https://datascience-project.eu/

Training Fiche

Title	Software
Keywords (meta tag)	SQL Github
Language	English
Objectives / Goals / Learning outcomes	1. **Understanding when to use SQL** 2. **Important fields of application of SQL** 3. **The basic SQL commands** 4. **How to develop code together in a team** 5. **Basic functionalities of GitHub**

Training course:	
Data Science Literacy	
Data Visualisation and Visual Analytics Module	
Software	X
Introduction to Data science for Human & Social Sciences	
Data Science for good	
Data Journalism and Storytelling	

• **Description**	• This course will briefly introduce the most important programming languages and tools that Data Scientists use on a daily basis. • The context and purpose in which they are typically used will be outlined and the most valuable commands for beginners will be presented: o **SQL** has become a cornerstone of modern data management. In this course, we will explore different ways SQL can be used to retrieve data from databases. o We will discuss what **GitHub** is, what features it offers, and how software developers can benefit from it. • At the end of the course, students will know the field of activity and the commands that are most common.
Contents arranged in 3 levels	**2. Introduction to SQL** **2.1 Background Information** **2.1.1 SQL: the global standard language for database management and analysis** Structured Query Language (SQL) is a widely used database language that is designed to manage and manipulate relational databases. It is a powerful tool that allows users to analyse, manipulate, and process data in various ways, making it a popular choice in many industries and application areas, including finance, e-commerce, health care, and government.

With the support of the Erasmus+ programme of the European Union. This document and its contents reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

One of the key features of SQL is its flexibility. It offers a standard method for maintaining and processing large amounts of data, making it a first choice for companies that need to manage large amounts of information. Its flexibility also allows for customized queries and reports to be generated, providing users with valuable insights into the data they are working with.

In addition to its flexibility, SQL is also highly effective in its ability to process and manipulate data quickly and efficiently. With the use of complex SQL statements and indexing techniques, SQL can quickly locate and retrieve data, which is especially important when working with large databases.

Another advantage of SQL is its easy-to-use nature. For most users, it is relatively easy to learn and use, making it a popular choice for businesses and organizations of all sizes. There are also numerous sources available to learn the language, from online tutorials and courses to textbooks and manuals.

Despite its many advantages, SQL does have some limitations. For example, it is not always the best choice for working with unstructured data, such as images, videos, or audio files. Additionally, it may not be the most effective choice for certain types of analyses, such as those that require advanced statistical techniques.

2.1.2 Purpose of SQL

One of the main uses of SQL is to retrieve data from a database. This can include selecting specific columns of data, filtering data based on specific criteria, or combining data from multiple tables. For example, let's say you have a database of customers and their orders. With SQL, you can easily retrieve a list of all orders for a particular customer or all orders for a particular time period.

Another use of SQL is to add new data to a database, edit existing data, or delete data that is no longer needed. This can be especially useful when you need to update large amounts of data at once. For example, if you need to update the shipping address for all customers who live in a particular zip code, you can make this change quickly and easily with SQL.

In addition to managing data, SQL can also be used to create and manage entire tables in databases. This includes creating

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

new tables, modifying existing tables, and deleting tables that are no longer needed. For example, if you want to create a new table to record your company's sales data, you can use SQL to define the table's structure and specify the data types for each column.

Finally, SQL is particularly well suited for processing large amounts of data. This is because it is designed for high efficiency and can handle complex queries and operations effortlessly. This makes it the best choice for companies and organizations that need to manage and analyse large amounts of data on a regular basis.

2.1.3 There are only 4 essential things you need to make valuable use of SQL

If you want to start using SQL, you may be wondering what tools and knowledge you need to get started. Fortunately, you only need 4 essential things.

First, you need a database management system (DBMS). A DBMS is a software system that allows you to create, manage, and manipulate databases. There are many different DBMSs, but among the most popular are MySQL, Oracle, PostgreSQL and Microsoft SQL Server. These systems provide you with the ability to organize and store data, as well as access and manipulate that data by using SQL.

The **second** thing you need is a database. A database is a collection of data organized in a specific way to facilitate access and editing. There are many different types of databases, but among the most popular are Oracle, PostgreSQL, MySQL, and SQL Server. You can download the open-source PostgreSQL database from their website: https://www.postgresql.org/

The **third** thing you need is an SQL client. An SQL client is a tool that allows you to connect to a database and execute SQL statements. There are many different SQL clients, but among the most popular are MySQL Workbench, SQL Developer, and SQL Server Management Studio. Alternatively, you can use a programming language such as Java, Python, or C# to execute SQL statements for a database.

Fourth, and finally, you need a basic knowledge of SQL syntax and concepts. This includes knowing how to create and manipulate tables, how to use SELECT, INSERT, UPDATE, and

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

DELETE statements to interact with data, and how to use WHERE clauses to filter data. Once you understand these concepts, you can use SQL to extract, manipulate, and manage data in a variety of situations.

2.2 Relational databases are key

Relational databases are a fundamental part of modern data management in organizations of all sizes. They are used to store, manage, and analyse data in a variety of environments, from small businesses to large corporations, government agencies, and non-profit organizations.

One of the main advantages of a relational database is that large amounts of data are stored in a structured and organized manner, which allows data to be retrieved (found) very quickly. For example, you could use a relational database to store and manage all data of the employees in your company.

Relational databases are based on a simple but powerful idea: data can be stored in tables, which consist of rows and columns. Each **table** represents a particular type of data, and a **row** in the table represents a single item. For example, in our table of employees, each row contains a different individual, and each **column** represents a different **attribute** of the employee (e.g. name, address and phone number). A row is also called a **record** and the individual attributes are stored in **fields**. The header row of the table is also called the **scheme**, because here is the reference, which data is stored in which column.

Another benefit of a relational database is the ability to ensure the integrity and consistency of the data. This means that you can set rules and restrictions on how data is entered into the database and then ensure that those rules are followed. For example, you can require that an address be stored for all employees, or a unique phone number. This prevents data errors and inconsistencies that can lead to problems.

There are other types of databases. For example, NoSQL databases are designed to handle unstructured and semi-structured data, such as documents, graphs, and key-value pairs. They are often used for big data analytics and real-time web applications.

With the support of the Erasmus+ programme of the European Union. This document and its contents reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

	Other types are object-oriented databases, graph databases or In-memory databases. The choice of database type depends on the specific needs and requirements of the application used.

With the support of the Erasmus+ programme of the European Union. This document and its contents reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

2.3 SQL Statements

2.3.1 Structure and Syntax of SQL Statements

In this script, we will learn about the common **keywords** (also known as a clause) used in SQL statements and the general **structure and syntax** of SQL statements.

Keywords identifies the type of operation to be performed. The following are the most common keywords used in SQL statements:

- **SELECT:** Used to retrieve data from one or more tables
- **INSERT:** Used to insert new data into a table
- **UPDATE:** Used to update existing data in a table
- **DELETE:** Used to delete data from a table
- **ALTER:** Used to modify the structure of a table
- **DROP:** Used to delete a table or database object
- **CREATE:** Used to create a new table or database object
- **USE:** Used to select a database to work with
- **SHOW:** Used to display information about a database object

The **structure and syntax** of SQL statements can vary depending on the specific database management system (DBMS) being used. However, there are some general guidelines that apply to most SQL statements. A basic SQL statement typically consists of the following elements:

Keyword: The keyword or clause that identifies the type of operation to be performed

Arguments: One or more arguments or parameters that provide additional information about the operation to be performed

Semicolon: All SQL statements end with a semicolon (;)

For example, a basic SELECT statement would look like this:

```
SELECT column1, column2 FROM table_name;
```

In this statement, "SELECT" is the keyword, "column1, column2" are the arguments, and "table_name" is the table from which data is being retrieved. The semicolon at the end of the statement indicates the end of the SQL statement.

2.3.2 SELECT statements

An easy example of a **SELECT** statement in respect with our relational database might look like this:

```
SELECT * FROM employees;
```

"**SELECT**" is the keyword and

"*****" and "**FROM employees**" are the arguments

This SQL statement is retrieving all data from our table called "employees". The * represents all and in this case all columns. In most cases, however, we do not want all the stored data but only a part of it. For example, the name and the associated phone number of all employees:

With the support of the Erasmus+ programme of the European Union. This document and its contents reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

SELECT Name, TelNr **FROM** employees;

Now let's look at the special arguments **FROM** and **WHERE:**
FROM determines the table or tables for this query
WHERE is used to add a condition to your query
if the condition is a text, enclose the text in single quotes
For example, if we want only the phone number of our employees whose last name is Mouse, then we use:
SELECT Name, TelNr **FROM** employees **WHERE** Name = 'Mouse';
SELECT statements are very powerful and support various insights to Data Scientists. For more special arguments see the annex of this script.

2.3.3 Other useful SQL statements

The DML - Data Manipulation Language - commands of SQL are SELECT, INSERT, UPDATE, and DELETE and are essential for managing and manipulating data within a relational database. By learning these 4 commands and how to use them effectively, users can become proficient in SQL and use it to handle a wide range of database tasks.

SELECT: The SELECT command is used to retrieve data from one or more tables in a database. It allows users to specify the columns they want to retrieve and filter the data based on specific criteria. We have worked with them in the last chapter.

INSERT: The INSERT command is used to add new data to a table in a database. It allows users to specify the values to be inserted into the table, and the columns in which those values should be inserted.

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

INSERT INTO is followed by the name of the table into which the data is to be inserted, in the example "table_name"
The next part of the statement specifies the columns in the table where the data is to be inserted, which is represented by "(column1, column2 ...)". After that, the VALUES keyword is used to specify the actual data that will be inserted into the field. The values must match the order of the columns specified in the previous section and are represented by "(value1, value2 ... valueX)".
For example, if we insert data in our table called "employees" with columns "Name", "Adresse" and "TelNr", the SQL statement to insert a new record might look like this:
INSERT INTO employees (Name, Adresse, TelNr) VALUES ('John Smith', 'High Street', 50573);
Note that not all columns in a table need to be specified in the INSERT INTO statement. If a column is left out, the database will

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

either assign a default value or insert a NULL value into that column, depending on how the table was created.
Also, the order in which the values are specified must match the order of the columns. If the order is different or if a value is missing, the database will report an error.

UPDATE: The UPDATE command is used to modify existing data in a table in a database. It allows users to specify the new values that should replace the existing values, and to filter the data based on specific criteria.

UPDATE table_name
SET column1 = value1, column2 = value2 ... columnN = valueN
[**WHERE** condition];

UPDATE table_name specifies the name of the table to update.
SET column1 = value1, column2 = value2 ... columnN = valueN sets the values of one or more columns in the table to new values. Each column and its corresponding value are separated by an equal sign (=), and multiple columns are separated by commas.
[WHERE condition] is an optional clause that specifies the condition(s) that must be met in order for the update to take place. If no condition is specified, all rows in the table will be updated.
Examples for valid conditions:

WHERE Name = 'Mouse';
WHERE Age > 50;
WHERE category = 'Electronics' AND stock_quantity > 0;
WHERE order_date < '2022-01-01';

DELETE: The DELETE command is used to remove data from a table in a database. It allows users to filter the data based on specific criteria, and to remove all or a subset of the data that matches those criteria.

DELETE FROM table_name **WHERE** {condition};

DELETE FROM table_name specifies the name of the table from which rows should be deleted.
WHERE {condition} is an optional clause that specifies the condition(s) that must be met for the rows to be deleted. If no condition is specified, all rows in the table will be deleted!
Examples for valid conditions:

WHERE NameID = 123456;

This statement would delete the row with a NameID of 12345

WHERE NameID **IN** (12345, 23456, 34567);

This statement would delete all rows where the "NameID" is either 12345 or 23456 or 34567.
It is important to exercise caution when using the DELETE statement, as it permanently remove data from a table. It is

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

	recommended to make a backup of the data or use a transaction to rollback the changes if necessary. In conclusion, SQL is a powerful and flexible database language that is widely used in many different industries and application areas. Its ease of use, flexibility, and efficiency make it an excellent choice for businesses and organizations that need to process and maintain large amounts of data. With its many advantages and resources available for learning, SQL is a valuable tool for anyone working with relational databases.
Self-assessment (multiple choice queries and answers)	1. Who uses SQL? A) Humans to work with data stored in a relational database B) Web-developers to code Websites C) Database administrators to manage the data 2. What is SQL used for? A) To create and delete data in a database B) For web shop payments C) To retrieve data from a non-database environment 3. Which keywords can start a SQL statement? A) FROM B) WHERE C) SELECT
Resources (videos, reference link)	https://www.postgresql.org/ https://www.youtube.com/watch?v=HXV3zeQKqGY
Related material	
Contents arranged in 3 levels	**3. Understanding GitHub** **3.1 Purpose of GitHub** **3.1.1 Managing Large Software Code** For all software developers, managing the code is an essential part of programming. Version control systems allow developers to control their code changes, collaborate with others, and manage their projects efficiently. A popular version control system is Git, and GitHub is a cloud-based version control service for software development projects built on top of Git. GitHub provides a user-friendly interface for creating and managing Git repositories, as well as tools for collaborating with other developers on a project. GitHub can be used for both personal and commercial projects and is free in its basic edition.

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

GitHub can be characterized as a kind of social network for software developers. Members can follow each other, rate each other's work, get updates on specific projects, and communicate publicly or privately. It enables developers to share their work, learn from others, and build a community.

GitHub also provides a comprehensive set of project management tools that make it easy for developers to manage their projects. It includes tools for tracking issues, organizing tasks, and collaborating with others.

GitHub integrates with many other tools and services, including continuous integration and deployment services. This allows developers to automate the process of testing, building, and deploying their code changes.

Since the end of 2018, GitHub Inc is owned by Microsoft.

3.1.2 What is version control?

When working on complex software projects, keeping track of changes and managing multiple versions of code can be a challenging task. Version control systems like Git provide a solution to this problem by tracking and recording the changes.

Git provides functionalities to recover old versions of a project, compare, analyse, merge changes, and much more. This process is called **version control**, and it helps developers keep track of the history of their code. Git is not the only version control system available. Other version control systems like Perforce, Mercurial, CVS, and SVN are also available. However, Git has become the most popular and widely used version control system among developers due to its flexibility, speed, and ease of use.

One of the unique features of Git is that it is a decentralized version control system. Unlike others, Git doesn't depend on a central server to keep old versions of files. Instead, it works completely locally, storing that data as folders on the user's hard drive. This is called a **repository**. This allows developers to work on their code offline and makes it easy to track changes without relying on a central server.

If a user wants to work with others on the same code, they can provide a copy of their repository online for the whole team to

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

access. This allows multiple developers to work on the same codebase without overwriting each other's changes.

3.2 How to use

3.2.1 Install it on your Device

To use Github in a productive way, you need Git on your local computer. There are many Git programs that can be used for free or with a fee:

Windows - "Git for Windows" provides a GUI client and a BASH command line emulator.
https://git-scm.com/downloads/win or
https://gitforwindows.org/

Linux - just open a new terminal and install Git via the package manager of your Linux distribution. For Ubuntu, the command is e.g. sudo apt-get install git

Mac OS - The easiest way is to install homebrew and then just run brew install git from your terminal.
https://brew.sh/ or https://git-scm.com/downloads/mac

3.2.2 Create an account and configure GitHub

Before you can use it, create a Github account on
www.github.com

There are several configurations to the appearance and functionality of the client. Decide according to your preferences to work efficiently.

However, important is the configuration of the **username** and the corresponding **email address**
open "Git Bash" and execute these commands on the command line:
git config --global user.name "Mary"
git config --global user.email "mary.smith@email.eu"

All activities in Git are linked to the respective username and e-mail address specified in the configuration! This makes modifications traceable because other users always know who made the specific changes and provides overview in projects where many developers are working.

3.3 How to use GitHub

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

3.3.1 Common Terms in GitHub

When working with Git, it is essential to understand its terminology to be able to use it effectively.

A **repository**, or repo for short, is a folder in which all files and their version histories are stored. It is the central location for managing and organizing code. A repository can be hosted on a remote server, such as GitHub or Bitbucket, or it can be stored locally on your computer.

A **branch** is a workspace in which you can make isolated changes that won't affect others and has its own history. It is like a separate timeline in which you can experiment with changes without affecting the main codebase. Developers can work on different branches simultaneously, making it easier to collaborate and avoid conflicts.

A **commit** is a saved record of modifications made to a file within the repo. It is the state of our repository at a point in time. A snapshot to which the code can be restored. Commits are essential in Git because they allow you to track changes over time and revert to a previous version of your code if necessary.

After updates are made to a repository, other developers can download all changes with a **pull request** or PR. A pull request is a request to merge changes from one branch into another. This allows developers to review and approve changes before they are merged into the main codebase.

Push is the process of adding a local change to the remote repository. When you push your changes, they become available for others to see and download. This is an essential step in collaborating with other developers.

After a pull request is approved, the commit will be **merged** from one branch to another. Merging combines changes from one branch into another, typically the main branch. This process ensures that all changes are incorporated into the main codebase.

A **clone** is a fully functional copy of a project. Copying a repository from a remote server is cloning. This allows you to have a complete copy on the local machine, making it easier to work on your code without relying on an internet connection.

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

3.3.2 First steps in Github

When using Git, first a folder must be created

mkdir learning-git (=**m**a**k**e **dir**ectory)

in which the repository can be created for each program or project

cd learning-git (= **c**hange **d**irectory)

git init

Following this, you can link a local and an online repo with

git remote add origin https://github.com/[username]/[projectname].git

Git works with the concept of a "staging area". At the beginning, the staging area is empty. Files can added (or even single lines and parts of files) with the git add command and finally commit everything (create a snapshot) with git commit:

git add learning_script.txt

git commit –m «firstsnapshot»

To upload code to a remote repo, first connect to it

git remote add origin https://github.com/learning.git

Then, the local commits can be transferred to the server, performed each time we want to update the remote repository

git push server_origin local_master

Once this is done, other developers can download the changes from the remote repository with a single command

git pull origin master

To clone a whole program or project use

git clone https://github.com/tutorial.git

3.3.3 Branch related commands

When developing a new feature in software development, it is best practice to work on a copy of the original project, called a branch. A branch is a separate copy of the codebase that allows developers to make changes without affecting the live version of the code.

Each branch has its own history and isolates the changes made to it from other branches until it is decided to merge them. The benefits of this approach are:

1. A stable (live) version of the code is not affected by unwanted bugs: When working on a separate branch, any bugs or errors that occur during the development process do not affect the live version of the code.
2. A team of developers can work on many features at the same time: Branches allow multiple developers to work on different

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

	features at the same time without interfering with each other's work. 3. Every developer can work on their own branch without the risk of having their codebase changed by another developer's work: Working on separate branches ensures that developers can work independently and avoid conflicts between their code changes. 4. Multiple versions of the same feature can be developed on different branches and then compared to find out the best version: Branches enable developers to experiment with different approaches to a feature and compare the results before merging the changes into the main codebase. In summary, using branches is a best practice in software development as it helps to avoid conflicts and errors, enables teamwork, and allows for experimentation and comparison of different approaches to a feature. The default branch of each repository is called master. To create more branches, use the command git branch <name> Switch to the newly created branch by using git checkout <name> To merge two branches switch to one and use git merge <name> To delete a branch use git branch –d <name> In conclusion, many companies are using GitHub. So, if you're looking for a job, you'll do well if you're already familiar with GitHub. GitHub is also a learning and collaboration platform. Explore it and expand your knowledge and community.
Self-assessment (multiple choice queries and answers)	1. Why is version control important? A) Quick develop code B) Develop a program with a team in a traceable process C) Needed for selling the project 2. Who can use GitHub? A) Everyone with a Git account B) Everyone who has bought it C) Only trained developer 3. What is a branch? A) A licensed company

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

	B) A specific project C) A copy of the original project
Resources (videos, reference link)	https://www.github.com https://git-scm.com/downloads/win https://gitforwindows.org/ https://brew.sh/ https://git-scm.com/downloads/mac
Related material	
Related PPT	
Bibliography	
Provided by	[Women in AI Austria]

With views

Co-funded by the Erasmus+ Programme of the European Union

the support of the Erasmus+ programme of the European Union. This document and its contents reflect the only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.