

Administracio de Sistemes Gestors de Bases de Dades - Sebastian Ciscar

None

None

Copyright - Sebastian Ciscar - Professor Departament d'Informàtica IES Jaume II el Just (Tavernes de la Valldigna)

Table of contents

1. Presentació del mòdul ASGBD	4
2. Continguts del mòdul. Unitats didàctiques	5
2.1 Material Didàctic	7
2.2 Programari	8
2.3 Metodologia	9
3. 1 Amazon Web Services (AWS)	11
3.1 1.1 Què és AWS Academy?	11
3.2 1.2 Com accedisc a AWS Academy?	11
3.3 1.3 Què és AWS Academy Learner Lab?	11
3.4 1.4 Com puc utilitzar AWS Academy Learner Lab?	11
3.5 1.5 Serveis d'Amazon Web Services (AWS)	11
3.6 1.6 Com crear instàncies EC2?	12
3.7 1.7 Com connectar-nos a una instància EC2 per SSH des de Linux	24
3.8 1.7.3 Pas 3. Descarreguem la clau privada .PEM des del Learner Lab	26
3.9 1.7.4 Pas 4. Connectar-nos per SSH des de Linux	27
4. UD1	28
4.1 Instalación MySQL Server	28
4.2 Objectiu de la pràctica	40
4.3 Descripció de la pràctica	40
4.4 Tasques a realitzar	41
4.5 Documentació del procés:	42
4.6 Materials necessaris	42
4.7 Lliurament	42
4.8 Avaluació	42
4.9 Orientacions per a l'alumne	43
5. UD2	46
5.1 Instalación MongoDB	46
5.2 Imagen de MongoDB en docker	47
5.3 Probar la conexión a MongoDB	48
5.4 Asegurar MongoDB	48
5.5 NoSql: MongoDB	49
5.6 1.Introducció a NoSql.	49
5.7 2. MongoDB	56
5.8 3. Modelat Mongo	57
5.9 3. Disseny Físic. DDL Mongo	61

6. Arxius	72
6.1 Arxius del mòdul	72

1. Presentació del mòdul ASGBD



El mòdul d'Administració de Sistemes Gestors de Bases de Dades (ASGBD) pertany al 2on curs del CFGS d'Administració de Sistemes Informàtics en Xarxa. Aquest mòdul està dissenyat per dotar els alumnes de les habilitats necessàries per a l'administració, configuració i manteniment de bases de dades, així com per garantir la seguretat i integritat de la informació.

2. Continguts del mòdul. Unitats didàctiques



• UD 1. Instal·lació de MySQL Server (3 hores)

Aprendràs a instal·lar i configurar un servidor MySQL, crear bases de dades i gestionar taules, així com una introducció als procediments emmagatzemats.

• UD 2. MongoDB (3 hores)

Coneixeràs les bases de dades NoSQL amb MongoDB i la seva aplicació en la gestió d'informació no estructurada. També integrarem MongoDB en aplicacions web.

• UD 3. AWS i Bases de Dades en el Núvol (3 hores)

Introducció als serveis de bases de dades en el núvol amb AWS, configuració d'AWS RDS, DynamoDB i S3, i pràctiques amb aquests serveis.

• UD 4. Administració del Servidor MySQL (9 hores)

Configuració avançada del servidor MySQL, gestió de variables del sistema, estadístiques, i la implementació de procediments emmagatzemats i funcions.

• UD 5. Usuaris, Permisos i Rols (9 hores)

Creació i gestió d'usuaris a MySQL, assignació de permisos i rols, i la implementació de mesures de seguretat per protegir les dades.

• UD 6. Còpies de Seguretat (9 hores)

Coneixeràs els diferents tipus de còpies de seguretat, estratègies efectives per fer còpies i la restauració de dades en MySQL.

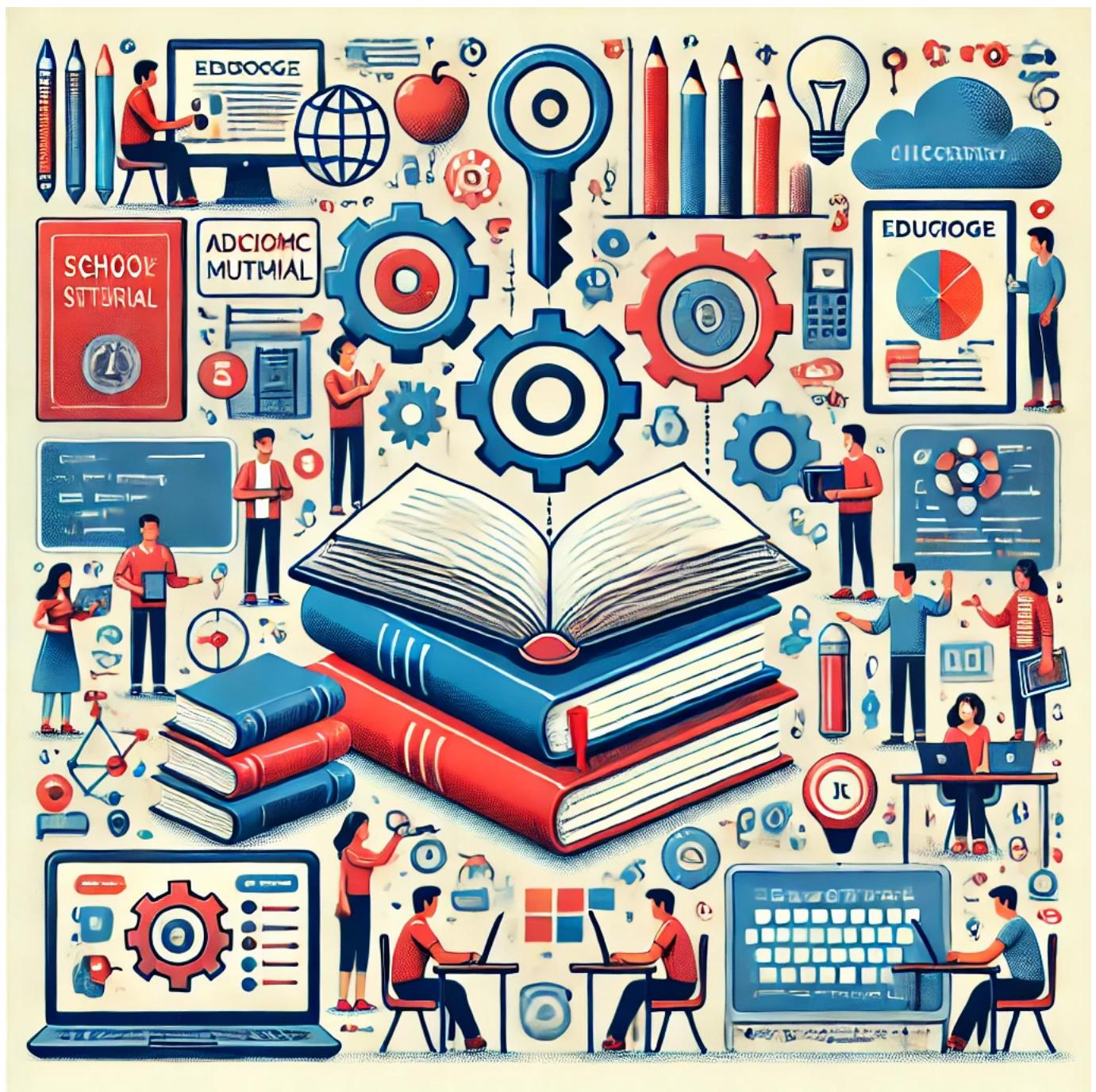
• UD 7. Rèpliques en Bases de Dades (9 hores)

Introducció a les rèpliques en bases de dades MySQL, configuració de servidors mestre-rèplica, i pràctiques de rèpliques i recuperació de dades.

• UD 8. Procediments Emmagatzemats (9 hores)

Creació i gestió de procediments emmagatzemats a MySQL, amb aplicacions pràctiques i implementació en casos reals.

2.1 Material Didàctic



- Apunts proporcionats pel professor.
- Textos d'ampliació i enllaços a articles i documentació oficial relacionats amb cada unitat.
- Pràctiques i exercicis resolts per reforçar el que s'ha exposat als apunts.

Tot aquest material s'oferirà a través de l'aula virtual durant el desenvolupament de cada unitat.

A més, dins l'aula virtual, disposarem d'un fòrum general per comentar aspectes globals del mòdul, i un fòrum per cada unitat didàctica, per tal deresoldre dubtes i tractar aspectes relacionats amb la unitat.

2.2 Programari

El programari a utilitzar serà principalment lliure, i es donaran instruccions en cada unitat per a la seu descàrrega i instal·lació:

- **Kubuntu 24.04** com a sistema operatiu de base utilitzat a l'aula. No obstant això, l'alumne pot utilitzar qualsevol altre sistema operatiu, ja que tot el programari amb què treballarem és multiplataforma.
-

- **MySQL Server**, com a sistema gestor de bases de dades relacional.
- **MongoDB**, com a sistema gestor de bases de dades NoSQL.
- **AWS**, com a entorn per desplegar maquines virtuals i bases de dades en el núvol.
- **Docker**, com a contenidor d'aplicacions.

2.2.1 Eines web i col·laboratives

A banda del programari esmentat anteriorment, també s'utilitzaran el següent portals web i plataformes de treball col·laboratiu:

- **Portals Aules**: Com a aula virtual, i que articularà el funcionament del mòdul. Serà aici on s'ubiquen els diferents recursos, fòrums, etc.
- **MS Teams**: El nostre centre és *Centre Digital Col·laboratiu*, pel que cada alumne i professor disposa d'una identitat digital que li proporciona accés a tota la suite d'enies d'Office 365. Entre aquestes, la que utilitzarem de forma més freqüent serà *MS Teams*, a través de la qual s'organitzaran les diferents tutoríes .
- **Github**: En alguns projectes en grup, serà de gran ajuda treballar amb sistemes de control de versions distribuïts, com *Github* o *Gitlab*, de manera que puguen realitzar desenvolupaments de forma col·laborativa.

2.3 Metodologia



Per a cada unitat disposareu de la teoria, projectes i exemples pràctics, i d'orientacions per tal de realitzar les activitats pràctiques que se us plantegen, així com enllaços a documentació addicional i complementària per a les activitats d'ampliació.

En línies generals, la manera de treballar en cada unitat serà la següent:

- En iniciar cada unitat didàctica es realitzarà una presentació inicial d'aquesta, dels conceptes bàsics, i de què seran capaços de fer en finalitzar-la.

- Disposareu del material per a la seu lectura comprensiva i estudi
 - També disposareu de documentació addicional per ampliar o puntualitzar allò exposat a classe.
 - Se us proporcionaran exercicis i pràctiques guiades que accompanyen la teoria de la unitat i que us ajudaran a entendre els conceptes de la unitat.
 - També es realitzaran activitats i pràctiques de consolidació i reforç
-

3. 1 Amazon Web Services (AWS)

Amazon Web Services (AWS) és una col·lecció de serveis de computació en el núvol pública que, en conjunt, formen una plataforma de computació en el núvol, oferida per Amazon a través d'Internet.

3.1 1.1 Què és AWS Academy?

AWS Academy és un programa d'AWS que ofereix gratuïtament a les institucions d'educació superior un pla d'estudis de computació en el núvol, preparant els estudiants per obtenir certificacions reconegudes en la indústria.

Referències: Preguntes freqüents sobre AWS Academy.

3.2 1.2 Com accedisc a AWS Academy?

L'URL per accedir a AWS Academy és la següent:

https://www.awsacademy.com/LMS_Login

3.3 1.3 Què és AWS Academy Learner Lab?

AWS Academy Learner Lab és un laboratori que permet als estudiants treballar amb alguns dels serveis d'AWS durant un període de temps. Cada estudiant disposa d'un crèdit de 100 dòlars per utilitzar en serveis de la plataforma AWS.

3.4 1.4 Com puc utilitzar AWS Academy Learner Lab?

En aquesta guia trobaràs tots els passos necessaris per accedir i utilitzar la plataforma AWS Academy Learner Lab.

Guia de l'estudiant d'AWS Academy Learner Lab. (pdf)

3.5 1.5 Serveis d'Amazon Web Services (AWS)

A continuació es mostren alguns dels serveis d'AWS que utilitzarem durant el curs.

- **Amazon EC2 (Elastic Compute Cloud):** Servei web que proporciona capacitat de càlcul escalable en el núvol.
- **Amazon S3 (Simple Storage Service):** Servei d'emmagatzematge d'objectes.
- **Amazon EBS (Elastic Block Store):** Servei d'emmagatzematge de blocs.
- **Amazon EFS (Elastic File System):** Servei d'emmagatzematge de fitxers en xarxa, compatible amb NFS.
- **Amazon RDS (Relational Database Service):** Servei de bases de dades gestionat per AWS.
- **Amazon VPC (Virtual Private Cloud):** Servei per crear xarxes virtuals en AWS.
- **AWS IAM (Identity and Access Management):** Servei per administrar usuaris i permisos d'accés als recursos d'AWS.
- **AWS KMS (Key Management Service):** Servei per crear i administrar claus de xifratge.
- **Amazon Lambda:** Servei basat en esdeveniments per executar codi sense gestionar servidors.
- **Amazon Elastic Beanstalk:** Servei per desplegar i escalar aplicacions web sense servidors.
- **Amazon ECS (Elastic Container Service):** Servei d'orquestració de contenidors.
- **Amazon EKS (Elastic Kubernetes Service):** Servei per executar Kubernetes en AWS.
- **AWS Fargate:** Servei per executar contenidors sense gestionar servidors.
- **Amazon Lightsail:** Solució VPS senzilla per a desenvolupadors i petites empreses.

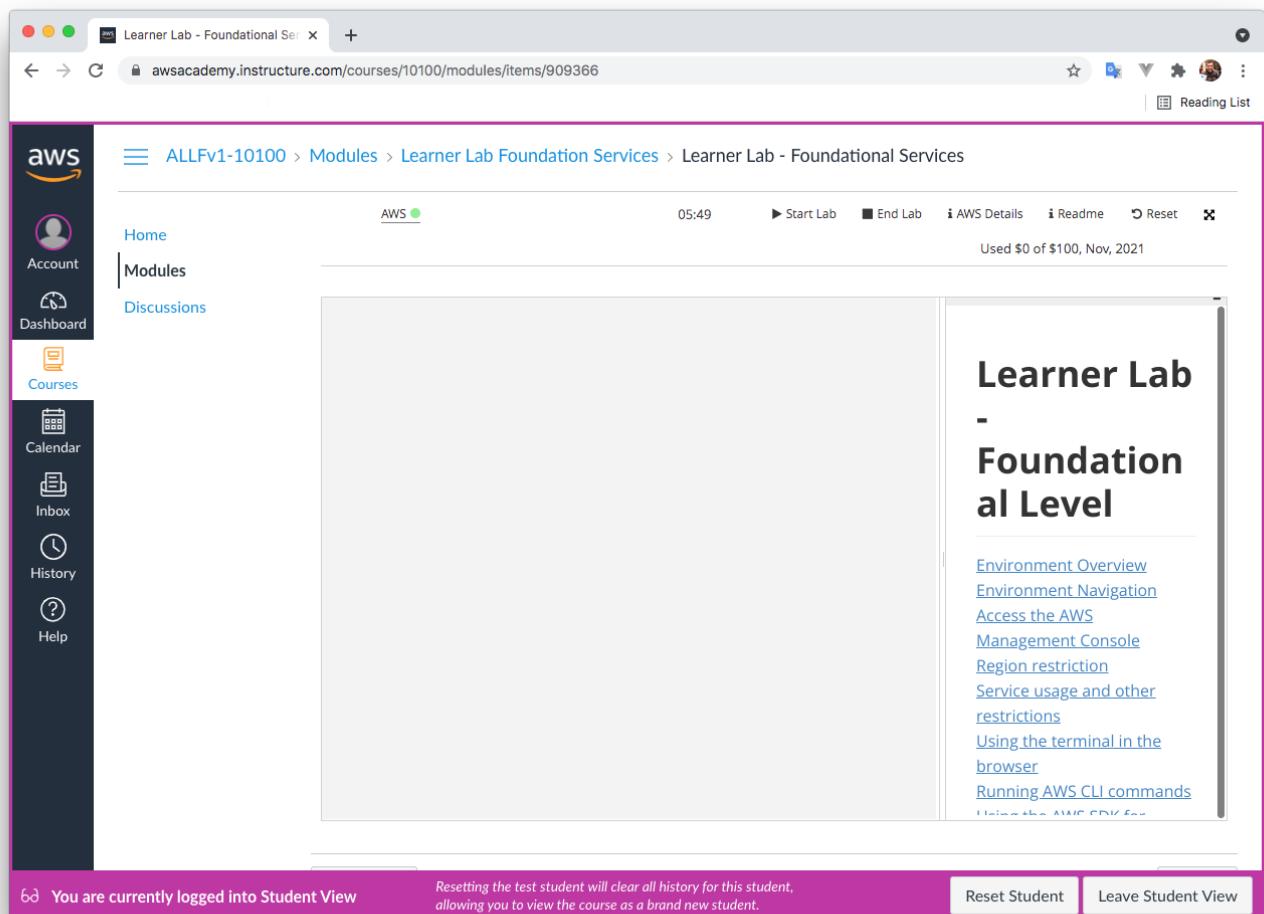
3.6 1.6 Com crear instàncies EC2?

En aquesta guia trobaràs els passos per crear instàncies EC2 en AWS.

Tutorial bàsic per crear instàncies EC2 en Amazon Web Services (AWS). (pdf)

3.6.1 1.6.1 Pas 1. Iniciem el laboratori

Des del curs d'AWS Academy iniciem el laboratori prement *Start Lab* i esperem que l'ícone al costat del text d'AWS es torne verd.



3.6.2 1.6.2 Pas 2. Accedim a la consola de gestió d'AWS

Premem sobre el text d'AWS, i s'obrirà una nova pestanya amb la consola de gestió d'AWS.

Seleccionem el servei EC2.

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and account information. Below the header, the main title 'AWS Management Console' is displayed. On the left, a sidebar titled 'AWS services' shows 'Recently visited services' with 'EC2' listed. The main content area features several sections: 'Build a solution' with options like 'Launch a virtual machine' (With EC2, 2-3 minutes) and 'Build a web app' (With Elastic Beanstalk, 6 minutes); 'Explore AWS' with 'AWS Machine Learning Training' and 'Join the Graviton Challenge'; and a 'Stay connected to your AWS resources on-the-go' section about the AWS Console Mobile App. At the bottom, there are links for feedback, language selection (English US), and legal notices.

