Telerik Academy

Database Systems

DBMS Fundamental Concepts

Telerik Academy

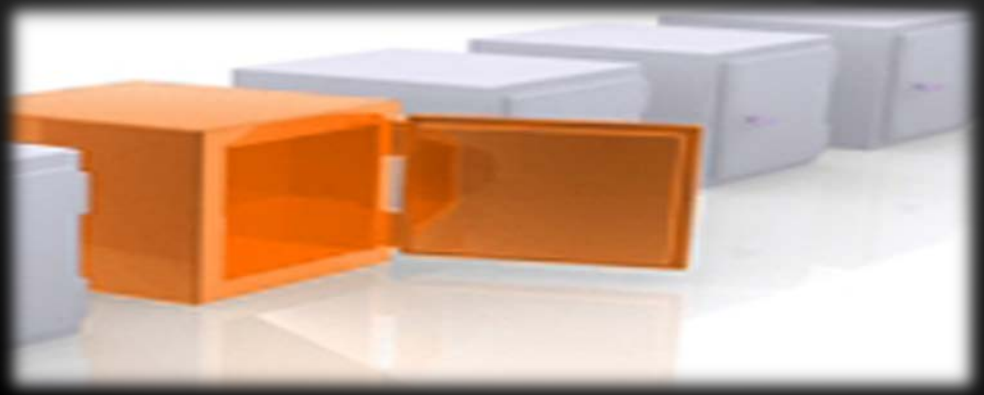
Table of Contents

1. Database Models
2. Relational Database Model
3. DBMS & RDBMS Systems
4. Tables, Relationships, Multiplicity, E/R Diagrams
5. Normalization
6. Constraints
7. Indices
8. The SQL language

Telerik Academy

Table of Contents (2)

9. Stored Procedures
10. Views
11. Triggers
12. Transactions and Isolation
13. NoSQL Databases



Telerik Academy

RDBMS Systems

Relational Databases, Database Servers and RDBMS

Relational Databases

- Database models
  - Hierarchical (tree)
  - Network / graph
  - Relational (table)
  - Object-oriented

- Relational databases
  - Represent a bunch of tables together with the relationships between them
  - Rely on a strong mathematical foundation: the relational algebra

Relational Database Management System (RDBMS)

- Relational Database Management Systems (RDBMS) manage data stored in tables
- RDBMS systems typically implement
  - Creating / altering / deleting tables and relationships between them (database schema)
  - Adding, changing, deleting, searching and retrieving of data stored in the tables
  - Support for the SQL language
  - Transaction management (optional)

RDBMS Systems

- RDBMS systems are also known as:
  - Database management servers
  - Or just database servers
- Popular RDBMS servers:
  - Microsoft SQL Server
  - Oracle Database
  - MySQL
  - IBM DB2
  - PostgreSQL
  - SQLite

Telerik Academy

Tables and Relationships

Database Tables, Relationships, Multiplicity

Tables

- Database tables consist of data, arranged in rows and columns
  - For example (table `Persons`):

Id	First Name	Last Name	Employer
1	Svetlin	Nakov	Telerik
2	Stephen	Forte	Telerik
3	Steve	Jobs	Apple

- All rows have the same structure
- Columns have name and type (number, string, date, image, or other)

Table Schema

- The schema of a table is an ordered sequence of column specifications (name and type)
- For example the Persons table has the following schema:

```
Persons (
   Id: number,
   FirstName: string,
   LastName: string,
   Employer: string
)
```

Primary Key

- Primary key is a column of the table that uniquely identifies its rows (usually its is a number)

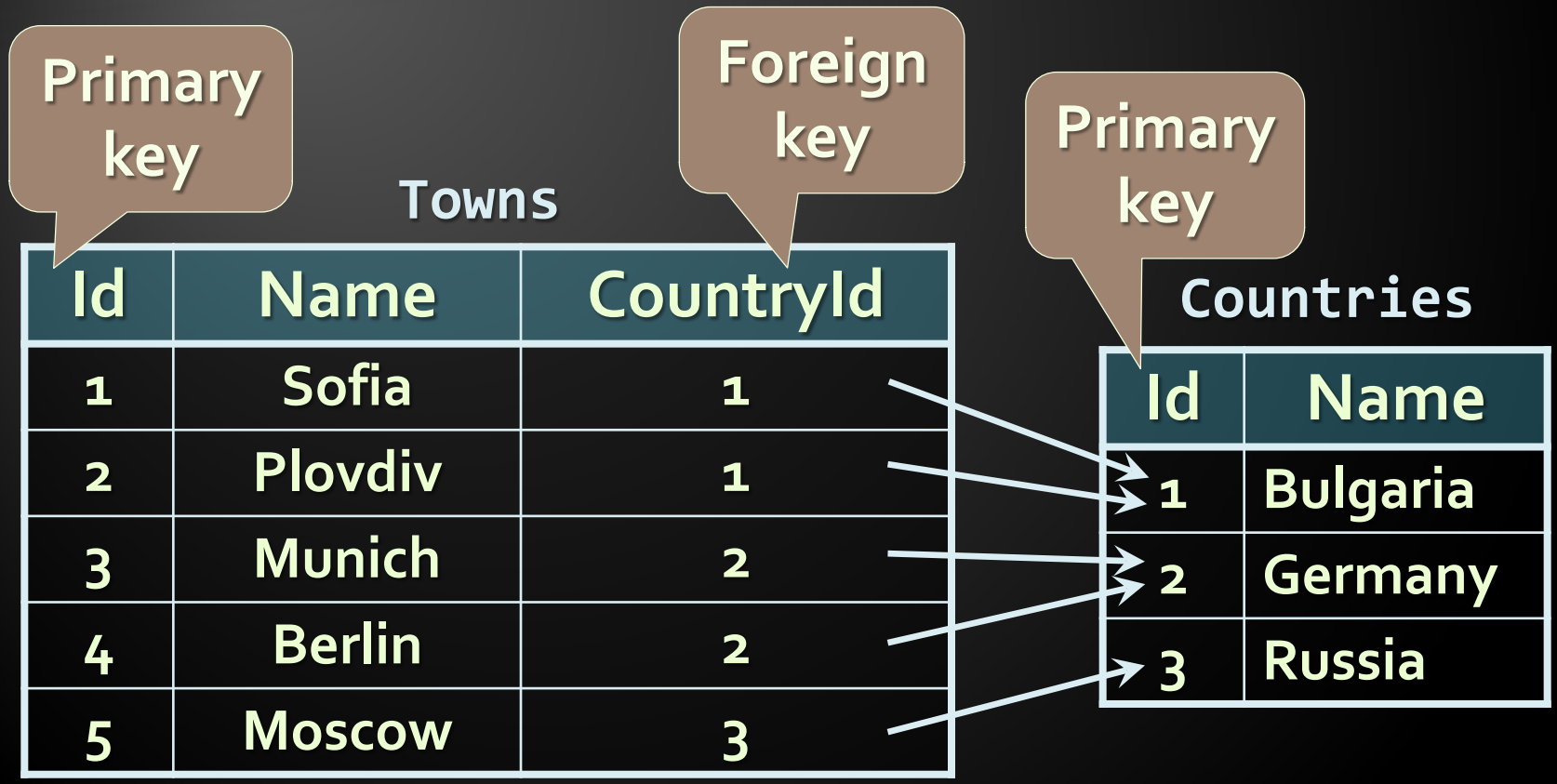
Primary key

Id	First Name	Last Name	Employer
1	Svetlin	Nakov	Telerik
2	Stephen	Forte	Telerik
3	Steve	Jobs	Apple

