**CÔNG TY CỔ PHẦN DỊCH VỤ iERP**

-----🙞🙜🕮🙞🙜-----

Text

Description automatically generated with low confidence

**TÀI LIỆU ĐÀO TẠO POWER BI**

new2***Hà Nội, tháng 10 năm 2021***

MỤC LỤC

[I. Giới thiệu chung 2](#_Toc84516521)

[1. Một số ứng dụng chính của Power BI 2](#_Toc84516522)

[2. Các thành phần của Power BI: Visualizations, Datasets, Reports, Dashboards, Tiles 2](#_Toc84516523)

[3. 2](#_Toc84516524)

[II. 2](#_Toc84516525)

[1. 2](#_Toc84516526)

[2. 2](#_Toc84516527)

**LỜI MỞ ĐẦU**

Với một nền văn hóa dữ liệu hiện nay, Power BI được xem là xu hướng quan trọng để sử dụng trong thời đại mà mọi quyết định kinh doanh phải dựa trên thực tế chứ không phải ý kiến. Vì thế, việc phân tích dữ liệu với Power BI sẽ giúp các nhà quản trị đưa ra các quyết định kinh doanh quan trọng.

**Power BI là gì?**

Power BI có thể giúp bạn mỗi ngày cho việc xử lý số liệu, dữ liệu. Bạn nhận được các loại dữ liệu, email với biểu đồ, thậm chí cả tài liệu in phát tay. Khi dữ liệu tích lũy, bạn khó khăn để xử lý, nguy cơ bỏ sót nhiều dữ liệu quan trọng. Power BI của Microsoft có thể giúp công việc của bạn đồng thời dễ dàng và hiệu quả hơn. Bằng cách chuyển đổi tất cả dữ liệu đó thành biểu đồ và đồ thị. Giúp bạn hình dung dữ liệu của mình theo những cách có ý nghĩa.

Ngoài ra, mọi người có thể làm việc từ cùng một lúc trên báo cáo, trên các thiết bị tin cậy. Dữ liệu sẽ tự động cập nhật và làm mới và được bảo. Nội dung được cập nhật liên tục giúp tìm kiếm các xu hướng, thông tin chi tiết nhanh chóng. Tính năng trực quan của PBI cho phép kéo thả hoặc đặt ra các câu hỏi để xem kết quả. Power BI giám sát dữ liệu, gửi các cảnh báo khi dữ liệu thay đổi trên. Thông các ngưỡng giới hạn với một tham số nào đó do bạn đặt. Tất cả dữ liệu luôn khả dụng bất cứ lúc nào, trên đám mây hay on-premise, bất kỳ thiết bị.

Như vậy Power BI là gì?

* **Power BI**là công cụ “phân tích kinh doanh” phân tích dữ liệu chia sẻ thông tin chi tiết. Kết nối dữ liệu dễ dàng, hiển thị nhanh trên Dashboard – bảng điều khiển, Reports.
* Power BI là tập hợp nhiều ứng dụng và connectors. Biến các nguồn dữ liệu không liên quan, thành thông tin chi tiết mạch lạc, trực quan và tương tác. Nguồn dữ liệu đa dạng, có thể giản tệp Excel hoặc bảng của website, đến Azure hoặc AWS.
* Đảm báo tính bảo mật cao là yếu tố quan trọng trong thời đại này.

# Giới thiệu chung

## Một số ứng dụng chính của Power BI

Ba ứng dụng chính của Power BI:

* Power BI Desktop: Một ứng dụng cho máy tính để bàn.
* Power BI service Một dịch vụ SaaS (Phần mềm dưới dạng Dịch vụ) trực tuyến.
* Power BI mobile: Phiên bản dành cho điện thoại có các hệ điều hành Android, iOS, Windows.

Ba ứng dụng này được thiết kế giúp bạn tạo, chia sẻ và sử dụng thông tin về hoạt động kinh doanh. Tùy vào vai trò và vị trí công việc của mỗi người, sẽ sử dụng ứng dụng khác nhau.

## Các thành phần của Power BI: Visualizations, Datasets, Reports, Dashboards, Tiles

**Visualizations:**

Visualizations là các biểu đồ sử dụng để hiển thị số liệu dưới dạng hỉnh ảnh hóa.

**Datasets:**

Dataset là một tập hợp dữ liệu được import hoặc kết nối vào không gian làm việc (workspaces). Power BI cho phép bạn kết nối và import tất cả các loại tập dữ liệu từ nhiều nguồn dữ liệu khác nhau.

**Reports:**

Reports là một chế độ xem đa góc độ vào một tập dữ liệu, với các visuals đại diện cho các phát hiện và thông tin chi tiết khác nhau từ tập dữ liệu đó. Một báo cáo có thể có một hoặc nhiều visual.

**Dashboards và Tiles:**

Là một tính năng của Power BI service.

Dashboard là một trang duy nhất, sử dụng các hình ảnh trực quan để kể một câu chuyện. Chính vì thế, Dashboard chỉ chứa các yếu tố quan trọng nhất.

Tiles là một ảnh chụp nhanh dữ liệu, được ghim vào trang tổng quan. Tiles có thể được tạo ra từ một report, một Dataset, một Dashboard và nhiều hơn thế nữa.

## Luồng thao tác với Power BI

Một quy trình công việc phổ biến trong Power BI bắt đầu bằng cách kết nối với các nguồn dữ liệu trong Power BI Desktop và xây dựng báo cáo. Sau đó, xuất bản báo cáo từ Power BI Desktop lên Power BI service để chia sẻ. Các người dùng khác xem được trên thiết bị di động và tương tác với báo cáo.

Power BI Desktop là một công cụ tạo báo cáo và phân tích dữ liệu. Bao gồm Power Query, kết nối với nhiều nguồn dữ liệu khác nhau thành một mô hình dữ liệu. Sau đó, bạn thiết kế một báo cáo dựa trên mô hình dữ liệu đó.

Power BI service là một dịch vụ dựa trên đám mây. Nó hỗ trợ chỉnh sửa báo cáo và cộng tác cho các nhóm và tổ chức. Bạn cũng có thể kết nối với các nguồn dữ liệu, nhưng hạn chế việc lập mô hình.

Hầu hết người dùng sử dụng Power BI Desktop để tạo báo cáo. Sau đó sử dụng dịch vụ Power BI service để cộng tác và phân phối báo cáo. Cụ thể:

* **Step 1:** Kết nối với dữ liệu:

Bước đầu tiên khi chúng ta bắt đầu làm việc trên Power BI Desktop là kết nối với nguồn dữ liệu và thêm dữ liệu bằng tùy chọn Get Data trên thanh menu. Tại đây, bạn dễ dàng chọn và kết nối với nguồn dữ liệu từ một danh sách dài các nguồn dữ liệu có sẵn.

* **Step 2:** Chuyển đổi và chỉnh sửa dữ liệu:

Ngay sau khi bạn kết nối thành công với nguồn dữ liệu, bạn có thể chuyển đổi và chỉnh sửa dữ liệu của mình với Power Query Editor. Trong Power Query Editor, bạn thực hiện đổi tên bảng, thay đổi loại dữ liệu, xóa hoặc thêm cột, áp dụng các bộ lọc và kết hợp dữ liệu từ nhiều nguồn.

* **Step 3:** Tạo visuals:

Khi dữ liệu được chuyển đổi và được tải vào Power BI Desktop, bạn có thể tạo các visual bằng mô hình dữ liệu trong khung trực quan được đưa ra ở trên cùng bên phải.

* **Step 4:** Tạo report:

Bộ công cụ hoàn hảo để tạo báo cáo hấp dẫn và nhiều thông tin được Power BI Desktop cung cấp. Báo cáo Power BI bao gồm một hoặc nhiều trang (Page) là một tập hợp các hình ảnh trực quan (Visuals), cùng nhau cung cấp một tài khoản thông tin và cái nhìn sâu sắc về dữ liệu đã nhập.

- **Step 5:** Publish báo cáo:

Sau khi tạo báo cáo trong Power BI Desktop, bạn sử dụng tùy chọn Publish để chia sẻ nó với người khác. Chúng ta có thể chia sẻ báo cáo tại bất kỳ vị trí nào của Power BI Service như không gian làm việc cá nhân của bạn, không gian làm việc nhóm hoặc ở nơi khác.

# Power Bi service

## Giao diện Power BI service

Khi mở Power BI service, người dùng sẽ thấy một Dashboard được hiển thị.

Graphical user interface, application

Description automatically generated

Các tính năng chính của giao diện người dùng Power BI servicecủa bạn là:

**1. Thanh menu**

Sử dụng để di chuyển và thao tác trong giữa các mục trong không gian làm việc như dashboards, reports, workbooks, and datasets

Graphical user interface, application

Description automatically generated

- Click vào mục “Get Data” để thêm mới một datasets, reports, and dashboards lên Power BI

- Mở rộng và thu gọn thanh menu bằng biểu tượng 

- Mục “Favorites” cho phép người dùng mở hoặc quản lý nội dung yêu thích.

- Mục “Recent” cho phép xem và mở nội dung được truy cập gần đây nhất.

- Mục “App” hiển thị mục đã xem gần đây, mở hoặc xóa một ứng dụng.

- Mục “Shared with me” hiển thị những nội dung được chia sẻ.

- Mục “Workspaces” hiển thị và mở các report.

**2. Dashboard:**

Hiển thị những nội dung chính

**3. Hộp Q&A question:**

Dùng nó để đặt câu hỏi với dữ liệu. Câu trả lời nhận được hiển thị dưới dạng visual

4. Các Icon, bao gồm tính năng trợ giúp và phản hồi

5. Dashboard title

6. Trình khởi chạy ứng dụng Office 365

7. Nút quay về trang chủ Power BI

8. Các biểu tượng icon.

## Hộp thoại Q&A trên PowerBI service.

Đôi khi, cách nhanh nhất để nhận được câu trả lời từ dữ liệu của bạn là đặt câu hỏi bằng ngôn ngữ tự nhiên. Ví dụ: "tổng doanh thu năm ngoái là bao nhiêu". Sử dụng hộp thoại Q&A để khám phá dữ liệu của bạn bằng khả năng trực quan, ngôn ngữ tự nhiên và nhận câu trả lời dưới dạng biểu đồ và đồ thị.

A picture containing graphical user interface

Description automatically generated

Q&A khác với công cụ tìm kiếm - Q&A chỉ cung cấp kết quả về dữ liệu trong Power BI.

***Làm sao Q&A biết cách trả lời các câu hỏi cụ thể?***

Nó phụ thuộc vào tên các bảng, các cột, và các trường được tính toán trong dataset.

Ví dụ: giả sử bạn có một bảng Excel có tên là “Doanh số”, với các cột có tiêu đề “Sản phẩm”, “Tháng”, “Số đã bán”, “Tổng doanh số” và “Lợi nhuận”. Bạn có thể đặt câu hỏi về bất kỳ thực thể nào trong số đó. Bạn có thể hỏi "hiển thị doanh số bán hàng," tổng lợi nhuận theo tháng "," sắp xếp sản phẩm theo đơn vị đã bán ", vv…. Q&A có thể trả lời các câu hỏi dựa trên cách tổ chức trên dataset.

## Xuất Report/Dashboard

Trong Power BI Desktop, bạn có thể tải báo cáo lên Power BI service bằng cách lưu báo cáo và chọn Publish. Ngược lại trên PowerBi service, bạn cũng có thể download báo cáo Power BI xuống PBI desktop. Tệp tải xuống sẽ có đuôi là .pbix.

## Các tính năng chia sẻ, cộng tác trên PowerBI service.

Power BI Service là nơi chứa các thư viện báo cáo và từ đó chia sẻ để người dùng theo dõi hoặc phối hợp chỉnh sửa.

Để chia sẻ một báo cáo, chúng ta có thể truy cập vào báo cáo đó và tương tác trên thanh công cụ phía trên. Click mục file -> chọn Embed report chúng ta sẽ thấy có nhiều các cách chia sẻ khác nhau.