3.6.3 1.6.3 Pas 3. Cerquem el botó "Launch Instance" per crear una instància EC2

La manera més ràpida de crear una instància EC2 és a través del botó *Launch Instance* que apareix en el tauler del servei EC2.

The screenshot shows the AWS EC2 Management Dashboard. On the left, there's a sidebar with options like 'New EC2 Experience', 'EC2 Dashboard', 'Instances', 'Images', and 'Feedback'. The main area has a grid of statistics: Instances (running) 3, Dedicated Hosts 0, Elastic IPs 4, Instances 3, Key pairs 1, Load balancers 0, Placement groups 0, Security groups 16, Snapshots 0, and Volumes 3. Below this is a callout box with text about Microsoft SQL Server Always On availability groups. To the right, there's a 'Service health' section with a 'Launch Instance' button highlighted with a red box. Further right are sections for 'Default VPC', 'Settings', 'Explore AWS', and copyright information.

3.6.4 1.6.4 Pas 4. Assignem un nom a la instància EC2 que anem a crear

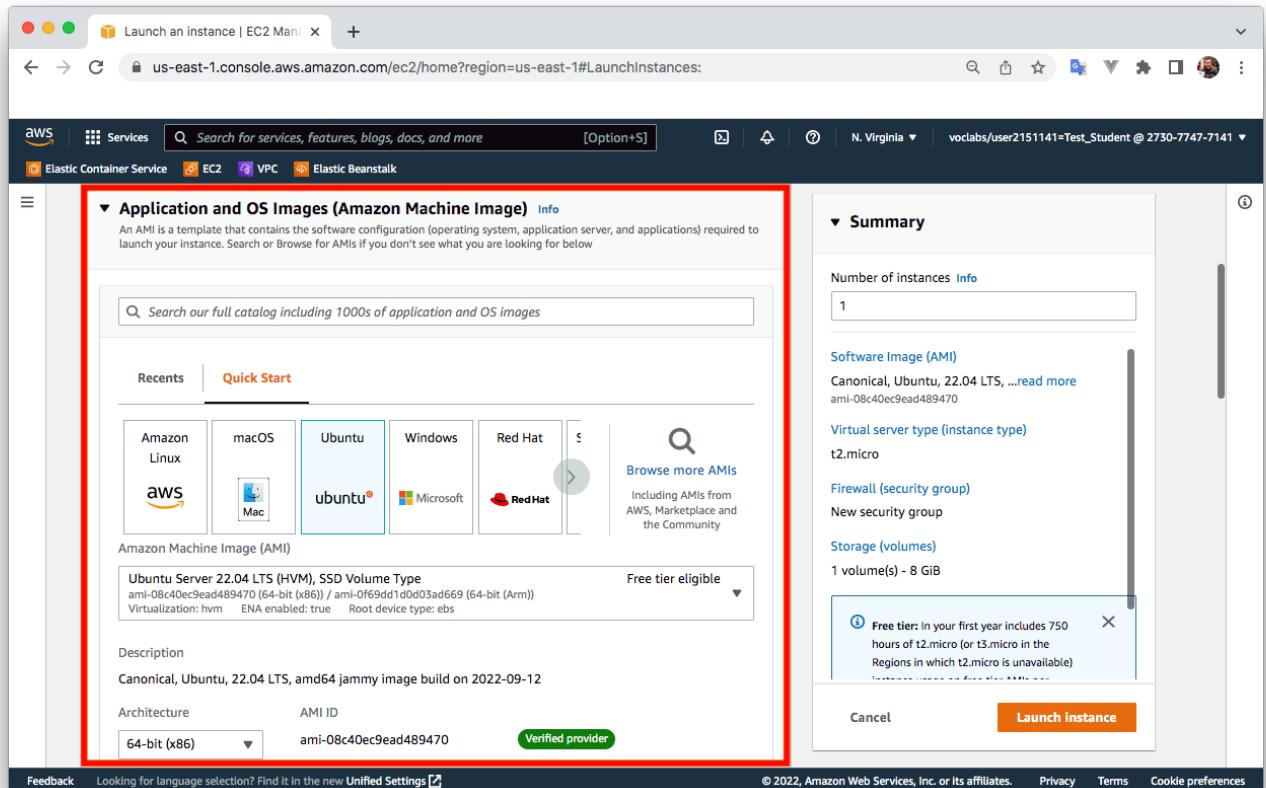
En aquest pas podem assignar-li un nom a la instància EC2 i afegir etiquetes amb metadades (parelles clau-valor). Les etiquetes permeten organitzar o filtrar les instàncies.

The screenshot shows the AWS EC2 'Launch an instance' wizard. The 'Name and tags' section is highlighted with a red box. Inside this box, there is a 'Name' input field containing the text 'e.g. My Web Server'. To the right of the input field is a link 'Add additional tags'.

3.6.5 1.6.5 Pas 5. Seleccionem la AMI i l'arquitectura de la instància EC2

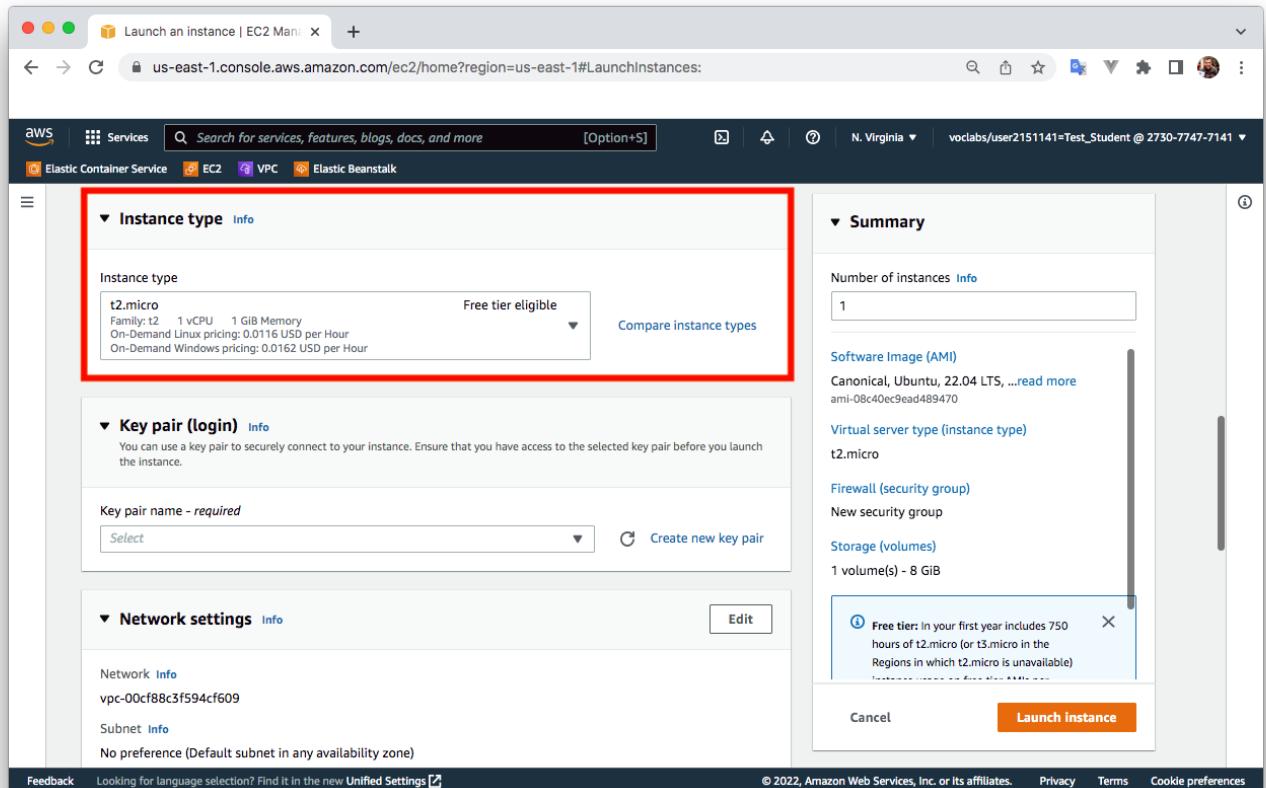
Seleccionem la AMI (Amazon Machine Image) i l'arquitectura (x86 o Arm).

En aquest exemple, seleccionem la imatge *Ubuntu Server 22.04 LTS (HVM), SSD Volume Type, 64 bits (x86)*.



3.6.6 1.6.6 Pas 6. Seleccionem el tipus d'instància EC2

Seleccionem crear una instància de tipus *t2.micro*, amb 1 vCPU i 1 GiB de RAM.



3.6.7 1.6.7 Pas 7. Seleccionem la clau pública SSH que injectarem a la instància EC2

Seleccionem la clau pública SSH que injectarem a la instància. Podem generar un parell de claus SSH o utilitzar les claus associades al nostre compte en AWS Learner Lab.

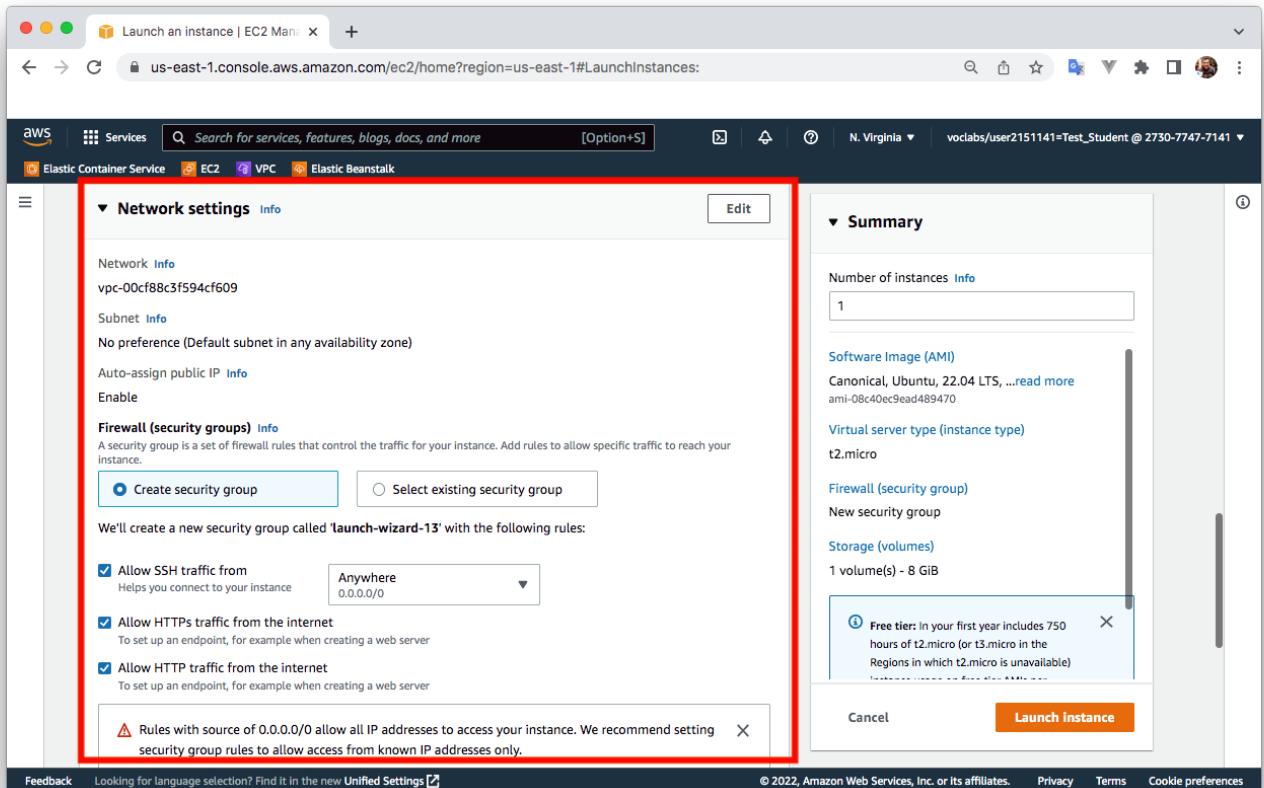
En aquest exemple, utilitzem la clau pública `vockey`.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Key pair (login)' step, a red box highlights the 'Key pair name - required' input field, which contains the value 'vockey'. Below this, the 'Network settings' section is shown, containing fields for Network (vpc-00cf88c3f594cf609), Subnet (No preference), and Auto-assign public IP (Enable). Under Firewall (security groups), there are two options: 'Create security group' (selected) and 'Select existing security group'.

3.6.8 1.6.8 Pas 8. Configurem la xarxa i els grups de seguretat

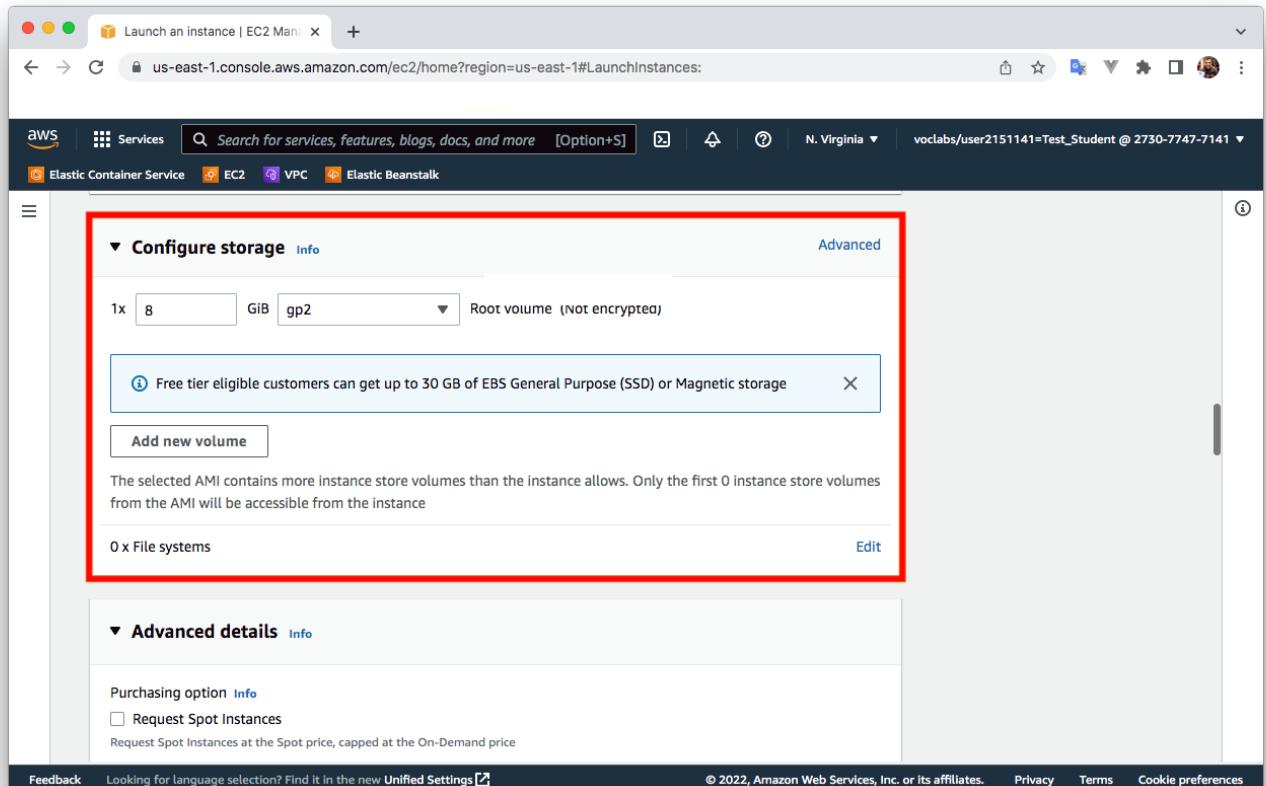
Podem configurar la xarxa VPC i les regles del grup de seguretat. En aquest exemple, creem un nou grup de seguretat amb les següents regles:

- **SSH.** Protocol: TCP. Port: 22. Origen: 0.0.0.0/0
- **HTTP.** Protocol: TCP. Port: 80. Origen: 0.0.0.0/0
- **HTTPS.** Protocol: TCP. Port: 443. Origen: 0.0.0.0/0



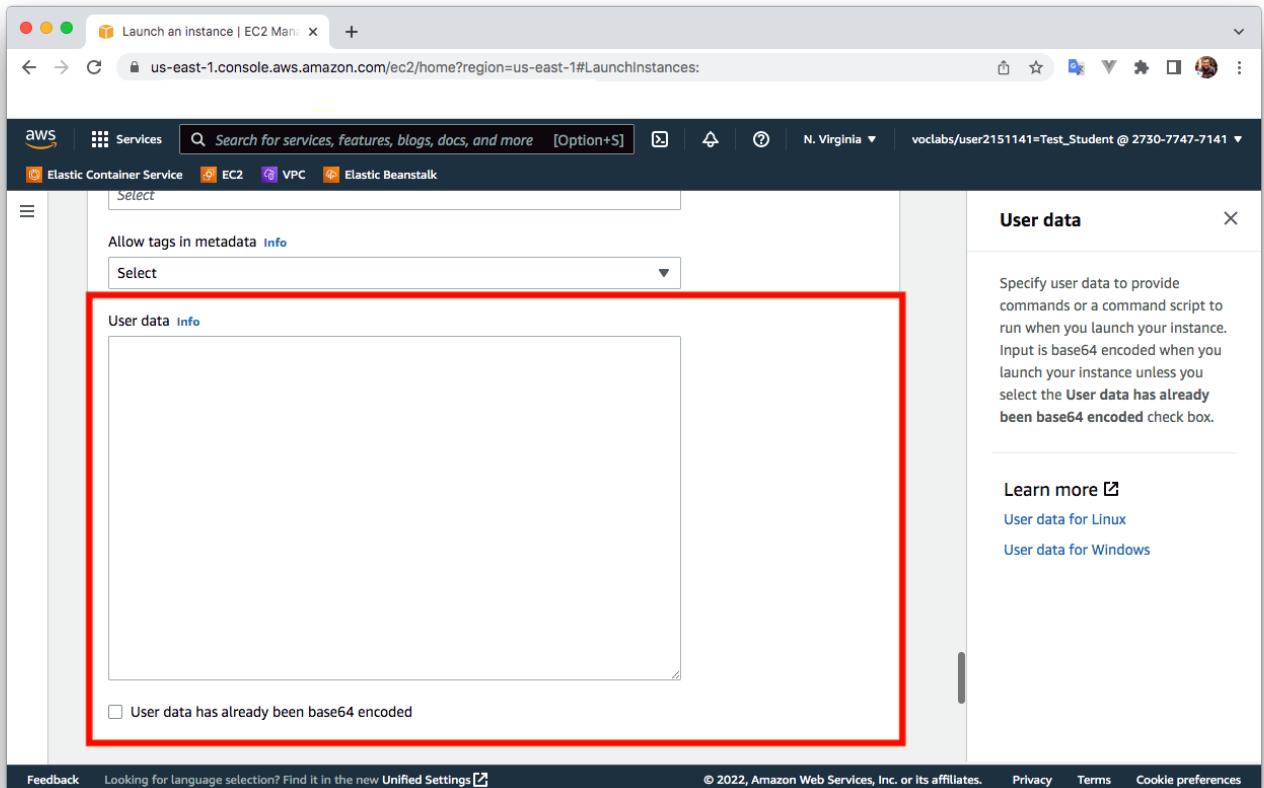
3.6.9 1.6.9 Pas 9. Afegim l'emmagatzematge que utilitzarà la instància

Configurem l'emmagatzematge de la instància EC2. En aquest exemple, utilitzem un disc SSD de 8 GB per defecte.



