Data Warehouse Analyst

DBT: Продвинутые трансформации

otus.ru

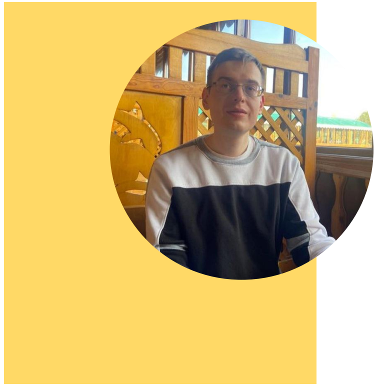
● REC **Проверить, идет ли запись**

Меня хорошо видно && слышно?

Ставим "+", если все хорошо
"-", если есть проблемы

DBT: Продвинутые трансформации

Андрей Поляков



В отрасли бэкенд-разработки на Java более 6 лет. Занимался fullstack-разработкой приложений, разработкой высоконагруженных compute-grid систем, а также микросервисов и etl-пайплайнов. Сейчас в роли старшего разработчика работаю над сервисами платежных систем в Unlimint.

Есть опыт работы с сервисами Hadoop (HDFS, HBase), оркестраторами (Airflow, Spring Cloud Data Flow), MPP-базами (Cassandra, Greenplum, Clickhouse).

Интересы: BigData, Blockchain, NFT

Образование: Master Degree in Computer Science and IT, ЮУрГУ, факультет ВШЭКН.

Преподает на курсах

- **Highload Architect**
- **Cloud Solution Architecture**
- **Архитектура и шаблоны проектирования**
- **Microservice Architecture**
- **Data Warehouse Analyst**
- **Data Engineer**
- **Java Developer. Basic**

Unlimint

Старший разработчик

3 года в Otus **261 занятие** **2259 студентов**

Правила вебинара

Активно участвуем

Off-topic обсуждаем в Telegram @DWH-2024-12

Задаем вопрос в чат или голосом

Вопросы вижу в чате, могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое на активность

Пишем в чат

Говорим голосом

Документ

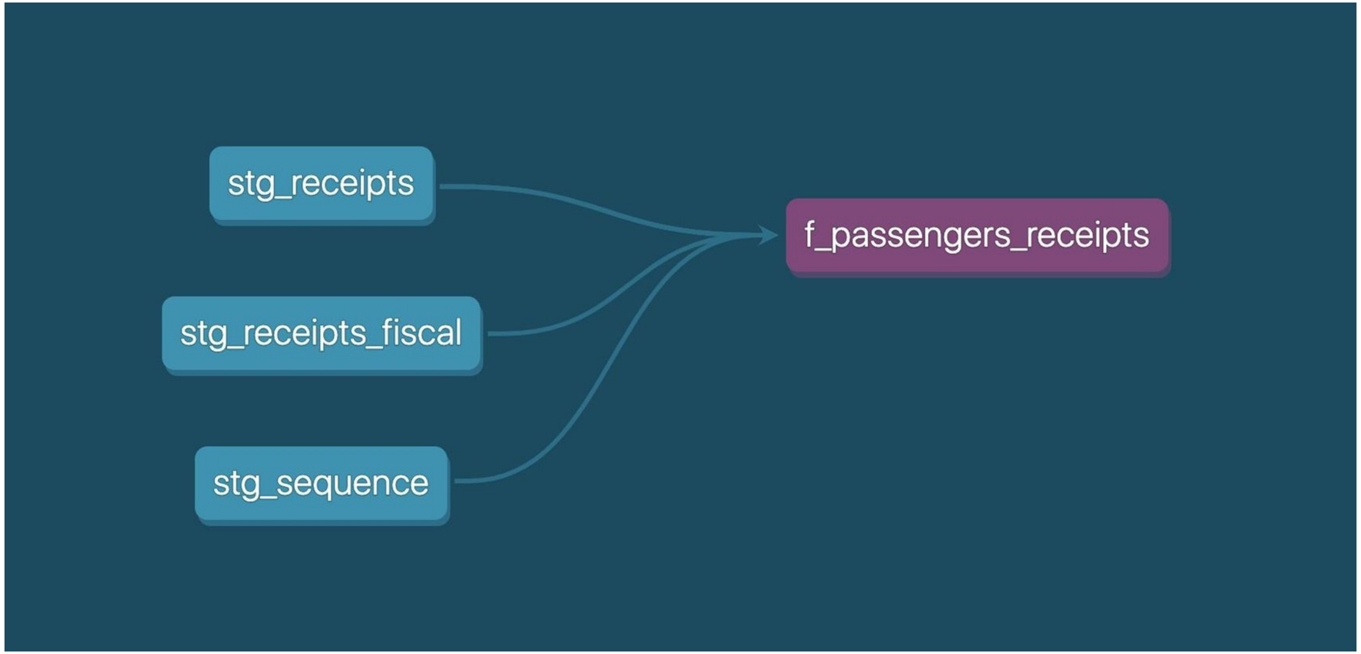
Ответьте себе или задайте вопрос

Цели вебинара

К концу занятия вы сможете

1. Применять DBT для формирования структуры DWH
2. Реализовывать трансформации данных с использованием расширений DBT
3. Делать снепшоты данных, генерировать документацию, тестировать данные и структуру хранилища