Ngoài ra nút Share trên thanh công cụ cho phép người dùng chia sẻ báo cáo qua đường link publish (publish to web). Đây là cách sử dụng thông dụng nhất đặc biệt là khi người dùng chỉ có tài khoản Power BI miễn phí. Power BI cho phép chia sẻ báo cáo của mình qua một đường link mã hóa. Đường link này khi bạn cung cấp cho bất cứ ai họ đều sẽ xem và theo dõi báo cáo của bạn dễ dàng qua các trình duyệt web mà không cần có tài khoản Power BI (không yêu cầu đăng nhập).

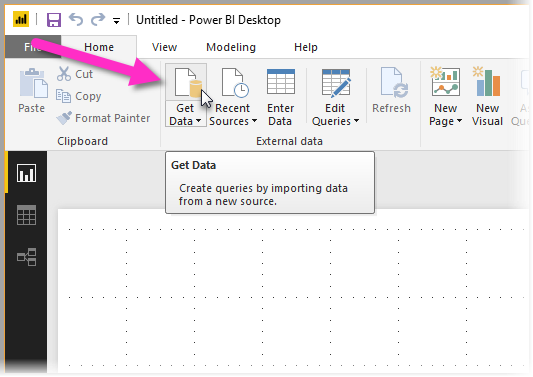
Graphical user interface, application

Description automatically generated

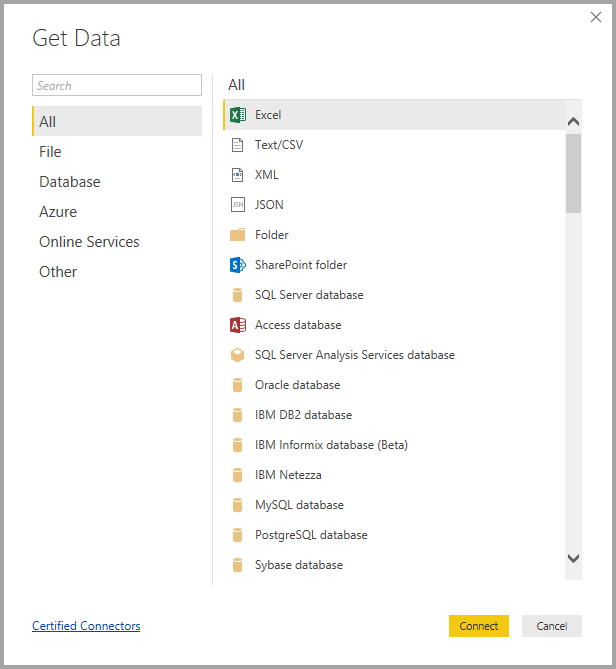
# Power BI Desktop

## Kết nối dữ liệu

Hiện nay, bạn có thể dễ dàng kết nối với Power BI từ rất nhiều các loại dữ liệu khác nhau như Excel, SQL, CSV, Web… Thậm chí công cụ này còn cho phép kết nối đến các dịch vụ lưu trữ online nơi chứa file dữ liệu của bạn như Salesforce, Microsoft Dynamics, Azure Blob Storage…



Thao tác khá đơn giản, tại giao diện chính chọn “Home” rồi đến “Get Data”. Lúc này, cửa sổ Get Data sẽ mở ra và cho phép bạn chọn kiểu file sẽ tải lên.



## Các kiểu kết nối dữ liệu trong PBI

Khi tiến hành lấy dữ liệu từ các nguồn dữ liệu, ngoài việc chỉ ra đường dẫn đến Data Source, người dùng cần chọn loại kết nối (Data Connection Types) phù hợp. Tùy vào nguồn dữ liệu như: tập tin CSV, tập tin Excel, SQL Server Database…mà chúng ta sẽ có một hoặc nhiều lựa chọn kết nối tương ứng.

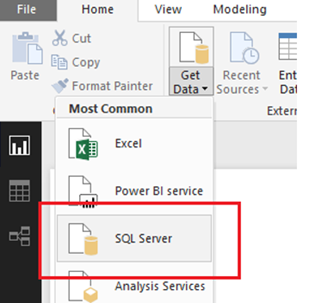
Trong PowerBI hỗ trợ ba loại kết nối:

* **Import**.
* **DirectQuery.**
* **Live Connection.**

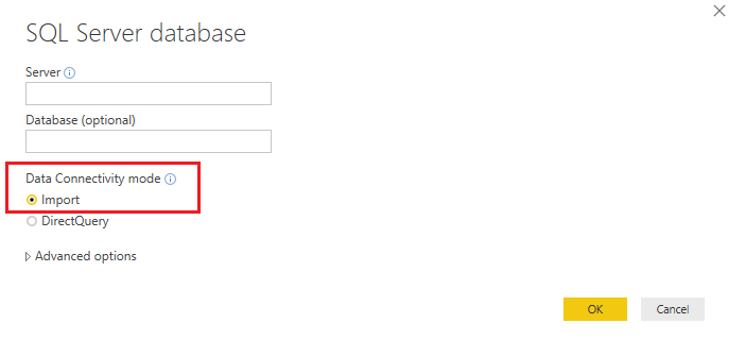
### Import

Import là loại kết nối thường gặp nhất, đa phần các Data Source đều hỗ trợ phương thức này. Khi chọn cách này, Power BI sẽ nạp **toàn bộ dữ liệu thô** từ nguồn dữ liệu vào Power BI Desktop. Thông qua những dữ liệu thô này, chúng ta hoàn toàn có thể tạo ra các mối quan hệ (mối quan hệ kiểu như trong SQL hoặc Access) giữa các bảng dữ liệu, từ đó chỉnh sửa & lựa chọn những dữ liệu cần thiết và tạo ra báo cáo như mong muốn.

**Ví dụ:** Để Import dữ liệu từ nguồn dữ liệu SQL Server Database. Tại Menu **Home**, chọn **Get Data** là **SQL Server.**



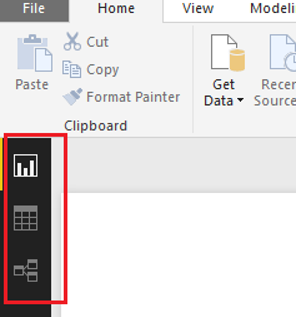
Điền vào các thông tin của Source Data (Server, Database name, …). Tại mục Data Connectivity mode, chọn **Import**.



Nhấn Ok và chờ một lúc để dữ liệu được nạp vào. (Người dùng có thể sẽ được yêu cầu lựa chọn các bảng dữ liệu mong muốn trong quá trình nạp).

Sau khi nạp xong, chúng ta có toàn quyền trong việc tạo các kết nối giữa các bảng dữ liệu xem thông tin của từng bảng dữ liệu và tạo các báo cáo dựa trên dữ liệu đã nạp.

Để thực hiện việc này, người dùng có thể chuyển qua lại giữa ba góc nhìn: **Report**, **Data** và **Relationship** theo thanh công cụ như hình dưới (thứ tự từ trên xuống).

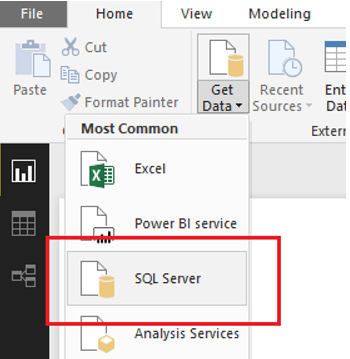


### DirectQuery

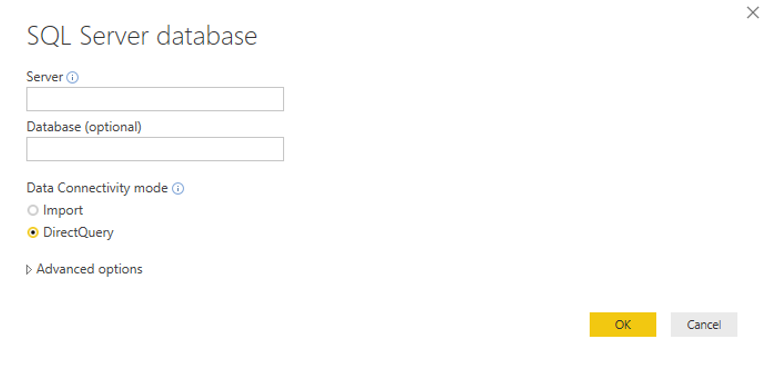
Cách thức lấy dữ liệu phổ biến thứ hai là DirectQuery. Phương thức này không nạp trực tiếp dữ liệu vào mà chỉ **thực hiện câu Query** lên Data Source để lấy những dữ liệu thật sự cần thiết. Dĩ nhiên, ngôn ngữ Query phải phù hợp với Data Source. Hiện tại, chỉ một số Data Source hỗ trợ phương thức này ([danh sách hỗ trợ](https://docs.microsoft.com/en-us/power-bi/desktop-directquery-data-sources)).

**Ví dụ:** Để lấy dữ liệu từ nguồn dữ liệu SQL Server Database,

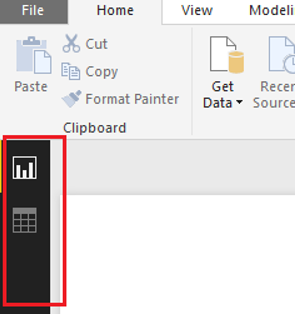
Tại Menu Home, chọn **Get Data** là **SQL Server**.



Điền vào các thông tin của Source Data (Server, Database name, …). Tại mục **Data Connectivity** mode, chọn **DirectQuery**.

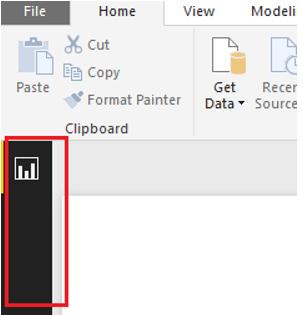


Về bản chất, cách thức lấy dữ liệu này chỉ thực hiện những câu Query để lấy dữ liệu lên, nên dữ liệu vẫn còn nằm ở Data Source, đồng thời các mối liên hệ giữa các bảng dữ liệu (nếu có) đều được giữ nguyên như ở Data Source. Vậy nên, chúng ta không có lựa chọn điều chỉnh mối liên hệ ở đây (thiếu mục Relationship so với phương thức Import ở trên).



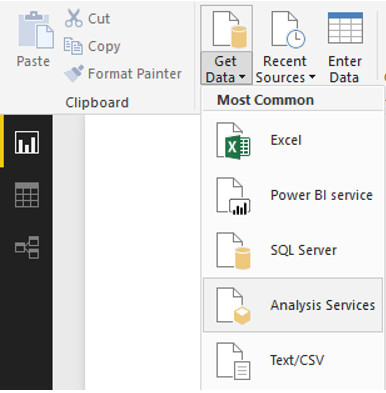
### Live Connection

Phương thức kết nối này mặc định các công đoạn tiền xử lý đã được thực hiện ở Data Source. Power BI sẽ chỉ đóng vai trò là công cụ tạo các báo cáo. Các lựa chọn Data và Relationship ở đây đều bị mất.

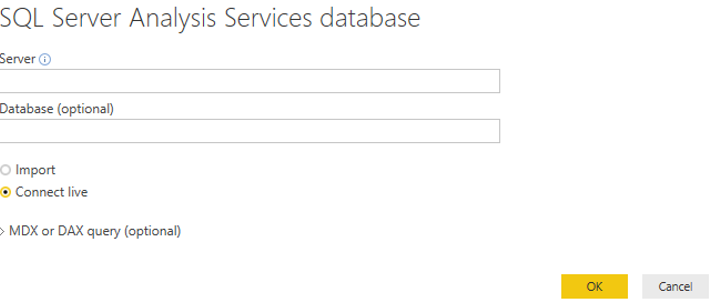


Hiện tại, các Data Source hỗ trợ phương thức này đều dựa trên **SQL Server Analysis Service**, bao gồm: Multidimensional, Azure Tabular và Tabular on premises.

**Ví dụ:** chọn Get **Data** là **Analysis Services**



Tại mục chọn**Connection Type**, chọn **Connect Live**.