- Two records (rows) are different if and only if their primary keys are different
- The primary key can be composed by several columns (composite primary key)

Relationships

- Relationships between tables are based on interconnections: primary key / foreign key



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

Relationships (2)

- The foreign key is an identifier of a record located in another table (usually its primary key)
- By using relationships we avoid repeating data in the database
  - In the last example the name of the country is not repeated for each town (its number is used instead)
- Relationships have multiplicity:
  - One-to-many – e.g. country / towns
  - Many-to-many – e.g. student / course
  - One-to-one – e.g. example human / student

Relationships' Multiplicity

- Relationship one-to-many (or many-to-one)
  - A single record in the first table has many corresponding records in the second table
  - Used very often

Towns

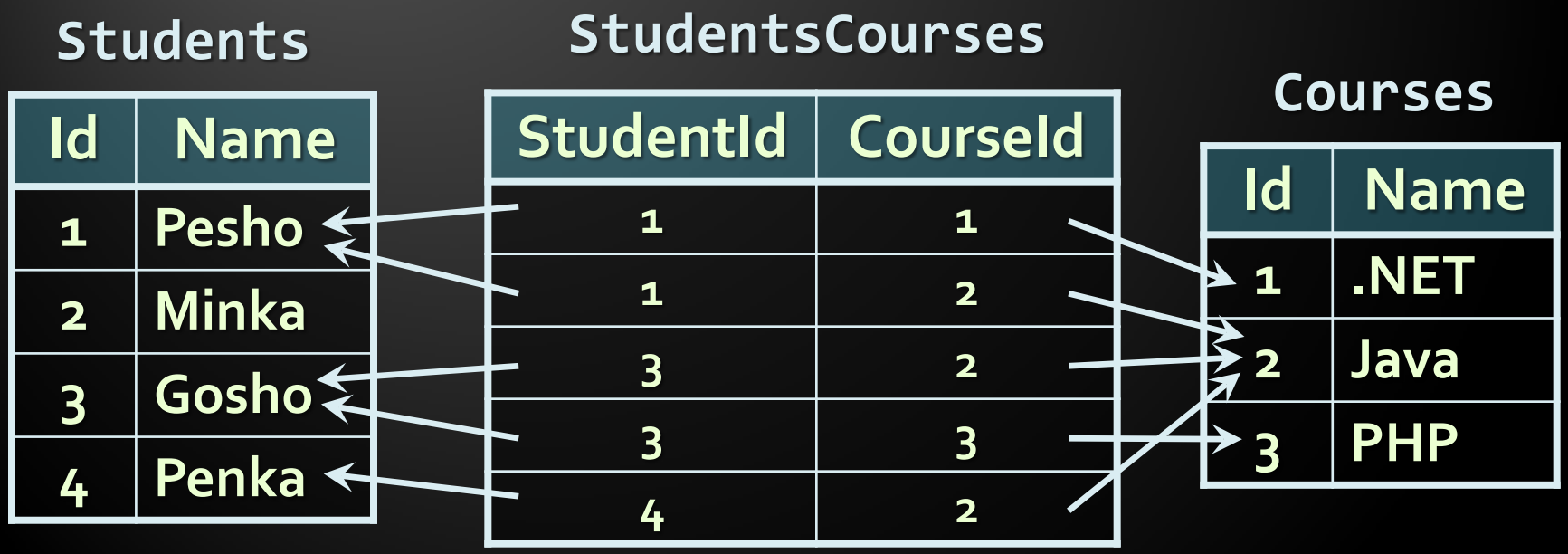
Id	Name	CountryId
1	Sofia	1
2	Plovdiv	1
3	Munich	2
4	Berlin	2
5	Moscow	3

Countries

Id	Name
1	Bulgaria
2	Germany
3	Russia

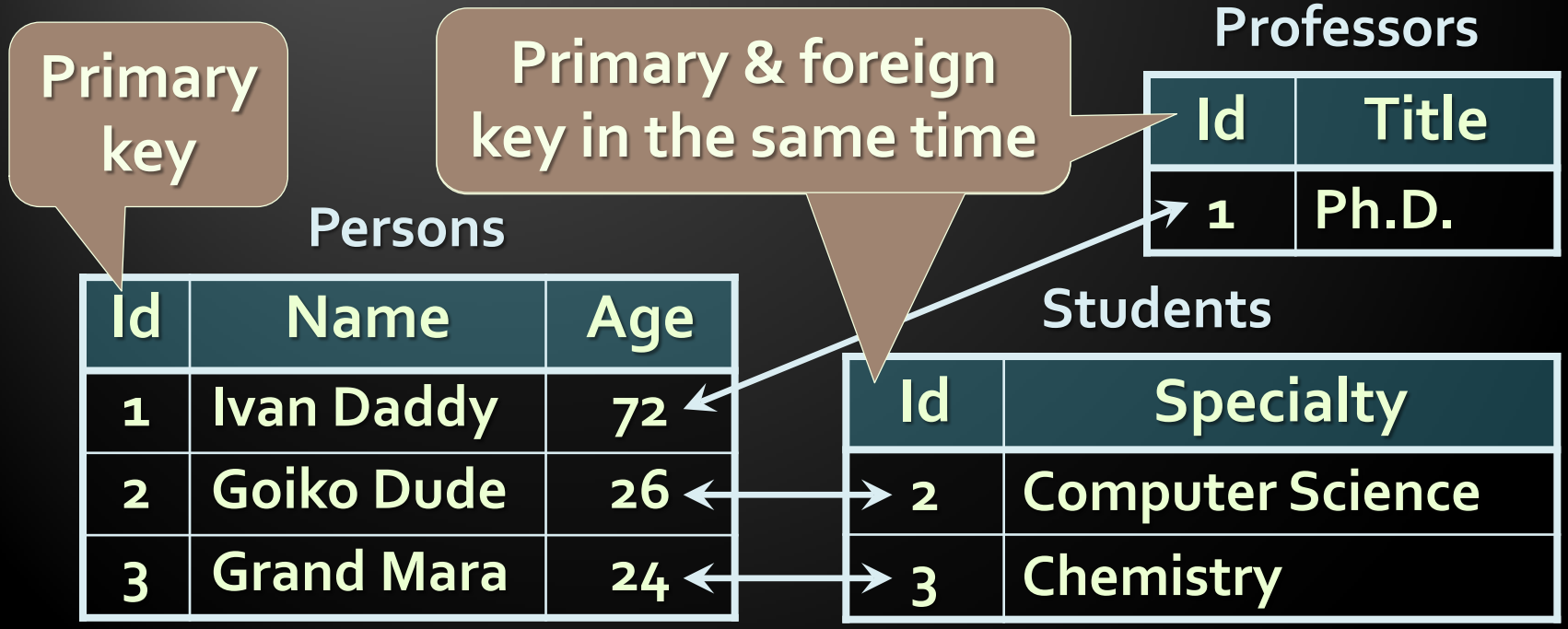
Relationships' Multiplicity (2)

