Introducción a Databricks y el ecosistema Spark

Databricks

Data Engineering

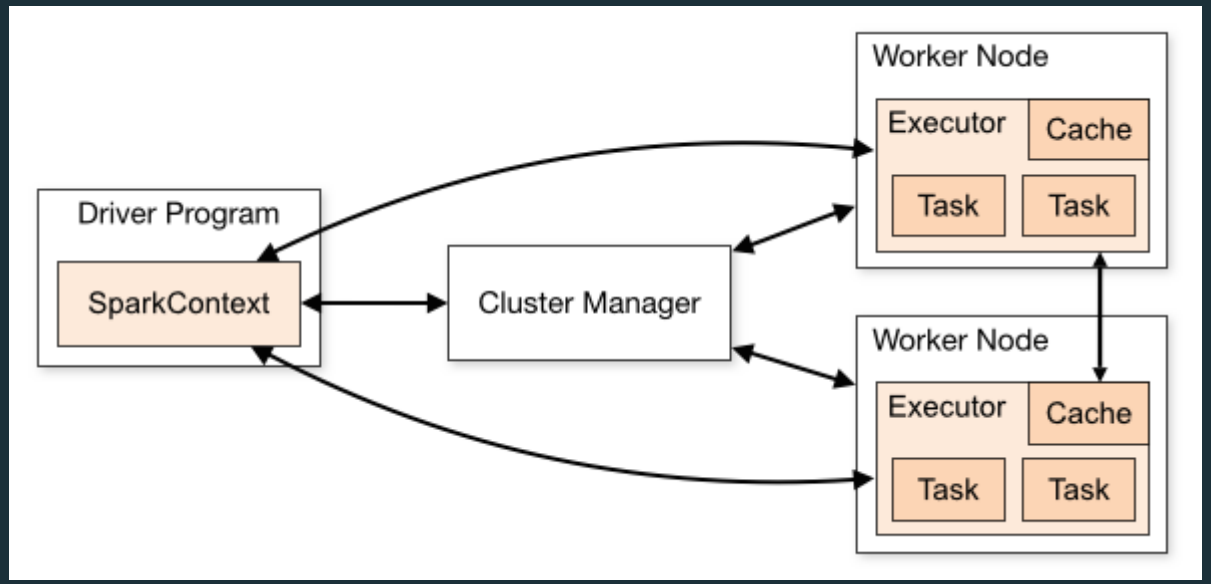
databricks

IMMUNE
TECHNOLOGY INSTITUTE

Spark – Modelo de Ejecución

Apache Spark opera bajo un modelo de ejecución distribuida, permitiendo el procesamiento de grandes volúmenes de datos en paralelo a través de un cluster formado por:

- Driver: organiza las tareas a ejecutar y distribuye el trabajo entre los nodos del cluster.
- Nodo (Worker): Ejecuta una parte del trabajo en paralelo utilizando sus *executors*.

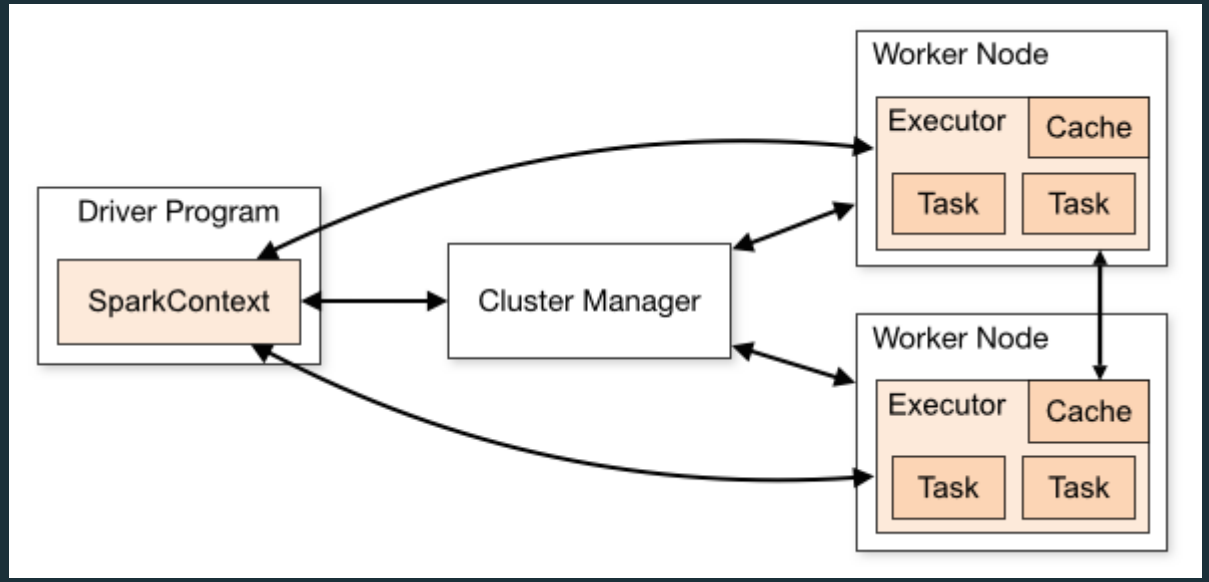


Este enfoque permite a Spark manejar eficientemente datos de gran tamaño, optimizando el uso de recursos y reduciendo tiempos de ejecución.

Spark – Modelo de Ejecución

El modelo de ejecución de Spark se basa en el concepto de RDD (Resilient Distributed Dataset), que es una colección de elementos particionados a lo largo del cluster que pueden ser operados en paralelo.

