

# Lecture 3

## Topics of the lectures

- The conceptual model
- Subtypes and supertypes
- Specificity /concrete definition/
- Synthesis /generalization/
- Data structure in an RDBMS
- Relational algebra
- Major relational operators:
  - select /sample/
  - project
  - Cartesian /cross product/
  - union
  - intersection
  - difference /subtraction/
  - compound /left, right, full join/
  - division

# Questions to the previous lecture

## □ **Data Model**

- What data models do you know?
- Which model is historically the first?
- Which model is the fastest data processing?
- Which model is used for the analytic representation?
- What models are based on strings, records?
- Which model of the connection can't be represented as a many-to-many?
- Which model is the most advanced? Why is it so popular?

## **What can you say about each stage of the database design?**

- Preliminary design
  - Analysis of feasibility
  - Determine requirements
  - Conceptual design
- Implementation
  - Testing and maintenance of the database.

# conceptual design

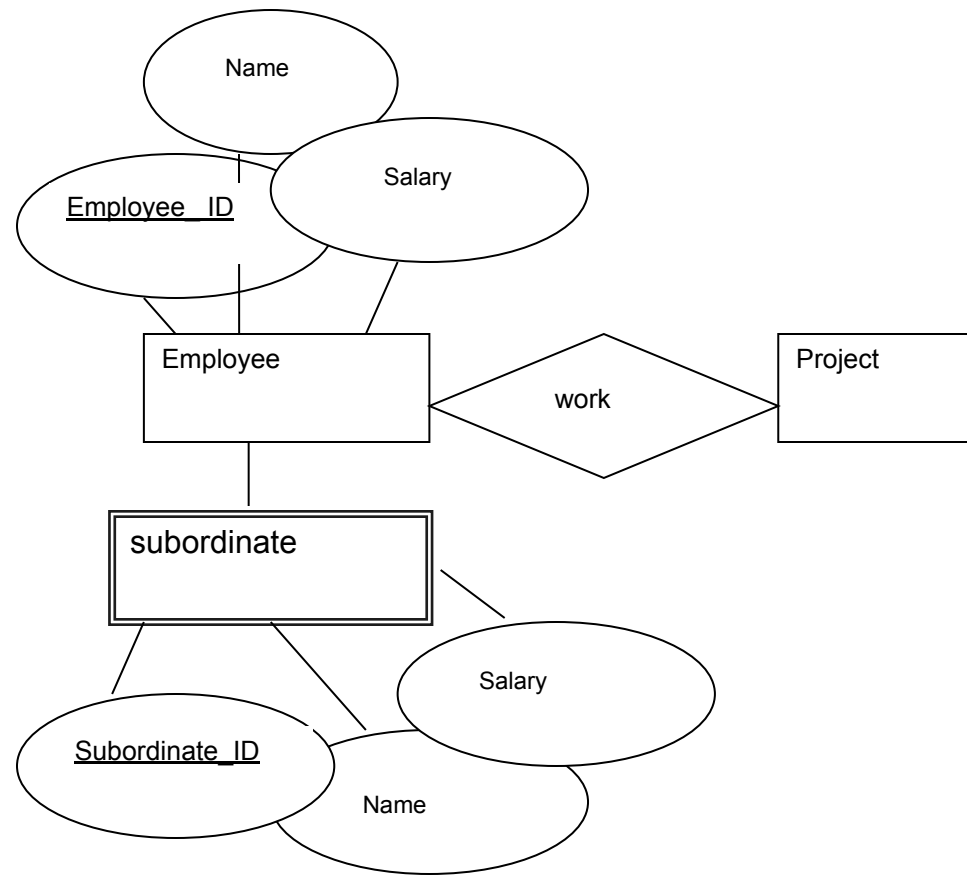
- ❑ Conceptual model reflects the entities and relationships between them in relation to the needs of the organization of data processing.
- ❑ Conceptual model can be converted into a relational, hierarchical or network model.
- ❑ The conceptual model is independent of individual applications, database management systems, hardware and physical storage method.
- ❑ Data analysis is the first step in the development of the conceptual model, and it begins with the collection of data.
- ❑ Data analysis includes the determination of entities, their attributes and the relationships between them on the basis of the data collected.
- ❑ The next step is to check all of the operational use of the organization associated with their treatment, and avoid unnecessary or duplicate data.
- ❑ After completing the analysis of the data, you draw a diagram of the "entity - relationship." This scheme provides an intuitive overview of the project and is particularly useful for the exchange of ideas among the users.

# Model "Entity-Relationship"

- There are a variety of object-oriented models. The most widely used model is the "entity - relationship" (ER model).
- Model "entity - relationship" is based on a realistic view which encompasses a set of objects or entities and their relationships.
- Schema components of ER are:
  - entity ;
  - connection;
  - attributes.