## Làm sạch và chuyển đổi dữ liệu

Trong khi làm báo cáo với Power BI, chắc hẳn sẽ có lúc bạn gặp phải một số lỗi nhất là khi phân tích dữ liệu từ nhiều nguồn khác nhau, chả hạn:

* Nhiều cột có một lỗi trong đó
* Giá trị rỗng trong nhiều cột
* Trùng lặp dữ liệu
* Cột cần dữ liệu dạng chữ lại chỉ chứa các giá trị số
* ………

Để giải quyết được vấn đề này, điều cần thiết là phải làm sạch dữ liệu của bạn một cách thường xuyên giúp duy trì khả năng phân tích dữ liệt chính xác và nhanh chóng.

### Khái niệm chung

* Làm sạch dữ liệu là một quá trình loại bỏ và phát hiện các bản ghi nhận không chính xác từ một tập tài liệu. Nó được sử dụng để thay đổi dữ liệu trong một nguồn lưu trữ nhất định để đảm bảo rằng nó là chính xác.
* Chuyển đổi là một quá trình lọc ra các bản ghi nhận không dùng được của cơ sở dữ liệu trong Power BI. Những dữ liệu không hữu ích để tạo báo cáo và phân tích cần được loại bỏ.

### Khi nào nên sử dụng làm sạch và chuyển đổi dữ liệu?

Khi bất kỳ dữ liệu nào có định dạng không chính xác và gặp một số lỗi trong tạo báo cáo trực quan thì chúng ta cần phải làm sạch và biến đổi dữ liệu đó để có thể xóa các bản ghi không đúng khỏi tập dữ liệu đó. Khi nhập từ hai tập dữ liệu trở lên trong Power BI chúng ta cần quản lý mối liên hệ giữa giữa chúng vì vậy việc làm sạch dữ liệu không thể thực hiện, đồng thời gặp phải một số vấn đề khác. Trong tình huống này, chúng ta cần thực hiện theo quy trình làm sạch và chuyển đổi dữ liệu để giải quyết sự cố bằng cách sử dụng truy vấn nguồn và lưu lại vào tệp đã nhập.

### Các bước làm sạch và chuyển đổi dữ liệu trong Power BI

#### Nhập dữ liệu

Nhập dữ liệu là bước đầu tiên của quá trình làm sạch dữ liệu. Đầu tiên nhấp vào tab **Get Data** sau đó chọn **File** và **Workboo**k trong menu. Sẽ có một danh sách tệp hiện ra trên màn hình để điều hướng tệp Excel cần nhập.

Sau khi chọn File để nhập dữ liệu sẽ xuất hiện cửa sổ Bộ điều hướng (Navigator window) cho phép bạn chọn trang tính từ Workbook được kết hợp với dữ liệu đã nhập.

Trong khi chọn dữ liệu, bạn có thể xem bản xem trước của dữ liệu ở phía bên phải

Nếu mục dữ liệu của bạn có nhiều trang tính, hãy nhấp vào chọn nhiều mục.

Bạn có thể chọn giữa tùy chọn Tải và Chỉnh sửa theo ý mình. Bạn làm sạch dữ liệu của mình sau đó nhấp vào nút Chỉnh sửa và mở nó trong trình chỉnh sửa truy vấn (Query editor)

#### Power Query Editor

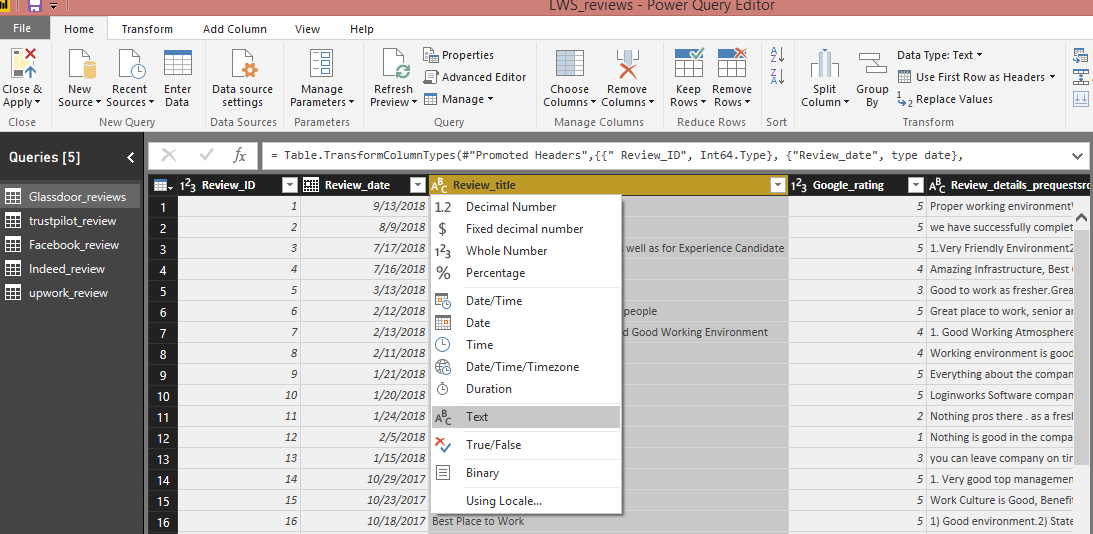
Trong trình chỉnh sửa này, chúng ta có thể bắt đầu chuyển đổi dữ liễu để cung cấp cho nó một định dạng bảng thích hợp.

Bất kỳ biến đổi nào bạn thực hiện trong dữ liệu, nó sẽ xuất hiện trong khu vực Applied step. Bạn có thể nhấp vào bất kỳ bước nào trong danh sách.

Diagram

Description automatically generated

Khi mở cửa sổ chỉnh sửa cho bất kỳ bảng nào trong Power BI thì bạn có thể thực hiện nhiều thao tác nếu bạn cần thay đổi kiểu dữ liệu của bất kỳ cột nào. Sau đó nhấp chuột phải vào tên cột và nhận danh sách kiểu dữ liệu của cột.



Sử dụng Power Query Editor chúng ta có thể thực hiện các bước chỉnh sửa trên bảng như sau:

* Hợp nhất cột bằng cách sử dụng merge
* Thêm cột từ tab Add column
* Nhập dữ liệu mới bằng cách sử dụng nguồn mới theo yêu cầu
* Tạo tham số và chỉnh sửa tham số từ quản lý tham số
* Cung cấp tham chiếu, xóa và sao chép từ tab Manage
* Chọn cột và loại bỏ cột
* Giữ hàng và xóa hàng
* Chia cột thành hai phần
* Thay thế giá trị
* Tạo thêm truy vấn mới
* Chuyển đổi, hoàn nguyên và đếm hàng
* Thay đổi định dạng
* Tạo kết nối với R Script bằng Run R script
* Sử dụng định dạng điều kiện trên dữ liệu
* Viết mã trong trình chỉnh sửa nâng cao
* Quản lý các tùy thuộc truy vấn từ cơ sở dữ liệu
* Thiết lập cài đặt của chỉnh sửa theo yêu cầu
* Làm mới và tải báo cáo bằng cách sử dụng các thuộc tính trong trình chỉnh sửa truy vấn
* Để lưu dữ liệu, hãy nhấn tùy chọn Close và Load từ trang chủ của trình chỉnh sửa truy vấn.

Bạn có thể thực hiện nhiều quy trình theo yêu cầu báo cáo trong Power Query Editor. Nếu muốn có bất kỳ thay đổi nào về dữ liệu, bạn có thể cập nhật nhiều tùy chọn. Khi bạn có dữ liệu ở định dạng không chính xác hoặc định dạng không có cấu trúc thì có thể áp dụng quy trình làm sạch và chuyển đổi dữ liệu và có thể tiến hành xử lý lỗi.

### Append và Merge/Join

#### Append

Append Queries có nghĩa là nối hai hoặc nhiều bảng lại với nhau theo cách:

* Các hàng sẽ được nối với nhau.
* Các cột phải có cùng kiểu cấu trúc và định dạng với nhau.

Ví dụ: có 2 hai bảng dữ liệu trích từ doanh thu bán hàng tháng 2 và tháng 3 file excel như hình bên dưới:

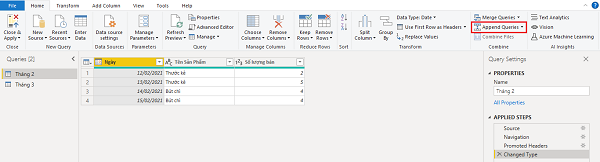
Table

Description automatically generated

Table

Description automatically generated

Các bạn hãy thực hiện đưa chúng vào Power Query Editor. Sau đó thực hiện nối hai bảng lại với nhau bằng cách các bạn chọn 1 trong 2 bảng cần nối, vào tab**Home**, chọn **Append Queries**.



Nếu bạn muốn giữ nguyên kết quả hiện có, và tạo một bảng mới với kết quả được thêm vào, hãy chọn “**Append Queries as New**”, nếu không chỉ cần chọn “**Append Queries**”.

Sau khi chọn “**Append Queries as New**” hộp thoại **Append** được mở ra:

**First table:**bảng đầu tiên là bảng thường được chọn trước khi thực hiện Append  
**Second table:**chọn các bảng sẽ được nối vào bảng chính.

Graphical user interface, application

Description automatically generated

Kết quả sau khi thực hiện Append

Graphical user interface, application, table

Description automatically generated

#### MERGE/JOIN

Tính năng Merge/Join trong Power BI tạo ra một Query mới từ hai Query có sẵn. Một Query chứa tất cả các cột từ một bảng chính (Primary Table), trong đó có một cột đóng vai trò là một cột chứa liên kết điều hướng đến một bảng có liên quan- tức bảng phụ. Bảng liên quan chứa tất cả các hàng khớp với từng hàng từ một bảng chính dựa trên một giá trị cột chung. Tính năng Merge/Join có thể thay thế cho việc sử dụng Vlookup trong Excel.

Lưu ý, để sử dụng tính năng Merge/Join:

* Dữ liệu phải được nạp vào Power BI
* Dữ liệu ở dạng bảng
* Giữa các bảng phải có ít nhất 01 cột làm trung gian để có thể liên kết các bảng với nhau
* Cần xác định bảng nào là bảng đầu, bảng nào là bảng sau để lựa chọn được kiểu Merge phù hợp.

Có 4 kiểu Merge/Joinb hay gặp trong Power BI:

1. Inner Join
2. Full Outer
3. Left Outer
4. Right Outer

* **Inner Join**

Chọn kiểu Merge là Inner Join khi ta muốn lấy ra dữ liệu trùng của 2 bảng hiện có – tức những dòng dữ liệu tồn tại ở trong cả hai bảng.

* **Full Outer**

Chọn kiểu Merge là Full Outer khi ta muốn lấy tất cả các dòng từ cả hai bảng. Cụ thể, khi một giá trị tồn tại trong một bảng, nhưng không tồn tại trong bảng còn lại, giá trị này vẫn được hiển thị trong kết quả Merge dưới dạng một dòng mới chứa giá trị null.

* **Left Outer**

Tính năng này được sử dụng như một loại VLOOKUP trong đó các bản ghi từ một bảng tham chiếu được mang sang một bảng khác, nhưng nó cũng có thể được sử dụng theo cách khác tùy thuộc vào việc chúng ta muốn sử dụng thao tác Mở rộng (Expand) hay Tổng hợp (Aggregation) sau thao tác Merge. Bảng mới sau khi sử dụng tính năng Left Outer Join sẽ mang tất cả các bản ghi từ bảng đầu tiên (First Table) và chỉ các bản ghi từ bảng thứ hai (Second Table) khớp với bản ghi từ bảng đầu tiên.

* **Right Outer**

Với lựa chọn Merge là Right Outer, bảng thứ hai (Second Table) là bảng chính được giữ lại, nó sẽ đóng vai trò như một bộ lọc áp dụng lên bảng thứ nhất, tức là bảng thứ hai sẽ kiểm soát giá trị nào hay dòng nào của bảng thứ nhất sẽ được hiển thị trong bảng kết quả.

# Tổng quan về DAX trong POWER BI

## Khái niệm

**DAX** là viết tắt của cụm từ **Data Analysis Expressions** là tập hợp các hàm, toán tử và hằng số, được sử dụng để giải quyết những bài toán từ cơ bản đến phức tạp. Từ đó, trả về một hoặc nhiều giá trị từ dữ liệu có sẵn.

