Introduction to Databases
Data Definition and Datatypes

How Do RDBMS Work?

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

Table of Contents

1. Data Management

2. Database Engine

3. Structured Query Language

4. MySQL

5. Table Relationships

6. Programmability

Table of Contents

sli.do

# #

Data Management

Storage vs. Management (1)

SALES RECEIPT

Date: 07/16/2016
Order#:[00315]

Customer: David Rivers
Product: Oil Pump
S/N: OP147-0623

Unit Price: 69.90
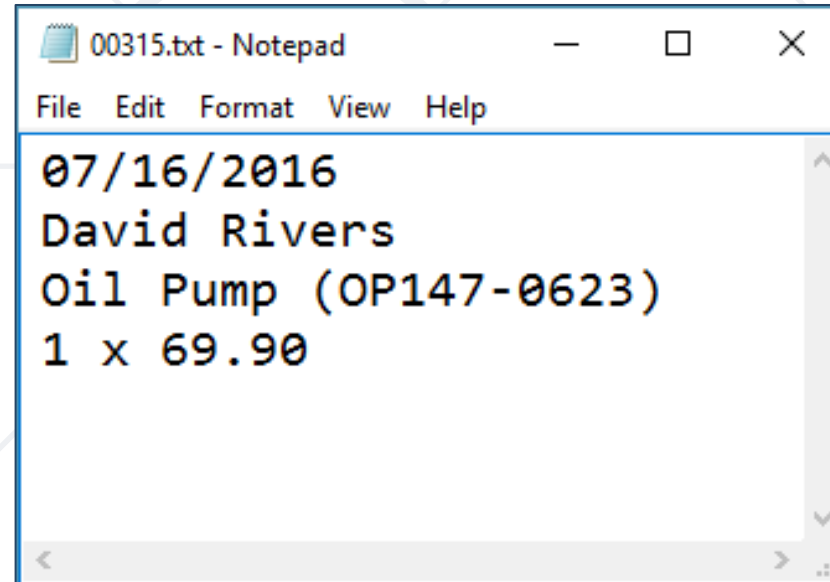Qty: 1

Total: 69.90

> 00315 – 07/16/2016
> David Rivers
> Oil Pump (OP147-0623)
> 1 x 69.90

Storage vs. Management (2)

```
00315.txt - Notepad
File  Edit  Format  View  Help

07/16/2016
David Rivers
Oil Pump (OP147-0623)
1 x 69.90
```

Order#	Date	Customer	Product	S/N	Qty
00315	07/16/2016	David Rivers	Oil Pump	OP147-063	1

Storage vs. Management (3)

- Storing data is **not** the primary reason to use a database

- Flat storage **eventually** runs into **issues** with

  - Size

  - Ease of updating

  - Accuracy

  - Security

  - Redundancy

  - Importance

Databases

- A database is an **organized** collection of **related** information
  - It imposes **rules** on the contained data
  - Access to data is usually provided by a "**system**" (DBMS) **database management**
  - Relational storage first proposed by Edgar Codd in 1970
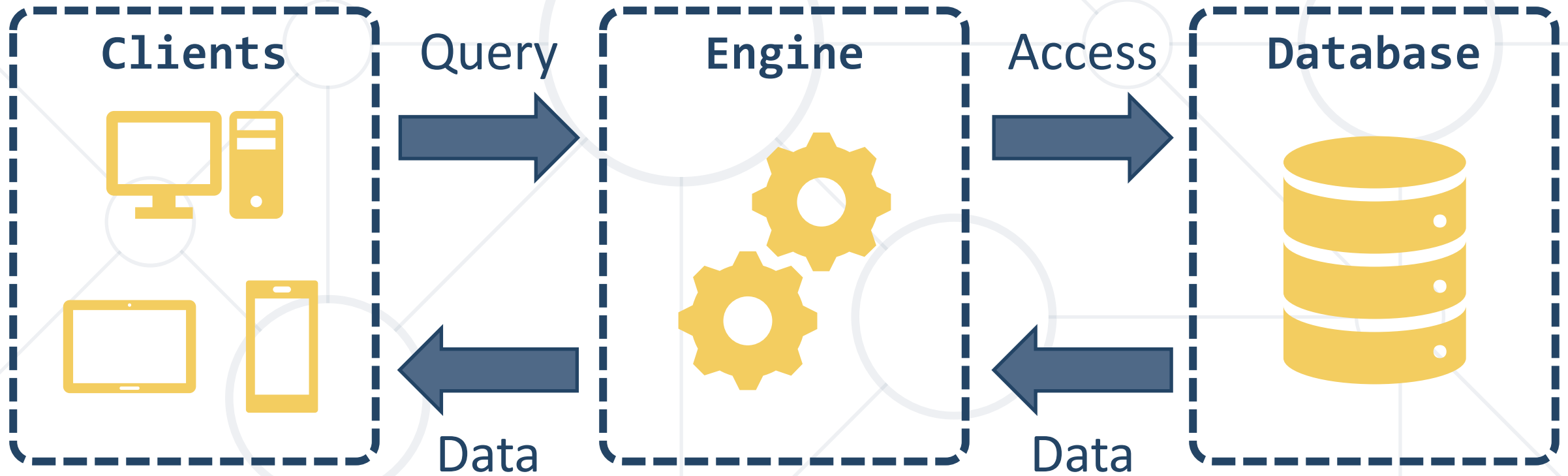
RDBMS

- **R**elational **D**ata **B**ase **M**anagement **S**ystem
  - Database **management**
  - It **parses requests** from the user and takes the **appropriate** action
  - The user **doesn't have direct access** to the stored data
  - Data is presented by **relations** – collection of tables related by **common fields**
  - MS SQL Server, DB2, Oracle and MySQL

Database Engine

Database Engine Flow

- SQL Server uses the Client-Server Model

Top Database Engines

Rank			DBMS	Database Model	Score		
Aug 2022	Jul 2022	Aug 2021			Aug 2022	Jul 2022	Aug 2021
1.	1.	1.	Oracle ➕	Relational, Multi-model ℹ️	1260.80	-19.50	-8.46
2.	2.	2.	MySQL ➕	Relational, Multi-model ℹ️	1202.85	+7.98	-35.37
3.	3.	3.	Microsoft SQL Server ➕	Relational, Multi-model ℹ️	944.96	+2.83	-28.39
4.	4.	4.	PostgreSQL ➕	Relational, Multi-model ℹ️	618.00	+2.13	+40.95
5.	5.	5.	MongoDB ➕	Document, Multi-model ℹ️	477.66	+4.68	-18.88
6.	6.	6.	Redis ➕	Key-value, Multi-model ℹ️	176.39	+2.77	+6.51
7.	7.	7.	IBM Db2	Relational, Multi-model ℹ️	157.23	-3.99	-8.24
8.	8.	8.	Elasticsearch	Search engine, Multi-model ℹ️	155.08	+0.75	-2.01
9.	9.	↑10.	Microsoft Access	Relational	146.50	+1.41	+31.66
10.	10.	↓9.	SQLite ➕	Relational	138.87	+2.20	+9.06

