Databases

Relational and Non-Relational Databases

SoftUni Team

Technical Trainers

Software University

Software University

https://about.softuni.bg

Table of Contents

Have a Question?

sli.do

#qa-fund

Databases

Introduction

What is a Database?

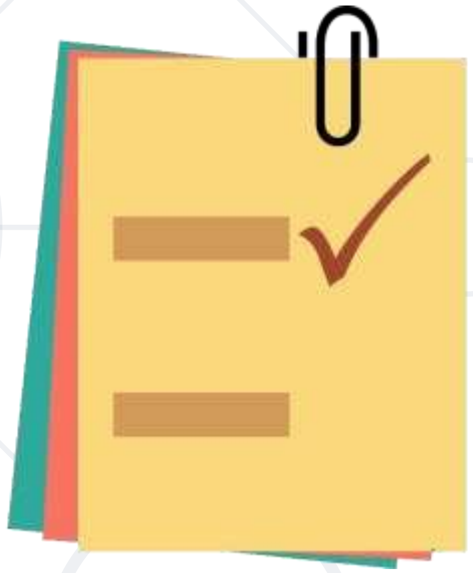
- A **database** is a collection of data, organized to be easily accessed, managed and updated
- Modern databases are managed by **Database Management Systems** (DBMS)
  - Define database **structure**, e.g. tables, collections, columns, relations, indexes
  - Create / Read / Update / Delete data (**CRUD** operations)
  - Execute **queries** (filter / search data)

Relational and NoSQL Databases

- **Databases** hold and manage data in the back-end systems
- **Relational databases (RDBMS)**
  - Hold data in **tables** + **relationships**
  - Use the **SQL** language to query / modify data
  - Examples: MySQL, PostgreSQL, Web SQL in HTML5
- **NoSQL databases**
  - Hold **collections** of documents or key-value pairs
  - Examples: MongoDB, IndexedDB in HTML5

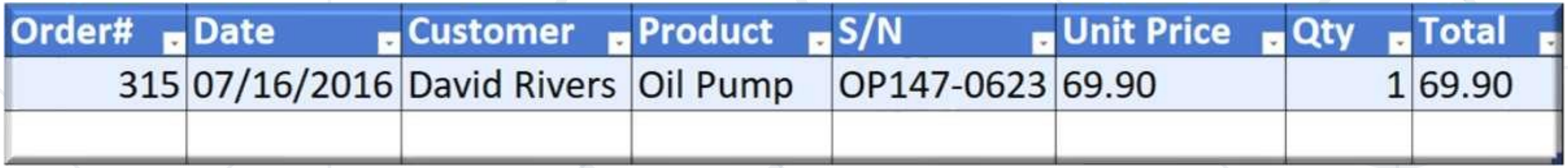
Data Storage

- Conventional **data storage**
  - Orders
  - Receipts



From Data Storage to Databases

- We can **group related pieces of data** into separate columns:



Order#	Date	Customer	Product	S/N	Unit Price	Qty	Total
315	07/16/2016	David Rivers	Oil Pump	OP147-0623	69.90	1	69.90

Why Do We Need Databases?

- Storing data is **not** the primary reason to use a database
- Flat storage runs into **issues** with:
  - Ease of searching
  - Ease of updating
  - Performance
  - Accuracy and consistency
  - Security and access control
  - Redundancy

Relational Database

SQL Databases (Relational Databases)

- Relational (**SQL**) databases organize data in **tables**
  - Tables have strict structure (**columns** with certain **data types**)
  - Can have **relationships** to other tables
- Relational databases use the **structured query language** (SQL) for defining and manipulating data
  - Extremely powerful for complex queries
- **Relational databases** are the most widely used data management technology

SQL Databases (Relational Databases) (2)

- Relational DB model organizes data into one or more **tables** of columns and rows with a **unique key** identifying each row and **foreign keys** defining **relationships**

Items

ID	Order ID	Name	Quantity	Price
5	1	Table	1	200.00
6	1	Chair	1	123.12

Customers

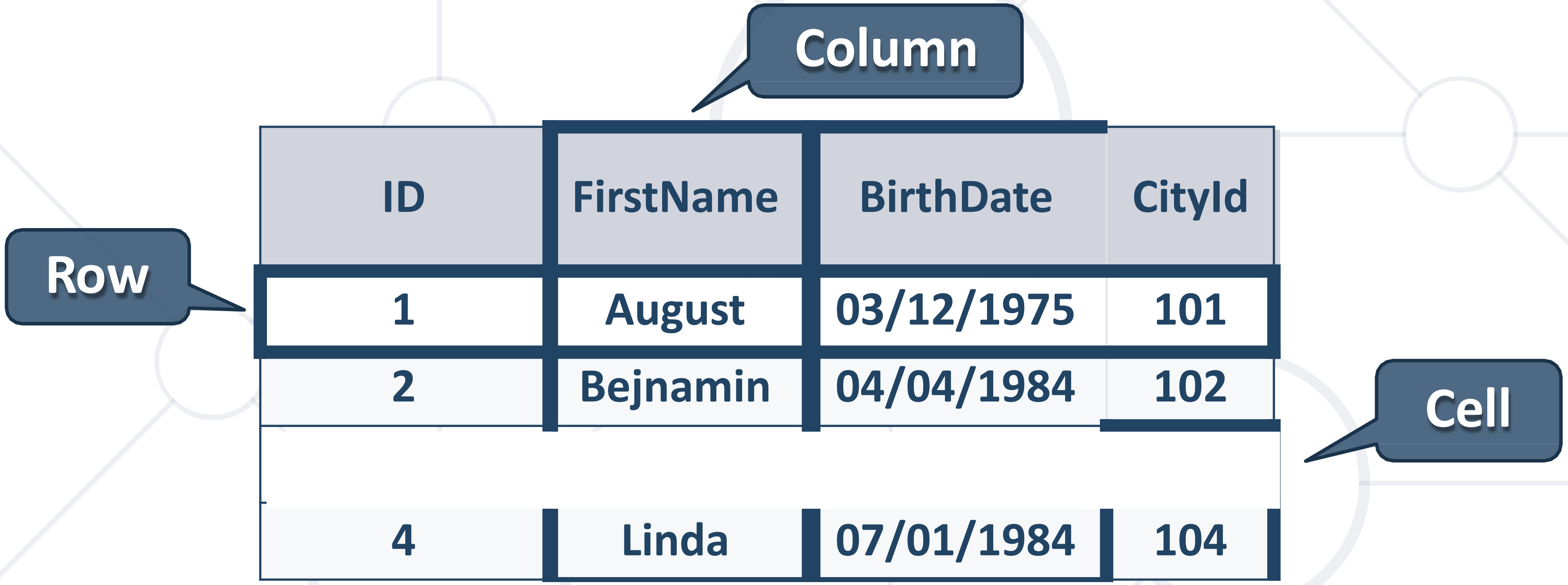
ID	Name	Email
5	Peter	peter@gmail.com
6	Jayne	jayne@gmail.com

