Relational databases: fundamental concepts

Agenda

- Overview of Relational Databases
- Relational Database Management Systems (RDBMS)
- RDBMS Architectures

Agenda

- Relational Database Schema
- Relational Database Design
- The Structured Query Language (SQL)

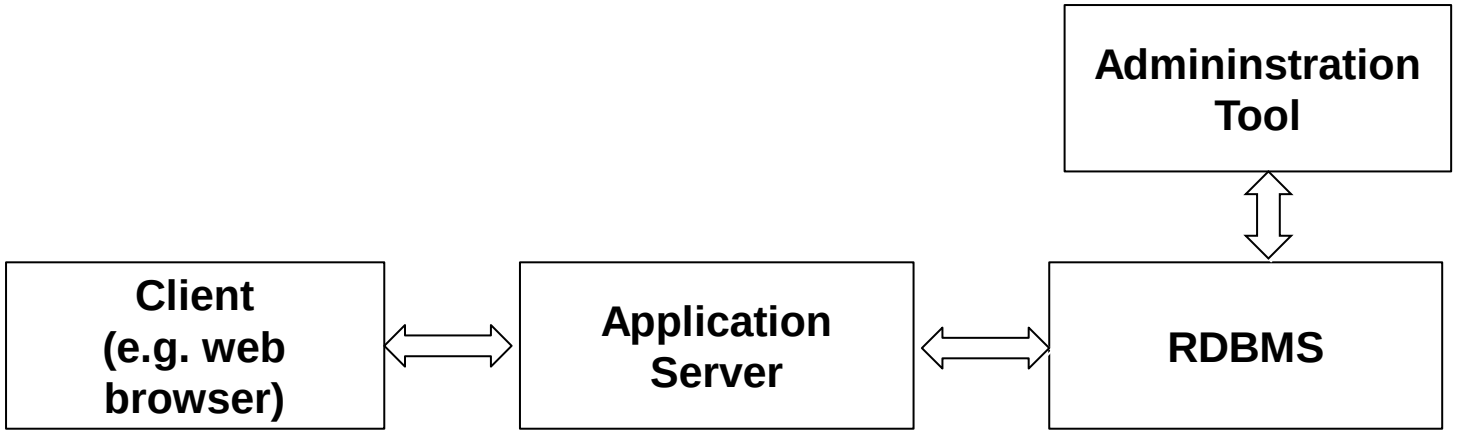
Overview of Relational Databases

Overview of Relational Databases

- Relational databases:
  - are systems that provide efficient, reliable, convenient and safe multi-user storage of and access to massive amounts of data
  - have a strong mathematical foundation: relational algebra
  - provide strict schema for storing the relational data
  - store data in database tables (relations) that has a set of attributes (columns) with a type (domain) and rows (tuples) that have values for each column
  - Store additional objects such as views, indexes, constraints, sequences, procedures, functions and others (most of them are covered in the course)

Overview of Relational Databases

- Relational databases:
  - are key component of many enterprise systems



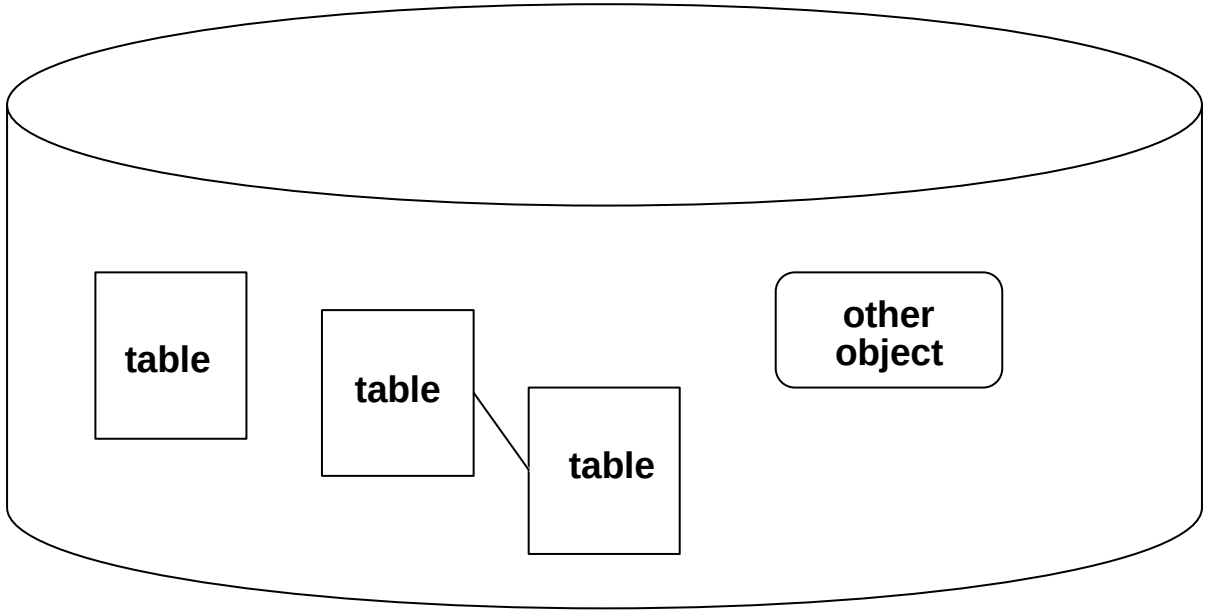
Overview of Relational Databases

- Example database table:

Id	Name	Family	Employer
1	Bill	Edwards	Company A
2	Ivan	Ivanov	Company B
3	Martin	Toshev	Company C

Overview of Relational Databases

- Example database:



Relational Database Management Systems

Relational Database Management Systems

- Relational Database Management Systems (RDBMS):
  - are also known as "database servers"
  - provide mechanisms for storing (persisting) data in a relational database
  - provide frameworks for manipulation of databases and database data - such as create/read/update/delete (CRUD) operations on database data and database configuration
  - Have different aspects such as: DBMS implementation, database design, database administration, database manipulation and database application development