# entity

- The entity is any object, place, person, or action, details of which are recorded.
- Entities are represented as rectangles, on which are written the names assigned to them.
- There are two types of entities:
  - Dependent /weak/;
  - Independent /regular entity/.
- Affiliated entities are also referred to as weak entities, and independent - regular entities.
- Weak entity represented by a rectangle outlined by the double line.



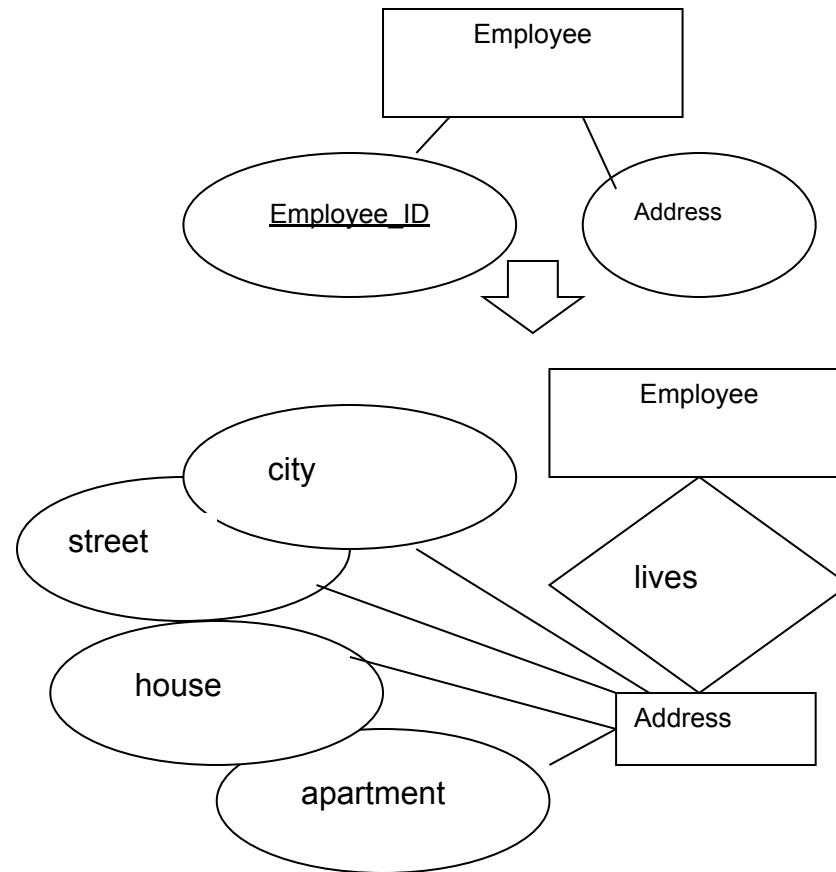
An example of a weak entity

# connection

- Combining entities are called connection.
- Connection is depicted in the form of rhomb with the name of the link.
- Can attach an entity to itself.
- Between the same entities may also be multiple connections.
- Connections are of three types:
  - one-to-one;
  - one-to-many;
  - many-to-many.

# attributes

- Attribute called property of this entity.
- Attributes are represented as ellipses, equipped name properties.
- Key attributes are underlined.
- Connection can also have attributes.



Attributes can be entities

## **Simple and composite attributes:**

- Simple attribute - an attribute that consists of a single component with an independent existence.
- Compound attribute - an attribute that consists of several components, each of which is characterized by an independent existence.

## **Unambiguous and multi-valued attributes:**

- Unambiguous attribute - the attribute that contains one value for each entity instance certain type.
- Multi-valued attribute - the attribute that contains multiple values for each instance of a certain type of entity.