Spark utiliza un planificador avanzado que divide el trabajo en tareas que se ejecutan en los executors.



databricks

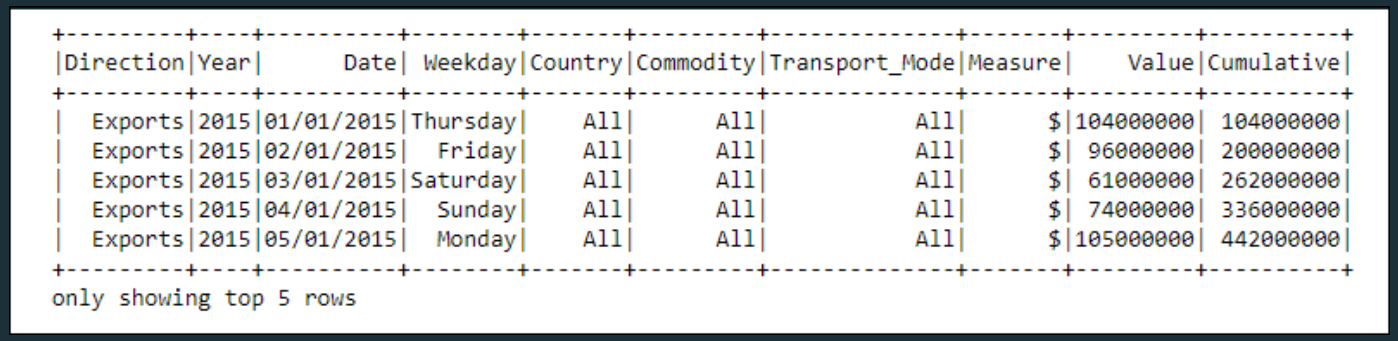
IMMUNE
TECHNOLOGY INSTITUTE

Spark – Dataframe

Un DataFrame es una colección distribuida de datos organizados en columnas, similar a una tabla en una base de datos relacional o un dataframe en pandas de Python.

Ofrece una API de alto nivel que facilita las operaciones de manipulación de datos, como selección, filtrado y agregación, de manera eficiente y con sintaxis sencilla.

Los DataFrames están optimizados internamente por el motor de Spark, lo que permite realizar operaciones complejas de manera rápida y eficaz.



Direction	Year	Date	Weekday	Country	Commodity	Transport_Mode	Measure	Value	Cumulative
Exports	2015	01/01/2015	Thursday	All	All	All	$	104000000	104000000
Exports	2015	02/01/2015	Friday	All	All	All	$	96000000	200000000
Exports	2015	03/01/2015	Saturday	All	All	All	$	61000000	262000000
Exports	2015	04/01/2015	Sunday	All	All	All	$	74000000	336000000
Exports	2015	05/01/2015	Monday	All	All	All	$	105000000	442000000

only showing top 5 rows

databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Dataframe

Los DataFrames en Spark soportan diferentes formatos de datos (JSON, Parquet, CSV, Avro, etc..) y fuentes de datos (bases de datos, sistemas de archivos distribuidos, streaming, etc...).

Direction	Year	Date	Weekday	Country	Commodity	Transport_Mode	Measure	Value	Cumulative
Exports	2015	01/01/2015	Thursday	All	All	All	$	104000000	104000000
Exports	2015	02/01/2015	Friday	All	All	All	$	96000000	200000000
Exports	2015	03/01/2015	Saturday	All	All	All	$	61000000	262000000
Exports	2015	04/01/2015	Sunday	All	All	All	$	74000000	336000000
Exports	2015	05/01/2015	Monday	All	All	All	$	105000000	442000000

only showing top 5 rows

databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Dataframe

Los DataFrames en Spark soportan diferentes formatos de datos (JSON, Parquet, CSV, Avro, etc..) y fuentes de datos (bases de datos, sistemas de archivos distribuidos, streaming, etc...).

Direction	Year	Date	Weekday	Country	Commodity	Transport_Mode	Measure	Value	Cumulative
Exports	2015	01/01/2015	Thursday	All	All	All	$	104000000	104000000
Exports	2015	02/01/2015	Friday	All	All	All	$	96000000	200000000
Exports	2015	03/01/2015	Saturday	All	All	All	$	61000000	262000000
Exports	2015	04/01/2015	Sunday	All	All	All	$	74000000	336000000
Exports	2015	05/01/2015	Monday	All	All	All	$	105000000	442000000

only showing top 5 rows

databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Transformaciones

Las transformaciones en Spark son operaciones que crean un nuevo DataFrame a partir de uno existente, sin modificar los datos originales.

Estas operaciones son "perezosas", es decir, Spark no ejecuta la transformación hasta que se realiza una acción que requiera un resultado.

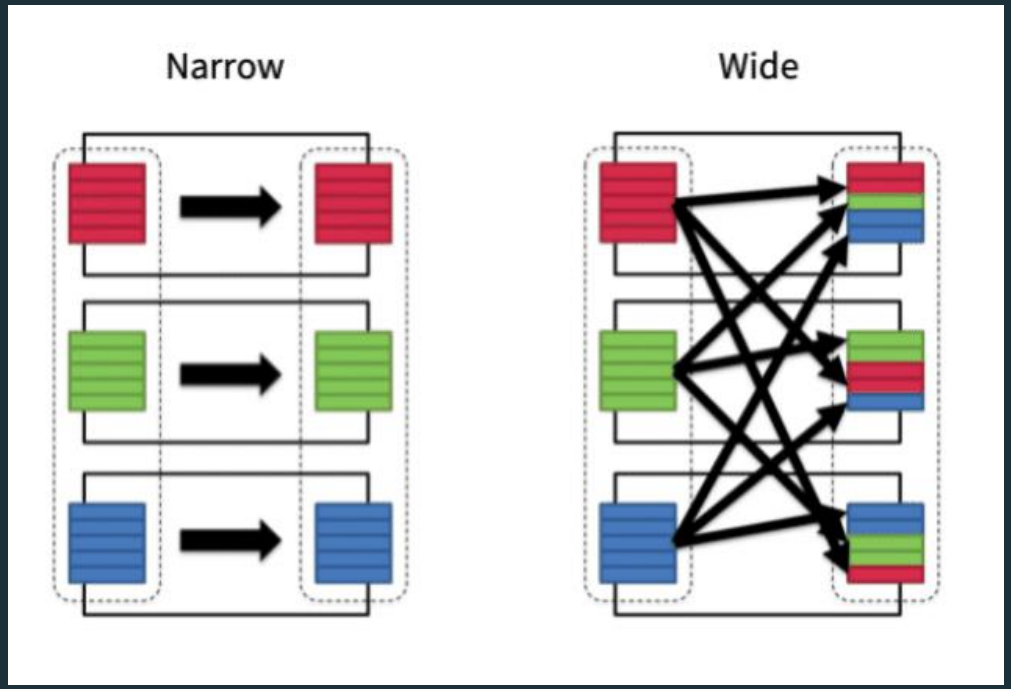
Ejemplos de transformaciones incluyen **map, filter, y join,** entre otras. Las transformaciones permiten construir pipelines de procesamiento de datos más complejos de manera eficiente.

```
filtered_df = df.filter(df["rank"] < 6)
select_df = df.select("City", "State")
subset_df = filtered_df.withColumn(“someColumn“,lit(1)).filter(df["rank"] < 11).select("City")
```

Spark – Transformaciones simples vs complejas

Transformaciones simples (narrow): son aquellas operaciones en las que los datos de entrada están en una sola partición, es decir, no requieren datos de otras particiones (Ej: **filter, select, map**).

