Web Programming

CS-406

Lecture # 08

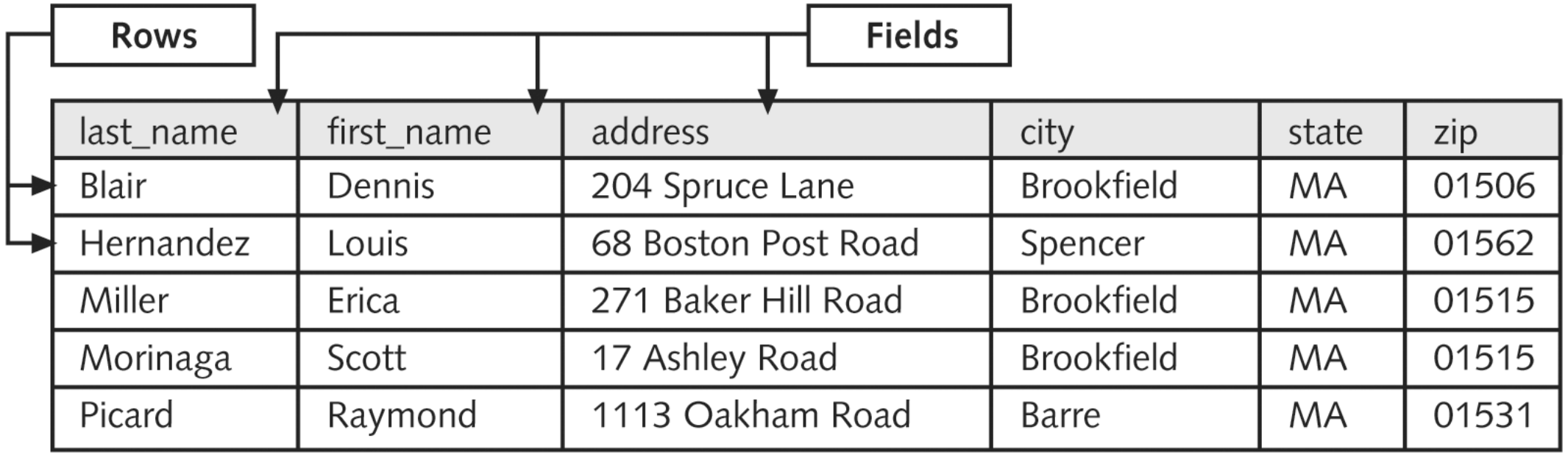
PHP and Databases

Databases

Introduction to Databases

- A **database** is an ordered collection of information from which a computer program can quickly access information
- Each row in a database table is called a record
- A **record** in a database is a single complete set of related information
- Each column in a database table is called a field
- **Fields** are the individual categories of information stored in a record

Introduction to Databases (continued)



last_name	first_name	address	city	state	zip
Blair	Dennis	204 Spruce Lane	Brookfield	MA	01506
Hernandez	Louis	68 Boston Post Road	Spencer	MA	01562
Miller	Erica	271 Baker Hill Road	Brookfield	MA	01515
Morinaga	Scott	17 Ashley Road	Brookfield	MA	01515
Picard	Raymond	1113 Oakham Road	Barre	MA	01531

Introduction (cntd……)

- A **flat-file database** stores information in a single table
- A **relational database stores** information across multiple related tables

Relational Databases

- Relational databases consist of one or more related tables
- A **primary table** is the main table in a relationship that is referenced by another table
- A **related table** (or “child table”) references a primary table in a relational database
- A **primary key** is a field that contains a unique identifier for each record in a primary table