Orders

ID	Customer ID	Date	Total Price
1	5	11/1/17	323.12
2	1	11/15/17	13.99

Database Table Elements

- The **table** is the main **building block** in the relational databases



ID	FirstName	BirthDate	CityId
1	August	03/12/1975	101
2	Bejnamin	04/04/1984	102
4	Linda	07/01/1984	104

- Each **row** is called a **record** or **entity**
- Columns (**fields**) define the **type** of data they contain

Non-Relational Database

NoSQL Database

NoSQL (Non-Relational) Databases

- A **NoSQL** databases have dynamic schema for **unstructured** data
- Data is stored in many ways:
  - Document-oriented
  - Column-oriented
  - Graph-based
  - Key-value store

NoSQL Databases

- **NoSQL databases** don't use tables and SQL
  - Instead, use **document collections** or **key-value pairs**
- More **scalable** and provide **superior performance**
- Examples: **MongoDB**, **Cassandra**, **Redis**, etc.

```
{
  ObjectId("59d3fe7ed81452db0933a871"),
  "email": peter@gmail.com,
  "age": 22
}
```

Example of JSON document in MongoDB

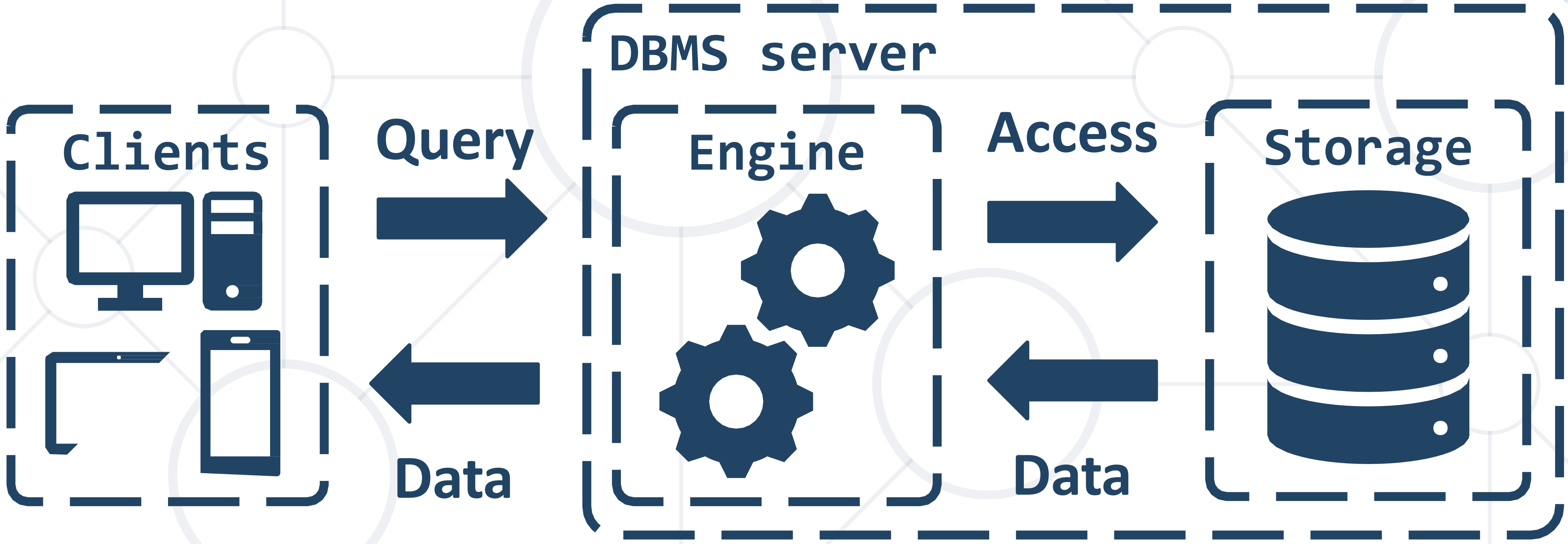
Database Management Systems

Database Management Systems (DBMS)

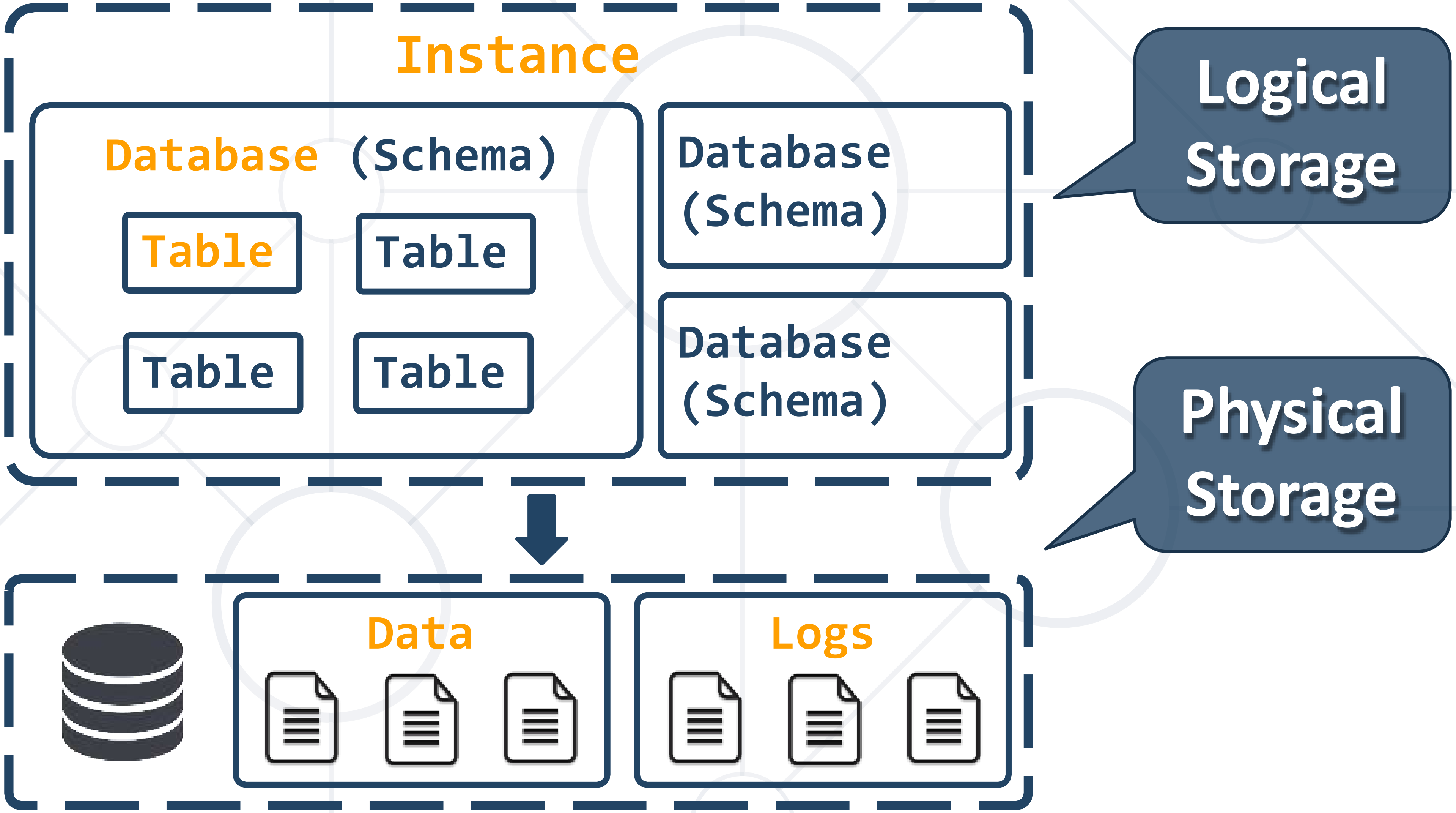
- A **D**ata**b**ase **M**anagement **S**ystem (**DBMS**) is a software, used to **define**, **manipulate**, **retrieve** and **manage** data in a database
- DBMS generally **manipulates** the data itself, the data format, field names and data types, record structure and file structure
- DBMS **examples**:
  - MySQL, MS SQL Server, Oracle, PostgreSQL
  - MongoDB, Cassandra, Redis, HBase
  - Amazon DynamoDB, Azure Cosmos DB

