Product safe harbor statement

This information is provided to outline Databricks' general product direction and is for informational purposes only. Customers who purchase Databricks services should make their purchase decisions relying solely upon services, features, and functions that are currently available. Unreleased features or functionality described in forward-looking statements are subject to change at Databricks discretion and may not be delivered as planned or at all

DATA+AI SUMMIT
BY databricks

PYTHON STORED PROCEDURES

SUPERCHARGE YOUR DATA WAREHOUSE

Go for it!

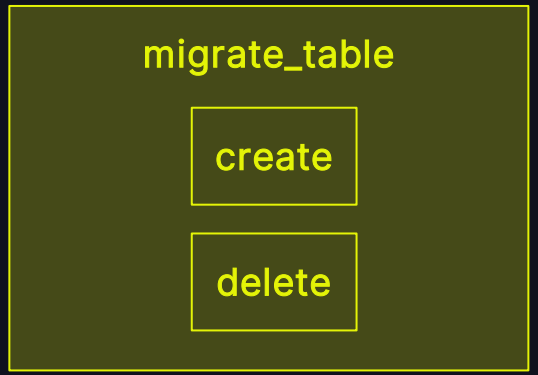
Allison Wang,
Sr. Software Engineer, Databricks

Jakob Mund,
Staff Product Manager, Databricks

(PYTHON) STORED PROCEDURES?

(PYTHON) STORED PROCEDURES?

Encapsulated and named

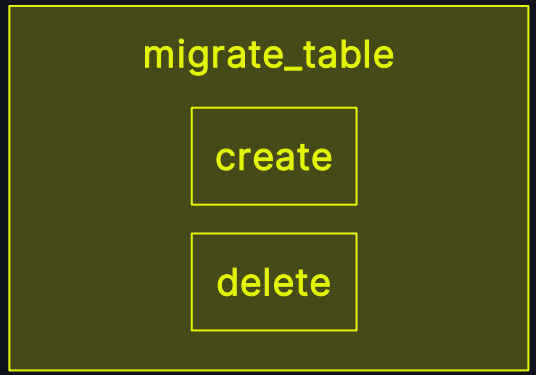


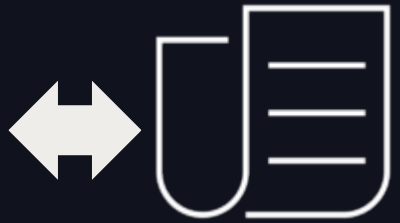
Unity Catalog

Stored and governed in Unity Catalog

(PYTHON) STORED PROCEDURES?

