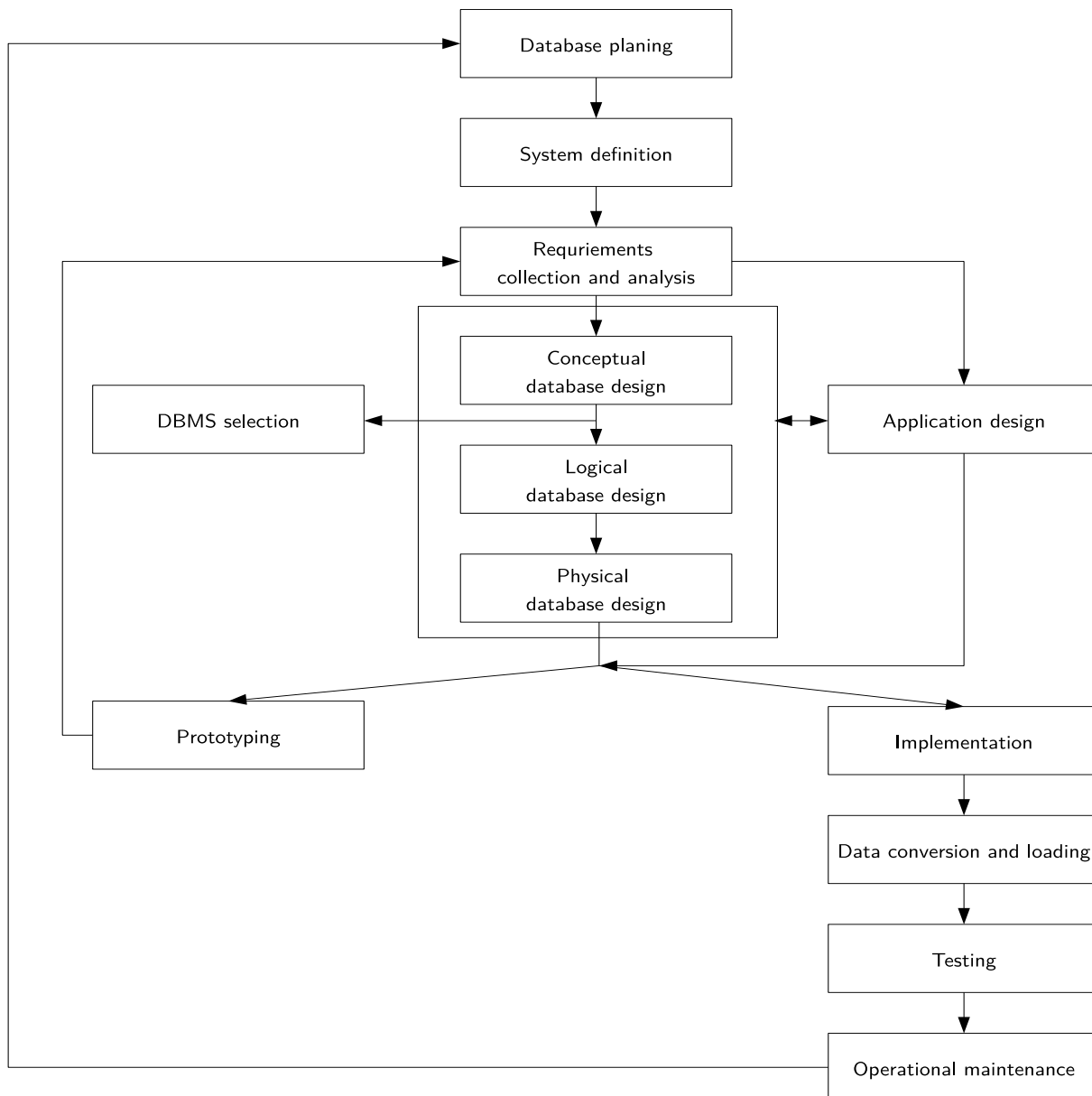


1 Database system development life cycle



Source:

Thomas Connolly, Carolyn Begg, Systemy baz danych, Wydawnictwo RM, 2004

Summary of the main activities involved in each phase of the database system development life cycle [3]:

- Database planing - planning the most effective methods of implementing life cycle phases.
- System definition - determining the scope and boundaries of a database system.
- Requiements collection and analysis - collection and analysis of requirements for the database system.
- Database design - conceptual, logical and physical database design.