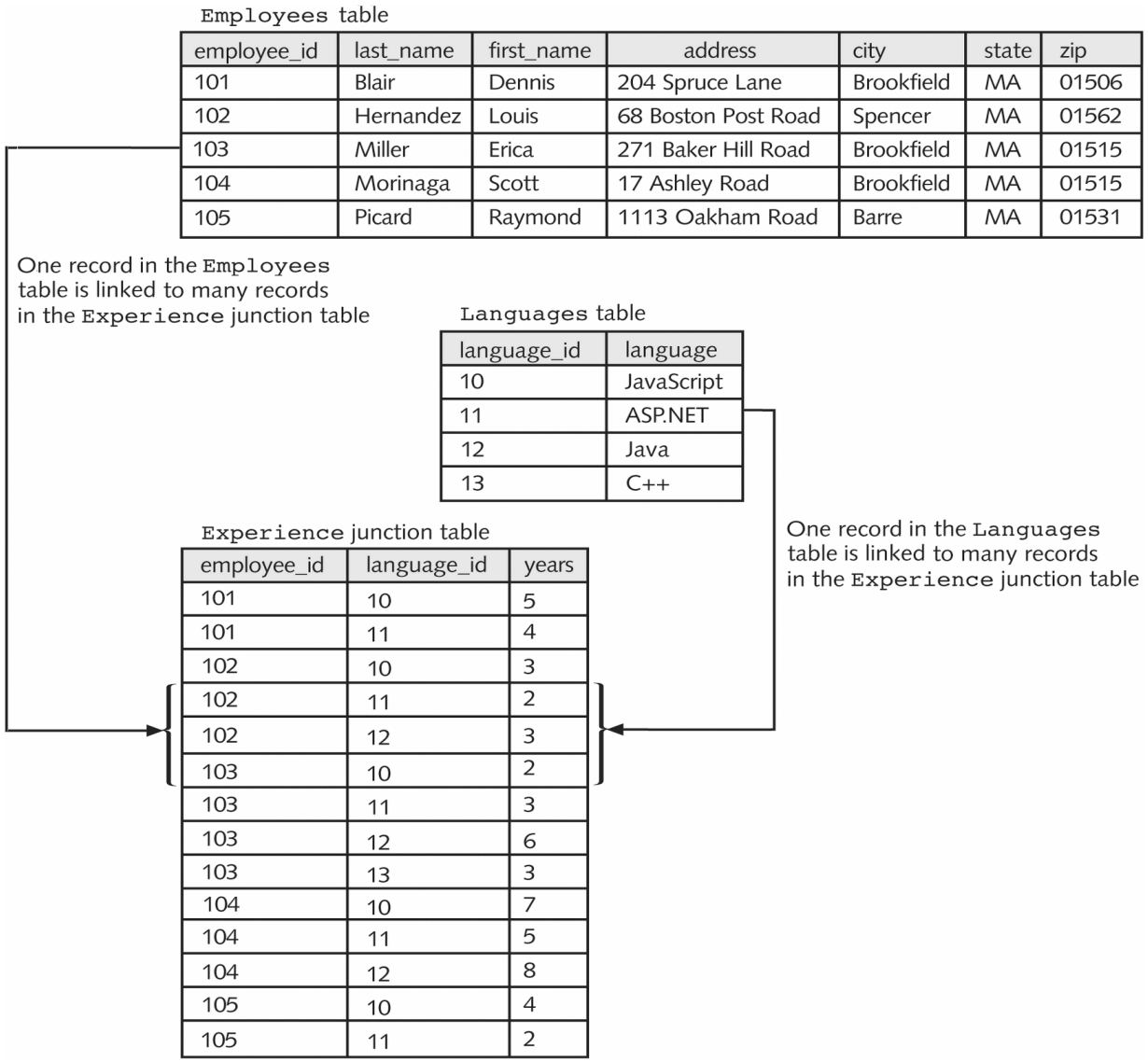
Relational Databases

- A **primary key** is a type of index, which identifies records in a database to make retrievals and sorting faster
- A **foreign key** is a field in a related table that refers to the primary key in a primary table
- Primary and foreign keys link records across multiple tables in a relational database

Database Management System

- A **database management system** (or DBMS) is an application or collection of applications used to access and manage a database
- A **schema** is the structure of a database including its tables, fields, and relationships
- A **flat-file database management system** is a system that stores data in a flat-file format
- A **relational database management system** (or RDBMS) is a system that stores data in a relational format

Working with Database Management Systems (continued)



`Employees` table

employee_id	last_name	first_name	address	city	state	zip
101	Blair	Dennis	204 Spruce Lane	Brookfield	MA	01506
102	Hernandez	Louis	68 Boston Post Road	Spencer	MA	01562
103	Miller	Erica	271 Baker Hill Road	Brookfield	MA	01515
104	Morinaga	Scott	17 Ashley Road	Brookfield	MA	01515
105	Picard	Raymond	1113 Oakham Road	Barre	MA	01531

One record in the `Employees` table is linked to many records in the `Experience` junction table

`Languages` table

language_id	language
10	JavaScript
11	ASP.NET
12	Java
13	C++

One record in the `Languages` table is linked to many records in the `Experience` junction table

