Introduction to Databricks

Martin Oharek

October 4th, 2023

Outline

Outline

1. **History context**
   - Challenges of traditional/legacy solutions
   - What managed platforms (e.g. Databricks) bring to the table and why the companies want it
2. **Databricks intro**
   - Lakehouse concept
   - Databricks architecture
   - Building blocks (cloud integration, Apache Spark, MLFlow, Delta Lake,…)
   - Use cases
3. **Core Databricks features**
   - Compute x Storage, Delta Lake
   - Clusters
   - Databricks workflows
   - Data object types
4. **Real world Databricks use cases**

History context

History context

PROFINIT

Big Data Phase 1 – Structured Content	Big Data Phase 2 – Web Based Unstructured Content	Big Data Phase 3 – Mobile and Sensor Based Content
Period: 1970-2000	Period: 2000 – 2010	Period: 2010 - Present
• RDBMS & data warehousing • Extract Transfer Load • Online Analytical Processing • Dashboards & scorecards • Data mining & statistical analysis	• Information retrieval and extraction • Opinion mining • Question answering • Web analytics and web intelligence • Social media analytics • Social network analysis • Spatial-temporal analysis	• Location-aware analysis • Person-centred analysis • Context-relevant analysis • Mobile visualization • Human-Computer-Interaction

History context

- Structured data (RDBMS - ~1970) -> (semi/un)structured data (Data lake)
- Data volumes increased rapidly (internet (~1991), mobile/sensor devices and applications (~2010),…)
- Traditional data processing and storage approaches couldn‘t handle so much data
- Hadoop ecosystem (2006 initial release) – distributed parallel computing and storage
- Managed (cloud) services (AWS EMR, Kinesis, Azure Data Factory, Azure Synapse,…)
- Cloud-based data platforms and unified (scalable) environments (Databricks, Snowflake,…)

Legacy technology challenges

PROFINIT

- **Infrastructure challenges**
  - On-premise infrastructure management
  - No scaling (not easily achievable and expensive)
  - Over-provisioning to keep-up with increasing compute and storage demands (expensive)
  - Security/availability/reliability/back ups, etc. of data centers must be handled by the company -> a lot of people involved
  - Software version updates and patching
  - Keynote: **A LOT OF TASKS THAT DON‘T PRODUCE ANY VALUE FOR THE CUSTOMER‘S BUSINESS**, but must be done

- **Operational challenges**
  - A lot of different data-related tasks (ingestion, ETL, analysis, dashboarding, data science) handled by different pieces of technology – usually leads to complex architectures
  - Data silos – problems with access to data throughout the organization (among different teams,..), often leads to data redundancy and duplication, „many sources of truth“
  - Complexity
  - Bad performance


I have one problem
I have two problems
I have a lot of problems

Databricks intro

How do Databricks solve the challenges?

PROFINIT

- Cloud-based (AWS, Azure, GCP)
  - Infrastructure is managed by the cloud vendor, you just need to provision it
- Auto-scaling support (alleviate the over-provisioning issue)
- Provide tools for handling all data-related processing demands (batch, streaming, ML, data sharing,…), all unified under single platform
- Software versions, libraries and runtimes are managed by Databricks, also come with handy libraries preinstalled
- On-demand cluster provisioning -> no need to run machines when idle
- Lakehouse concept + centralized data governance solution – supports the „single source of truth“

