Section 01- Databricks lake house platform

Wednesday, October 11, 2023 3:58 PM

Outline

What is Databricks?
Datawarehouse vs data lake
Data lakehouse, intro, architecture, main components and high level component
Delta lake
Cluster and hands on cluster Creation.

What is Databricks?
Multi cloud lake house platform based on the architecture of Apache spark.

Data warehouse vs Data lake

Topic	Data warehouse	data lake
Definition	It is an structured data management system, which stores your current and historical data from multiple sources.	Used to store any format of data structured, semi structured and unstructured.
Use cases	BI and SQL Analytics	It serve as centralize repo of unstructured data.
Pros	Maintained, reliability, consistency, data quality and accuracy	• In expensive, Provide quick and seamless integration, Consolidate, prepare transform
Cons	Not support for unstructured data, data science and ML, expensive to scale	Reliability issues, Slow performance and lack of security features.

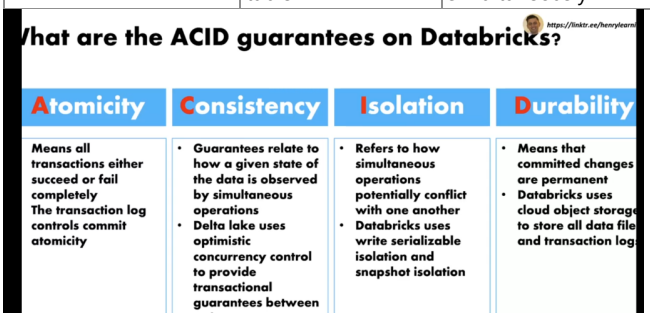
What is lake house?

- One platform which unify all of your data engineering analytics, AI and data science workloads, It delivers open, flexible and ML support of data lake, and adapt reliability, strong governance, and performance of data warehouse.
- Combines ACID transactions and data governance of enterprise data warehouses with the flexibility, cost efficiency of data lake to enable BI and AI workloads.

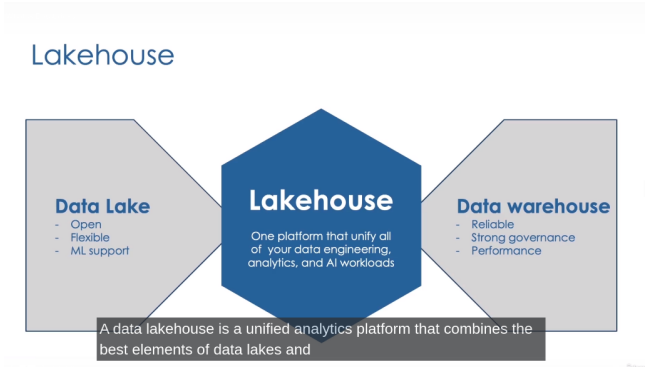
What are ACID guaranties on data bricks?
Acid are the four key properties which defines the transaction, if database operations has ACID properties, it will be called as ACID transaction , data storage systems. The system which apply ACID operations and properties are called as transactional systems.

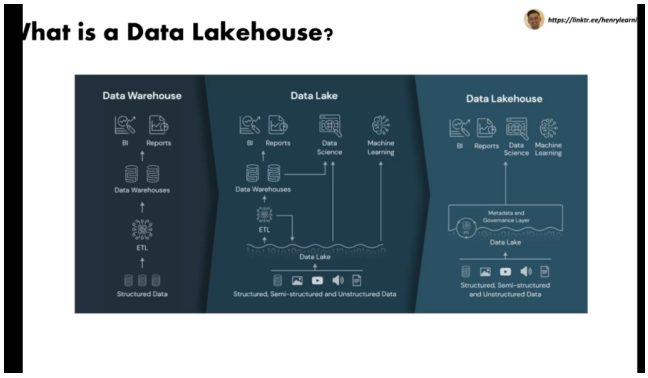
It guarantees that each read, write, and update operation has the following properties.

Atomicity	Consistency	Isolation	Durability
Each statement in a transaction (to read, write, update or delete data) is treated as a single unit. Either the entire statement is executed, or none of it is executed. This property prevents data loss and corruption from occurring if, for example, if your streaming data source fails mid-stream.	ensures that transactions only make changes to tables in predefined, predictable ways. Transactional consistency ensures that corruption or errors in your data do not create unintended consequences for the integrity of your table	when multiple users are reading and writing from the same table all at once, isolation of their transactions ensures that the concurrent transactions don't interfere with or affect one another. Each request can occur as though they were occurring one by one, even though they're actually occurring simultaneously	ensures that changes to your data made by successfully executed transactions will be saved, even in the event of system failure

