Database design I

Linus Falk

November 16, 2022

Contents

Lecture 2: DBMS and RE models

friday 04 nov 15:15

1 DBMS

meta is a description of the data.

typical DBMS functionality

- define database in terms of data types
- construct or load database content
- Manipulate the database, Retrieval, Querying, modification, Accessing the database through web applications

Application activities against a database. Quires that access the different part of data and gives a result. Transactions a set of actions, read and update. Applications should not be accesses by non authorized personal.

Additional DBMS functionality

- active processing to take internal action on data

Example of database

Mini-world is for example a organization. entities of the mini-world are for example in the university case: students, courses and sections (of courses).

This entities have different meta data. Courses have course name, credits etc.

Main characteristics of the database

Insulation between the program and the data. Means that me as user can change the data in the database without changing the code. Allow changing the program/code ontop without change the DMBS code. Database abstraction the DBMS can give different visualization of the database for the user.

DBMS allow concurrent user to retrieve and update the database. Can keep log to undo operation when transactions goes wrong.

The users can be different groups. The one who manages who design, just "watchers" etc.

Actors

System analysts understand the user and what they need. Application programmers taking the information from the the analysis and implement from the specification.

Why use databases

Controlling redundancy in data storage. sharing with multiple users, security etc. Optimization of queries for efficient processing. Provides backup services, Gives different interfaces to different user classes,. Potential to implement standards.

2 Entity relationship model

The system analysis must understand the need/scenario and build a model. ER-model is a popular method to model the needs/requirements before implementing.

We use models to have a formal way to present the formula for the system.

For entities we use nouns, the relationships describe connections between entities and we can use verbs for this as name, capital letter for Entities and relationship.

Design

Begin in natural language and putting together a conceptual model (ER model) then continue with logical modelling, programing and then

Design progress

- Database deign
- Application design

Why a conceptual model

More formal than natural language, to avoid misconception. Can be understood without technical background. Can be used as documentation. Rules how to make this modelling, and can therefore be transformed and mapped to the implementation.

Starting with a ER model

first try to identify the three categories first. Relationships should be links between entities. Entities got different attributes.

Tips good practice to do it minimally. Start from top left and go to bottom right. The employee works on project, going from left to right. Easier to read and understand.

Lecture 3: DBMS and RE models

monday 07 nov 15:15

3 ER modelling, recap COMPANY example

- Tips, use colors to identify entities, relationships and attributes. Solutions are not unique and there could be many solutions. Naming scheme, nouns for entities, verbs for relationships. Capital from entitles and relationship. Lower case for attributes.

- Conceptual models are abstract and lacks details but are more formal than natural language.
- Double oval multi-value attribute.
- Trick for many to many, think of table how they are connected. John can only be a manager for one department. The department can have many employees, John, Joseph and Jenny.
- In relationship, 1:1 1:N or N:M. Double it should be a connection, single line is optional. An department may not have any employees but all employees should belong to a department.
- Key should be unique in entity set. Key is underlined. Can be a composite of two attributes. The combination should then be unique. For an example a project can have the same name but the combination with a project number should be unique.
- Weak entity vs strong: weak can not exist without the existence of another strong entity, does not have a key. Dependent is a weak entity, need Employee that is strong and have key.
- Read the ER diagram from top left to bottom right and try to model it this way.
- A way to choose a key is to look at the size of the key, if one is smaller this one could be better to use.
- Relation is a table, be careful with the

Attributes and value sets

Domains is a range of values for the attribute.

Relationship of higher degree

- degree 2 called binary
- degree 3 called ternary

Avoid using relationship degree higher than two (binary). Constraint are harder to specify for higher (>2) degree relationships.

4 Exercise 1

Galleries keep information about artists, their names (which are unique), birthplaces, age, and style of art. For each piece of artists, the artist, the year it was made, its unique title, its type of art (e.g., painting, lithograph, sculpture, photograph), and its price must be stored. Pieces of artwork are also classified/-belongs into groups of various kinds, for example, portraits, still lifes, works by Picasso, or works of the 19th century; a given piece may belong to more than one group. Each group is identified by a name (like those just given) that describes the group. Finally, galleries keep information about customers. For

each customer, galleries keep that person's unique name, address, total amount of dollars spent in the gallery (very important!), and the artists and groups of art that the customer tends to like.

Entities

Relationship

Lecture 5: Relation database desing by ER and EER

monday 15 nov 15:15

5 Mapping of ER- and EERR to relational database

What do we want to achive with the mapping?