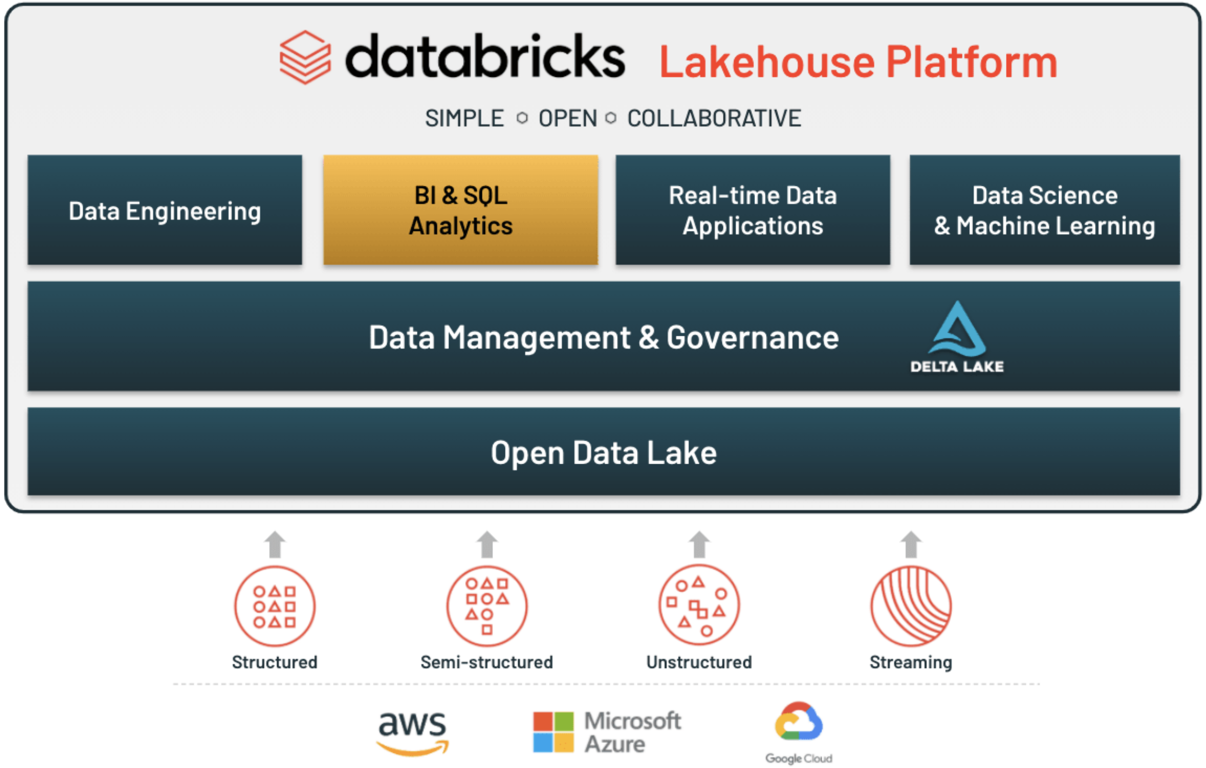
Databricks

PROFINIT

- Founded in 2013
- Unified, data analytics platform for building, deploying, sharing, and maintaining enterprise-grade data, analytics, and AI solutions at scale
- Integrated with cloud vendors – AWS, Azure, GCP
- Cloud agnostic
- Databricks Lakehouse platform
- ~ 15% of market share in big-data-analytics domain (https://6sense.com/tech/big-data-analytics/databricks-market-share)
- Databricks account -> Databricks workspaces associated with the account