Transformaciones complejas (wide): Requieren datos de varias particiones para ser combinados (Ej: **groupBy y join**), lo que generalmente resulta en un **shuffle**, una redistribución costosa de datos entre múltiples nodos



databricks

IMMUNE
TECHNOLOGY INSTITUTE

Spark – Acciones

Las acciones, por otro lado, son operaciones que desencadenan la ejecución de las transformaciones acumuladas en DataFrame y devuelven un resultado al driver o almacenan el resultado en el sistema de archivos.

Ejemplos de acciones son **count, collect, write**, etc...

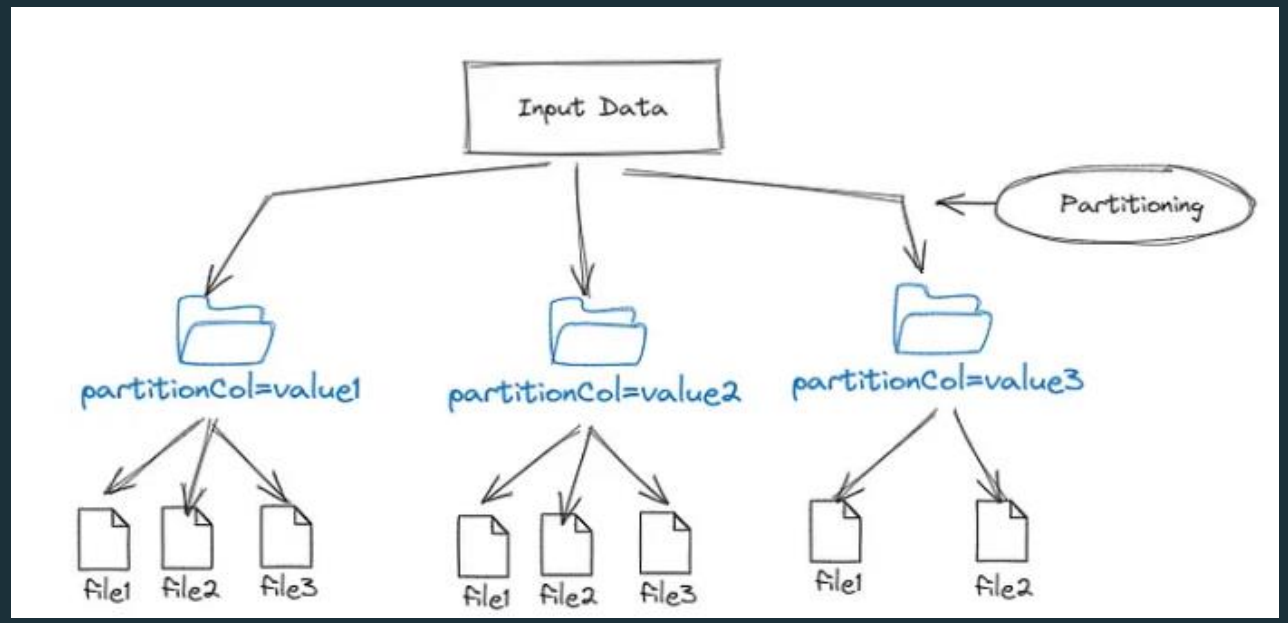
Las acciones son cruciales para obtener resultados de las operaciones de procesamiento de datos y provocan la evaluación de todo el plan de ejecución

```
df.write.format("json").save("/tmp/json_data")
```

Spark – Particionamiento de datos en escritura

El particionado de datos al guardar en Apache Spark es una técnica que implica dividir los datos en bloques o particiones más manejables antes de almacenarlos en un sistema de archivos o base de datos.

Esta estrategia permite organizar los datos de manera que se optimicen las consultas futuras y se mejore el rendimiento de lectura.



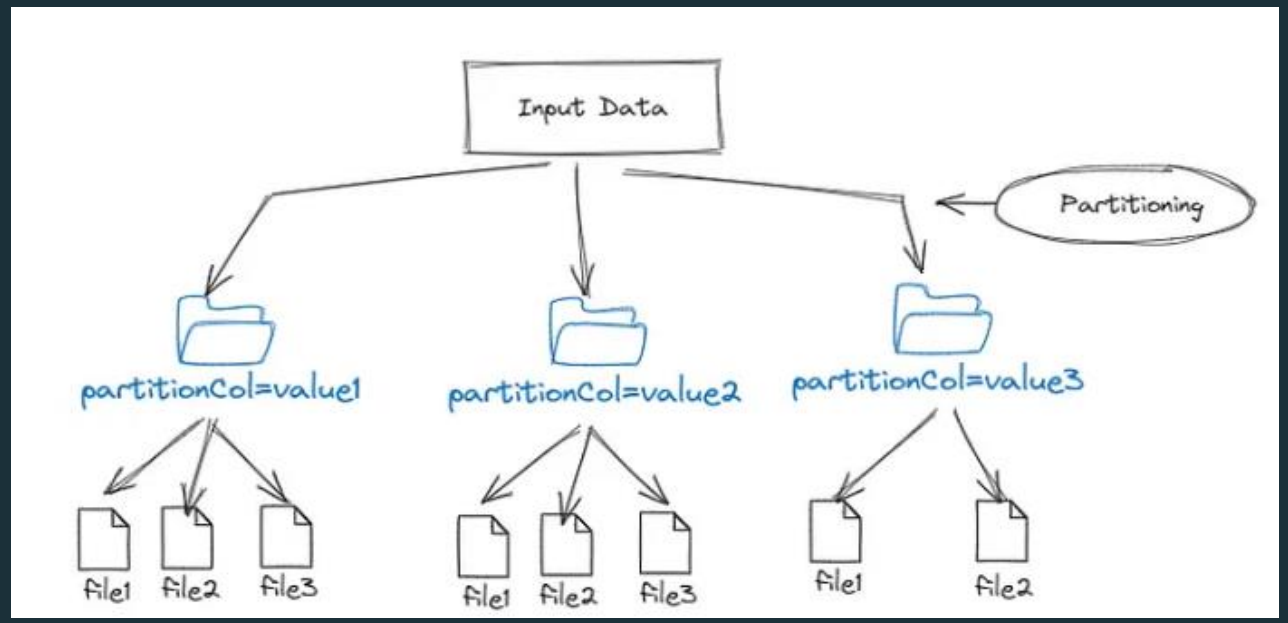
databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Particionamiento de datos en escritura

