



**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA HỆ THÔNG THÔNG TIN**



**BÁO CÁO ĐỒ ÁN MÔN HỌC  
KHO DỮ LIỆU VÀ OLAP**

**ĐỀ TÀI: PHÂN TÍCH DỮ LIỆU GIAO DỊCH THẺ TÍN DỤNG**

Lớp: IS217.P11

Giảng viên hướng dẫn: ThS. Đỗ Thị Minh Phụng

Sinh viên thực hiện:

Trần Đại Hiển – 22520426@gm.uit.edu.vn

Trần Lương Văn Nhi – 22521044@gm.uit.edu.vn

# MỤC LỤC

1. Giới thiệu tổng quan .....	6
1.1 Lý do chọn đề tài.....	6
1.2 Giới thiệu về dataset .....	6
1.2.1 Mô tả về dataset .....	6
1.2.2 Lý do chọn bộ dữ liệu .....	6
1.2.3 Nguồn kho dữ liệu.....	6
1.2.5 Mô tả chi tiết các cột thuộc tính sử dụng .....	6
1.3 Thiết kế kho dữ liệu .....	7
1.3.1 Lược đồ kho dữ liệu .....	7
1.3.2.1 Bảng fact_transaction .....	8
1.3.2.2 Bảng dim_cardholder .....	8
1.3.2.3 Bảng dim_datetime .....	8
1.3.2.4 Bảng dim_category .....	8
1.3.2.6 Bảng dim_merchant.....	9
2. Nội dung câu truy vấn .....	9
<b>PHẦN 2: QUÁ TRÌNH XÂY DỰNG KHO DỮ LIỆU (SSIS) .....</b>	<b>10</b>
2.1 Chuẩn bị công cụ.....	10
2.2 Chuẩn bị cơ sở dữ liệu và tạo project SSIS .....	10
2.3 Tạo mới project SSIS.....	11
2.4 Tạo các connection.....	12
2.4.1 Flat file connection.....	12
2.4.2 OLE DB connection .....	15
2.5 Cleaning and Staging data .....	17
2.5.1 Cleaning data.....	17
2.5.2 Tạo các bảng Dim và Fact.....	24
2.5.2.1 Tạo bảng DimDate .....	24
2.5.2.2 Tạo bảng DimCardHolder .....	28
2.5.2.3 Tạo bảng DimMerchant .....	31
2.5.2.4 Tạo bảng DimCategory .....	34
2.5.2.5 Tạo bảng StagingFact.....	36
2.6 Merge Table.....	39
2.7 Create Constraint.....	59
2.8 Chạy dự án SSIS .....	61
<b>PHẦN 3: THỰC HIỆN PHÂN TÍCH TRỰC TUYẾN TRÊN KHO DỮ LIỆU (SSAS) .....</b>	<b>63</b>
3.1 Tạo project SSAS mới .....	63

3.2 Xác định dữ liệu nguồn (Data Sources) .....	64
3.3 Xác định khung nhìn dữ liệu nguồn (Data Source View) .....	67
3.4. Xây dựng các khối (Cube) và deploy Cube.....	70
3.4.1. Tạo Cube và Dimension.....	70
3.4.2. Thêm thuộc tính cho Dim .....	72
3.4.3 Tạo hierarchy cho chiều thời .....	73
3.5 Chạy project SSAS.....	81
3.6 Thực hiện các câu truy vấn sử dụng ngôn ngữ MDX, Manual và Pivot Excel .....	82
3.6.1 Câu 1: Tổng số lượng giao dịch và tổng tiền giao dịch theo từng năm của từng bưu cục/cửa hàng .....	82
3.6.2 Câu 2: Số tiền giao dịch trung bình tại các bưu cục/cửa hàng.....	83
3.6.3 Câu 3: Top 5 nghề nghiệp có số lượng giao dịch nhiều nhất trong năm 2020 .....	85
3.6.4 Câu 4: Top 5 thành phố có nhiều giao dịch nhất .....	87
3.6.5 Câu 5: Trung bình số tiền giao dịch của các chủ thẻ có nghề nghiệp là Science writer, Systems analyst, và Hospital pharmacist trong năm 2020 và sống tại bang có mã là “NY”, “CA” .....	88
3.6.6 Câu 6: Top 10 tên chủ thẻ có tổng số giao dịch cao nhất ở hạng mục category là gas_transport ..	90
3.6.7 Câu 7: Thống kê số lượng giao dịch, tổng số tiền giao dịch và trung bình số tiền giao dịch theo giờ trong ngày .....	92
3.6.8 Câu 8: Tính toán số lượng giao dịch theo khoảng cách giữa địa điểm giao dịch và địa chỉ khách hàng lớn hơn 100 km. ....	93
3.6.9 Câu 9: Tính toán tổng số lượng giao dịch, tổng số tiền giao dịch và số tiền giao dịch trung bình theo giới tính là nữ và danh mục sản phẩm/dịch vụ là shopping_pos .....	96
3.6.10 Câu 10: Thống kê các chủ thẻ có giao dịch trung bình lớn hơn 750, xếp theo thứ tự giảm dần của giá trị giao dịch trung bình .....	97
3.6.11 Câu 11: Tổng số giao dịch của các chủ thẻ có năm sinh sau 2000 .....	98
3.6.12 Câu 12: Với mỗi thành phố, lấy ra được khách hàng có tổng số tiền giao dịch lớn nhất .....	100
3.6.13 Câu 13: Thống kê phân bố tổng giá trị giao dịch theo từng category, xếp theo thứ tự bảng chữ cái của danh mục .....	103
3.6.14 Câu 14: Thống kê tổng số lần giao dịch và tổng số tiền giao dịch theo từng quý trong năm 2020 của category “home” hoặc “kids_pets” theo giới tính .....	106
3.6.15 Câu 15: Thống kê tổng số giao dịch và tổng số tiền theo category của từng nghề nghiệp.....	108
<b>PHẦN 4: QUÁ TRÌNH TẠO LẬP BÁO CÁO .....</b>	<b>110</b>
4.1 Chuẩn bị công cụ.....	110
4.2 Trích xuất dữ liệu từ kho dữ liệu dưới dạng .csv .....	111
4.3 Thực hiện tạo bảng biểu bằng Power BI.....	112
4.3.1 Tạo mới báo cáo và upload dữ liệu vào Power BI.....	112
4.3.2 Báo cáo 1 .....	117
4.3.3 Báo cáo 2 .....	119
4.3.4 Báo cáo 3 .....	121
4.4 Thực hiện tạo báo cáo bằng Looker Studio.....	123
4.3.1 Quá trình upload dữ liệu vào Looker Studio.....	123

4.3.2 Báo cáo 1 .....	125
4.3.3 Báo cáo 2 .....	128
4.3.4 Báo cáo 3 .....	130
<b>PHẦN 5: QUÁ TRÌNH KHAI PHÁ DỮ LIỆU (DATA MINING) .....</b>	<b>133</b>
<b>5.1 Khám phá dữ liệu và tiền xử lý dữ liệu.....</b>	<b>133</b>
<b>5.2 Mã hóa và lựa chọn thuộc tính .....</b>	<b>138</b>
<b>5.2.1 Mã hóa thuộc tính .....</b>	<b>138</b>
<b>5.2.2 Lựa chọn thuộc tính.....</b>	<b>139</b>
<b>5.3 Lựa chọn mô hình .....</b>	<b>142</b>
<b>5.3.1 Decision Tree .....</b>	<b>142</b>
<b>5.3.1.1 Các bước hoạt động của thuật toán Decision Tree: .....</b>	<b>142</b>
<b>5.3.1.2 Ưu điểm và nhược điểm của Decision Tree:.....</b>	<b>142</b>
<b>5.3.1.3 Ứng dụng Decision Tree trong Fraud Detection .....</b>	<b>142</b>
<b>5.3.2 KNN.....</b>	<b>143</b>
<b>5.3.2.1 Các bước hoạt động của thuật toán KNN:.....</b>	<b>143</b>
<b>5.3.2.2 Ưu điểm và nhược điểm của KNN: .....</b>	<b>144</b>
<b>5.3.2.3 Ứng dụng trong Fraud Detection: .....</b>	<b>144</b>
<b>5.3.3 LightGBM.....</b>	<b>144</b>
<b>5.3.3.1 Gradient Boosting Decision Tree (GBDT).....</b>	<b>144</b>
<b>5.3.3.2 Gradient-based One-Side Sampling (GOSS).....</b>	<b>145</b>
<b>5.3.3.3 Exclusive Feature Bundling (EFB).....</b>	<b>145</b>
<b>5.3.3.4 Ứng dụng LightGBM trong Fraud Detection.....</b>	<b>145</b>
<b>5.4 Độ đo sử dụng.....</b>	<b>146</b>
<b>5.4.1 Accuracy (Độ chính xác).....</b>	<b>146</b>
<b>5.4.2 Precision (Độ chính xác khi dự đoán dương) .....</b>	<b>146</b>
<b>5.4.3 Recall (Độ nhạy).....</b>	<b>146</b>
<b>5.4.4 F1 Score.....</b>	<b>147</b>
<b>5.5 Chiến thuật tối ưu hóa tham số.....</b>	<b>147</b>
<b>5.5.1 RandomizedSearchCV.....</b>	<b>147</b>
<b>5.5.2 Các tham số tối ưu .....</b>	<b>147</b>
<b>5.5.2.1 Decision Tree:.....</b>	<b>148</b>
<b>5.5.2.2 K-Nearest Neighbors (KNN) .....</b>	<b>148</b>
<b>5.5.2.3 LightGBM.....</b>	<b>149</b>
<b>5.6 Các bước thực hiện .....</b>	<b>149</b>
<b>5.7 Đánh giá và so sánh mô hình .....</b>	<b>151</b>
<b>5.7.1 Kết quả:.....</b>	<b>151</b>
<b>5.7.2 So sánh kết quả trước và sau khi tối ưu hóa tham số .....</b>	<b>151</b>
<b>5.7.2.1 Decision Tree .....</b>	<b>152</b>

5.7.2.2 K-Nearest Neighbors (KNN) .....	152
5.7.2.3 LightGBM.....	152
5.7.3 Nhận xét tổng quan và đề xuất mô hình .....	152
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>153</b>

## **1. Giới thiệu tổng quan**

### **1.1 Lý do chọn đề tài**

Thẻ tín dụng (credit card) là một loại thẻ thanh toán được cung cấp bởi các ngân hàng hoặc tổ chức tài chính khác, cho phép người sử dụng mượn tiền từ ngân hàng để thực hiện các giao dịch mua sắm hoặc thanh toán dịch vụ. Xu hướng sử dụng thẻ tín dụng ngày nay đang phát triển mạnh mẽ nhờ sự tiện lợi, an toàn, và tích hợp nhiều công nghệ mới đáp ứng nhu cầu đa dạng và ngày càng cao của người tiêu dùng trong bối cảnh hiện đại hóa và số hóa toàn cầu.

Việc có thể hiểu rõ thông tin và có khả năng phân tích dữ liệu giao dịch thẻ tín dụng là yếu tố quan trọng đối với các tổ chức doanh nghiệp tài chính. Phân tích dữ liệu giúp doanh nghiệp hiểu rõ hơn về thói quen hành vi chi tiêu của khách hàng, từ đó đưa ra được những insight chính xác giúp doanh nghiệp có thể tối ưu hóa dịch vụ. Bên cạnh đó, việc nắm bắt tốt thông tin quản lý có thể giúp phát hiện các gian lận tài chính, giao dịch bất thường để có thể tăng cường an ninh hệ thống và đưa ra các hỗ trợ nhanh chóng cho khách hàng.

Kho dữ liệu phân tích về giao dịch có thể giúp doanh nghiệp theo dõi và quản lý một lượng lớn dữ liệu một cách hiệu quả. Nhóm chúng em nghĩ rằng đề tài "Phân tích dữ liệu giao dịch thẻ tín dụng" là một lựa chọn thiết thực và có giá trị ứng dụng cao thực tiễn.

### **1.2 Giới thiệu về dataset**

#### **1.2.1 Mô tả về dataset**

Tên bộ dữ liệu: Credit Card Transaction D

Tác giả: Ryan Lee

Bộ dữ liệu giao dịch thẻ tín dụng cung cấp hồ sơ chi tiết về các giao dịch thẻ tín dụng từ ngày 01 tháng 01 năm 2019 đến ngày 10 tháng 03 năm 2020, thông tin bao gồm thời gian giao dịch, số tiền và thông tin chi tiết về cá nhân và thương gia liên quan

#### **1.2.2 Lý do chọn bộ dữ liệu**

Tuy bộ dữ liệu này chỉ lưu trữ các dữ liệu giao dịch thuộc giai đoạn 2019 - 2020 nhưng nhóm vẫn đánh giá đây vẫn là bộ dữ liệu có giá trị lớn trong việc phân tích. Dữ liệu này cung cấp thông tin lịch sử về hành vi chi tiêu, xu hướng giao dịch trong một giai đoạn cụ thể, từ đó hỗ trợ nhóm nghiên cứu rút ra các bài học. Hơn nữa, các đặc điểm giao dịch thẻ tín dụng thường có tính ổn định tương đối qua thời gian, nên việc phân tích dữ liệu cũ vẫn có thể mang lại những phân tích hữu ích có thể áp dụng cho hiện tại.

#### **1.2.3 Nguồn kho dữ liệu**

Bộ dữ liệu được lấy từ nền tảng Kaggle

#### **1.2.4 Thuộc tính kho dữ liệu**

Dữ liệu gồm 1 296 675 dòng và 24 thuộc tính

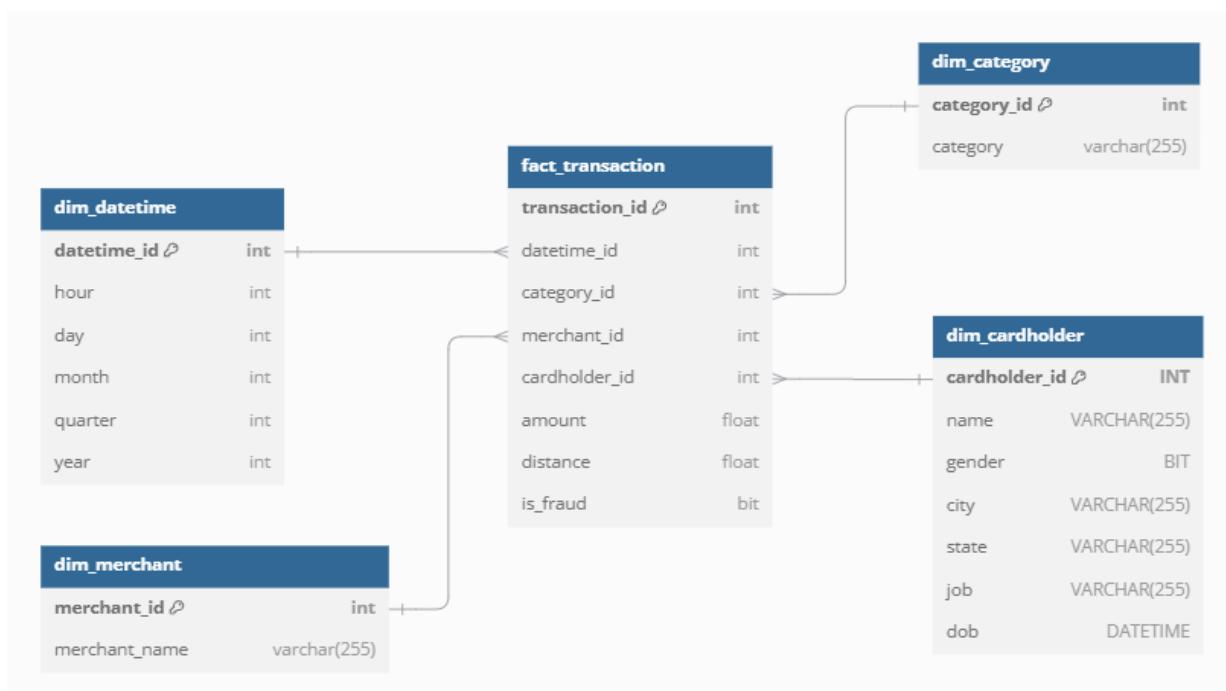
#### **1.2.5 Mô tả chi tiết các cột thuộc tính sử dụng**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	trans_date trans_time	DATETIME	Thời gian phát sinh giao dịch
2	cc_num	VARCHAR (50)	Số thẻ tín dụng

3	merchant	VARCHAR (255)	Tên thương nhân hoặc tên cửa hàng nơi giao dịch diễn ra
4	category	VARCHAR (255)	Loại giao dịch (VD: gas_transport, entertainment, health fitness, ....)
5	amt	FLOAT	Số tiền giao dịch
6	first	VARCHAR (255)	Tên của chủ thẻ
7	last	VARCHAR (255)	Họ của chủ thẻ
8	gender	BIT	Giới tính của chủ thẻ (1: Nam, 0: Nữ)
9	street	VARCHAR (255)	Tên địa chỉ của chủ thẻ
10	city	VARCHAR (255)	Thành phố trong địa chỉ của chủ thẻ
11	state	VARCHAR (255)	Tiểu bang trong địa chỉ của chủ thẻ
12	zip	VARCHAR (255)	Mã bưu điện của chủ thẻ
13	lat	DOUBLE	Tọa độ vĩ độ của giao dịch
14	long	DOUBLE	Tọa độ kinh độ của giao dịch
15	city pop	INT	Dân số của thành phố nơi giao dịch diễn ra
16	job	VARCHAR (255)	Nghề nghiệp của chủ thẻ
17	dob	DATETIME	Ngày sinh chủ thẻ
18	trans num	VARCHAR (255)	Mã giao dịch
19	unix time	BIGINT	thời gian giao dịch ở dạng unix
20	merch lat	DOUBLE	Tọa độ vĩ độ của thương nhân hay cửa hàng
21	merch long	DOUBLE	Tọa độ kinh độ của thương nhân hay cửa hàng
22	is fraud	BIT	Nhận đánh dấu xem giao dịch có phải là gian lận hay không (1: có, 2: không)
23	merch zipcode	VARCHAR (255)	Mã bưu điện của thương nhân hay cửa hàng

### 1.3 Thiết kế kho dữ liệu

#### 1.3.1 Lược đồ kho dữ liệu



### 1.3.2.1 Bảng fact\_transaction

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
Khóa chính	transaction_id	int	Khóa
	datetime_id	int	Mã ngày tháng năm
	category_id	int	Mã loại giao dịch
	merchant_id	int	Mã cửa hàng
	cardholder_id	int	Mã chủ thẻ
	transaction_location_id	int	Mã địa điểm giao dịch
	amount	float	Số tiền giao dịch
	distance	float	Khoảng cách
	is_fraud	bit	Gian lận (0: không, 1: có)

Trong đó: cột distance được tính bởi khoảng cách giữa tọa độ xảy ra giao dịch (lat, long) và tọa độ của cửa hàng (merch\_lat, merch\_long)

### 1.3.2.2 Bảng dim\_cardholder

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
Khóa chính	cardholder_id	int	Mã chủ thẻ
	name	varchar(255)	Họ và tên của chủ thẻ
	gender	varchar(1)	Giới tính của chủ thẻ (M: Nam, F: Nữ)
	city	varchar(255)	Thành phố trong địa chỉ của chủ thẻ
	state	varchar(255)	Tiểu bang trong địa chỉ của chủ thẻ
	job	varchar(255)	Nghề nghiệp của chủ thẻ
	dob	datetime	Ngày sinh của chủ thẻ

### 1.3.2.3 Bảng dim\_datetime

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
Khóa chính	datetime_id	int	Mã thời gian
	hour	int	Giờ
	day	int	Ngày
	month	int	Tháng
	quarter	int	Quý
	year	int	Năm

### 1.3.2.4 Bảng dim\_category

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
Khóa chính	category_id	int	Mã loại giao dịch

	category	varchar (255)	Loại giao dịch
--	----------	---------------	----------------

### 1.3.2.6 Bảng dim\_merchant

Khóa	Tên thuộc tính	Kiểu dữ liệu	Mô tả
Khóa chính	merchant_id	int	Mã cửa hàng
	merchant_name	varchar (255)	Tên cửa hàng

## 2. Nội dung câu truy vấn

Câu 1: Tổng số lượng giao dịch và tổng tiền giao dịch theo từng năm của từng bưu cục/cửa hàng

Câu 2: Số tiền giao dịch trung bình tại các bưu cục/cửa hàng

Câu 3: Top 5 nghề nghiệp có số lượng giao dịch nhiều nhất trong năm 2020

Câu 4: Top 5 thành phố có nhiều giao dịch nhất

Câu 5: Trung bình số tiền giao dịch của các chủ thẻ có nghề nghiệp là Science writer, Systems analyst, và Hospital pharmacist trong năm 2020 và sống tại bang có mã là "NY", "CA"

Câu 6: Top 10 tên chủ thẻ có tổng số giao dịch cao nhất ở hạng mục category là gas\_transport

Câu 7: Thống kê số lượng giao dịch, tổng số tiền giao dịch và trung bình số tiền giao dịch theo giờ trong ngày

Câu 8: Thống kê số giao dịch mà khoảng cách giữa địa điểm giao dịch và địa chỉ khách lớn 100km

Câu 9: Tính toán tổng số lượng giao dịch, tổng số tiền giao dịch và số tiền giao dịch trung bình theo giới tính là nữ và danh mục sản phẩm/dịch vụ là shopping\_pos

Câu 10: Thống kê các chủ thẻ có giao dịch trung bình lớn hơn 750, xếp theo thứ tự giảm dần của giá trị giao dịch trung bình

Câu 11: Tổng số giao dịch của các chủ thẻ có năm sinh sau 2000

Câu 12: Với mỗi thành phố, lấy ra được khách hàng có tổng số tiền giao dịch lớn nhất

Câu 13: Thống kê phân bố tổng giá trị giao dịch theo từng category, xếp theo thứ tự bảng chữ cái của mặt hàng

Câu 14: Thống kê tổng số lần giao dịch và tổng số tiền giao dịch theo từng quý trong năm 2020 của category "home" hoặc "kids\_pets" theo giới tính

Câu 15: Thống kê tổng số giao dịch và tổng số tiền theo category của từng nghề nghiệp

## PHẦN 2: QUÁ TRÌNH XÂY DỰNG KHO DỮ LIỆU (SSIS)

### 2.1 Chuẩn bị công cụ

Microsoft Visual Studio 2022

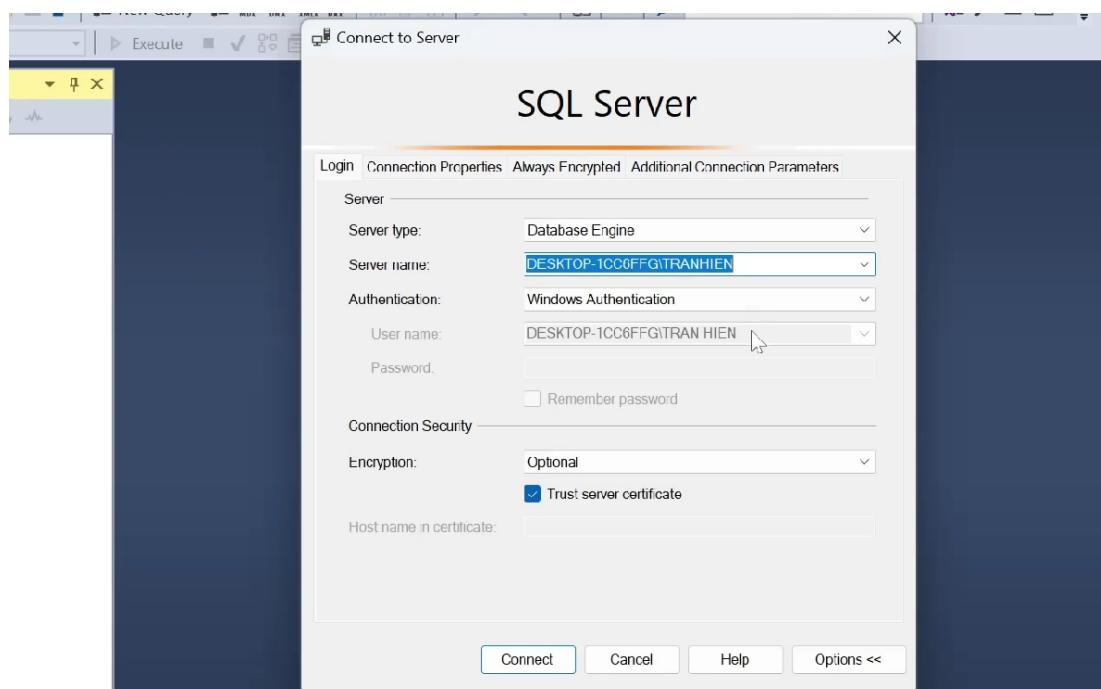


SQL Server Management Studio

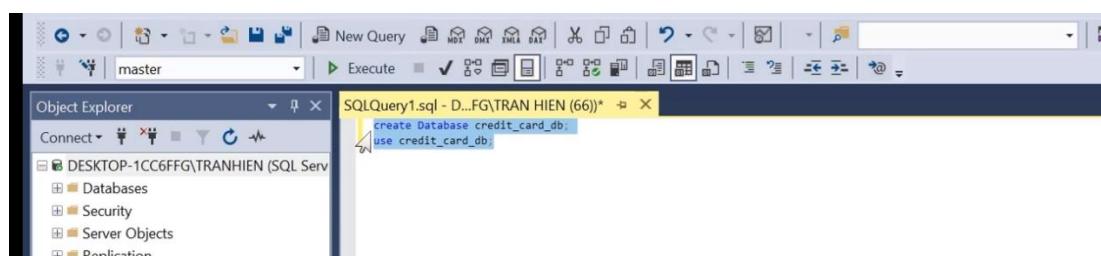


### 2.2 Chuẩn bị cơ sở dữ liệu và tạo project SSIS

Bước 1: Mở SQL Server và kết nối với server bằng tài khoản user của window

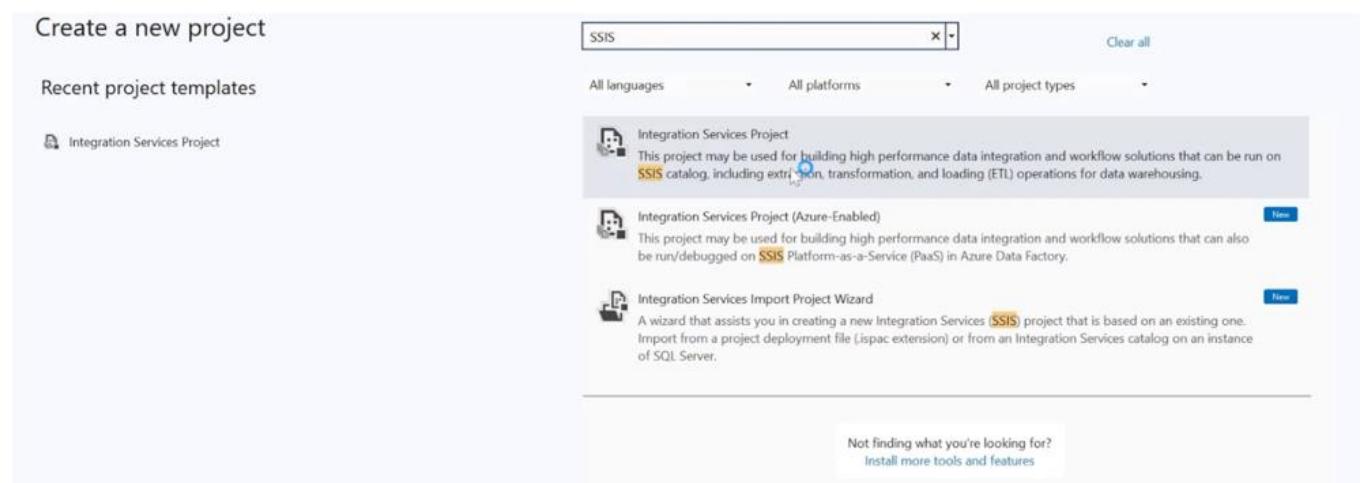
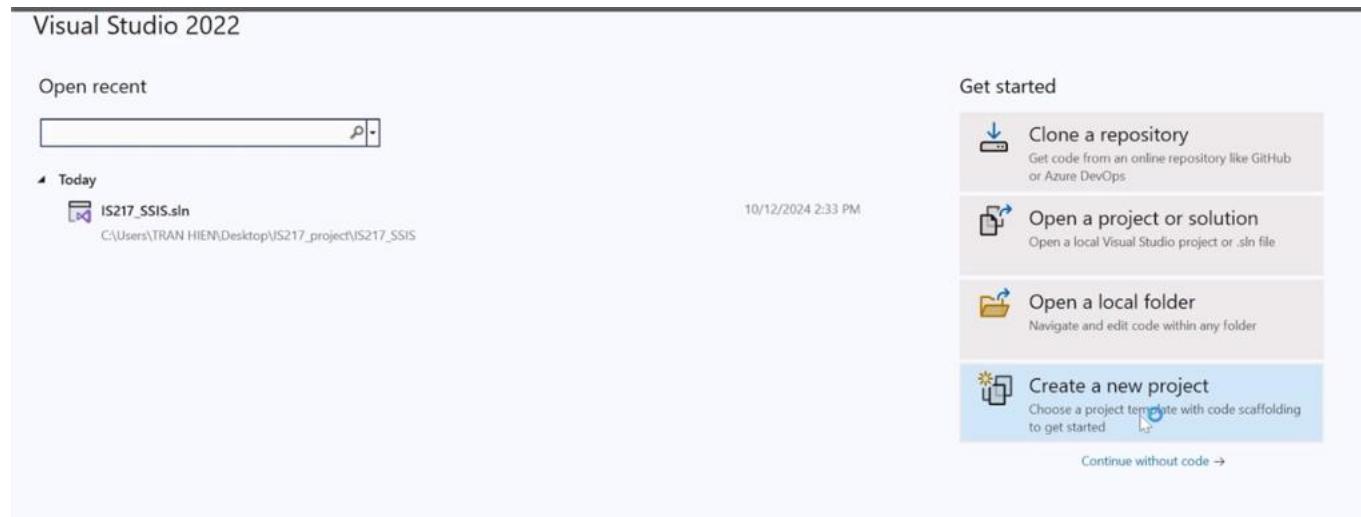


Bước 2: Tạo một database mới, đặt tên là credit\_card\_db để chứa dữ liệu các bảng Fact và bảng Dim

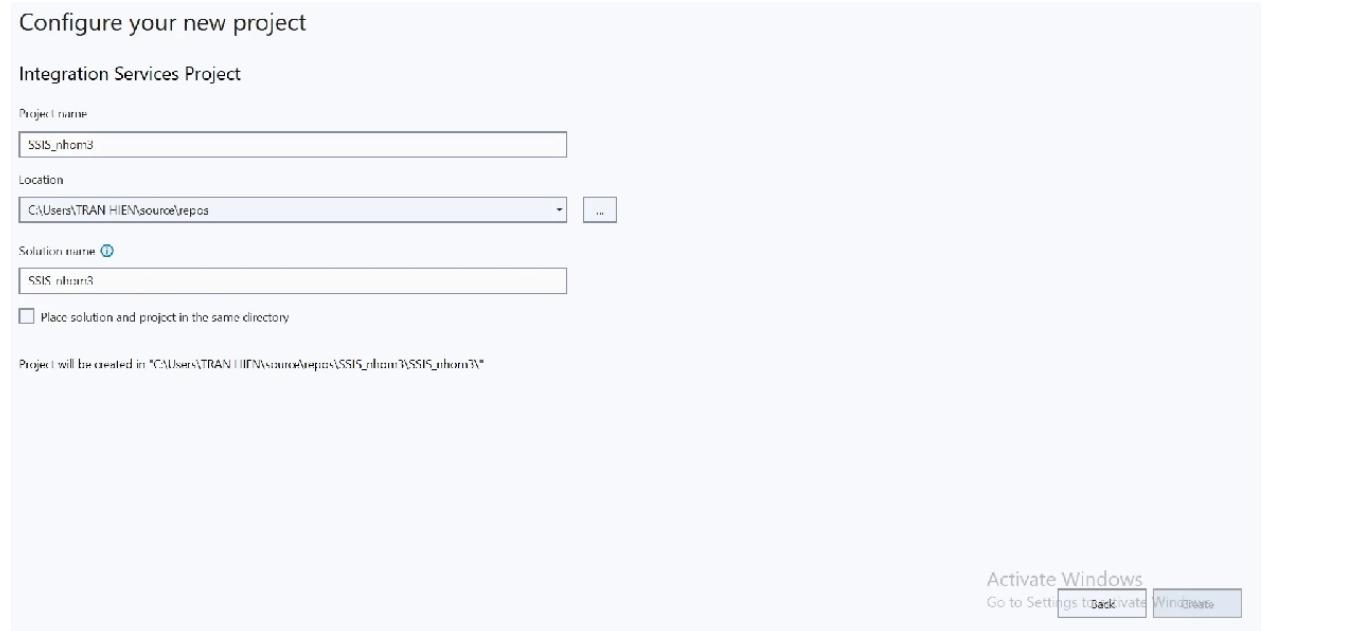


## 2.3 Tạo mới project SSIS

Bước 1: Vào Visual Studio và chọn “Create a new project”, chọn Integration Services Project và chọn Next.



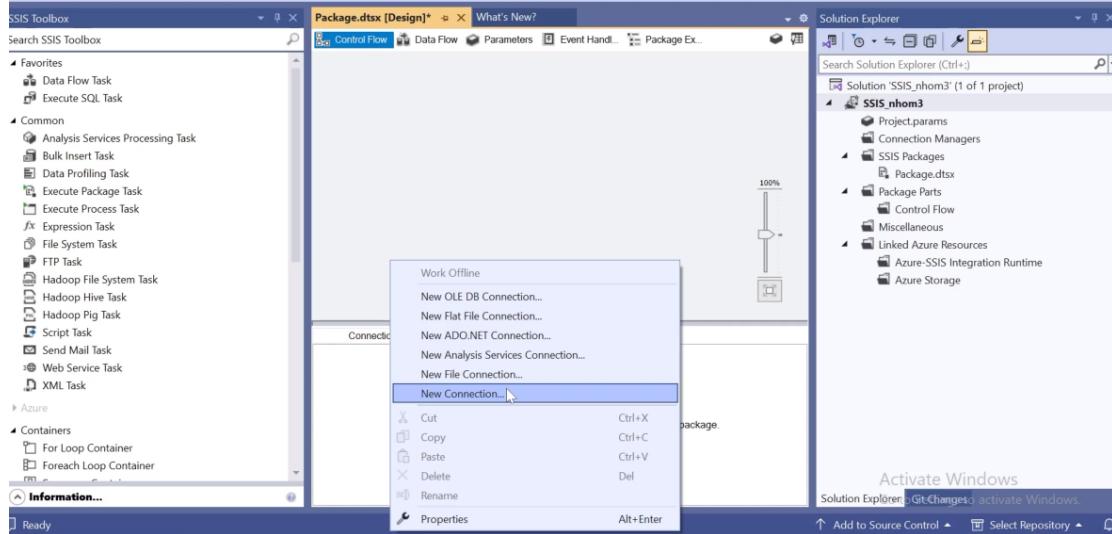
Bước 2: Đặt tên, thiết lập đường dẫn và hoàn tất



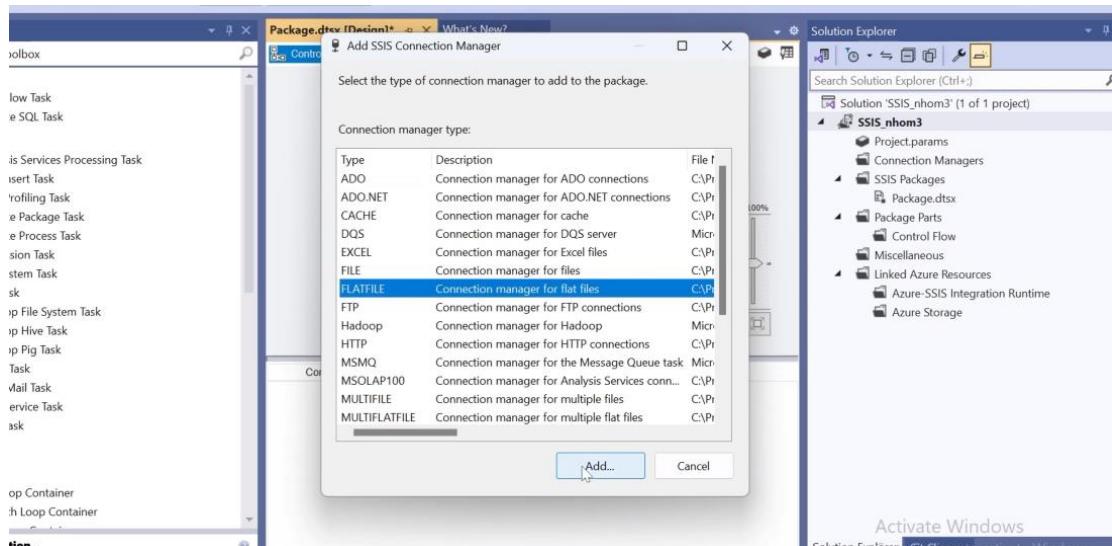
## 2.4 Tạo các connection

### 2.4.1 Flat file connection

Bước 1: Ở Connection Manager bấm chuột phải và chọn New connection

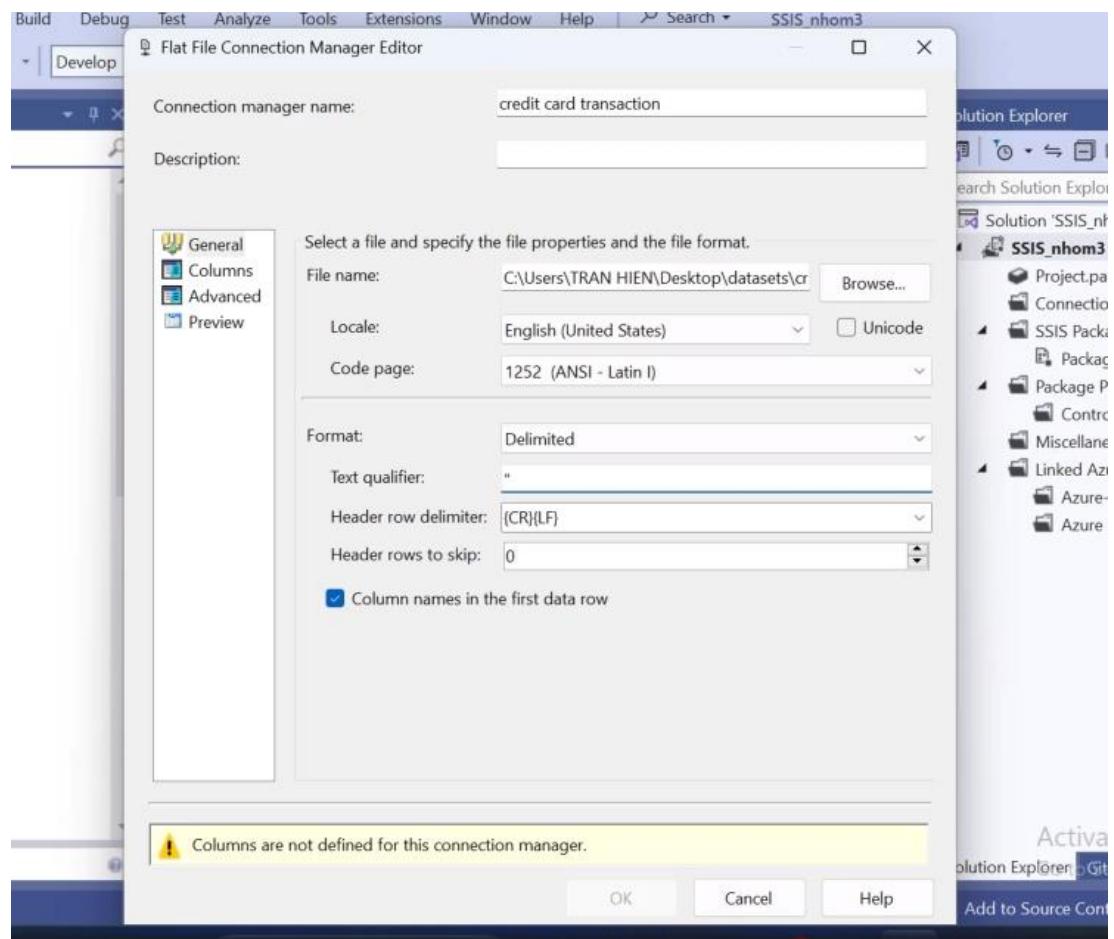


Bước 2: Chọn FLATFILE



Đặt connection manager name là: credit card transaction

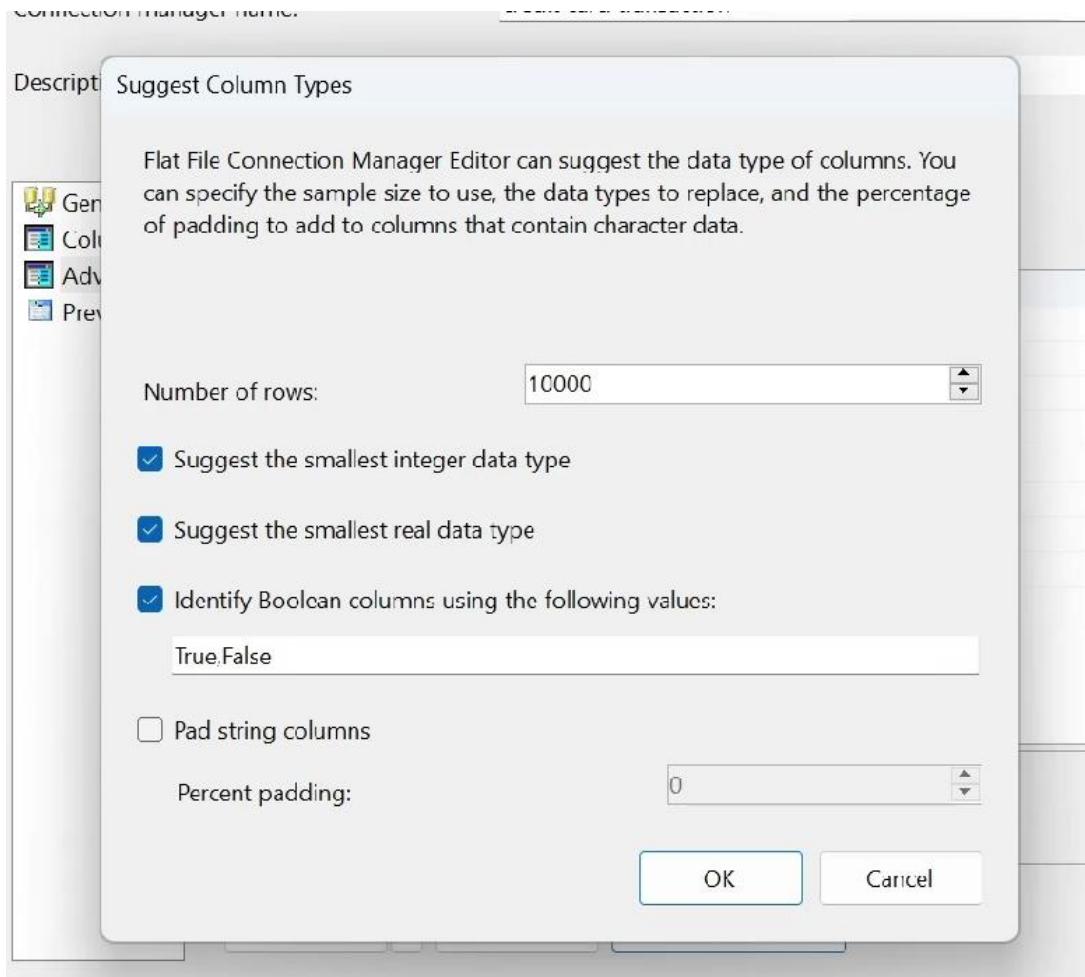
Chỉnh Text qualifier thành dấu nháy kép: “



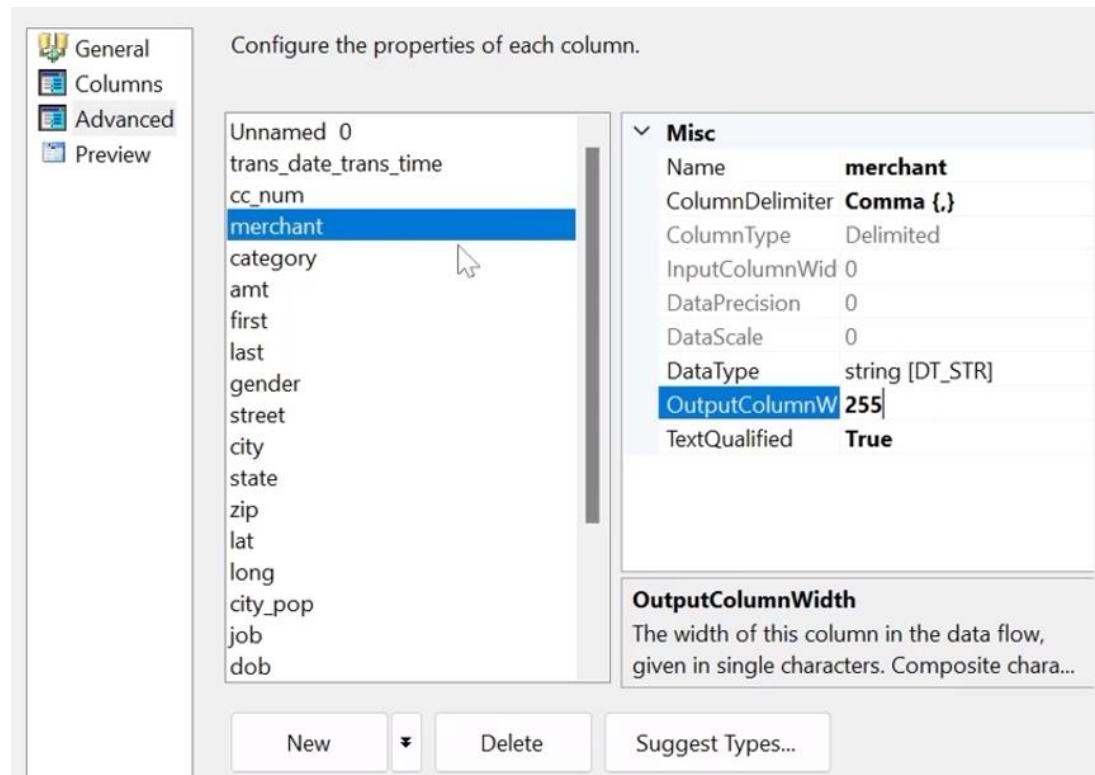
Bước 3: Qua mục advanced chọn Suggest để SSIS chọn giúp kiểu dữ liệu của các cột.

Name	Type	Width
trans_date_trans_time	string [DT_STR]	50
cc_num	string [DT_STR]	50
merchant	string [DT_STR]	50
category	string [DT_STR]	50
amt	string [DT_STR]	50
first	string [DT_STR]	50
last	string [DT_STR]	50
gender	string [DT_STR]	50
street	string [DT_STR]	50
city	string [DT_STR]	50
state	string [DT_STR]	50
zip	string [DT_STR]	50
lat	string [DT_STR]	50
long	string [DT_STR]	50
city_pop	string [DT_STR]	50
job	string [DT_STR]	50
dob	string [DT_STR]	50

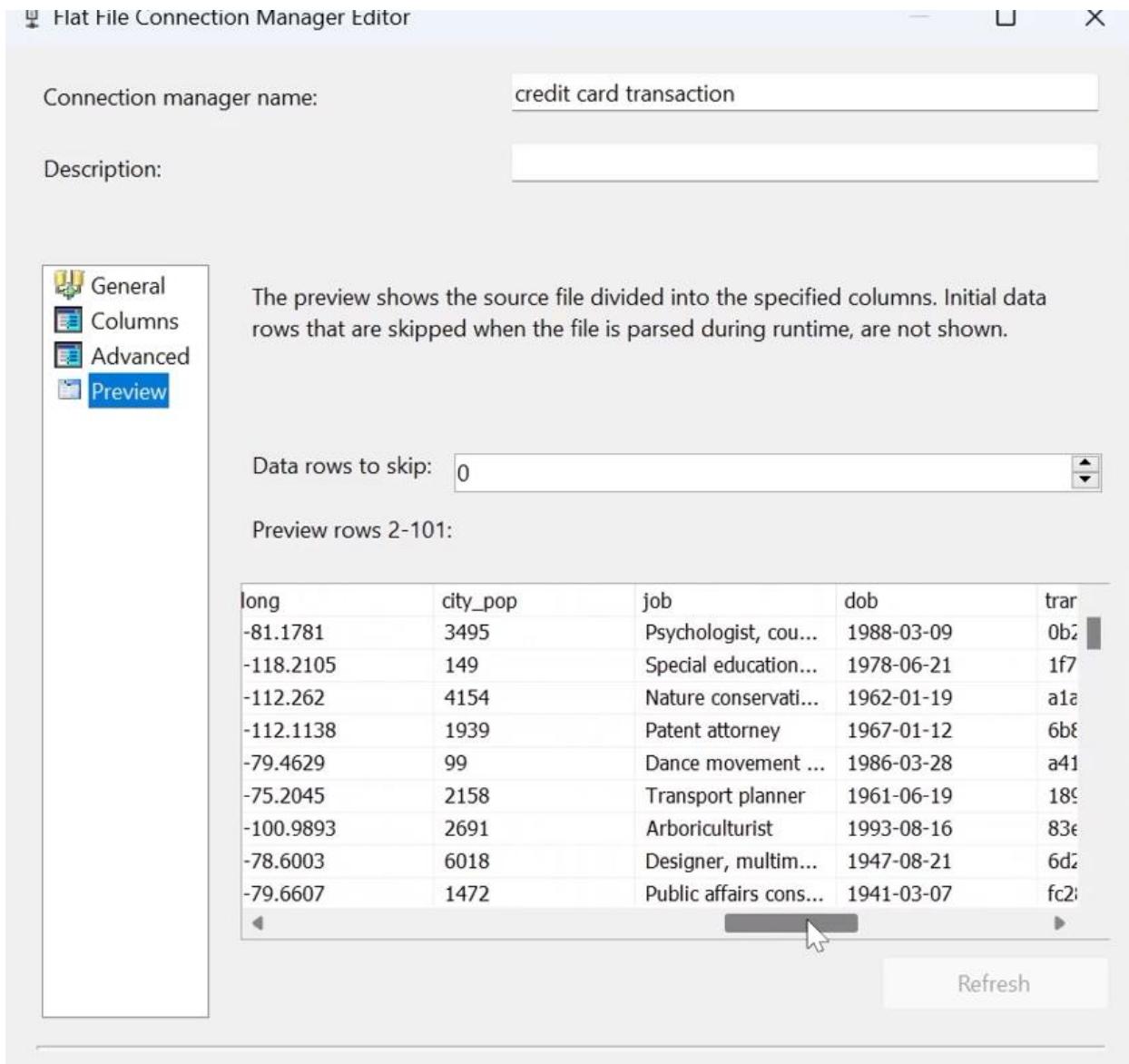
#### Bước 4: Chọn 10000 để SSIS xem xét và để xuất kiểu:



#### Bước 5: Chính các cột có kiểu dữ liệu là string thành độ dài là 255 để tránh bị lỗi truncate:

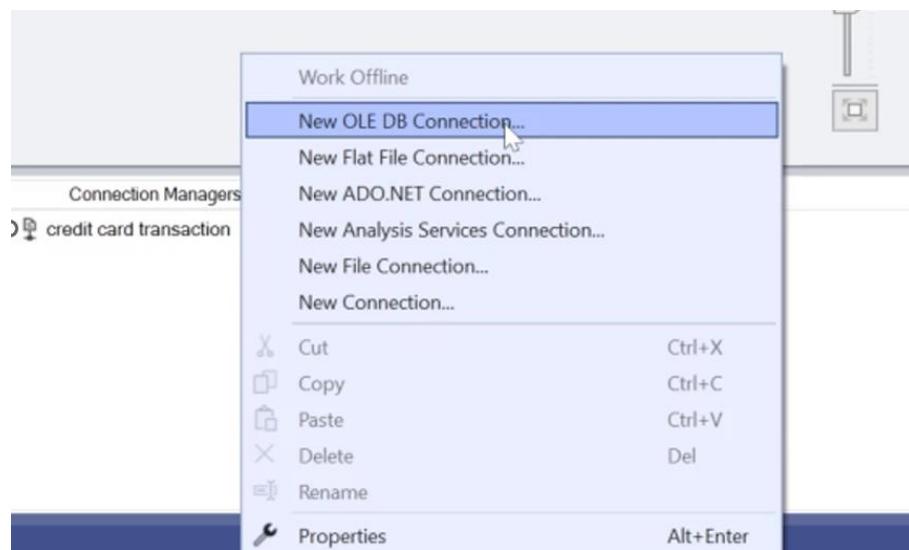


Bước 6: Nhấn qua phần preview để đảm bảo các cột hoàn chỉnh và đã đúng, sau đó nhấn ok để tiếp tục.