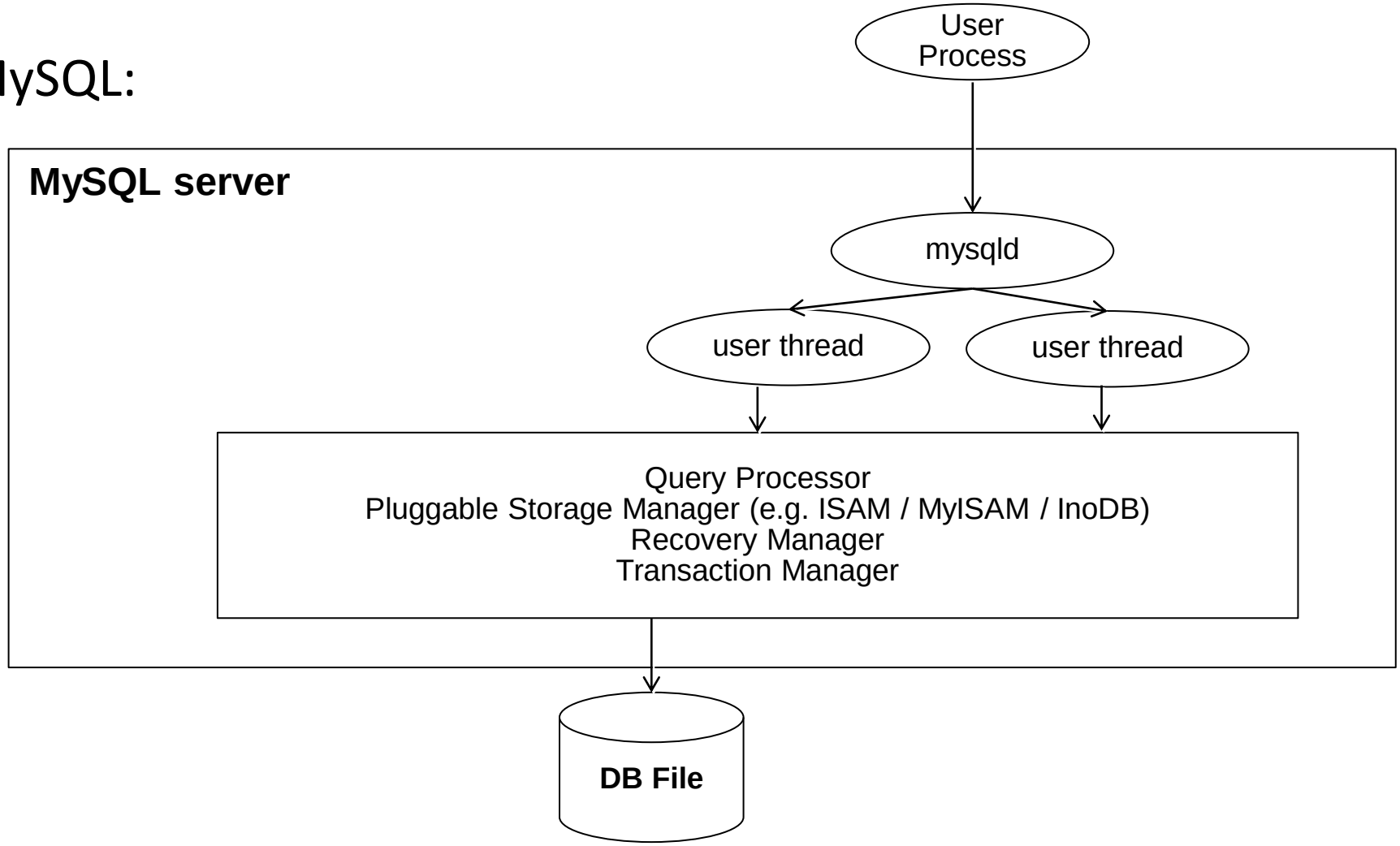
Relational Database Management Systems

- Famous RDBMS implementations:
  - MySQL (used throughout the course)
  - Microsoft SQL Server
  - IBM DB2
  - PostgreSQL
  - Borland Interbase

RDBMS Architectures

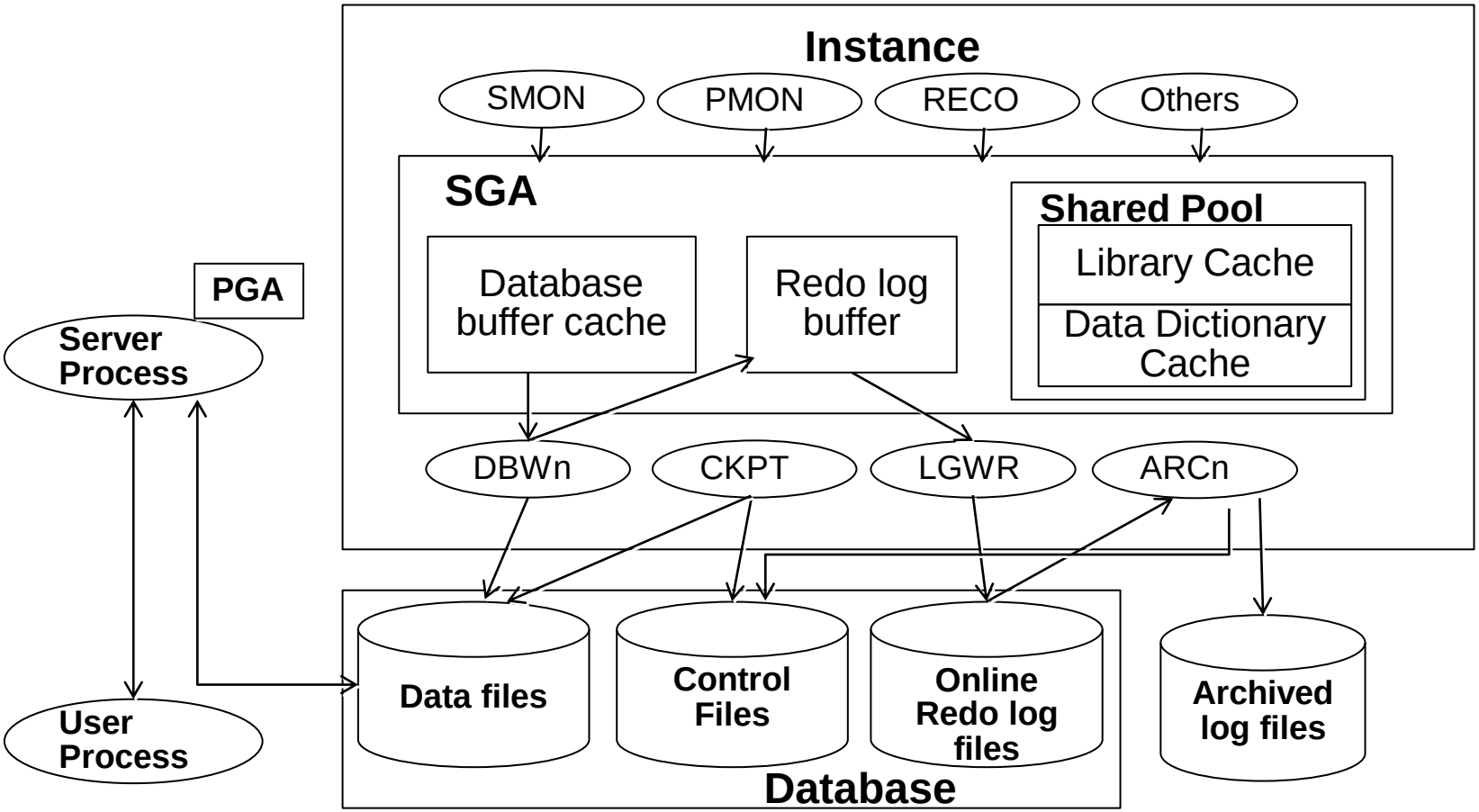
RDBMS architectures

- MySQL:



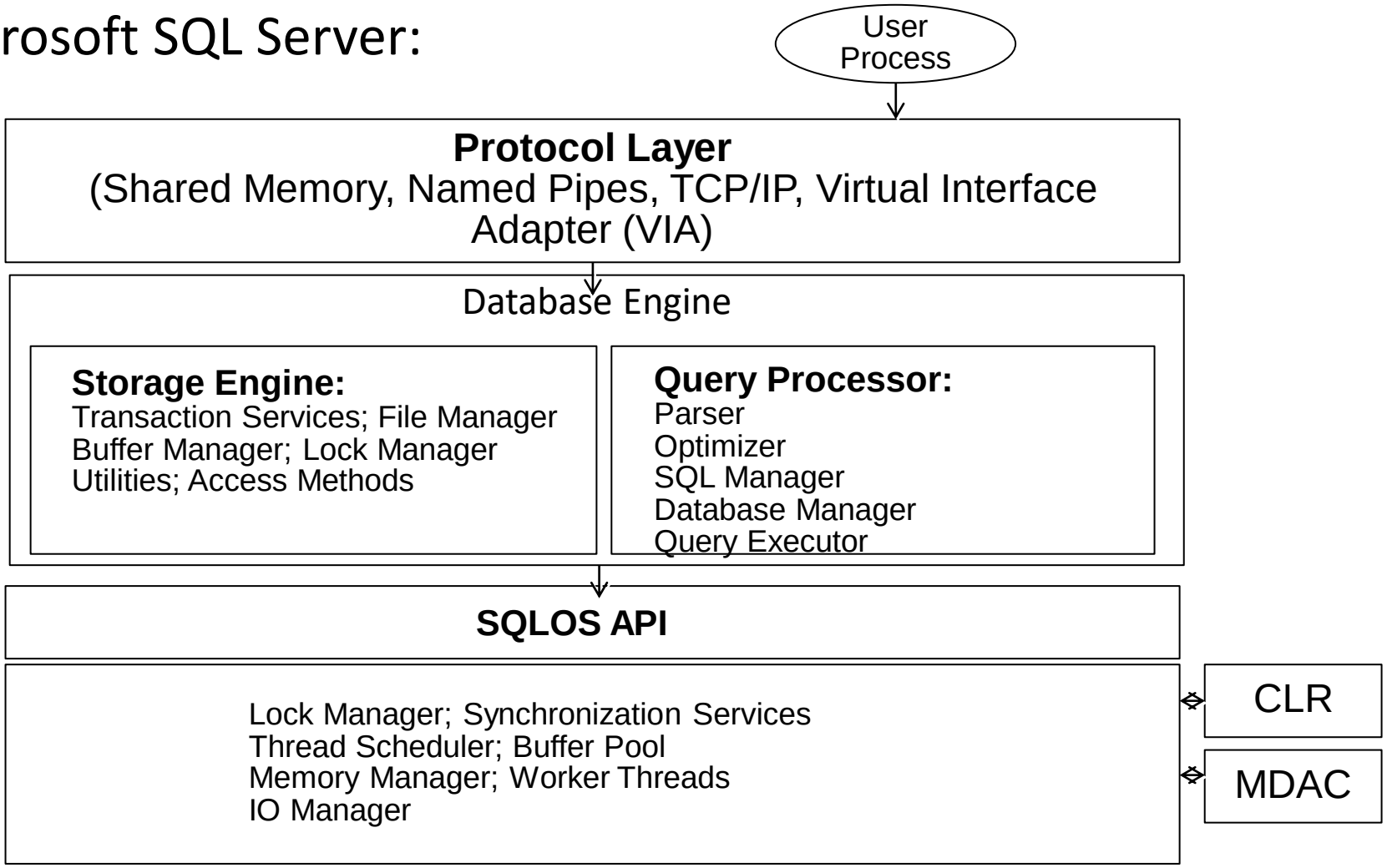
RDBMS architectures

- Oracle RDBMS:



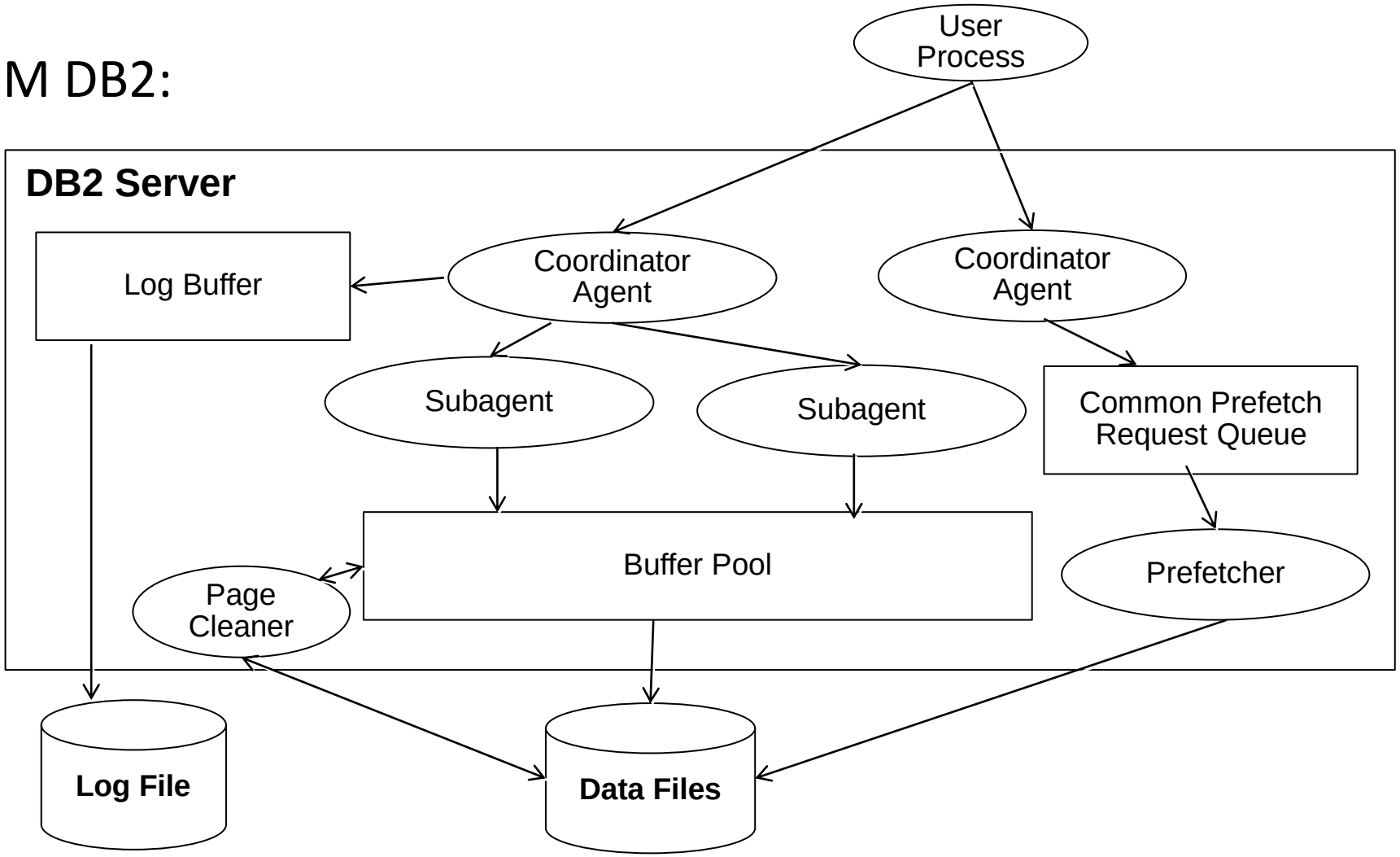
RDBMS architectures

- Microsoft SQL Server:



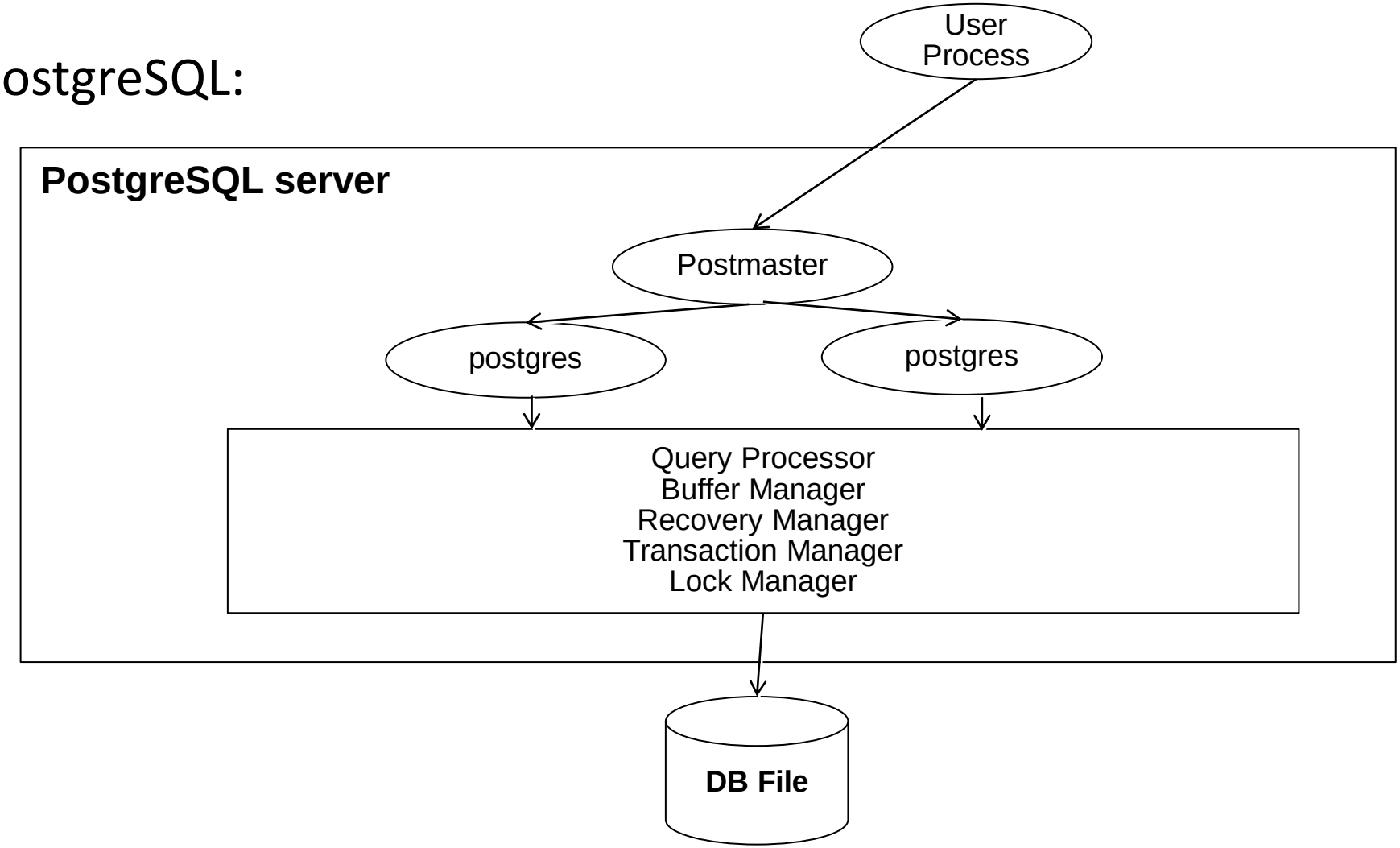
RDBMS architectures

- IBM DB2:



RDBMS architectures

- PostgreSQL:



Relational Database Schema

Relational Database Schema

- Describes the structure of a database in a formal way
- Provides definitions for tables, fields, relationships, views, indexes, packages, procedures, functions, queues, triggers, sequences, synonyms and other objects
- Defines constraints over the database objects
- In many cases normalization is a key requirement for a database schema

Relational Database Schema

- Sample table schema:

```
PEOPLE (
   id: number,
   name: string,
   family: string,
   employer: string
)
```

Relational Database Schema

- Constraints provide restrictions on the columns of a table
- Can be defined either as part of the table definition or separately
- Types of constraints:
  - not-null constraint - a value cannot be NULL
  - primary key constraint - a set of columns that identifies uniquely each row; defined over some of the columns; values cannot be NULL; is at most one
  - unique key constraint - a set of columns that must be unique for each row; values can be NULL

Relational Database Schema

- Types of constraints:
  - foreign key constraint - a set of columns in one table references a set of columns (a unique or primary key) in another table
  - check constraint - values in a column meet some condition

    For example:
    - `(hour>=0) AND (hour<=24)`
    - `name = upper(name)`

Relational Database Schema

- Primary key is a set of columns in a table, that can be used to uniquely identify its rows

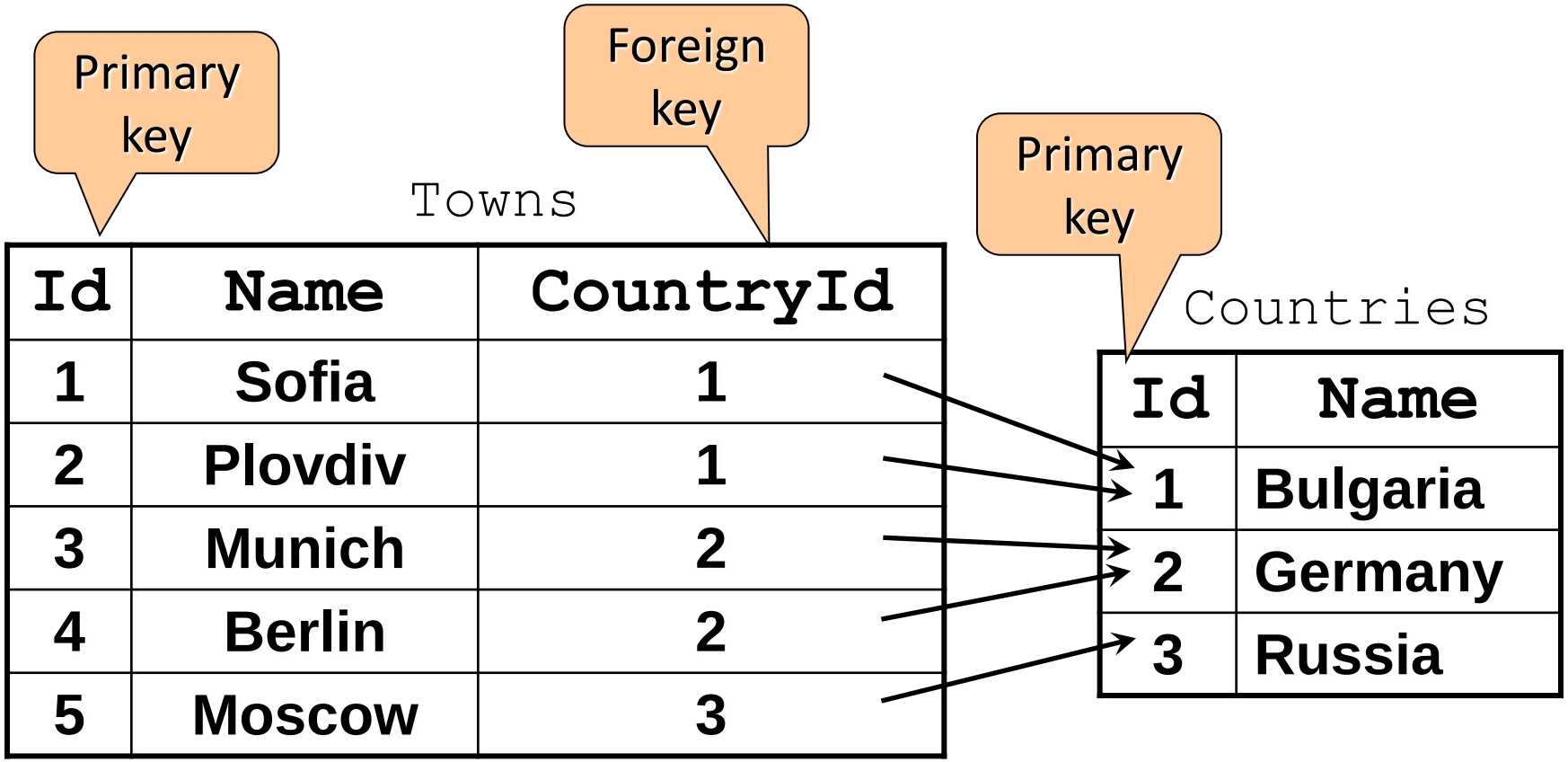
Primary key

Id	Name	Family	Employer
1	Bill	Edwards	Company A
2	Ivan	Ivanov	Company B
3	Martin	Toshev	Company C

- If the primary key is composed more than one column then it is called composite primary key

Relational Database Schema

- Relations between tables are defined using foreign keys



Id	Name	CountryId
1	Sofia	1
2	Plovdiv	1
3	Munich	2
4	Berlin	2
5	Moscow	3

Id	Name
1	Bulgaria
2	Germany
3	Russia

Relational Database Schema

- By using relations we avoid repeating information in our database (in the example the name of the country is not repeated for every town)

- Relations have multiplicity:
  - 1 x many (e.g. country / town)
  - many x 1 (e.g. town / country)
  - many x many (e.g. student / course)
  - 1 x 1 (human / student)

Relational Database Schema

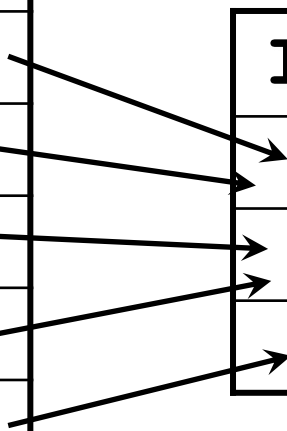
- In a relation of type 1 x many one record in the first table has many corresponding records in the second one (many x 1 is the opposite).