Esto es importante por las siguientes razones:

1. **Optimización de Consultas**: Almacena datos en particiones para reducir la cantidad de datos leídos durante las consultas, acelerando así el análisis.
2. **Mejora del Rendimiento de Escritura**: Distribuye el proceso de almacenamiento a través de múltiples nodos para un almacenamiento distribuido eficiente.
3. **Escalabilidad**: Facilita el manejo de grandes volúmenes de datos dividiéndolos en unidades gestionables.
4. **Accesibilidad y Mantenimiento**: Mejora el mantenimiento y acceso a los datos permitiendo operaciones más rápidas y específicas.



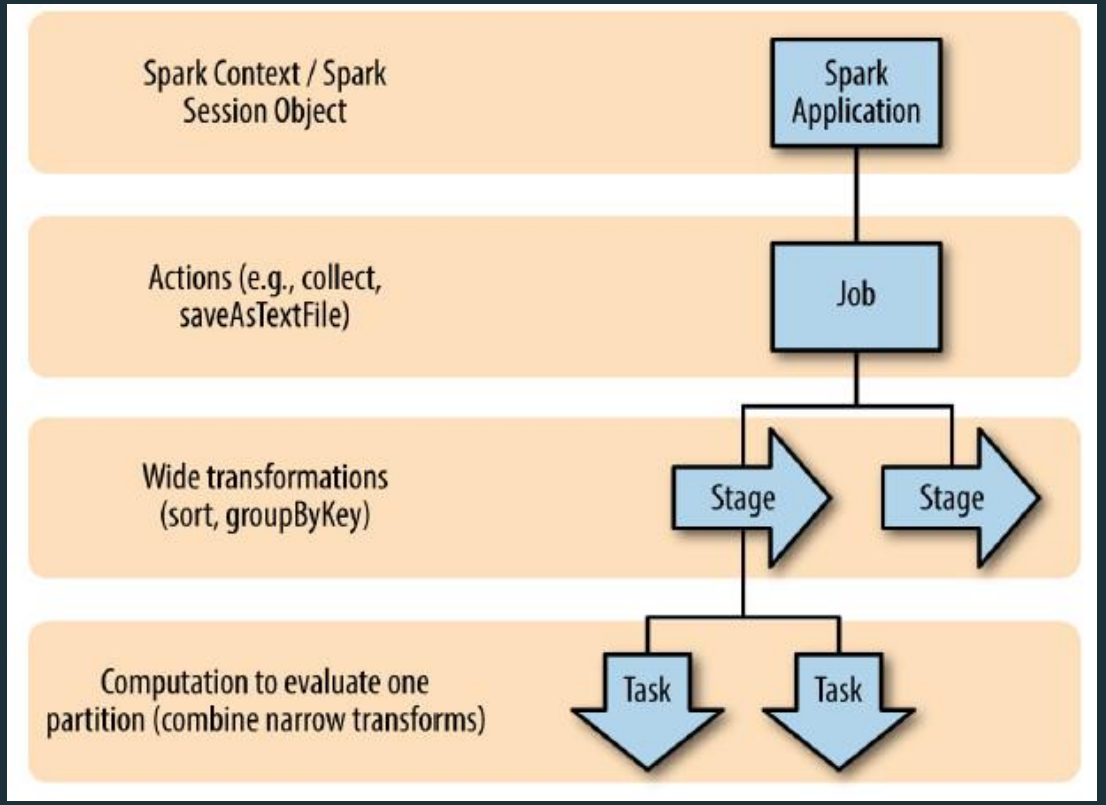
databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Partes de una aplicación

Una aplicación de Spark se compone de 3 varios conceptos clave: **jobs, stages, y tasks**.

- Job: corresponde a una acción ejecutada sobre un DataFrame, y puede constar de múltiples stages.
- Stage: son grupos de tareas que se pueden ejecutar en paralelo, separados por shuffles o por dependencias en los datos.
- Task: unidad de trabajo enviada a un executor, correspondiendo a la ejecución de una operación sobre una partición de datos.



