Database Management Systems (LIX022B05)

Instructor: Çağrı Çöltekin
c.coltekin@rug.nl

Information science/Informatiekunde

September 9, 2013

Why bother?

Why should you care about learning databases?

- Databases are everywhere, as an I(C)T professional you will have to deal with them.
- There are specific jobs in the industry where databases are the central piece: they may become your (professional) life.

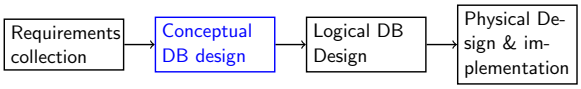
Databases (or database management systems) are good for

- isolating data from applications
- enforcing data integrity
- sharing data between multiple users/applications
- efficient data access manipulation (for most cases)
- and also, access control, backup/recovery ...

Some concepts

- Database architectures are typically classified as,
  - standalone (or serverless)
  - client-server
  - multi-tier (or three-tier)
- Relational data model and relational database management systems are the most common way of organizing data and running databases.
- A relational database is organized in the form of multiple tables (or relations).
- SQL is the standard language to create, manipulate and query databases.

Database Design Process



- DB design is generally part of a bigger software design process.
- These steps reflect the idealized case. Typically, you may need to re-iterate over some of the steps multiple times.
- This week, we are interested in the second step (and a bit of third step).

Database Design: what to avoid

- The obvious: incomplete design. The database design should handle all transactions required by the application/enterprise.
- The bad: Replication of the same information. Causes inconsistency, wastes space.
- The ugly: unnecessary/redundant information. No need to store things that are not necessary.

What not to do (example)

Is anything wrong with this table?

title	*author*	*genre*
I, Robot	Isaac Asimov	sci-fi
A Wizard of Earthsea	Ursula K. LeGuin	fantasy

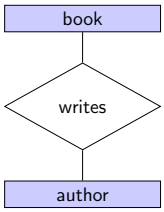
How to represent if ...

- an author writes multiple books
- a book is written by multiple authors
- we want a list of authors sorted by last name.
- there is a book we do not know who the author is
- there are books or authors with the same name
- multiple editions of the same book

The entity-relationship data model

The entity-relationship (E-R) model is commonly used for specifying high-level database design.

- Two important concepts: entity sets and relation sets.
- An E-R model specifies real-world entities (objects, things) and their relations.
- It has a diagrammatic representation. That particularly comes handy in communicating your design with non-specialists.



The entity sets and attributes

An entity is a real world (abstract) object.

- In DB design, we are interested in entity sets.

Entities are defined by their attributes.

- The attributes can be
  - simple,
  - composite,
  - multi-valued,
  - derived.
- The set of allowed values for an attribute is called its domain.
- An attribute may be allowed to have null value or not.

Weak entity sets

A weak entity set represents entities that cannot be identified without help of one or more other entities.

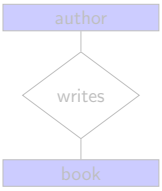
- Weak entity sets do not have a primary key.
- The set of attributes that identify a weak entity from others is called the discriminator.
- The existence of a weak entity set depends on other (strong) entity set(s) which are said to identify or own the weak entity set.
- The relationship set between a weak entity and its identifying entity set(s) are called identifying relationship(s).

The relationship sets

A relationship set states an interaction between two or more entity sets.

- Typically, we use binary relationships in database design. However, n-ary relationships may be more appropriate in certain cases.
- If all entities in a participating entity set has to participate in a relationship, then the participation is said to be total. Otherwise it is partial.
- A relationship set can be to and from the same entity set, in which case it is called a recursive relationship set.
- Relationships may have descriptive attributes.

Mapping cardinalities



An important constraint on relationship sets is mapping cardinalities.

A relationship can be,

- one to one: an author can write only one book, and a book can be written by only one author.
- one to many (from author to book): a book can be written by only one author, but an author can write many books.
- many to one (from author to book): more than one author can writea single book, but an author is allowed to write only one book.
- many to many: a book can be written by more than one author and an author can write multiple books.

Keys of entity sets

A primary key, formed by one or more attributes, uniquely identifies an entity.

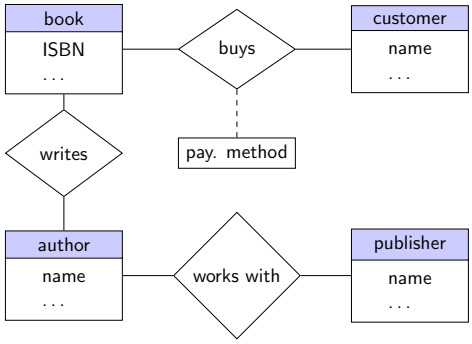
- At worst case, the values of all attributes of an entity forms the primary key.
- A *candidate key* is one with no redundant attributes.
- There may be more than one candidate keys. Choice of primary key is a database design decision.
- Sometimes the primary key is 'invented': ISBN, social security number, student ID numbers, ...

