Introduction to
database and relational model

General scheme of presentation

- What is a database and what is a DBMS.
- Different database models:
  - Hierarchical
  - Network
  - Relational
  - Object
- The relational databases
  - Fundamental concepts.
  - Different activities
  - Entity & relations models

What is a database and what is a DBMS

- ❖ A database can be viewed as
  - On or more files
  - A set of structured data
- ❖ The database is managed by a DBMS. In most cases it‘s a server (exception: Ms-Access).
- ❖ A DBMS can manage one or more databases.
- ❖ It is possible to create links between databases.

DBMS responsabilities

- The responsabilities of a DBMS are:
  - To manage the data and the definitions of the data structures (meta data).
  - To manage the security.
  - To manage the data integrity.
  - To give access to client applications.
  - To manage isolation between transactions.
  - ...

Databases models

- ❖ Different models exist :

1. Hierarchical : nodes of data (set of fields/value) with parent-child relations forming a hierachical tree : a child may have only one parent

2. Network : nodes of data with relations between any nodes : it forms a network : a child may have many parents. In hierachical and network databases to retrieve a node you have to navigate through relations.

3. Relational : tables of data. A table is a set of records having the same structure and the same meaning (entities of the same type). A record is composed of fields; each field has a value. A one to one or one to many relation is modelized as a data field.

4. Object/Relational Model : extension to the relational model to work with objects.

...

Relational model

- Values are atomic (we can‘t access only to a part of a value)
- The sequence of rows is insignificant
- In the different rows of the same table, the column values have the same type (the type is defined for the column)
- The sequence of columns is insignificant
- On each table you have to define a primary key (on a single column or on many). The primary key identifies the record. Values in primary key should be unique. Values in column participating to a primary key can‘t be null.
- On each table you may define indexes (on a single column or on many). Searches with creteria related to indexes are more efficient.

Relational data bases – fundamental concepts

- Data stored in tables.
- Tables formed of rows and columns.
- Primary keys.
- Surrogate.
- Foreign keys.
- Integrity constraints.
-

Database objects

- Schemas : collection of tables, indexes and views
- Tables : storage structure
- Views : stored select SQL
- Indexes : definition of indexation

Primary key, surrogate key

- A primary key is the field that will identify each record in a table.
- A surrogate key is an artificial column added to a relation to serve as a primary key:
  - Often supplied by the DBMS (usage of a sequence)
  - Short, numeric and never changes – an ideal primary key!
  - Has technical values that are meaningless to users
  - Normally hidden in forms and reports

Surrogate key, foreign keys

- You will use a surrogate key either for optimization reasons (less data, quick access) or when you have no “natural” key.
- A foreign key in a table is an identifier of a record of another table (in the table X, the foreign key on the table Y contains values of the primary key of the table Y). In other words, a foreign key is the primary key of one relation that is placed in another relation to form a link between the relations

Referential integrity constraint

- A referential integrity constraint is a statement that limits the values of the foreign key to those already existing as primary key values in the corresponding relation

Activities related to the usage of relational databases.

- Design : data modeling, data definition (meta-data) with SQL or with a tool
- Usage : data manipulation (SQL insert, update, delete) and retrieval (SQL select).
- Administration : security management, performance management, data integrity.

- Such activities are not always allocated to different people.

Data modeling - Entity-relationship model

- The entity & relationship model is a way to represent structures and links that can be stored in a relational database. It is a tool for conceptualisation; it is used mainly during the analysis stage.

- An entity set is a group of entities having the same type (= a table). The definition of this type is part of the definition of the entity set.

- An entity is a flat structure (set of fields) holding the characteristics of a concrete thing (for examples a person, a training session) or of an abstract thing (for examples a knowledge, an objective). (= a table row)

- A relation defines the possibility to create links between two or more entities.

- An entity-relationship model can be represented by textual expressions or by graphs.