Databricks

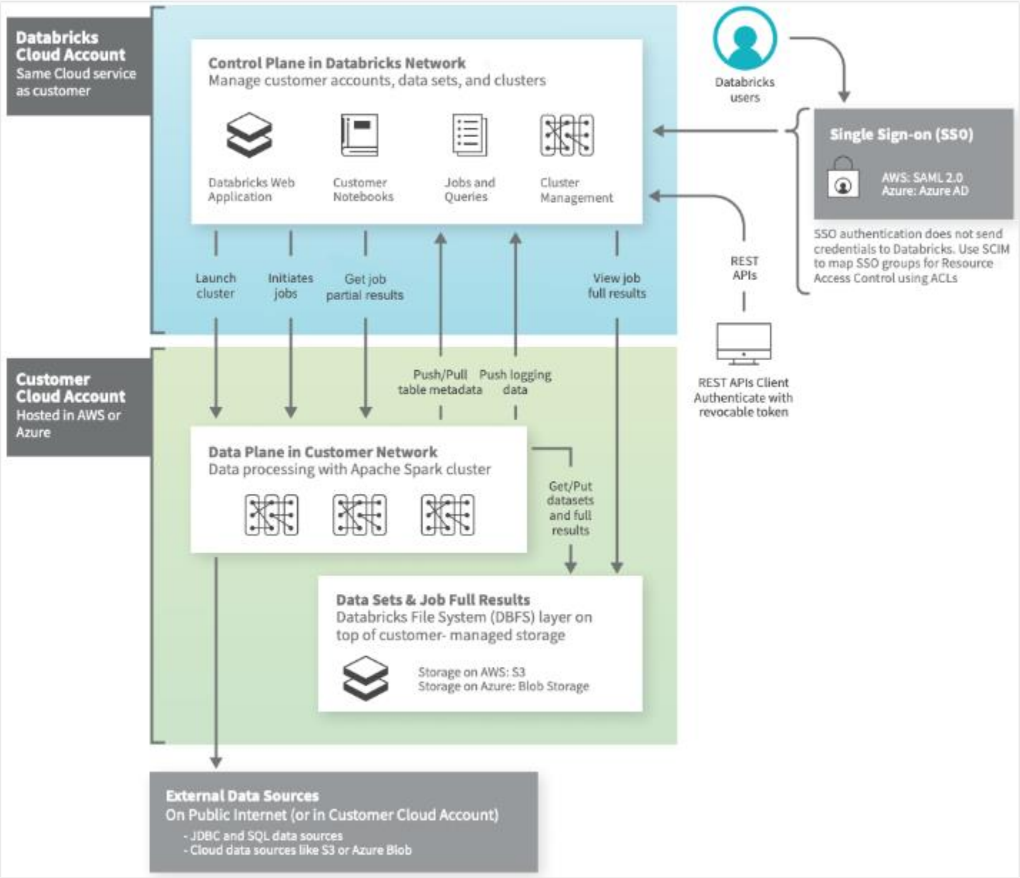
PROFINIT



Databricks spaces

- **Databricks SQL**
  - Compute resources for SQL queries, visualizations and dashboards executed against data sources in the lakehouse
  - SQL warehouse, optimized for processing large-scale data, multi-tenancy
  - Alerting
- **Data Science & Engineering**
  - Notebooks, Apache Spark, Spark Structured Streaming
  - Databricks Jobs
  - ETL – Delta Live Tables
- **Machine learning**
  - AutoML, MLFlow
  - Scalable machine learning - Spark MLLib, HyperOpt, EDA with Spark

