PyCon Australia 2023

Building 3D Trusted Data Pipelines With Dagster, Dbt, and Duckdb

1

Danh Phan

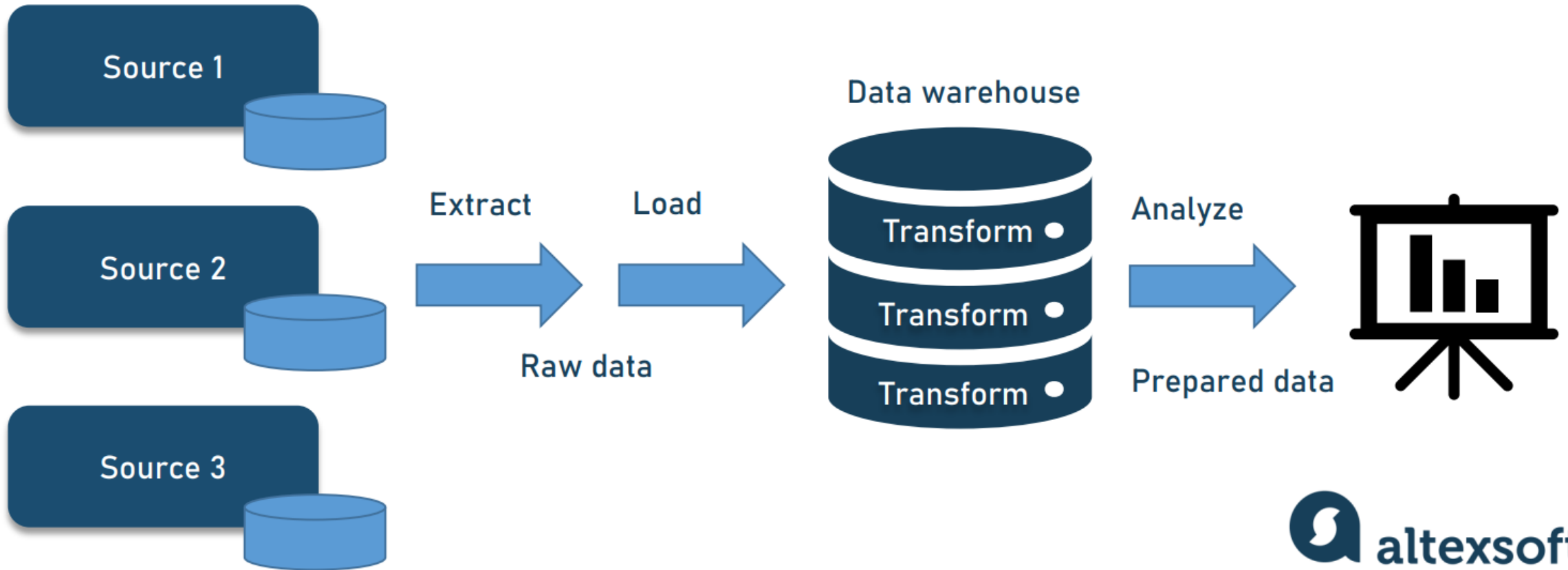**Adelaide, August 2023**

About me

Danh Phan

- Senior Data engineer at IDS, HESTA.
- An open-source contributor & a camper
- Ex. ML Researcher, Ex. Data analyst,
- Ex. Web dev, Ex. Database designer.
- https://danhphan.net

Views And Opinions Are My Own!

Overview

- Context

- Trusted data pipelines

- Demo

ELT PIPELINE

Source 1

Source 2

Source 3

Extract

Load

Raw data

Data warehouse

Transform

Transform

Transform

Analyze

Prepared data

altexsoft

Need robust/reliable pipelines

Interesting talks on this topic

- 2022 Open Data Science, **Sam Bail :** Building a Robust Data Pipeline with the "dag Stack": dbt, Airflow, and Great Expectations

- 2021 Bigeye, **Egor Gryaznov** : Data Reliability Engineering—Reliable Data Pipelines 101

- 2021 DataEngBytes, **Harmeet Sokhi** : Shift-left testing : Building reliable Data Pipelines

- 2019 DataBricks, **Steven Yu** : Building Robust Production Data Pipelines with Databricks Delta

- 2017 DataBricks, **Xiao Li** : Building Robust ETL Pipelines with Apache Spark

- 2016 Jfokus, **Lars Albertsson** : Data pipelines from zero to solid

- …

Lessons-learned

- Testing data pipelines (***)

- DevOps practice: code versioning, CI/CD

- Infrastructure as code (Terraform , CloudFormation, …)

- Container environment (Docker)

- Data lineage and monitoring

- Enhance data contracts from upstream

- …

Robust is not enough!