Source: http://db-engines.com/en/ranking

Structured Query Language

Structured Query Language (1)

- Programming language designed for managing data in a relational database

- Developed at **IBM** in the early 1970s
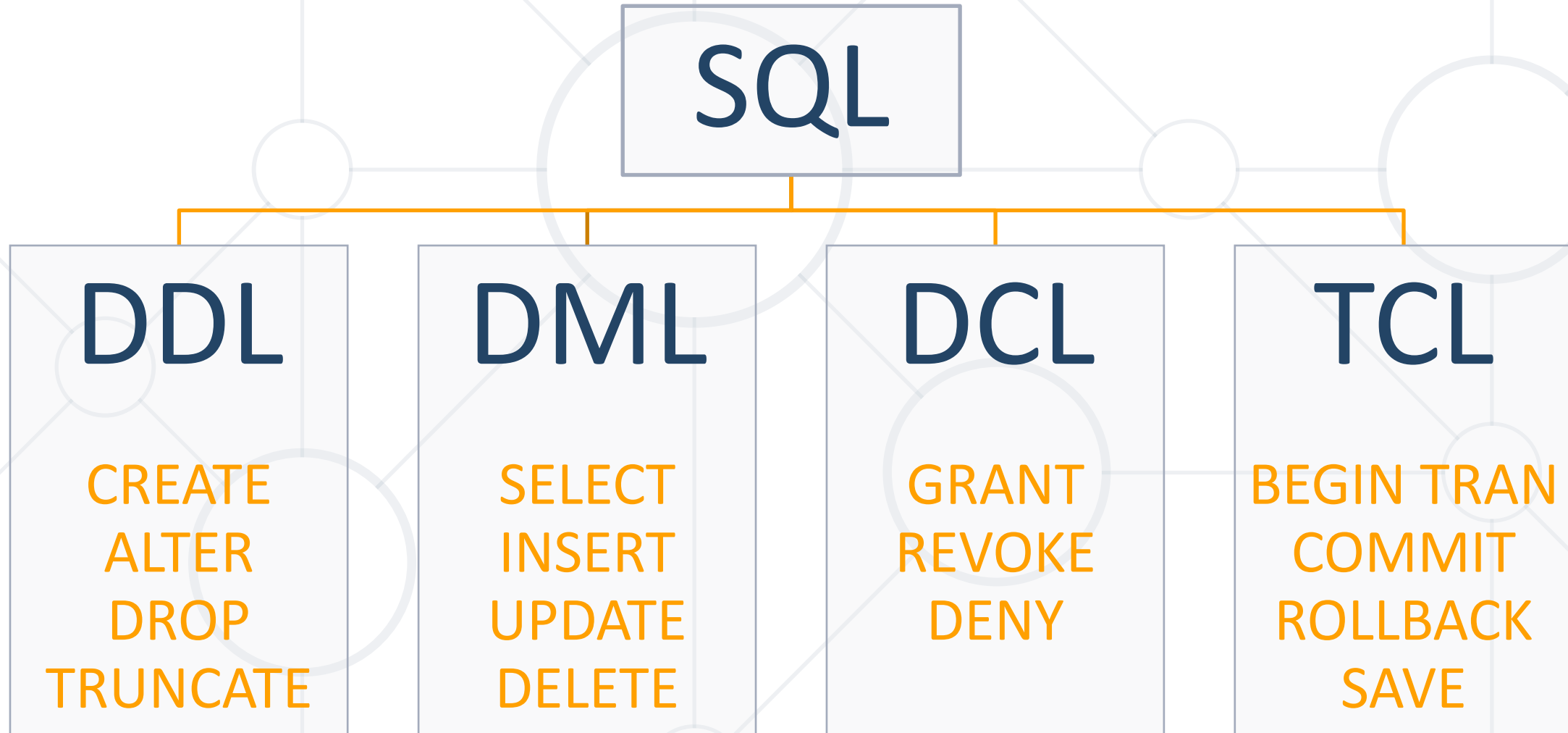
- To communicate with the Engine we use **SQL**

Structured Query Language (2)

- Subdivided into several language elements

  - Queries

  - Clauses

  - Expressions

  - Predicates

  - Statements

Update clause

Expression

Statement

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

Predicate

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

Structured Query Language (4)


```
                    ┌─────────────┐
                    │     SQL     │
                    └──────┬──────┘
          ┌────────────┬───┴───┬────────────┐
     ┌────┴────┐  ┌────┴────┐ ┌┴────────┐ ┌─┴────────┐
     │   DDL   │  │   DML   │ │   DCL   │ │   TCL    │
     │         │  │         │ │         │ │          │
     │ CREATE  │  │ SELECT  │ │ GRANT   │ │BEGIN TRAN│
     │ ALTER   │  │ INSERT  │ │ REVOKE  │ │ COMMIT   │
     │ DROP    │  │ UPDATE  │ │ DENY    │ │ ROLLBACK │
     │TRUNCATE │  │ DELETE  │ │         │ │ SAVE     │
     └─────────┘  └─────────┘ └─────────┘ └──────────┘
```