`Experience` junction table

employee_id	language_id	years
101	10	5
101	11	4
102	10	3
102	11	2
102	12	3
103	10	2
103	11	3
103	12	6
103	13	3
104	10	7
104	11	5
104	12	8
105	10	4
105	11	2

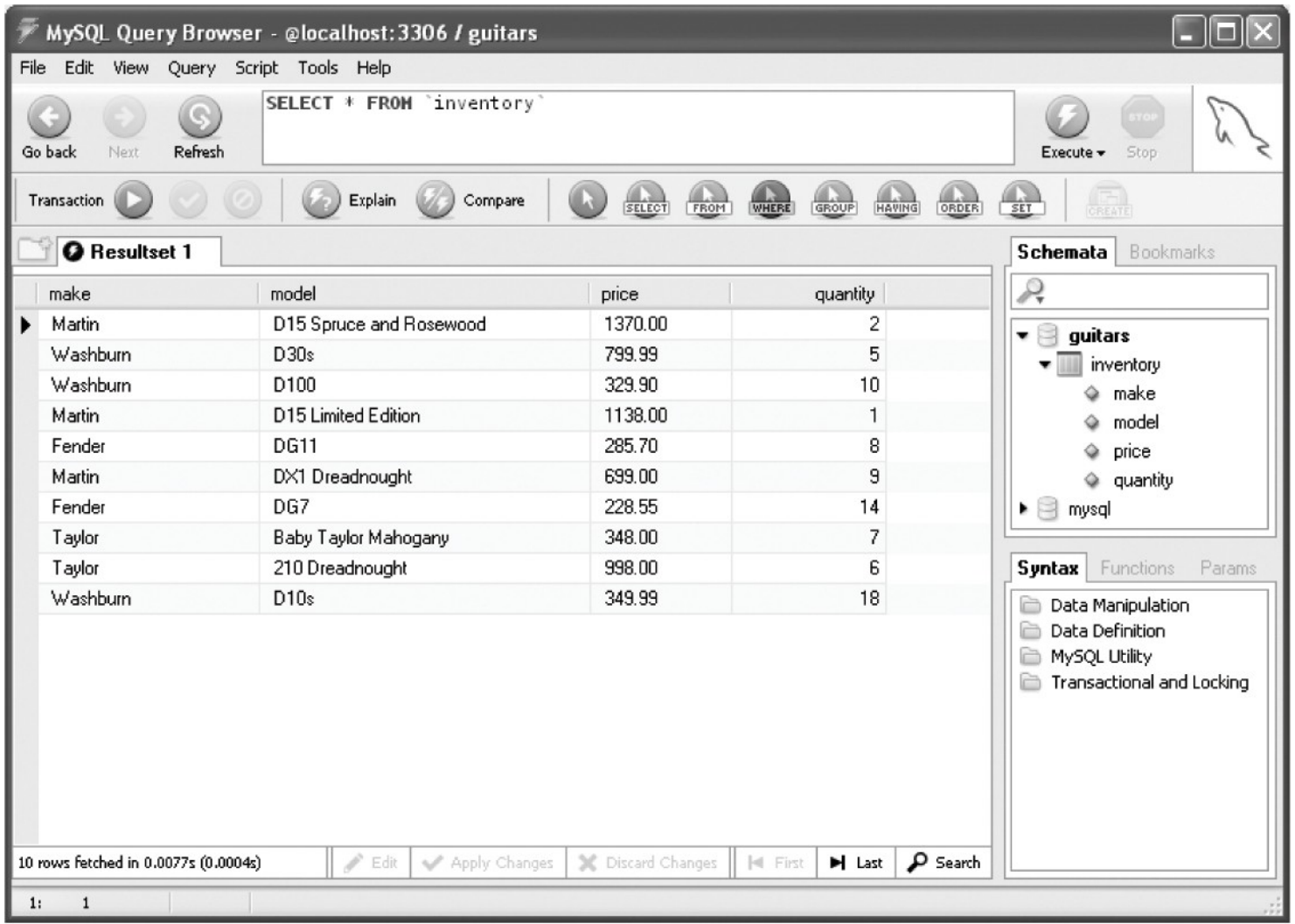
Database Management System

- Important aspects of database management systems:
  - The structuring and preservation of the database file
  - Ensuring that data is stored correctly in a database's tables, regardless of the database format
  - Querying capability