3.6.10 1.6.10 Pas 10. Details avançats

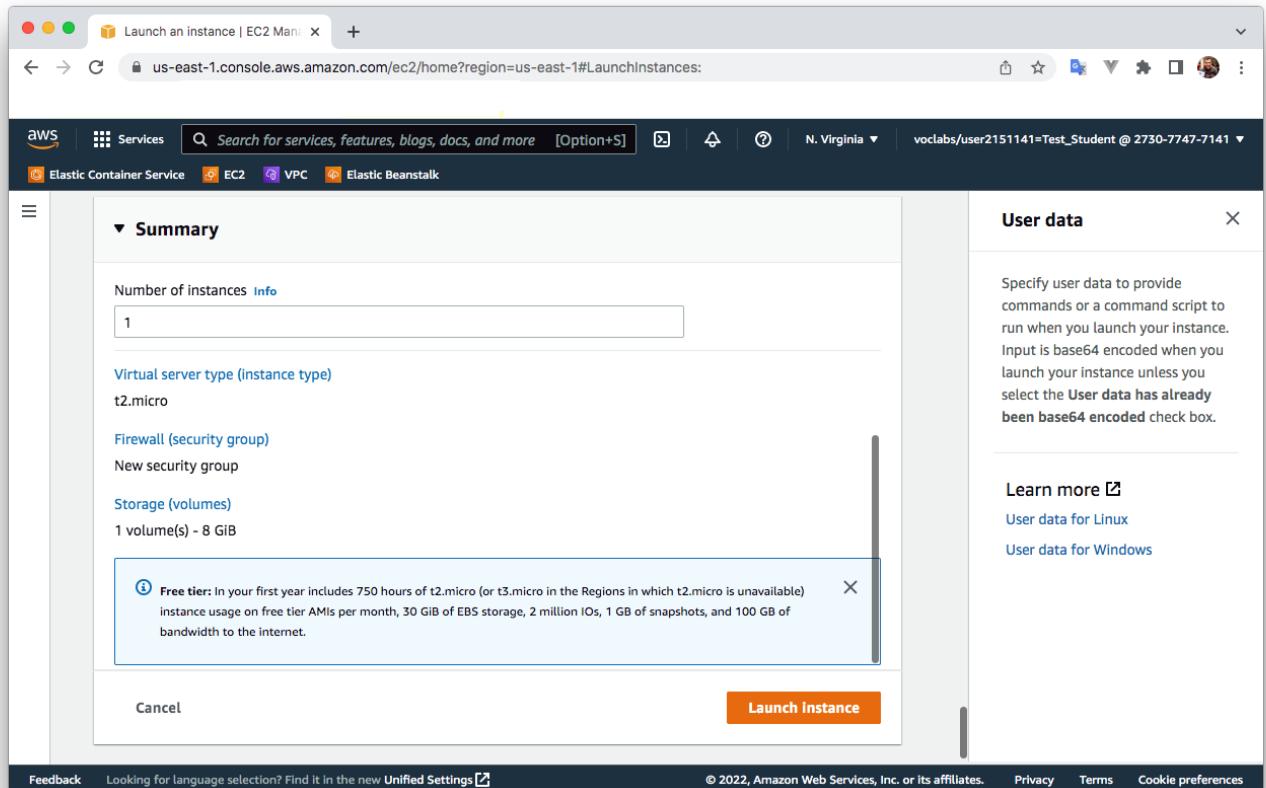
En aquest pas podem configurar alguns detalls avançats de la instància. Per exemple, en el camp *User data* podem indicar comandos o un script que s'executarà quan es reinicie la instància. Aquesta opció permet preparar la instància perquè s'inicie automàticament amb l'estat desitjat.



3.6.11 1.6.11 Pas 11. Revisem la configuració seleccionada

En aquest pas podem revisar la configuració seleccionada i triar el nombre d'instàncies que volem crear.

Per crear la instància, feu clic sobre el botó *Launch Instance*.



3.6.12 1.6.12 Pas 12. Estat de la instància

En aquest pas apareixerà un missatge indicant que la instància s'està creant. Farem clic sobre l'identificador de la instància que apareix en el quadre verd.

The screenshot shows the AWS Launch Instance Wizard interface. At the top, it says "Launch instance wizard | EC2". Below that is a navigation bar with "Services", a search bar, and user information "voclabs/user1690903=Test_Student @ 6171-1699-7141 N. Virginia Support". The main content area is titled "Launch Status". It contains a green box with a checkmark stating "Your instances are now launching" and a link to "View launch log". Below this is a blue box with an info icon and the text "Get notified of estimated charges", followed by a subtext about creating billing alerts. Further down, there's a section titled "How to connect to your instances" with a note about instances launching and accruing usage hours. It also lists helpful resources like the User Guide and Discussion Forum. At the bottom, there's a footer with the URL "https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:search=i-00b9a4e23df2d98c5;sort=instanceId", copyright information, and links to Privacy Policy, Terms of Use, and Cookie preferences.

3.6.13 1.6.13 Pas 13. Consultem l'estat de la instància des de la consola de gestió

Després de pocs segons, la instància que hem creat apareixerà a la consola. Quan la instància mostre l'estat *Running*, estarà llesta per connectar-nos.

The screenshot shows the AWS EC2 Management console interface. On the left, there's a sidebar with navigation links for New EC2 Experience, EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations), Images (AMIs), and Elastic Block Store. The main content area is titled 'Instances (1/1)' and displays a table with one row. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. The first row shows 'Name' as '-' and 'Instance ID' as 'i-00b9a4e23df2d98c5'. The 'Instance state' column shows a green circle with a checkmark and the word 'Running'. The 'Instance type' column shows 't2.micro'. The 'Status check' column shows a green circle with a checkmark and the word 'Initializing'. The 'Alarm status' column shows 'No alarms'. At the bottom of the main area, it says 'Instance: i-00b9a4e23df2d98c5'. The top of the page shows the URL 'console.aws.amazon.com/ec2/v2/home?region=us-east-1#instances:search=i-00b9a4e23df2d98c5' and the AWS logo.

3.7 1.7 Com connectar-nos a una instància EC2 per SSH des de Linux

3.7.1 1.7.1 Pas 1. Seleccionem la instància i fem clic sobre Connect

Quan la instància mostre l'estat *Running*, marcarem el checkbox que apareix al principi de la línia i després farem clic sobre *Connect*.

The screenshot shows the AWS EC2 Management console with the URL <https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#instances:search=i-00b9a4e23df2d98c5>. The left sidebar shows navigation options like EC2 Dashboard, Events, Tags, Limits, Instances (selected), Images, and Elastic Block Store. The main content area displays the 'Instances (1/1)' section with a table. The table has columns: Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. One row is selected, showing: Name - -, Instance ID - i-00b9a4e23df2d98c5, Instance state - Running, Instance type - t2.micro, Status check - Initializing, and Alarm status - No alarms.

3.7.2 1.7.2 Pas 2. Obtenim el comando per connectar-nos des d'un client SSH en Linux

En aquest pas apareixerà com podem connectar-nos a la instància que acabem de crear.

Seleccionarem l'opció *SSH client* i llegirem els passos per connectar-nos a la instància. Podem copiar el comando que apareix en aquest pas, que conté el nom de l'arxiu .pem de la clau privada, el nom d'usuari i el DNS públic de la instància.

Exemple:

```
1 $ ssh -i "voockey.pem" ubuntu@ec2-54-224-67-209.compute-1.amazonaws.com
```

The screenshot shows the AWS EC2 Instances page for an instance named i-00b9a4e23df2d98c5. The 'Connect to instance' section is open, specifically the 'SSH client' tab. It provides instructions for connecting via SSH, including:

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is vockey.pem.
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 vockey.pem
- Connect to your instance using its Public DNS:
ssh -i "vockey.pem" ubuntu@ec2-54-224-67-209.compute-1.amazonaws.com

A green box highlights the copied command message: "Command copied". A note at the bottom states: "Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name."

3.8 1.7.3 Pas 3. Descarreguem la clau privada .PEM des del Learner Lab

Per descarregar la clau, fem clic sobre *AWS Details*, i apareixerà la possibilitat de descarregar la clau com un arxiu .PEM o .PPK.

En Linux utilitzarem la clau .PEM, i en Windows, si usem PuTTY, utilitzarem la clau .PPK. Quan descarregueu la clau, es descarregarà com `labuser.pem` o `labuser.ppk`.

You are currently logged into Student View

Resetting the test student will clear all history for this student, allowing you to view the course as a brand new student.

[Reset Student](#) [Leave Student View](#)

3.9 1.7.4 Pas 4. Connectar-nos per SSH des de Linux

Quan descarregueu la clau privada `labuser.pem`, guardeu-la en un directori i, des d'allí, realitzeu els següents passos.

Renombrem l'arxiu `labuser.pem` com `vockey.pem`:

```
1 $ mv labuser.pem vockey.pem
```

IMPORTANT: canviem els permisos de l'arxiu perquè només el propietari tinga permisos de lectura:

```
1 $ chmod 400 vockey.pem
```

Executem el comando copiat en el pas 2 per connectar-nos per SSH a la instància EC2 des d'un terminal de Linux. El comando serà semblant a aquest, però amb un DNS públic diferent:

```
1 $ ssh -i "vockey.pem" ubuntu@ec2-54-224-67-209.compute-1.amazonaws.com
```

Esta pàgina forma part del curs Implantació d'Aplicacions Web de José Juan Sánchez Hernández i el seu contingut es distribueix sota una llicència [Creative Commons Reconeixement-NoComercial-CompartirIgual 4.0 Internacional](#)

4. UD1

4.1 Instalación MySql Server

4.1.1 Instalación utilizando binarios

Oracle proporciona un conjunto de distribuciones binarias de MySQL. Estas incluyen distribuciones genéricas binarias en forma de archivos **tar**, archivos con una tar.xz como extensión. Existen diferentes binarios específicos para cada plataforma. En esta sección vamos a ver la instalación de MySQL desde una distribución binaria de archivo **tar** comprimido para la plataforma **Unix/Linux**.

Ejemplo para descargar el binario:

```
1 wget https://dev.mysql.com/get/Downloads/MySQL-8.0/mysql-8.0.26-linux-glibc2.12-x86_64.tar.xz
```

Advertencias: * Si ha instalado previamente una versión de MySQL utilizando YUM/APT (Gestores de paquetes) puede experimentar problemas. Asegurarse de eliminar las aplicaciones y ficheros de configuración completamente del sistema * MySQL depende de la biblioteca **libaio**, si es necesario instalela utilizando el administrador de paquetes adecuado.

```
1 apt-cache search libaio # search for info
2 apt-get install libaio1 # install library for APT-based systems
```

Para instalar una distribución binaria de archivo tar comprimido , descomprimalo en la ubicación de instalación que elija (normalmente /usr/local/mysql). Esto crea los directorios que se muestran en la siguiente tabla.

Directorio	Contenido
bin	programas de utilidad, cliente y servidor mysqld
docs	Manual de MySQL en formato Info
man	Páginas de manual de Unix
include	Páginas de manual de Unix
lib	Bibliotecas
share	Mensajes de error, diccionario y SQL para la instalación de la base de datos
support-files	Archivos de soporte varios

La secuencia de comandos para instalar MySql es la siguiente:

```
1 groupadd mysql
2 useradd -r -g mysql -s /bin/false mysql
3 cd /usr/local
4 tar xvf /path/to/mysql-VERSION-OS.tar.xz
5 ln -s full-path-to-mysql-VERSION-OS mysql
6 export PATH=$PATH:/usr/local/mysql/bin
7 cd mysql
8 mkdir mysql-files
9 chown mysql:mysql mysql-files
10 chmod 750 mysql-files
11 bin/mysqld --initialize --user=mysql
12 bin/mysql_ssl_setup
13 bin/mysqld_safe --user=mysql & # Next command is optional
14 cp support-files/mysql.server /etc/init.d/mysql.server
```

Este proceso asume que tenemos acceso **root** al sistema. Alternativamente podemos prefijar cada comando usando el comando **sudo**

4.1.2 Crear un usuario y un grupo de mysql

Si su sistema aún no tiene un usuario y un grupo para usar para ejecutar **mysqld**, es posible que deba crearlos. La sintaxis de useradd y groupadd puede diferir ligeramente en diferentes versiones de **Unix/Linux**, o pueden tener diferentes nombres como adduser y addgroup.

```
1 groupadd mysql
2 useradd -r -g mysql -s /bin/false mysql
```

Debido a que el usuario solo es necesario para fines de propiedad, no de inicio de sesión, el comando useradd utiliza las opciones **-r y -s /bin/false** para crear un usuario que no tenga permisos de inicio de sesión en el servidor.

4.1.3 Obtenga y desempaque la aplicación

Elija el directorio en el que desea descomprimir la distribución y cambie la ubicación en él. El ejemplo aquí descomprime la distribución en /usr/local. Por lo tanto, las instrucciones asumen que tiene permiso para crear archivos y directorios en /usr/local. Si ese directorio está protegido, debe realizar la instalación como root. Obtenga un archivo de distribución siguiendo las instrucciones de la “[Cómo obtener MySQL](#)” de la documentación de MySQL. Para una versión determinada, las distribuciones binarias para todas las plataformas se crean a partir de la misma distribución fuente de MySQL. Desempaque la distribución, que crea el directorio de instalación. tar puede descomprimir y descomprimir la distribución si tiene la opción de soporte:

```
1 cd /usr/local
2 tar xvf /path/to/mysql-VERSION-OS.tar.xz
```

El comando **tar** crea un directorio llamado **.mysql-VERSION-OS**

Si su **tar** no es compatible con **xz** utilizar el siguiente comando:

```
1 xz -dc /path/to/mysql-VERSION-OS.tar.xz | tar x
```

Por último crear el enlace simbólico a la carpeta de instalación y agregar el directorio al path para que sea más sencillo acceder a los programas.

```
1 ln -s full-path-to-mysql-VERSION-OS mysql
2 export PATH=$PATH:/usr/local/mysql/bin
```

Por último saltar a: Configuración y pruebas posteriores a la instalación.

4.1.4 Instalación utilizando el gestor de paquetes APT de linux.

Al igual que en el punto anterior, asumimos que en el sistema no se ha instalado previamente ninguna versión de MySQL Server o ha sido eliminada completamente.

Pasos para una nueva instalación de MySql

Primero agragar el paquete .deb del repositorio APT de MySQL. Podemos descargarlo del [repositorio](#). Instalar el paquete con el siguiente comando (reemplazar x-y-z con la versión utilizada)

```
1 sudo dpkg -i mysql-apt-config_x.y.z_all.deb
```

Durante la instalación del paquete, se pedirá que elija las versiones del servidor MySQL y otros componentes (por ejemplo, MySQL Workbench) que quieras instalar. Si no estás seguro de qué versión elegir, no cambies las opciones predeterminadas seleccionadas. Actualizar la información de los repositorios **paso obligado**

```
1 sudo apt-get update
```

Instalación del paquete con APT

Instalar con el comando:

```
1 sudo apt-get install mysql-server
```

Esto instala el paquete para el servidor MySQL, así como los paquetes para el cliente y para los archivos comunes de la base de datos. Durante la instalación, se le pedirá que proporcione una **contraseña** para el usuario **root** para su instalación de MySQL.

Iniciar y detener el servidor

El servidor MySQL se inicia automáticamente después de la instalación. Puede verificar el estado del servidor MySQL con el siguiente comando:

```
1 systemctl status mysql
```

Si el sistema operativo está habilitado para systemd, los comandos estándar systemctl (o alternativamente, servicio con los argumentos invertidos) como detener, iniciar, estado y reiniciar deben usarse para administrar el servicio del servidor MySQL. El servicio mysql está habilitado de forma predeterminada y se inicia al reiniciar el sistema. Consulte Administrar MySQL Server con systemd para obtener información adicional.

Opciones de **systemctl**.