Users

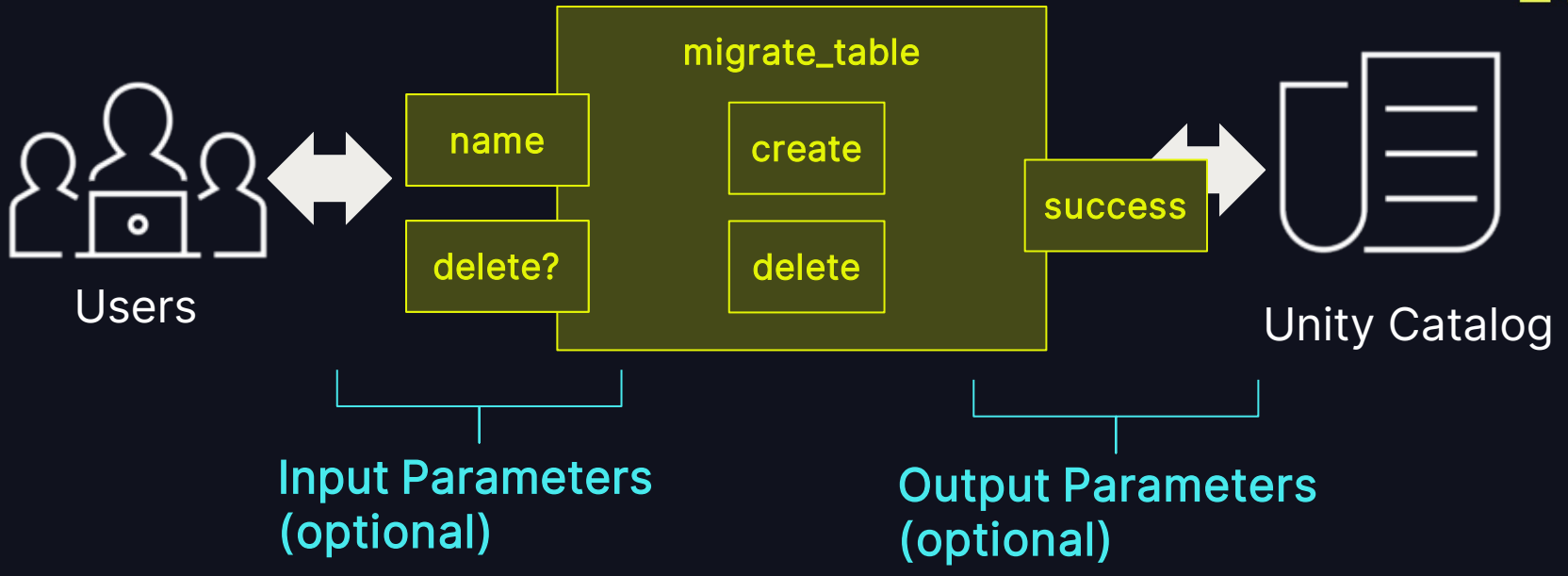




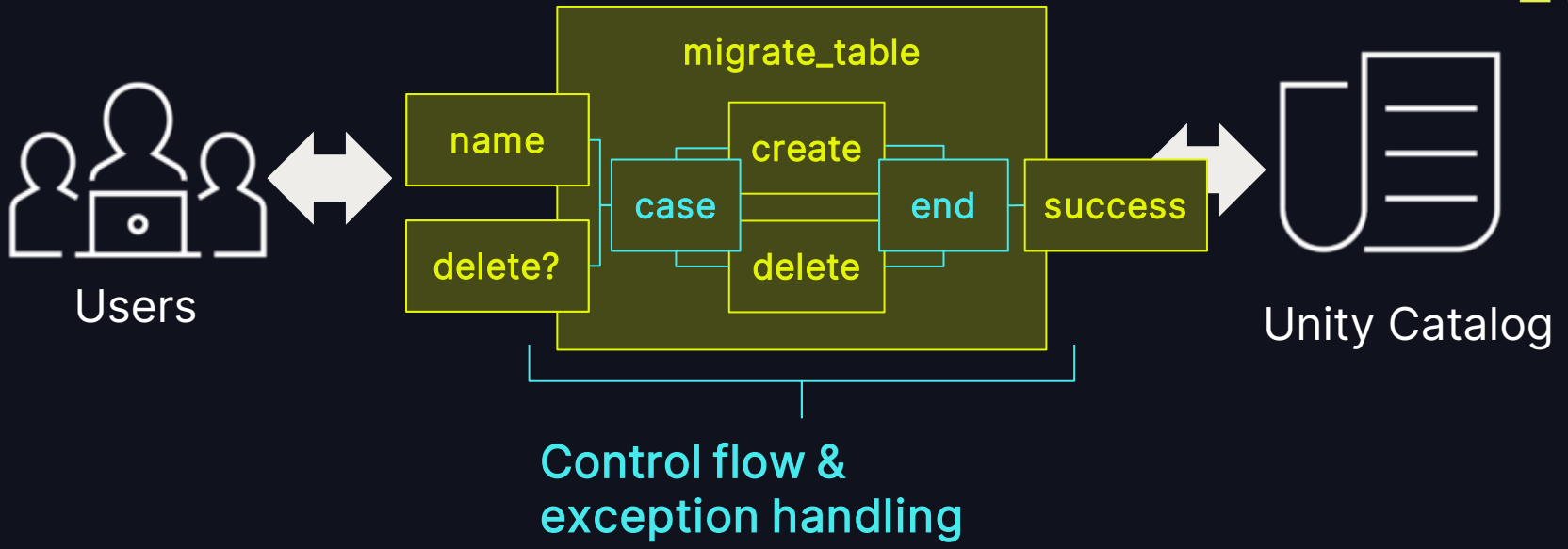
Unity Catalog