Nếu bạn đã từng sử dụng hàm trong **Excel** thì có thể hình dùng **DAX**là hàm trong **Power BI**, dù có một vài điểm khác biệt. Để dễ hiểu hơn, hãy lấy một ví dụ về dữ liệu bán hàng của công ty ở các quốc gia, bao gồm các thông tin: Thời gian, doanh số, thuế, tên quốc gia…

Khi phân tích dữ liệu này cần chỉ ra những con số chi tiết hơn như lợi nhuận sau thuế (Doanh số - thuế), tính trung bình lợi nhuận theo tháng ở mỗi quốc gia (Lợi nhuận năm/12)…

## Cấu trúc chung

Cấu trúc chuẩn thông thường của hàm DAX thường như sau:

Timeline

Description automatically generated

## Sự khác biệt DAX và các hàm trong Excel

Như đã chia sẻ, **DAX** có nhiều điểm tương đồng với hàm trong **Excel**, dù vẫn có nhiều điểm khác biệt. Vì vậy, cần làm rõ một số lưu ý để tránh việc các bạn đã từng dùng **Excel** có những hiểu lầm.

* Một số hàm**DAX** và **Excel** có cùng tên gọi, thậm chí là chức năng nhưng kiểu dữ liệu đầu vào và đầu ra có thể khác nhau. Khi kết nối các dữ liệu từ file **Excel**cần chú ý rằng bạn không thể dùng hàm **Excel**trong **Power BI.**
* Hàm **DAX** không lấy tham chiếu ô hoặc phạm vi làm tham chiếu mà chỉ sử dụng bảng hoặc cột.
* Nếu như các hàm ngày và thời gian trong **Excel** đươc trả về kiểu số nguyên thì hàm **DAX**trả về kiểu datetime.
* Nhiều hàm **DAX** có thể trả về bảng chứa các giá trị hoặc thực hiện các phép tính dựa trên dữ liệu đầu vào là một bảng chứa các giá trị. Trong khi đó, **Excel** không có hàm trả về bảng, nếu dữ liệu **Excel** của bạn không thể đáp ứng nhu cầu này bạn có thể kết nối đến **Power BI** và xử lý trực tiếp bằng các hàm **DAX**.
* **DAX** mang đến các hàm tra cứu mới tương tự tra cứu mảng và vec-tơ thường gặp trong **Excel.** Lưu ý, **DAX** yêu cầu thiết lập mối quan hệ giữa các bảng.
* Dữ liệu trong cùng cột sẽ luôn có cùng kiểu, nếu không **DAX**sẽ tự động thay đổi chúng thành kiểu tốt nhất có thể chứa tất cả dữ liệu trong cột.

# TỔNG HỢP BỘ HÀM DAX

## Nhóm hàm toán học (Math & Trig Functions)

Các hàm toán học trong DAX tương tự như các hàm toán học và lượng giác của Excel. Tuy nhiên, có một số khác biệt trong các kiểu dữ liệu số được các hàm DAX sử dụng.

Một số DAX hay sử dụng.

### SUMX ()

**Mục đích:** Tính tổng từng dòng trên bảng tham chiếu dựa vào phương thức tính toán được khai báo.

Cú pháp: **SUMX(<table>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | khai báo thông tin bảng chứa dữ liệu cần tính toán |
| expression | khai báo các phương thức tính toán, có thể kết hợp các hàm filter |

**Giá trị trả về:** giá trị thập phân

VD: thực hiện tính toán Tổng tiền dựa trên 2 thông tin: **Số lượng** & **Giá bán** trong bảng Banhang:

**SUMX(‘Banhang’,**

**‘Banhang'[Số lượng] \* ‘Banhang'[Giá bán])**

### RAND() & RANDBETWEEN()

Mục đích đều dùng để tạo một giá trị ngẫu nhiên, thường dùng trong trường hợp cần giả lập giá trị để thử nghiệm các tính toán. Vì 2 hàm này có mục đích sử dụng như nhau & không khác biệt nhau nhiều về cú pháp nên sẽ gom chung như sau:

RAND()

**Mục đích:** dùng để tạo giá trị ngẫu nhiên trong khoảng giá trị cho phép: >=0 && <1.

Cú pháp: **RAND()**

**Kết quả trả về:** số thập phân. VD: **RAND()**

-> Kết quả trả về số sẽ là số ngẫu nhiên trong dãy số từ 0 ->1.

RANDBETWEEN()

**Mục đích:** dùng tạo giá trị ngẫu nhiên trong khoảng giá trị được người dùng nhập liệu. Cú pháp: **RANDBETWEEN(value 1, value 2)**

**Kết quả trả về:** dạng số. VD: **RANDBETWEEN(1,100)**

-> Kết quả trả về số sẽ là số ngẫu nhiên trong dãy số từ 1 -> 100.

**Khả năng mở rộng:**

Hai hàm trên có thể dùng kết hợp với các công thức tính toán hay các hàm khác, thậm chí bạn có thể kết hợp 2 hàm này với nhau.

### DIVIDE()

**Mục đích:** thực hiện phép chia

Cú pháp: **DIVIDE(<numerator>,<denominator>[,<alternateresult>])**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| numerator | Khai báo tử số của hàm |
| denominator | Khai báo mẫu số cần chia |
| alternateresult | Giá trị trả về trong trường hợp chia cho giá trị 0, nếu không khai báo giá trị trả về mặc định là BLANK() |

**Kết quả trả về:** số thập phân VD: **DIVIDE(5,2)**

-> Kết quả trả về: 2.5

ROUND()

**Mục đích:** hàm thực hiện làm tròn giá trị. Cú pháp: **ROUND(<number>, <num\_digits>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| number | Khai báo số bạn muốn làm muốn thực hiện việc làm tròn giá trị. |
| num\_digits | Được linh hoạt khai báo tùy theo giá trị mong muốn như sau:   * Giá trị > 0: làm tròn giá trị thập phân * Giá trị = 0: làm tròn đến giá trị gần nhất * Giá trị < 0: làm tròn theo giá trị đầu tiên |

**Kết quả trả về:** số đã được làm tròn theo tham số num\_digits. VD1: Trường hợp giá trị tham số truyền vào > 0:

**ROUND(29.09, 1)**

-> Kết quả trả về: 29.10

VD2: Trường hợp giá trị tham số truyền vào = 0

**ROUND(29.09, 0)**

-> Kết quả trả về: 29.00

VD3: Trường hợp giá trị tham số truyền vào < 0

**ROUND(29.09, -1)**

-> Kết quả trả về: 30.00

## Nhóm hàm thống kê (Statistical Functions)

**Mô tả:** nhóm hàm này thực hiện các kết hợp. Ngoài việc tạo tổng và số trung bình, hoặc tìm các giá trị tối thiểu và tối đa, bạn cũng có thể lọc cột trước khi tổng hợp hoặc tạo các tập hợp dựa trên các bảng có liên quan.

### COUNT()

**Mục đích:** đếm số ô trong cột chứa giá trị số. Cú pháp: **COUNT(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị số cần thực hiện đếm. |

**Giá trị trả về:** trả về số dòng đếm được với kiểu định dạng số nguyên.

**Ví dụ**: Đếm giá trị số trong cột Ngày bán:

**COUNT(‘Sale '[Ngay ban])**

### COUNTA()

**Mục đích:** đếm số lượng ô trong cột không trống. Không chỉ đếm các hàng chứa giá trị số, mà còn bao gồm văn bản, ngày tháng và các giá trị logic.

Cú pháp: **COUNTA(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị số cần thực hiện đếm. |

**Giá trị trả về:** trả về số dòng đếm được với kiểu định dạng số nguyên. **Ví dụ**: Đếm số sản phẩm trong cột Sản phẩm – không đếm trường rỗng. **COUNTA(‘Sale '[San pham])**

### COUNTBLANK()

**Mục đích:** đếm số lượng ô trống trong cột chỉ định. Cú pháp: **COUNTBLANK(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị số cần thực hiện đếm. |

**Giá trị trả về:** trả về số dòng đếm được với kiểu định dạng số nguyên. Nếu không có dòng nào thỏa điều kiện sẽ trả về giá trị rỗng.

**Ví dụ**: Đếm số sản phẩm trong cột Sản phẩm – đếm trường rỗng.

**COUNTBLANK(‘Sale '[San pham])**

### COUNTROWS()

**Mục đích:** đếm số lượng dòng có giá trị trong cột chỉ định. Cú pháp: **COUNTROWS(<table>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng chứa dòng có giá trị số cần thực hiện đếm. |

**Giá trị trả về:** trả về số dòng đếm được với kiểu định dạng số nguyên.

**Ví dụ**: Đếm số sản phẩm trong bảng SALE:

**COUNTROWS(‘Sale ')**

### COUNTX()

**Mục đích:** đếm số lượng dòng có giá trị trong cột chỉ định.

Cú pháp: **COUNTX(<table>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng chứa dòng có giá trị số cần thực hiện đếm. |
| expression | Khai báo biểu thức trả về nhóm giá trị có chứa giá trị cần  đếm. |

**Giá trị trả về:** trả về số dòng đếm được.

**Ví dụ**: Đếm tất cả các dòng sản phẩm có định nghĩa giá bán:

**COUNTX(‘Sale ’, [Gia ban])**

### DISTINCTCOUNT()

**Mục đích:** đếm số lượng ô có giá trị trong cột chỉ định, chỉ đếm cột có giá trị khác nhau. Cú pháp: **DISTINCTCOUNT(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị số cần thực hiện đếm. |

**Giá trị trả về:** trả về số dòng đếm các cột có giá trị duy nhất.

**Ví dụ**: Đếm số sản phẩm trong cột Sản phẩm – không đếm sản phẩm cùng tên.

**DISTINCTCOUNT(‘Sale '[San pham])**

### CROSSJOIN()

**Mục đích:** tạo bảng mới có giá trị từ các cột trong các bảng chỉ định. Cú pháp: **CROSSJOIN(<table>,<table>[,<table>]…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo các bảng dùng tạo bảng mới |

**Giá trị trả về:** trả về bảng có giá trị từ các bảng chỉ định.

**Ví dụ**: Tôi tạo bảng mới từ 2 bảng Loại hình áp dụng (SMB, Enterprise) và bảng Tên sản phẩm (Office 365..)

**CROSSJOIN(Business\_Type, Product\_Name)**

### MIN()

**Mục đích:** trả về giá trị nhỏ nhất trong cột, không áp dụng cho giá trị logic hoặc văn bản. Cú pháp: **MIN(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột muốn tìm giá trị nhỏ nhất. |

**Giá trị trả về:** trả về giá trị nhỏ nhất trong cột chỉ định. **Ví dụ**: tôi thực hiện tìm giá trị doanh thu thấp nhất: **MIN(‘Sale '[Doanh thu])**

### MINX()

**Mục đích:** trả về giá trị số nhỏ nhất có kết quả từ việc đánh giá một biểu thức cho mỗi dòng của một bảng.

Cú pháp: **MINX(<table>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng muốn tìm giá trị nhỏ nhất. |
| expression | Thông tin biểu thức cần thực hiện kiểm tra trên từng dòng của bảng |

**Giá trị trả về:** trả về dòng có giá trị nhỏ nhất.

**Ví dụ**: tôi thực hiện tìm giá trị lợi nhuận thấp nhất trong năm 2020:

**MINX(**

**FILTER(‘Sale '[Nam] =2020),[Loi nhuan])**

### MAX()

**Mục đích:** trả về giá trị lớn nhất trong cột, không áp dụng cho giá trị logic hoặc văn bản. Cú pháp: **MAX (<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột muốn tìm giá trị lớn nhất. |

**Giá trị trả về:** trả về giá trị lớn nhất trong cột chỉ định. **Ví dụ**: tôi thực hiện tìm giá trị doanh thu lớn nhất: **MAX(‘Sale '[Doanh thu])**

### MAXX()

**Mục đích:** trả về giá trị số nhỏ lớn có kết quả từ việc đánh giá một biểu thức cho mỗi dòng của một bảng.

Cú pháp: **MAXX(<table>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng muốn tìm giá trị lớn nhất. |
| expression | Thông tin biểu thức cần thực hiện kiểm tra trên từng dòng của bảng |