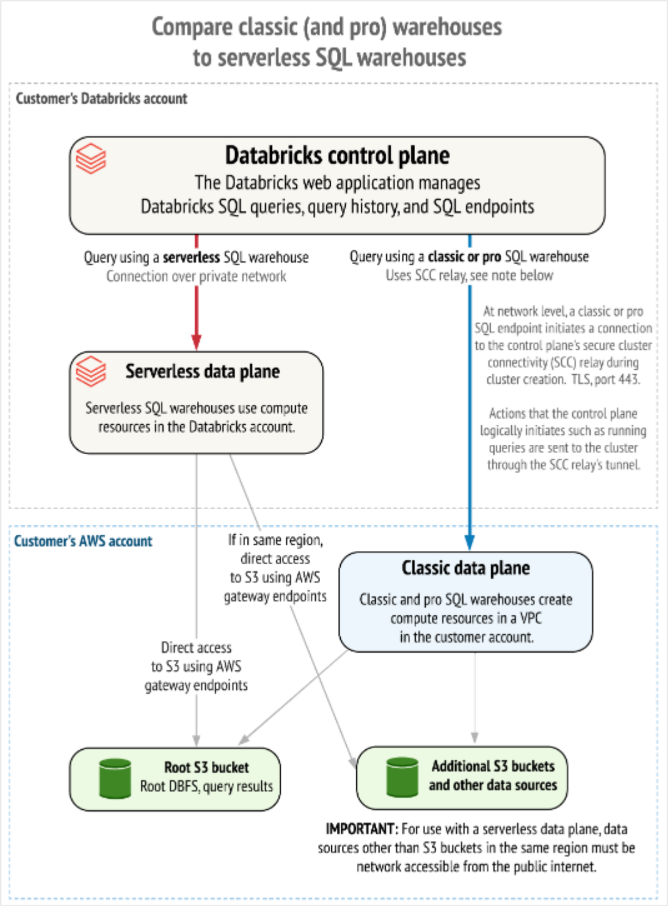
Databricks architecture



Databricks serverless architecture

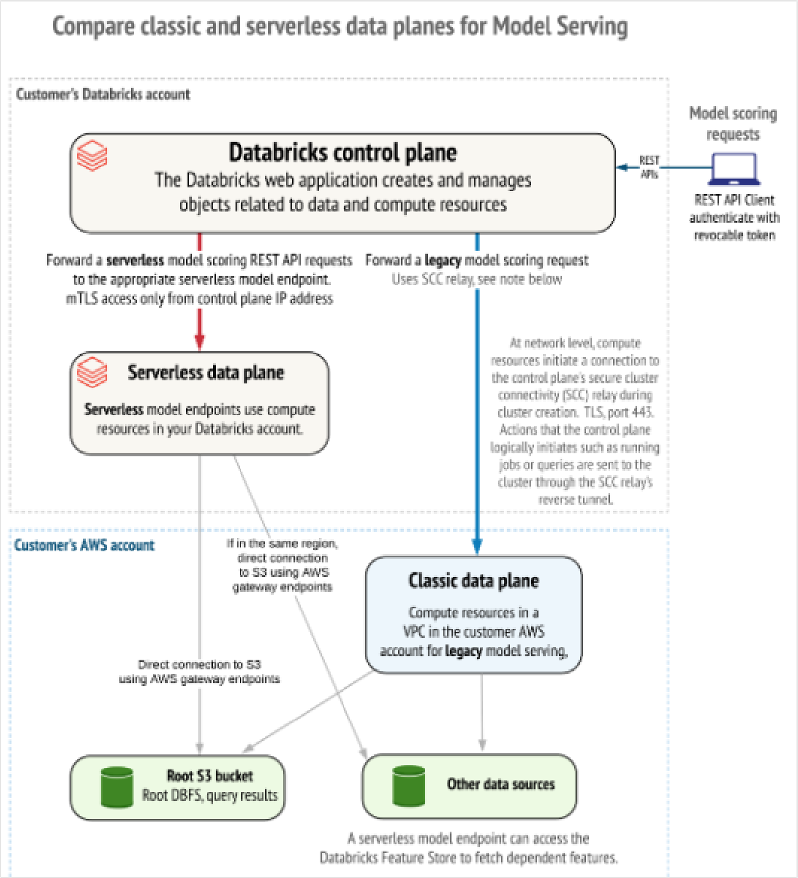
PROFINIT

- Databricks SQL serverless



Databricks serverless architecture

- Model serving

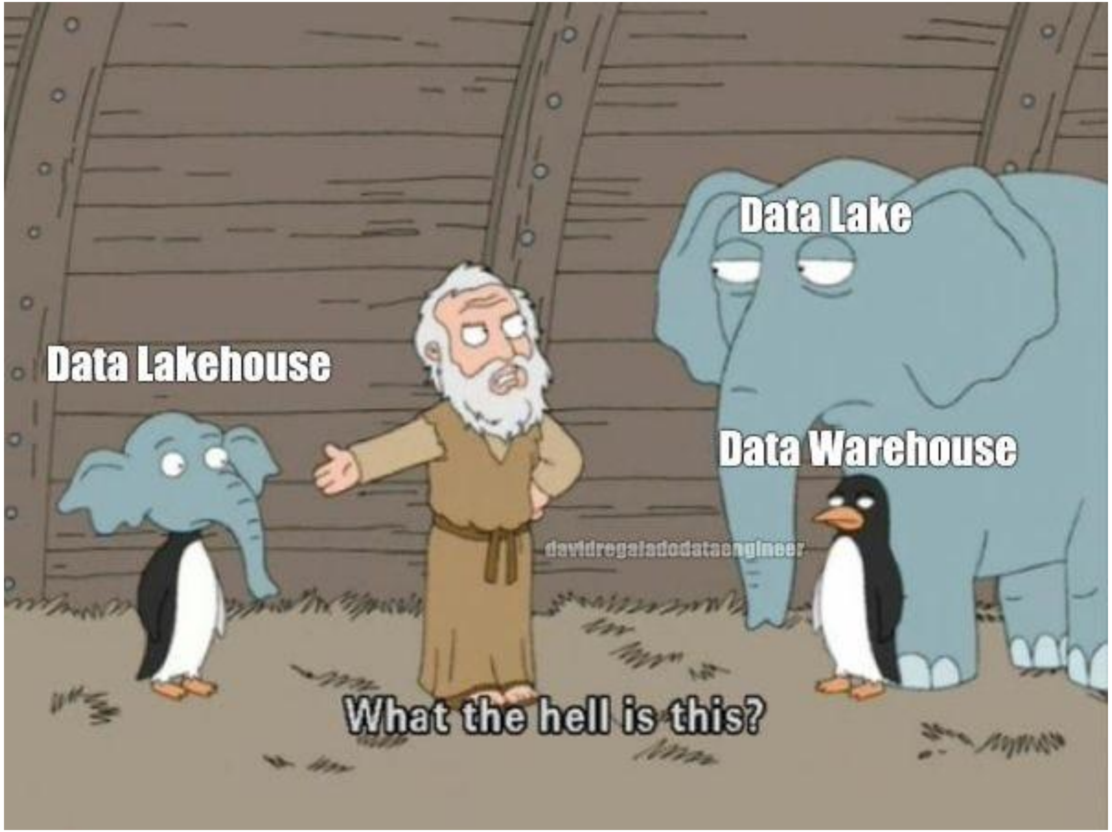


Databricks architecture

PROFINIT

- **Control plane**
  - Backend services managed by Databricks (in its own account)
  - Notebook commands, workspace configurations, etc.
- **Data plane**
  - Where data is processed (customer‘s AWS account)
  - *Classic data plane*
    - Data is stored in your cloud account
    - Notebooks, jobs, pro/classic Databricks SQL warehouses
  - *Serverless data plane*
    - Shared
    - For serverless compute of Databricks SQL or Model serving

Databricks Lakehouse



Databricks Lakehouse

PROFINIT

- https://docs.databricks.com/en/lakehouse/index.html
- Combines best elements from
  - Data warehouses
    - ACID transactions, data governance
  - Data lakes
    - Flexibility, cost-efficiency
- Built on top of open source technologies – Parquet, Apache Spark, Delta Lake, MLFlow – prevents vendor-lock
- **Delta tables** (stored with Delta Lake protocol)
  - ACID, Data versioning, ETL, indexing
- **Unity Catalog**
  - Data governance, Data sharing, Data auditing, Data lineage

Databricks core features

Databricks core features

PROFINIT

- **Decoupled compute from storage**
  - Storage provided by cloud object storage (e.g. AWS S3) or external locations
  - Compute provided by compute clusters
    - Clusters also have their own disk attached
- **Storage layer powered by Delta Lake**
  - Data versioning, historization
  - Indexing, optimization
  - ACID transactions
  - Optimized for structured streaming
- **Databricks workflows (jobs)**
  - Running non-interactive workloads
  - On schedule, on demand
  - Notifications

Databricks clusters

PROFINIT

- Computation resources for data engineering, data science and analytics workloads
- Created on classic data plane = your AWS account
- Running Spark
- **All-purpose clusters**
  - For interactive workloads, usually used with notebooks
  - Can be shared accross multiple users
- **Job clusters**
  - For non-interactive workloads, automated jobs
  - Is terminated when job is finished
- Controlled with UI, CLI, or REST API
- Pools
  - Keep warm instances as idle to reduce start and scale-up times