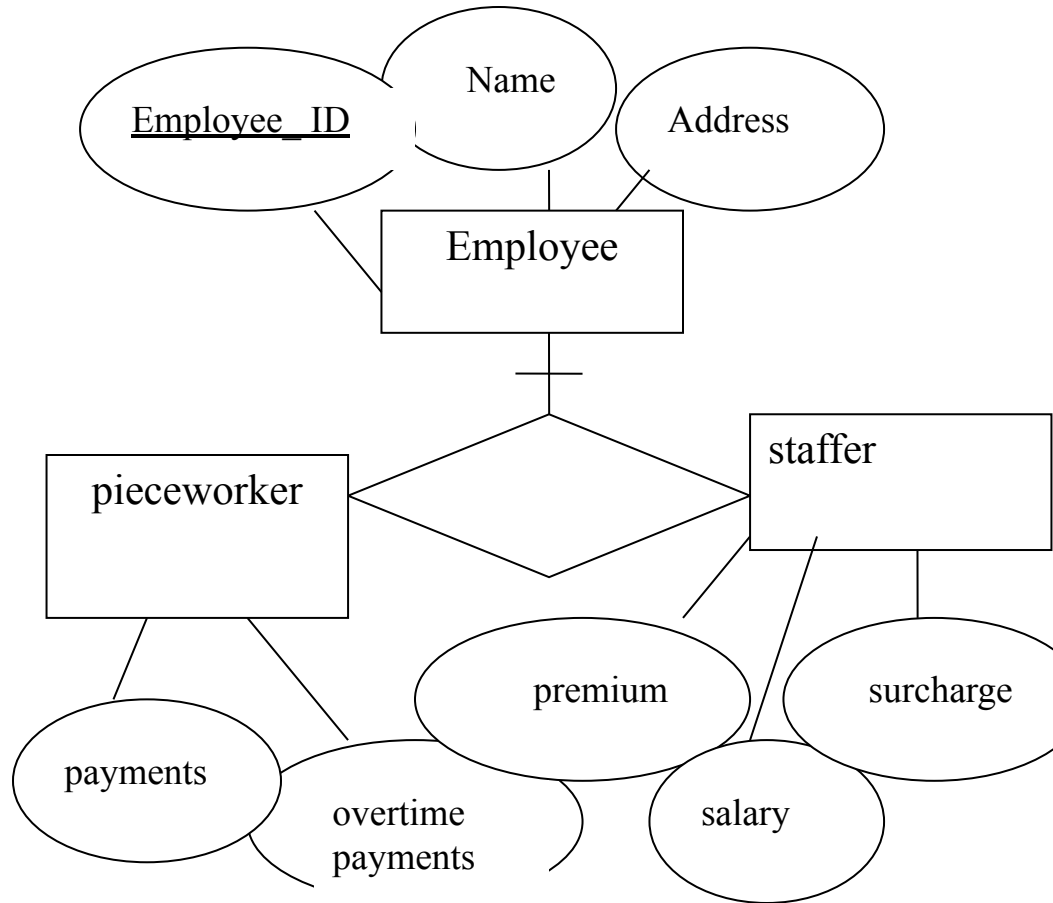
## **Derived attributes:**

- Derived attribute - an attribute that represents a value derived from the value of the associated attribute or a set of attributes that belong to some (not necessarily this) type of entity.



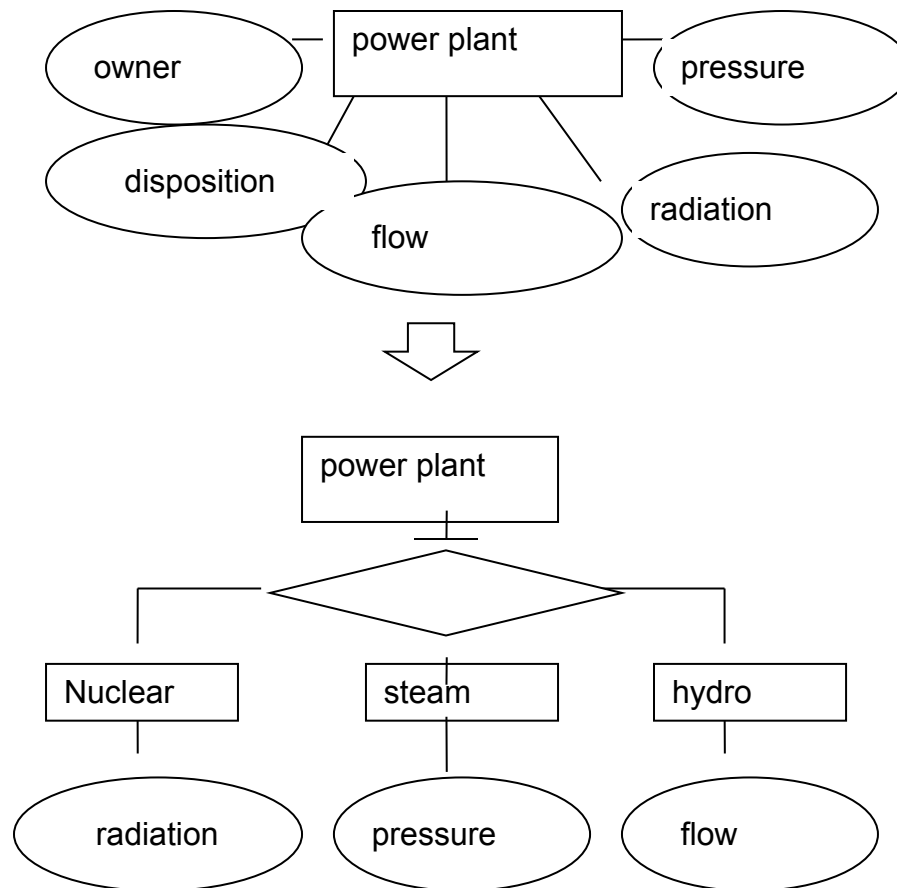
# Subtypes and supertypes

- Subtype is a subset of another entity. The existence of a subtype is always dependent on the supertype.



