

# Apunts CE\_5075 8.1

Iloc: [Institut d'Ensenyaments a Distància de les Illes Balears](#)  
Curs: Big data aplicat  
Llibre: Apunts CE\_5075 8.1

Imprès per: Carlos Sanchez Recio  
Data: dimarts, 22 d'abril 2025, 21:23

## Taula de continguts

### 1. Introducció

### 2. Solucions de Hadoop i Spark al núgol

- 2.1. Avantatges d'utilitzar Hadoop o Spark en el núgol
- 2.2. Azure HDInsight
- 2.3. Amazon EMR
- 2.4. Google Dataproc
- 2.5. Conclusió

### 3. Databricks

- 3.1. Què és Databricks i quina relació té amb Spark
- 3.2. Crear un recurs Databricks en Azure
- 3.3. Crear un clúster de Spark
- 3.4. Les dades en Databricks
- 3.5. Crear un quadern Databricks
- 3.6. Databricks SQL
- 3.7. Dashboards amb Databricks SQL
- 3.8. Machine Learning

## 1. Introducció

Al llarg d'aquest curs hem anat veient que l'anàlisi de dades massives pot arribar a ser crucial per a les organitzacions que volen prendre decisions informades i ben fonamentades, a partir de les seves dades. En aquest context, l'ecosistema Apache Hadoop ha esdevingut una eina fonamental en un entorn de Big Data.

D'altra banda, en el lliurament anterior vàrem introduir el framework Apache Spark, que proporciona un motor d'execució distribuït més eficient que MapReduce, la qual cosa permet treballar en temps real. Ja sabem també que Spark pot configurar-se de manera independent, sense emprar HDFS ni YARN.

També hem comentat en diversos moments del curs (especialment en el lliurament 4 del mòdul de Programació d'IA, on vàrem fer feina amb els serveis de Microsoft Azure), els avantatges que pot suposar una arquitectura basada en el níquid.

En el lliurament 1 vàrem aprendre a desplegar un petit cluster Hadoop sobre màquines virtuals, mentre que en el lliurament 7 hem vist com configurar un clúster Spark sobre contenidors Docker. En aquest lliurament volem anar una passa més enllà i veure com podem crear i utilitzar clústers Hadoop i Spark sobre plataformes del níquid.

Començarem introduint les solucions de les tres grans plataformes del níquid per crear clústers Hadoop i Spark: HDInsight de Microsoft Azure, Elastic MapReduce (EMR) d'Amazon Web Services i Dataproc de Google Cloud. I acabarem veient amb més profunditat Databricks, una plataforma per a l'anàlisi de dades massives en el níquid sobre Spark, que està disponible en Azure, AWS i Google Cloud.

## 2. Solucions de Hadoop i Spark al nigul

Aquí veurem les solucions que les tres grans plataformes de serveis al nigul proporcionen per a configurar clústers Hadoop i Spark. Desafortunadament, tots aquests serveis són de pagament i no podrem interactuar amb ells, ni tan sols amb la nostra subscripció Azure for Students. En tot cas, val la pena conèixer què ens poden oferir Azure, AWS i Google Cloud.

## 2.1. Avantatges d'utilitzar Hadoop o Spark en el nigul

Desplegar un clúster Hadoop o Spark en el nigul ofereix diversos avantatges que milloren la seva eficàcia i faciliten la gestió de les dades a gran escala.

### 1. Escalabilitat

El nigul proporciona una escalabilitat fluïda que permet a les organitzacions ajustar els recursos segons les necessitats de càrrega de treball, sense necessitat de preveure el pic màxim de dades. Això significa que podem augmentar o disminuir el nombre de nodes en el clúster de Hadoop o Spark dinàmicament, optimitzant així els costos i l'eficiència.

### 2. Cost-eficiència

Amb el model de pagament per ús del nigul, les organitzacions només paguen pels recursos que utilitzen. Això elimina la necessitat d'inversions inicials en *hardware* i manteniment, reduint significativament els costos operatius. A més, les millores en l'automatització i la gestió de clústers ajuden a reduir els costos administratius associats.

### 3. Agilitat i velocitat

El desplegament de Hadoop o Spark en el nigul pot realitzar-se en qüestió de minuts, comparat amb els dies o setmanes que es podria tardar en configurar-se un entorn en les instal·lacions físiques. Aquesta agilitat permet a les organitzacions ser més reactives als canvis en les necessitats de negoci i accelerar el llançament de noves iniciatives de dades.

### 4. Manteniment simplificat

Els proveïdors de serveis en el nigul gestionen la infraestructura, inclòs el manteniment regular i les actualitzacions de seguretat, alliberant als equips de TIC de les organitzacions de tasques de manteniment rutinàries. Això permet als professionals de les dades concentrar-se més en l'anàlisi i menys en l'administració del sistema.

### 5. Integració amb serveis de nigul

Els entorns de nigul ofereixen integració amb una àmplia gamma de serveis addicionals, com ara plataformes analítiques, eines de visualització de dades, i solucions d'emmagatzematge. Això facilita l'expansió dels ecosistemes de dades i la creació de *pipelines* de dades robustos, millorant així les capacitats analítiques globals de l'organització.

### 6. Accessibilitat i col·laboració

El nigul permet que les dades i les plataformes de processament siguin accessibles des de qualsevol lloc, facilitant la col·laboració entre equips distribuïts geogràficament. Això es tradueix en una major flexibilitat per al treball remot i per a projectes que involucren múltiples departaments o entitats.

### 7. Recuperació davant desastres

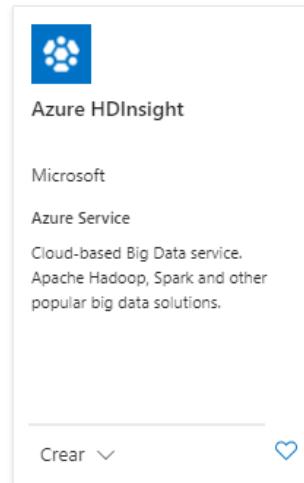
La infraestructura en el nigul sovint inclou opcions robustes de recuperació davant desastres i d'alta disponibilitat. Això significa que les dades i aplicacions són més resilients davant fallades del sistema o catàstrofes naturals, garantint la continuïtat del negoci i la protecció de dades valioses.

## 2.2. Azure HDInsight

Azure HDInsight és el servei de cloud ofert per Microsoft Azure per configurar clústers Hadoop. De fet, el nom HDInsight és una combinació de "HD", que fa referència a Hadoop, i "Insight", que vol dir un coneixement profund sobre un tema.

HDInsight també permet crear clústers d'altres frameworks basats en Hadoop, com poden ser Spark, HBase, Hive o Kafka. En qualsevol d'aquests casos, el clúster implementarà els components base de Hadoop: HDFS i YARN. A més, HDInsight també està integrat amb altres serveis d'Azure, com les solucions d'emmagatzematge Azure Blob Storage i Azure Data Lake Storage.

Una vegada que hem vist els fonaments d'Azure Portal en el lliurament 4 del mòdul de Programació d'intel·ligència artificial, configurar un clúster amb HDInsight resulta bastant senzill. Per fer-ho, des d'Azure Portal, hem de crear un nou recurs del tipus "Azure HDInsight".



**Imatge:** Recurs Azure HDInsight en Azure Portal

A continuació haurem de configurar els paràmetres habituals: subscripció d'Azure, nom del grup de recursos, nom del recurs (clúster) i regió. I aquí hem de seleccionar el tipus de clúster, entre diverses possibilitats: Hadoop, Spark, Kafka (per al processament de dades en streaming), HBase i Interactive Query (que internament és Hive).

The screenshot shows the Azure portal interface for creating a new resource. On the left, a navigation bar includes 'Inicio', 'Crear un recurso', 'Marketplace', 'Azure HDInsight', and a 'Crear clúster de HDInsight' button. The main area is titled 'Crear clúster de HDInsight'. It contains several sections: 'Conceptos básicos' (Subscription: Azure for Students, Resource Group: (Nuevo) test\_hdinsight), 'Detalles del proyecto' (Project name: cluster-hdinsight, Region: France Central, Availability zone: 1), 'Detalles del clúster' (Cluster type: Hadoop, Version: Hadoop 3.3.4 (HDI 5.1)), and 'Revisión y creación' (Review + Create). Below the main form is a 'Selección de tipo de clúster' modal window. This window lists five options with 'Seleccionar' buttons: 'Hadoop' (Apache Hadoop, MapReduce, Hive, Pig, Sqoop, Oozie), 'Spark' (Apache Spark, fast data processing), 'Kafka' (Apache Kafka, streaming platform), 'HBase' (Apache HBase, NoSQL database), and 'Interactive Query' (Apache Hive, SQL on Hadoop, LLAP). The 'Hadoop' section is currently selected.

**Imatge:** Tipus de clústers HDInsight

Podem anar avançant i arribam a la configuració de nodes del clúster. Un clúster HDInsight sempre necessita 2 nodes mestre (*head nodes* o nodes d'encapçalament) i 3 nodes Zookeeper (Zookeeper, com ja varem veure en el lliurament de Kafka, és un servei per a la coordinació de processos que té un paper important en la gestió del clúster). I podem triar el nombre de nodes treballador (*worker nodes*) que vulguem crear, 4 per defecte.

[Inicio](#) > [Crear un recurso](#) > [Marketplace](#) > [Azure HDInsight](#) >

## Crear clúster de HDInsight ...

Conceptos básicos Almacenamiento Seguridad y redes **Configuración y precios** Etiquetas Revisión y creación

Configure los precios y el rendimiento del clúster. [Más información](#)

### Configuración del nodo

Permite configurar el tamaño y el rendimiento del clúster y ver la información de los costos estimados.

Esta estimación de los costos representada en la tabla no incluye los descuentos ni los costos de la suscripción relativos al almacenamiento, la red o la transferencia de datos.

**X** La cuota de la familia de SKU standardev3family no es suficiente, valor disponible: 0, núcleos adicionales: 40; La cuota de la familia de SKU standardav2family no es suficiente, valor disponible: 0, núcleos adicionales: 6; No hay suficientes núcleos disponibles para admitir el número seleccionado de nodos. Ajuste la cantidad de nodos seleccionados, elija una región diferente o abra un caso de soporte para solicitar núcleos de HDInsight adicionales.  
[Ver uso de núcleos](#)  
[Abra un caso de soporte técnico de aumento de cuota de HDInsight](#)

### + Agregar aplicación

Tipo del nodo	Tamaño del nodo	Número de ...	Costo estimado p...
Nodo Encabez...	E4 V3 (4 Núcleos, 32 GB de RAM), 0.39 USD/h...	2	0.78 USD
Nodo Zookeeper	A2 v2 (2 Núcleos, 4 GB de RAM), 0.00 USD/ho...	3	0.00 (GRATIS)
Nodo Trabajador	E8 V3 (8 Núcleos, 64 GB de RAM), 0.78 USD/h...	4	3.10 USD

Ha alcanzado el límite de cuota de núcleos de la suscripción en France Central. Elija otra región o solicite soporte de facturación para aumentar el límite de France Central. El valor debe ser de entre 1 y 1.

Habilitar disco administrado

Habilitar escalado automático [Más información](#)

Costo total estimado/hora 3.88 USD

Acciones de secuencia de comandos

**Revisión y creación**

[« Anterior](#)

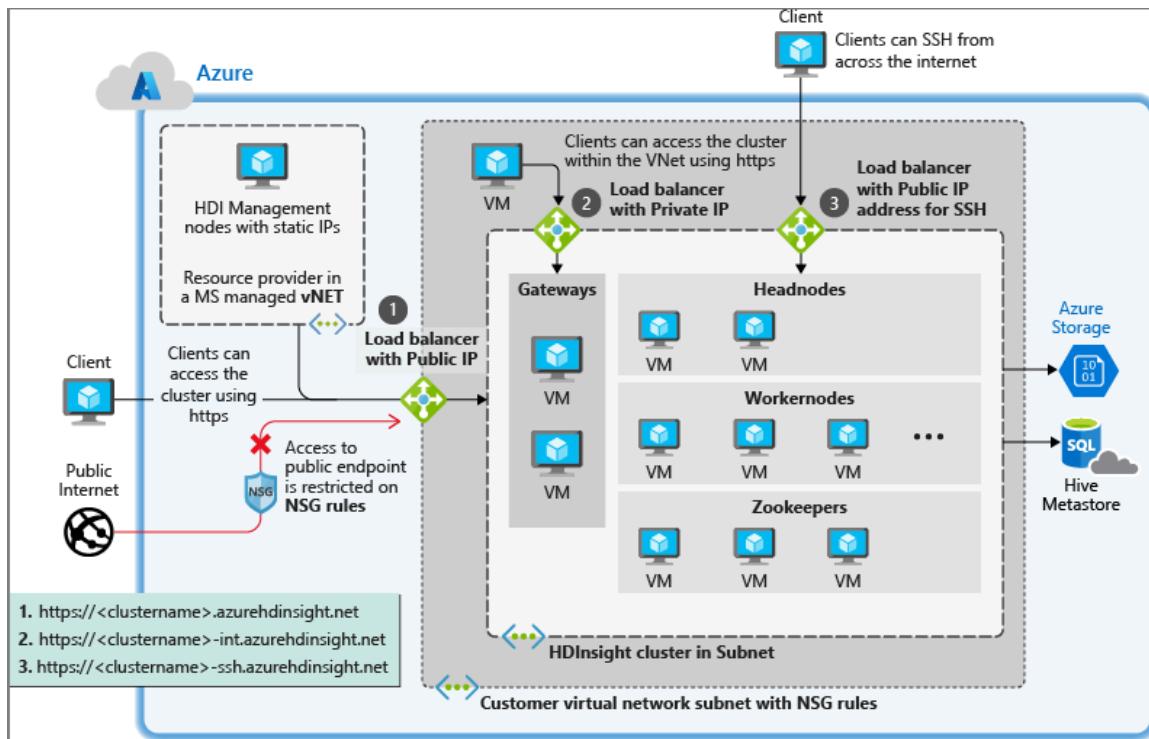
[Siguiente: Etiquetas»](#)

**Imatge:** Configuració de nodes del clúster

En la imatge podem veure que ens apareixen errors. La raó és que la subscripció Azure Students no permet la creació de clústers HDInsight. En una subscripció normal de pagament sí que seria possible. En tot cas, amb això ja podem veure com el procés és molt simple, molt més que el que ens va costar crear el nostre clúster Hadoop en el lliurament 1. Aquí no cal crear les màquines per a cada node, instal·lar-hi un sistema operatiu, configurar-hi Hadoop, etc. Tot el procés de creació del clúster és automàtic. I amb l'aventatge d'una gestió també molt simplificada, on afegir nodes quan les càrregues ho requereixin és molt senzill.

## Arquitectura

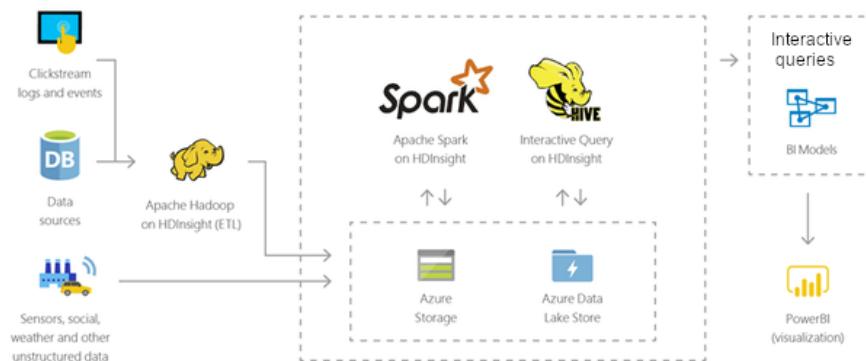
Entrant en els detalls, la següent imatge mostra l'arquitectura d'HDInsight. Podem veure com, en el nucli de l'arquitectura tenim les màquines virtuals corresponents als 2 nodes mestre (*head nodes*) del clúster, les tres corresponents als nodes de Zookeeper i  $n$  màquines virtuals per als nodes dels nodes *worker*.