- Robust/reliable is not enough

- We absolutely need it

- But we also need to deliver the data quality to our data consumers (data analysts, BI developers, or Data scientists)

Trusted data pipelines also focus on data quality!

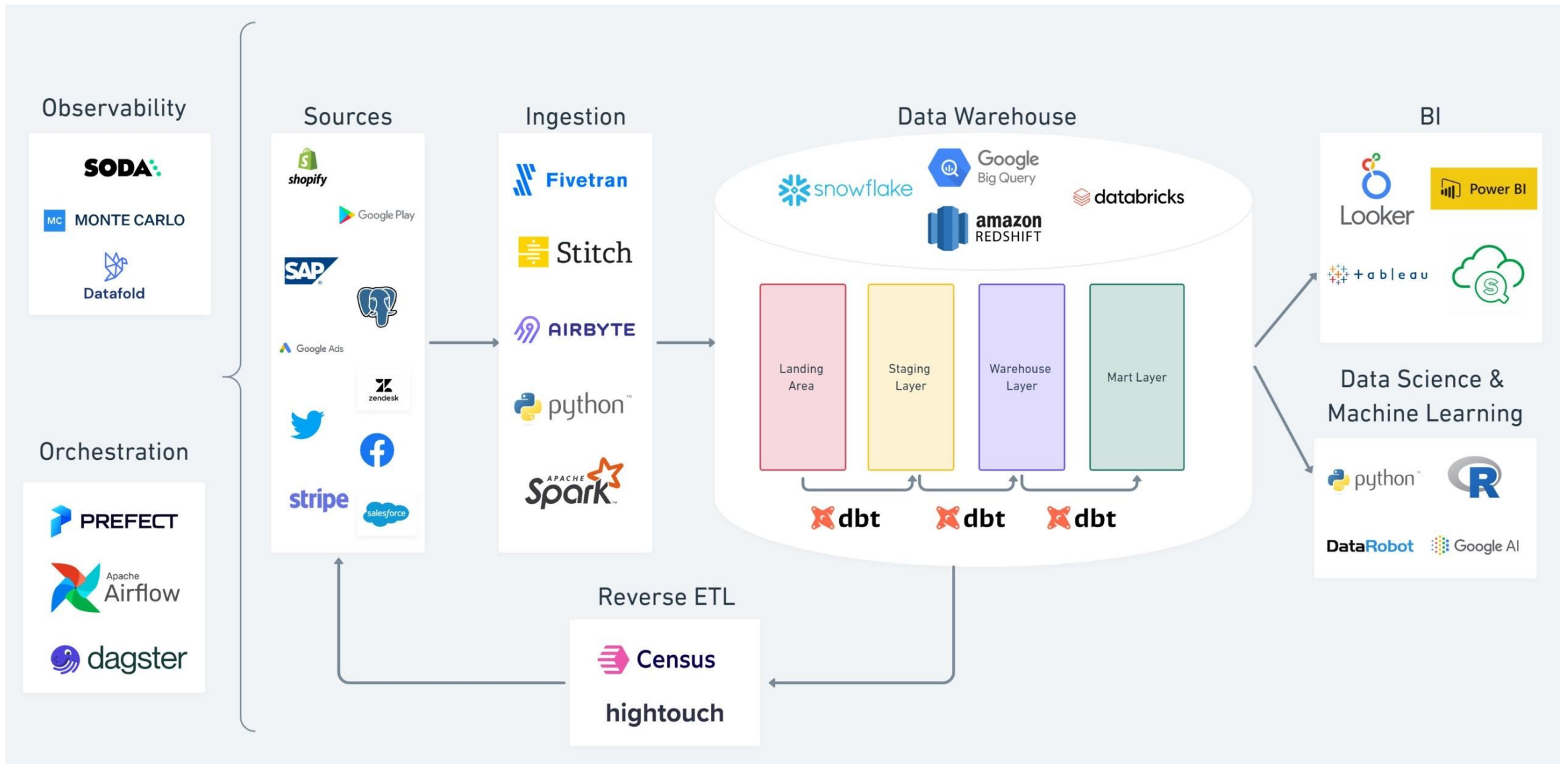# Robust pipeline vs. Trusted pipeline

Engineering focus

Data consumers' focus

Data quality questions?