# Specification / concrete definition/

- Specification is result of the determination a subset from the entity set of a high-level to form low-level entity set.

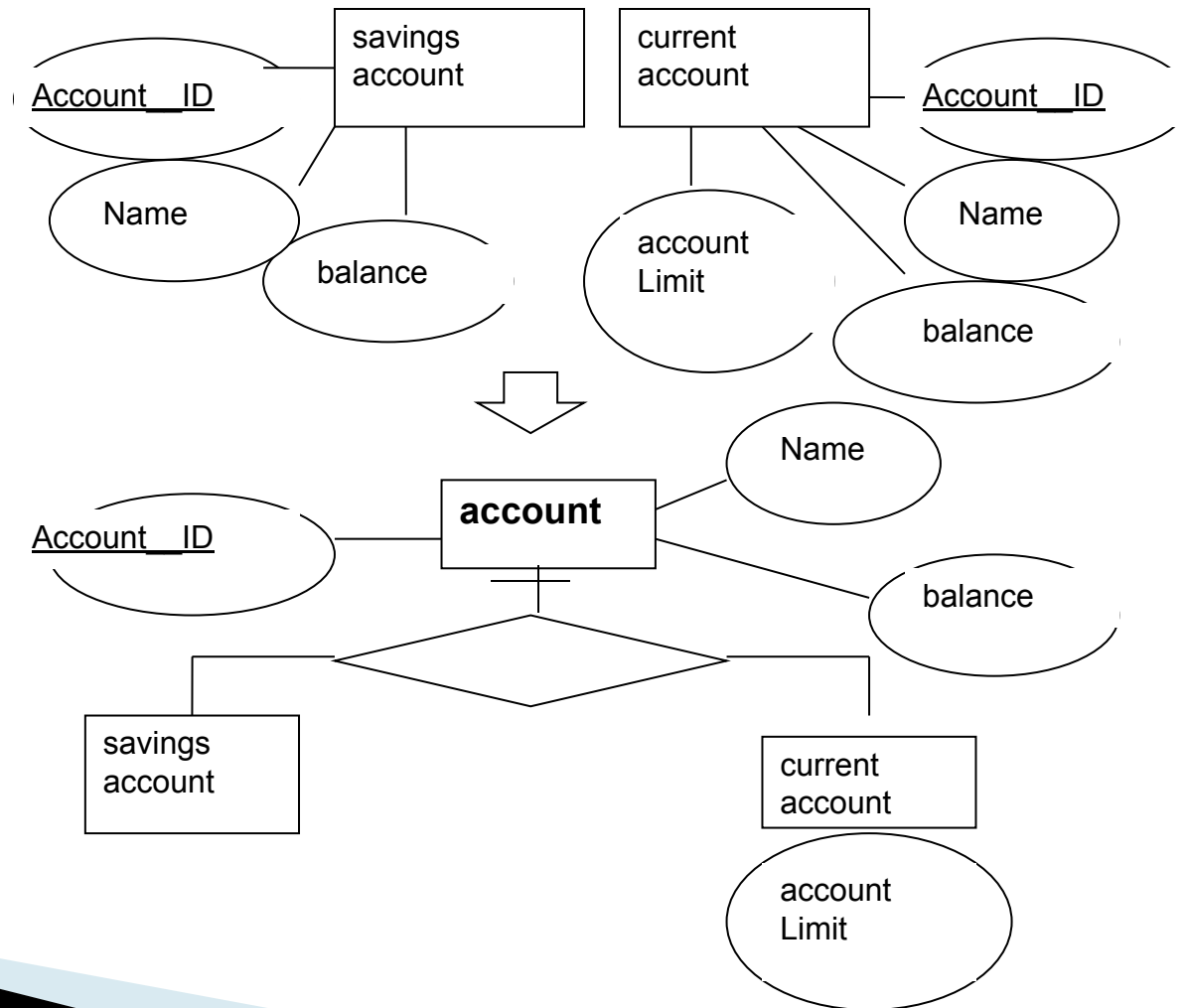


# Synthesis /generalization/

- Generalization is the result of combining two or more low-level entity sets to create higher level entities set.