**Giá trị trả về:** trả về dòng có giá trị lớn nhất.

**Ví dụ**: tôi thực hiện tìm giá trị lợi nhuận lớn nhất trong năm 2017:

**MAXX(**

**FILTER(‘Sale '[Nam] =2017),[Loi nhuan])**

### RANKX()

**Mục đích:** trả về thứ tự xếp hạng của một số trong danh sách các số cho mỗi dòng trong bảng. Cú pháp: **RANKX(<table>, <expression>[,<value>[,<order>[,<ties>]]])**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bất kỳ biểu thức DAX nào trả về một bảng dữ liệu mà biểu thức  được đánh giá. |
| expression | Bất kỳ biểu thức DAX nào trả về một giá trị vô hướng. Biểu thức được đánh giá cho mỗi hàng của bảng, để tạo ra tất cả các giá trị có thể có để xếp hạng |
| value | Bất kỳ biểu thức DAX nào trả về một giá trị vô hướng duy nhất có thứ hạng được tìm thấy.  Khi tham số giá trị bị bỏ qua, giá trị của biểu thức tại dòng hiện tại  được sử dụng thay thế. |
| order | Giá trị chỉ định cách xếp hạng giá trị, từ thấp đến cao hoặc cao  đến thấp. |

**Giá trị trả về:** danh sách thứ tự xếp hạng.

**Ví dụ**: tôi muốn xếp hạng khách hàng theo lợi nhuận bán hàng theo thứ tự từ cao đến thấp:

**RANKX(**

**ALL(‘Sale '[Khach hang], [Loi nhuan], ,Desc)**

### AVERAGE()

**Mục đích:** trả về giá trị trung bình (trung bình số học) của tất cả các số trong một cột. Cú pháp: **AVERAGE(<column>])**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Cột chứa các số mà bạn muốn tính trung bình. |

**Giá trị trả về:** giá trị trung bình các số trong cột chỉ định. **Ví dụ**: tôi muốn tính giá trị trung bình doanh thu bán hàng: **AVERAGE(‘Sale '[Doanh thu])**

### AVERAGEX()

**Mục đích:** tính trung bình (trung bình số học) của một tập hợp các biểu thức được đánh giá trên

một bảng.

Cú pháp: **AVERAGEX(<table>], <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng cần thực hiện tính trung bình |
| expression | Khai báo biểu thức cần đánh giá trên bảng. |

**Giá trị trả về:** giá trị trung bình theo kiểu thập phân.

**Ví dụ**: tôi muốn tính giá trị trung bình của Giá bán với Thuế áp dụng theo từng mặt hàng:

**AVERAGEX(‘Sale ',**

**‘Sale '[Gia ban] + ‘Sale '[VAT])**

### ADDCOLUMNS()

**Mục đích:** thêm cột vào bảng được chỉ định, cột thêm vào có thể là giá trị tham chiếu từ cột khác hoặc một biểu thức tính toán tạo giá trị theo dòng.

Cú pháp: **ADDCOLUMNS(<table>,<name>,<expression>[,<name>,<expression>]…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng cần thêm cột. |
| name | Khai báo tên cột mới. |
| expression | Khai báo biểu thức tính toán tạo giá trị theo từng dòng. |

**Giá trị trả về:** cột mới được thêm vào bảng chỉ định.

**Ví dụ**: tôi muốn thêm cột giá gốc và thuế vào bảng Sale :

**ADDCOLUMNS(‘Sale ', “Giá gốc”,**

**SUMX(**

**RELATEDTABLE(‘BangGia '), ‘BangGia '[Gia goc]) “Thuế”,**

**SUMX(**

**RELATEDTABLE(‘BangGia '), ‘BangGia '[VAT])**

### SUMMARIZE()

**Mục đích:** Trả về bảng tóm tắt cho tổng số được yêu cầu trên một nhóm các nhóm.

Cú pháp: **SUMMARIZE(<table>, <groupBy\_columnName>[,<groupBy\_columnName>]… [,<name>,<expression>]…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Khai báo bảng cần thêm cột. |
| groupBy\_columnName | Tên đủ điều kiện của cột hiện tại được sử dụng để tạo nhóm tóm tắt dựa trên các giá trị được tìm thấy trong đó. Tham số này không thể là một biểu thức. |
| name | Tên được đặt cho cột mới, kèm theo dấu ngoặc kép. |
| expression | Bất kỳ biểu thức DAX nào trả về một giá trị vô hướng duy nhất,  trong đó biểu thức được đánh giá nhiều lần (theo mỗi dòng / ngữ cảnh). |

**Giá trị trả về:** bảng có các cột được chọn cho các đối số *groupBy\_columnName* và các cột tóm tắt

được tạo bởi các đối số tên.

**Ví dụ**: tôi muốn tạo bảng tóm tắt doanh thu bán hàng và giá trị giảm giá bán hàng được nhóm lại

theo năm bán hàng và theo nhóm sản phẩm thực tế bán:

**SUMMARIZE(‘Thongke ',**

**‘Date [Nam],**

**“Product [Nhom san pham], “Doanh thu”, SUMX(**

**RELATEDTABLE(‘Sale '), ‘Sale '[Doanh thu])**

**“Giá giảm”, SUMX(**

**RELATEDTABLE(‘Sale '), ‘Sale '[Giá giảm]))**

## Nhóm hàm Thời gian (Date and Time Functions)

**Mô tả:** Nhóm hàm thời gian khá phức tạp và được sử dụng nhiều nhất trong phân tích dữ liệu. Nhiều hàm ngày tháng và thời gian trong DAX rất giống với hàm ngày tháng và thời gian của Excel. Tuy nhiên, các hàm DAX sử dụng kiểu dữ liệu datetime và có thể lấy các giá trị từ một cột làm đối số. DAX cũng bao gồm một tập các chức năng thông minh phân tích về thời gian cho phép bạn thao tác dữ liệu bằng các khoảng thời gian, bao gồm ngày, tháng, quý và năm, sau đó xây dựng và so sánh các phép tính trong các chu kỳ thời gian.

Trong nhóm này, tôi sẽ đồng thời gom chung các hàm về Thời gian cơ bản (Date & Time) cùng

nhóm hàm đặc biệt (Time Intelligence).

### DATE()

**Mục đích:** tạo ra ngày dựa trên tham số năm/tháng/ngày được truyền vào theo định dạng datetime chuẩn.

Cú pháp: **DATE(<year>, <month>, <day>) Giá trị trả về:** trả về ngày cụ thể.

**Ví dụ**: Tôi muốn tạo ngày 29/09/2020

**DATE(2020,9,29)**

### DATEVALUE()

**Mục đích:** chuyển định dạng date chuẩn cho cột có giá trị văn bản. Cú pháp: **DATEVALUE(date\_text)**

**Giá trị trả về:** trả về ngày với định dạng date chuẩn.

**Ví dụ**: Tôi có cột thời gian (Date) được Power BI nhận diện theo định dạng text khi import, thực hiện chuyển đổi định dạng như sau:

**DATEVALUE(‘Date ’)**

### TIMEVALUE()

**Mục đích:** chuyển định dạng time chuẩn cho cột có giá trị văn bản. Cú pháp: **TIMEVALUE(time\_text)**

**Giá trị trả về:** trả về thời gian với định dạng time chuẩn.

**Ví dụ**: Tôi muốn trả về đúng định dạng time chuẩn cho chuỗi sau:

**TIMEVALUE(“12:49:55”)**

### TODAY()

**Mục đích:** lấy giá trị ngày hiện hành. Cú pháp: **TODAY()**

**Giá trị trả về:** trả về ngày hiện hành.

**Ví dụ**: Tôi muốn biết tuổi của người có năm sinh 2020 ở thời điểm hiện tại:

**YEAR(TODAY()) - 2020**

### WEEKNUM()

**Mục đích:** lấy thông tin số tuần dựa trên thông tin ngày đưa vào.

Cú pháp: **WEEKNUM(<date>, <return\_type>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần lấy thông tin số tuần |
| return\_type | Có 2 kiểu sau:  1: ngày đầu tiên của tuần làm việc là Chủ Nhật  2: ngày đầu tiên của tuần làm việc là Thứ Hai |

**Giá trị trả về:** số tuần thực tế của ngày chỉ định.

**Ví dụ**: Tôi muốn biết số tuần theo các ngày bán hàng:

**WEEKNUM(‘Sale ’[Ngay ban])**

### EDATE()

**Mục đích:** trả về ngày với số tháng được chỉ định trước. Sử dụng EDATE để tính ngày đáo hạn hoặc ngày đến hạn rơi vào cùng ngày trong tháng chỉ định.

Cú pháp: **EDATE(<start\_date>, <months>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| start\_date | Khai báo ngày chỉ định cần tham chiếu |
| months | Khai báo số tháng cần xác định |

**Giá trị trả về:** giá trị theo định dạng ngày.

**Ví dụ**: Tôi có ngày bán hàng như sau: 01/01/2017 -> 12/31/2017, vậy hàm sau sẽ trả về khoảng thời gian sau: 06/01/2017 -> 31/05/2018

**EDATE(‘Sale ’[Ngay ban],5)**

### EOMONTH()

**Mục đích:** trả về ngày theo định dạng ngày giờ của ngày cuối cùng của tháng được chỉ định. Sử dụng EOMONTH để tính ngày đáo hạn hoặc ngày đến hạn rơi vào ngày cuối cùng của tháng.

Cú pháp: **EOMONTH(<start\_date>, <months>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| start\_date | Khai báo ngày cần cần tham chiếu |
| months | Khai báo số tháng cần xác định |

**Giá trị trả về:** giá trị ngày cuối tháng theo định dạng ngày.

**Ví dụ**: Tôi có ngày bán hàng như sau: 01/01/2017 -> 12/31/2017, vậy hàm sau sẽ chỉ trả về ngày cuối tháng trong từng tháng: 04/30/2017 -> 31/05/2018

**EOMONTH(‘Sale ’[Ngay ban],3)**

### DAY()

**Mục đích:** trả về giá trị ngày của ngày chỉ định, giá trị trả về trong khoảng thời gian từ ngày 01

đến 31. 12.

Cú pháp: **DAY(<date>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần lấy giá trị năm |

**Giá trị trả về:** giá trị là số nguyên trong khoảng thời gian từ ngày 01 đến 31.

**Ví dụ**: Hàm sau trả về ngày 07:

**DAY(“05,07,2018”)**

### MONTH()

**Mục đích:** trả về giá trị tháng của ngày chỉ định, giá trị trả về trong khoảng thời gian từ tháng 01

đến tháng 12.

Cú pháp: **MONTH(<date>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần lấy giá trị năm |

**Giá trị trả về:** giá trị là số nguyên trong khoảng thời gian từ tháng 01 đến tháng 12

**Ví dụ**: Hàm sau trả về tháng 05:

**MONTH(“05,07,2018”)**

### YEAR()

**Mục đích:** trả về giá trị năm của ngày chỉ định, giá trị trả về trong khoảng thời gian từ năm 1900 đến 9999.

Cú pháp: **YEAR(<date>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần lấy giá trị năm |

**Giá trị trả về:** giá trị là số nguyên trong khoảng thời gian từ 1900 đến 9999.

**Ví dụ**: Hàm sau trả về năm 2018:

**YEAR(“05,07,2018”)**

### DATEADD()

**Mục đích:** trả về một bảng có chứa giá trị theo định dạng ngày chuẩn, giá trị sẽ tăng hoặc giảm dựa vào tham số **number\_of\_intervals** được đưa vào theo ngữ thực tế.