- Is that table the same compared to the upstream table?

- This table needs to have these columns

- Where this data come from?

- Is the data arrived late?

- Is the values of a numeric column in an expected range?

- Is the summary stats make sense?

- Seem to have outliers?

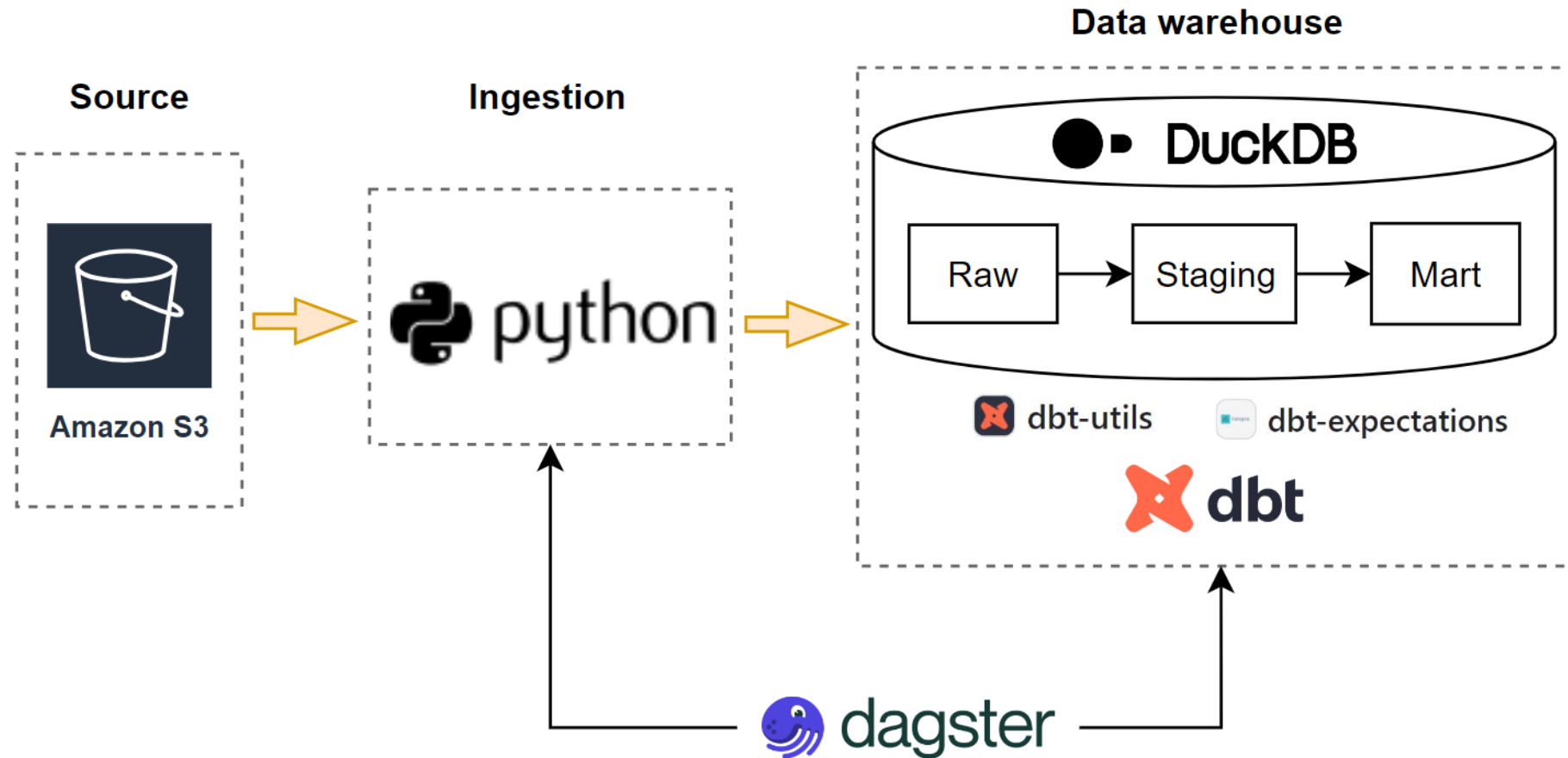- We expect the column to have no null value?