Imatge: Arquitectura d'Azure HDInsight. Font: Microsoft Learn

## Casos d'ús

És interessant veure dos casos d'ús habituals amb Azure HDInsight. El primer es tracta d'un ús centrat en *data warehousing*, on HDInsight es fa servir per poder executar queries interactives sobre grans volums (petabytes) de dades, tant estructurades com no estructurades. En la següent imatge es veu com Hadoop es pot fer servir com a base per construir una capa d'emmagatzematge d'Azure. En aquest context, Hive ens proporciona les funcionalitats de magatzem de dades, cosa que permet executar queries en HiveQL sobre aquestes dades. Les API de Spark també ens permeten consultar aquestes dades.

Imatge: HDInsight per a data warehousing. Font: <https://blog.hensongroup.com/what-is-azure-hdinsight/>

El segon cas d'ús se centra en l'anàltica de negoci (*business analytics*). Aquí el que volem és extreure informació rellevant a partir de dades que poden estar en un *data lake* d'Azure, emprant Spark com a infraestructura de processament distribuït. Aquesta aproximació permet fer una analítica descriptiva, obtenint visualitzacions (quadres de comandament) amb PowerBI. I també fer una analítica predictiva, aplicant els serveis de machine learning d'Azure (o l'API de Spark MLlib).



**imatge:** HDInsight per a business analytics. Font: <https://blog.hensongroup.com/what-is-azure-hdinsight/> (editada)



Podeu trobar informació més detallada sobre HDInsight a <https://learn.microsoft.com/es-es/azure/hdinsight/hdinsight-overview>

A <https://learn.microsoft.com/es-es/training/patterns/build-oss-analytical-solutions-az-hdinsight/> també podreu trobar una sèrie de cursos d'aprenentatge de Microsoft Learn sobre HDInsight.

AMPLIACIÓ

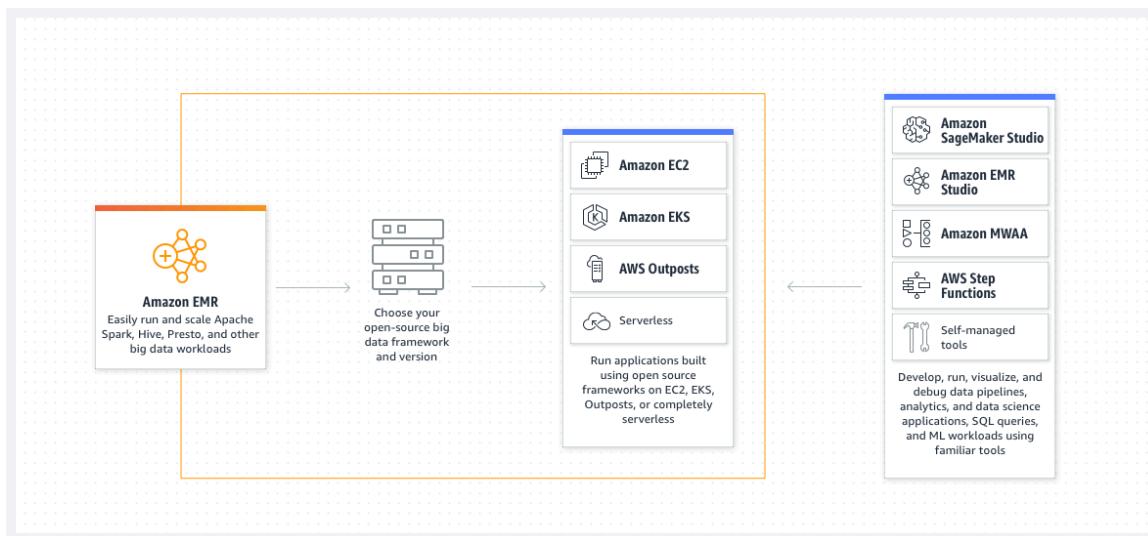
## 2.3. Amazon EMR

Amazon Elastic MapReduce (EMR) és la solució per al processament de dades massives, anàlisi interactiva i aprenentatge automàtic en el níquid ofert per Amazon Web Services (AWS).

Quan EMR va ser llançat per primera vegada, estava enfocat principalment en fer Hadoop (i totes les eines del seu ecosistema) més accessible i fàcil d'usar a gran escala, proporcionant una solució de Hadoop com a servei. Amb el temps, AWS va anar ampliant les capacitats d'EMR per incloure altres motors populars de processament distribuït de dades, com Apache Spark. Així, el cor d'EMR és Hadoop, utilitzant el seu model de computació distribuïda MapReduce (d'aquí precisament el nom d'Elastic MapReduce) i el seu sistema de fitxers distribuït (HDFS). Tot i això, quan avui en dia es fa feina amb Hadoop sobre EMR és molt més freqüent utilitzar Apache Tez (un altre motor d'execució sobre YARN, més eficient que MapReduce) en lloc de MapReduce. Per exemple, és molt habitual treballar en EMR amb Hive, però executant-se no sobre MapReduce sinó sobre Tez.

D'altra banda, EMR permet interactuar amb la resta de solucions de la plataforma AWS, com per exemple S3 per a l'emmagatzematge de dades. A més, Amazon proporciona dues IDE per al desenvolupament d'aplicacions en diversos llenguatges (Python i PySpark entre ells): EMR Studio, per a l'anàlisi de dades, basada en quaderns Jupyter, i SageMaker Studio per a l'aprenentatge automàtic.

La imatge següent mostra una visió general d'EMR. Podem veure que el mòdul que diu "Choose your open-source big data framework and version" és on tendríem Hadoop o Spark.



**Imatge:** Visió general d'Amazon EMR. Font: <https://aws.amazon.com/es/emr/>



Podeu trobar més informació sobre:

- Hadoop en EMR: <https://aws.amazon.com/es/emr/features/hadoop/>
- Spark en EMR: <https://aws.amazon.com/es/emr/features/spark/>
- Hive en EMR: <https://aws.amazon.com/es/emr/features/hive/>
- HBase en EMR: <https://aws.amazon.com/es/emr/features/hbase/>

## 2.4. Google Dataproc

Dataproc és el servei gestionat de Google Cloud que permet desplegar clústers Hadoop i Spark, entre d'altres, equivalent a Azure HDInsight i Amazon EMR.

Com passa amb les altres plataformes, Dataproc també s'integra amb la resta de serveis de Google Cloud. Per exemple, per a l'emmagatzematge, Dataproc pot treballar amb Google Cloud Storage (GCS), BigQuery (una solució de *data warehousing* de Google per a grans volums de dades estructurades) o BigTable (la base de dades NoSQL columnar de Google). Dataproc també pot utilitzar-se amb Vertex AI, la plataforma per a intel·ligència artificial de Google, que inclou Gemini, el MML (model de llenguatge extens) de Google.

Un dels punts forts de Dataproc és la seva extensió Dataproc Serverless, que permet executar treballs de Spark sense haver de provisionar ni administrar un clúster. És a dir, els recursos computacionals necessaris per executar el treball s'assignen automàticament de manera dinàmica, sobre la marxa. Això facilita enormement les tasques d'administració. Hem d'aclarir que Amazon EMR també té un mòdul EMR Serverless amb característiques semblants. Nosaltres emprarem capacitats *serverless* en Databricks, en l'apartat 3.6.



Podeu trobar més informació sobre Dataproc a <https://cloud.google.com/dataproc>

## 2.5. Conclusió

Hem pogut veure que els tres productes, Azure HDInsight, Amazon EMR i Google Dataproc, ofereixen unes característiques molt semblants. Tots tres ens permeten crear clústers Hadoop o Spark en el níquid, d'una manera senzilla i àgil, amb una alta escalabilitat. I tots tres estan integrats amb la resta d'eines de la seva plataforma.

Poden haver-hi diferències de preu en els serveis, que poden ser més o menys significatives en funció de la mida dels nostres treballs. Però, molt sovint, la tria entre ells depèn fonamentalment de quina sigui la plataforma de níquid que ja pugui estar utilitzant la nostra organització.

### 3. Databricks

[Databricks](#) ofereix una plataforma per a l'anàlisi de dades massives, basada en Apache Spark, completament en el níu. Databricks està disponible en les tres plataformes principals del níu, Azure, AWS i Google Cloud. En concret, nosaltres treballarem sobre Azure, fent servir l'usuari de la subscripció d'Azure for Students que ja varem emprar en el llurament 4 del mòdul de Programació d'intel·ligència artificial.

En aquest apartat veurem com crear un clúster Spark per a poder realitzar les nostres tasques d'anàlisi de dades i d'aprenentatge automàtic. Veurem també com Databricks gestiona les dades amb el que s'anomena un *data lakehouse* (magatzem de llac de dades), que es basa en dos components clau, Delta Lake i Unity Catalog. A continuació, tractarem els quaderns de Databricks, semblants als quaderns de Jupyter, que ens serveixen per a desenvolupar codi (en diversos llenguatges) i presentar resultats. Seguirem amb Databricks SQL, una funcionalitat que ens permet crear elements de computació *serverless* molt eficients, a més de poder interactuar amb les dades d'una manera senzilla. En concret, podrem definir quadres de comandament o *dashboards* per a visualitzar les nostres dades. Finalment, introduirem l'ús de Databricks per a l'aprenentatge automàtic.

### 3.1. Què és Databricks i quina relació té amb Spark

Databricks ofereix una plataforma unificada de dades massives en el núvol, que permet als usuaris configurar i gestionar clústers de Spark de manera senzilla i eficient, i desenvolupar aplicacions d'anàlisi de dades i aprenentatge automàtic a gran escala. Aquesta plataforma proporciona:

- **Gestió de clústers:** Automatització en la configuració, escalat i manteniment de clústers Spark en el núvol.
- **Quaderns integrats:** Una interfície basada en quaderns (*notebooks*) que permet als usuaris escriure codi, executar-lo i visualitzar resultats, tot en un entorn interactiu.
- **Interfície d'usuari amigable:** Facilita la gestió de *pipelines* de dades, des de l'ingesta fins a l'anàlisi i la visualització.
- **Integració amb altres serveis de núvol:** Databricks treballa eficientment dins dels ecosistemes cloud com AWS, Microsoft Azure i Google Cloud.

#### Orígens de Databricks i relació amb Apache Spark

Apache Spark va ser inicialment desenvolupat a l'Universitat de Califòrnia, Berkeley, dins del seu AMPLab (Algorithms, Machines, and People Lab). Ja sabem que Spark va ser creat com una millora al model de programació de MapReduce, proporcionant una manera més ràpida i flexible de processar grans conjunts de dades, especialment per a aplicacions que requereixen múltiples operacions de passada sobre el mateix conjunt de dades.

Els fundadors de Spark van veure l'oportunitat de comercialitzar aquesta tecnologia i van crear Databricks el 2013. La companyia va ser establerta amb l'objectiu de desenvolupar una plataforma basada en el núvol que simplificava l'ús de Spark, fent-lo més accessible i gestionable per a les empreses.

Databricks ha contribuït significativament al desenvolupament de Spark, aportant noves funcionalitats i millores en el rendiment. Algunes d'aquestes contribucions inclouen l'optimització del rendiment de Spark, la millora de les APIs, i l'extensió de les capacitats de Spark en àrees com l'aprenentatge automàtic i el processament en temps real.

En resum, Databricks i Apache Spark estan intrínsecament lligats des dels seus inicis, amb Databricks jugant un rol clau en la promoció i l'adopció de Spark en la indústria, així com en el seu desenvolupament continu. La plataforma Databricks proporciona una manera poderosa i simplificada d'utilitzar Spark en entorns empresarials, ajudant les organitzacions aaprofitar el poder de l'anàlisi de dades massives.

## 3.2. Crear un recurs Databricks en Azure

Com sempre que volem fer feina amb Azure hem de provisionar el recurs corresponent. Així que, una vegada hem entrat en [Azure Portal](#) amb el nostre usuari de la subscripció Azure for Students, hem de pitjar la icona de "Crear un recurso". A continuació hem de cercar "databricks". Veurem que el primer dels recursos que ens apareixen es diu "Azure Databricks".

The screenshot shows the Azure Marketplace interface. At the top, there's a search bar with the text 'databricks'. Below it, a checkbox labeled 'Azure services only' is unchecked. The search results show two items: 'Azure Databricks' and 'Access Connector for Azure Databricks'. Both items are from Microsoft and are categorized under 'Azure Service'. The 'Azure Databricks' item has a brief description: 'Azure Databricks is the fast, easy and collaborative Apache Spark-based analytics platform.' Below each item are 'Create' and 'Heart' buttons.

**Imagen:** Creación del recurso

Pitjarem el botó "Create" i seleccionam l'única opció, "Azure Databricks". Ens apareixerà ja la pantalla per començar a configurar el nostre recurs, un àrea de treball (workspace) d'Azure Databricks. En la primera pestanya, en dades bàsiques, hem de seleccionar les següents opcions:

- *Subscripción:* Azure for Students
- *Grupo de recursos:* cream un de nou i li posam, per exemple, el nom "databricks1group". Els grups de recursos ens permeten agrupar-los, per exemple, per projectes, de manera que tots els recursos associats a un projecte estiguin dins del mateix grup.
- *Nombre del área de trabajo:* "databricks1" (per exemple)
- *Región:* seleccionam "West Europe" (altres, com "France Central", poden no oferir totes les funcionalitats que necessitem)
- *Plan de tarifa:* "Standard (Apache Spark, Secure with Microsoft Entra ID)"
- *Nombre del grupo de recursos administrados:* "databricks1managedgroup" (per exemple)

[Datos básicos](#) [Redes](#) [Cifrado](#) [Security & compliance](#) [Etiquetas](#) [Revisar y crear](#)

### Detalles del proyecto

Seleccione la suscripción para administrar recursos implementados y los costes. Use los grupos de recursos como carpetas para organizar y administrar todos los recursos.

Suscripción *	Azure for Students
Grupo de recursos *	(Nuevo) databricks1group
	<a href="#">Crear nuevo</a>

### Detalles de instancia

Nombre del área de trabajo *	databricks1
Región *	West Europe
Plan de tarifa *	Standard (Apache Spark, Secure with Microsoft Entra ID)
Nombre del grupo de recursos administrados	databricks1managedgroup

**Imatge:** Configuració de l'àrea de treball d'Azure Databricks

A la resta de pestanyes podem deixar les opcions per defecte i en la darrera de totes ("Revisar y crear"), podem pitjar el botó "Crear". Després d'una estona s'haurà creat tota la infraestructura per poder treballar amb Databricks en Azure.

Inicio >

 **databricks1group\_databricks1 | Información general** [...](#)

Implementación

Buscar Eliminar Cancelar Volver a implementar Descargar Actualizar

**Información general**

Se completó la implementación

Nombre de implementación : databricks1group\_databricks1  
Suscripción : Azure for Students  
Grupo de recursos : databricks1group

Detalles de implementación

Pasos siguientes

**Ir al recurso**

**Imatge:** Recurs creat

Podem pitjar el botó "Ir al recurso" per accedir-hi.