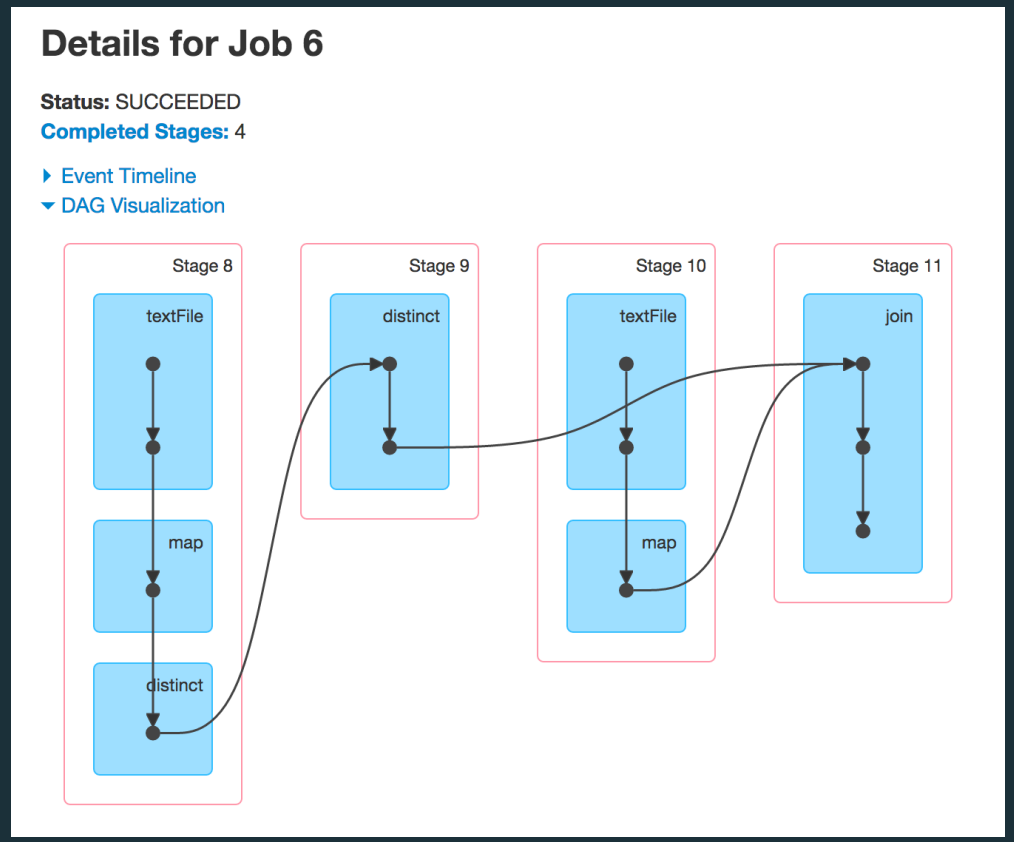
databricks

IMMUNE TECHNOLOGY INSTITUTE

Spark – Plan de ejecución y DAG

Cada vez que se ejecuta una acción, Spark crea un plan de ejecución que optimiza el procesamiento de los datos. Este plan se representa como un **DAG (Directed Acyclic Graph)**, que es un gráfico de las transformaciones a realizar, mostrando cómo los diferentes DataFrames están relacionados entre sí y cómo se transforman a través de las operaciones.

El DAG permite a Spark optimizar el procesamiento, combinando transformaciones y planificando la ejecución de manera que minimice el trabajo necesario y los shuffles. La visualización del DAG en la interfaz de usuario de Spark ofrece una herramienta muy útil para entender y optimizar las aplicaciones Spark



databricks

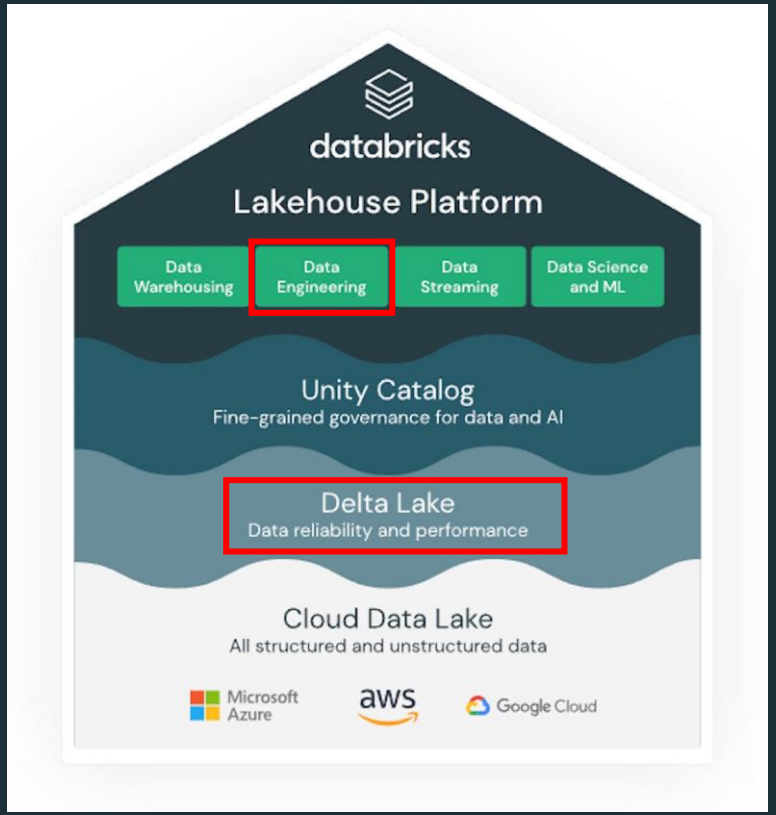
IMMUNE TECHNOLOGY INSTITUTE

Databricks - Lakehouse

Lakehouse es una arquitectura moderna que integra la flexibilidad de los data lakes con las capacidades analíticas de los data warehouses, utilizando Delta Lake para garantizar el rendimiento y fiabilidad, y gestionar metadatos.

Facilita el manejo y análisis de grandes volúmenes de datos estructurados y no estructurados en una plataforma unificada.

Implementa rigurosos controles de gobernanza y seguridad, proporcionando una herramienta única para el gobierno de datos



databricks

IMMUNE TECHNOLOGY INSTITUTE

Delta Lake

Delta Lake es una capa de almacenamiento de datos abierta, diseñada para brindar seguridad, confiabilidad y rendimiento a los sistemas de datos a gran escala.

Funciona sobre los sistemas de archivos existentes en la nube, proporcionando ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad)

databricks

IMMUNE
TECHNOLOGY INSTITUTE

Delta Lake

Delta Lake introduce varios beneficios:

- Transacciones ACID en big data
- Escalabilidad masiva
- Compatibilidad con múltiples lenguajes de programación y frameworks de procesamiento de datos
- Facilidades para la gestión de metadatos y evolución de esquemas
- Permite un manejo eficiente de datos históricos y versionamiento, facilitando la auditoría y rollbacks cuando es necesario

Delta Lake – ACID

- **Atomicidad**: Todas las operaciones de una transacción se completan con éxito o ninguna se aplica.
- **Consistencia**: Las transacciones llevan la base de datos de un estado válido a otro, respetando todas las reglas definidas.
- **Aislamiento**: Las operaciones de una transacción no son visibles para otras transacciones hasta que se completa.
- **Durabilidad**: Una vez finalizada, los resultados de una transacción permanecen y no se pierden, incluso ante fallos.