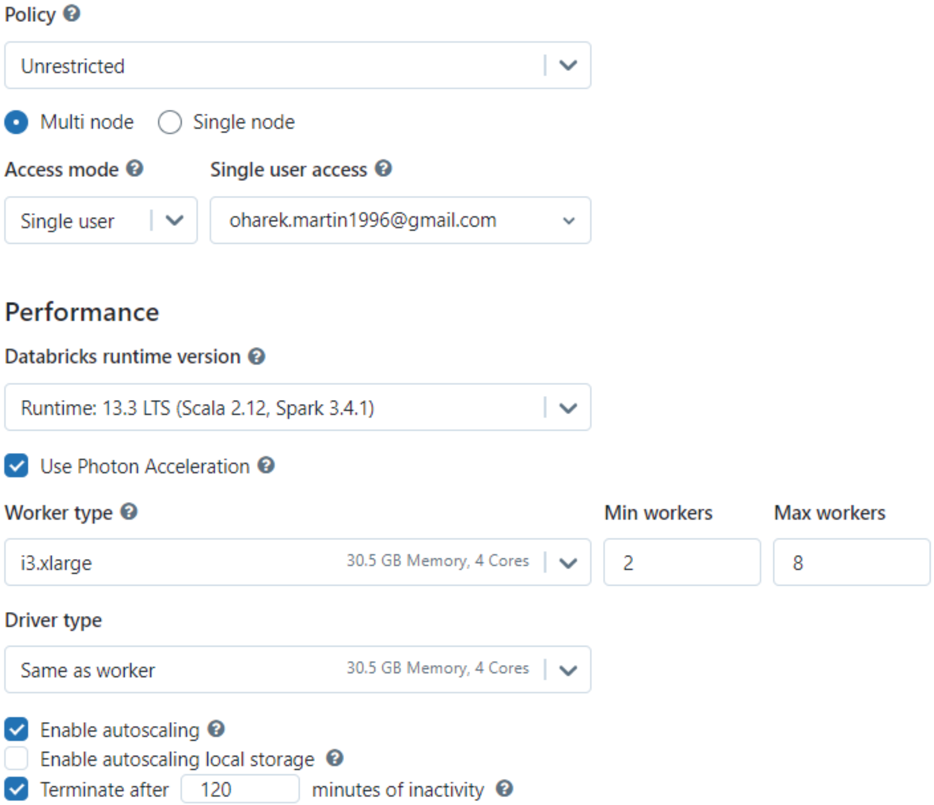
Databricks clusters

PROFINIT

- 1 driver node, 0-n worker nodes
- Autoscaling
  - Add or remove instances from the cluster based on the workload
- Init script for custom initializations
- Arbitrary Spark configurations
- Policy, access mode
- Databricks runtime
  - Scala, Spark preinstalled
- Autotermination
- Tags (Metadata)
- Arbitrary log destinations

Databricks clusters

PROFINIT



Delta Lake

PROFINIT

- Default data storage format
- Data stored as Parquet files
- ACID transactions
  - Secured by transaction log, tracks all changes made to the table
- Data are versioned
  - Keep data files for every version (w.r. to retention period)
  - Time travel