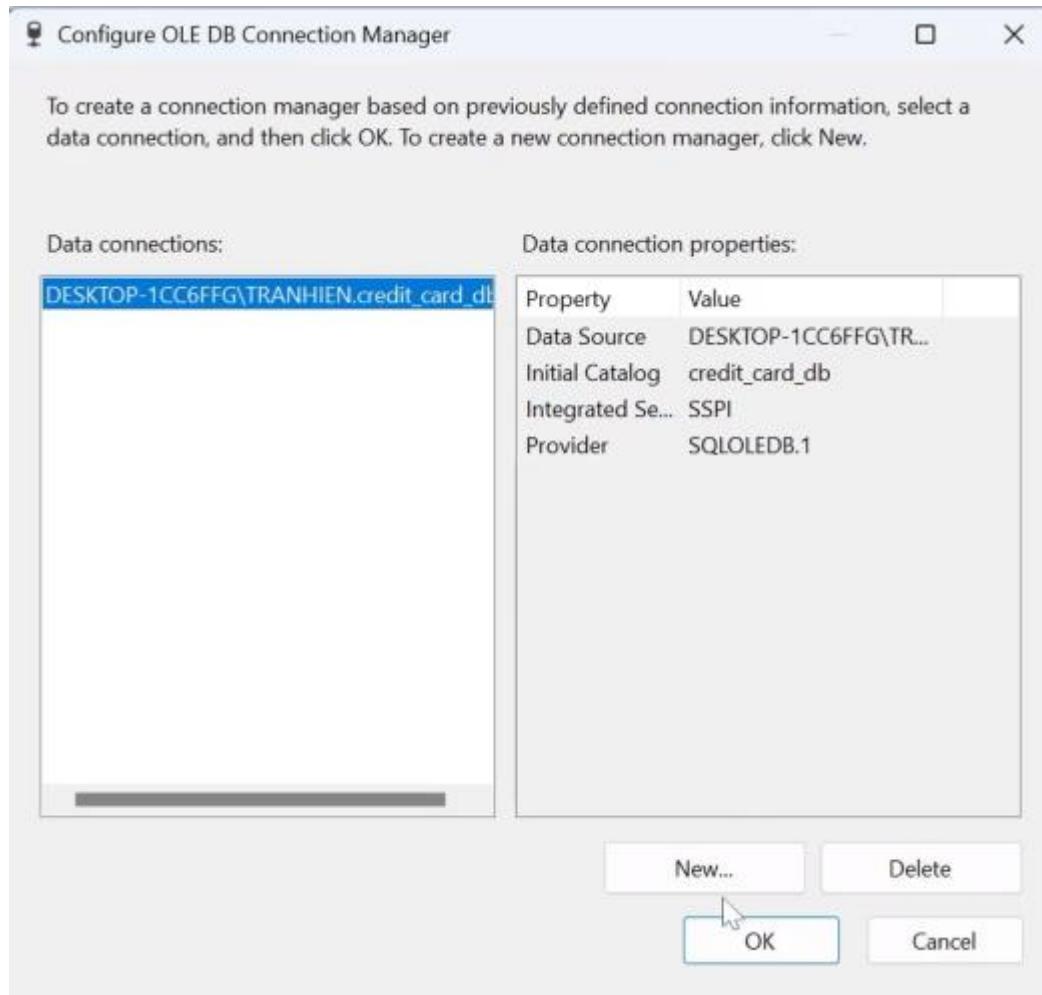


## 2.4.2 OLE DB connection

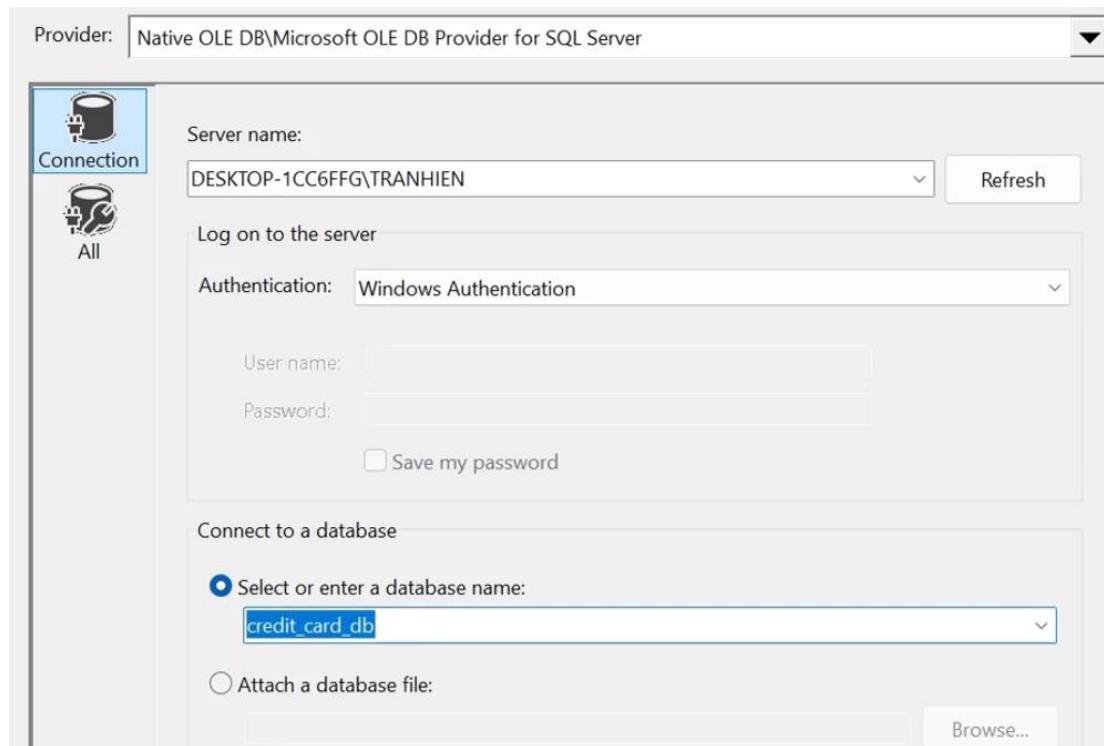
Bước 1: Nhấn chuột phải dưới phần connection manager và chọn New OLE DB connection



## Bước 2: Chọn New để tạo connection mới



Bước 3: Chọn Provider tương ứng với hình dưới, chọn server name và database tương ứng với phần đã tạo ở trên. Sau đó nhấn OK.

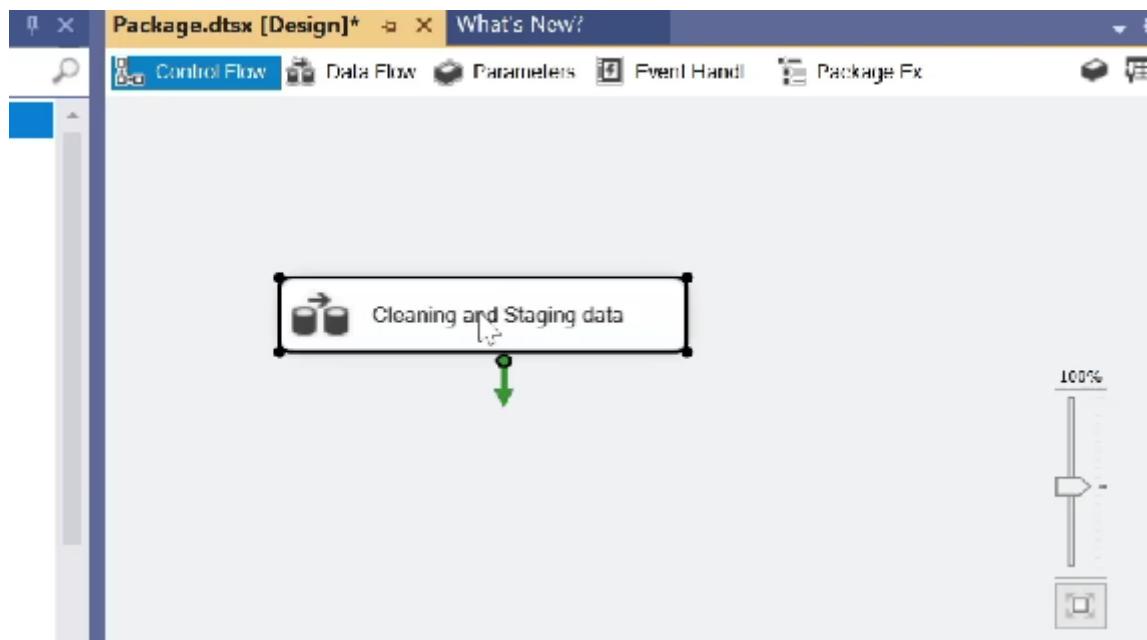


## 2.5 Cleaning and Staging data

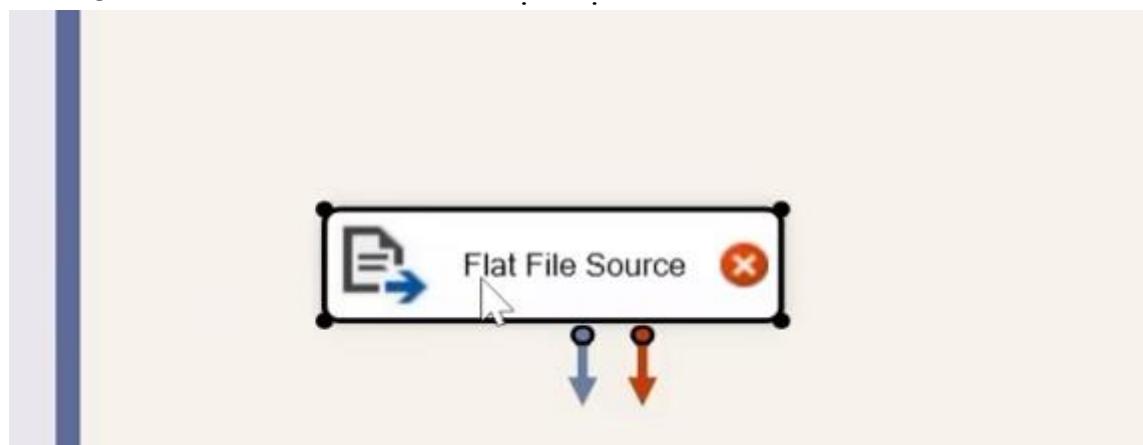
### 2.5.1 Cleaning data

Bước 1: Tạo mới Data Flow Task, Data Flow Task này sẽ đổ dữ liệu vào các bảng Dim và bảng StagingFact (để merge các id từ bảng dim ở phần sau)

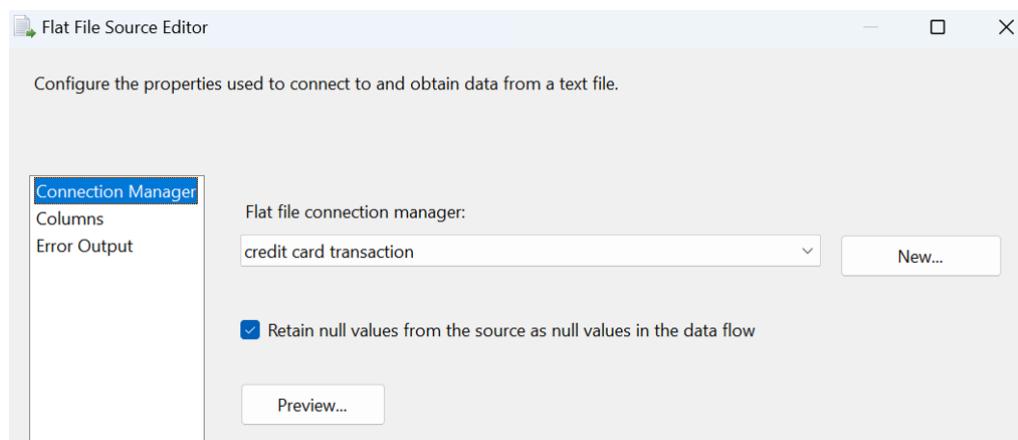
Bước 2: Tạo một Data Flow Task và đặt tên là Cleaning and Staging data



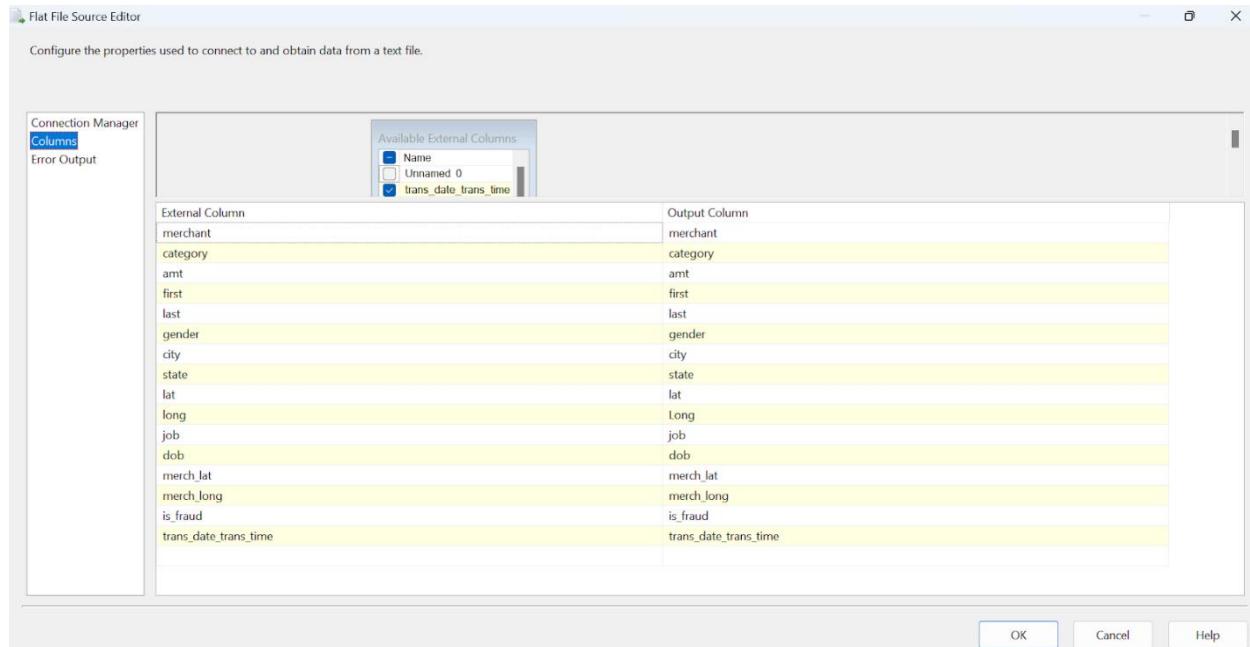
Bước 3: Vào Data Flow Task đó và tạo một Flat File Source



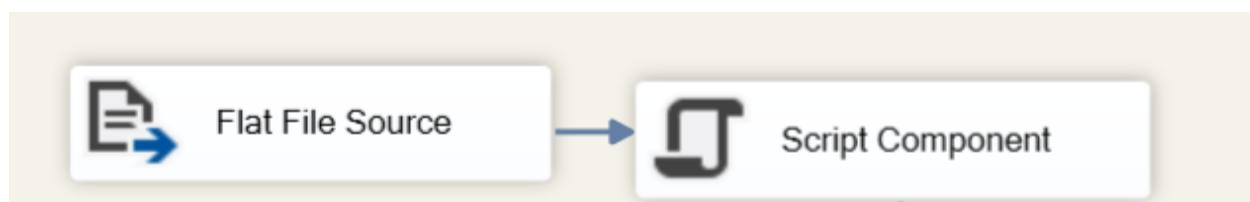
Bước 4: Chọn Flat File Connection đã tạo ở phần trước, click chọn Retain null values from the source as null values in the data flow



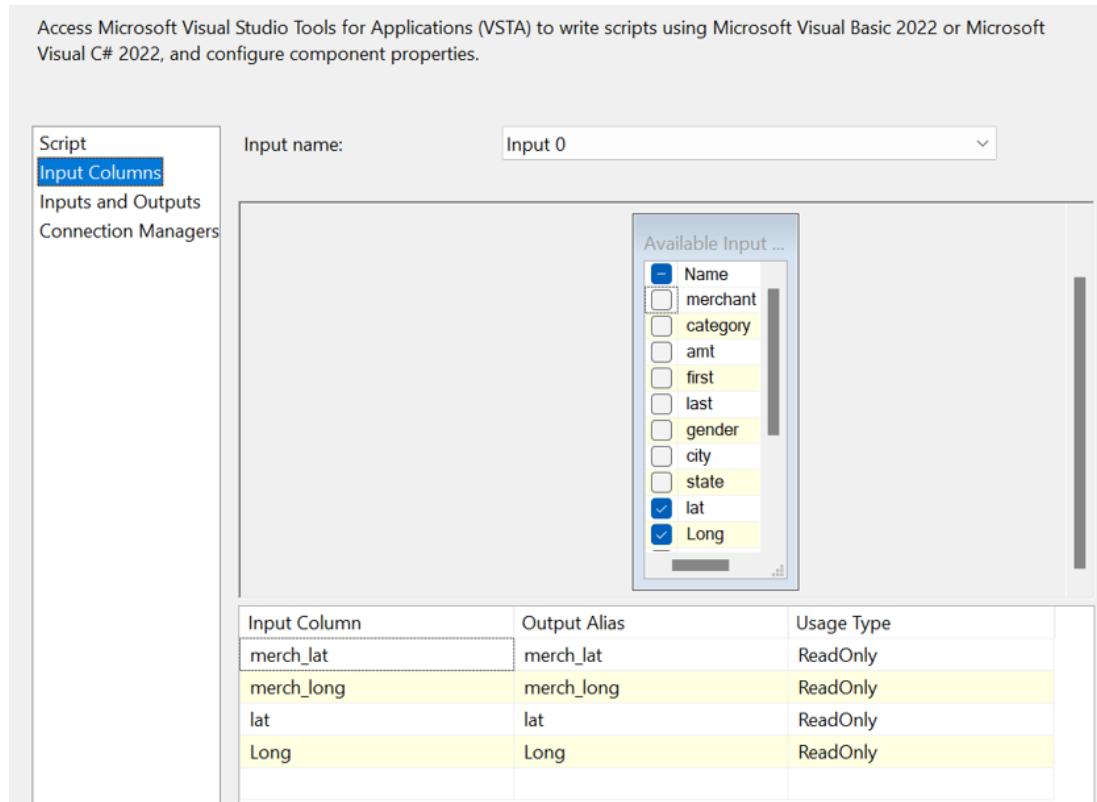
Bước 5: Ở phần Column, lọc ra các cột cần sử dụng, chú ý chỉnh sửa tên cột long thành Long để tránh bị lỗi trùng kiểu dữ liệu “long” khi thực hiện biến đổi dữ liệu bằng C#. Sau đó bấm OK để tiếp tục



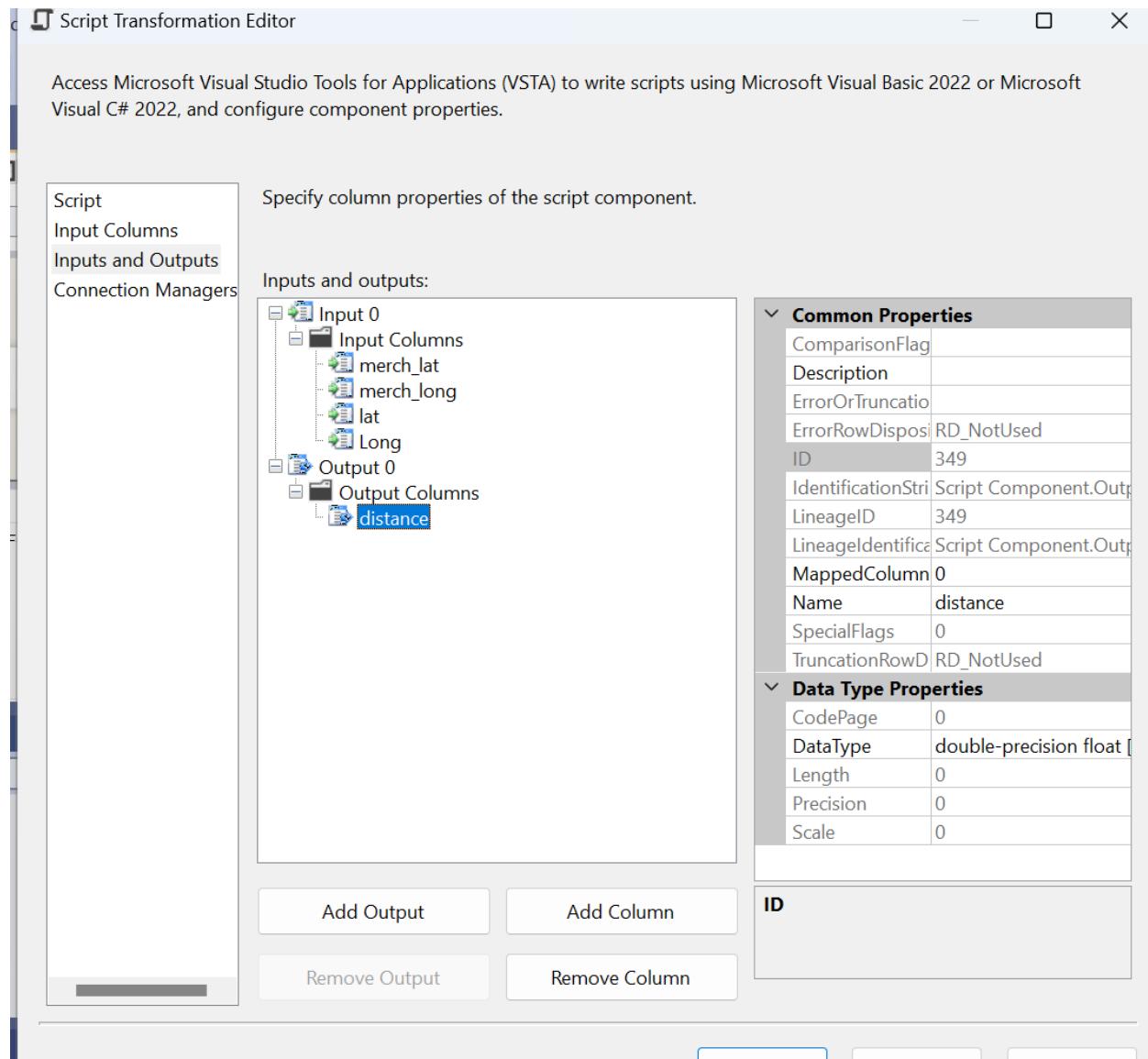
Bước 6: Tạo một Script Component và kết nối đến.



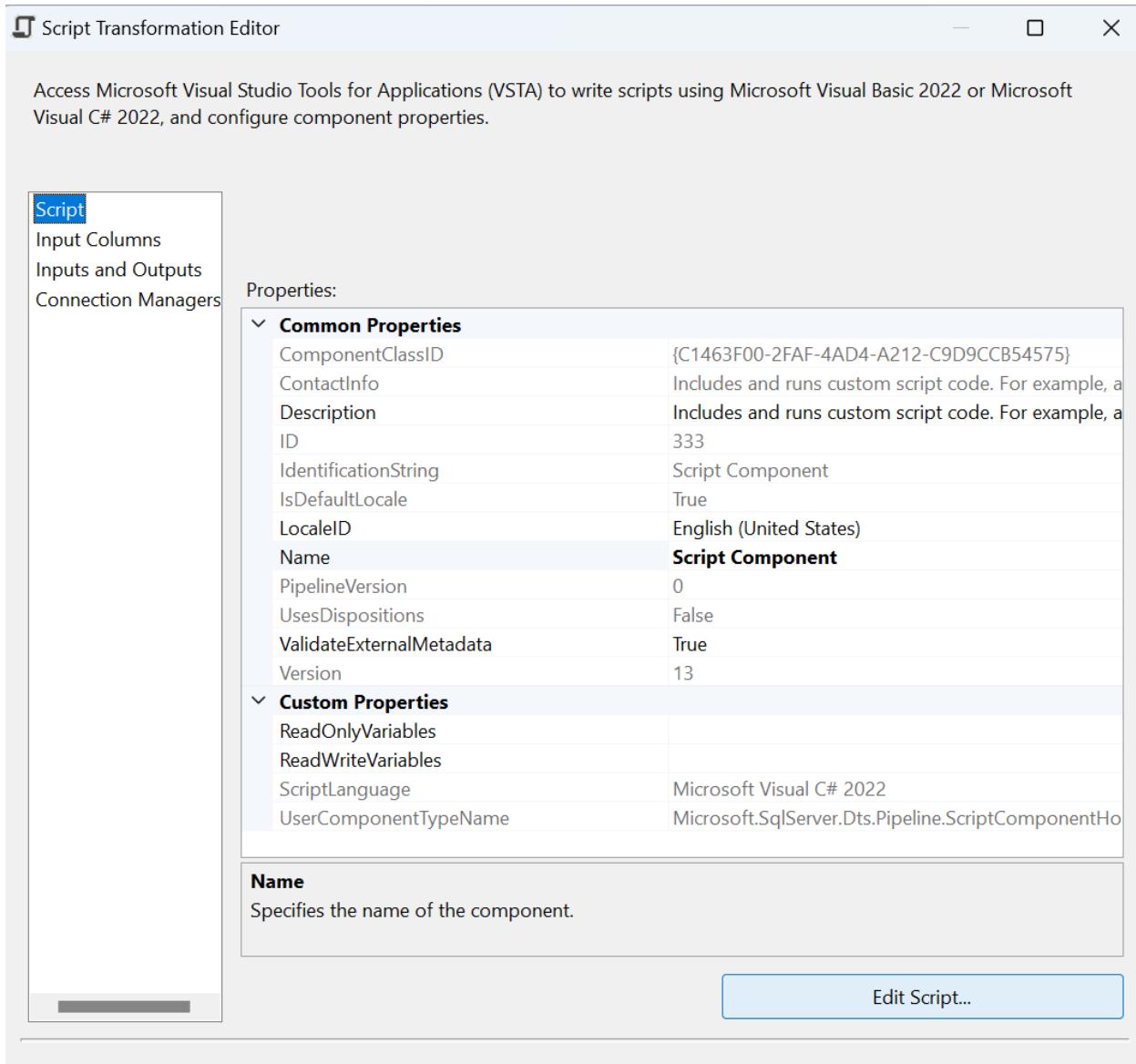
Bước 7: Vào Script Component, chọn Input Column là các cột như hình dưới.



Bước 8: Ở phần Inputs and Outputs, tạo cột Output mới là distance với kiểu dữ liệu là double-precision float.



Bước 9: Ở phần Script, nhấn Edit Script... để hiện code



Bước 10: Tại đây, chèn code có sẵn vô để tiến hành tạo ra cột distance. Sau đó lưu lại và tắt và nhấn OK để tiếp tục

```

    public override void Input0_ProcessInputRow(Input0Buffer Row)
    {
        double merchLat = Row.merchlat;
        double merchLong = Row.merchlong;
        double lat = Row.lat;
        double lng = Row.Long;

        // Calculate distance using Haversine formula
        double distance = CalculateDistance(merchLat, merchLong, lat, lng);

        // Assign the calculated distance to the output column
        Row.distance = distance;
    }

    private double CalculateDistance(double lat1, double lon1, double lat2, double lon2)
    {
        // Implementation of Haversine formula
    }
}

```

Code:

```
public override void Input0_ProcessInputRow(Input0Buffer Row)
{
    double merchLat = Row.merchlat;
    double merchLong = Row.merchlong;
    double lat = Row.lat;
    double lng = Row.Long;

    // Calculate distance using Haversine formula
    double distance = CalculateDistance(merchLat, merchLong, lat, lng);

    // Assign the calculated distance to the output column
    Row.distance = distance;
}

private double CalculateDistance(double lat1, double lon1, double lat2, double lon2)
{
    const double EarthRadius = 6371; // Earth radius in kilometers

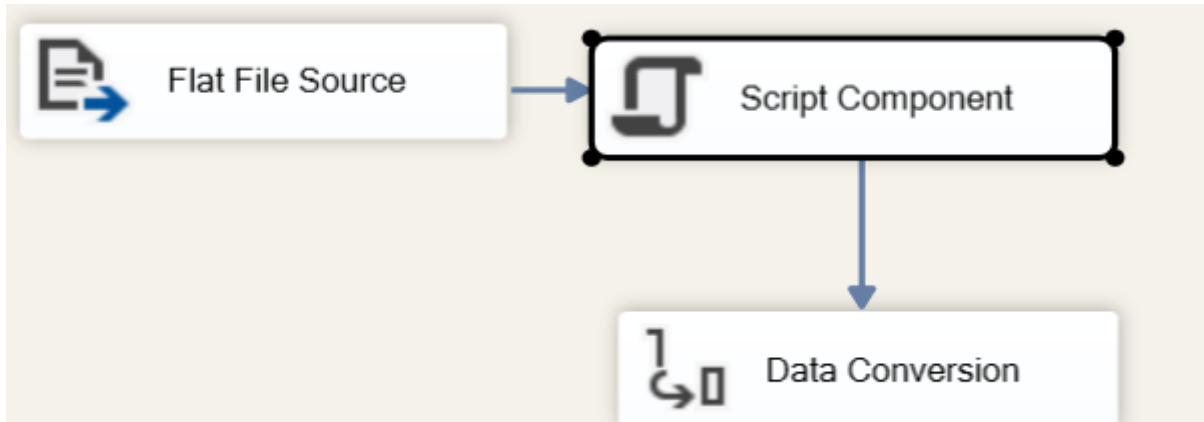
    // Convert degrees to radians
    lat1 = ToRadians(lat1);
    lon1 = ToRadians(lon1);
    lat2 = ToRadians(lat2);
    lon2 = ToRadians(lon2);

    // Haversine formula
    double dlon = lon2 - lon1;
    double dlat = lat2 - lat1;
    double a = Math.Sin(dlat / 2) * Math.Sin(dlat / 2) +
               Math.Cos(lat1) * Math.Cos(lat2) *
               Math.Sin(dlon / 2) * Math.Sin(dlon / 2);
    double c = 2 * Math.Atan2(Math.Sqrt(a), Math.Sqrt(1 - a));

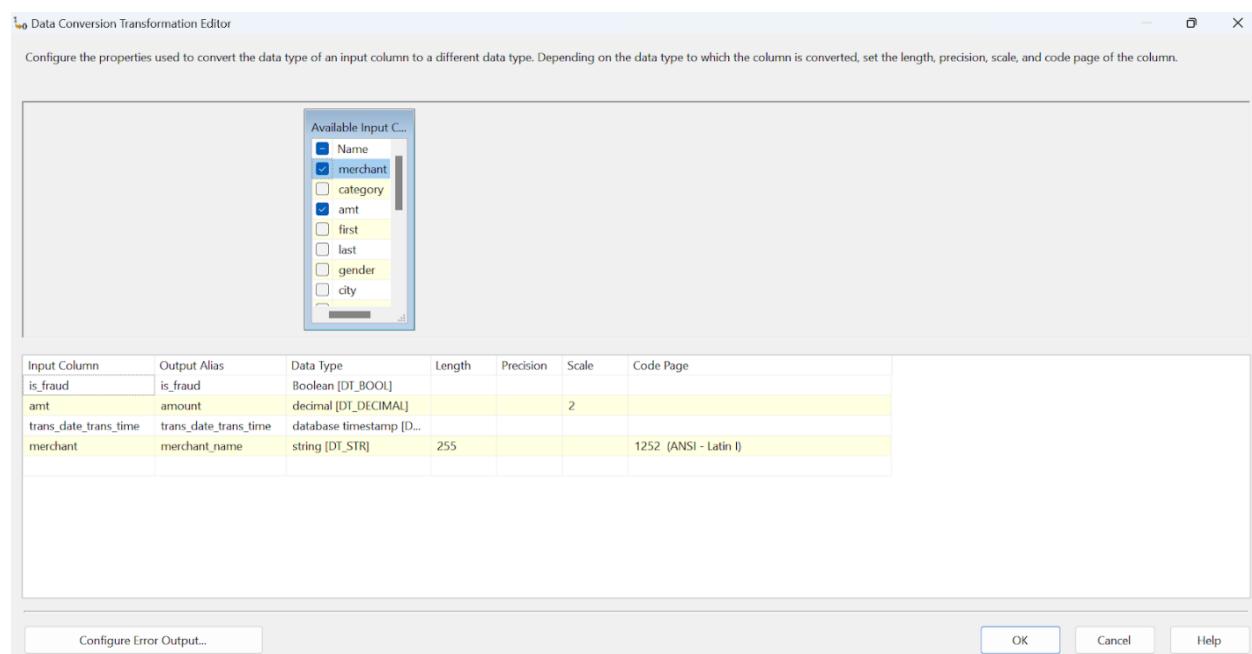
    // Calculate distance
    double distance = EarthRadius * c;
    return distance;
}

// Helper function to convert degrees to radians
private double ToRadians(double degrees)
{
    return degrees * (Math.PI / 180);
}
```

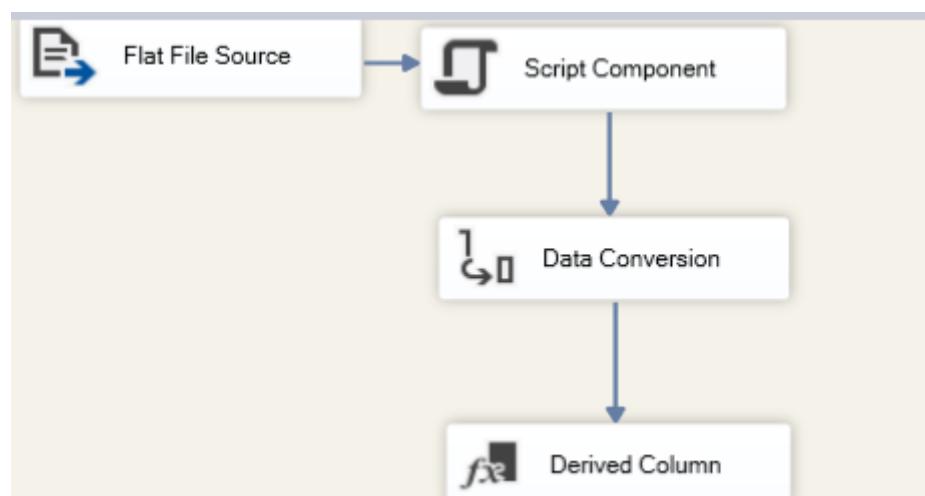
Bước 11: Tạo một Data Conversion và kết nối đến chúng.



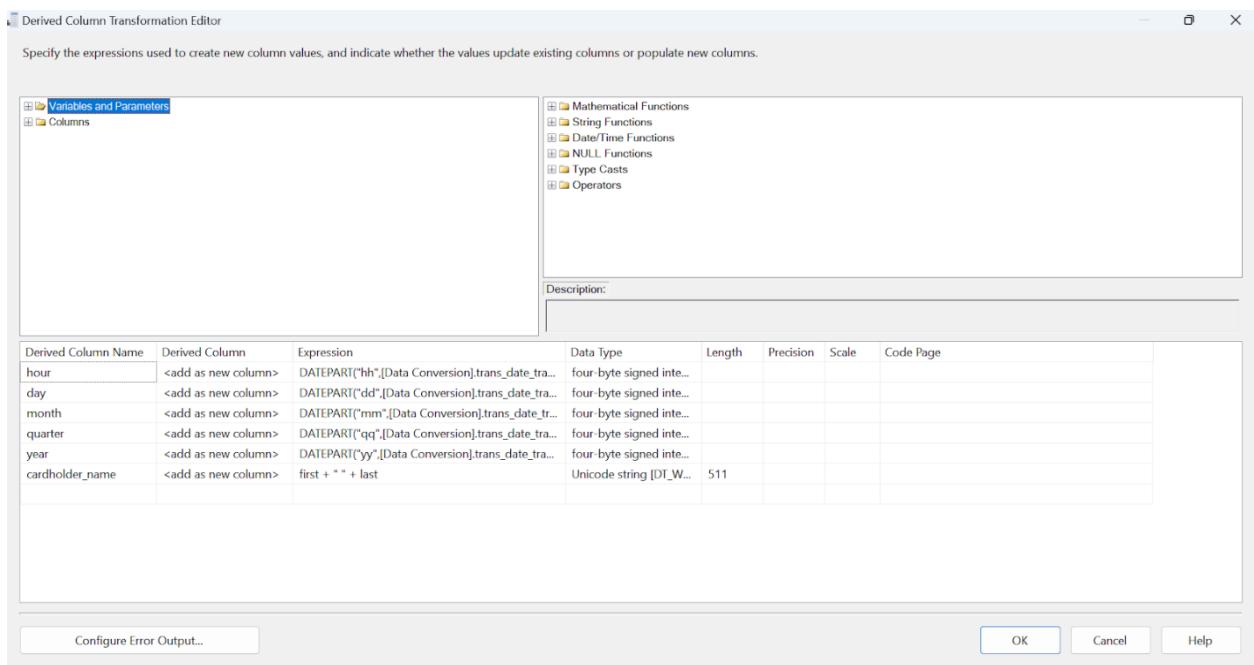
Bước 12: Vào Data Conversion, sửa tên và kiểu dữ liệu như sau và nhấn OK để tiếp tục..



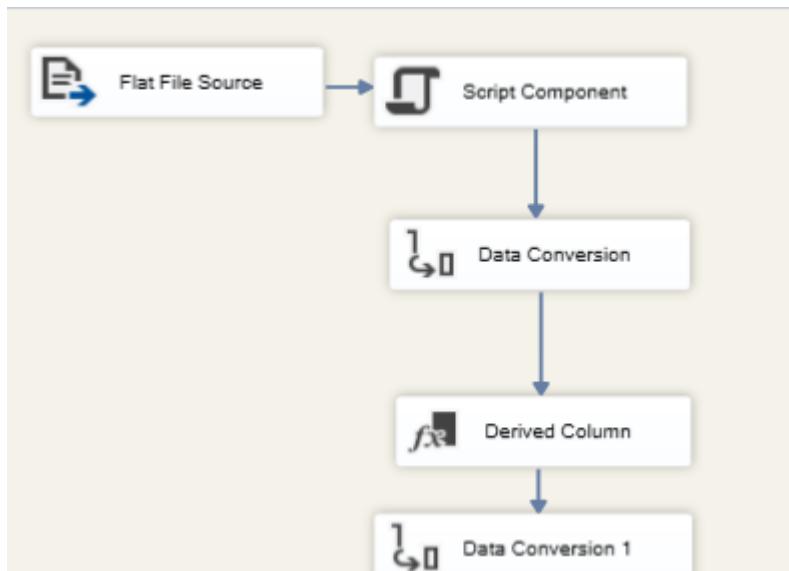
Bước 13: Tạo một Derived Column và kết nối đến



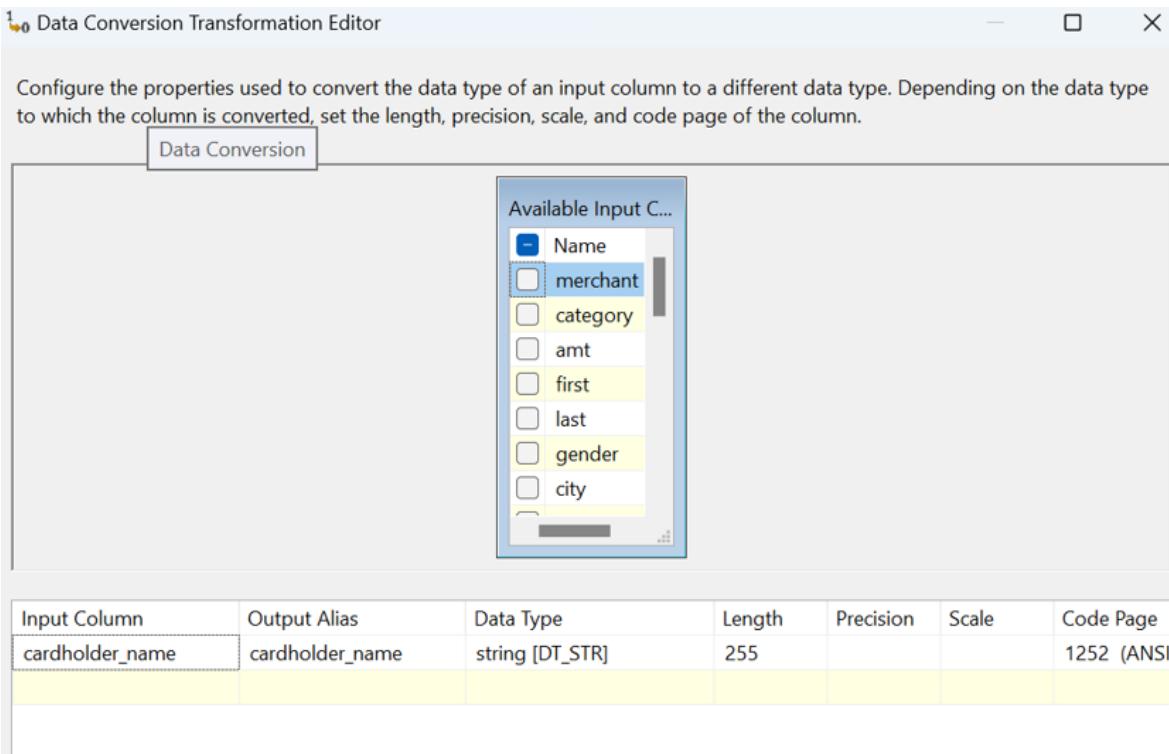
Bước 14: Vào Derived Column, tạo ra cột mới bằng các hàm có sẵn như hình dưới. Sau đó nhấn OK để tiếp tục.



### Bước 15: Tạo một Data Conversion 1 mới



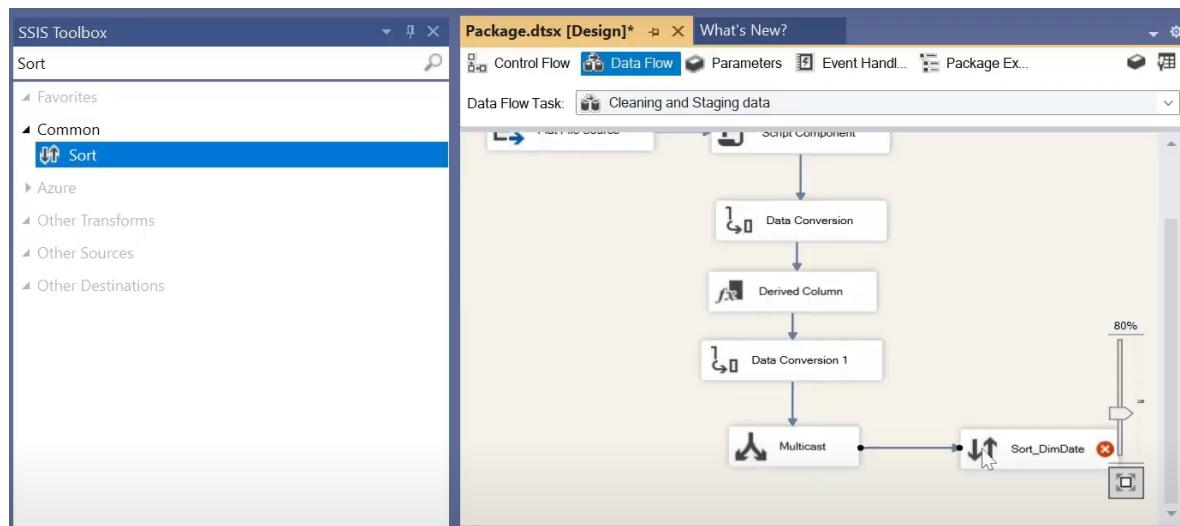
### Bước 16: Chuyển kiểu dữ liệu của cột cardholder\_name vừa tạo thành kiểu không có unicode để tránh bị lỗi kiểu dữ liệu, sau đó nhấn OK để tiếp tục

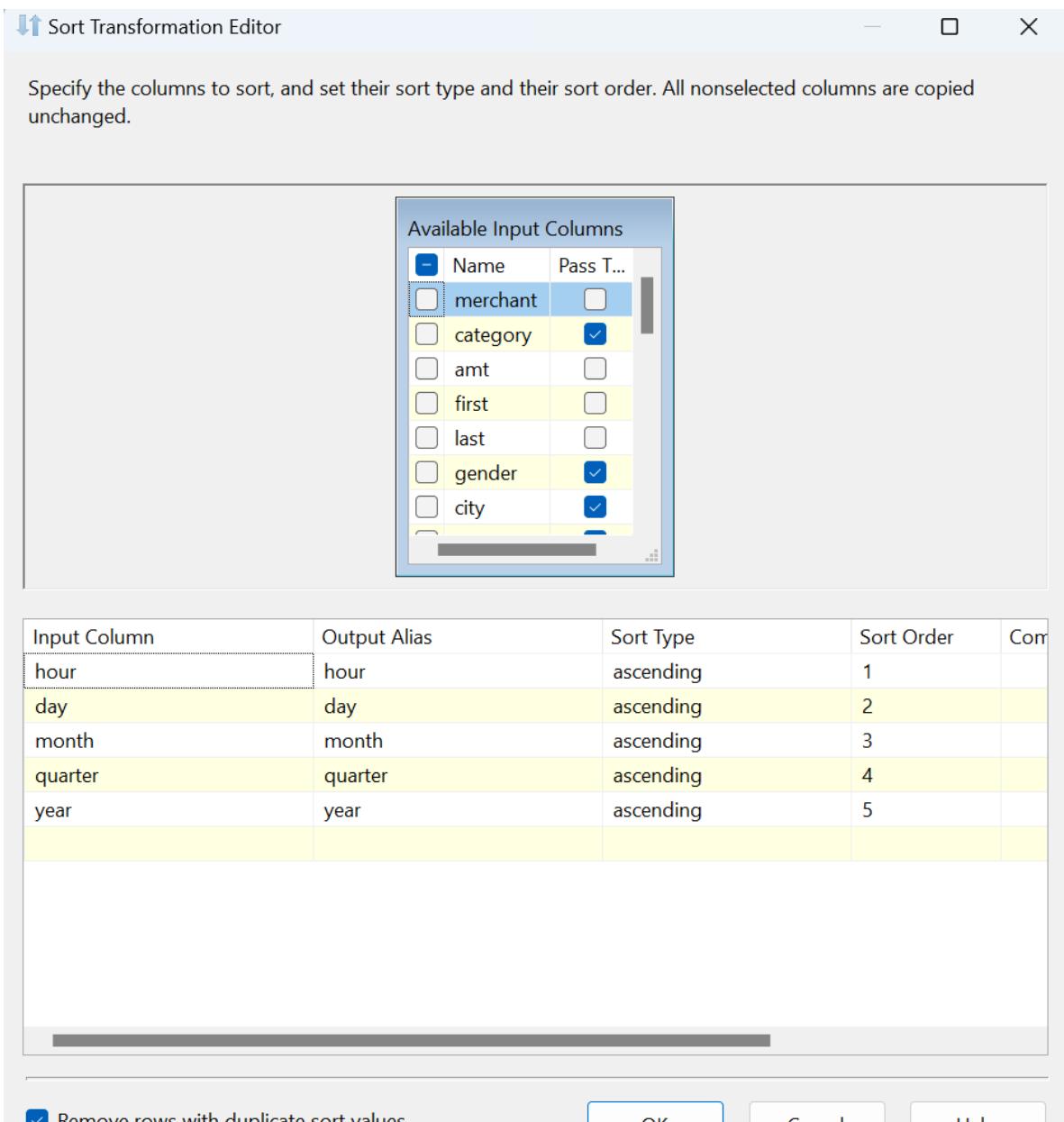


### 2.5.2 Tạo các bảng Dim và Fact

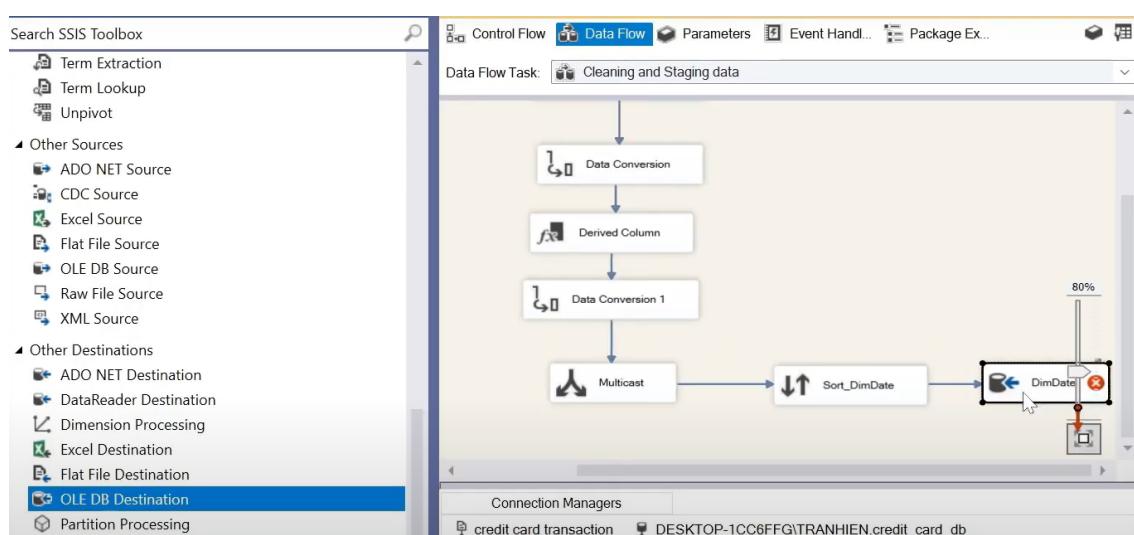
### 2.5.2.1 Tạo bảng DimDate

Bước 1: Tạo một Sort mới, đổi tên là Sort\_DimDate để lấy ra các cột dữ liệu cần thiết. Chọn Sort, và chọn các thuộc tính của bảng DimDate. Tick vào phần Remove rows with duplicate sort values) để xóa các dòng dữ liệu trùng nhau, sau đó chọn OK.





Bước 2: Tạo mới một OLE DB Destination, đổi tên thành DimDate và nối Sort\_DimDate tới OLE DB Destination này.



Chọn vào DimDate, chọn New... để tạo mới bảng.

Code tạo bảng:

```
CREATE TABLE dim_datetime (
```

```
    datetime_id INT IDENTITY(1,1) PRIMARY KEY, -- Surrogate key
```

```
    [hour] INT,
```

```
    [day] INT,
```

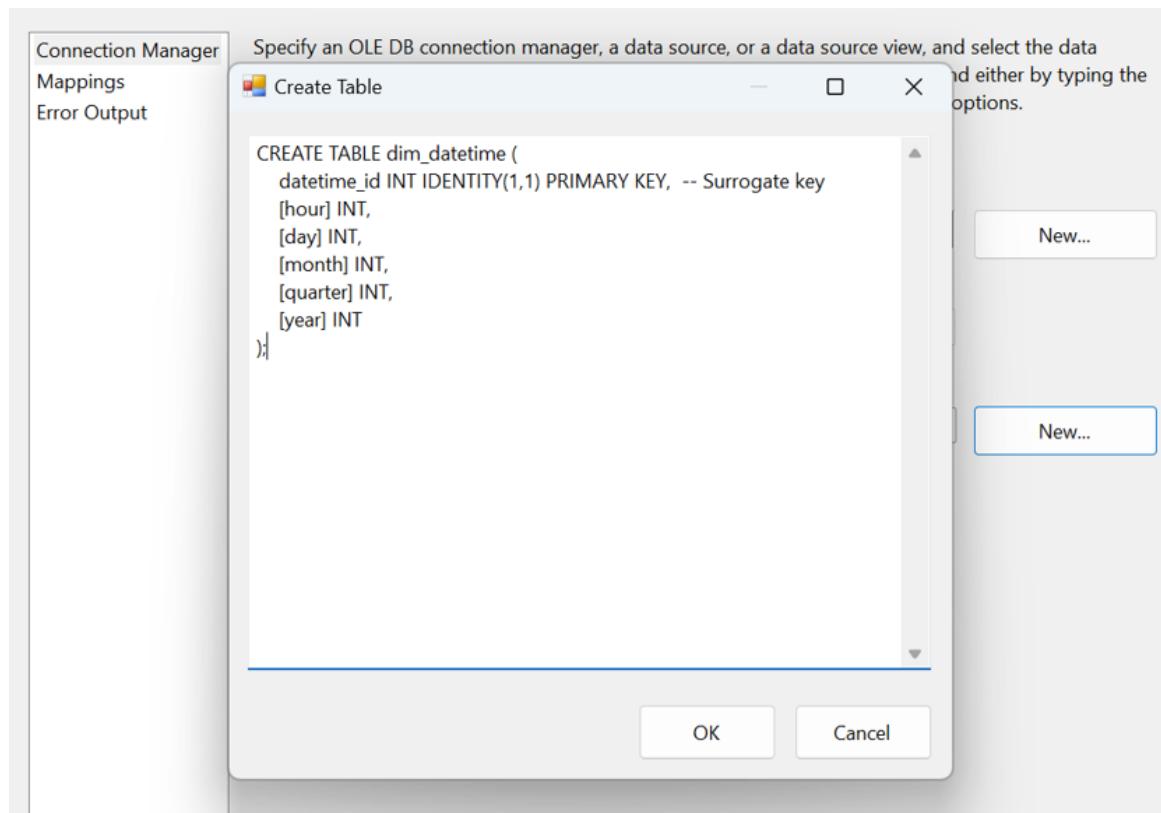
```
    day_of_week INT,
```

```
    [month] INT,
```

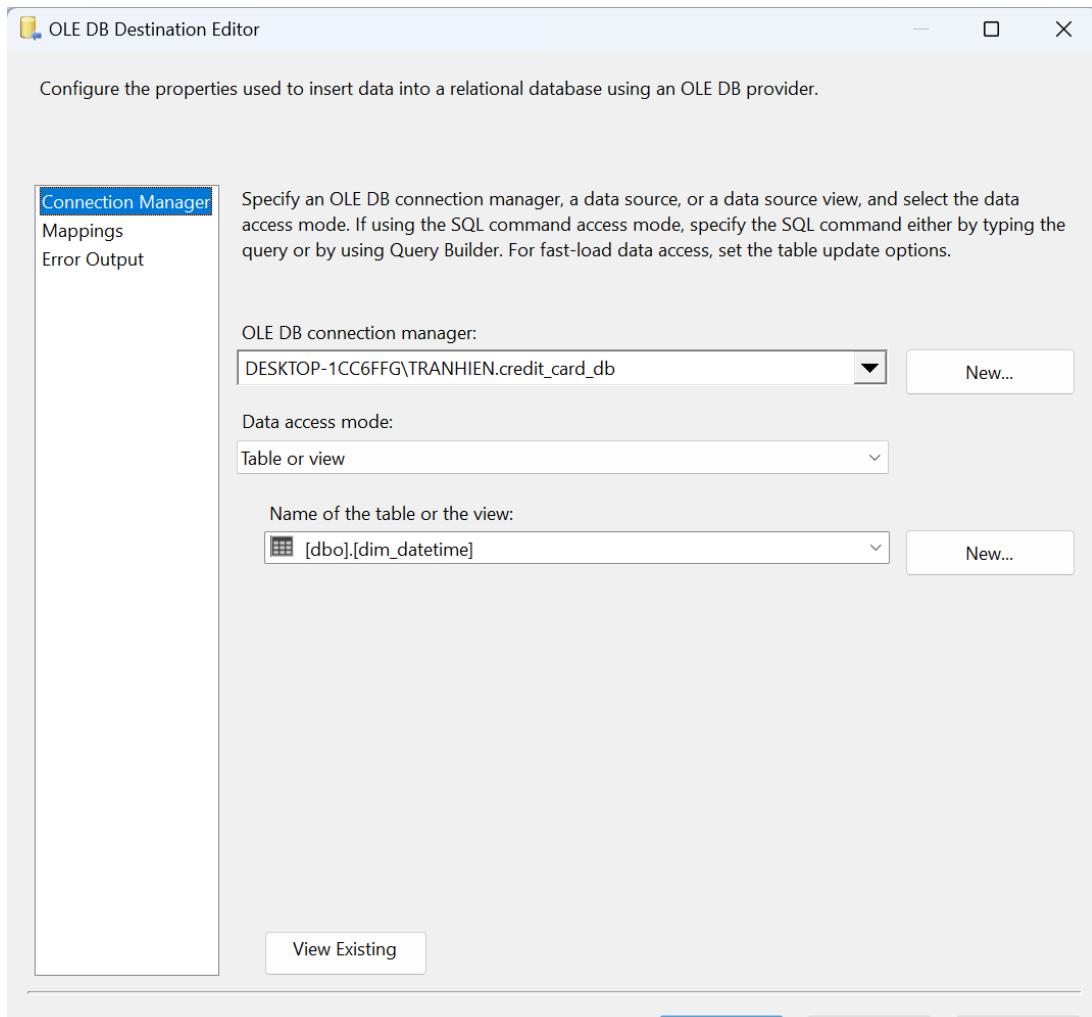
```
    [quarter] INT,
```

```
    [year] INT
```

```
);
```



Bước 3: Chọn Connection đã tạo



Bước 4: Tiếp đến ta cần chọn mục Mappings để xem xét việc ánh xạ các cột dữ liệu. Chọn OK để hoàn tất thiết lập.

