dbt SQL data transformation

CONCEPTS

A model is one SELECT in models/*.sql; dbt wraps it in CREATE VIEW or CREATE TABLE and runs models in dependency order.

Sources raw tables
↓ source()
Staging models views
↓ ref()
Marts tables
↓ dbt test
Docs lineage graph

Never hard-code table names: ref() and source() build the DAG that selection and run order depend on.

dbt_project.yml describes the project; profiles.yml (in ~/.dbt by default) holds connections and stays out of git.

PROJECT

dbt init *<project>*
Scaffold a project and profile

dbt debug
Check profile, project and connection

dbt deps
Install packages from packages.yml

dbt parse
Parse the project, write the manifest

dbt clean
Delete target/ and dbt_packages/

PROJECT FILES

Path	Holds
dbt_project.yml	name, profile, folder configs
profiles.yml	targets and credentials
models/*.sql	one SELECT per model
models/*.yml	tests, docs, sources
seeds/*.csv	small static tables
snapshots/	SCD type-2 history
macros/*.sql	reusable Jinja
packages.yml	package dependencies
target/	compiled SQL, artifacts

PROJECT FILES (cont.)

models/schema.yml

```
models:
  - name: orders
    columns:
      - name: id
        data_tests:
          - unique
          - not_null
```

JINJA IN MODELS

{{ ref('*<model>*') }}
Reference another model or seed

{{ source('*<src>*', '*<table>*') }}
Reference a raw table declared as a source

{{ config(materialized='table') }}
Set config at the top of a model

{{ var('*<name>*', '*<default>*') }}
Read a project variable

{{ this }}
The model's own relation

{% if is_incremental() %}
Filter only on incremental runs

RUN AND BUILD

dbt build
Seed, run, snapshot and test in DAG order

dbt run
Build all models

dbt test
Run data and unit tests

dbt seed
Load seeds/*.csv into tables

dbt snapshot
Record changes in snapshot tables

dbt source freshness
Check sources against freshness rules

dbt compile -s *<model>*
Print the rendered SQL of a model

dbt show -s *<model>* **--limit** *<n>*
Preview a model's rows

dbt show --inline "*<sql>***"**
Run an ad-hoc query with Jinja

RUN AND BUILD (cont.)

dbt ls -s *<selector>*
List what a selection matches

dbt retry
Rerun only what failed last time

dbt docs generate
Build catalog and docs site

dbt docs serve --port *<port>*
Serve the docs locally

SELECTION

dbt run -s *<model>*
Select one model (--select)

dbt run -s *<model>*+
Model and everything downstream

dbt run -s +*<model>*
Model and everything upstream

dbt run -s +*<model>*+
Upstream, model and downstream

dbt run -s 1+*<model>*
Model and direct parents only

dbt run -s "*<a>* *<b>***"**
Union: either selection

dbt run --exclude *<model>*
Leave nodes out

dbt run -s tag:*<tag>*
By tag

dbt run -s path:models/*<dir>*
By folder

COMMON FLAGS

-t *<target>*
Use another target from profiles.yml

--full-refresh
Rebuild incremental models and seeds

--vars '{*<key>***: *<value>*}'**
Set variables (YAML)

STATE AND CI

--state points at another run's target/ (its manifest.json); copy it aside first, since each run overwrites target/.

dbt build -s state:modified+ --state *<dir>*
Build changed nodes and children