W2BUSINESS
QA Academy
Wroclaw - Spring 2018
Chapter “Databases (DB)”

Lectors

Svitlana Samko

https://www.linkedin.com/in/svitlana-samko-b87532114/

Senior Developer in Test
over 10 years web development practice
over 10 delivered software projects for middle and large business

My most beneficial skill: *I like to learn business from the inside. Only so one can be sure that we build right product in the right way at any stage of development process.*

Andrii Stepura

https://www.linkedin.com/in/andriistepura/

Senior Quality Assurance Automation Engineer
over 14 years web development practice
over 300 delivered web projects as PO / Dev / Analyst / QA

My most beneficial skill: *Imagination to think like a stakeholder. Every piece of software starts from an idea. The first written code lines are just a half of the delivery of that idea.*

Definition of done

1. **Introduction to IT:**
   - Introduction to IT in basic terms
   - Software theory
     - SW goals, SW types, benefits
     - Software development life cycle, models
   - Fundamentals of Testing
2. Software Development basics:
   - Data
     - Formats - Text, graphic, binary
   - Languages
     - Formal XML, HTML, CSS, JSON
     - Programming Javascript, PHP
   - Test Levels and Types
   - Static Techniques
   - Review
3. **Databases (DB):**
   - Structured Query Language (SQL)
   - Relationship (MySQL DB examples)

Databases

What do you know about...

Databases?

Databases

A database is an organized collection of data. A relational database, more restrictively, is a collection of schemas, tables, queries, reports, views, and other elements. Database designers typically organize the data to model aspects of reality in a way that supports processes requiring information, such as (for example) modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

Evolution of databases

Databases have evolved since their inception in the 1960s, beginning with hierarchical and network databases, through the 1980s with object-oriented databases, and today with SQL and NoSQL databases and cloud databases.

In one view, databases can be classified according to content type: bibliographic, full text, numeric and images. In computing, databases are sometimes classified according to their organizational approach. There are many different kinds of databases, ranging from the most prevalent approach, the relational database, to a distributed database, cloud database or NoSQL database.

Codd's 12 rules

Codd's twelve rules are a set of thirteen rules (numbered zero to twelve) proposed by Edgar F. Codd, a pioneer of the relational model for databases... Codd produced these rules as part of a personal campaign to prevent the vision of the original relational database from being diluted, as database vendors scrambled in the early 1980s to repackage existing products with a relational veneer. Rule 12 was particularly designed to counter such a positioning.

Databases

SQL (Structured Query Language):

is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). In comparison to older read/write APIs like ISAM (Indexed Sequential Access Method) or VSAM (Virtual Storage Access Method), SQL offers two main advantages: first, it introduced the concept of accessing many records with one single command; and second, it eliminates the need to specify how to reach a record, e.g. with or without an index.

© https://en.wikipedia.org/wiki/SQL

Databases

SQL Data Types

Data type	Description
CHAR(size)	Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to 255 characters
VARCHAR(size)	Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to 255 characters. **Note:** If you put a greater value than 255 it will be converted to a TEXT type
TINYTEXT	Holds a string with a maximum length of 255 characters
TEXT	Holds a string with a maximum length of 65,535 characters
BLOB	For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data
MEDIUMTEXT	Holds a string with a maximum length of 16,777,215 characters
MEDIUMBLOB	For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data
LONGTEXT	Holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
ENUM(x,y,z,etc.)	Let you enter a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. **Note:** The values are sorted in the order you enter them. You enter the possible values in this format: ENUM('X','Y','Z')
SET	Similar to ENUM except that SET may contain up to 64 list items and can store more than one choice

Databases

SQL Number data types:

Data type	Description
TINYINT(size)	-128 to 127 normal. 0 to 255 UNSIGNED*. The maximum number of digits may be specified in parenthesis
SMALLINT(size)	-32768 to 32767 normal. 0 to 65535 UNSIGNED*. The maximum number of digits may be specified in parenthesis
MEDIUMINT(size)	-8388608 to 8388607 normal. 0 to 16777215 UNSIGNED*. The maximum number of digits may be specified in parenthesis
INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
BIGINT(size)	-9223372036854775808 to 9223372036854775807 normal. 0 to 18446744073709551615 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE stored as a string , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

Databases

Date data types:

Data type	Description
DATE()	A date. Format: YYYY-MM-DD **Note:** The supported range is from '1000-01-01' to '9999-12-31'
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP()	*A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC
TIME()	A time. Format: HH:MI:SS **Note:** The supported range is from '-838:59:59' to '838:59:59'
YEAR()	A year in two-digit or four-digit format. **Note:** Values allowed in four-digit format: 1901 to 2155. Values allowed in two-digit format: 70 to 69, representing years from 1970 to 2069

Databases

SQL (Structured Query Language):

Source	Common name	Full name
ANSI/ISO Standard	SQL/PSM	SQL/Persistent Stored Modules
Interbase / Firebird	PSQL	Procedural SQL
IBM DB2	SQL PL	SQL Procedural Language (implements SQL/PSM)
IBM Informix	SPL	Stored Procedural Language
IBM Netezza	NZPLSQL[20]	(based on Postgres PL/pgSQL)
Microsoft / Sybase	T-SQL	Transact-SQL
Mimer SQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MySQL	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
MonetDB	SQL/PSM	SQL/Persistent Stored Module (implements SQL/PSM)
NuoDB	SSP	Starkey Stored Procedures
Oracle	PL/SQL	Procedural Language/SQL (based on Ada)
PostgreSQL	PL/pgSQL	Procedural Language/PostgreSQL Structured Query Language (implements SQL/PSM)
Sybase	Watcom-SQL	SQL Anywhere Watcom-SQL Dialect
Teradata	SPL	Stored Procedural Language
SAP	SAP HANA	SQL Script

Software Development basics

CRUD

create, read, update and delete (as an acronym CRUD) are the four basic functions of persistent storage. Alternate words are sometimes used when defining the four basic functions of CRUD, such as retrieve instead of read, modify instead of update, or destroy instead of delete. CRUD is also sometimes used to describe user interface conventions that facilitate viewing, searching, and changing information; often using computer-based forms and reports. The term was likely first popularized by James Martin in his 1983 book Managing the Data-base Environment. The acronym may be extended to CRUDL to cover listing of large data sets which bring additional complexity such as pagination when the data sets are too large to hold easily in memory.

Databases

CRUD in SQL

Operation	SQL	HTTP	DDS
Create	INSERT	PUT / POST	write
Read (Retrieve)	SELECT	GET	read / take
Update (Modify)	UPDATE	PUT / POST / PATCH	write
Delete (Destroy)	DELETE	DELETE	dispose

Databases

SQL CREATE DATABASE Statement

The CREATE DATABASE statement is used to create a new SQL database.

CREATE DATABASE databasename;

```
CREATE DATABASE databasename;
```

SQL CREATE TABLE

The CREATE TABLE statement is used to create a new table in a database.

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

Example

```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

Databases

SQL INSERT INTO Statement

INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);

INSERT INTO table_name
VALUES (value1, value2, value3, ...);

Example

```
INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

Try it Yourself »

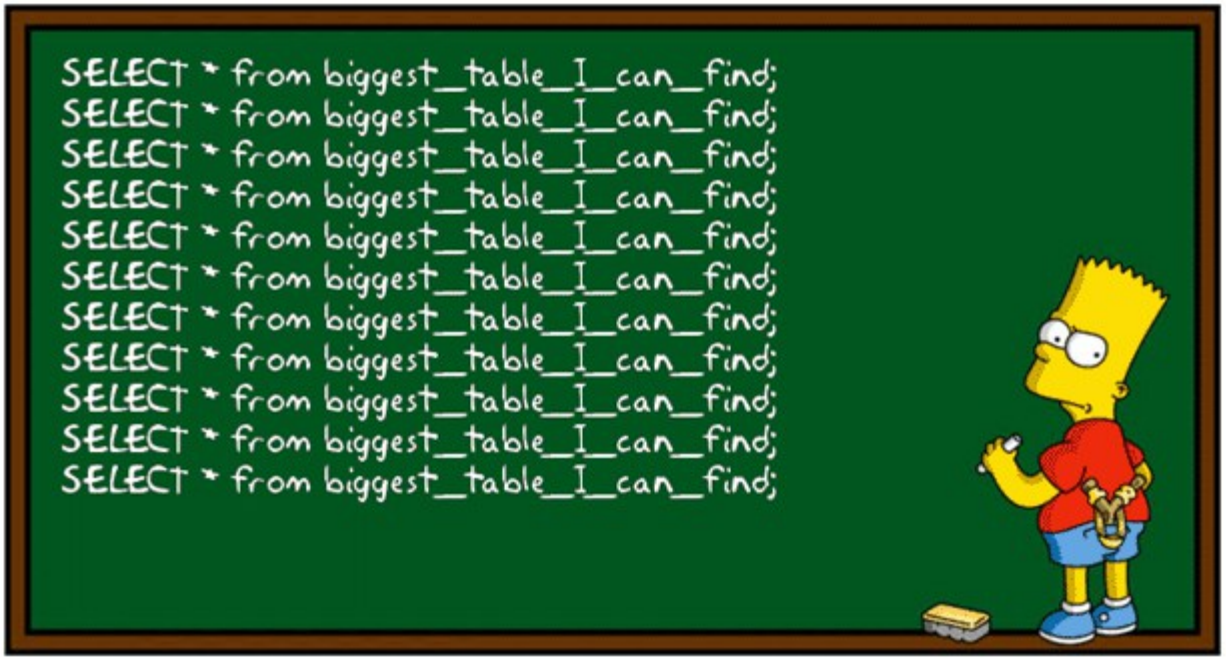
The selection from the "Customers" table will now look like this:

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

SQL SELECT Statement

SELECT column1, column2, ...
FROM table_name;

*SELECT * FROM table_name;*



© https://www.w3schools.com/sql/sql_select.asp & https://www.toadworld.com/products/toad-for-sql-server/b/weblog/archive/2010/05/17/select-the-good-the-bad-amp-more-bad

SQL ORDER BY Keyword

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC;
```

```
SELECT * FROM Customers
ORDER BY Country DESC;
```

```
SELECT * FROM Customers
ORDER BY Country ASC, CustomerName DESC;
```

SELECT *additional commands*

The SELECT **DISTINCT** statement is used to return only different values.

SELECT DISTINCT column1, column2, … FROM table_name;

```
SELECT DISTINCT Person.PName
FROM Person;
```

Id	Name	Address	Hobby
1123	Anita	Damauli	stamps
1123	Anita	Damauli	coins
5556	Binod	Kathmandu	hiking
9876	Barsha	Kathmandu	stamps

PName
Anita
Barsha
Binod

Databases

SELECT *additional commands*

The SELECT **WHERE** clause is used to filter records.

*SELECT column1, column2, … FROM table_name WHERE **{condition}**;*

WHERE Country='Mexico'; WHERE CustomerID=1; WHERE …

Operator	Description
=	Equal
<>	Not equal. **Note:** In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

Databases WHERE additional commands

AND, OR, NOT

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;
```

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;
```

```
table_name
WHERE
WHERE NOT condition;
```

The WHERE clause can be combined with AND, OR and NOT operators.
The AND and OR operators are used to filter records based on more than one condition:

- The AND operator displays a record if all the conditions separated by AND is TRUE.
- The OR operator displays a record if any of the conditions separated by OR is TRUE.
- The NOT operator displays a record if the condition(s) is NOT TRUE.

Databases

WHERE

additional commands

BETWEEN (1,10)
IN (1,2,3,4,5,6,7,8,9,10)
LIKE (_ame%)

!

//[TODO]

© https://www.w3schools.com/sql/sql_like.asp

Databases

LIKE

additional commands

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that starts with "a"
WHERE CustomerName LIKE '%a'	Finds any values that ends with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%_%'	Finds any values that starts with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that starts with "a" and ends with "o"

!

//[TODO]

© https://www.w3schools.com/sql/sql_like.asp

SQL ALTER TABLE Statement

The ALTER TABLE statement is used to add, delete or modify columns in an existing table and also used to add and drop various constraints on an existing table.

```
ALTER TABLE table_name
ADD column_name datatype;
```

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

SQL UPDATE Statement

UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

Databases

SQL DELETE Statement

DELETE FROM table_name
WHERE condition;

```
DELETE FROM Customers
WHERE CustomerName='Alfreds Futterkiste';
```

It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

```
DELETE FROM table_name;
```

or:

```
DELETE * FROM table_name;
```

The DROP TABLE statement is used to drop an existing table in a database.

```
DROP TABLE table_name;
```

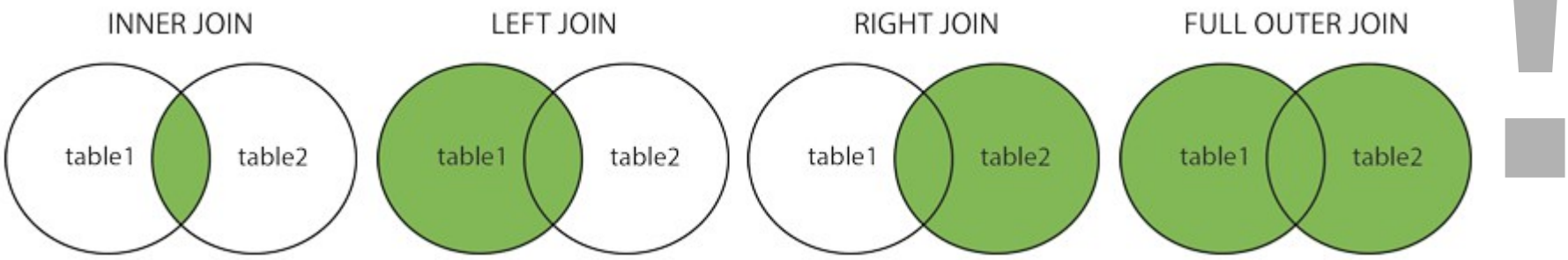
The DROP DATABASE statement is used to drop an existing SQL database.

```
DROP DATABASE databasename;
```

SQL Joins

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

(INNER) JOIN: Returns records that have matching values in both tables
LEFT (OUTER) JOIN: Return all records from the left table, and the matched records from the right table
RIGHT (OUTER) JOIN: Return all records from the right table, and the matched records from the left table
FULL (OUTER) JOIN: Return all records when there is a match in either left or right table

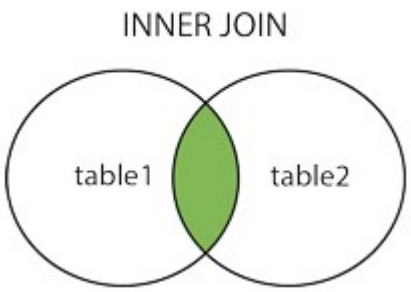


© https://www.w3schools.com/sql/sql_insert.asp

SQL INNER JOIN Keyword

The INNER JOIN keyword selects records that have matching values in both tables.

```
SELECT column_name(s)
FROM table1
INNER JOIN table2 ON table1.column_name =
table2.column_name;
```

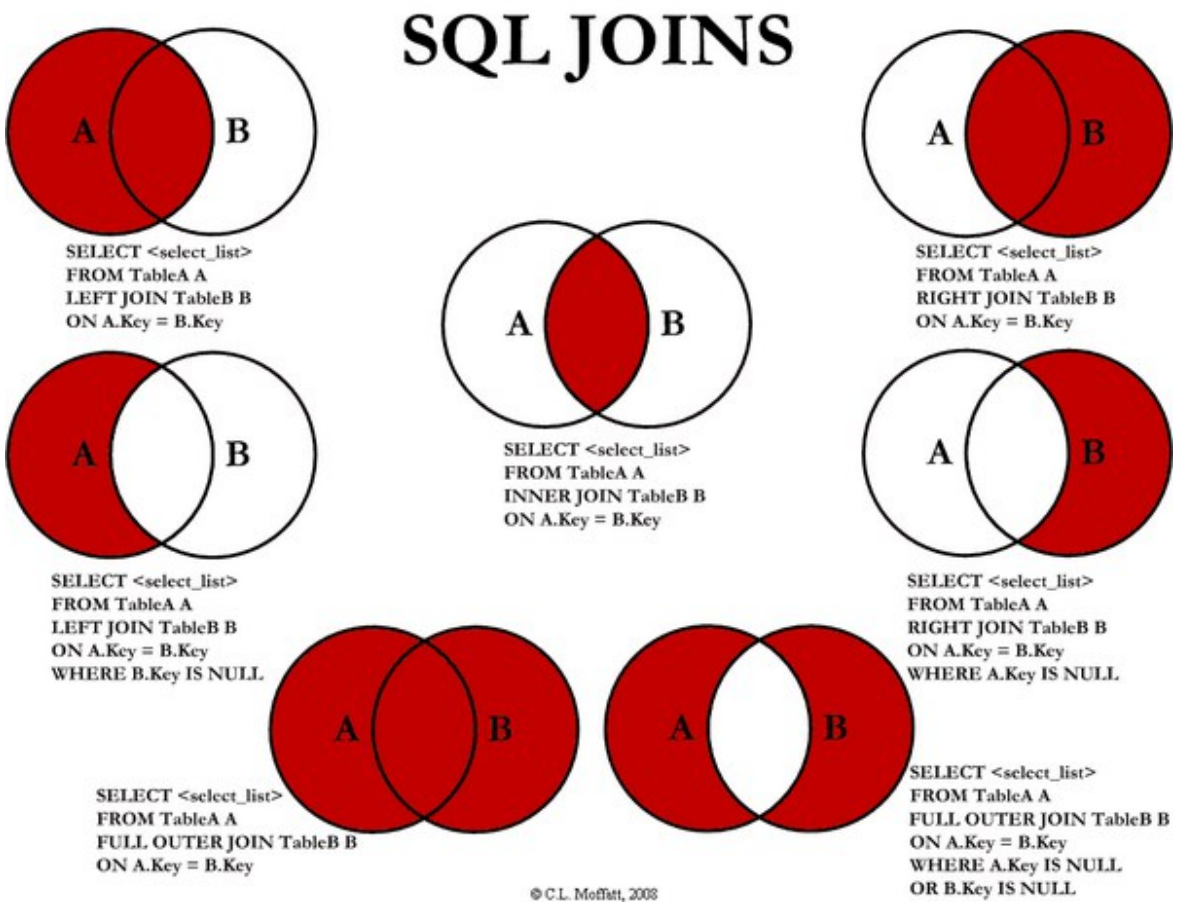


Example

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

Databases

SQL JOINS



© https://stackoverflow.com/questions/4560471/how-to-exclude-rows-that-dont-join-with-another-table/4560613

Self-training home tasks:

//[TODO]

0. Get updates from repo https://github.com/AndriiStepura/W2BUSINESS_QA_Academy with command: git bash pull
1. In Asana project assign any tasks from "To Do" with title "Lecture #3 - Homework task theory #WQAA-..." to yourself and set as "In Progress".

2. Read ISTQB syllabus 5 and 6 chapters (45-66 pages)
3. Log in phpmyadmin administration tool for MySQL DB at server with credentials:
panel url: **http://144.76.160.118/phpmyadmin/**
login: **"qa2018_" + {#from_task}**
pass: **qa2018123?%^&**
3.1.Import file "3.Questions.sql" at DB
4. Create table for answers, as example with "`CREATE TABLE qa2018_0.Answers ( ID INT NULL , Your_Answer VARCHAR(1) NOT NULL );`**"**
4.1. SELECT Question, Answers 1 to 10 as ex.: FROM `Questions` WHERE ID=1
4.2. Set answers with "`INSERT INTO Answers (ID, Your_Answer) VALUES ('1', 'A')`**" OR "**`INSERT INTO Answers VALUES (1,"A")`**"**
4.3. OR UPDATE answers with "UPDATE Answers SET Your_Answer="A" WHERE ID=1"
5. Export answers table, as result – file "Answers.sql" attach to task and set task as "RfR"
6. Assign any tasks from "RfR" with title "Lecture #3 - Homework task theory #WQAA-..." to yourself and set as "In Review".
6.1. Import answers from task with rename table, as ex. "rfr_Answers.sql" at file and change table in file, as ex. "RfR_Answers"
6.2. Compare answers with yours by SQL join, as ex: "SELECT Answers.Your_Answer,RfR_Answers.Your_Answer FROM Answers INNER JOIN RfR_Answers ON Answers.ID = RfR_Answers.ID;" if ok, set task as DONE, else to TODO

W3Schools SQL Quiz

Additional:
7. In Asana project assign any tasks from "To Do" with title "Lecture #3 - Homework task practice #WQAA-..." to yourself and set as "In Test". Pass the quiz https://www.w3schools.com/quiztest/quiztest.asp?qtest=SQL with result >90% then add screenshot to task and set as "Done".

Gratitude:

Thanks for review:

http://w2business.pl/
W2BUSINESS
QA academy

Thanks for tech background:

FUNDACJA UKRAINA

https://www.linkedin.com/company/10948809/

https://www.facebook.com/fundacja.ukraina/