- To transform an entity-relationship model into a database model, entities are mapped to tables, relations are mapped as supplementary fields of entities tables or as separated tables.

Entity-relation graphs

- Main represented elements are:
  - Entities sets
  - Attributes
  - Relations between entity sets
  - Cardinality of such relations.

Relations cardinality

- Cardinality means “count,” and is expressed as a number.
- Maximum cardinality is the maximum number of entity instances that can participate in a relationship.
- Minimum cardinality is the minimum number of entity instances that must participate in a relationship.
- There are three types of maximum cardinality:
  - One-to-One [1:1]
  - One-to-Many [1:N]
  - Many-to-Many [N:M]

ONE-TO-ONE Relationship

Example: AUTOMOBILE and REGISTRATION

VIN_NUM	MAKE	MODEL	YEAR

REG_NUM	ADRESS	STATUS

May become

VIN_UM	MAKE	MODEL	YEAR	REG_NUM	ADDRESS	STATUS

or

VIN_NUM	MAKE	MODEL	YEAR	REG_NUM

REG_NUM	ADRESS	STATUS

or

MAKE	MODEL	YEAR	VIN_NUM

VIN_NUM	REG_NUM	ADRESS	STATUS

ONE-TO-MANY Relationship

Example: EMPLOYEE and DEPARTMENT (one department, many employees, one employee, only one department)

EMP_NUM	NAME	GRADE

DEP_NUM	NAME	ADRESS

becomes

EMP_NUM	NAME	GRADE	DEP_NUM

DEP_NUM	NAME	ADRESS

MANY-TO-MANY Relationship

Example: STUDENT and COURSE (many students in one course, many courses, per one student)