- Relationship many-to-many
  - Records in the first table have many corresponding records in the second one and vice versa
  - Implemented through additional table



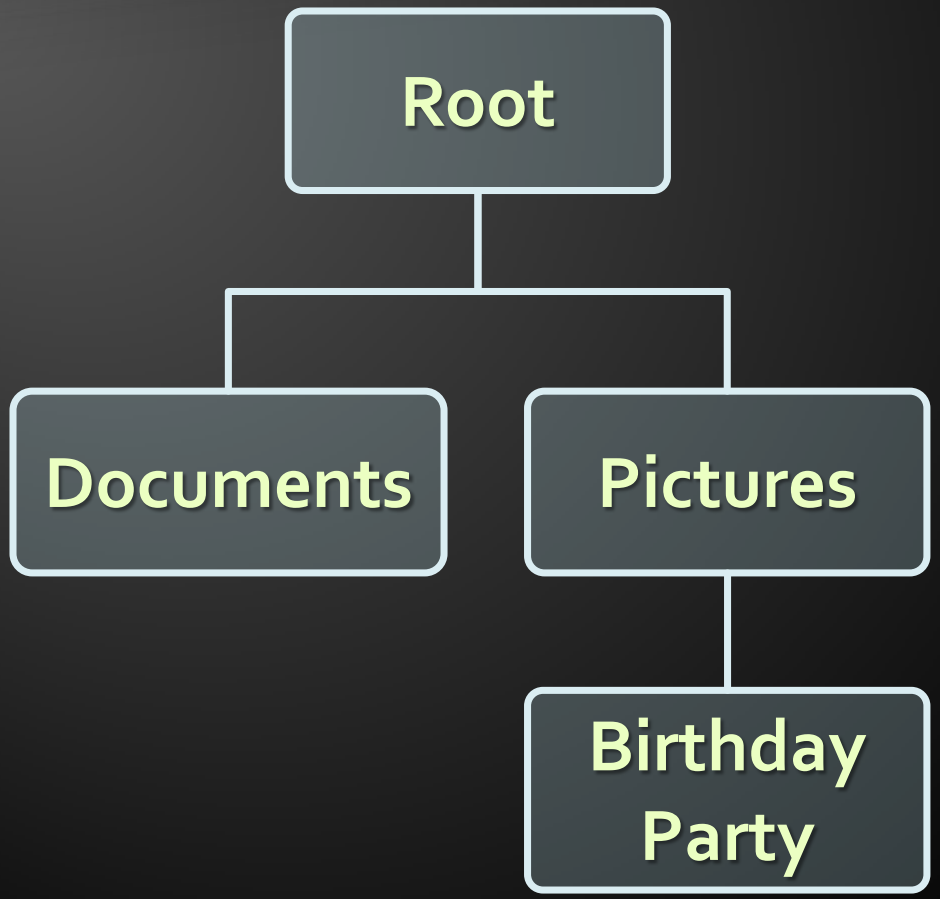
Relationships' Multiplicity (3)

- Relationship one-to-one
  - A single record in a table corresponds to a single record in the other table
  - Used to model inheritance between tables



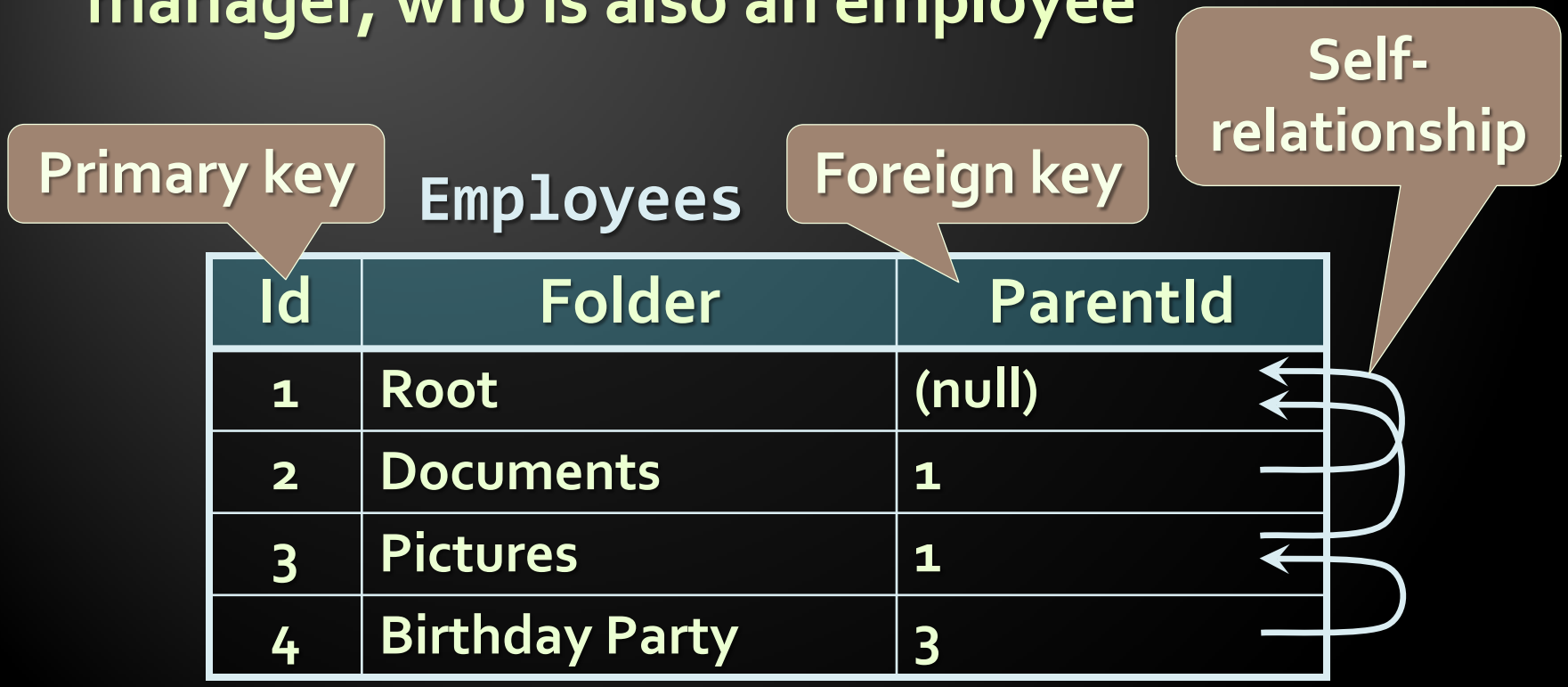
Representing Hierarchical Data

- How do we represent trees and graphs?