**databricks1** Servicio de Azure Databricks

Información general

Estado: Active  
Grupo de recursos: databricks1group  
Ubicación: West Europe  
Suscripción: Azure for Students  
Id. de suscripción: 64a5c9f9-a949-4d26-8c28-445f8eb77b0b  
Etiquetas (editar): Agregar etiquetas

Vista JSON

Información esencial

Iniciar área de trabajo

Actualizar a Premium

Documentación, Introducción, Importar datos de archivo, Importar datos de Azure Storage, Libreta, Guía de administración, Vincular área de trabajo de Azure ML.

**Imatge:** Pàgina del recurs

I ara ja podem pitjar a "Iniciar área de trabajo" per a, finalment, accedir a la pàgina on tendrem accés a totes les funcionalitats de Databricks en Azure:

Microsoft Azure | **databricks**

+ Nuevo

Workspace, Recientes, Catálogo, Flujos de trabajo, Cómputo, Ingeniería de datos, Ejecuciones, Machine Learning, Zona de pruebas, Experimentos, Características, Modelos, Servicio.

Bienvenido a Databricks

Buscar datos, cuadernos, recientes y más... CTRL + P

Set up your workspace

Follow this step-by-step guide that walks you through setting up the workspace for your new Databricks account.

Get started

Recientes, Favoritos, Popular, Mosaic AI, Novedades

Comience su recorrido

Pruebe el menú «Nuevo», donde puede subir datos o conectarse a ellos y, a continuación, explorarlos en un Notebook o tablero.

+ Nuevo

**Imatge:** Àrea de treball de Databricks en Azure

En els apartats següents seguirem treballant amb Databricks des d'aquesta àrea de treball.

### 3.3. Crear un clúster de Spark

Per poder fer feina amb Databricks, hem de crear un clúster de Spark sobre el qual poder executar els nostres processos. Per fer-ho, en l'àrea de treball de Databricks, hem de pitjar el botó "Nuevo" i dins l'opció "Más" trobarem "Clúster".

Cómputo > Nuevo cómputo

### Proves Azure's Cluster

Multi-nodo  Nodo único

Modo de acceso ? Acceso de usuario único ?

Usuario único ▼ Proves Azure ▼

#### Rendimiento

Versión de Databricks Runtime ?

Runtime: 15.4 LTS (Scala 2.12, Spark 3.5.0) ▼

Utilizar aceleración Photon ?

Tipo de worker <small>?</small>	Workers mín.	Workers máx.
Standard_D4ds_v5 16 GB de memoria y 4 núcleos <small>▼</small>	2	8

Instancias spot ?

#### Tipo de driver

El mismo que el worker  
16 GB de memoria y 4 núcleos ▼

Activar la auto expansión ?

Terminar después de  minutos de inactividad ?

#### Etiquetas ?

Añadir etiquetas

Key	Value	Añadir
-----	-------	--------

> Etiquetas añadidas automáticamente

▶ Opciones avanzadas

**Imagen:** Creació d'un clúster multi-node

Veim aquí que podríem crear un clúster amb diversos nodes, on podem configurar el número mínim i màxim de nodes *worker* de Spark, per defecte 2 i 8 respectivament. Podríem seleccionar amb quina versió de Scala i Spark volem fer feina (Scala 2.12 amb Spark 3.5.0, per defecte, la versió més moderna amb LTS -Long Time Support-, i quin tipus de màquina volem emprar per a cada node *worker* (16GB de RAM i 4 nuclis, per defecte) i per al node *driver* (el mateix que els nodes *worker*, per defecte). Podem observar que la configuració del clúster és molt senzilla.

No obstant això, la nostra subscripció d'Azure for Students no ens permet crear clústers multi-node, així que ens hem de conformar amb crear un clúster amb un únic node. En tot cas, per als exemples i exercicis que volem fer, en tenim prou.

Cómputo > Nuevo cómputo >

## Toni Prova's Cluster

Multi-nodo  Nodo único

Modo de acceso  Acceso de usuario único 

Usuario único |  Toni Prova 

### Rendimiento

Versión de Databricks Runtime 

Runtime: 14.3 LTS (Scala 2.12, Spark 3.5.0) 

Utilizar aceleración Photon 

Tipo de nodo 

Standard\_DS3\_v2 14 GB de memoria y 4 núcleos  

Terminar después de  minutos de inactividad 

### Etiquetas

Añadir etiquetas

Key	Value	
-----	-------	---

> Etiquetas añadidas automáticamente

► Opciones avanzadas

**Imatge:** Creació d'un clúster de node únic

Començam seleccionant la versió del Databricks Runtime. Podem deixar l'opció per defecte, la versió 15.4 LTS amb Scala 2.12 i Spark 3.5.0, la més moderna amb LTS.

A continuació hem de decidir si volem emprar Photon, un component que accelera les queries en Databricks, ja sigui emprant SQL o dataframes de Spark. Això redueix el temps d'execució dels processos, tot i que augmenta el preu per hora del clúster. Per fer els exemples no té massa impacte perquè els datasets són petits i els costos molt baixos.

Seguidament hem de seleccionar el tipus de node. L'opció per defecte és Standard\_D4ads\_v5, amb 16GB de RAM i 4 nuclis, que és el node més lleuger dels que tenen [Cache Delta](#), una opció interessant per a l'optimització del rendiment. No obstant això, la nostra subscripció tampoc no ens permet emprar aquests tipus de nodes.

Seleccionarem el més lleuger dels d'ús general que no tenen Cache Delta, el Standard\_D4s\_v2 amb 14GB de RAM i 4 nuclis.

A la part dreta podem veure el cost del clúster, en aquest cas 1,5 DBU/h (unitat Databricks per hora). Si no haguéssim seleccionat Photon, el preu seria la meitat, 0,75 DBU/h. Per als nodes amb Cache Delta tendrem seria respectivament 2 i 1 DBU/h.

### Resumen

1 driver	14 GB de memoria y 4 núcleos
Runtime	15.4.x-scala2.12
 Photon	 Standard_DS3_v2

**Imatge:** Preu del clúster

I quin preu té un DBU/h? Una DBU és una unitat de capacitat de processament i depèn de les màquines virtuals (i si fan ús de Photon) seleccionades per al clúster. El preu d'aquestes DBU pot variar segons la regió i si hem seleccionat una tarifa estàndard (com nosaltres) o Premium. Actualment, per una tarifa estàndard, el preu per a

processar treballs és de 0,14 euros per DBU/hora, mentre que si són treballs lleugers és de 0,07 euros per DBU/hora. Podeu trobar tots els detalls sobre la facturació de Databricks en Azure a <https://azure.microsoft.com/es-es/pricing/details/databricks/>

També pot ser interessant modificar el següent paràmetre, el que indica quants minuts espera d'inactivitat espera el sistema per aturar els recursos. Per defecte són 120, podem baixar-ho a 30, per evitar costos si ens oblidam d'aturar manualment.

Una vegada ja hem configurat totes les opcions, podem pitjar "Crear cómputo" i, després d'una estona, veurem que ens apareixerà un punt verd al costat del nom del clúster, indicant que s'ha creat el nostre clúster Spark (en el nostre cas, amb un únic node). Si anam al menú "Cómputo", veurem que ens apareix:

The screenshot shows the 'Compute' blade in the Azure Databricks interface. At the top, there are tabs for 'Cómputo interactivo', 'Cómputo de trabajos', 'Pools', and 'Aplicaciones'. Below the tabs is a search bar with placeholder text 'Filtre los cómputos a los que tiene acceso' and a dropdown menu set to 'Creador'. There are also two checkboxes: 'Solo los anclados' and 'Crear cómputo'. The main area displays a table with the following columns: Estado, Nombre, Runtime, Memoria activa, Núcleos activos, DBU/h activo, Fuente, Creador, Cuadernos, and three more columns represented by icons. A single row is visible, showing the status as 'En ejecución' (Running), the name as 'Proves Azure's Cluster', runtime of 15.4, active memory of 14 GB, active cores of 4, active DBUs of 1.5, and the source as 'Proves Azure'. The 'Creador' column shows a user icon and the name 'Proves Azure'. The 'Cuadernos' column has a minus sign. The bottom right of the table has three more icons: a blue square, a red square, and a green square.

**imatge: Clúster creat**

Una vegada hem creat el clúster, ja estam llistos per afegir-hi dades i interactuar amb elles.



Una vegada hem acabat totes les tasques que volem fer amb el nostre clúster, és convenient aturar-lo per evitar despeses innecessàries. En tot cas, passat un determinat temps d'inactivitat (120 minuts per defecte, que hem dit de reduir-lo a 30), el clúster s'aturarà de manera automàtica.

I quan estem segurs que ja no hem de tornar a utilitzar el nostre clúster, també és recomanable eliminar-lo.

## 3.4. Les dades en Databricks

Si, en l'àrea de treball de Databricks, anam a la secció de "Catàlogo", veurem que tenim dues entrades. Una d'elles és un *metastore* de Hive, amb una base de dades "default", que està buida. Xerrarem més tard sobre Hive i ens centrarem ara en l'altra entrada, una col·lecció amb dades d'exemple ("samples"), que conté tres esquemes o bases de dades. Per exemple, la segona base de dades, "nyctaxi", conté la taula "trips" amb les dades dels viatges dels taxis de Nova York del mes de febrer de 2016 (recordareu que vàrem emprar aquestes dades en el lliurament 5 de Sistemes de big data).

Columna	Tipo	Comentario
tpep_pickup_datetime	timestamp	
tpep_dropoff_datetime	timestamp	
trip_distance	double	
fare_amount	double	
pickup_zip	int	
dropoff_zip	int	

imatge: Dades en el catàleg

Realment, el que estam veient aquí és una visió lògica. Recordem que en el lliurament 3 ([apartat 4 dels apunts](#)) vàrem introduir els conceptes de **magatzems de dades (data warehouses)** i **llacs de dades (data lakes)**. Tots dos són repositoris de dades que ens serveixen per fer analítica de dades, ja sigui descriptiva, predictiva (aplicant models d'aprenentatge automàtic) o prescriptiva. No obstant això, hi ha una diferència fonamental entre les dues solucions. Un magatzem de dades fa feina amb dades estructurades, que normalment s'han transformat prèviament. En canvi, un llac de dades emmagatzema dades estructurades, semiestructurades i no estructurades de tot tipus, en el seu format original, sense cap transformació prèvia.

Databricks combina els avantatges de les dues aproximacions i defineix el que denomina **data lakehouse**, que s'ha traduït com a **magatzem de llac de dades**. Així doncs, un *data lakehouse* és una plataforma integrada que unifica l'emmagatzematge de dades massives en el seu format original (*data lake*) amb les capacitats analítiques (consulta, ànalisi i visualització) i de gestió de dades estructurades (*data warehouse*), proporcionant així una solució més eficient i escalable per a l'anàlisi de dades.

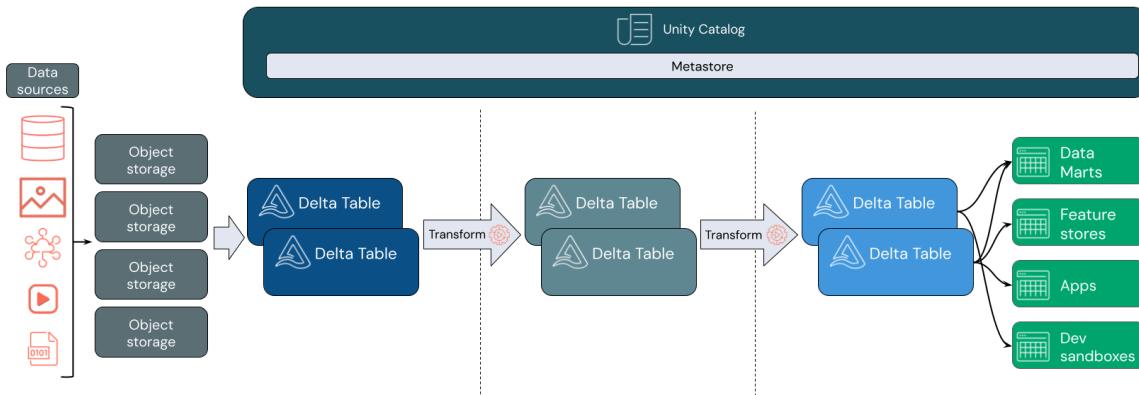
Per fer això possible, Databricks es basa en dos components, que s'executen sobre Spark: Delta Lake i Unity Catalog.

**Delta Lake** (sovint denominada només Delta) és una capa d'emmagatzematge optimizada que soporta transaccions ACID sobre dades estructurades i semi-estructurades. Aquestes dades s'emmagatzemen en les anomenades taules Delta (*Delta tables*). També proporciona una comprovació d'esquemes (*schema enforcement* o aplicació d'esquema), que permet verificar que les dades s'ajusten a un esquema de dades predefinit; en cas contrari, no es podran inserir dins de les taules Delta.

**Unity Catalog** és el catàleg de dades, és a dir proporciona un *metastore* o repositori centralitzat de les metadades de totes les taules Delta. Per tant, és el que fa visibles les taules Delta. Unity Catalog també proporciona un sistema de governança de dades, incloent control d'accés basat en rols, auditores de dades, compliment amb polítiques de seguretat, etc.

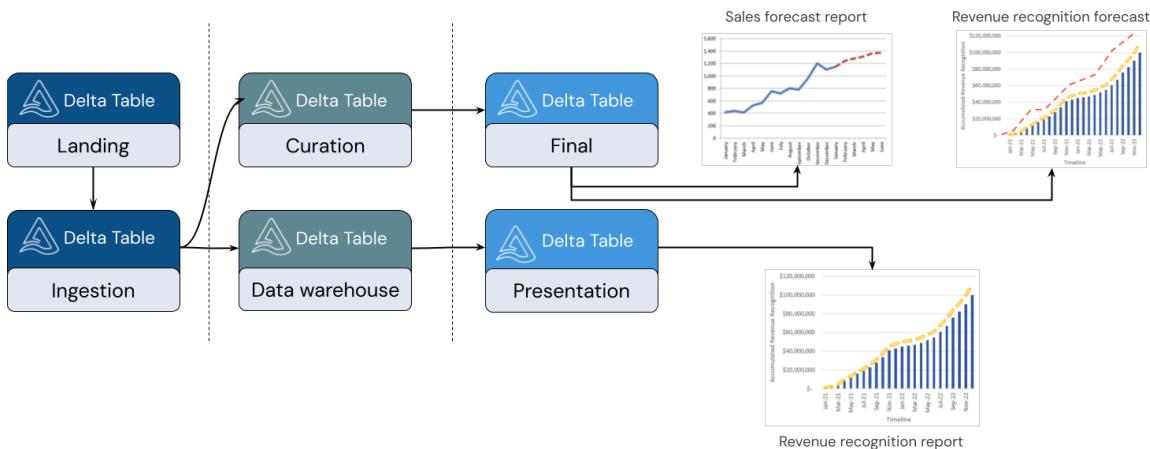
El que aconseguim amb Delta Lake i Unity Catalog és que l'emmagatzematge físic sigui completament transparent per als usuaris (científics de dades i enginyers de dades). L'usuari només veu taules i no necessita saber ni on ni com estan emmagatzemades. A més de proporcionar les funcionalitats transaccionals, d'aplicació d'esquema i de governança de dades que hem mencionat, en un entorn distribuït, escalable i d'alt rendiment.

La imatge següent mostra una visió general d'una instància de *data lakehouse*, on podem veure com tenim diverses fonts de dades, que acaben originant, aplicant una sèrie de transformacions, un conjunt de taules Delta, que es fan públiques a través de Unity Catalog i que acaben essent accedides per aplicacions.



**imatge:** Visió general d'un data lakehouse de Databricks. Font: Microsoft Learn

La imatge següent mostra com es modelen i mantenen les dades (taules Delta) a mesura que es van fent diverses transformacions, donant lloc a un sistema que ofereix diverses visions (o capes) de les dades.