Databricks workflows

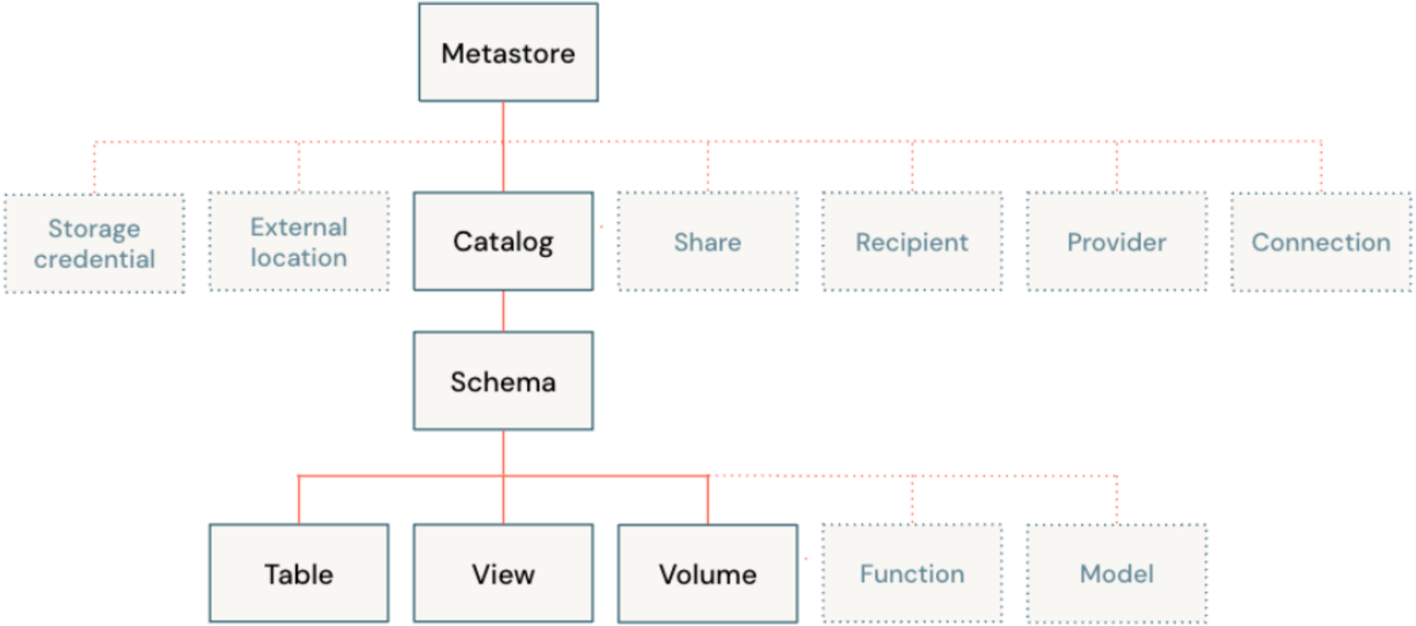
PROFINIT

- Using job clusters
  - Job clusters are terminated immediately after job is finished
- Consisting of tasks
  - Python script
  - Spark submit
  - Notebook
  - JAR, Python wheel
  - SQL – Query, Dashboard, Alert
  - Job
- Compute can be shared or different cluster can be selected for different tasks
- Run on demand/schedule/trigger
- Databricks native alternative to open source orchestration tools (AirFlow, Dagster, etc.)
- Can show nice DAG (graphical view)

Data objects

PROFINIT

- Kept and organized in cloud object storage (AWS S3, Azure Blob Storage,…)



Data objects

PROFINIT

- **Metastore**
  - Contains metadata of data objects
  - Configured with root storage in cloud object storage (e.g. S3 bucket in AWS)
  - Can be assigned to multiple workspaces
  - One workspace may have only a single metastore
- **Catalog**
  - The highest abstraction in DBX Lakehouse relational model
  - Collection of schemas (databases)
  - Default catalog is *hive_metastore*
- **Schema**
  - LOCATION on cloud object storage
  - Collection of tables, views and functions

Data objects

PROFINIT

- **Table**
  - Collection of structured data
  - Default storage provider – Delta Lake (https://delta.io/)
    - ACID transcations
    - Optimized performance (OPTIMIZE, Z-ORDER,…)
    - Driven by parquet
  - Managed table
    - In the same location as database
    - Metadata and data is managed by Databricks
    - DROP = delete data and metadata

```
CREATE TABLE table_name AS SELECT * FROM another_table
```

  - Unmanaged table
    - Only metadata is managed by Databricks
    - DROP = data is preserved

```
CREATE TABLE table_name
(field_name1 INT, field_name2 STRING)
LOCATION '/path/to/empty/directory'
```

Data objects

PROFINIT

- **View**
  - Query text is registered to the metastore (database)
  - No actual data is written

```sql
CREATE VIEW main.default.experienced_employee
  (id COMMENT 'Unique identification number', Name)
  COMMENT 'View for experienced employees'
AS SELECT id, name
   FROM all_employee
   WHERE working_years > 5;
```