Графы исполнения моделей DAGs



DAG

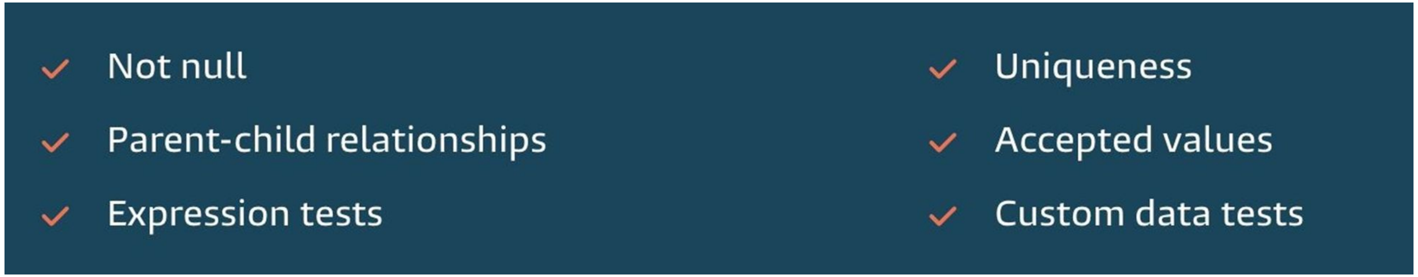
Полезные модули

`dbt_codegen`	Use the package to help you generate YML files for your models and sources and SQL files for your staging models.
`dbt_utils`	The package contains macros useful for daily development. For example, `date_spine` generates a table with all dates between the ones provided as parameters.
`dbt_project_evaluator`	The package compares your dbt project against a list of our best practices and provides suggestions and guidelines on how to update your models.
`dbt_expectations`	The package contains many tests beyond those built into dbt.

Тесты

```
{% test not_null(model, column_name) %}

    select *
    from {{ model }}
    where {{ column_name }} is null

{% endtest %}
```

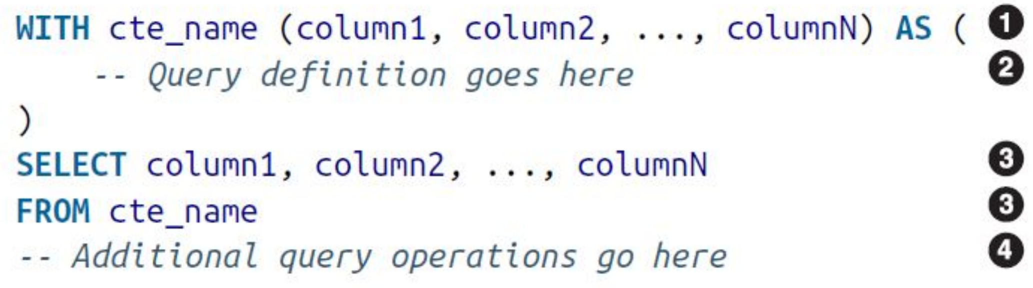
Schema testing



- ✓ Not null
- ✓ Parent-child relationships
- ✓ Expression tests
- ✓ Uniqueness
- ✓ Accepted values
- ✓ Custom data tests

```yaml
version: 2
models:
- name: my_model
  tests:
  - not_null_columns:
    columns:
    - column1
    - column2
```

CTE



```
WITH cte_name (column1, column2, ..., columnN) AS (  ❶
    -- Query definition goes here                     ❷
)
SELECT column1, column2, ..., columnN                 ❸
FROM cte_name                                         ❸
-- Additional query operations go here                ❹
```

Что такое CTE? Для чего они нужны?

Сроки выполнения: 1 мин (пишем в чат)