**imatge:** Modelat de dades. Font: Microsoft Learn

## Les taules Delta i el sistema d'arxius DBFS

Les taules Delta s'emmagatzemem mitjançant arxius en format Parquet. Parquet és un format de fitxer columnar altament eficient per a l'ús amb tecnologies de processament distribuït de dades massives com Hadoop i Spark. Aquest format optimitza tant l'eficiència d'emmagatzematge, com la velocitat de les operacions de lectura, especialment per consultes que no requereixen la lectura de totes les columnes de les dades. El problema de Parquet és que no soporta transaccions ACID, cosa que ens vendrà proporcionada per Delta Lake.

Delta Lake pot operar sobre diversos sistemes d'arxius compatibles amb Hadoop, principalment HDFS (Hadoop Distributed File System) i també sobre sistemes d'emmagatzematge basats en el níquid com Amazon S3, Azure Data Lake Storage (ADLS) o Google Cloud Storage (GCS). Aquesta capacitat permet a Delta Lake integrar-se fàcilment en diversos entorns, tant de *cloud* com *on-premise*.

Per donar una visió unificada de totes aquestes possibles fonts de dades, Databricks fa servir un sistema d'arxius abstracte anomenat Databricks File System (DBFS). DBFS actua com una capa de sistema d'arxius virtual, de manera que qualsevol objecte (per exemple, una taula Delta) tengui un *path* DBFS. Això permet que els usuaris i desenvolupadors puguin referenciar una taula Delta mitjançant aquesta ruta DBFS, de manera que sigui

transparent per a ells en quin sistema estan les dades emmagatzemades. La imatge següent ens mostra el *path* de la taula Delta "trips", de la qual hem xerrat anteriorment: `dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled`

## Explorador de catàlegs [Enviar comentaris](#)

Created At	Wed Apr 09 10:36:40 UTC 2025
Last Access	UNKNOWN
Created By	Spark
Type	EXTERNAL
Location	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled
Properties	<ul style="list-style-type: none"> <li>delta:           <ul style="list-style-type: none"> <li>minReaderVersion: "1"</li> <li>minWriterVersion: "2"</li> </ul> </li> </ul>

**Imatge:** Path DBFS de la taula Delta trips

## Afegir dades al catàleg

Si tornam a la pàgina del catàleg, podem pitjar el botó "Añadir datos" (i després "Añadir datos" de nou) situat a la part superior dreta.

**Imatge:** Afegir dades al catàleg

Si pitjam a "Ver todo" veurem que tenim multitud de tipus de fonts de dades disponibles, incloent-hi bases de dades relacionals (MySQL, PostgreSQL, SQL Server o Oracle) o NoSQL (per exemple, MongoDB). També solucions d'emmagatzematge pròpies d'Azure com Azure Blob Storage o Azure Data Lake Storage.

Ens quedarem amb el primer botó de tots, "Crear o modificar una tabla", que ens permetrà carregar arxius de dades estructurades (CSV, JSON, XML, entre d'altres) en una nova taula Delta.

Per veure un exemple, anam a carregar el fitxer amb els [allotjaments d'Airbnb de Menorca](#) que hem utilitzat en altres lliuraments.

Añadir datos >

Crear o modificar tabla a partir de una carga de archivos

listings.csv cargado 584.70KB

Modo de vista previa Catalogo Esquema Nombre de la tabla Crear nueva tabla

hive\_metastore default listings Atributos avanzados

O Visualizando 50 filas, 18 columnas

id	name	host_id	host_name	neighbourhood_g	neighbourhood	latitude	longitude	roo
44085	Villa in Addaia great sea view ...	193043	Manuela	null	Es Mercadal	40.00974	4.19958	Entire hom
102558	appartment in Menorca -Cala ...	536023	Yolanda	null	Ciutadella de Menorca	39.9434	3.9613	Entire hom
294398	Son Esquella - Charming Men...	312145	Sol	null	Alaior	39.89082	4.11595	Entire hom
295705	The Most Perfect Villa in Men...	312145	Sol	null	Alaior	39.89028	4.11476	Entire hom
357026	Coqueto apartamento Son Bo...	1740878	Emilio	null	Alaior	39.8052	4.07072	Entire hom
358378	MENORCA APARTMENT SON ...	1807049	Miguel Angel	null	Alaior	39.90183	4.07595	Entire hom
366338	Nature Farmhouse, Private Po...	1849697	Alessandro	null	Ciutadella de Menorca	40.03581	3.96544	Entire hom
366636	Apartamento en Son Bou Men...	1851146	Carmen	null	Alaior	39.90128	4.07565	Entire hom
372756	Beautiful Farm House in Meno...	312145	Sol	null	Alaior	39.87626	4.13161	Entire hom
425853	Peaceful, comfortable apartm...	2117118	Jonathan	null	Es Mercadal	40.048770904541016	4.109414100646973	Entire hom
434190	Ideal para parejas	2155055	Maria Isabel	null	Ciutadella de Menorca	39.92443	3.82661	Entire hom
494480	Nice appartment in front of a ...	715601	Jose Luis	null	Ciutadella de Menorca	39.99876	3.79916	Entire hom
517301	Preciosa casa en el campo con...	2546568	Paqui	null	Mahón	39.93384	4.24331	Entire hom
518633	Rustic farmhouse	1273331	Anne	null	Es Castell	39.85917	4.29107	Entire hom
545844	The beach on your doorstep!	2684640	Maria Jose	null	Es Mercadal	40.02113	4.18331	Entire hom

Cerrar Crear tabla

**Imatge:** Importació d'un fitxer CSV

Aquí és important seleccionar `hive_metastore` en el catàleg, ja que les nostres dades han d'estar publicades en Hive. Pitjam "Crear tabla" i una vegada importat el fitxer, es crea la taula Delta `listings`, que pot ser referenciada mitjançant el seu path DBFS (`dbfs:/user/hive/warehouse/listings`), com qualsevol altra taula Delta.

## Explorador de catàlegs Enviar comentarios

Escribir para buscar...

Comparticiones de Delta recibidas

- > samples
- > Heredado
- > hive\_metastore
- > default
- listings

default >

listings

Información general Datos de muestra Detalles Historial

Created At	Wed Apr 09 11:06:46 UTC 2025
Last Access	UNKNOWN
Created By	Spark 3.5.0
Type	MANAGED
Location	dbfs:/user/hive/warehouse/listings
Is Managed Location	true
Properties	<ul style="list-style-type: none"> <li>delta:</li> <li>enableDeletionVectors: "true"</li> <li>feature.deletionVectors: "supported"</li> <li>minReaderVersion: "3"</li> <li>minWriterVersion: "7"</li> </ul>

**Imatge:** Taula Delta creada i la seva ruta DBFS

Podeu trobar més informació sobre Delta Lake i el concepte de *data lakehouse* a la documentació de Databricks en Azure:

<https://learn.microsoft.com/es-es/azure/databricks/lakehouse/>



### 3.5. Crear un quadern Databricks

Com ja sabem, els quaderns són una eina habitual en la ciència de dades i l'aprenentatge automàtic per a desenvolupar codi i presentar resultats. Databricks permet crear quaderns, amb un funcionament semblant als de Jupyter/Colab que ja coneixem. No obstant això, els quaderns de Databricks tenen algunes diferències importants respecte als de Jupyter/Colab: principalment, els quaderns de Databricks estan ja configurats per treballar directament sobre el framework Spark de Databricks i permeten emprar diversos llenguatges: Python, SQL, Scala i R.

Anem a crear el nostre primer quadern. En l'àrea de treball de Databricks, podem pitjar el botó blau "Nuevo" i seleccionar l'opció "Cuaderno". Alternativament, podem entrar en la secció de "Workspace" (on podrem veure els quaderns que anem creant) i pitjar el botó "Crear" i seleccionar "Cuaderno". De qualsevol de les dues maneres, ens crearà un nou quadern de Databricks. El primer que farem és donar-li un nom i seleccionar (al costat del nom) el llenguatge per defecte per al nostre quadern, en el nostre cas, Python.

Com hem comentat abans, ja tenim configurat tot l'entorn per fer feina amb Spark i les seves API. Així doncs, tenim disponible l'objecte **spark**, una instància de *SparkSession*, que ens servirà com a punt d'entrada únic, permet-nos interactuar amb totes les funcionalitats de Spark i els seus mòduls, incloent l'API de DataFrame.

Per tant, podem crear un dataframe que llegeixi les dades de la taula *trips* de les dades d'exemple dels taxis de Nova York. Podem fer-ho de dues maneres. La primera és mitjançant `spark.read.table`, emprant el nom complet de la taula (`samples.nyctaxi.trips`).

```
df = spark.read.table('samples.nyctaxi.trips')
```

tpep_pickup_datetime	tpep_dropoff_datetime	trip_distance	fare_amount	pickup_zip	dropoff_zip
2016-02-14 16:52:13	2016-02-14 17:16:04	4.94	19.0	10282	10171
2016-02-04 18:44:19	2016-02-04 18:46:00	0.28	3.5	10110	10110
2016-02-17 17:13:57	2016-02-17 17:17:55	0.7	5.0	10103	10023
2016-02-18 10:36:07	2016-02-18 10:41:45	0.8	6.0	10022	10017
2016-02-22 14:14:41	2016-02-22 14:31:52	4.51	17.0	10110	10282
2016-02-06 06:45:02	2016-02-05 06:50:26	1.8	7.0	10009	10065
2016-02-15 15:03:28	2016-02-15 15:18:45	2.58	12.0	10153	10199
2016-02-02 19:09:26	2016-02-25 19:24:50	1.4	11.0	10112	10069
2016-02-13 16:28:18	2016-02-13 16:36:36	1.21	7.5	10023	10153
2016-02-14 00:03:48	2016-02-14 00:10:24	0.6	6.0	10012	10003
2016-02-27 15:02:58	2016-02-27 15:08:31	2.02	8.0	10002	11211
2016-02-17 07:52:40	2016-02-17 08:01:21	1.5	8.0	10019	10199
2016-02-14 21:55:55	2016-02-14 22:01:31	0.93	6.0	10019	10018
2016-02-20 22:27:07	2016-02-05 22:39:44	2.34	10.5	10110	10014
2016-02-09 09:51:47	2016-02-09 09:57:27	0.91	5.5	10110	10199
2016-02-21 11:15:39	2016-02-21 11:40:24	11.6	33.5	10010	11371
2016-02-23 13:20:28	2016-02-23 13:36:25	1.4	11.0	10016	10022
2016-02-24 13:07:46	2016-02-24 13:26:13	2.43	13.0	10065	10119

Imatge: Lectura de dades en un quadern de Databricks

La segona manera és utilitzant el path DBFS (`dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled`). És convenient especificar també que les dades estan en el format de Delta:

```
df = spark.read.format("delta").load('dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled')
```

En el cas que vulguem carregar les dades de la taula *listings* que hem pujat anteriorment dins de Hive, també podem emprar aquestes dues maneres. Amb el nom complet de la taula (`hive_metastore.default.listings`):

```
allotjaments =
spark.read.table('hive_metastore.default.listings')
```

o amb la ruta DBFS (`dbfs:/user/hive/warehouse/listings`):

```
allotjaments = spark.read.format("delta").load('dbfs:/user/hive/warehouse/listings')
```

A partir d'aquí, podem emprar l'API de DataFrame, així com vulguem. Per exemple, si volem veure el nom dels allotjaments d'Es Castell:

```
allotjaments.filter("neighbourhood = 'Es Castell'").select("name").show()
```

O també ho podríem fer mitjançant el mètode `sql` de l'objecte `spark`:

```
allotjaments.createOrReplaceTempView("allotjaments")
spark.sql("SELECT name FROM allotjaments WHERE neighbourhood = 'Es Castell'").show()
```

Si volem guardar el nostre quadern en local, hem d'anar al menú "Archivo", seleccionar "Exportar" i triar l'opció "Archivo DBC". DBC és el format natiu dels quaderns de Databricks. En tot cas, també podem exportar-lo a quadern IPython (arxiu `.ipynb`), però pot no funcionar si el carregam en Colab o el nostre JupyterLab.

## Utilitzar SQL en el quadern

Hem configurat que el llenguatge del nostre quadern sigui Python. Però podem especificar que el llenguatge d'una cella determinada sigui un altre, en concret SQL. Ho feim afegint `%sql` al principi de la cella. Vegem com podem així executar una query SQL sobre la taula `trips`:

```
%sql
select * from samples.nyctaxi.trips
```

O també recuperar el nom de tots els allotjaments d'Es Castell:

```
%sql
select name from hive_metastore.default.listings where neighbourhood = 'Es Castell'
```

```
%sql
select name from hive_metastore.default.listings where neighbourhood = 'Es Castell'

(1) trabajos de Spark
_sqldf: pyspark.sql.dataframe.DataFrame = [name: string]
```

	A_B name
1	Rustic farmhouse
2	Fondouc
3	Villa Starbal
4	APARTAMENTO con vistas al mar.
5	Hort des Milord - Villa en Es Castell
6	Casa Bárbara
7	SANTA ANA
8	Starbal Villa
9	Trebaluger 3 bedroom villa, Es Castell
10	Preciosa casa en primera línea del mar
11	Villa with swimming pool and view to the sea
12	Family-friendly Villa in Menorca sea views & pool
13	Menorca retreat: lush gardens, fantastic views!
14	Holiday Villa "Ca Na Llum" from Menorca
15	Pardela Menorca Double Superior

↓ 52 filas | 0,47 s de tiempo de ejecución

Actualizado ahora

Este resultado se almacena como `_sqldf` y se puede utilizar en otras celdas de Python y SQL.

Imatge: Execució d'una query SQL en un quadern de Databricks

### Ordres màgiques (*magic commands*) i dbutils

Començant amb % tenim una sèrie d'ordres disponibles, com per exemple, la que hem vist (%sql) per canviar el llenguatge d'una cella. Se les denomina ordres màgiques o *magic commands*. Vegem-ne les més habituals:

Magic command	Descripció
%python %r scala %sql	Canvia el llenguatge d'una cella
%sh	Executa ordres del shell (només en el node driver, no en els workers)
%fs	Interactua amb el sistema de fitxers (a través de <code>dbutils.fs</code> , que veurem a continuació)
%md	Markdown, per donar estil a un text
%run	Executa un altre quadern des d'un quadern
%pip	Installa llibreries Python

Taula: Ordres màgiques de Databricks.