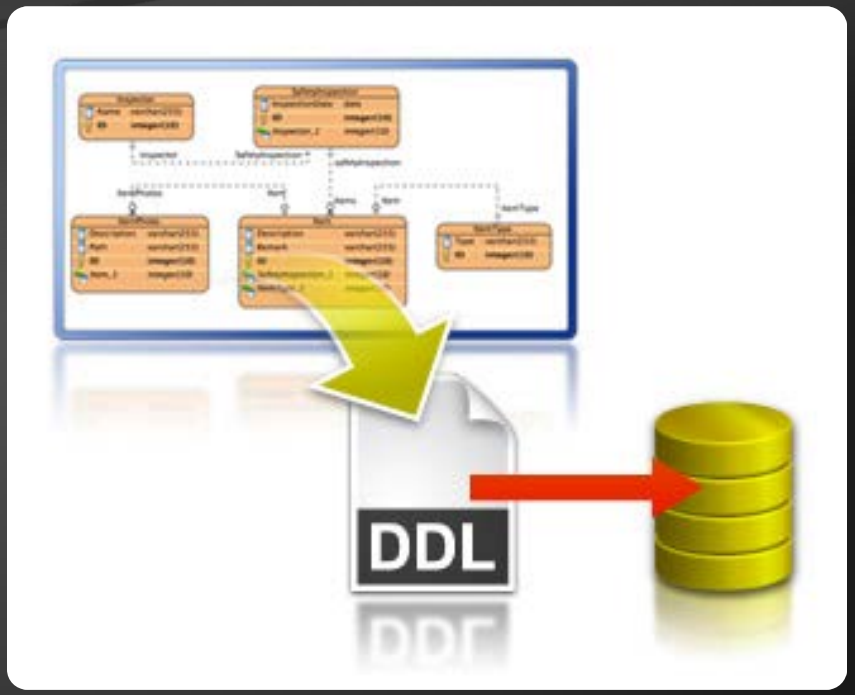
Self-Relationships

- The primary / foreign key relationships can point to one and the same table
  - Example: employees in a company have a manager, who is also an employee