CTE

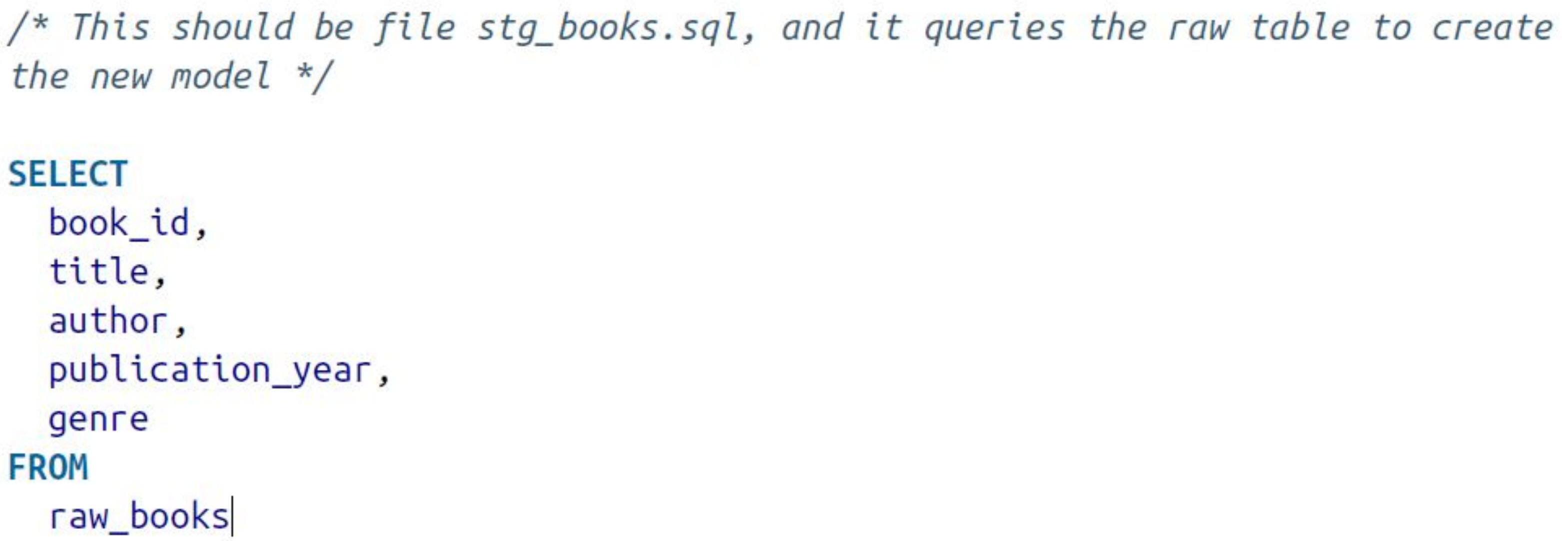
Без CTE

```sql
SELECT pb.book_id,
       pb.title,
       pb.author,
       s.total_sales
FROM (
    SELECT book_id,
           title,
           author
    FROM books
    WHERE rating >= 4.6
) AS pb
JOIN sales s ON pb.book_id = s.book_id
WHERE s.year = 2022
ORDER BY s.total_sales DESC
LIMIT 5;
```

С CTE

```sql
WITH popular_books AS (
    SELECT book_id,
           title,
           author
    FROM books
    WHERE rating >= 4.6
),
best_sellers AS (
    SELECT pb.book_id,
           pb.title,
           pb.author,
           s.total_sales
    FROM popular_books pb
    JOIN sales s ON pb.book_id = s.book_id
    WHERE s.year = 2022
    ORDER BY s.total_sales DESC
    LIMIT 5
)
SELECT *
FROM best_sellers;
```

Модели Stage



```sql
/* This should be file stg_books.sql, and it queries the raw table to create
the new model */

SELECT
  book_id,
  title,
  author,
  publication_year,
  genre
FROM
  raw_books
```

Модели Intermediate

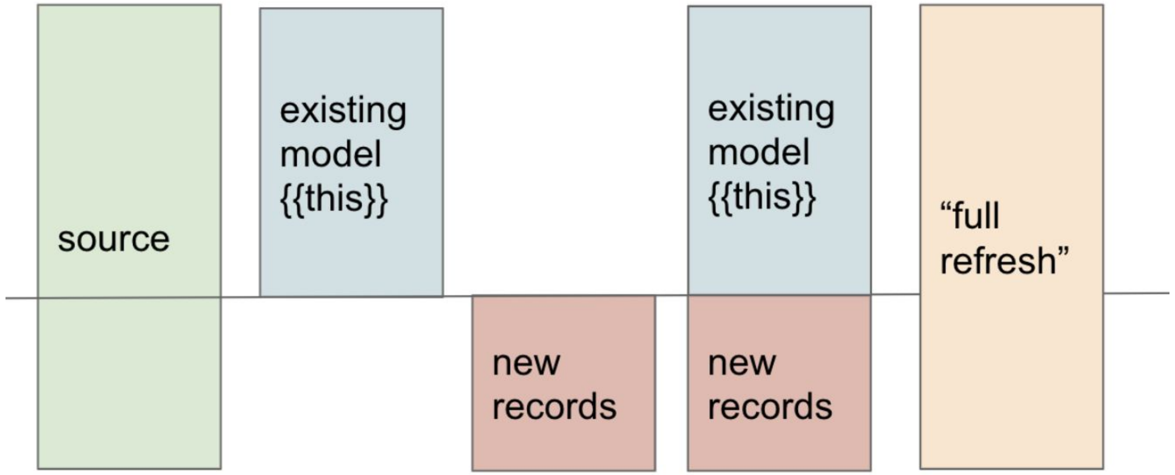
```sql
-- This should be file int_book_authors.sql

-- Reference the staging models
WITH
  books AS (
    SELECT *
    FROM {{ ref('stg_books') }}
  ),
  authors AS (
    SELECT *
    FROM {{ ref('stg_authors') }}
  )

-- Combine the relevant information
SELECT
  b.book_id,
  b.title,
  a.author_id,
  a.author_name
FROM
  books b
JOIN
  authors a ON b.author_id = a.author_id
```