Podeu trobar més detalls a la documentació de databricks: <https://docs.databricks.com/en/notebooks/notebooks-code.html#language-magic>

També tenim disponible un objecte `dbutils`, que conté una col·lecció d'objectes que Databricks proporciona per ajudar a simplificar tasques comunes en un quadern de Databricks. Algunes de les utilitats més habituals són:

Utilitat	Descripció	Exemple
fs	Proporciona funcions per interactuar amb el sistema de fitxers de Databricks (DBFS). Permet realitzar operacions com llistar arxius, moure, eliminar, llegir i escriure arxius en DBFS.	<code>display(dbutils.fs.ls("/mnt/my-data/"))</code>

Utilitat	Descripció	Exemple
secrets	Permet accedir de manera segura a secrets emmagatzemats en Databricks Secrets Store. Aquests secrets poden incloure credencials d'accés, contrasenyes, claus API, etc.	<code>password = dbutils.secrets.get(scope = "myScope", key = "myKey")</code>
notebook	Proporciona funcions per interactuar amb altres quaderns, com per exemple executar un quadern des d'un altre quadern.	<code>dbutils.notebook.run("/Shared/my-other-notebook", 60, {"param": "value"})</code>
widgets	Ofereix funcions per crear i gestionar <i>widgets</i> en els notebooks. Aquests widgets poden ser utilitzats per recollir entrades d'usuari, com a selector de data, entrades de text, desplegables, etc.	<code>dbutils.widgets.text("input", "", "Enter your input")</code>
library	Permet gestionar biblioteques dins dels quaderns de Databricks, inclos la càrrega de biblioteques.	<code>dbutils.library.installPyPI("pandas")</code>

**Taula:** Utilitats de *dbutils*

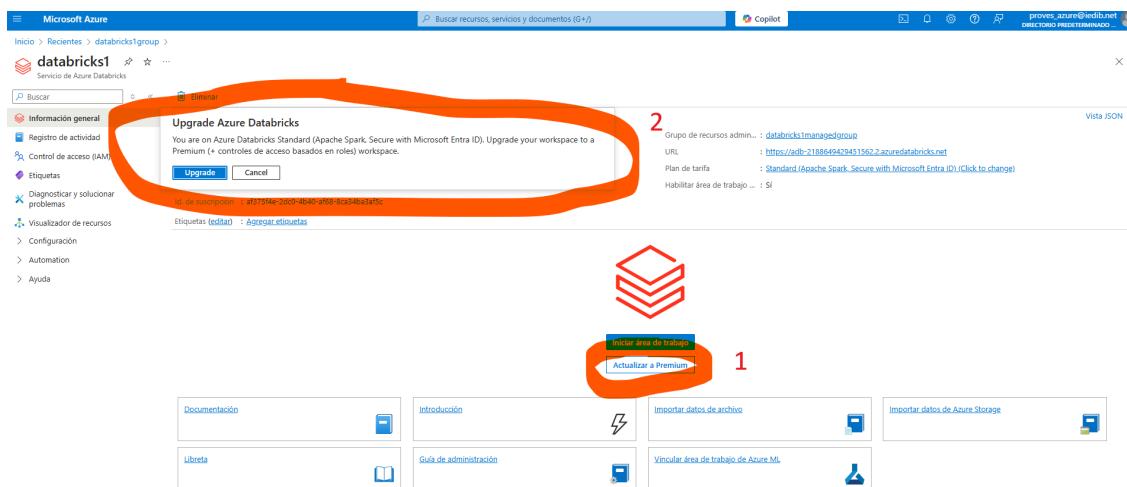
Podeu trobar una referència completa de *dbutils* a la documentació de Databricks: <https://docs.databricks.com/en/dev-tools/databricks-utils.html>

## 3.6. Databricks SQL

Databricks SQL és un magatzem de dades (*data warehouse*) sense servidor (*serverless*) constituit sobre l'arquitectura de Data Lakehouse de Databricks, molt eficient i que proporciona una interfície web per analitzar i visualitzar les dades (molt més senzilla que interactuar mitjançant quaderns).

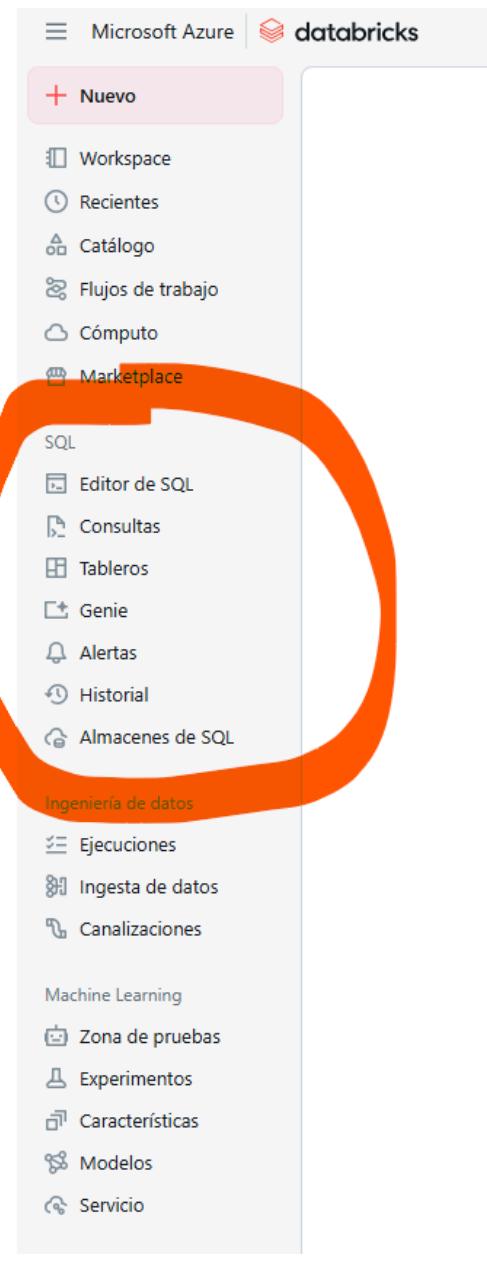
Quan es diu que Databricks SQL és *serverless* es fa referència a que no necessitam gestionar explícitament els servidors o recursos de computació. Abans hem creat un clúster, que en el nostre cas era de node únic (però que amb una altra llicència hauríem pogut configurar-lo amb un número determinat de nodes *worker*) i hem hagut d'especificar els detalls (RAM i CPU necessaris, versions de Scala i Spark) dels nodes. Els magatzems de Databricks SQL proporcionen un càlcul SQL instantani i elàstic (desacoblat de l'emmagatzematge) que s'escala automàticament per proporcionar concorrència il·limitada sense interrupció. Això permet executar totes les càrregues de treball BI i ETL d'una manera molt més eficient, amb una relació preu/rendiment fins a 12 vegades millor.

Per poder fer feina amb Databricks SQL necessitam **canviar el pla de tarifa** del nostre recurs d'Azure Databricks. Per fer-ho, hem d'anar a Azure Portal i obrir el nostre recurs *databricks1* (un servei d'Azure Databricks). Aquí hem de pitjar el botó "Actualizar a Premium" (també ho podem fer amb un clic sobre el pla de tarifa actual).



**Imatge:** Actualització a Premium

Una vegada fet l'*upgrade* (tarda una estona) podem tornar a pitjar "Iniciar àrea de treball" i tornam a la nostra àrea de treball de Databricks. Si ens fixam, al menú de l'esquerra ens ha aparegut una nova secció "SQL".



imatge: Secció SQL

Ara anam a definir un còmput serverless. Pitjarem "Nuevo" i en la secció de "Más", veurem que ara ens apareix una nova opció "Almacén de SQL" (aquesta opció no estava disponible amb la tarifa estàndard que teníem seleccionada anteriorment). També hi podem arribar des de l'opció "Almacenes de SQL" de la secció "SQL" i, a continuació, pitjant el botó "Crear almacén de SQL".

En la finestra que ens apareix hem de donar-li un nom al nostre magatzem SQL (per exemple databrick1sqlwarehouse) i seleccionar una mida. Per a les nostres proves ens bastarà l'opció més petita, "2X-Pequeño". I és important en el "Tipo" seleccionar l'opció Serverless, que apareix marcada per defecte. Fixau-vos que no hem hagut de seleccionar el número de nodes, ni quants nuclis i RAM necessitam, tot és reserva de manera dinàmica.

## Nuevo almacén de SQL

X

Nombre

Tamaño del clúster  4 DBU/h

Parada automática  Tras  minutos de inactividad

Expansión   clústeres (4 DBU)

Tipo  Serverless  Pro  Clásico

Opciones avanzadas

**Imatge:** Creació d'un magatzem SQL serverless

Podem veure que la creació d'aquest tipus de recurs de computació és molt més ràpida que la d'un clúster, pràcticament immediata.

Almacenes SQL >

## databrick1sqlwarehouse •

[Información general](#) [Detalles de conexión](#) [Monitoreo](#)

Estado	<span style="color: green;">● En ejecución</span>
Nombre	databrick1sqlwarehouse (ID: a1c19c69d84e5656)
Tipo	Serverless
Tamaño del clúster	2X-Pequeño
Terminación automática	Tras 10 minutos de inactividad
Expansión	Conteo de clústeres: activo 1 mín. 1 máx. 1
Canal	Actual (v 2025.10)
Creador	 Proves Azure

**Imatge:** Magatzem SQL serverless creat

Una vegada creat el nostre magatzem SQL, podem veure les eines que ens proporciona la secció "SQL".

### Editor de SQL

Aquí podem escriure una query SQL sobre les taules que tenim al nostre catàleg i executar-la.

The screenshot shows the Databricks interface with the SQL editor open. The query executed is:

```
1 |SELECT * FROM listings WHERE neighbourhood = 'Es Castell'
```

The results table displays 20 rows of data from the 'listings' table, filtered by 'neighbourhood' = 'Es Castell'. The columns include: id, name, host\_id, host\_name, neighbourhood\_group, neighbourhood, latitude, longitude, room\_type, price, minimum\_nights, number\_of\_reviews, last\_review, reviews\_per\_month, calculated\_host\_listings\_count, availability\_365, number\_of\_reviews\_l365d, and license.

At the bottom of the table, it says "Actualizado hace 1 minuto". Below the table, there's a note: "Desactivar el nuevo navegador de esquemas" and "0 2 s 586 ms | 52 filas devueltas".

**imatge:** Execució d'una query en l'editor de SQL

Podem guardar una query i ens quedarà en la secció "Consultas". També podem programar l'execució d'una query.

## Tableros

Aquesta opció ens permet construir un *dashboard* o quadre de comandament. Podem veure un exemple molt atractiu sobre les dades dels viatges de taxis de Nova York, pitjant a "Destacado" i seleccionant el primer exemple, "NYC Taxi Trip Analysis".

**imatge:** Quadre de comandaments dels viatges en taxi de Nova York

Una vegada tenim el nostre *dashboard* configurat, podem publicar-lo perquè altres usuaris de la nostra organització puguin visualitzar-lo. Hem d'assegurar-nos de deixar marcada l'opció "Incrustar credencials" perquè els altres usuaris puguin accedir-hi.



## Publicar NYC Taxi Trip Analysis (1)



Incrustar credenciales (default)  
⚠️ Permisos de datos compartidos

All viewers of the published dashboard use your data permissions to run queries. This enables a shared cache which can improve performance.



No incrustar credenciales  
🔒 Permisos de datos individuales

Cada espectador de un tablero publicado utiliza su propio permiso de datos para ejecutar consultas. Esto puede llevar a operaciones de refresh más frecuentes.

Personas con acceso

Los usuarios con al menos permisos de «Puede ver» pueden ver el tablero publicado. Haga clic [aquí](#) para cambiar los permisos de usuario.

proves\_azure@iedib.net, Admins, All workspace users

Notifique a los espectadores (incluye correos electrónicos a individuos y grupos pequeños)

Copiar link

Publicar

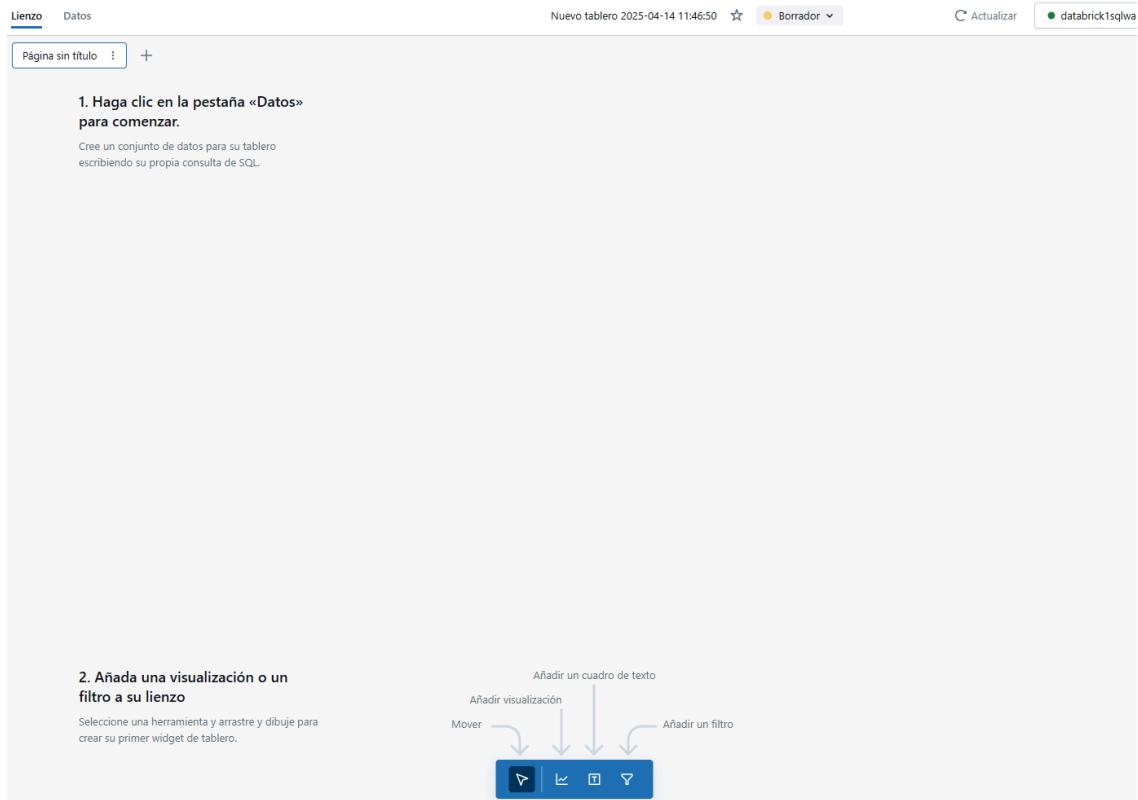
**Imatge:** Publicació del quadre de comandaments

En l'apartat següent veurem com construir aquest quadre de comandaments des de zero.

## 3.7. Dashboards amb Databricks SQL

En l'apartat anterior hem vist com carregar un quadre de comandament o *dashboard* de la galeria d'exemples sobre els viatges en taxis de Nova York. Ara construirem aquest *dashboard* des de zero.

Des de l'opció "Tableros" de la secció "SQL", hem de pitjar el botó "Create tablero". Ens apareix el nostre llenç (*lienzo* en castellà o *canvas* en anglès) en blanc. Podem veure que ens indica que, en primer lloc, haurem de configurar el conjunt de dades que volem emprar a partir d'una consulta SQL, i després podrem anar afegint elements visuals (en Databricks es denominen *visualizaciones* o *widgets*) al llenç.