STUD_ID	NAME	YEAR

COURSE_ID	NAME	LEVEL

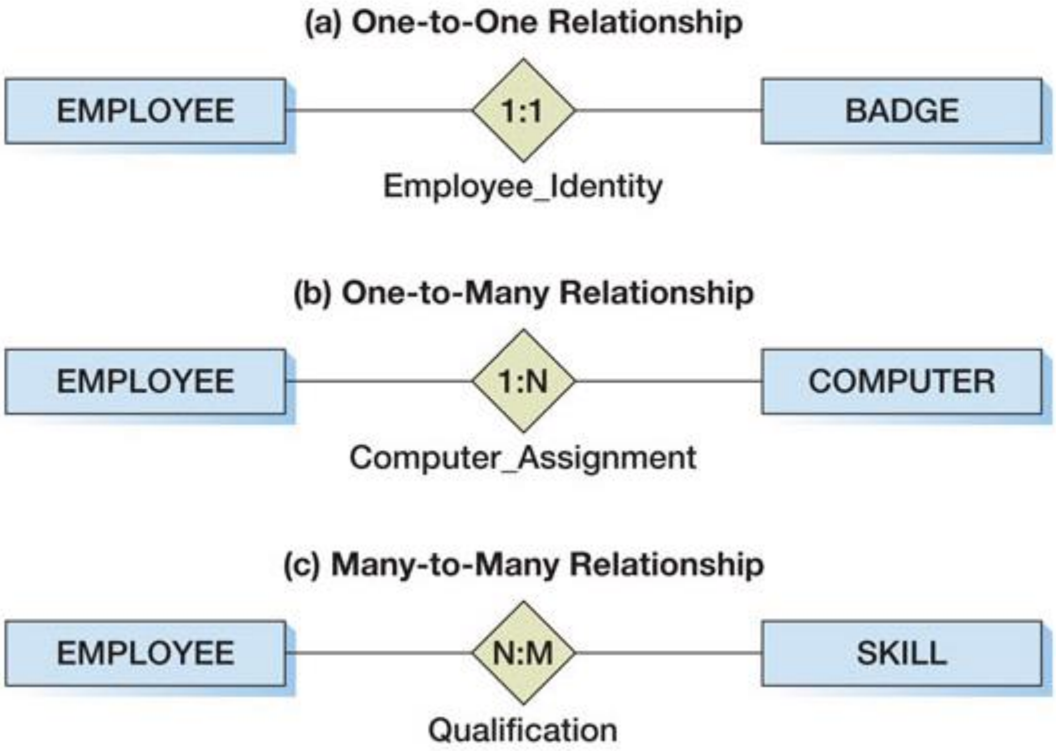
becomes

NAME	YEAR	STUD_ID

COURSE_ID	NAME	LEVEL

STUD_ID	COURSE_ID

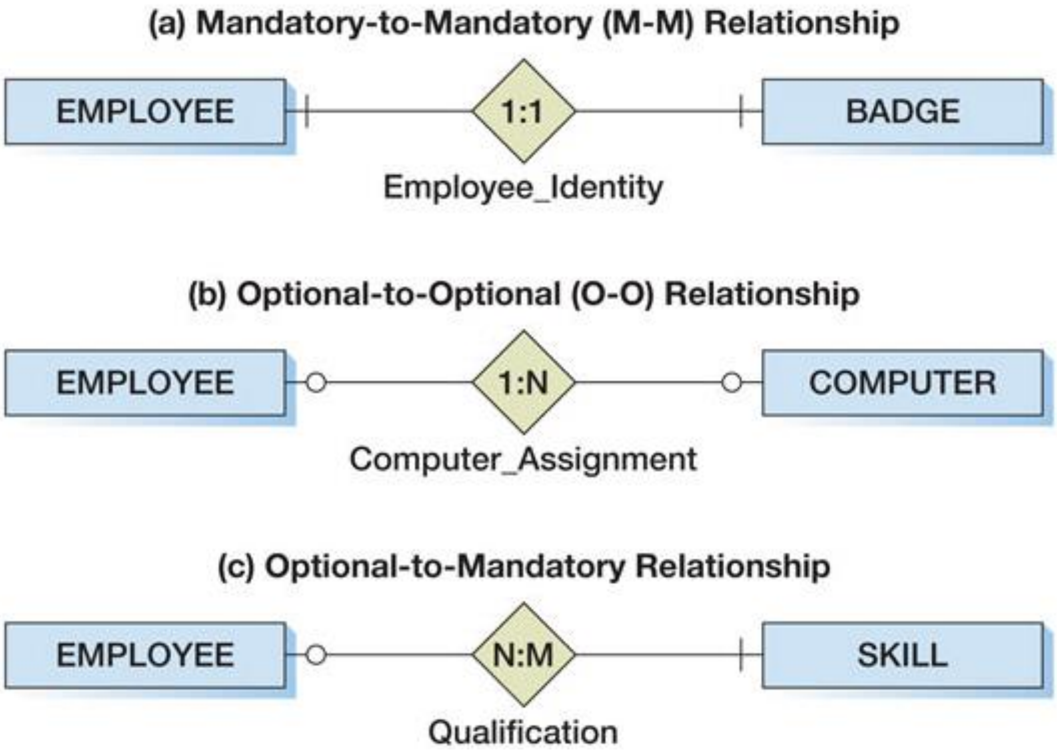
Relations – graphical representation of max cardinality



Relations – min cardinality

- Minimums are generally stated as either zero or one:
  - IF **zero [0]** THEN participation in the relationship by the entity is **optional**, and **no** entity instance must participate in the relationship.
  - IF **one [1]** THEN participation in the relationship by the entity is **mandatory**, and **at least one** entity instance must participate in the relationship.

