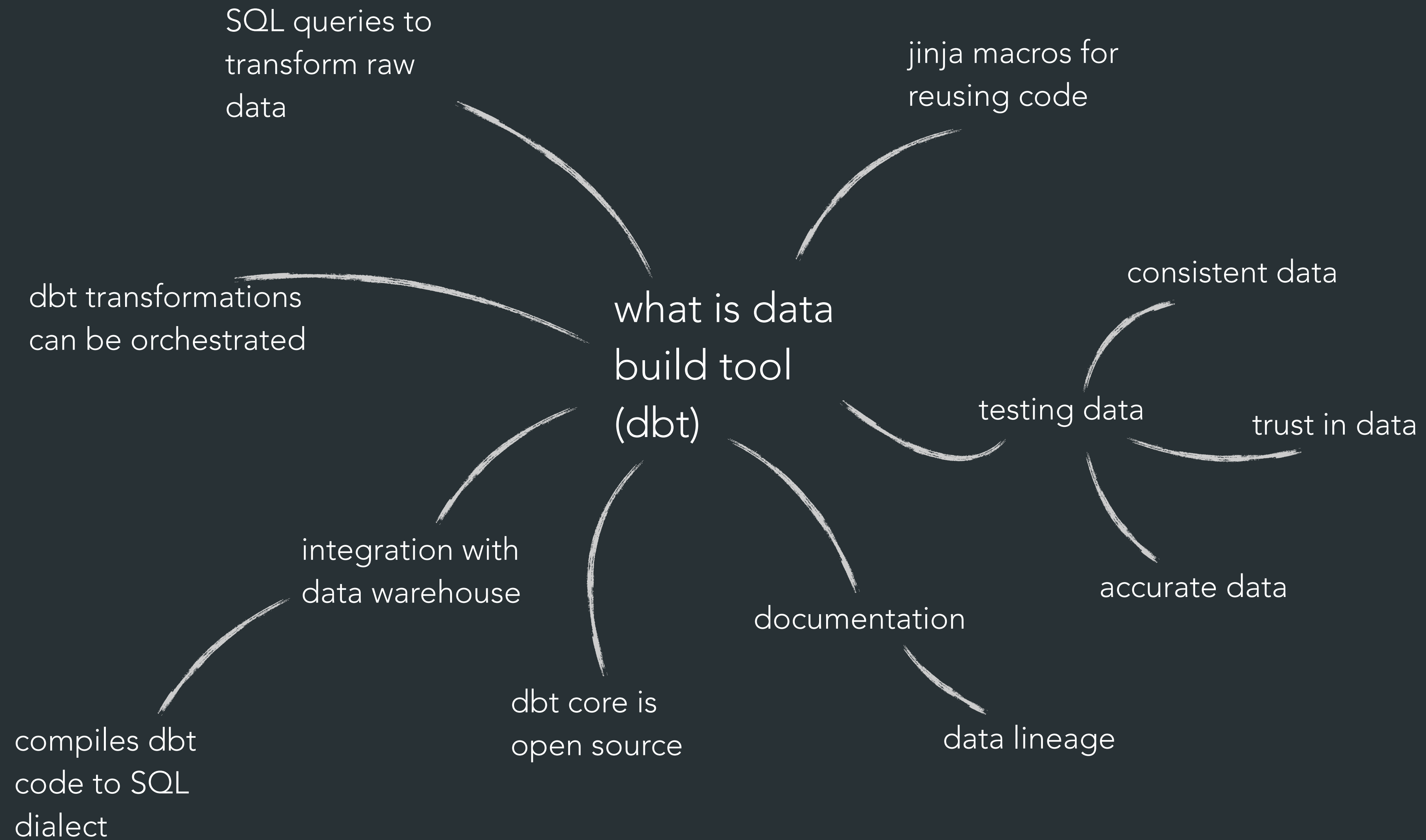


kokchun giang

using **dbt** to  
transform and refine  
the data into





# web-based autogenerated **documentation** to understand the data models

Search for models...

Overview

Project

Database

Group

Tables and Views

job\_ads

marts

mart\_job\_listings

staging

warehouse

dim\_employer

dim\_job\_details

fct\_job\_ads

Job ad project

This is the job ads projects documentation. Take a look at the dimensional model before moving on to lineage and the different models.

[!NOTE] Some of the dimensional model has not been implemented yet. Also the test suite should be made more comprehensive.

