


## DBT: data build tool

### Contents

DBT (Data Build Tool) Là Gì? Những Thứ Cơ Bản Về DBT_May 2023 .....	2
I. DBT là gì? .....	2
Tại sao lại là DBT? .....	3
Ưu điểm của DBT: .....	4
II. Các điểm nổi bật của dbt .....	4
1. Rút gọn boilerplate (đoạn code mang tính hình thức) .....	4
2. Hỗ trợ đa dạng Data Platform - Data Warehouse .....	5
3. Models hóa abstraction và dependency .....	6
4. Data Lineage và documentation .....	6
5. Hỗ trợ Jinja template .....	7
6. Test tự động .....	7
7. Có bản Cloud .....	7
Tạo Data Pipeline bằng DBT - Cách Dùng DBT Phần 2 (Vịt data)_Feb 2023 .....	7
Cách Dùng DBT - Phần 1 Tạo Bảng (Vịt data)_Jan 2023 .....	8
Hướng Dẫn Cài Đặt DBT (Vịt data)_Dec 2022 .....	8
Làm Data Pipeline cho data warehouse bằng DBT SIÊU TIỆN 😊 (Vịt data)_Sep 2022 .....	8
DBT (Data Build Tool) là gì – Jul 2022 .....	8
DBT là gì có ăn được không? .....	8
DBT có gì đặc biệt .....	8
Điểm khác biệt cốt lõi của DBT .....	8
Xóa boilerplate là như thế nào .....	9
Thủ công .....	9
DBT .....	9
Một số tính năng của DBT .....	10
1. Transform data without boilerplate .....	10
2. Models ( hoặc tạm hiểu là table cũng được ) abstraction and dependency .....	10
3. Data Lineage .....	11
4. Jinja Support .....	11
5. Data Documentation support .....	11
6. Automated Testing .....	11
Tải DBT ở đâu .....	11

Data build tool



dbt is an open-source command line tool that helps analysts and engineers transform data in their warehouse more effectively. [Wikipedia](#)

**Developer(s):** dbt-Labs

**Available in:** [Python](#)

**Initial release:** December 3, 2021; 23 months ago

**License:** [Apache License 2.0](#)

**Operating system:** [Microsoft Windows](#), [macOS](#), [Linux](#)

**Repository:** [github.com/dbt-labs/dbt-core](https://github.com/dbt-labs/dbt-core)

**Stable release:** 1.6.5 / October 2, 2023; 54 days ago

## DBT (Data Build Tool) Là Gì? Những Thứ Cơ Bản Về DBT\_May 2023

<https://viblo.asia/p/DBT-data-build-tool-la-gi-nhung-thu-co-ban-ve-DBT-ok9VyxbxLQR>



Các bạn DE, **Data Analyst** (DA) khi tìm search từ khóa "**DBT**" trên Google thường sẽ ra nhiều kết quả khác nhau về những cụm từ có viết tắt là "DBT", và không may thay là cái mà DE, DA cần lại thường không được suggest đầu tiên. Cái mình nhắc đến hôm nay là **Data Build Tool**, một công cụ mã nguồn mở hỗ trợ chủ yếu việc **transform data**. Dạo gần đây thấy nó nổi lên khá là mạnh và được sử dụng tương đối nhiều.

### I. DBT là gì?

# What is dbt?

dbt™ is a SQL-first transformation workflow that lets teams quickly and collaboratively deploy analytics code following software engineering best practices like modularity, portability, CI/CD, and documentation. Now anyone on the data team can safely contribute to production-grade data pipelines.

Đối với các bạn **DA**, muốn xây dựng một BI Model, việc module hóa quá trình **transform data** là thực sự cần thiết. Có rất nhiều tool hỗ trợ transform mạnh mẽ có thể kể đến như Pandas (Python), Apache Spark, R, Apache Nifi, ... Tuy nhiên để có thể dễ dàng sử dụng cho các bạn đã biết SQL thì DBT là một lựa chọn

rất tốt vì nó hỗ trợ module hóa các câu lệnh SQL, vậy nên các bạn chỉ cần có nền tảng SQL có thể dễ dàng sử dụng.