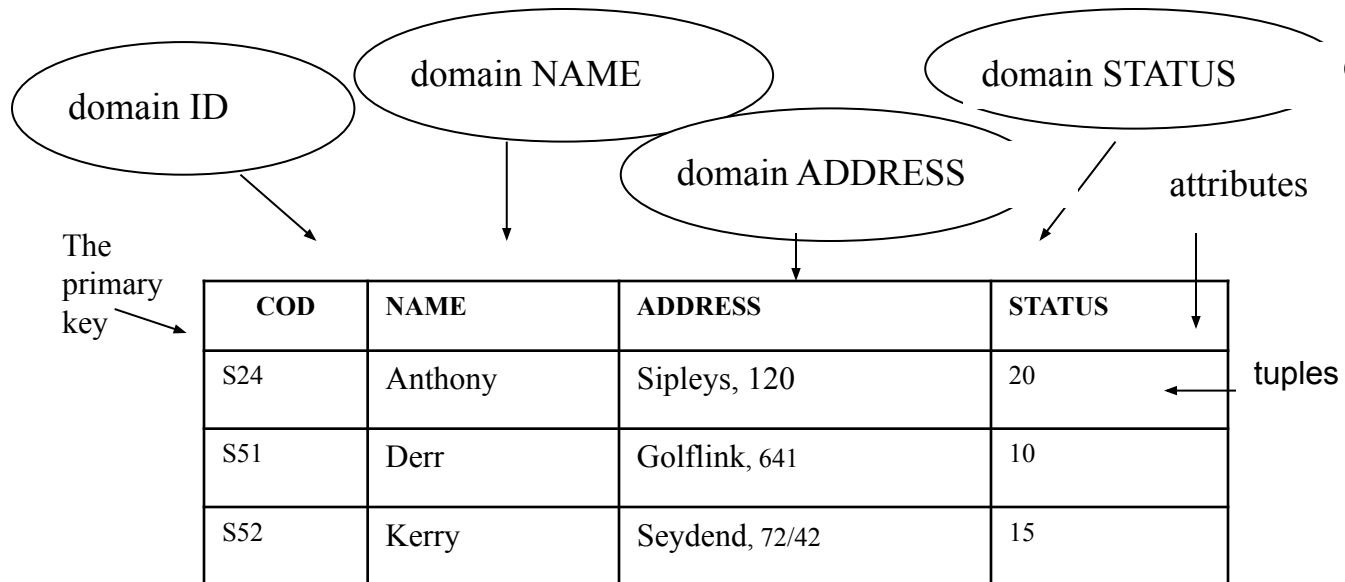
Synthesis - a method opposite specification.

Each entity in the generalization of high-level and should also be the entity of a lower level.




# The relational model

- ❑ Relational DBs were launched in 1970 with the publication of the article "A Relational Model of Data for Large Shared Databanks" by E.F. Codd.
- ❑ Codd defined the basis for relational DB theory, and provided the set-theoretic relational algebra for manipulating such DBs.
- ❑ Relational DBMSs largely won the competition for the DBMS layer, and most popular DB products.
- ❑ Data structure in an RDBMS.

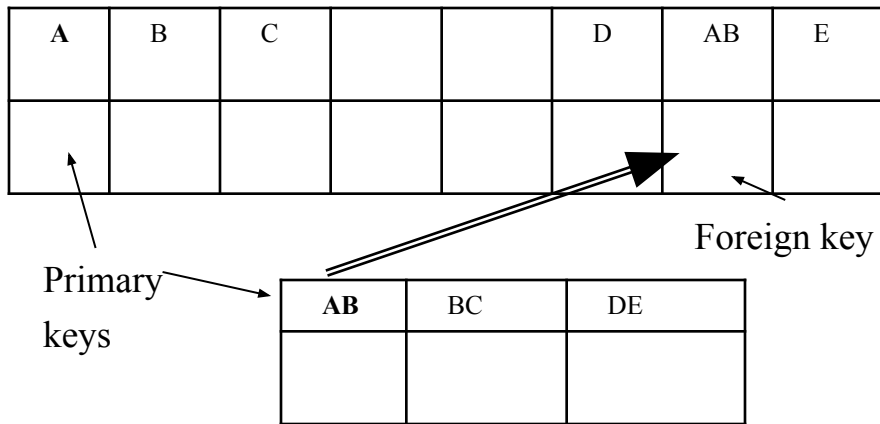


# Data structure in an RDBMS

- ❑ Organizing principle of the relational database is a table showing the data values are placed at the intersections of row-column. Each table in the database has a unique name that identifies its contents. The table is called a relation.
  - ❑ Relation is a set of elements called tuples. Visual form, a relationship is a familiar human readable two-dimensional table.
  - ❑ Table has rows (records) and columns. Each row has the same structure and is made up of fields. Lines correspond to the tuples, and columns - the attributes of the relationship.
  - ❑ Because, in a single table can't be described more complex logical data structure of the domain, use the binding table.
  - ❑ Each table should have a column or combination of columns that uniquely identify each row in the table. This column (or columns) is the primary key.
  - ❑ A domain is a group of values from which one or more attributes (columns) calculate their actual values.
- 

# keys

Parent-child relationships in the relational model



There are different types of keys:

- primary;
- foreign;
- candidate;
- alternative;
- composite.