- DBMS selection - selection of a DBMS appropriate for the database application.
- Application design - designing user interfaces and application programs that will use the database.
- Prototyping (optional) - building a working model of a database system that allows designers and users to visualize and evaluate how the final system will work and look.
- Implementation - creating database definitions and application programs.
- Data conversion and loading - transferring data from the old system to the new one.
- Testing - testing database application and checking compliance with user requirements.
- Operational maintenance - the database application is fully implemented. The system is monitored and maintained. As needed, new requirements are added to database system by going through the above phases again.

Design methodology - A structured approach using procedures, techniques, tools and documentation aids for the supporting and facilitating of the design process.

Conceptual database design [3] - the process of creating a model of the data used in the enterprise, independent of any details of the physical representation.

2 Entity-Relationship Model

Based on:

Thomas Connolly, Carolyn Begg, Systemy baz danych, Wydawnictwo RM, 2004

Entity-Relationship model [3] - Entity relationship modeling representing a top-down approach to database design. It begins by defining of entities and relationships between them. Then more and more details are added to the model, such as attributes constraints (conditions) regarding entities, relationships and attributes.

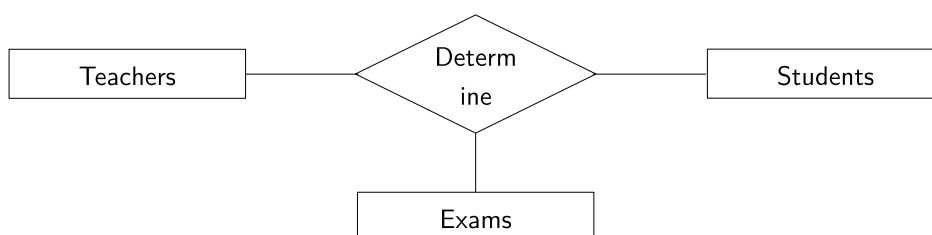
To present entity relationship model a unified modeling language UML was used.

Definitions [3]:

- Entity set - is a group of objects with the same properties that were considered as independent entities within a company.
- Strong entity - is an entity set whose existence does not depend on other entity sets.
- Weak entity set - is an entity set whose existence depends on other entity sets. An entity from such set can not be identifies using only attributes assigned to this entity set.
- Entity - is a unique and recognizable object from a entity set.
- Relationship - is a set of significant connections between entity sets.
- Recursive relationship - is a relationship in which the same set of entities occurs more than once in different roles, for example: for entity set Employees there can be relationship "Has boss".
- Instance of a relationship - is a unique and recognizable relationship between single entities from the entities participating in the relationship.

Students	Assigned to ▷	◁ Contains	Groups
studentID	12..32	1..1	groupID

- Degree of relationship - is a number of entity sets participating in it. In UML notation diamond symbol is used when degree is greater than 2, for example Teacher determines a date of Exam with Students.



- Attribute - is a specific feature of an entity or relationship, for example surname.
- Derived attribute - an attribute representing value, which is calculated based on other attribute or sets of attributes, from the same or other set of entities, for example age or number of students in a group.
- Attribute domain - a set of admissible values for at least one attribute.

Students
studentID
first_name
last_name

- Candidate key - the smallest set of attributes which uniquely identifies every entity in set of entities.
- Primary key (PK) - a candidate key chosen to uniquely identifies every entity in set of entities.
- Alternate key - a candidate key not chosen to be a primary key.
- Simple key - a key consisting of one attribute.
- Composite key - a key consisting of more than one attribute.
- Cardinality - a number or range of possible entities from one entity set which can be in a relationship with another entity from other or the same entity set.