Towns

Id	Name	CountryId
1	Sofia	1
2	Plovdiv	1
3	Munich	2
4	Berlin	2
5	Moscow	3

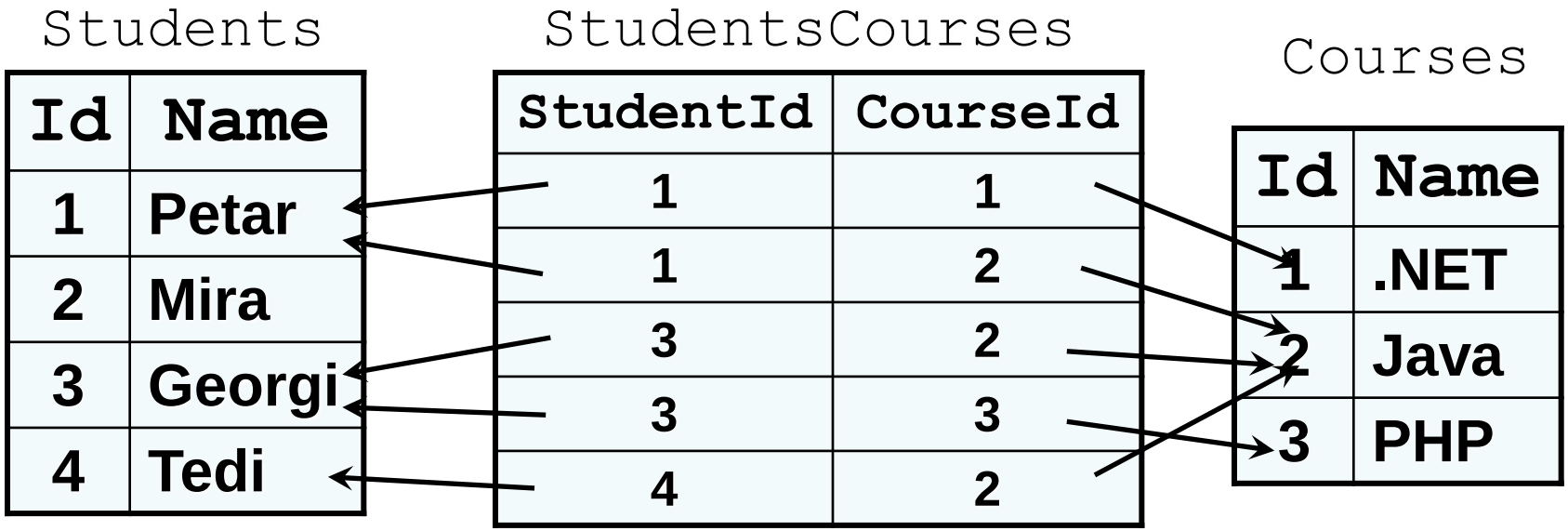
Countries

Id	Name
1	Bulgaria
2	Germany
3	Russia



Relational Database Schema

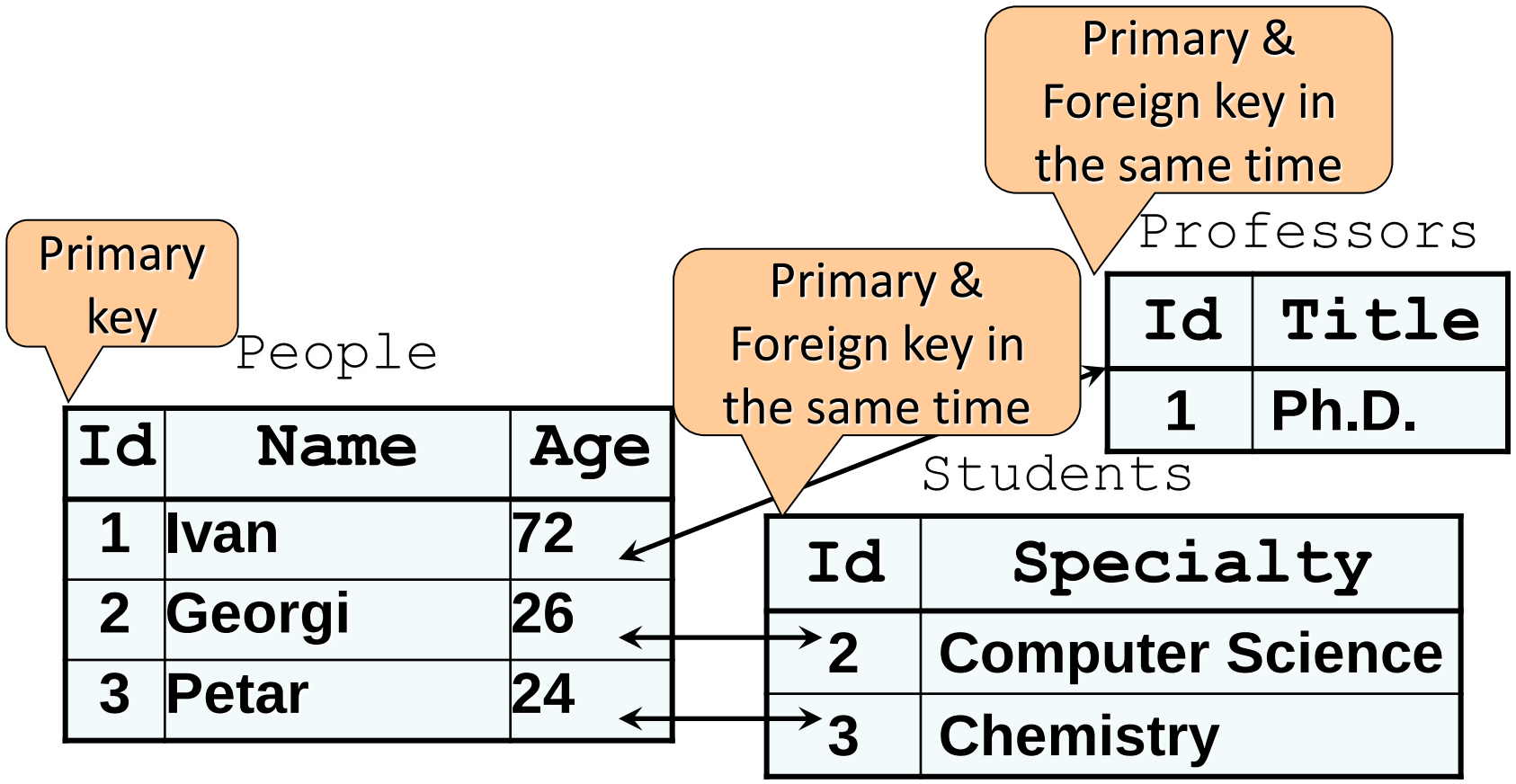
- In a relation of type many x many one record in the first table has many corresponding records in the second one and vice versa. It is implemented using an extra table



(**many x many** is split into **one x many** and **many x one** so that foreign keys can be used).

Relational Database Schema

- A relation of type 1 x 1 one record in a table corresponds to 1 record in the other table



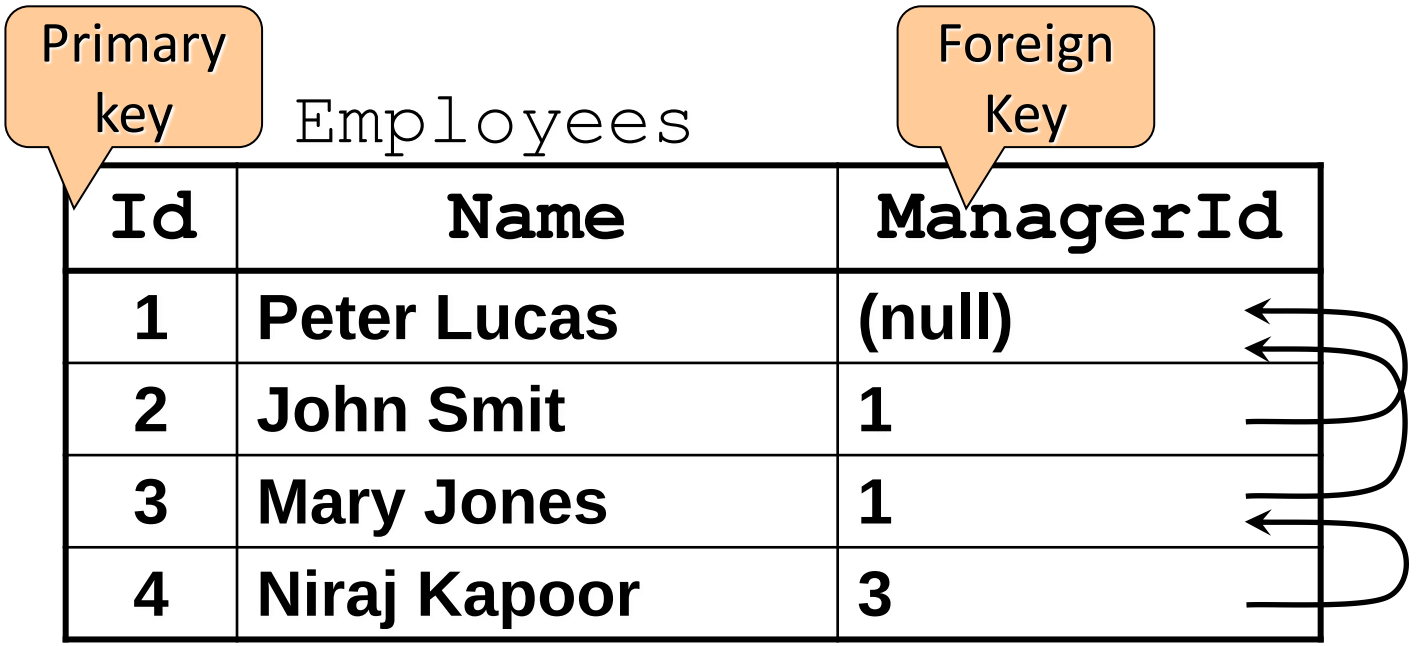
Id	Title
1	Ph.D.