Connection Manager  
Mappings  
Error Output

Available Input Columns

- Name
- city
- category
- amount
- state
- hour
- day
- month
- quarter
- year
- distance

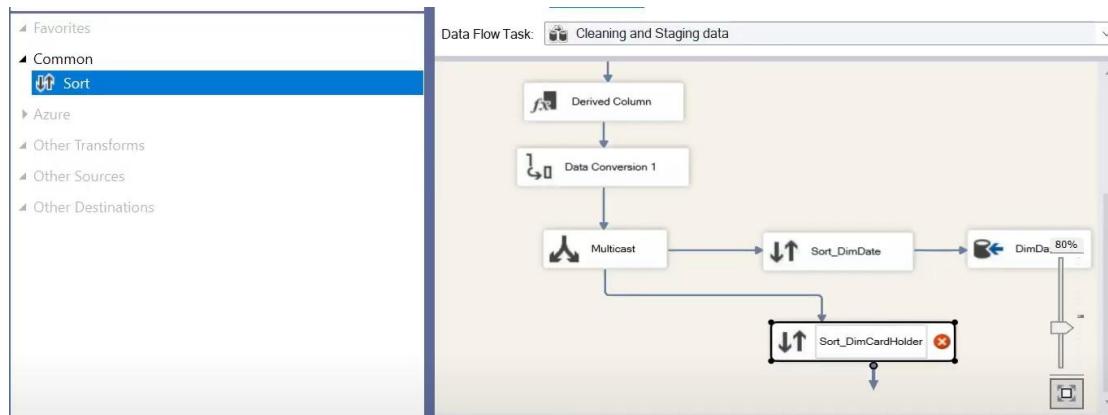
Available Dest...

- datetime\_id
- hour
- day
- month
- quarter

Input Column	Destination Column
<ignore>	datetime_id
hour	hour
day	day
month	month
quarter	quarter
year	year

### 2.5.2.2 Tạo bảng DimCardHolder

Bước 1: Tạo một Sort mới, đổi tên là Sort\_CardHolder để lấy ra các cột dữ liệu cần thiết. Chọn Sort, và chọn các thuộc tính của bảng DimCardHolder. Tick vào phần Remove rows with duplicate sort values) để xóa các dòng dữ liệu trùng nhau, sau đó chọn OK.



Sort Transformation Editor

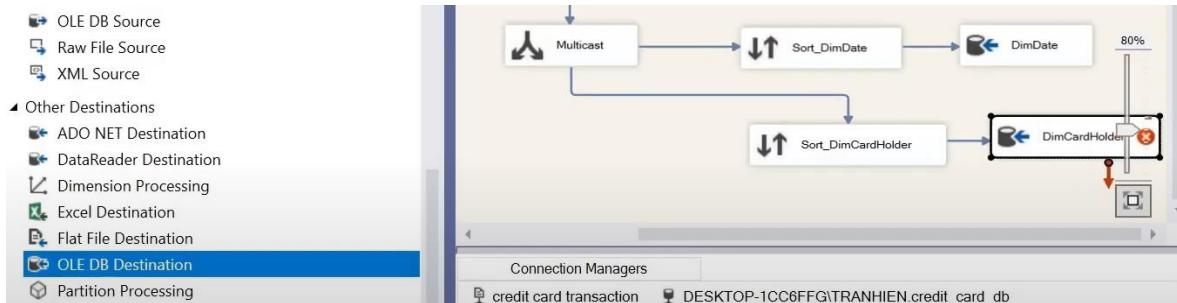
Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

Available Input Columns		
Name	Pass T...	
merchant	<input checked="" type="checkbox"/>	
category	<input checked="" type="checkbox"/>	
amt	<input type="checkbox"/>	
first	<input type="checkbox"/>	
last	<input type="checkbox"/>	
gender	<input checked="" type="checkbox"/>	
city	<input checked="" type="checkbox"/>	

Input Column	Output Alias	Sort Type	Sort Order	Com
Data Conversion 1.cardhold...	cardholder_name	ascending	1	
job	job	ascending	2	
gender	gender	ascending	3	
dob	dob	ascending	4	
city	city	ascending	5	
state	state	ascending	6	

Remove rows with duplicate sort values OK Cancel Help

Bước 2: Tạo mới một OLE DB Destination, đổi tên thành DimCardHolder và nối Sort\_CardHolder tới OLE DB Destination này.



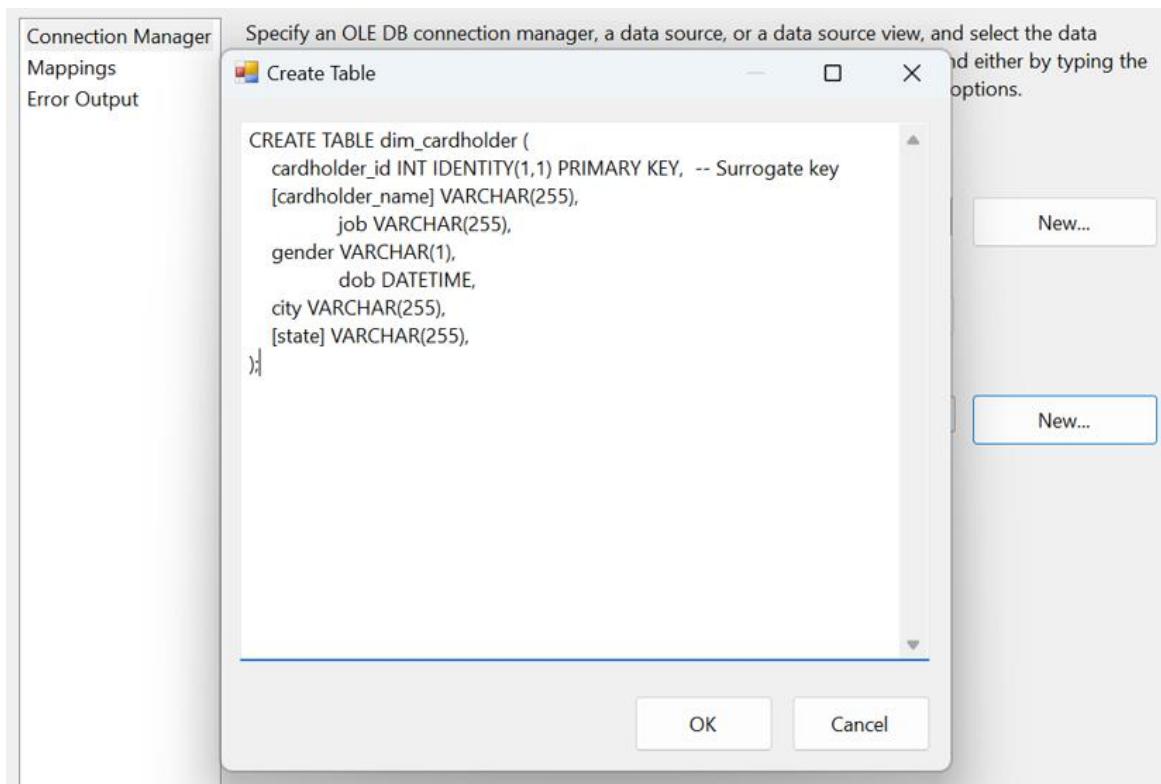
Chọn vào DimCardHolder, chọn New... để tạo mới bảng.

Code tạo bảng:

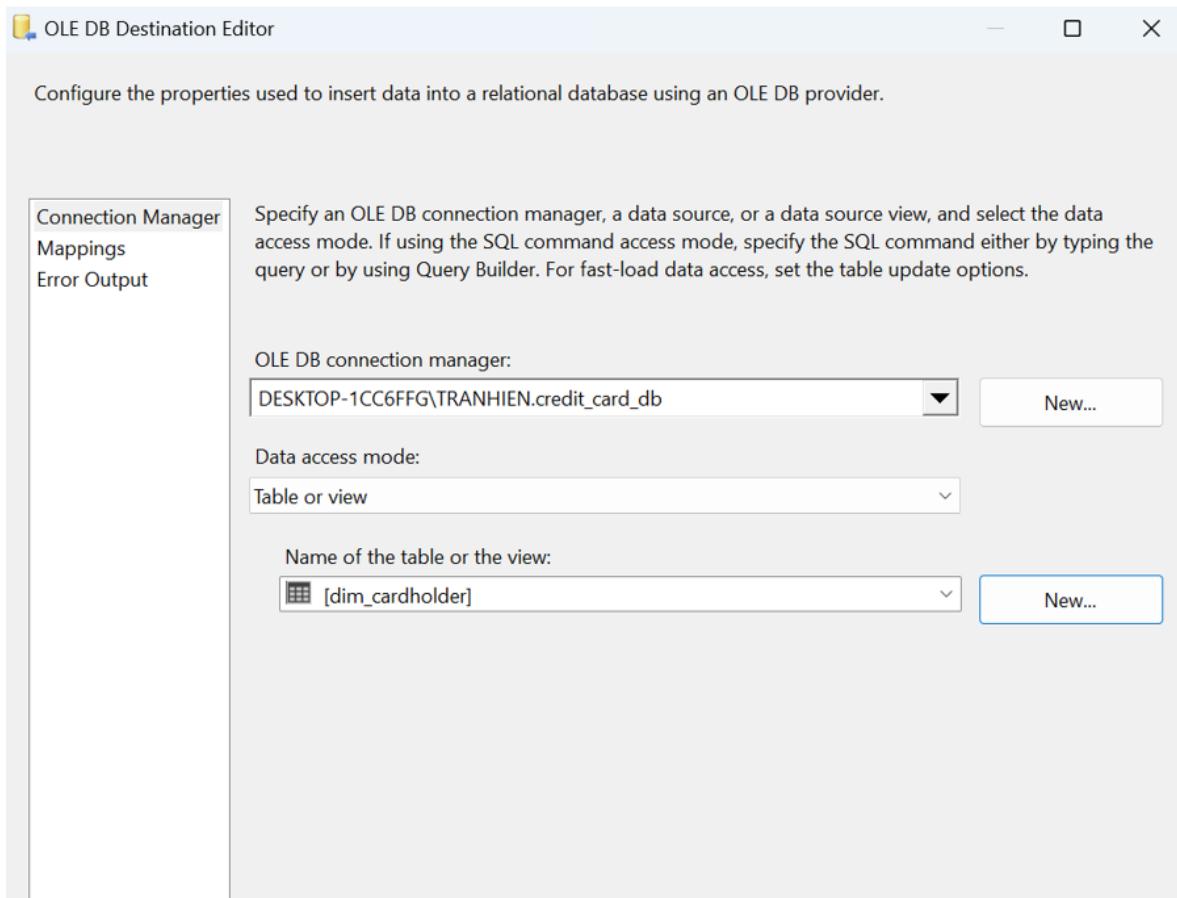
```
CREATE TABLE dim_cardholder (
```

```
    cardholder_id INT IDENTITY(1,1) PRIMARY KEY, -- Surrogate key
    [name] VARCHAR(255),
    gender VARCHAR(1),

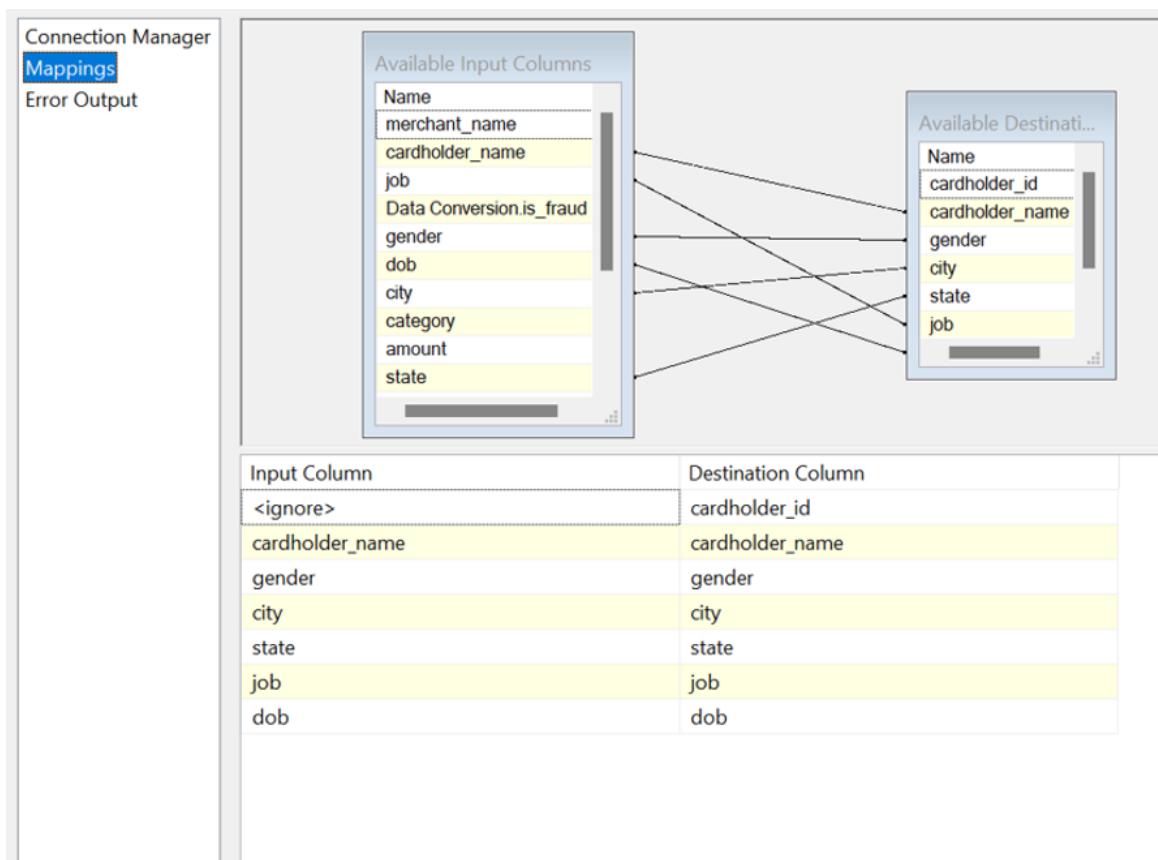
    city VARCHAR(255),
    [state] VARCHAR(255),
    job VARCHAR(255),
    dob DATETIME);
```



Bước 3: Chọn Connection đã tạo

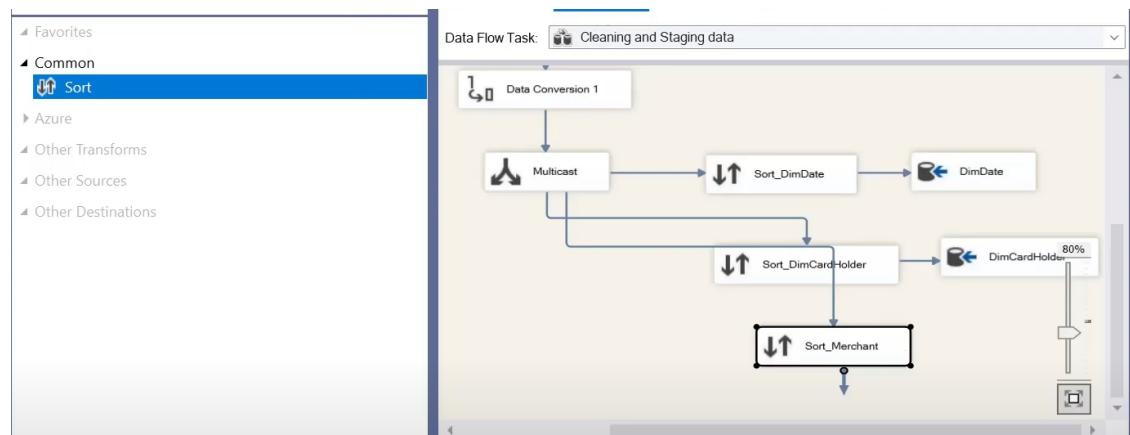


Bước 4: Tiếp đến ta cần chọn mục Mappings để xem xét việc ánh xạ các cột dữ liệu. Chọn OK để hoàn tất thiết lập.



### 2.5.2.3 Tạo bảng DimMerchant

Bước 1: Tạo một Sort mới, đổi tên là Sort\_Merchant để lấy ra các cột dữ liệu cần thiết. Chọn Sort, và chọn các thuộc tính của bảng DimMerchant. Tick vào phần Remove rows with duplicate sort values) để xóa các dòng dữ liệu trùng nhau, sau đó chọn OK.



Sort Transformation Editor

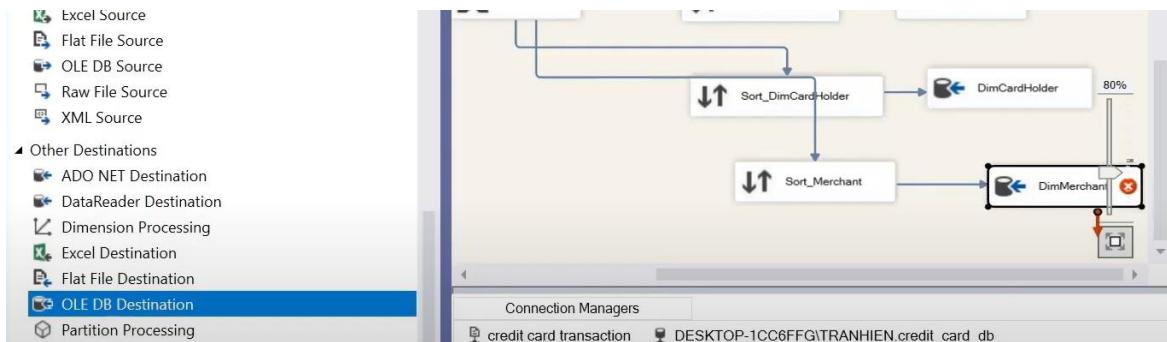
Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

Available Input Columns

Name	Pass T...
merchant	<input type="checkbox"/>
category	<input checked="" type="checkbox"/>
amt	<input type="checkbox"/>
first	<input type="checkbox"/>
last	<input type="checkbox"/>
gender	<input checked="" type="checkbox"/>
city	<input checked="" type="checkbox"/>

Input Column	Output Alias	Sort Type	Sort Order	Conn...
merchant_name	merchant_name	ascending	1	

Bước 2: Tạo mới một OLE DB Destination, đổi tên thành DimMerchant và nối Sort\_Merchant tới OLE DB Destination này.

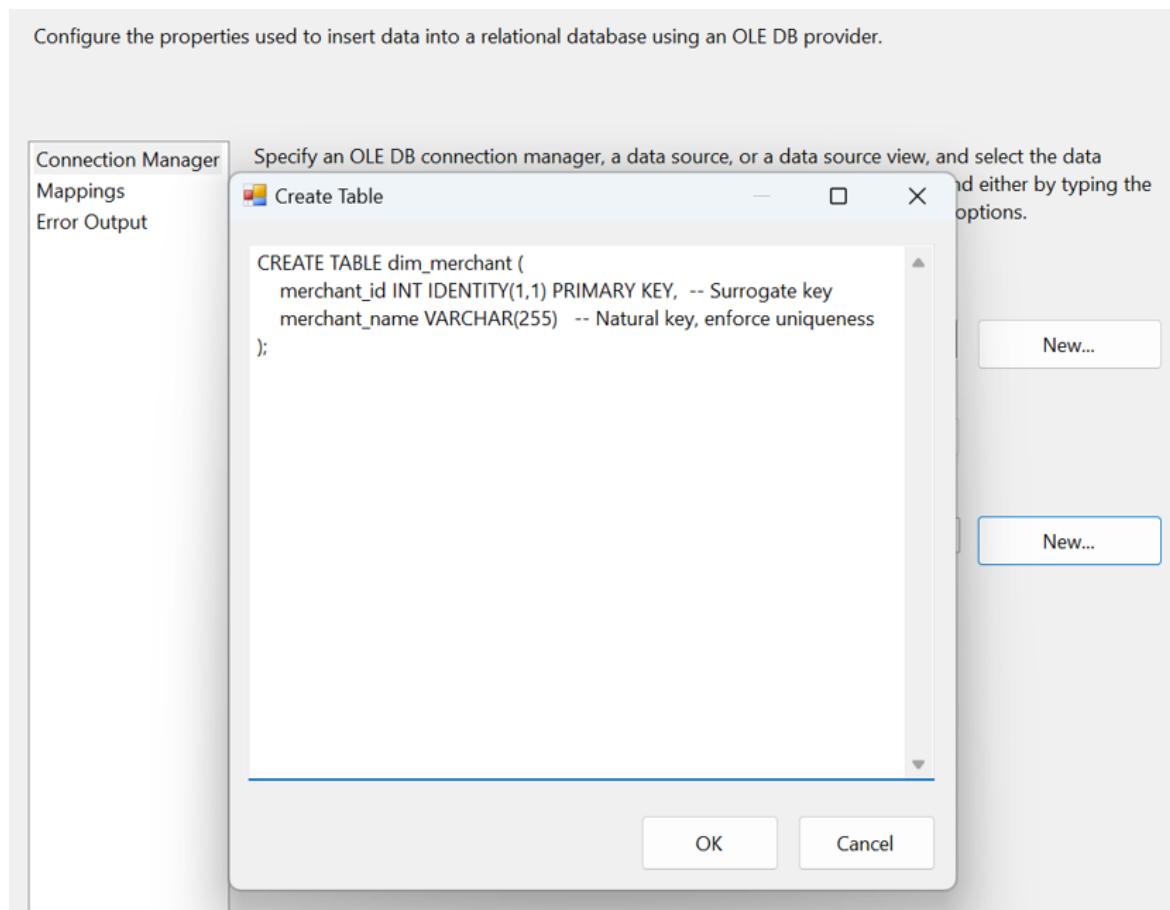


Chọn vào DimMerchant, chọn New... để tạo mới bảng.

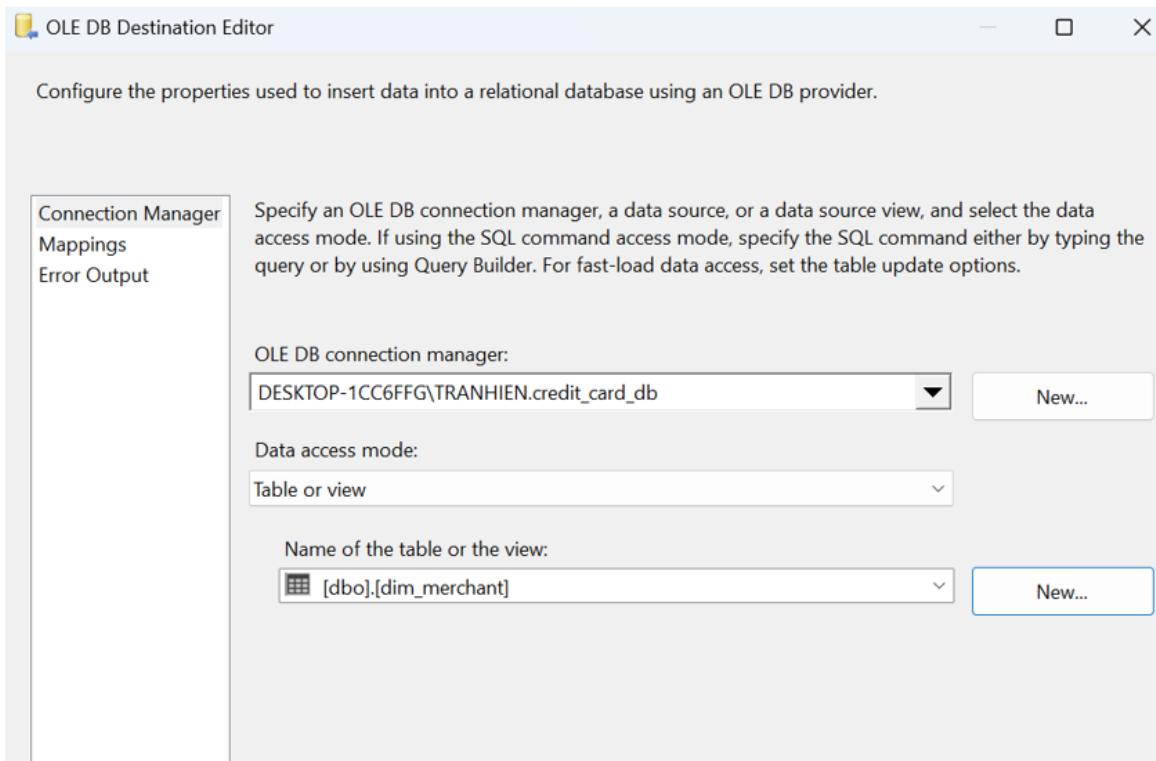
Code tạo bảng:

```
CREATE TABLE dim_merchant (
```

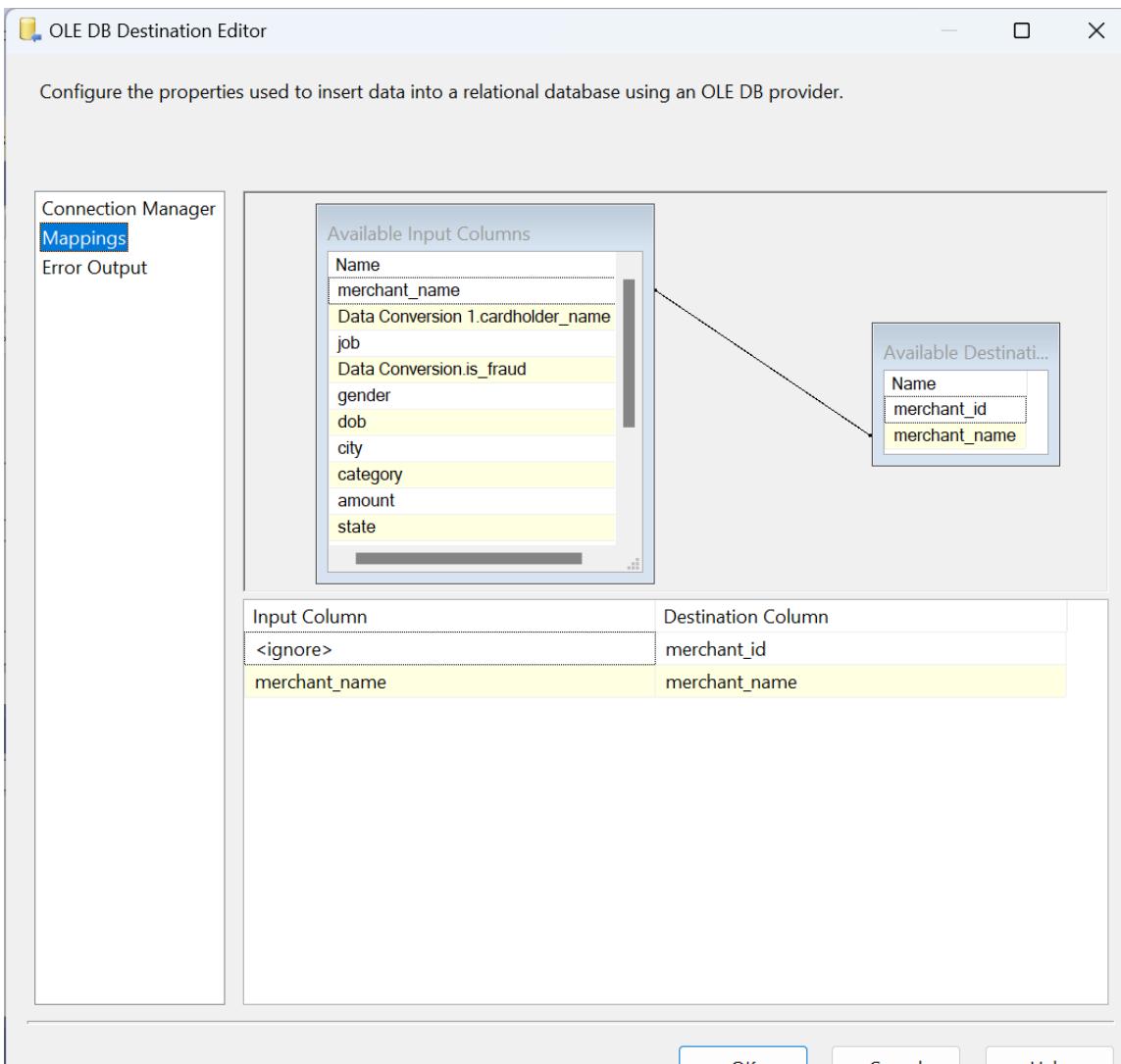
```
    merchant_id INT IDENTITY(1,1) PRIMARY KEY, -- Surrogate key
    merchant_name VARCHAR(255) UNIQUE NOT NULL -- Natural key, enforce
uniqueness
);
```



Bước 3: Chọn Connection đã tạo

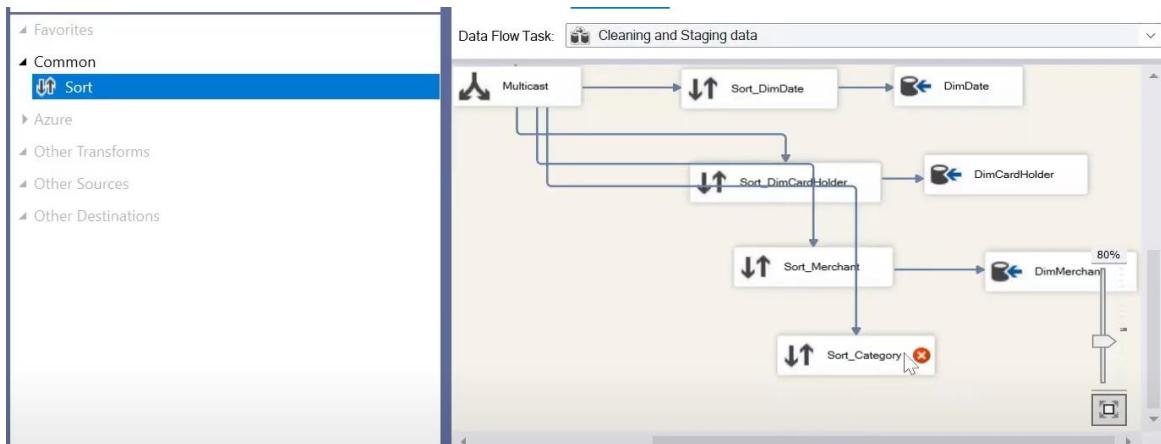


Bước 4: Tiếp đến ta cần chọn mục Mappings để xem xét việc ánh xạ các cột dữ liệu. Chọn OK để hoàn tất thiết lập.



#### 2.5.2.4 Tạo bảng DimCategory

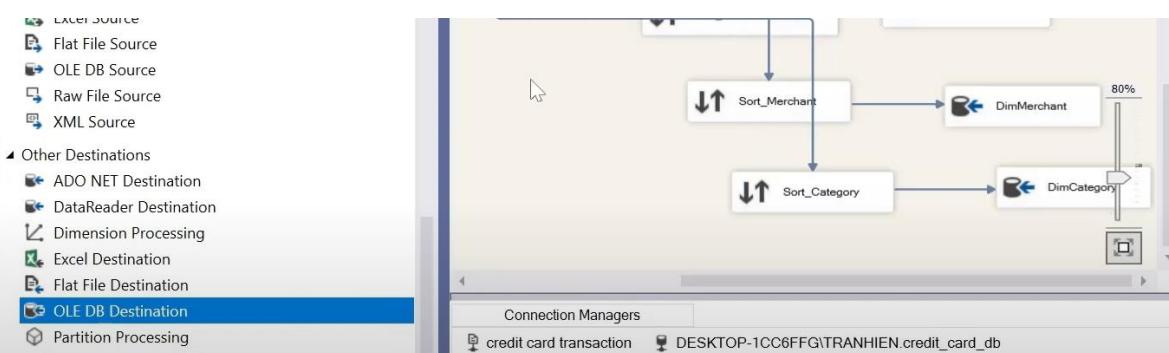
Bước 1: Tạo một Sort mới, đổi tên là Sort\_Category để lấy ra các cột dữ liệu cần thiết. Chọn Sort, và chọn các thuộc tính của bảng DimCategory. Tick vào phần Remove rows with duplicate sort values) để xóa các dòng dữ liệu trùng nhau, sau đó chọn OK



Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

Input Column	Output Alias	Sort Type	Sort Order	Comparison Flags
category	category	ascending	1	

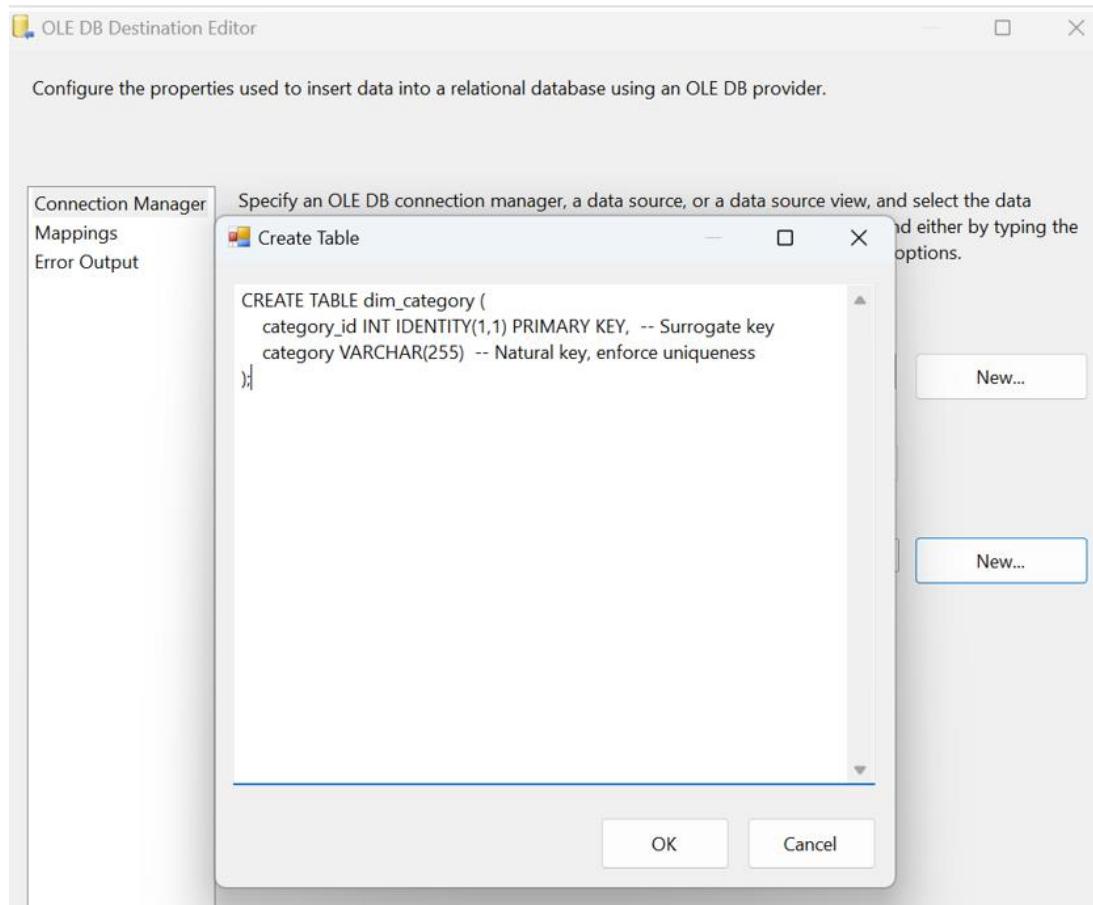
Bước 2: Tạo mới một OLE DB Destination, đổi tên thành DimCategory và nối Sort\_Category tới OLE DB Destination này.



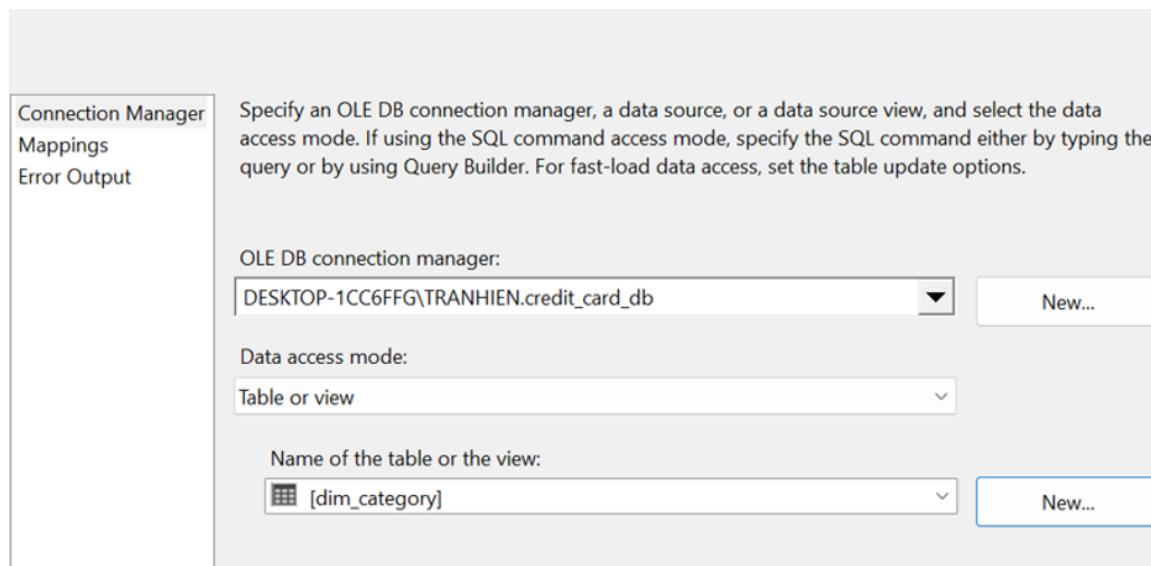
Chọn vào DimCategory, chọn New... để tạo mới bảng.

Code tạo bảng:

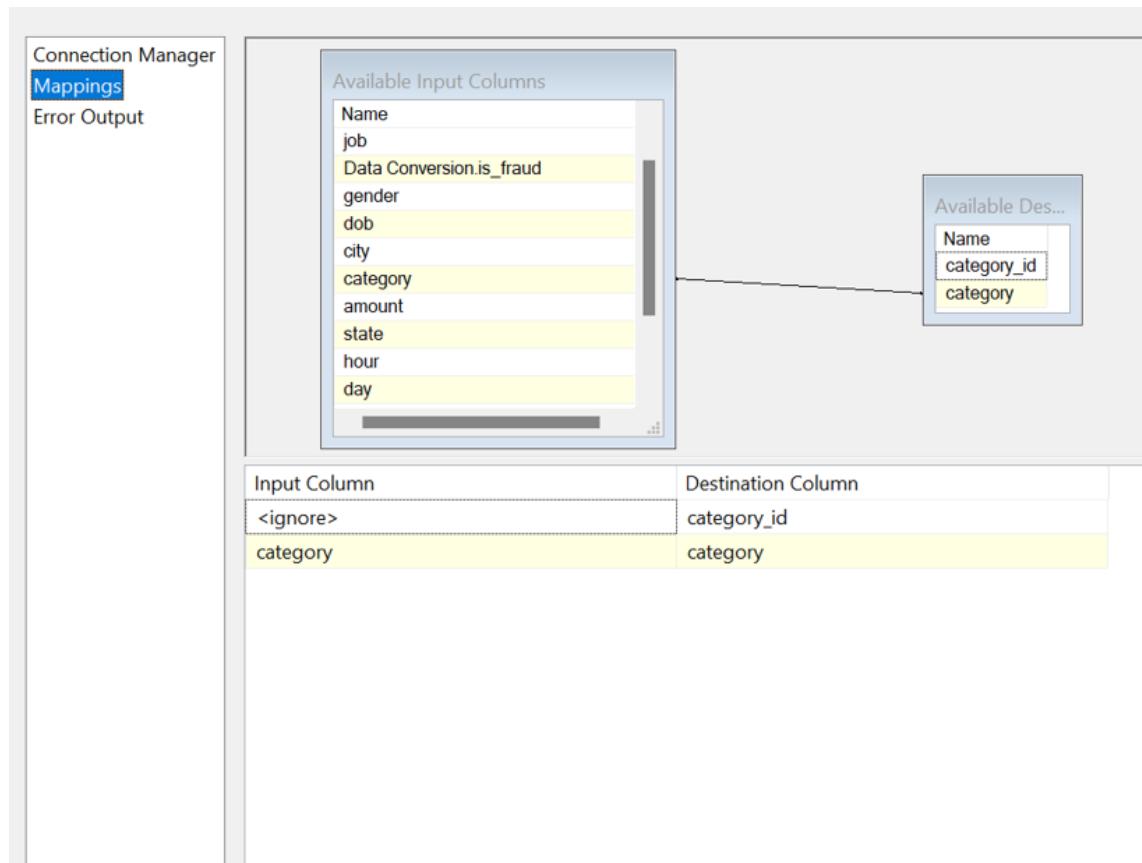
```
CREATE TABLE dim_category (
    category_id INT IDENTITY(1,1) PRIMARY KEY, -- Surrogate key
    category VARCHAR(255) UNIQUE NOT NULL -- Natural key, enforce uniqueness
);
```



Bước 3: Chọn Connection đã tạo

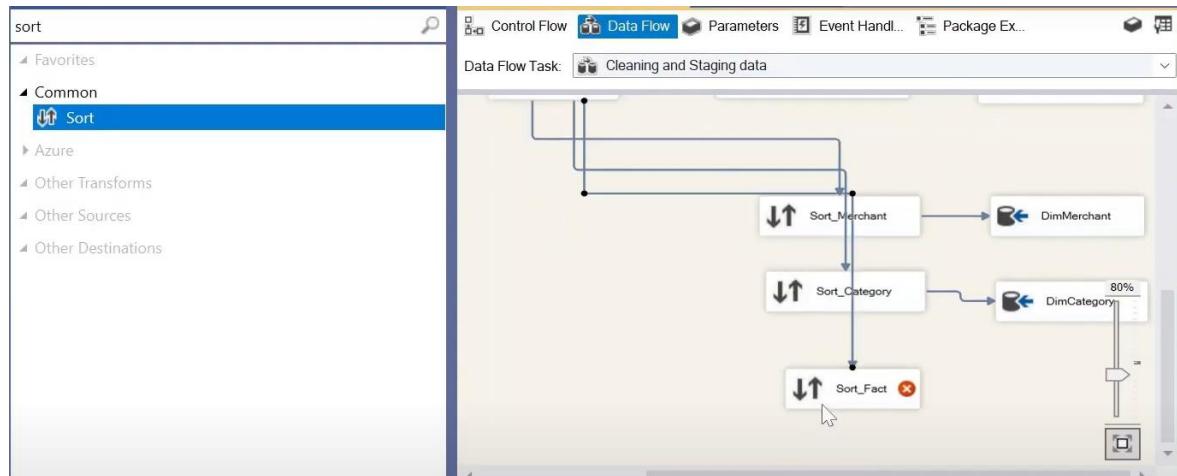


Bước 4: Tiếp đến ta cần chọn mục Mappings để xem xét việc ánh xạ các cột dữ liệu. Chọn OK để hoàn tất thiết lập.



### 2.5.2.5 Tạo bảng StagingFact

Bước 1: Tạo một Sort mới, đổi tên là Sort\_Fact để lấy ra các cột dữ liệu cần thiết. Chọn Sort, và chọn các thuộc tính của bảng StagingFact. Tick vào phần Remove rows with duplicate sort values) để xóa các dòng dữ liệu trùng nhau, sau đó chọn OK



Sort Transformation Editor

Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

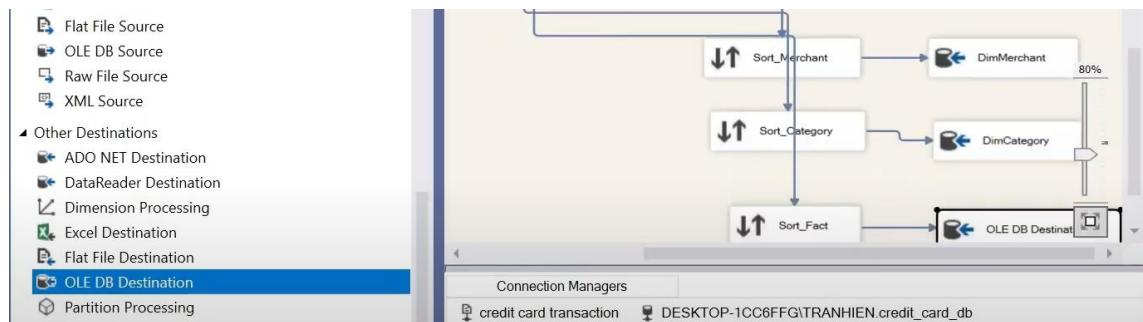
Available Input Columns  
 Name Pass T...  
 merchant

Input Column	Output Alias	Sort Type	Sort Order	Comparison Flags
merchant_name	merchant_name	ascending	1	
Data Conversion 1.cardholder...	cardholder_name	ascending	2	
job	job	ascending	3	
Data Conversion.is_fraud	is_fraud	ascending	4	
gender	gender	ascending	5	
dob	dob	ascending	6	
city	city	ascending	7	
category	category	ascending	8	
amount	amount	ascending	9	
state	state	ascending	10	
hour	hour	ascending	11	
day	day	ascending	12	
month	month	ascending	13	
year	year	ascending	14	
quarter	quarter	ascending	15	
distance	distance	ascending	16	

Remove rows with duplicate sort values

OK Cancel Help

Bước 2: Tạo mới một OLE DB Destination, đổi tên thành StagingFact và nối Sort\_Fact tới OLE DB Destination này.

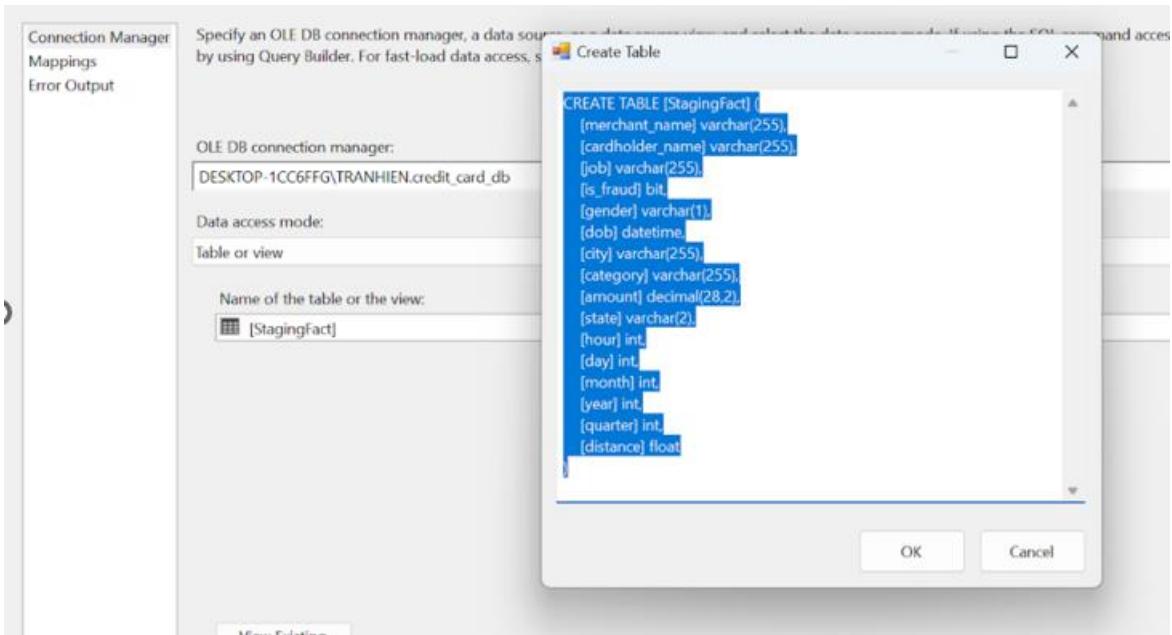


Chọn vào StagingFact, chọn New... để tạo mới bảng.

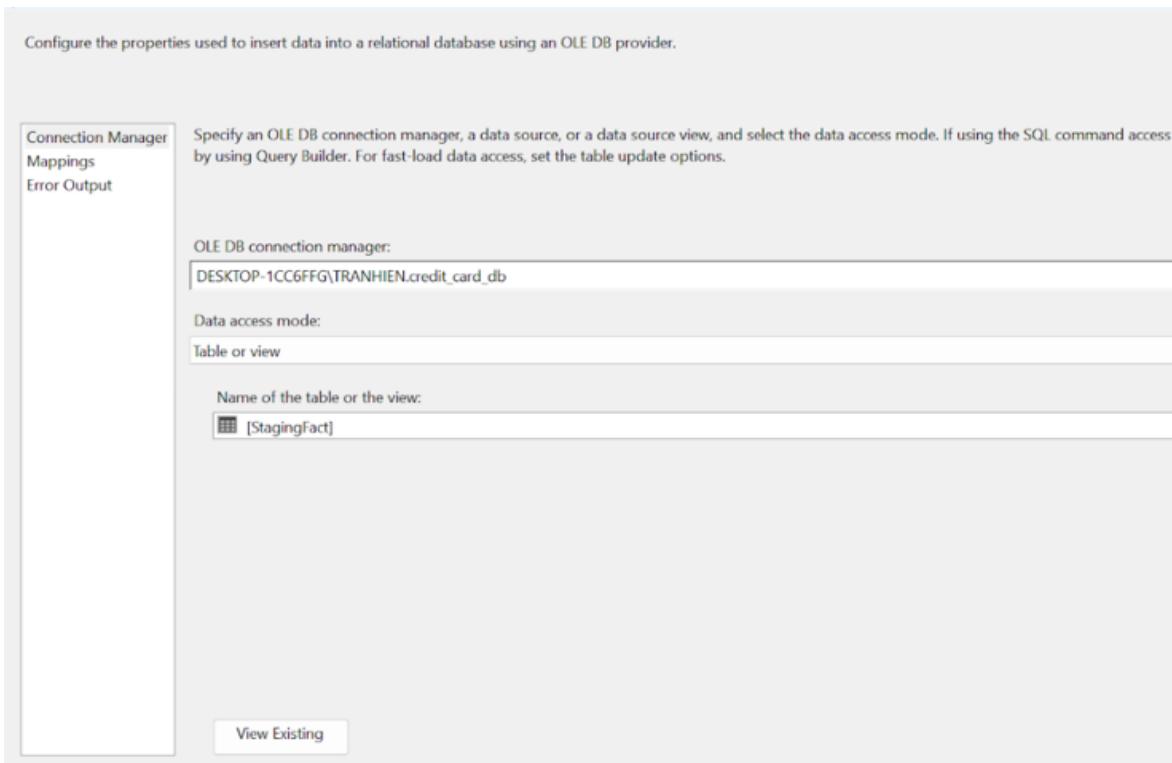
Code tạo bảng:

```

CREATE TABLE [staging_fact] (
    [merchant_name] varchar(255),
    [merch_long] decimal(28,7),
    [merch_lat] decimal(28,7),
    [long] decimal(28,7),
    [lat] decimal(28,7),
    [last_name] varchar(255),
    [job] varchar(255),
    [is_fraud] bit,
    [gender] varchar(1),
    [first_name] varchar(255),
    [dob] datetime,
    [city] varchar(255),
    [category] varchar(255),
    [amount] real,
    [trans_date_trans_time] datetime,
    [state] varchar(255)
);
    
```

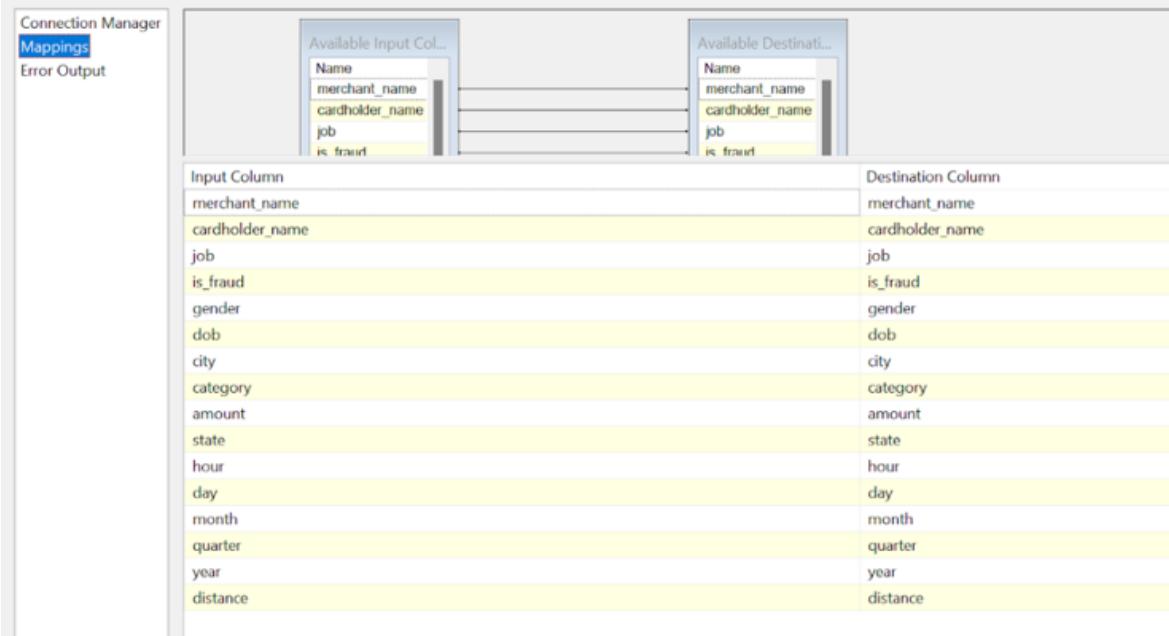


### Bước 3: Chọn Connection đã tạo

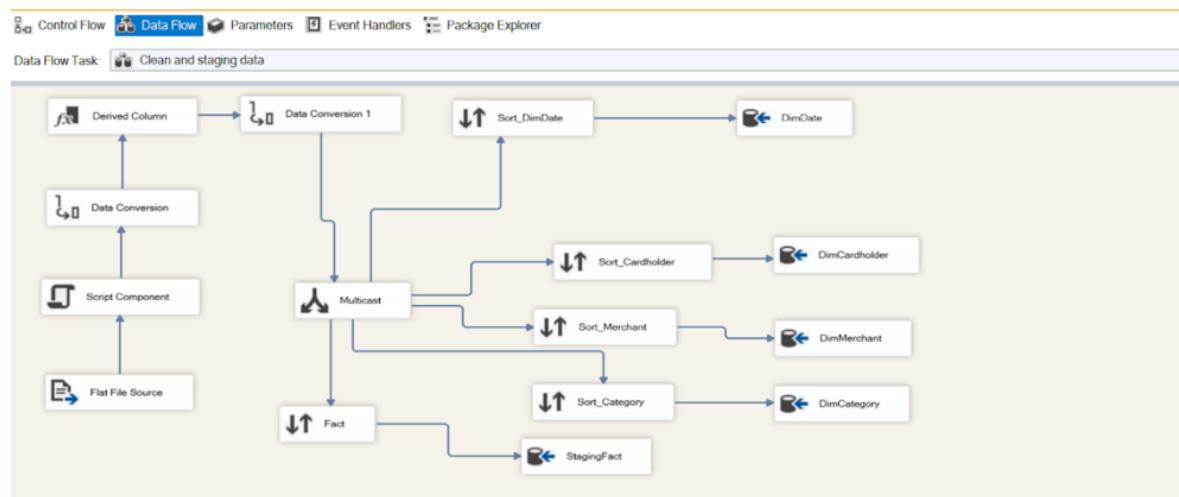


### Bước 4: Tiếp đến ta cần chọn mục Mappings để xem xét việc ánh xạ các cột dữ liệu. Chọn OK để hoàn tất thiết lập.

Configure the properties used to insert data into a relational database using an OLE DB provider.