Relations – graphical representation of min cardinality


(a) Mandatory-to-Mandatory (M-M) Relationship
EMPLOYEE
1:1
BADGE
Employee_Identity
(b) Optional-to-Optional (O-O) Relationship
EMPLOYEE
1:N
COMPUTER
Computer_Assignment
(c) Optional-to-Mandatory Relationship
EMPLOYEE
N:M
SKILL
Qualification

Data Modeling – Crow's Foot representation

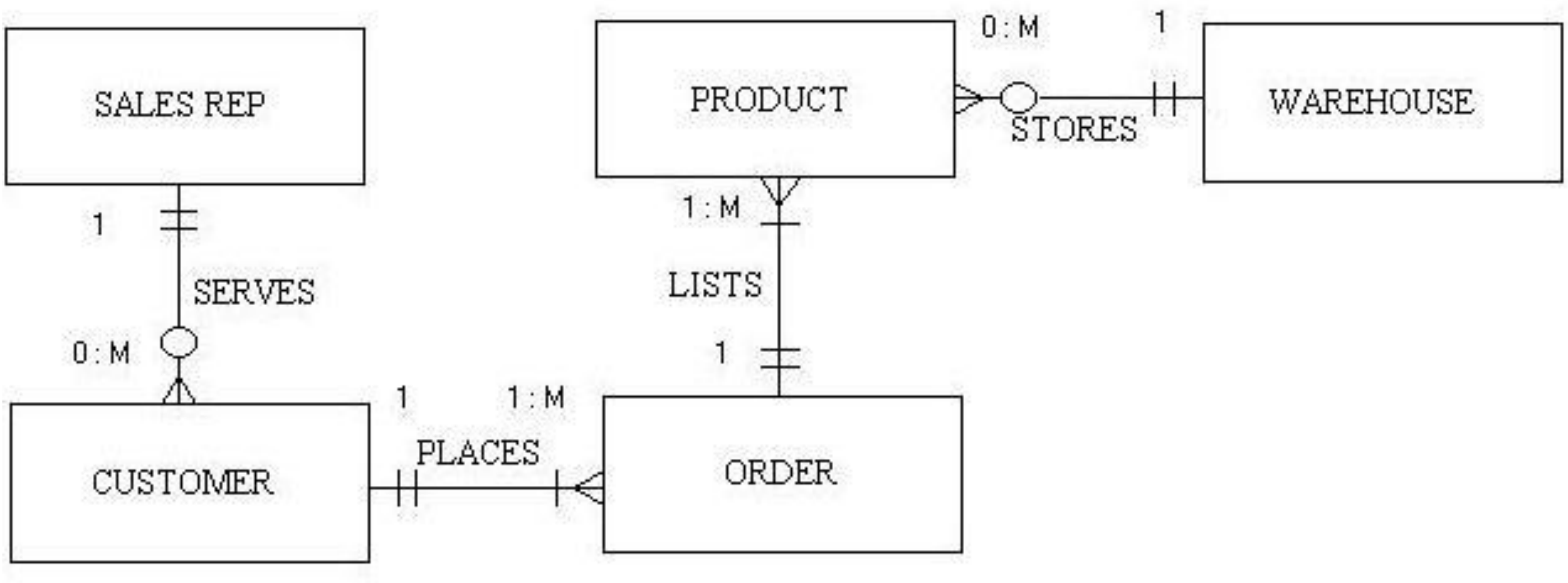


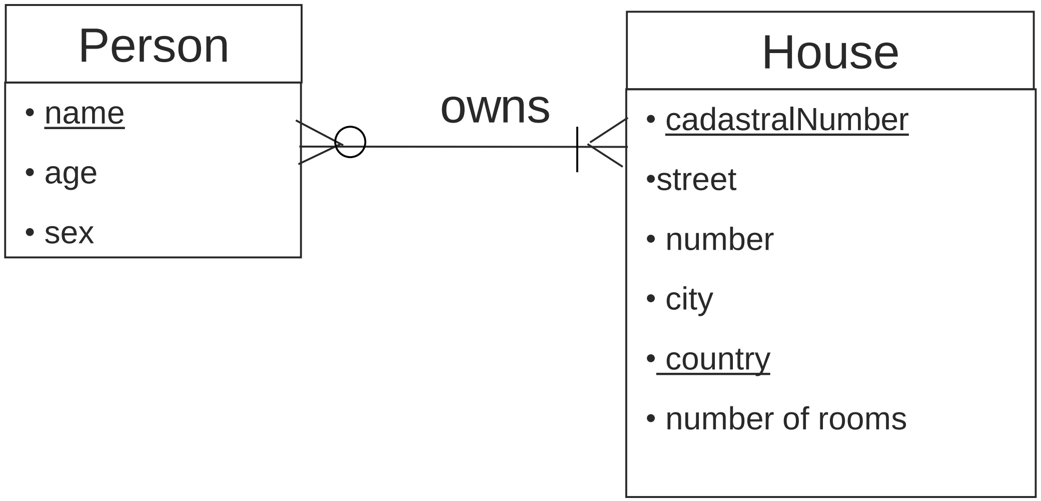
Figure 1. Entity-Relationship Diagram

* 1 INSTANCE OF A SALES REP SERVES 1 TO MANY CUSTOMERS
* 1 INSTANCE OF A CUSTOMER PLACES 1 TO MANY ORDERS
* 1 INSTANCE OF AN ORDER LISTS 1 TO MANY PRODUCTS
* 1 INSTANCE OF A WAREHOUSE STORES 0 TO MANY PRODUCTS

Data modeling - Logical vs Physical Model

- ❖ The logical model is created as support of the design phase. Normally, the designer will work on the logical model to obtain a normalized form (the normalization process is explained here after).
- ❖ The physical model directly represents what will be created (tables and relations) in the database.
- ❖ The main difference are:
- ○ In the physical model all “many to many” relations are represented via an intermediary table.
- ○ In the logical model, foreign keys are not shown.
- ❖ Physical considerations may cause the physical data model to be quite different from the logical data model (de-normalization).