Id	Folder	ParentId
1	Root	(null)
2	Documents	1
3	Pictures	1
4	Birthday Party	3

Telerik Academy



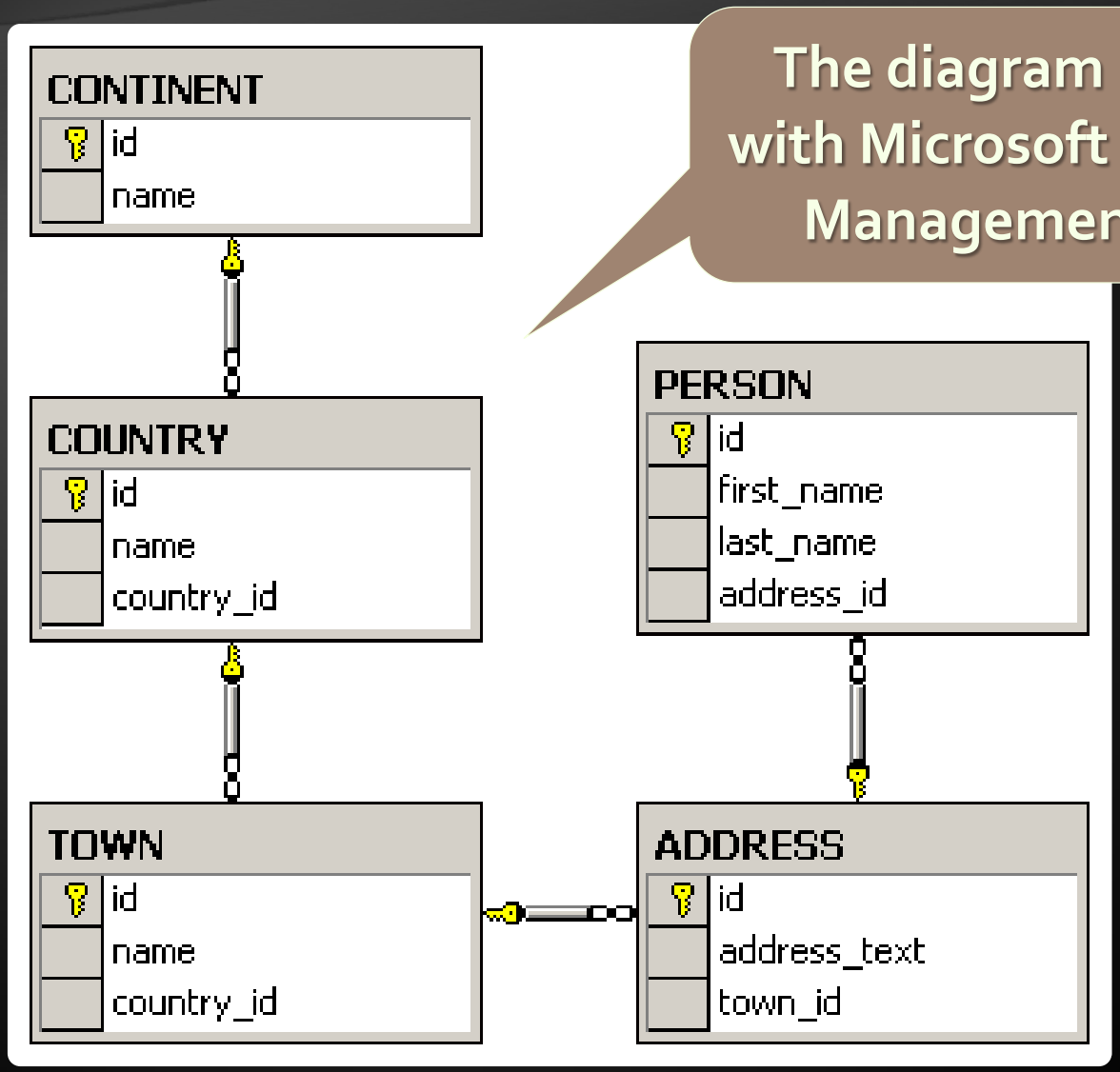
E/R Diagrams

Entity / Relationship Diagrams and DB Modeling Tools

Relational Schema

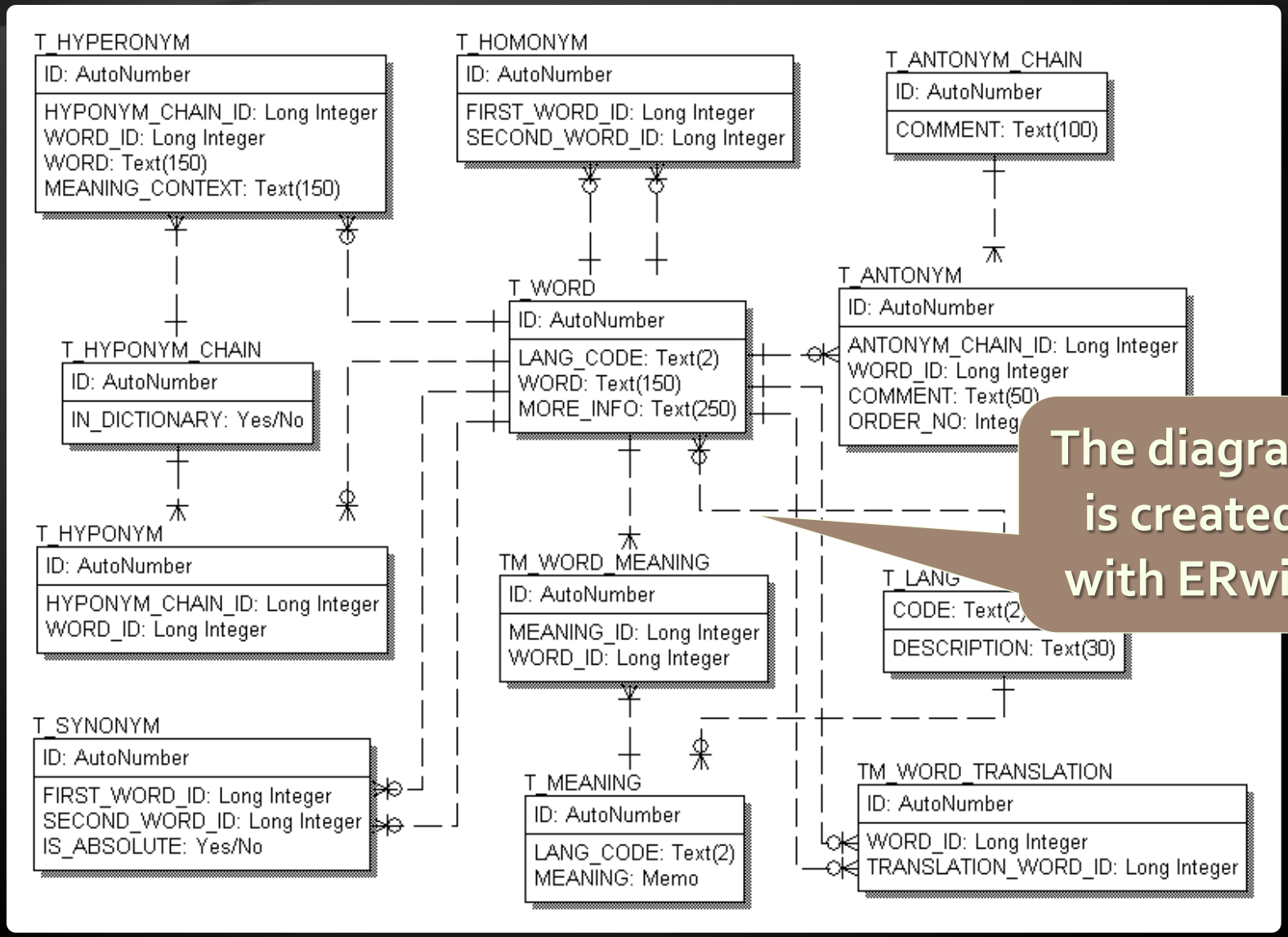
- Relational schema of a DB is the collection of:
  - The schemas of all tables
  - Relationships between the tables
  - Any other database objects (e.g. constraints)
- The relational schema describes the structure of the database
  - Doesn't contain data, but metadata
- Relational schemas are graphically displayed in Entity / Relationship diagrams (E/R Diagrams)

E/R Diagrams – Examples

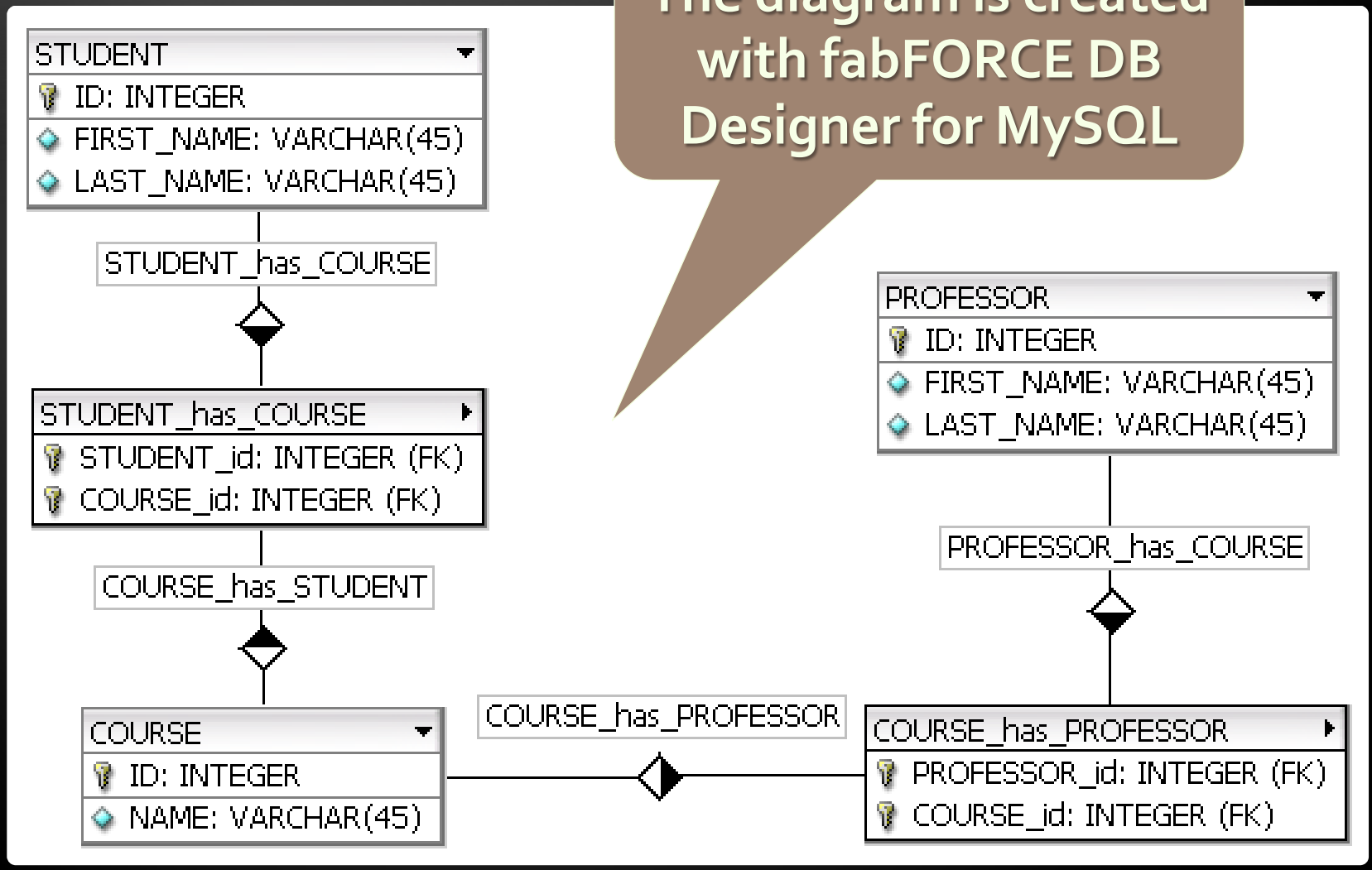


Telerik Academy

E/R Diagrams – Examples (2)