**imatge:** Nou dashboard

Comencem creant el conjunt de dades. El que aquí es denomina conjunt de dades no és més que una vista sobre les dades, que cream a partir d'una sentència SELECT de SQL. Anirem a la pestanya "Datos" i pitjam el botó "Crear a partir de SQL". Ens crea un nou conjunt de dades (una vista) amb el nom "Datos sin nombre", al qual li podem canviar el nom per "nyctaxys" (amb el menú dels 3 punts). A continuació li podem assignar una consulta. Podem escriure-la directament o bé fer-ho navegant pel catàleg:

The screenshot shows the Databricks SQL interface. On the left, the Catalogo sidebar lists databases like samples, default, nyctaxi, and trips. A search bar at the top says "Escribir para buscar...". In the main area, a query is being run: "select \* from samples.nyctaxi.tripsamples.nyctaxi.trips". Below the query, a table titled "Tabla de resultados" displays the results. The table has columns: tpep\_pickup\_datetime, tpep\_dropoff\_datetime, trip\_distance, fare\_amount, pickup\_zip, and dropoff\_zip. The data consists of 21 rows of taxi trip information.

	tpep_pickup_datetime	tpep_dropoff_datetime	trip_distance	fare_amount	pickup_zip	dropoff_zip
1	2016-02-16T22:40:50.000+0000	2016-02-16T22:59:25.000+0000	5.35	18.5	10003	11238
2	2016-02-05T16:06:44.000+0000	2016-02-05T16:26:03.000+0000	6.5	21.5	10282	10001
3	2016-02-08T07:39:53.000+0000	2016-02-08T07:44:14.000+0000	0.9	5.5	10119	10003
4	2016-02-29T22:53:33.000+0000	2016-02-29T22:38:09.000+0000	3.5	13.5	10001	11222
5	2016-02-03T17:23:12.000+0000	2016-02-03T17:23:24.000+0000	0.3	3.5	10028	10028
6	2016-02-10T00:47:40:000+0000	2016-02-10T00:53:04:000+0000	0	5	10038	10005
7	2016-02-19T03:24:23.000+0000	2016-02-19T03:44:56.000+0000	6.57	21.5	10001	11377
8	2016-02-02T14:05:21.000+0000	2016-02-02T14:23:07.000+0000	1.08	11.5	10103	10167
9	2016-02-20T15:42:02.000+0000	2016-02-20T15:50:40.000+0000	0.8	7	10003	10011
10	2016-02-14T16:19:53.000+0000	2016-02-14T16:32:10.000+0000	1.3	9	10199	10020
11	2016-02-16T21:01:22.000+0000	2016-02-16T21:11:29.000+0000	2.74	11	10002	11211
12	2016-02-22T20:13:11:000+0000	2016-02-22T20:27:35.000+0000	2.09	11.5	10012	10009
13	2016-02-12T11:35:59.000+0000	2016-02-12T11:50:33.000+0000	2.8	13.5	10103	10016
14	2016-02-27T07:50:33.000+0000	2016-02-27T07:56:38.000+0000	1	6	10022	10110
15	2016-02-28T12:04:09.000+0000	2016-02-28T12:07:01.000+0000	0.7	4.5	10065	10021
16	2016-02-16T16:11:04:000+0000	2016-02-16T17:04:11.000+0000	11.54	40	10110	11371
17	2016-02-11T12:44:24.000+0000	2016-02-11T13:07:43.000+0000	3.8	18	10021	10027
18	2016-02-28T18:25:18.000+0000	2016-02-28T18:31:08.000+0000	0.98	6	10119	10009
19	2016-02-13T20:00:15.000+0000	2016-02-13T20:17:20.000+0000	2.83	13.5	10003	10282
20	2016-02-21T01:07:04.000+0000	2016-02-21T01:09:43.000+0000	0.6	4	10018	10018
21	2016-02-27T15:56:54.000+0000	2016-02-27T16:04:51.000+0000	1.53	8	10023	10025
22	2016-02-18T07:40:20.000+0000	2016-02-18T07:40:51.000+0000	0.03	4	10001	10167

**imatge:** Assignam una consulta SQL al conjunt de dades

Acabam pitjant "Ejecutar".

Ara ja podem tornar al llenç, on a la part inferior trobam una barra d'eines de color blau. Pitjarem la segona icona de la barra per afegir un nou element visual (*visualización* o *widget*) al *dashboard*. Situam el *widget* on volem i el configuram amb el panell de la dreta.

## Diagrama de barres

Començarem fent un diagrama de barres per veure els moments del mes amb més activitat (segons l'hora de recollida). Per tant, en "*Visualización*" hem de seleccionar l'opció "*Barras*".

A l'eix X hem de seleccionar la columna *tpep\_pickup\_datetime* (data i hora de recollida, d'inici del viatge) i fent-hi clic configurarem l'escala com a temporal, ho agruparem per hores i li posarem el "*Display name*" com a "*Pickup Hour*".

The screenshot shows the configuration of a bar chart in Tableau. The X-axis is labeled 'Eje X' and has a dropdown menu showing 'HOURLY(tpep\_pickup\_dat...'. The Y-axis is labeled 'Eje Y' and has a dropdown menu showing 'COUNT(\*)'. A modal window is open for the X-axis configuration, titled 'Campo'. It shows the selected field 'tpep\_pickup\_datetime' and its type as 'Continuo'. Under 'Transformar', 'HOURLY' is selected. The 'Nombre en pantalla' field is set to 'Pickup Hour'.

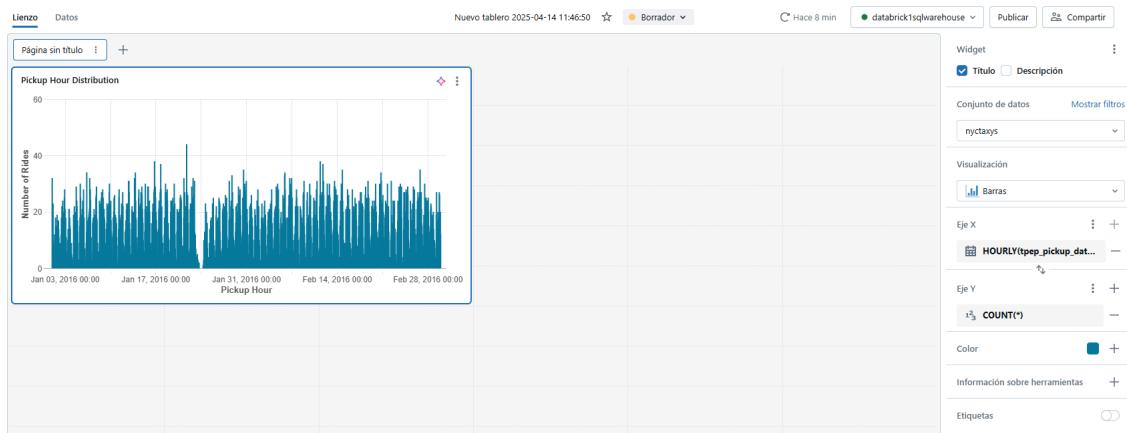
**Imatge:** Configuració de l'eix X del diagrama de barres

Per a l'eix Y, en lloc de triar una columna, ara anam a seleccionar "COUNT(\*)", per saber el número total de viatges en cada franja horària. Al "Display name" li podem posar "Number of Rides".

The screenshot shows the configuration of a bar chart in Tableau. The X-axis is labeled 'Eje X' and has a dropdown menu showing 'HOURLY(tpep\_pickup\_dat...'. The Y-axis is labeled 'Eje Y' and has a dropdown menu showing 'COUNT(\*)'. A modal window is open for the Y-axis configuration, titled 'Campo'. It shows the selected field 'COUNT(\*)' and its type as 'Continuo'. The 'Nombre en pantalla' field is set to 'Number of Rides'.

**Imatge:** Configuració de l'eix Y del diagrama de barres

Finalment, podem posar-li un títol ("Pickup Hour Distribution") al widget, activant el checkbox "Título" i escrivint el títol directament sobre el widget.



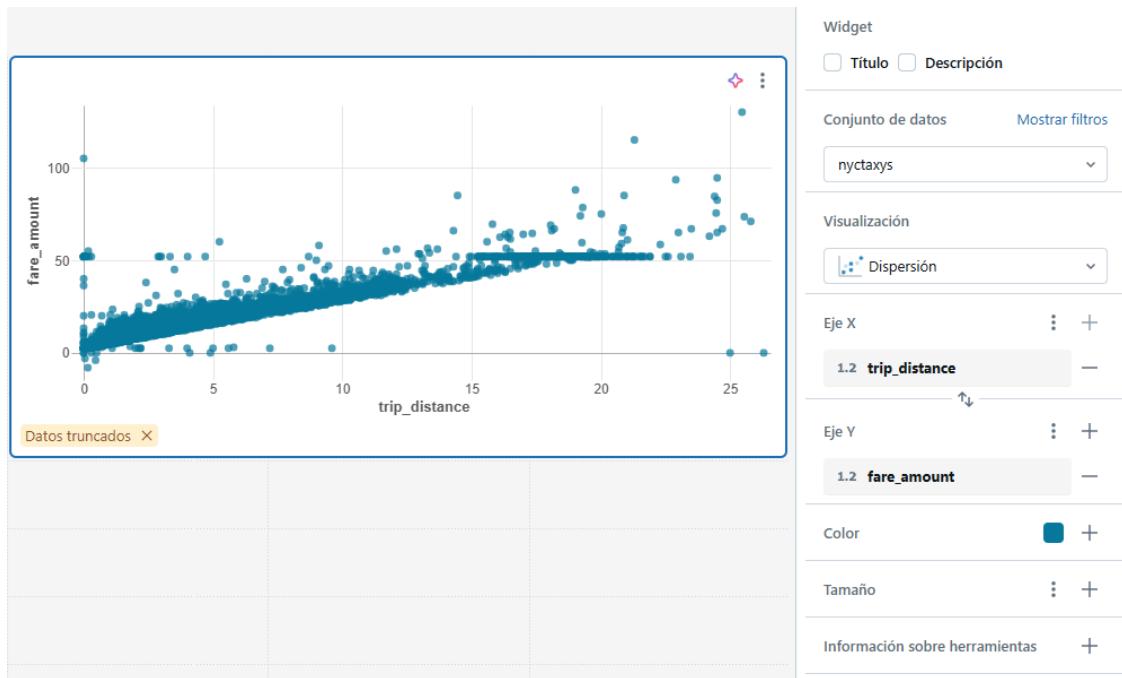
**Imatge:** Visualització final del diagrama de barres en el llenç

Per fer el diagrama de barres de la distribució horària de les hores d'acabament del viatge (*dropoff*), seguiríem el mateix procés, però canviant el color del gràfic. Si seguim el model de la galeria d'exemples, triaríem el groc.

## Diagrama de dispersió

Veim que el *widget* central del *dashboard* d'exemple és un diagrama de dispersió. Anem a crear un altre *widget* amb la segona icona de la barra d'eines i seleccionam "Dispersión" en el tipus de visualització.

En l'eix X posarem *trip\_distance* amb una escala quantitativa i sense aplicar-hi cap transformació. En l'eix Y posarem *fare\_amount*, també amb escala quantitativa i sense cap transformació.



**Imatge:** Diagrama de dispersió ( primera versió)

Veim que aquí ens surten tots els punts del mateix color, mentre que en l'exemple de la galeria ens apareix un color diferent per a cada dia de la setmana. Podríem posar *tpep\_pickup\_datetime* al "Color/Group by", però el problema és que no tenim cap opció automàtica per extreure'n els dies de la setmana.

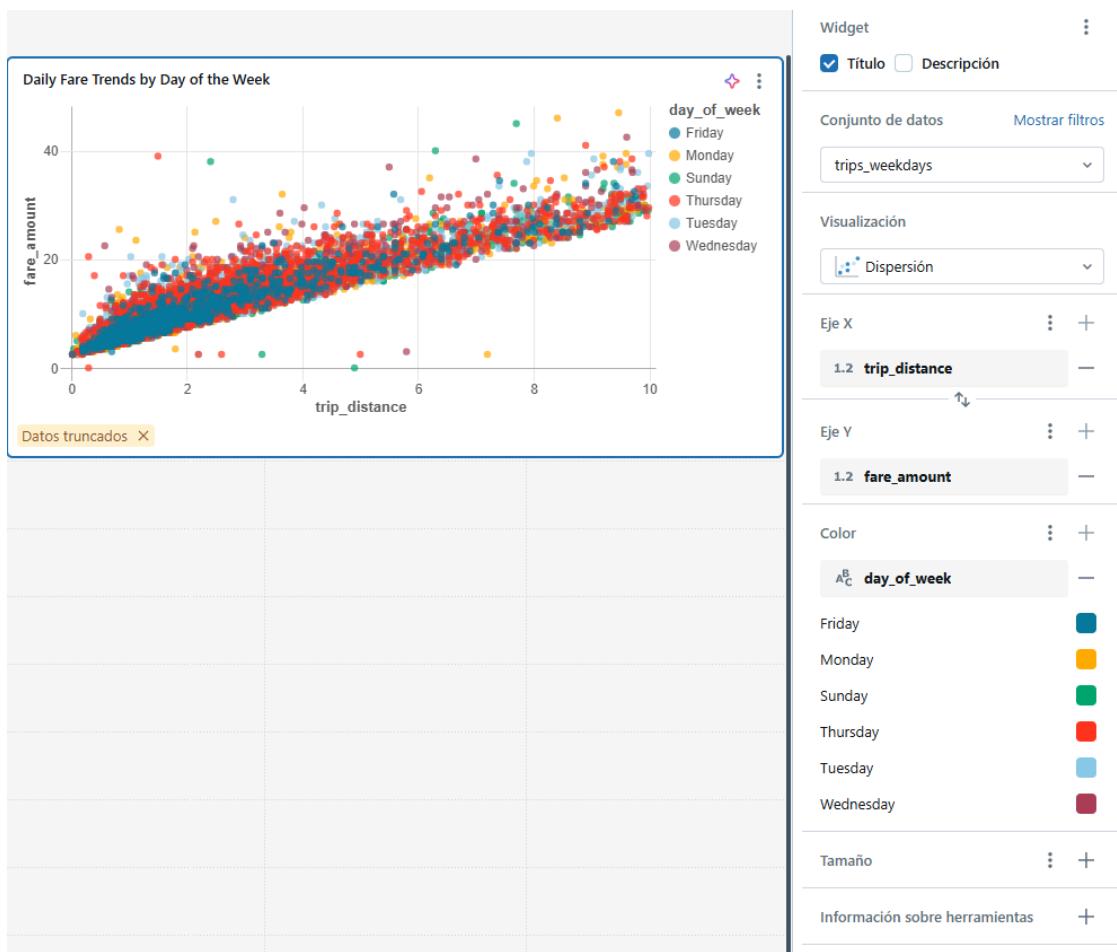
La solució és definir un nou conjunt de dades (li direm *trips\_weekdays*), emprant la següent consulta SQL que ens afegeix dues columnes *weekday* i *day\_of\_week*, respectivament amb el número de dia (1 per al diumenge, 2 per al dilluns, etc.) i el nom del dia (*sunday*, *monday*, etc.), a partir de la data i hora de recollida.

```

SELECT
    T.tpep_pickup_datetime,
    T.tpep_dropoff_datetime,
    T.fare_amount,
    T.pickup_zip,
    T.dropoff_zip,
    T.trip_distance,
    T.weekday,
    CASE
        WHEN T.weekday = 1 THEN 'Sunday'
        WHEN T.weekday = 2 THEN 'Monday'
        WHEN T.weekday = 3 THEN 'Tuesday'
        WHEN T.weekday = 4 THEN 'Wednesday'
        WHEN T.weekday = 5 THEN 'Thursday'
        WHEN T.weekday = 6 THEN 'Friday'
        WHEN T.weekday = 7 THEN 'Saturday'
        ELSE 'N/A'
    END AS day_of_week,
FROM
(
    SELECT
        dayofweek(tpep_pickup_datetime) as weekday,
        *
    FROM
        `samples`.`nyctaxi`.`trips`
    WHERE
        trip_distance > 0
        AND trip_distance < 10
        AND fare_amount > 0
        AND fare_amount < 50
) T
ORDER BY
    T.weekday

```

Una vegada afegit això, hem de canviar el conjunt de dades associat al nostre diagrama de dispersió i seleccionar *trip\_weekdays*. I ara ja sí que podem posar *day\_of\_week* en "Color/Group by". Podem afegir-li també el títol "Daily Fare Trends by Day of the Week"



Imatge: Diagrama de dispersió (versió final)

## Taula

Per construir la taula amb els ingressos (revenues) per rutes, haurem de crear un altre conjunt de dades (li direm *routes\_revenues*), amb aquesta consulta SQL:

```

SELECT
    T.pickup_zip,
    T.dropoff_zip,
    T.route AS `Route`,
    T.frequency AS `Number Trips`,
    T.total_fare AS `Total Revenue`
FROM
(
    SELECT
        pickup_zip,
        dropoff_zip,
        concat(pickup_zip, '-', dropoff_zip) AS route,
        count(*) AS frequency,
        SUM(fare_amount) AS total_fare
    FROM
        `samples`.`nyctaxi`.`trips`
    GROUP BY
        1,2,3
) T
ORDER BY
    1 ASC

```

Una vegada creat el conjunt de dades, afegim un nou *widget* i seleccionam *routes\_revenues* com a conjunt de dades i "Tabla" com a tipus de visualització. Només ens queda activar les columnes que volem que s'incloguin en la taula (amb la icona de l'ull) i ordenar-les al nostre gust. També li afegirem el títol "Route Revenue Attribution".