Tóm lại, **DBT** (Data Build Tool), là công cụ **hỗ trợ quá trình transform data** trở nên nhanh và trở nên dễ dàng hơn.

### Tại sao lại là DBT?

Một vài câu hỏi mà chúng ta thường gặp là:

- Vì sao lại phải sử dụng DBT, trong khi đang sử dụng SQL ngon lành mà?
- Ủa, sao kỳ vậy? Đang quen thuộc với SQL tự dưng cái phải học thêm DBT ha?

Thì vậy nè, nhìn cái tên mình cũng thấy sượng sượng đó, DBT được thiết kế để giải quyết phần **T** (**transform**) trong ELT, nó sẽ làm với raw data trong quá trình Transform. DBT cung cấp ít chức năng trong quá trình transform hơn các cái OSS(open-source) ETL tool khác như Airflow, Luigi,... nhưng bù lại nó được cái là dễ hiểu, dễ sử dụng cho mấy non-engineer.



(nguồn: <https://viblo.asia/p/DBT-la-gi-may-thu-nho-nhat-minh-lum-lat-ve-DBT-5pPLkjOyJRZ>)

Như đã nói, nó là một tool open-source được xây dựng và phát triển bởi [RJMetrics](#) vào năm 2016. Mục đích ban đầu và cũng là mục đích chính của DBT là **transform** data. Data build tool sử dụng **SQL** đó đó quá trình transform trở nên nhanh và dễ dàng hơn.

Điều làm DBT trở nên đặc biệt là DBT có thể giúp một Analyst bình thường có thể thực hiện được những công việc cơ bản của Data Engineer (chủ yếu là transform - biến đổi dữ liệu). DBT giúp việc transform, document, test data trở nên dễ dàng hơn và có thể nhân rộng được. Thực hiện các điều trên cũng trở nên đơn giản hơn thông qua việc sử dụng các công cụ của DBT chứ bạn không cần phải set up hệ thống test và viết document tách biệt.



Nguồn: <https://www.getdbt.com/product/what-is-dbt/>

### ***Ưu điểm của DBT:***

Mấy năm gần đây thì **DW** (Data Warehouse) nó mạnh mẽ và linh hoạt, nó có thể tách biệt phần lưu trữ và phần xử lý, elastic scale (kiểu linh hoạt trong việc scale) và áp dụng Machine Learning. Xong cái nhiều công ty sử dụng, mà nhiều công ty sử dụng dẫn tới nhu cầu, càng dễ sử dụng càng tốt.

Nói chung về lợi ích thì khá nhiều, chủ yếu là dễ học, dễ sử dụng, dễ dàng sử dụng lại (dễ dàng chuyển đổi giữa các loại SQL với nhau, ví dụ đang sử dụng SQL, cái muốn đổi sang BigQuery thì cũng k mất nhiều công sức), đơn giản hóa cái việc coding của mình (ví dụ như SQL cần create table, insert data into table đó thì DBT chỉ cần viết select là nó tự tạo model, tự insert)

## ***II. Các điểm nổi bật của dbt***

Lý do dbt được lựa chọn trong việc transform dữ liệu cho cả những người mới và trong cả các công ty là bởi dbt có rất nhiều tính năng hay ho. Dưới đây mình sẽ liệt kê một vài tính năng và công cụ rất tiện lợi mà dbt cung cấp:

### ***1. Rút gọn boilerplate (đoạn code mang tính hình thức)***

Mỗi DBMS (hệ quản trị cơ sở dữ liệu) sẽ sử dụng một loại SQL khác nhau. Ví dụ như Oracle sử dụng PL-SQL trong khi Microsoft SQL Server sử dụng T-SQL, .... Cơ bản thì nó vẫn là SQL tuy nhiên mỗi loại lại có những cú pháp, cách sử dụng có phần khác nhau đôi chút. Kiểu như có loại sẽ dùng LIMIT, có loại dùng

TOP, có loại dùng kiểu STRING có loại lại là VARCHAR, db có gốc Java thì có kiểu DOUBLE còn mấy cái khác thì không v.v....