Keys of relationship sets

Primary keys of relationship sets are formed by primary keys of participating entity sets.

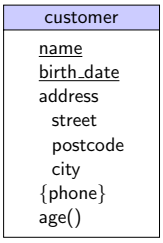
- many-to-many, the primary key of the relationship set is the union of the primary keys of the participating entities.
- one-to-many, the primary key of the entity set on the 'one' side is the primary key of the relationship set.
- one-to-many, primary keys of either set can serve as the primary key of the relationship set.

E-R diagrams: a simple example

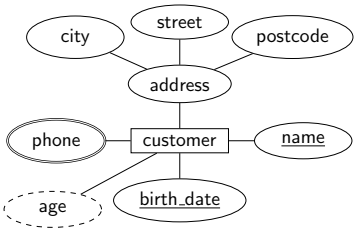


The entity sets (the textbook notation)

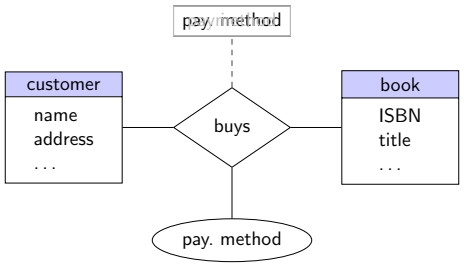
- Entity sets are represented with split rectangles.
- The top part is name of the entity set, bottom part lists the attributes.
- Attributes that form the primary key are underlined.
- Composite attributes are listed by indenting the part items.
- Multi-valued attributes are listed in curly braces.
- Derived attributes are suffixed with '()'.



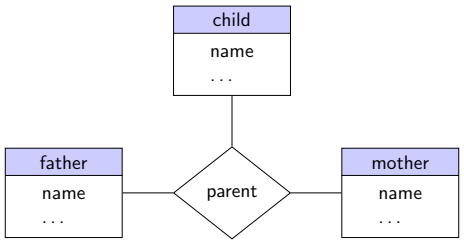
The entity sets (more common notation)



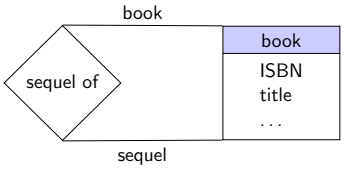
Relationship sets



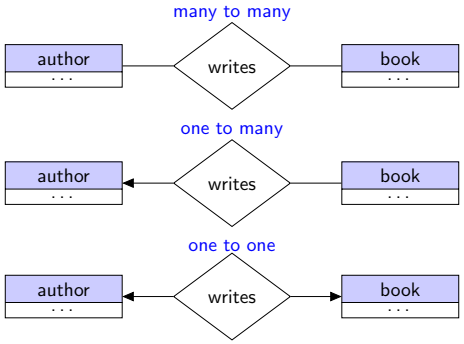
N-ary relations



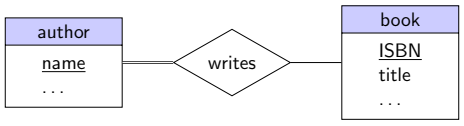
Recursive relationship sets/roles



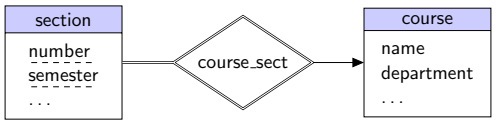
Mapping cardinalities



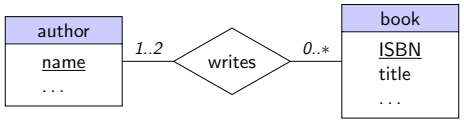
Participation constraints



Weak entities

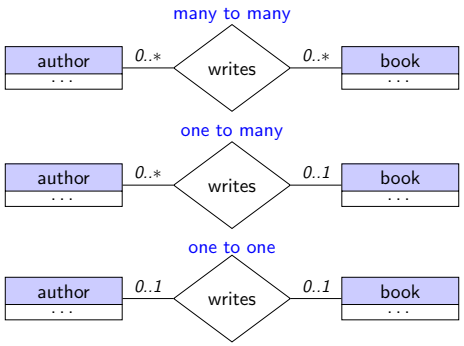


Alternative notation for constraints



- Lower limit of 1 means that the participation is total.
- Upper limit of 1 on both sides means a one-to-one relationship set.
- Upper limit of 1 only one side means a one-to-many relationship set.