Opcion	Significado	Comando
stop	Detener el servicio	systemctl stop mysql
start	Iniciar el servicio	systemctl start mysql
restart	Reiniciar el servicio (star/stop)	systemctl restart mysql
reload	Recargar la configuración	systemctl reload mysql

Instalar componentes adicionales de MySQL

Puedes usar APT para instalar componentes individuales de MySQL desde el repositorio APT de MySQL. Suponiendo que ya tiene el repositorio APT de MySQL en la lista de repositorios de su sistema. Primero, use el siguiente comando para obtener la información más reciente del paquete del repositorio APT de MySQL:

```
1 sudo apt-get update
```

Ejemplos:

```
1 sudo apt-get install mysql-workbench-community #cliente workbench
2 sudo apt-get install libmysqlclient21 #bibliotecas cliente compartidas
```

4.1.5 Actualizar la versión de MySQL Server

Advertencias: * Tener una copia de seguridad previa de la información antes de actualizar * Si es posible probar primero en un servidor de preproducción * Los comandos que veremos a continuación no sirven si la instalación previa no es de Oracle... Si es MariaDB o Percona

Primero asegurarse que tenemos el repositorio MySQL apt en nuestra lista de repositorios Actualizar el sistema

```
1 sudo apt-get update
```

Como regla general, para actualizar de una serie de lanzamientos a otra, vaya a la siguiente serie en lugar de omitir una serie. Por ejemplo, si actualmente estás ejecutando MySQL 5.5 y deseas actualizar a una serie más nueva, actualiza primero a MySQL 5.6 antes de actualizar a 5.7, y así sucesivamente. Actualizar con el siguiente comando:

```
1 sudo apt-get install mysql-server
```

Si hay que actualizar otros paquetes utilizar el mismo comando:

```
1 sudo apt-get install package-name
```

Para saber el nombre de los paquetes instalados desde el repositorio APT de MySQL, use el siguiente comando:

```
1 dpkg -l | grep mysql | grep ii
```

Advertencia: Si realizas una actualización de todo el sistema mediante **apt-get upgrade**, solo la biblioteca MySQL y los paquetes de desarrollo se actualizan con versiones más recientes (si están disponibles). Para actualizar otros componentes, incluido el servidor, el cliente, la suite de pruebas, etc., use el comando apt-get install. El servidor MySQL siempre se reinicia después de una actualización de APT. Antes de MySQL 8.0.16, ejecute **mysql_upgrade** después de que el servidor se reinicia para verificar y posiblemente resolver cualquier incompatibilidad entre los datos antiguos y el software actualizado. **mysql_upgrade** también realiza otras funciones; consulte [mysql_upgrade](#). A partir de MySQL 8.0.16, este paso no es necesario, ya que el servidor realiza todas las tareas previamente manejadas por **mysql_upgrade**.

Eliminando MySQL con APT

Para desinstalar el servidor MySQL y los componentes relacionados que se han instalado usando el repositorio APT de MySQL, elimine el servidor MySQL y cualquier otro software que se haya instalado automáticamente usando los siguientes comandos:

```
1 sudo apt-get remove mysql-server
2 sudo apt-get autoremove
```

Para desinstalar otros paquetes instalados manualmente utilizar:

```
1 sudo apt-get remove package-name
```

Para ver una lista de los paquetes que han instalado desde el repositorio APT de MySQL, use el siguiente comando:

```
1 dpkg -l | grep mysql | grep ii
```

4.1.6 Paquetes disponibles del repositorio APT de Mysql

Package Name	Description
mysql-server	Metapackage for installing the MySQL server
mysql-community-server	MySQL server
mysql-client	Metapackage for installing the MySQL client
mysql-cluster-community-auto-installer *	Auto installer for NDB Cluster
mysql-cluster-community-client	MySQL client for NDB Cluster
mysql-cluster-community-data-node	NDB Cluster data node
mysql-cluster-community-java	NDB Cluster Java drivers
mysql-cluster-community-management-server	NDB Cluster management node
mysql-cluster-community-memcached	NDB Cluster memcached server
mysql-cluster-community-nodejs	NDB Cluster Node.js adapters
mysql-cluster-community-server	MySQL server for NDB Cluster
mysql-cluster-community-source	Source package for NDB Cluster
mysql-cluster-community-test	NDB Cluster testsuite
mysql-community-client	MySQL client
mysql-common	MySQL database common files
libmysqlclient21	MySQL database client library
libmysqlclient-dev	MySQL database development files
libmysqld-dev	MySQL embedded database development files
mysql-testsuite	Metapackage for installing the MySQL test suite
mysql-community-test	MySQL test suite
mysql-community-bench	MySQL benchmark suite
mysql-community-source	MySQL source code
mysql-workbench-community	MySQL Workbench (not available for Debian platforms)
mysql-connector-python-py3	MySQL Connector/Python for supported Ubuntu versions with Python 3.2 or later
mysql-connector-python	MySQL Connector/Python for supported Debian versions with Python 2.6.3 or later, and for supported Ubuntu versions with Python 2.6.3 to 3.1
mysql-router	MySQL Router
ndbclient	NDB Cluster client
ndbclient-dev	NDB Cluster client development library

4.1.7 Configuración y pruebas posteriores a la instalación

Después de instalar MySQL, se debe inicializar el directorio de datos, incluidas las tablas en el esquema mysql del sistema: *

Para algunos métodos de instalación de MySQL, la inicialización del directorio de datos es automática * Para otros métodos de instalación, se debe inicializar el directorio de datos manualmente. Estos incluyen la instalación desde distribuciones genéricas binarias y de origen en sistemas Unix y similares a Unix, y la instalación desde un paquete ZIP en Windows. En general, cuando instalamos desde un instalador (exe o msi en Windows o apt/yum en linux) la inicialización del directorio es automática. Cuando la instalación es mediante la copia de binarios o compilación hay que iniciar estos directorios manualmente.

Inicialización del directorio de datos

DESCRIPCIÓN GENERAL DE LA INICIALIZACIÓN

Esta sección describe cómo inicializar el directorio de datos manualmente para los métodos de instalación de MySQL para los cuales la inicialización del directorio de datos no es automática. Advertencia. En MySQL 8.0, el complemento de autenticación predeterminado ha cambiado de *mysql_native_password* a *caching_sha2_password*, y la cuenta '*root*'@'*localhost*' administrativa utiliza *caching_sha2_password* de forma predeterminada. Si prefiere que la cuenta **root** utilice el complemento de autenticación predeterminado anterior (*mysql_native_password*), consulte [caching_sha2_password y la cuenta administrativa raíz](#). Los pasos son los siguientes:

- * Cambiar la ubicación al nivel superior de instalación de mysql
- * Crear un directorio
- * Otorgar la propiedad de la carpeta al usuario y grupo mysql
- * Utilizar el servidor para inicializar el directorio de datos

```
1 cd /usr/local/mysql
2 mkdir mysql-files
3 chown mysql:mysql mysql-files
4 chmod 750 mysql-files
5 bin/mysqld --initialize --user=mysql
```

Si queremos implementar el soporte automático para conexiones seguras, hay que utilizar la utilidad **mysql_ssl_rsa_setup** para crear los archivos SSL y RSA respectivamente

```
1 bin/mysql_ssl_rsa_setup
```

PROCEDIMIENTO DE INICIALIZACIÓN DEL DIRECTORIO DE DATOS

Cambiar la ubicación al nivel superior de instalación de mysql.

```
1 cd /usr/local/mysql
```

Para inicializar el directorio de datos, invoque mysqld con la opción *--initialize* o *--initialize-insecure*, dependiendo de si desea que el servidor genere una contraseña inicial aleatoria para la cuenta '*root*'@'*localhost*', o cree esa cuenta sin contraseña:

- Use *--initialize* para la instalación "segura por defecto" (es decir, incluida la generación de una rootcontraseña inicial aleatoria). En este caso, la contraseña se marca como caducada y debe elegir una nueva.
- Con *--initialize-insecure*, no se genera ninguna contraseña para root. **Esto es inseguro;** se supone que tiene la intención de asignar una contraseña a la cuenta de manera oportuna antes de poner el servidor en uso de producción.

En sistemas Unix y similares a Unix, es importante que los directorios y archivos de la base de datos sean propiedad de la cuenta *mysql* de inicio de sesión para que el servidor tenga acceso de lectura y escritura a ellos cuando lo ejecute más tarde. Para garantizar esto inicia mysqld desde la cuenta root del sistema e incluye la opción *--user* como se muestra aquí:

```
1 bin/mysqld --initialize --user=mysql
2 bin/mysqld --initialize-insecure --user=mysql
```

Podría ser necesario especificar otras opciones, como *--basedir* o *--datadir* si mysqld no puede identificar las ubicaciones correctas para el directorio de instalación o el directorio de datos. Por ejemplo (ingrese el comando en una sola línea):

```
1 bin/mysqld --initialize --user=mysql
2 -basedir=/opt/mysql/mysql
3 --datadir=/opt/mysql/mysql/data
```

Alternativamente, coloca la configuración de opciones relevante en un archivo de opciones y pase el nombre de ese archivo a mysqld. Para sistemas Unix y similares a Unix, supon que el nombre del archivo de opciones es `/opt/mysql/mysql/etc/my.cnf`. Pon estas líneas en el archivo:

```
1 [mysqld]
2 basedir=/opt/mysql/mysql
3 datadir=/opt/mysql/mysql/data
```

ACCIONES EN EL SERVIDOR DURANTE LA INICIALIZACIÓN DEL DIRECTORIO DE DATOS

Cuando se invoca con la opción `--initialize` o `--initialize-insecure`, mysqld realiza las siguientes acciones durante la secuencia de inicialización del directorio de datos: * El servidor verifica la existencia del directorio de datos (si no existe lo crea) * Si el directorio de datos existe pero no está vacío, el servidor se cierra después de producir un mensaje de error:

```
1 [ERROR] --initialize specified but the data directory exists. Aborting.
```

En este caso, elimine o cambie el nombre del directorio de datos y vuelva a intentarlo. Se permite que un directorio de datos existente no esté vacío si cada entrada tiene un nombre que comienza con un punto (.). * Dentro del directorio de datos el servidor crea el esquema de mysql y sus tablas, incluidas las tablas del diccionario de datos, las tablas de concesión, las tablas de zona horaria y las tablas de ayuda del lado del servidor. * El servidor inicializa el espacio de la tabla del sistema y las estructuras de datos relacionadas necesarias para administrar las tablas InnoDB * El servidor crea una cuenta 'root'@'localhost' de superusuario y otras cuentas reservadas + Con `--initialize` pero sin `--initialize-insecure`, el servidor genera una contraseña aleatoria, la marca como caducada y escribe un mensaje que muestra la contraseña.

```
1 * Con --initialize-insecure, (ya sea con o sin --initialize porque --initialize-insecure implica --initialize), el servidor no genera una contraseña ni la marca caducada, y escribe un mensaje de advertencia:
```

```
1 #initialize
2 [Warning] A temporary password is generated for root@localhost:
3 iTag*AfrH5ej
```

```
1 #initialize-insecure
2 [Warning] root@localhost is created with an empty password ! Please
3 consider switching off the --initialize-insecure option.
```

* El servidor completa las tablas de ayuda del lado del servidor que se utilizan para la declaración HELP * Si se proporcionó la variable `init_file` del sistema para nombrar un archivo de sentencias SQL, el servidor ejecuta las sentencias en el archivo. Esta opción le permite realizar secuencias de arranque personalizadas. * Por último se cierra el servidor

ASIGNACIÓN DE LA CONTRASEÑA ROOT POSTERIOR A LA INICIALIZACIÓN

Estos son los pasos a seguir: * Iniciar el servidor * Conectarse al servidor

```
1 mysql -u root -p
```

* Luego, cuando se le solicite la contraseña, ingrese la contraseña aleatoria que generó el servidor durante la secuencia de inicialización:

```
1 Enter password: (enter the random root password here)
```

* Si inició con `--initialize-insecure` conectarse sin contraseña. Después de conectarse use **ALTER USER** para generar una nueva contraseña.

```
1 mysql -u root --skip-password
2 ...
3 mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'root-password';
```

Iniciar el servidor

Para iniciar el servidor:

```
1 systemctl start mysql
```

ó si la instalación incluye *mysqld_safe*

```
1 bin/mysqld_safe --user=mysql &
```

Es importante que el servidor MySQL se ejecute con una cuenta root de inicio de sesión sin privilegios (sin). Para garantizar esto, ejecute *mysqld_safe* como root e incluya la opción *--user* como se muestra. De lo contrario, debe ejecutar el programa mientras está conectado como mysql, en cuyo caso puede omitir la opción *--user* del comando.

Probar el servidor

Después de inicializar el directorio de datos y de haber iniciado el servidor, realizar algunas pruebas sencillas para asegurarse de que funciona satisfactoriamente. Esta sección asume que su ubicación actual es el directorio de instalación de MySQL y que tiene un subdirectorio bin que contiene los programas de MySQL usados aquí. Si eso no es cierto, ajuste los nombres de la ruta de comando en consecuencia.

Alternativamente, agregue el directorio bin a su configuración de variable de entorno PATH. Eso permite que su terminal (intérprete de comandos) encuentre programas MySQL correctamente, de modo que pueda ejecutar un programa escribiendo solo su nombre, no su nombre de ruta.

Utilice **mysqladmin** para verificar que el servidor se esté ejecutando. Los siguientes comandos proporcionan pruebas simples para verificar si el servidor está funcionando y respondiendo a las conexiones:

```
1 bin/mysqladmin version
2 bin/mysqladmin variables
```

Si no puedes conectarte al servidor, especifica una opción *-u root* para conectarte. Si ya ha asignado una contraseña para la rootcuenta, también deberá especificar *-p*. Por ejemplo:

```
1 bin/mysqladmin -u root -p version
2 Enter password: (enter root password here)
```

La salida debería ser similar a:

```
1 bin/mysqladmin version
2 mysqladmin Ver 14.12 Distrib 8.0.26, for pc-linux-gnu on i686
3 ...
4
5 Server version      8.0.26
6 Protocol version   10
7 Connection          Localhost via UNIX socket
8 UNIX socket         /var/lib/mysql/mysql.sock
9 Uptime:              14 days 5 hours 5 min 21 sec
10
11 Threads: 1  Questions: 366  Slow queries: 0
12 Opens: 0  Flush tables: 1  Open tables: 19
13 Queries per second avg: 0.000
```

Verifique que puede apagar el servidor (incluya una opción *-p* si la cuenta root ya tiene una contraseña):

```
1 bin/mysqladmin -u root shutdown
```

Verifique que pueda iniciar el servidor nuevamente. Haga esto usando *mysqld_safe* o invocando *mysqld* directamente. Por ejemplo:

```
1 bin/mysqld_safe --user=mysql &
```

Utilice `mysqlshow` para ver qué bases de datos existen:

```

1 bin/mysqlshow
2 +-----+
3 | Databases      |
4 +-----+
5 | information_schema |
6 | mysql           |
7 | performance_schema |
8 | sys             |
9 +-----+

```

Asegurar la cuenta inicial de MySQL

Esta sección describe cómo asignar una contraseña a la cuenta root inicial creada durante el procedimiento de instalación de MySQL, si aún no lo has hecho.

Advertencia: * En todas las plataformas, la distribución de MySQL incluye `mysql_secure_installation`, una utilidad de línea de comandos que automatiza gran parte del proceso de asegurar una instalación de MySQL.

Es posible que ya se haya asignado una contraseña a la cuenta inicial en las siguientes circunstancias: * Las instalaciones que utilizan paquetes Debian le dan la opción de asignar una contraseña. * Para la inicialización del directorio de datos realizada manualmente usando `mysqld --initialize`, mysqld genera una contraseña aleatoria inicial, la marca como vencida y la escribe en el registro de errores del servidor.

La tabla `mysql.user` de concesión define la cuenta de usuario inicial de MySQL y sus privilegios de acceso. La instalación de MySQL crea solo una cuenta '`root'@'localhost'` de **superusuario** que tiene todos los privilegios y puede hacer cualquier cosa. Si la cuenta root tiene una contraseña vacía, la instalación de MySQL está desprotegida: cualquiera puede conectarse al servidor MySQL root sin una contraseña y recibir todos los privilegios.

Para asignar una contraseña para la cuenta root MySQL inicial, utilice el siguiente procedimiento. Reemplace **root-password** en los ejemplos con la contraseña que desea usar.

Si la cuenta root existe con una contraseña aleatoria inicial que ha caducado, conéctate al servidor usando root esa contraseña, luego elige una nueva contraseña.

```

1 mysql -u root -p
2 Enter password: (enter the random root password here)
3 Elija una nueva contraseña para reemplazar la contraseña aleatoria:
4 ...
5 ...
6 mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'root-password';

```

Si la cuenta root existe pero no tiene contraseña, conéctate al servidor root sin usar contraseña, luego asigna una contraseña. Este es el caso si inicializó el directorio de datos usando **mysqld --initialize-insecure**.

Conéctate al servidor root sin usar contraseña:

```

1 mysql -u root --skip-password
2 Asignar una contraseña:
3 ...
4 mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'root-password';

```

4.1.8 Actualización de mysql

Rutas de actualización

- Se admite la actualización de MySQL 5.7 a 8.0. Sin embargo, la actualización solo se admite entre las versiones de disponibilidad general (GA). Para MySQL 8.0, es necesario que actualice desde una versión de MySQL 5.7 GA (5.7.9 o superior). No se admiten actualizaciones de versiones no GA de MySQL 5.7.
- Se recomienda actualizar a la última versión antes de actualizar a la siguiente versión. Por ejemplo, actualice a la última versión de MySQL 5.7 antes de actualizar a MySQL 8.0.
- No se admite la actualización que omite versiones. Por ejemplo, no se admite la actualización directa de MySQL 5.6 a 8.0.

- Una vez que una serie de lanzamientos alcanza el estado de Disponibilidad general (GA), se admite la actualización dentro de la serie de lanzamientos (de una versión GA a otra versión GA). Por ejemplo, actualizar desde MySQL 8.0.x a 8.0.y está apoyado. (No se admite la actualización que involucre versiones de estado de desarrollo que no sean de GA). También se admite la omisión de una versión. Por ejemplo, actualizar desde MySQL 8.0.x a 8.0.z está apoyado. MySQL 8.0.11 es la primera versión de estado GA dentro de la serie de versiones MySQL 8.0.