Delta Lake – Compactado de archivos

Delta Lake mejora significativamente el rendimiento mediante el **compactado de archivos**, lo que reduce la sobrecarga de lectura y mejora la eficiencia de almacenamiento al combinar archivos pequeños en otros más grandes.

La indexación de datos es otra optimización clave, permitiendo accesos más rápidos a los datos específicos mediante la **creación de índices** que aceleran las consultas.

Delta Lake – Versionado y Time Travel

El versionado de datos es una característica clave de Delta Lake, ofreciendo la capacidad de mantener múltiples versiones de los mismos datos. Esto permite realizar auditorías, experimentación y rollbacks de manera eficiente, asegurando que los datos puedan ser restaurados a estados anteriores sin pérdida de información.

La funcionalidad de **Time Travel** de Delta Lake permite a los usuarios consultar versiones anteriores de los datos, facilitando el análisis histórico y la comparación de datos a lo largo del tiempo. Esta característica es crítica para la resolución de problemas de datos y la reproducción de análisis o informes basados en datos históricos.

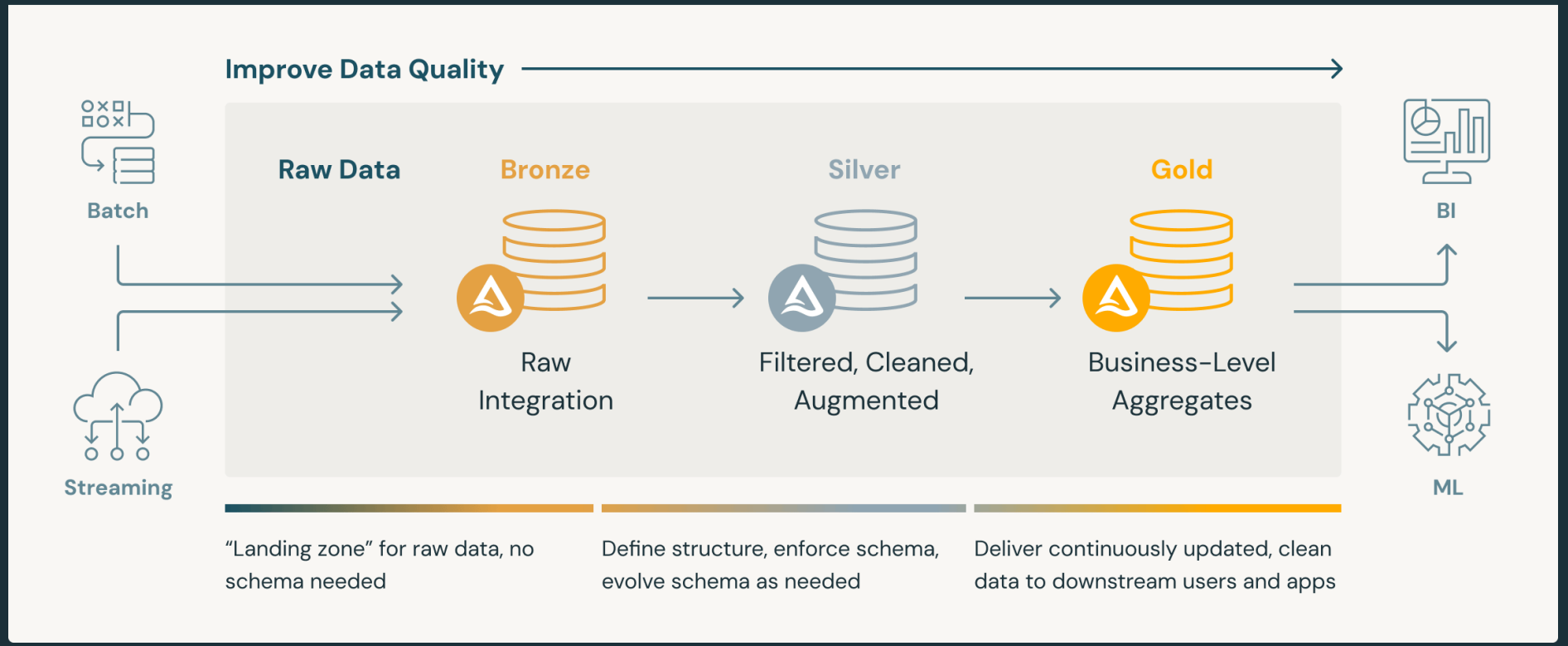
Arquitectura Datos de tipo Medallion

La arquitectura de datos de tipo medallón organiza la información en tres capas clave para optimizar el acceso, análisis y seguridad de los datos:

- Capa **Bronze**: Almacena datos crudos en formato tabular procedentes de la información original, sirviendo como la base para la limpieza y transformación. También es recomendable guardar los datos en el formato original. Esto fundamental para la trazabilidad y el cumplimiento.
- Capa **Silver**: Contiene datos limpios, enriquecidos y transformados, preparados para análisis más detallados. Actúa como un puente entre el almacenamiento crudo y los datos de alto nivel.
- Capa **Gold**: Aloja datos agregados o resumidos, listos para la toma de decisiones y análisis predictivos. Representa la vista más refinada y valiosa de los datos para los stakeholders y gente de negocio.