Cú pháp: **DATEADD(<dates>, <number\_of\_intervals>, <interval>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |
| number\_of\_intervals | Khai báo số nguyên xác định số lượng khoảng thời gian để cộng vào hoặc trừ đi |
| interval | Khoảng thời gian muốn thay đổi ngày tháng, hỗ trợ theo: year, quarter, month, day |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Hàm sau trả về ngày/tháng/năm của năm trước ngày trong ngữ cảnh hiện tại:

**DATEADD(‘Sale ’[Ngay ban], -1, YEAR)**

### DATESBETWEEN()

**Mục đích:** Trả về bảng chứa cột ngày bắt đầu bằng **start\_date** và tiếp tục cho đến **end\_date**. Cú pháp: **DATESBETWEEN(<dates>, <start\_date>, <end\_date>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| dates | Khai báo cột ngày cần tham chiếu |
| start\_date | Khai báo biểu thức cho ngày bắt đầu |
| end\_date | Khai báo biểu thức cho ngày kết thúc |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Tôi thực hiện tính doanh số bán hàng trong 3 tháng đầu năm 2018:

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

**DATESBETWEEN(‘Sale ’[Ngay ban],**

**DATE(2018,1,1),**

**DATE(2018,3,31)))**

### DATESINPERIOD()

Logo

Description automatically generated**Mục đích:** trả về một bảng có chứa giá trị theo định dạng ngày chuẩn có ngày bắt đầu bằng **start\_date** giá trị sẽ tăng hoặc giảm dựa vào tham số **number\_of\_intervals** được đưa vào theo ngữ thực tế.

Cú pháp: **DATESINPERIOD(<dates>,<start\_date>,<number\_of\_intervals>,<interval>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |
| start\_date | Khai báo biểu thức cho ngày bắt đầu |
| number\_of\_intervals | Khai báo số nguyên xác định số lượng khoảng thời gian để cộng vào hoặc trừ đi |
| interval | Khoảng thời gian muốn thay đổi ngày tháng, hỗ trợ theo: year, quarter, month, day |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Ngày chốt kỳ kế toán của công ty là 31/5/2018, tôi muốn tính doanh số bán hàng của 31 ngày trong tháng cuối kỳ:

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

**DATESINPERIOD(‘Sale ’[Ngay ban],**

**DATE(2018,5,31),**

**-31,**

**DAY))**

### DATESMTD()

Logo

Description automatically generated**Mục đích: T**rả về một bảng có chứa một cột của các ngày cho tháng đến nay. Cú pháp: **DATESMTD(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng. **Ví dụ**: Tôi muốn tính lũy kế doanh thu trong tháng 5: **CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

**DATESMTD(**

**FILTER(**

**MONTH(‘Sale ’[Ngay ban]) = 5)))**

DATESMTD()

**Mục đích: T**rả về một bảng có chứa một cột của các ngày cho tháng đến nay. Cú pháp: **DATESMTD(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng. **Ví dụ**: Tôi muốn tính lũy kế doanh thu trong tháng 5: **CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

**DATESMTD(**

**FILTER(**

**MONTH(‘Sale ’[Ngay ban]) = 5)))**

### DATESQTD()

Logo

Description automatically generated**Mục đích:** trả về một bảng có chứa một cột của các ngày cho quý đến nay. Cú pháp: **DATESMTD(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Tôi muốn tính lũy kế doanh thu theo quý:

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

**DATESMQTD(‘Sale ’[Ngay ban])**

### DATESYTD()

**Mục đích:** trả về một bảng có chứa một cột của các ngày cho năm đến nay. Cú pháp: **DATESYTD(<dates>, <year\_end\_date>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |
| year\_end\_date | Chuỗi ký tự có ngày xác định **year\_end\_date**.  Mặc định là ngày 31 tháng 12 (giá trị không bắt buộc khai báo) |

**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Tôi muốn tính lũy kế doanh thu theo năm:

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu]),**

Logo

Description automatically generated**DATESYTD(‘Sale ’[Ngay ban])**

### ENDOFMONTH()

**Mục đích:** trả về ngày theo định dạng ngày giờ của ngày cuối cùng của tháng được chỉ định. Cú pháp: **ENDOFMONTH(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần cần tham chiếu |

**Giá trị trả về:** giá trị ngày cuối tháng theo định dạng ngày.

**Ví dụ**: Tôi có ngày bán hàng như sau: 01/01/2017 -> 12/31/2017, vậy hàm sau sẽ chỉ trả về ngày cuối tháng trong từng tháng: 01/31/2017 -> 12/31/2017:

**ENDOFMONTH(‘Sale ’[Ngay ban])**

### ENDOFQUARTER()

**Mục đích:** trả về ngày theo định dạng ngày giờ của ngày cuối cùng của quý được chỉ định. Cú pháp: **ENDOFQUARTER(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần cần tham chiếu |

**Giá trị trả về:** giá trị ngày cuối quý theo định dạng ngày.

**Ví dụ**: Tôi thực hiện hàm trả về ngày cuối của từng quý trong năm của ngày bán hàng:

**ENDOFQUARTER(‘Sale ’[Ngay ban])**

### ENDOFYEAR()

Logo

Description automatically generated**Mục đích:** trả về ngày theo định dạng ngày giờ của ngày cuối cùng của năm được chỉ định. Cú pháp: **ENDOFYEAR(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo ngày cần cần tham chiếu |
| year\_end\_date | Chuỗi ký tự có ngày xác định **year\_end\_date**.  Mặc định là ngày 31 tháng 12 (giá trị không bắt buộc khai báo) |

**Ví dụ**: Tôi thực hiện hàm trả về ngày cuối của từng năm của ngày bán hàng:

**ENDOFYEAR(‘Sale ’[Ngay ban])**

### FIRSTDAY()

**Mục đích:** trả về ngày đầu tiên trong ngữ cảnh hiện tại cho cột ngày được chỉ định. Cú pháp: **FIRSTDAY(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn và một dòng có giá trị ngày tháng.

**Ví dụ**: Tôi thực hiện hàm trả về ngày đầu tiên của cột ngày bán (giá trị theo từng dòng):

**FIRSTDAY(‘Sale ’[Ngay ban])**

### FIRSTNONBLANK()

**Mục đích:** trả về giá trị đầu tiên trong cột được kiểm tra không có giá trị rỗng thông qua biểu thức đi cùng.

Cú pháp: **FIRSTNONBLANK(<column>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị ngày |
| Expression | Biểu thức kiểm tra giá trị rỗng |

**Giá trị trả về:** một bảng có chứa một cột và một hàng duy nhất với giá trị đầu tiên được trả về.

**Ví dụ**: Tôi muốn liệt kê những sản phẩm có doanh thu bán hàng tốt nhất, hàm sẽ được viết như

sau (có kết hợp TOPN hỗ trợ thống kê):

**FIRSTNONBLANK(**

**TOPN(**

**1)**

### LASTDAY()

**1,**

**VALUES('Product '[san pham]), [Doanh thu]),**

**Mục đích:** trả về ngày cuối trong ngữ cảnh hiện tại cho cột ngày được chỉ định. Cú pháp: **LASTDAY(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn và một dòng có giá trị ngày tháng.

**Ví dụ**: Tôi thực hiện hàm trả về ngày cuối của cột ngày bán (giá trị theo từng dòng):

**LASTDAY(‘Sale ’[Ngay ban])**

### LASTNONBLANK()

**Mục đích:** trả về giá trị cuối trong cột được kiểm tra không có giá trị rỗng thông qua biểu thức đi

cùng.

Cú pháp: **LASTNONBLANK(<column>, <expression>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột chứa giá trị ngày |
| Expression | Biểu thức kiểm tra giá trị rỗng |

**Giá trị trả về:** một bảng có chứa một cột và một hàng duy nhất với giá trị cuối được trả về.

**Ví dụ**: Tôi muốn lấy giá trị doanh thu bán hàng sau cùng có kết hợp kiểm tra giá trị rỗng của ngày phát sinh giao dịch bán hàng:

**CALCULATE(**

**SUM('Sale '[doanh thu]), LASTNONBLANK('Dates '[date],**

**CALCULATE(**

**SUM('Sale '[doanh thu])))**

### NEXTDAY()

**Mục đích:** trả về một bảng có chứa một cột của tất cả các ngày kế tiếp, dựa trên ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **NEXTDAY(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về ngày kế tiếp của ngày trong cột Ngày bán: **NEXTDAY(‘Sale ’[Ngay ban])**

### NEXTMONTH()

**Mục đích:** trả về một bảng có chứa một cột của tất cả các tháng kế tiếp, dựa trên ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **NEXTMONTH(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về tháng kế tiếp của tháng trong cột Ngày bán: **NEXTMONTH(‘Sale ’[Ngay ban])**

### NEXTQUARTER()

**Mục đích:** trả về một bảng có chứa một cột của tất cả các quý kế tiếp, dựa trên ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **NEXTQUARTER(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |

|  |  |
| --- | --- |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: hàm sau trả về quý kế tiếp của quý trong Ngày bán:

Logo

Description automatically generated**NEXTQUARTER(‘Sale ’[Ngay ban])**

### NEXTYEAR()

**Mục đích:** trả về một bảng có chứa một cột của tất cả các năm kế tiếp, dựa trên ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **NEXTYEAR(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: hàm sau trả về năm kế tiếp của năm trong cột Ngày bán:

**NEXTYEAR ‘Sale ’[Ngay ban]**

### PARALLELPERIOD()

**Mục đích:** trả về một bảng có chứa giá trị theo định dạng ngày chuẩn, giá trị sẽ tăng hoặc giảm dựa vào tham số **number\_of\_intervals** được đưa vào theo ngữ thực tế.

Cú pháp: **PARALLELPERIOD(<dates>, <number\_of\_intervals>, <interval>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

|  |  |
| --- | --- |
| number\_of\_intervals | Khai báo số nguyên xác định số lượng khoảng thời gian để cộng vào hoặc trừ đi |
| interval | Khoảng thời gian muốn thay đổi ngày tháng, hỗ trợ theo: year, quarter, month. |

Logo

Description automatically generated**Giá trị trả về:** một bảng có chứa một cột các giá trị ngày tháng.

**Ví dụ**: Tôi thực hiện tính tổng doanh thu bán hàng của năm trước dựa trên năm trong ngữ cảnh hiện tại:

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu’],**

**PARALLELPERIOD(‘Sale ’[Ngay ban], -1, YEAR)**

### PREVIOUSDAY()

**Mục đích:** trả về một bảng có chứa một cột của tất cả các ngày trước ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **PREVIOUSDAY(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về ngày trước của ngày trong cột Ngày bán: **PREVIOUSDAY(‘Sale ’[Ngay ban])**

### PREVIOUSMONTH()

**Mục đích:** trả về một bảng có chứa một cột của tháng trước ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Logo

Description automatically generatedCú pháp: **PREVIOUSMONTH(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về tháng trước của tháng trong cột Ngày bán: **PREVIOUSMONTH(‘Sale ’[Ngay ban])**

### PREVIOUSQUARTER()

**Mục đích:** trả về một bảng có chứa một cột của quý trước ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **PREVIOUSQUARTER(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: hàm sau trả về quý trước của quý trong cột Ngày bán:

**PREVIOUSQUARTER(‘Sale ’[Ngay ban])**

### PREVIOUSYEAR()

**Mục đích:** trả về một bảng có chứa một cột của năm trước ngày được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **PREVIOUSYEAR(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: hàm sau trả về năm trước của năm trong cột Ngày bán:

**PREVIOUSYEAR(‘Sale ’[Ngay ban])**

### SAMEPERIODLASTYEAR()

**Mục đích:** trả về một bảng có chứa cột giá trị cùng kỳ của năm trước của năm được chỉ định trong cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **SAMPERIODLASTYEAR(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: Tôi thực hiện tính tổng doanh thu bán hàng của năm trước dựa trên năm trong ngữ cảnh hiện tại (tính toán doanh số theo doanh thu cùng kỳ với năm hiện tại):

**CALCULATE(**

**SUM(‘Sale ’[Doanh thu’],**

**SAMPERIODLASTYEAR(‘Sale ’[Ngay ban])**

### STARTOFMONTH()

**Mục đích:** trả về ngày đầu tiên của tháng dựa trên cột ngày tháng trong ngữ cảnh hiện tại. Cú pháp: **STARTOFMONTH(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về ngày đầu tiên của tháng trong cột Ngày bán: **STARTOFMONTH(‘Sale ’[Ngay ban])**

### STARTOFQUARTER()

**Mục đích:** trả về ngày đầu tiên của quý dựa trên cột ngày tháng trong ngữ cảnh hiện tại. Cú pháp: **STARTOFQUARTER(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng.