MySQL

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
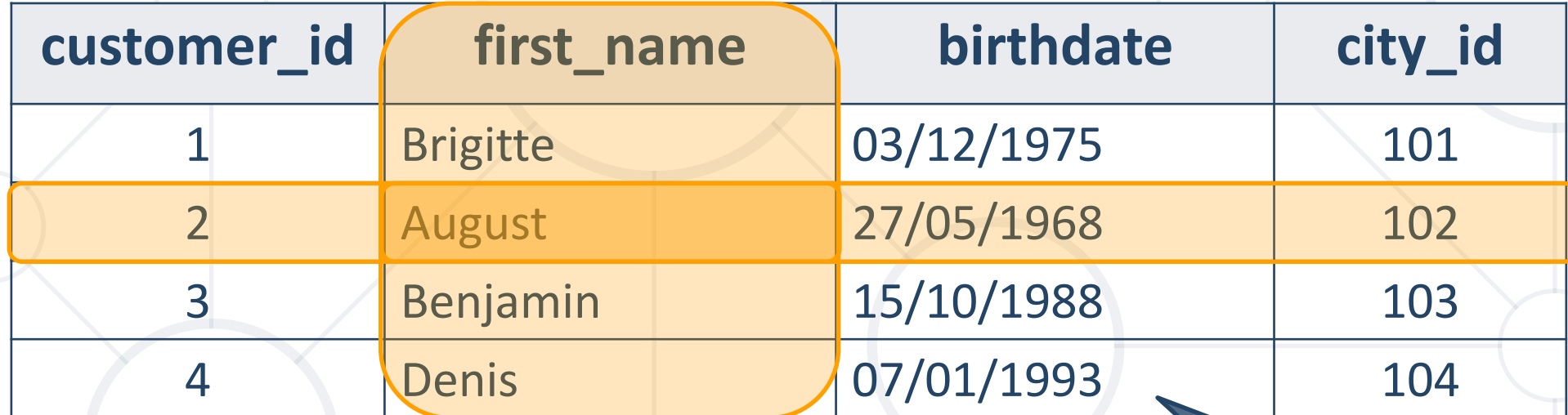
- Download **MySQL Server**

  - **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

Instance

Database(Schema)

| Table | Table |

| Table | Table |

Database(Schema)

Database(Schema)

Data

Logs

22
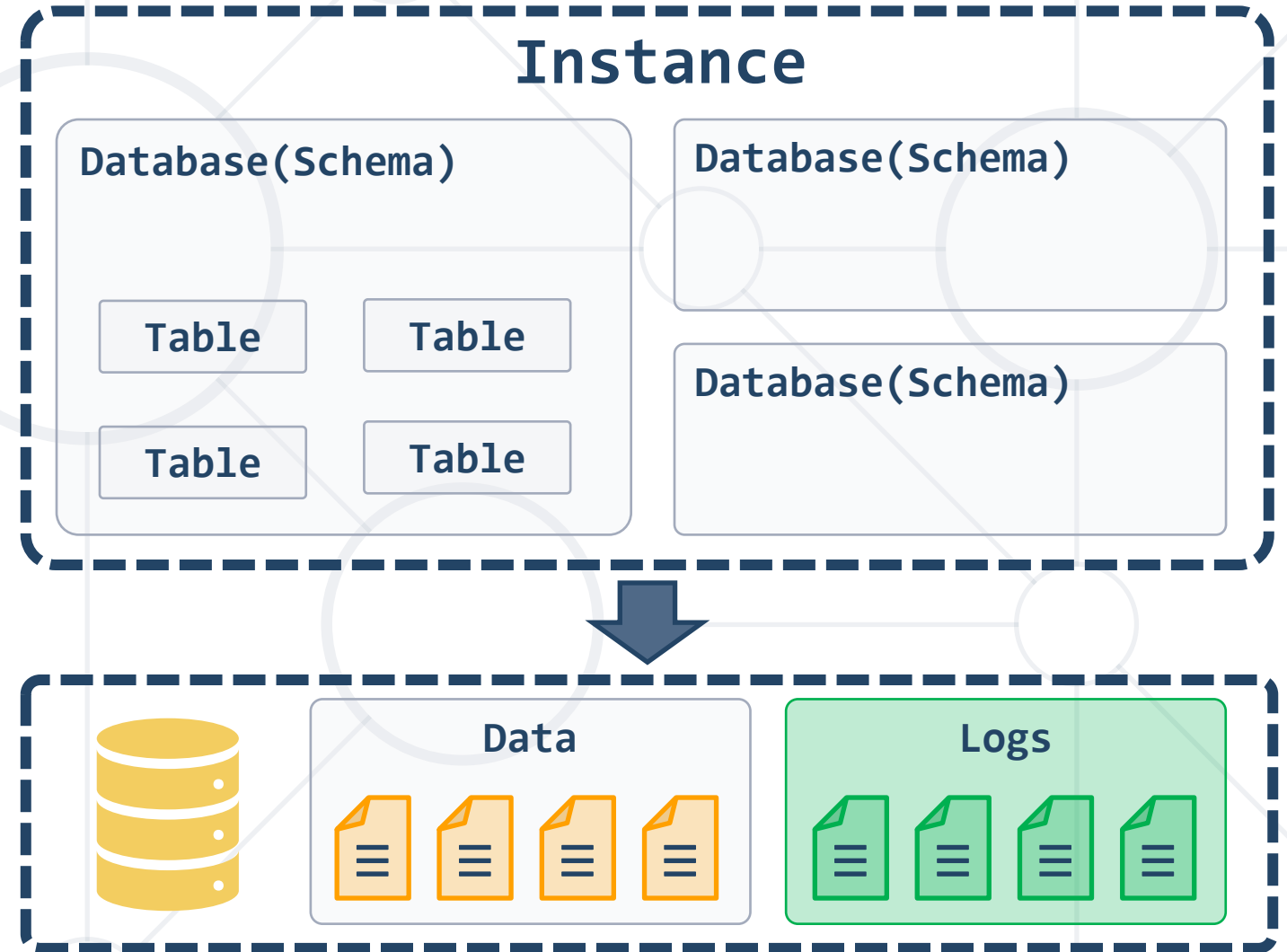
Database Table Elements

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

Column

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

Row

Cell

- Each **row** is called a **record** or **entity**

- Columns (**fields**) define the **type** of data they contain
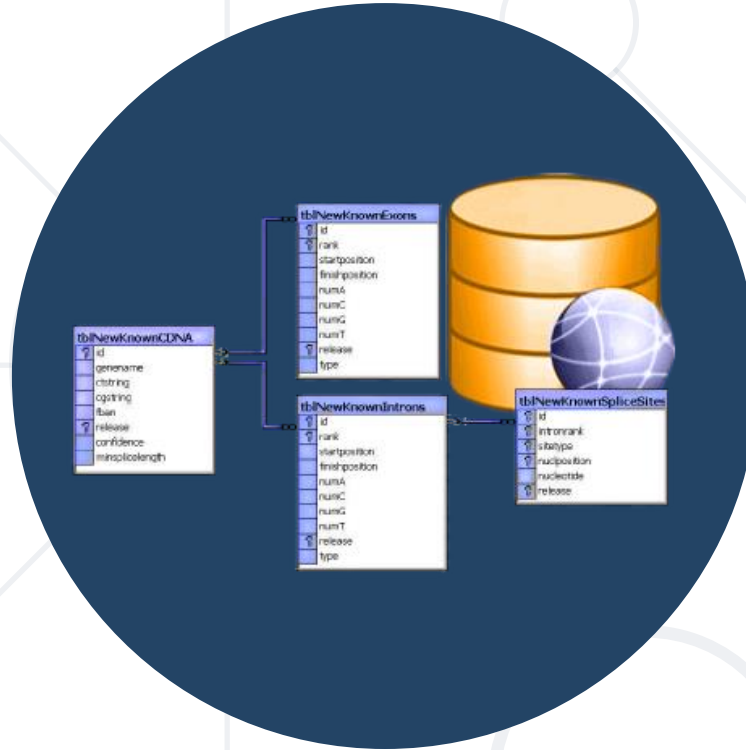
Table Relationships