Arquitectura Datos de tipo Medallion

Building reliable, performant data pipelines with DELTA LAKE



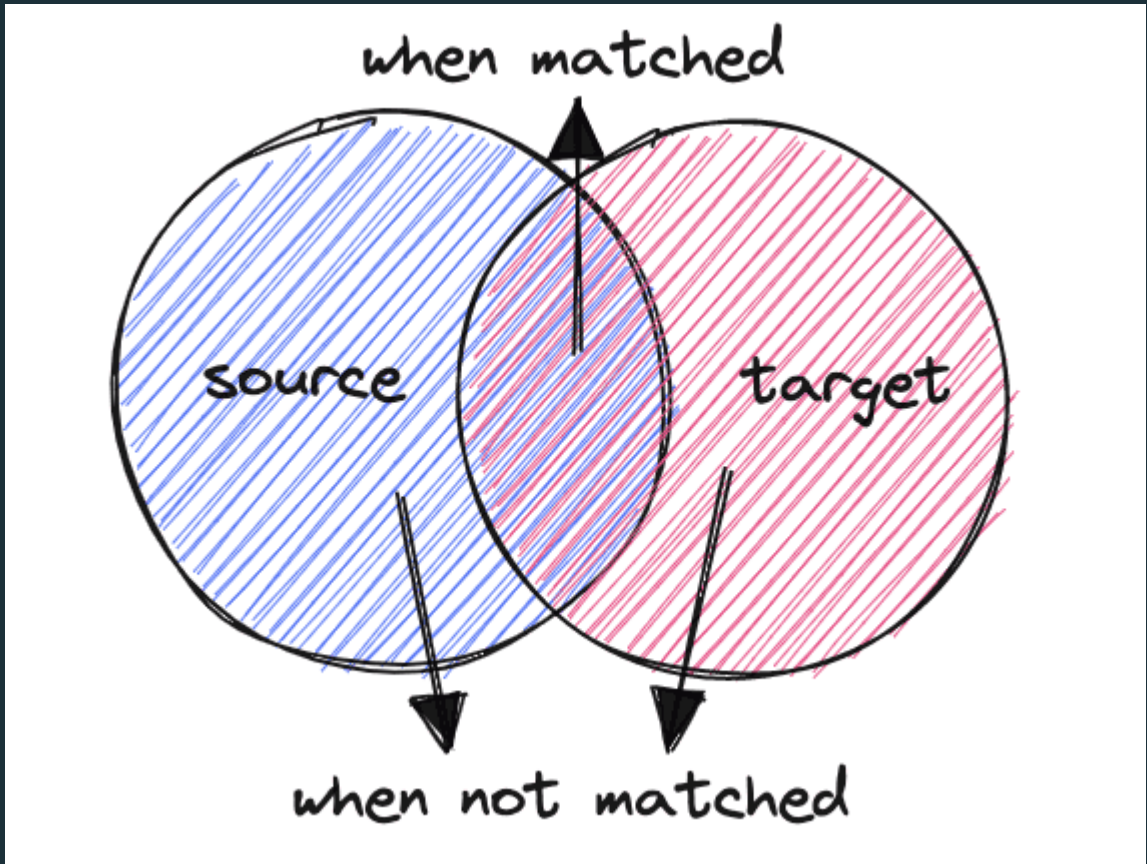
databricks

IMMUNE TECHNOLOGY INSTITUTE

Delta Merge

Permite combinar dos conjuntos de datos: uno existente (la tabla objetivo) y otro nuevo o modificado (la tabla fuente), basándose en una condición de coincidencia especificada.

Si se cumplen las condiciones de coincidencia, se pueden actualizar los registros existentes en la tabla objetivo o insertar nuevos registros si no hay coincidencias (UPSERT).



databricks

IMMUNE
TECHNOLOGY INSTITUTE

Delta Merge

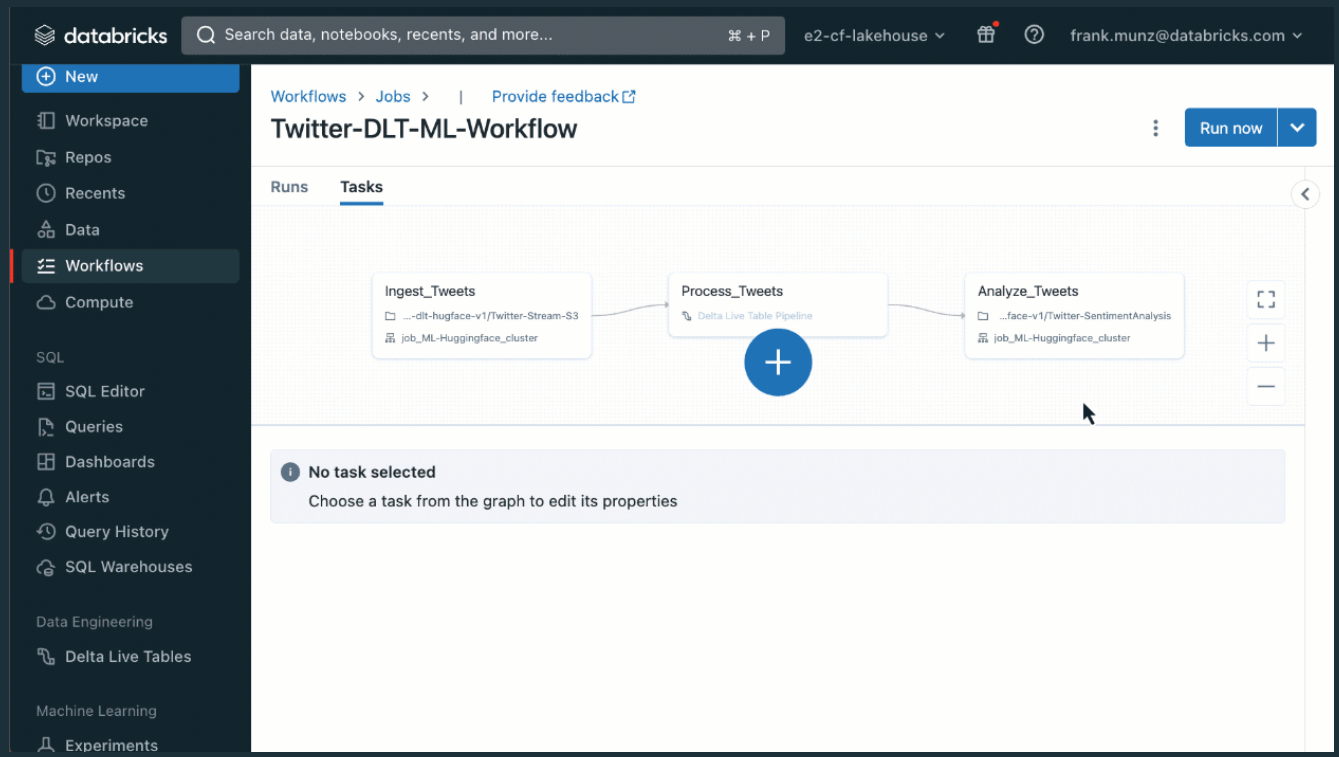
Usar Delta Merge tiene las siguientes ventajas:

- **Eficiencia en el Manejo de Datos**: Reduce la necesidad de operaciones de reescritura completas de los datos, lo que puede ser muy costoso en términos de tiempo y recursos de computación. En su lugar, permite modificar solo las partes necesarias de los datos.
- **Simplificación de ETL**: Simplifica los procesos de ETL (Extract, Transform, Load) al permitir la actualización incremental de los datos, lo que hace que los flujos de trabajo de procesamiento de datos sean más eficientes y fáciles de mantener.
- **Control de Versiones y Rollback**: Delta Lake mantiene un historial de las transacciones, lo que permite el control de versiones de los datos y la capacidad de revertir a estados anteriores si es necesario. Esto es útil para la auditoría de datos y para corregir errores de manera eficaz.
- **Optimización de Consultas**: Mejora el rendimiento de las consultas al permitir optimizaciones como el pruning de archivos (ignorar archivos que no son relevantes para una consulta) y la compresión de datos, lo que resulta en tiempos de respuesta más rápidos.

Orquestación de Procesos con Databricks Workflows

Databricks Workflows ofrece una solución de orquestación de procesos poderosa y flexible, permitiendo a los usuarios diseñar, programar y monitorizar pipelines de datos complejos dentro del ecosistema de Databricks.

Esta herramienta facilita la automatización de flujos de trabajo de datos, desde la ingesta hasta el análisis, asegurando eficiencia y coherencia en los procesos de ingeniería de datos.



databricks

IMMUNE TECHNOLOGY INSTITUTE

Diseño y Programación de Workflows

Con Databricks Workflows, los usuarios pueden diseñar workflows con una interfaz visual intuitiva o mediante código, y programar su ejecución para que se realice de manera automática según un horario.

Esto permite una gestión eficiente de los recursos y asegura que los datos estén siempre actualizados y listos para el análisis

databricks

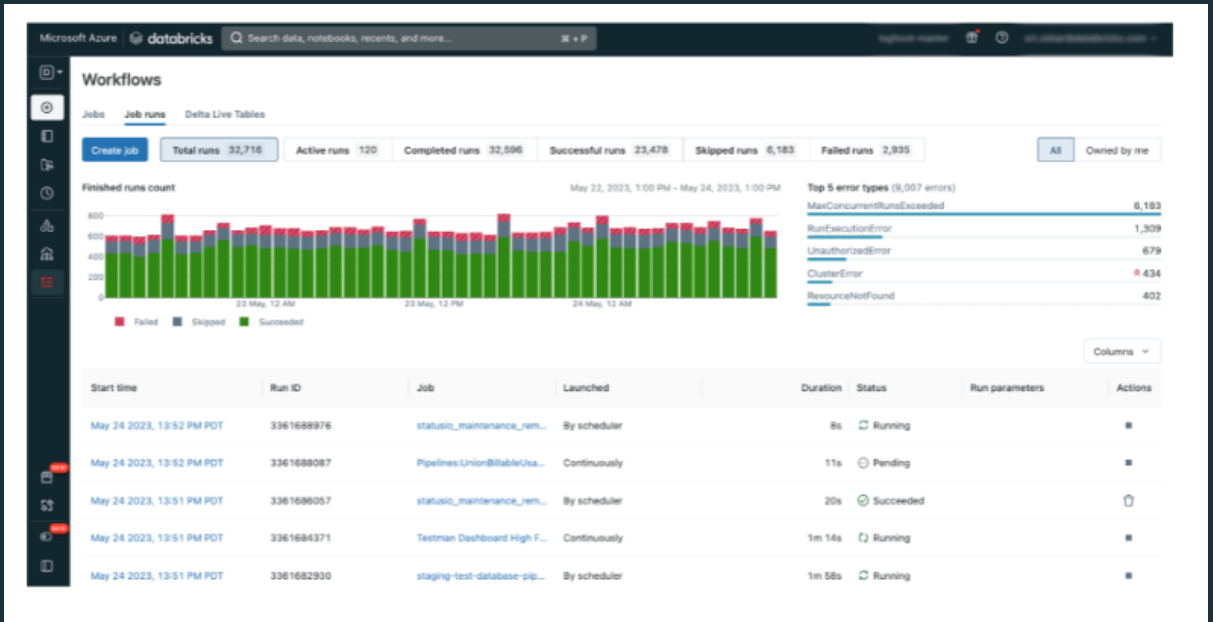
IMMUNE TECHNOLOGY INSTITUTE

Monitorizacion y Optimización de Workflows

La capacidad de monitorear y optimizar workflows en tiempo real es una ventaja clave de Databricks Workflows.

Los usuarios pueden rastrear el rendimiento, identificar cuellos de botella y ajustar los recursos asignados para maximizar la eficiencia operativa.

Esto asegura que los pipelines de datos sean escalables, confiables y optimizados para el rendimiento.



databricks

IMMUNE TECHNOLOGY INSTITUTE

Databricks CLI: Interacción y Automatización desde la Línea de Comandos

Databricks CLI es una interfaz de línea de comandos que ofrece a los usuarios una forma potente y flexible de interactuar con la plataforma Databricks.

Proporcionando acceso a una amplia gama de funcionalidades, como la gestión de workspaces, clusters, workflows y notebooks, Databricks CLI es ideal para la automatización de tareas y la integración con sistemas de CI/CD.

Su compatibilidad con scripts permite a los desarrolladores y ingenieros de datos orquestar complejas cadenas de trabajo y administrar recursos de Databricks de manera programática, optimizando los flujos de trabajo y aumentando la eficiencia operativa.

```
{
  "cluster_name": "my-cluster",
  "spark_version": "7.3.x-scala2.12",
  "node_type_id": "Standard_DS3_v2",
  "spark_conf": {},
  "aws_attributes": {
    "zone_id": "us-west-2a",
    "first_on_demand": 1,
    "availability": "SPOT_WITH_FALLBACK",
    "spot_bid_price_percent": 100
  },
  "num_workers": 2
}
```

```
databricks clusters create --json-file cluster-config.json
```

databricks

IMMUNE TECHNOLOGY INSTITUTE

databricks

Lab 2: Ingesta de datos en Delta Lake

IMMUNE
TECHNOLOGY INSTITUTE