Mapping cardinalities (example)



Choices in E-R model

Usually, the modeling decisions in an E-R model are intuitive.

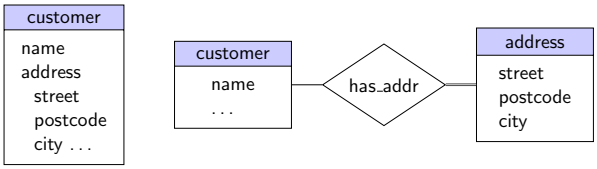
However,

- the same real-world situation can be modeled in more than one way,
- the decisions change depending on what is being modeled,

DB design is more of an art than science, but there are some useful guidelines.

Entity set or attribute

Often it is not clear whether we should use an entity or attribute.

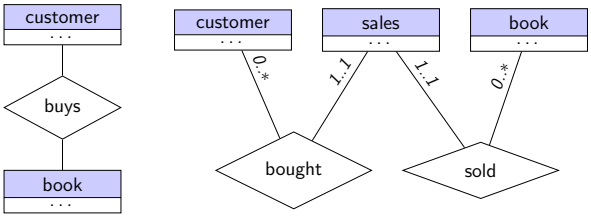


Questions to ask:

- Can one customer have multiple addresses?
- Do other entities also have relationships with 'address' entity?
- Does 'address' entity set require additional information that 'customer' entity set does not need?

Entity set or relationship

Often it is not clear whether we should use an entity or attribute.

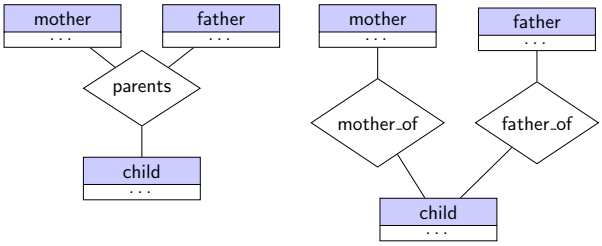


Unless necessary, more complex options should be avoided.
Question to ask:

- Do 'sales' records participate in other relationship sets?

Binary or n-ary relationships

Every n-ary relationships can (mechanically) be decomposed into a number of binary relationships.



- N-ary relationship may reflect the real-world better.
- Binary decomposition may increase complexity.
- Some constraints may be difficult/impossible to express in binary form.

Weak entities

Unless necessary, avoid weak entities.

Use a weak entity set

- if the entity set does not have a primary key.
- if the entity participates in relationships other than the owning relations.

Rules of thumb

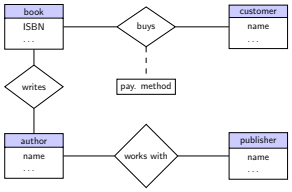
Conceptual database design is an art as much as it is a science. Generally, there is no single correct solution.

- Do try to represent the real-world as faithful as possible.
- Don't model same thing twice: avoid redundancy.
- Don't model things that are not needed: look for simpler solutions.

From E-R data model to relational DB schema

- E-R model gives us a way to formalize our ideas, and communicate them using E-R diagrams.
- In relational databases everything should be stored in relations (tables).
- Good E-R modeling does not guarantee good DB design: there is more work to be done.

We have:



We want:

```
book(ISBN, title, year)
customer(name, phone, street)
author(name,...)
```


Database schema

- A database schema is a set of textual table descriptions.

customer(ID, name, phone, street, postcode, . . .)

In a database schema,

- we specify name of the tables,
- attributes (column names) of the table as they would appear on the database
- the primary key choice for the tables by underlining the attributes that form the primary key

E-R entity to relation schema

- Simple attributes are mapped directly to attributes of the table
- Each component attribute of a composite attribute is mapped to standalone attributes of the table. The top level composite attribute is not used.
- Derived attributes are not modeled.
- Multi-valued attributes treated specially.
- Primary key of an entity becomes the primary key of the corresponding table.

E-R multi-valued attribute to relation schema

- Multi-valued attributes get separate tables.
- The table includes all component attributes (if multi valued) of the attribute, and the primary key of the owning entity set.
- Primary key of a table created for a multi-valued attribute is the primary key of the owning entity set and discriminating set of attributes of the multi-valued attribute.
- A foreign key constraint is added to the table generated from multi-valued attribute that references to the owning entity set's primary key.

E-R weak entity to relation schema

Weak entities are similar to multi-valued attributes.