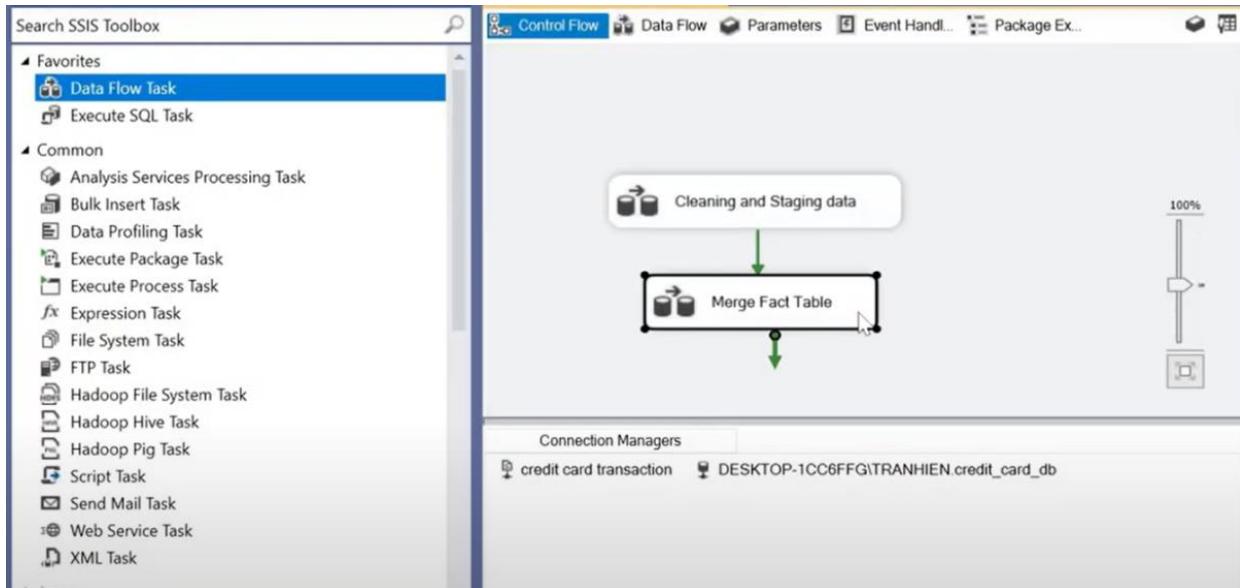


## Data flow Task hoàn chỉnh:

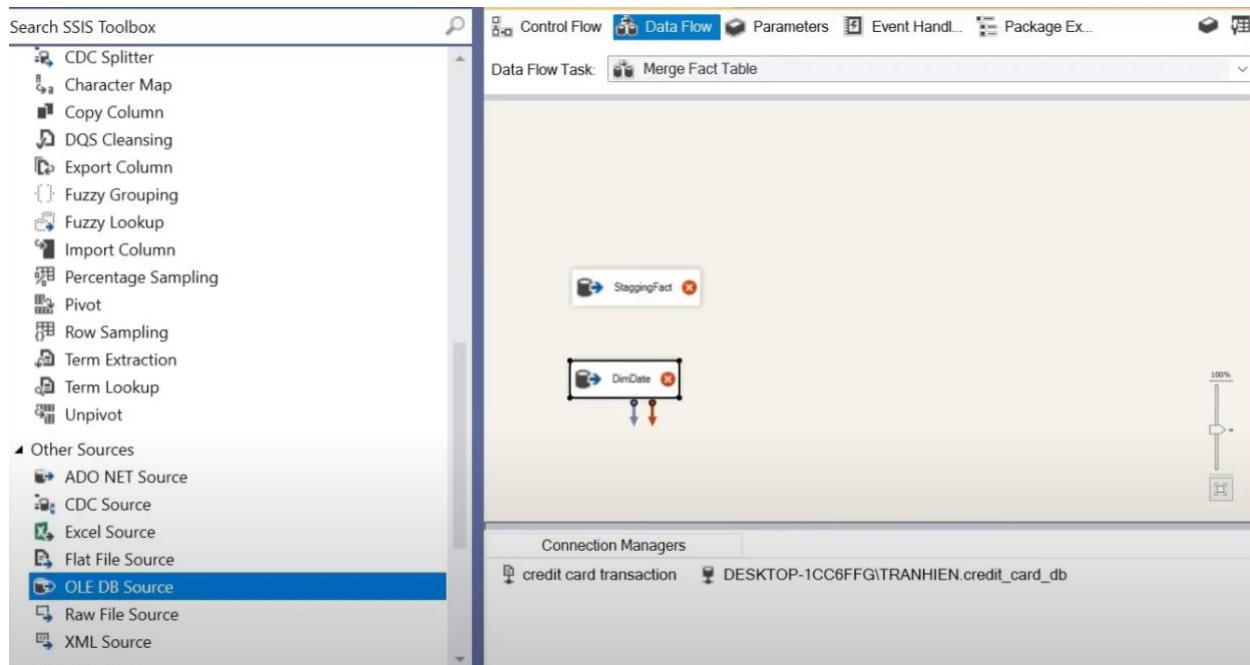


## 2.6 Merge Table

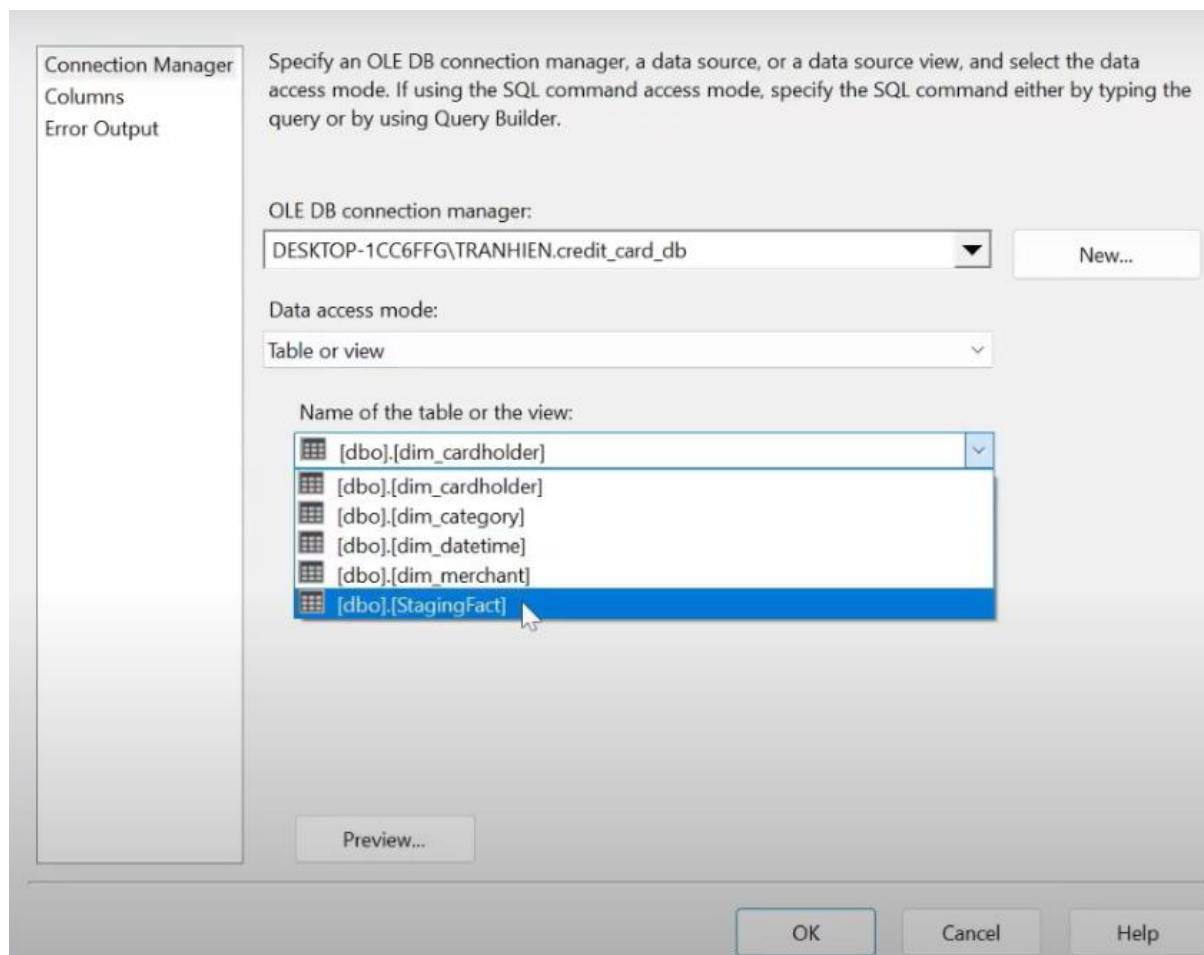
Bước 1: Tạo một Data Flow Task mới, chuột phải vào Data Flow Task rồi chọn Rename, đổi tên thành Merge Table



Bước 2: Trong Data Flow Task, trong, tạo 2 OLE DB Source và đổi tên StaggingFact và DimDate



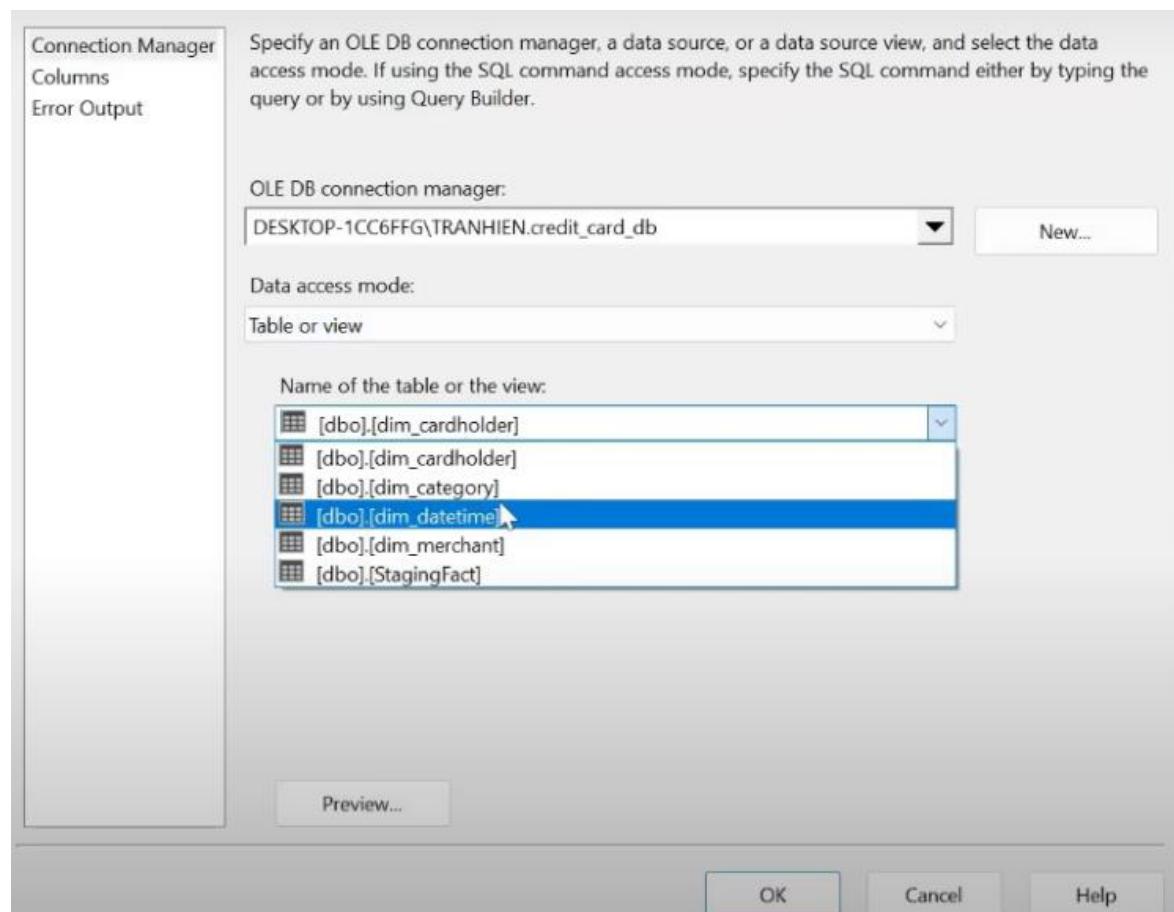
Bước 3: Chọn StaggingFact, sau đó chọn bảng StaggingFact đã tạo trước đó làm data source cho bảng mới này.



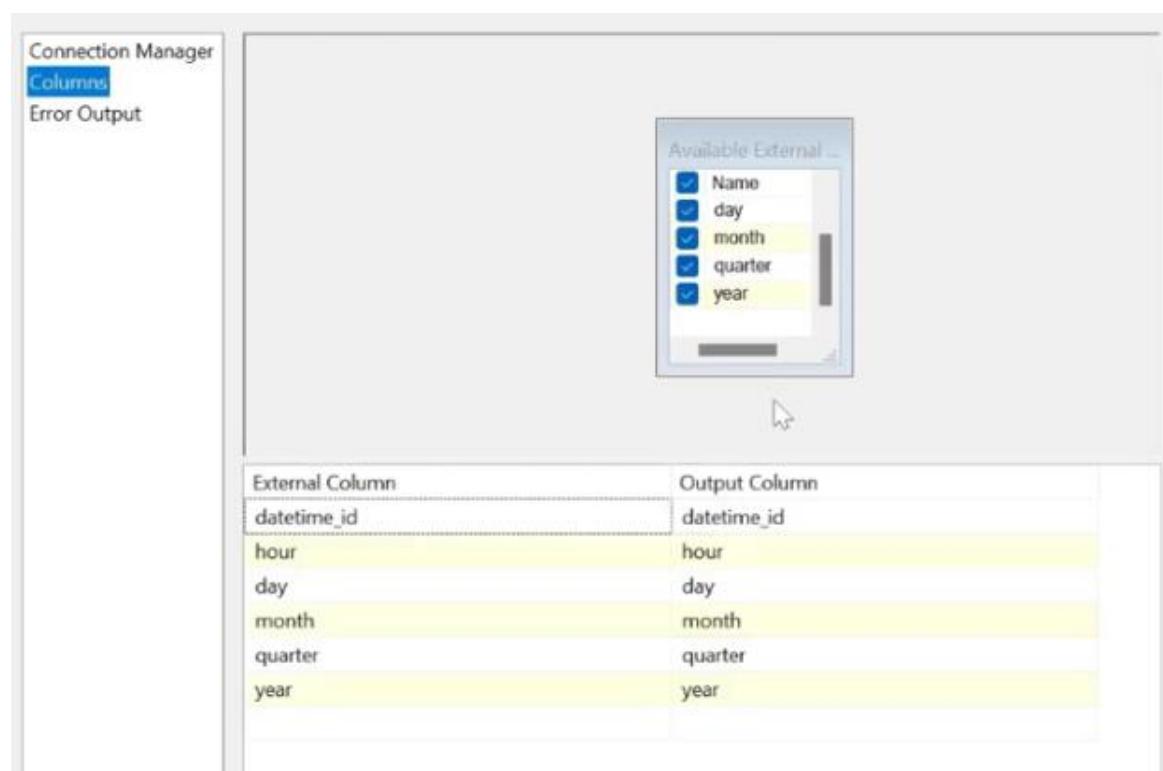
Bước 4: Chọn Column để xem các thuộc tính đã được ánh xạ đúng chưa. Nhấn OK để hoàn thành

External Column	Output Column
amount	amount
state	state
hour	hour
day	day
month	month
year	year
quarter	quarter
distance	distance

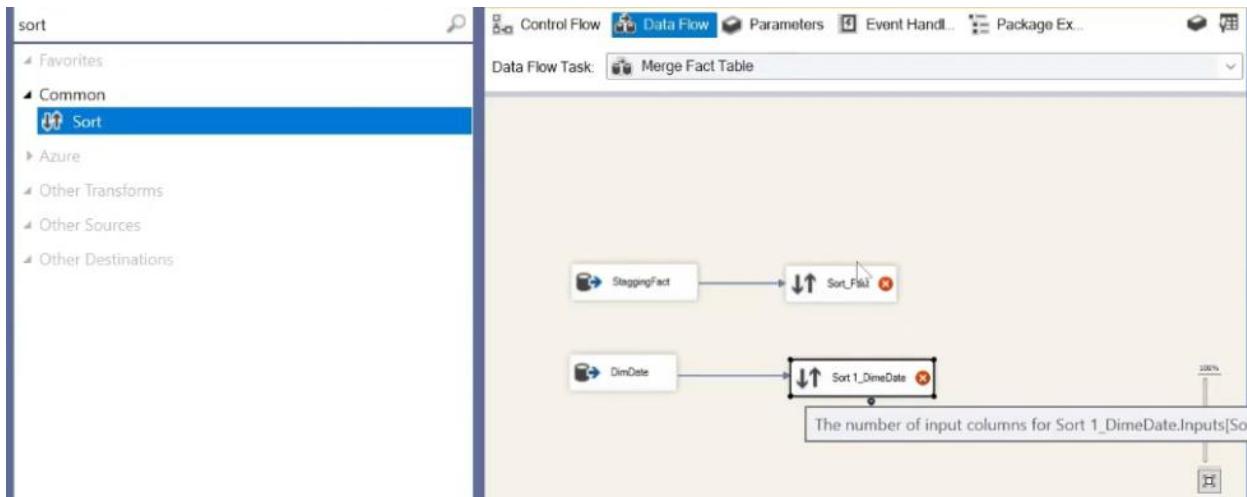
Bước 5: Chọn DimDate, sau đó chọn bảng dim\_datetime đã tạo trước đó làm data source cho bảng mới này.



Bước 6: Chọn Column để xem các thuộc tính đã được ánh xạ đúng chưa. Nhấn OK để hoàn thành



Bước 7: Tạo 2 Sort tương ứng với mỗi bảng, đổi tên thành Sort\_Fact và Sort\_DimDate và kết nối đến bảng tương ứng



Bước 8: Ở Sort\_Fact, chọn các cột theo thứ tự giống với bảng DimDate để chuẩn bị cho quá trình merge.

The screenshot shows the 'Available Input Columns' dialog and the 'Sort' configuration table for the 'Sort\_Fact' component.

**Available Input Columns Dialog:**

Name	Pass T...
state	<input checked="" type="checkbox"/>
hour	<input checked="" type="checkbox"/>
day	<input checked="" type="checkbox"/>
month	<input checked="" type="checkbox"/>
year	<input checked="" type="checkbox"/>
quarter	<input checked="" type="checkbox"/>
distance	<input type="checkbox"/>

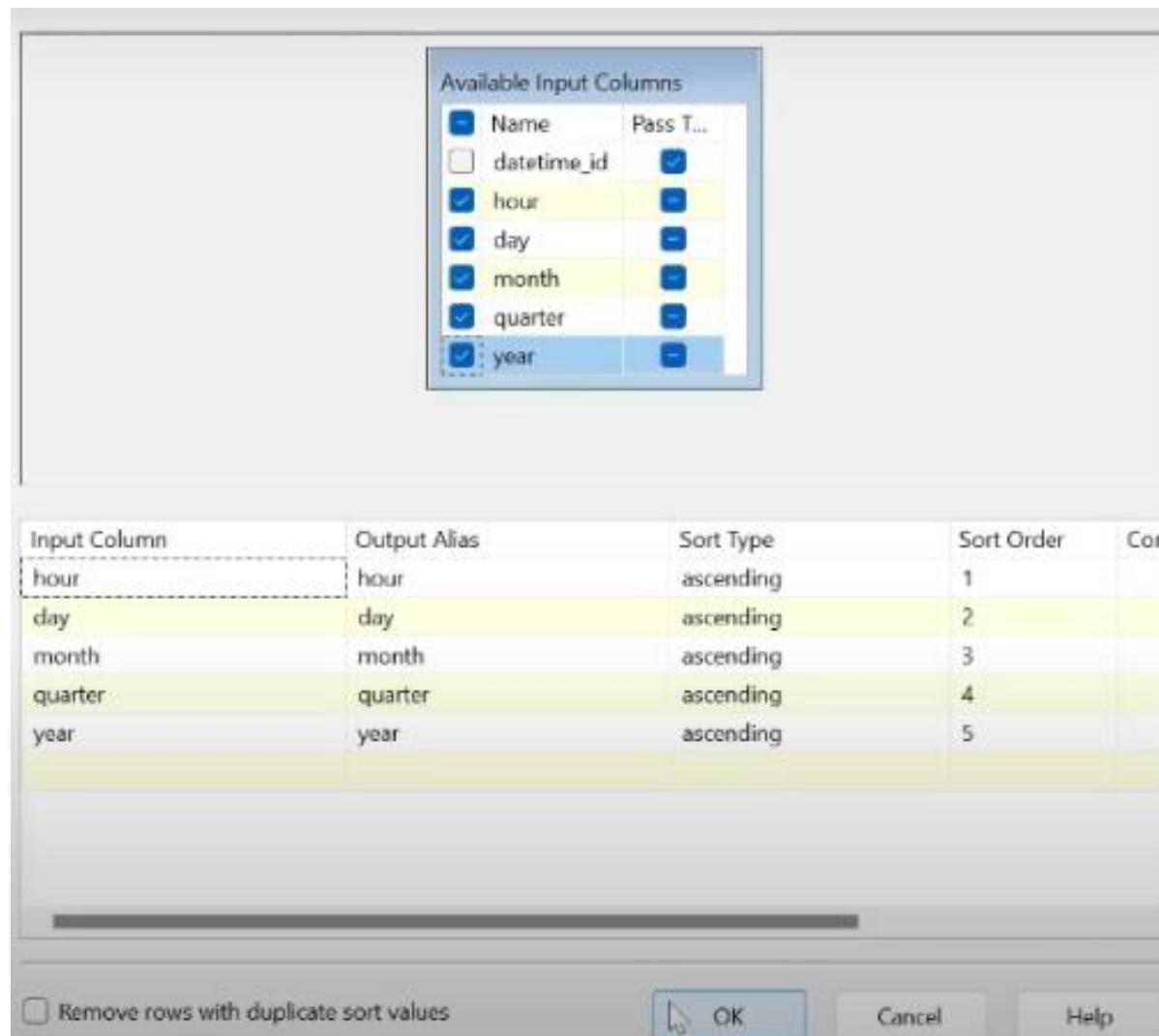
**Sort Configuration Table:**

Input Column	Output Alias	Sort Type	Sort Order	Conn...
hour	hour	ascending	1	
day	day	ascending	2	
month	month	ascending	3	
year	year	ascending	4	
quarter	quarter	ascending	5	

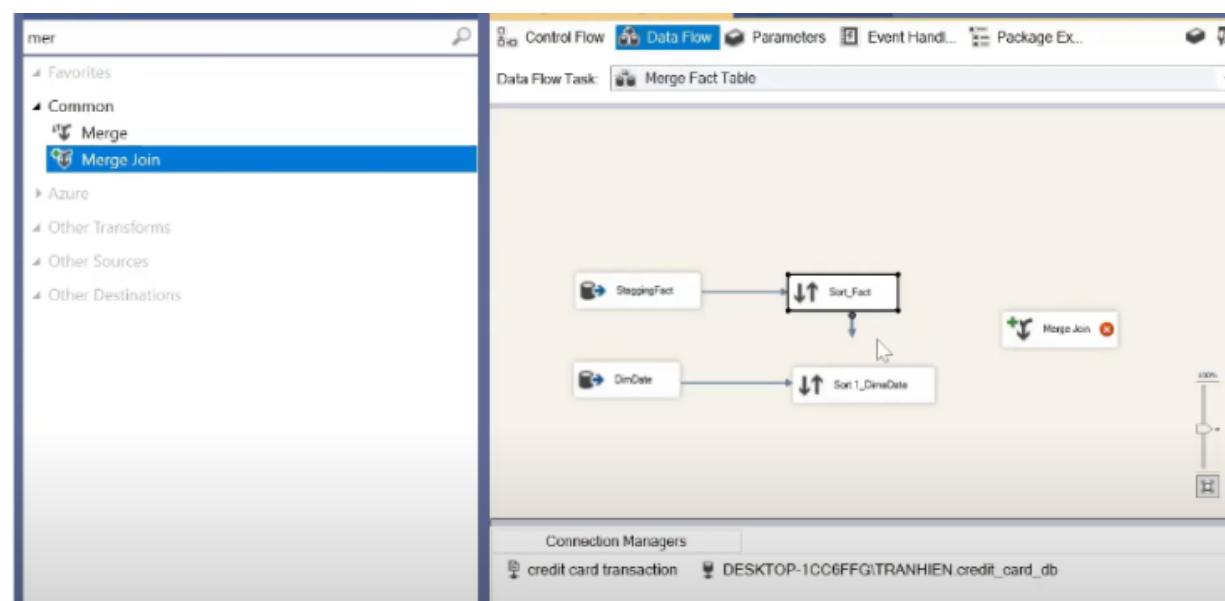
**Buttons at the bottom:**

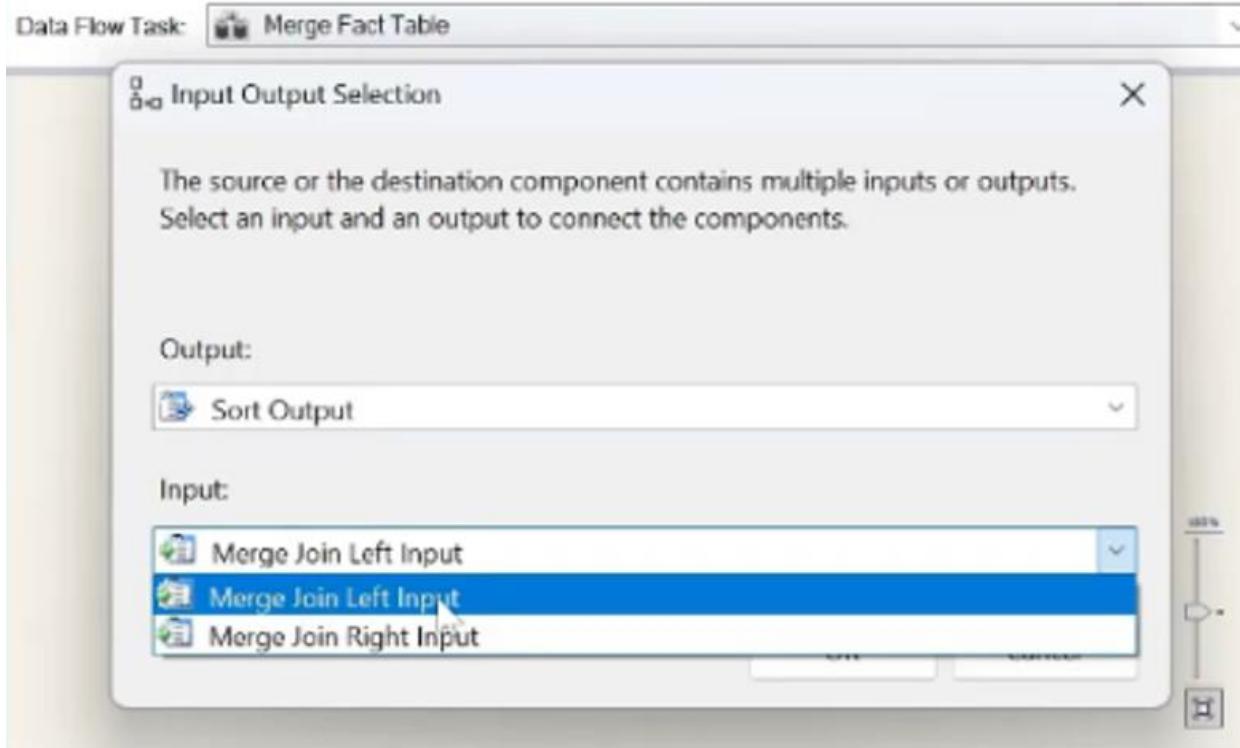
- Remove rows with duplicate sort values
- OK
- Cancel
- Help

Bước 9: Tương tự ta chọn các cột tương ứng cho Sort\_DimDate (lưu ý không chọn datetime\_id)



Bước 10: Tạo một Merge Join và nối với Sort\_Fact, tiếp theo ta chọn Merge Join, chọn Left Input để giữ lại toàn bộ các dòng trong bảng Fact bất kể có kết quả khi thực hiện phép kết trái với cột id của bảng DimDate hay không.





Bước 11:

Chọn vào Merge Join và nhấn Edit, một hộp thoại merge editor xuất hiện:

Chọn tất cả các cột của Sort nhưng không lấy các cột tương ứng với bảng DimDate.

Tiếp theo ta chọn datetime\_id ở Sort\_DimDate để merge vào StagingFact

Kết quả sau khi merge là bảng StagingFact không còn các thuộc tính của bảng DimDate và có thêm 1 thuộc tính mới là datetime\_id.

Join type: Inner join Swap Inputs

The screenshot shows the 'Sort' transformation configuration window. It has two main sections: 'Sort\_Fact' on the left and 'Sort 1\_DimeDate' on the right.

- Sort\_Fact:**

Name	Order	Join
Name	0	<input type="checkbox"/>
city	0	<input checked="" type="checkbox"/>
category	0	<input checked="" type="checkbox"/>
amount	0	<input type="checkbox"/>
state	0	<input type="checkbox"/>
hour	1	<input checked="" type="checkbox"/>
day	2	<input checked="" type="checkbox"/>
month	3	<input checked="" type="checkbox"/>
- Sort 1\_DimeDate:**

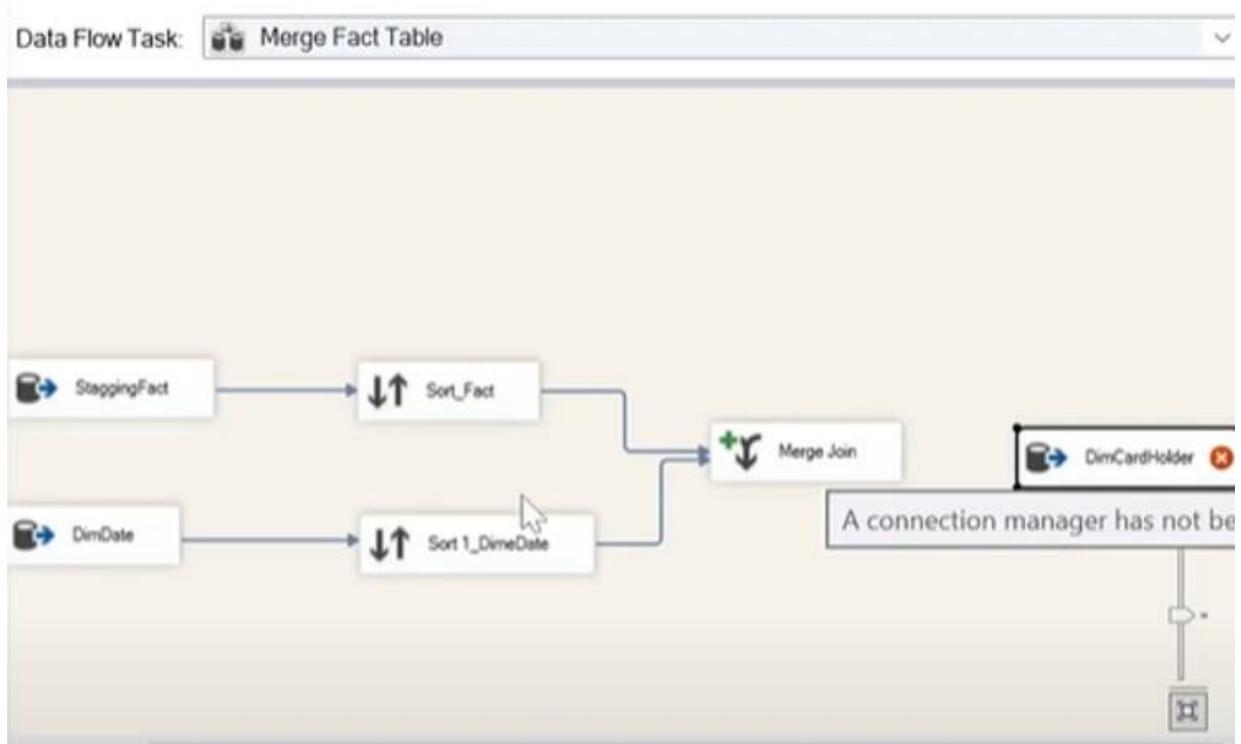
Name	Order	Join
Name	0	<input type="checkbox"/>
datetime_id	0	<input checked="" type="checkbox"/>
hour	1	<input checked="" type="checkbox"/>
day	2	<input checked="" type="checkbox"/>
month	3	<input checked="" type="checkbox"/>
quarter	4	<input checked="" type="checkbox"/>
year	5	<input checked="" type="checkbox"/>

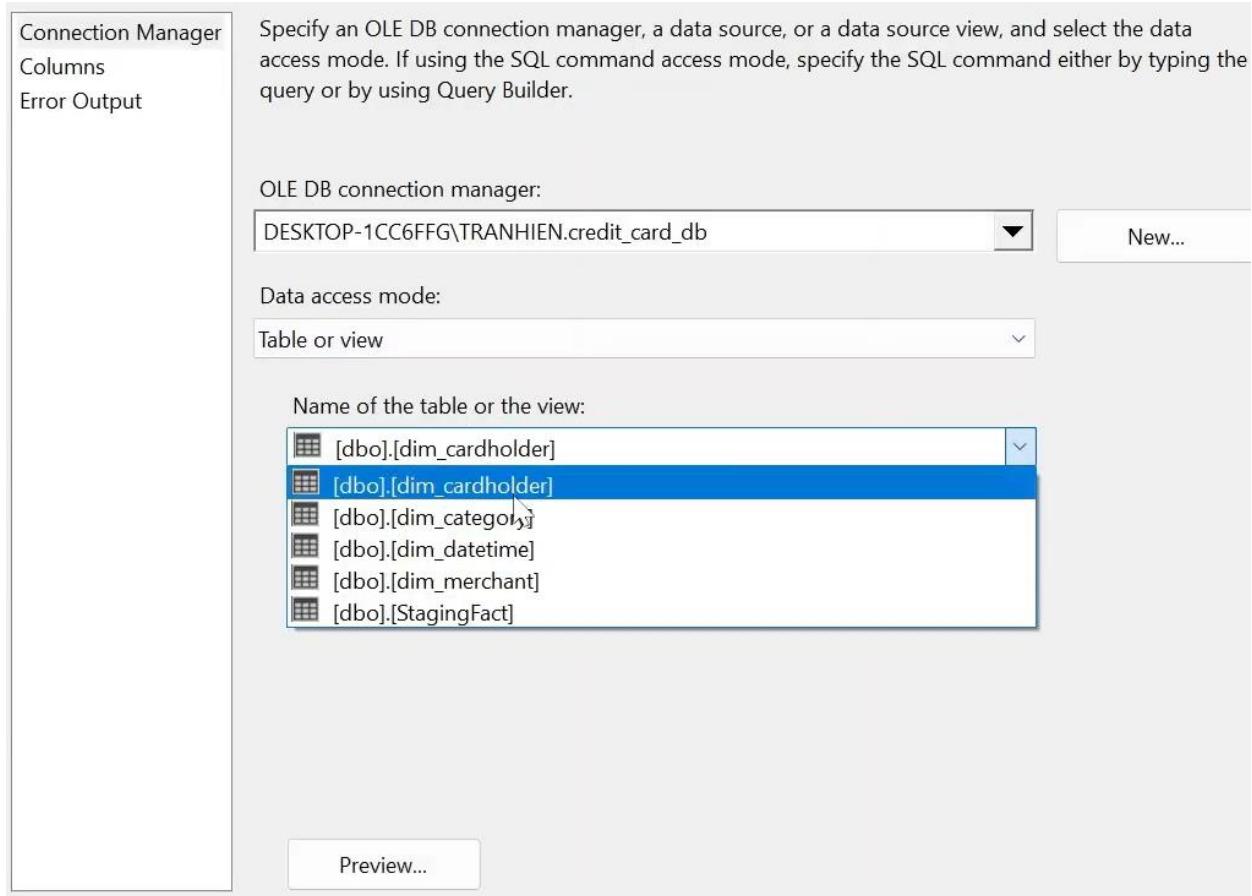
Below these sections is a table mapping input columns to output aliases:

Input	Input Column	Output Alias
Sort_Fact	is_fraud	is_fraud
Sort_Fact	gender	gender
Sort_Fact	dob	dob
Sort_Fact	city	city
Sort_Fact	category	category
Sort_Fact	amount	amount
Sort_Fact	state	state
Sort_Fact	distance	distance
Sort 1_DimeD...	datetime_id	datetime_id

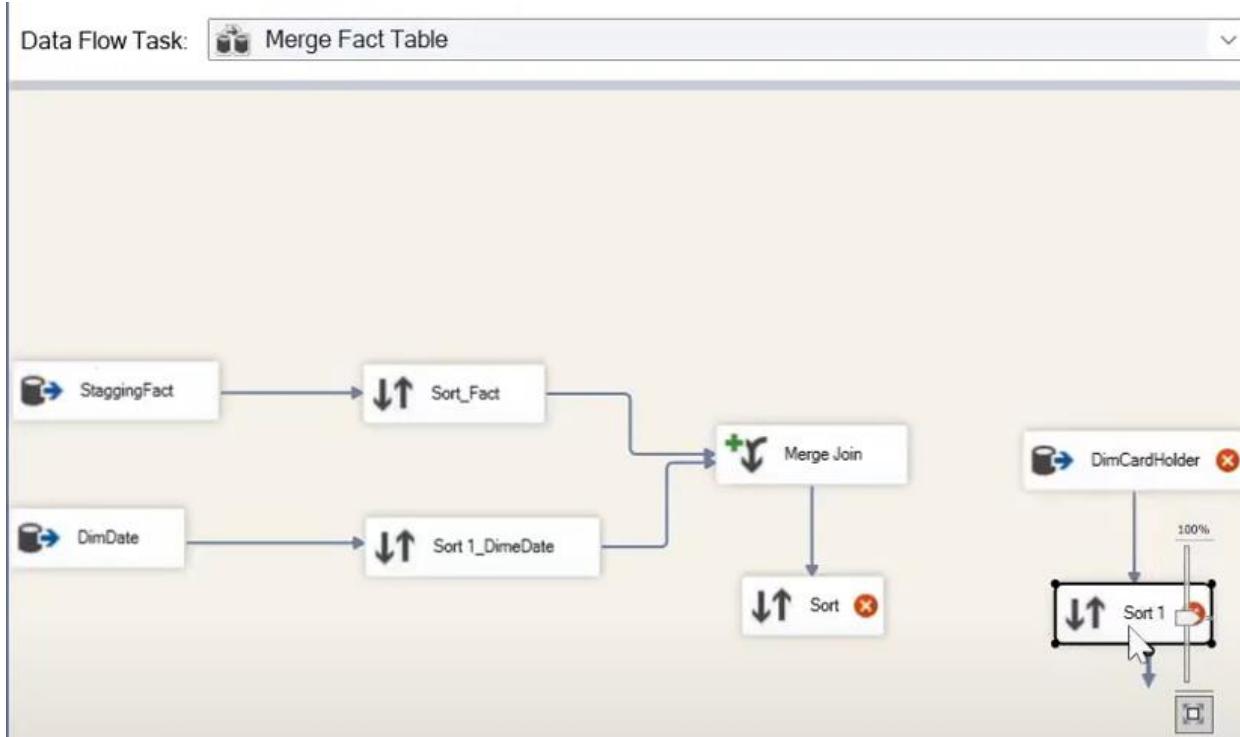
At the bottom right are buttons for OK, Cancel, and Help.

Bước 12: Tạo OLE DB Source mới và đổi thành DimCardHolder, sau đó chọn bảng dim\_cardholder đã tạo trước đó làm data source cho bảng mới này.





Bước 13: Tạo 2 Sort tương ứng với mỗi bảng và kết nối đến bảng tương ứng cho quá trình merge



Bước 14: Ở Sort, chọn các cột theo thứ tự giống với bảng DimCardHolder để chuẩn bị cho quá trình

Available Input Columns

Name	Pass T...
<input checked="" type="checkbox"/> cardholder_name	<input type="checkbox"/>
<input checked="" type="checkbox"/> dob	<input type="checkbox"/>
<input checked="" type="checkbox"/> city	<input type="checkbox"/>
<input type="checkbox"/> category	<input type="checkbox"/>
<input type="checkbox"/> amount	<input type="checkbox"/>
<input checked="" type="checkbox"/> state	<input type="checkbox"/>
<input type="checkbox"/> distance	<input type="checkbox"/>
<input type="checkbox"/> datetime_id	<input type="checkbox"/>

Input Column	Output Alias	Sort Type	Sort Order	Con
cardholder_name	cardholder_name	ascending	1	
job	job	ascending	2	
gender	gender	ascending	3	
dob	dob	ascending	4	
city	city	ascending	5	
state	state	ascending	6	

Bước 15: Ở Sort1, chọn các cột tương ứng với DimCardHolder (không chọn cardholder\_id)

Available Input Columns

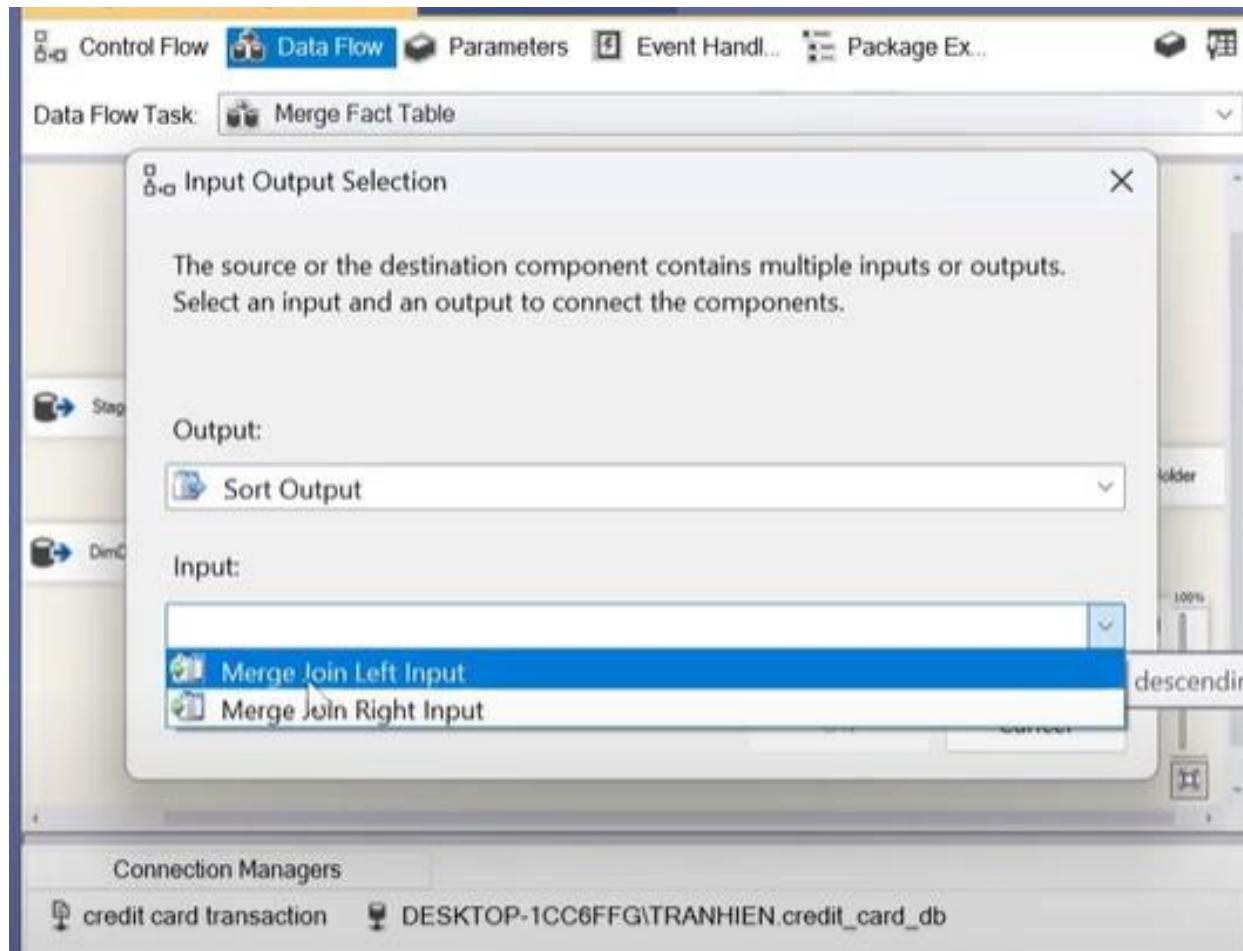
Name	Pass T...
<input type="checkbox"/> cardholder_id	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> cardholder_name	<input type="checkbox"/>
<input checked="" type="checkbox"/> job	<input type="checkbox"/>
<input checked="" type="checkbox"/> gender	<input type="checkbox"/>
<input checked="" type="checkbox"/> dob	<input type="checkbox"/>
<input checked="" type="checkbox"/> city	<input type="checkbox"/>

Input Column	Output Alias	Sort Type	Sort Order	Con
cardholder_name	cardholder_name	ascending	1	
job	job	ascending	2	
gender	gender	ascending	3	
dob	dob	ascending	4	
city	city	ascending	5	
state	state	ascending	6	

Bước 16: Tạo mới Merge Join 1 và nối với Sort\_Fact, tiếp theo ta chọn Merge Join 1 rồi chọn

Left Input để giữ lại toàn bộ các dòng trong bảng Fact bất kể có kết quả khi thực hiện phép kết trái với cột ID của bảng DimCardHolder hay không.

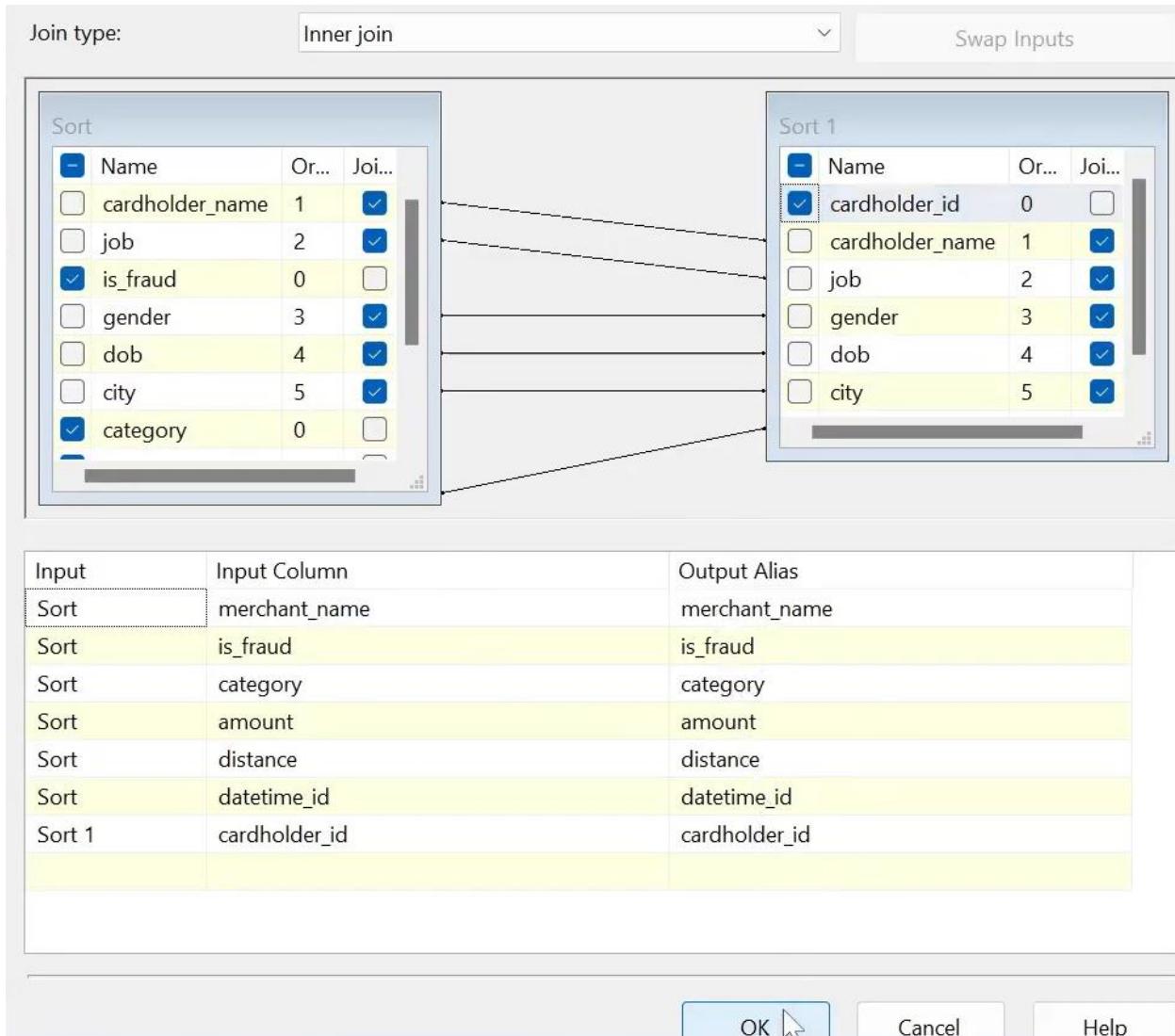


Bước 17:

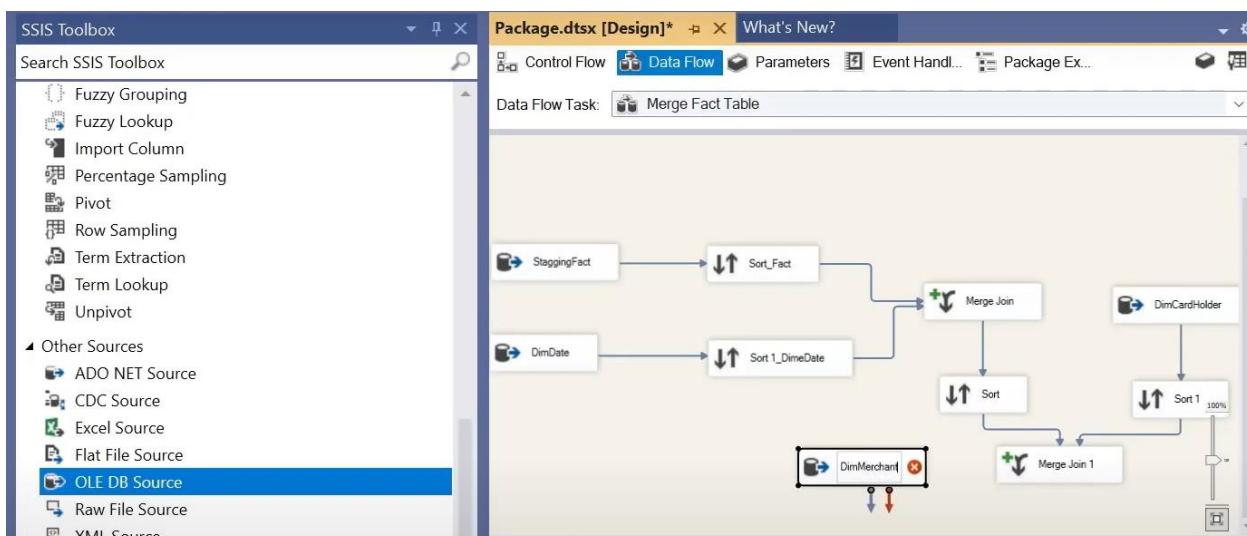
Chọn vào Merge Join 1, một hộp thoại merge editor xuất hiện: ở đây ta tick chọn tất cả các cột của Sort nhưng không lấy các cột tương ứng với bảng DimCardholder.

