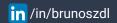


dbt deps

# dbt cheat sheet



#### Main commands

dbt build Builds and tests all selected resources (models, seeds, snapshots, tests)

Deletes all folders specified in the clean-targets list dbt clean specified in dbt\_project.yml, like target.

Compiles (but does not run) the models in a project

Clones selected nodes from the specified state to the

Downloads dependencies for a project

target schema(s)

dbt debug Debugs dbt connections and projects

dbt init Initializes a new dbt project (dbt Core only)

dbt list Lists resources defined in a dbt project

Parses a project and writes detailed timing info dbt parse

Retries the last run dbt command from the point of failure dbt retry

Runs the models in a project dbt run

Invokes a macro dbt run-operation

> Loads CSV files into the database dbt seed

Previews table rows post-transformation dbt show

dbt snapshot Executes "snapshot" jobs defined in a project

Provides tools for working with source data (including validating that sources are "fresh") dbt source

dbt test Executes tests defined in a project

## Main Flags

--debug Shows debug-level info in the terminal

--fail-fast Makes dbt exit immediately if a resource fails --full-refresh Causes table and seeds ito be recreated

--help Shows available commands and arguments

--log-level Limits the info in terminal to the set log level

--profiles-dir Sets the path for profiles.yml

-project\_dir Sets the path for dbt\_project.yml cord-timing-info Saves performance profiling information to a file

Makes dbt store failed rows in a table --store-failures

> Specifies the number of threads to use in the run --threads

Supplies variables to the project Shows dbt version

Makes warnings act like errors

### dbt docs

dbt docs generate Generates your projects' documentation website. Use the --no-compile argument to skip re-compilation

dbt docs serve Starts a webserver to serve your documentation locally. Use the --port to choose the port

### Node selection

# Syntax overview

command	argument(s)
run	select (-s),exclude,selector,defer
test	select (-s),exclude,selector,defer
seed	select (-s),exclude,selector
snapshot	select (-s),exclude,selector
list	select (-s),exclude,selector,resource-type
compile	select (-s),exclude,selector
freshness	select (-s),exclude,selector
build	select (-s),exclude,selector,resource-type,defer
docs generate	select (-s),exclude,selector

## Specifying resources

--version

The --select flag accepts one or more args. Each arg can be one of compatibility. A packages name

A model name

A fully-qualified path to a directory of models
A selection method (path:, tag:, config:, test\_type:, test\_name: etc Examples:

\$ dbt run --select my\_dbt\_project\_name a specific model

\$ dbt run --select my\_dbt\_model \$ dbt run --select path.to.my.models

all models in a specific directory a specific model in a specific \$ dbt run --select my\_package.some\_model

\$ dbt run --select tag:nightly \$ dbt run --select path/to/models

\$ dbt run --select path/to/my\_model.sql

models with the "nightly" tag

models contained in path/to/models

a specific model by its path

# **Excluding models**

dbt provides an --exclude flag with the same semantics as --select. Models specified with the --exclude flag will be removed from the set of models selected with --select.

\$ dbt run --select my\_package.\*+ --exclude my\_package.a\_big\_model+

# **Graph operators**

# Plus operator(+)

- \$ dbt run --select my model+
- \$ dbt run --select +my\_model \$ dbt run --select +my\_model+
- select my\_model and all children - select my\_model and all parents
  - select my\_model, and all of its parents and children

- select my\_model, its first-degree parents, and its

#### N-plus operator \$ dbt run --select my\_model+1

- \$ dbt run --select 2+mv model
- \$ dbt run --select 3+mv model+4
- At operator(@)
- \$ dbt run --models @my\_model Star operator(\*) \$ dbt run --select finance.base.\*
- second-degree parents ("grandparents") select my\_model, its parents up to the 3rd degree, and its children down to the 4th degree

- select my\_model and its first-degree children

- select my\_model, its children, and the parents of its - run all of the models in models/finance/base

# Methods examples

dbt run -s "tag:nightly" dbt run -s "source:snowplow+ dbt list -s "resource\_type:test" source resource\_type dbt run -s "models/staging/github" path package dbt run -s "package:snowplow dbt run -s "config.materialized:incremental" confia dbt test -s "test\_type:generic

test\_type dbt test -s "test\_name:unique"
dbt run -s "state:modified" --state path/to/artifacts test\_name state

dbt run -s "+exposure:weekly\_kpis"
dbt build -s "+metric:weekly\_active\_users exposure metric result dbt run -s "result:error" --state path/to/artifacts dbt build -s "source\_status:fres source\_status

dbt run -s "group:finance" dbt list -s "access:public" group access version dbt list -s "version:latest"
dbt list -s "unit\_test:\*" unit test

### Set operators

Unions (space-delineated)

\$\text{dbt run --select +snowplow\_sessions +fct\_orders} dbt run --select +my\_model}\$

Intersections (comma-separated) \$ dbt run --select +snowplow\_sessions,+fct\_orders

- sions, all ancestors of snowplow\_sessions, fct\_orders, and all ancestors of fct\_orders)
- run all the common ancestors of snowplow\_sessions and fct\_orders
- run models that are in the marts/finance subdirectory and tagged nightly

State

\$ dbt run --select 3+my\_model+4 \$ dbt run --select marts.finance,tag:nightly

state, like the state of a previous invocation or the state of the project in production.

Some methods require a manifest file to compare the current state of the project with another

#### Defer

Defer allows you build your project without having to build upstream resources. It requires a state. It is commonly used for Slim CI: dbt build -s "state"modified+"

The path of this manifest can be passed using the --state flag. odified--defer

source: https://docs.getdbt.com/