- …   **It is critical to test and measure data quality!**
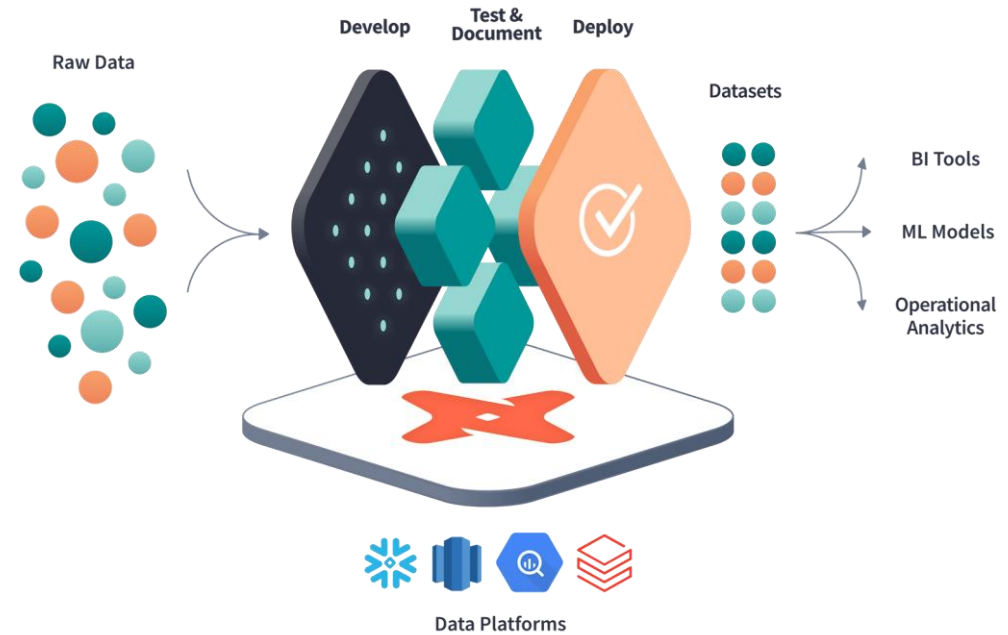
IN DATA

WE TRUST

https://bitestreams.com/nl/blog/modern_data_stack/

3D (Dagster, Dbt, and Duckdb) trusted pipeline

The 3Ds

Converting an ETL script to Software-Defined Assets

🌀 dagster

DuckDB is an in-process
SQL OLAP database management system

Testing data quality

- **Between tables**
  - Relationships
  - Raw (unprocessed) truth vs processed truth

- **In a table**
  - Row counts
  - Column counts
  - Column exist
  - …

- **Column level:**
  - Generic test:
    - Not null (with a threshold level)
    - Unique
  - Text data:
    - Following a pattern?
    - Accepted values?
  - Numeric data:
    - Min, max, mean, median
    - Outliers
  - Date:
    - Recency, range, latest, min, max

Testing data quality

Four generic tests already defined in Dbt:

- unique,
- not_null,
- accepted_values
- relationships.

```yaml
version: 2

models:
  - name: orders
    columns:
      - name: order_id
        tests:
          - unique
          - not_null
      - name: status
        tests:
          - accepted_values:
              values: ['placed', 'shipped', 'completed', 'returned']
      - name: customer_id
        tests:
          - relationships:
              to: ref('customers')
              field: id
```

15

Testing data quality

- Add testing packages into dbt *packages.yml* file:

```
1  packages:
2    - package: dbt-labs/dbt_utils
3      version: 1.1.1
4    - package: calogica/dbt_expectations
5      version: [">=0.8.0", "<0.9.0"]
```

- Install dbt packages: **dbt deps**

- Run test: **dbt tests**

Testing data quality

- **Dbt-utils** Generic Tests

- equal_rowcount (source)
- fewer_rows_than (source)
- equality (source)
- expression_is_true (source)
- recency (source)
- at_least_one (source)
- not_constant (source)
- not_empty_string (source)
- cardinality_equality (source)