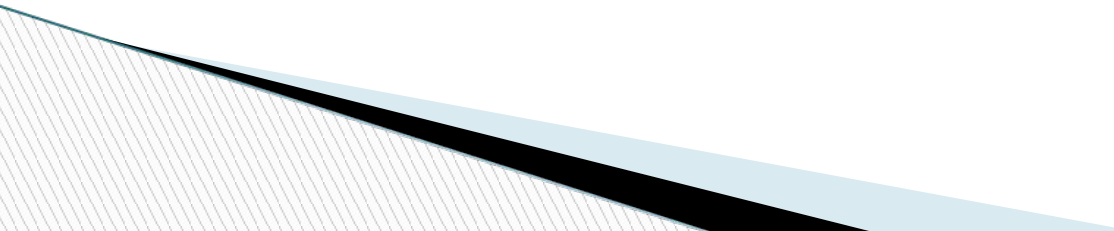
- ✓ Any attribute (or set of attributes) that uniquely identifies a row in the table can be a primary key. Such an attribute is called a candidate key.
- ✓ One of the possible options /candidate/ is chosen to be the primary key based on its prevalence, increasing use, etc.
- ✓ Attribute, which is a candidate /possible key/, but not the primary, called alternative key.
- ✓ If the key that uniquely identifies a row in the table consists of more than one attribute, it is called a composite key.
- ✓ Foreign keys always display connection.

# Relational algebra

- The relational model is based on the principles of relational algebra.
- Relational algebra is a set of operators that work with relationships.
- Each operator takes one or two relations as input and returns a new relation on the output.

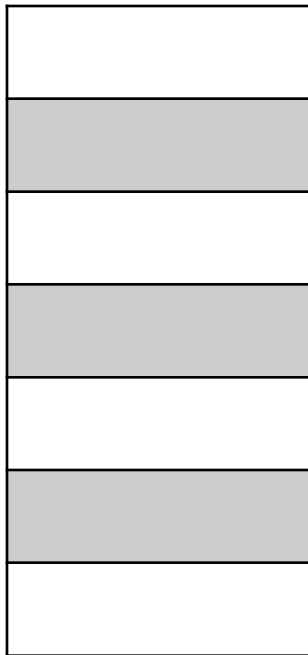
## Relational algebra operators

We give a symbolic representation, and a brief description of the major relational operators:

- select /sample/
  - project
  - Cartesian /cross product/
  - union
  - intersection
  - difference /subtraction/
  - compound /left, right, full join/
  - division
- 

# sample /sampling /

"Sampling" retrieves tuples and strings.



The operator selects the sample tuples or rows from relation, based on the condition.

The table has the attributes of students "list\_number" (number on the list), "students NAME", "age" and "gender".

Condition is the selection of tuples only those students older than 25 years.

The resulting relation is:

<b>list_number</b>	<b>students NAME</b>	<b>age</b>	<b>gender</b>
0910	Anthony	26	M
0976	Sarah	28	Ж

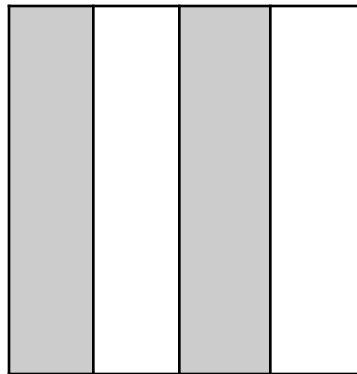
An example of the result set



# projection /plane/

Projection operator selects the attributes or columns of the relation. If you want to retrieve only the name and age of the students, the resulting relation is as follows (assuming that the table contains a total of six students of tuples):

"Projection" retrieves the attributes or columns



<b>students NAME</b>	<b>age</b>
Jerry	20
Susan	23
Nancy	21
Anthony	26
Raimi	24
Sarah	28