DBMS Systems and Data Flow

- **DBMS servers** use the **client-server model**:



DBMS Server Architecture



DBMS Systems: Examples

- **SQL databases** examples:
  - MySQL
  - PostgreSQL
  - Oracle
  - Microsoft SQL Server
  - SQLite and Web SQL
- **NoSQL databases** examples:
  - MongoDB
  - Redis
  - Google BigTable
  - Amazon DynamoDB
  - Azure Cosmos DB
  - Cassandra

Structured Query Language

Query Basics

Structured Query Language (SQL)

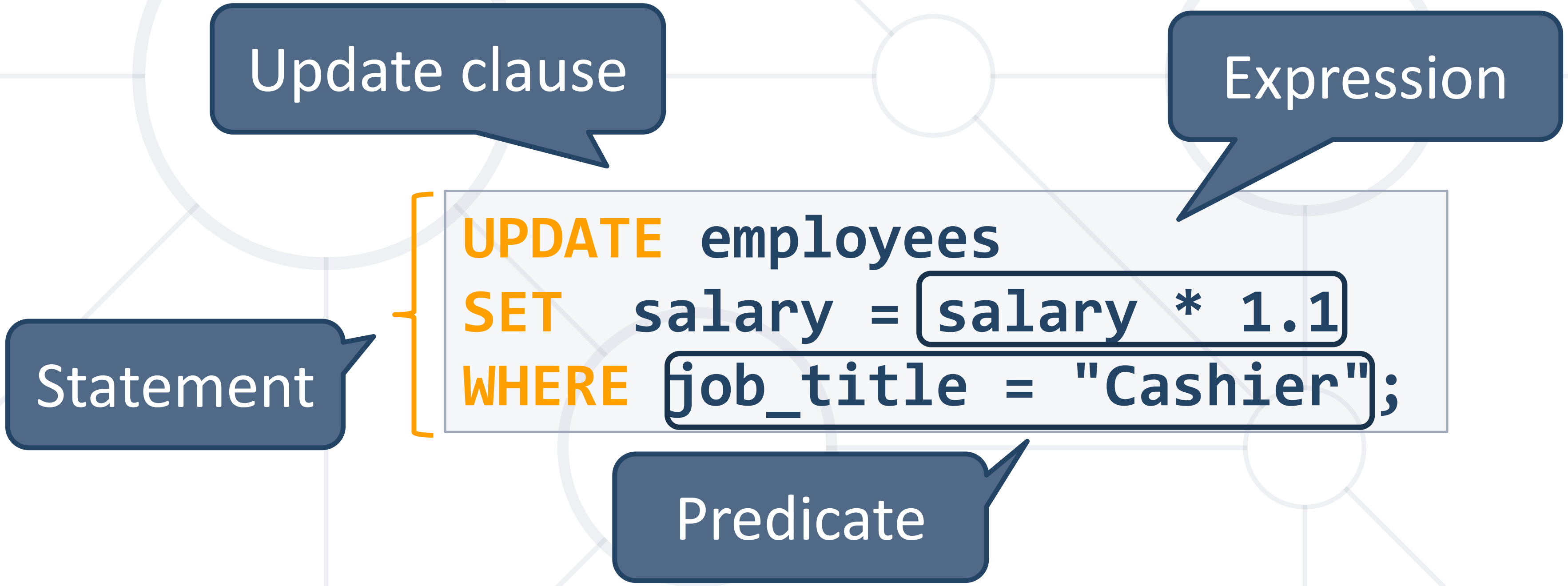
- SQL == query language designed for managing data in relational databases (RDBMS)
  - Used to communicate with the database engine
- Logically, SQL is divided into four sections:
  - **Data definition**: describe the **structure** of data
  - **Data manipulation**: **store** and **retrieve data**
  - **Data control**: define who can **access the data**
  - **Transaction control**: bundle **operations** together and perform **commit** / **rollback**

Structured Query Language (1)

- Programming language designed for managing data in a relational database
- Developed at **IBM** in the early 1970s
- To communicate with the Engine we use **SQL**

Structured Query Language (2)

- Subdivided into several language elements
  - Queries
  - Clauses
  - Expressions
  - Predicates
  - Statements



```
UPDATE employees
SET   salary = salary * 1.1
WHERE job_title = "Cashier";
```

Structured Query Language (3)

- Logically divided in four sections
  - **Data Definition** – describe the structure of our data
  - **Data Manipulation** – store and retrieve data
  - **Data Control** – define who can access the data
  - **Transaction Control** – bundle operations and allow rollback

SQL Commands

- We can **communicate** with the database engine via **SQL**
- **SQL commands** provide greater **control** and **flexibility**
- To **create a database** in MySQL:

```
CREATE DATABASE employees
```