- not_null_proportion (source)
- not_accepted_values (source)
- relationships_where (source)
- mutually_exclusive_ranges (source)
- sequential_values (source)
- unique_combination_of_columns (source)
- accepted_range (source)
- Grouping in tests

Testing data quality

- **Dbt-expectations**

Table shape

- expect_column_to_exist
- expect_row_values_to_have_recent_data
- expect_grouped_row_values_to_have_recent_data
- expect_table_aggregation_to_equal_other_table
- expect_table_column_count_to_be_between
- expect_table_column_count_to_equal_other_table
- expect_table_column_count_to_equal
- expect_table_columns_to_not_contain_set
- expect_table_columns_to_contain_set
- expect_table_columns_to_match_ordered_list
- expect_table_columns_to_match_set
- expect_table_row_count_to_be_between
- expect_table_row_count_to_equal_other_table
- expect_table_row_count_to_equal_other_table_times_factor

Missing values, unique values, and types

- expect_column_values_to_be_null
- expect_column_values_to_not_be_null
- expect_column_values_to_be_unique
- expect_column_values_to_be_of_type
- expect_column_values_to_be_in_type_list
- expect_column_values_to_have_consistent_casing

Sets and ranges

- expect_column_values_to_be_in_set
- expect_column_values_to_not_be_in_set
- expect_column_values_to_be_between
- expect_column_values_to_be_decreasing
- expect_column_values_to_be_increasing

Testing data quality

- **Dbt-expectations**

Multi-column

- expect_column_pair_values_A_to_be_greater_than_B
- expect_column_pair_values_to_be_equal
- expect_column_pair_values_to_be_in_set
- expect_compound_columns_to_be_unique
- expect_multicolumn_sum_to_equal
- expect_select_column_values_to_be_unique_within_record

Distributional functions

- expect_column_values_to_be_within_n_moving_stdevs
- expect_column_values_to_be_within_n_stdevs
- expect_row_values_to_have_data_for_every_n_datepart

String matching

- expect_column_value_lengths_to_be_between
- expect_column_value_lengths_to_equal
- expect_column_values_to_match_like_pattern
- expect_column_values_to_match_like_pattern_list
- expect_column_values_to_match_regex
- expect_column_values_to_match_regex_list
- expect_column_values_to_not_match_like_pattern
- expect_column_values_to_not_match_like_pattern_list
- expect_column_values_to_not_match_regex
- expect_column_values_to_not_match_regex_list