Id	Name	Age
1	Ivan	72
2	Georgi	26
3	Petar	24

Id	Specialty
2	Computer Science
3	Chemistry

Relational Database Schema

- The foreign key can also point to one and the same table (this is also called an auto relation) - e.g. employees in a company have a manager, who is also an employee



Relational Database Schema

- Normalization of the relational scheme removes repeating data

- Non-normalized data may contain many repetitions. For example:

Product	Producer	Price	Category	Shop	Town
Yoghurt	Mlexis LTD	0.67	food	store "Mente"	Sofia
bread "Dobrudja"	Bakery "Smoky"	0.85	food	store "Mente"	Sofia
beer "Zagorka"	Zagorka CO	0.68	soft drinks	stall "non-stop"	Varna
beer "Tuborg"	Shumen Drinks CO	0.87	soft drinks	stall "non-stop"	Varna

Relational Database Schema

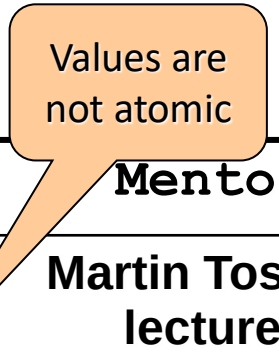
- When designing a database typically only the first, second and third normal forms are considered (however there are higher level forms such as fourth and fifth)

- Sample denormalized table:

ProjectID	Project	Participants	Mentor
1	Social Network	Ivan Ivanov, Petar Petrov	Martin Toshev, lecturer
2	Security Testing Tool	Elena Petrova Maria Dimitrova	Martin Toshev, lecturer

Relational Database Schema

- First Normal Form - there no duplicate rows (i.e. there must be a primary key) and columns in a table; the fields in the rows are atomic (inseparable) values



ProjectID (PK)	Project	Participants	Mentor
1	Social Network	Ivan Ivanov, Petar Petrov	Martin Toshev, lecturer
2	Security Testing Tool	Elena Petrova Maria Dimitrova	Martin Toshev, lecturer

Relational Database Schema

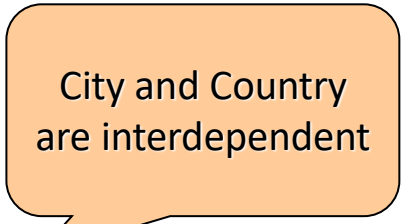
- Second Normal Form - Retains all requirements of First Normal Form; there are no columns that depend on part of the primary key (i.e. columns that do not depend on the primary key are extracted to a separate table)

Project does not depend on the MentorId

MentorId (PK)	Project	Mentor
1	Social Network	Martin Toshev
2	Security Testing Tool	Martin Toshev

Relational Database Schema

- Third Normal Form - retains all requirements of Second Normal Form; removes columns that are interdependent



MentorId	Mentor	City	Country
1	Martin Toshev	Sofia	Bulgaria

Relational Database Schema

- Fourth Normal Form - Retains all requirements of Third Normal Form; there is one column at most in each table that can have many possible values for a single key (multi-valued attribute)

One mentor can participate in many courses

One mentor can participate in many projects

MentorId (PK)	Course (PK)	Project (PK)
1	.NET Programming	Regular Expressions in .NET
2	Mastering J2EE	Best Practices in J2EE

Relational Database Schema

- Views:
  - are the result from a stored query on a data
  - can represent a subset of the data contained in a table
  - can be queried in the same manner as tables
  - can join and simplify multiple tables
  - can hide data complexity
  - allow for more fine-grained security - users may have only access to certain views and not to the tables that these views use

Relational Database Schema

- Sample views:

Id	Company	TownId
1	Mente LTD	1
2	BulkSoft Inc.	2
3	HardSoft CO	4
4	Sputnick CO	3

V_Companies

Id	Town	CountryId
1	Sofia	1
2	New York	3
3	Moscow	2
4	Plovdiv	1

V_Towns

V_Countries

Id	Country
1	Bulgaria
2	Russia
3	USA

Relational Database Schema

- Indexes:
  - can be outer (outside the table) or built-in
  - when defined slow down adding or deleting records in indexed tables
  - when unique imply a unique constraint
  - speed up searching of values in a certain column or group of columns
  - are typically defined in big tables
  - are usually implemented as B-trees or hash tables

Relational Database Schema

- Stored procedures:
  - are procedures defined in the database server
  - can be used by external applications connecting to the database
  - are stored in the data dictionary of the database
  - faster than an outer code (moreover data is locally accessible)
  - may reduce network usage
  - can accept parameters

Relational Database Schema

- Stored procedures:
  - are procedures defined in the database server
  - can be used by external applications connecting to the database
  - are stored in the data dictionary of the database
  - faster than an outer code (moreover data is locally accessible)
  - may reduce network usage
  - can accept parameters

Relational Database Schema

- Stored procedures:
  - typically have different implementations in the different RDBMS:
    - in PL/SQL for Oracle
    - in Transact-SQL for Microsoft SQL Server
    - custom format in MySQL
    - in SQL PL for DB2
    - in PL/pgSQL for PostgreSQL

Relational Database Schema

- Functions:
  - are stored procedures that can return result
    - single value
    - record set
  - must not be confused with standard SQL functions (discussed later)

Relational Database Schema

- Triggers:
  - are database level procedures that activate when some event occurs, for instance:
    - when inserting a record
    - when updating a record
    - when deleting a record
  - are commonly used to audit changes (e.g. by logging information about users and roles involved in changes), execute business rules, replicate data or enhance performance

Relational Database Schema

- Sequences:
  - database objects from which multiple users may generate unique integers (e.g. for use by primary key values)
- Synonyms:
  - are aliases (alternative names) for a table, view, sequence or other database schema object

Relational Database Design

Relational Database Design

- The purpose of a database design is to provide a data model of the database

- Typically database design involves the database tables and the relationships between them - additional constraints and other database object might be specified either at design time or after a concrete schema description is generated from the model

Relational Database Design

- Typical design process:

  1) determine the purpose of the database

  2) gather the required information and divide into tables (typically normalized at least to the Third Normal Form unless there aren't other specific requirements)

  3) specify primary keys for the tables

  4) specify the relations between tables in terms of foreign keys

  5) specify constraints on the columns

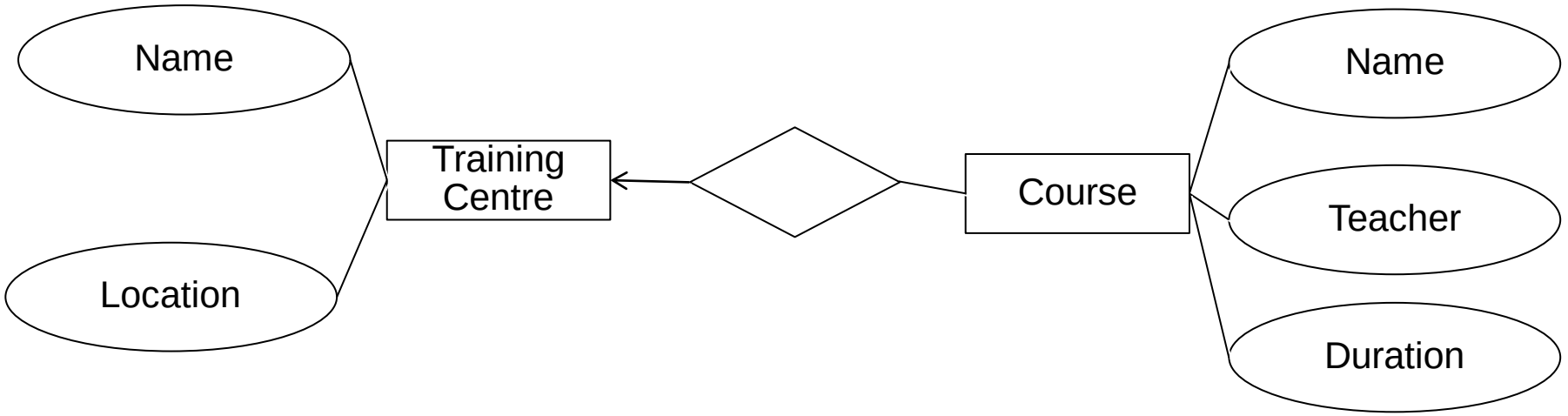
Relational Database Design

- A relational database schema can be designed graphically using Entity/Relationship diagrams (E/R Diagrams)

- An E/R diagram has three main types of elements:
  - Entities (tables)
  - Attributes (columns)
  - Relationships

