## What is ODP

* ODP allows individuals on data teams to build data pipelines using JSON/YAML config instead of code.
* It is a python library built on top of the dagster orchestrator.
* The library translates user defined config to dagster primitives such as assets, resources, sensors, schedules, partitions and asset checks.
* Supports ELT pipelines through Dagster’s DBT integration. ETL pipelines are not supported yet (there isn’t a pre-built spark integration yet)

## Advantages

### Pipelines without code

* Use the predefined tasks or custom tasks to build no-code pipelines
* Allows data scientists or data analysts to build pipelines using tasks defined by data engineers

### Declarative pipelines

* Decouple task logic from DAG/pipeline logic
* Iterate on task logic without touching pipelines that use those tasks
* Separate CI/CD and review processes for tasks and pipelines

### Task re-usability

* Define tasks once and re-use them across pipelines

### Pre-defined task libraries

* Common GCP tasks
* Common DuckDB tasks
* Run shell commands

### Pre-built integrations

* DLT integration allows users to ingest data from pre-defined sources without code
* Define Data Quality checks in yaml and incorporate them in dagster asset checks without code

### Validation

* Pydantic validation for all config files
* Auto validation of parameters defined in custom tasks

## Dagster functionality supported

1. Assets
2. Asset materialization metadata
3. Resources
4. Sensors
5. Schedules
6. DBT integration
7. Only time partitions ( are not supported)
8. Asset checks through Soda

## Dagster functionality on the roadmap

1. Static, dynamic and multi-dimensioned partitions
2. Config schema