Database name

- **Display all databases** in MySQL:

```
SHOW DATABASES
```

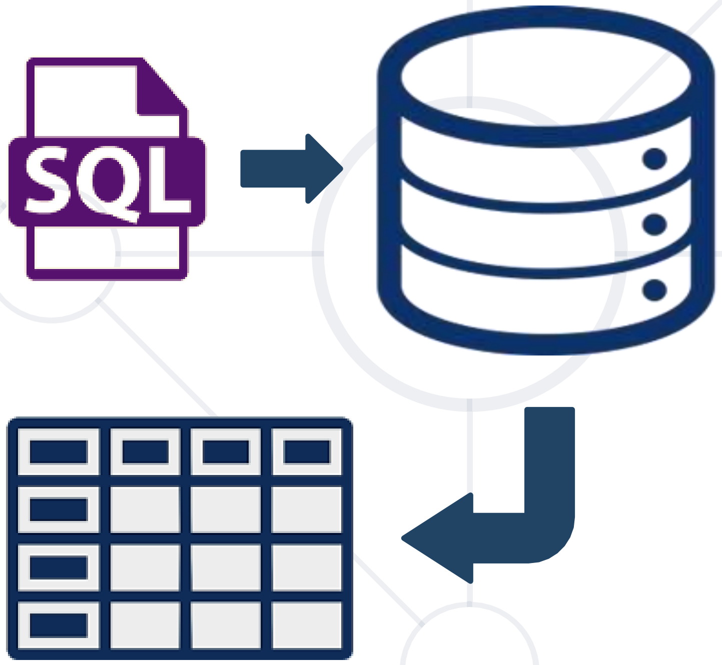
SQL – Example

- Example of **SQL query**:

```
SELECT * FROM people
```

- The query is executed by the **DBMS** system
  - It returns a sequence of **data rows**, e.g.

id	email	first_name	last_name
1	smith@yahoo.co.uk	John	Smith
2	pwh@gmail.com	Peter	White
3	anne@anne.com	Anne	Green
4	jason.jj@gmail.com	Jason	Anderson



SELECT – Example

- Selecting **all** columns from the "Departments" table

```sql
SELECT * FROM Departments
```

DepartmentID	Name	ManagerID
1	Engineering	12
2	Tool design	4
3	Sales	273
…	…	…

- Selecting **specific** columns

```sql
SELECT DepartmentId, Name
  FROM Departments
```

DepartmentID	Name
1	Engineering
2	Tool design
3	Sales
…	…

SQL – Examples

```
SELECT FirstName, LastName, JobTitle FROM Employees
```

```
SELECT * FROM Projects WHERE StartDate = '1/1/2006'
```

```
INSERT INTO Projects(Name, StartDate)
VALUES('Introduction to SQL Course', '1/1/2006')
```

```
UPDATE Projects
   SET EndDate = '8/31/2006'
 WHERE StartDate = '1/1/2006'
```

```
DELETE FROM Projects
      WHERE StartDate = '1/1/2006'
```

Filtering the Selected Rows

- Use **DISTINCT** to eliminate **duplicate** results

```
SELECT DISTINCT DepartmentID
  FROM Employees
```

- Filter rows by specific **conditions** using the **WHERE** clause

```
SELECT LastName, DepartmentID
  FROM Employees
 WHERE DepartmentID = 1
```

- Other **logical operators** can be used for greater control

```
SELECT LastName, Salary FROM Employees
 WHERE Salary <= 20000
```

Sorting Result Sets

- Sort rows with the **ORDER BY** clause
  - **ASC**: ascending order, default
  - **DESC**: descending order

```
  SELECT LastName, HireDate
    FROM Employees
ORDER BY HireDate
```

LastName	HireDate
Gilbert	1998-07-31
Brown	1999-02-26
Tamburello	1999-12-12
…	…

```
  SELECT LastName, HireDate
    FROM Employees
ORDER BY HireDate DESC
```

LastName	HireDate
Valdez	2005-07-01
Tsoflias	2005-07-01
Abbas	2005-04-15
…	…

JSON Data Format

Definition and Syntax

JSON Data Format

- **JSON** (**J**ava**S**cript **O**bject **N**otation) is a lightweight data format
  - Human and machine-readable plain text
  - Based on **JavaScript** objects
  - Independent of development platforms and languages
  - JSON data consists of:
    - Values (strings, numbers, etc.)
    - Key-value pairs: **`{ key : value }`**
    - Arrays: **`[value1, value2, …]`**

```
{
    "firstName": "Pesho",
    "courses": ["C#", "JS", "ASP.NET"]
    "age": 23,
    "hasDriverLicense": true,
    "date": "2012-04-23T18:25:43.511Z",
    // ...
}
```

JSON Data Format (2)

- The JSON data format follows the rules of object creation in JS
  - **Strings**, **numbers** and **Booleans** are valid JSON:

```
"this is a string and is valid JSON"
```

```
3.14
```

```
true
```

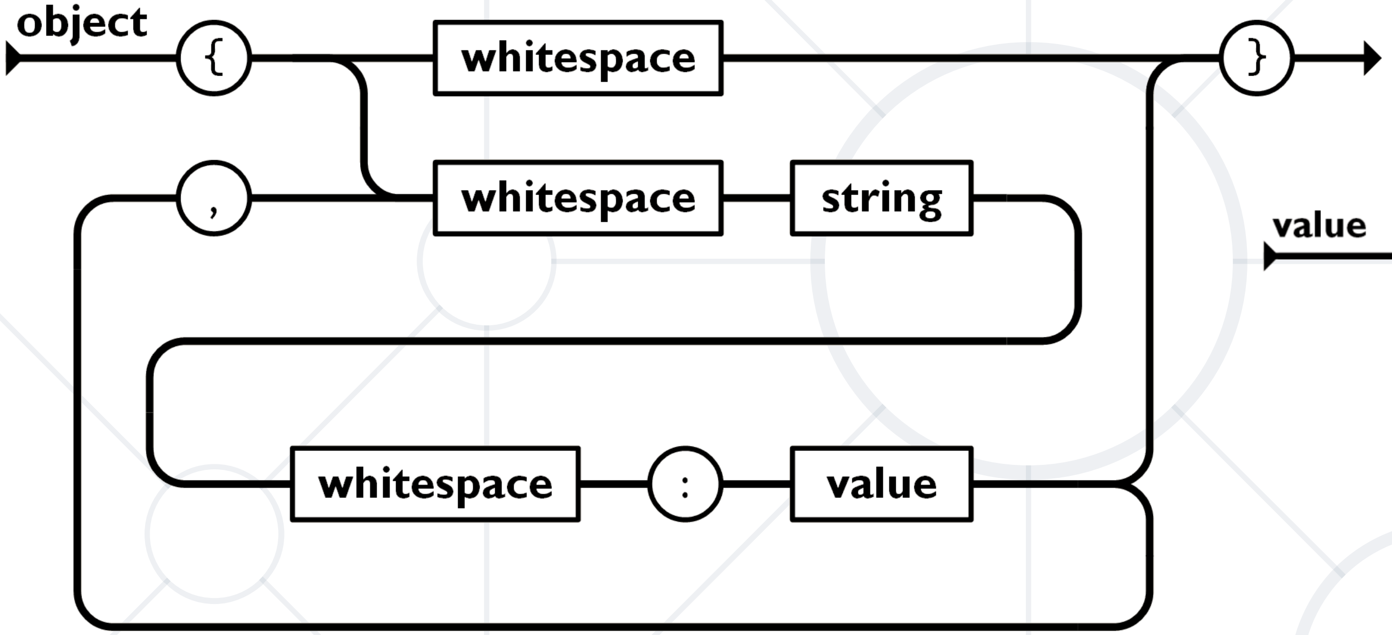
  - **Arrays** are valid JSON:

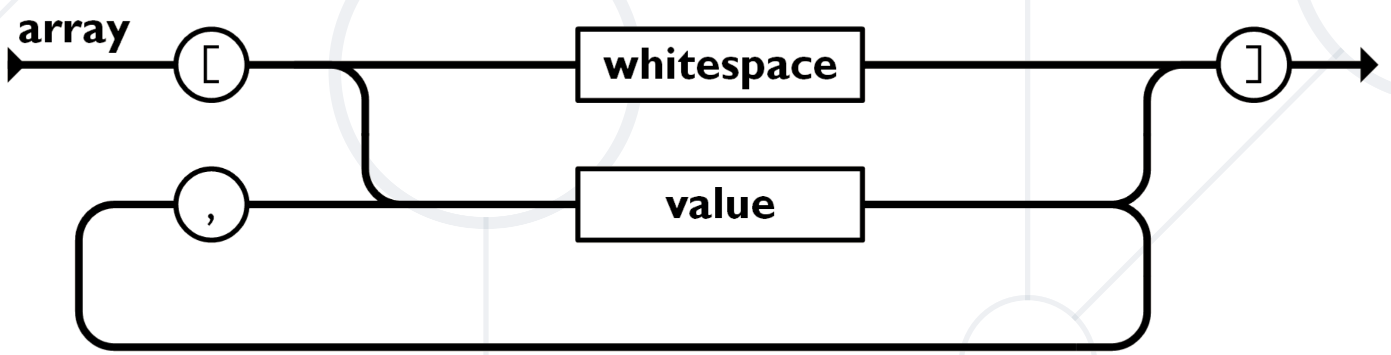
```
[5, "text", true]
```

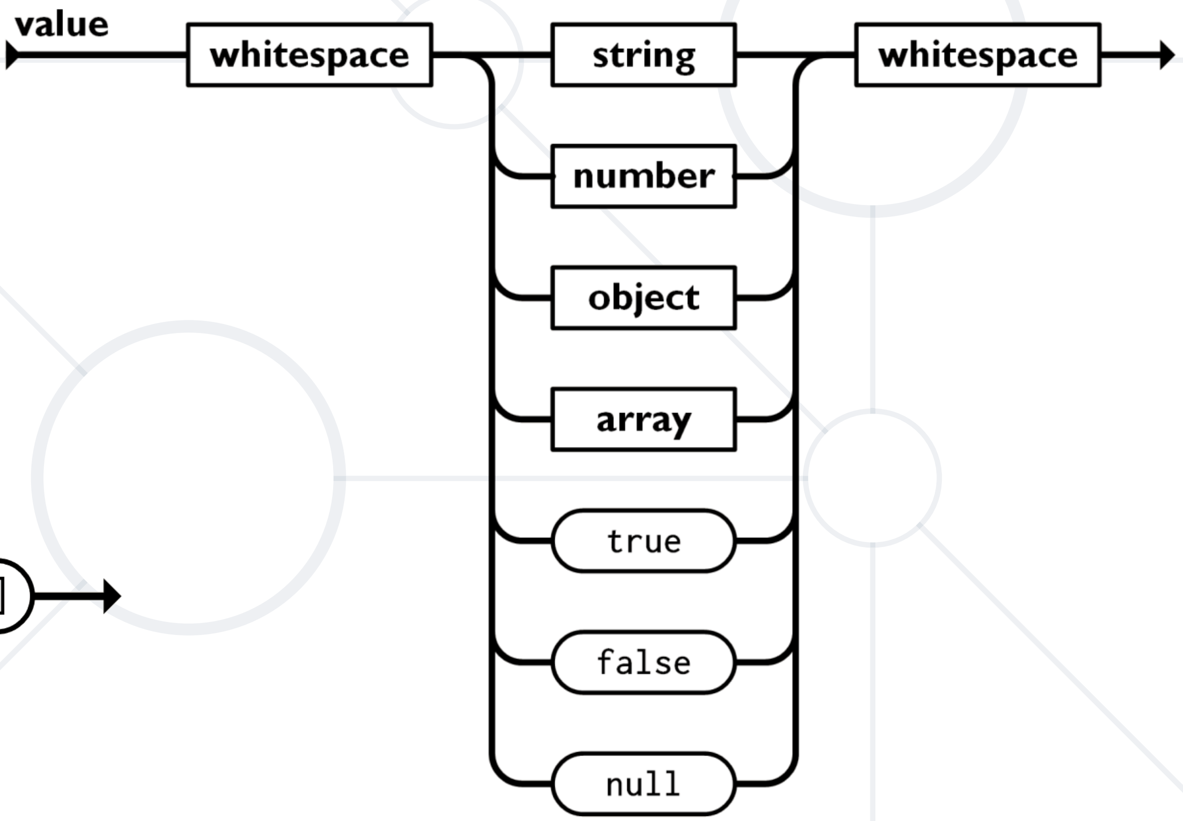
  - **Objects** are valid JSON (key-value pairs):

```
{
  "firstName": "Svetlin", "lastName": "Nakov",
  "jobTitle": "Technical Trainer", "age": 40
}
```

