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.

PROJECT

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

`dbt debug`
Check profile, project and connection

`dbt deps`
Install packages from packages.yml

`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

JINJA IN MODELS

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

JINJA IN MODELS (cont.)

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

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

`{% 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 compile -s <model>`
Print the rendered SQL of a model

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

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

`dbt retry`
Rerun only what failed last time

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 --exclude <model>`
Leave nodes out

COMMON FLAGS

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