Lakeflow Declarative Pipelines

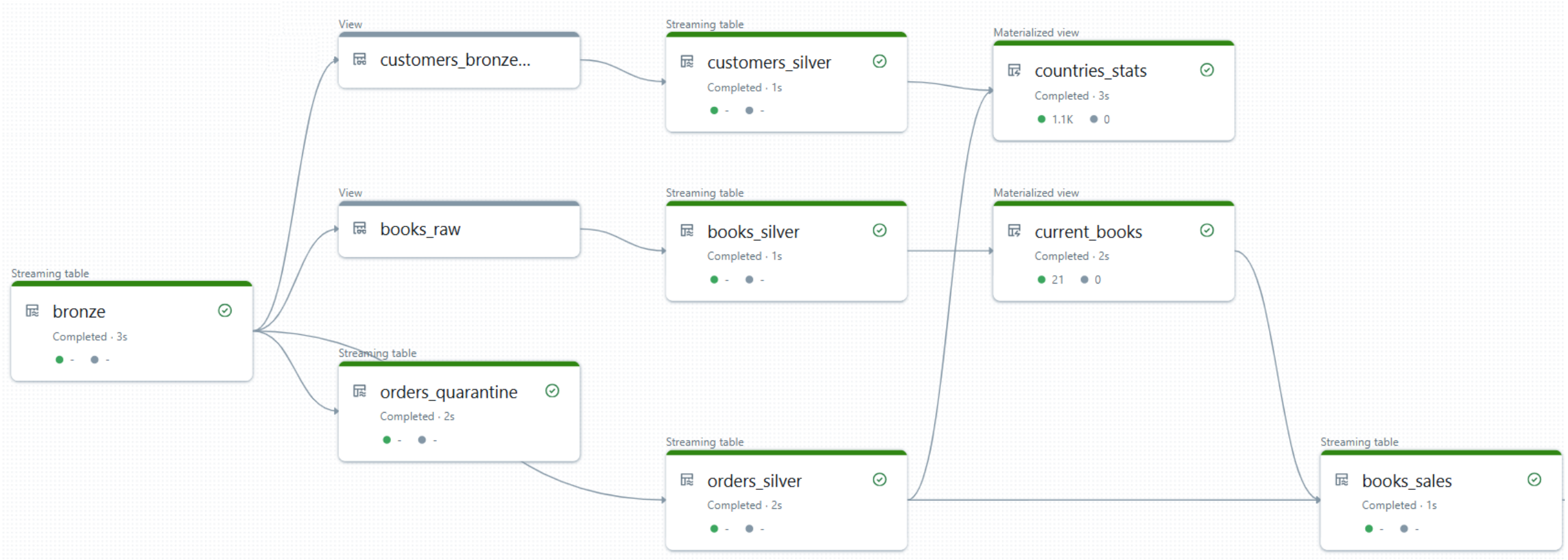
Learning Objectives

- Definition of Lakeflow Declarative Pipelines
- Differences with Apache Spark
- Object Types

Lakeflow Declarative Pipelines

- LDP is a declarative ETL framework powered by Apache Spark for building reliable and maintainable data pipelines.
- Previously known as Delta Live Tables (DLT)
- Open source: Spark Declarative Pipelines

Example


Streaming table
bronze
Completed · 3s
View
customers_bronze...
View
books_raw
Streaming table
orders_quarantine
Completed · 2s
Streaming table
customers_silver
Completed · 1s
Streaming table
books_silver
Completed · 1s
Streaming table
orders_silver
Completed · 2s
Materialized view
countries_stats
Completed · 3s
1.1K
0
Materialized view
current_books
Completed · 2s
21
0
Streaming table
books_sales
Completed · 1s

Benefits

- Automatic orchestration
- Handle checkpoints, retires, and optimization
- Easy to implement CDC, SCD Type 2, and data quality control

Spark vs. LDP

Spark

```
spark.readStream
        .format("cloudFiles")
        .option("cloudFiles.format", "json")
        .load('/some/path/')
    .writeStream
        .option("checkpointLocation", "/path")
        .table("orders_raw")
```

LDP

```
from pyspark import pipelines as dp

@dp.table
def orders_raw():
    return (spark.readStream
                .format("cloudFiles")
                .option("cloudFiles.format", "json")
                .load("/some/path")
            )
```