Object, Array and Value in JSON







MySQL

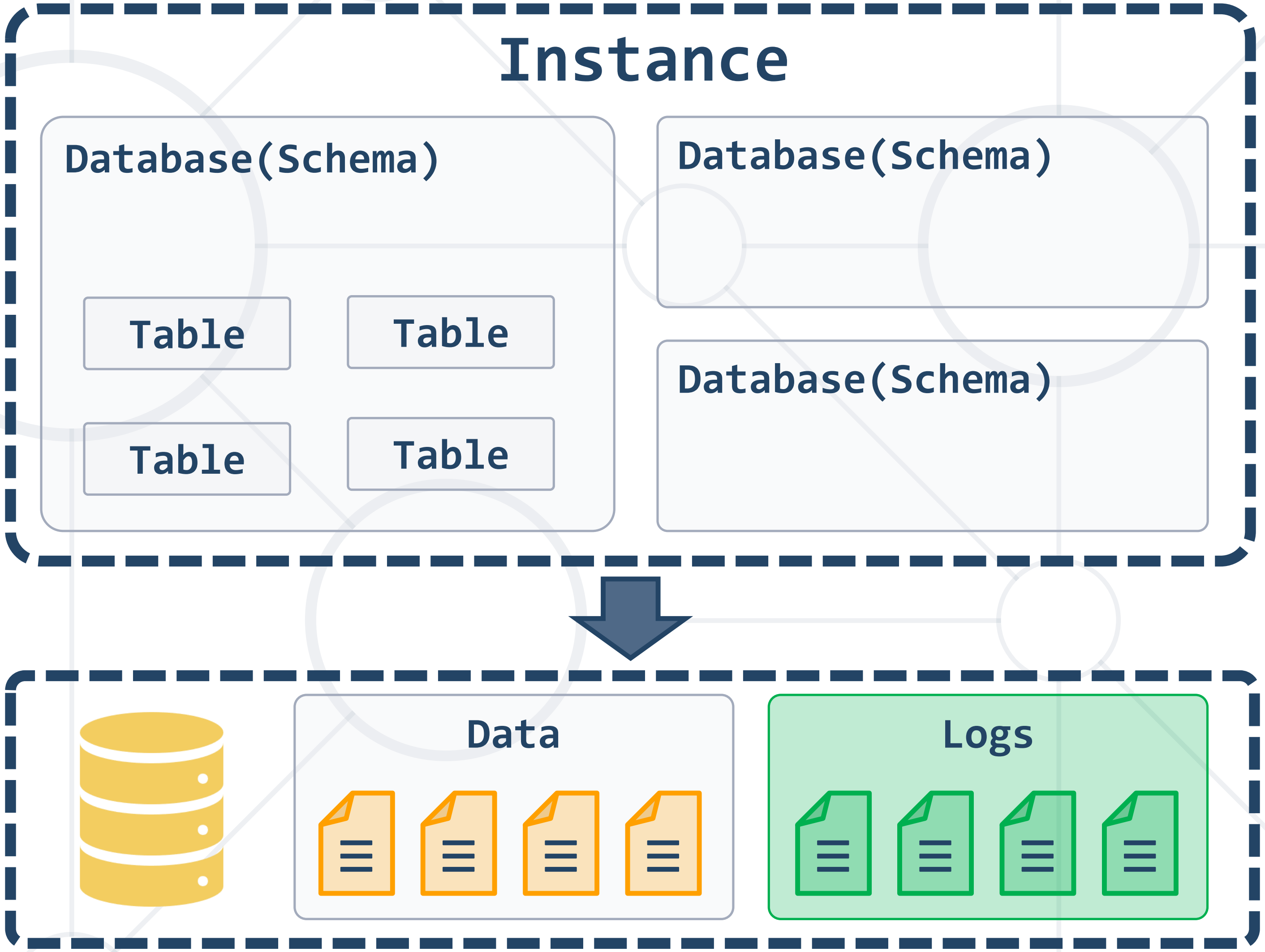
Working with Relational Database

MySQL

- **Open-source** relational database management system
- Used in many **large-scale websites** like including Google, Facebook, YouTube etc.
- Works on **many** system platforms – MAC OS, Windows, Linux
- Download **MySQL Server**
  - **Windows**: https://dev.mysql.com/downloads/mysql/
  - **Ubuntu/Debian**: https://dev.mysql.com/downloads/repo/apt/

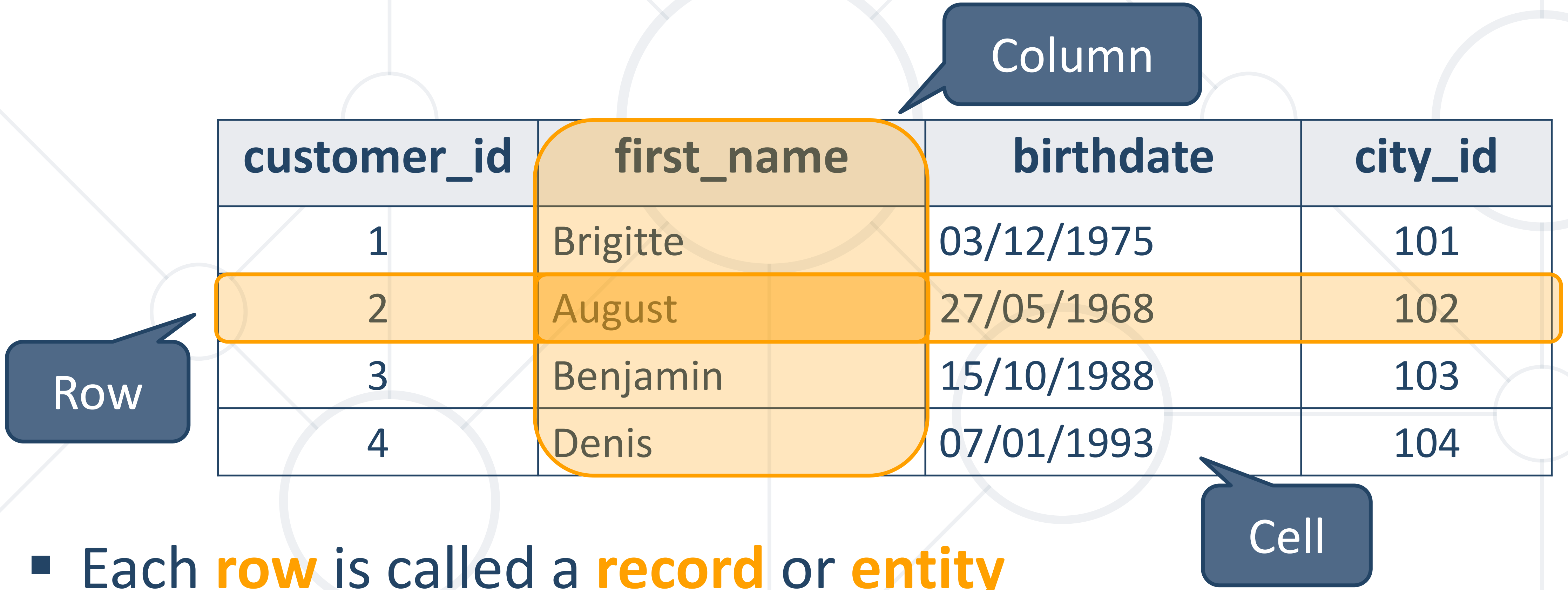
MySQL Server Architecture

- Logical Storage
  - Instance
  - Database/Schema
  - Table
- Physical Storage
  - Data files and Log files
  - Data pages



Database Table Elements

- The table is the main **building block** of any database



customer_id	first_name	birthdate	city_id
1	Brigitte	03/12/1975	101
2	August	27/05/1968	102
3	Benjamin	15/10/1988	103
4	Denis	07/01/1993	104

- Each **row** is called a **record** or **entity**
- Columns (**fields**) define the **type** of data they contain

Why Split Related Data?