Модели Mart

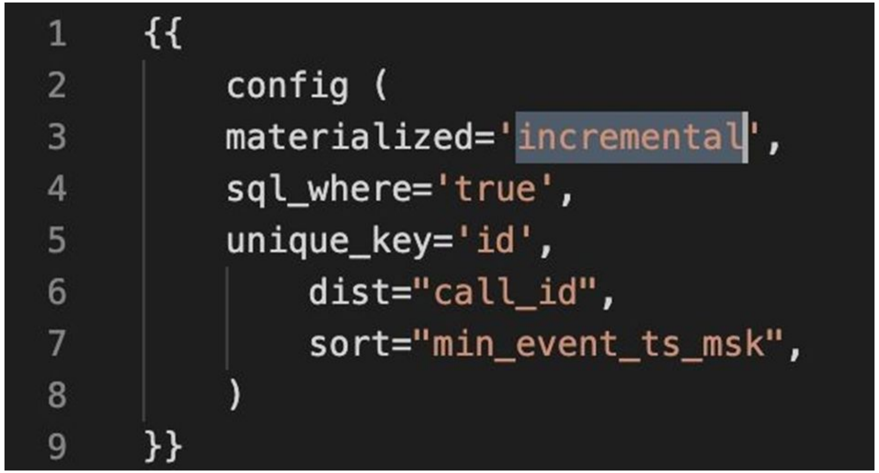
```
-- This should be file mart_book_authors.sql

{{
  config(
    materialized='table',
    unique_key='author_id',
    sort='author_id'
  )
}}

WITH book_counts AS (
  SELECT
    author_id,
    COUNT(*) AS total_books
  FROM {{ ref('int_book_authors') }}
  GROUP BY author_id
)
SELECT
  author_id,
  total_books
FROM book_counts
```

Инкрементальные модели

https://docs.getdbt.com/docs/build/incremental-strategy



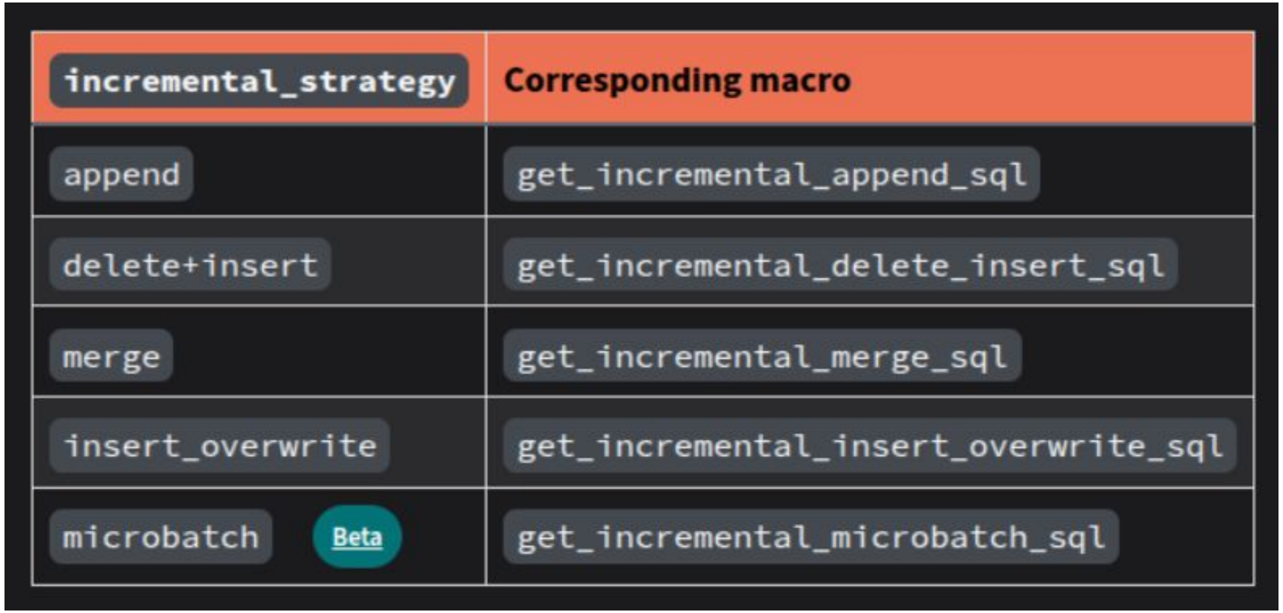
Полная и инкрементальная загрузка



```
{{
    config (
    materialized='incremental',
    sql_where='true',
    unique_key='id',
        dist="call_id",
        sort="min_event_ts_msk",
    )
}}
```

Инкрементальные модели

https://docs.getdbt.com/docs/build/incremental-strategy



incremental_strategy	Corresponding macro
append	get_incremental_append_sql
delete+insert	get_incremental_delete_insert_sql
merge	get_incremental_merge_sql
insert_overwrite	get_incremental_insert_overwrite_sql
microbatch Beta	get_incremental_microbatch_sql