Cũng vì lý do này mà không mấy ai thích việc phải viết và define từng table, column hay insert data vào database trực tiếp bằng SQL (thậm chí developer còn tránh làm điều này mà dùng ORM để thay thế 🤖).

DBT sinh ra để một phần cải thiện vấn đề này. Nó giúp **lược bỏ** những đoạn **code mang tính hình thức**.

Điều này có thể giúp bạn thao tác tốt hơn với các table với số lượng lớn và phụ thuộc vào nhau.

Ví dụ, bạn có một table tên `customer`, bạn muốn tạo một bảng mới tên là `person` với một số cột từ bảng `customer`.

Với **SQL bình thường** thì phải làm như sau:

```
DROP TABLE IF EXISTS person;

CREATE TABLE person AS (
  SELECT ID, FirstName, LastName, City, Address FROM customer;
)
```

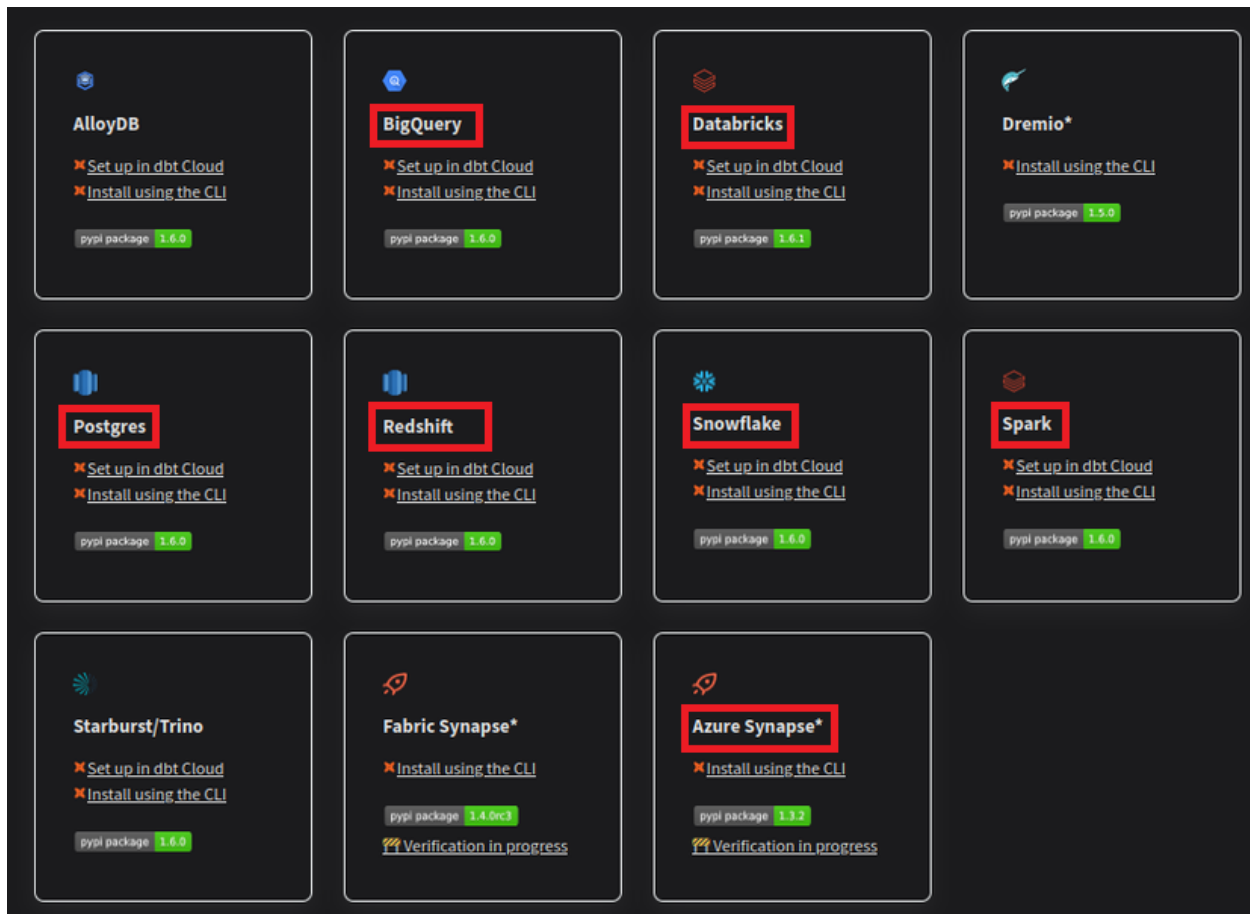
Còn với dbt, bạn chỉ cần viết:

```
SELECT ID, FirstName, LastName, City, Address FROM customer
```