The screenshot shows a dashboard interface. On the left is a table titled "Route Revenue Attribution" with columns: Route, Number Trips, and Total Revenue. The data includes rows like 7002-7002 (2 trips, 21.00), 7030-7030 (1 trip, 40.00), etc. On the right is a configuration sidebar for a "Widget". Under "Widget", "Título" is checked. In the "Conjunto de datos" section, "routes\_revenues" is selected. Under "Visualización", "Tabla" is chosen. The "Columns" section lists columns from the data source: pickup\_zip, dropoff\_zip, Route, Number Trips, and Total Revenue. The "Grid" tab is also visible.

imatge: Taula

## Comptador

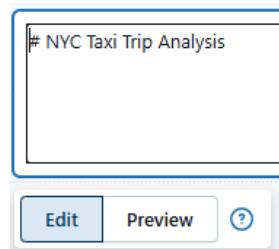
Volem incorporar un comptador del nombre total de viatges. Afegim un nou widget, podem seleccionar *nyctaxis* com a conjunt de dades i "Contador" com a tipus de visualització. A "Valor" hem de seleccionar "COUNT(\*)". També li posarem el títol "Total Trips".

The screenshot shows a dashboard interface. On the left is a large digital-style display showing the number "21,93 mil". On the right is a configuration sidebar for a "Widget". Under "Widget", "Título" is checked. In the "Conjunto de datos" section, "nyctaxis" is selected. Under "Visualización", "Contador" is chosen. The "Valor" section contains the expression "COUNT(\*)". There is also an "Objetivo" section with a plus sign.

imatge: Comptador

## Títol

Per fer el títol, afegirem un quadre de text mitjançant el tercer botó de la barra d'eines de color blau. Al quadre que ens apareix hi podem escriure el text que vulguem donant-li format amb Markdown. Per exemple, amb # hi posam una capçalera de nivell 1.



Imatge: Títol

## Afinadors o filters

Ens queda definir els afinadors per filtrar les dades que es mostren en cada un dels *widgets*. Per afegir-ne un de nou, ho feim mitjançant el quart botó de la barra d'eines de color blau.

Començarem definint el del codi postal de recollida (*pickup\_zip*). A "Fields", hem de pujar el botó "+" i seleccionar la columna *pickup\_zip* de *nyctaxis*. Podem deixar-ho com a valor únic o permetre que l'usuari en seleccioni diversos codis postals, canviant el filtre a "Valors múltiples"

**Widget**

Título  Descripción

**Filtro**

Valores múltiples

**Campos**

nyctaxis.pickup\_zip

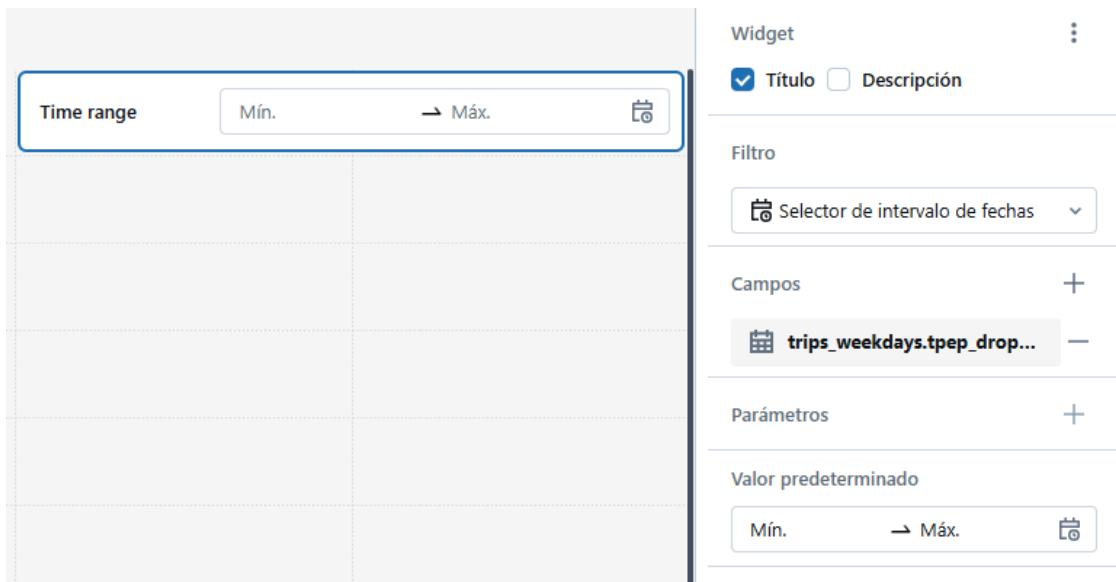
**Parámetros**

**Valor predeterminado**

Imatge: Filtre de valors múltiples

Crearem un altre afinador per al codi postal de destinació (*dropoff\_zip*) de la mateixa manera.

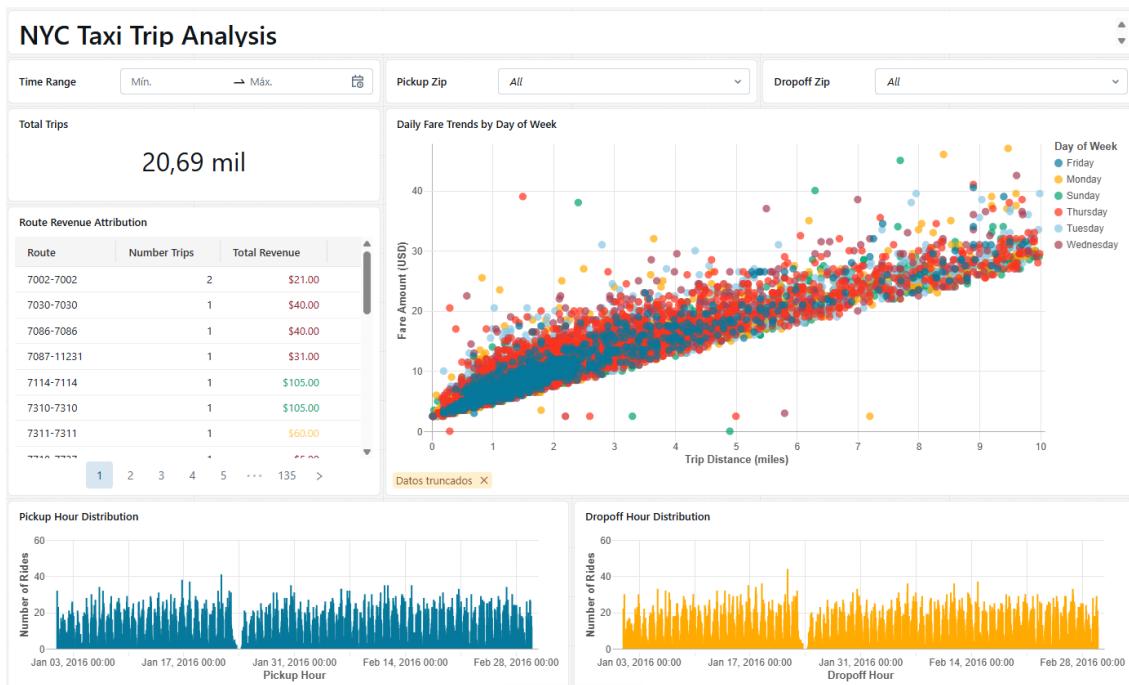
L'altre filtro que tenim en l'exemple és de tipus temporal, per a la columna *dropoff\_datetime*. Afegim un altre filtre, per al qual seleccionarem "Selector de intervalo de fechas" com a tipus de filtre i *dropoff\_datetime* en Fields.



Imatge: Filtre d'interval de dates

## Composició final

Només ens queda ajustar les mides de cada widget i col·locar-los tots com ens agradi per tenir la versió final del nostre *dashboard*, que ja podrem publicar.



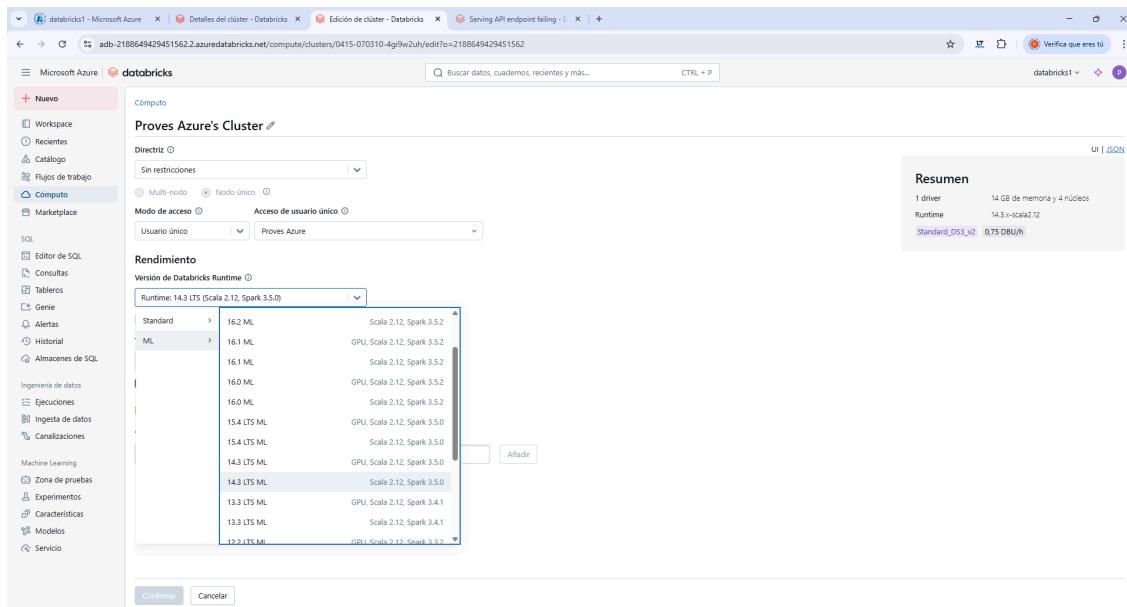
Imatge: Versió final del dashboard

## 3.8. Machine Learning

Databricks proporciona capacitats per a treballar en l'entrenament, reproducció, avaluació i desplegament de models d'aprenentatge automàtic a gran escala.

Anteriorment hem vist com crear un clúster Spark en Databricks per a l'anàlisi de dades. Recordem que havíem de seleccionar una imatge base per als nodes del clúster. És el que s'anomena Databricks runtime. En aquell moment vàrem triar el *runtime* 15.4 LTS, que incorpora Scala 2.12 i Spark 3.5.0. Però Databricks proporciona uns runtimes específics per fer feina en un entorn d'aprenentatge automàtic. És l'anomenat Machine Learning Runtime (MLR), un *runtime* optimitzat per a l'aprenentatge automàtic que, a més de Scala i Spark, ja du pre-installades diverses llibreries específiques per a Machine Learning i Deep Learning, com Spark MLlib, Scikit-Learn, Tensorflow, PyTorch o Keras. A més, podem seleccionar si volem que les nostres màquines virtuals facin feina amb acceleració per GPU o no.

En la imatge següent podem veure alguns dels *runtimes* disponibles quan cream un nou clúster per a ML (en aquest cas de node únic). En concret, seleccionarem la versió 14.3 LTS ML (amb Scala 2.12 i Spark 3.5.0), sense GPU, ni acceleració Photon. N'hi ha una versió LTS més moderna, però dona alguns problemes de llibreries quan volem generar un servei REST per al model. Com a tipus de node, podem seleccionar Standard\_DS3\_v2, amb 14 GB de RAM i 4 nuclis. El cost associat és de 0,75 DBU/h.



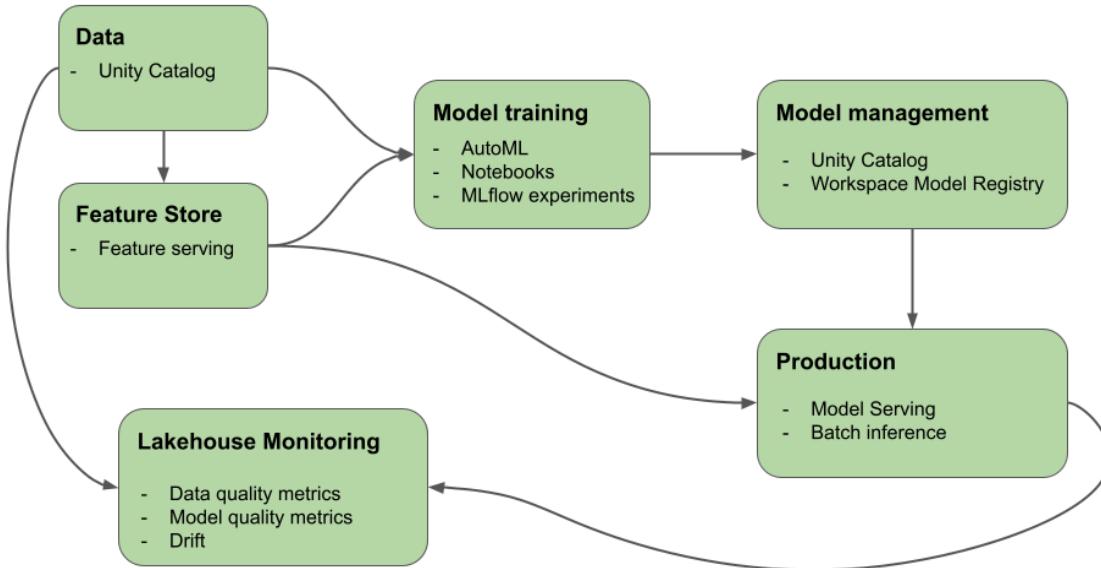
**Imatge:** Creació d'un clúster amb MLR

Podem consultar tots els detalls de la configuració d'aquest *runtime* concret, amb les versions de cada una de les llibreries installades, a <https://learn.microsoft.com/en-us/azure/databricks/release-notes/runtime/14.3lts-ml>.

Una vegada hem creat un clúster Spark amb un MLR, podem crear un quadern i escriure el nostre codi utilitzant les llibreries que hem mencionat. A més, quan treballam amb Databricks en Azure, també podem utilitzar AutoML (que vàrem veure en el lliurament 4 de Programació d'intel·ligència artificial) per entrenar ràpidament models predictius, sobre el nostre clúster Spark.

Una altra característica de Databricks relacionada amb l'aprenentatge automàtic és l'ús de MLflow. MLflow és una plataforma open source desenvolupada per Databricks que està dissenyada per ajudar a gestionar el cicle de vida complet del ML, incloent-hi l'experimentació, la reproducció de resultats, l'organització del desplegament, i el monitoratge dels models de ML. L'objectiu de MLflow és simplificar tot el procés complex del ML, facilitant als científics de dades i enginyers la tasca de passar dels experiments a la producció de manera ràpida i eficaç. Les darreres actualitzacions de MLflow introduceixen funcionalitats per a la gestió de IA generativa i LLM (models de llenguatge extens). Azure Databricks integra MLflow directament, cosa que permet als usuaris registrar experiments, gestionar models, i desplegar-los en producció dins de l'ecosistema Azure.