Example of the result of the projection

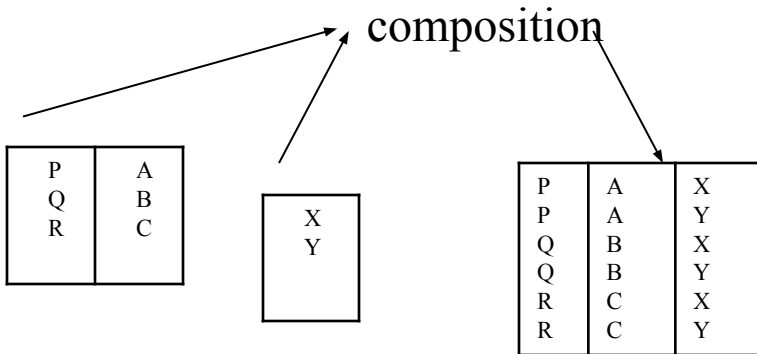
# composition /cross join/

- It consists of all possible combinations of tuples, taken one by one from each of the two relations.
- Example output products

For compatibility, the two tables must have common attributes.

Product operator provides the Cartesian product of two tables.

For example, consider the following two tables.



Cartesian product of the tables is all possible combination of tuples.

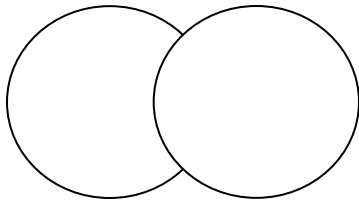
CODE FOR TEACHERS	NAME
I1001	Nancy Matthews
I1002	Catherine
I1003	Mack Thames

GROUP CODE	CODE FOR TEACHERS
B001	I1001
B002	I1002
B003	I1003

CODE FOR TEACHERS	NAME	GROUP CODE
I1001	Nancy Matthews	B001
I1001	Nancy Matthews	B002
I1001	Nancy Matthews	B003
I1002	Catherine	B001
I1002	Catherine	B002
I1002	Catherine	B003
I1003	Mack Thames	B001
I1003	Mack Thames	B002
I1003	Mack Thames	B003

# union

- Union operator creates relations of tuples contained in each or either of the relation.



union relations

Consider two tables A and B.

"A" contains the numbers on the list and the names of students whose main subject is Computer science.

"B" contains the numbers and names of all the students, the principal of which is the discipline of mathematics.

These tables are compatible combinations, so they can use the union operator.

A

<b>number on the list</b>	<b>students NAME</b>
0910	Anthony
0856	Nancy

B

<b>number on the list</b>	<b>students NAME</b>
0856	Nancy
0976	Susan

For compatibility, the two tables must have the same attribute types (sets of columns that have the same data type).

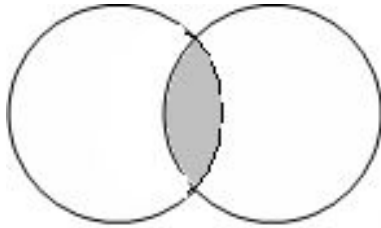
AB

<b>number on the list</b>	<b>students NAME</b>
0910	Anthony
0856	Nancy
0976	Susan

Example of the result of "unity"

# intersection

Intersection operator creates a relation consisting of tuples belonging to both relations.



"Crossing" of relations

Consider Table "A" and "B". Nancy examines two main disciplines. So her name appears in both tables. Intersection operator Tables A and retrieves the string that is common to both tables. Intersection operator works on tables that are compatible for the union.

A

<b>list_number</b>	<b>students NAME</b>
0910	Anthony
0856	Nancy

B

<b>list_number</b>	<b>students NAME</b>
0856	Nancy
0976	Susan

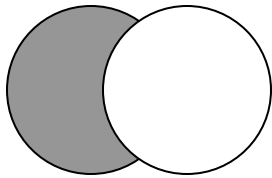
AB

<b>list_number</b>	<b>students NAME</b>
0856	Nancy

Example of the result of "intersection" of relations

# exception /subtraction/

The subtraction generates relation of tuples belonging to the first, but absent in the second of the two relations.



Subtraction operator also works for tables that are compatible combination /union, intersection /. In the case of tables “A” and “B” operation "A subtraction in" all of the rows that belong to “A”, but not in “B”.

"Subtract" relations

A

list_number	students NAME
0910	Anthony
0856	Nancy

B

list_number	students NAME
0856	Nancy
0976	Susan

AB

list_number	students NAME
0910	Anthonxy

Example of the result of "subtraction" of relations

# compound /left, right, full join/

- Join operator forms the attitude of the two relations.
- The operator forms the relation from two relations, consisting of all possible tuples combination, taken in pairs from each relation and satisfied the condition.
- Join operator requires a common attribute.