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
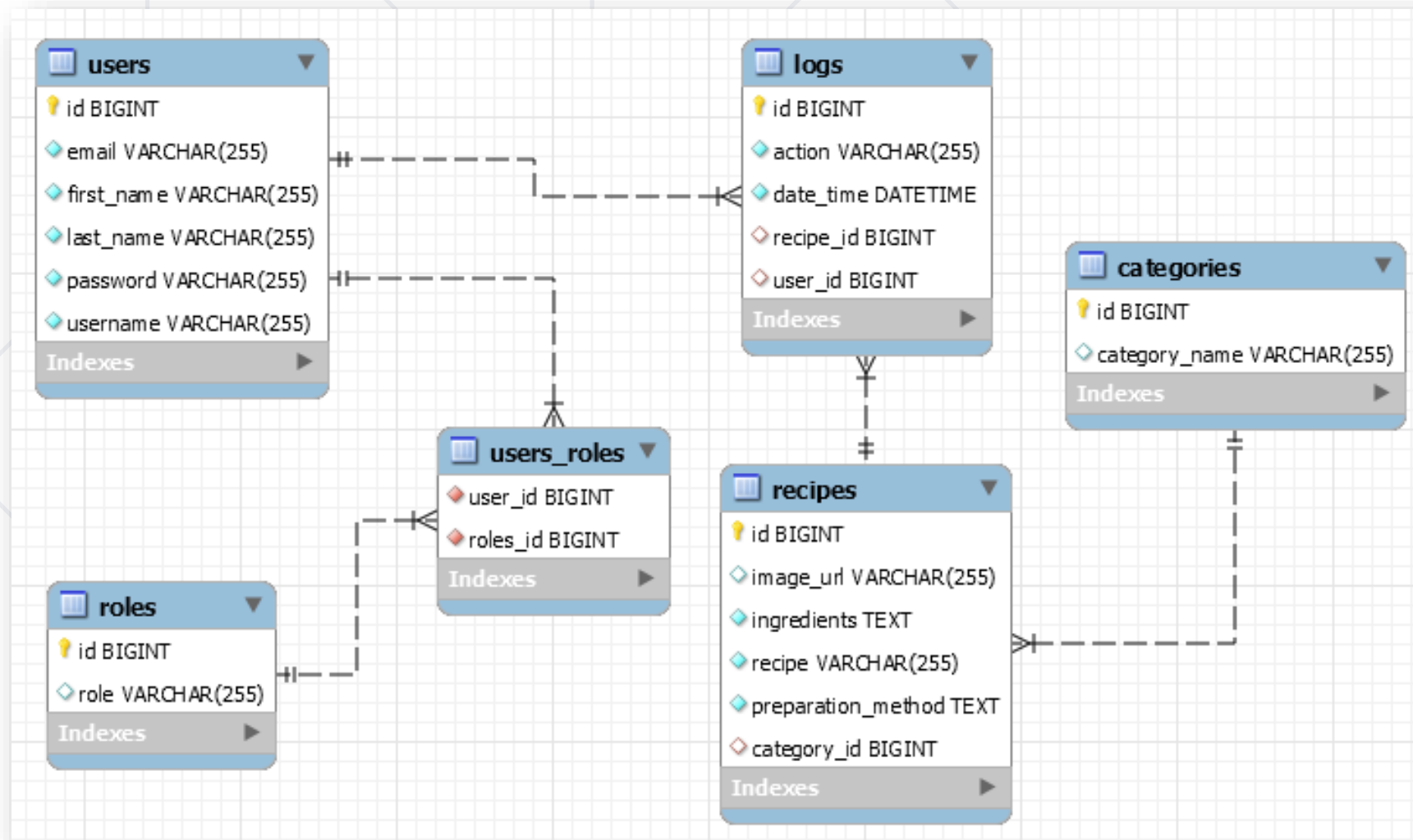
Programmability

Indices

- Indices make data lookup faster

  - Clustered – bound to the **primary key**, physically sorts data

  - Non-Clustered – can be **any field**, references the primary index

- Structured as an **ordered tree**
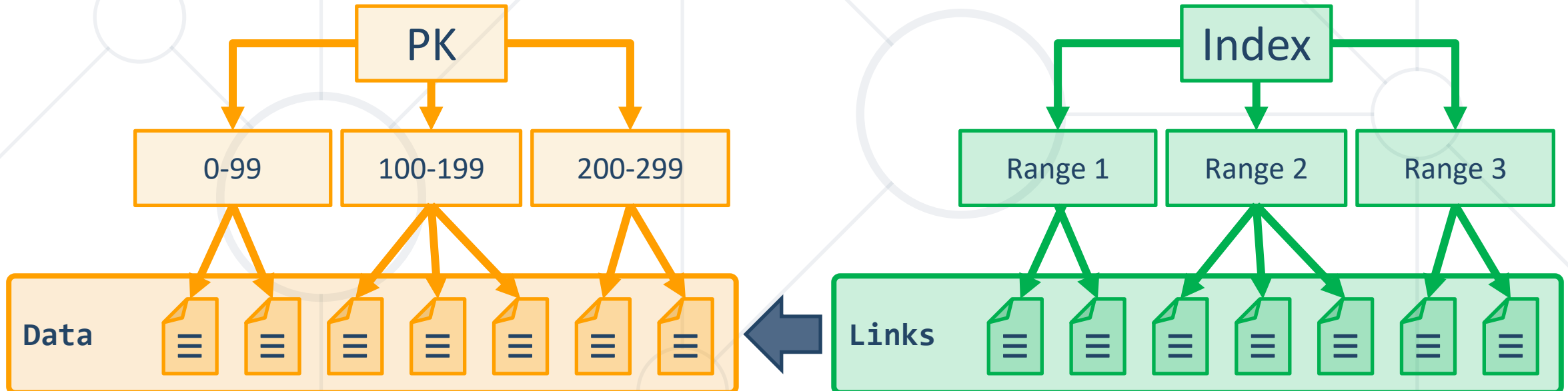
Views