Và lưu lại với file tên `person.sql`, đơn giản và nhanh gọn là bạn đã có table tên `person`, đầy đủ data y hệt như khi viết SQL bình thường. Ở một ví dụ nhỏ như thế này chúng ta có thể thấy sự khác biệt không nhiều, nhưng trên thực tế rất có ích với quy mô hàng trăm tables cùng 1 lúc và phụ thuộc vào nhau.

## 2. Hỗ trợ đa dạng Data Platform - Data Warehouse

DBT hỗ trợ kết nối và làm việc cùng với rất đa dạng các Data Platform khác nhau. Dưới đây là một số những data platform phổ biến mà dbt có thể kết nối và làm việc.



Nguồn: <https://docs.getdbt.com/docs/supported-data-platforms>

Đối với mỗi data platform mà bạn muốn làm việc, cần cài đặt 2 thư viện là `dbt-core` và thư viện tương ứng với platform đó.

Ví dụ, nếu muốn làm việc với **Google BigQuery** thì cài `dbt-bigquery` nữa là được

```
pip install dbt-core dbt-bigquery
```

### 3. Models hóa abstraction và dependency

Các table trong một database có một abstract dependency (tạm gọi là sự lệ thuộc ảo) với nhau. Khi chạy SQL thông qua dbt, thì các câu SQL sẽ được chạy theo tuần tự như dependency đã được khởi tạo trong dbt-models.

### 4. Data Lineage và documentation

Thêm nữa, dbt còn cung cấp công cụ document, khi bạn tạo ra dependency, dbt sẽ tự động tạo ra các tài liệu để biểu thị sự phụ thuộc giữa các model. Bạn sẽ không cần phải tự ghi tài liệu mà vẫn có tài liệu để nắm rõ nguồn gốc và mối liên hệ của data. Khá là tiện lợi!!!

Ngoài việc có đồ thị cho data lineage, dbt còn hỗ trợ tự động tạo tài liệu cho data. Bạn có thể ghi lại ý nghĩa của từng column, table do bạn tạo ra (tương tự Metadata đã nhắc tới trong [Dagster](#), tránh trường hợp sau này quên mất chúng là gì).

## 5. Hỗ trợ Jinja template

Jinja là một template engine rất nổi tiếng, ai làm Front-End web chắc hẳn đã nghe qua và sử dụng. Trong trường hợp của dbt, nó sẽ giúp bạn thu gọn những câu lệnh SQL rất dài thành ngắn lại.

```
select *  
from {{ ref('my_first_dbt_model') }}  
where id = 1
```

Ví dụ như sau đây:

Ở đây thay vì các bạn phải ghi chi tiết từ database đến table .... để kết nối tới table `my_first_dbt_model` thì các bạn chỉ cần ghi ngắn gọn lại như vậy và dbt sẽ xử lý đầy đủ.

## 6. Test tự động

DBT có rất nhiều loại test tự động khác nhau được soạn sẵn, hoặc bạn có thể tự viết bằng dbt macros. Như vậy bạn có thể chuẩn hóa quy trình bằng cách yêu cầu tất cả models pass các test được đề ra mà không cần phải check thủ công từng table.

```
version: 2  
  
models:  
  - name: my_first_dbt_model  
    description: "A starter dbt model"  
    columns:  
      - name: id  
        description: "The primary key for this table"  
        tests:  
          - unique  
          - not_null
```

## 7. Có bản Cloud

DBT có 2 bản Cloud và CLI, bản Cloud có giao diện rất đẹp cùng nhiều tính năng tiện lợi hay hơn bản CLI nhiều, và đương nhiên là mất phí rồi 😊))