Graph operators

https://docs.getdbt.com/reference/node-selection/graph-operators

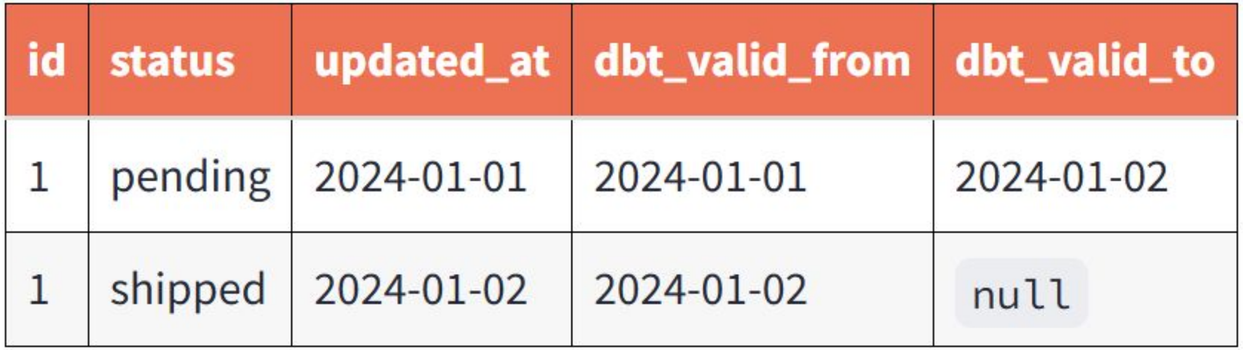
```
dbt run --select "my_model+"        # select my_model and all descendants
dbt run --select "+my_model"        # select my_model and all ancestors
dbt run --select "+my_model+"        # select my_model, and all of its ancestors and descendants
```

Аналитика

https://github.com/dbt-labs/quickbooks

Снепшоты

https://en.wikipedia.org/wiki/Slowly_changing_dimension#Type_2:_add_new_row



id	status	updated_at	dbt_valid_from	dbt_valid_to
1	pending	2024-01-01	2024-01-01	2024-01-02
1	shipped	2024-01-02	2024-01-02	`null`

Снепшоты

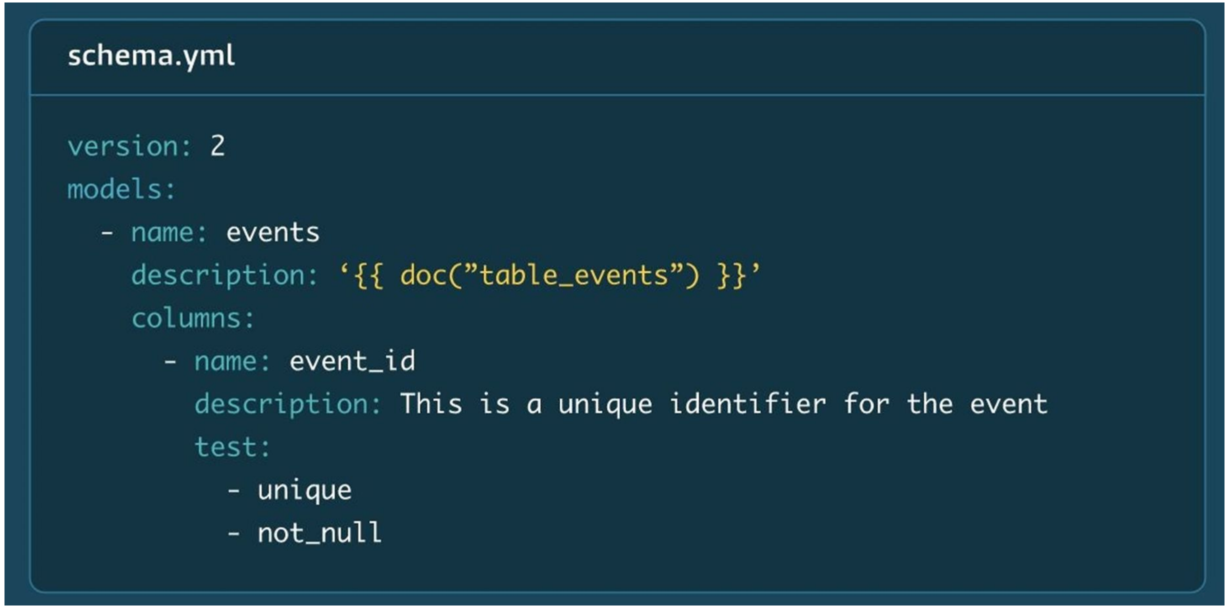
```
{% snapshot orders_snapshot_timestamp %}

    {{
        config(
          target_schema='snapshots',
          strategy='timestamp',
          unique_key='id',
          updated_at='updated_at',
        )
    }}

    select * from {{ source('jaffle_shop', 'orders') }}

{% endsnapshot %}
```

Снепшоты

```
{% snapshot orders_snapshot_check %}

    {{
        config(
          target_schema='snapshots',
          strategy='check',
          unique_key='id',
          check_cols=['status', 'is_cancelled'],
        )
    }}

    select * from {{ source('jaffle_shop', 'orders') }}

{% endsnapshot %}
```