Relational Database Design

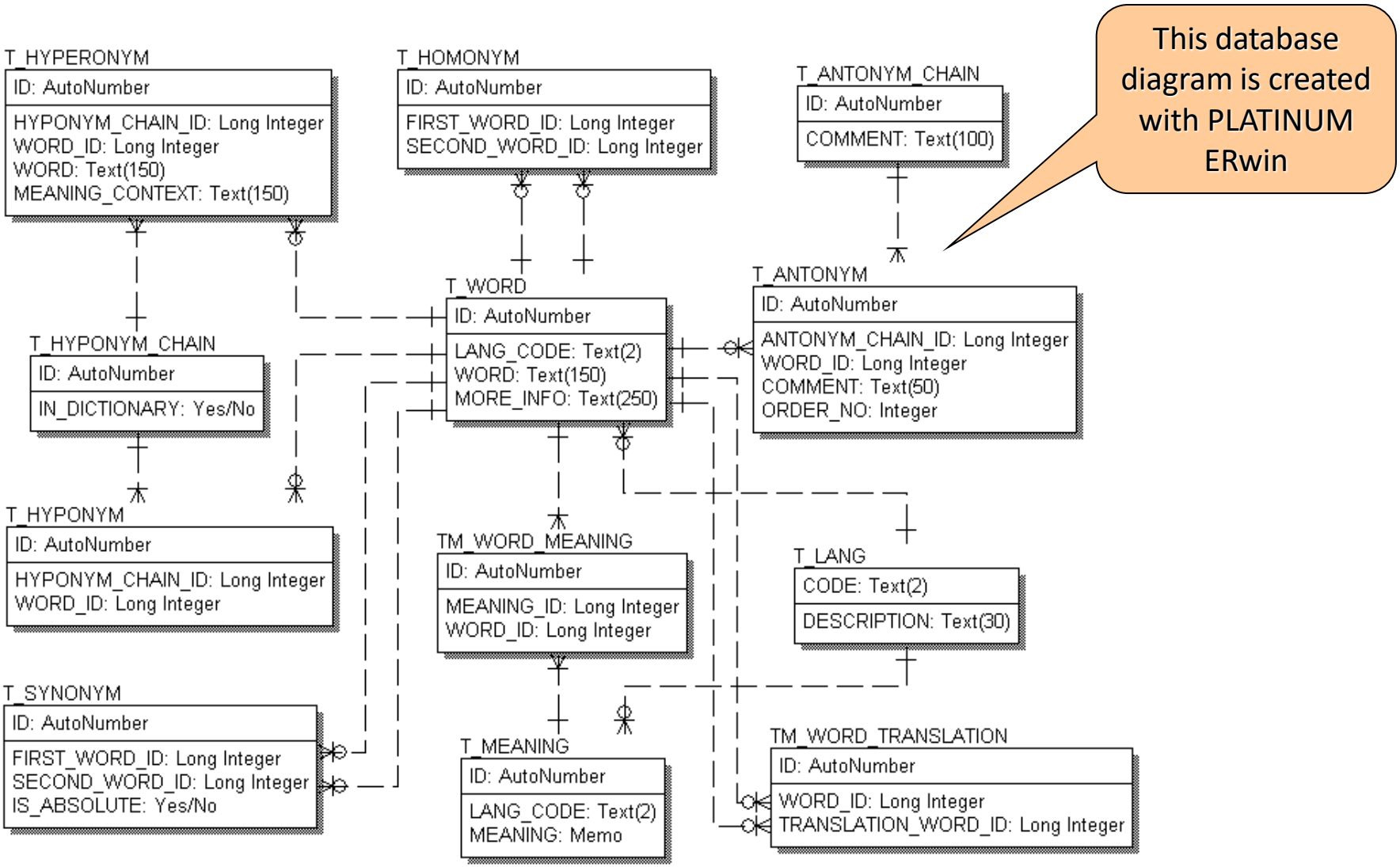
- Example E/R diagram:



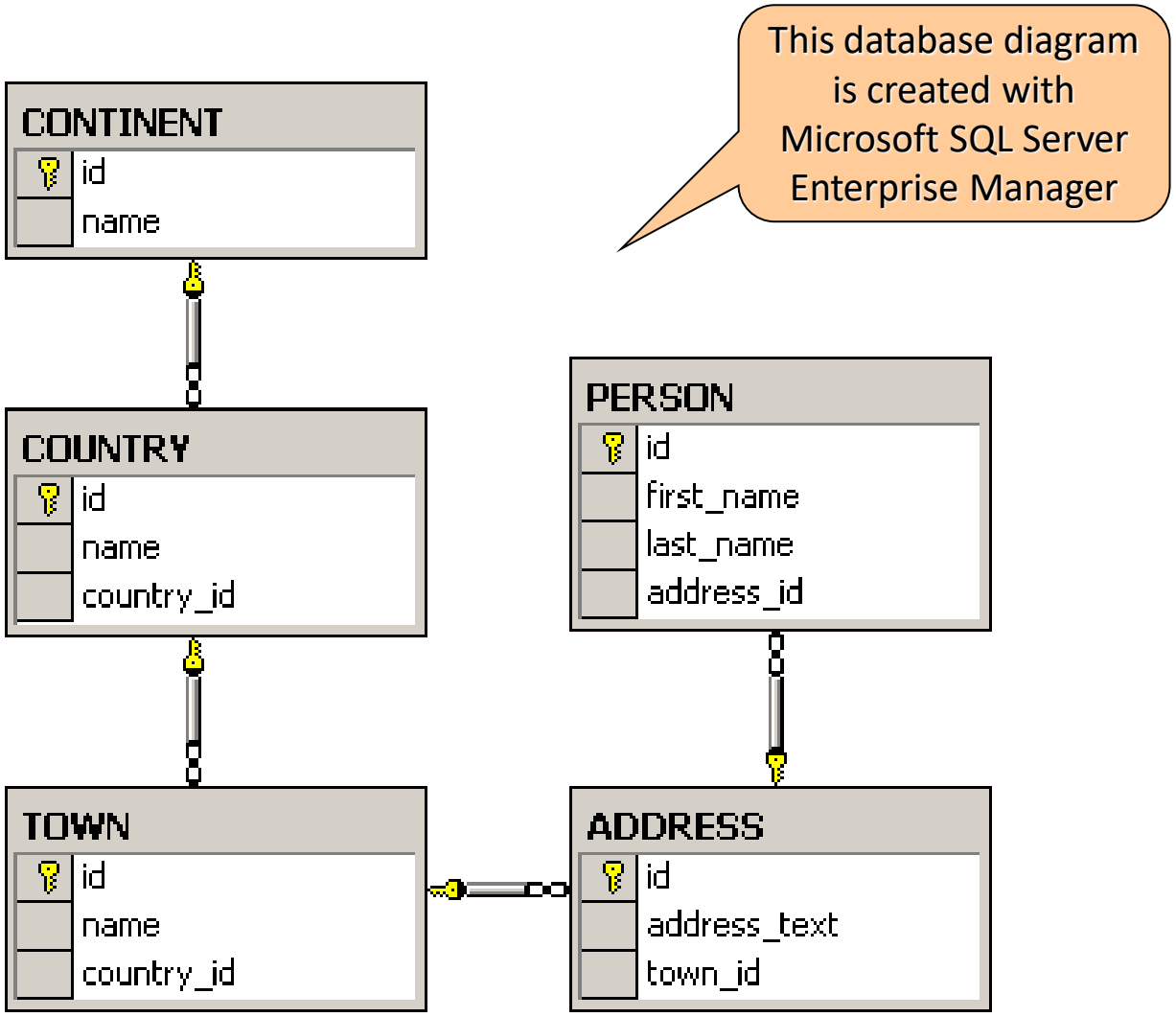
Relational Database Design

- Tools for creating E/R or similar database diagrams (Data Modelling Tools):
  - SQL Server Enterprise Manager
  - Oracle Designer
  - Microsoft Visio
  - Computer Associates ERwin
  - CASE Studio
  - IBM Rational Rose
  - theKompany Data Architect