### Lời kết

Cơ bản thì không có một công cụ nào là hoàn hảo về mọi mặt. DBT cũng chỉ là một lựa chọn trong số rất nhiều công cụ khác. Trong một dự án phải sử dụng kết hợp nhiều công cụ mới có thể tối ưu điểm mạnh. Tuy nhiên DBT cũng là một công cụ khá hay ho mà các bạn nên trải nghiệm vì một phần nó không khó để bắt đầu và cũng vì nó đang dần trở nên phổ biến thời gian gần đây. Bài viết mình tổng hợp từ vài nguồn và hiểu biết, thiếu sót mong mọi người góp ý. Cảm ơn mọi người đã đọc!

### Reference

<https://tuananalytic.com/dbt-data-build-tool-la-gi/>

<https://www.getdbt.com/>

<https://www.getdbt.com/product/what-is-dbt/>

**Tạo Data Pipeline bằng DBT - Cách Dùng DBT Phần 2 (Vịt data)\_Feb 2023**

<https://www.youtube.com/watch?v=Dm63EVrzJC0>

### **Cách Dùng DBT - Phần 1 Tạo Bảng (Vịt data)\_Jan 2023**

<https://www.youtube.com/watch?v=qkGJMh61dNI>

### **Hướng Dẫn Cài Đặt DBT (Vịt data)\_Dec 2022**

[https://www.youtube.com/watch?v=CF4p1d20D\\_s](https://www.youtube.com/watch?v=CF4p1d20D_s)

### **Làm Data Pipeline cho data warehouse bằng DBT SIÊU TIỆN 😊 (Vịt data)\_Sep 2022**

<https://www.youtube.com/watch?v=mlvj3VuCbTA>

### **DBT (Data Build Tool) là gì – Jul 2022**

Link: [tuananalytic.com](https://tuananalytic.com)

Dạo gần đây mình để ý có một khái niệm cứ bị google suggest **không đúng** khi data engineer, hoặc data analyst search trên google. Khi bạn search DBT sẽ ra kết quả về Dialectical Behavior Therapy. Bạn vào tìm hiểu và chả thấy tool nào cả! Mình quyết định viết bài viết này để giúp đỡ những người đang search về nó mà **bị lạc đường trên google**.

#### **DBT là gì có ăn được không?**

DBT là viết tắt cho Data Build Tool (Không phải là Dialectical Behavior Therapy google suggest).

DBT là công cụ được công ty **RJMetrics** tại Philadelphia, Mỹ phát triển và open-source vào năm 2016.

Mục đích ban đầu chủ yếu là để hỗ trợ thêm cho việc **transform data nội bộ**.

Sau nhiều năm được sử dụng bởi các công ty. DBT hiện nay là công cụ hỗ trợ việc transform data bằng SQL phổ biến nhất hiện nay.

#### **DBT có gì đặc biệt**

Điều đặc biệt của DBT chính là việc DBT có thể giúp một analyst bình thường có thể **thực hiện được công việc của một data engineer**. Thậm chí có thể giúp các analyst này phát triển các model phức tạp và sát với nhu cầu business mà không cần phải sử dụng nguồn lực (thường là luôn bị quá tải) của các data engineer.

DBT giúp việc transform, document, test data trở nên dễ dàng hơn và có thể nhân rộng được. Thực hiện các tác vụ trên cũng trở nên đơn giản hơn thông qua việc sử dụng các công cụ của DBT chứ bạn không cần phải set up hệ thống test và viết document tách biệt.

#### **Điểm khác biệt cốt lõi của DBT**

Thực ra có nhiều tool khác nhau cũng có thể làm điều tương tự như DBT, vậy điểm khác biệt cốt lõi ở đâu? Điểm khác biệt cốt lõi nhất của DBT đó chính là khả năng xóa bớt boilerplate (tạm vậy ch là **code mang tính hình thức**) khỏi SQL.