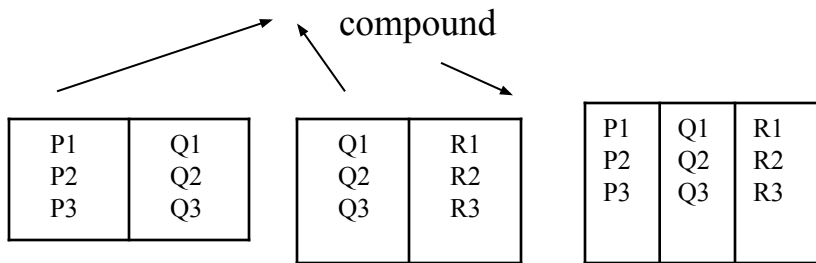


Table **A** contains the directory numbers of students and codes of courses they attend.

Table **B** contains the identification numbers of teachers and codes of courses that they taught.

"Compound" relations

**A**

number on the list	course code
0910	A21
0856	D21
0976	C67
0768	D21
0752	C67

**B**

teacher code	course code
0081	A21
0075	D21
0002	H42
0075	C67
0052	A21

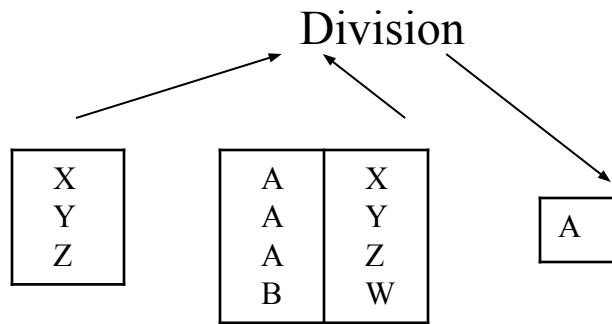
**AB**

number on the list	teacher code	course code
0910	0081	A21
0856	0075	D21
0976	0075	C67
0768	0075	D21
0910	0052	A21
0752	0075	C67

Output compound

# division

- The division operator takes two relations and build the third relation consisting of the values of the attributes of one relationship, coinciding with all the attribute values from the second relationship.
- Division operation is the inverse of the composition operation.



"Division" of relations

## employee

NAME	CITY
Anthony	New York
Anthony	California
Anthony	Washington
Nancy	Los Angeles

## city

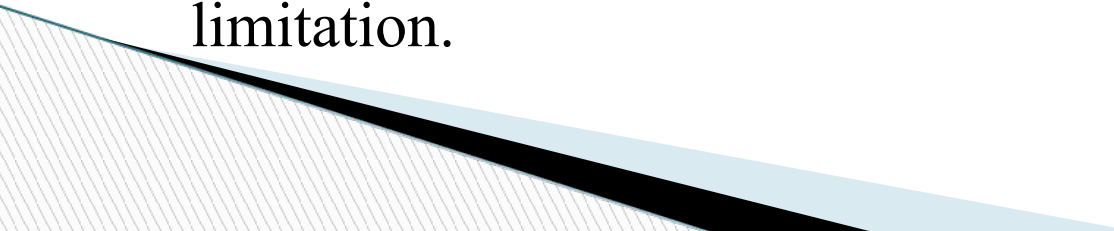
CITY
New York
California
Washington

As a result, the division:

NAME
Anthony

Example of an operator "division"

# Tips on logical database design

- ❑ Do not enter the attributes that are not necessary.
  - ❑ In the process of designing a database of some attributes may require additional attributes to clarify, and they become entities.  
To represent important recurring attribute groups you can create a new entity.
  - ❑ Concretization the result of view of a subset of the set of entities in the form of a high-level entity set low.
  - ❑ The union results from a merger of two or more sets of entities to create a low-level entity set high.
  - ❑ The union simplifies the multiple references.
  - ❑ By combining a high level of each entity should be the essence of both low. However, the specification does not have this limitation.
- 



# Conclusion

- ❑ Conceptual model reflects the entity and their connections.
- ❑ The conceptual model is independent of the system in which it is proposed for implementation.
- ❑ Regular entities are independent. They can exist in isolation, independently of any other entity.
- ❑ Each entity is displayed table. Each attribute in the diagram E/R is shown in the attribute table.
- ❑ Entities with common attributes be merged. Attributes may require additional attributes, and they become entities.
- ❑ Mapping relations depends on the type of communication. Depending on the relational database systems, each of the types of links to tables set in different ways.
- ❑ Called a weak entity whose existence is dependent of any other entity.
- ❑ Subtype is a subset of another entity. The existence of a subtype is always dependent on the supertype.
- ❑ Optional attributes should be replaced by sub-entity. This operation is called a specialization.
- ❑ To simplify the multiple references must introduce a new super-entity. This operation is called the union.