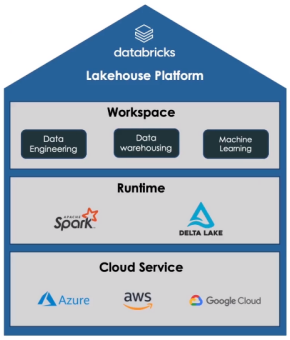


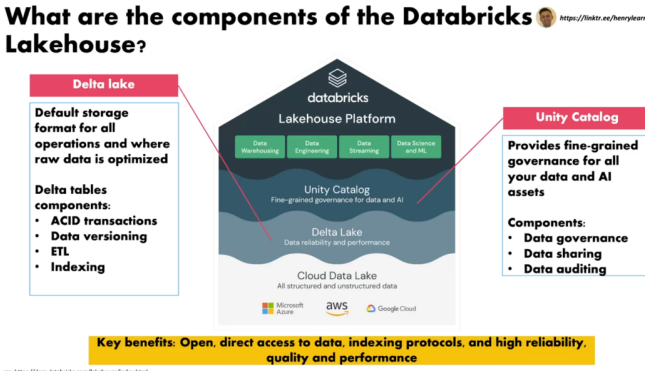
Why are ACID transactions a good thing to have?
ACID transactions ensure the highest possible data reliability and integrity. They ensure that your data never falls into an inconsistent state because of an operation that only partially completes. For example, without ACID transactions, if you were writing some data to a database table, but the power went out unexpectedly, it's possible that only some of your data would have been saved, while some of it would not. Now your database is in an inconsistent state that is very difficult and time-consuming to recover from.





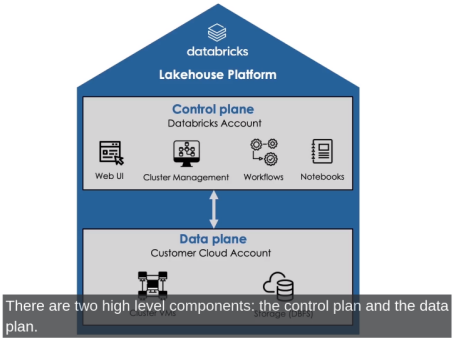
Data lake house architecture





There are two main high level components of data bricks

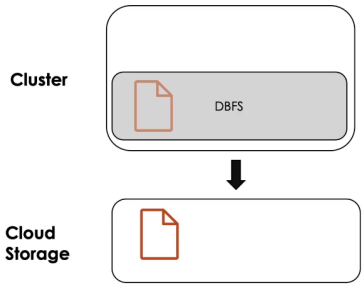
Control Plane	Data Plane
When you create a workspace, It is deployed in the control plane along with databricks services like UI, cluster management, workflow services, and notebooks.	A storage account is deployed in your respective cloud subscription, e.g Azure data will be stored in stoarge account (E.G dbfs) and VMs will be deployed on your azure cloud based on the clusters and it's type you create.
Handle by data bricks	Handled in your cloud subscription.



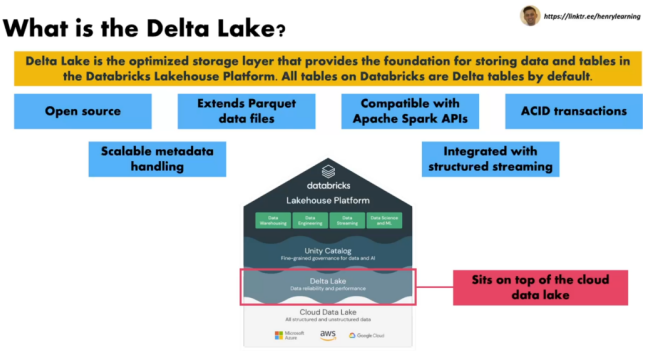
There are two high level components: the control plan and the data plan.

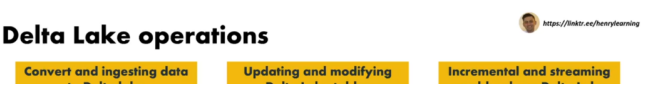
Spark on data bricks
Databricks has been founded by the same engineers who build the Apache spark, the data is distributed and processed in the memory of multiple nodes in cluster.

- Support all languages similar to spark include java, python, sql, R and scala
- Batch and stream processing
- Any data format.
- Support distributed file system called as DBFS (data bricks file system)
- When a cluster is created in data bricks, it comes with pre-installed dbfs.
- **Abstraction layer:** It uses underlying storage to persist the data, even when cluster is terminated it saves your data in the cloud storage.



I hope this introduction was clear and informative for you.





2. Delta lake.mp4

6.1 Section 2_S1

Hands On Creating a cluster
A cluster is set of node or computers working together like a single entity, It contains the master node called the driver node and some worker nodes.

Driver nodes
It is responsible for coordinating the workers and their parallel execution of tasks.
Single node cluster
It has no workers run spark jobs on the driver node.

Databricks runtime
It is an VM image that comes with pre installed libraries, which has specific versions of spark scala, python and other libraries.

Activated photon
It is vectorized query engine developed in C++ in order to enhance the spark performance.

Worker type
These are the different VM sizes provided by your respective cloud, you can select as per your requirement and cloud subscription.

Summary of properties in side defines
No of workers with sized and configs
No of selected driver nodes with size and configs
Runtime type
Worker type selected
Number of DBU/h (databricks unit per hour) pay according to the number of dbu units.

After cluster creation
You can see event log: which shows information about cluster creation, running, and driver health, cluster activity etc.

Driver logs: It shows log information within the clusters, notebook and libraries.

Magic Commands hands on
Language Commands
%sql %python etc.
%md markdown command for formatting
%run another notebook from current notebook, you can call function and variable from the notebook, provides code reusability and modularity
%fs to interact with file system

Another way to interact with files system is dibutils, you can interact with different databricks services and tools like credentils, file systems using dbutils.