## Xóa boilerplate là như thế nào

Như bạn đã biết SQL là ngôn ngữ của database nên đôi lúc sẽ có rất nhiều tính năng “không chuẩn” so với Standard SQL. Để hiểu rõ thêm điều này hãy đọc bài viết tạo sao có lỗi SQL của mình.

Câu lệnh được sử dụng nhiều nhất trong SQL chính là **SELECT** thần thánh. Đối với một data analyst mọi chuyện sẽ bắt đầu sau chữ **SELECT** (vì mục đích là query dữ liệu ra mà). Tuy nhiên ngoài **SELECT** ra thì ở các loại SQL khác nhau lại có các loại code khác nhau để define table, có datatype khác nhau, có SQL thì gọi là string có bên thì gọi là varchar, các db có gốc java thì hỗ trợ double các db khác thì không, v.v.

Do đó, không mấy ai thích việc phải viết và define từng table, hay insert data vào database bằng SQL (Thậm chí developer còn tránh làm điều này mà dùng **ORM** để thay thế).

Giả sử bạn phải tạo một table Persons và load data từ một bảng Customers chẳng hạn. Chúng ta hãy so sánh 2 cách làm của DBT và thủ công.

## Thủ công

```
1 CREATE TABLE Persons (  
2     PersonID int,  
3     LastName varchar,  
4     FirstName varchar,  
5     Address varchar,  
6     City varchar  
7 );
```

SQL

Sau khi đã có table chúng ta sẽ từ từ insert data vào bằng insert

```
1 INSERT INTO Persons ( columns,..... ) VALUES ( values,.... )
```

SQL

Hoặc nếu DB có hỗ trợ CTE function chúng ta có thể viết

```
1 CREATE TABLE Persons  
2 AS (SELECT *  
3     FROM COMPANY.CUSTOMERS);
```

## DBT

Các bạn chỉ cần viết

```
1 | SELECT * FROM COMPANY.CUSTOMERS
```

Persons.sql

Thế là ... xong, bạn đã load dữ liệu của customers vào persons, đủ cột đủ dòng đủ datatype, không tin bạn hoàn toàn có thể vào database kiểm tra sau khi chạy DBT ( Nhớ đặt tên file là Persons.SQL ).

Ở một ví dụ nhỏ như thế này chúng ta có thể thấy sự khác biệt không nhiều, nhưng trên thực tế rất có ích với quy mô hàng trăm tables cùng 1 lúc và phụ thuộc vào nhau.

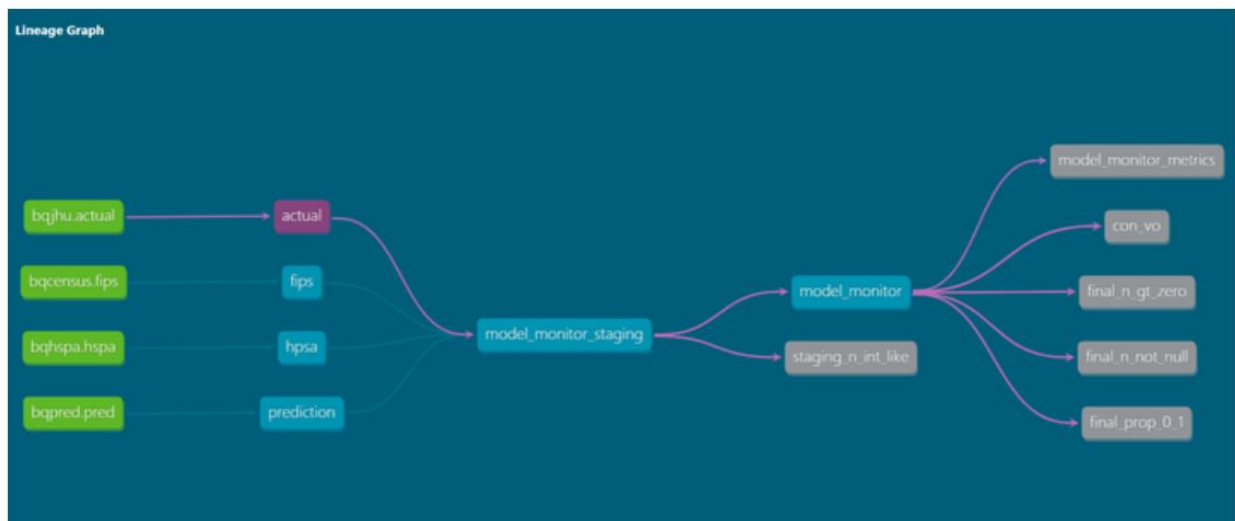
### Một số tính năng của DBT

Ngoài ví dụ trên mình xin liệt kê một số tính năng của DBT để các bạn có thể nắm sơ lược.