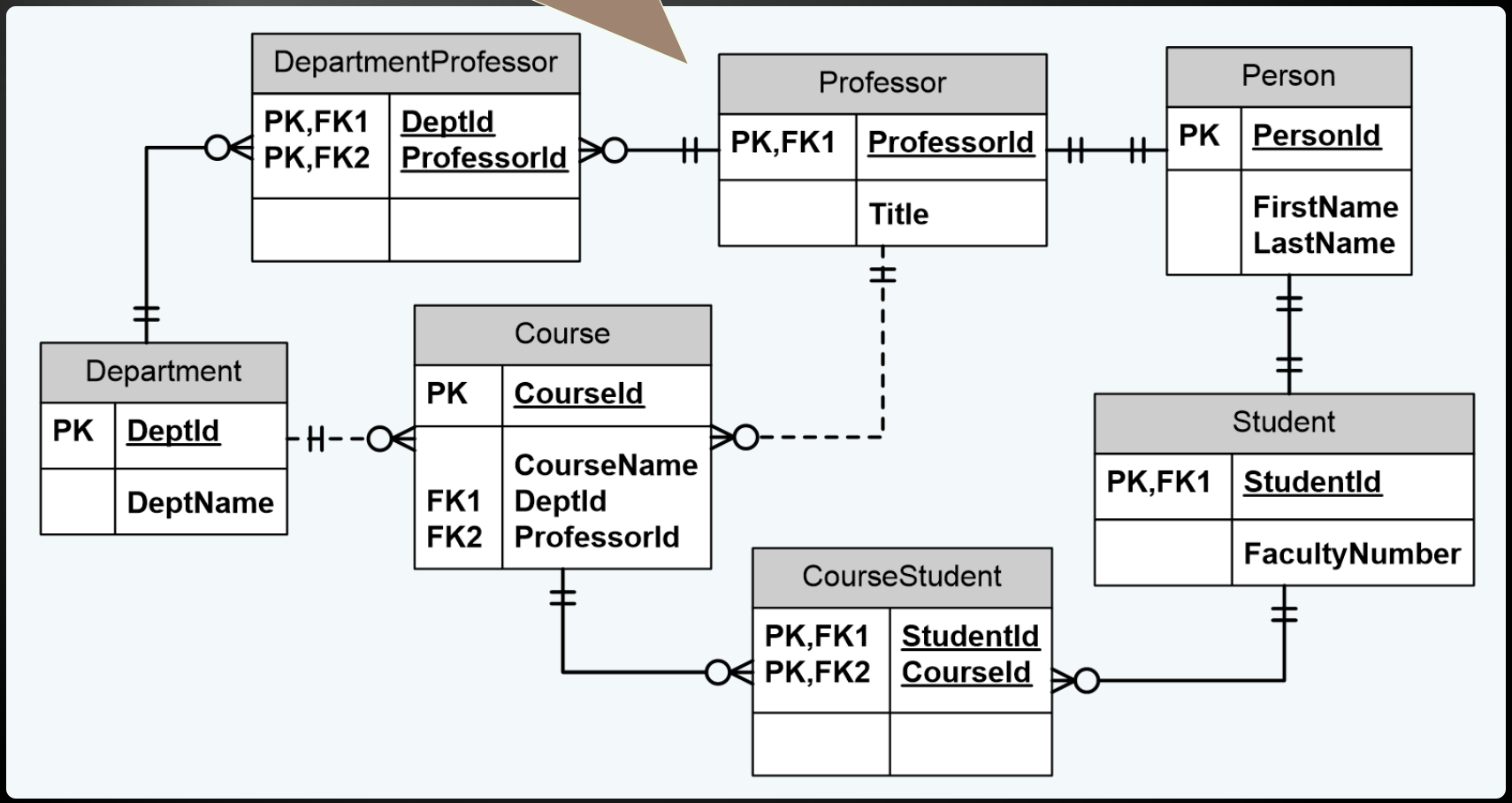


E/R Diagrams – Examples (3)



E/R Diagrams – Examples (4)

The diagram is created with MS Visio



Tools for E/R Design

- Data modeling tools allow building E/R diagrams, generate / import DB schemas:
  - SQL Server Management Studio
  - MySQL Workbench
  - Oracle JDeveloper
  - Microsoft Visio
  - CASE Studio
  - Computer Associates ERwin
  - IBM Rational Rose

Telerik Academy

DB Normalization

Avoiding Duplicated Data through Database Schema Normalization

Normalization

- Normalization of the relational schema removes repeating data
- Non-normalized schemas can contain many data repetitions, e.g.

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 Corp.	0.68	soft drinks	stall "non-stop"	Varna
beer "Tuborg"	Shoumen Drinks Corp.	0.87	soft drinks	stall "non-stop"	Varna

Normalization (6)

- Example of fully normalized schema (in 4th Normal Form):

Products

Id	Product	ProducerId	Price	CategoryId	ShopId	TownId
1	Youghurt	2	0.67	2	4	1
2	bread "Dobrudja"	3	0.55	2	4	1
3	rakia "Peshtera"	6	4.38	5	2	1
4	beer "Tuborg"	4	0.67	4	1	3

Producers

Id	Name
2	"Milk" Ltd.
4	"Zagorka" AD

Categories

Id	Name
4	beer
2	food

Shops

Id	Name
1	Billa
4	METRO

Towns

Id	Name
1	Sofia
3	Varna

Telerik Academy

Other Database Objects

Constraints, Indices, SQL, Stored Procedures, Views, Triggers

Integrity Constraints

- Integrity constraints ensure data integrity in the database tables
  - Enforce data rules which cannot be violated
- Primary key constraint
  - Ensures that the primary key of a table has unique value for each table row
- Unique key constraint
  - Ensures that all values in a certain column (or a group of columns) are unique

Integrity Constraints (2)

- Foreign key constraint
  - Ensures that the value in given column is a key from another table
- Check constraint
  - Ensures that values in a certain column meet some predefined condition
  - Examples:

```
(hour >= 0) AND (hour < 24)
```

```
name = UPPER(name)
```

Indices

- Indices speed up searching of values in a certain column or group of columns
  - Usually implemented as B-trees
- Indices can be built-in the table (clustered) or stored externally (non-clustered)
- Adding and deleting records in indexed tables is slower!
  - Indices should be used for big tables only (e.g. 50 000 rows)

The SQL Language

- SQL (Structured Query Language)
  - Standardized declarative language for manipulation of relational databases
  - SQL-99 is currently in use in most databases
  - http://en.wikipedia.org/wiki/SQL#Standardization
- SQL language supports:
  - Creating, altering, deleting tables and other objects in the database
  - Searching, retrieving, inserting, modifying and deleting table data (rows)

The SQL Language (2)

- SQL consists of:
  - DDL – Data Definition Language
    - CREATE, ALTER, DROP commands
  - DML – Data Manipulation Language
    - SELECT, INSERT, UPDATE, DELETE commands
- Example of SQL SELECT query:

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

Stored Procedures

- Stored procedures (database-level procedures)
  - Consist of SQL-like code stored in the database
  - Code executed inside the database server
  - Much faster than an external code
  - Data is locally accessible
  - Can accept parameters
  - Can return results
    - Single value
    - Record sets

Stored Procedures (2)

- Stored procedures are written in a language extension of SQL
  - T-SQL – in Microsoft SQL Server
  - PL/SQL – in Oracle
- Example of stored procedure in Oracle PL/SQL:

```
CREATE OR REPLACE PROCEDURE
spInsertCountry(countryName varchar2) IS
BEGIN
  INSERT INTO Countries(Name)
  VALUES(countryName);
END;
```

Views

- Views are named SQL SELECT queries which are used as tables
  - Simplify data access
  - Facilitate writing of complex SQL queries
- Used also to apply security restrictions:
  - E.g. a certain user isn't given permissions on any of the tables in the database
  - The user is given permissions on few views (subset of DB) and few stored procedures only

Views – Example

Companies

Id	Company	TownId
1	Mente LTD	1
2	BulkSoft Inc.	2
3	HardSoft Corp.	4
4	Sputnik Corp.	3

Towns

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

Countries

Id	Country
1	Bulgaria
2	Russia
3	USA

Views – Example (2)

```
CREATE VIEW V_BGCompanies AS
  SELECT
    Companies.Id AS Id,
    Companies.Company AS Company
  FROM Companies INNER JOIN
    (Towns INNER JOIN Countries ON
     Towns.CountryId = Countries.Id)
    ON Companies.TownId = Towns.Id
  WHERE
    Countries.Country = "Bulgaria";
```

V_BGCompanies

Id	Company
1	Mente Ltd.
3	HardSoft Corp.

Triggers

- Triggers are special stored procedures that are activate when some event occurs, for instance:
  - When inserting a record
  - When changing a record
  - When deleting a record

- Triggers can perform additional data processing of the affected rows, e.g.
  - To change the newly added data
  - To maintain logs and history

Triggers – Example

- We have a table holding company names:

```
CREATE TABLE Companies(
  Id number NOT NULL,
  Name varchar(50) NOT NULL)
```

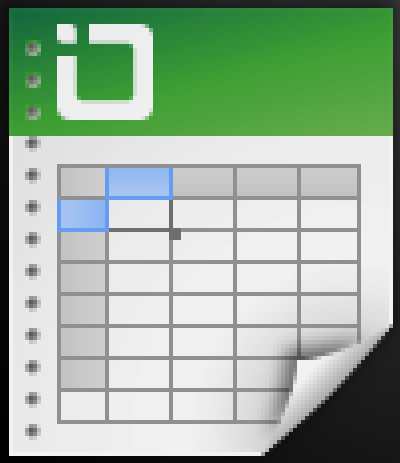
- A trigger that appends "Ltd." at the end of the name of a new company:

```
CREATE OR REPLACE TRIGGER trg_Companies_INSERT
  BEFORE INSERT ON Company
  FOR EACH ROW
BEGIN
  :NEW.Name := :NEW.Name || ' Ltd.';
END;
```

Telerik Academy

Transactions

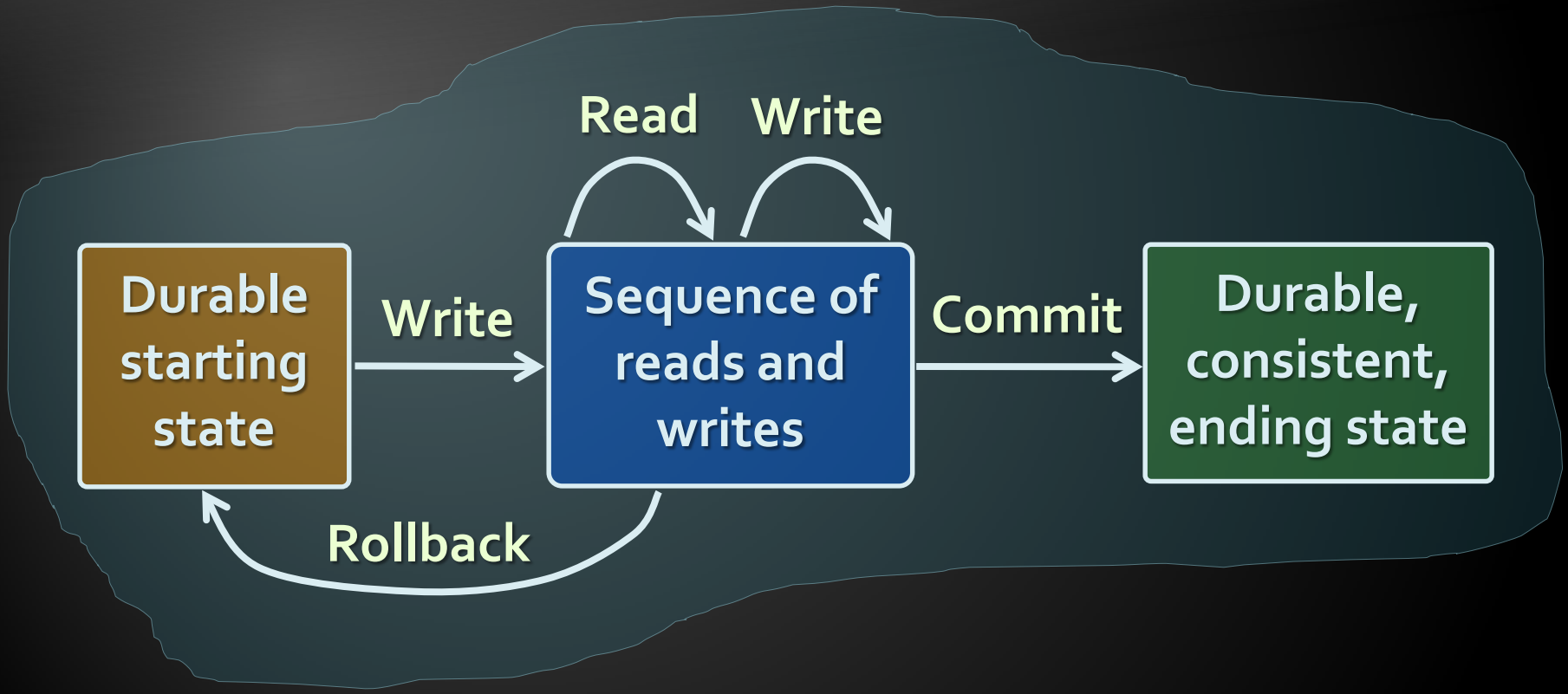
ACID Transactions and Isolation



Transactions

- Transactions are a sequence of database operations which are executed as a single unit:
  - Either all of them execute successfully
  - Or none of them is executed at all
- Example:
  - A bank transfer from one account into another (withdrawal + deposit)
  - If either the withdrawal or the deposit fails the entire operation should be cancelled