La imatge següent mostra com tots els components que hem mencionat treballen de manera conjunta per ajudar l'usuari a implementar el procés de desenvolupament i desplegament de models.



Imatge: Components relacionats amb ML en Databricks

Per entendre millor el funcionament del ML en Databricks, anam a crear un experiment de ML amb AutoML sobre el nostre clúster. Treballarem amb el conegut dataset de flors iris. Recordem que es tracta de les dades d'unes flors del gènere Iris (n'hi ha 3 espècies diferents), amb 4 variables (longitud i amplada del pètal i del sèpal) i que és un dels datasets més populars per a provar models de classificació. Podem descarregar-lo des de <https://raw.githubusercontent.com/tnavarrete-iedib/bigdata-24-25/refs/heads/main/iris.csv>. A continuació, carregarem les dades com una taula Delta en el nostre catàleg (dins del hive\_metastore), seguint les passes que hem vist a l'apartat 3.4.

Per començar el nostre experiment ML, hem d'entrar en l'opció "Experimentos" de la secció "Machine Learning" (o pitjar "Nuevo" i "Experimento"). Aquí hem de seleccionar classificació com el tipus de model de ML que volem entrenar (hem de pitjar "Iniciar entrenamiento en el bloc "Classificación"). A continuació, hem de seleccionar el nostre clúster ML, les dades d'entrada per a l'entrenament, pitjant el botó "Explorar" i triant la nostra taula Delta *iris*. Deixam marcades les 4 columnes d'entrada amb les longituds i amplades dels pètals i sèpals i triam *Species* com a la columna objectiu. A més, en les opcions de configuració avançada podem canviar la mètrica d'avaluació del model i seleccionar l'exactitud (*accuracy*). També podem limitar la durada de l'experiment a 10 minuts, atès que el dataset és molt petit i en aquest temps serà capaç de trobar algun model amb una exactitud molt propera a 1.

Experiments >

### Configurar el experimento de clasificación

1 Configurar 2 Unir características 3 Entrenar 4 Evaluar

**Configuración del cómputo**

- \* Cluster (Databricks Runtime 9.1 LTS ML o superior) ○  
Provee Azure's Cluster

**Configuración del experimento**

- \* Datos de entrada para entrenamiento  
Explorar default.iris
- \* Columna objetivo a predecir  
Species
- \* Nombre del experimento ○  
Species\_iris-2025\_04\_15-09\_25

**Configuración avanzada (opcional)**

- \* Métrica de evaluación ○  
Exactitud
- Directorio de experimentos ○  
/Users/proves\_azure@iedib.net/databricks\_automl/
- \* Frameworks de entrenamiento ○  
lightgbm X sklearn X xgboost X
- \* Tiempo de espera (minutos) ○  
10

Columna de tiempo para la separación de entrenamiento/validación/pruebas ○  
Seleccionar una columna de tiempo

Unir características (opcional) > Iniciar AutoML >

**Imatge:** Configuració de l'experiment d'AutoML

Finalment, pitjam el botó "Iniciar AutoML" per iniciar l'experiment.

Veurem que el procés comença preparant l'entorn i a continuació fa l'entrenament de diversos models, en diferents execucions amb diferents hiperparàmetres. Per a cada un d'ells podem veure el valor de la mètrica que hem triat. Podem veure que en els primers minuts ja troba models amb una exactitud molt propera a 1. Podriem aturar ja l'entrenament, però deixarem que passin els 10 minuts i acabi. La imatge següent en mostra el resultat:

Experiments >  
**Species\_iris-2025\_04\_15-09\_25** ○ Send feedback Add Description

1 AutoML 2 Configurar 3 Entrenar 4 Evaluar

Conjunto de datos de entrenamiento: default.iris  
Categoría de destino: Species  
Métrica de evaluación: val\_accuracy\_score  
Interrupción por tiempo: 10 minutos

Evaluación AutoML ○ completado  
Se han completado todos las ejecuciones y se han añadido a la tabla que figura a continuación. Haga clic en una ejecución específica para ver los detalles.

Modelo con la mejor val\_accuracy\_score  
El modelo está listo para registrarse e implementarse. O bien, acceda al código fuente del entrenamiento del modelo para realizar modificaciones haciendo clic en un cuaderno bajo la columna Source en la tabla siguiente.

Ver cuaderno del mejor modelo Ver cuaderno de exploración de datos

Los cuadernos generados por AutoML ahora se guardan como artefactos de Mlflow. Haga clic aquí para obtener más información.

Ejecuciones Evaluación Preview Rastros

Run Name	Created	Dataset	Duration	Source	Models	Metrics	Tags
abrasive-carp-803	Hace 5 minutos	dataset (32f4da30) Training... [+2]	1.0min	-	sklearn	0.96875	logistic_regr...
angry-lark-337	Hace 9 minutos	dataset (32f4da30) Training... [+2]	1.2min	Notebook	sklearn	0.96875	logistic_regr...
flawless-carp-820	Hace 3 minutos	dataset (32f4da30) Training... [+2]	1.0min	-	sklearn	0.9375	random_for...
calm-perch-399	Hace 3 minutos	dataset (32f4da30) Training... [+2]	57.2s	-	sklearn	0.9375	decision_tre...
judicious-carp-684	Hace 5 minutos	dataset (32f4da30) Training... [+2]	1.0min	-	sklearn	0.9375	decision_tre...
bittersweet-dove-315	Hace 6 minutos	dataset (32f4da30) Training... [+2]	57.3s	-	sklearn	0.9375	random_for...
fearless-sow-439	Hace 6 minutos	dataset (32f4da30) Training... [+2]	1.1min	-	sklearn	0.9375	decision_tre...
crawling-hound-396	Hace 8 minutos	dataset (32f4da30) Training... [+2]	1.3min	-	sklearn	0.9375	random_for...
traveling-boar-896	Hace 8 minutos	dataset (32f4da30) Training... [+2]	1.0min	-	sklearn	0.9375	decision_tre...

21 ejecuciones coincidentes

**Imatge:** Resultat de l'entrenament

Veim que de totes les execucions que ha fet (de diversos models amb diferents hiperparàmetres), n'hi ha dues amb una precisió de 0.96875, ambdues amb models de regresió logística. En concret, ha considerat que la millor és la que té el nom *angry-lark-337*. Podem fer clic sobre el seu quadern (a "Notebook" a la columna "Source" de la taula, o en el botó "Ver cuaderno del mejor modelo") i veure un quadern amb tot el procés d'entrenament d'aquest model. El quadern finalitza amb la matriu de confusió.

També podem veure un altre quadern (botó "Ver cuaderno de exploración de datos"), que conté nombrosos gràfics i indicadors estadístics sobre les dades d'entrada, com per exemple, la correlació entre les 4 variables d'entrada.

Si feim clic sobre el nom del model, podrem veure els detalls del model, amb els seus hiperparàmetres i els valors de les mètriques.

The screenshot shows the Databricks interface for a selected experiment. The top navigation bar includes 'Experiments > /Users/proves\_azure@iedib.net/databricks\_autom/Species\_iris-2025\_04\_15-09\_25 > angry-lark-337'. Below the navigation are tabs for 'Información general', 'Métricas del modelo', 'Métricas del sistema', 'Rastros', 'Resultados de la evaluación', and 'Artefactos'. A 'Send feedback' button is also present. On the right, there are buttons for 'Reproducir la ejecución' and 'Registrar modelo'. The main content area displays detailed information about the model's creation, creator, execution ID, state, duration, datasets used, and the training pipeline. It also lists registered models and their types. Two side panels are visible: 'Parámetros (61)' and 'Métricas (24)', each with search bars and tables of data.

**Imatge:** Detalls del model seleccionat

Podem registrar aquest model, pitjant el botó "Registrar modelo". Li donarem el nom *iris-logistic-regression*.

The dialog box has a title 'Registrar modelo' and a close button 'X'. The first section is 'Model' with a dropdown menu containing '+ Create New Model'. The second section is 'Model Name' with an input field containing 'iris-logistic-regression'. At the bottom are two buttons: 'Cancelar' (Cancel) and 'Registrar' (Register).

**Imatge:** Registre del model

Aquí podem veure el model ja registrat (ens queda disponible en "Modelos" de la secció "Machine Learning"), entrant en la "Versión 1":

Modelos registrados > iris-logistic-regression >

### Versión 1

Registrado en: 15 abr 2025, 09:41 Creador: proveis\_azure@iedib.net Estatus del seguimiento: Siguéndose Último cambio: 15 abr 2025, 09:41 Ejecución de origen: angry-lark-337 Estadio: None

> Descripción Editar

> Solitudes pendientes

Solicitud	Solicitud realizada por	Acciones
No hay ninguna solicitud pendiente.		

> Etiquetas

> Esquema

Nombre Tipo

- Entradas (4)
 

Sepal length (required)	double
Sepal width (required)	double
Petal length (required)	double
Petal width (required)	double
- Salidas (1)
 

- (required)	Tensor (dtype: object, shape: [-1])
--------------	-------------------------------------

> Actividades

**Imatge:** Model registrat (versió 1)

I una vegada ja l'hem registrat, podem treballar amb ell per fer prediccions, pitjant el botó "Usar modelo para inferencia". Aquí tenim tres opcions per configurar la inferència. Nosaltres triarem la primera ("En tiempo real"), per crear un *endpoint* o punt de servei per a interactuar amb el model en temps real.

## Configurar la inferencia del modelo Send feedback

X

[En tiempo real](#) [Transmisión \(Delta Live Tables\)](#) [Inferencia Batch](#)

Servir este modelo a través de un punto. Para configurar varios modelos servidos desde este punto, actualice su configuración a través de la página de detalles del punto. [Más información](#)

### Versión del modelo

Version 1

### Nombre del punto

iris

### Cómputo

Small

4 concurrency (4 DBU)

Escalar a cero

[Cancelar](#)

[Crear endpoint](#)

**Imatge:** Creació d'un endpoint per a prediccions

El procés tardarà una bona estona, en el meu cas uns 45 minuts. El resultat és que ha creat un punt de servei a <https://adb-729002784819399.19.azuredatabricks.net/serving-endpoints/iris/invocations>

Puntos de servicio >

### iris

Estado del punto de servicio: Listo

Creator: Proves Azure

URL: <https://adb-2188649429451562.azuredatabricks.net/serving-endpoints/iris/involciones>

Etiquetas:

Política de presupuestos serverless:

Optimización de rutas: Desactivado

⋮ Permisos Editar Detener Usar

**Configuración activa**

Entidad	Versión	Nombre	Estado	Cómputo	Tráfico, %
iris-logistic-regression	Versión 1	iris-logistic-regression-1	<span style="color: green;">Listo</span>	CPU, Small 4 concurrency (4 DBU)	100

**Eventos**

Métricas	Eventos	Logs

Timestamp

Tipo de evento

Nombre de la entidad servida

Mensaje

15 abr 2025, 09:53	ENDPOINT\_EVENT		Endpoint 'iris' entered READY state.
15 abr 2025, 09:53	ENDPOINT\_UPDATE\_EVENT		Endpoint update succeeded for endpoint 'iris', config version 1.
15 abr 2025, 09:53	SERVED\_ENTITY\_SERVICE\_EVENT	iris-logistic-regression-1	Served entity creation succeeded for served entity 'iris-logistic-regression-1', config version 1.
15 abr 2025, 09:53	SERVED\_ENTITY\_SERVICE\_EVENT		Served entity 'iris-logistic-regression-1' entered DEPLOYMENT\_READY state.
15 abr 2025, 09:50	SERVED\_ENTITY\_SERVICE\_EVENT		Served entity 'iris-logistic-regression-1' entered DEPLOYMENT\_CREATING state: Deploying
15 abr 2025, 09:49	SERVED\_ENTITY\_SERVICE\_EVENT		Served entity 'iris-logistic-regression-1' entered DEPLOYMENT\_CREATING state: Provisioning resources
15 abr 2025, 09:49	SERVED\_ENTITY\_CONTAINER\_EVENT	iris-logistic-regression-1	Container image creation finished successfully
15 abr 2025, 09:44	SERVED\_ENTITY\_CONTAINER\_EVENT	iris-logistic-regression-1	Container image creation initiated
15 abr 2025, 09:44	SERVED\_ENTITY\_SERVICE\_EVENT		Served entity 'iris-logistic-regression-1' entered DEPLOYMENT\_CREATING state: Pending Container Creation
15 abr 2025, 09:44	SERVED\_ENTITY\_CREATION\_EVENT	iris-logistic-regression-1	Served entity created for served entity 'iris-logistic-regression-1', config version 1.
15 abr 2025, 09:43	ENDPOINT\_UPDATE\_EVENT		Endpoint updated by proves\_azure@iedib.net.
15 abr 2025, 09:43	ENDPOINT\_CREATION\_EVENT		Endpoint created by proves\_azure@iedib.net.

Imatge: endpoint creat

Si pitjam el botó "Usar", podem enviar una petició al nostre endpoint. En aquest cas hem enviat el JSON següent amb els valors de 3 flors diferents i el model ha predit de quina espècie és cada una.

```
{
  "dataframe_split": {
    "columns": [
      "Sepal length",
      "Sepal width",
      "Petal length",
      "Petal width"
    ],
    "data": [
      [
        5.1,
        3.5,
        1.4,
        0.2
      ],
      [
        6.4,
        3.2,
        4.5,
        1.4
      ],
      [
        7.1,
        2.6,
        6.0,
        2.0
      ]
    ]
  }
}
```

## Consultar punto de servicio

X

Navegador    Curl    Python    SQL

**Solicitud** ?

```
{
  "dataframe_split": {
    "columns": [
      "Sepal length",
      "Sepal width",
      "Petal length",
      "Petal width"
    ],
    "data": [
      [
        5.1,
        3.5,
        1.4,
        0.2
      ],
      [
        6.4,
        3.2,
        4.5,
        1.5
      ]
    ]
  }
}
```

**Respuesta** ?

desde iris-logistic-regression-1

```
{
  "predictions": [
    "Iris-setosa",
    "Iris-versicolor",
    "Iris-virginica"
  ]
}
```

5c

Enviar petición    Restablecer ejemplo

Imatge: Petició a l'endpoint

Si pitjam la pestanya Python, podem veure el codi per interactuar amb l'endpoint, que podríem utilitzar des d'un quadern o una aplicació externa (hauríem de configurar els permisos).

```
import os
import requests
import numpy as np
import pandas as pd
import json

def create_tf_serving_json(data):
    return {'inputs': {name: data[name].tolist() for name in data.keys()} if isinstance(data, dict) else data.tolist()}

def score_model(dataset):
    url = 'https://adb-2188649429451562.2.azuredatabricks.net/serving-endpoints/iris/invocations'
    headers = {'Authorization': f'Bearer {os.environ.get("DATABRICKS_TOKEN")}', 'Content-Type': 'application/json'}
    ds_dict = {'dataframe_split': dataset.to_dict(orient='split')} if isinstance(dataset, pd.DataFrame) else create_tf_serving_json(dataset)
    data_json = json.dumps(ds_dict, allow_nan=True)
    response = requests.request(method='POST', headers=headers, url=url, data=data_json)
    if response.status_code != 200:
        raise Exception(f'Request failed with status {response.status_code}, {response.text}')
    return response.json()
```

Amb això acabam aquest recorregut per Databricks.