Example of a logical model



Person

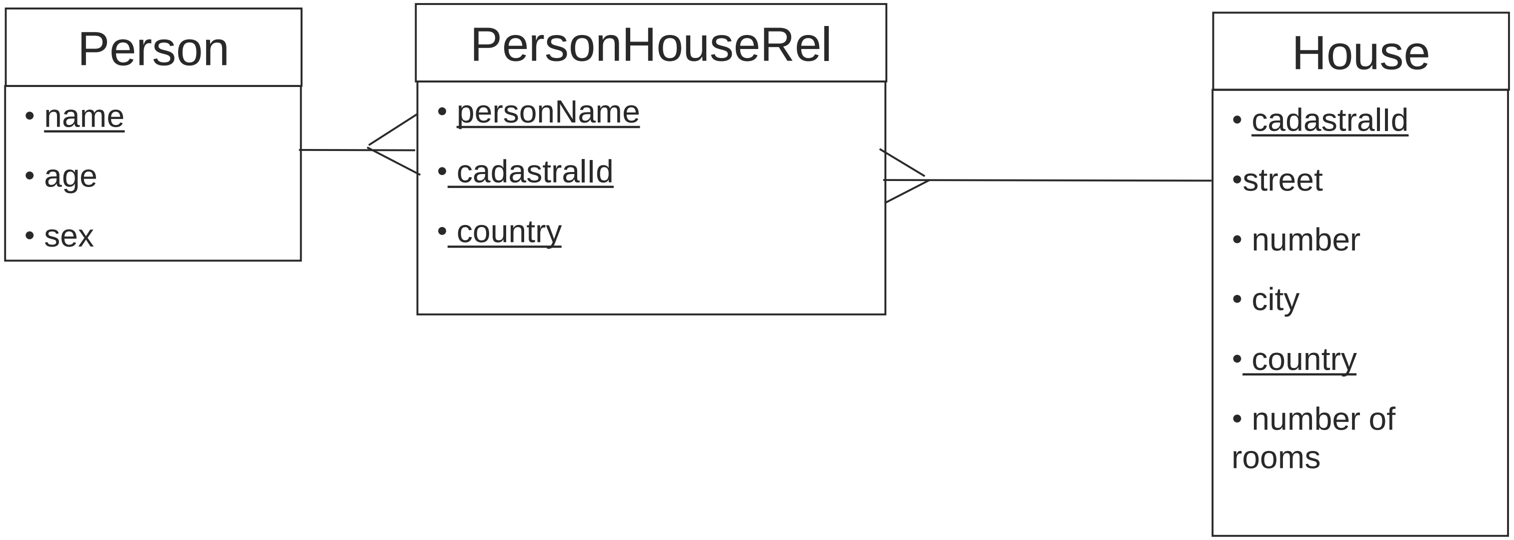
- name
- age
- sex

owns

House

- cadastralNumber
- street
- number
- city
- country
- number of rooms

Example of a physical model



Person

- name
- age
- sex

PersonHouseRel

- personName
- cadastralId
- country

House

- cadastralId
- street
- number
- city
- country
- number of rooms

Normalization forms

- Normalization = respect of guide lines. Enforce the data consistence (no redundant data).
- 1 NF: each field has a single value, we can’t have a variable number of fields in two records of the same table.
- 2 NF: 1NF + a non-key field must give an information (a fact) related to the key, as the whole key, and nothing but the key.
- 3 NF: 2NF + a non-key field can not be an information related to another non-key field
- 4 NF: 3NF + a table should not contain two multi-valued facts about an entity.
- …

1NF

- Each field has a single value, we can’t have a variable number of fields in two records of the same table.
- Bad example:

PersonName	Age	Children
Gaston	65	Benoit,Françoise
André	47	Alice, Pascal, Eloïse

1NF

- Each field has a single value, we can’t have a variable number of fields in two records of the same table.
- Bad example:

Person Name	Age	Child1	Child2	Child3	Child4	Child5
Gaston	65	Benoît	Françoise	Null	Null	Null
André	47	Alice	Pascal	Eloïse	Null	Null

1NF

- Each field has a single value, we can’t have a variable number of fields in two records of the same table.
- A possible solution:

table1

Person Name	age
Gaston	65
André	47

table2

FatherName	Child
Gaston	Benoît
Gaston	Françoise
André	Alice
André	Pascal
André	Eloïse

2NF

- 2 NF: A non-key field must give an information (a fact) related to the key, as the whole key, and nothing but the key.

- Bad example:

Headquarters

Company	Country	NbEmpl	Currency
CW	BE	450	Euro
CW	GB	50	GBP
CW	LU	60	Euro

2NF

- 2 NF: A non-key field must give an information (a fact) related to the key, as the whole key, and nothing but the key.
- A possible solution:

Headquarters

Company	Country	NbEmpl
CW	BE	450
CW	GB	50
CW	LU	60

Countries

Country	Currency
BE	Euro
GB	GBP
LU	Euro

3NF

- A non-key field can not be an information related to another non-key field
- Bad example:

Countries

Country	Currency	Roundup
BE	Euro	0.01
GB	GBP	0.01
LU	Euro	0.01

3NF

- A non-key field can not be an information related to another non-key field
- A possible solution:

Countries

Country	Currency
BE	Euro
GB	GBP
LU	Euro

Currencies

Currency	Roundup
Euro	0.01
GBP	0.01

4NF

- A table should not contain two multi-valued facts about an entity.
- Bad example:

EmployeeId	skill	Language
awe	C	FR
awe	Java	EN
awe	SQL	Null
xyz	Accounting	EN

4NF

- A table should not contain two multi-valued facts about an entity.
- A possible solution:

EmployeeId	skill
awe	C
awe	Java
awe	SQL
xyz	Accounting

EmployeeId	Language
awe	FR
awe	EN
awe	Null
xyz	EN

Thanks for your attention!