Tiếp theo ta chọn cardholder\_id ở Sort\_DimDate để merge vào StaggingFact

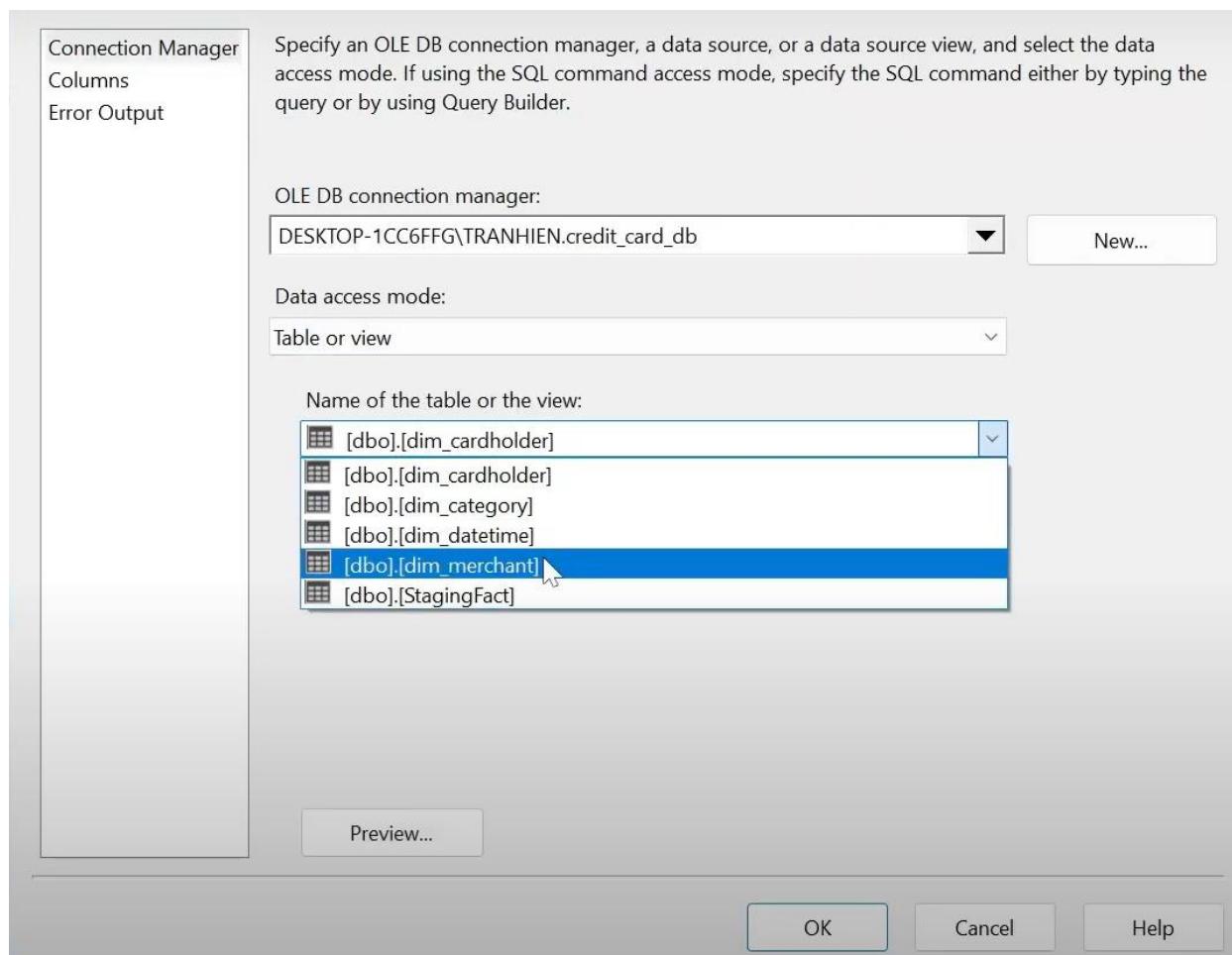
Kết quả sau khi merge là bảng StaggingFact không còn các thuộc tính của bảng CardHolder và có thêm 1 thuộc tính mới là cardholder\_id.



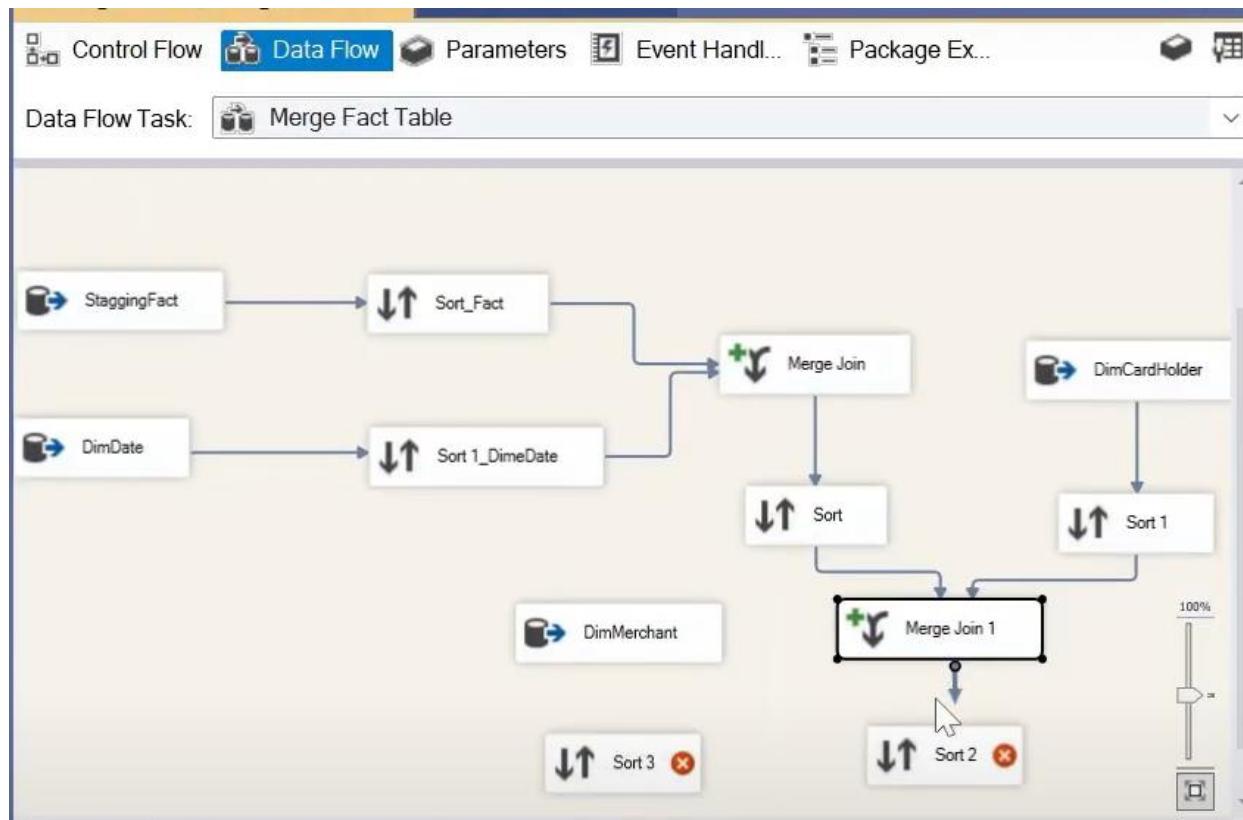
Bước 18: Tạo OLE DB Source mới và đổi thành DimMerchant, sau đó chọn bảng dim\_merchant đã tạo trước đó làm data source cho bảng mới này.



Chọn Column để xem các thuộc tính đã được ánh xạ đúng chưa. Nhấn OK để hoàn thành



Bước 19: Tạo 2 Sort tương ứng với mỗi bảng và kết nối đến bảng tương ứng để chuẩn bị cho quá trình merge



Bước 20: Ở Sort 2, chọn các cột theo thứ tự giống với bảng DimMerchant

The screenshot shows the 'Available Input Columns' dialog box with the following columns:

Name	Pass T...
merchant_name	<input checked="" type="checkbox"/>
is_fraud	<input checked="" type="checkbox"/>
category	<input checked="" type="checkbox"/>
amount	<input checked="" type="checkbox"/>
distance	<input checked="" type="checkbox"/>
datetime_id	<input checked="" type="checkbox"/>

Below the dialog is a table mapping:

Input Column	Output Alias	Sort Type	Sort Order	Con
merchant_name	merchant_name	ascending	1	

Bước 21: Ở Sort 3, chọn các cột tương ứng với DimMerchant (không chọn merchant\_id)

The screenshot shows the 'Available Input Columns' dialog box with the following columns:

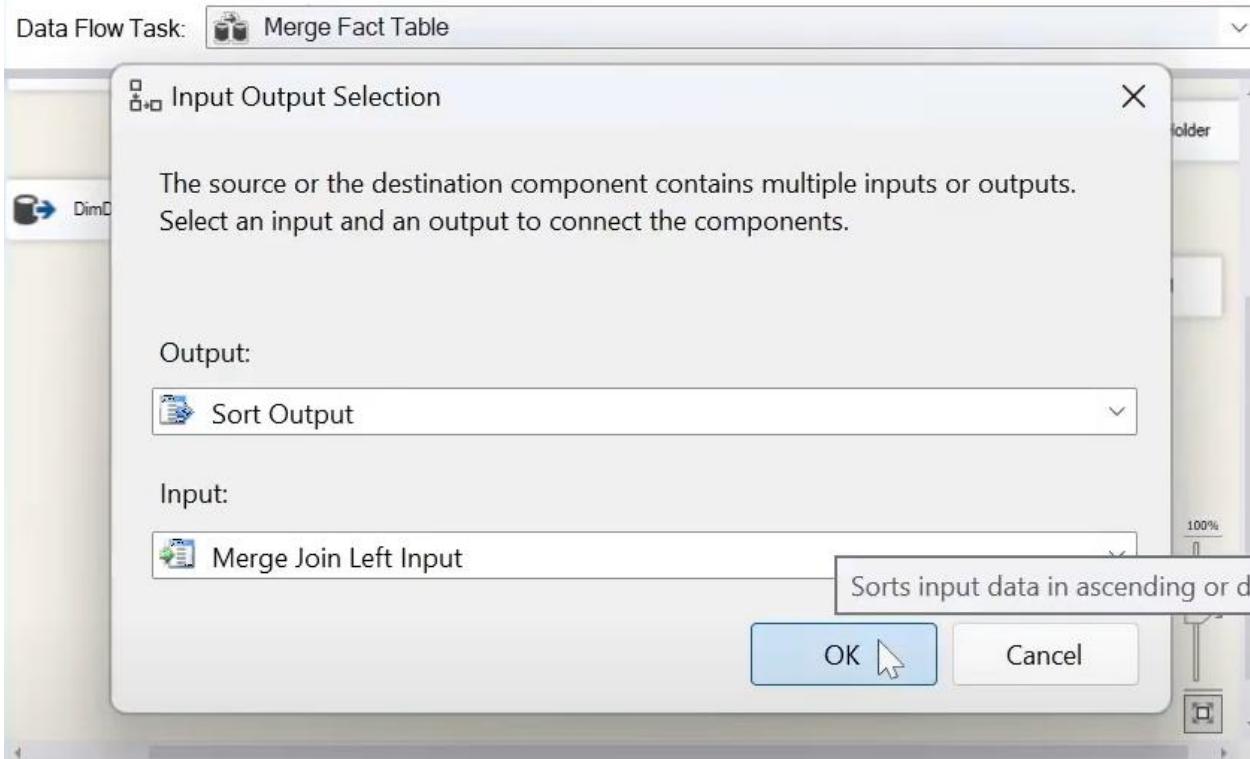
Name	Pass T...
merchant_id	<input checked="" type="checkbox"/>
merchant_name	<input checked="" type="checkbox"/>

Below the dialog is a table mapping:

Input Column	Output Alias	Sort Type	Sort Order	Con
merchant_name	merchant_name	ascending	1	

Bước 22: Tạo mới Merge Join 2 và nối với Merge Join 1

Tiếp theo ta chọn Merge Join 2 rồi chọn Left Input để giữ lại toàn bộ các dòng trong bảng StagingFact bất kể có kết quả khi thực hiện phép kết trái với cột ID của bảng DimMerchant hay không.

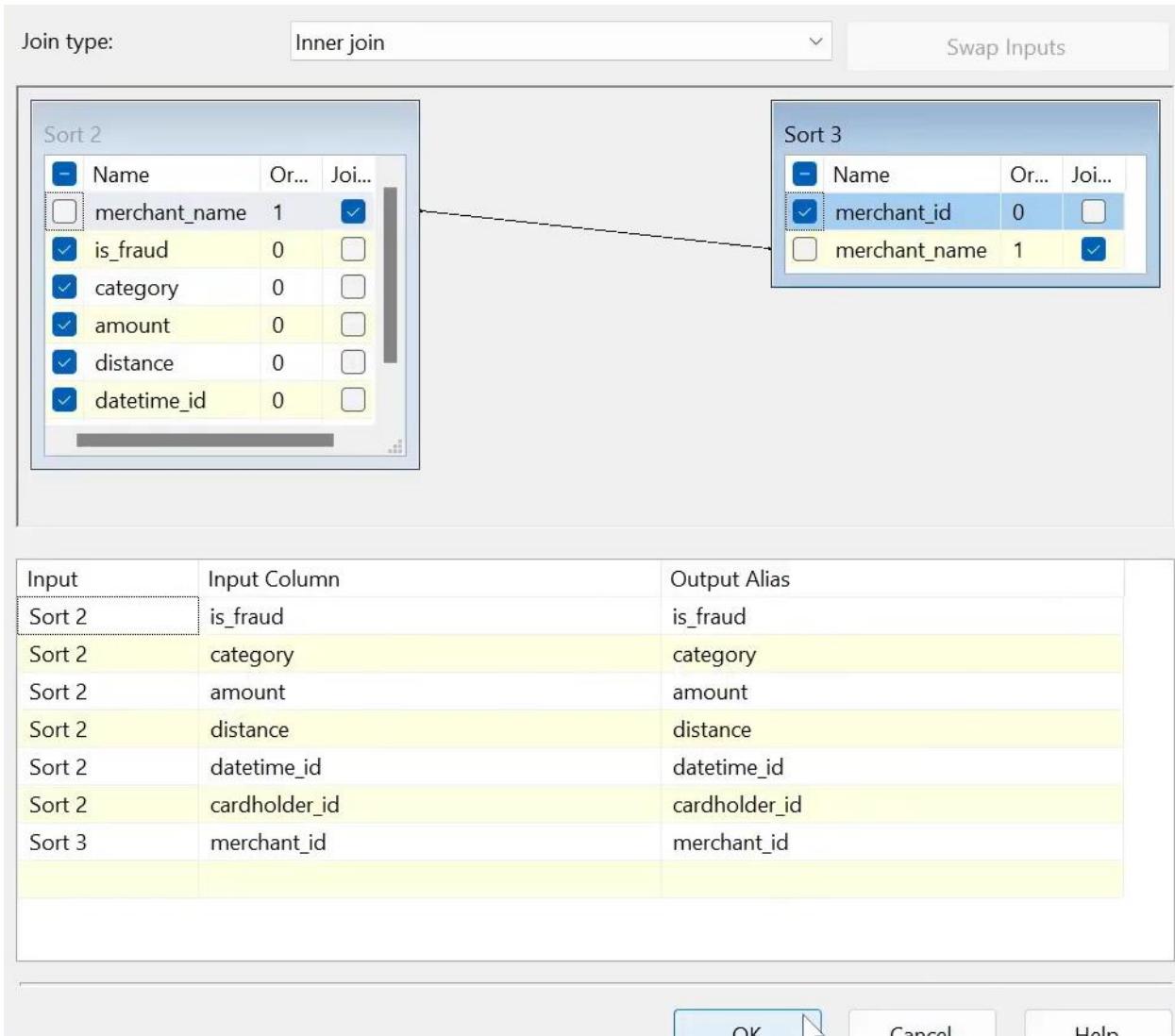


Bước 23: Chọn vào Merge Join 2, một hộp thoại merge editor xuất hiện:

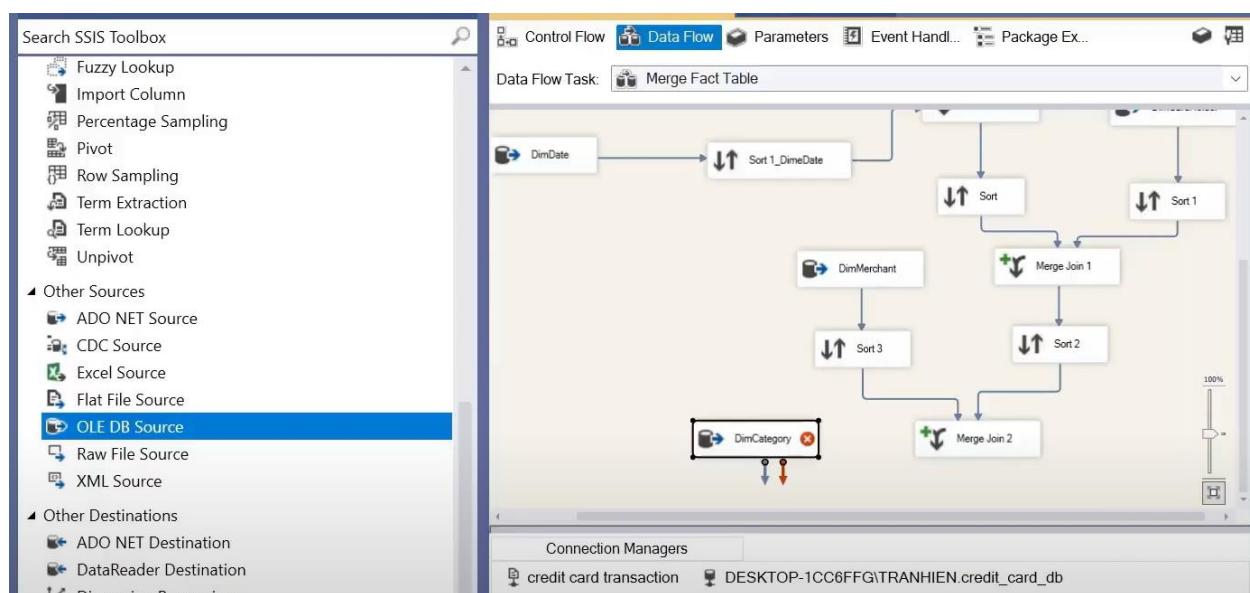
Chọn tất cả các cột của Sort 2 nhưng không lấy các cột tương ứng với bảng DimMerchant.

Tiếp theo ta chọn merchant\_id ở Sort 3 để merge vào StaggingFact

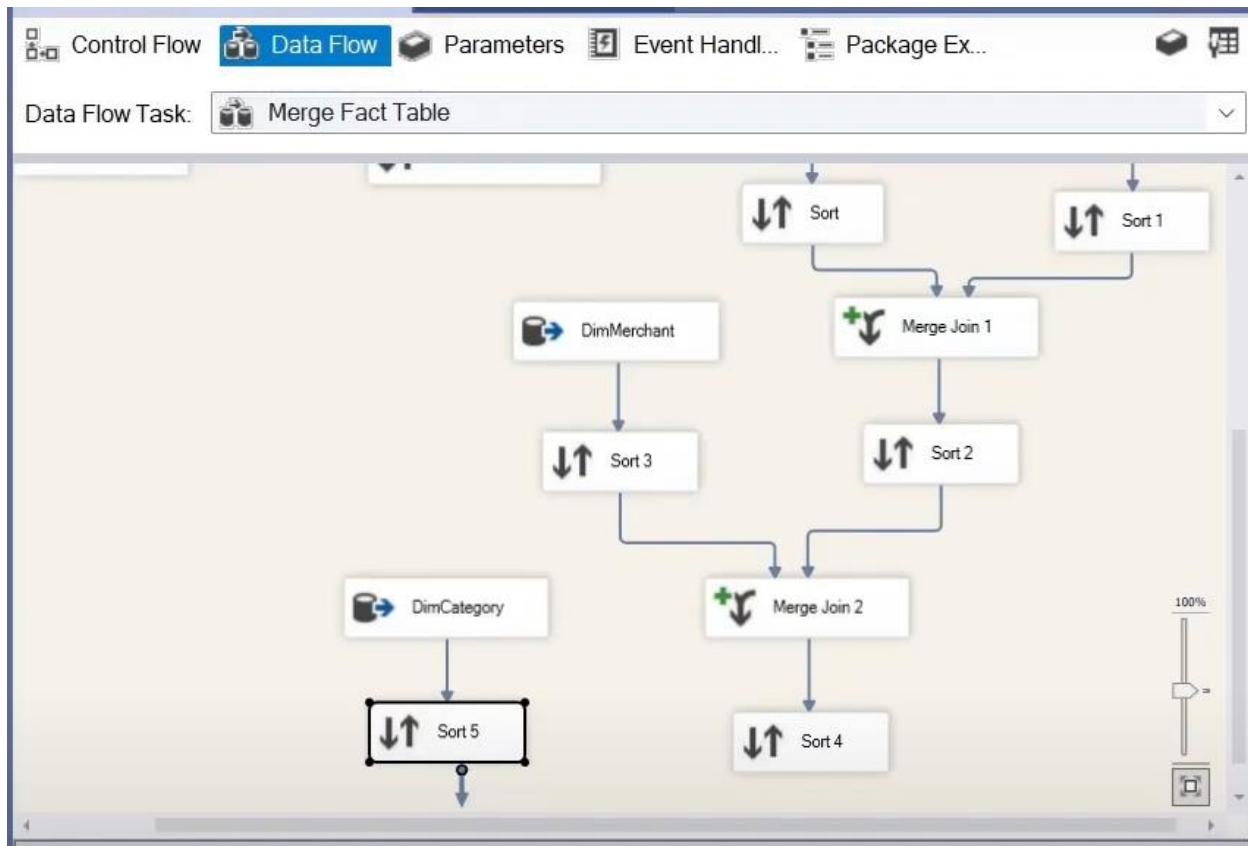
Kết quả sau khi merge là bảng StaggingFact không còn các thuộc tính của bảng DimMerchant và có thêm 1 thuộc tính mới là merchant\_id.



Bước 24: Tạo OLE DB Source mới và đổi thành DimCategory, sau đó chọn bảng dim\_category đã tạo trước đó làm data source cho bảng mới này.



Bước 25: Tạo 2 Sort tương ứng với mỗi bảng và kết nối đến bảng tương ứng để chuẩn bị cho quá trình merge



Bước 26: Ở Sort 4, chọn các cột theo thứ tự giống với bảng DimCategory để chuẩn bị cho quá trình

Available Input Columns	
Name	Pass T...
is_fraud	<input checked="" type="checkbox"/>
<b>category</b>	<input checked="" type="checkbox"/>
amount	<input checked="" type="checkbox"/>
distance	<input checked="" type="checkbox"/>
datetime_id	<input checked="" type="checkbox"/>
cardholder_id	<input checked="" type="checkbox"/>

Input Column	Output Alias	Sort Type	Sort Order	Com
category	category	ascending	1	

Bước 27: Ở Sort 5, chọn các cột tương ứng với DimCategory (không chọn category\_id)

The screenshot shows the SSIS Data Flow Task configuration window. At the top, there is a 'Available Input Columns' dialog box with three columns: Name, Pass T..., and category\_id. The 'category\_id' column has a checked checkbox. Below this, there is a table titled 'Sort Transformation' with columns: Input Column, Output Alias, Sort Type, Sort Order, and Com. A single row is present: 'category' under Input Column, 'category' under Output Alias, 'ascending' under Sort Type, and '1' under Sort Order.

Bước 28: Tạo mới Merge Join 3 và nối với Sort 4, tiếp theo ta chọn Merge Join 3 rồi chọn Left Input để giữ lại toàn bộ các dòng trong bảng StagingFact bất kể có kết quả khi thực hiện phép kết trái với cột ID của bảng DimCategory hay không.

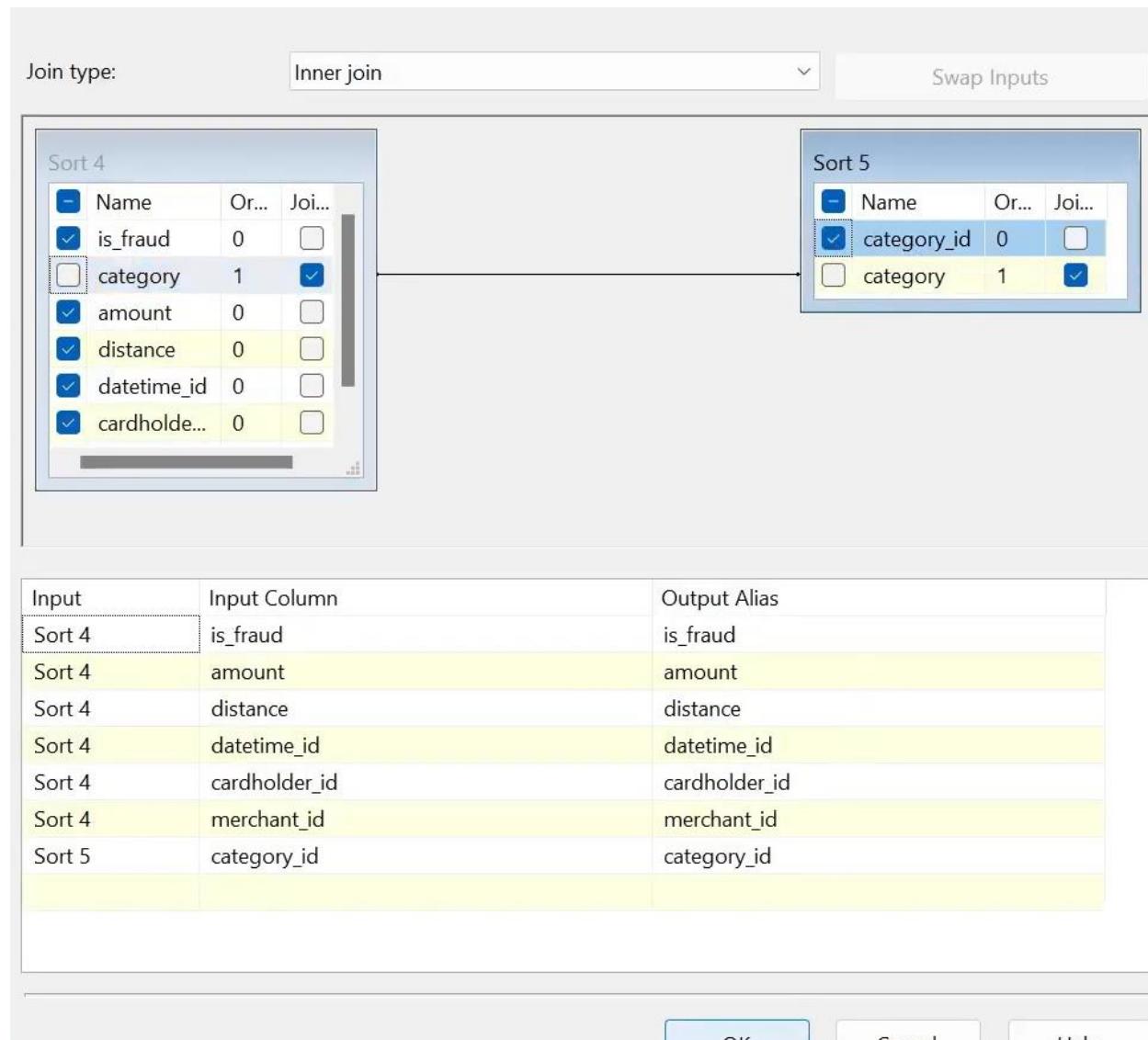
The screenshot shows the SSIS Data Flow Task configuration window with a 'Merge Fact Table' selected. A 'Input Output Selection' dialog box is open. It contains a message: 'The source or the destination component contains multiple inputs or outputs. Select an input and an output to connect the components.' Below this, there are two sections: 'Output:' with a dropdown menu showing 'Sort Output' and 'Input:' with a dropdown menu showing 'Merge Join Left Input'. At the bottom of the dialog is a note: 'Sorts input data in ascending or descending order' and two buttons: 'OK' and 'Cancel'.

Bước 29: Chọn vào Merge Join 3, một hộp thoại merge editor xuất hiện:

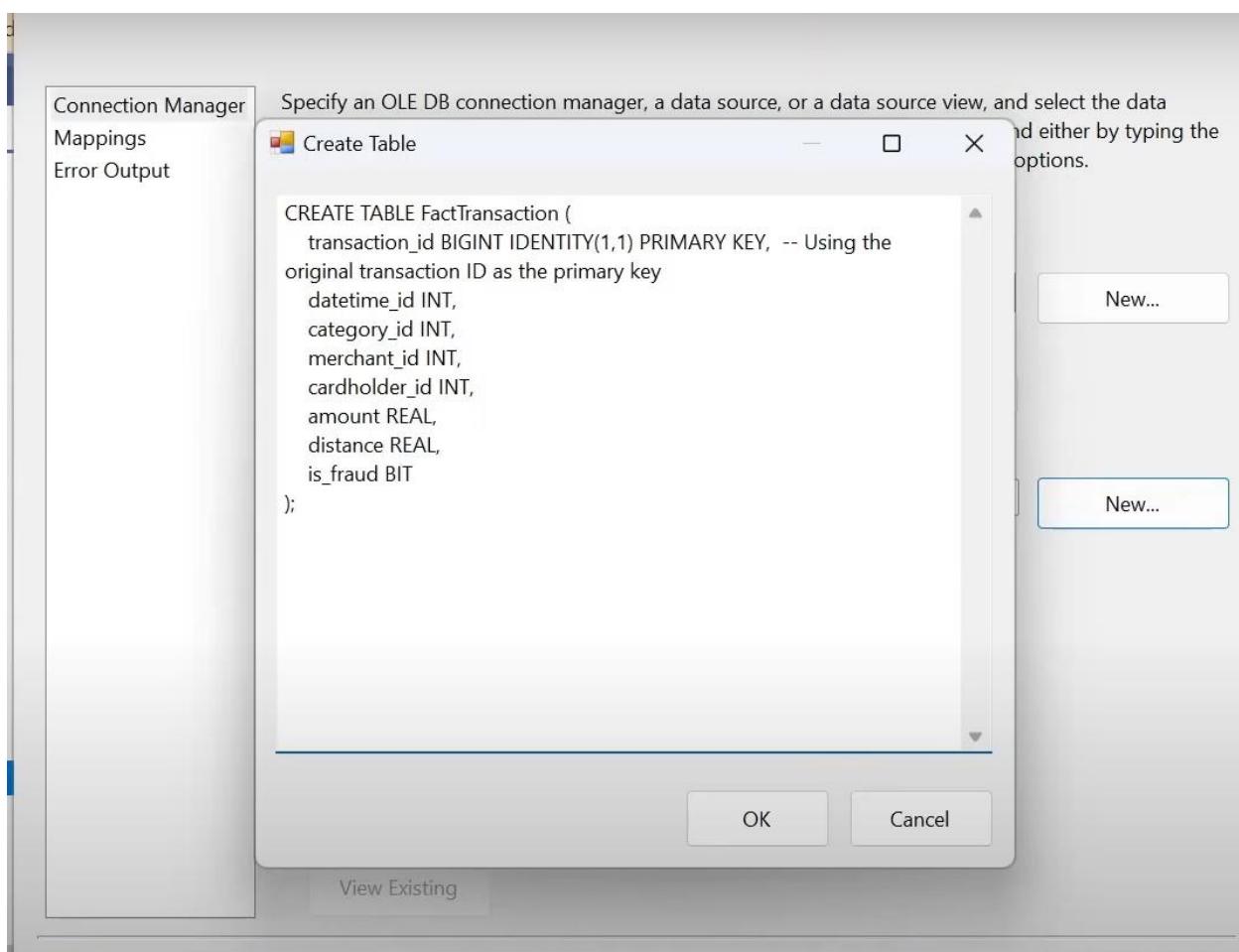
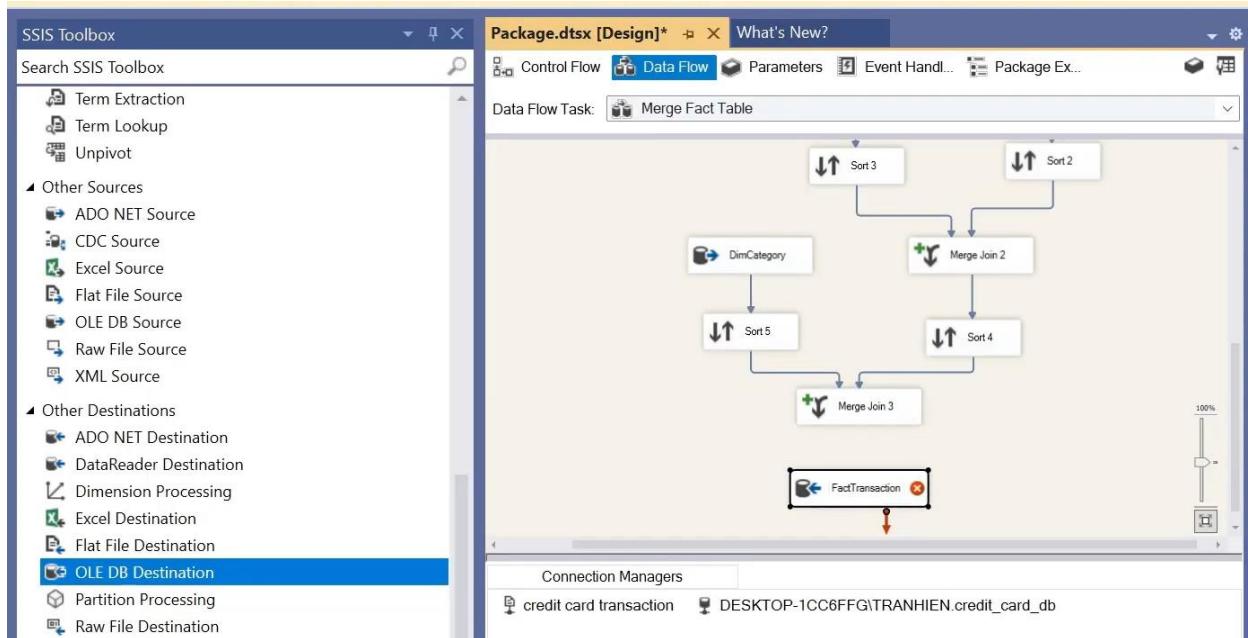
Chọn tất cả các cột của Sort 4 nhưng không lấy các cột tương ứng với bảng DimCategory

Tiếp theo ta chọn category\_id ở Sort 5 để merge vào StagingFact

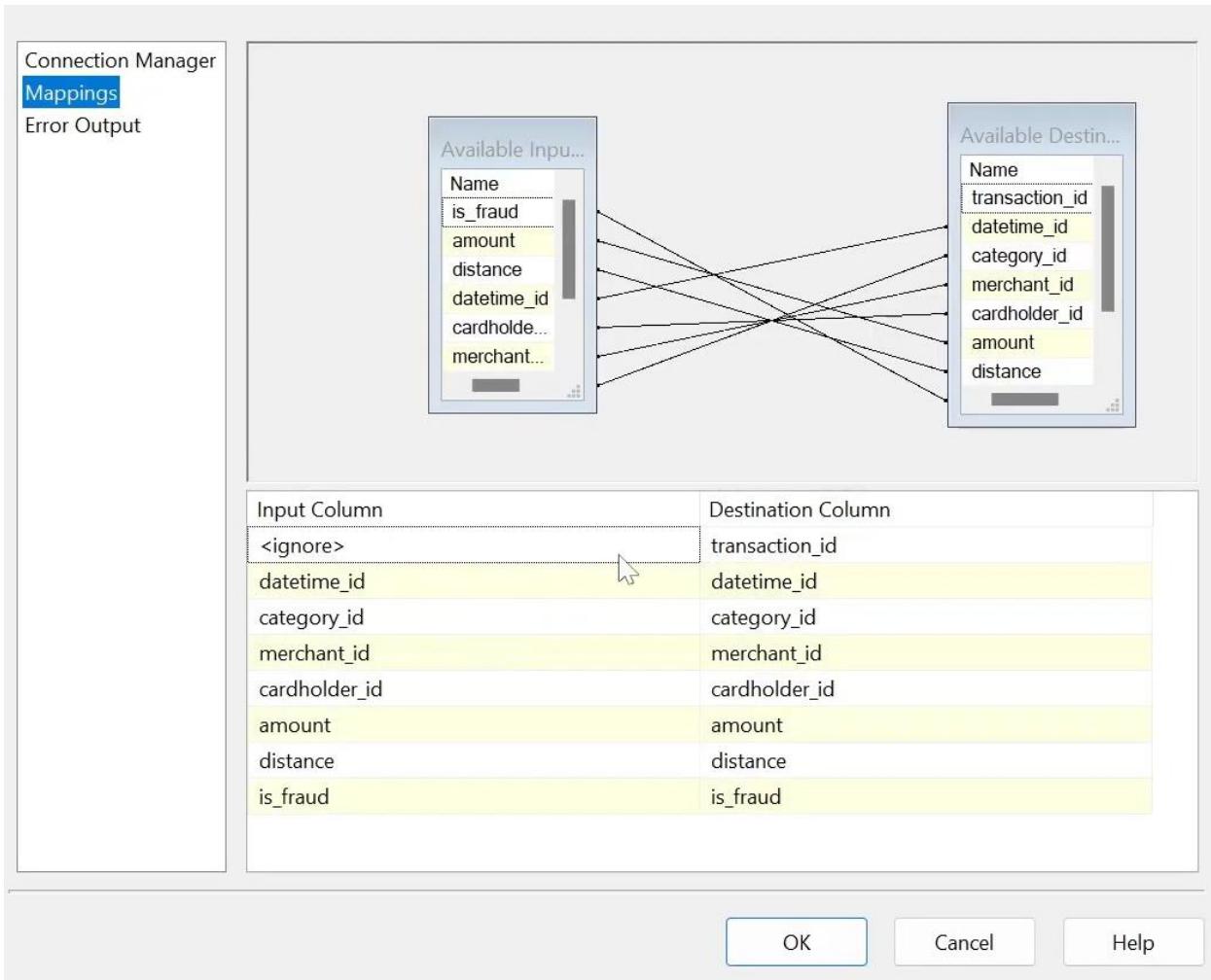
Kết quả sau khi merge là bảng StagingFact không còn các thuộc tính của bảng DimCategory và có thêm 1 thuộc tính mới là category\_id.



Bước 30: Tạo mới một OLE DB Destination, đổi tên thành FactTransaction để chứa tất cả những gì đã merge

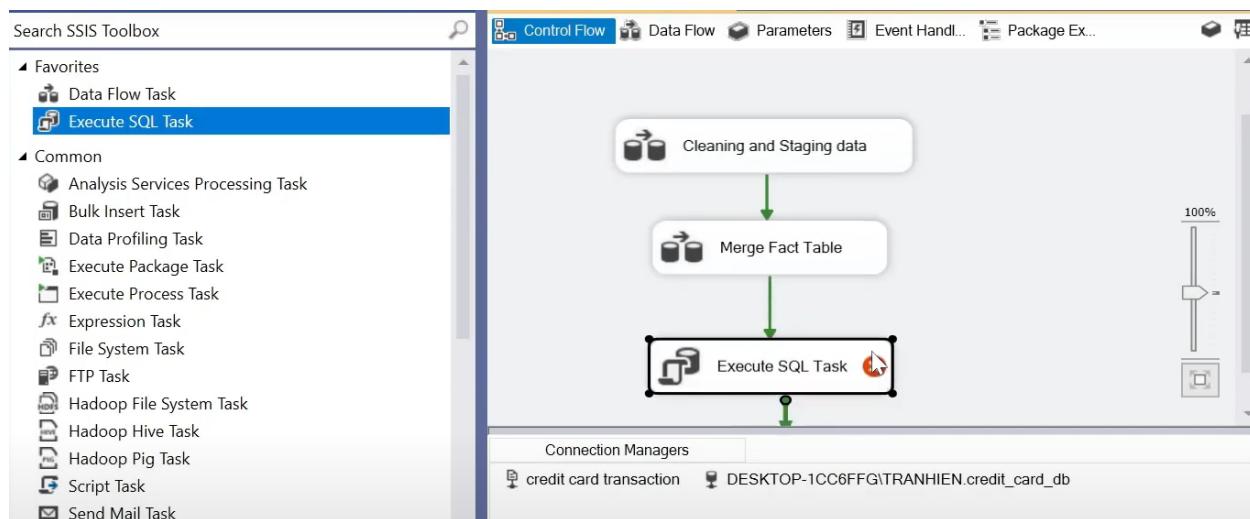


Bước 31: Chọn Mapping để xem ánh xạ dữ liệu đã đúng hay chưa. Nhấn OK để hoàn tất

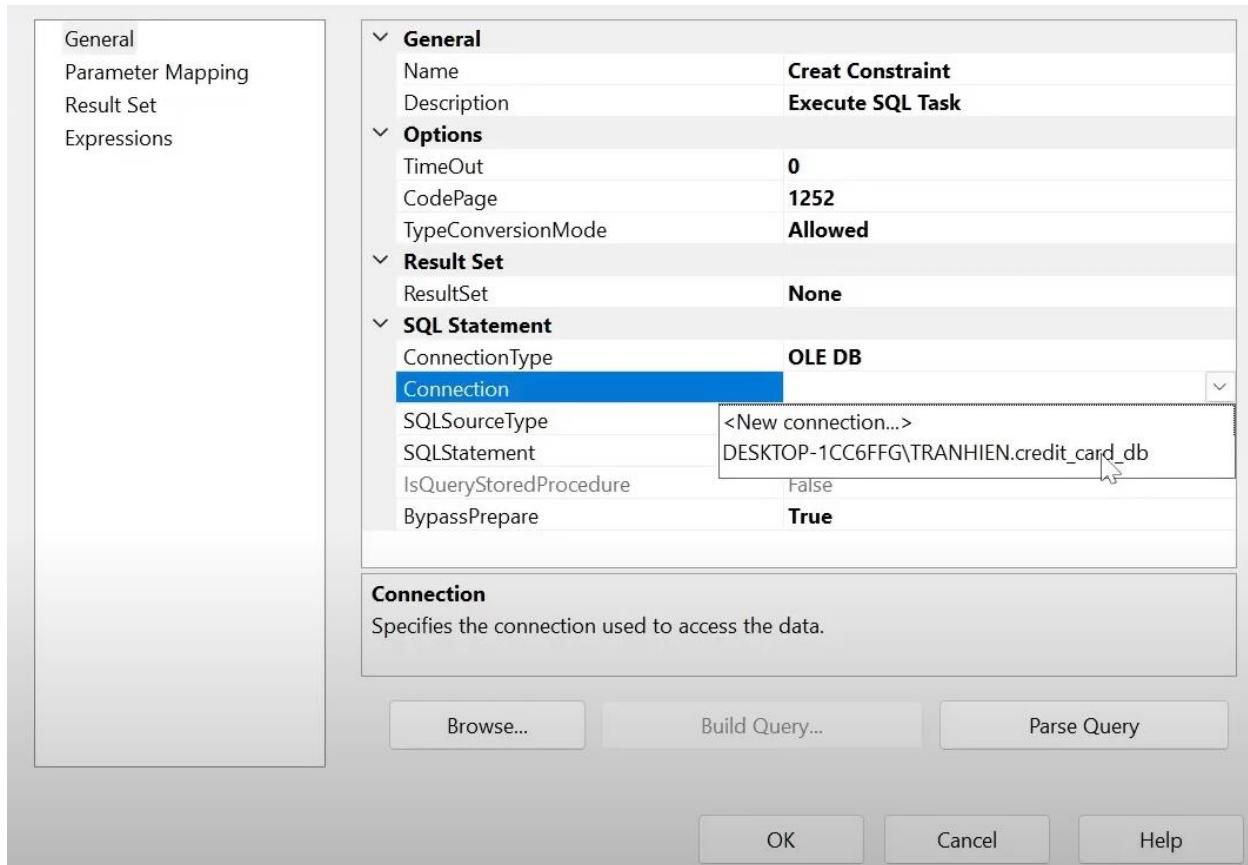


## 2.7 Create Constraint

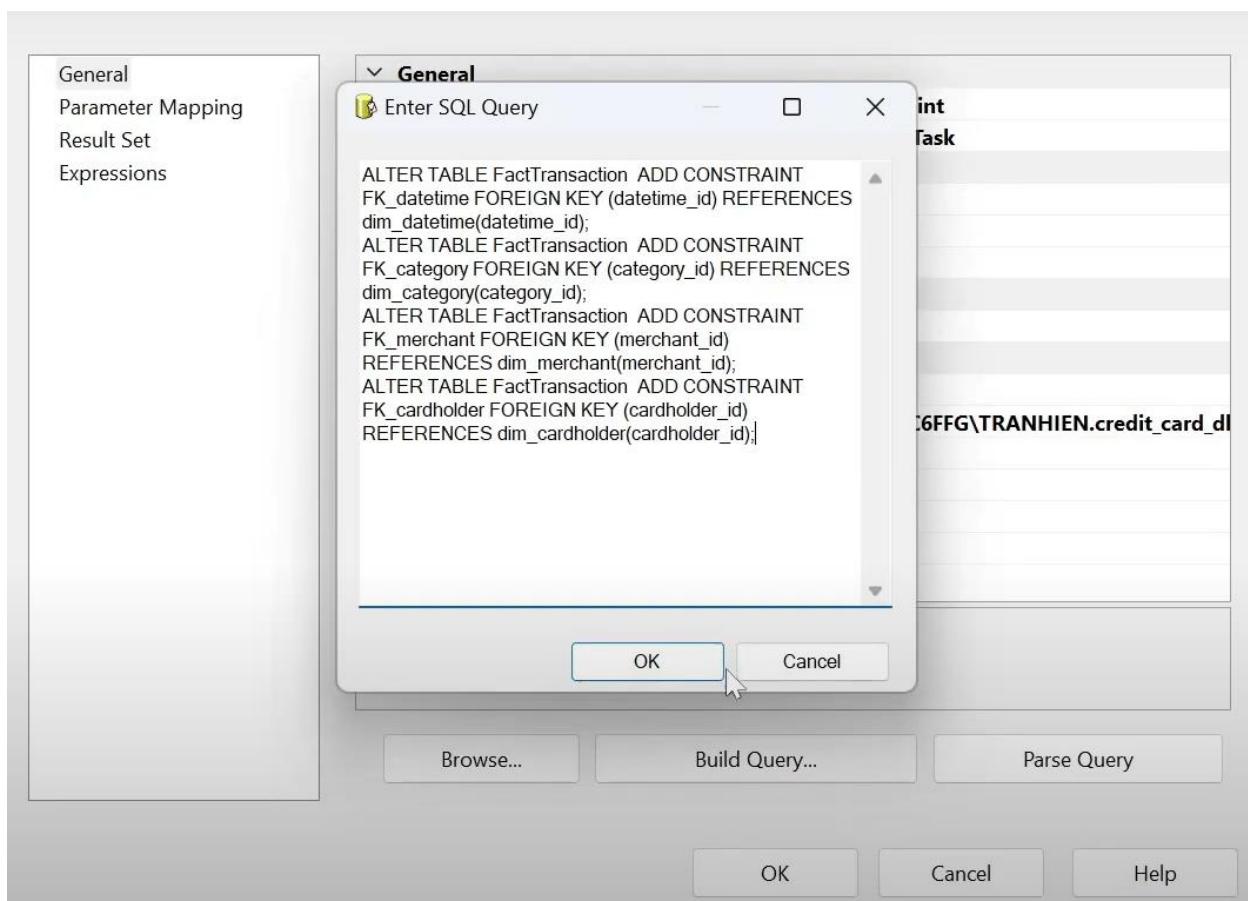
Bước 1: Chọn mới Data Flow Task, đổi tên thành Create Constraint. Ta sẽ tạo khóa ở bước này.



Bước 2: Chọn Execute SQL Task, ở ô Connection, chọn connection đã thiết lập đến data warehouse trong SQL Server.



Bước 3: Ở ô SQLStatement, thêm các câu truy vấn SQL thực hiện tạo các khóa ngoại từ các Dim đến bảng Fact. Nhấn OK để hoàn tất quá trình.

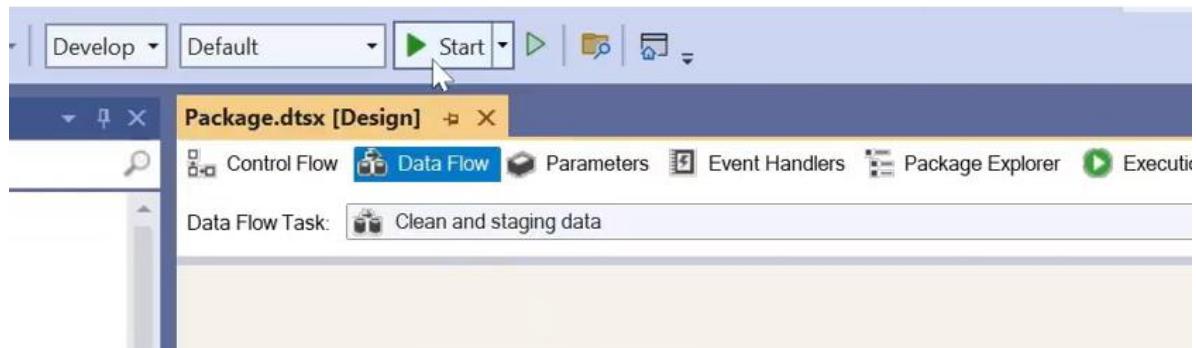


Code để tạo khóa:

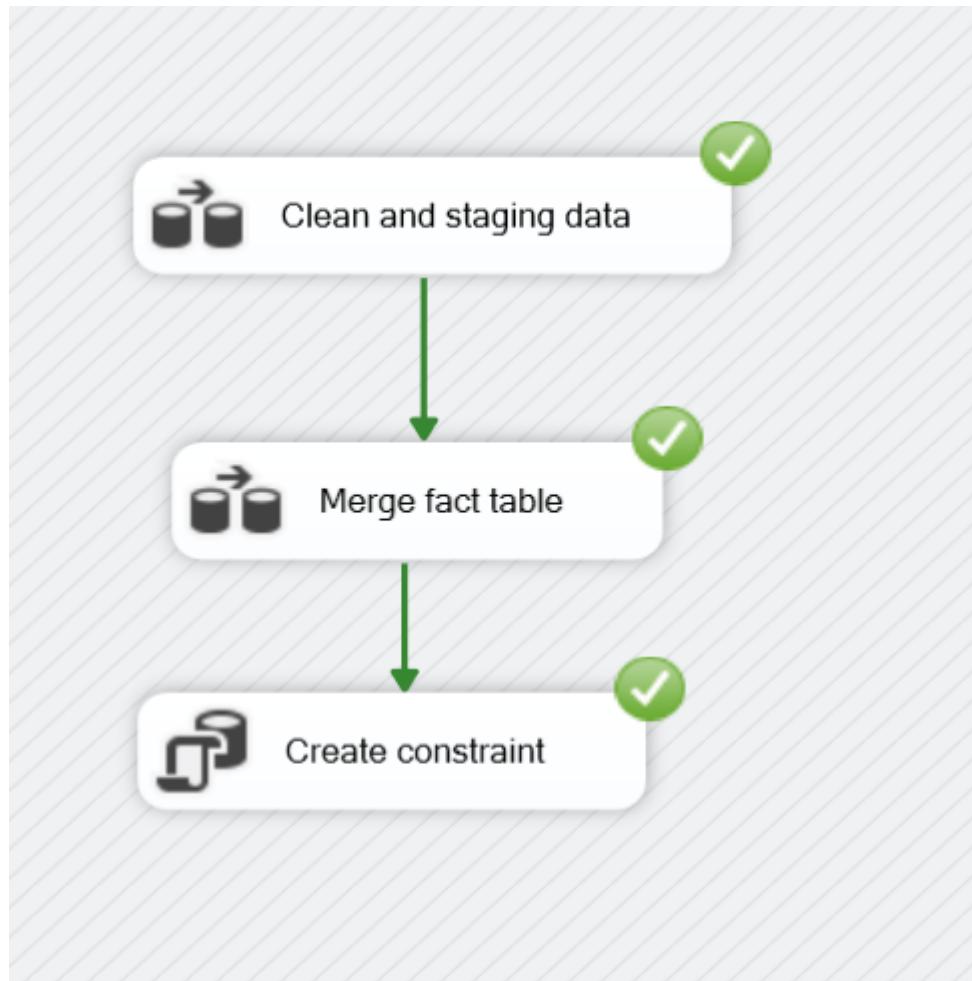
```
ALTER TABLE FactTrans ADD CONSTRAINT FK_datetime FOREIGN KEY (datetime_id)
REFERENCES dim_datetime(datetime_id);
ALTER TABLE FactTrans ADD CONSTRAINT FK_category FOREIGN KEY (category_id)
REFERENCES dim_category(category_id);
ALTER TABLE FactTrans ADD CONSTRAINT FK_merchant FOREIGN KEY (merchant_id)
REFERENCES dim_merchant(merchant_id);
ALTER TABLE FactTrans ADD CONSTRAINT FK_cardholder FOREIGN KEY (cardholder_id)
REFERENCES dim_cardholder(cardholder_id)
```

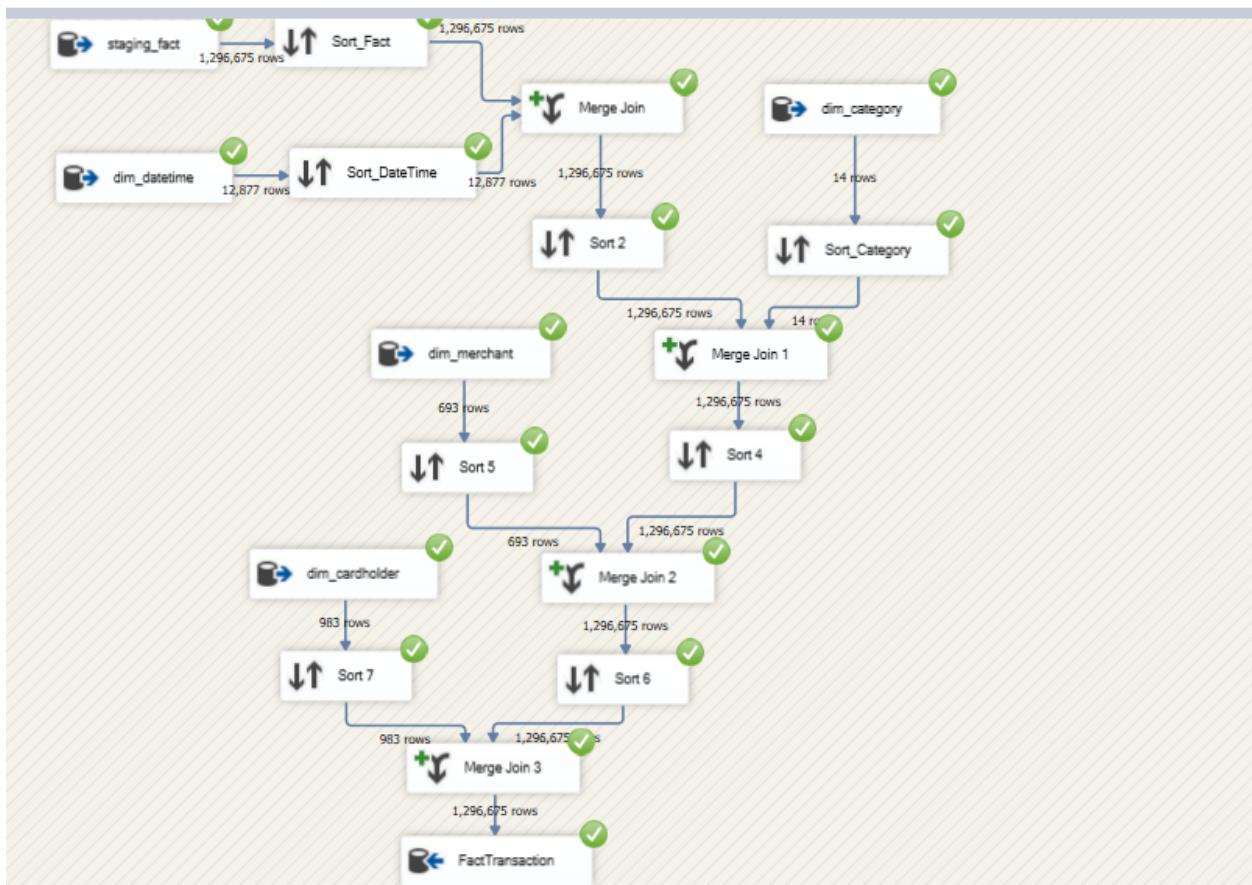
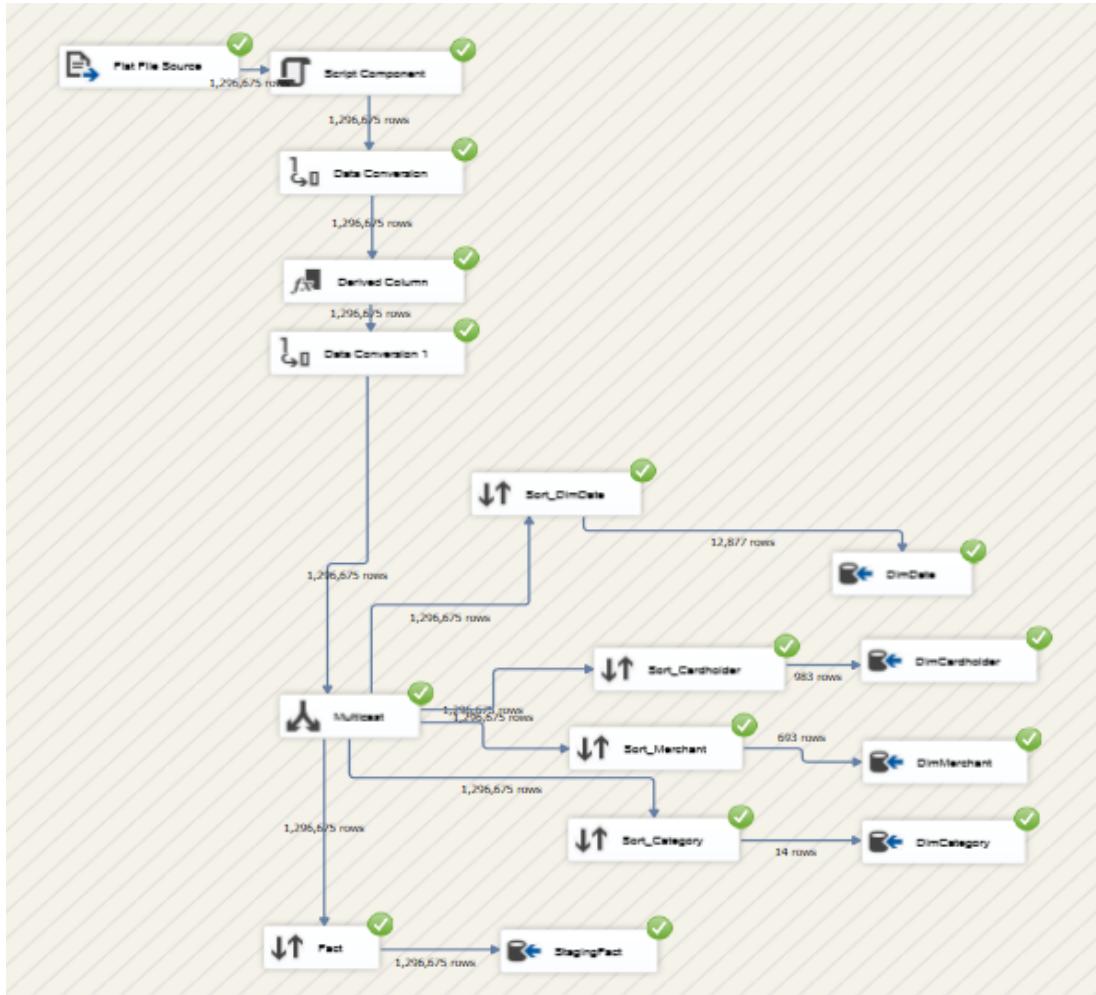
## 2.8 Chạy dự án SSIS

Nhấn nút Start trên thanh menu để tiến hành chạy project.