Binary relationships:

- "one to one" (1:1),
- "one to many" (1:*),
- "many to many" (*:*)

3 Conceptual database design

Conceptual database design [3] - process of information model creation used in a company, independent of physical representation details.

Step 1

Create local conceptual data model for every perspective

Perspective in this context means area of application, point of view of user or user group.

Step 1.1

Specify entity sets

Can be done on the basis of user requirements specification (based on nouns or noun expressions).

After this step we should have entity sets description containing: entity set name, description, aliases (synonyms) and properties.

Step 1.2

Specify types of relationships

Can be done on the basis of user requirements specification (based on verbs or verb expressions).

Attention should be given to:

- degree of relationship (binary or higher),

- recursive relationship,
- cardinality,
- checking if relationships are a representation of real dependencies,
- fan traps and chasm traps,
- occurrence of each entity set in at least one relationship.

After this step we should get relationships descriptions containing: relationships names, entity set names, and relationship degree. It be presented as UML diagrams or text.

Fan traps [3] occurs when a model represents a relationship between entity sets, but paths instances between entities instances are not unambiguous.

Chasm traps [3] occurs when a model suggests a relationship between entity sets, but there are no paths connecting certain entities.

Step 1.3

Specify attributes of entity sets and relationships

Can be done on the basis of user requirements specification (based on nouns or noun expressions, which describe the properties of entities or relationships).

Attention should be given to:

- single and composite attributes,
- single-value and multi-value attributes,
- derived attributes,
- potential problems.

After this step we should get attributes description containing entity set name to which it should be assigned, attribute name, description, data type and its length, aliases (if exists), default value (if exists), simple attributes of which it is composed (if it is composite attribute), information whether it is an single-value and multi-value attribute, information whether it is a derived attribute and if so, how to determine its value.

Step 1.4

Specify attributes domains

After this step we should get domains description, a set of admissible values, admissible range of length, format of attributes, and optionally a set of admissible operations on attributes, list of attributes that can be compared with each other or used together.

Step 1.5

Specify candidate keys and primary keys

When choosing a primary key from candidate keys attention should be given to:

- key with the fewest attributes;
- a key whose values are changed least often;
- candidate key with the smallest number of characters or the smallest maximum value,
- a key that will be most convenient for users.

Other solution is adding a new attribute that uniquely identifies each entity.

After this step we should get information about candidate and primary keys. Not for every entity set a primary key can be determine now. For weak entity sets, primary keys can be determined only after step 2.2.

Step 1.6

Consider advanced modeling methods (optional step)

Enhanced Entity-Relationship model (EER) [3] it is an ER model extended with concepts such as specialization-generalization, aggregation and composition.

Superclass [3] - Is a set of entities containing at least one subgroup composed of their entities, which should be separately represented in a data model.

Subclass [3] - This is a subgroup of entities from an entity set, which should be separately represented in a data model.

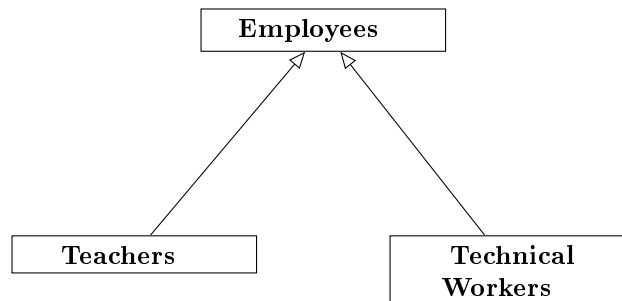
For example: employee (superclass) - teacher (subclass)

Specialization [3] - It is a process of maximizing differences between entity elements, realized by identifying their distinctive characteristics.

Example: Employees $\rightarrow$ Teachers, Technical Workers

Generalization [3] - It is a process of minimizing differences between entities, realized by determining their common features.

Example: Teachers, Technical Workers $\rightarrow$ Employee



Aggregation [3] - Represents relationships of the type 'is part' or 'has' between sets of entities in which one of participants is 'the whole' and the other 'part'.