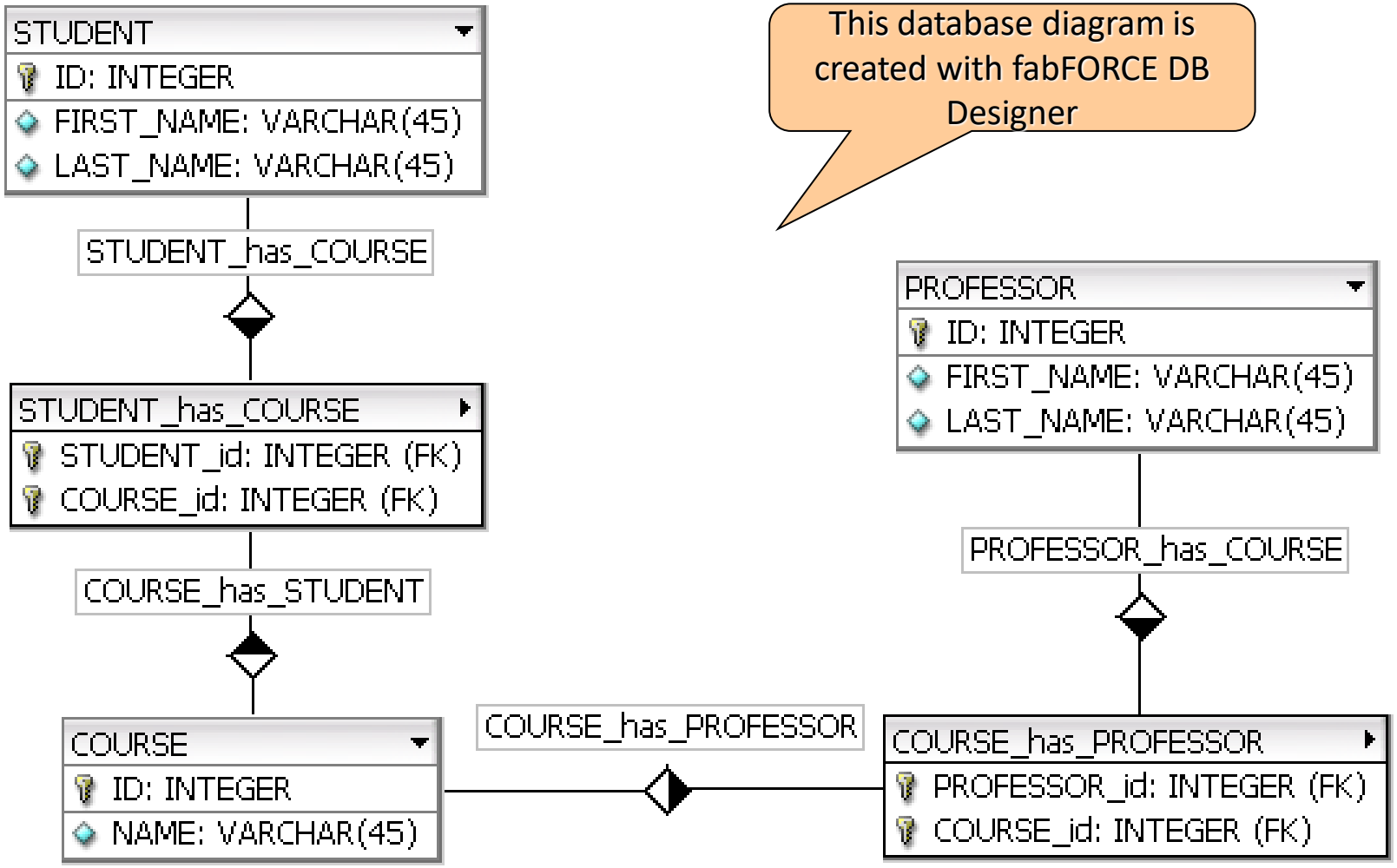
Relational Database Design



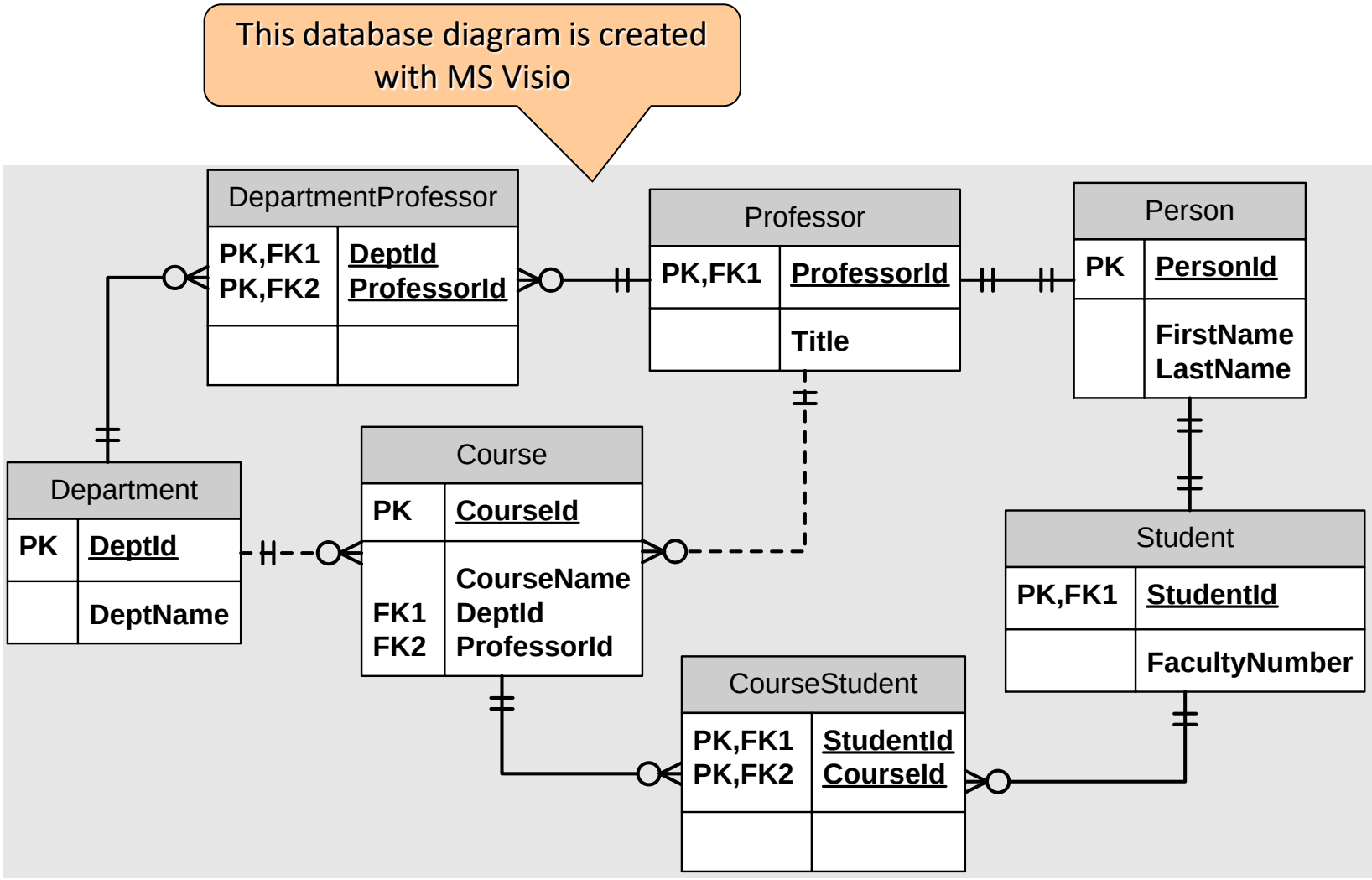
Relational Database Design



Relational Database Design



Relational Database Design



The SQL language

The SQL language

- SQL (Structured Query Language):
  - standardized declarative language for manipulation of relational databases
  - SQL-99 is currently into use (SQL-2003, SQL-2006, SQL-2008 and SQL-2011 revisions are either partially or fully supported by the RDBMS)
  - supports creating, altering, deleting tables and other objects in the database
  - supports searching for, retrieving, inserting, changing and deleting records

The SQL language

- SQL (Structured Query Language):
  - DDL – Data Definition Language (CREATE, ALTER, DROP commands)
  - DML – Data Manipulation Language (SELECT, INSERT, UPDATE, DELETE commands)
- Example SQL SELECT query:

```
SELECT Towns.Name, Countries.Name
FROM Towns, Countries
WHERE Towns.CountryId = Countries.Id
```

Questions ?

Exercises (1)

1. What RDBMS do you know?
2. Which are the main functions performed by a Relational Database Management System (RDBMS)?
3. Define "table" in terms of relational databases.
4. Explain the difference between a primary and a foreign key.
5. Point out the different types of relationships between tables.

Exercises (2)

6. When is a certain database normalized? What are the advantages of a normalized database?
7. What are constraints used for in a database?
8. Point out the pros and cons of using indexes in a database.
9. What's the main purpose of the SQL language?
10. What common features do RDBMS share ?
11. How can a database schema be designed ?