Kết quả chạy dự án:

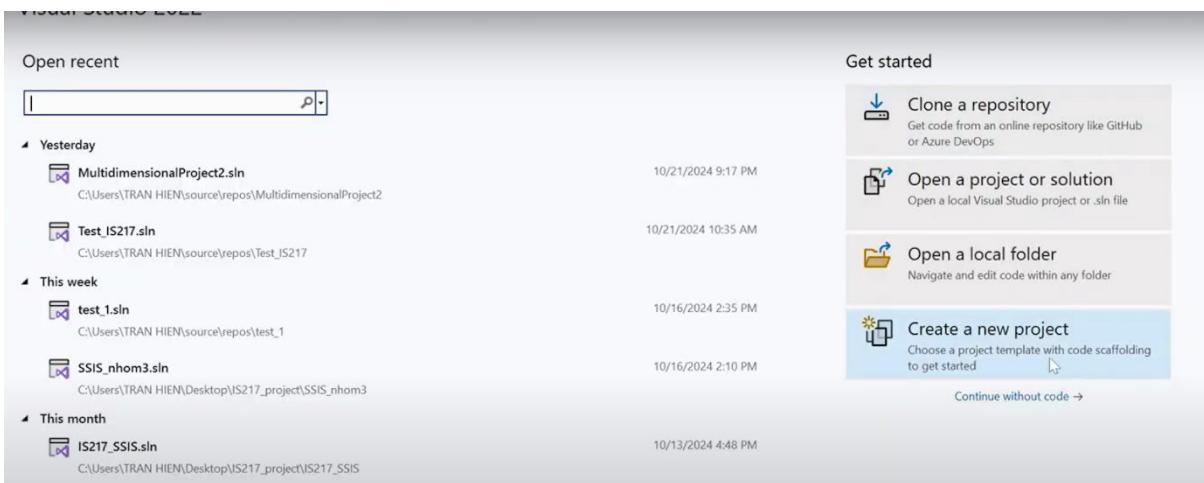




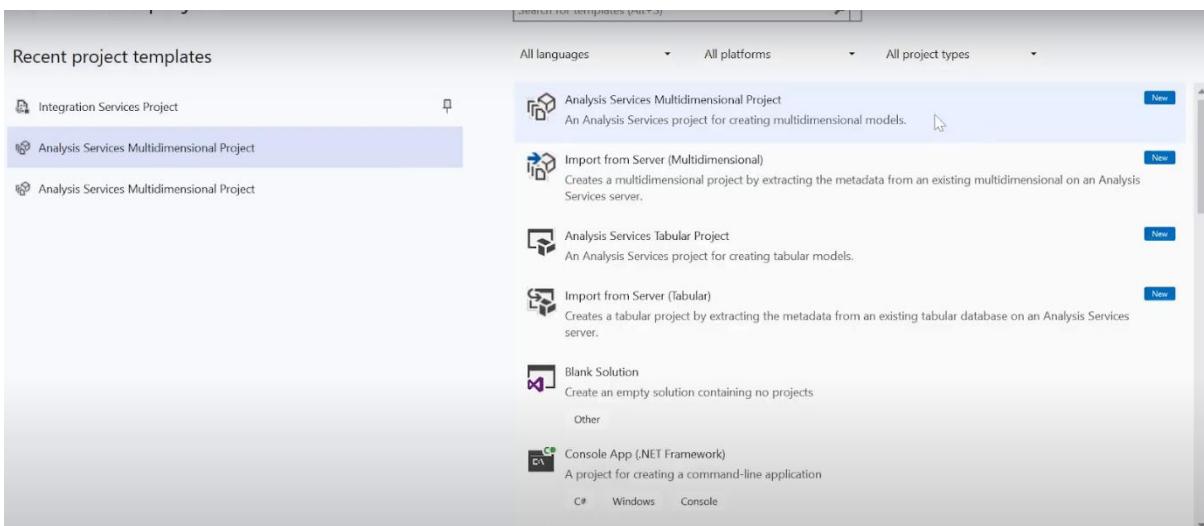
## PHẦN 3: THỰC HIỆN PHÂN TÍCH TRỰC TUYẾN TRÊN KHO DỮ LIỆU (SSAS)

### 3.1 Tạo project SSAS mới

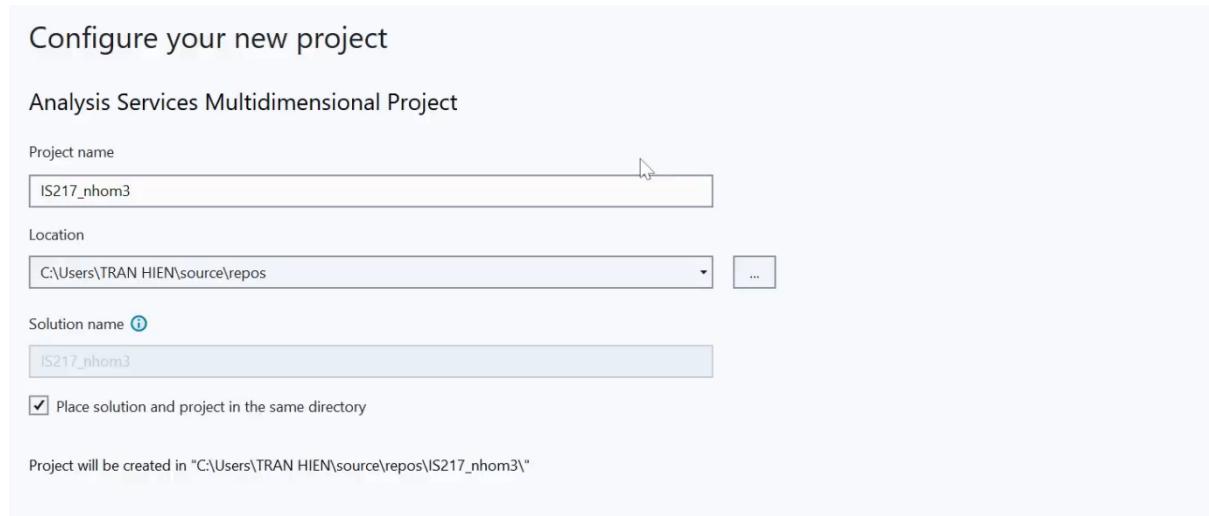
Bước 1: Mở Visual Studio và chọn “Create a new project”.



Bước 2: Chọn Analysis Services Multidimensional Project và chọn Next.

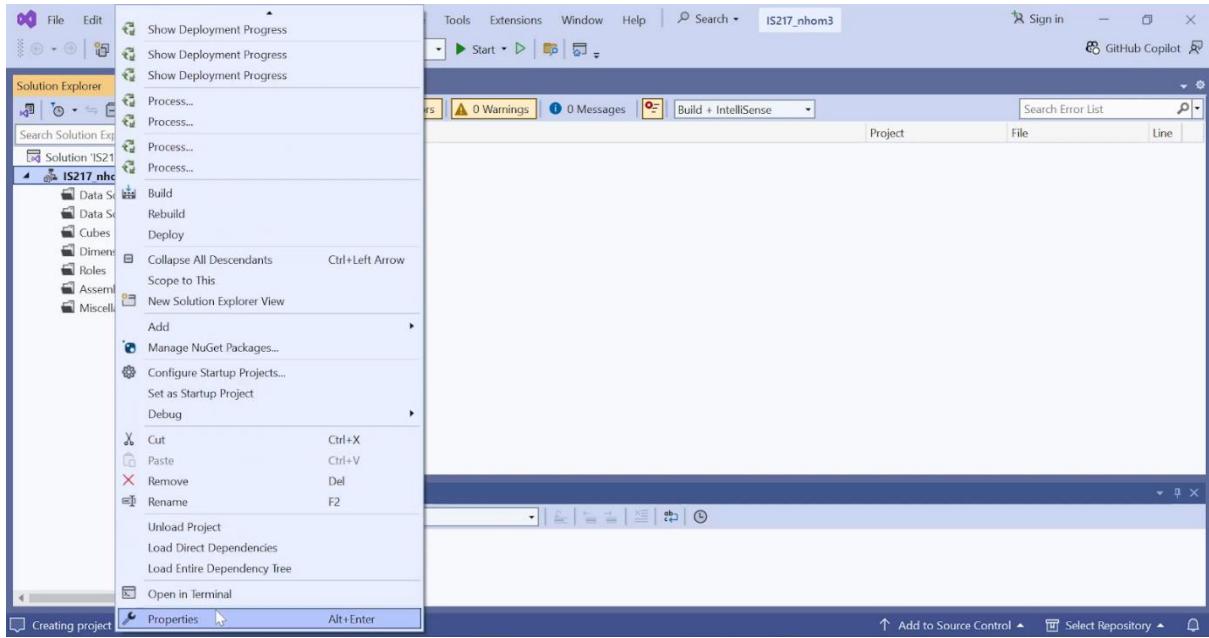


Bước 3: Đặt tên và thiết lập đường dẫn cho Project. Sau đó chọn Create.

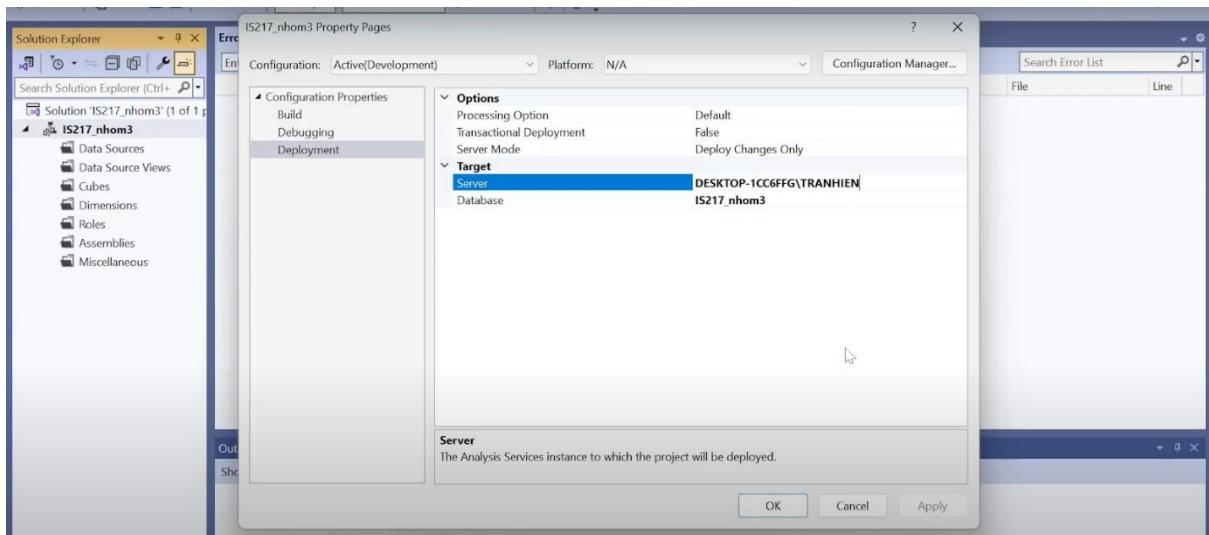


### 3.2 Xác định dữ liệu nguồn (Data Sources)

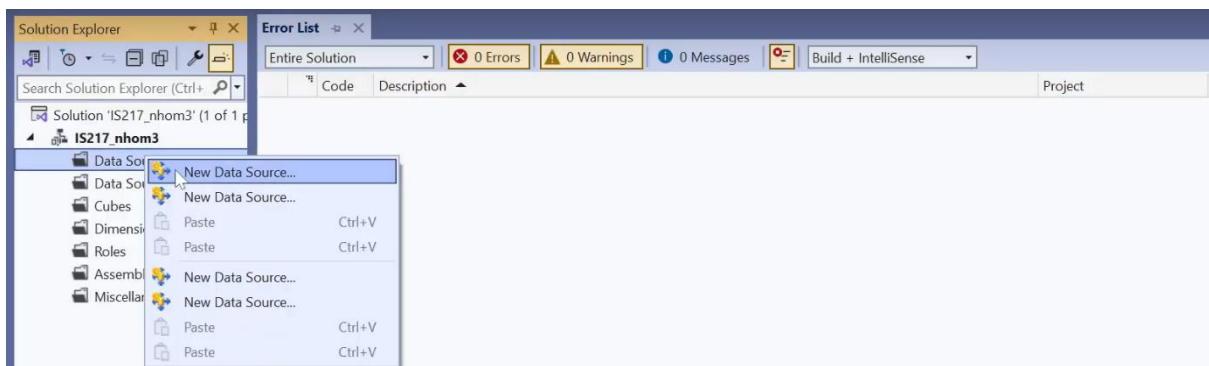
Bước 1: Chọn tên project rồi chọn Properties để chọn server



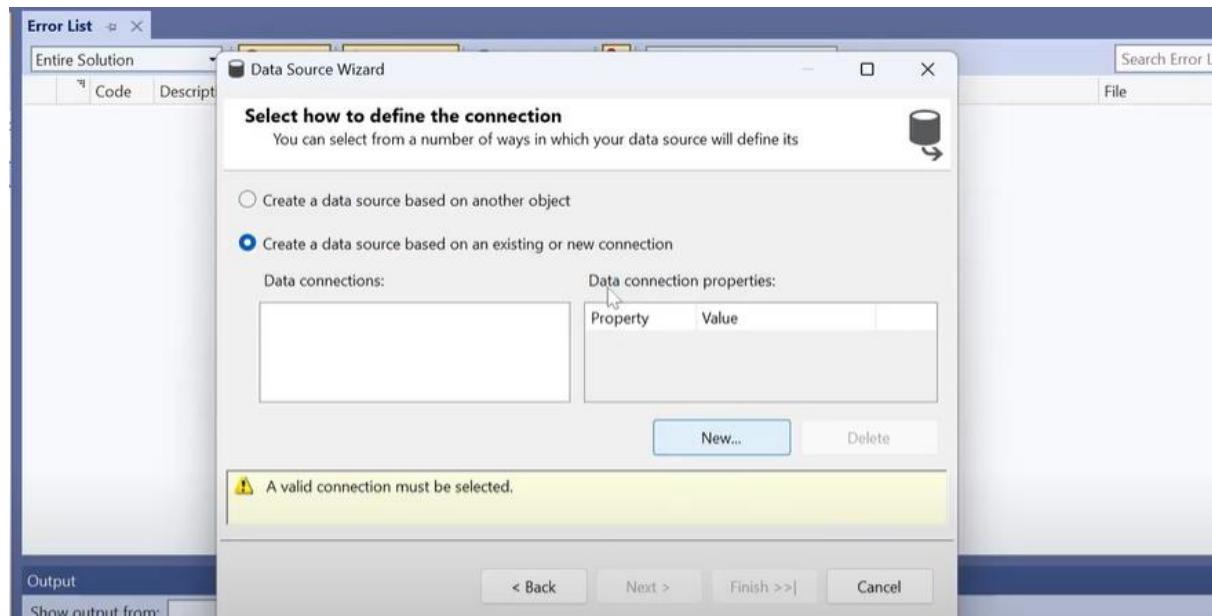
Trong cửa sổ Properties, chọn Deployment. Tại Server đổi từ localhost sang server name của máy



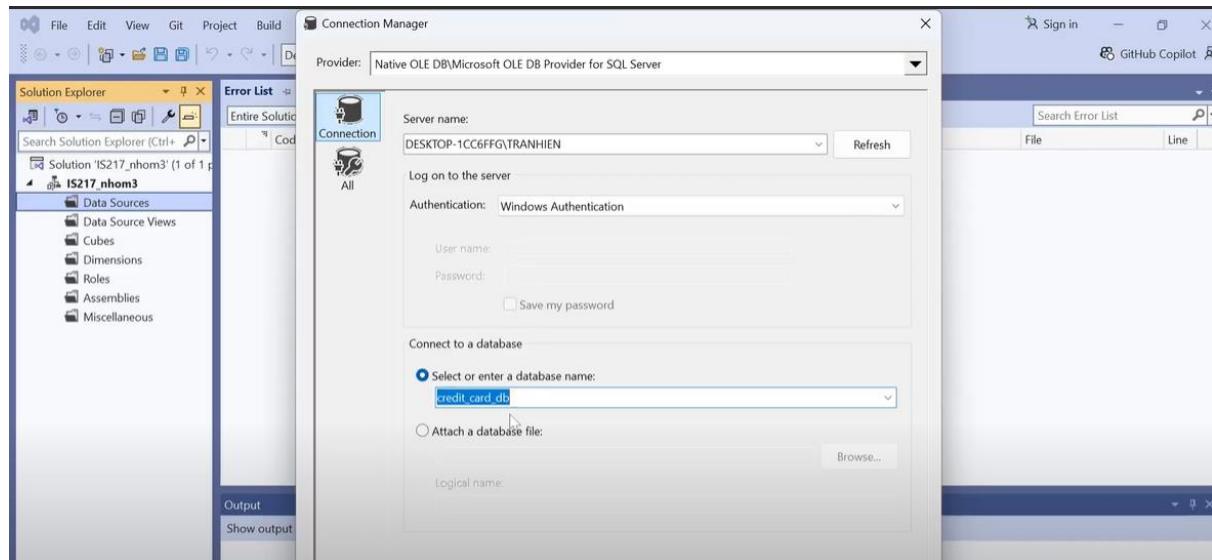
Bước 2: Tại Solution Explorer, chuột phải vào thư mục Data Sources, chọn New Data Source.



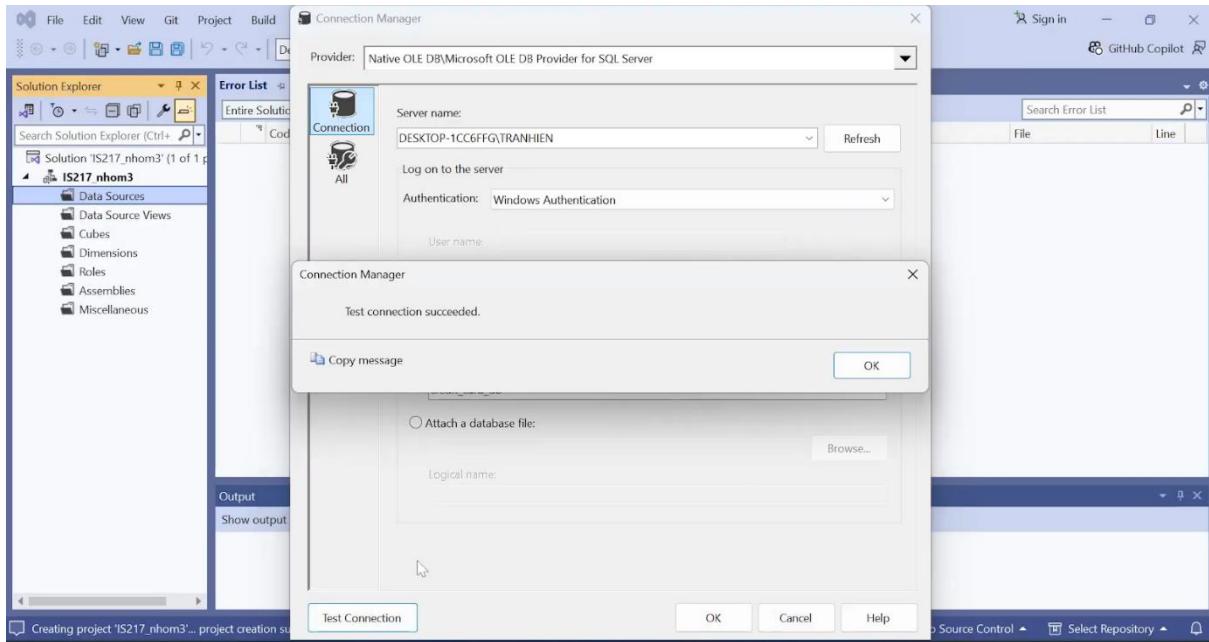
Bước 3: Chọn Create a data source based on an existing or new connection. Sau đó chọn New để tạo mới Data



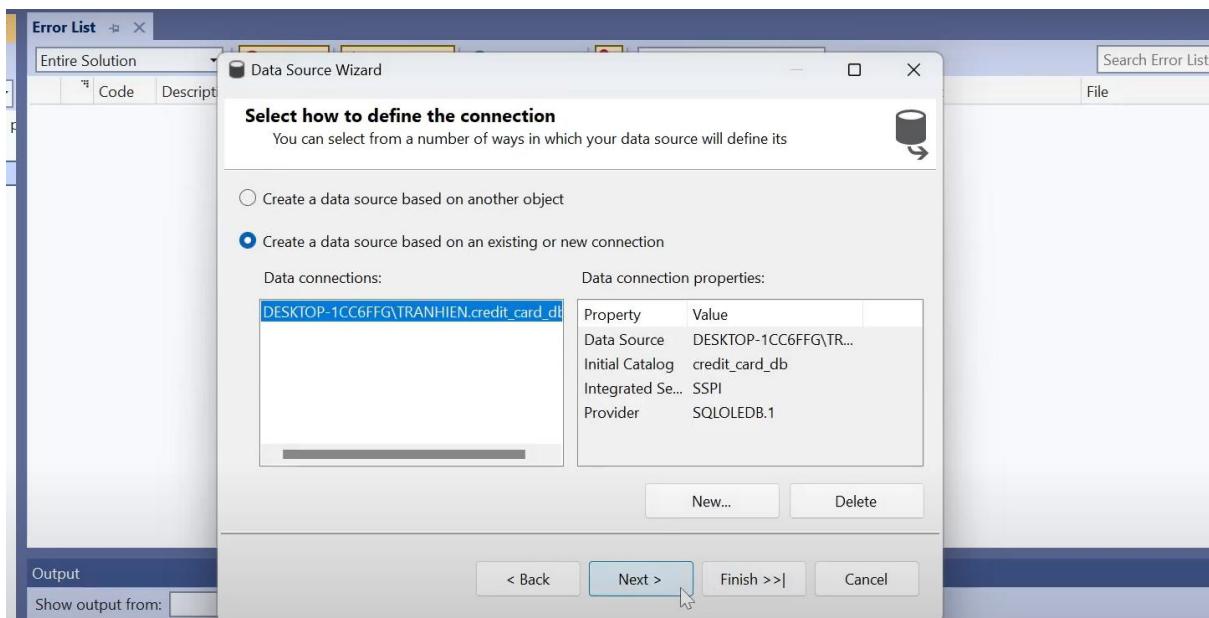
Bước 4: Tại Provider chọn Microsoft OLE DB Provider for SQL Server, tại Server name chọn Server của máy. Cuối cùng chọn Select or enter a database name và chọn Database đang sử dụng



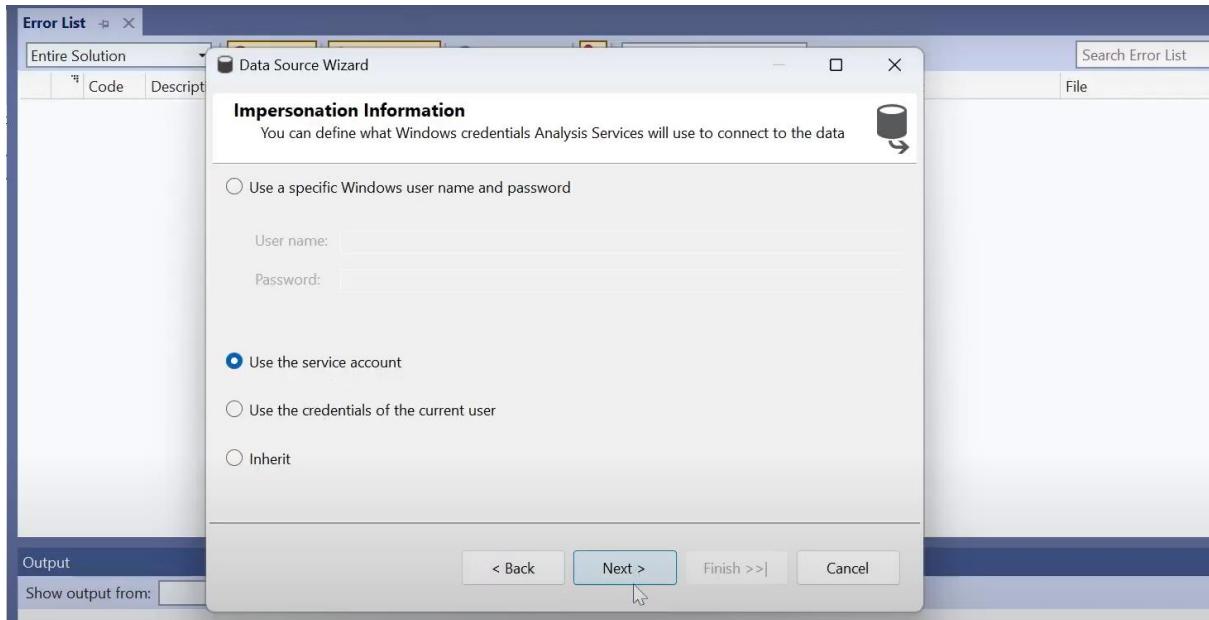
Bước 5: Chọn Test Connection, nếu màn hình hiện Test Connection Succeeded thì chọn OK.



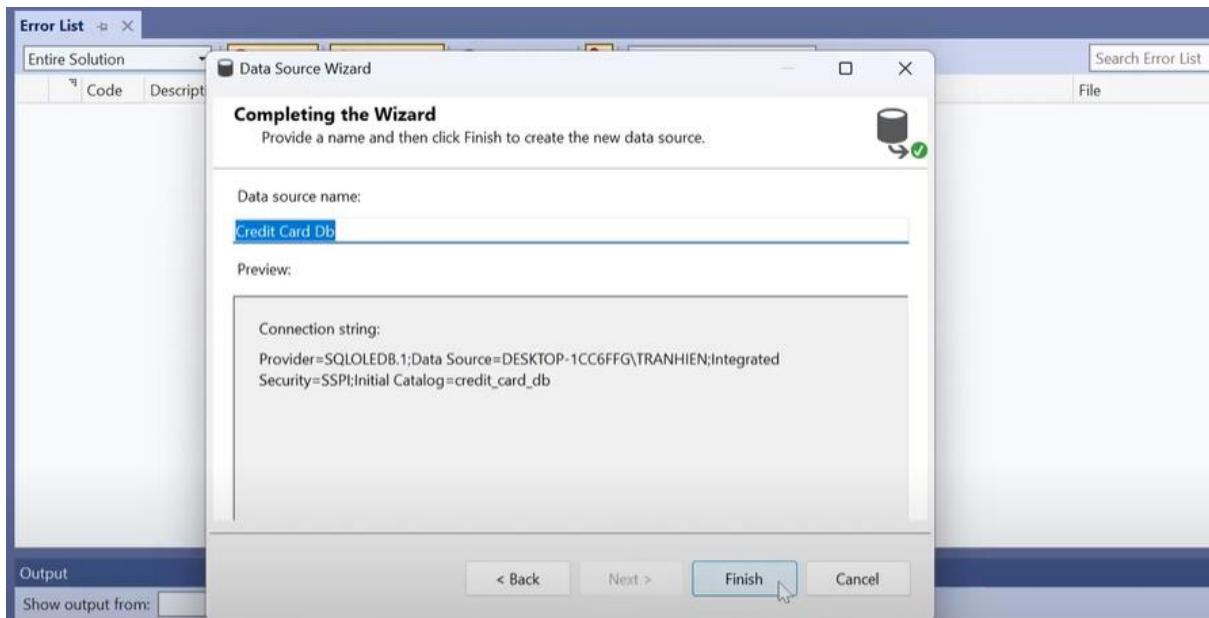
Bước 6: Chọn Next để tiếp tục



Bước 7: Chọn “Use the service account”, sau đó chọn Next để tiếp tục.

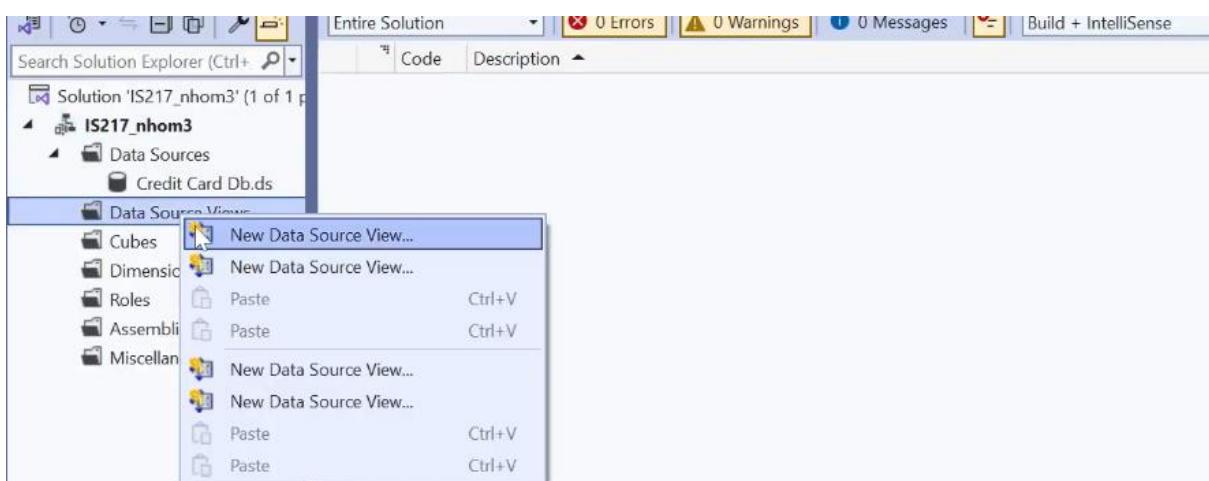


Bước 8: Chọn Finish để hoàn tất quy trình định nghĩa nguồn dữ liệu.

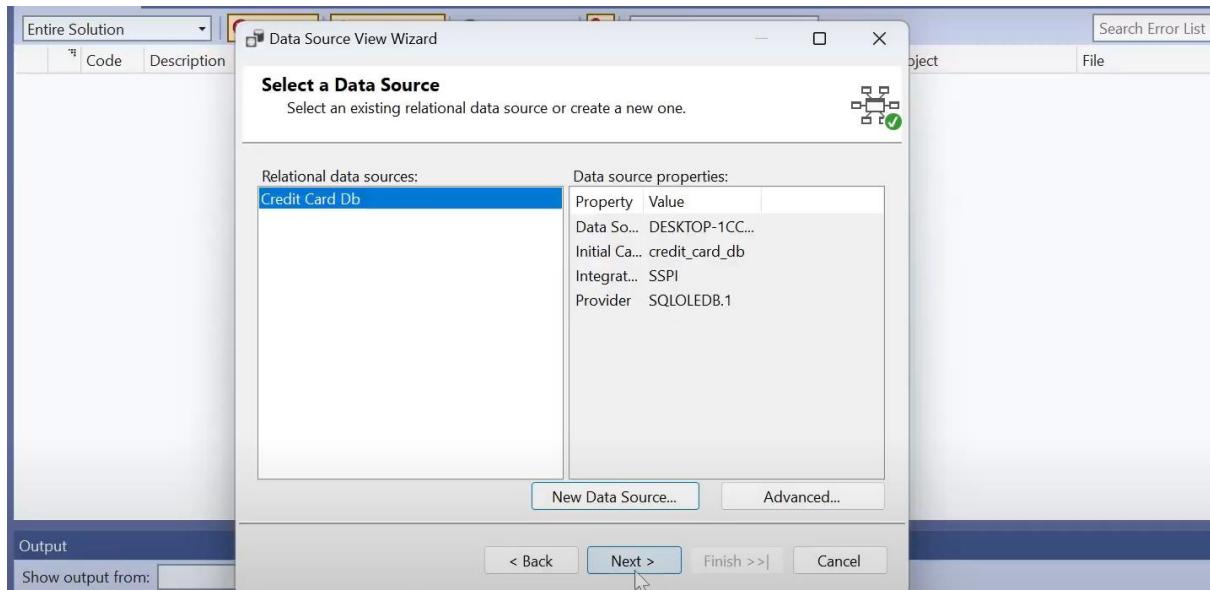


### 3.3 Xác định khung nhìn dữ liệu nguồn (Data Source View)

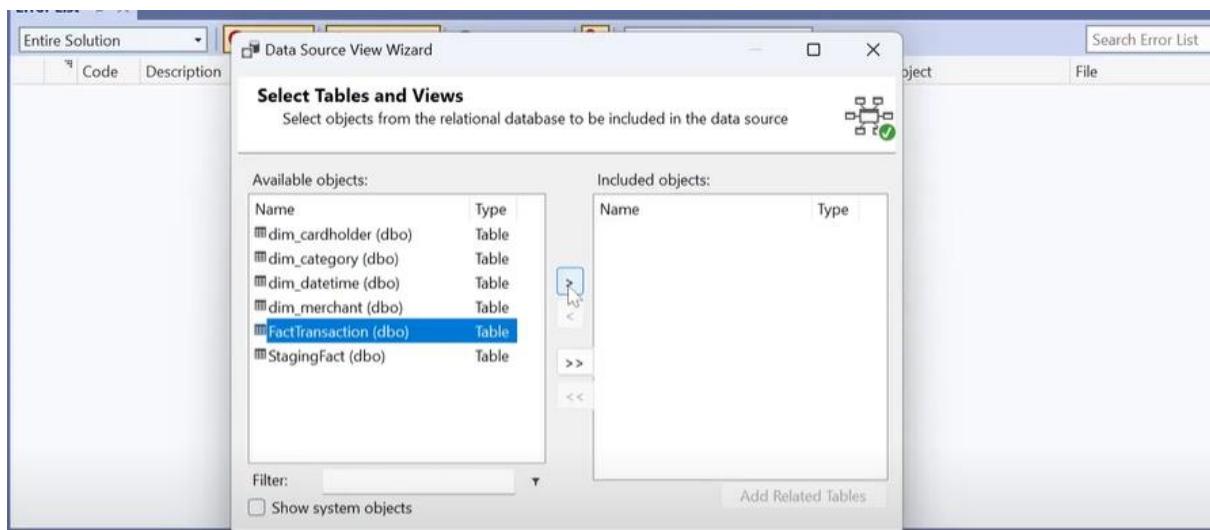
Bước 1: Tại Solution Explorer, chuột phải vào thư mục Data Source Views và chọn New Data Source View.



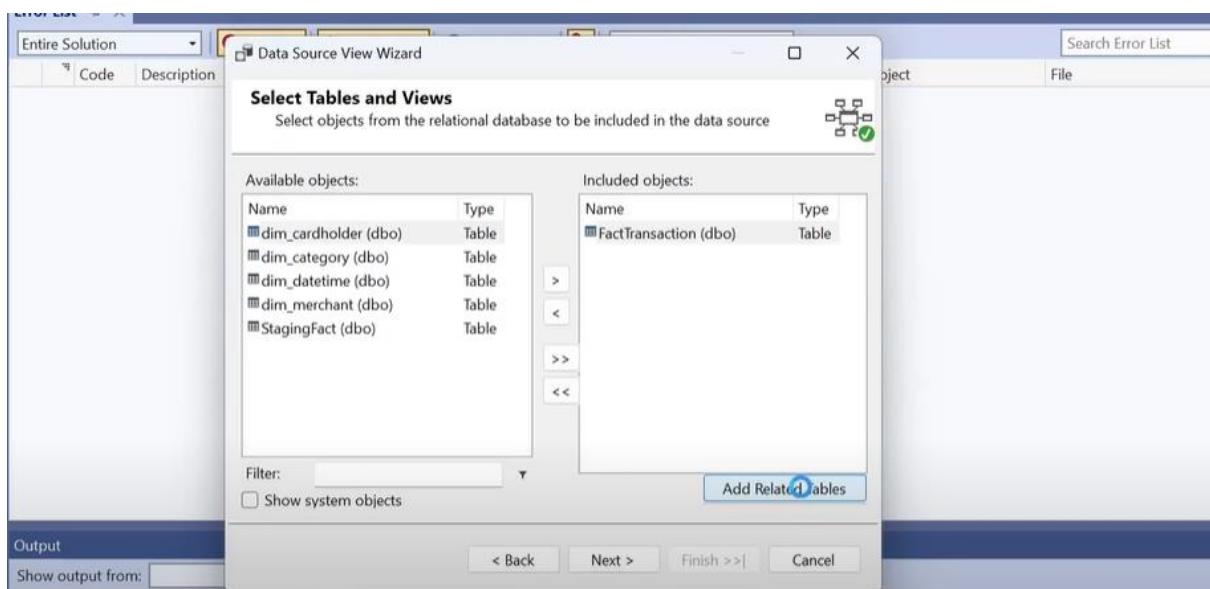
## Bước 2: Chọn Next để tiếp tục



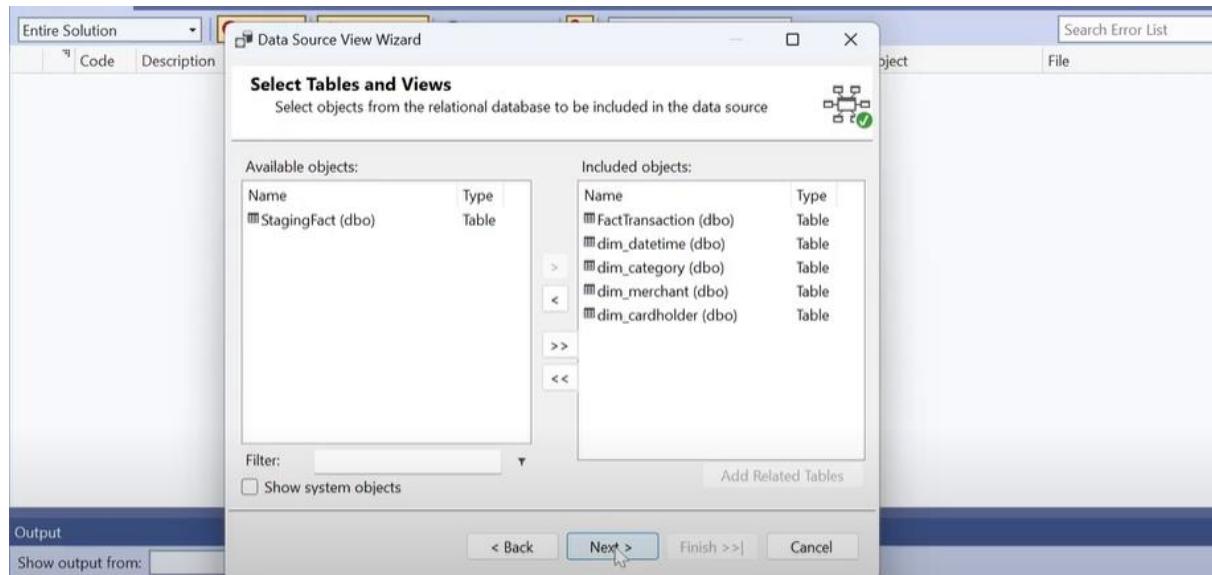
Bước 3: Chọn bảng Fact, sau đó chọn nút “>” để thêm bảng Fact vào Data source view.



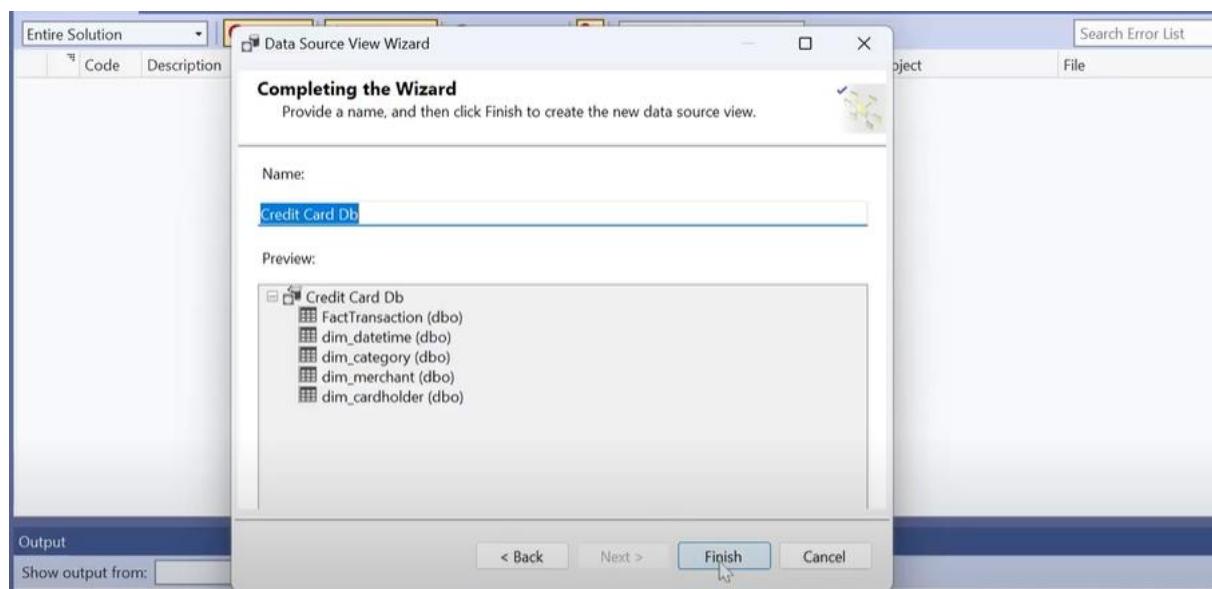
Bước 4: Chọn Add Related Tables, sau đó chọn >> để thêm tất cả bảng Dim vào Data source view



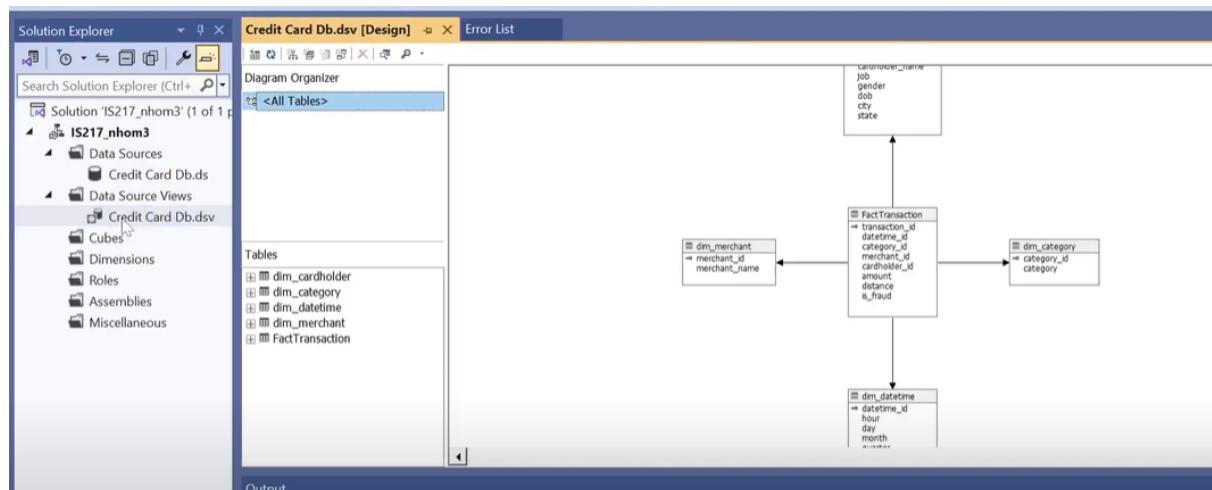
## Bước 5: Chọn Next để tiếp tục



## Bước 6: Chọn Finish để hoàn tất



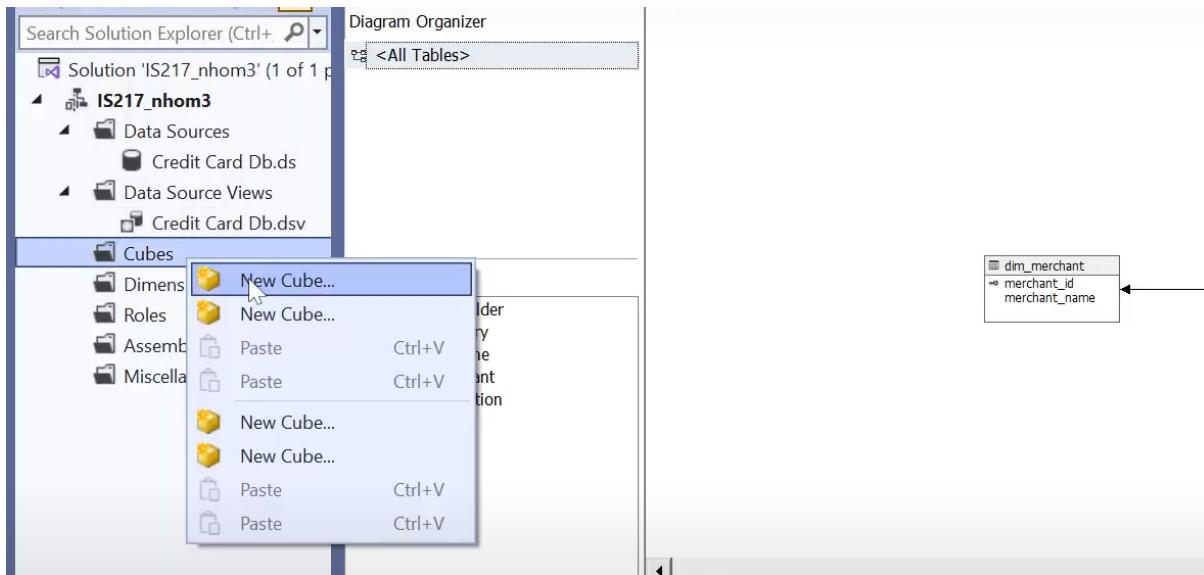
Sau khi hoàn tất quá trình, chọn file .dsv để xem data source view



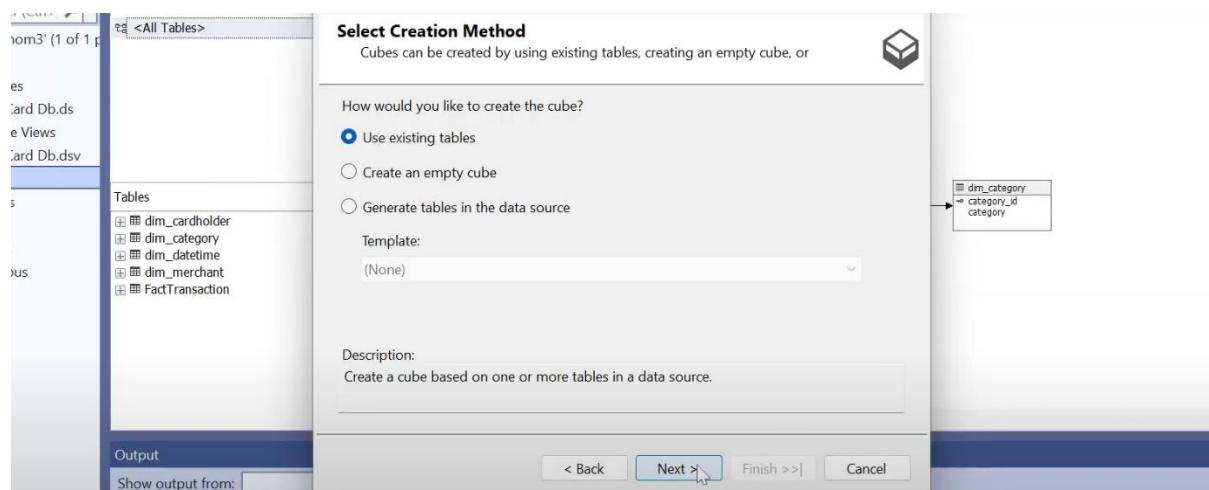
### 3.4. Xây dựng các khối (Cube) và deploy Cube

#### 3.4.1. Tạo Cube và Dimension

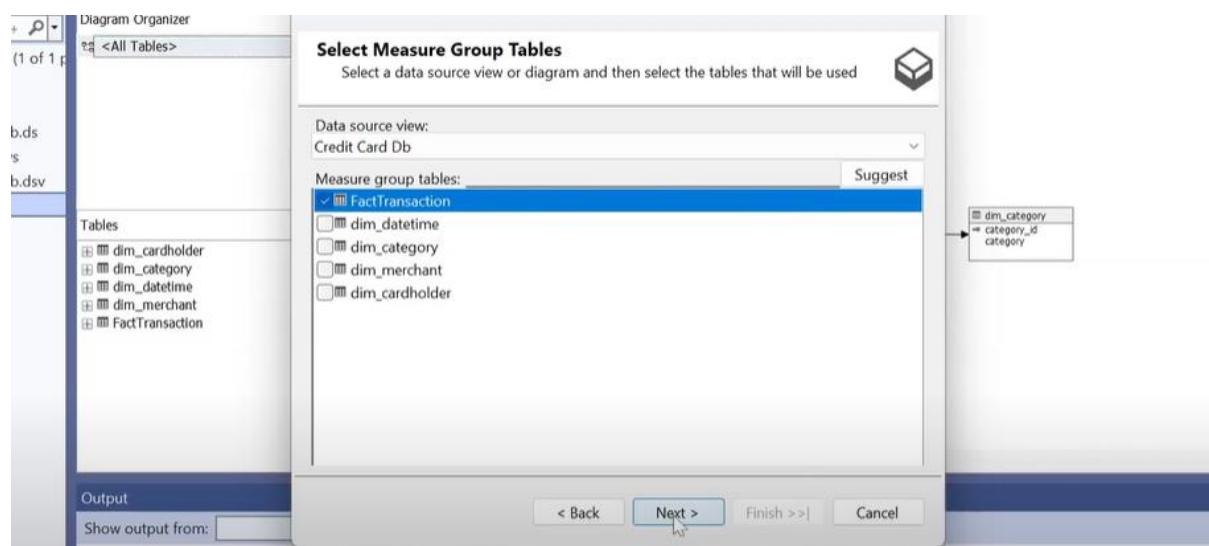
Bước 1: Tại Solution Explorer, chuột vào thư mục Cubes và chọn New Cube.



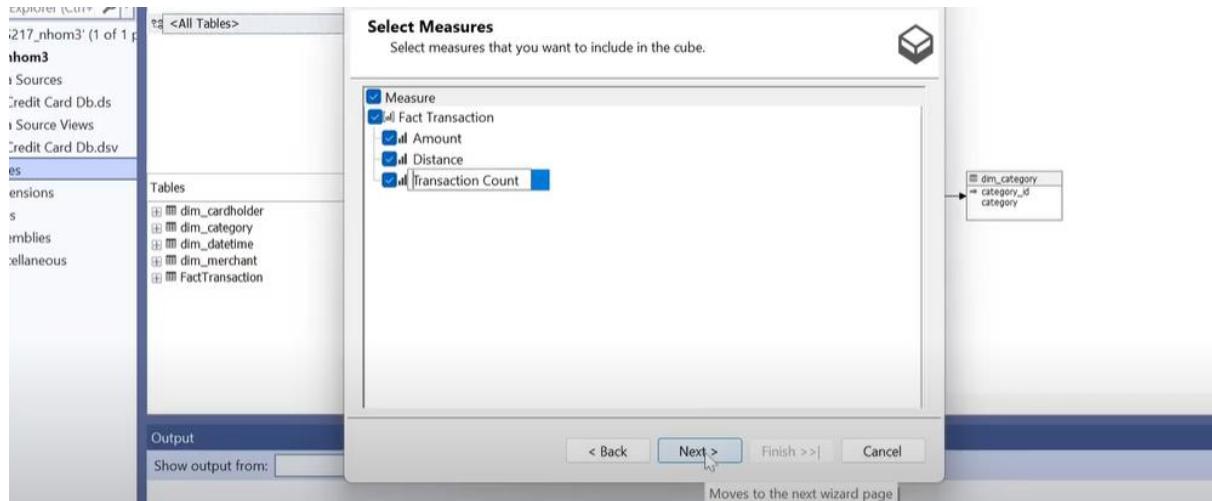
Bước 2: Chọn Use existing tables, và chọn Next để tiếp tục.