- Each weak entity set is mapped to a table.
- The table includes all attributes weak entity set, and the primary key of the owning entity set.
- Primary key of a table created for a weak entity is the primary key of the owning entity set(s) and discriminator of the weak entity set.
- A foreign key constraint is added to the table generated for the weak entity set that references to the owning entity set's primary key.

E-R relationship to relation schema

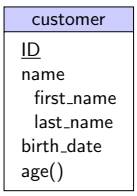
Each relationship set is mapped to a table schema that includes primary keys of participating entities, and the descriptive attributes of the relationship set. Note that we may need to rename some attribute names.

The primary key of the relation schema depends on the mapping constraint of the relationship set:

- For many-to-many relationship sets the primary key is the union of the primary keys of the participating entity sets.
- For one-to-many relationship sets the primary key is the primary key of the 'many' side.
- For one-to-one relationship sets, primary keys of the either participating entity sets can be chosen.

For every attribute of the resulting relation schema which is derived from the primary key of the participating entity sets, a foreign key constraint is added.

E-R entity to relation schema (example)

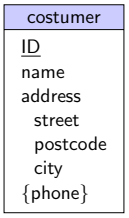



```
customer(ID, first_name,
         last_name, birth_date)

⇒ create table customer (ID int,
                         fname varchar(20),
                         lname varchar(20),
                         birth_date date,
                         primary key (ID));
```


E-R multi-valued attribute to relation schema (example)

```
customer(ID, name, street, pcode, city)
phone(cust_id, phone_nr)
```

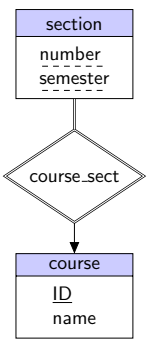


⇒

```
create table
    customer (ID int,
              name varchar(50),
              street varchar(20),
              postcode char(6),
              primary key (ID));

create table
    phone (cust_id int, phone_nr int,
           primary key(cust_id, phone_nr),
           foreign key (cust_id)
               references customer(ID));
```

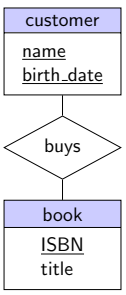

E-R weak entity to relation schema (example)




```
course(ID, name)
section(c_ID, num, semester)

create table
    course (ID int,
            name varchar(50),
            primary key (ID));
⇒
create table
    section (c_ID intint,
             num int, semester int,
             primary key(c_id, num, smester),
             foreign key(c_id)
                 references course(ID);
```


E-R relationship to relation schema (example)

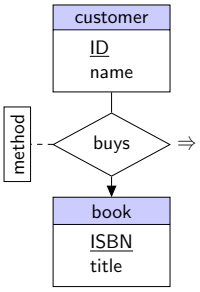


⇒

```
customer(name, birth_date)
book(ISBN, title)
sales(c_name, c_bdate, ISBN);

create table customer (name varchar(50),
                       birth_date date,
                       primary key (name, birth_date));
create table book (ISBN char(13),
                   title varchar(50),
                   primary key(ISBN));
create table
    sales(c_name varchar(50),
          c_bdate date,
          ISBN char(13),
          primary key(c_name, c_bdate, ISBN),
          foreign key (c_name, c_bdate) references
                      customer(name, birth_date),
          foreign key (ISBN) references book);
```

E-R relationship to relation schema (example 2)



$\Rightarrow$

book(ISBN, title)
customer(ID, name)
buys(cID, ISBN, method)

```
create table customer (ID int,
                       name varchar(50),
                       primary key (ID));
create table book (ISBN int,
                   title varchar(50),
                   primary key(ISBN));
create table buys (cID int, ISBN int,
      method char(10),
      primary key(cID),
      foreign key (cID) references customer(ID),
      foreign key (ISBN) references book(ISBN));
```


We are not done yet...

The conceptual design is just the beginning.

- Usage requirements: typical/frequent queries, performance concerns..
- Authorization requirements.
- Even a good E-R modeling does not guarantee a good relational database design. Logical database design process aims to eliminate potential problems (next week).

An overall summary

- A conceptual design using E-R data model allows us to
  - think about the DB requirements systematically and formalize the ideas from the requirement analysis,
  - communicate the overall design of the database using a graphical representation.
- E-R notation is varied and non-standard, one can alternatively use another representation like UML.
- E-R constructs can be reduced to a database schema.
- Conceptual modeling is helpful, however, it does not guarantee correct relational database design.

What is next?

- Reading for next week: Relational database design (Chapter 8).
- Assignment: posted today, due next week Friday (Sept 20, 23:59).
- Step-by-step practical exercises for warming up with MySQL (optional, but strongly recommended).
- No lab session on Friday, feel free to email me with your questions.
- You should already have access to the MySQL server.