Query

- A **query** is a structured set of instructions and criteria for retrieving, adding, modifying, and deleting database information
- **Structured query language** (or SQL) is a standard data manipulation language used among many database management systems

Working with Database Management Systems (continued)



Querying Databases with Structured Query Language

Keyword	Description
`DELETE`	Deletes a row from a table
`FROM`	Specifies the tables from which to retrieve or delete records
`INSERT`	Inserts a new row into a table
`INTO`	Determines the table into which records should be inserted
`ORDER BY`	Sorts the records returned from a table
`SELECT`	Returns information from a table
`UPDATE`	Saves changes to fields in a record
`WHERE`	Specifies the conditions that must be met for records to be returned from a query

Database - Create

—

- Use the CREATE DATABASE statement to create a new database:

```
mysql> CREATE DATABASE Student;[ENTER° ]
```

- To use a new database, select it by executing the USE DATABASE statement

Selecting a Database

- Use the DATABASE( ) function to return the name of the currently active database

```
mysql> SELECT DATABASE();[ENTER° ]
```

- View the available databases using the SHOW DATABASES statement

```
mysql> SHOW databases;[ENTER° ]
```

- Use the DROP DATABASE statement to remove all tables and delete a database

```
mysql> DROP DATABASE database;
```

Defining Database Tables

- Data types that are assigned to fields determine how much storage space the computer allocates for the data in the database
- Choose the smallest data type possible for each field

Defining Database Tables (continued)

Type	Storage	Range	Special information
BOOL	1 byte	–128 to 127	0 is considered FALSE
TINYINT	1 byte	–128 to 127	
SMALLINT	2 bytes	–32,768 to 32,767	
MEDIUMINT	3 bytes	–8,388,608 to 8,388,607	
INT or INTEGER	4 bytes	–2,147,483,648 to 2,147,483,647	
BIGINT	8 bytes	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	
FLOAT	4 bytes	–3.402823466E+38 to –1.175494351E-38, 0, and 1.175494351E-38 to 3.402823466E+38	0 to 24 bits of precision
DOUBLE or DOUBLE PRECISION	8 bytes	–1.7976931348623157E+308 to –2.2250738585072014E-308, 0, and 2.2250738585072014E-308 to 1.7976931348623157E+308	25-53 bits of precision
DATE	3 bytes	'0000-00-00', '1000-01-01' to '9999-12-31'	
TIME	3 bytes	'–838:59:59' to '838:59:59'	
CHAR(*m*)	Number of bytes specified by m	Fixed-length string between 0 to 255 characters	
VARCHAR(*m*)	Varies up to the number of bytes specified by m	Variable-length string with a maximum length between 0 to 65,535 characters	Maximum length is 255 in older versions
ENUM	Varies	One of a set of predefined strings	
SET	Varies	Zero or more of a set of predefined strings, separated by commas	

Table 7-3 Common MySQL data types

Creating Tables

- Use the CREATE TABLE statement to create a new table and define the column names and data types for each column

```
mysql> CREATE TABLE vehicles
 (license VARCHAR(10), make VARCHAR(25),
  model VARCHAR(50), miles FLOAT,
 assigned_to VARCHAR(40));[ENTER°]
```

Viewing Table Structure

- Use the DESCRIBE *table_name* statement to view the structure of the table

```
mysql> DESCRIBE vehicles;[ENTER↵]
+------------+------------+------+-----+---------+-------+
| Field      | Type       | Null | Key | Default | Extra |
+------------+------------+------+-----+---------+-------+
| license    | varchar(10) | YES  |     | NULL    |       |
| make       | varchar(25) | YES  |     | NULL    |       |
| model      | varchar(50) | YES  |     | NULL    |       |
| miles      | float      | YES  |     | NULL    |       |
| assigned_to | varchar(40) | YES  |     | NULL    |       |
+------------+------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Deleting Tables

- Execute the DROP TABLE statement to remove all data and the table definition from a database

  ```
  DROP TABLE table;
  ```

- In MySQL Monitor, enter the following at the mysql> prompt:

  ```
  mysql> DROP TABLE company_cars;[ENTER⏎]
  ```

- You must be logged in as the `root` user or have `DROP` privileges to delete a table.

Adding Records

- Use the `INSERT` statement to add individual records to a table
- The syntax for the `INSERT` statement is:

```
INSERT INTO table_name (column1, column2, …)
  VALUES(value1, value2, ...);
```

- The values entered in the VALUES list must be in the same order in which you defined the table fields
- Specify `NULL` in any fields for which you do not have a value

Adding Records

- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> INSERT INTO company_cars(license,
   model_year, make, model, mileage)
   VALUES('CK-2987', 2009, 'Toyota',
   'Corolla', 3508.4);[ENTER°]
```

Retrieving Records

- Use the SELECT statement to retrieve records from a table:

  ```
  SELECT criteria FROM table_name;
  ```

- Use the asterisk (*) wildcard with the SELECT statement to retrieve all fields from a table
- To return multiple fields, separate field names with a comma

Retrieving Records (continued)

- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> SELECT model, mileage FROM
 company_cars;[ENTER↵]
```

Sorting Query Results

- Use the `ORDER BY` keyword with the `SELECT` statement to perform an alphanumeric sort of the results returned from a query
- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> SELECT make, model FROM inventory
  ORDER BY make, model;[ENTER]
```

Sorting Query Results

- To perform a reverse sort, add the DESC keyword after the name of the field by which
you want to perform the sort
- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> SELECT make, model FROM
   company_cars ORDER BY make DESC,
   model;[ENTER°]
```

Filtering Query Results

- The **criteria** portion of the SELECT statement determines which fields to retrieve from a table
- You can also specify which records to return by using the WHERE keyword
- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> SELECT * FROM inventory WHERE
  make='Martin‘;[ENTER°]
```

Filtering Query Results

- Use the keywords AND and OR to specify more detailed conditions about the records you want to return
- In MySQL Monitor, enter the following code using the AND keyword at the `mysql>` prompt:

```
mysql> SELECT * FROM company_cars
    WHERE model_year=2007 AND
   mileage<60000;[ENTER° ]
```

Filtering Query Results

- In MySQL Monitor, enter the following code using the OR keyword at the mysql> prompt:

```
mysql> SELECT * FROM company_cars
  WHERE make='Toyota’OR
  make='Honda‘ORDER BY mileage;[ENTER°]
```

Updating Records

- To update records in a table, use the UPDATE statement
- The syntax for the UPDATE statement is:

```
UPDATE table_name
SET column_name=value
WHERE condition;
```

  - The UPDATE keyword specifies the name of the table to update
  - The SET keyword specifies the value to assign to the fields in the records that match the condition in the WHERE keyword

Updating Records (continued)

- In MySQL Monitor, enter the following code using the OR keyword at the `mysql>` prompt:

```
mysql> UPDATE company_cars SET
  mileage=368.2
   WHERE make='Ford’ AND model='Fusion';
  [ENTER°]
```

Deleting Records

- Use the DELETE statement to delete records in a table
- The syntax for the DELETE statement is:

```
DELETE FROM table_name
   WHERE condition;
```

- The DELETE statement deletes all records that match the condition
- To delete all the records in a table, leave off the WHERE keyword

Deleting Records

- In MySQL Monitor, enter the following code at the `mysql>` prompt:

```
mysql> DELETE FROM company_cars WHERE
  model_year=2006 AND make='Honda'
   AND model='Accord';[ENTER° ]
```

- To delete all records from a table, omit the `WHERE` clause

Alter

- ALTER TABLE Customers
  ADD Email varchar(255);

- ALTER TABLE *table_name*
  DROP COLUMN *column_name*;

- ALTER TABLE *table_name*
  MODIFY COLUMN *column_name datatype*;

Constraints

- Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

- CREATE TABLE *table_name* (
  column1 datatype constraint,
  column2 datatype constraint,
  column3 datatype constraint,
  ....
  );

Constraints

- The following constraints are commonly used in SQL:
- **NOT NULL** - Ensures that a column cannot have a NULL value
- **UNIQUE** - Ensures that all values in a column are different
- **PRIMARY KEY** - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
- **FOREIGN KEY** - Uniquely identifies a row/record in another table
- **CHECK** - Ensures that all values in a column satisfies a specific condition
- **DEFAULT** - Sets a default value for a column when no value is specified
- **INDEX** - Used to create and retrieve data from the database very quickly

References

- https://www.w3schools.com/php/php_mysql_intro.asp
- https://www.php.net/manual/en/migration5.databases.php
- https://www.siteground.com/tutorials/php-mysql/
- https://www.cloudways.com/blog/connect-mysql-with-php/