Spark vs. LDP

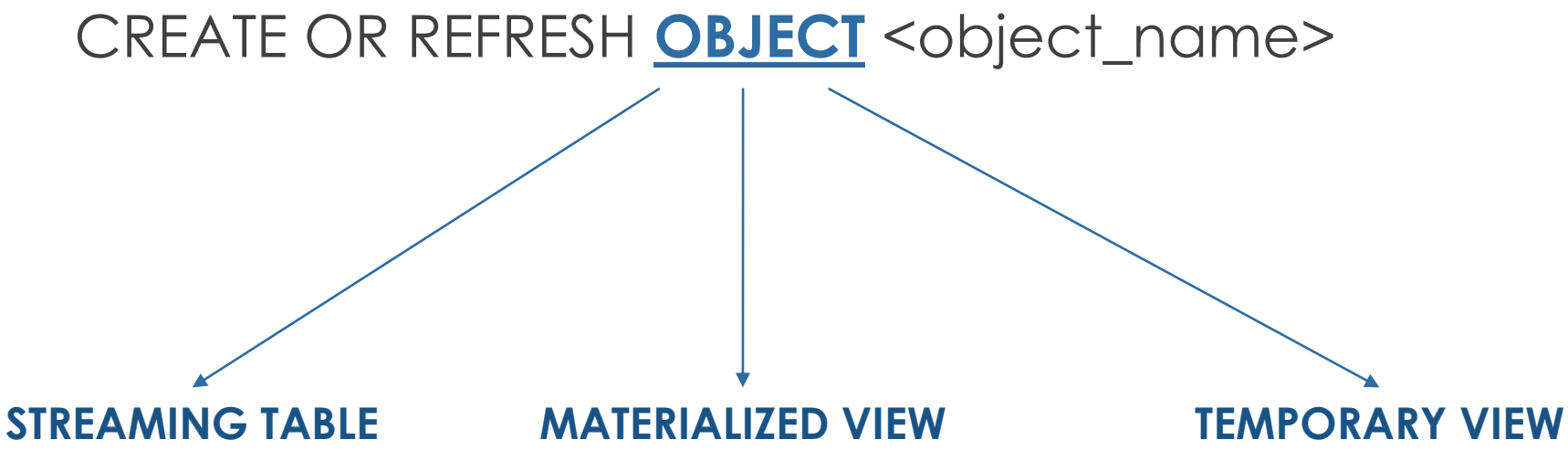
Spark

- Can Not create streaming tables in Spark SQL syntax only. Need to pass by PySpark to register streaming tables

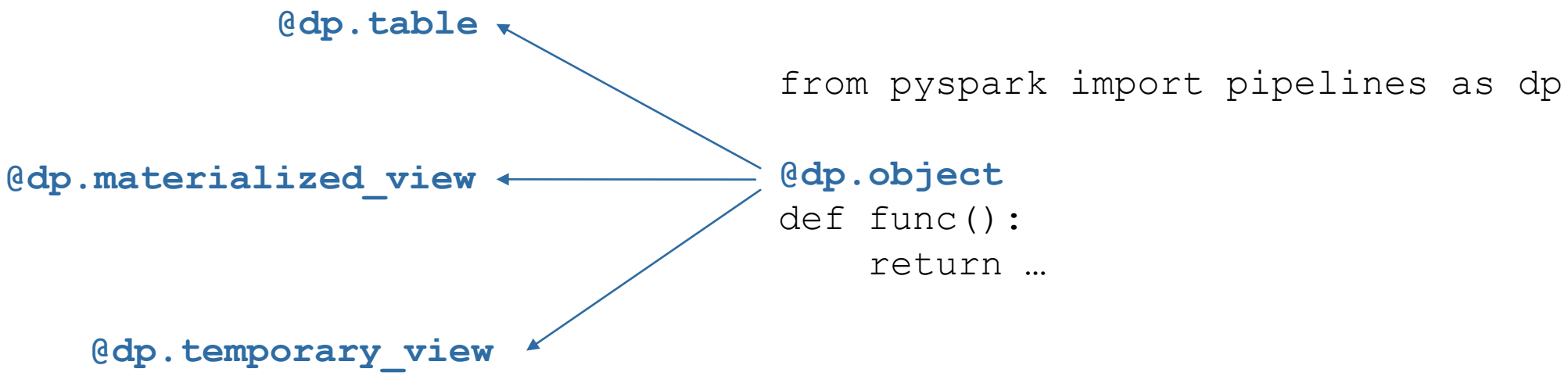
LDP

- Support creating streaming tables in SQL via **CREATE STREAMING TABLE** command

Creating LDP Objects (SQL)



Creating LDP Objects (Python)



LDP Object Types

Streaming Tables

- Permanent objects
- Handle incremental refresh
- Used for data ingestion from streaming sources (append-only):
  - spark.**readStream**
  - **STREAM()** SQL function
- Support near real-time data ingestion

Materialized Views

- Permanent objects
- Handle full or incremental refresh (Serverless-only)
- Used for precomputing complex BI queries, or for data ingestion from non-streamable sources:
  - spark.**read**
- Not designed for low-latency use cases

Temporary Views

- Temporary objects
- Handle temporarily processed data
- Used for intermediate transformations and data quality checks

LDP vs DLT

	LDP	DLT (Old Syntax)
Module import	`from pyspark import pipelines as dp`	`import dlt`

Streaming Table	`@dp.table`
	`def func()`
	`  return spark.readStream …`
		`@dlt.table`
		`def func()`
		`  return spark.readStream …`

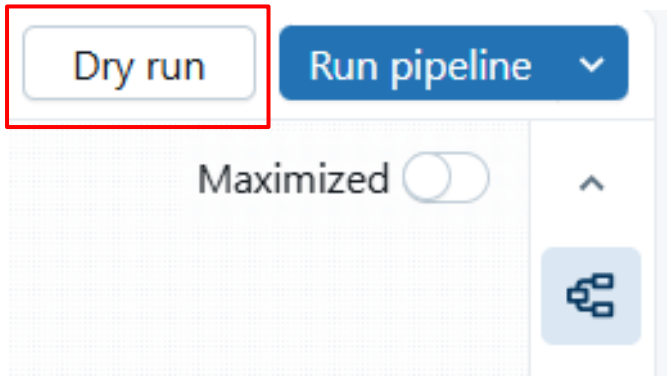
Materialized View	`@dp.materialized_view`
	`def func()`
	`  return spark.read …`
		`@dlt.table`
		`def func()`
		`  return spark.read …`

Temporary View	`@dp.temporary_view`
	`def func ()`
	`  return …`
		`@dlt.view`
		`def func()`
		`  return …`

Validating Code

LDP

- Files (.py or .sql)



DLT

- Notebooks