Shareable among users

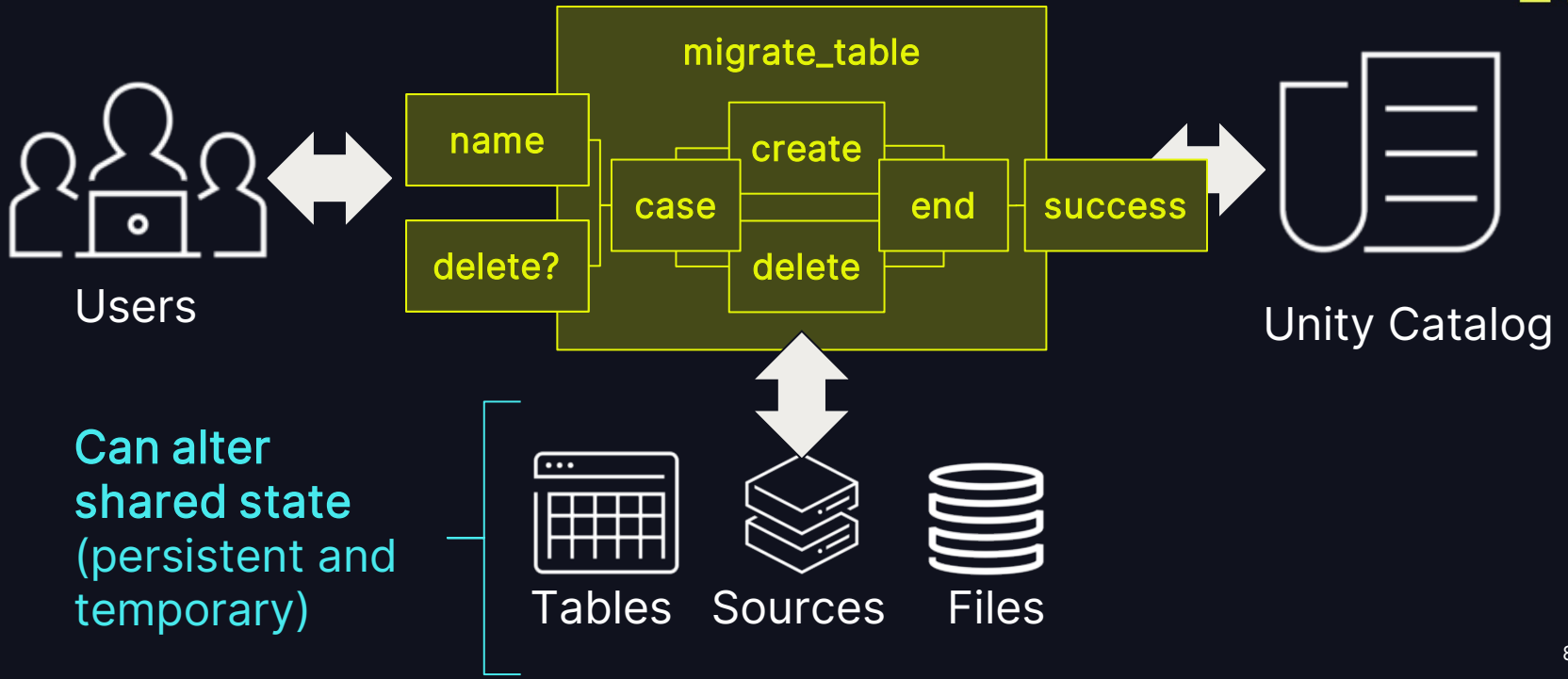
(PYTHON) STORED PROCEDURES?



(PYTHON) STORED PROCEDURES?