Документирование

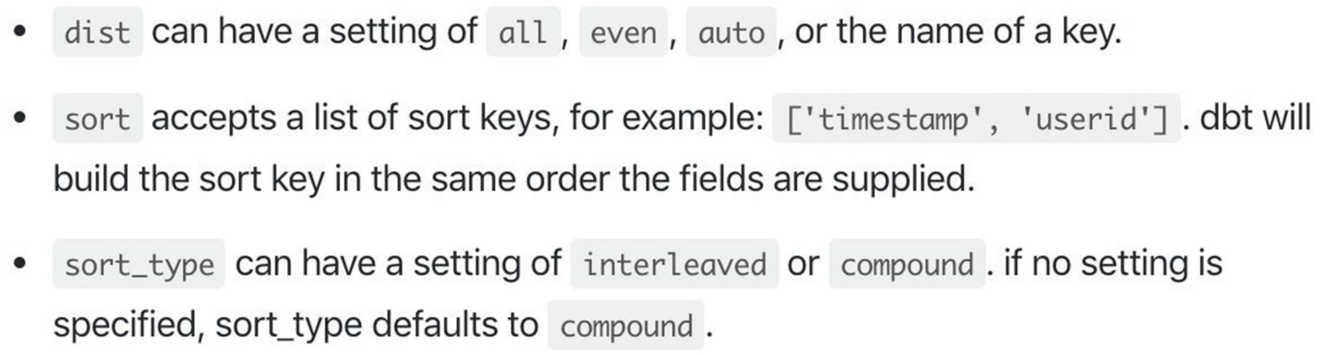
dbt docs generate
dbt docs serve
# localhost:8080



schema.yml

```
version: 2
models:
  - name: events
    description: '{{ doc("table_events") }}'
    columns:
      - name: event_id
        description: This is a unique identifier for the event
        test:
          - unique
          - not_null
```

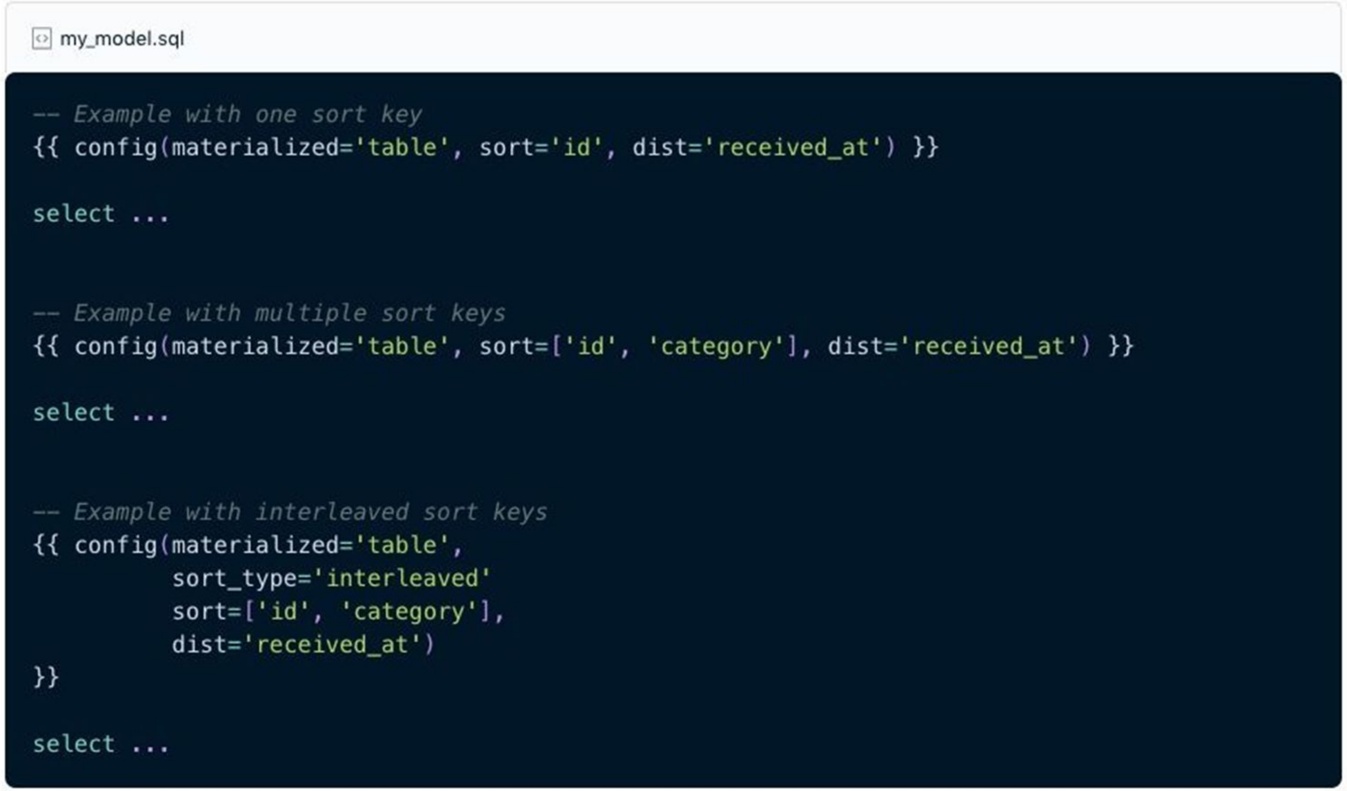
Database specific configurations



- `dist` can have a setting of `all`, `even`, `auto`, or the name of a key.
- `sort` accepts a list of sort keys, for example: `['timestamp', 'userid']`. dbt will build the sort key in the same order the fields are supplied.
- `sort_type` can have a setting of `interleaved` or `compound`. if no setting is specified, sort_type defaults to `compound`.