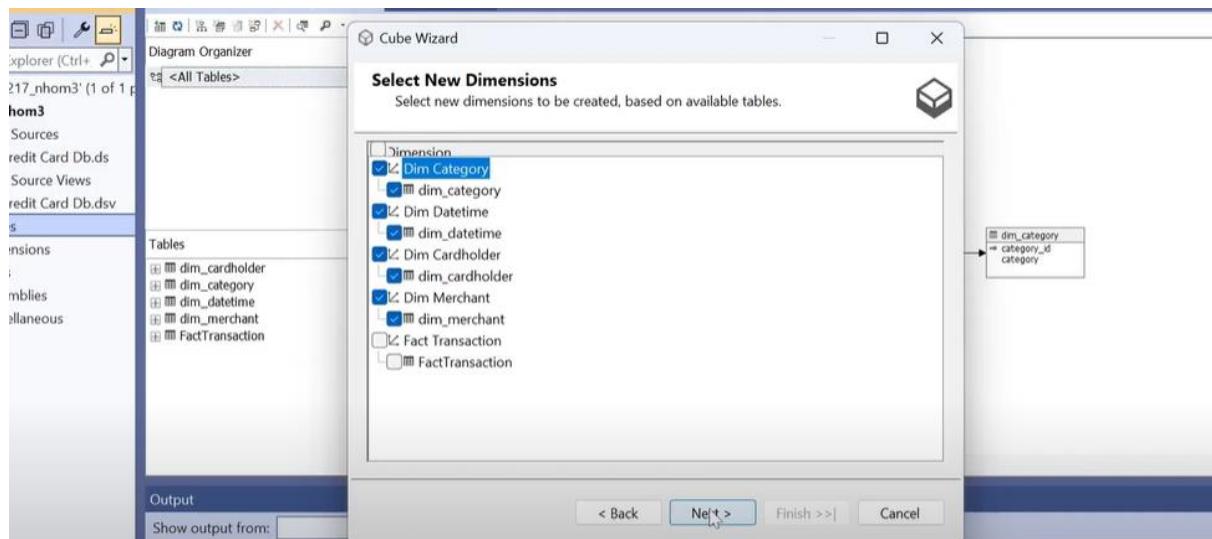
Bước 3: Chọn bảng Fact và chọn Next



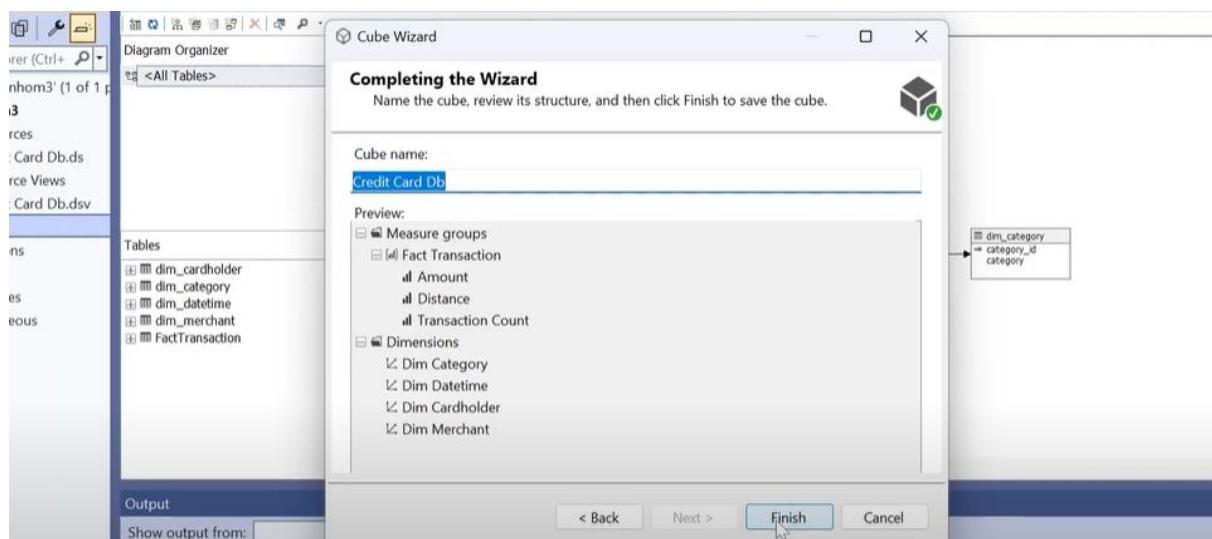
Bước 4: Chọn những độ đo để xuất, đổi tên Fact Transaction Count thành Transaction Count, sau đó chọn Next để tiếp tục.



Bước 5: Chọn danh sách bảng Dim, sau đó chọn Next để tiếp tục.



Bước 7: Xem lại các độ đo được tạo, các bảng Dim và chọn Finish để hoàn tất.



### 3.4.2. Thêm thuộc tính cho Dim

Ở khung Dimensions, lần lượt chọn vào từng bảng Dimension → Edit bảng Dimension để tiến hành chỉnh sửa từng bảng Dimension:

#### Bảng Dim\_Category

Kéo thả thuộc tính category từ cột Data Source View sang cột Attributes

The screenshot shows the 'Dimension Structure' interface. The 'Attributes' pane on the left lists 'Dim Category' with sub-items 'Category' and 'Category Id'. The 'Hierarchies' pane in the center has a placeholder message: 'To create a new hierarchy, drag an attribute here.' The 'Data Source View' pane on the right contains a table named 'dim\_category' with columns 'category\_id' and 'category'.

#### Bảng Dim\_Datetime

Kéo thả thuộc tính Day, Hour, Month, Quarter, Year từ cột Data Source View sang cột Attributes

The screenshot shows the 'Dimension Structure' interface. The 'Attributes' pane on the left lists 'Dim Datetime' with sub-items 'Datetime Id', 'Day', 'Hour', 'Month', 'Quarter', and 'Year'. The 'Hierarchies' pane in the center has a placeholder message: 'To create a new hierarchy, drag an attribute here.' The 'Data Source View' pane on the right contains a table named 'dim\_datetime' with columns 'datetime\_id', 'hour', 'day', 'month', 'quarter', and 'year'.

#### Bảng Dim\_Cardholder

Kéo thả thuộc tính Cardholder Name, City, Dob, Gender, Job, State từ cột Data Source View sang cột Attributes

The screenshot shows the 'Dimension Structure' interface. The 'Attributes' pane on the left lists 'Dim Cardholder' with sub-items 'Cardholder Id', 'Cardholder Name', 'City', 'Dob', 'Gender', 'Job', and 'State'. The 'Hierarchies' pane in the center has a placeholder message: 'To create a new hierarchy, drag an attribute here.' The 'Data Source View' pane on the right contains a table named 'dim\_cardholder' with columns 'cardholder\_id', 'cardholder\_name', 'job', 'gender', 'dob', 'city', and 'state'.

## Bảng Dim\_Merchant

Kéo thả thuộc tính Merchant Name từ cột Data Source View sang cột Attributes

The screenshot shows the 'Dimension Structure' interface. On the left, under 'Attributes', there is a list: 'Dim Merchant' (selected), 'Merchant Id', and 'Merchant Name'. In the center, under 'Hierarchies', a tooltip says: 'To create a new hierarchy, drag an attribute here.' On the right, under 'Data Source View', there is a box containing 'dim\_merchant' with attributes 'merchant\_id' and 'merchant\_name'. A cursor is hovering over the 'Merchant Name' attribute in the 'Attributes' list.

### 3.4.3 Tạo hierarchy cho chiều thời

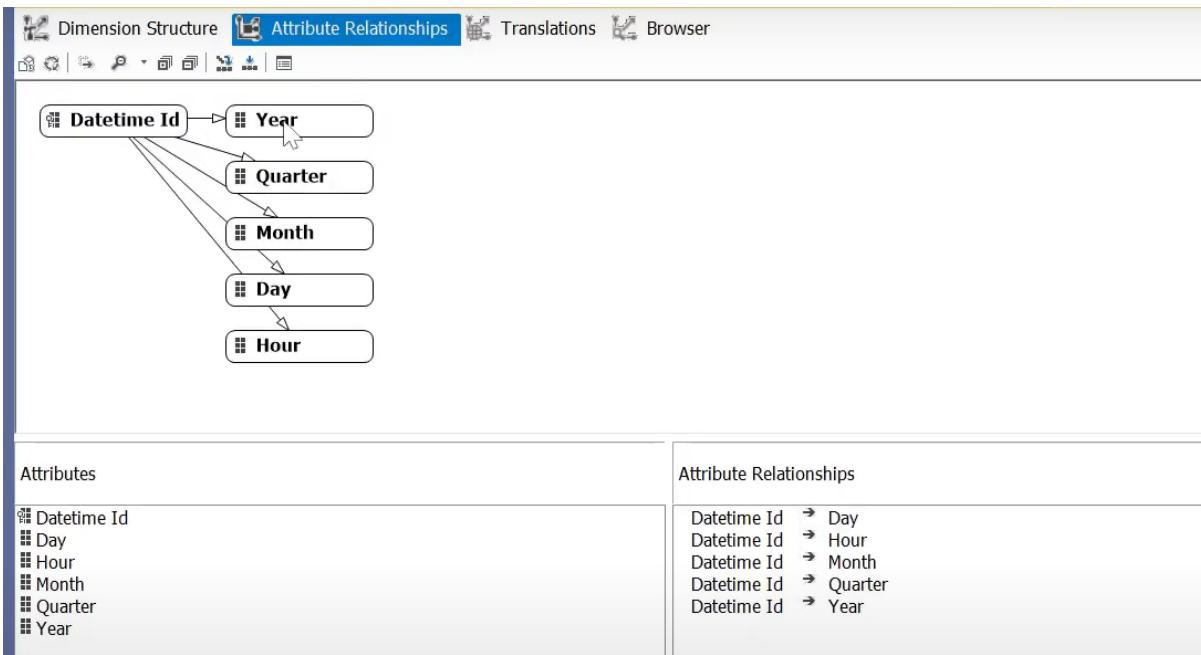
Bước 1: Chọn bảng Dim\_Datetime, tạo hierarchies theo thứ tự Year > Quarter > Month > Day > Hour bằng cách lần lượt kéo thả từng thuộc tính từ cột Attributes vào <new level> bên cột Hierarchies

The screenshot shows the 'Dimension Structure' interface. On the left, under 'Attributes', there is a list: 'Dim Datetime' (selected), 'Datetime Id', 'Day', 'Hour' (highlighted in blue), 'Month', 'Quarter', and 'Year'. In the center, under 'Hierarchies', a tooltip says: 'To create a new hierarchy, drag an attribute here.' A context menu is open over the 'Hour' attribute, with the option '<new level>' highlighted. A cursor is hovering over the '<new level>' option.

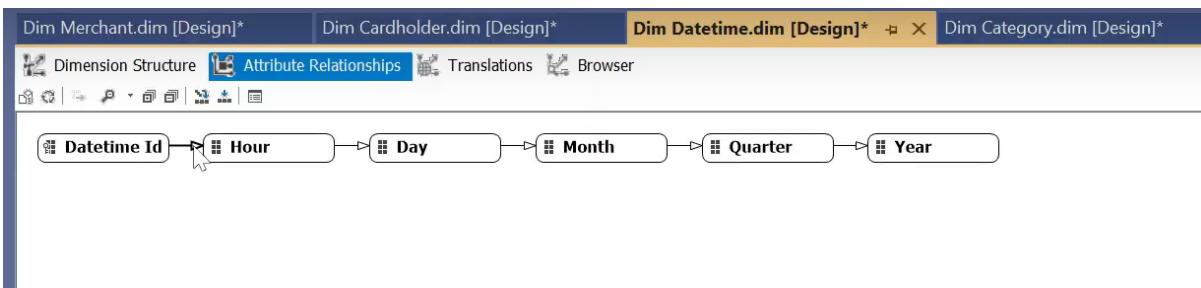
Bước 2: Có thể Remane Hierarchies để thuận tiện cho quá trình xử lý

The screenshot shows the 'Solution Explorer' interface. On the left, there is a tree view of project files: 'IS217\_nhom3' (selected), 'Data Sources', 'Data Source Views', 'Cubes', 'Dimensions' (selected), 'Dim Category.dim', 'Dim Datetime.dim' (highlighted in blue), 'Dim Cardholder.dim', 'Dim Merchant.dim', 'Roles', 'Assemblies', and 'Miscellaneous'. In the center, under 'Dimensions', there is a list: 'Dim Datetime' (selected), 'Datetime Id', 'Day', 'Hour', 'Month', 'Quarter', and 'Year'. A context menu is open over the 'Dim Datetime' dimension, with the 'Rename' option highlighted. A cursor is hovering over the 'Rename' option.

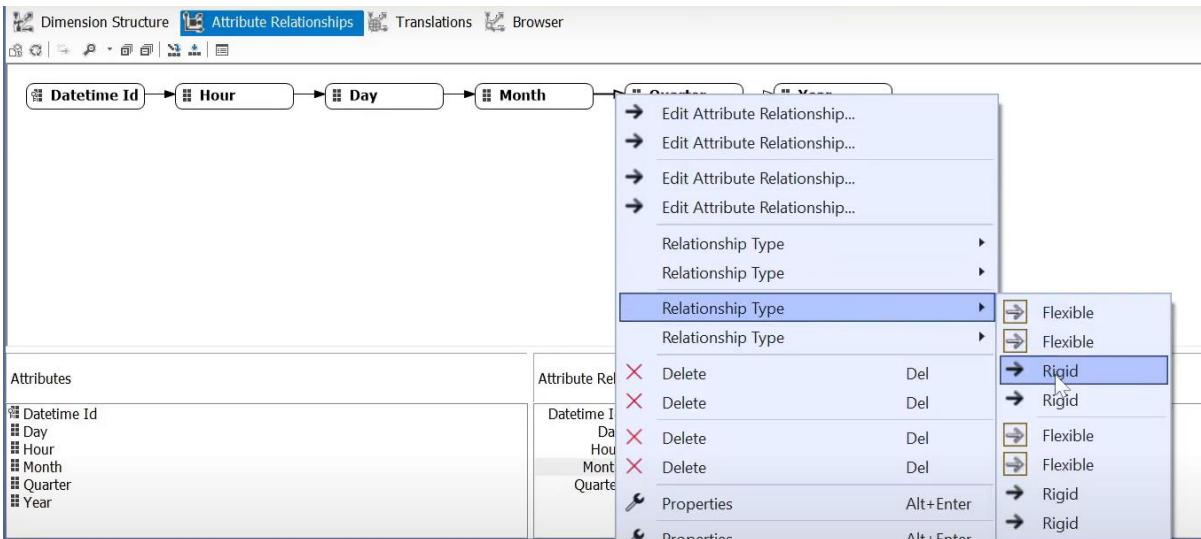
Bước 3: Tại Attribute Relationships



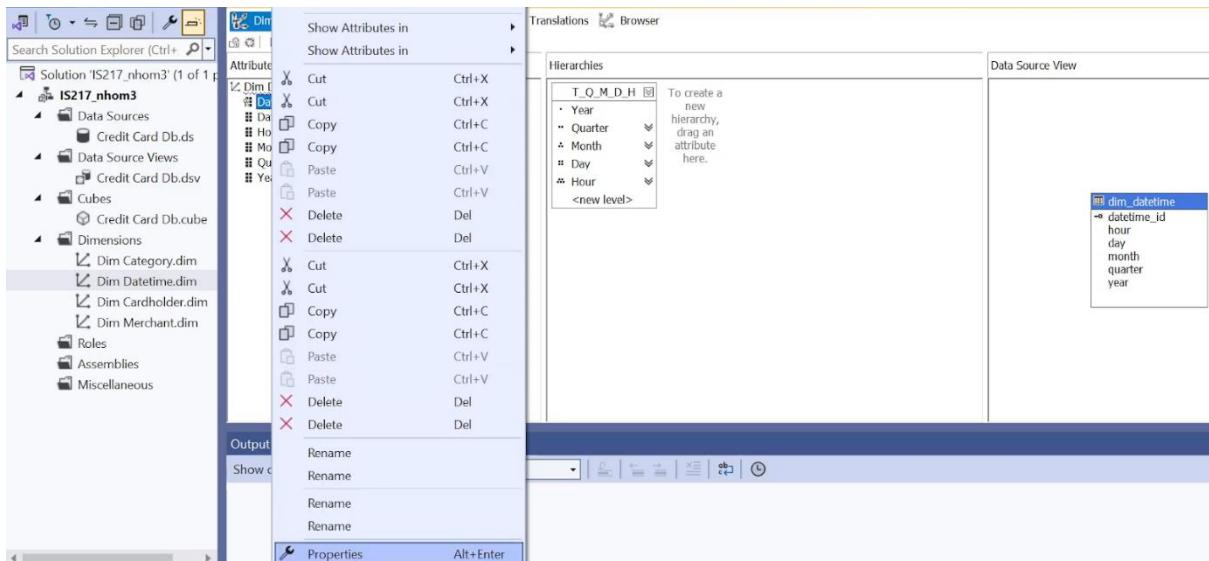
Thực hiện nối các mối quan hệ như hình sau



Bước 4: Lần lượt nhấp chuột phải vào các đường dẫn, chọn Relationship Type và đổi thành Rigid.



Bước 5: Quay lại Dimension Structure, chuột phải vào datetime\_id và chọn properties



Bước 6: Chọn thuộc tính Hour và tiến hành chỉnh sửa KeyColumn

Lần lượt chọn các thuộc tính Day, Month, Quarter, Year và nhấn >. Nhấn OK

## Bước 7: Tại NameColumn chọn Hour, nhấn OK

The screenshot shows the 'Dim Datetime.dim [Design]' tab selected. In the 'Name Column' dialog, the 'Source column' dropdown is set to 'datetime\_id'. The 'KeyColumns' field in the properties pane is highlighted.

## Bước 8: Tại OrderBy sửa thành Key

The screenshot shows the 'Dim Datetime.dim [Design]' tab selected. In the 'Hierarchy' section, the 'T\_Q\_M\_D\_H' hierarchy is shown. The 'OrderBy' field in the properties pane is highlighted and changed to 'Key'.

## Bước 9: Làm tương tự với các thuộc tính còn lại. Chọn thuộc tính Day và tiến hành chỉnh sửa KeyColumn

The screenshot shows the 'Dim Datetime.dim [Design]' tab selected. In the 'Hierarchy' section, the 'T\_Q\_M\_D\_H' hierarchy is shown. The 'KeyColumns' field in the properties pane is highlighted and changed to 'datetime.day (Integer)'.

Lần lượt chọn các thuộc tính Month, Quarter, Year và nhấn >. Nhấn OK

Bước 10: Tại NameColumn chọn Day, nhấn OK

Bước 11: Tại OrderBy sửa thành Key

Bước 12: Chọn thuộc tính Month và tiến hành chỉnh sửa KeyColumn

Lần lượt chọn các thuộc tính Quarter, Year và nhấn >. Nhấn OK

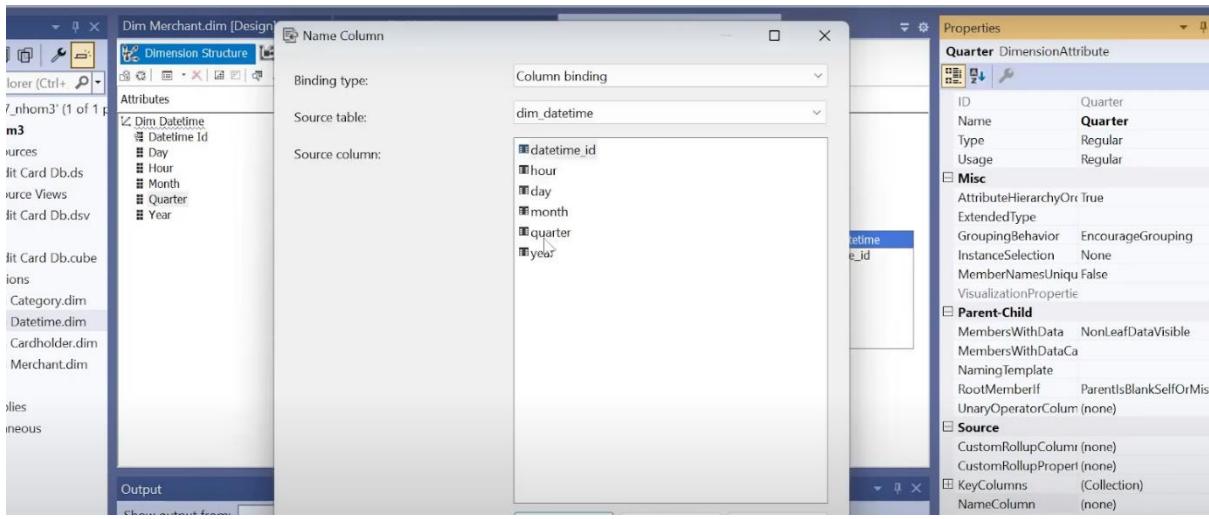
Bước 13: Tại NameColumn chọn Month, nhấn OK

Bước 14: Tại OrderBy sửa thành Key

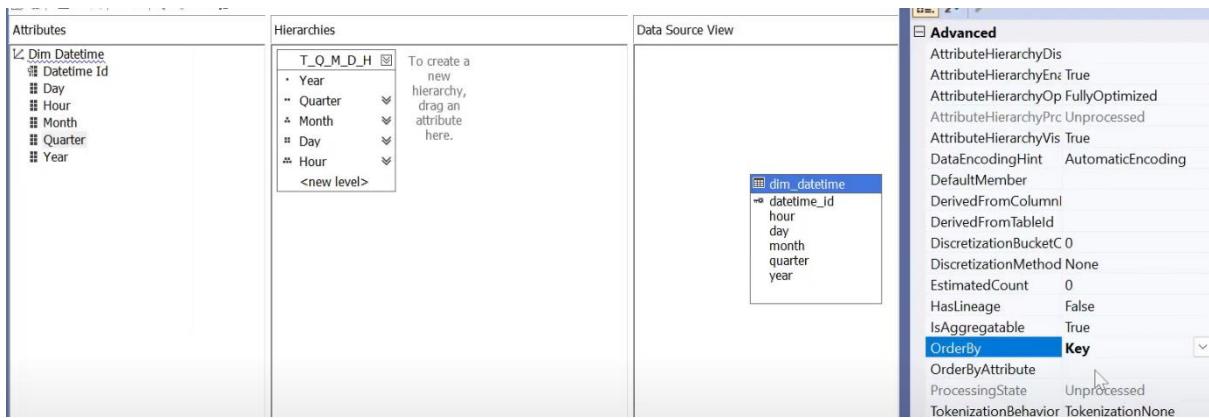
Bước 15: Chọn thuộc tính Quarter và tiến hành chỉnh sửa KeyColumn

Lần lượt chọn các thuộc tính Year và nhấn >. Nhấn OK

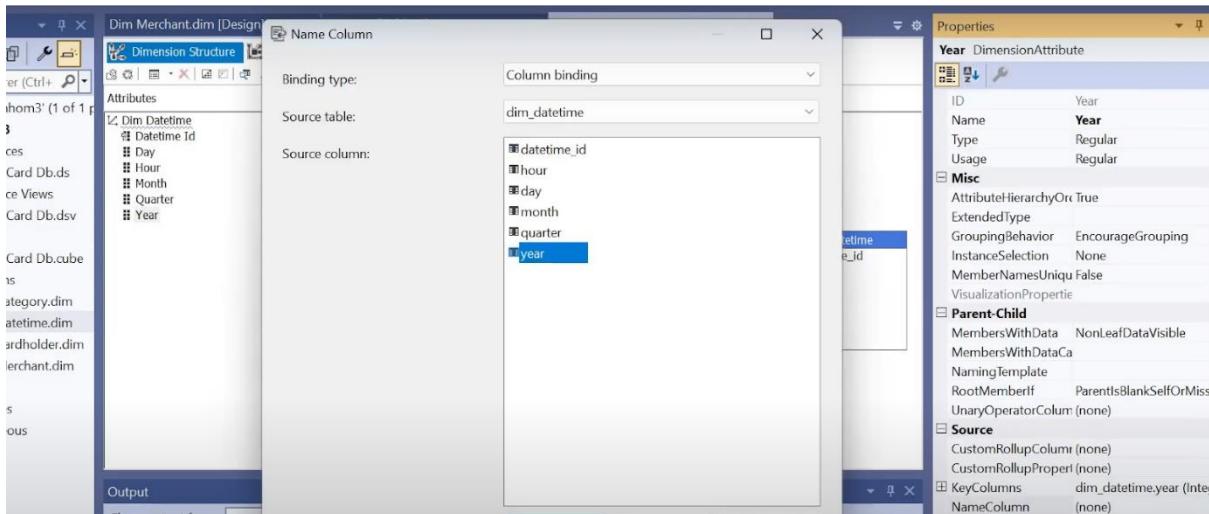
Bước 16: Tại NameColumn chọn Quarter, nhấn OK



Bước 17: Tại OrderBy sửa thành Key



Bước 18: Chọn thuộc tính Year, tại NameColumn chọn Quarter, nhấn OK



Bước 19: Tại OrderBy sửa thành Key

### 3.5 Chạy project SSAS

Tại Solution Explorer, chuột phải ở tên project và deploy

Khi Deploy thành công, màn hình sẽ hiện

Server : DESKTOP-1CC6FFG\TRANHien

Database : IS217\_nhom3

- Command
- Command
  - Processing Database 'IS217\_nhom3' completed.  
Start time: 10/22/2024 8:48:16 AM; End time: 10/22/2024 8:48:20 AM; Duration: 0:00:03
  - Processing Cube 'Credit Card Db' completed.  
Start time: 10/22/2024 8:48:18 AM; End time: 10/22/2024 8:48:20 AM; Duration: 0:00:02
  - Processing Measure Group 'Fact Transaction' completed.
  - Processing Dimension 'Dim Cardholder' completed.
  - Processing Dimension 'Dim Category' completed.
  - Processing Dimension 'Dim Datetime' completed.
  - Processing Dimension 'Dim Merchant' completed.

### 3.6 Thực hiện các câu truy vấn sử dụng ngôn ngữ MDX, Manual và Pivot Excel

#### 3.6.1 Câu 1: Tổng số lượng giao dịch và tổng tiền giao dịch theo từng năm của từng bưu cục/cửa hàng

a. MDX

```
SELECT
{
    [Measures].[Transaction Count],
    [Measures].[Amount]
} ON COLUMNS,
NON EMPTY
CROSSJOIN(
    [Dim Datetime].[T_0_M_D_H].[Year].Members,
    [Dim Merchant].[Merchant Name].Members
) ON ROWS
FROM [Credit Card Db]
```

Kết quả:

64 %

Messades Results

		Transaction Count	Amount
2019	All	924850	6.498495E+07
2019	Abbott-Rogahn	1311	83727.98
2019	Abbott-Steuber	1248	63010.31
2019	Abernathy and Sons	1245	65883.74
2019	Abshire PLC	1328	89924.98
2019	Adams, Kovacek and Kuhlman	657	34720.84
2019	Adams-Barrows	1258	64807.91
2019	Altenwerth, Cartwright and Koss	1322	99162.61

b. Manual

Language: Default

Edit as Text Import... MDX

Credit Card Db Metadata

Search Model Measure Group: <All>

Measure Group: Measures Fact Transaction Amount Distance Is Fraud Transaction Average Transaction

Dimension Hierarchy Operator Filter Expression Parameter

<Select dimension>

Year	Merchant Name	Transaction Count	Amount
2019	Abbott-Rogahn	1311	83727.98
2019	Abbott-Steuber	1248	63010.31
2019	Abernathy and Sons	1245	65883.74
2019	Abshire PLC	1328	89924.98
2019	Adams, Kovacek and ...	657	34720.84
2019	Adams-Barrows	1258	64807.91
2019	Altenwerth, Cartwrig...	1322	99162.61
2019	Altenwerth-Kilback	1754	101223.6
2019	Ankunding LLC	1392	121026
2019	Ankunding-Carroll	590	76034.77
2019	Armstrong, Walter a...	1356	70604.94
2019	Auer LLC	1321	61561.82
2019	Auer-Mosciski	1714	201062.3

c. Pivot Excel

	A	B	C	D
1				
2				
3	Row Labels	Amount	Transaction Count	
4	Abbott-Rogahn			
5	⊕ 2019	83728	1311	
6	⊕ 2020	32930.54688	533	
7	Abbott-Steuber			
8	⊕ 2019	63010.30859	1248	
9	⊕ 2020	25816.69141	515	
10	Abernathy and Sons			
11	⊕ 2019	65883.73438	1245	
12	⊕ 2020	24696.47852	506	
13	Abshire PLC			
14	⊕ 2019	89924.98438	1328	
15	⊕ 2020	37208.26953	567	
16	Adams, Kovacek and Kuhlman			
17	⊕ 2019	34720.83984	657	
18	⊕ 2020	15497.92871	283	
19	Adams-Barrows			
20	⊕ 2019	64807.90625	1258	
21	⊕ 2020	27409.07813	488	

3.6.2 Câu 2: Số tiền giao dịch trung bình tại các bưu cục/cửa hàng

a. MDX

```

WITH
    MEMBER [Measures].[Average Transaction Amount] AS
        IIF(
            [Measures].[Transaction Count] = 0,
            NULL,
            [Measures].[Amount] / [Measures].[Transaction Count]
        )

    SELECT
    {
        [Measures].[Average Transaction Amount]
    } ON COLUMNS,
    NON EMPTY
    [Dim Merchant].[Merchant Name].Members ON ROWS

FROM [Credit Card Db]

```

Kết quả:

64 %

Messages Results

	Average Transaction Amount
All	70.3509946594174
Abbott-Rogahn	63.2638540536876
Abbott-Steuber	50.3840089690868
Abernathy and Sons	51.7305602155911
Abshire PLC	67.0887862796834
Adams, Kovacek and Kuhlman	53.4242229055851
Adams-Barrows	52.8161377792096
Altenwerth, Cartwright and Koss	79.5639689469538
Altenwerth-Kilback	58.4323249600479
Ankunding LLC	84.015015600624
Ankunding-Carroll	110.637351552984

## b. Manual

Tạo new calculated member là [Average Transaction Amount]

Script Organizer

- 1 Command
- 2 CALCULATE
- 3 [query 2]
- 4 [query 3]

Calculation Tools

- Metadata Functions Templates
- Search Model

Measure Group:

Credit Card Db

Name: [Average Transaction Amount]

Parent Properties

Parent hierarchy: Measures

Parent member: Change

Expression

```
IIF(
    [Measures].[Transaction Count] = 0,
    NULL,
    [Measures].[Amount] / [Measures].[Transaction Count]
)
```

No issues found

Additional Properties

Format string: "Standard"

Kết quả:

Credit Card Db

Metadata

Search Model

Measure Group:

<All>

Credit Card Db

Measures

KPIs

Dim Cardholder

Dim Category

Dim Datetime

Dim Merchant

Calculated Members

Dimension	Hierarchy	Operator	Filter Expression	Param...
<Select dimension>				
Merchant Name	Average Transaction Am...			
Abbott-Rogahn	63.2638540536876			
Abbott-Steuber	50.3840089690868			
Abernathy and...	51.7305602155911			
Abshire PLC	67.0887862796834			
Adams, Kovac...	53.4242229055851			
Adams-Barrows	52.8161377792096			
Altenwerth, C...	79.5639689469538			
Altenwerth-Kil...	58.4323249600479			
Ankunding LLC	84.015015600624			
Ankunding-Ca...	110.637351552984			
Armstrong, W...	52.4965563595327			
Auer LLC	46.4668181508715			
Auer-Mosciski	117.336137983707			

c. Pivot Excel

A	B
Row Labels	Average Transaction Amount
Abbott-Rogahn	63.26385405
Abbott-Steuber	50.38400897
Abernathy and Sons	51.73056022
Abshire PLC	67.08878628
Adams, Kovacek and Kuhlman	53.42422291
Adams-Barrows	52.81613778
Altenwerth, Cartwright and Koss	79.56396895
Altenwerth-Kilback	58.43232496
Ankunding LLC	84.0150156
Ankunding-Carroll	110.6373516
Armstrong, Walter and Gottlieb	52.49655636
Auer LLC	46.46681815
Auer-Mosciski	117.336138
Auer-West	88.93603438
Bahringer Group	53.59953452
Bahringer, Bergnaum and Quitzon	59.34072073
... (remaining 14 rows)	... (remaining 14 average values)

### 3.6.3 Câu 3: Top 5 nghề nghiệp có số lượng giao dịch nhiều nhất trong năm 2020

a. MDX

```

SELECT
    { [Measures].[Transaction Count] } ON COLUMNS,
    TOPCOUNT(
        [Dim Cardholder].[Job].[Job],
        5,
        [Measures].[Transaction Count]
    ) ON ROWS
FROM [Credit Card Db]
WHERE ([Dim Datetime].[Year].&[2020])

```

Kết quả

	Transaction Count
Film/video editor	2758
Exhibition designer	2678
Naval architect	2488
Surveyor, land/geomatics	2482
Materials engineer	2357

## b. Manual

Tạo name set mới  
name set [query 3]

ORDER(

TOPCOUNT(

[Dim Cardholder].[Job].[Job],

5,

[Measures].[Transaction Count]

),

[Measures].[Transaction Count],

DESC

)

Script Organizer

Name: [query\_3]

Expression:

```
ORDER(
    TOPCOUNT(
        [Dim Cardholder].[Job].[Job],
        5,
        [Measures].[Transaction Count]
    ),
    [Measures].[Transaction Count],
    DESC
)
No issues found
```

Additional Properties:

Type: Dynamic

Display folder:

Kết quả:

Job	Transaction Count
Exhibition designer	2678
Film/video editor	2758
Materials engineer	2357
Naval architect	2488
Surveyor, land/geomatics	2482

### c Pivot Excel

Year	2020
Row Labels	Transaction Count
Exhibition designer	2678
Film/video editor	2758
Materials engineer	2357
Naval architect	2488
Surveyor, land/geomatics	2482

### 3.6.4 Câu 4: Top 5 thành phố có nhiều giao dịch nhất

a. MDX

```
SELECT
  {
    [Measures].[Transaction Count]
  } ON COLUMNS,
  NON EMPTY
  TOPCOUNT(
    [Dim Cardholder].[City].[City].Members, -- Lấy cấp thành phố, không bao gồm All
    5,
    [Measures].[Transaction Count]
  ) ON ROWS
FROM [Credit Card Db]
```

Kết quả:

54 %

Messages Results

	Transaction Count
Birmingham	5617
San Antonio	5130
Utica	5105
Phoenix	5075
Meridian	5060

b. Manual

Tạo name set mới

Script Organizer

```

1 CALCULATE
2 [Average Transaction Amount]
3 [query 4]
4 [query 2]
5 [query 3]

```

Calculation Tools

- Metadata
- Functions
- Templates

Search Model

Measure Group:

Credit Card Db

Measures

Name: [query 4]

Expression

```

TOPCOUNT(
    [Dim Cardholder].[City].[City].Members, 5,
    [Measures].[Transaction Count])

```

No issues found

Ln: 3 Ch: 40 SPC LF

Additional Properties

Type: Dynamic

Display folder:

Kết quả:

Edit as Text Import... MDX

Dimension	Hierarchy	Operator	Filter Expression	Param...
Dim Cardholder	City	In	query 4	<input type="checkbox"/> <input checked="" type="checkbox"/>
<Select dimension>				

City	Transaction Count
Birmingham	5617
Meridian	5060
Phoenix	5075
San Antonio	5130
Utica	5105

Calculated Members

c. Pivot Excel

A	B	C
Row Labels	Transaction Count	
Birmingham		5617
Meridian		5060
Phoenix		5075
San Antonio		5130
Utica		5105

3.6.5 Câu 5: Trung bình số tiền giao dịch của các chủ thẻ có nghề nghiệp là Science writer, Systems analyst, và Hospital pharmacist trong năm 2020 và sống tại bang có mã là “NY”, “CA”.

a. MDX

```

SELECT
    {[Dim Cardholder].[State].&[NY],
     [Dim Cardholder].[State].&[CA]}
    * {[Measures].[Average Transaction Amount]} ON COLUMNS,
    {
        [Dim Cardholder].[Job].&[Science writer],
        [Dim Cardholder].[Job].&[Systems analyst],
        [Dim Cardholder].[Job].&[Pharmacist, hospital]} ON ROWS
FROM [Credit Card Db]
WHERE ([Dim Datetime].[Year].&[2020])

```

Kết quả:

64 %

Messages Results

	NY	CA
	Average Transaction Amount	Average Transaction Amount
Science writer	78.57	51.46
Systems analyst	90.46	67.80
Pharmacist, hospital	90.79	78.57

### b. Manual

Edit as Text Import... MDX

Dimension	Hierarchy	Operator	Filter Expression	Param...
Dim Cardholder	State	In	query_5_state	<input type="checkbox"/> <input checked="" type="checkbox"/>
Dim Cardholder	Job	In	query_5_job	<input type="checkbox"/> <input checked="" type="checkbox"/>
<Select dimension>				

Search Model Measure Group: <All>

- {...} query 3
- {...} query 4
- {...} query 5\_job
- {...} query 5\_state
- {...} query 6
- + Cardholder Id
- + Cardholder Name

Calculated Members

Job	State	Average Transaction Am...
Pharmacist, hospital	CA	72.0593257050669
Pharmacist, hospital	NY	82.6469800544508
Science writer	CA	73.1836163294798
Science writer	NY	80.4289070151812
Systems analyst	CA	73.6239909686701
Systems analyst	NY	77.0499177631579

### c. Pivot Excel

Year 2020

Average Transaction Amount Column Labels

Row Labels	CA	NY
Pharmacist, hospital	78.56664861	90.78934602
Science writer	51.46142893	78.56852378
Systems analyst	67.80462346	90.45633371

### 3.6.6 Câu 6: Top 10 tên chủ thẻ có tổng số giao dịch cao nhất ở hạng mục category là gas\_transport

a. MDX

```
SELECT
{
    [Measures].[Transaction Count]
} ON COLUMNS,
TOPCOUNT(
    [Dim Cardholder].[Cardholder Name].[Cardholder Name].Members,
    10,
    [Measures].[Transaction Count]
) ON ROWS

FROM [credit Card Db]

WHERE (
    [Dim Category].[Category].&[gas_transport]
)
```

Kết quả:

Transaction Count	
Scott Martin	567
Sara Ramirez	480
Kayla Jones	477
Jennifer Black	460
Kathryn Smith	456
Carol Dillon	454
Janet Turner	452
Ana Howell	444

b. Manual

Tạo name set mới

Name: [query 6]

Expression

```
ORDER(
    TOPCOUNT(
        [Dim Cardholder].[Cardholder Name].[Cardholder Name].Members,
        10,
        [Measures].[Transaction Count]
    ),
    [Measures].[Transaction Count],
    BDESC
)
```

No issues found

Ln: 6 Ch: 11 SPC CRLF

Additional Properties

Type: Dynamic

Display folder:

Kết quả:

Cube Structure Dimension Usage Calculations KPIs Actions Partitions Aggregations Perspectives Translations Browser

Edit as Text Import... MDX

Dimension	Hierarchy	Operator	Filter Expression	Param...
Dim Cardholder	Cardholder Name	In	query 6	
Dim Category	Category	Equal	{ gas_transport }	
<Select dimension>				
Cardholder Name	Transaction Count			
Ana Howell	444			
Barbara Taylor	439			
Christine Harris	431			
Jeffrey Smith	278			
Jessica Perez	96			
Keith Sanders	391			
Mark Wood	399			
Monica Cohen	442			
Scott Martin	567			
Tammy Ayers	114			

### c. Pivot Excel

Category	gas_transport
Row Labels	Transaction Count
Ana Howell	444
Carol Dillon	454
Janet Turner	452
Jennifer Black	460
Kathryn Smith	456
Kayla Jones	477
Lauren Anderson	444
Rachel Lowe	444
Sara Ramirez	480
Scott Martin	567

### 3.6.7 Câu 7: Thống kê số lượng giao dịch, tổng số tiền giao dịch và trung bình số tiền giao dịch theo giờ trong ngày

#### a. MDX

```

SELECT
{
    [Measures].[Transaction Count],
    [Measures].[Amount],
    [Measures].[Average Transaction Amount]
} ON COLUMNS,

NON EMPTY {DRILLDOWNLEVEL(
    DRILLDOWNLEVEL(
        DRILLDOWNLEVEL(
            DRILLDOWNLEVEL(
                DRILLDOWNLEVEL([Dim Datetime].[T_Q_M_D_H])))))
} ON ROWS

FROM [Credit Card Db]

```

Kết quả:

76 %

Messages Results

	Transaction Count	Amount	Average Transaction Amount
All	1296675	9.122238E+07	\$70.35
2019	924850	6.498492E+07	\$70.27
1	173330	1.239228E+07	\$71.50
1	52525	3759751	\$71.58
1	2414	156487.1	\$64.82
0	81	6803.91	\$84.00
1	81	6384.101	\$78.82
2	78	5855.26	\$75.07
3	68	6308.96	\$92.78
4	72	5116.59	\$71.06

Activate Window

#### b. Manual

Edit as Text Import... MDX

Credit Card Db

Dimension Hierarchy Operator Filter Expression

<Select dimension>

Measure Group: <All>

Year Quarter Month Day Hour Transaction Count Amount Average Transaction...

Year	Quarter	Month	Day	Hour	Transaction Count	Amount	Average Transaction...
2019	1	1	1	0	81	6803.91	83.9988908179012
2019	1	1	1	1	81	6384.1...	78.8160566165123
2019	1	1	1	2	78	5855.26	75.0674328926282
2019	1	1	1	3	68	6308.96	92.7788229549632
2019	1	1	1	4	72	5116.59	71.0637478298611
2019	1	1	1	5	80	5871.38	73.3922485351563
2019	1	1	1	6	78	6668.21	85.4898712940705
2019	1	1	1	7	76	5292.9...	69.6435418379934
2019	1	1	1	8	77	5622.9...	73.0255935470779
2019	1	1	1	9	86	7728.76	89.8692996002907
2019	1	1	1	10	90	8109.51	90.1056640625
2019	1	1	1	11	70	5772.1	74.0012720422602

Calculated Members

c. Pivot Excel

Row Labels	Transaction Count	Amount	Average Transaction Amount
2019			
1			
1			
0	81	6803.910156	83.99889082
1	81	6384.100586	78.81605662
2	78	5855.259766	75.06743289
3	68	6308.959961	92.77882295
4	72	5116.589844	71.06374783
5	80	5871.379883	73.39224854
6	78	6668.209961	85.48987129
7	76	5292.90918	69.64354184
8	77	5622.970703	73.02559355
9	86	7728.759766	89.8692996
10	90	8109.509766	90.10566406
11	78	5772.099609	74.00127704
12	95	4964.799805	52.26105058
13	124	4903.549805	39.54475649
14	122	6709.219727	54.99360432
--	--	--	--

3.6.8 Câu 8: Tính toán số lượng giao dịch theo khoảng cách giữa địa điểm giao dịch và địa chỉ khách hàng lớn hơn 100 km.

a. MDX

```

SELECT
{
[Measures].[Transaction Count]
} ON COLUMNS,

NON EMPTY
FILTER(
[Dim Merchant].[Merchant Name].Members,
[Measures].[Distance].MEMBERVALUE > 100
) ON ROWS

FROM [Credit Card Db];

```

Kết quả:

76 %

Messages Results

	Transaction Count
All	1296675
Abbott-Rogahn	1844
Abbott-Steuber	1763
Abernathy and Sons	1751
Abshire PLC	1895
Adams, Kovacek and Kuhlman	940
Adams-Barrows	1746
Altenwerth, Cartwright and Koss	1904
Altenwerth-Kilback	2503
Ankunding LLC	1923

Activ

b. Manual

Tạo name set mới

Name: [query 8]

Expression

```
FILTER(
    [Dim Merchant].[Merchant Name].Members,
    [Measures].[Distance] > 100
)
No issues found
```

Ln: 3 Ch: 36 SPC CRLF

Additional Properties

Type: Dynamic

Display folder:

Kết quả:

Dimension: Dim Merchant    Hierarchy: Merchant Name    Operator: In    Filter Expression: query 8

Merchant Name	Transaction Count
Abbott-Rogahn	1844
Abbott-Steuber	1763
Abernathy and...	1751
Abshire PLC	1895
Adams, Kovac...	940
Adams-Barrows	1746
Altenwerth, C...	1904
Altenwerth-Kil...	2503
Ankunding LLC	1923
Ankunding-Ca...	821
Armstrong, W...	1883
Auer LLC	1836
Auer-Mosciski	2455

### c. Pivot Excel

	A	B
1	Row Labels	Transaction Count
2	Abbott-Rogahn	1844
3	Abbott-Steuber	1763
4	Abernathy and Sons	1751
5	Abshire PLC	1895
6	Adams, Kovacek and Kuhlman	940
7	Adams-Barrows	1746
8	Altenwerth, Cartwright and Koss	1904
9	Altenwerth-Kilback	2503
10	Ankunding LLC	1923
11	Ankunding-Carroll	821
12	Armstrong, Walter and Gottlieb	1883
13	Auer LLC	1836
14	Auer-Mosciski	2455
15	Auer-West	1887
16	Bahringer Group	1722
17	Bahringer, Bergnaum and Quitzon	2481
18	Bahringer, Osinski and Block	1721
19	Bahringer, Schoen and Corkery	2286
20	Bahringer-Larson	829
21	Bahringer-Streich	1812

### 3.6.9 Câu 9: Tính toán tổng số lượng giao dịch, tổng số tiền giao dịch và số tiền giao dịch trung bình theo giới tính là nữ và danh mục sản phẩm/dịch vụ là shopping\_pos

a. MDX

```
SELECT
  {
    [Measures].[Transaction Count],
    [Measures].[Amount],
    [Measures].[Average Transaction Amount]
  } ON COLUMNS
FROM [Credit Card Db]
WHERE (
  [Dim Cardholder].[Gender].&[F],
  [Dim Category].[Category].&[shopping_pos]
)
```

Kết quả:

Transaction Count	Amount	Average Transaction Amount
67831	5343501	\$78.78

b. Manual

Dimension	Hierarchy	Operator	Filter Expression
Dim Cardholder	# Gender	Equal	{ F }
Dim Category	# Category	Equal	{ shopping_pos }
<Select dimension>			

Amount	Average Transaction A...	Transaction Count
5343501	78.7766802789285	67831

c. Pivot Excel

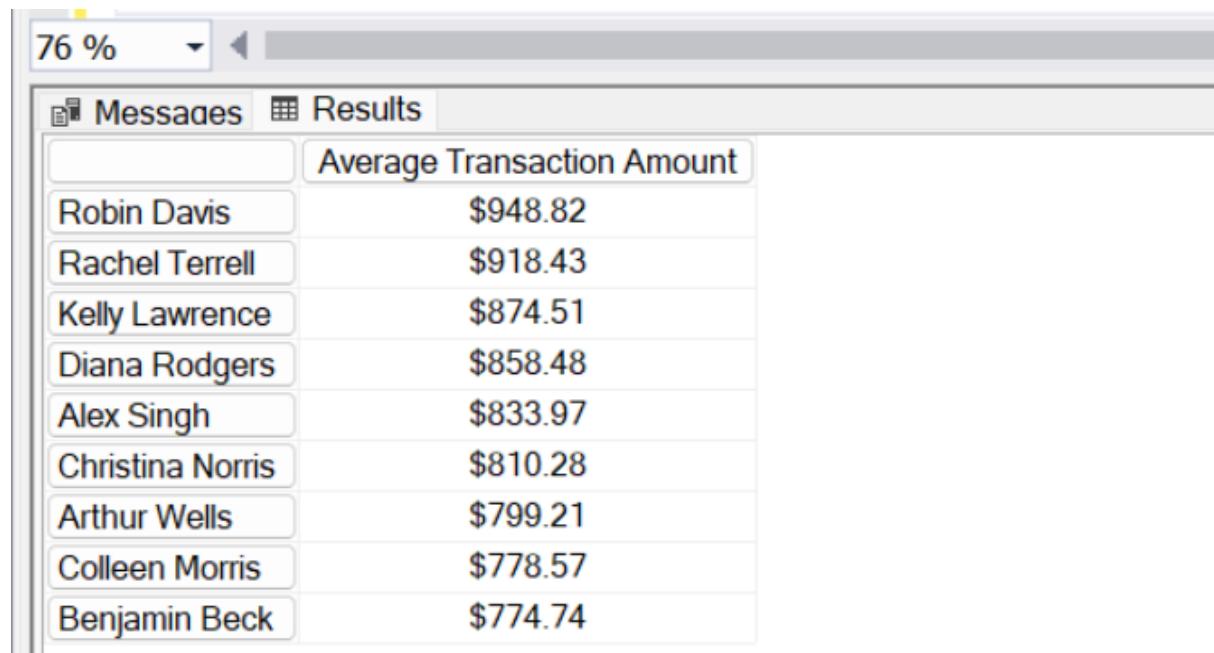
Gender	Category	Amount	Average Transaction Amount	Transaction Count
F	shopping_pos	5343501	78.77668028	67831

**3.6.10 Câu 10: Thống kê các chủ thẻ có giao dịch trung bình lớn hơn 750, xếp theo thứ tự giảm dần của giá trị giao dịch trung bình**

a. MDX

```
SELECT  
    [Measures].[Average Transaction Amount] ON COLUMNS,  
  
    NON EMPTY  
    ORDER (  
        EXCEPT(  
            [Dim Cardholder].[Cardholder Name].Members,  
            FILTER(  
                [Dim Cardholder].[Cardholder Name].Members,  
                [Measures].[Average Transaction Amount] < 750  
            )  
        ),  
        [Measures].[Average Transaction Amount],  
        BDESC  
    ) ON ROWS  
  
FROM [Credit Card Db];
```

Kết quả:



	Average Transaction Amount
Robin Davis	\$948.82
Rachel Terrell	\$918.43
Kelly Lawrence	\$874.51
Diana Rodgers	\$858.48
Alex Singh	\$833.97
Christina Norris	\$810.28
Arthur Wells	\$799.21
Colleen Morris	\$778.57
Benjamin Beck	\$774.74

b. Manual

The screenshot shows the SSAS MDX Query Editor interface. On the left, there's a navigation pane with sections for 'Credit Card Db', 'Metadata', 'Search Model', 'Measure Group:', and 'Calculated Members'. Below these are treeviews for 'Fact Transaction' and 'Calculated Members'. The main area displays an MDX query result grid. At the top of the grid, there's a header row with columns for 'Dimension', 'Hierarchy', 'Operator', 'Filter Expression', and 'Param...'. A filter is applied for 'Dim Cardholder' under 'Cardholder Name' with the operator 'In' and the filter expression 'query 11'. The data table below contains 10 rows of cardholder names and their average transaction amounts.

Cardholder Name	Average Transaction Am...
Alex Singh	833.96904296875
Arthur Wells	799.213324652778
Benjamin Beck	774.743774414063
Christina Norris	810.278599330357
Colleen Morris	778.571821732955
Diana Rodgers	858.480102539063
Kelly Lawrence	874.505719866071
Rachel Terrell	918.425564236111
Robin Davis	948.818093039773

c. Pivot Excel

	A	B	C
1	Row Labels	Average Transaction Amount	
2	Alex Singh	833.969043	
3	Arthur Wells	799.2133247	
4	Benjamin Beck	774.7437744	
5	Christina Norris	810.2785993	
6	Colleen Morris	778.5718217	
7	Diana Rodgers	858.4801025	
8	Kelly Lawrence	874.5057199	
9	Rachel Terrell	918.4255642	
10	Robin Davis	948.818093	
11			

### 3.6.11 Câu 11: Tổng số giao dịch của các chủ thẻ có năm sinh sau 2000

a. MDX:

WITH

```
 MEMBER [Measures].[DOBDate] AS
    [Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE
```

SELECT

```
{
    [Measures].[Transaction Count],
    [Measures].[Amount],
    [Measures].[Average Transaction Amount],
    [Measures].[DOBDate]
} ON COLUMNS,
NON EMPTY
FILTER(
```

```

[Dim Cardholder].[Cardholder Id].Members,
[Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE > CDate('2000-01-
01')
) ON ROWS
FROM [Credit Card Db];

```

```
--query 10
WITH
    MEMBER [Measures].[DOBDate] AS
        [Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE

SELECT
{
    [Measures].[Transaction Count],
    [Measures].[Amount],
    [Measures].[Average Transaction Amount],
    [Measures].[DOBDate]
} ON COLUMNS,
NON EMPTY
FILTER(
    [Dim Cardholder].[Cardholder Id].Members,
    [Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE > CDate('2000-01-01')
) ON ROWS
FROM [Credit Card Db];

```

76 %

	Transaction Count	Amount	Average Transaction Amount	DOBDate
All	1296675	9.122241E+07	\$70.35	All
4	537	28704.05	\$53.45	4/22/2002
10	2060	110733.9	\$53.75	7/1/2004
12	2002	119177.4	\$59.53	9/11/2000
25	3106	284013.3	\$91.44	4/8/2000
35	2042	112761.9	\$55.22	3/13/2001
39	2915	209256.4	\$71.79	5/8/2011

## b. Manual

Tạo name set

FILTER(

```

[Dim Cardholder].[Cardholder Id].Members,
[Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE > CDate('2000-01-
01'))

```

Name:

[query 10]

≈ Expression

```

FILTER(
    [Dim Cardholder].[Cardholder Id].Members,
    [Dim Cardholder].[Dob].CURRENTMEMBER.MEMBER_VALUE > CDate('2000-01-01'))

```

✓ No issues found

Ln: 4 Ch: 6

Dimension	Hierarchy	Operator	Filter Expression	Para ▲
Dim Cardholder	Cardholder Id	In	query 10	<input type="checkbox"/> <input type="checkbox"/>
<Select dimensio...				