- Preserve information, attributes, map them correctly
- maintain constraints, cardinality 1:N etc
- Minimize the use of Null values

Mapping algorithm

- **Step 1**

In most cases enititys becomes tables. For each strong/regular enitity E in the ER schema, create a relation R that includes all simple(?) attributes of E. Choose on of the key attributes as primary key for R. If this key is a composite, the set of simple attriubutes will together form the key.

We cant have multi value attributes but we can change it to a new entity.

read about strong and weak relationships a weak relationships is weak if the key is dependent on an entity above the relationshio, double line relationship.

- **Step 2**

For each weak entitity W in the ER schema with a owner entity E, create a Relation R and include all simple attributes. Also include as foreign key the attributes of R the primary key attributes of the relations(s) that correspond to owner enitity E type(s) QUE?

- **Step 3**

For each binary 1:1 relationship type R in the ER Schema we should identify the relations S and T that correspond to the entity types that are participating in R. Here there is three different approaches:

1. Foreign key (2 relations) approach. Choose one of the relations S and include a foreign key in S the primary key of T. QUE?
2. Merged relation (1 relation) option. An alternative mapping of a 1:1 relatiopnship type is possible by merging the two entitys types and the relationship inta a single relation.
3. Cross reference or relationship relation (3 relations) options. The last alternative is to setup a third relation R for the purpose of cross referencing the primary keys of the two relations S and T

- **Step 4**

Mapping of the Binary 1:n relationships types. For each of regular binary 1:N relationships identify the relation S that represent the participating entitiy type at the N side of the relationship. Include as foriegn key in S the primary ket of the relation T that represent the other entity participating in R. Include any simple attributes of the 1:N relation type as attributes of S.

- **Step 5**

Mapping of binary M:N relationships. For each binary M:N relationship R, create a new relation S to represent R, this is a **relationship relation**. Include as a foreign key attributes in S the primary keys of the relations that represent the participating enitity types. **their combination will form the primary key of S**. Also invlude any simple attributes of the M:N relatioship type as attributes of S.

- **Step 6**

Mapping multivalued attributes. For each multivalued attribute A, create a new relation R. This relation will include an attribute corresponding to A, plus the primary key of attribute K-as a foreign key in R of the relation that represent the entitity type of relationship type that A has as an attribute. The primary key of R is the combination of A and K. If this multivalued attribute is composite we incluide its simple components.

- **Step 7**

Mapping of N-ary relationship types. For each n-ary QUE? relationship type R where n >2, we create a new relationship S to represent R. We include R as foreign key attributes in S the pirmary keys of the attributes of the n-ary relationship type or simple components of composite attributes as attributes of S again, QUE?

6 Mapping of generalization and specialization hierarchies

Mapping EER model constructs to realtions

- **Step 8**

Some options for mapping specialization or generalization. Convert each specialization with m subclasses $\{S_1, S_2, \ldots, S_m\}$ and generlized superclass C, where te attributes of C are $\{k, a_1, a_2, \ldots, a_n\}$, (where k is the primary key) into relationalm schemas using on of these options

1. Multiple relation-Superclass relations only.

2. Multiple relations-Subclass relations only

3. Single relation with one type attribute

4. Single relation with multiple type attributes.

Normalization

Guideline 1

Information is stored redundantly: We should not waste space with having redundant information in the relation table, will making it hard do delete change values or insert new information.

Functional depencincies

Are used to specify formal measrues of the "goodness" of relational designs. keys are used to define noermal forms for relations. thet are contraints that are derived from the meaning and internrelationships check.

example of FD constraints. SSN determines employee name. Project number determines project name and location. Employee ssn and project number determine the number of hours of the week the employee works with the project.
.

In order to understand FDs, we need to understand the meaining of the attriubtes. We need to discuss with the costumer to understand attributes and relationships involved.

1NF we remove repeated information.
2NF remove partial dependencies.
3NF trying to elimnate trancietive depencecies.

Lecture 6: Normailization

wednesday 16 nov 10:15

normalization is a formal way to transform the er model to relation tables and a database. With this method we avoid

- Waste storage
- insertion anomloies
- modiciation anamolies
- deletion anomaies

in the context of this course we will go to **Normal form 3** or **NF3**

Functional depencency can proven if its false but its only possible that the other way around. It may hold but from one table its not possible to determine.

partial dependency, depends partly of the key. Full dependency depends on the whole key.

- **1NF** All attributes depend on the key
- **2NF** All attributes depend on the whole key
- **3NF** All attributes depend on nothing but the key

We want to get rid off partial dependencys.

7 Basic SQL

We will get a lot of error messages if we dont set up the relational tables correct. SQL not straight forward as Python or Java.

remeber that foreign key shall have the same structure (like char(9)) as the "original" ›