

# RELATIONAL DATABASES TOOLS

## DataBase Foundations

Author: Eng. Carlos Andrés Sierra, M.Sc.  
cavirguezs@udistrital.edu.co

Lecturer  
Computer Engineer  
School of Engineering  
Universidad Distrital Francisco José de Caldas

2024-III



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment
- 5 DevOps



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment
- 5 DevOps



# What is a DBMS?

- A **Database Management System** (DBMS) is a **software system** that uses a **standard** method to **store** and **organize data**.
- The **data** can be stored in **tables**, which are **related** to each other.
- A **DBMS** is a **software system** that allows users to **define**, **create**, **maintain**, and **control access** to the database.
- A **DBMS** is a **software package** designed to **manipulate**, **retrieve**, and **manage data** in a database.



# What is a DBMS?

- A **Database Management System** (DBMS) is a **software system** that uses a **standard** method to **store** and **organize data**.
- The **data** can be stored in **tables**, which are **related** to each other.
- A **DBMS** is a **software system** that allows users to **define**, **create**, **maintain**, and **control access** to the database.
- A **DBMS** is a **software package** designed to **manipulate**, **retrieve**, and **manage data** in a database.



# What is a DBMS?

- A **Database Management System** (DBMS) is a **software system** that uses a **standard** method to **store** and **organize data**.
- The **data** can be stored in **tables**, which are **related** to each other.
- A **DBMS** is a **software system** that allows users to **define**, **create**, **maintain**, and **control access** to the database.
- A **DBMS** is a **software package** designed to **manipulate**, **retrieve**, and **manage data** in a database.



# What is a DBMS?

- A **Database Management System** (DBMS) is a **software system** that uses a **standard** method to **store** and **organize data**.
- The **data** can be stored in **tables**, which are **related** to each other.
- A **DBMS** is a **software system** that allows users to **define**, **create**, **maintain**, and **control access** to the database.
- A **DBMS** is a **software package** designed to **manipulate**, **retrieve**, and **manage data** in a database.



# Pros & Cons of DBMS

- **Pros:**

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

- **Cons:**

- **Complexity:** DBMS are complex systems that require specialized knowledge to use.
- **Cost:** DBMS can be expensive to purchase and maintain.
- **Performance:** DBMS can be slow to respond to queries.





# Pros & Cons of DBMS

- **Pros:**

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

- **Cons:**

- **Complexity:** DBMSs are complex systems that require specialized knowledge to use effectively.
- **Cost:** DBMSs can be expensive to purchase and maintain.
- **Performance:** DBMSs can be slower than other data storage methods.
- **Security:** DBMSs can be vulnerable to security attacks.



# Pros & Cons of DBMS

- **Pros:**

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

- **Cons:**

- **Complexity:** DBMS are complex systems.
- **Performance:** DBMS can be slow.
- **Cost:** DBMS can be expensive.



# Pros & Cons of DBMS

## • Pros:

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

## • Cons:

- **Complexity:** DBMS are complex systems.
- **Cost:** DBMS are expensive for bigger data volumes.



# Pros & Cons of DBMS

## • Pros:

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

## • Cons:

- **Complexity:** DBMS are **complex systems**.
- **Cost:** DBMS are **expensive** for bigger data volumes.
- **Performance:** DBMS can be **slow**.



# Pros & Cons of DBMS

- **Pros:**

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

- **Cons:**

- **Complexity:** DBMS are **complex systems**.
- **Cost:** DBMS are **expensive** for **bigger data volumes**.
- **Performance:** DBMS can be **slow**.



# Pros & Cons of DBMS

- **Pros:**

- **Data Independence:** Data is **stored independently** of the applications that use it.
- **Data Integrity:** Data is **consistent** and **accurate**.
- **Data Security:** Data is **protected** from **unauthorized access**.
- **Data Recovery:** Data can be **recovered** in case of **failure**.

- **Cons:**

- **Complexity:** DBMS are **complex systems**.
- **Cost:** DBMS are **expensive** for **bigger data volumes**.
- **Performance:** DBMS can be **slow**.



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment
- 5 DevOps



# GUI Assistants

- A **Graphical User Interface** (*GUI*) is a type of user interface that allows **users** to **interact** with electronic devices using **graphical icons** and **visual indicators**.
- **GUIs** are easier to use than **Command Line Interfaces** (*CLI*) because they allow **users** to **interact** with the system using **visual elements** such as **windows**, **buttons**, and **menus**.
- **GUIs** are more **intuitive** and **user-friendly** than CLIs, which makes them ideal for **users** who are **not familiar** with the system.





# Case of Study: DBeaver

The screenshot shows the DBeaver 22.1.3 interface. On the left, the 'Database Navigator' tree shows the hierarchy: system -> localhost:26257 -> Databases -> movr -> public -> Tables -> rides. The main panel displays the 'Properties' tab for the 'rides' table. The 'Table Name' is 'rides', 'Object ID' is '111', and 'Owner' is 'root'. Below this, the 'Columns' tab shows a list of columns with their data types and constraints.

Column Name	#	Data type	Identity	Collation	Not Null	Default	Comm
id	1	uuid			[v]		
city	2	varchar		default	[v]		
vehicle_city	3	varchar		default	[ ]		
rider_id	4	uuid			[ ]		
vehicle_id	5	uuid			[ ]		
start_address	6	varchar		default	[ ]		
end_address	7	varchar		default	[ ]		
start_time	8	timestamp			[ ]		
end_time	9	timestamp			[ ]		
revenue	10	numeric(10, 2)			[ ]		

At the bottom, the 'Project - General' tab shows the 'DataSource' field with the value 'PST\_en\_US'.



# Command Line

- A **Command Line Interface (CLI)** is a type of **user interface** that allows **users** to **interact** with electronic devices using **text-based commands**.
- CLIs are more **powerful** and **flexible** than GUIs because they allow users to perform **complex tasks** using **simple commands**.
- **CLIs** are more **efficient** than GUIs because they do **not require** users to **navigate** through menus and windows to perform tasks.



# Case of Study: MariaDB CLI

```
arnel@arnel.com [~]# mysql -u arnel_test2 -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8643
Server version: 10.1.25-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| arnel_test1 |
| arnel_test2 |
| information_schema |
+-----+
3 rows in set (0.00 sec)

MariaDB [(none)]> use arnel_test1
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [arnel_test1]>
```



# Why use an agnostic tool?

- An **agnostic tool** is a tool that is **not tied** to a specific **technology** or **platform**.
- **Agnostic tools** are useful because they **allow users** to work with **multiple databases** without having to **learn different tools**.
- **Agnostic tools** are also useful because they **allow users** to work with **multiple databases** without having to **switch between different tools**.



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud**
- 4 Local Environment
- 5 DevOps



# What is the cloud Computing?

- **Cloud computing** is the delivery of **computing services** over the **internet**.
- **Cloud computing** allows users to **access computing resources** such as **servers, storage, and databases on demand**.



# What is the cloud Computing?

- **Cloud computing** is the delivery of **computing services** over the **internet**.
- **Cloud computing** allows users to **access computing resources** such as **servers, storage, and databases on demand**.



# What is On-Premises Computing?

- **On-premises computing** is the **traditional** way of accessing **computing resources**.
- **On-premises computing** requires users to **purchase** and **maintain** their own **computing** resources such as **servers**, **storage**, and **databases**.





# What is On-Premises Computing?

- **On-premises computing** is the **traditional** way of accessing **computing resources**.
- **On-premises computing** requires users to **purchase** and **maintain** their own **computing** resources such as **servers**, **storage**, and **databases**.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a cost-effective way to access computing resources.
- **Scalable:** Cloud computing is a scalable way to access computing resources.
- **Flexible:** Cloud computing is a flexible way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be a security risk for businesses that store sensitive data in the cloud.
- **Performance:** Cloud computing can be slower than local computing, especially for applications that require high performance.
- **Complexity:** Cloud computing can be complex to set up and manage, especially for businesses that are not familiar with cloud technology.
- **Dependency:** Cloud computing can be a dependency on a single provider, which can be a risk if the provider goes out of business.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a cost-effective way to access computing resources.
- **Scalable:** Cloud computing is a scalable way to access computing resources.
- **Flexible:** Cloud computing is a flexible way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be less secure than on-premises computing.
- **Performance:** Cloud computing can be slower than on-premises computing.
- **Cost:** Cloud computing can be more expensive than on-premises computing.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a **cost-effective** way to access computing resources.
- **Scalable:** Cloud computing is a **scalable** way to access computing resources.
- **Flexible:** Cloud computing is a **flexible** way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be less secure than on-premises computing.
- **Performance:** Cloud computing can be slower than on-premises computing.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a **cost-effective** way to access computing resources.
- **Scalable:** Cloud computing is a **scalable** way to access computing resources.
- **Flexible:** Cloud computing is a **flexible** way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be **less secure** than on-premises computing.
- **Performance:** Cloud computing can be **slower** than on-premises computing.
- **Reliability:** Cloud computing can be **less reliable** than on-premises computing.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a **cost-effective** way to access computing resources.
- **Scalable:** Cloud computing is a **scalable** way to access computing resources.
- **Flexible:** Cloud computing is a **flexible** way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be **less secure** than on-premises computing.
- **Performance:** Cloud computing can be **slower** than on-premises computing.
- **Reliability:** Cloud computing can be **less reliable** than on-premises computing.



# Pros & Cons of Cloud Computing

- **Pros:**

- **Cost-Effective:** Cloud computing is a **cost-effective** way to access computing resources.
- **Scalable:** Cloud computing is a **scalable** way to access computing resources.
- **Flexible:** Cloud computing is a **flexible** way to access computing resources.

- **Cons:**

- **Security:** Cloud computing can be **less secure** than on-premises computing.
- **Performance:** Cloud computing can be **slower** than on-premises computing.
- **Reliability:** Cloud computing can be **less reliable** than on-premises computing.



# SaaS Vs. IaaS Vs. PaaS

- **Software as a Service** (*SaaS*) is a **software distribution** model in which a **third-party** provider **hosts applications** and makes them available to customers over the **internet**.
- **Infrastructure as a Service** (*IaaS*) is a **cloud computing** model that provides **virtualized computing** resources over the **internet**.
- **Platform as a Service** (*PaaS*) is a **cloud computing** model that provides a **platform for developers to build, deploy, and manage applications** over the **internet**.





# SaaS Vs. IaaS Vs. PaaS

- **Software as a Service** (*SaaS*) is a **software distribution** model in which a **third-party** provider **hosts applications** and makes them available to customers over the **internet**.
- **Infrastructure as a Service** (*IaaS*) is a **cloud computing** model that provides **virtualized computing** resources over the **internet**.
- **Platform as a Service** (*PaaS*) is a **cloud computing** model that provides a **platform for developers to build, deploy, and manage applications** over the **internet**.

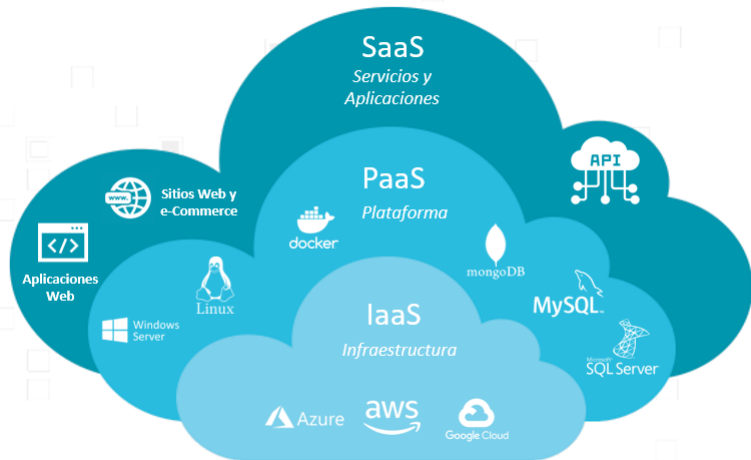


# SaaS Vs. IaaS Vs. PaaS

- **Software as a Service** (*SaaS*) is a **software distribution** model in which a **third-party** provider **hosts applications** and makes them available to customers over the **internet**.
- **Infrastructure as a Service** (*IaaS*) is a **cloud computing** model that provides **virtualized computing** resources over the **internet**.
- **Platform as a Service** (*PaaS*) is a **cloud computing** model that provides a **platform for developers** to **build, deploy, and manage applications** over the **internet**.



# Cloud Levels

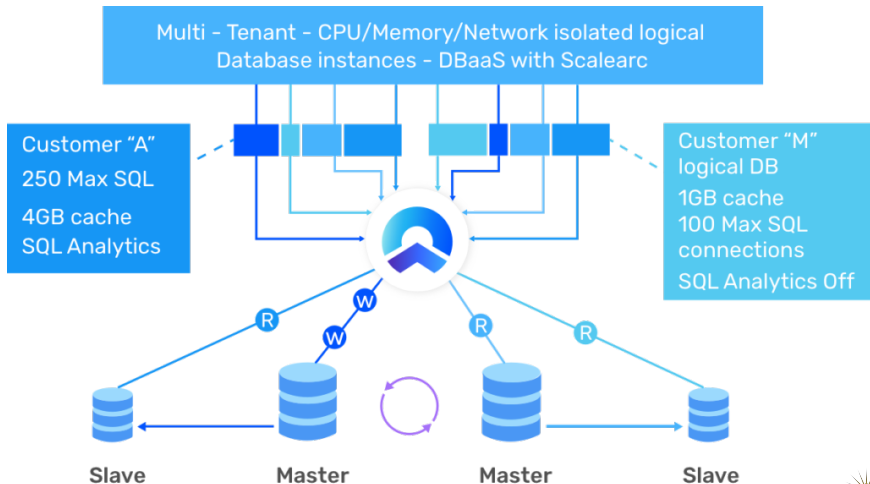


# DataBases as a Service

**Database as a Service** (DBaaS) is a **cloud computing model** that provides **database services** over the **internet**.



# Case of Study: DBaaS Custom for Clients



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment**
- 5 DevOps



# Localhost

- **Localhost** is a **hostname** that refers to the **local computer** that a **program** is **running on**.
- **Localhost** is used to **access the services** that are **running on the local computer**.
- **Localhost** is used to **access the database services** that are **running on the local computer**.
- **Localhost** is used to **access the web services** that are **running on the local computer**.



# Monolithic Architecture

- **Monolithic Architecture** is a **software architecture** in which all the **components** of the software are **combined** into a **single program**.
- **Monolithic Architecture** is a traditional software architecture that was used to **build large** and **complex** software systems.
- **Monolithic Architecture** is a simple and easy-to-understand software architecture that is used to **build** software systems that do **not require** high scalability and flexibility.





# Monolithic Architecture

- **Monolithic Architecture** is a **software architecture** in which all the **components** of the software are **combined** into a **single program**.
- **Monolithic Architecture** is a **traditional software architecture** that was used to **build large** and **complex software systems**.
- **Monolithic Architecture** is a simple and easy-to-understand software architecture that is used to build software systems that do not require high scalability and flexibility.



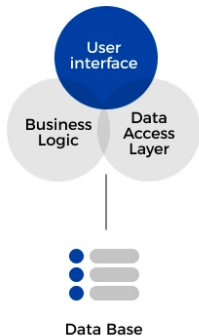
# Monolithic Architecture

- **Monolithic Architecture** is a **software architecture** in which all the **components** of the software are **combined** into a **single program**.
- **Monolithic Architecture** is a **traditional software architecture** that was used to **build large** and **complex software systems**.
- **Monolithic Architecture** is a simple and **easy-to-understand software architecture** that is used to **build software systems** that do **not require** high scalability and flexibility.

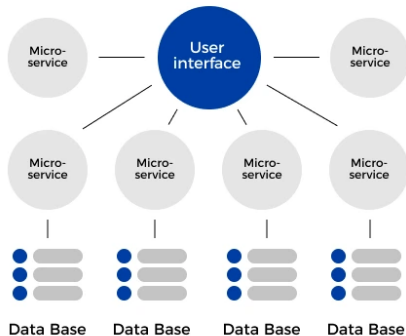


# Monolithic Architecture Schema

## MONOLITHIC ARCHITECTURE



## MICROSERVICE ARCHITECTURE



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment
- 5 DevOps



# Continuous Integration

- **Continuous Integration** is a software development practice in which developers integrate code into a shared repository frequently.
- Continuous Integration is a software development practice that helps developers to detect and fix integration errors early.



# Continuous Integration

- **Continuous Integration** is a software development practice in which developers integrate code into a shared repository frequently.
- **Continuous Integration** is a software development practice that helps developers to detect and fix integration errors early.



# Continuous Deployment

- **Continuous Deployment** is a software development practice in which developers **deploy code** into **production frequently**.
- Continuous Deployment is a software development practice that helps developers to **deliver new features** to customers quickly.
- Continuous Deployment is a software development practice that helps developers to **improve the quality** of the software.



# Continuous Deployment

- **Continuous Deployment** is a software development practice in which developers **deploy code** into **production frequently**.
- **Continuous Deployment** is a software development practice that helps developers to **deliver new features** to customers **quickly**.
- **Continuous Deployment** is a software development practice that helps developers to **improve the quality** of the software.



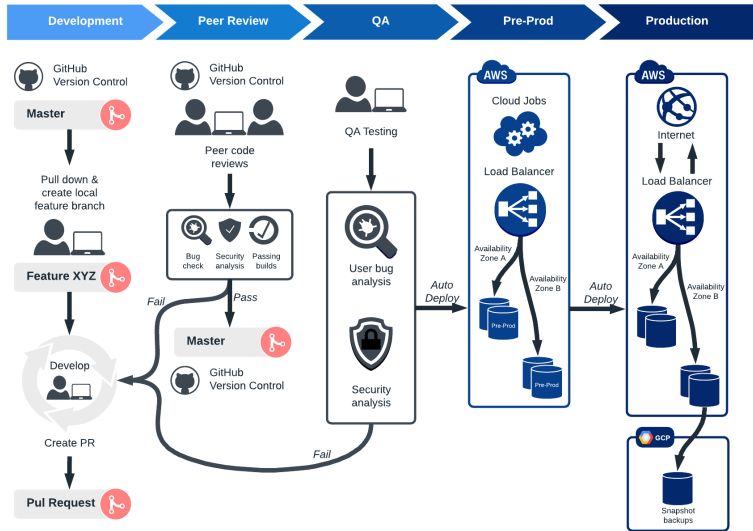


# Continuous Deployment

- **Continuous Deployment** is a software development practice in which developers **deploy code** into **production** frequently.
- **Continuous Deployment** is a software development practice that helps developers to **deliver new features** to customers **quickly**.
- **Continuous Deployment** is a software development practice that helps developers to **improve** the **quality** of the **software**.



# Development Workflow using CI/CD



# Containers and Docker

- **Containers** are a **lightweight** and **portable** way to **package software**.
- **Containers** are a way to **isolate applications** from the underlying system.
- **Docker** is a **platform** that allows developers to **build, ship, and run** containers.



# Containers and Docker

- **Containers** are a **lightweight** and **portable** way to **package software**.
- **Containers** are a way to **isolate applications** from the underlying system.
- **Docker** is a **platform** that allows developers to **build, ship, and run** containers.

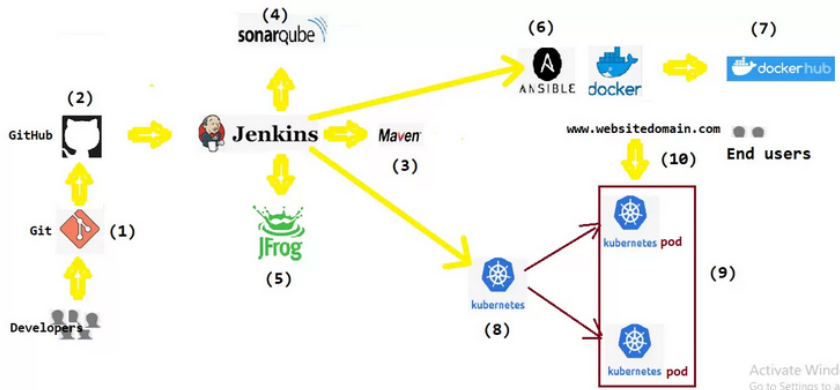


# Containers and Docker

- **Containers** are a **lightweight** and **portable** way to **package software**.
- **Containers** are a way to **isolate applications** from the underlying system.
- **Docker** is a **platform** that allows developers to **build**, **ship**, and **run containers**.



# From Code to Docker



Activate Wind  
Go to Settings to a



# Outline

- 1 DataBase Management Systems — DBMS
- 2 UI: GUI vs. CLI
- 3 DataBases in the Cloud
- 4 Local Environment
- 5 DevOps



# Thanks!

## Questions?



Repo: <https://github.com/EngAndres/ud-public/tree/main/courses/databases-foundations>