- Views are **prepared queries** for displaying **sections** of our data

```
CREATE VIEW v_employee_names AS
     SELECT employee_id,
            first_name,
            last_name
     FROM employees
```

```
SELECT * FROM v_employee_names
```

- Evaluated at **run time** – they do not increase performance

Procedures, Functions and Triggers

- A database can further be customized with reusable code

- **Procedures** – carry out a predetermined **action**
  - E.g. get all employees with salary above 35000

- **Functions** – receive **parameters** and return a **result**
  - E.g. get the age of a person using their birthdate and current date

- **Triggers** – **watch** for activity in the database and **react** to it
  - E.g. when a record is deleted, write it to an archive

Data Types in MySQL Server

Numeric Data Types

- Numeric data types have certain range

- Their range can be changed if they are:

  - **Signed** - represent numbers both in the positive **and** negative ranges

  - **Unsigned** - represent numbers **only** in the positive range

- E.g. signed and unsigned INT:

Signed Range		Unsigned Range	
Min Value	Max Value	Min Value	Max Value
-2147483648	2147483648	0	4294967295

Numeric Data Types

- **INT** [(M)] [UNSIGNED]
  - TINYINT, SMALLINT, MEDIUMINT, BIGINT

- **DOUBLE** [(M, D)] [UNSIGNED]

Digits stored for value

Decimals after floating point

  - E.g. DOUBLE[5, 2] – 999.99

- **DECIMAL** [(M, D)] [UNSIGNED] [ZEROFILL]

String Types (1)

- String column definitions include attributes that specify the **character set** or **collation**
  - **CHARACTER SET** (Encoding)

    > Determines the storage of each character (single or multiple bytes)

    - E.g. utf8, ucs2
  - **CHARACTER COLLATION** – rules for encoding comparison
    - E.g. latin1_general_cs, Traditional_Spanish_ci_ai etc.

    > Determines the sorting order and case-sensitivity
- Set and collation can be defined at the database, table or column level

- **ORDER BY** with different collations

latin1_swedish_ci	latin1_german1_ci	latin1_german2_ci
Muffler	Muffler	Müller
MX Systems	Müller	Muffler
Müller	MX Systems	MX Systems
MySQL	MySQL	MySQL

- **CHAR** [(M)] - up to 255 characters
  - fixed-length character type (example CHAR(30))
- **VARCHAR**(M) - up to 65 535. The effective maximum length is a subject to the maximum row size (65,535 bytes, which is shared among all columns) and the character set used
  - Variable max size
- **TEXT** - up to 65 535 characters
  - TINYTEXT, MEDIUMTEXT, LONGTEXT
- **BLOB** - **B**inary **L**arge **Ob**ject [(M)] - 65 535 (2^{16} - 1) characters
  - TINYBLOB, MEDIUMBLOB, LONGBLOB

CHAR vs VARCHAR

- Storing data in CHAR and VARCHAR examples

Value	CHAR(4)	Storage Required	VARCHAR(4)	Storage Required
''	' '	4 bytes	''	1 bytes
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes

Date Types (1)

- **DATE** - for values with a date part but **no time part**

- **TIME** - for values with time but **no date part**

- **DATETIME** - values that contain both date **and** time parts

- **TIMESTAMP** - both date **and** time parts

> **DATETIME and TIMESTAMP have different time ranges**

Column name	Column Type
birthdate	DATE
last_time_online	TIMESTAMP
start_at	TIME
deleted_on	DATETIME

Date Types (2)

- MySQL retrieves values for a given date type in a **standard output format**
  - E.g. as a string in either 'YYYY-MM-DD' or 'YY-MM-DD'

Data Type	Column Type
DATE	'0000-00-00'
TIME	'00:00:00'
DATETIME	'0000-00-00 00:00:00'
TIMESTAMP	'0000-00-00 00:00:00'
YEAR	0000

Database Modeling

Working with IDEs

- We will **manage** databases with **MySQL Workbench**

- Enables us:

  - To **create** a new database

  - To create **objects in the database** (tables, stored procedures, relationships and others)