https://docs.getdbt.com/reference/dbt-jinja-functions/adapter

Physical model optimization (Redshift)


my_model.sql

```
-- Example with one sort key
{{ config(materialized='table', sort='id', dist='received_at') }}

select ...


-- Example with multiple sort keys
{{ config(materialized='table', sort=['id', 'category'], dist='received_at') }}

select ...


-- Example with interleaved sort keys
{{ config(materialized='table',
          sort_type='interleaved'
          sort=['id', 'category'],
          dist='received_at')
}}

select ...
```

LIVE DEMO

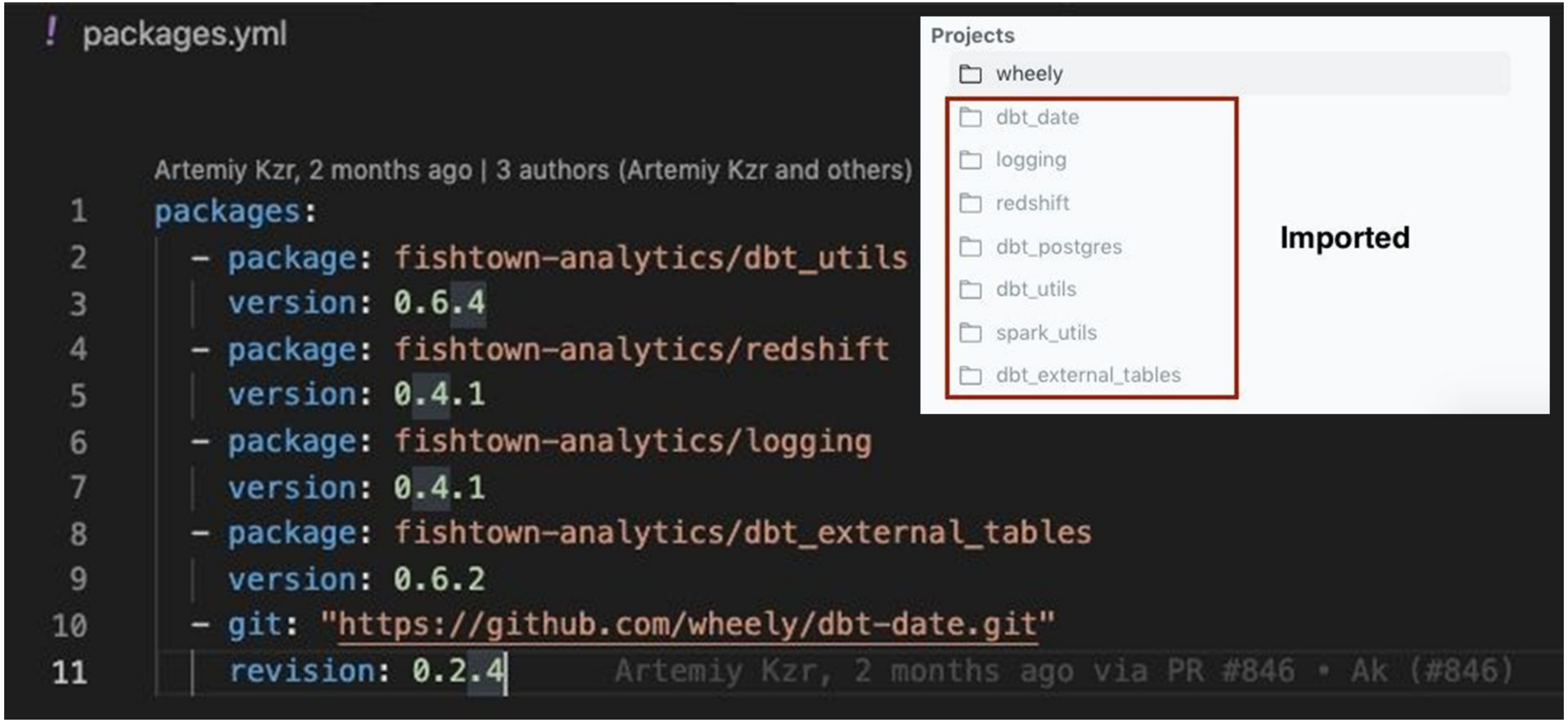
Extensibility – модульная структура

Importing modules – dbt utilities

dbt_utils

Created by fishtown-analytics

Importing modules allows reusing code

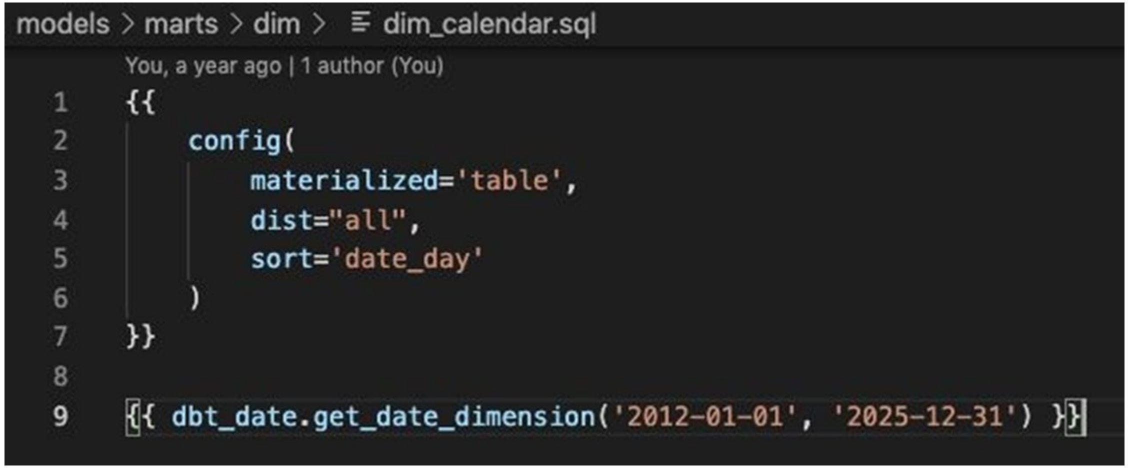


packages.yml

Artemiy Kzr, 2 months ago | 3 authors (Artemiy Kzr and others)

```yaml
packages:
  - package: fishtown-analytics/dbt_utils
    version: 0.6.4
  - package: fishtown-analytics/redshift
    version: 0.4.1
  - package: fishtown-analytics/logging
    version: 0.4.1
  - package: fishtown-analytics/dbt_external_tables
    version: 0.6.2
  - git: "https://github.com/wheely/dbt-date.git"
    revision: 0.2.4      Artemiy Kzr, 2 months ago via PR #846 • Ak (#846)
```

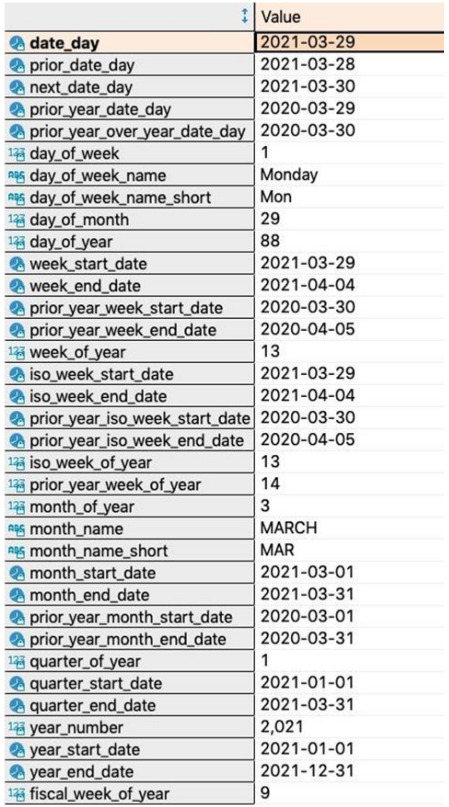
Generating calendar in one line



models > marts > dim > dim_calendar.sql

You, a year ago | 1 author (You)

```
{{
    config(
        materialized='table',