Qué actualiza el proceso de actualización de MySQL

La instalación de una nueva versión de MySQL puede requerir la actualización de estas partes de la instalación existente:

- * Paso 1: Actualización del diccionario de datos
- * Paso 2: Actualización del servidor

La siguiente lista muestra los comandos de actualización anteriores a MySQL 8.0.16 y los comandos equivalentes para MySQL 8.0.16 y versiones posteriores:

- Realice una actualización normal (pasos 1 y 2 según sea necesario):
- Antes de MySQL 8.0.16: [**mysqld**] seguido de [**mysql_upgrade**]
- A partir de MySQL 8.0.16: [**mysqld**]
- Realice solo el paso 1 según sea necesario:
- Antes de MySQL 8.0.16: no es posible realizar todas las tareas de actualización descritas en el paso 1 excluyendo las descritas en el paso 2. Sin embargo, puede evitar actualizar los esquemas de usuario y el esquema sys usando [**mysqld**] seguido de [**mysql_upgrade**] con las opciones [--upgrade-system-tables] y [--skip-sys-schema].
- A partir de MySQL 8.0.16: [**mysqld --upgrade = MINIMAL**]
- Realice el paso 1 según sea necesario y fuerce el paso 2:
- Antes de MySQL 8.0.16: [**mysqld**] seguido de [**mysql_upgrade --force**]
- A partir de MySQL 8.0.16: [**mysqld --upgrade = FORCE**]

Actualización con el repositorio APT de MySQL

- Antes de realizar cualquier actualización a MySQL, siga cuidadosamente las instrucciones en [Actualización de MySQL]. Entre otras instrucciones discutidas allí, *es especialmente importante hacer una copia de seguridad de su base de datos antes de la actualización* .
- Las siguientes instrucciones asumen que MySQL se ha instalado en su sistema usando el repositorio APT de MySQL; si ese no es el caso, siga las instrucciones que se dan en [Reemplazo de una distribución nativa de MySQL usando el repositorio APT de MySQL] o [Reemplazo de un servidor MySQL instalado por una descarga directa del paquete deb]. También tenga en cuenta que no puede usar el repositorio APT de MySQL para actualizar una distribución de MySQL que haya instalado desde un repositorio de software no nativo (por ejemplo, de MariaDB o Percona).
- Asegúrese de que tiene el repositorio APT de MySQL en la lista de repositorios de su sistema
- Asegurarse de que tienes la información del paquete más actualizado

```
1 sudo apt-get update
```

- Utilice el comando:

```
1 apt-get install mysql-server
```

4. Para acutlizar cualquier otro paquete:

```
1 apt-get install package-name
```

5. Para ver los nombres de los paquetes intalados

```
1 dpkg -l | grep mysql | grep ii
```

6. El servidor se reiniciará después de una actualización. Antes de MySQL 8.0.16, ejecute `mysql_upgrade` después de que el servidor se reinicie para verificar y posiblemente resolver cualquier incompatibilidad entre los datos antiguos y el software actualizado. A partir de MySQL 8.0.16, este paso no es necesario, ya que el servidor realiza todas las tareas previamente manejadas por `mysql_upgrade`.

4.1.9 Uso de imagen Docker para MySql

Descargar la imagen de Docker

Para descargar la imagen de Docker de MySQL, ejecute el siguiente comando:

```
1 docker pull mysql/mysql-server:tag
```

Donde **tag** es la versión de la imagen de MySQL que desea descargar. Por ejemplo, para descargar la imagen de MySQL 8.0, ejecute el siguiente comando:

```
1 docker pull mysql/mysql-server:8.0
```

Iniciar un contenedor de Docker con MySQL

Para iniciar un contenedor de Docker con MySQL, ejecute el siguiente comando:

```
1 docker run --name=mysql-container -e MYSQL_ROOT_PASSWORD=root-password -d mysql/mysql-server:tag
```

Donde **mysql-container** es el nombre del contenedor, **root-password** es la contraseña de root de MySQL y **tag** es la versión de la imagen de MySQL que desea utilizar. Por ejemplo, para iniciar un contenedor de Docker con MySQL 8.0 y la contraseña de root **root**, ejecute el siguiente comando:

```
1 docker run --name=mysql-container -e MYSQL_ROOT_PASSWORD=root -d mysql/mysql-server:8.0
```

Conectarse al contenedor de Docker con MySQL

Para conectarse al contenedor de Docker con MySQL, ejecute el siguiente comando:

```
1 docker exec -it mysql-container mysql -uroot -p
```

Donde **mysql-container** es el nombre del contenedor. Se le pedirá que ingrese la contraseña de root de MySQL. Una vez que haya ingresado la contraseña, se conectará al servidor MySQL en el contenedor de Docker.

Detener y eliminar el contenedor de Docker con MySQL

Para detener el contenedor de Docker con MySQL, ejecute el siguiente comando:

```
1 docker stop mysql-container
```

Para eliminar el contenedor de Docker con MySQL, ejecute el siguiente comando:

```
1 docker rm mysql-container
```

Persistencia de datos en contenedores de Docker

Para persistir los datos en un contenedor de Docker con MySQL, puede montar un volumen de Docker en el contenedor. Para hacer esto, agregue la opción **-v** al comando **docker run**. Por ejemplo, para montar un volumen de Docker en el contenedor de MySQL, ejecute el siguiente comando:

```
1 docker run --name=mysql-container -e MYSQL_ROOT_PASSWORD=root -v /path/to/mysql-data:/var/lib/mysql -d mysql/mysql-server:8.0
```

Donde **/path/to/mysql-data** es la ruta en su sistema host donde desea almacenar los datos de MySQL. Esto montará el volumen en el directorio **/var/lib/mysql** en el contenedor de Docker, lo que permitirá que los datos de MySQL persistan incluso si el contenedor se detiene o se elimina.

4.1.10 Docker Compose para MySQL

Crear un archivo docker-compose.yml

Para usar Docker Compose con MySQL, primero debe crear un archivo **docker-compose.yml** que defina la configuración de su contenedor de MySQL. Aquí hay un ejemplo de un archivo **docker-compose.yml** que define un contenedor de MySQL:

```

1 version: '3.1'
2
3 services:
4   mysql:
5     image: mysql/mysql-server
6     container_name: mysql-container
7     environment:
8       MYSQL_ROOT_PASSWORD: root
9     command: --default-authentication-plugin=mysql_native_password
10    ports:
11      - "3306:3306"
12    volumes:
13      - /path/to/mysql-data:/var/lib/mysql

```

```

1 version: '3.1'
2
3 services:
4   mysql:
5     image: mariadb
6     container_name: mysql-container
7     environment:
8       MYSQL_ROOT_PASSWORD: root
9     ports:
10      - "3306:3306"
11    volumes:
12      - /path/to/mysql-data:/var/lib/mysql

```

Pràctica 1: Instal·lació i Configuració Avançada de MySQL Server

4.2 Objectiu de la pràctica

Aquesta pràctica té com a objectiu avaluar els següents Resultats d'Aprenentatge (RA) i Criteris d'Avaluació (CA):

4.2.1 RA1: Implaça sistemes gestors de bases de dades analitzant les seves característiques i ajustant-se als requeriments del sistema.

CA:

- a) Reconeixement de la utilitat i funció dels elements d'un SGBD.
- b) Anàlisi de les característiques dels principals SGBD.
- c) Selecció del SGBD adequat.
- d) Identificació del programari necessari per a la instal·lació.
- e) Verificació dels requisits hardware.
- f) Instal·lació del SGBD.
- g) Documentació del procés d'instal·lació.
- h) Interpretació dels missatges d'error i fitxers de registre.
- i) Resolució d'incidències durant la instal·lació.
- j) Verificació del funcionament del SGBD.

4.2.2 RA2: Configura el sistema gestor de bases de dades interpretant les especificacions tècniques i els requisits d'explotació.

CA:

- a) Descripció de les condicions d'inici i parada del SGBD.
- b) Selecció del motor de base de dades.
- c) Assegurament dels comptes d'administració.
- d) Configuració de les eines i programari client.
- e) Configuració de la connectivitat en xarxa.
- f) Definició de les característiques per defecte de les bases de dades.
- g) Definició dels paràmetres relatius a les connexions.
- h) Documentació del procés de configuració.

4.2.3 RA3 (parcial): Implaça mètodes de control d'accés utilitzant assistents, eines gràfiques i comandaments del llenguatge del SGBD.

CA:

- c) Definició i eliminació de comptes d'usuari.
- h) Garantia del compliment dels requisits de seguretat.

4.3 Descripció de la pràctica

L'objectiu és realitzar la instal·lació i configuració avançada de MySQL Server en un entorn amb múltiples màquines virtuals, seguint els passos detallats a continuació, i documentant tot el procés en format Markdown.

4.4 Tasques a realitzar

4.4.1 Preparació de l'entorn amb múltiples màquines:

1. Crea dues màquines virtuals amb sistema operatiu Linux (per exemple, Ubuntu Server).
2. Verifica la connectivitat entre les màquines.

4.4.2 Anàlisi i selecció del SGBD:

1. Investiga les diferents versions disponibles de MySQL Server i selecciona la més adequada segons els requeriments del sistema i les característiques necessàries.
2. Justifica la teva elecció en base a factors com estabilitat, compatibilitat i funcionalitats.

4.4.3 Preparació del sistema:

1. Verifica els requisits hardware i software necessaris per a la instal·lació de MySQL Server.

4.4.4 Instal·lació de MySQL Server:

1. Instal·la MySQL Server en cada màquina virtual, en una per mig de apt o apt-get. en l'altra per mig de docker.
2. Inicia el servei de MySQL i verifica que s'està executant correctament.
3. Documenta el procés d'instal·lació, incloent les comandes utilitzades i captures de pantalla del terminal.

4.4.5 Configuració avançada del SGBD:

Canvi del port de MySQL:

1. En la maquina que has instal·lat per docker, canvia el port de MySQL a un valor diferent del port per defecte (3306).
2. Reinicia el servei i verifica que MySQL està escoltant al nou port.

Configuració de la connectivitat en xarxa:

1. Configura MySQL per permetre connexions remotes des de l'altra màquina.
2. Assegura't que el servidor està escoltant en totes les interfícies de xarxa (bind-address).
3. La maquina que has instal·lat per apt o apt-get, configura MySQL per NO permetre connexions remotes.
4. La maquina de docker, configura MySQL per permetre connexions remotes.

Utilització de l'script mysql_secure_installation :

1. Executa l'script mysql_secure_installation per millorar la seguretat de la instal·lació de MySQL en la màquina que has instal·lat per apt o apt-get.
2. Documenta les opcions seleccionades durant l'execució de l'script.

4.4.6 Gestió de comptes d'usuari i permisos:

1. Crea un usuari en cada servidor MySQL que tingui permís per connectar-se des de l'altra màquina.
2. Assigna els privilegis necessaris per accedir com a root.
3. Documenta els comandos SQL utilitzats per crear els usuaris i assignar els permisos.

4.4.7 Verificació de la connectivitat entre màquines:

1. Des de cada màquina, i desde el teu host, connecta't al servidor MySQL de l'altra màquina utilitzant l'usuari creat.

2. Utilitza les comandes necessàries per realitzar la connexió.
3. Executa consultes bàsiques per verificar l'accés a la base de dades.

4.5 Documentació del procés:

Elabora un document en format Markdown que inclogui tots els passos realitzats, les comandes utilitzades, captures de pantalla (o còpies del terminal) i explicacions clares. Inclou una secció de resolució d'incidències on expliquis els problemes trobats i com els has solucionat. Finalitza amb unes conclusions personals sobre l'experiència i els coneixements adquirits.

4.6 Materials necessaris

- Dues màquines virtuals amb sistema operatiu Linux (preferiblement Ubuntu Server).
- Programari de virtualització (VirtualBox, VMware, etc.).
- Connexió a Internet per descarregar els paquets i documentació necessària.
- Accés a una compte d'usuari amb permisos d'administrador (root) a les màquines Linux.
- Documentació oficial de MySQL i recursos del Tema 01.

4.7 Lliurament

4.7.1 Format:

Document en Markdown (.md) amb el desenvolupament complet de la pràctica.

4.7.2 Contingut del document:

- **Títol i introducció:** Nom de l'alumne, assignatura, títol de la pràctica i data.
- **Índex de continguts.**
- **Introducció:** Objectius de la pràctica i breu descripció del que es farà.
- **Desenvolupament:** Explicació detallada de cada tasca realitzada, amb comandes, captures de pantalla (inserides en el Markdown) i explicacions.
- **Resolució d'incidències:** Descripció dels problemes trobats i solucions aplicades.
- **Conclusions:** Reflexió personal sobre l'activitat i aprenentatges obtinguts.
- **Bibliografia i fonts consultades.**

4.7.3 Nom del fitxer:

Practica1_NomCognoms.md

4.7.4 Data de lliurament:

[Especificar data de lliurament]

4.8 Avaluació

L'avaluació es basarà en els següents aspectes:

4.8.1 Assoliment dels Resultats d'Aprenentatge i Criteris d'Avaluació:

RA1 i CA associats:

- Capacitat per reconèixer i explicar la utilitat i funció dels elements de MySQL Server.
- Anàlisi i justificació de la selecció de la versió de MySQL.
- Identificació correcta del programari i requisits necessaris.
- Realització efectiva de la instal·lació en múltiples màquines.
- Documentació detallada del procés d'instal·lació.
- Interpretació i resolució d'errors durant la instal·lació.
- Verificació del funcionament del SGBD en ambdues màquines.

RA2 i CA associats:

- Descripció clara de les condicions d'inici i parada del servei MySQL.
- Configuració adequada del port i connectivitat en xarxa.
- Assegurament efectiu dels comptes d'administració utilitzant `mysql_secure_installation`.
- Configuració correcta de les eines i programari client.
- Definició dels paràmetres relatius a les connexions.
- Documentació completa del procés de configuració.

RA3 (parcial) i CA associats:

- Creació i gestió de comptes d'usuari amb privilegis adequats per a l'accés remot.
- Aplicació de mesures per garantir la seguretat del sistema gestor.

4.8.2 Qualitat de la documentació:

- Claredat i organització del document en format Markdown.
- Precisió tècnica i ús correcte del llenguatge.
- Inclusió de captures de pantalla o còpies del terminal pertinents.
- Ús adequat de la sintaxi Markdown per a una presentació clara.

4.8.3 Resolució d'incidències:

- Capacitat per identificar problemes durant la instal·lació i configuració.
- Eficàcia en la recerca i aplicació de solucions.

4.8.4 Conclusions:

- Reflexió sobre els coneixements adquirits i dificultats superades.
- Valoració personal de l'experiència.

4.9 Orientacions per a l'alumne

4.9.1 Planificació:

- Organitza bé el temps per a realitzar la pràctica, deixant marge per a possibles incidències.
- Assegura't que les màquines virtuals tenen els recursos necessaris (memòria, disc dur, etc.).

4.9.2 Documentació:

- Documenta tots els passos que realitzis, encara que semblin senzills.
- Utilitza la sintaxi Markdown per estructurar el document (títols, subtítols, codi, llistes, etc.).
- Pots utilitzar eines com Visual Studio Code o Typora per editar Markdown.

4.9.3 Seguretat:

- Posa especial atenció en la configuració de la seguretat, especialment en obrir l'accés remot a MySQL.
- Assegura que les contrasenyes són robustes i que els usuaris tenen els privilegis mínims necessaris.

4.9.4 Proves i verificacions:

- Després de cada canvi, realitza proves per verificar que tot funciona correctament.
- Si tens dubtes durant la realització de la pràctica, consulta la documentació oficial o altres recursos fiables.

4.9.5 Revisió:

- Revisa el document abans de lliurar-lo, assegurant-te que està ben estructurat i que no conté errors ortogràfics o tècnics.
- Verifica que les captures de pantalla s'inclouen correctament en el Markdown.

Tasca \ RA	RA1	RA2	RA3
	a)	b)	c)
Tasca 1	100%		
Tasca 2		100%	100%
Tasca 3			
Tasca 4			
Tasca 5			
Tasca 6			
Tasca 7			
Tasca 8			

4.9.6 Llegenda:

RA1:

- CA a) Reconeixement de la utilitat i funció dels elements d'un SGBD.
- CA b) Anàlisi de les característiques dels principals SGBD.
- CA c) Selecció del SGBD adequat.
- CA d) Identificació del programari necessari per a la instal·lació.
- CA e) Verificació dels requisits hardware.
- CA f) Instal·lació del SGBD.
- CA g) Documentació del procés d'instal·lació.
- CA h) Interpretació dels missatges d'error i fitxers de registre.
- CA i) Resolució d'incidències durant la instal·lació.
- CA j) Verificació del funcionament del SGBD.

RA2:

- CA a) Descripció de les condicions d'inici i parada del SGBD.
- CA b) Selecció del motor de base de dades.
- CA c) Assegurament dels comptes d'administració.
- CA d) Configuració de les eines i programari client.
- CA e) Configuració de la connectivitat en xarxa.
- CA f) Definició de les característiques per defecte de les bases de dades.
- CA g) Definició dels paràmetres relatius a les connexions.
- CA h) Documentació del procés de configuració.

RA3 (parcial):

- CA c) Definició i eliminació de comptes d'usuari.
- CA h) Garantia del compliment dels requisits de seguretat.

4.9.7 Explicació de l'assignació de percentatges:

- **Criteris coberts per una sola tasca:** S'ha assignat el 100% a la tasca corresponent. Exemple: CA a) de RA1 és cobert únicament per la Tasca 1, per tant, s'assigna el 100% a aquesta tasca.
- **Criteris coberts per dues tasques:** S'ha dividit el percentatge entre les tasques que el cobreixen. Exemple: CA g) de RA1 és cobert per les Tasques 4 i 8; s'assigna el 50% a cadascuna.

4.9.8 Observacions:**RA1:**

- CA g), CA h) i CA i) són criteris coberts per les Tasques 4 i 8, per tant, s'ha assignat un 50% a cada tasca.
- La resta de criteris de RA1 són coberts per una única tasca i tenen un 100% assignat.

RA2:

- CA c) (Assegurament dels comptes d'administració) és cobert per les Tasques 5 i 6; s'assigna un 50% a cadascuna.
- CA e) (Configuració de la connectivitat en xarxa) és cobert per les Tasques 5 i 7; s'assigna un 50% a cadascuna.
- CA h) (Documentació del procés de configuració) és cobert únicament per la Tasca 8; s'assigna el 100%.

RA3:

- CA c) (Definició i eliminació de comptes d'usuari) és cobert únicament per la Tasca 6; s'assigna el 100%.
- CA h) (Garantia del compliment dels requisits de seguretat) és cobert per les Tasques 5 i 6; s'assigna un 50% a cadascuna.

4.9.9 Com utilitzar aquesta taula:

- **Avaluació dels criteris:** Aquesta taula permet veure quines tasques cobreixen cada criteri d'avaluació i amb quin percentatge. Això facilita l'avaluació del grau d'assoliment de cada criteri per part de l'alumne en funció de les tasques realitzades.
- **Planificació de l'estudi:** Els alumnes poden utilitzar aquesta taula per assegurar-se que, en realitzar les tasques assignades, estan cobrint tots els criteris d'avaluació necessaris per assolir els resultats d'aprenentatge.

5. UD2

5.1 Instalación MongoDB

5.1.1 Introducción

MongoDB es una base de datos NoSQL de código abierto, muy popular y ampliamente utilizada en aplicaciones modernas para almacenar datos en documentos JSON. MongoDB es una base de datos NoSQL de alto rendimiento y escalable que se puede utilizar para almacenar datos de cualquier tipo sin tener que definir la estructura de los datos de antemano.