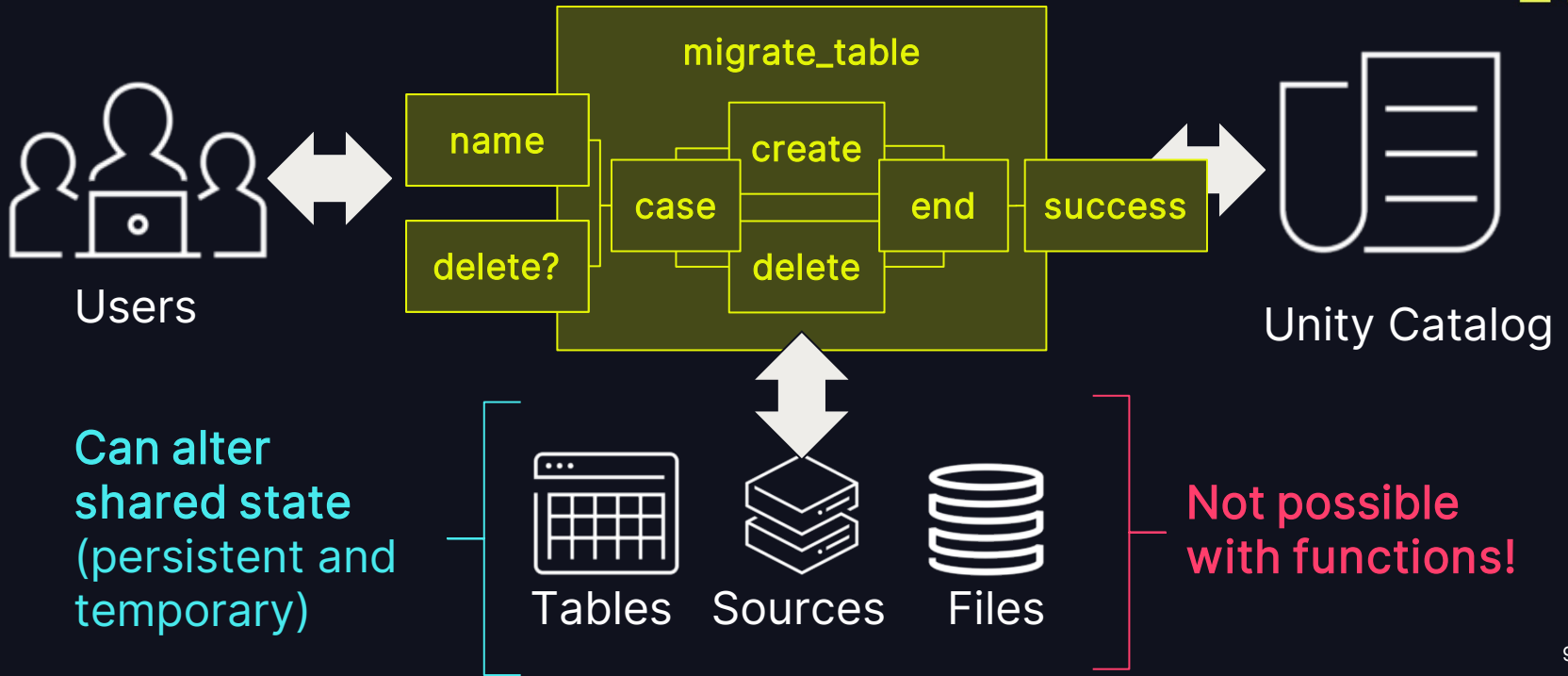


(PYTHON) STORED PROCEDURES?



Can alter shared state (persistent and temporary)

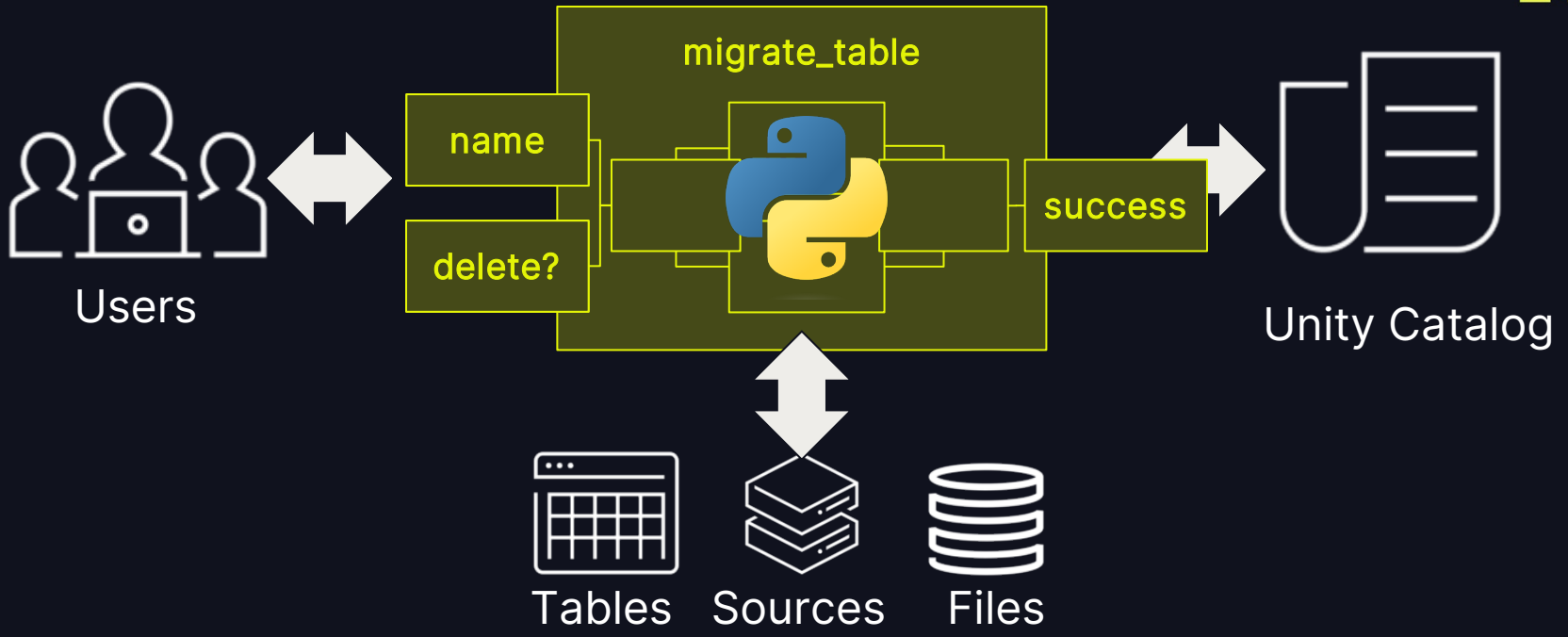
(PYTHON) STORED PROCEDURES?