Global Asset Lineage

⟳ Reload definitions

✳️ Type an asset subset... (ex: ++jaffle_ɛ)   0:15 ⟳   ✨ **Materialize all** ▾
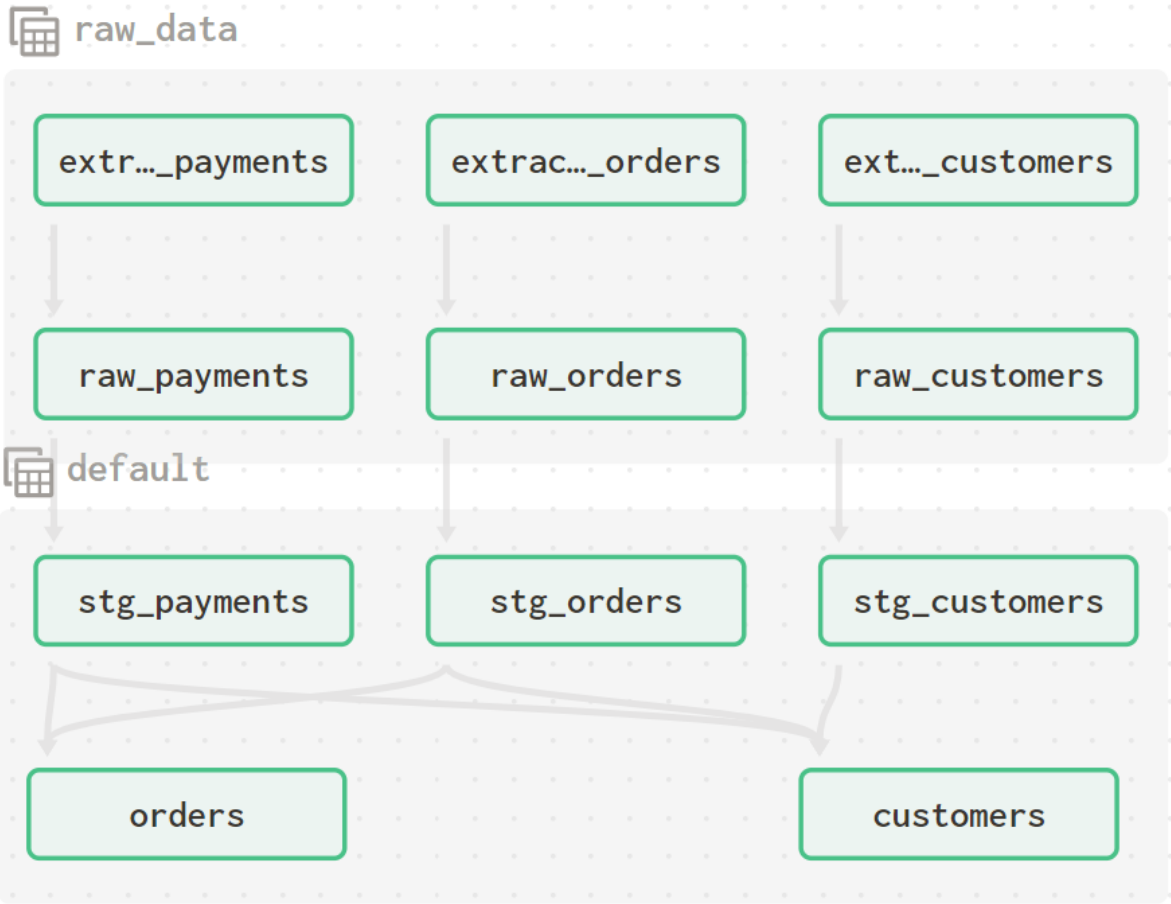
🗂️ **raw_data**

extr..._payments	extrac..._orders	ext..._customers
raw_payments	raw_orders	raw_customers

🗂️ default

stg_payments	stg_orders	stg_customers
orders		customers

🔍+

🔍−

⬇

20

🔍 Search...   /   ⚙️

Global Asset Lineage

🔄 Reload definitions

✳️ Type an asset subset... (ex: ++jaffle_a   0:00 🔄   ✨ Materialize all ▾

🔳 raw_data

🔲 extract_raw_payments		🔲 extract_raw_orders		🔲 extract_raw_customers	
No description		No description		No description	
Materialized	30 July, 5:48 pm	Materialized	30 July, 5:48 pm	Materialized	30 July, 5:48 pm

🔲 raw_payments		🔲 raw_orders		🔲 raw_customers	
No description		No description		No description	
Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm

🔳 default

🔍+

🔍−

21

Global Asset Lineage

↻ Reload definitions

✳️ Type an asset subset... (ex: ++jaffle_a   0:06 ↻   ✨ **Materialize all** ▾
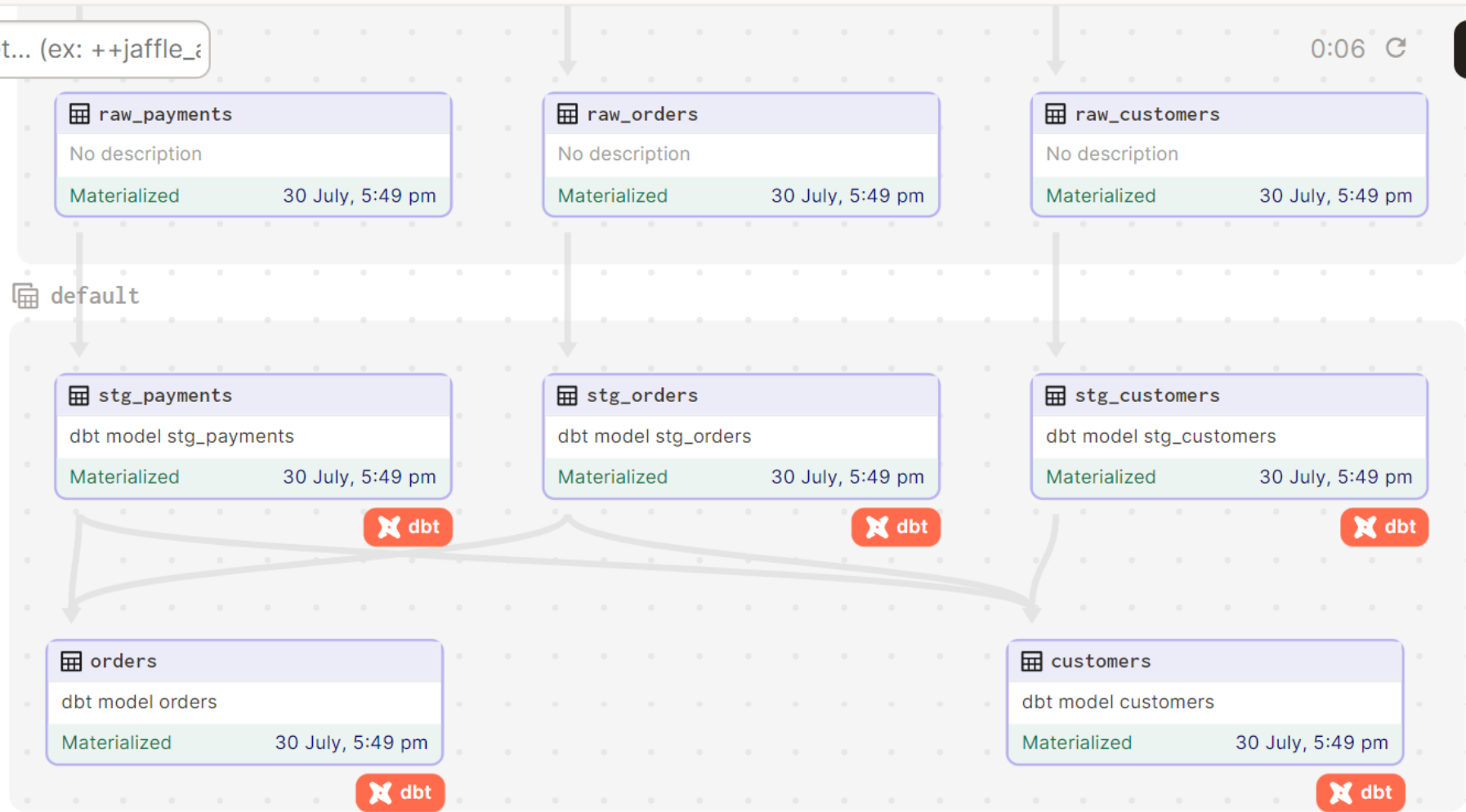
⊞ raw_payments		⊞ raw_orders		⊞ raw_customers	
No description		No description		No description	
Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm

⊞ default

⊞ stg_payments		⊞ stg_orders		⊞ stg_customers	
dbt model stg_payments		dbt model stg_orders		dbt model stg_customers	
Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm

✖ dbt   ✖ dbt   ✖ dbt

⊞ orders		⊞ customers	
dbt model orders		dbt model customers	
Materialized	30 July, 5:49 pm	Materialized	30 July, 5:49 pm

✖ dbt   ✖ dbt

🔍+
🔍−
⊙

Demo

Building trust is a journey!

- Building trusted data is a journey, which needs a good road map

- And strong leaderships with a right data strategy

- And buy-in of data stakeholders and data customers

- Well designed data schema

- Control data quality from upstream (data contract), on the way (data warehouse / data lake), and downstream

References

➢ The Demo Project repo : https://github.com/danhphan/trusted-data-pipeline

➢ Dbt tests: https://docs.getdbt.com/docs/build/tests

➢ Dbt Workshop: Advanced Testing: https://www.youtube.com/watch?v=fo7lUn6vgtg

➢ Build a poor man's data lake from scratch with DuckDB : https://dagster.io/blog/duckdb-data-lake

➢ Building a robust data pipeline with Dbt, Airflow, and Great Expectations: https://www.getdbt.com/coalesce-2020/building-a-robust-data-pipeline-with-dbt-airflow-and-great-expectations/

➢ Dagster repo: https://github.com/dagster-io/dagster

➢ Dbt repo: https://github.com/dbt-labs/dbt-core

➢ Duckdb repo: https://github.com/duckdb/duckdb

➢ Dbt-utils repo: https://github.com/dbt-labs/dbt-utils

➢ Dbt-expectation repo: https://github.com/calogica/dbt-expectations

Thank you!