**Ví dụ**: hàm sau trả về ngày đầu tiên của quý trong cột Ngày bán:

**STARTOFQUARTER (‘Sale ’[Ngay ban])**

### STARTOFYEAR()

**Mục đích:** trả về ngày đầu tiên của năm dựa trên cột ngày tháng trong ngữ cảnh hiện tại.

Cú pháp: **STARTOFYEAR(<dates>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| date | Khai báo cột chứa giá trị ngày |

Logo

Description automatically generated**Giá trị trả về:** một bảng có chứa một cột đơn có giá trị ngày tháng. **Ví dụ**: hàm sau trả về ngày đầu tiên của năm trong cột Ngày bán: **STARTOFYEAR(‘Sale ’[Ngay ban])**

## Nhóm hàm Bộ lọc (Filter Functions)

**Mô tả:** Nhóm hàm bộ lọc trong DAX bao gồm một số hàm phức tạp và mạnh mẽ nhất, có sự khác biệt rất nhiều so với các hàm Excel mà chúng ta quen dùng. Việc tra cứu thông qua các mối quan hệ giữa các bảng dữ liệu, từ đây cho phép bạn tạo ra các ngữ cảnh tra cứu linh động.

### Logo Description automatically generatedCALCULATE()

**Mục đích:** là loại hàm trong DAX thường dùng để lọc giá trị theo 1 biểu thức tính toán nào đó.

Cú pháp: **CALCULATE(<expression>, <filter1>, <filter2>,…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| Expression | Khai báo biểu thức tính toán |
| filter1, filter2 | Khai báo các điều kiện lọc dữ liệu cho biểu thức tính toán |

**Giá trị trả về:** trả về kết quả là giá trị từ biểu thức tính toán

**Ví dụ**: Tôi muốn lấy tổng doanh thu trong năm 2018 của 1 sản phẩm nào đó từ các thông số Doanh thu, Năm, Sản phẩm trong bảng Sale :

**CALCULATE(**

**SUM(‘Sale '[Doanh thu]), ’Sale '[Năm]=2018,**

**’Sale '[Sản phẩm]=”TV”)**

### ALL()

**Mục đích:** Trả về toàn bộ các dòng trong bảng hay toàn bộ giá trị trong cột chỉ định, bỏ qua thiết lập của các bộ lọc khác. Nói cách khác hàm này dùng trong ngữ cảnh bạn muốn tính toán toàn bộ giá trị mà không cần quan tâm đến việc giá trị có áp dụng bộ lọc nào trước đó.

Logo

Description automatically generatedCú pháp: **ALL( {<table> [ <column>[, <column>[, <column>[,…]]]} )**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bảng muốn bỏ qua các thiết lập lọc trước đó |
| column | Cột muốn bỏ qua các thiết lập lọc trước đó |

**Giá trị trả về:** bảng hay cột đã loại bỏ bộ lọc.

**Ví dụ**: Tôi muốn lấy tổng doanh thu của tất cả sản phẩm:

**CALCULATE(**

**SUM(‘Sale '[Doanh thu]), ALL(’Sale '[Sản phẩm]))**

### ALLEXCEPT()

**Mục đích:** Trả về toàn bộ các dòng trong bảng hay toàn bộ giá trị trong cột, bỏ qua thiết lập của các bộ lọc khác ngoại trừ bộ lọc được áp dụng cho những cột được chỉ định.

Cú pháp: **ALLEXCEPT( {<table> [ <column>[, <column>[, <column>[,…]]]} )**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bảng muốn bỏ qua các thiết lập lọc trước đó, bao gồm cả những cột muốn giữ lại bộ lọc |

|  |  |
| --- | --- |
| column | Cột muốn giữ lại các thiết lập lọc trước đó |

**Giá trị trả về:** bảng hay cột đã loại bỏ bộ lọc ngoại trừ những cột được khai báo trong hàm.

**Ví dụ**: Tôi muốn lấy tổng doanh thu của tất cả sản phẩm nhưng muốn giữ lại những thiết lập lọc

đã áp dụng cho năm bán hàng:

Logo

Description automatically generated**CALCULATE(**

**SUM(‘Sale '[Doanh thu]),**

**ALLEXCEPT(’Sale '[Năm]))**

### ALLNOBLANKROW()

**Mục đích:** Trả về toàn bộ các dòng trong bảng không có giá trị rỗng (Blank)

Cú pháp: **ALLNOBLANKROW( {<table> [ <column>[, <column>[, <column>[,…]]]} )**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bảng muốn thực hiện kiểm tra rỗng |
| column | Cột muốn thực hiện kiểm tra rỗng |

**Giá trị trả về:** bảng hay cột đã loại bỏ giá trị rỗng.

**Ví dụ**: Tôi muốn thực hiện đếm số dòng của bảng Bán hàng với điều kiện chỉ đếm những dòng có khai báo thông tin Ngày bán:

**COUNTROWS(**

**ALLNOBLANKROW(’Sale '[Ngay ban]))**

### FILTER()

**Mục đích:** Trả về bảng đại diện cho tập hợp con của một bảng hoặc biểu thức khác. Cú pháp: **FILTER(<table>, <filter>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bảng muốn thực hiện lọc |
| filter | Điều kiện lọc |

Logo

Description automatically generated**Giá trị trả về:** bảng chứa các dòng thỏa điều kiện lọc.

**Ví dụ**: Tôi muốn lấy tổng doanh thu của tất cả sản phẩm có giá bán> 3,000,000 VND:

**SUMX(**

**FILTER(‘Sale ,**

**‘Sale '[Gia ban] > 3,000,000), ‘Sale '[Doanh thu])**

### DISTINCT()

**Mục đích:** trả về bảng có cột chứa các giá trị duy nhất từ cột được chỉ định. Nói cách khác, các giá trị trùng lặp sẽ bị xóa và chỉ các giá trị duy nhất được trả về.

Cú pháp: **DISTINCT(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột cần lấy giá trị duy nhất. |

**Giá trị trả về:** cột chứa giá trị duy nhất.

**Ví dụ**: Tôi muốn biết có bao nhiêu khách hàng đã giao dịch:

**COUNTROWS(**

**DISTINCT(‘Sale ’[Khach hang]))**

### EARLIER()

Logo

Description automatically generated**Mục đích:** trả về giá trị hiện tại của cột được chỉ định. Cú pháp: **EARLIER(<column>, <number>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột cần lấy giá trị duy nhất. |
| number | Số lần đánh giá bên ngoài. |

**Giá trị trả về:** Giá trị hiện tại của dòng từ cột, tại số lần đánh giá bên ngoài.

**Ví dụ**: Tôi muốn thực hiện công thức xếp hạng khách hàng theo lợi nhuận bán hàng đạt được:

**COUNTROWS(**

**FILTER(‘Sale ’[Khach hang],**

**EARLIER(‘Sale ’[Loi nhuan] < ‘Sale ’[Loi nhuan]))**

**+ 1**

### HASONEFILTER()

**Mục đích:** trả về TRUE khi số lượng giá trị được lọc trực tiếp trên **columnName** là một, ngược lại trả về FALSE.

Cú pháp: **HASONEFILTER(<columnName>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| columnName | Khai báo cột cần kiểm tra. |

**Giá trị trả về:** trả về TRUE khi số lượng giá trị được lọc trực tiếp trên **columnName** là một, ngược lại trả về FALSE.

**Ví dụ**: Tôi muốn kiểm tra có giá trị nào được lọc trực tiếp, nếu có thì trả về giá trị đó ngược lại trả về BLANK().

**IF(**

Logo

Description automatically generated**HASONEFILTER(‘Product ’[Product ID], FILTERS(‘Product ’[Product ID], BLANK())**

### HASONEVALUE()

**Mục đích:** trả về TRUE khi số lượng giá trị được lọc trên **columnName** là một, ngược lại trả về FALSE.

Hàm này cũng tương tự như HASONEFILTER() nhưng áp dụng cho các trường hợp lọc phức tạp (cross-filters)

Cú pháp: **HASONEVALUE(<columnName>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| columnName | Khai báo cột cần kiểm tra. |

**Giá trị trả về:** trả về TRUE khi số lượng giá trị được lọc trên **columnName** là một, ngược lại trả về FALSE.

**Ví dụ**: Tôi muốn kiểm tra cột sau có giá trị không, nếu có thì trả về giá trị đó ngược lại trả về BLANK().

**IF(**

**HASONEVALUE(‘Product ’[Product ID], VALUES(‘Product ’[Product ID], BLANK())**

### VALUES ()

**Mục đích:** trả về bảng có cột chứa các giá trị duy nhất từ cột được chỉ định. Nói cách khác, các giá trị trùng lặp sẽ bị xóa và chỉ các giá trị duy nhất được trả về.

Cú pháp: **VALUES(<tableNameOrcolumnName>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| tableNameOrcolumnName | Khai báo bảng cột cần lấy giá trị duy nhất. |

Logo

Description automatically generated**Giá trị trả về:** cột chứa giá trị duy nhất.

**Ví dụ**: Tôi muốn biết có bao nhiêu đơn hàng đã thực hiện:

**COUNTROWS(**

**VALUES(‘Sale ’[OrderNumber]))**

### RELATED()

**Mục đích:** trả về một giá trị liên quan từ một bảng khác. Cú pháp: **RELATED(<column>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| column | Khai báo cột cần lấy giá trị. |

**Giá trị trả về:** một giá trị duy nhất có liên quan đến hàng hiện tại.

**Ví dụ**: Tôi muốn lọc những giao dịch trừ nhóm sản phẩm liên quan đến Microsoft Azure:

**FILTER('Sale '[OrderID],**

**RELATED('Product '[Nhom san pham]) <> "Microsoft Azure")**

## Nhóm hàm Thông tin (Information Functions)

**Mô tả:** nhóm hàm toán học sẽ thực hiện…

## CONTAINS()

Logo

Description automatically generated**Mục đích:** Trả về TRUE nếu tồn tại các giá trị cho tất cả các cột được đề cập hoặc được chứa trong các cột đó, ngược lại nếu không hàm trả về FALSE.

Cú pháp: **CONTAINS(<tables>),<columnName>,<value>[,columnName>,<value>]…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| table | Bảng muốn thực hiện tìm thông tin |
| columnName | Cột cần tìm giá trị |
| value | Nhập giá trị cần tìm |

**Giá trị trả về:** trả về TRUE nếu tồn tại các giá trị cho tất cả các cột được đề cập hoặc được chứa trong các cột đó, ngược lại nếu không hàm trả về FALSE.