        dist="all",
        sort='date_day'
    )
}}

{{ dbt_date.get_date_dimension('2012-01-01', '2025-12-31') }}
```



	Value
date_day	2021-03-29
prior_date_day	2021-03-28
next_date_day	2021-03-30
prior_year_date_day	2020-03-29
prior_year_over_year_date_day	2020-03-30
day_of_week	1
day_of_week_name	Monday
day_of_week_name_short	Mon
day_of_month	29
day_of_year	88
week_start_date	2021-03-29
week_end_date	2021-04-04
prior_year_week_start_date	2020-03-30
prior_year_week_end_date	2020-04-05
week_of_year	13
iso_week_start_date	2021-03-29
iso_week_end_date	2021-04-04
prior_year_iso_week_start_date	2020-03-30
prior_year_iso_week_end_date	2020-04-05
iso_week_of_year	13
prior_year_week_of_year	14
month_of_year	3
month_name	MARCH
month_name_short	MAR
month_start_date	2021-03-01
month_end_date	2021-03-31
prior_year_month_start_date	2020-03-01
prior_year_month_end_date	2020-03-31
quarter_of_year	1
quarter_start_date	2021-01-01
quarter_end_date	2021-03-31
year_number	2,021
year_start_date	2021-01-01
year_end_date	2021-12-31
fiscal_week_of_year	9

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Рефлексия

Список материалов для изучения

1. dbt Getting Started Tutorial
2. dbt Documentation
3. dbt FAQ
4. How we structure our dbt projects
5. The Modern Data Stack: Past, Present, and Future
6. Five principles that will keep your data warehouse organized
7. The Analytics Engineering Guide

Делитесь своими материалами в Slack

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!