# Project Phase-1

28 August 2020
11:36

## Introduction to mini world

The mini world we are going to deal with is the Hospital Management System which keep tracks of the patients, employees and some other entities that are a part of this system. Storing these information in a database helps users with the process like whether the patient was a previously admitted one so that the information may not be entered so many times etc., and this system has the ability to optimize and digitize all the processes within the system, which will help to improve customer service, reduce processes costs, streamline the search of medical records, bills, patients, doctors etc.,

## Database Requirements

1. This hospital management systems have employees working in it. Each employee can be either a doctor, nurse or a driver(we are not storing the information of remaining as they are somewhat less crucial).
2. A doctor attends a patient, he examines him and based on that tests will be performed and based on the tests, the doctor prescribes the medicine to the patient(Here we are considering that a specific doctor examines the patient in his own way and is given tests based on the doctor's advice and prescribes his a medicine. As for a particular disease there may be several medicines to cure it but here the doctor prescribes a medicine in those several medicines based on his knowledge)
3. A patient can be either an in-patient(to be treated for some days/months in the hospital) or out-patient(treated in a small span of time). These in-patients are allotted a room which is under a control of one nurse.
4. A driver has his own vehicle and drives a patient to the hospital.
5. A doctor also has a senior who is also a doctor based on the experience he has. Like a doctor who joined right after his completion of studies, sometimes referred to as a junior doctor.

## Entities

1. **Patient**
   Type: Strong entity
   Attributes:
   1. Simple- blood-group[varchar()], gender[char()], date-of-

birth[varchar()]
2. Composite- name[char()], address[varchar()], p-id[varchar()]
3. Multi-valued- phone-number[INT()], email[varchar()]
4. Derived- age[int()]

Note: Key attribute is p-id, email, phone-number.

**2. Employee**

Type: Strong entity

Attributes:

1. Simple- e-id[int()], joining-date[varchar()], date-of-birth[varchar()], salary[int()]
2. Multi-valued- phone-number[int()]
3. Composite- name[char()], address[varchar()]
4. Derived- age[int()]

Note: Key attribute here is e-id, phone-number.

**3. Nurse**

Type: Weak entity

Attributes:

1. Simple- Qualification[varchar()], communication-skills[varchar()]

**4. Driver**

Type: Strong entity

Attributes:

1. Simple- Driving License number[varchar()]

**5. Doctor**

Type: Strong entity

Attributes:

1. Simple- Qualification[varchar()], specialization[varchar()]

**6. Room**

Type: Strong entity

Attributes:

1. Simple- room-number[int()], room-cost[int()], room-type[varchar()]
2. Derived- number of patients[int()]

Note: Key attribute here is room number

**7. In-patient**

Type: Strong entity

Attributes:

1. Simple- arrival-date[varchar()], discharge-date[varchar()], disease[varchar()]

**8. Out-patient**

Type: Strong entity
Attributes:

1. Simple- arrival-date[varchar()], mild-disease[varchar()]

9. **Relative**
   Type: Strong entity
   Attributes:
   1. Simple- relation[char()], phone-number[int()]
   2. Composite- name[char()]

10. **Test**
    Type: Strong entity
    Attributes:
    1. Simple- test-id[int()], test-cost[int()]
    2. Composite- test-name[char()]

    Note: Key attribute here is test-id.

11. **Medicine**
    Type: Strong entity
    Attributes:
    1. Simple- id[int()], cost[int()], quantity[varchar()], expiry-date[varchar()]
    2. Composite- medicine-name[char()]

12. **Bill**
    Type: Weak entity
    Attributes:
    1. Simple- bill-date[varchar()], medicine-cost[int()], room-cost[int()], other-charges[int()]
    2. Derived- Total cost[int()]

13. **Vehicle**
    Type: Weak Entity
    Attributes:
    1. Simple- Vehicle-number[int()], year[int()], Model[varchar()]

Note: Here the composed attribute name has first name, middle name and last name attributes, address has door-number, street, city, state as attributes. In the patient attributes p-id is formed by the combination of phone number and email of that patient.

The ones written in [] are describing the data types of the attribute.

## Relationship Types

1. **Prescribes**

Entity types Doctor, Patient, Test and Medicine are having the relationship type Prescribes.
Patient is in total participation and remaining three are in partial participation with the given relationship type.
The cardinality ratio here is 1:1:M:N(Doctor : Patient : Test : Medicine)
The cardinality constraint of an entity in this relationship is defined by a pair of three entity instances associated with the other single entity instance.

2. **Has**
   In-patient entity type has one to one weak relationship with Relative entity type.
   In-patient is in partial participation and Relative is in total participation with the entity type.

3. **Allocated**
   In-patient entity type has many to one relationship with Room entity type.
   Both In-patient and Room are in total participation with relationship type.
   Here one room has at most 15 in-patients.

4. **Governs**
   Nurse entity type has one to one weak relationship with Room entity type.
   Both Room and Nurse are in total participation with the relationship type.

5. **Drives**
   Driver entity type has one to one weak relationship with Vehicle entity type.
   Both Driver and Vehicle are in total participation with the relationship type.
   Here the relationship type attribute is p-id which is id of the particular patient.

6. **Paid**
   Bill entity type has one to one weak relationship with Patient entity type.
   Both Patient and Bill are in total participation with the relationship type.
   Here the relationship type attribute is bill-date(simple attribute, varchar())

7. **Supervise**
   Entity Doctor is in a relation supervise with the same entity doctor.
   Here it is a many to many relationship.

## Subclasses

1. Entities Doctor, Nurse, Driver are the subclasses of the entity Employee

Here all the entities are in partial relationship with this subclass.

2. Entities In-patient and Out-patient are the subclasses of the entity Patient. Here Patient entity is in total participation and remaining two are in partial participation with this subclass relationship.

## Functional Requirements

1. **Retrieval**
    1. **Select:** Retrieve the complete data tuples of patients that are male.
    2. **Projection:** List the names of all patients that are discharged on a same date.
    3. **Aggregate**: Sum of total costs of the bill payed by all the patients.
    4. **Search:** Retrieving the entries in the entity type Medicine that the match the given partial text.
2. **Analysis**
    1. **A-1:** Number of patients taking the medicine whose quantity is greater than 15mg and undergone treatment under a ENT doctor.
    2. **A-2**: On total number of different rooms that at least one patient resides having a heart disease.
3. **Modification**
    1. **Insert**:  Insertion of the admitted patient information and checking for integrity constraints such as number of digits of phone number and like if a specific attribute is of integer type then only integer values are given for attributes, likewise this applies to remaining attributes.
    2. **Update**: Updating the information of costs of medicines and rooms whenever they change.
    3. **Remove**: Deleting the information of the driver when he is removed or left from the job.

**Done by:**

1. Revanth- **2019101052**
2. Praneeth- **2019101002**
3. Raghavendra- **2019101074**