Dimensional model

dim\_job\_details

job_details_id	INTEGER
headline	STRING NN
description	STRING NN
description_html_formatted	STRING NN
employment_type	STRING NN
duration	STRING NN
salary_type	STRING NN
scope_of_work_min	NUMBER(3,0) NN
scope_of_work_max	NUMBER(3,0) NN

fct\_job\_ads

job_id	INTEGER
job_details_id	INTEGER
employer_id	INTEGER
auxilliary_attributes_id	INTEGER
number_vacancies	INTEGER NN
relevance	FLOAT NN
application_deadline	DATETIME NN

dim\_employer

employer_id	INTEGER
employer_name	STRING NN
employer_workplace	STRING NN
employer_organization_number	STRING NN
workplace_street_address	STRING NN
workplace_region	STRING NN
workplace_postcode	STRING NN
workplace_city	STRING NN
workplace_country	STRING NN

dim\_auxilliary\_attributes

auxilliary_attributes_id	INTEGER
--------------------------	---------

dim\_job\_details

fct\_job\_ads

dim\_employer

dim\_auxilliary\_attributes

dim\_job\_details

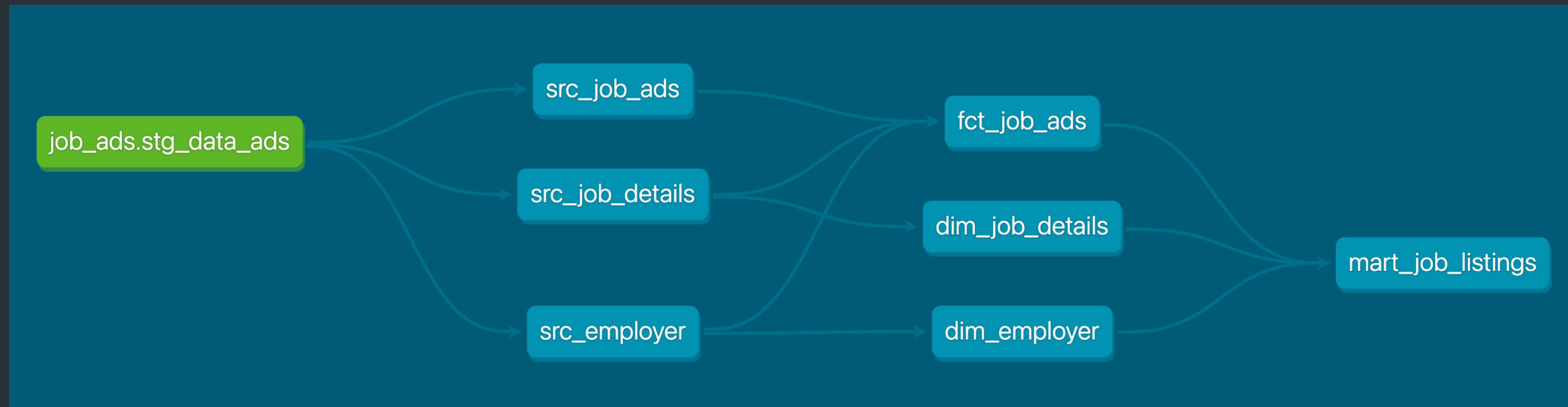
fct\_job\_ads

dim\_employer

dim\_auxilliary\_attributes



**data lineage** to visualize how data flows through various transformations



source code is **compiled** to snowflake code

jinja macros

Source Compiled

```
1 WITH ja AS (SELECT * FROM {{ ref('src_job_ads') }}),
2
3 jd AS (SELECT * FROM {{ ref('src_job_details') }}),
4
5 e AS (SELECT * FROM {{ ref('src_employer') }})
6
7 SELECT
8     {{dbt_utils.generate_surrogate_key(['jd.id', 'jd.headline'])}} AS job_details_key,
9     {{dbt_utils.generate_surrogate_key(['e.id', 'e.employer_name'])}} AS employer_key,
10    {# TODO: key to junk dimension #}
11    relevance, -- if null, aggregation functions can handle it properly
12    coalesce(vacancies, 1) as vacancies,
13    application_deadline
14    -- for verifying FK relationships work
15    {# jd.id,
16     e.workplace_city,
17     e.employer_name,
18     jd.description #}
19 FROM
20     ja
21 LEFT JOIN
22     jd ON ja.id = jd.id
23 LEFT JOIN
24     e ON ja.id = e.id
```

dbt code

Source Compiled

```
1 WITH __dbt__cte__src_job_ads as (
2
3
4
5 WITH stg_job_ads AS (SELECT * FROM job_ads.staging.data_field_job_ads
6 )
7
8 SELECT
9     id,
10    headline,
11    number_of_vacancies AS vacancies,
12    relevance,
13    application_deadline
14 FROM stg_job_ads ORDER BY application_deadline ASC
15 ), __dbt__cte__src_job_details as (
16
17
18 WITH stg_job_ads AS (SELECT * FROM job_ads.staging.data_field_job_ads
19 )
20
21 SELECT
22     id,
23    headline,
24    description__text AS "DESCRIPTION",
25    description_text_formatted AS description_html_formatted
```

compiled snowflake SQL

# file structure of a dbt project