Composition [3] - This is a special form of aggregation representing a relationship between entities, in which there is a strong relationship where 'the whole' 'own' 'parts' and compliance of periods of their existence.

Step 1.7

Verify redundancy occurrence in a model

In this step should be checked:

- check again one to many relationships, whether they represent different or the same objects,
- remove redundant relationships, which provide information that may be obtained in different way.

Step 1.8

Verify realization of transactions in a local conceptual model

In the first method description of transactions requirements is prepared and checked whether all information necessary to perform them, such as entities, relationships and attributes, are in the model.

In the second method, paths representing transactions are marked on entity diagram. This make possible to find elements used in transactions and other which are never used.

Usually, problems are caused by omission of entity set, relationship or attribute.

For example simple transaction like calculating number of students in every require two entity sets: Students and Groups and relationship between them.

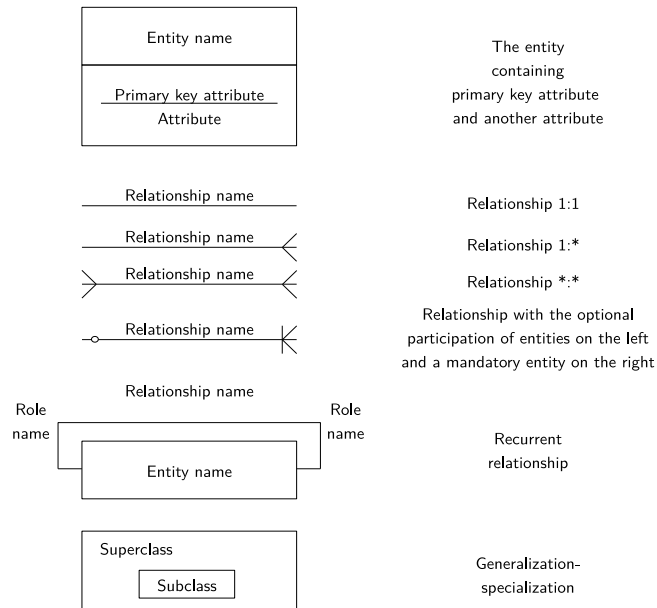
Students	Assigned to ▷	◁ Contains	Groups
studentID	12..32	1..1	groupID

Step 1.9

Discuss local conceptual data model with user

The user should verify the correctness of the implementation of the local conceptual data model, which consists of the ER diagram and documentation.

4 'Crow's foot' notation



'Crow's foot' notation . Thomas Connolly, Carolyn Begg, Systemy baz danych, Wydawnictwo RM, 2004

References

- [1] Jeffrey D. Ullman, Jennifer Widom, "Podstawowy wykład z systemów baz danych", WNT, 2001
- [2] Jeffrey D. Ullman, Jennifer Widom, "Systemy baz danych, Pełny wykład", WNT, 2006
- [3] Thomas Connolly, Carolyn Begg, "Systemy baz danych", Wydawnictwo RM, 2004
- [4] Kevin Loney, Bob Bryla, Oracle Database 10g, Podręcznik administratora baz danych. Helion 2008
- [5] Scott Urman, Ron Hardman, Michael McLaughlin, "Oracle Database 10g, Programowanie w języku PL/SQL", Helion, 2008
- [6] Price J, "Oracle Database 11g i SQL Programowanie", Helion, 2009
- [7] Ramez Elmasri, Shamkant B. Navathe, "Wprowadzeni do systemów baz danych", Helion, 2005
- [8] Sharon Allen, "Modelowanie danych", Helion, 2006
- [9] Jose A. Ramalho, Oracle 8i, Mikom, 2001
- [10] Dokumentacja do bazy danych Oracle dostępna na stronie www.oracle.com
- [11] Sadalage P. J., Fowler M., "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence", Addison-Wesley Professional, 2012
- [12] Michael Stonebraker, "New Sql: An Alternative to Nosql and Old Sql For New Oltp Apps", June 16, 2011

Materials for the subject are available at: <http://weaii-moodle.tu.kielce.pl/>