5.1.2 Instalación utilizando binarios

Para instalar MongoDB en un sistema Linux, primero debemos descargar el archivo binario de MongoDB desde el sitio web oficial de MongoDB. A continuación, descomprimimos el archivo binario y movemos el directorio de MongoDB a la ubicación deseada.

```
1 wget https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-4.4.6.tgz
2 tar -xvzf mongodb-linux-x86_64-4.4.6.tgz
3 mv mongodb-linux-x86_64-4.4.6 /usr/local/mongodb
```

5.1.3 Instalación utilizando el administrador de paquetes

MongoDB también se puede instalar utilizando el administrador de paquetes de la distribución de Linux. A continuación se muestra cómo instalar MongoDB en Ubuntu utilizando el administrador de paquetes apt.

[Enlace documentación servidor](#) [Enlace documentacion cliente](#)

directorios de MongoDB

MongoDB utiliza varios directorios para almacenar datos, registros y archivos de configuración. A continuación se muestra una lista de los directorios utilizados por MongoDB:

Directorio	Descripción
/data/db	Directorio predeterminado para los datos de MongoDB
/var/log/mongodb	Directorio de registros de MongoDB
/etc/mongod.conf	Archivo de configuración de MongoDB
/usr/local/mongodb	Directorio de instalación de MongoDB

5.1.4 Iniciar el servicio de MongoDB

Para iniciar el servicio de MongoDB, primero debemos crear el directorio de datos de MongoDB y luego iniciar el servicio de MongoDB utilizando el comando mongod.

```
1 mkdir -p /data/db
2 /usr/local/mongodb/bin/mongod
```

También podemos iniciar el servicio de MongoDB utilizando el administrador de servicios systemctl.

```
1 sudo systemctl start mongod
```

5.1.5 Detener el servicio de MongoDB

Podemos detener el servicio de MongoDB utilizando el comando mongod --shutdown.

```
1 /usr/local/mongodb/bin/mongod --shutdown
```

Mediante systemctl

```
1 sudo systemctl stop mongodb
```

5.1.6 Conexión a MongoDB

Para conectarse a MongoDB, primero debemos iniciar el shell de MongoDB utilizando el comando mongo. A continuación, podemos conectarnos a la base de datos de MongoDB utilizando el comando use .

```
1 /usr/local/mongodb/bin/mongo
2 > use testdb
```

5.1.7 Crear usuario administrador

Para crear un usuario administrador en MongoDB, primero debemos conectarnos a la base de datos de administración y luego crear un usuario administrador utilizando el comando db.createUser().

```
1 /usr/local/mongodb/bin/mongo
2 > use admin
3 > db.createUser(
4 (
5   {
6     user: "admin",
7     pwd: "admin123",
8     roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
9   }
10 )
```

5.2 Imagen de MongoDB en docker

Para ejecutar una instancia de MongoDB en un contenedor de Docker, primero debemos descargar la imagen oficial de MongoDB desde Docker Hub y luego ejecutar el contenedor de MongoDB utilizando el comando docker run.

```
1 docker pull mongo
2 docker run -d -p 27017:27017 --name mongodb mongo
```

5.2.1 Persistencia de datos

Para persistir los datos de MongoDB en un contenedor de Docker, podemos montar un volumen de Docker en el directorio de datos de MongoDB utilizando la opción -v.

```
1 docker run -d -p 27017:27017 -v /data/db:/data/db --name mongodb mongo
```

5.2.2 Conexión a MongoDB en Docker

Para conectarse a la instancia de MongoDB en un contenedor de Docker, primero debemos iniciar el shell de MongoDB utilizando el comando docker exec y luego conectarnos a la base de datos de MongoDB utilizando el comando mongo.

```
1 docker exec -it mongodb mongo
2 > use testdb
```

5.3 Probar la conexión a MongoDB

Para probar la conexión a MongoDB, podemos crear una base de datos y una colección en MongoDB y luego insertar algunos documentos en la colección.

```
1 /usr/local/mongodb/bin/mongo
2 > use testdb
3 > db.createCollection("users")
4 > db.users.insert({name: "Alice", age: 30})
5 > db.users.insert({name: "Bob", age: 25})
6 > db.users.find()
```

5.4 Asegurar MongoDB

Para asegurar MongoDB, podemos habilitar la autenticación de MongoDB y crear usuarios con roles específicos en MongoDB.

```
1 /usr/local/mongodb/bin/mongo
2 > use admin
3 > db.createUser(
4   {
5     user
6     : "admin",
7     pwd: "admin123",
8     roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
9   }
10 )
11 > db.auth("admin", "admin123")
```

5.5 NoSql: MongoDB

[1. Introducció a NoSql.](#)

[Principis NoSql.](#)

[Tipus de NoSql.](#)

[Clau- Valor](#)

[Documentals.](#)

[Grafs.](#)

[NoSql o Relacional](#)

[Diferències entre NoSql i Relacional. Nomenclatura.](#)

[Diferències NoSql i relacional](#)

[Comparativa Relacional vs NoSql](#)

[Que ofereix NoSql](#)

[Quan triar NoSQL](#)

[Els entorns NoSql més usuals són:](#)

[Exemples de quan podem triar cada model:](#)

[2. MongoDB](#)

[Que ofereix MongoDB](#)

[JSON: JavaScript Object Notation](#)

[3. Modelat Mongo](#)

[Modelat de dades](#)

[Opcions de modelat](#)

[Embeure o referenciar](#)

[3. Disseny Físic. DDL Mongo](#)

[Crear una Bases de dades](#)

[Crear una col·lecció](#)

[Eliminar col·leccions i bases de dades](#)

[4. DML Bàsic Mongo](#)

[Consideracions iniciais](#)

5.6 1. Introducció a NoSql.

Que és NoSql, no es refereix a que no utilitze sql, sinó que significa NO SOLS SQL (NoSQL != Not Only SQL).

No es tant una cuestió de relacional o no relacional, sinó que estem parlant d'una estructura alternativa per tractar les bases de dades davant un problema de rendiment.

És mirar la vessant relacional d'una base de dades desde un altre punt de vista, amb una estructura més flexible y no tan llimitant com el relacional.

Els dos models apareixen en els 70-80, el relacional continua a dia de hui com el més popular, Sql ha començat a ser-ho en els últims anys. El NoSql no és un substitut del relacional, sinó més bé una alternativa en determinades circumstàncies.

Per què sorgeix? Apareix amb el Web 2.0, fins eixe moment sols les grans companyies pujaven contingut a internet. Però amb l'aparició de plataformes com Google, Amazon, Facebook, etc.. qualsevol usuari podia pujar contingut, el que provocà un increment de les dades.

És en eixe moment quan comencen a aparèixer els primers problemes de rendiment en els sistemes de bases de dades relacionals. En un primer moment s'opta per ampliar el HW, però era molt costós i no solucionava bé el problema. Al final opten per crear un sistema específic al problema, el NoSql.

NoSql permet superar els problemes d'escalabilitat i rendiment que es produueixen en els sistemes de bases de dades relacionals quan es troben grans volúmens de dades, gran quantitat d'usuaris en concorrència i milions de consultes diaries.

Estos sistemes NoSql estan optimitzats per a tasques de recuperació i agregació d'informació, la seu actualització no és una prioritat.

5.6.1 Principis NoSql.

NoSql es basa en 4 principis:

El control transaccional ACID no és important.

Els JOINS tampoc ho són. En especial els complexes i distribuïts.

Alguns elements relacionals són necessaris i aconsellables: claus (keys).

Gran capacitat d'escalabilitat i de replicació en múltiples servidors.

5.6.2 Tipus de NoSql.

Les bases de dades NoSql són sistemes d'emmagatzemament d'informació que no compleixen l'esquema Entitat-Reació, ni l'estructura de taula en la que s'emmagatzemen les dades. ESTos sistemes emmagatzemen la informació mitjançant:

- Clau - Valor
- Documents
- Mapeig de columnes
- Grafs

Les BD documentals, en el nostre cas MongoDB, els documents seran estructures JSON.

Les orientades a grafs, componen un conjunt d'objectes anomenats nodes connectats mitjançant una sèrie d'aristes. (neo4j)

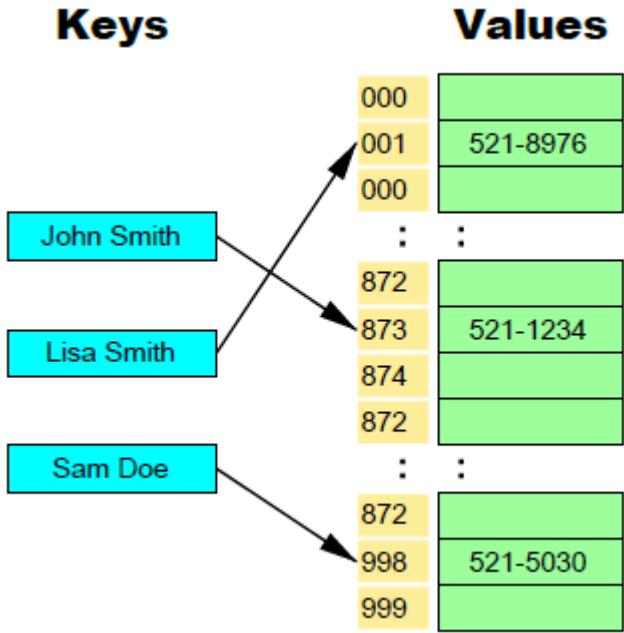
Les calu-valor, la base de dades emmagatzema en parells clau/valor. (Redis)

Clau- Valor

Són les habituals en els SGBD NoSQL perquè son les més senzilles en quant a funcionalitat.

En este sistema, cada element és identificat per una clau única, permetent així una recuperació ràpida. Són molt eficients en lectura i escriptura.

Per exemple Cassandra i Redis.



Documentals.

L'informació s'emmagaatzema com un document, utilitza una estructura simple com JSON o XML, en ell s'utilitza una clau única per cada registre.

A més de realitzar cerques per clau-valor, es poden fer cerques mes avançades per el contingut del document.

Per exemple MongoDB i CouchDB.

```
{
    Nombre: "Alberto",
    Dirección: "Castaños 17",
    Hijos: [
        {Nombre: "Andrés", Edad: 12},
        {Nombre: "Susana", Edad: 7},
        {Nombre: "Verónica", Edad: 4}
    ]
}
```

Grafs.

L'informació es representa com nodes d'un graf i les seus relacions com les aristes entre els nodes, de forma que per recorrer la informació es pot aplicar la teoria de grafs.

Permet mantindre el rendiment, sense importar el que creixca la base de dades.

5.6.3 NoSql o Relacional

El primer que s'ha de dir és que no és una decisió superimportant, podem dissenyar qualsevol cosa amb els dos models. Està clar que en determinades situacions ens facilita més la vida el tindre un model o un altre, però en altres en té exactament igual.

El que realment importa, es tria el model que es tria, és fer un bon disseny. Poc importa si encertem el model si després ho dissenyem malament.

5.6.4 Diferències entre NoSql i Relacional. Nomenclatura.

SQL	NoSQL
Base de dades	Base de dades
Taula	Col·lecció
Fila	Document
Columna	Camp
Estructura tabular	Objectes JSON

Imaginem que tenim una plataforma online de cursos d'idioma, i triem emprar un model relacional, però dissenyem mal i fem açò:

fecha	cliente	idioma	nivel	suscripción	precio	descuento_%	precio final
25/06/2018	Pedro	Inglés	Intermedio	Mensual	7	0	7
25/06/2018	Pedro	Chino	Principiante	Mensual	9	0	9
01/07/2018	Aurelia	Francés	Avanzado	Anual	8	25	6
03/07/2018	Federico	Inglés	Intermedio	Trimestral	7	10	6.3

Com veiem tenim els problemes de falta de normalització.

Ara normalitzem:

Nombre	Idioma	Nivel	precio
Pedro	Alemán	Principiante	7
Aurelia	Alemán	Intermedio	8
Federico	Inglés	Principiante	7
Tipo	Descuento		
Mensual	0		
Trimestral	10		
Anual	25		

Nombre	Idioma	Nivel	Tipo	Fecha
Pedro	Inglés	Intermedio	Mensual	25/06/2018
Pedro	Chino	Principiante	Mensual	25/06/2018
Aurelia	Francés	Avanzado	Anual	01/07/2018
Federico	Inglés	Intermedio	Trimestral	03/07/2018

En NoSql tenim l'alternativa desnormalitzada:

```
[{"Nombre": "Pedro",
  "cursos": [
    {"Idioma": "Inglés",
     "Nivel": "Intermedio",
     "Tipo": "Mensual",
     "Fecha": "25/06/2018"
    },
    {"Idioma": "Chino",
     "Nivel": "Principante",
     "Tipo": "Mensual",
     "Fecha": "25/06/2018"
    }
  ],
  {"Nombre": "Aurelia",
    "cursos": [
      {"Idioma": "Francés",
       "Nivel": "Avanzado",
       "Tipo": "Anual",
       "Fecha": "01/07/2018"
      }
    ],
    {"Nombre": "Federico",
      "cursos": [
        {"Idioma": "Inglés",
         "Nivel": "Intermedio",
         "Tipo": "Trimestral",
         "Fecha": "03/07/2018"
        }
      ]
    }
  ],
  {"Idioma": "Inglés",
   "Nivel": "Intermedio",
   "Tipo": "Mensual",
   "Alumnos": [
     {"Nombre": "Pedro",
      "Fecha": "25/06/2018"
     },
     {"Nombre": "Pablo",
      "Fecha": "28/06/2018"
     }
   ],
   {"Idioma": "Chino",
    "Tipo": "Mensual",
    "alumno": {"Nombre": "Pedro",
               "Fecha": "25/06/2018"
              }
   },
   {"Idioma": "Francés",
    "Nivel": "Avanzado",
    "Tipo": "Anual",
    "Alumnos": {"Nombre": "Aurelia",
                "Fecha": "01/07/2018"
               }
   },
   {"Idioma": "Inglés",
    "Nivel": "Intermedio",
    "Tipo": "Trimestral",
    "Alumno": {"Nombre": "Federico",
               "Fecha": "03/07/2018"
              }
   }
  ]
}
```

Dos implementacions vàlides.

5.6.5 Diferències NoSql i relacional

- No utilitzen el llenguatge SQL per les consultes.
- No utilitzen estructures d'emmagatzemament fixes per guardar la informació, fan ús de altres models, com sistemes clau-valor, grafs o objectes.
- No solen permetre operacions de JOIN. al tindre un volum de dades extremadament gran, resulta incòmode utilitzar JOIN atès que sobrecarrega el sistema. La solució a este problema pot ser la desnormalització o fer el JOIN per software en la capa d'aplicació.
- Arquitectura distribuida. Les bases de dades relacionals soLEN estar centralitzades en una màquina o amb estructura client-servidor. Però els sistemes NoSql podEN contindre la informació compartida entre varIES màquines mitjançant taules Hash distribuïdes.

5.6.6 Comparativa Relacional vs NoSql

Ventajas Relacional	Ventajas NoSql	Desventajas Relacional	Desventajas NoSql
Madurez	Versatilidad y manejo de grandes cantidades de datos (Estructura distribuida)	Mantenimiento complejo y lentitud con la integridad referencial	ACID. <ul style="list-style-type: none"> Consistencia eventual, podrían leerse datos de nodos secundarios que aún no están actualizados. Sin soporte transaccional.
ACID	Crecimientos horizontal	Poca flexibilidad frente a cambios del diseño original	Documentación
Estandares bien definidos (SQL)	Baja necesidad de recursos e instalación	Complejidad e instalación	No hay estándar
Sencillez de estructura	Flexibilidad de estructuras	Estructuras fijas y con muchas validaciones.	Dificultad para consultas complejas.
Aplicaciones clientes más sencillas, el modelo relacional lo da todo hecho.	Fácil de integrar con aplicaciones usando BSON y JSON.	Necesidad de APIs de conexión, ODBC, JDBC, OCI...	Aplicaciones clientes más complejas de desarrollar al trabajar con esquemas flexibles, desnormalizados y dinámicos

5.6.7 Que ofereix NoSql

- Són ideones si les dades no segueixen una estructura fixa.
- No garantzen els principis ACID
- EL model és flexible. Milloren la flexibilitat i escalabilitat respecte de les BD SQL. Les dades que utilitzen les nostres aplicacions web, mòbils o socials es veuen constantment sota canvís.
- Versatilitat. La modificació és més senzilla que en els models relacionals. A l'hora de modificar o introduir nous tipus de dades no tenim la necessitat de modificar per complet l'estructura de la base de dades.
- Alt rendiment. La velocitat és un dels factors més importants.

5.6.8 Quan triar NoSQL

- Quan l'escalabilitat en un model relacional no és viable o molt costós, tant a nivell tècnic com de costos.
- Quan el sistema té pics d'ús molt elevats per part dels usuaris.
- Quan el nombre de dades creix ràpidament en situacions puntuals, arribant a superar el Terabyte d'informació.
- Quan la informació no és homogènia, és a dir, quan en cada inserció podem tindre diferents camps.
- Quan ens interessa més la velocitat de les lectures que la seguretat de l'escriptura.
- Quan no tenim la necessitat d'una gran normalització de les dades i podem tindre-ho desnormalizado.

Els entorns NoSql més usuals són:

- Xarxes socials: quasi obligatori.

- Desenvolupament Web: a causa de la poca uniformitat de la informació que es troba en Internet; tot i que també pot emprar-se SQL.
- Desenvolupament Mòbil: a causa de la tendència (en creixement) de Bring Your Own Device.
- BigData: a causa de l'administració de grandíssimes quantitats d'informació i la seua evident heterogeneïtat.
- Cloud (XaaS): "Everything as a service"; NoSQL pot adaptar-se quasi a qualsevol necessitat del client, i les seues particularitats.

Exemples de quan podem triar cada model:

- Quan el nostre pressupost no es pot permetre grans màquines i ha de destinarse a màquines de menor rendiment, **NoSQL**.
- Quan les estructures de dades que manegem són variables, o no la tenim clara inicialment **NoSQL**.
- Anàlisi de grans quantitats de dades en manera lectura, **NoSQL**.
- Captura i processament d'esdeveniments, **NoSQL**.
- Botigues en línia amb motors d'intel·ligència complexos, **NoSQL**.
- Quan les dades han de ser consistents sense donar possibilitat a l'error utilitzar una base de dades relacional, **SQL**
- En quasi tots els altres casos, **SQL**.

5.6.9

En NoSQL, les dades es recuperen, en general, de forma més ràpida que el Model Relacional, no obstant això les consultes que es poden realitzar són més limitades. La complexitat es trasllada a l'aplicació.

Si la teua aplicació requereix suport transaccional, has d'usar un model Relacional, no inventes la roda.