**Ví dụ**: Tôi muốn tìm thông tin khách hàng ‘ICT24h’ đã mua sản phẩm “Office 365 Business”:

**CONTAINS(‘Sale ,**

**‘Sale '[Khach hang],“ICT24h”,**

**‘Sale '[San pham], “Office 365 Business”)**

## ISBLANK()

**Mục đích:** Trả về TRUE nếu tồn tại giá trị rỗng, ngược lại nếu không hàm trả về FALSE. Cú pháp: **ISBLANK(<value>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| value | Nhập giá trị hoặc biểu thức muốn kiểm tra rỗng. |

**Giá trị trả về:** Giá trị Boolean TRUE nếu giá trị trống; nếu không FALSE.

**Ví dụ**: Tôi có hàm tính tổng doanh thu năm 2017 (Doanhthu\_2017), tôi muốn tính tỷ lệ doanh thu cùng kỳ năm nay so với năm trước nhưng muốn kiểm tra nếu doanh thu năm trước có giá trị rỗng thì trả về BLANK(), ngươc lại sẽ thực hiện tính tỷ lệ doanh thu:

**IF(**

Logo

Description automatically generated**ISBLANK(Doanhthu\_2017, BLANK(),**

**(Doanhthu\_2018 – Doanhthu\_2017/Doanhthu\_2017)**

## ISERROR()

**Mục đích:** Trả về TRUE nếu tồn tại giá trị bị lỗi, ngược lại nếu không hàm trả về FALSE. Cú pháp: **ISERROR(<value>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| value | Nhập giá trị hoặc biểu thức muốn kiểm tra lỗi. |

**Giá trị trả về:** Giá trị Boolean TRUE nếu giá trị lỗi; nếu không FALSE.

**Ví dụ**: Tương tự ví dụ trên, tôi muốn kiểm tra có lỗi không – thường là kiểm tra xem có chia cho giá trị 0 – nếu có giá trị lỗi thì trả về BLANK(), ngươc lại sẽ thực hiện tính tỷ lệ doanh thu:

**IF(ISERROR(**

**Doanhthu\_2018/Doanhthu\_2017 BLANK(),**

**Doanhthu\_2018/Doanhthu\_2017)**

## ISNUMBER()

**Mục đích:** Trả về TRUE nếu tồn tại giá trị là số, ngược lại nếu không hàm trả về FALSE.

Cú pháp: **ISNUMBER(<value>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| value | Nhập giá trị hoặc biểu thức muốn kiểm tra. |

**Giá trị trả về:** Giá trị Boolean TRUE nếu giá trị là số; nếu không FALSE.

Logo

Description automatically generated**Ví dụ**:

**IF(ISNUMBER(1980), “Giá trị là số”, “Không phải số”)**

→ Kết quả trả về: “Giá trị là số”

**IF(ISNUMBER(“1988”), “Giá trị là số”, “Không phải số”)**

→ Kết quả trả về: “Không phải số”

## ISTEXT()

**Mục đích:** Trả về TRUE nếu tồn tại giá trị là ký tự, ngược lại nếu không hàm trả về FALSE. Cú pháp: **ISTEXT(<value>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| value | Nhập giá trị hoặc biểu thức muốn kiểm tra. |

**Giá trị trả về:** Giá trị Boolean TRUE nếu giá trị là ký tự; nếu không FALSE.

**Ví dụ**:

**IF(ISTEXT(1980), “Giá trị là ký tự”, “Không phải ký tự”)**

→ Kết quả trả về: “Không phải ký tự”

**IF(ISTEXT(“1988”), “Giá trị là ký tự”, “Không phải ký tự”)**

→ Kết quả trả về: “Giá trị là ký tự”

## LOOKUPVALUE()

**Mục đích:** Trả về các dòng có giá trị thỏa điều kiện tìm kiếm.

Cú pháp: **LOOKUPVALUE(<result\_columnName>,<search\_columnName>,<search\_**

**value>[,<search\_columnName>,<search\_value>]…)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| result\_columnName | Cột sẽ trả về kết quả các dòng nếu thỏa điều kiện truy vấn |
| search\_columnName | Khai báo cột có giá trị cần truy vấn |
| Search value | Nhập giá trị hoặc biểu thức cần truy vấn |

Logo

Description automatically generated**Giá trị trả về:** các dòng trong cột result\_columnName nếu thỏa điều kiện có cột & có giá trị cần tìm.

**Ví dụ**: Tìm những sản phẩm thuộc nhóm sản phẩm có mã “Microsoft Office 365”

**LOOPUPVALUE(**

**‘Sale '[San pham],**

**‘Sale '[Nhom san pham], “Microsoft Office 365”)**

## Nhóm hàm Logic (Logical Functions)

**Mô tả:** nhóm hàm logic hoạt động khi một biểu thức trả về thông tin về các giá trị hoặc các tập hợp trong biểu thức. Ví dụ, bạn có thể sử dụng hàm IF để kiểm tra kết quả của một biểu thức và tạo các kết quả có điều kiện.

## Logo Description automatically generatedAND()

**Mục đích:** Kiểm tra xem cả hai đối số đều đúng và hợp lệ thì trả về TRUE, nếu không trả về false. Cú pháp: **AND(<logical1>, <logical2>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| logical1, logical2 | Nhập giá trị logic muốn kiểm tra. |

**Giá trị trả về:** Giá trị Boolean TRUE hay FALSE dựa trên sự kết hợp các giá trị mà bạn muốn kiểm tra.

**Ví dụ**:

**IF(AND(10 > 9, -10 < -1), “Tất cả đều đúng”, “Có giá trị sai”)**

Kết quả trả về: “Tất cả đều đúng”

## OR()

**Mục đích:** Tương tự như AND() nhưng chỉ trả về giá trị FALSE khi cả 2 đối số có giá trị sai. Cú pháp: **OR(<logical1>, <logical2>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| logical1, logical2 | Nhập giá trị logic muốn kiểm tra. |

**Giá trị trả về:** Giá trị Boolean TRUE hay FALSE dựa trên sự kết hợp các giá trị mà bạn muốn kiểm tra.

**Ví dụ**:

Các hàm sau cho ra kết quả: “Tất cả đều đúng”

Logo

Description automatically generated**IF(OR(10 > 9, -10 < -1), “Tất cả đều đúng”, “Có giá trị sai”) IF(OR(10 > 9, -10 > -1), “Tất cả đều đúng”, “Có giá trị sai”) IF(OR(10 < 9, -10 < -1), “Tất cả đều đúng”, “Có giá trị sai”)** Hàm sau cho kết quả: “Có giá trị sai”

**IF(OR(10 <9, -10 > -1), “Tất cả đều đúng”, “Có giá trị sai”)**

## IF()

Mục đích: Kiểm tra xem một điều kiện được cung cấp như là đối số đầu tiên được đáp ứng. Trả về một giá trị nếu điều kiện là TRUE và trả về một giá trị khác nếu điều kiện là FALSE.

Cú pháp: **IF(logical\_test>, <value\_if\_true>, value\_if\_false)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| logical\_test | Bất kỳ giá trị hoặc biểu thức nào có thể được kiểm tra là TRUE hoặc FALSE. |
| value\_if\_true | Giá trị được trả về nếu kết quả kiểm tra là TRUE. Nếu bỏ qua,  TRUE được trả về. |
| value\_if\_false | Giá trị được trả về nếu kết quả kiểm tra là FALSE. Nếu bỏ qua, FALSE được trả về. |

**Giá trị trả về:** Bất kỳ giá trị nào có thể được trả lại bằng một biểu thức.

**Ví dụ**: Tôi muốn thực hiện phân loại Khách hàng theo điều kiện Doanh thu sau:

* Doanh thu < 5,000,000 VND sẽ đưa vào nhóm Đạt
* Doanh thu < 3,000,000 VND sẽ đưa vào nhóm Không Đạt
* Ngoài ra, doanh thu vượt mức trên sẽ đưa vào nhóm Tốt.

**IF(‘Sale '[Doanh thu] < 3,000,000, “Không Đạt”,**

**IF((‘Sale '[Doanh thu] < 5,000,000, “Đạt”, “Tốt”))**

## SWITCH()

Mục đích: Đánh giá một biểu thức dựa vào danh sách các giá trị và trả về một trong nhiều biểu thức kết quả có thể có.

Logo

Description automatically generatedCú pháp: **SWITCH(<expression>, <value>, <result>[,<value>,<result>]…[,<else>])**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| expression | Bất kỳ biểu thức DAX nào trả về một giá trị vô hướng duy nhất trong đó biểu thức được đánh giá nhiều lần (cho mỗi hàng / ngữ cảnh). |
| value | Giá trị không đổi được so khớp với kết quả biểu thức. |
| result | Bất kỳ biểu thức vô hướng nào được đánh giá nếu kết quả của biểu thức khớp với giá trị tương ứng. |
| else | Bất kỳ biểu thức vô hướng nào được đánh giá nếu kết quả của biểu thức không khớp với bất kỳ đối số giá trị nào. |

**Giá trị trả về:** Giá trị vô hướng đến từ một trong các biểu thức kết quả, nếu có giá trị khớp hoặc từ biểu thức khác (VD: thông báo) nếu không khớp với bất kỳ giá trị nào.

**Ví dụ**: Tôi muốn thực hiện thêm cột thể hiện tên của tháng tương ứng với tháng cụ thể (theo số):

**SWITCH([Month],**

**1, ”January”,**

**2, ”February”,**

**3, ”March”,**

**4, ”April”,**

**5, ”May”,**

**6, ”June”,**

**7, ”July”,**

**8, ”August”,**

**9, ”September”,**

**10, ”October”,**

**11, ”November”, 12,”December”**

**,”Giá trị không xác định”)**

## Logo Description automatically generatedNOT()

**Mục đích:** Chuyển đổi trạng thái giá trị Boolean từ TRUE sang FALSE và ngược lại. Cú pháp: **NOT(<logical1>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| logical1 | Nhập giá trị hoặc biểu thức muốn đổi trạng thái. |

**Giá trị trả về:** Giá trị Boolean TRUE hay FALSE.

**Ví dụ**:

Tôi muốn chuyển trạng thái TRUE của hàm có giá trị đúng sau thành FALSE

**NOT(**

**IF(10 > 9, “TRUE”, “FALSE”))**

## IFERROR()

**Mục đích:** Thường dùng trong trường hợp bẫy lỗi giá trị hay biểu thức, thay giá trị lỗi bằng giá trị hay biểu thức mong muốn.

Cú pháp: **IFERROR(<value>, < value\_if\_error>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| value | Nhập giá trị hoặc biểu thức muốn kiểm tra |

|  |  |
| --- | --- |
| value\_if\_error | Nhập giá trị hoặc biểu thức trả về nếu có lỗi xảy ra |

**Giá trị trả về:** Giá trị vô hướng có cùng kiểu với tham số value.

**Ví dụ**:

Kết quả sẽ là 999999 nếu biểu thức xảy ra lỗi:

Logo

Description automatically generated**IFERROR(10/0, 999999)**

# Nhóm hàm Text (Text Functions)

**Mô tả:** nhóm hàm text trả về một phần của chuỗi, tìm kiếm văn bản trong một chuỗi hoặc ghép các ký tự.

## BLANK()

Logo

Description automatically generated**Mục đích:** Trả về giá trị rỗng. Cú pháp: **BLANK()**

**Giá trị trả về:** Giá trị rỗng.

**Ví dụ**: Kiểm tra nếu Doanh thu năm 2017 có giá nhỏ hơn hoặc bằng 0 thì trả về Blank, ngược lại tiến hành thực hiện phép chia.

**IF(Doanhthu\_2017 <= 0, BLANK(),**

**Doanhthu\_2018/Doanhthu\_2017)**

## FIND()

**Mục đích:** Trả về vị trí bắt đầu của một chuỗi văn bản trong một chuỗi văn bản khác. FIND phân biệt chữ hoa chữ thường.

Cú pháp: **FIND(<find\_text>,<within\_text>[,[<start\_num>][,<NotFoundValue>]])**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| find\_text | Chuỗi văn bản cần tìm, có thể dùng các ký tự đặc biệt: (?), (\*) theo  quy định sau:   * Dấu chấm hỏi khớp với bất kỳ ký tự đơn nào * Dấu sao thị khớp với bất kỳ chuỗi ký tự nào |
| within\_text | Chuỗi văn bản có chứa chuỗi hay ký tự cần tìm. |
| start\_num | Các ký tự mà tại đó để bắt đầu tìm kiếm; nếu bỏ qua, start\_num =  1. Ký tự đầu tiên trong within\_text là ký tự số 1. |

|  |  |
| --- | --- |
| NotFoundValue | Giá trị trả về trong trường hợp không tìm thấy chuỗi văn bản, ký tự khớp với nội dung cần tìm. Thông thường là: 0, -1, hoặc BLANK(). |

**Ví dụ**: tìm vị trí của chữ cái đầu tiên của tên Sản phẩm “Office 365”, trong chuỗi chứa mô tả sản phẩm.

**FIND(“Office 365”,”Microsoft Office 365 Enterprise E5”)**

## CONCATENATE()

**Mục đích:** Nối 2 chuỗi lại với nhau.

Cú pháp: **CONCATENATE(<text1>, <text2>)**

|  |  |
| --- | --- |
| **Tham số** | **Mô tả** |
| text1, text2 | Khai báo các chuỗi ký tự hoặc số cần nối lại với nhau. Có thể áp dụng cho cột tham chiếu |

**Kết quả trả về:** chuỗi đã nối.

**Ví dụ**: nối 2 chuỗi sau:

#### CONCATENATE(“ ”, “a member of ICT24h TEAM”)