Can alter shared state (persistent and temporary)

Not possible with functions!

(PYTHON) STORED PROCEDURES?



USE CASES

Administrative Tasks

- Run Python logic to accomplish sophisticated administrative tasks specific to you
- E.g., migrations, data integrity checks, custom audits

Customized Permissions

- Write logic to execute specific actions otherwise requiring a (higher) priviledge
- Grant users the permission to execute the procedure instead of granting the privilege
- E.g., validate naming schemas or enforce a certain location when creating a table

Uplevel users & workloads

- Securely share and re-use logic written in Python
- Leverage Python libraries to achieve more with less
- E.g., advanced ETL transformations, writing to multiple tables/destinations,

ANYTHING MISSING?

PYTHON STORED PROCEDURES

ON DATABRICKS

CREATING A STORED PROCEDURE

How to create a Python stored procedure

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

CREATING A STORED PROCEDURE

How to create a Python stored procedure

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

<-Name of the procedure

Can also be a three-level namespace: catalog.schema.procedure_name

CREATING A STORED PROCEDURE

How to create a Python stored procedure

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

<- Parameters

- IN: Input parameter. Default parameter mode. Optional to specify.
- OUT: Output parameter.
- INOUT: Input/Output parameter.

CREATING A STORED PROCEDURE

How to create a Python stored procedure

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

<- Language of the procedure

CREATING A STORED PROCEDURE

How to create a Python stored procedure

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

<- Procedure body: the Python code that calculates the area and updates the accumulator.

INVOKING A STORED PROCEDURE

How to execute a stored procedure

```
> CALL area_of_rectangle(
    x => 1,
    y => 2,
    area => null,
    acc => 0)


    +----+---+
    |area|acc|
    +----+---+
    |   2|  2|
    +----+---+
```

<- CALL command

- Specify the IN and the INOUT parameters
- Use NULL as a placeholder for the OUT parameters
- Support named arguments

<- Return value of the procedure

- One row with multiple columns
- Each column represents an OUT or INOUT parameter specified in the procedure parameters

VARIABLE BINDING

How to bind IN, OUT and INOUT parameters

```
CREATE OR REPLACE PROCEDURE area_of_rectangle(
    IN x INT,
    IN y INT,
    OUT area INT,
    INOUT acc INT
)
LANGUAGE PYTHON
AS $$
  area = x * y
  acc = acc + area
$$
```

<- All three types of parameters (IN, OUT and INOUT) can be referenced directly in the procedure body using their names.
<- For OUT and INOUT parameters, the assigned values will be returned as the output of the CALL command.

ACCESS CONTROL

Unified governance with Unity Catalog

- Grant and revoke permissions to execute stored procedures

```
GRANT EXECUTE ON PROCEDURE area_of_rectangle TO `user`

REVOKE EXECUTE ON PROCEDURE area_of_rectangle FROM `user`
```

WAIT, WHY PROCEDURES?

Why do we want to use procedures instead of functions?

- You can access Spark session in procedures!
- This means you can:
  - Run SQL queries: `spark.sql("SELECT * FROM my_table")`
  - Use PySpark: `df.groupby(...).select(...)`
- Benefits:
  - Perform data transformations and analysis right within the procedure
  - Seamlessly encapsulate your data processing logic into reusable components.