Empty records

first	last	registered	email	email2
David	Rivers	05/02/2016	drivers@mail.cx	*NULL*
Sarah	Thorne	07/17/2016	sarah@mail.cx	*NULL*
Michael	Walters	11/23/2015	walters_michael@mail.cx	*walters_michael@abv.bg*

Redundant information

		customer	product	s/n	price
00315	07/16/2016	David Rivers	Oil Pump	OP147-0623	69.90
00315	07/16/2016	David Rivers	Accessory Belt	AB544-1648	149.99
00316	07/17/2016	Sarah Thorne	Wiper Fluid	WF000-0001	99.90
00317	07/18/2016	Michael Walters	Oil Pump	OP147-0623	69.90

Related Tables

- We split the data and introduce **relationships** between the tables to **avoid** repeating information

user_id	first	last	registered
203	David	Rivers	05/02/2016
204	Sarah	Thorne	07/17/2016
205	Michael	Walters	11/23/2015

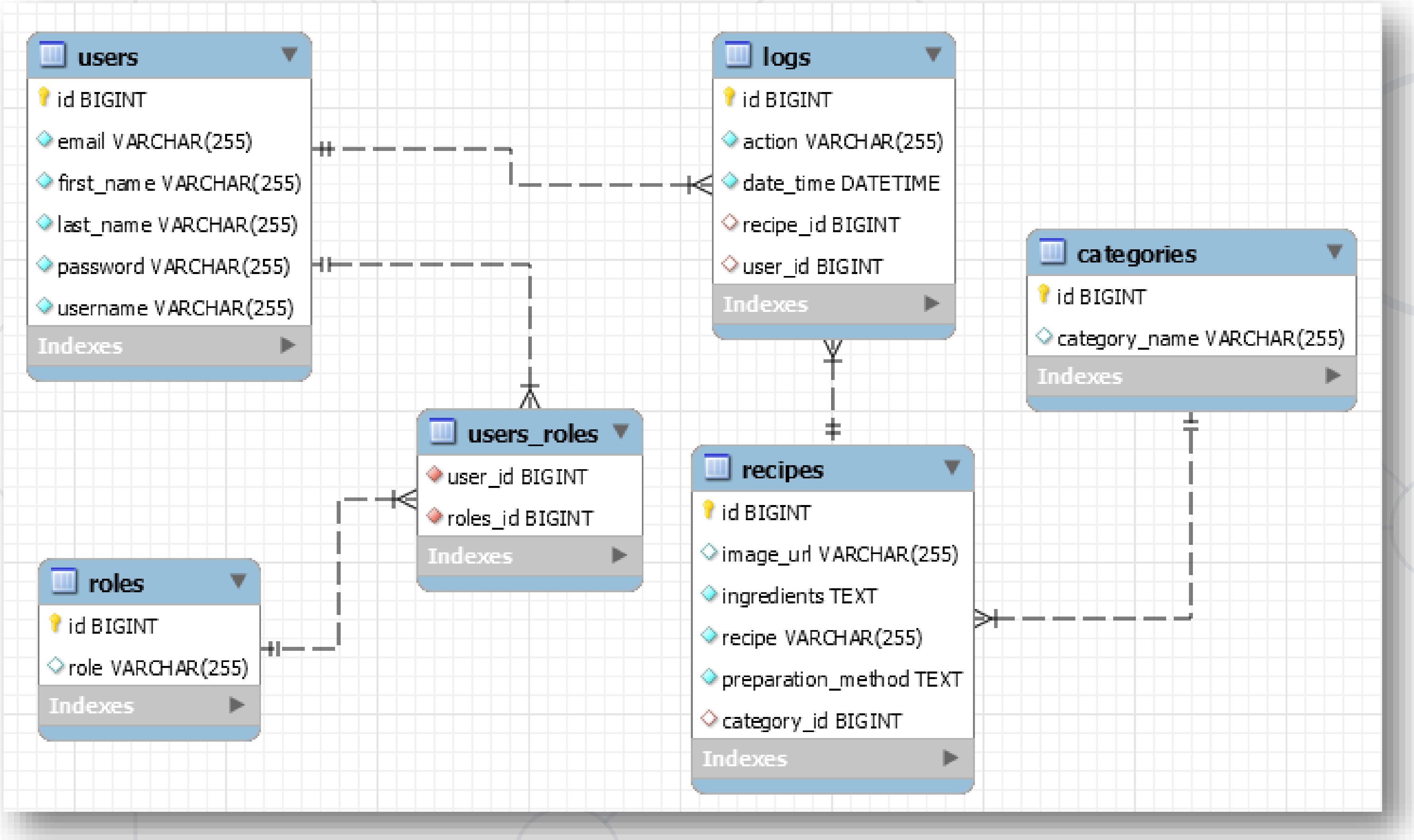
user_id	email
203	drivers@mail.cx
204	sarah@mail.cx
205	walters_michael@mail.cx
203	david@homedomain.cx

Primary Key

Foreign Key

- Connection via **Foreign Key** in one table pointing to the **Primary Key** in another

E/R Diagrams


users
id BIGINT
email VARCHAR(255)
first_name VARCHAR(255)
last_name VARCHAR(255)
password VARCHAR(255)
username VARCHAR(255)
Indexes
logs
id BIGINT
action VARCHAR(255)
date_time DATETIME
recipe_id BIGINT
user_id BIGINT
Indexes
categories
id BIGINT
category_name VARCHAR(255)
Indexes
users_roles
user_id BIGINT
roles_id BIGINT
Indexes
roles
id BIGINT
role VARCHAR(255)
Indexes
recipes
id BIGINT
image_url VARCHAR(255)
ingredients TEXT
recipe VARCHAR(255)
preparation_method TEXT
category_id BIGINT
Indexes

Mongo DB

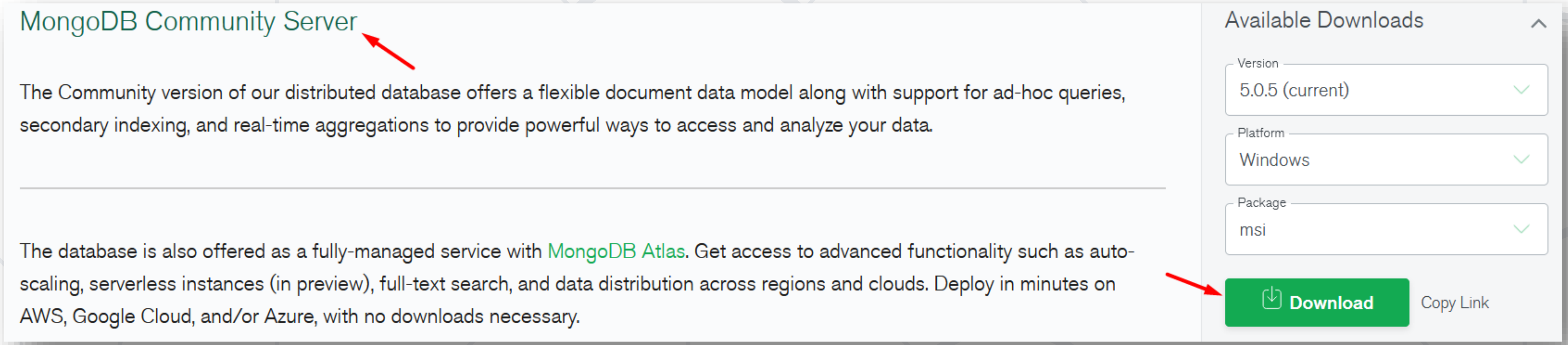
Working with Non-Relational Database

MongoDB

- **MongoDB** == free **open-source** cross-platform **document-oriented database**
  - Keeps collections of **JSON** documents (with or without schema)
- Sample usages: **mobile app** backend, product **catalog**, **poll** system, **blog** system, Web content management system (**CMS**)
- Supports evolving data requirements
  - The DB structure **may change** over the time
- Supports **indexing** for increased performance

Install MongoDB

- Download from: mongodb.com/try/download/community



MongoDB Community Server

The Community version of our distributed database offers a flexible document data model along with support for ad-hoc queries, secondary indexing, and real-time aggregations to provide powerful ways to access and analyze your data.