  - To **change** the properties of objects

  - To **enter records** into the tables

Creating a New Database

- Select **Create new schema** from the **command menu**
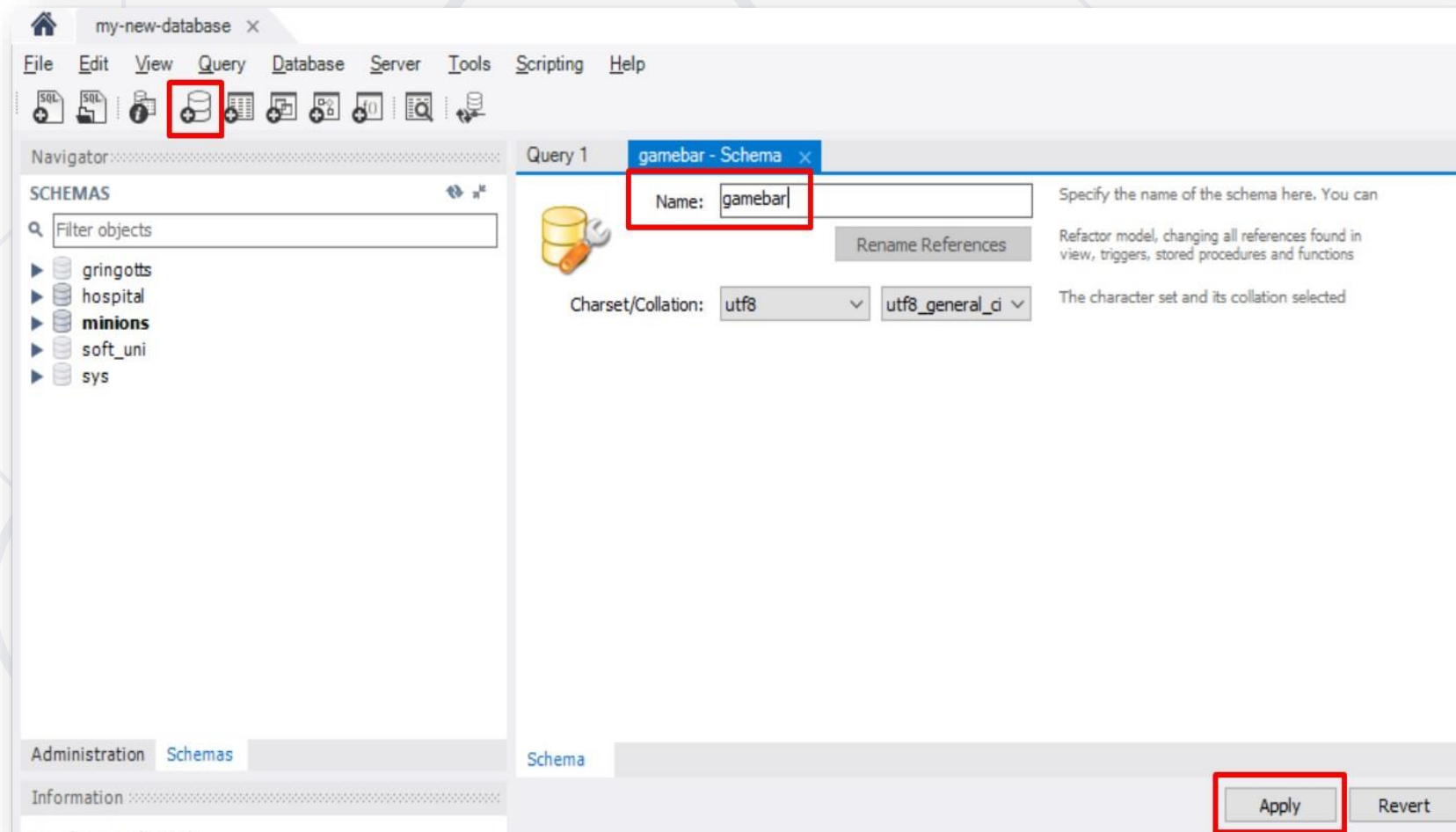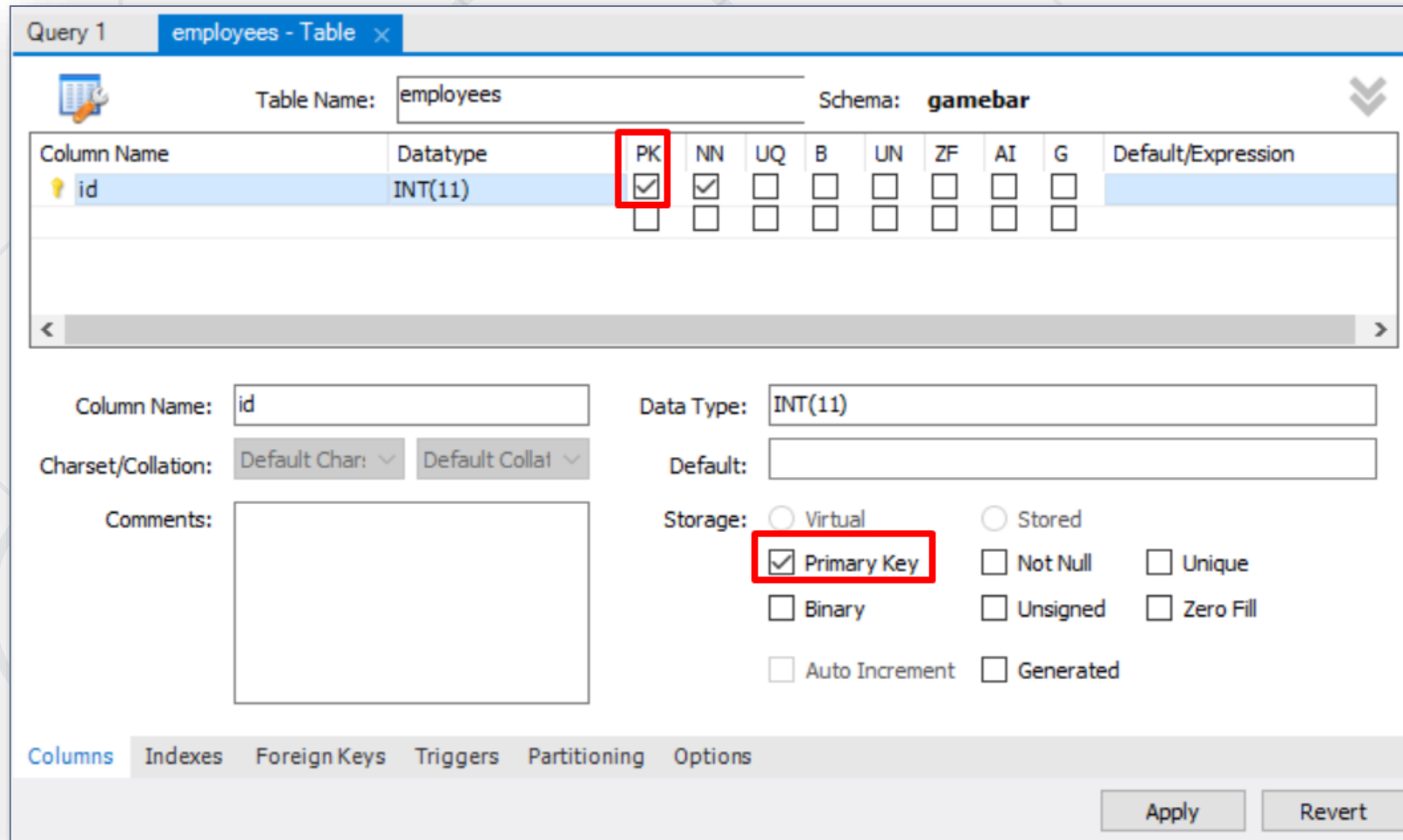
Creating Tables