SPARK SESSION IN STORED PROCEDURES

The super power of Python stored procedures

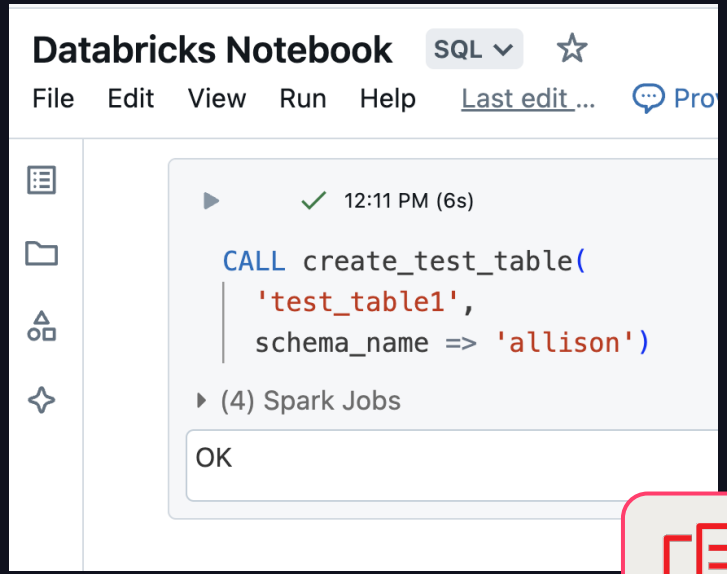
```
CREATE OR REPLACE PROCEDURE create_test_table(table_name STRING)
LANGUAGE PYTHON
AS $$
if not table_name.startswith("test_"):
  raise Exception(f"Table name must starts with 'test_'")

spark.sql(f"""CREATE TABLE {table_name} AS SELECT 1 AS id""")
$$
```

Access the Spark session and execute SQL statements via an implicit variable called `spark`.

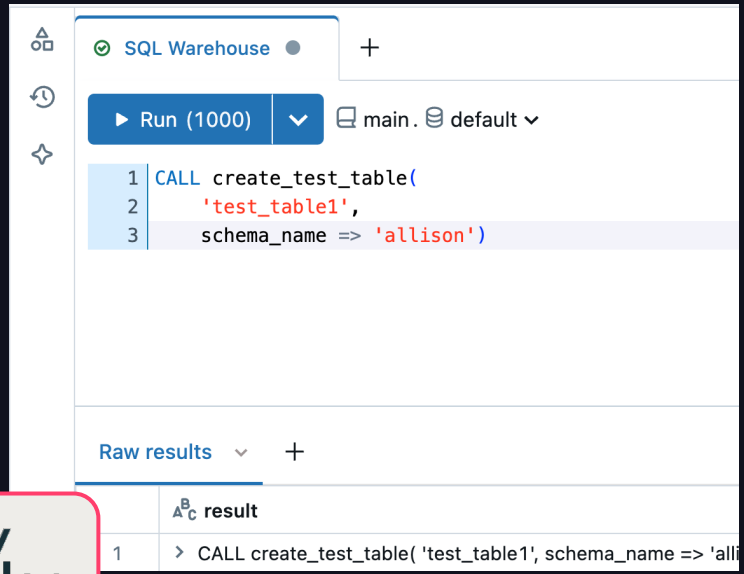
SPARK SESSION IN STORED PROCEDURES

Executable from any Unity Catalog Enabled Compute

Databricks Clusters



Databricks SQL Warehouse



unity catalog

MORE EXAMPLES

Perform an ETL operation using the PySpark API

```
CREATE OR REPLACE PROCEDURE etl_demo(source STRING, target STRING)
LANGUAGE PYTHON
AS $$
 from pyspark.sql.functions import col

 source_df = spark.table(source)

 transformed_df = (source_df
   .groupBy("category")
   .agg({"value": "sum"})
   .withColumnRenamed("sum(value)", "total_value"))

 transformed_df.write.format("delta").mode("overwrite").saveAsTable(target)
$$
```

MORE EXAMPLES

Perform an ETL operation using the PySpark API

```
CREATE OR REPLACE PROCEDURE etl_demo(source STRING, target STRING)
LANGUAGE PYTHON
AS $$
 from pyspark.sql.functions import col

 source_df = spark.table(source)

 transformed_df = (source_df
   .groupBy("category")
   .agg({"value": "sum"})
   .withColumnRenamed("sum(value)", "total_value"))

 transformed_df.write.format("delta").mode("overwrite").saveAsTable(target)
$$
```