NoSQL no és adequat per a aplicacions que generen informes amb consultes complexes (necessitat de JOINs), encara que existeixen operacions en NoSQL que permet paralelitzar operacions complexes com a agregacions, filtres, etc..

La tendència actual és cap a la combinació de sistemes SQL i NoSQL

5.7 2. MongoDB

MongoDB és un gestor de dades estructurat sota NoSQL que compta amb un format d'emmagatzematge de documents en un format similar al JSON (JavaScript Object Notation). Està escrit en llenguatge C++, és multiplataforma en codi obert i completament gratuït. Els seus orígens es remunten a la fi de l'any 2007 quan era un projecte intern d'una empresa de nom 10Gen. Mongo ve de la paraula anglesa «humongous», que significa «enorme, extraordinàriament deixe anar». El seu disseny va ser desenvolupat per a ser implementat en una aplicació d'internet que l'empresa desenvolupava i no va ser fins a l'any 2009 que decideixen alliberar-lo en format OpenSource.

MongoDB compta amb un emmagatzematge flexible basat en JSON. A més té suport per a la creació d'índexs des de qualsevol atribut. També és important destacar que té un alt rendiment per a consultes i actualitzacions. És increïblement flexible i potent la seua capacitat de creixement, replicació i escalabilitat.

MongoDB és ideal per a crear aplicacions en línia que requerisquen el registre de diverses dades. O per a la creació i disseny de programari que siguen flexibles quant a tractament d'alts volums d'informació. Les Bases construïdes sota aquest disseny poden rebre centenars de milers de lectures per segon sense que el seu rendiment es veja afectat o disminuït.

5.7.1 Que ofereix MongoDB

- Orientada a documents (JSON).
- No segueix cap esquema (schemaless).
- Disposa d'una consola (shell) construïda sobre JavaScript el que permet executar moltes de les seues funcions.
- Àmpliament utilitzada en aplicacions per a Internet i de BigData.
- Escriptura ràpida: Les bases de dades documentals estableixen un ordre de priorització de la disponibilitat d'escriptura per damunt de l'estreta consistència de les dades.

- Llenguatge de Consultes Avançat. Els seus mètodes d'agregació per a bigdata és el millor que hi ha.
- Alta disponibilitat: Replicació dels set i balanceig de càrrega és molt senzill i es fa en 4 passos.
- Escalabilitat horitzontal: Alçar un clúster és senzill.
- Consola JS: Es treballa en aquest llenguatge per als comandos de Mongo.

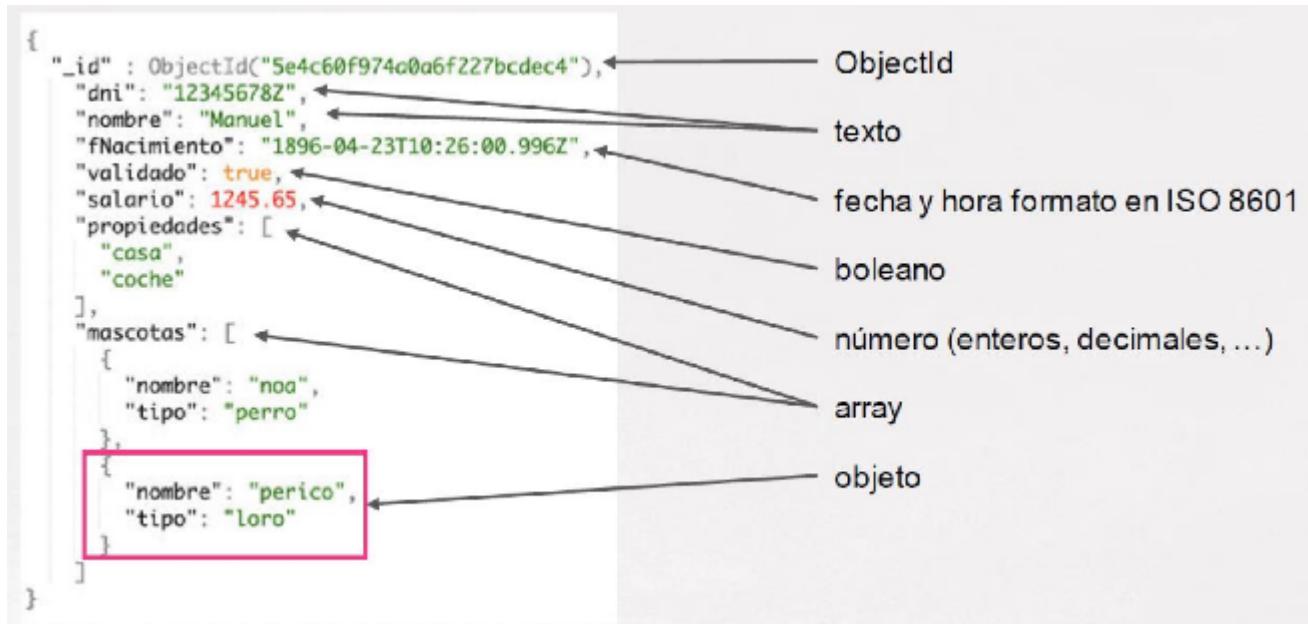
5.7.2 JSON: JavaScript Object Notation

És un format de text senzill per a l'intercanvi de dades. Presenta un alternaiva a XML perque és més senzill de llegir i escriure. A més a més, esta suportat per un gran nombre de llenguatges.

Un Objecte JSON està format per un o varios parells string:valor.

La sintaxi en JSON és molt senzilla, partim de la dupla identificador:valor i segons l'estructura formarem variables, arrays, etc.

- Creació d'arrays amb claudàtors ([]).
- Creació d'objectes amb claus ({}).
- Separació del nom i/o assignació de valor amb dos punts (:).
- Separació de valors amb coma (,).



Exemple detallat de document JSON

5.8 3. Modelat Mongo

5.8.1 Modelat de dades

El model NoSQL és àgil i flexible i val tot. Com ja hem vist, accepta qualsevol cosa i qualsevol estructura. Podem tindre un E-R de partida i el seu disseny conceptual (traducció), és el més recomanable, encara que si tenim un model gran amb moltes relacions, és una pista que potser hauríem d'usar el model relacional i no el NoSql. O també no tindre cap model i anar afegint estructures sobre la marxa, generalment quan ens decantem per NoSQL és perquè no hi ha moltes entitats, ni les relacions són massa importants. Al final sempre acabarem dissenyant un model quan l'anem tenint clar, però tenim la flexibilitat de no requerir un model predefinit, almenys al principi.

5.8.2 Opcions de modelat

En Mongo disposem de dues estructures diferents per a dissenyar relacions en les nostres col·leccions:

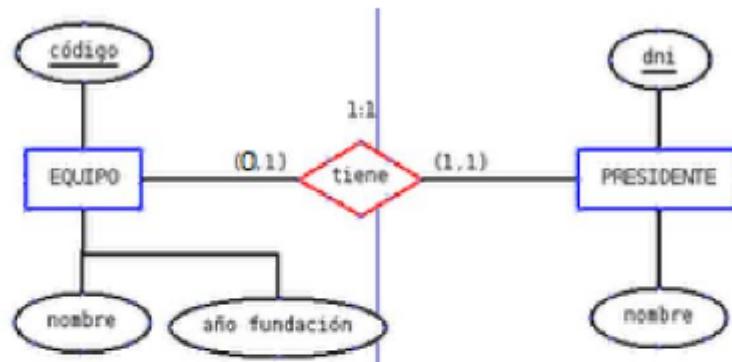
Embeigudes: El que anomenaríem "entitats" en el disseny conceptual, en aquest cas tindríem una entitat dins d'una altra formant 1 sola col·lecció. El que podria dir-se subdocuments.

Referenciades: Quan ens interessa tindre les "entitats" separades i llavors referenciem d'una forma semblant a com ho fem en el model relacional, amb unes foreign key simulades. En aquest cas haurem de tindre alguna cosa semblant a una clau primària (per defecte mongo té el camp `_id`).

NoSQL està pensat per a embeure, però tampoc podem crear un document infinit ni repetitiu, a vegades ens convindrà referenciar, simulant les famoses FK (encara que moltíssim menys restrictives), i com veurem les col·leccions poden tindre PK i UK.

- Triar entre embeure o referenciar depen totalment de l'aplicació:
- Com treballarem amb les dades
- Com preveiem que vages a evolucionar
- Com creixen
- Quan s'accedeix a ells i a quins exactament
- Com volem emmagatzemar les dades
-

Relación 1:1



Relacional

EQUIPO (codigo,nombre,año_fundación,dni_presidente)
PK: (codigo)
UK: (dni_presidente)
FK: (dni_presidente → EQUIPO)
PRESIDENTE (dni,nombre)
PK: (dni)

NoSQL

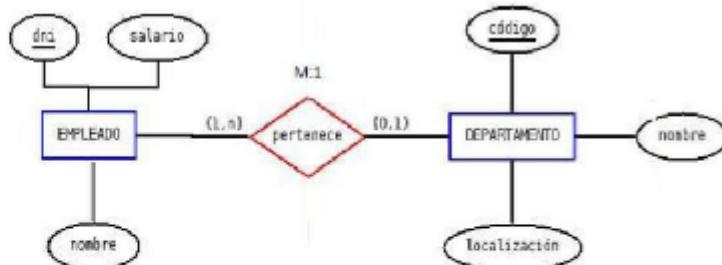
```

EQUIPO={
    codigo: "ESP",
    nombre: "ESPAÑA",
    anyo_fundacion: 1920,
    presidente:{
        dni: "111C",
        nombre: "Luis Rubiales"
    }
}
    
```

Relación 1:M

Relacional

EMPLEADO(dni,nombre,salario,codigo_departamento)
PK: (dni)
FK: Código_departamento → DEPARTAMENTO
DEPARTAMENTO(codigo,nombre,localización)
PK: (codigo_nombre)



NoSQL Embebido

```

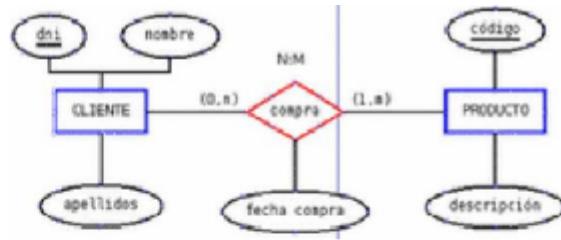
Departamento-{
    codigo:"Inf",
    nombre:"Dpto Informática",
    localización:"Alicante",
    telefones:[1234,5678],
    empleado:[
        {
            dni:"111C",
            nombre:"Pablo Garramone",
            salario:1000
        },
        {
            dni:"222B",
            nombre:"Diego",
            salario:2000
        },
        {
            dni:"333B",
            nombre:"Javi",
            salario:3000
        }
    ]
}
    
```

NoSQL Referenciado

```

Departamento-{
    codigo:"Inf",
    nombre:"Dpto Informática",
    localización:"Alicante",
    telefones:[1234,5678],
    empleado:[{"dni": "111C", "nombre": "Pablo Garramone", "salario": 1000, "cod_dep": "Inf"}, {"dni": "222B", "nombre": "Diego", "salario": 2000, "cod_dep": "Inf"}, {"dni": "333B", "nombre": "Javi", "salario": 3000, "cod_dep": "Inf"}]
}
    
```

Relación M:M



Relacional

CLIENTE (dni, nombre, apellidos)
 CP: (dni)
PRODUCTO (codigo, descripción)
 CP: (codigo)
COMPRA (dni_cliente, codigo_producto, fecha_compra)
 CP (dni_cliente, codigo_producto)
 CAj: dni_cliente -> CLIENTE
 CAj: codigo -> PRODUCTO

NoSQL

```

Cliente={
  dni:"111C",
  nombre:"Pablo",
  apellidos:"Garramone Ramirez",
  productos:[
    {cod_prod:1,fecha:"12/12/2020"}, 
    {cod_prod:2,fecha:"12/11/2020"}, 
    {cod_prod:3,fecha:"12/10/2020"}
  ]
}

Producto={
  codigo:1,
  descripcion:"Manzana",
  clientes:["111C","223B"]
}
  
```

5.8.3 Embeure o referenciar

- En les relacions 1:1 interessa embeure, ja que d'aquesta manera tenim tota la informació en la mateixa col·lecció el que facilita la seu consulta.
- En una relació 1:M, dependrà de la relació:
 - Si l'element de l'1 no es relaciona amb molts del costat M, ens pot interessar embeure ja que sempre facilitarà la consulta.
 - Però si es pot relacionar amb molts (1 milió per exemple) tindrem dos problemes, que si es donen serà millor referenciar:
 - De grandària: Cal anar amb compte perquè Mongo té una limitació de 16Mb per cada document i no podrem inserir-lo.
 - De cerca, si tenim tants subelements, i hem de fer una consulta i just busquem l'últim no serà gens òptim, en aqueix cas ens interessarà referenciar.
 - Si requerim moltes consultes sobre els subelements exclusivament, pot interessar-nos tindre-ho en una col·lecció solta, per la qual cosa llavors serà millor referenciar.
- En les M:M, normalment serà convenient referenciar, perquè si tenim que es relaciona 1 document amb molts de l'altra col·lecció, aquest es repetirà en cadascun dels documents relacionats el que generarà una tremenda redundància.
- Per norma general hem d'embeure, de manera que tinguem el màxim d'informació possible dins del mateix document, evitant "JOINS", ja que en Mongo l'atomicidad és per document, de manera que els JOINS no són trivials.
- Si el subdocument embegut es repeteix moltíssimes vegades o realitzem cerques exclusives sobre ell, llavors és millor referenciar.

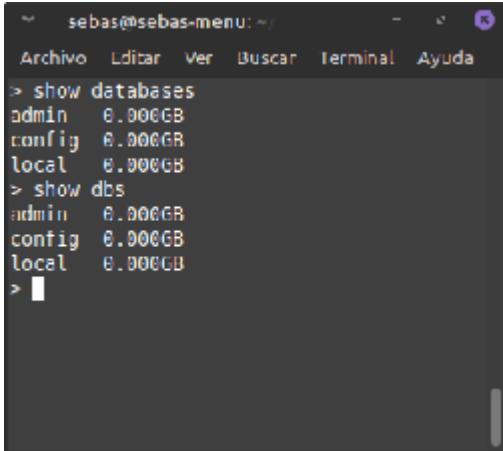
- Si la llista de subdocuments creixerà indiscriminadament, és millor referenciar (tindre en compte la limitació dels 16Mb) ja que si es nia massa informació la consulta és ineficient.

Embeure	Referenciar
Xicotets subdocuments	Grans subdocuments
Lectura ràpida. TEnim tota la informació en un sol document	Lectura òptima, si embeguts hi haguera molts subdocuments. Inconvenient del JOIN
Documents que creixen en xicotetes quantitats o no canvien regularment	Documents que creixen en gran quantitat.
Escriptures òptimes si no s'actualitza informació relacionada.	Escriptures ràpides. En actualitzar informació relacionada, es fa directament sense haver de relacionar la col·lecció externa.

5.9 3. Disseny Físic. DDL Mongo

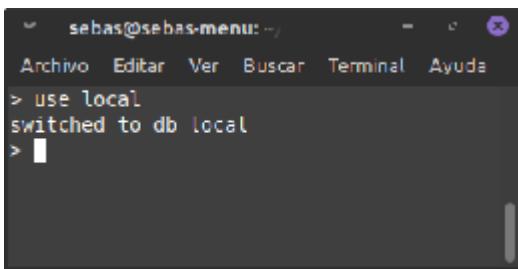
Primers comandaments que podem provar:

- show dbs / show databases : ens mostra totes les bases de dades que tenim en mongo



```
sebas@sebas-menu: ~
Archivo Editar Ver Buscar Terminal Ayuda
> show databases
admin 0.000GB
config 0.000GB
local 0.000GB
> show dbs
admin 0.000GB
config 0.000GB
local 0.000GB
> |
```

- use <basededades> : per accedir a una base de dades.



```
sebas@sebas-menu: ~
Archivo Editar Ver Buscar Terminal Ayuda
> use local
switched to db local
> |
```

- show collections : per vore les col·leccions que tenim a una BD.

```
sebas@sebas-menu: ~
Archivo Editar Ver Buscar Terminal Ayuda
> show collections
startup_log
> [ ]
```

- db.stats() : ens dona els detalls i estadístiques generals de la base de dades.

```
sebas@sebas-menu: ~/D...
Archivo Editar Ver Buscar Terminal Ayuda
> use nova
switched to db nova
> db.stats()
{
    "db" : "nova",
    "collections" : 1,
    "views" : 0,
    "objects" : 1,
    "avgObjSize" : 92,
    "dataSize" : 92,
    "storageSize" : 20480,
    "indexes" : 1,
    "indexSize" : 20480,
    "totalSize" : 40960,
    "scaleFactor" : 1,
    "fsUsedSize" : 151588634624,
    "fsTotalSize" : 195785797632,
    "ok" : 1
}
> [ ]
```

- db.help() : proporciona ajuda dels comandaments de mongo.
- db : indica la base de dades actual.

Crear una Bases de dades {#crear-una-bases-de-dades}

No hi ha cap mètode específic per crear una base de dades, en el moment que creem una col·lecció automàticament es crea la base de dades.

- Primer accedim a la base de dades nova (encara que no existixca)

```
sebas@sebas-menu: ~/
Archivo Editar Ver Buscar Terminal Ayuda
> use nova
switched to db nova
> 
```

- Una vegada dins hem de crear una col·lecció. Atenció!, amb “use” encara no s’ha creat res, sinó creem la col·lecció la base de dades no s’haurà creat.

5.9.1 Crear una col·lecció

Tenim dues opcions per crear les col·leccions:

- db.createCollection(<nomColeccio>, [opcions]) : ens permet crear una col·lecció utilitzant les distintes opcions de configuració. Esta opció és interessant quan el projecte està clar, definit i estan clares les llimitacions que anem a posar-li.

```
sebas@sebas-menu: ~/
Archivo Editar Ver Buscar Terminal Ayuda
> db.createCollection("persones")
{ "ok" : 1 }
> 
```

- db.<nomColeccio>.insert(<JSON>) : li diguem a mongo que faça una inserció en la col·lecció i si mongo detecta que no existeix la crea. Esta es l’opció més habitual quan comencem i encara no tenim clares les limitacions que anem a donar-li a la col·lecció.

```
sebas@sebas-menu: ~/Documentos/bbdd/mongo
Archivo Editar Ver Buscar Terminal Ayuda
db.coches.insertOne({matricula:"2966bfl",marca:"ford", model:"fiesta", kms: 30000})
{
    "acknowledged" : true,
    "insertedId" : ObjectId("6272a9aebc8146d8f27504e2")
}
> 
```

Eliminar col·leccions i bases de dades {#eliminar-col·leccions-i-bases-de-dades}

- db.<nomColeccio>.drop() : elimina la col·lecció i retorna true o false si s’ha eliminat correctament o ha ocorregut algun problema.
- db.dropDatabase() : elimina la base de dades en al que estem.