- **Temporary view**
  - Limited scope and persistence
  - Not registered to metastore
  - Scopes:
    - Notebooks and jobs
    - Databricks SQL – query level
    - Global temporary views – cluster level

```sql
CREATE TEMPORARY VIEW subscribed_movies
  AS SELECT mo.member_id, mb.full_name, mo.movie_title
       FROM movies AS mo
       INNER JOIN members AS mb
          ON mo.member_id = mb.id;
```

Data objects

PROFINIT

- **User-defined function**
  - Associate user-defined logic with a database
  - In SQL or Python/Scala/Java
    - Code in Python can have a negative impact on performance
      - Outside of JVM – data serialization
      - Databricks have code optimizers for SQL, not Python
  - Usually not good for production workloads (instead use native Apache Spark methods if possible)

```sql
CREATE FUNCTION convert_f_to_c(unit STRING, temp DOUBLE)
RETURNS DOUBLE
RETURN CASE
  WHEN unit = "F" THEN (temp - 32) * (5/9)
  ELSE temp
END;

SELECT convert_f_to_c(unit, temp) AS c_temp
FROM tv_temp;
```

```python
def convertFtoC(unitCol, tempCol):
  from pyspark.sql.functions import when
  return when(unitCol == "F", (tempCol - 32) * (5/9)).otherwise(tempCol)

from pyspark.sql.functions import col

df_query = df.select(convertFtoC(col("unit"), col("temp"))).toDF("c_temp")
display(df_query)
```

Data objects

PROFINIT

- **Volume**
  - Represents logical volume of storage in cloud object storage location
  - Accessing, storing, governing and organizing files
  - Add governance over also to non-tabular datasets
  - Only in Unity Catalog
  - **Managed**

```
CREATE VOLUME myManagedVolume
  COMMENT 'This is my example managed volume';
```

```
SELECT * FROM csv.`dbfs:/Volumes/mycatalog/myschema/mymanagedvolume/sample.csv`
```

  - **External**

```
CREATE EXTERNAL VOLUME IF NOT EXISTS myCatalog.mySchema.myExternalVolume
      COMMENT 'This is my example external volume'
      LOCATION 's3://my-bucket/my-location/my-path'
```

```
SELECT * FROM csv.`/Volumes/mycatalog/myschema/myexternalvolume/sample.csv`
```

Advanced Databricks features – to be continued

PROFINIT

- Machine learning tooling
  - MLFlow, Scalable ML with Spark, AutoML, Model serving
- Delta Live Tables
  - ETL tool
  - Declarative definitions
  - A lot of „self optimization and maintanance“
  - Development or production modes
- Photon
  - New generation data processing engine
  - Written in C++
  - Compatible with Apache Spark APIs
- SQL warehouses
- Lakehouse federation
- LakehouseIQ

Real-world Databricks use cases

PROFINIT

- **Gucci**
  - **Use case: media budget allocation to maximize ROI** https://www.youtube.com/watch?v=mq3IxO_toDA
  - MLOps
  - Trying to adopt community-recommended best practices
  - Speed-up time to market
  - Benefit from managed ML services – distributed hyperparameter tuning with HyperOpt and Spark, MLFlow, AutoML (kick-off stage)
- **CDQ**
  - **Use case: migrate custom reporting ETL pipeline to Databricks**
  - Get scalable solution with usage of Delta Live Tables
  - Exploit Lakehouse architecture
  - Performance boost
- **Shell**
  - **Use case: Databricks as key tool in Shell.ai platform** https://www.databricks.com/customers/shell
  - Democratize data access in organization, supported cross-team collaboration, develop over 100 AI models

Q&A

PROFINIT

Thanks for attention!

Profinit EU, s.r.o.
Tychonova 2, 160 00 Praha 6

Tel.: + 420 224 316 016, web: www.profinit.eu

LINKEDIN
linkedin.com/company/profinit

TWITTER
@profinit_EU

FACEBOOK
facebook.com/Profinit.EU

YOUTUBE
Profinit EU, s.r.o.