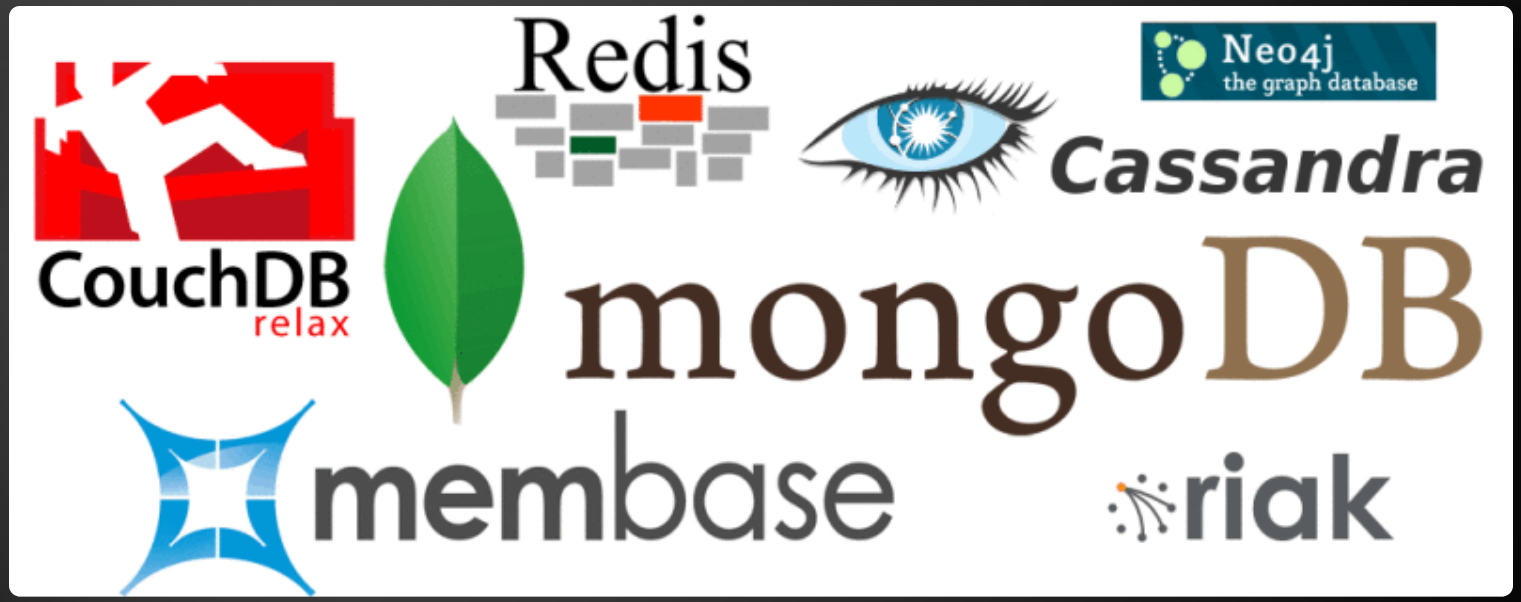
DB Transactions Lifecycle



Transactions Behavior

- Transactions guarantee the consistency and the integrity of the database
  - All changes in a transaction are temporary
  - Changes become final when COMMIT is successfully executed
  - At any time all changes done in the transaction can be canceled by executing ROLLBACK
- All operations are executed as a single unit
  - Either all of them pass or none of them

Telerik Academy



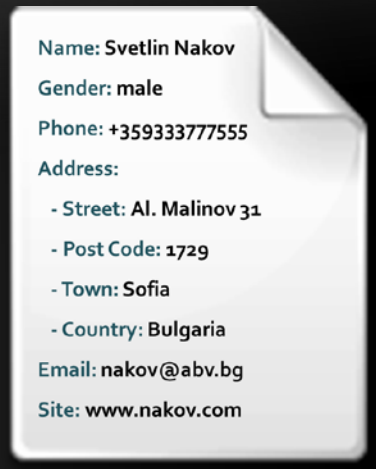
NoSQL Databases

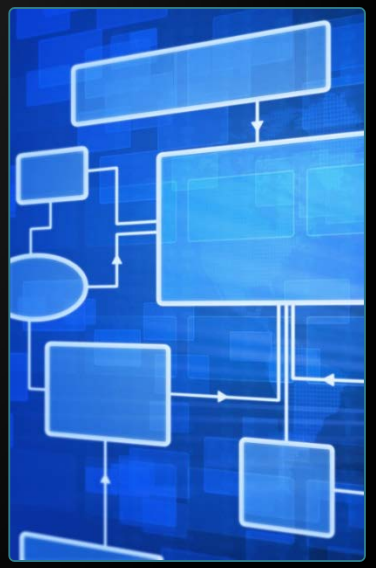
Non-Relational Database Systems

Telerik Academy

Non-Relational Data Models

- Document model
  - Set of documents, e.g. JSON strings
- Key-value model
  - Set of key-value pairs
- Hierarchical key-value
  - Hierarchy of key-value pairs
- Wide-column model
  - Key-value model with schema
- Object model
  - Set of OOP-style objects





What is NoSQL Database?

- NoSQL (non-relational) databases
  - Use document-based model (non-relational)
  - Schema-free document storage
    - Still support CRUD operations (create, read, update, delete)
    - Still support indexing and querying
    - Still supports concurrency and transactions
  - Highly optimized for append / retrieve
  - Great performance and scalability
  - NoSQL == "No SQL" or "Not Only SQL"?

Relational vs. NoSQL Databases

- Relational databases
  - Data stored as table rows
  - Relationships between related rows
  - Single entity spans multiple tables
  - RDBMS systems are very mature, rock solid

- NoSQL databases
  - Data stored as documents
  - Single entity (document) is a single record
  - Documents do not have a fixed structure

Relational vs. NoSQL Models

Relational Model

Name	Svetlin Nakov
Gender	male
Phone	+359333777555
Email	nakov@abv.bg
Site	www.nakov.com

*
1

Street	Al. Malinov 31
Post Code	1729

*
1

Town	Sofia

*
1

Country	Bulgaria

Document Model

Name: Svetlin Nakov

Gender: male

Phone: +359333777555

Address:

- **Street:** Al. Malinov 31
- **Post Code:** 1729
- **Town:** Sofia
- **Country:** Bulgaria

Email: nakov@abv.bg

Site: www.nakov.com

NoSQL Database Systems

- Redis 

  - Ultra-fast in-memory data structures server
- MongoDB 

  - Mature and powerful JSON-document database
- CouchDB 

  - JSON-based document database with REST API
- Cassandra 

  - Distributed wide-column database
- DB Ranking: http://db-engines.com/en/ranking

Telerik Academy

Database Systems

http://academy.telerik.com

Exercises

1. What database models do you know?
2. Which are the main functions performed by a Relational Database Management System (RDBMS)?
3. Define what is "table" in database terms.
4. Explain the difference between a primary and a foreign key.
5. Explain the different kinds of relationships between tables in relational databases.
6. When is a certain database schema normalized? What are the advantages of normalized databases?

Exercises (2)

7. What are database integrity constraints and when are they used?
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 are transactions used for? Give an example.
11. What is a NoSQL database?
12. Explain the classical non-relational data models.
13. Give few examples of NoSQL databases and their pros and cons.

Telerik Academy

Free Trainings @ Telerik Academy

- C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com