- Right click on "Tables" **Select Create Table**

Create new table

Set up table name

Add new record
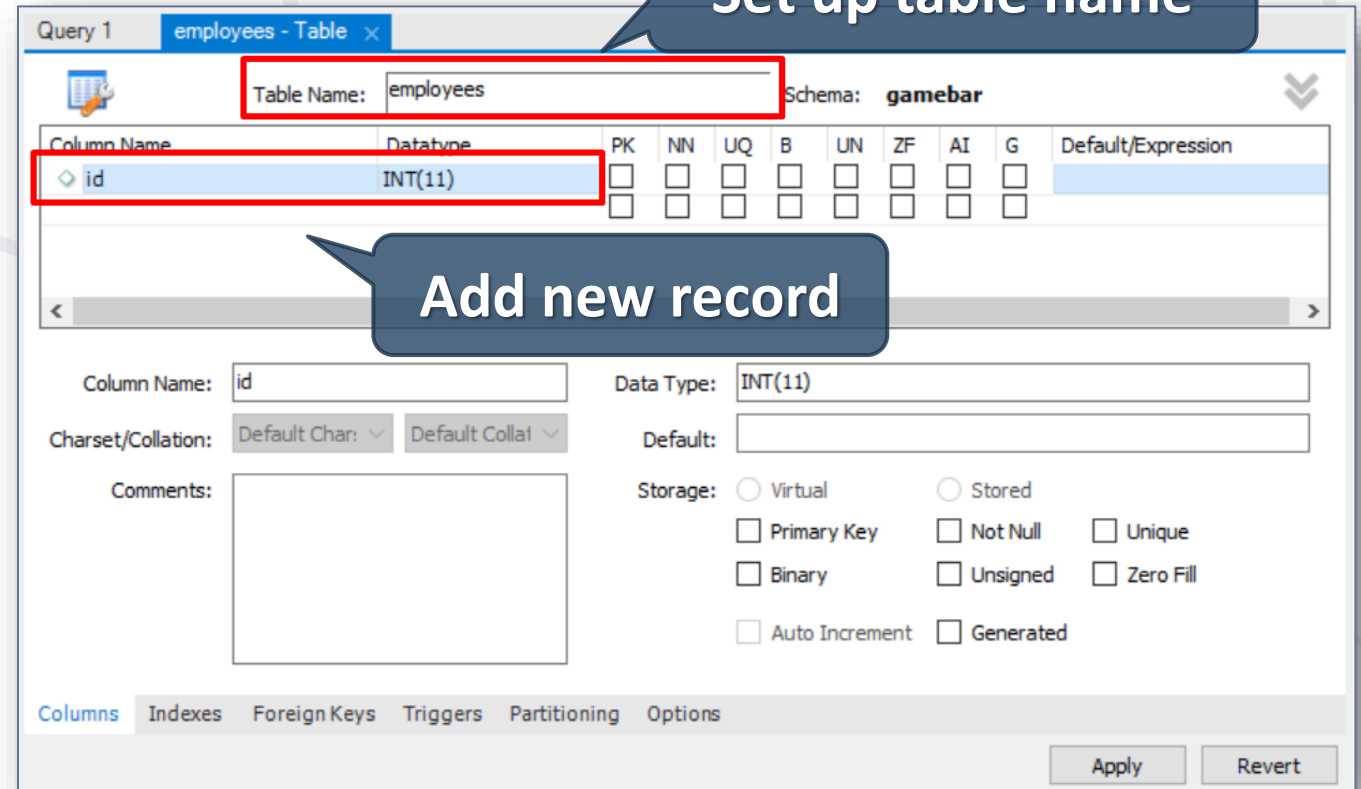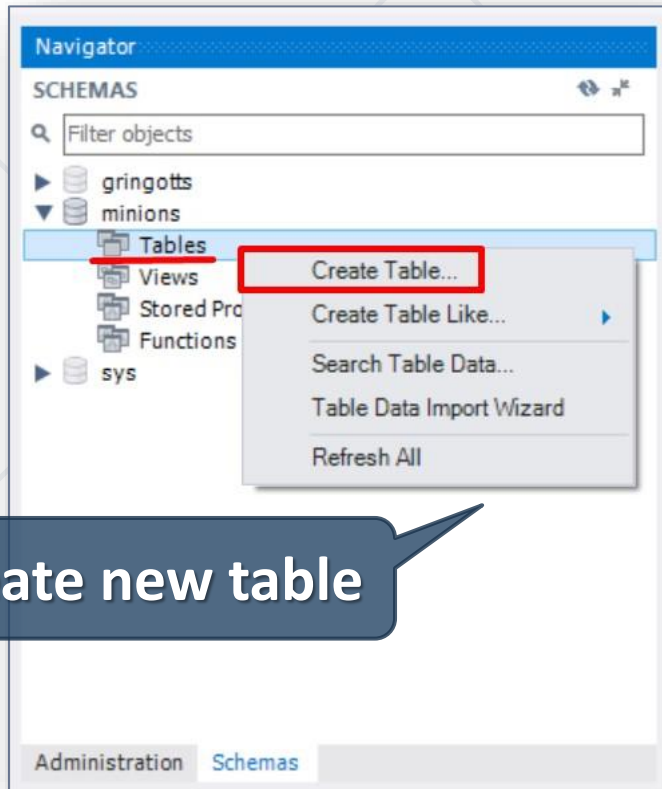
Creating Tables (2)

- A **Primary Key** is used to uniquely identify and index records

Creating Tables (3)

- Auto increment – on the "**Default**" field

Storing and Retrieving Data

- We can **add**, **modify** and **read** records with **GUI Clients**

- To insert or edit a record, click inside the **cell**

```
CREATE TABLE people
(
    id INT NOT NULL,
    email VARCHAR(50) NOT
NULL,
    first_name VARCHAR(50),
    last_name VARCHAR(50)
);
```

Basic SQL Queries

SQL Queries

- We communicate with the database engine using SQL

- Queries provide greater **control** and **flexibility**

- To create a database using SQL:

Database name

```
CREATE DATABASE employees;
```

- SQL keywords are conventionally **capitalized**

Table Creation in SQL

```sql
CREATE TABLE people
(
  id INT NOT NULL,
  email VARCHAR(50) NOT NULL,
  first_name VARCHAR(50),
  last_name VARCHAR(50)
);
```

Table name

Custom properties

Data type

Column name

50

Retrieve Records in SQL

- Get all information from a table

```
SELECT * FROM employees;
```

Table name

- You can limit the columns and number of records

```
SELECT first_name, last_name FROM
employees LIMIT 5;
```

List of columns

Number of records

Table Customization

Custom Column Properties

- Primary Key

```
id INT NOT NULL PRIMARY KEY
```

- Auto-Increment (Identity)

```
id INT AUTO_INCREMENT PRIMARY KEY
```

- Unique constraint – no repeating values in entire table

```
email VARCHAR(50) UNIQUE
```

- Default value – if not specified (otherwise set to NULL)

```
balance DECIMAL(10,2) DEFAULT 0
```

Problems: Create and Insert

- **Create** new Database "gamebar"

- **Create Tables**:

  - **employees** - id, first_name, last_name

  - **categories** - id, name

  - **products** - id, name, category_id

- **Insert** Data:

  - Populate the **employees** table with 3 test values

Altering Tables

Altering Tables Using SQL (1)

- A table can be changed using the keywords **ALTER TABLE**

```
ALTER TABLE employees;
```

> **Table name**

- Add new column

```
ALTER TABLE employees
ADD COLUMN salary DECIMAL;
```

> **Column name**

> **Data type**

Altering Tables Using SQL (2)

- Delete existing column

```
ALTER TABLE people            Column name
DROP COLUMN full_name;
```

- Modify data type of existing column

```
ALTER TABLE people
MODIFY COLUMN email VARCHAR(100);
```

Column name   New data type

- Add primary key to existing column

```
ALTER TABLE people
ADD CONSTRAINT pk_id
PRIMARY KEY (id);
```

Constraint name

**Column name
(more than one for
composite key)**

- Add unique constraint

```
ALTER TABLE people
ADD CONSTRAINT uq_email
UNIQUE (email)
```

Constraint name

Columns name(s)

Altering Tables Using SQL (4)

- Set default value

```
ALTER TABLE people
ALTER COLUMN balance SET DEFAULT 0;
```

Default value

Column name

Problems: Alter Tables

- **Alter** table
  - Add a new column – "middle_name" to the "employees" table
- Adding **Constraints**
  - Make "category_id" **foreign key** linked to "id" in the "categories" table
- **Modifying** Columns
  - Change the property "VARCHAR(50)" to "VARCHAR(100)" to the "middle_name" column in "employees" table

Deleting Data and Structures

Deleting from Database

- Deleting structures is called **dropping**
  - You can drop **keys**, **constraints**, **tables** and entire **databases**
- Deleting all data in a table is called **truncating**
- Both of these actions **cannot be undone** – use with caution!

Dropping and Truncating

- To delete all the entries in a table

```
TRUNCATE TABLE employees;
```
Table name

- To drop a table – delete data and structure

```
DROP TABLE employees;
```
Table name

- To drop entire database

```
DROP DATABASE soft_uni;
```
Database name

Dropping and Truncating (2)

- To remove a constraining rule from a column

  - Primary keys, value constraints and unique fields

```
ALTER TABLE employess        Table name
DROP CONSTRAINT pk_id;
                             Constraint name
```

- To remove **DEFAULT** value (if not specified, revert to **NULL**)

```
ALTER TABLE employess        Table name
ALTER COLUMN clients
                             Columns name
DROP DEFAULT;
```
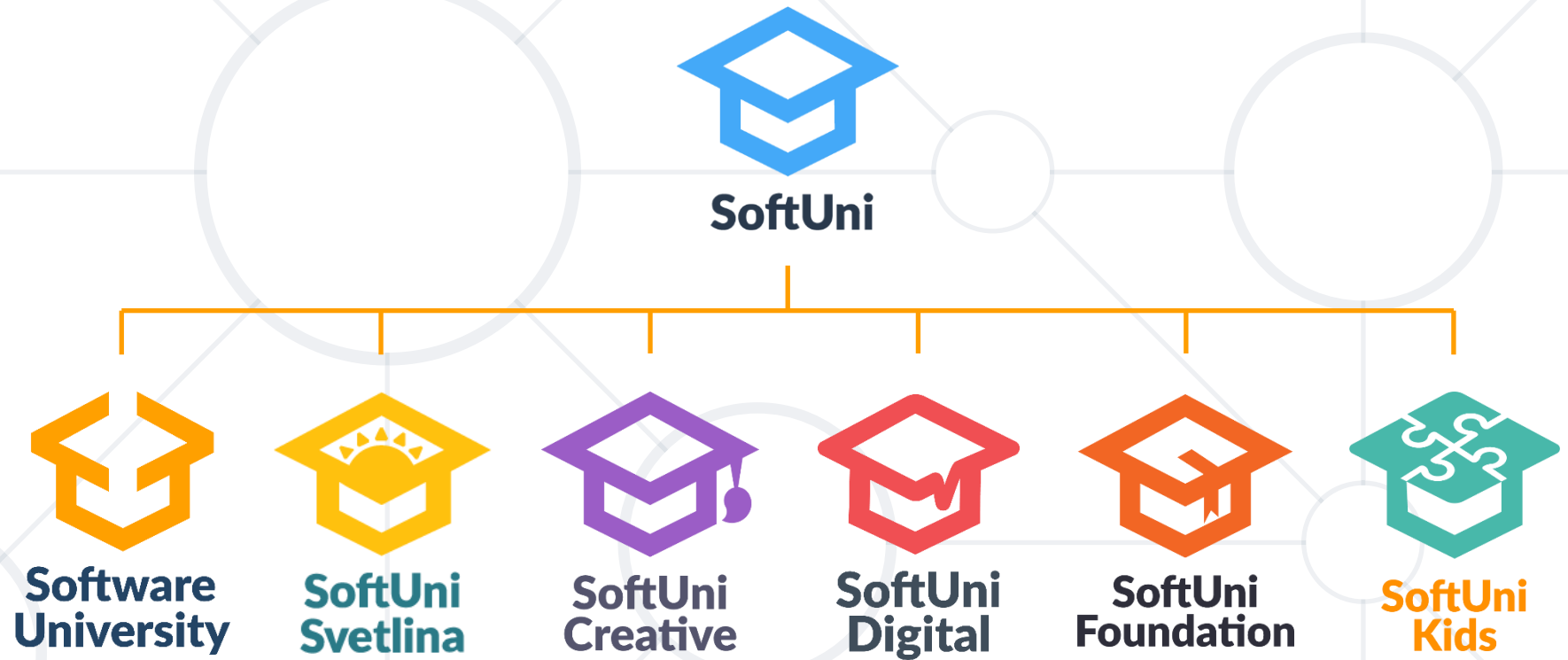
Summary

- We communicate with the DB engine via **SQL**

- MySQL is a **multiplatform** RDBMS using SQL

- Table columns have a **fixed type**

- We can use GUI Clients to **create** and **customize** tables

- SQL provides **greater control**

Questions?

SoftUni Diamond Partners

Educational Partners

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers

  - softuni.bg

- Software University @ Facebook

  - facebook.com/SoftwareUniversity

- SoftUni Global

  - softuni.org

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**

- Unauthorized copy, reproduction or use is illegal

- © SoftUni – https://about.softuni.bg/

- © Software University – https://softuni.bg

- © SoftUni Global – https://softuni.org