ER Model for Hospital Database

Team Floppy Disks

Devesh Marwah (2020115005), Hariharan Kalimuthu (2020115015) & Sudha Tanay Doddi (2020115010)

Introduction

This mini-world deals with a hospital database, through which information can be accessed and modified. Data regarding patient records, doctor credentials and users, along with treatments and medicine, can be obtained. Mini-worlds like these need to exist in order to comprehend the vast array of data and information when dealing with large organized institutions.

Purpose

This database aims to provide a comprehensive and exhaustive outlook of everything related to this hospital and establish a model for categorizing data and establishing relationships among said data. The database mainly focuses on the economic aspect of the hospital management and makes it more efficient to handle, store and carry out transactions with transparency.

Users

The users of this database can be anybody who is part of the hospital ecosystem. This can range from doctors trying to access patient records, patients who wish to view their information, or hospital staff managing financial logistics.

Applications

The applications of this database have several uses from:

a) Maintaining a coherent and organized database for hospitals or similar institutions with a similar structure.
b) Accessibility of records and related information for detailed processing and comprehension. This can be patient/doctor/medicine/treatment data.
c) Provides a bird's eye view of the structural integrity of the database and helps with streamlined database management through information retrieval and modification for data storage purposes.

Database Requirements

Entities

1. **admin**
    a. id [varchar(10)] (Primary key)
       Mention all the constraints of the attributes.
    b. name *(Composite attribute)*
        - first_name [varchar(15)]
        - middle_name [varchar(15)]
        - last_name [varchar(15)]
    c. login [varchar(15)]
    d. password [varchar(16)]

2. **users (Superclass entity type)**
    a. id [varchar(10)] (Primary key)
    b. name *(Composite attribute)*
        - first_name [varchar(15)]
        - middle_name [varchar(15)]
        - last_name [varchar(15)]
    c. gender [char(1)]
    d. address [varchar(100)] (*Mutlivalued attribute*)
    e. dob [date]
    f. permissions *(Composite attribute)*
        - p_id [varchar(10)]
        - p_name [varchar(30)]
    g. tel_no [bigint(12)] *(Multivalued attribute)*
    h. age [int(3)] *(Derived attribute)* can be derived from DOB as given
    i. emailID [varchar(30)]

This entity has patients, doctors,other_workers as its subclasses.

3. **other_worker (Superclass and subclass both)**
    a. salary [decimal(10,2)] (>0)
    b. job [varchar(25)]

This subclass behaves like subclass and superclass. It is a subclass of users and a superclass of entities grade_1 & grade_2.

4. **patient**

    a. adm_date [date]
    b. complaints [varchar(100)]

5. **doctor**
    a. salary [decimal(10,2)] (>0)
    b. department [varchar(15)]
    c. doc_ssn [varchar(15)] (Unique Key)
6. **treatment**
    a. t_id [varchar(10)] (Primary key)
    b. t_name [varchar(30)]
    c. cost [decimal(12,2)] (>0)
    d. m_use [varchar(10)](Multivalued) :- Store ids of medicines used for a treatment.
7. **medicine**
    a. m_id [varchar(10)] (Primary key)
    b. m_name [varchar(30)]
    c. mrp [decimal(8,2)] (>0)
    d. discount [decimal(4,1)] (>=0)
    e. cost [decimal(8,2)] *(Derived attribute)* can be derived from mrp & discount.
8. **grade_1**
    a. doc_ssn [varchar(15)]
    b. equipments_used[varchar(30)]
9. **grade_2**
    a. hours [int] (>0)
    b. contracter_name[varchar(30)]
10. **dosage (Weak Entity)**
    a. med_availability [varchar(15)]
    b. med_salt [varchar(15)]
11. **maintenanceSchedule(Weak Entity)**
    a. costPerWeek [int] (>0)
    b. timeOfUnavailiablity [int] (>0)

Weak Entity:

1. dosage
    a. category
    b. dosageAmount
    c. costPerDose
2. maintenanceSchedule
    c. costPerWeek
    d. timeOfUnavailiablity

Multivalued attribute:- users->address & users -> tel_no
Derived attributes:- users -> age & medicines -> cost

Composite attributes:- users -> permissions & users -> name
doctors is an entity with 2 potential primary key attributes -> doc_ssn, id(inherited from user superclass entity type)

Relationships

1. **admin controls** the **user** data
    a. Many to many
    b. Binary relationship as **n=2**
    c. This relationship connects the **admin** entity to the **user** entity. The admin has full access to the database of the user and hence, contains the permission to read/edit any data that is present in the database. The data regarding which admins have control over which users can be obtained from this relationship.
    d. Cardinality Ratio: N:M
    e. Total participation as each entity in the entity set occurs at least one relationship in that relationship set.

2. **department** has **maintenanceSchedule**
    a. One to One
    b. Binary relationship as **n=2**
    c. This relationship connects the weak entity **maintenanceSchedule** to the parent entity **department**. Each department has a regular maintenance schedule to check for the integrity of the instruments and availability of staff, etc., which incurs some costs. All these details regarding maintenance will have a unique identity only when connected with the parent entity.
    d. **Cardinality Ratio: 1:1**
    e. Relationship Constraint: Total participation by each entity occurs.

3. **Medicine** has **dosage**
    f. One to One
    g. Binary relationship as **n=2**
    h. This relationship connects the weak entity **dosage** to the parent entity **medicine**. Prescribed medicine would have details regarding the dosage and a particular category(tablet/lotion/syrup etc.). These details regarding the dosage cannot be uniquely identified unless it is paired to its parent entity which contains the exact name of the medication.
    i. **Cardinality Ratio: 1:1**
    j. Relationship Constraint: Total participation by each entity occurs.

4. **Generalisation and Specialisation**
    a. As we move from users to the entities such as doctor, patient and other hospital workers, we move from an entity having general attributes to ones with specialisation. Hence, the superclass here is the 'users' entity, and the subclass here are the three entities mentioned above.
    b. The entity 'other workers' in turn acts as a superclass for the entities grade_1 ,grade_2 workers who have their specific attributes.

n>3 Relationships:

1. medicalRecord (Relationship between patient, doctor, treatment, medicine)

- Quaternary relationship as **n=4**
- This relationship connects the patient entity to the doctor, treatment and medicine entity. The details regarding who consulted a particular doctor, the treatment procedure, and prescribed medicines would be obtainable via this relationship. Hence, the entire details of any medical consultation in the hospital can be accessed from here.
- M-N-1-P relationship between the participating entities treatment-patient-doctor-medicine
- Relationship constraint: partial participation as treatment and medicines need not be involved in all consultations.
- It also has an attribute called disease [varchar(15)]

Functional Requirements

Modifications

1. Insert:
    a. Insert details of medicines.
    b. Insert details of patients/doctors/workers.
    c. Add a new admin.

All of the above will be done while checking the constraints of the data inputted.

2. Delete:
    a. Delete details of medical records.
    b. Delete details of patients/doctors/workers.
    c. Delete admins.

3. Update/Modification statements:
    a. Update details of medicine records and check if medicine cost etc follow constraints.
    b. Update details of patients/doctors/workers.
    c. Whenever a new patient is added, a new treatment for him should be assigned.
    d. Modify data requirements of attribute id in admin.

Retrievals

1. Selection:
    a. Get details of doctors from a department.
    b. Get details of all patients.
2. Projection:
    a. Retrieve tuples of patients admitted after a specific date.
3. Aggregate:
    a. Get the maximum salary from all the doctor.
4. Search:
    a. Get the name of the patients that start with the string "xyz".
    b. Get names of all the patients who reside in the locality "xyz".
5. Analysis:
    a. Get the area-wise data on the diseases/illnesses patients are suffering from.
    b. Get the expenditure of people on medical expenses according to each area.
    c. Get the average expenses spent by the people for a particular illness.

Bonus Part:-

1. A. It is interesting and convenient to convert the ternary relationship into four binary relationships in a cyclic way, or another approach can be taken, modelling using weak entities. The first technique is called the conversion technique. We can keep the relationship developed in the centre as an entity(generally weak) and model 4 relationships out of it. Each of the four relationships will have cardinality 1:1 or 1:N as specified earlier in the relationships part, ratio overall for each entity. That will help us model the entire relationship as binary relationships only. The implementation is the same in this technique.

    B. We can also use another way which is to kind of make a cycle in which we can remove the ternary relationship and make three binary relationships among different entities. This is called the decompositional technique.

2. We have included Superclass users. Under them, we have a section of other_workers that has subclasses as grade_1 & grade_2. Hence the entity other_workers behaves as a superclass for grade_1 and is a subclass of users.
3. Since every entity has either an attribute or a set of attributes that help to uniquely identify a row, every entity has a super key. Now in the 'User' entity, we see that ID and emailID can both be unique and hence, are two candidate keys. If we choose ID as the primary key, emailID becomes the alternate key.