Read from a source table

MORE EXAMPLES

Perform an ETL operation using the PySpark API

```
CREATE OR REPLACE PROCEDURE etl_demo(source STRING, target STRING)
LANGUAGE PYTHON
AS $$
 from pyspark.sql.functions import col

 source_df = spark.table(source)

 transformed_df = (source_df
   .groupBy("category")
   .agg({"value": "sum"})
   .withColumnRenamed("sum(value)", "total_value"))

 transformed_df.write.format("delta").mode("overwrite").saveAsTable(target)
$$
```

Transform the dataframe using the PySpark API

MORE EXAMPLES

Perform an ETL operation using the PySpark API

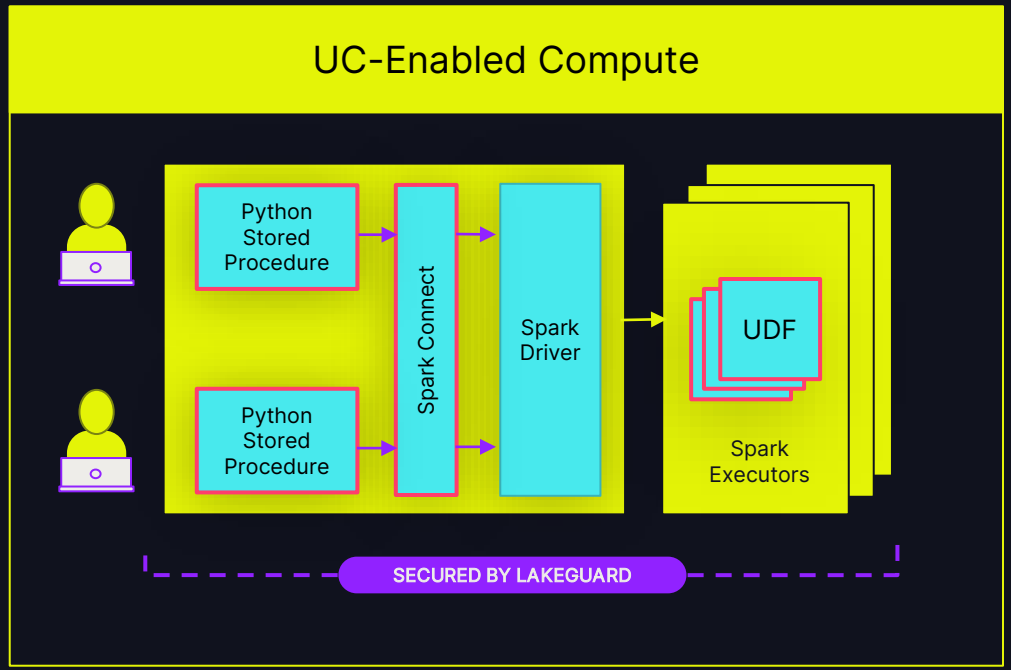
```
CREATE OR REPLACE PROCEDURE etl_demo(source STRING, target STRING)
LANGUAGE PYTHON
AS $$
 from pyspark.sql.functions import col

 source_df = spark.table(source)

 transformed_df = (source_df
   .groupBy("category")
   .agg({"value": "sum"})
   .withColumnRenamed("sum(value)", "total_value"))

 transformed_df.write.format("delta").mode("overwrite").saveAsTable(target)
$$
```

Write to a target table

HOW IT WORKS

Unity Catalog Lakeguard and Spark Connect

- Lakeguard enforces governance at runtime
- User code is executed in a sandbox
- Access to Spark session via Spark Connect



DEMO TIME

PYTHON STORED PROCEDURES

Recap & How to get involved

Write Stored Procedures in Unity Catalog for

- Administrative Tasks
- Customized Permissions
- Uplevel users & workloads

Provide feedback & sign up for preview and updates

https://forms.gle/PNvJ6sMajgAM2fZ6A

Don't be a stranger, reach out to us!

I'm hosting a Braindate™ at DATA+AI SUMMIT by databricks

Re-use and share Python code on SQL warehouses and UC compute using Python Stored Procedures

Email: python-stored-procedures@databricks.com

DATA+AI SUMMIT