#### 1. Transform data without boilerplate

Như đã ví dụ phía trên.

#### 2. Models ( hoặc tạm hiểu là table cũng được ) abstraction and dependency.



Models dependency

Các table có một abstract dependency ( Tạm gọi là sự lệ thuộc ảo ) với nhau. Khi chạy SQL thông qua DBT, thì các câu SQL sẽ được chạy theo tuần tự như dependency đã được khởi tạo trong DBT-models.

#### Example:

Khi chạy project DBT phía trên hình chúng ta có thể thấy được model\_monitor có nguồn dữ liệu phụ thuộc vào model\_monitor\_staging và tiếp tục phụ thuộc vào actual và bqjhu.actual.

Như vậy tuần tự chạy của các query sẽ là bqjhu.actual >> actual >> model\_monitor\_staging >> model\_monitor.

### 3. Data Lineage

Như hình phía trên, DBT còn cung cấp công cụ document, khi bạn tạo ra dependency, DBT sẽ tự động tạo ra các tài liệu để biểu thị sự phụ thuộc giữa các model. Bạn sẽ không cần phải tự ghi tài liệu mà vẫn có tài liệu để nắm rõ nguồn gốc và mối liên hệ của data.

### 4. Jinja Support

Jinja là một templating engine rất nổi tiếng với các bạn code frontend. Nói khái quát Jinja là các thẻ để bạn có thể viết một câu SQL cực kỳ dài nhưng lại “cực kỳ ngắn” trong model của DBT.

Để giữ độ dài của bài viết hợp lý mình xin mời các bạn đọc về điều này ở link sau

<https://docs.getDBT.com/guides/getting-started/learning-more/using-jinja>

### 5. Data Documentation support

Ngoài việc có đồ thị cho data lineage, DBT còn hỗ trợ tự động tạo tài liệu cho data.

Bạn có thể **ghi lại ý nghĩa** của từng column, table do bạn tạo ra, tránh trường hợp sau này quên mất chúng là gì.

### 6. Automated Testing

DBT có rất nhiều loại test tự động khác nhau được soạn sẵn, hoặc bạn có thể tự viết bằng DBT macros. Như vậy bạn có thể chuẩn hóa quy trình bằng cách yêu cầu tất cả models pass các test được đề ra mà không cần phải check thủ công từng table.

```
version: 2
models:
  - name: dim_product
    description: "The dimension for the product data"
    columns:
      - name: PRODUCTID
        description: Product Identifier
        tests:
          - not_null
          - unique
```

tests từng column

### Tải DBT ở đâu

Hiện tại DBT được maintain chính bởi DBT labs và có 2 phiên bản là: **DBT cli**, DBT cloud,

- **DBT cli**, là phần mềm hoàn toàn **miễn phí** và open source.

Để tìm hiểu thêm về DBT cli các bạn có thể truy cập vào các đường link sau:

<https://github.com/DBT-labs/DBT-core>

<https://docs.getDBT.com/DBT-cli/install/overview>

- **DBT cloud** sẽ có giao diện rất đẹp và có nhiều tính năng riêng dành cho DBT cloud, tuy nhiên nền tảng này **tính phí**.

Để tìm hiểu về DBT các bạn có thể truy cập vào đây cho DBT cloud: <https://www.getDBT.com/>

Không có một công cụ nào là hoàn hảo. Chúng ta phải kết hợp nhiều công cụ cần thiết các nhau để đạt được hiệu quả công việc tối ưu. Mong rằng bài viết của mình đã có thể giúp bạn hình dung sơ bộ được DBT là gì.