyService, so name is considered as non clustered index.

Table : **AdverseEffect**

Columns:

adverse\_id

name

preventive

Justification: From adverse\_id we can uniquely identify the **AdverseEffect**.

Normalization:The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: adverse\_id

Justification: As adverse\_id is a primary key, database by default considers it as a clustered index.

Table: **Role**

Columns:

role\_id

role\_name

privilege

Justification:From role\_id we can uniquely identify the **Role**.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: role\_id

Justification: As role\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: role\_name

Justification: In most of the queries the search will be based on the role\_name, so role\_name is considered as non clustered index.

Table: **User**

Columns:

user\_name

password

role\_id [foreign key; references **role\_id** of **Role** table]

Justification : As each **User** will have specific role\_id of role table, foreign key approach has been followed. Every **User** will have unique user\_name.

Normalization: The above table is in 4th normal form

Indexes:

Index: clustered

Columns: user\_name

Justification: As user\_name is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: role\_id

Justification: As role\_id is a foreign key, database by default considers it as a non clustered index.

Table : **DisAdvEff**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

adverse\_id [foreign key; references **adverse\_id** of **AdverseEffect** table]

Foreign Key Justification : As a disaster\_id can be linked to multiple adverse\_id and a adverse\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

Normalization:The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: disaster\_id,adverse\_id

Justification: As disaster\_id, adverse\_id is a primary key, database by default will create **clustered index** that consists of **two columns**.

Table : **DisEmgSvr**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

emg\_svr\_id [foreign key; references **emg\_svr\_id** of **EmergencyService** table]

pre

post

Justification : As a disaster\_id can be linked to multiple emg\_svr\_id and a emg\_svr\_id can be linked to multiple disaster\_id, cross reference approach has been followed to establish relationship.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: disaster\_id, emg\_svr\_id

Justification: As disaster\_id, emg\_svr\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: disaster\_id

Justification: As disaster\_id is a foreign key, database by default considers it as a non clustered index.

Index: non clustered

Columns: emg\_svr\_id

Justification: As emg\_svr\_id is a foreign key, database by default considers it as a non clustered index.

Table: **DisasterInjured**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

ssn[foreign key; references **ssn** of **Injured** table]

injury\_type\_id[foreign key; references **injury\_type\_id** of **InjuryType** table]

Foreign Key Justification: Cross reference approach has been followed to establish many to many relationship betweenInjuredand DisasterInjuredrelationship. Here it is assumed that a person can be affected in multiple disasters.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: disaster\_id, ssn, injury\_type\_id

Justification: As disaster\_id, ssn, injury\_type\_id is a primary key, database by default considers it as a clustered index.

Index: non-clustered

Columns: injury\_type\_id

Justification: As injury\_type\_id is a foreign key, database by default considers it as a non clustered index.

Index: non-clustered

Columns: disaster\_id

Justification: As disaster\_id is a foreign key, database by default considers it as a non clustered index.

Table: **HealthCareMngmtSvcDis**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

service\_id [foreign key; references **service\_id** of **HealthCareManagementService** table]

num\_served

Foreign key Justification: cross reference approach has been followed to establish many to many relationship between Disasterand HealthCareManagementService.

Note: num\_served is the number of people who were served with particular service during the disaster

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: disaster\_id, service\_id

Justification: As disaster\_id, service\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: service\_id

Justification: As service\_id is a foreign key, database by default considers it as a non clustered index.

Table: **Funding**

Columns:

disaster\_id [foreign key; references **disaster\_id** of **Disaster** table]

funder\_id [foreign key; references **funder\_id** of **FunderDetail** table]

amount(in USD)

Foreign key Justification: cross reference approach has been followed to establish many to many relationship between Disasterand Funding.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: disaster\_id, funder\_id

Justification: As disaster\_id, funder\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: funder\_id

Justification: As funder\_id is a foreign key, database by default considers it as a non clustered index.

Table: **FunderDetail**

Columns:

funder\_id

funder\_org\_name

Primary Key Justification: From the funder\_id we can uniquely identify the every funder/organization.

Normalization: The above table is in 4th normal form.

Indexes:

Index: clustered

Columns: funder\_id

Justification: As funder\_id is a primary key, database by default considers it as a clustered index.

Index: non clustered

Columns: funder\_org\_name

Justification: In most of the queries the search will be based on the funder\_org\_name, so funder\_org\_name is considered as non clustered index.

**Views and Stored Programs**

**Views**

View: **DisasterAdverseEffects**

Goal: name of the disaster, type of disaster and all the adverse effects of that disaster(comma separated) is reported for all the disasters in the system.

A Reporter or a Adverse Effect Reporter will be able to view the multiple adverse effects of all disasters in one table without need for writing complex queries

View: **DisasterDeathCount**

Goal: name of the disaster, type of disaster and death count for a disaster is reported for all the disasters in the system.

A Disaster Casualty Reporter can use this view to see the total count of fatality for each disaster without having to write the query every time.

View: **DisasterInjuredCount**

Goal: name of the disaster, type of disaster and Injured count for a disaster is reported for all the disasters in the system.

A Disaster Casualty Reporter can use this view to see the total count of injured people for each disaster without having to write the query every time.

View: **DisasterFundingDetails**

Goal: name of the disaster, type of funding and amount given as funds for a disaster is reported for all the disasters in the system.

A Funding Analyst can use this view to see the funding provided by each funder for every disaster without having to write the query every time.

View: **EmergencyServiceDetails**

Goal: name of the disaster, name of service provided, service provider and type of service provided for a disaster is reported for all the disasters in the system.

A Emergency Service Provider can use this view to see the information regarding emergency services provided for every disaster without having to write the query every time.

View: **HealthCareServicesForDisaster**

Goal: name of the disaster, type of disaster,type of service provided and total count of people who got served for a disaster is reported for all the disasters in the system.

A Medical Analyst can use this view to see the overall medical services provided and total count of people who got served per diasaster for all disasters in the system.

View: **userRole**

Goal: All the users in the system with their roles and privileges is reported in the system

A Database Administrator can view all the available users, roles and their privileges using this view

**Procedures**

Stored Procedure: **DisasterSpecificAdverseEffects**

Parameters: in name\_input varchar(255)

Goal: This stored procedure takes disaster name as the input and gives adverse effect specific information of that particular disaster

Stored Procedure: **DisasterSpecificFatalityInfo**

Parameters: in name\_input varchar(255)

Goal: This stored procedure takes disaster name as the input and gives information of the people and also thier cause of death and date of death

Stored Procedure: **DisasterSpecificInjuredInfo**

Parameters: in name\_input varchar(255)

Goal: This stored procedure takes disaster name as the input and gives information of the injured people and also injuries sustained by them in that disaster

Stored Procedure: **DisasterTypeSpecificHealthCareManagementServices**

Parameters: in type\_input varchar(255),in service\_input varchar(255)

Goal: This stored procedure takes disaster type and service type as the input and gives information of number of people who opted that particular service type for all the disasters with given disaster type

Stored Procedure: **DisasterSpecificFundingInfo**

Parameters: in name\_input varchar(255)

Goal: This stored procedure takes disaster name as the input and gives the information about the funding provided by the funding organisations to a particular disaster

Stored Procedure: **DisasterSpecificEmergencyServices**

Parameters: in name\_input varchar(255)

Goal: This stored procedure takes disaster name as the input and gives information of services provided(for the disaster) by every department