The database is also offered as a fully-managed service with MongoDB Atlas. Get access to advanced functionality such as auto-scaling, serverless instances (in preview), full-text search, and data distribution across regions and clouds. Deploy in minutes on AWS, Google Cloud, and/or Azure, with no downloads necessary.

Available Downloads

Version: 5.0.5 (current)

Platform: Windows

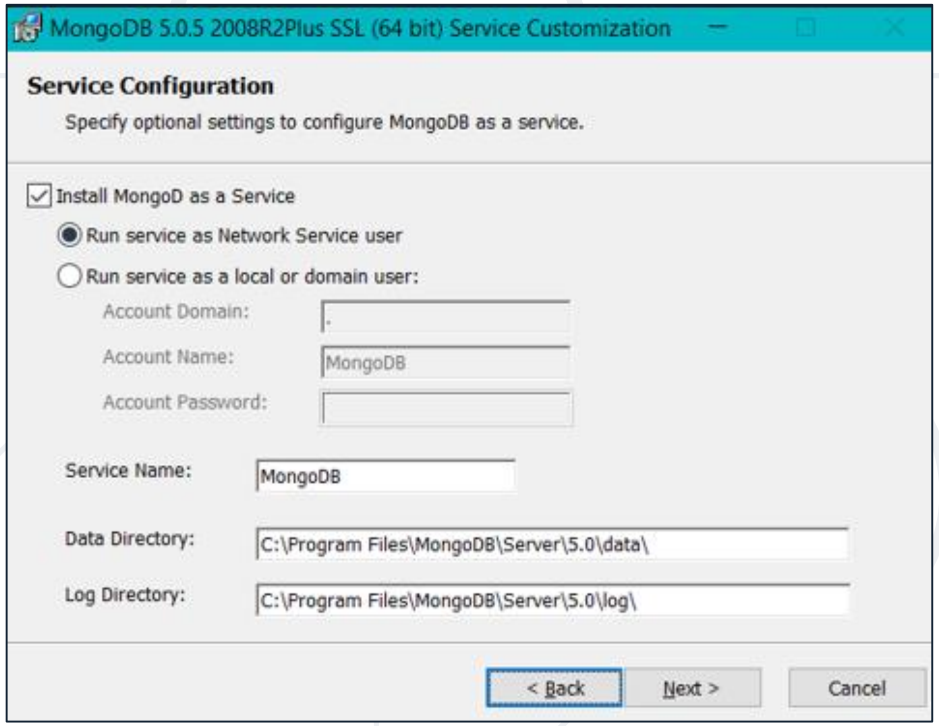
Package: msi

- The package includes **MongoDB Compass**
- When **installed**, MongoDB needs a **driver** (for every project)
  - Install MongoDB **driver** for Node.js
  - We will be using **Mongoose** (includes a driver)

```
npm install mongodb
```

MongoD Windows Service

- During installation, configure the **MongoDB service**:



MongoDB 5.0.5 2008R2Plus SSL (64 bit) Service Customization

Service Configuration

Specify optional settings to configure MongoDB as a service.

[✓] Install MongoD as a Service

(●) Run service as Network Service user

( ) Run service as a local or domain user:

Account Domain:	.
Account Name:	MongoDB
Account Password:	

Service Name:	MongoDB
Data Directory:	C:\Program Files\MongoDB\Server\5.0\data\
Log Directory:	C:\Program Files\MongoDB\Server\5.0\log\

Manual Service Configuration

- Required if you **skipped** the service installation (and for Linux)
  - Go to installation folder and **run** a command prompt as an **administrator**
  - Type the following command

Usually in C:\Program Files\MongoDB\Server\3.4\bin

```
<path to mongod.exe> mongod --dbpath <path to store data>
```

  - Additional information at https://docs.mongodb.com/manual/tutorial/

Working with MongoDB Shell Client

- Start the shell from **another** CLI
  - Type the command **mongo**

```
show dbs
```

```
use mytestdb
```

```
db.mycollection.insertOne({"name":"George"})
```

```
db.mycollection.find({"name":" George"})
```

```
db.mycollection.find({})
```

- Additional information at
  - https://docs.mongodb.com/manual/reference/mongo-shell/

Working with MongoDB GUI

- Choose one of the many (**Compass** is included in the installer)
- For example
  - Compass- https://www.mongodb.com/products/compass
  - Robo 3T- https://robomongo.org/download
  - NoSQLBooster- https://nosqlbooster.com

Mongoose Queries

- Mongoose defines **all** queries of the native MongoDB driver in a more **clear** and **useful** way
  - Instead of

```
{
  $or: [
    {conditionOne: true},
    {conditionTwo: true}
  ]
}
```

  - Do

```
.where({ conditionOne: true })
.or({ conditionTwo: true })
```

Mongoose Queries Example

- Mongoose supports **many** queries
  - For equality/non-equality

```
Student.findOne({'lastName':'Petrov'})
```

```
Student.find({}).where('age').gt(7).lt(14)
```

```
Student.find({}).where('facultyNumber').equals('12399')
```

  - Selection of some properties

```
Student.findOne({'lastName':'Kirilov'}).select('name age')
```

Mongoose Queries Example 2

- Sorting

```
Student.find({}).sort({age:-1})
```

- Limit & skip

```
Student.find({}).sort({age:-1}).skip(10).limit(10)
```

- Different methods could be **stacked** one upon the other

```
Student.find({})
      .where('firstName').equals('gosho')
      .where('age').gt(18).lt(65)
      .sort({age:-1})
      .skip(10)
      .limit(10)
```

Summary

- Databases - Introduction
- Relational and Non-Relational Databases
- DBMS
- SQL Commands
- JSON Data Format
- Working with MySQL + Workbench
- Working with Mongo DB + Compass

Questions?

Software University

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

Finance Academy

SoftUni Diamond Partners

Software University

SUPER HOSTING .BG

Coca-Cola HBC Bulgaria

POKERSTARS
POKER | CASINO | SPORTS
a Flutter International brand

INDEAVR
Serving the high achievers

AMBITIONED

DRAFT KINGS

SOFTWARE GROUP

createX

Postbank
Решения за твоето утре

BOSCH

DXC TECHNOLOGY

SmartIT

Educational Partners

License

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg