Advanced Coding and Cloud Computation

Romolo Politi

PhD Course 2024

Lecture List

Elenco Lezioni

- September 16th 2024

Lecture September 16th 2024

Course Overview

Course Overview

Cloud
- Cloud structure
- Data in the Cloud
- Cloud Computing

Data
- Data and Metadata
- Archives
- Relational and not-relational Database

Computing
- Retrieval
- Manipulation
- Visualization

Environment
- Virtualization and Containers
- Microservices
- DevOps

Coding
- Fundamentals of Coding
- Python
- Versioning and Documentation

Tools

- Slides and Examples available on GitHub:
  - https://github.com/RomoloPoliti-INAF/PhDCourse2024
- The example will be written in Python 3.12
- Microsoft Visual Studio Code will be used as framework
  - https://code.visualstudio.com

Struttura del Corso

- The list of topics shown earlier was organized by categories.
- We will follow an example-driven approach to better understand the philosophy behind it.
- After the introduction to programming, we will develop an example of a complex program (State Machine).
- Lastly, we will develop a WebApp and prepare it for deployment in containers.
- For some topics, we will not go into detail because the purpose of the course is to provide a general overview of the subject.
- Even though they won't be discussed, many details will be available in the slides or through the provided links.

Cloud Definition

What's Cloud

It is the on-demand availability of computer system resources, especially data storage (cloud storage) and computing power, without direct active management by the user.

wikipedia

Cloud Types

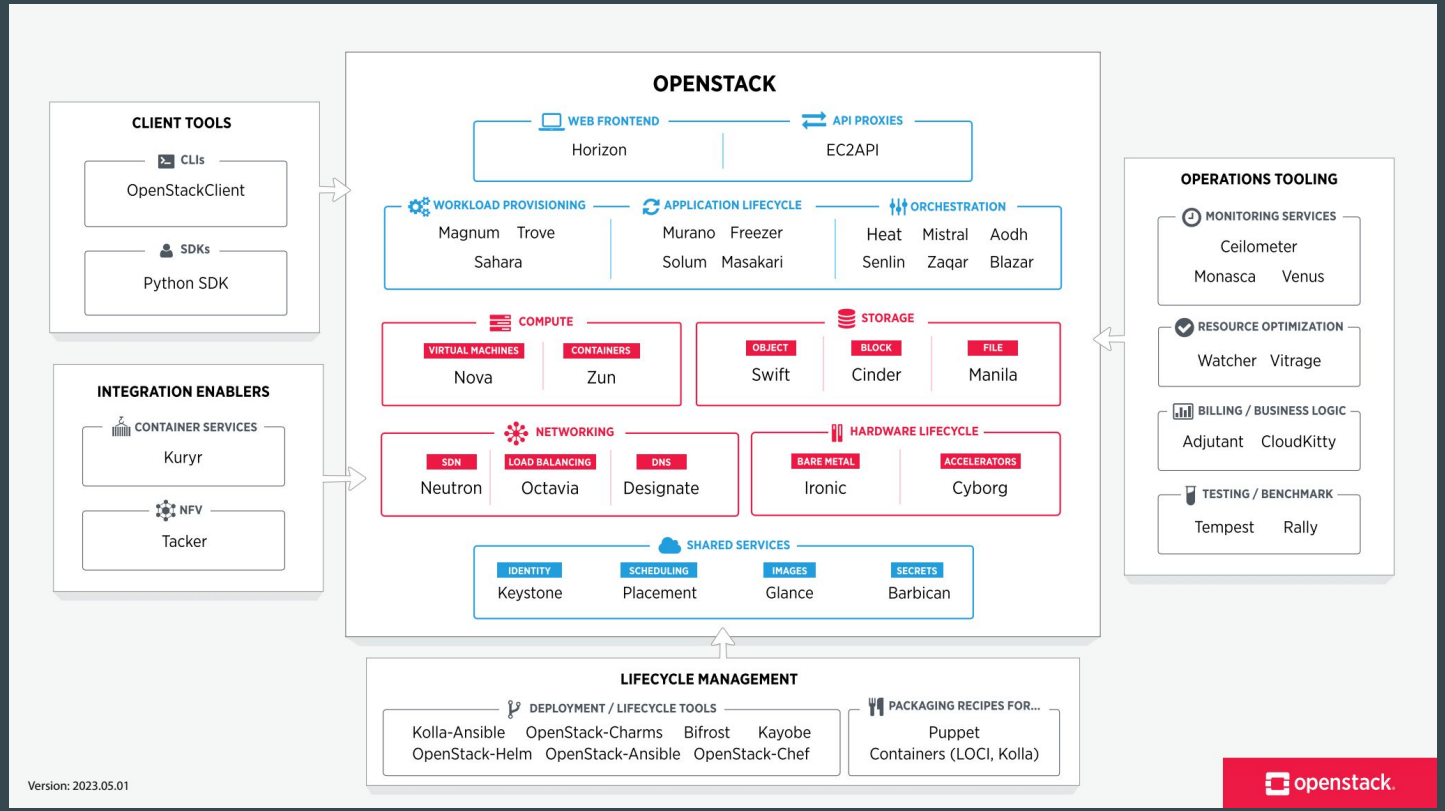
In Promise

Out Promise

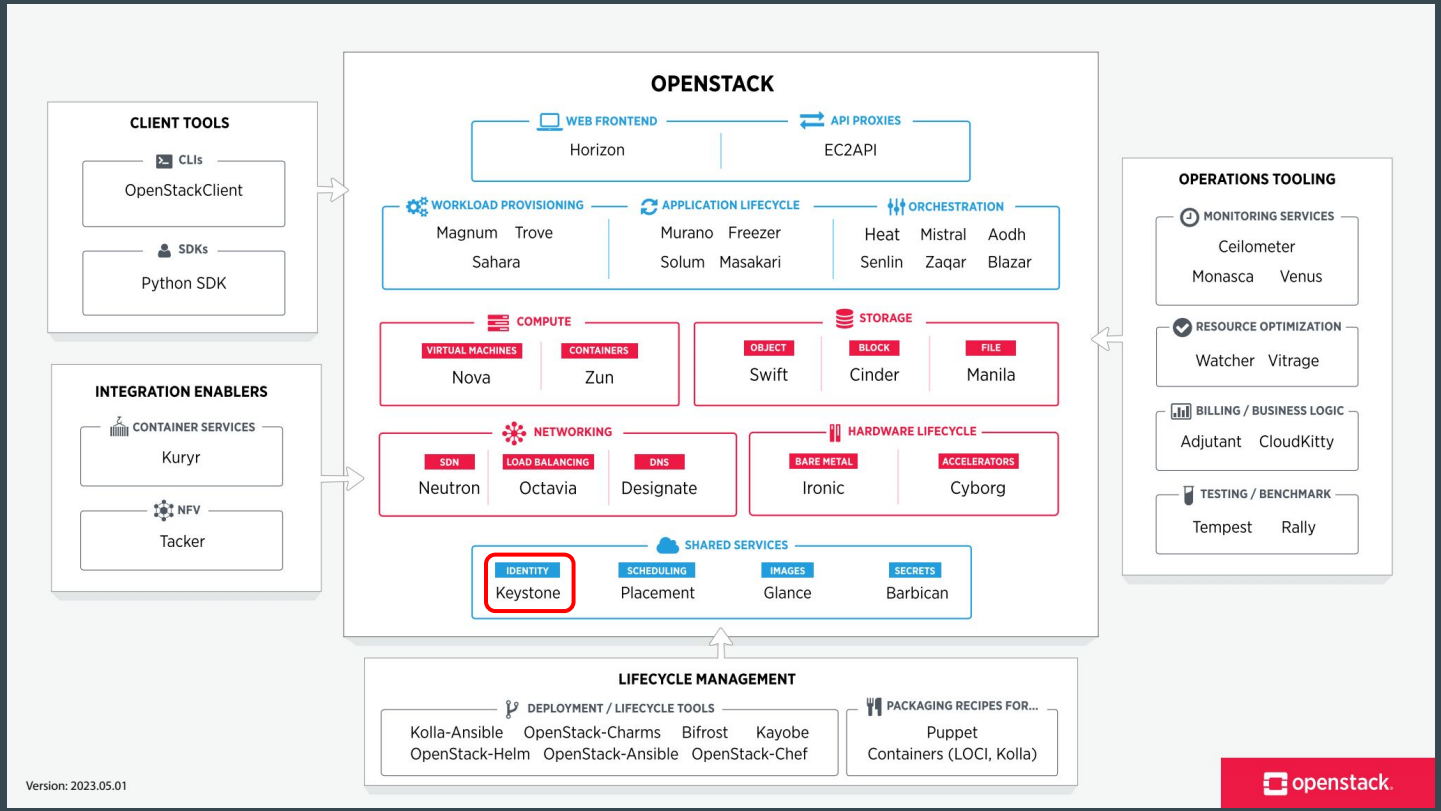
Cloud Structure

Cloud Structure

https://www.openstack.org/

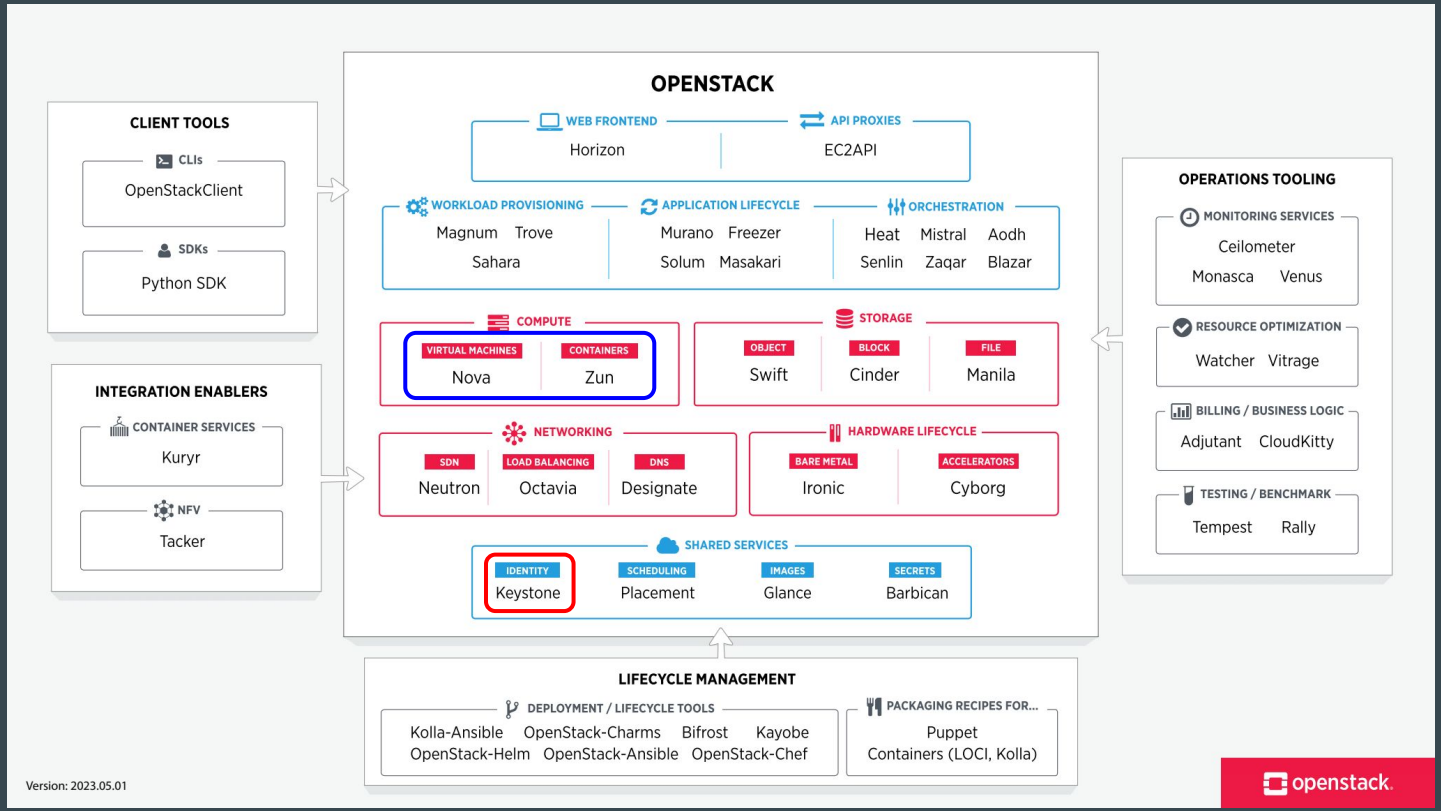


Cloud Structure



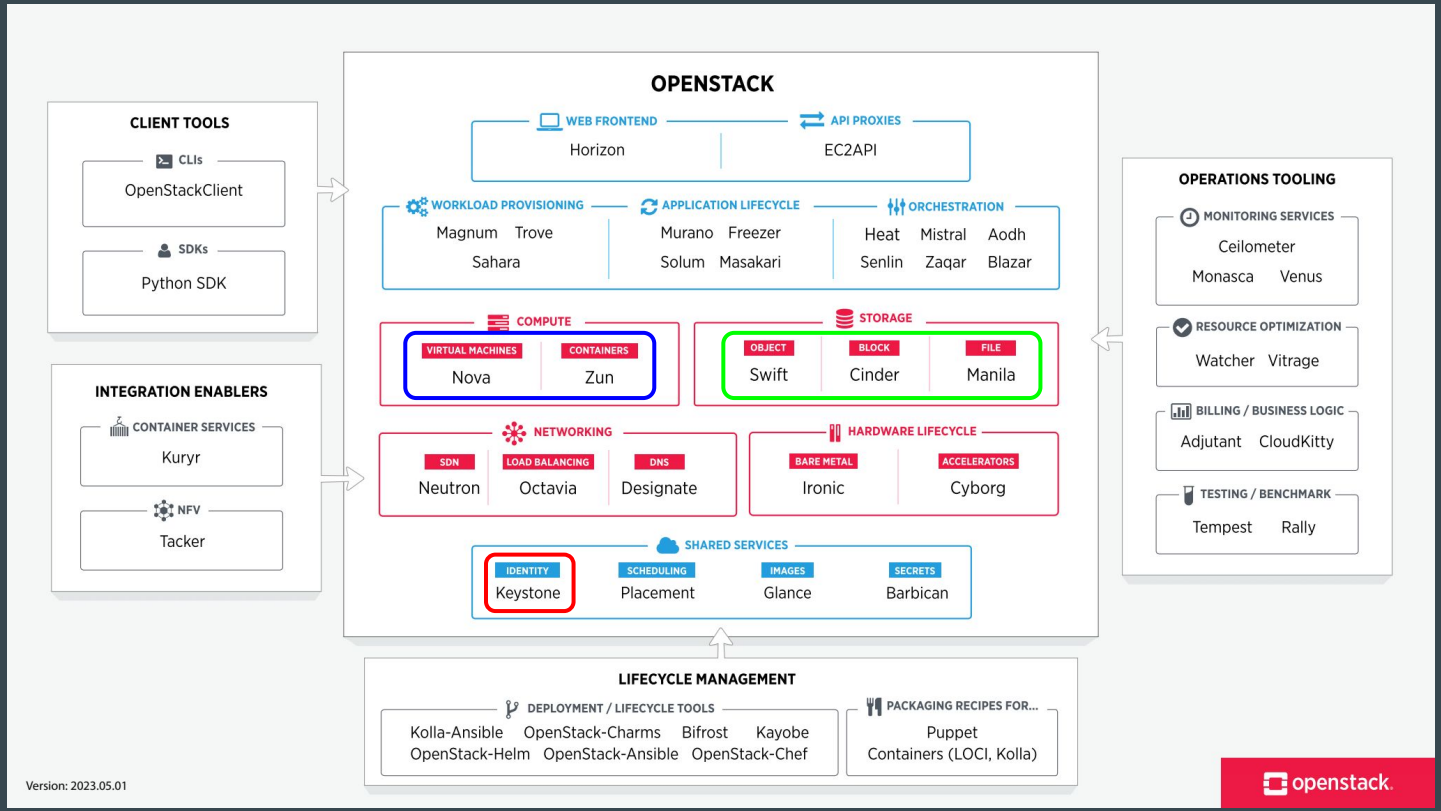
- Identity

Cloud Structure



- Identity
- Compute

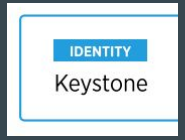
Cloud Structure



- Identity
- Compute
- Storage

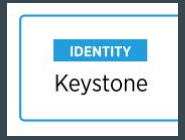
Main Components

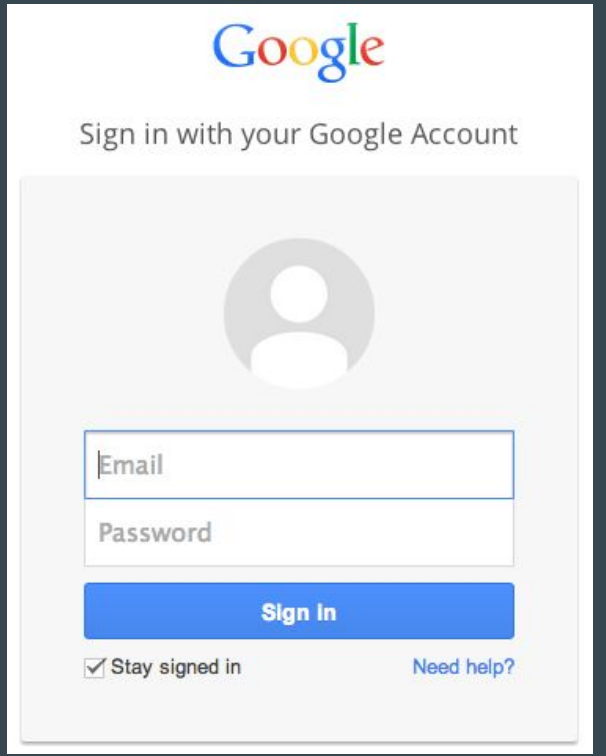
- IAM (Identity and Access Management)



Main Components

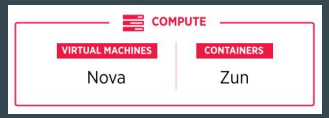
- IAM (Identity and Access Management)
  - identity check
  - list of resources
  - privileges
  - credits (cloud off premise)





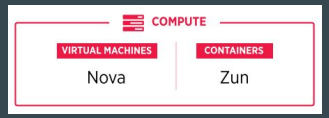
Main Components

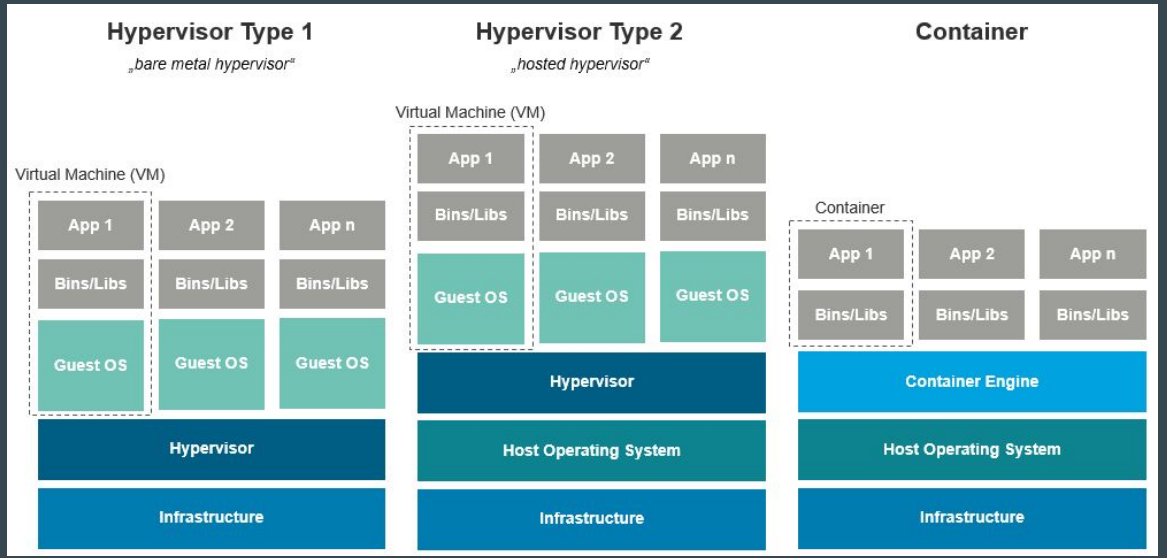
- IAM (Identity and Access Management)
  - identity check
  - list of resources
  - privileges
  - credits (cloud off premise)
- Compute Services



Main Components

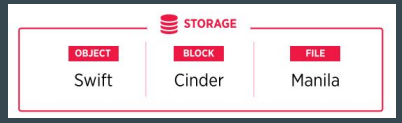
- IAM (Identity and Access Management)
  - identity check
  - list of resources
  - privileges
  - credits (cloud off premise)
- Compute Services





Main Components

- IAM (Identity and Access Management)
  - verifica identità
  - lista di risorse dedicate
  - privilegi
  - Credito (cloud off premise)
- Compute Services
- Storage Services

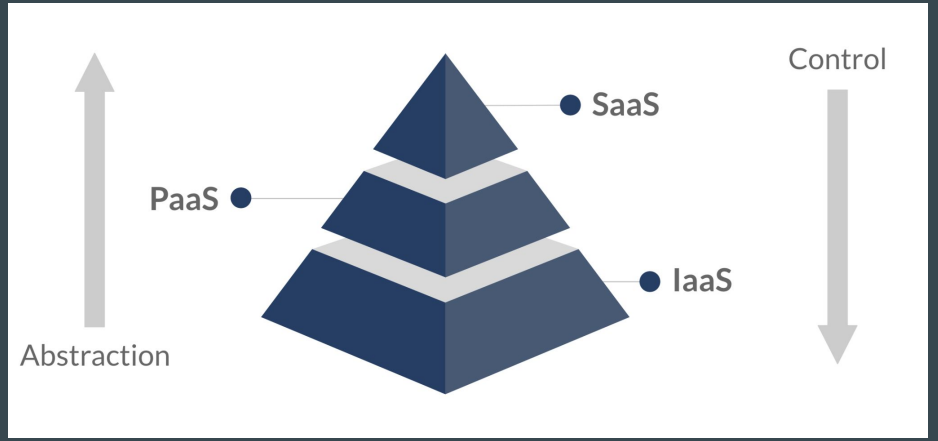


Servizi Cloud

- Infrastructure as a Service
- Platform as a Service
- Software as a Service

Cloud Services

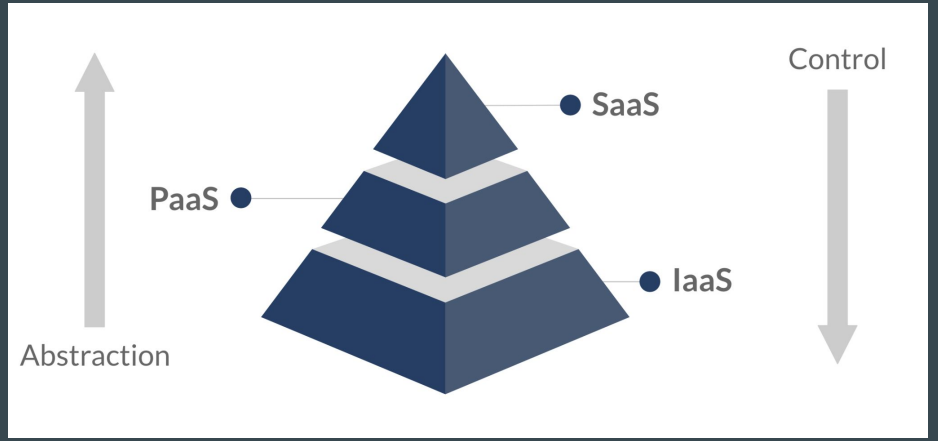
- Infrastructure as a Service
- Platform as a Service
- Software as a Service



IaaS
provides virtualized computing resources (CPU, RAM, disks, etc.) over the internet, allowing users to manage and scale hardware infrastructure without physical ownership.

Cloud Services

- Infrastructure as a Service
- Platform as a Service
- Software as a Service

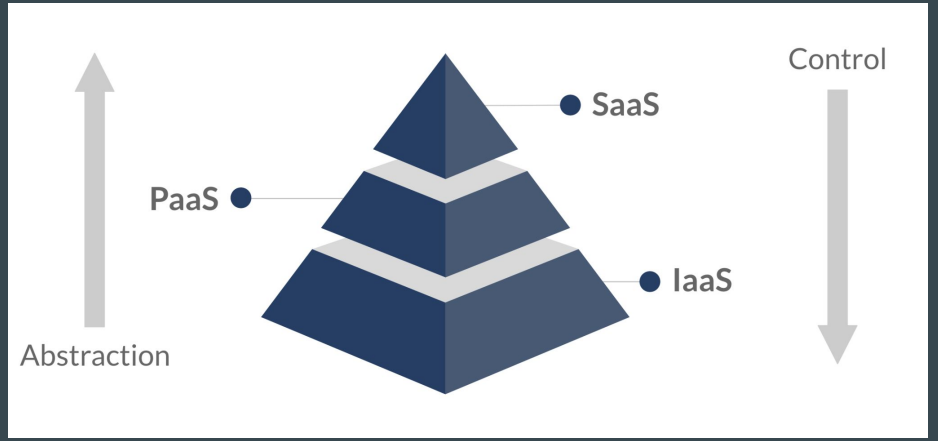


PaaS

It provides a cloud-based environment where developers can build, deploy, and manage applications without dealing with the underlying infrastructure.

Servizi Cloud

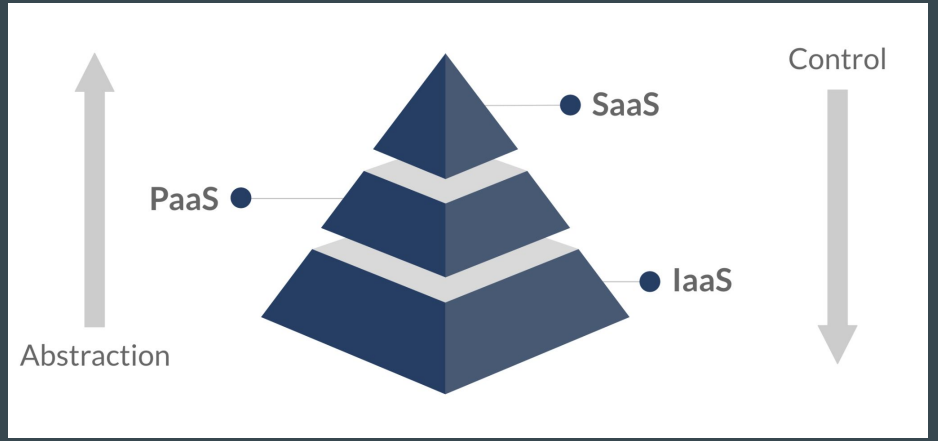
- Infrastructure as a Service
- Platform as a Service
- Software as a Service

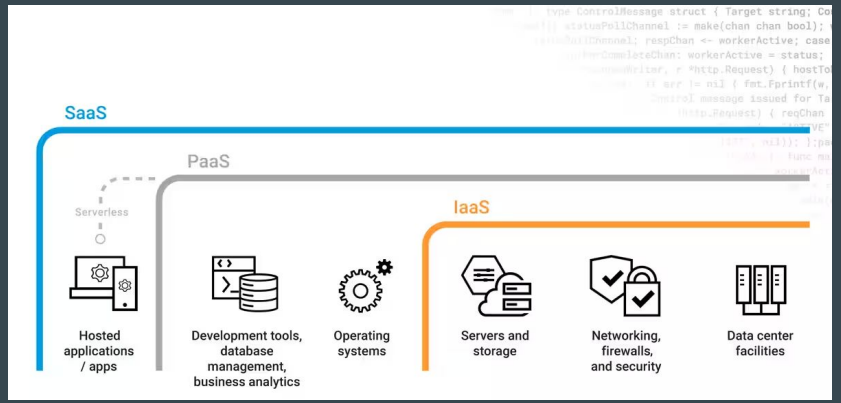


SaaS
it is a cloud-based model where applications are hosted and provided over the internet, allowing users to access and use software without managing the underlying infrastructure.

Servizi Cloud

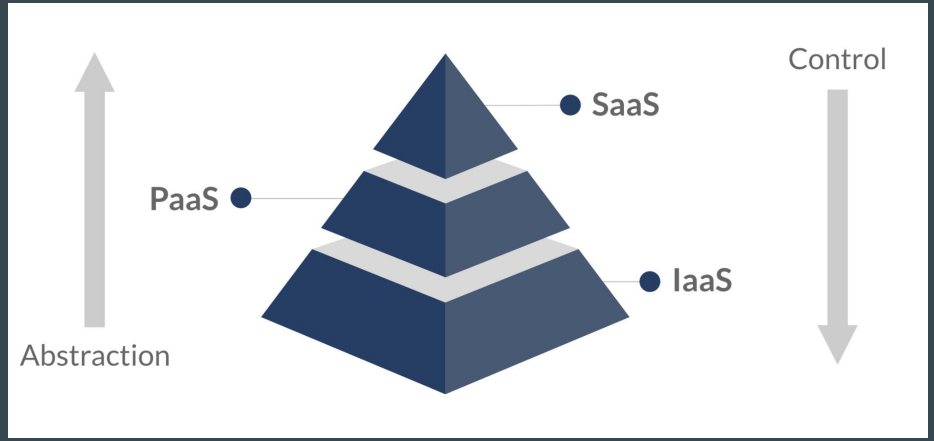
- Infrastructure as a Service
- Platform as a Service
- Software as a Service

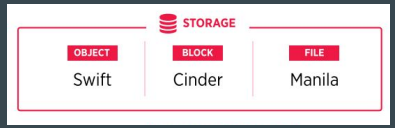


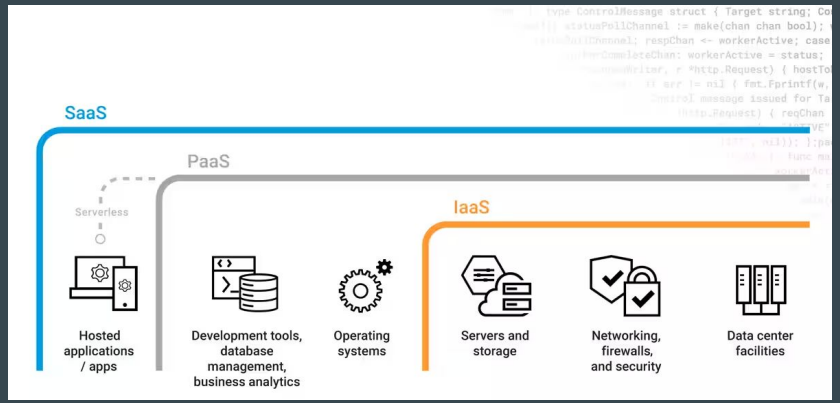


Servizi Cloud

- Infrastructure as a Service
- Platform as a Service
- Software as a Service
- Data as a Service

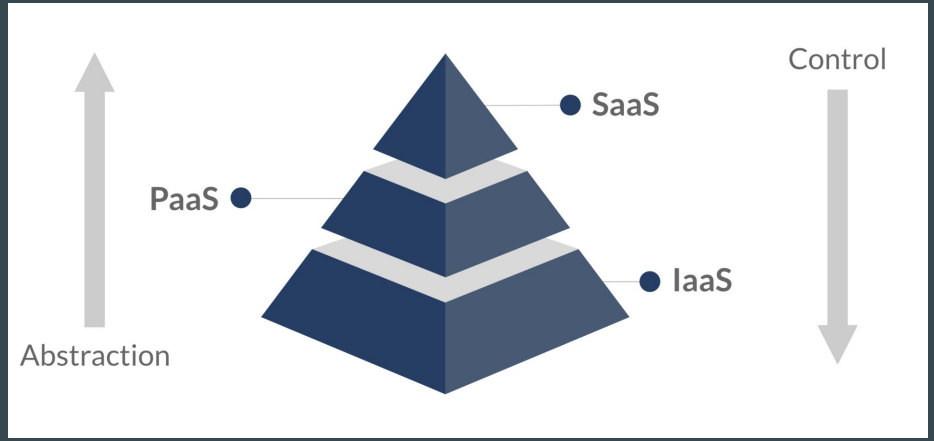


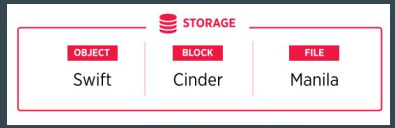


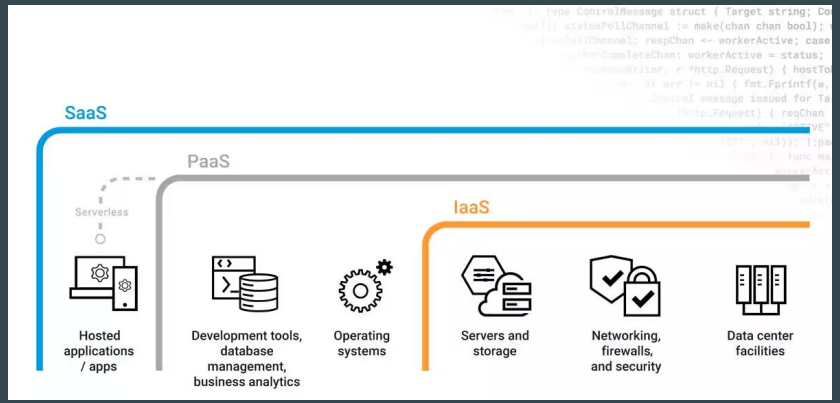


Servizi Cloud

- Infrastructure as a Service
- Platform as a Service
- Software as a Service
- Data as a Service







https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/

Data and Metadata

Data Definition

In computing, a data is a collection of facts, figures, or details that can be processed or analyzed. It often consists of numbers, text, or other types of information that are recorded and stored electronically.

Data serves as the foundation for creating information and insights through analysis and interpretation. It can be raw, unprocessed input or structured and organized to facilitate meaningful conclusions. In various contexts, data is used to make decisions, generate reports, or drive machine learning algorithms. Proper management and understanding of data are crucial for effective decision-making and problem-solving.

Metadata Definition

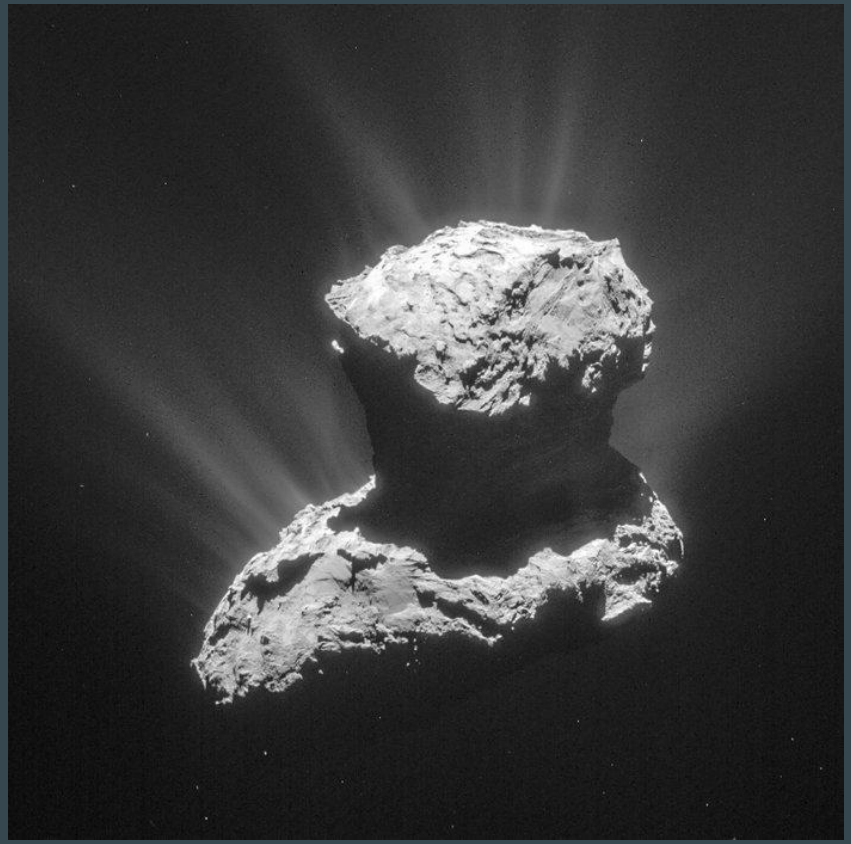
A metadata is data that provides information about other data. It describes various attributes of data, such as its origin, format, and relationships to other data, which helps in organizing, managing, and retrieving it efficiently.

Metadata can include details like the creator of a file, the date it was created, and how it should be used. It is essential for data cataloging and improving the accessibility and usability of information.

By offering context and structure, metadata enhances data searchability and interoperability across different systems and platforms.

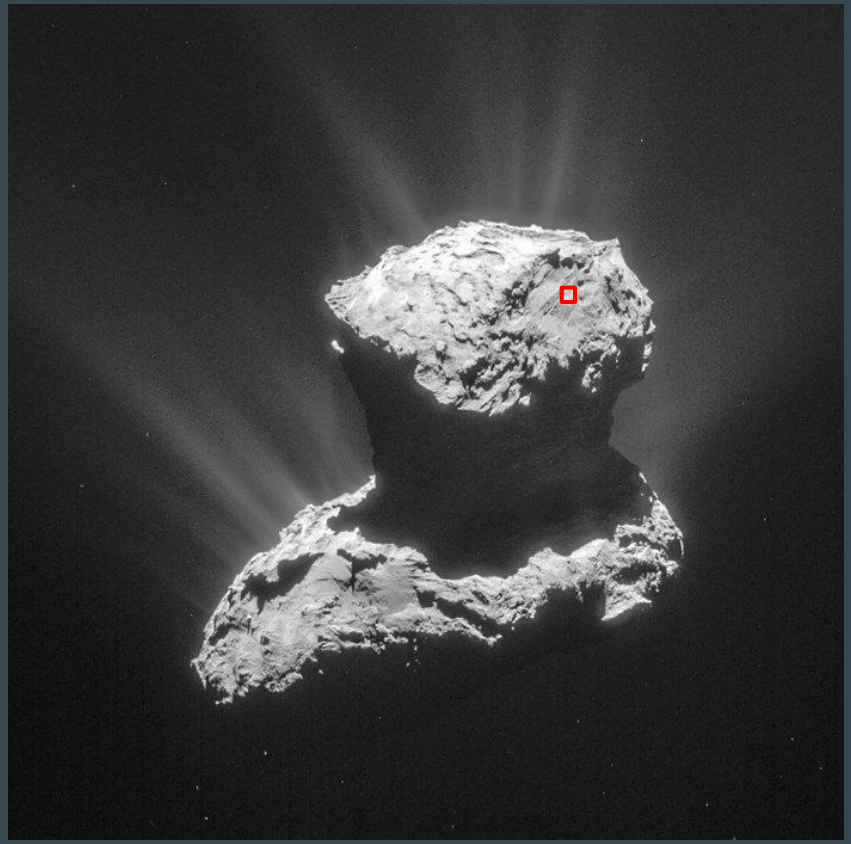
Data and Metadata Example

Planetological Example



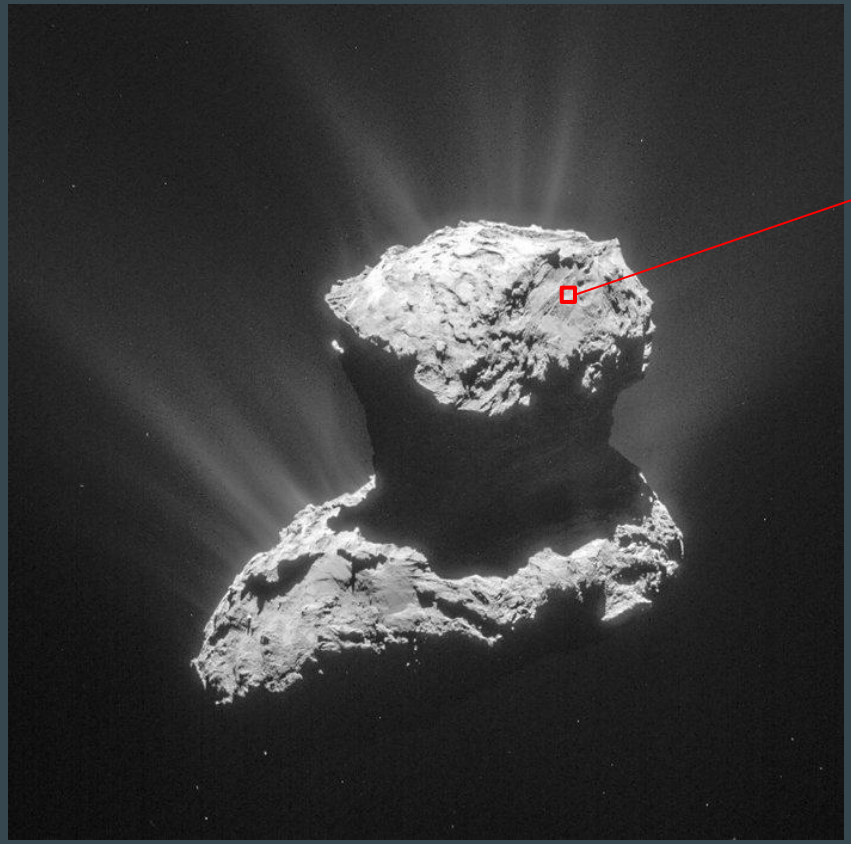
Which is the data?

Planetological Example



Which is the data?

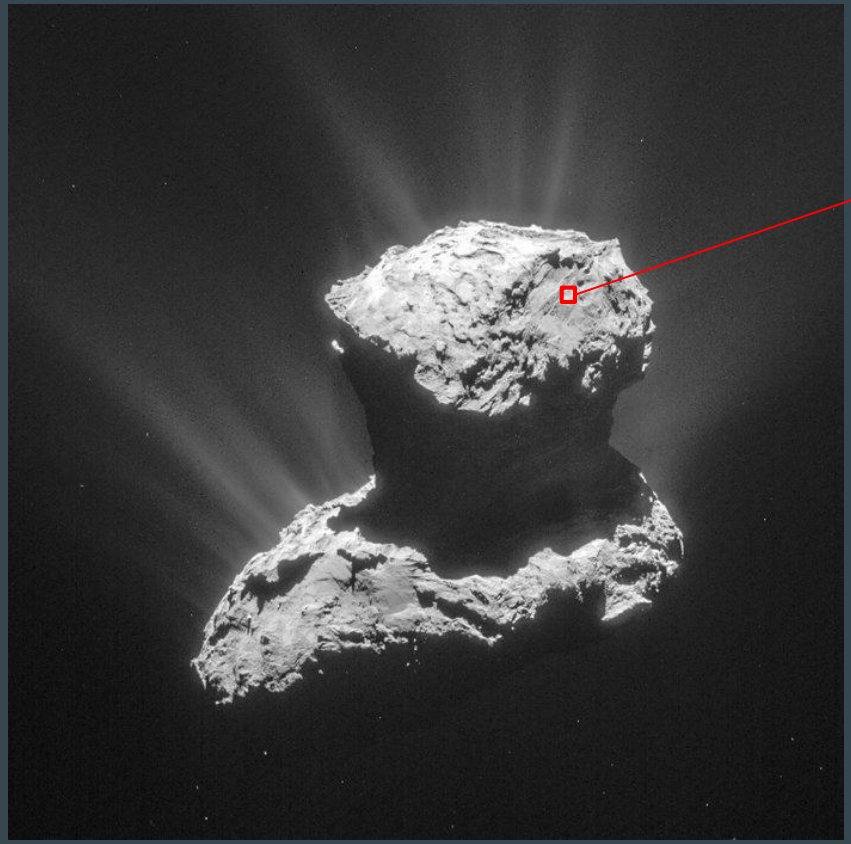
Planetological Example



Which is the data?

The pixel is represented as a 32-bit floating-point number.

Planetological Example

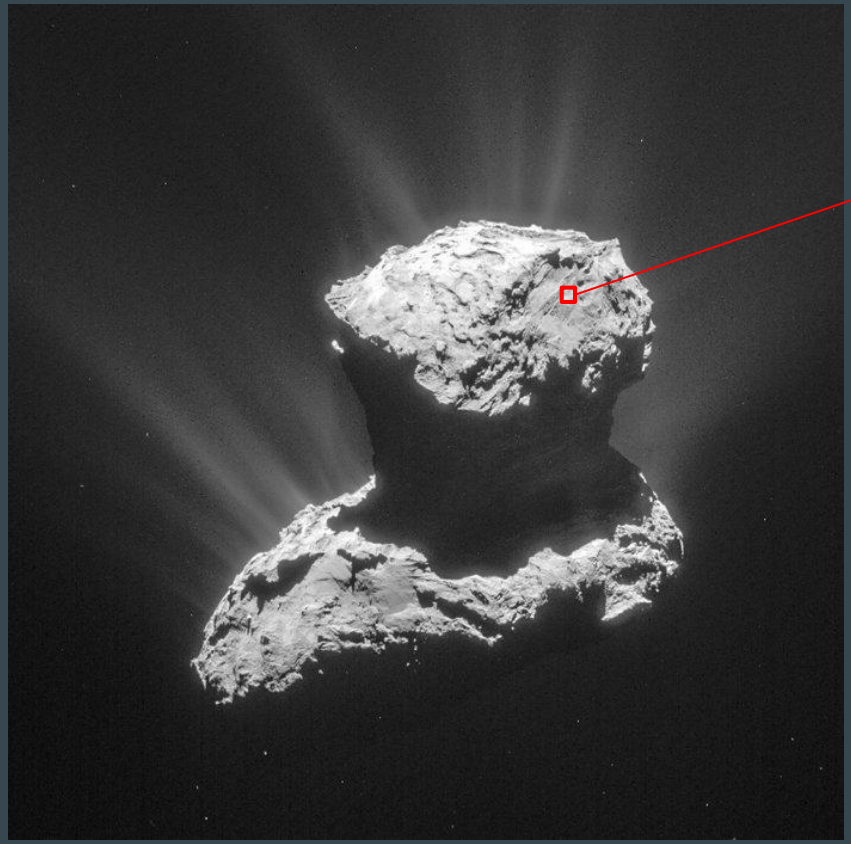


Which is the data?

The pixel is represented as a 32-bit floating-point number.

Does this have meaning?

Planetological Example



Which is the data?

The pixel is represented as a 32-bit floating-point number.

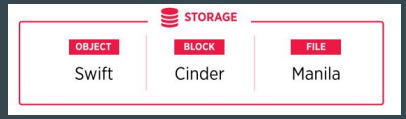
Does this have meaning?

The answer is **NO**.
It is necessary to know

- lighting,
- comet position,
- spacecraft position,
- exposure times,
- acquisition mode,
- pixel georeferencing.

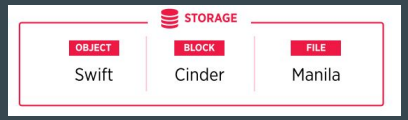
Data in the Cloud

Cloud Storage



https://www.ibm.com/topics/block-storage

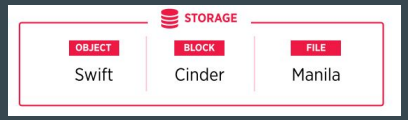
Cloud Storage



Block storage divides data into separate components made up of fixed-size data blocks, each with a unique identifier. Block storage allows the underlying storage system to retrieve it regardless of where it is stored.

https://www.ibm.com/topics/block-storage

Cloud Storage

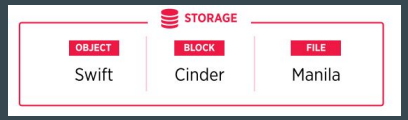


Block storage divides data into separate components made up of fixed-size data blocks, each with a unique identifier. Block storage allows the underlying storage system to retrieve it regardless of where it is stored.

File storage is the most well-known storage format: data is stored in files that can be interacted with, contained in folders within a hierarchical file directory.

https://www.ibm.com/topics/block-storage

Cloud Storage



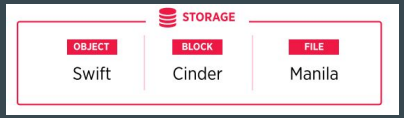
Block storage divides data into separate components made up of fixed-size data blocks, each with a unique identifier. Block storage allows the underlying storage system to retrieve it regardless of where it is stored.

File storage is the most well-known storage format: data is stored in files that can be interacted with, contained in folders within a hierarchical file directory.

Object storage is a storage format in which data is stored in separate units called objects. Each unit has a unique identifier, or key, that allows it to be located independently of where it is stored in a distributed system.

https://www.ibm.com/topics/block-storage

Cloud Storage



Block storage divides data into separate components made up of fixed-size data blocks, each with a unique identifier. Block storage allows the underlying storage system to retrieve it regardless of where it is stored.

File storage is the most well-known storage format: data is stored in files that can be interacted with, contained in folders within a hierarchical file directory.

Object storage is a storage format in which data is stored in separate units called objects. Each unit has a unique identifier, or key, that allows it to be located independently of where it is stored in a distributed system.

https://www.ibm.com/topics/block-storage

File Storage

- **Name**
- Path
- Type
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

Maxlength 255 characters

Unix and Unix Like

- All file names are "Case Sensitive." This means that vivek.txt, Vivek.txt, and VIVEK.txt are three different files.
- File names can use uppercase and lowercase letters as well as the symbols “.” (dot) and “_” (underscore).
- Other special characters like " " (blank space) can also be used, but they require a more complex handling (they must be quoted) and are generally discouraged.
- In practice, a file name can contain any character except “/” (root folder), which is reserved as a separator between files and folders in the pathname.
- The ***null*** character cannot be used.
- Using a “.” is not necessary but increases readability, especially if used to identify the file extension.
- The file name must be unique within a folder.
- A folder and a file with the same name cannot coexist within the same folder.

Unix and Unix Like

File Storage

- **Name**
- Path
- Type
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

Lunghezza massima 255 caratteri

Il set di nomi richiesto per specificare un particolare file in una gerarchia di folder è detto percorso del file o path.
percorso e nome del file formano il cosiddetto pathname.

Il percorso può essere assoluto o relativo:

- nel path assoluto si specifica tutto il percorso dall'inizio del disco (/,root):

  `/u/politi/projectb/plans/1dft`

- nel path relativo si può indicare il percorso a partire dal folder in cui ci si trova.

  `projectb/plans/1dft`

Un path relativo non può iniziare con /.
Simboli speciali:

. indica il folder corrente

.. indica il folder di livello superiore

Unix and Unix Like

File Storage

- Name
- **Path**
- Type
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

The set of names required to specify a particular file in a hierarchy of folders is called the file path.
The path and the filename together form what is known as the pathname.

The path can be absolute or relative:

- In an absolute path, the entire path is specified starting from the beginning of the disk (/, root):
  /u/politi/projectb/plans/1dft
- In a relative path, the path can be indicated starting from the folder in which you are located:
  projectb/plans/1dft

A relative path cannot start with /.
Special symbols:

. indicates the current folder
.. indicates the parent folder

Maxlength 1024 characters

Unix and Unix Like

File Storage

- Name
- Path
- Type
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

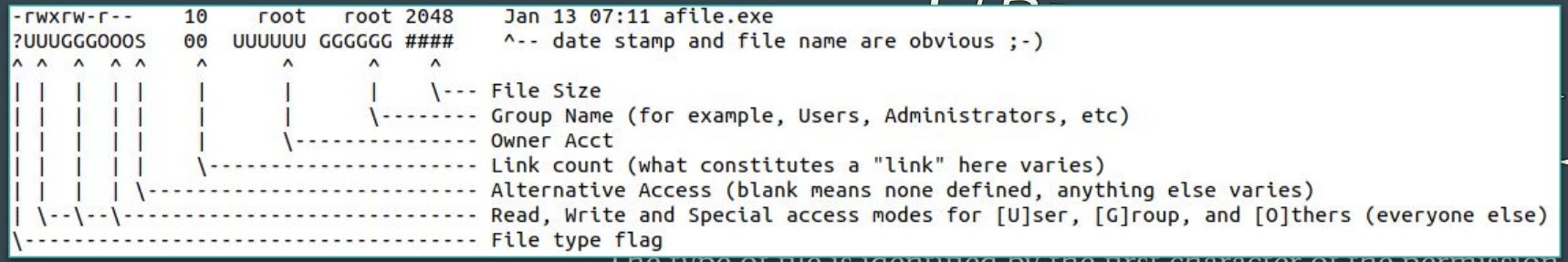
The type of file is identified by the first character of the permission string.

```
-rwxrwxrwx 1 romolo romolo       658 apr 30 09:56 manage.py
```

The types could be:

-	regular file
d	directory
l	symbolic link
c	Character file device
b	block device
s	local socket
p	named pipe

Uni...like



- **Type**
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

The type of file is identified by the first character of the permission string.

```
-rwxrwxrwx 1 romolo romolo       658 apr 30 09:56 manage.py
```

The types could be:

-	regular file
d	directory
l	symbolic link
c	Character file device
b	block device
s	local socket
p	named pipe

Unix-like

```
-rwxrw-r--    10    root   root 2048    Jan 13 07:11 afile.exe
?UUUGGGOOOS   00  UUUUUU GGGGGG ####    ^-- date stamp and file name are obvious ;-)
^ ^   ^   ^ ^     ^       ^      ^    ^
| |   |   | |     |       |      |    \--- File Size
| |   |   | |     |       |      \-------- Group Name (for example, Users, Administrators, etc)
| |   |   | |     |       \--------------- Owner Acct
| |   |   | |     \----------------------- Link count (what constitutes a "link" here varies)
| |   |   | \------------------------------ Alternative Access (blank means none defined, anything else varies)
| \--\--\-------------------------------- Read, Write and Special access modes for [U]ser, [G]roup, and [O]thers (everyone else)
\------------------------------------- File type flag
```

- **Type**
- Size
- Owner (UID, GID)
- Permission

The type of file is identified by the first character of the permission string.

```
-rwxrwxrwx 1 romolo romolo       658 apr 30 09:56 manage.py
```

The types could be:

	Character	Effect on files	Effect on directories
Read permission (first character)	`-`	The file cannot be read.	The directory's contents cannot be shown.
	`r`	The file can be read.	The directory's contents can be shown.
Write permission (second character)	`-`	The file cannot be modified.	The directory's contents cannot be modified.
	`w`	The file can be modified.	The directory's contents can be modified (create new files or folders; rename or delete existing files or folders); requires the execute permission to be also set, otherwise this permission has no effect.
Execute permission (third character)	`-`	The file cannot be executed.	The directory cannot be accessed with **cd**.
	`x`	The file can be executed.	The directory can be accessed with **cd**; this is the only permission bit that in practice can be considered to be "inherited" from the ancestor directories, in fact if *any* folder in the path does not have the `x` bit set, the final file or folder cannot be accessed either, regardless of its permissions; see **`path_resolution(7)`** for more information.
	`s`	The **setuid** bit when found in the **u**ser triad; the **setgid** bit when found in the **g**roup triad; it is not found in the **o**thers triad; it also implies that `x` is set.	
	`S`	Same as `s`, but `x` is not set; rare on regular files, and useless on folders.	
	`t`	The **sticky** bit; it can only be found in the **o**thers triad; it also implies that `x` is set.	
	`T`	Same as `t`, but `x` is not set; rare on regular files, and useless on folders.	

https://it.wikipedia.org/wiki/Permessi_(informatica)

Unix and Unix Like

File Storage

- Name
- Path
- **Type**
- Size
- Owner (UID, GID)
- Permission
- Timestamps
  - creation
  - modify

The type of file is identified by the first character of the permission string.

```
-rwxrwxrwx 1 romolo romolo       658 apr 30 09:56 manage.py
```

The types could be:

-	regular file
d	directory
l	symbolic link
c	Character file device
b	block device
s	local socket
p	named pipe

Object Storage

In object storage, data is broken down into discrete units called objects and stored in a single repository (Bucket) instead of as files within folders or as blocks on servers.

The volumes of object storage function as modular units: each one is an independent repository that contains the data, a unique identifier that allows an object to be located in a distributed system, and the metadata that describes the data.

Metadata is important and includes details such as age, privacy/security, and access restrictions.

Object Storage

In object storage, data is broken down into discrete units called objects and stored in a single repository (Bucket) instead of as files within folders or as blocks on servers.

The volum pendent repository e located in a distrib

> Object storage **metadata** can be extremely detailed and capable of storing information about where a video was filmed, the type of camera used, and the actors appearing in each frame.

Metadata is important and includes details such as age, privacy/security, and access restrictions.

The Data Preservation

In data management, Data Preservation is the act of safeguarding and maintaining both the security and integrity of data. Preservation is achieved through formal activities governed by policies, regulations, and strategies designed to protect and prolong the existence and authenticity of data and its associated metadata.

The Data Preservation

In data management, Data Preservation is the act of safeguarding and maintaining both the security and integrity of data. Preservation is achieved through formal activities governed by policies, regulations, and strategies designed to protect and prolong the existence and authenticity of data and its associated metadata.

- short-term
- medium-term
- long-term

The Data Preservation

In data management, Data Preservation is the act of safeguarding and maintaining both the security and integrity of data. Preservation is achieved through formal activities governed by policies, regulations, and strategies designed to protect and prolong the existence and authenticity of data and its associated metadata.

- short-term
- ~~medium-term~~
- long-term

The Data Preservation

In data management, Data Preservation is the act of safeguarding and maintaining both the security and integrity of data. Preservation is achieved through formal activities governed by policies, regulations, and strategies designed to protect and prolong the existence and authenticity of data and its associated metadata.

- **short-term**
- ~~medium-term~~
- long-term

Short-term Preservation.
Access to digital materials for a defined period during which use is expected, but which does not extend beyond the foreseeable future and/or until it becomes inaccessible due to technological changes.

The Data Preservation

In data management, Data Preservation is the act of safeguarding and maintaining both the security and integrity of data. Preservation is achieved through formal activities governed by policies, regulations, and strategies designed to protect and prolong the existence and authenticity of data and its associated metadata.

- short-term
- ~~medium-term~~
- **long-term**

Long-term Preservation
Continuous access to digital materials, or at least to the information contained in them, indefinitely.

Version Control

Git

Git is a distributed version control software that can be used via the command line interface, created by Linus Torvalds in 2005.

A **Distributed Version Control System** (DVCS) is a type of version control that allows tracking changes and versions made to software source code without needing to use a central server, as in traditional cases.

With this system, developers can collaborate individually and in parallel, working on their own development branch, recording their changes (commits), and later sharing or merging them with others' changes—all without the need for a centralized server. This system enables various modes of collaboration, as the server is merely a support tool.

https://github.com/torvalds/linux

Glossary

repository: It is a "folder" that contains all the files needed for your project, including the files that track all versions of the project.

clone: It is the local version of the repository.

remote: It is the remote version of the repository that can be modified by anyone with access to the repository.

branch: "Branches" are used in Git to implement isolated features, that is, developed independently from each other but starting from the same root.

fork: A copy of a repository belonging to another user.

commit: A snapshot of the local repository compressed with SHA, ready to be transferred from the clone to the remote or vice versa.

tag: A marker used to highlight specific commits.

First Steps

Git can be downloaded from https://git-scm.com/downloads (all Linux distributions have Git among the available packages).

Once the software is installed, to "copy" a repository locally, you simply use the clone command.

For example, for the course repository:

git clone https://github.com/RomoloPoliti-INAF/PhDCourse2024.git

First steps

Git can be downloaded from https://git-scm.com/downloads (all Linux distributions have Git among the available packages).

Once the software is installed, to "copy" a repository locally, you simply use the clone command.

For example, for the course repository:

git clone https://github.com/RomoloPoliti-INAF/PhDCourse2024.git

Windows users need to install the Git application by downloading it from the GitHub website.
After installation, it will be necessary to restart the machine.

Fundamental Git commands

clone: Create a local copy of a remote repository

pull: Update the local copy of the repository

add: Add one or more files to the list of contents in the local repository

commit: Record changes to the repository

push: Update the remote repository

Fundamental Git commands

clone: Create a local copy of a remote repository

pull: Update the local copy of the repository

add: Add one or more files to the list of contents in the local repository

commit: Record changes to the repository

push: Update the remote repository

Examples of using git can be found in the '*Examples/01 - git*' folder of the course repository.

Course Overview

Cloud
- ~~Cloud structure~~
- ~~Data in the Cloud~~
- ~~Cloud Computing~~

Data
- ~~Data and Metadata~~
- ~~Archives~~
- Relational and not-relational Database

Computing
- Retrieval
- Manipulation
- Visualization

Environment
- Virtualization and Containers
- Microservices
- DevOps

Coding
- Fundamentals of Coding
- Python
- Versioning and Documentation

Relational Database

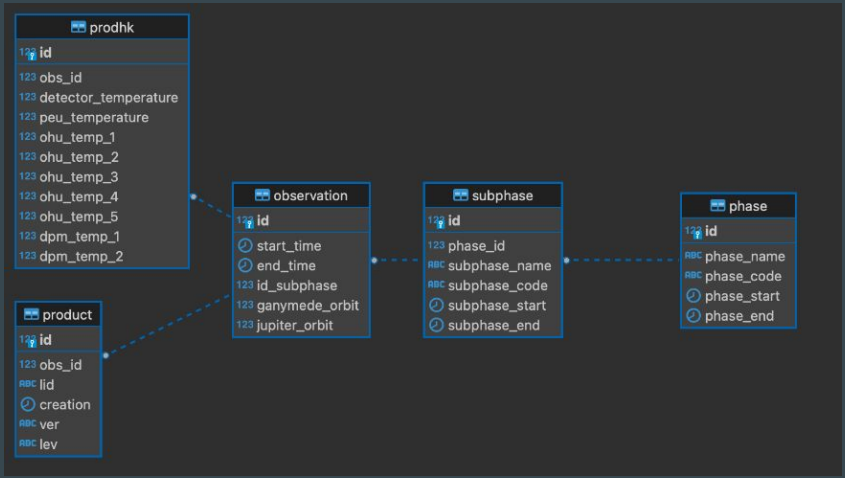
Introduction

The term "relational database management system" (**RDBMS**) refers to a database management system based on the relational model introduced by Edgar F. Codd.

In addition to these, although less commercially widespread, there are other database management systems that implement data models alternative to the relational one: hierarchical, network, and object-oriented models.

Among the various relational and object-oriented databases, PostgreSQL is the most widely used.

Entity Relationship Diagram



IDEF1X model

Entity Relationship Diagram - Glossary

Schema	A group of entities with their relationships.
Entity	They represent classes of objects (facts, things, people, ...) that have common properties and autonomous existence for the purposes of the application of interest. An occurrence of an entity is an object or instance of the class that the entity represents. This is not about the value that identifies the object, but about the object itself. An interesting consequence of this is that an occurrence of an entity has an existence independent of the properties associated with it. In a schema, each entity has a name that uniquely identifies it and is graphically represented by a rectangle with the entity's name inside it.
Relationship	They represent a relationship between two or more entities. The number of entities linked is indicated by the degree of the association: a good E-R schema is characterized by a prevalence of associations with a degree of two.
Tupla	A series of attributes that describe the entities. All objects of the same entity class have the same attributes: this is what is meant when talking about similar objects.
Attribute	A characteristic of the entity.

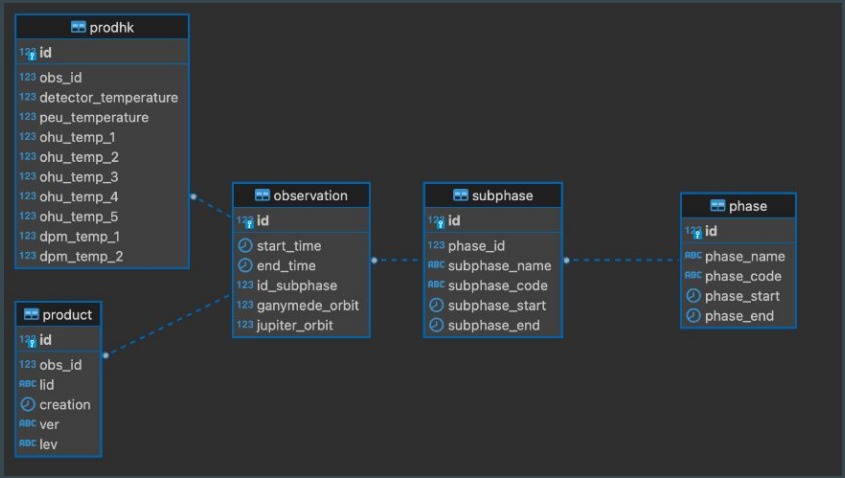
Entity Relationship Diagram - Glossary

The choice of attributes reflects the level of detail with which we want to represent the information of individual entities and relationships.

For each entity or association class, a key is defined.

The key is a minimal set of attributes that uniquely identifies an entity instance.

Entity Relationship Diagram



UML (Unified Modeling Language)

IDEF1X model

SQL Language

To give some examples of SQL language, we will use SQLite.

You can find a frontend here: https://sqlitebrowser.org

SQL Language

To give some examples of SQL language, we will use SQLite.

You can find a frontend here: https://sqlitebrowser.org

Schema	Entity	Tupla	Attribute	Relationship

Database	Table	Record	Field	Foreign Key

SQL - Basic Commands

CREATE	Create a database or a table
INSERT	Create one or more records in a table
DROP	Delete a database or a table
DELETE	delete one or more records
ALTER	Modify a database or a table
UPDATE	Modify a record
SELECT	Select a series of records.

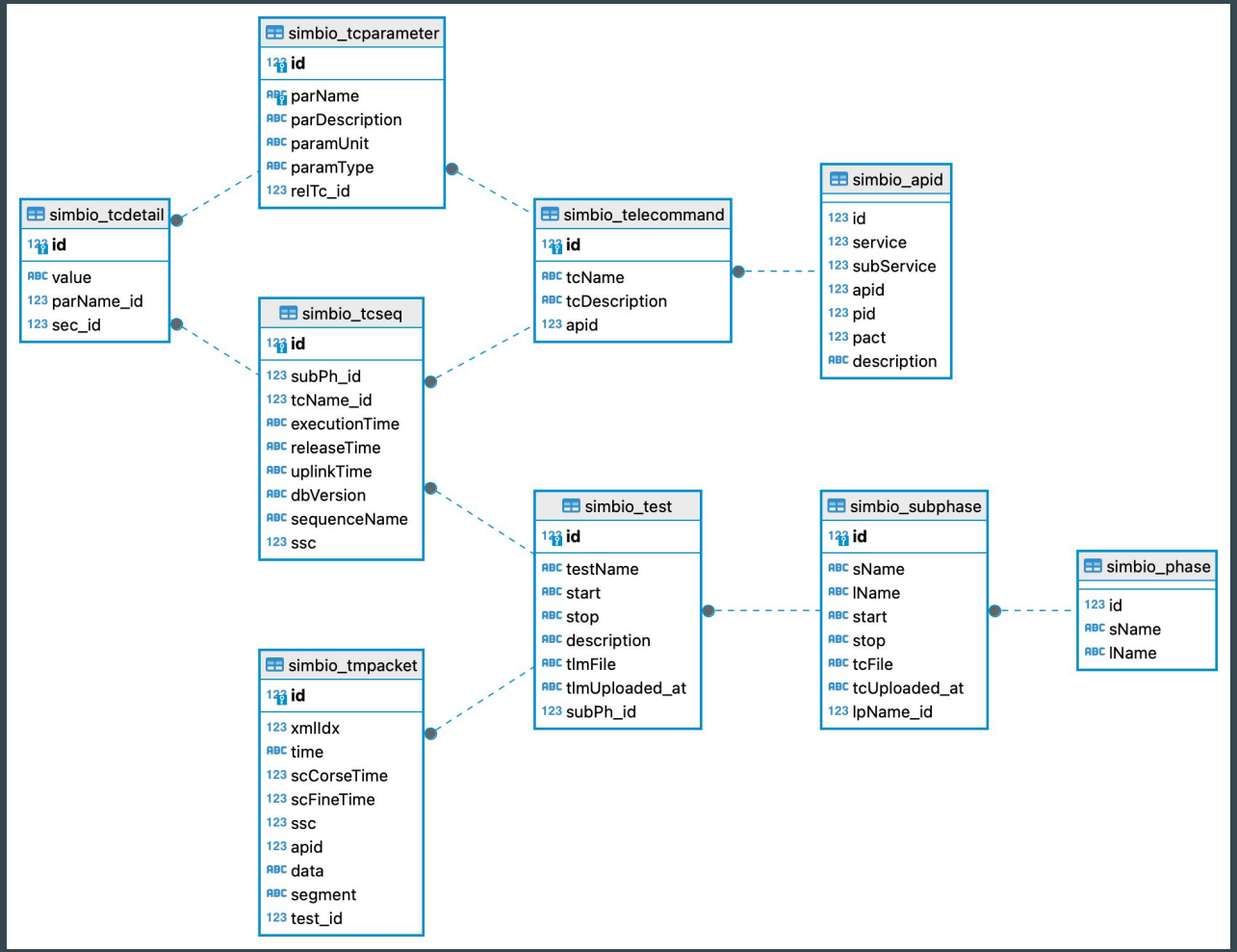
SQL - Basic Commands

CREATE	Create a database or a table
INSERT	Create one or more records in a table
DROP	Delete a database or a table
DELETE	delete one or more records
ALTER	Modify a database or a table
UPDATE	Modify a record
SELECT	Select a series of records.

Query

Examples

The examples use the database example.sqlite found in the *Examples/03 - sqlLite* folder of the repository. In the same folder is the file Script.sql with all the examples. On the side, you'll find the ER diagram of the database.



Select

```
SQL> SELECT table.field FROM table;
```

Select

```
SQL> SELECT tabella.campo FROM tabella;
```

Example 1: I want the list of all the tests (*simbio_test* table) present in my DB.

```
SQL> SELECT * FROM simbio_test;
```

Select

```
SQL> SELECT tabella.campo FROM tabella;
```

Example 1: I want the list of all the tests (*simbio_test* table) present in my DB.

```
SQL> SELECT * FROM simbio_test;
```

Example 2: From the previous list, I only want the name and start and end times.

```
SQL> SELECT st.testName, st.start, st.stop FROM simbio_test st;
```

Select

```
SQL> SELECT tabella.campo FROM tabella;
```

Example 1: I want the list of all the tests (*simbio_test* table) present in my DB.

```
SQL> SELECT * FROM simbio_test;
```

Example 2: From the previous list, I only want the name and start and end times.

```
SQL> SELECT st.testName, st.start, st.stop FROM simbio_test st;
```

Table alias. Equivalent to:
simbio_test AS *st*

Select

```
SQL> SELECT tabella.campo FROM tabella;
```

Example 1: I want the list of all the tests (*simbio_test* table) present in my DB.

```
SQL> SELECT * FROM simbio_test;
```

Example 2: From the previous list, I only want the name and start and end times.

```
SQL> SELECT test_name, start, stop FROM simbio_test;
```

Example 3: the same info as in example 2 but only those performed on 11/12/2018.

```
SQL> SELECT test_name, start, stop FROM simbio_test WHERE start > "2018-12-11
     00:00:00" AND stop < "2018-12-11 23:59:59";
```

Select

Example 4: I want all the fields of the sub-phases of the "CRUISE" phase ordered chronologically (knowing that phase and sub-phase are linked through an ID)

```
SQL> SELECT sp.* FROM simbio_subphase sp, simbio_phase p WHERE p.id = sp.lpName_id
        AND p.sName = "CRUISE" ORDER BY sp .start;
```

Exercise

Select the first VIHI science telemetry performed on 11/12/2018 and provide the time it was executed and the integration time.

Exercise - Solution

Select the first VIHI science telemetry performed on 11/12/2018 and provide the time it was executed and the integration time.

```
SQL>SELECT id FROM simbio_telecommand tc WHERE tc.tcDescription LIKE "%VIHI
      science%" LIMIT 1;

[OUT] 306

SQL> SELECT executionTime, id FROM simbio_tcseq  WHERE
      simbio_tcseq.tcName_id=306 AND executionTime > "2018-12-11" AND
      executionTime < "2018-12-12";

[OUT] 2018-12-11 15:54:37.657847+01; 9784
```

Exercise - Solution

Select the first VIHI science telemetry performed on 11/12/2018 and provide the time it was executed and the integration time.

```
[OUT] 2018-12-11 15:54:37.657847+01; 9784

SQL> SELECT id FROM simbio_tcparameter WHERE parDescription LIKE "%VIHI
       integration%";

[OUT] 578

SQL> SELECT value FROM simbio_tcdetail WHERE sec_id=9784 AND parName_id=578;

[OUT] 3
```

Exercise - More elegant solution

Select the first VIHI science telemetry performed on 11/12/2018 and provide the time it was executed and the integration time.

```
SQL>SELECT tseq.executionTime, simbio_tcdetail.value FROM simbio_tcseq AS tseq JOIN
    simbio_tcdetail ON tseq.id = simbio_tcdetail.sec_id WHERE tseq.tcName_id =
    (SELECT stc.id FROM simbio_telecommand stc WHERE stc.tcDescription LIKE
    "%VIHI%" AND stc.tcDescription LIKE "%science%") AND tseq.executionTime >
    "2018-12-11" AND tseq.executionTime < "2018-12-12" AND
    simbio_tcdetail.parName_id = (SELECT tcp.id FROM simbio_tcparameter AS tcp
    WHERE tcp.parDescription LIKE "%VIHI%" AND tcp.parDescription LIKE
    "%integration%") ORDER BY tseq.executionTime LIMIT 1
```

eXtensible Markup Language - XML

What is I'XML

XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.

What is I'XML

XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.

In logic and the theory of formal languages, a **metalanguage** is understood as a formally defined language whose purpose is to define other artificial languages, known as target languages or object languages (in the context of SGML and XML, the term applications is also used). Such a definition tends to be formally rigorous and complete, so it can be used for the construction or validation of computer tools that support the target languages.

What is l'XML

XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.

```
<?xml version="1.0" encoding="UTF-8"?>
<users>
	<user years="20">
		<name>Ema</name>
		<surname>Princi</surname>
		<address>Torino</address>
	</user>
	<user years="54">
		<name>Max</name>
		<surname>Rossi</surname>
		<address>Roma</address>
	</user>
</users>
```

What is l'XML

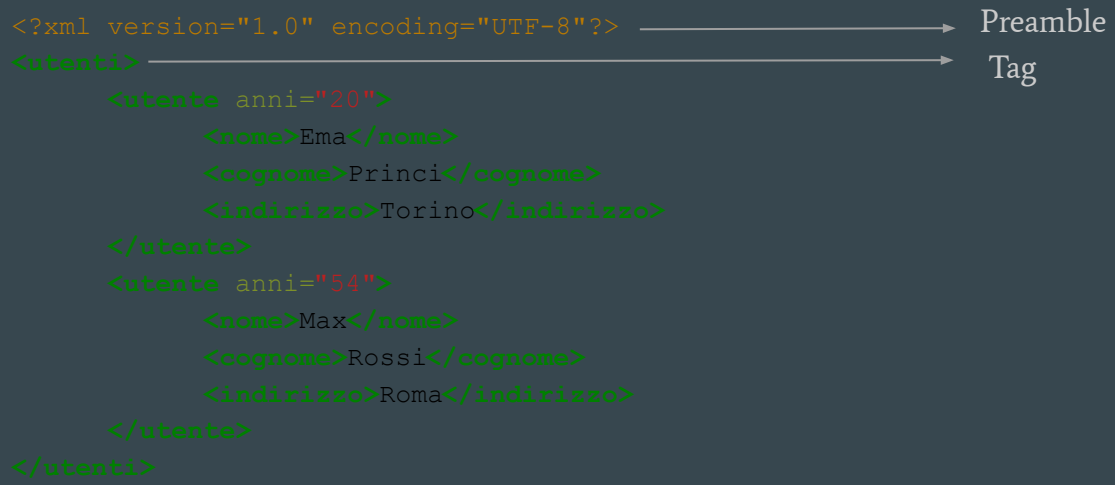
XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.

```
<?xml version="1.0" encoding="UTF-8"?>
<utenti>
      <utente anni="20">
            <nome>Ema</nome>
            <cognome>Princi</cognome>
            <indirizzo>Torino</indirizzo>
      </utente>
      <utente anni="54">
            <nome>Max</nome>
            <cognome>Rossi</cognome>
            <indirizzo>Roma</indirizzo>
      </utente>
</utenti>
```

Preamble

What is l'XML

XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.



What is l'XML

XML (short for eXtensible Markup Language) is a metalanguage for defining markup languages, meaning a language based on a syntactic mechanism that allows for defining and controlling the meaning of elements contained in a document or text.

```
<?xml version="1.0" encoding="UTF-8"?>
<utenti>
      <utente anni="20">
            <nome>Ema</nome>
            <cognome>Princi</cognome>
            <indirizzo>Torino</indirizzo>
      </utente>
      <utente anni="54">
            <nome>Max</nome>
            <cognome>Rossi</cognome>
            <indirizzo>Roma</indirizzo>
      </utente>
</utenti>
```

Preamble

Tag

Element