4. DML Bàsic Mongo {#4.-dml-bàsic-mongo}

5.9.2 Consideracions inicials

- Hem de tindre en compte que en Mongo tot són documents JSON.
- Pel que tindrem SEMPRE, camp:valor. I els valors SEMPRE entre cometes.
- En els elements de Mongo es posarà un \$ davant.
- En ocasions quan vulguem accedir a elements dels subdocuments haurem de posar-los entre cometes i fins i tot en funcions complexes amb \$.
- Qualsevol element tindrà {} i tindrem alguns amb array []
- Totes les operacions comencen per db, que indica data base (base de dades) i serà l'última que haurem posat USE.

5.9.3 Operacions bàsiques. CRUD

- INSERIR: insert / insertOne / insertMany
- ESBORRAR: remove / deleteOne / deleteMany
- MODIFICAR: update/ updateOne / updateMany /replaceOne / save
- CONSULTAR: find i les seues variants.

5.9.4 Operacions d'inserció

Tenim 3 mètodes d'inserir:

- insert(JSON): Pots inserir un o diversos documents alhora. Està obsolet en alguns drivers.
- insertOne(JSON): És la nova versió per a quan només volem inserir 1 document.
- insertMany([JSON]): És la nova versió per a quan volem inserir més d'1 document.

Sintaxis: db.\<colecció>.insert(\<JSON>)

Tenim dos formes d'insertar:

Definit en una variable Javascript

```
sebas@sebas-menu: ~/Documentos/bbdd/mongo/compas
Archivo Editor Ver Buscar Terminal Ayuda
> let coche = {matricula:"5467GCC",marca:"citroen",model:"c2", kms:3200}
> db.coches.insertOne(coche)
{
  "acknowledged" : true,
  "insertedId" : ObjectId("6274cccd142116fae9c99ccb4")
}
>
```

Directament:

```
sebas@sebas-menu: ~/Documentos/bbdd/mongo
Archivo Editor Ver Buscar Terminal Ayuda
db.coches.insertOne({matricula:"2966bfl",marca:"ford", model:"fiesta", kms: 30000})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("6272a9aebe8146d8f27504e2")
}
>
```

Insertar varies:

db.\<colecció>.insertMany([{document1},{document2}...])

```
sebas@sebas-mono: ~/Documentos/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.insertMany([{matricula:'9999nnn', marca:'seat', model:'ibiza', kms:090},{matricula:'0000ppp', marca:'peugeot',model:'2000', kms:23110}]);
```

Recordar que Mongo és atòmic per document, si alguna de les Insert falla, les que s'hagen realitzat anteriorment S'INSEREIXEN!, no va a agafar les posteriors a la fallada.

5.9.5 Operacions de borrat

Per eliminar tenim opcions com en insert:

- deleteOne(filtre), sols borra el primer document que trobe que complisca el filtre.
- deleteMany(filtre), borra tots els documents que complisquen el filtre.

```
sebas@sebas-mono: ~/Documentos/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.find({marca:'ford'})
{ '_id' : ObjectId('6272a9ae0c8146d8f27504e2') , 'matricula' : '2955bf1' , 'marca' : 'ford' , 'model' : 'fiesta' , 'kms' : 30000 }
> db.coches.deleteOne({marca:'ford'})
{ 'acknowledged' : true , 'deletedCount' : 1 }
> db.coches.find({marca:'ford'})
```

5.9.6 Operacions de modificació

- db.\<colecció>.updateOne(filtre, accio,opcions)
- db.\<colecció>.updateMany(filtre, accio,opcions)
- db.\<colecció>.replaceOne(filtre, canvi, opcions), canvia un document per un altre.

filtre: és com el where.

acció: les accions que anem a realitzar per fer actualitzacions.

opcions: Opcions alternatives que podem configurar per realitzar coses extres en l'operació.

Accions

- \$set: Definir la clau que s'actualitzarà (o crearà si no existira). El més habitual.
- \$inc: Incrementar el valor d'un camp numèric.
- \$rename: Canviar de nom el nom d'una clau.
- \$unset: Definir la clau del document que s'eliminara.
- \$currentDate: Actualitza un camp a la data actual.
- \$, \$[...], \$[\<id>]: Marques d'elements
- \$pull: Eliminar els valors d'un array que complisquen el filtre.
- \$pullAll: Elimina els valors específicats d'un array.
- \$pop: Elimina el primer o últim valor d'un array.
- \$push: Afig un element a un array.
- \$position: S'usa amb push per a indicar la posició exacta on afegir elements.

- `$slice`: S'usa amb `push` per a limitar la grandària final de l'array.
- `$sort`: S'usa amb `push` per a reordenar els elements afegits.
- `$addToSet`: Afig elements a un array si no existeixen.
- `$each`: S'empra amb `$addToSet` i `$push` per a permetre l'agregació múltiple d'elements a un array.

Exemple d'update:

```
sebas@sebas-menus: ~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.find({marca:'alfa'})
{ "_id" : ObjectId("620e14bce719e3642c0b2d2c"), "matricula" : "5550nnn", "marca" : "alfa", "model" : "rom", "kms" : 890 }
> db.coches.update({marca:'alfa'}, { $set:{matricula:'5555nnn',model:'romeo'}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.coches.find({marca:'alfa'})
{ "_id" : ObjectId("620e14bce719e3642c0b2d2c"), "matricula" : "5555nnn", "marca" : "alfa", "model" : "romeo", "kms" : 890 }
> 
```

Cap remarcar que si fem un update d'un document que té un objecte embedut, cal posar totes les dades de dit objecte, sinó, es perdren els camps de l'objecte.

Ací tenim un exemple de com es perd la informació del propietari que no hem modificat.

```
sebas@sebas-menus: ~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.insert({matricula:"7700nnn", marca:"saab", model:"93", kms:890, propietari:{nom:"sebas", dni:"1234567a", carrer:"muro", poblacio:"tavernes"}});
WriteResult({ "nInserted" : 1 })
> db.coches.find({marca:'saab'}).pretty();
{
  "_id" : ObjectId("620f372c12ed9a6002faf4e3"),
  "matricula" : "7700nnn",
  "marca" : "saab",
  "model" : "93",
  "kms" : 890,
  "propietari" : {
    "nom" : "sebas",
    "dni" : "1234567a",
    "carrer" : "muro",
    "poblacio" : "tavernes"
  }
}
> db.coches.update({marca:'saab'}, {$set:{model:95,kms:99,propietari:{nom:"sebastian"}}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.coches.find({marca:'saab'}).pretty();
{
  "_id" : ObjectId("620f372c12ed9a6002faf4e3"),
  "matricula" : "7700nnn",
  "marca" : "saab",
  "model" : "95",
  "kms" : 99,
  "propietari" : {
    "nom" : "sebastian"
  }
}
> 
```

Exemple de update amb \$inc per augmentar un valor numèric

```
~ sebas@sebas-menu:~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.find({marca:'saab'}).pretty();
{
  "_id" : ObjectId("628f372c12ed9a6002faf1e3"),
  "matricula" : "7780nnn",
  "marca" : "saab",
  "model" : 95,
  "kms" : 99,
  "propietari" : {
    "nom" : "sebastian"
  }
}
> db.coches.update({marca:'saab'},{$inc:{kms:1}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.coches.find({marca:'saab'}).pretty();
{
  "_id" : ObjectId("628f372c12ed9a6002faf1e3"),
  "matricula" : "7780nnn",
  "marca" : "saab",
  "model" : 95,
  "kms" : 100,
  "propietari" : {
    "nom" : "sebastian"
  }
}
>
```

Exemple de update amb \$rename, on veurem que es canvia el nom del camp, però es recolocarà al final del document.

```
~ sebas@sebas-menu:~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.insert({matricula:"7780nnn", marca:"lancia",estat:"nou",model:"epsilon", kms:898,propietari:{nom:"ana", dni:"2345678b", carrer:"sequers",poblacio:"tavernes"}});
WriteResult({ "nInserted" : 1 })
> db.coches.find({marca:'lancia'}).pretty();
{
  "_id" : ObjectId("628f39eb12ed9a6002faf1e4"),
  "matricula" : "7780nnn",
  "marca" : "lancia",
  "estat" : "nou",
  "model" : "epsilon",
  "kms" : 898,
  "propietari" : {
    "nom" : "ana",
    "dni" : "2345678b",
    "carrer" : "sequers",
    "poblacio" : "tavernes"
  }
}
> db.coches.update({marca:'lancia'},[$rename:{estat:'conservacio'}]);
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.coches.find({marca:'lancia'}).pretty();
{
  "_id" : ObjectId("628f39eb12ed9a6002faf1e4"),
  "matricula" : "7780nnn",
  "marca" : "lancia",
  "model" : "epsilon",
  "kms" : 898,
  "propietari" : {
    "nom" : "ana",
    "dni" : "2345678b",
    "carrer" : "sequers",
    "poblacio" : "tavernes"
  },
  "conservacio" : "nou"
}
```

Per suposat podem canviar el nom del camp en tots els documents, en aquest cas sols un document tenia aquest camp.

Exemple de update amb \$unset, on eliminem un camp d'un document

```
sebas@sebas-menu:~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.find({marca:'lancia'}).pretty();
{
  "_id" : ObjectId("628f39eb12ed9a6002fa14e4"),
  "matricula" : "7700nnn",
  "marca" : "lancia",
  "model" : "epsilon",
  "kms" : 800,
  "propietari" : {
    "nom" : "ana",
    "dni" : "2345678b",
    "carrer" : "sequers",
    "poblacio" : "tavernes"
  },
  "conservacio" : "nou"
}
> db.coches.update({marca:'lancia'},{$unset:{'conservacio':''}});
WriteResult({ "nMatched": 1, "nUpserted": 0, "nModified": 1 })
> db.coches.find({marca:'lancia'}).pretty();
{
  "_id" : ObjectId("628f39eb12ed9a6002fa14e4"),
  "matricula" : "7700nnn",
  "marca" : "lancia",
  "model" : "epsilon",
  "kms" : 800,
  "propietari" : {
    "nom" : "ana",
    "dni" : "2345678b",
    "carrer" : "sequers",
    "poblacio" : "tavernes"
  }
}
>
```

Opcions

```
db.collection.update(
  <query>,
  <update>,
  {
    upsert: <boolean>,
    multi: <boolean>,
    writeConcern: <document>,
    collation: <document>,
    arrayFilters: [ <filterdocument1>, ... ],
    hint: <document|string>           // Available starting in MongoDB 4.2
  }
)
```

- No posar cap: Recordar que posar opcions és optatiu.
- multi: Booleà, per defecte false. L'operació update només modifica el primer document que complisca el "where". Si posem aquest booleà a true, s'actualitzaren tots els documents que ho complisquen (farà com un updateMany).
- upsert: Booleà, per defecte false. Si un update no compleix el "where", l'operació no fa res, si posem aquest booleà a true, l'inserida com un nou document.
- hint: Per a forçar que utilitze un índex determinat, si l'índex que indiquem no existeix, donarà error.

MODIFICAR. REPLACEONE

A diferència d'update, que només es modifiquen o afügen els elements que posem en l'apartat d'acció, amb replaceOne substituïm el document sencer pel que posem.

Amb aquesta operació evidentment només es pot anar d'1 en 1

```

sebas@sebas-menu:~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.insert({matricula:"7788nnn", marca:"talbot", model:"gran", kms:890});
WriteResult({ nInserted: 1 })
> db.coches.replaceOne({marca:"talbot"}, {"marca": "talbot", model:"vell"});
{ acknowledged : true, matchedCount : 1, modifiedCount : 1 }
> db.coches.find({marca:"talbot"}).pretty();
{
  "_id" : ObjectId("620f3d2112ed9a6002faf4e5"),
  "marca" : "talbot",
  "model" : "vell"
}
>

```

5.9.7 Operacions de consulta

Per realitzar consultes s'utilitza l'operador FIND. És important recordar que FIND busca DOCUMENTS, no subdocuments concrets.

La sintaxi:

db.<colecció>.find(<filtrat o where> ,<projecció>).<operacions especials>

No hi ha cap filtrat obligatori, si no posem res obtindrem tots els documents de la col·lecció.

- Les condicions en Mongo van en moltes operacions (find, aggregate, update...)
- \$type: Per a consultar per la tipus de dada.
- \$exists: Per a consultar si el camp existeix.
- \$regex: Per a indicar una expressió regular a complir.
- \$and, \$nor, \$or i \$not: → Operadors booleans
- \$in, \$nin: → contingut en, no contingut en
- \$eq: → true si els valors són equivalents
- \$gt, \$gte, \$lt, \$lte, \$ne: → Major que, major que o igual...
- \$where: Satisfet una expressió en javascript
- \$all: Arrays que complisquen que tinguen tots els elements.
- \$elementMatch: Especificacions a complir en Arrays
- \$size: Indicar la grandària de l'array.

```

sebas@sebas-menu:~/OneDrive/bbdd/mongo/compas
Archivo Editar Ver Buscar Terminal Ayuda
> db.coches.find({marca:'tunca10'}).pretty();
[
  {
    "_id" : ObjectId("620f3d2112ed9a6002fa14e4"),
    "marca" : "tunca10",
    "model" : "tunca10",
    "kms" : 0,
    "conservacio" : 1,
    "propietari" : "ANA",
    "telefono" : "934567890",
    "telefono2" : "934567890",
    "telefono3" : "934567890",
    "correo" : "ana@tunca10.com"
  }
]
> db.coches.update({marca:'tunca10'}, {$set:{conservacio:'1'}});
WriteResult({ nModified: 1, nMatched: 1, nAffected: 1 })
> db.coches.find({marca:'tunca10'}).pretty();
[
  {
    "_id" : ObjectId("620f3d2112ed9a6002fa14e4"),
    "marca" : "tunca10",
    "model" : "tunca10",
    "kms" : 0,
    "conservacio" : 1,
    "propietari" : "ANA",
    "telefono" : "934567890",
    "telefono2" : "934567890",
    "telefono3" : "934567890",
    "correo" : "ana@tunca10.com"
  }
]
> db.coches.update({marca:'tunca10'}, {$set:{conservacio:'0'}});
WriteResult({ nModified: 1, nMatched: 1, nAffected: 1 })
> db.coches.find({marca:'tunca10'}).pretty();
[
  {
    "_id" : ObjectId("620f3d2112ed9a6002fa14e4"),
    "marca" : "tunca10",
    "model" : "tunca10",
    "kms" : 0,
    "conservacio" : 0,
    "propietari" : "ANA",
    "telefono" : "934567890",
    "telefono2" : "934567890",
    "telefono3" : "934567890",
    "correo" : "ana@tunca10.com"
  }
]
>

```

exemple de find

FIND \$type

Com en NoSql podem tindre un camp que unes vegades tinga un tipus de dada i altres vegades un altre, pot interessar-nos traure aquells documents on un determinat camp tinga un tipus concret.

Exemple: En una base de dades de pel·lícules i comentaris traure totes aquelles en els que el títol està en un string
`db.movies.find({title:{$type:"string"} }).pretty()`

FIND \$exists

S'utilitza per traure aquells documents on existeix o no existeix el camp.

Per exemple traure les pel·lícules on no apareixen actors:

`db.movies.find{cast:{$exists:false}}.pretty()`

FIND \$regex

Per utilitzar expressions regulars.

Per exemple pel·lícules que el seu títol comença per "The".

`db.movies.find({title:{$regex:^The.*}}).pretty()`

FIND \$and

Serveix per afegir restriccions y deuen de complir-se totes. Es fa de forma implícita per tant no és necessari utilitzar-la. Però alerta, en el cas de fer la consulta sobre el mateix camp, unit amb \$and SI que és necessari posar-lo.

Per exemple pel·lícules que comencen per "The" i siguen espanyoles

*countries és un array de països.

`db.movies.find({$and:[{title:{$regex:^The.*}},countries:"Spain"]}).pretty()`

`db.movies.find({title:{$regex:^The.*},countries:"Spain"}).pretty()`

Però alerta, en el cas de fer la consulta sobre el mateix camp, unit amb \$and SI que és necessari posar-lo.

Per exemple pel·lícules que siguen de Espanya i USA

`db.movies.find({$and:[countries:"Spain",countries:"USA"]}).pretty()`

FIND \$or

En aquest cas fa un or lògic i és necessari posar-lo.

Per exemple les pel·lícules que comencen per "El" o siguen espanyoles.

`db.movies.find({$or:[{title:{$regex:^El.*}}, {countries:"Spain"}]}).pretty()`

FIND \$not, \$nor

\$not és negar l'element, és a dir, el distint.

Per exemple pel·lícules que no siguen espanyoles.

`db.movies.find({countries:{$not:{$eq:"Spain"} }}).pretty()`

\$nor és negar les dos opcions, és a dir, ni açò ni allò

Per exemple pel·lícules que no son ni espanyoles ni de USA

`db.movies.find({$nor:[countries:"Spain",countries:"USA"]}).pretty()`

FIND \$eq

S'utilitza com operador d'igualtat, no sol utilitzar-se però hi ha cas que és necessari, com per exemple quan volem negar que siga espanyola

`db.movies.find({countries:{$not:{$eq:"Spain"} }}).pretty()`

Projectar

Com en SQL podem projectar i traure sols alguns elements o traure dades combinades. La sintaxis és la següent: {<element1>: 1,<element2>:1,.....}

Es posen els camps a traure junt a un 1 o true, si posem un 0 o false el camp no ix (també podem no posar-lo directament).

Alerta, el camp id sempre apareix, per tant en cas de no voler que ixca hem de dir explicitament que no aparega.

```
db.movies.find({}, {_id:0,title:1, countries:1}).pretty();
```

FIND Opcions extra

Ordenar de forma ascendent (1) o descendente(-1)

```
Sort({element1:1 o -1, element2:1 o -1,.....})
```

Ordenar les películes espanyoles per número de premis de forma descendente

```
1 db.movies.find({countries:"Spain"}).sort({awards.win.numberInt:-1}).pretty()
```

6. Arxius

6.1 Arxius del mòdul

Tattoo bd