Dob	Transaction Count	Amount	Average Transaction Amount
2000-04-08 00:00:00	3106	28401...	91.4402064552479
2000-04-29 00:00:00	2054	12399...	60.3683323697663
2000-05-10 00:00:00	2561	158571	61.9175981062085
2000-05-14 00:00:00	2525	15112...	59.8517945544554
2000-05-27 00:00:00	532	32485....	61.0622540237312
2000-07-05 00:00:00	1047	62659....	59.8469996119866
2000-08-16 00:00:00	2057	18572...	90.2898031113272
2000-09-11 00:00:00	2002	11917...	59.5291895604396
2000-09-29 00:00:00	2045	19811...	96.8778881418093
2000-10-05 00:00:00	2594	24825...	95.7020648612182
2000-10-07 00:00:00	1001	10460...	104.504386238761
2000-10-12 00:00:00	1535	14837...	96.6607390065147
2000-10-25 00:00:00	1058	99938....	94.4598003308129

c. Pivot excel

A	B	C	D
Row Labels	Transaction Count	Amount	Average Transaction Amount
4			
2002-04-22 00:00:00	537	28704.05078	53.45260853
10			
2004-07-01 00:00:00	2060	110733.875	53.75430825
12			
2000-09-11 00:00:00	2002	119177.4375	59.52918956
25			
2000-04-08 00:00:00	3106	284013.2813	91.44020646
35			
2001-03-13 00:00:00	2042	112761.9453	55.22132483
39			
2011-05-08 00:00:00	2915	209256.4219	71.78607955
43			
2005-05-20 00:00:00	501	22403.38281	44.71733096
52			
2007-06-13 00:00:00	2949	214441.9219	72.71682668
54			
2011-06-19 00:00:00	483	39034.92188	80.81764363
63			
2002-09-11 00:00:00	2588	124081.9766	47.94512232

**3.6.12 Câu 12: Với mỗi thành phố, lấy ra được khách hàng có tổng số tiền giao dịch lớn nhất**

a. MDX

SELECT

```
{[Measures].[Transaction Count]} ON COLUMNS,
NON EMPTY
Generate(
[Dim Cardholder].[City].Members,
```

```

TopCount(
CrossJoin(
    {[Dim Cardholder].[City].CURRENTMEMBER},
    [Dim Cardholder].[Cardholder Name].[Cardholder Name].Members
),
1,
[Measures].[Transaction Count]
)
) ON ROWS
FROM [Credit Card Db];

```

--12

```

SELECT {[Measures].[Transaction Count]} ON COLUMNS,
NON EMPTY
Generate(
    [Dim Cardholder].[City].Members,
    TopCount(
        CrossJoin(
            {[Dim Cardholder].[City].CURRENTMEMBER},
            [Dim Cardholder].[Cardholder Name].[Cardholder Name]
        ),
        1,
        [Measures].[Transaction Count]
    )
) ON ROWS
FROM [Credit Card Db];

```

76 %

Messages Results

		Transaction Count
All	Scott Martin	4618
Achille	Felicia Thomas	532
Acworth	Jesse Roberts	2097
Adams	Sonya Jensen	516
Afton	Dalton Jones	1028
Akron	Gregory Wood	511
Albany	Michelle Anderson	1045
Albuquerque	Kathleen Heath	519
Alder	Michael Orozco	1019
Alodo	Jordan May	1034

## b. Manual

Tạo name set

Generate(

[Dim Cardholder].[City].Members,

TopCount(

Exists(

[Dim Cardholder].[Cardholder Name].[Cardholder Name].Members,

[Dim Cardholder].[City].CurrentMember

),

1,

[Measures].[Transaction Count]

)

)

The screenshot shows the SSAS Script Editor interface. On the left, the 'Script Organizer' pane lists several items under 'CALCULATE': 'query 12' (selected), 'Fraud Transactions', 'query 10', 'query 13', 'query 11', and 'Average Transaction Amount'. Below it are 'Calculation Tools' and 'Search Model' buttons. In the center, the 'Expression' pane displays the MDX script for 'query 12'. The script uses the 'Generate' function to create a calculated member based on the specified logic. A status bar at the bottom right indicates 'No issues found' with line and character counts (Ln: 11 Ch: 6). On the right, 'Additional Properties' are set to 'Type: Dynamic'. At the bottom, the 'MDX' tab is selected in the ribbon, and the results pane shows a table of data from the 'Fact Transaction' measure group, filtered by the 'query 12' dimension.

Dimension	Hierarchy	Operator	Filter Expression	Par...
Dim Cardholder	Cardholder ...	In	query 12	<input type="checkbox"/> <input type="checkbox"/>
<Select dimen...				

City	Cardholder Name	Transaction Co...
Achille	Felicia Thomas	532
Acworth	Jesse Roberts	2097
Adams	Sonya Jensen	516
Afton	Dalton Jones	1028
Akron	Gregory Wood	511
Albany	Michelle Anderson	1045
Albuquerque	Kathleen Heath	519
Albuquerque	Susan Garcia	506
Alder	Michael Orozco	1019
Aledo	Tordan May	1034

c. Pivot excel

	A	B	C
1	Row Labels	Transaction Count	
2	Aaron Murray		
3	Meadville	2050	
4	Aaron Pena		
5	Burke	1476	
6	Aaron Rogers		
7	Valentine	508	
8	Aaron Stewart		
9	Winthrop	537	
10	Adam Keller		
11	Sardis	521	
12	Adam Kirk		
13	Big Indian	1556	
14	Adam Mcdonald		
15	South Hero	1022	
16	Adam Riddle		
17	Mount Saint Joseph	2082	
18	Adam Santos		
19	Glendale	2603	
20	Adam Stark		
21	Mc Veytown	2060	

3.6.13 Câu 13: Thống kê phân bố tổng giá trị giao dịch theo từng category, xếp theo thứ tự bảng chữ cái của danh mục

a. MDX:

```

SELECT
    [Measures].[Amount] ON COLUMNS,
    NON EMPTY
    ORDER(
        [Dim Category].[Category].Members,
        [Dim Category].[Category].CURRENTMEMBER.NAME,
        ASC
    ) ON ROWS
FROM [Credit Card Db];

```

```
--query 13
SELECT
    [Measures].[Amount] ON COLUMNS,
    NON EMPTY
    ORDER(
        [Dim Category].[Category].Members,
        [Dim Category].[Category].CURRENTMEMBER.NAME,
        ASC
    ) ON ROWS
FROM [Credit Card Db];
```

76 %

	Amount
All	9.122241E+07
entertainment	6036680
food_dining	4672459
gas_transport	8351735
grocery_net	2439413
grocery_pos	1.446082E+07
health_fitness	4653109
home	7173930
kids_pets	6503676
misc_net	5117708
misc_pos	5009582
personal_care	4353449
shopping_net	8625147
shopping_pos	9307989
travel	4516722

b. Mannual

Tạo name set

Name:

[query 13]

✗ Expression

```
ORDER(
    [Dim Category].[Category].Members,
    [Dim Category].[Category].CURRENTMEMBER.NAME,
    ^ASC
)
```

✓ No issues found

Ln: 5 Ch: 6

Category	Amount
entertainment	6036680
food_dining	4672459
gas_transport	8351734
grocery_net	2439413
grocery_pos	1.446082...
health_fitness	4653109
home	7173930
kids_pets	6503676
misc_net	5117708
misc_pos	5009582
personal_care	4353449
shopping_net	8625148
shopping_pos	9307990
travel	4516722

c. Pivot excel

Row Labels	Amount
entertainment	6036679
food_dining	4672459
gas_transport	8351734
grocery_net	2439412.5
grocery_pos	14460821
health_fitness	4653108.5
home	7173930
kids_pets	6503675.5
misc_net	5117708.5
misc_pos	5009582
personal_care	4353449
shopping_net	8625148
shopping_pos	9307990
travel	4516722
<b>Grand Total</b>	<b>91222440</b>

**3.6.14 Câu 14: Thống kê tổng số lần giao dịch và tổng số tiền giao dịch theo từng quý trong năm 2020 của category “home” hoặc “kids\_pets” theo giới tính**

a. MDX

```

SELECT
    {
        [Measures].[Transaction Count],
        [Measures].[Amount]
    } ON COLUMNS,
    NON EMPTY
        [Dim Datetime].[Year].[2020] *
        [Dim Datetime].[Quarter].Members *
        [Dim Cardholder].[Gender].Members ON ROWS
FROM [Credit Card Db]
WHERE (
    FILTER(
        [Dim Category].[Category].Members,
        [Dim Category].[Category].CurrentMember.MEMBERVALUE = "home"
    OR
        [Dim Category].[Category].CurrentMember.MEMBERVALUE = "kids_pets"
    )
)
```

```
-- query 14
SELECT
{
    [Measures].[Transaction Count],
    [Measures].[Amount]
} ON COLUMNS,
NON EMPTY
[Dim Datetime].[Year].[2020] *
[Dim Datetime].[Quarter].Members *
[Dim Cardholder].[Gender].Members ON ROWS

FROM [Credit Card Db]

WHERE (
    FILTER(
        [Dim Category].[Category].Members,
        [Dim Category].[Category].CurrentMember.MEMBERVALUE = "home"
        OR
        [Dim Category].[Category].CurrentMember.MEMBERVALUE = "kids_pets"
    )
)
```

76 %

Messages Results

			Transaction Count	Amount
2020	All	All	67657	3925101
2020	All	F	36641	2056413
2020	All	M	31016	1868688

76 %

Messages Results

			Transaction Count	Amount
2020	All	All	67657	3925101
2020	All	F	36641	2056413
2020	All	M	31016	1868688
2020	1	All	31629	1831240
2020	1	F	17162	957909.1
2020	1	M	14467	873331.1
2020	2	All	36028	2093861
2020	2	F	19479	1098504

b. Manual

The screenshot shows the SSAS MDX Editor interface. On the left, there's a navigation pane with 'Credit Card Db' selected, followed by 'Metadata' and 'Search Model'. Below these are sections for 'Measure Group' (with '<All>'), 'Calculated Members', and a dimension hierarchy tree for 'Dim Category' (Job, State, Category, Category Id) and 'Dim Datetime'. On the right, there's a 'Dimension' configuration panel with filters for 'Dim Datetime' (Year: 2020, Operator: Equal, Filter Expression: { 2020 }) and 'Dim Category' (Category: kids\_pets, home, Operator: Equal, Filter Expression: { kids\_pets, home }). A results grid displays transaction data:

Year	Quarter	Gender	Transaction Count	Amount
2020	1	F	17162	957909.1
2020	1	M	14467	873331.1
2020	2	F	19479	1098504
2020	2	M	16549	995356.6

c. Pivot excel

	A	B	C
1	Year	2020	
2	Category	(Multiple Items)	
3			
4	Row Labels	Transaction Count	Amount
5	1		
6	F	17162	957909.125
7	M	14467	873331.125
8	2		
9	F	19479	1098504.25
10	M	16549	995356.5625
11	Grand Total	67657	3925101.25
12			

### 3.6.15 Câu 15: Thống kê tổng số giao dịch và tổng số tiền theo category của từng nghề nghiệp

a. MDX

```

SELECT
{
[Measures].[Transaction Count],
[Measures].[Amount]
} ON COLUMNS,
NON EMPTY
[Dim Cardholder].[Job].[Job].Members *
[Dim Category].[Category].Members ON ROWS
FROM [Credit Card Db]
    
```

```
-- query 15
SELECT
    {
        [Measures].[Transaction Count],
        [Measures].[Amount]
    } ON COLUMNS,
    NON EMPTY
    {[Dim Cardholder].[Job].[Job].Members * 
    [Dim Category].[Category].Members} ON ROWS
FROM [Credit Card Db]
```

76 %

Messages Results

		Transaction Count	Amount
Academic librarian	All	1041	72478.21
Academic librarian	entertainment	82	4749.46
Academic librarian	food_dining	62	2853.13
Academic librarian	gas_transport	116	8250.4
Academic librarian	grocery_net	54	2705.36
Academic librarian	grocery_pos	95	6561.919
Academic librarian	health_fitness	79	4393.09
Academic librarian	home	111	7799.658
Academic librarian	kids_pets	68	4373.379

## b. Manual

The screenshot shows the SSAS Management Studio environment. At the top, there's a toolbar with various icons like Edit as Text, Import..., MDX, and others. Below the toolbar is a message bar indicating the query is at 76% completion.

The main area has two tabs: Messages and Results. The Results tab is active and displays the same data as the previous screenshot, showing transaction counts and amounts for different job categories.

To the left, there's a navigation pane with a tree view of the cube structure. It shows the cube name 'Credit Card Db', a 'Metadata' node, and a 'Search Model' node. Under 'Measure Group', it lists '' and 'Fact Transaction'. The 'Fact Transaction' group contains measures: Amount, Average Transaction Amount, Distance, Is Fraud, and Transaction Count.

At the bottom right, there's a preview pane showing the results of the MDX query. The columns are Job, Category, Transaction Count, and Amount. The data matches the table above.

## c. Pivot excel

Row Labels	Transaction Count	Amount
Academic librarian		
entertainment	82	4749.460449
food_dining	62	2853.129883
gas_transport	116	8250.400391
grocery_net	54	2705.360352
grocery_pos	95	6561.919434
health_fitness	79	4393.089844
home	111	7799.657715
kids_pets	68	4373.379395
misc_net	51	3920.840088
misc_pos	76	5657.060547
personal_care	53	1722.309814
shopping_net	76	15367.21094
shopping_pos	85	3776.209961
travel	33	348.1799622
Accountant, chartered		
gas_transport	1	9.840000153
grocery_pos	5	1569.73999
misc_net	1	761.6300049
misc_pos	1	7.840000153

## PHẦN 4: QUÁ TRÌNH TẠO LẬP BÁO CÁO

### 4.1 Chuẩn bị công cụ

Để thực hiện quá trình tạo báo cáo, cần chuẩn bị các công cụ sau:

- Power BI



- Looker Studio



#### 4.2 Trích xuất dữ liệu từ kho dữ liệu dưới dạng .csv

Bước 1: Tại SQL Server, kết nối tới Database Engine chứa database đã tạo ở quá trình SSIS.

Bước 2: Thực thi câu lệnh

SELECT

```
ft.transaction_id,  
dm.merchant_name,  
dc.cardholder_name,  
dc.job,  
dc.gender,  
dc.dob,  
dc.city,  
dcat.category,  
dc.state,  
dd.hour,  
dd.day,  
dd.month,  
dd.quarter,  
dd.year,  
ft.amount,  
ft.distance,  
ft.is_fraud  
FROM [dbo].[FactTransaction] AS ft  
INNER JOIN [dbo].[dim_cardholder] AS dc ON ft.cardholder_id = dc.cardholder_id  
INNER JOIN [dbo].[dim_category] AS dcat ON ft.category_id = dcat.category_id  
INNER JOIN [dbo].[dim_datetime] AS dd ON ft.datetime_id = dd.datetime_id  
INNER JOIN [dbo].[dim_merchant] AS dm ON ft.merchant_id = dm.merchant_id
```

Bước 3: Nhấn chuột phải vào vùng kết quả, chọn Save Result As. Sau đó lựa chọn nơi lưu file .csv và nhấn Save.

SQLQuery1.sql - D...\FG\TRAN HIEN (59)\*

```

SELECT
    dm.merchant_name,
    dc.cardholder_name,
    dc.job,
    dc.gender,
    dc.dob,
    dc.city,
    dcat.category,
    dc.state,
    dd.hour,
    dd.day,
    dd.month,
    dd.quarter,
    dd.year,
    ft.amount,
    ft.distance,
    ft.is_fraud
FROM [dbo].[FactTransaction] AS ft
INNER JOIN [dbo].[dim_cardholder] AS dc ON ft.cardholder_id = dc.cardholder_id
INNER JOIN [dbo].[dim_category] AS dcat ON ft.category_id = dcat.category_id
INNER JOIN [dbo].[dim_datetime] AS dd ON ft.datetime_id = dd.datetime_id
INNER JOIN [dbo].[dim_merchant] AS dm ON ft.merchant_id = dm.merchant_id

```

76 %

Results Messages

merchant_name	Copy Ctrl+C	gender	dob	city	category	state	hour	day	month	
Abshire PLC	Copy	centre manager	M	1981-12-23 00:00:00.000	Meadville	entertainment	MO	14	28	5
Altenwerth, Cartwright	Copy with Headers Ctrl+Shift+C	centre manager	M	1981-12-23 00:00:00.000	Meadville	shopping_net	MO	8	19	9
Auer LLC	Select All Ctrl+A	centre manager	M	1981-12-23 00:00:00.000	Meadville	personal_care	MO	12	24	6
Bahringer, Schoen	Save Results As...	centre manager	M	1981-12-23 00:00:00.000	Meadville	shopping_pos	MO	10	5	8
Bartoletti-Wunsch	Page Setup...	centre manager	M	1981-12-23 00:00:00.000	Meadville	gas_transport	MO	9	21	8
Bauch-Raynor	Print... Ctrl+P	centre manager	M	1981-12-23 00:00:00.000	Meadville	grocery_pos	MO	5	19	11
Baumbach, Strosik		centre manager	M	1981-12-23 00:00:00.000	Meadville	shopping_pos	MO	16	3	8

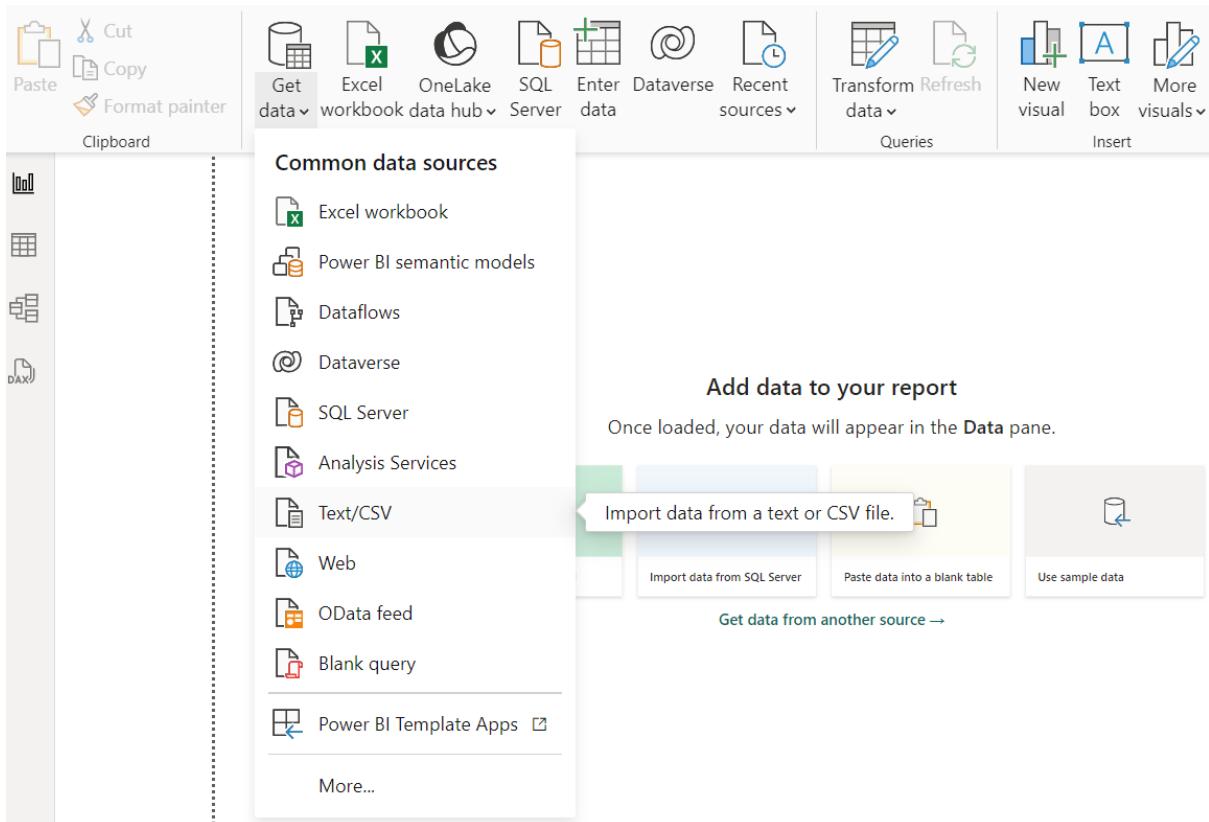
## 4.3 Thực hiện tạo bảng biểu bằng Power BI

### 4.3.1 Tạo mới báo cáo và upload dữ liệu vào Power BI

Bước 1: Tại giao diện chính của Power BI, chọn Blank report

The screenshot shows the Power BI Home page. On the left, there are navigation links for 'Home', 'Open', 'Sign in', 'Options and settings', and 'About'. The main area has two sections: 'Select a data source or start with a blank report' and 'Recommended'. In the first section, there are six options: 'Blank report' (selected), 'OneLake data hub', 'Excel workbook', 'SQL Server', 'Learn with sample data', and 'Get data from other sources'. Below this is a 'Getting started' card with a diagram of data flow and the text 'Intro—What is Power BI?'. At the bottom of the page are buttons for 'Recent', 'Shared with me', a search bar 'Filter by keyword', and a 'Filter' dropdown.

Bước 2: Chọn Get data tại mục Home trên thanh công cụ. Chọn Text/CSV.



### Bước 3: Chọn file csv đã lưu từ ô đĩa

The screenshot shows the Power BI file browser. At the top, there's a navigation bar with a folder icon, the path '« OLAP > PBI', a search bar labeled 'Search PBI', and a help icon. Below this is a table with columns 'Name' and 'Date modified'. A single file, 'credit\_card\_transactions.csv', is listed with a modified date of '10/11/2024 2:03 CH'. This file is highlighted with a blue border. At the bottom of the screen, a file selection dialog is open. It shows the file name 'credit\_card\_transactions.csv' in the 'File name:' field and 'Text Files (\*.txt;\*.csv;\*.prn)' in the 'File type:' dropdown. There are 'Open' and 'Cancel' buttons at the bottom right of the dialog.

## Bước 4: Tại giao diện, chọn Transform Data

credit\_card\_transactions.csv

File Origin: 65001: Unicode (UTF-8)

Delimiter: Comma

Data Type Detection: Based on first 200 rows

Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8
Abshire PLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	entertainment	MO
Altenwerth, Cartwright and Koss	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	shopping_net	MO
Auer LLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	personal_care	MO
Bahringer, Schoen and Corkery	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	shopping_pos	MO
Bartoletti-Wunsch	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	gas_transport	MO
Bauch-Raynor	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	grocery_pos	MO
Baumbach, Strosin and Nicolas	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	shopping_pos	MO
Bednar PLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	kids_pets	MO
Beier and Sons	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	home	MO
Berge-Ullrich	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	home	MO
Bernhard, Grant and Langworth	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	shopping_pos	MO
Block Group	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	misc_pos	MO
Block Group	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	misc_pos	MO
Bogisich-Weimann	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	kids_pets	MO
Boyer-Haley	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	travel	MO
Breitenberg-Hermiston	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	kids_pets	MO
Brown Inc	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	kids_pets	MO
Brown-Greenholt	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	entertainment	MO
Cartwright-Harris	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	grocery_pos	MO
Cole PLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00 SA	Meadville	grocery_pos	MO

Extract Table Using Examples      Load      Transform Data      Cancel

## Bước 5: Kiểm tra các kiểu dữ liệu xem đã tương ứng với thuộc tính hay chưa

Queries [1]

credit\_card\_transactions

Column1	Column2	Column3	Column4	Column5
Abshire PLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Altenwerth, Cartwright and Koss	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Auer LLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bahringer, Schoen and Corkery	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bartoletti-Wunsch	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bauch-Raynor	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Baumbach, Strosin and Nicolas	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bednar PLC	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Beier and Sons	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Berge-Ullrich	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bernhard, Grant and Langworth	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Block Group	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Block Group	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00
Bogisich-Weimann	Aaron Murray	Tourist information centre manager	M	23/12/1981 12:00:00

## Bước 6: Nhấp chuột phải vào từng cột và dùng lệnh Rename để đổi tên các cột cho đúng

ABC Column1 ABC Column2

- Valid
- Error
- Empty

467 distinct, 206 unique

1	Abshire PLC
2	Altenwerth, Cartwright
3	Auer LLC
4	Bahringer, Schoen and
5	Bartoletti-Wunsch
6	Bauch-Raynor
7	Baumbach, Strosin and
8	Bednar PLC
9	Beier and Sons
10	Berge-Ullrich
11	Bernhard, Grant and La
12	Block Group
13	Block Group
14	Bogisich-Weimann
15	<

Copy

Remove

Remove Other Columns

Duplicate Column

Add Column From Examples...

Remove Duplicates

Remove Errors

Change Type

Transform

Replace Values...

Replace Errors...

Split Column

Group By...

Fill

Unpivot Columns

Unpivot Other Columns

Unpivot Only Selected Columns

Rename...

Move

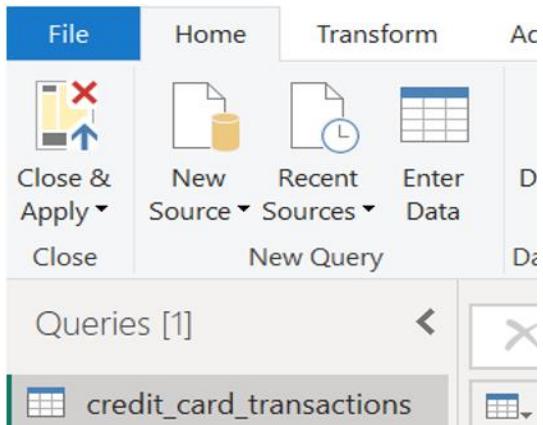
Drill Down

Add as New Query

credit\_card\_transactions

	ABC merchant_name	ABC cardholder_name	ABC job	ABC gender	ABC dob
	Valid 0% Error 0% Empty 0%	100%	Valid 0% Error 0% Empty 0%	100%	Valid 0% Error 0% Empty 0%
	467 distinct, 206 unique	12 distinct, 0 unique	12 distinct, 0 unique	2 distinct, 0 unique	12 distinct, 0 un
1	Abshire PLC	Aaron Murray	Tourist information centre manager	M	23/12
2	Altenwerth, Cartwright and Koss	Aaron Murray	Tourist information centre manager	M	23/12
3	Auer LLC	Aaron Murray	Tourist information centre manager	M	23/12
4	Bahringer, Schoen and Corkery	Aaron Murray	Tourist information centre manager	M	23/12
5	Bartoletti-Wunsch	Aaron Murray	Tourist information centre manager	M	23/12
6	Bauch-Raynor	Aaron Murray	Tourist information centre manager	M	23/12
7	Baumbach, Strosin and Nicolas	Aaron Murray	Tourist information centre manager	M	23/12
8	Bednar PLC	Aaron Murray	Tourist information centre manager	M	23/12
9	Beier and Sons	Aaron Murray	Tourist information centre manager	M	23/12
10	Berge-Ullrich	Aaron Murray	Tourist information centre manager	M	23/12
11	Bernhard, Grant and Langworth	Aaron Murray	Tourist information centre manager	M	23/12
12	Block Group	Aaron Murray	Tourist information centre manager	M	23/12
13	Block Group	Aaron Murray	Tourist information centre manager	M	23/12
14	Bogisich-Weimann	Aaron Murray	Tourist information centre manager	M	23/12
15	<				

## Bước 7: Khi đã hoàn thành, nhấn Close & Apply



## Bước 8:

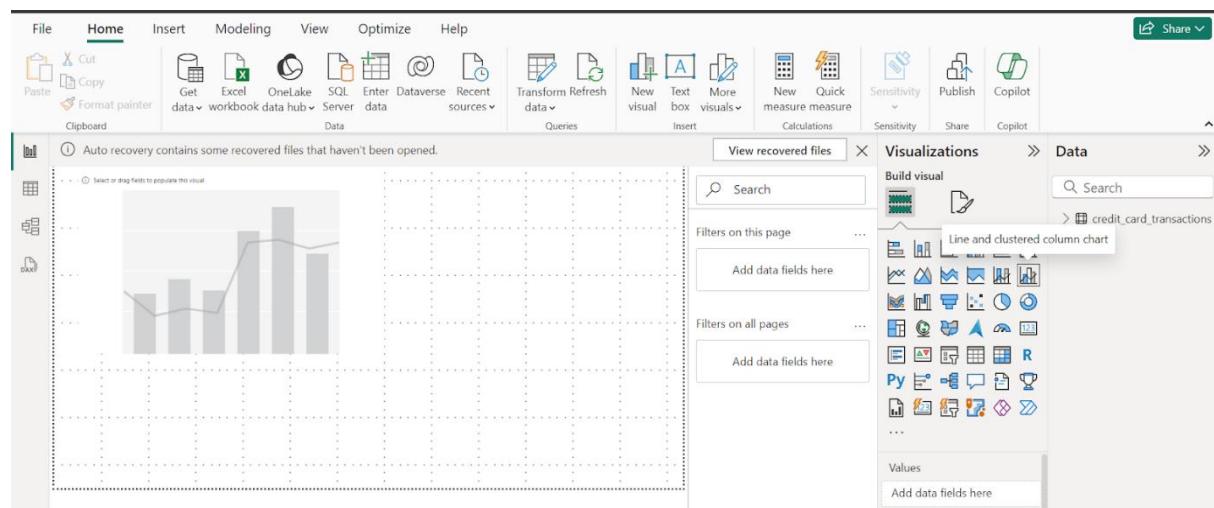
The screenshot shows the Power BI Data view pane. The 'credit\_card\_transactions' query is expanded, displaying a list of columns. Each column is preceded by a checkbox and a mathematical operator ( $\sum$  or  $\prod$ ). The columns listed are: amount, cardholder\_name, category, city, day, distance, dob, gender, hour, is\_fraud, job, merchant\_name, month, quarter, state, and transaction id.

Column	Operator
amount	$\sum$
cardholder_name	$\prod$
category	$\prod$
city	$\prod$
day	$\sum$
distance	$\sum$
dob	$\prod$
gender	$\prod$
hour	$\sum$
is_fraud	$\sum$
job	$\prod$
merchant_name	$\prod$
month	$\sum$
quarter	$\sum$
state	$\prod$
transaction id	$\sum$

### 4.3.2 Báo cáo 1

Nội dung: Tổng số giao dịch và tổng giá trị giao dịch theo giờ trong ngày

Bước 1: Chọn biểu đồ Line and clustered column chart

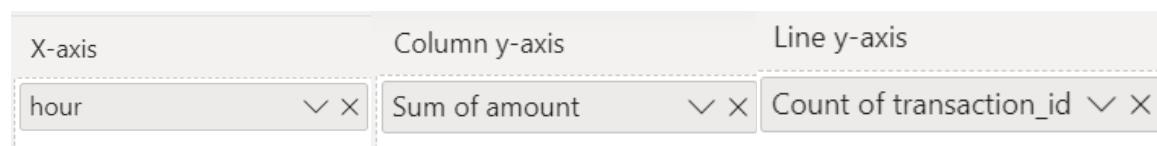


Bước 2: Chọn từ dữ liệu từ các thuộc tính:

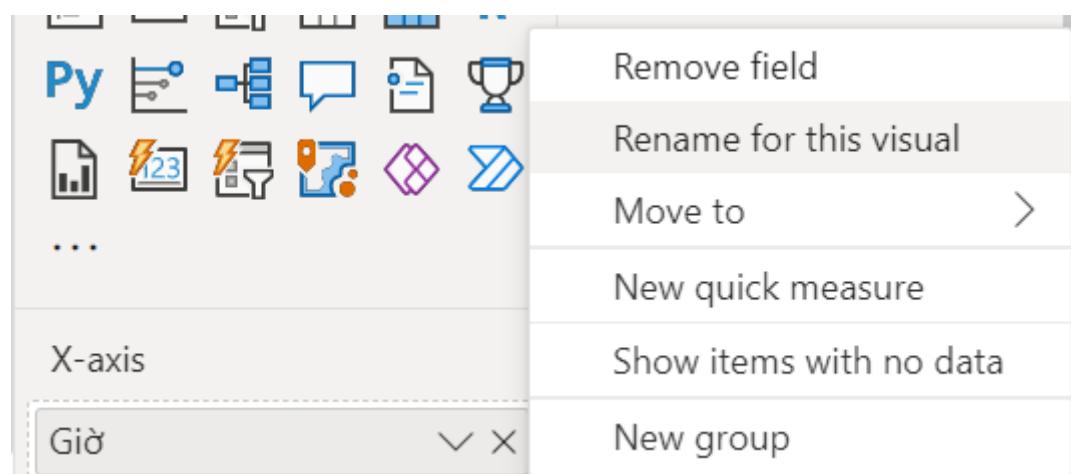
X-axis: hour

Column y-axis: dùng hàm sum đối với amount

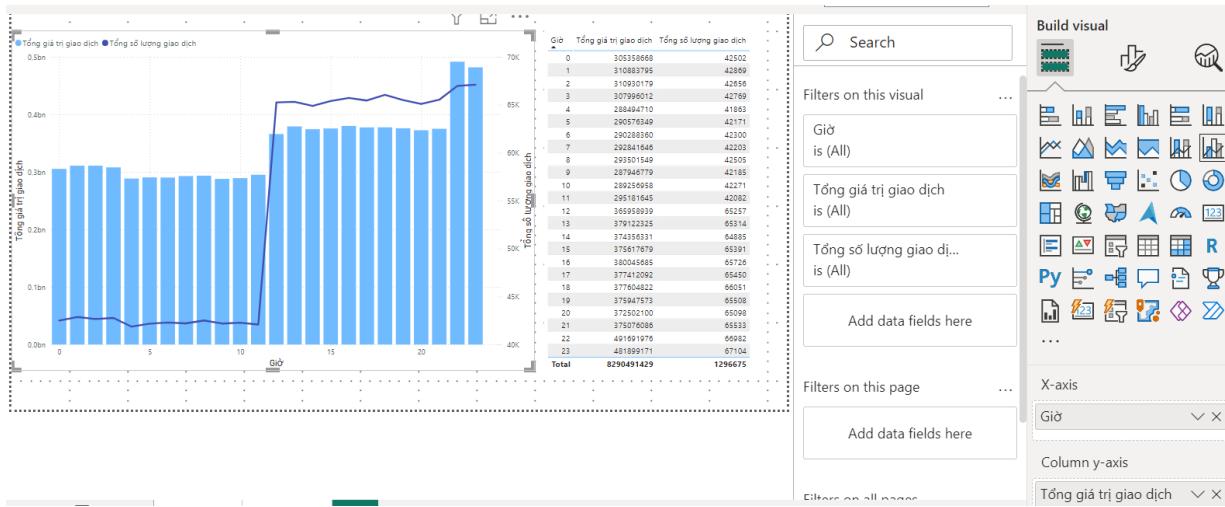
Line y-axis: dùng hàm count đối với transaction\_id



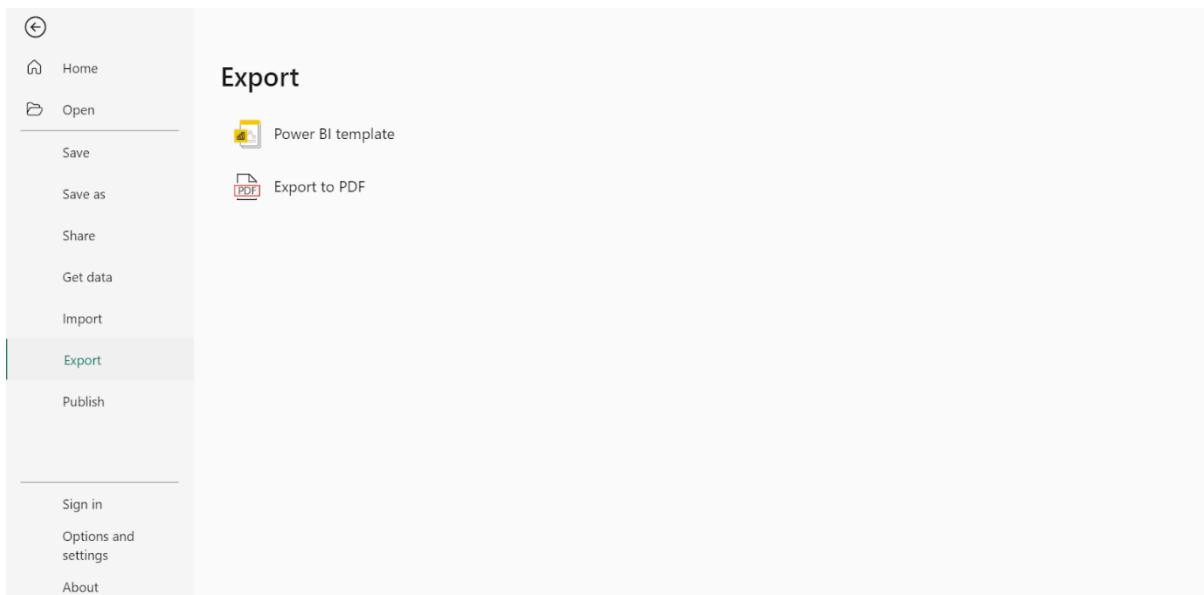
Bước 3: Có thể đổi tên các column bằng cách nhấn chuột phải vào cột và chọn lệnh Rename



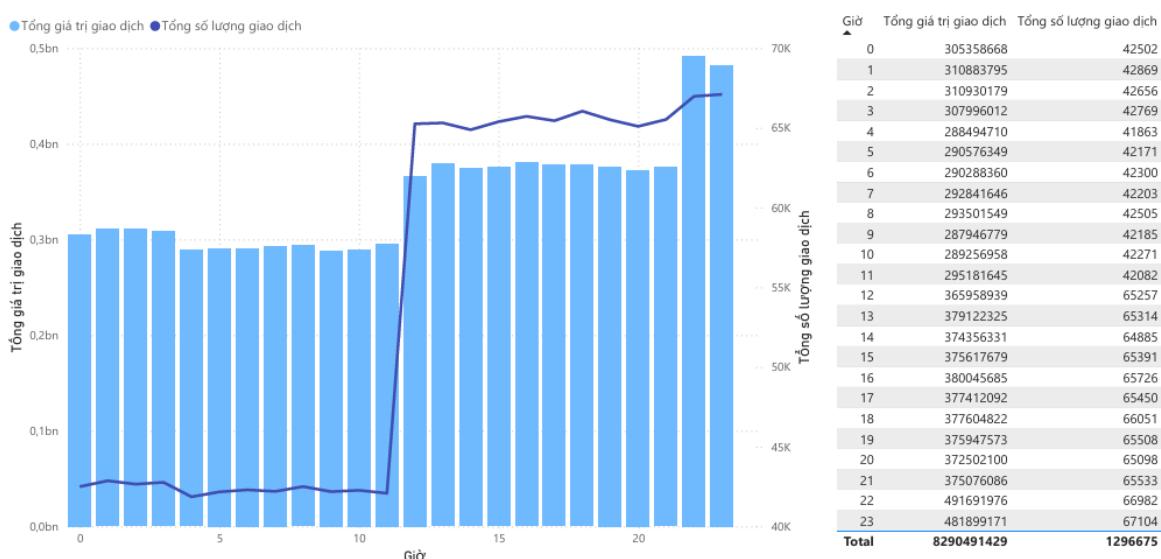
Bước 4: Có thể trình bày thêm báo cáo dưới dạng Table để có thể thấy số liệu rõ ràng hơn.



Bước 5: Xuất báo cáo dưới dạng PDF. Chọn File → Export → Export to PDF.



Báo cáo dưới dạng PDF:



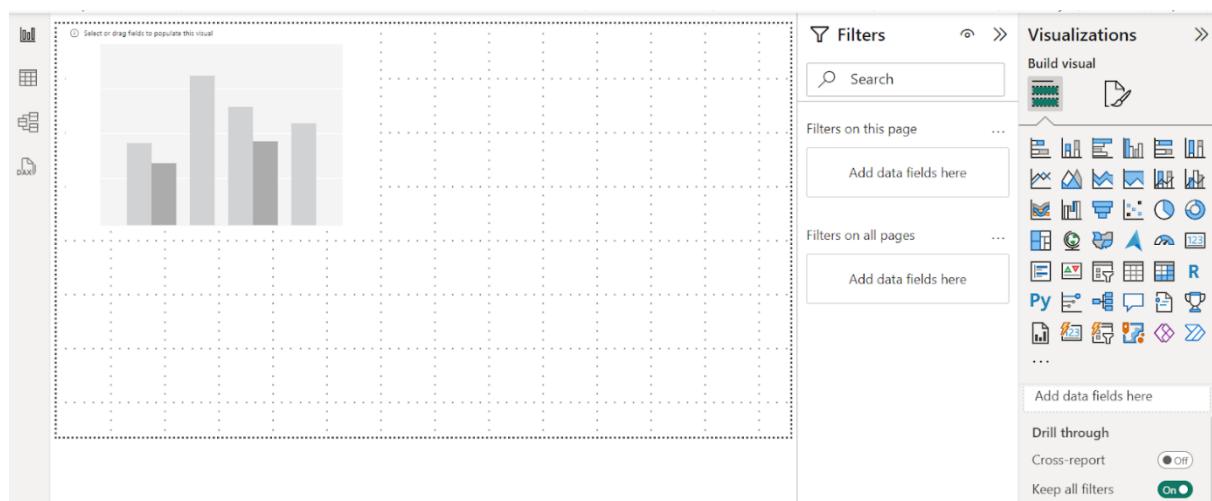
Nhận xét:

- Từ 0 giờ tới 11 giờ, tổng số lượng giao dịch dao động trong khoảng 42 000, với tổng số tiền giao dịch từ 288 495 000 tới 310 883 700 dollar.
- Từ 12 giờ đến 23h, tổng số giao dịch có xu hướng tăng lên đến hơn 65 000 và nhất là 67 104 giao dịch trong một giờ. Theo đó, tổng tiền giao dịch trong giờ cũng tăng lên trong khoảng 365 950 000 cho tới 491 691 000 dollar.
- Điểm chung của khoảng thời gian từ 0 giờ đến 11 giờ và 12 giờ đến 23 giờ là sự ổn định cả về số lượng giao dịch và tổng giá trị, khi không có điểm thời gian nào xảy ra những biến động bất ngờ. Tuy nhiên có thể thấy rõ, khoảng thời gian sau 12 giờ tập trung các giao dịch nhiều hơn.

#### 4.3.3 Báo cáo 2

Nội dung: Top 5 nghề nghiệp có tổng số giá trị giao dịch lớn nhất trong năm 2020

Bước 1: Chọn biểu đồ Clustered column chart



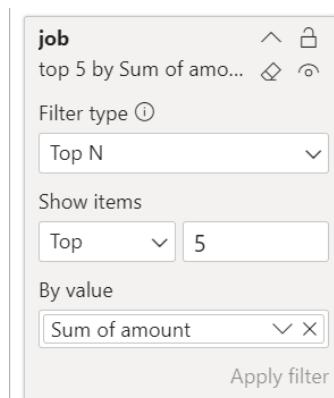
Bước 2: Chọn từ dữ liệu từ các thuộc tính: job, amount, year

- Dùng hàm sum đối với amount

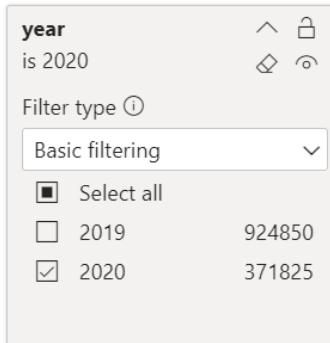


Bước 3: Chọn điều kiện lọc

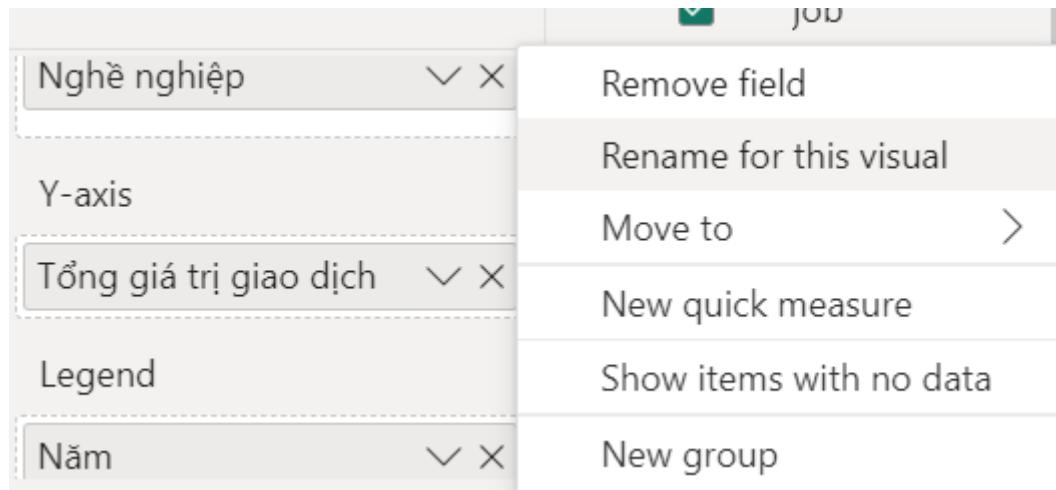
- Với job, Filter type chọn Top N, show items chọn Top 5 và set By value là Tổng giá trị giao dịch



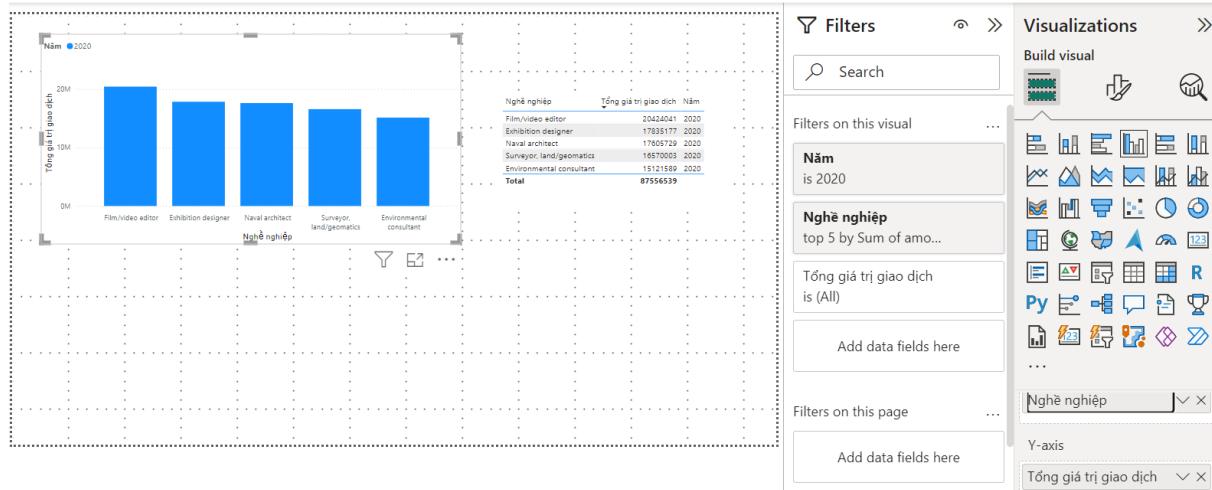
- Với year, tại Filter type chọn Basic filtering và tick 2020



Bước 4: Có thể đổi tên các column bằng cách nhấn chuột phải vào cột và chọn lệnh Rename

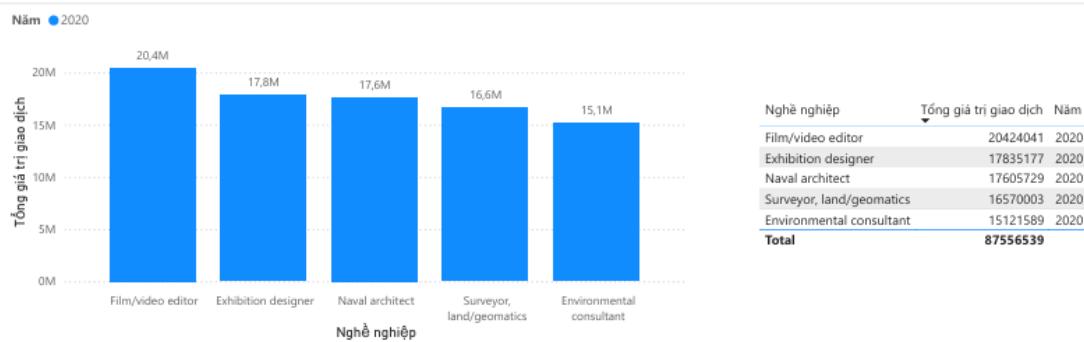


Bước 5: Có thể trình bày thêm báo cáo dưới dạng Table để có thể thấy số liệu rõ ràng hơn.



Bước 6: Xuất báo cáo dưới dạng PDF. Chọn File → Export → Export to PDF.

## Báo cáo dưới dạng PDF:



## Nhận xét:

Không quá rõ ràng để xét điểm chung giữa các ngành thuộc top 5 ngành nghề có tổng số lượng giao dịch cao nhất. Tuy vậy có thể thấy, các ngành này đều là những ngành có tính chuyên môn cao, áp dụng nhiều kỹ thuật công nghệ trong quá trình làm việc và những dự án của các ngành nghề này cũng thường yêu cầu chi phí cao.

### 4.3.4 Báo cáo 3

Nội dung: Tỷ lệ số lượng giao dịch ở các mục gas\_transport, home và pet\_kids của nam và nữ

#### Bước 1: Chọn biểu đồ Stacked column chart

The screenshot shows the Power BI desktop interface. A stacked column chart is selected on the left. The ribbon at the top includes tabs for Clipboard, Data, Queries, Insert, Calculations, Sensitivity, Share, and Copilot. The 'Insert' tab is active, showing the 'Visualizations' pane on the right. The 'Stacked column chart' icon is highlighted. Below the chart, there are three sections: 'Filters on this visual', 'Filters on this page', and 'Filters on all pages', each with a 'Add data fields here' button. At the bottom right of the visualization area, there are buttons for 'Y-axis' and 'Legend'.

#### Bước 2: Chọn từ dữ liệu từ các thuộc tính:

X-axis: gender

Y-axis: Dùng hàm count cho transaction\_id

Legend: category

A screenshot of the Power BI settings for the stacked column chart. It shows three dropdown menus: 'X-axis' set to 'gender', 'Y-axis' set to 'Count of transaction\_id', and 'Legend' set to 'category'. Each menu has a '✓' and 'X' button for confirmation and cancellation.

#### Bước 3: Chọn điều kiện lọc

– Với category, tại Filter type chọn Basic filtering và tick personal\_pos, home và pet\_kids

**Category**  
is gas\_transport, kids\_...

Filter type ⓘ

Basic filtering

Search

- Select all
- entertainment 94014
- food\_dining 91461
- gas\_transport 131659
- grocery\_net 45452
- grocery\_pos 123638
- health\_fitness 85879

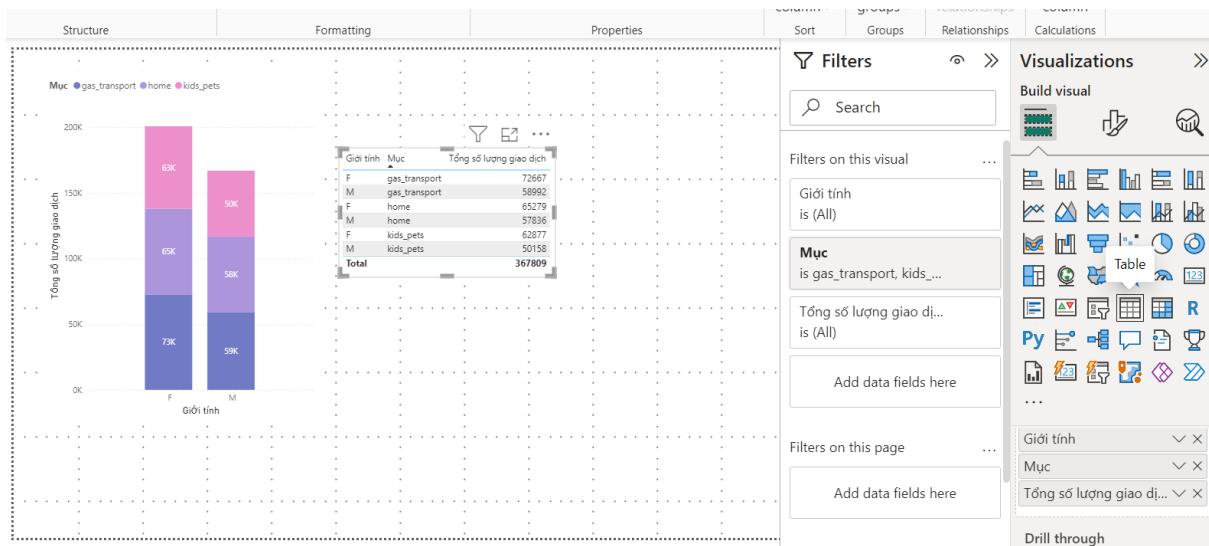
Require single selection

Bước 4: Có thể đổi tên các column bằng cách nhấn chuột phải vào cột và chọn lệnh Rename

The screenshot shows a Power BI interface with a context menu open over a column titled "Tổng số lượng giao dịch". The menu options are:

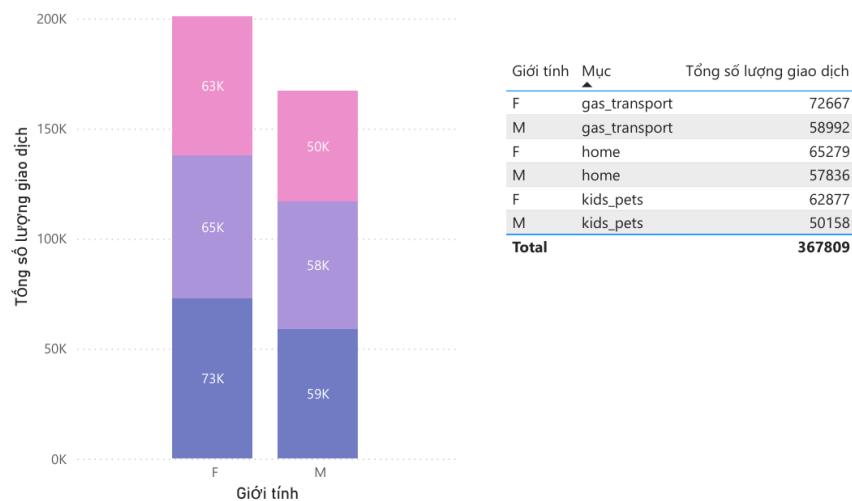
- Remove field
- Rename for this visual
- Move to >
- New quick measure
- Show items with no data

Bước 5: Có thể trình bày thêm báo cáo dưới dạng Table để có thể thấy số liệu rõ ràng hơn.



Bước 6: Xuất báo cáo dưới dạng PDF. Chọn File → Export → Export to PDF.

Mục ●gas\_transport ●home ●kids\_pets



Nhận xét:

- Các danh mục gas\_transport, home và pet\_kids đều là những mặt hàng trong gia đình.
- Theo biểu đồ, có thể thấy nữ giới có số lượng giao dịch lớn hơn nhiều so với nam giới ở cả 3 danh mục.
- Điều này phản ánh việc phụ nữ thường quan tâm và chi tiêu nhiều cho các mặt hàng liên quan đến gia đình nhiều hơn nam giới, hoặc có thể do xu hướng nữ giới thường là người quản lý chi tiêu trong gia đình nhiều hơn.

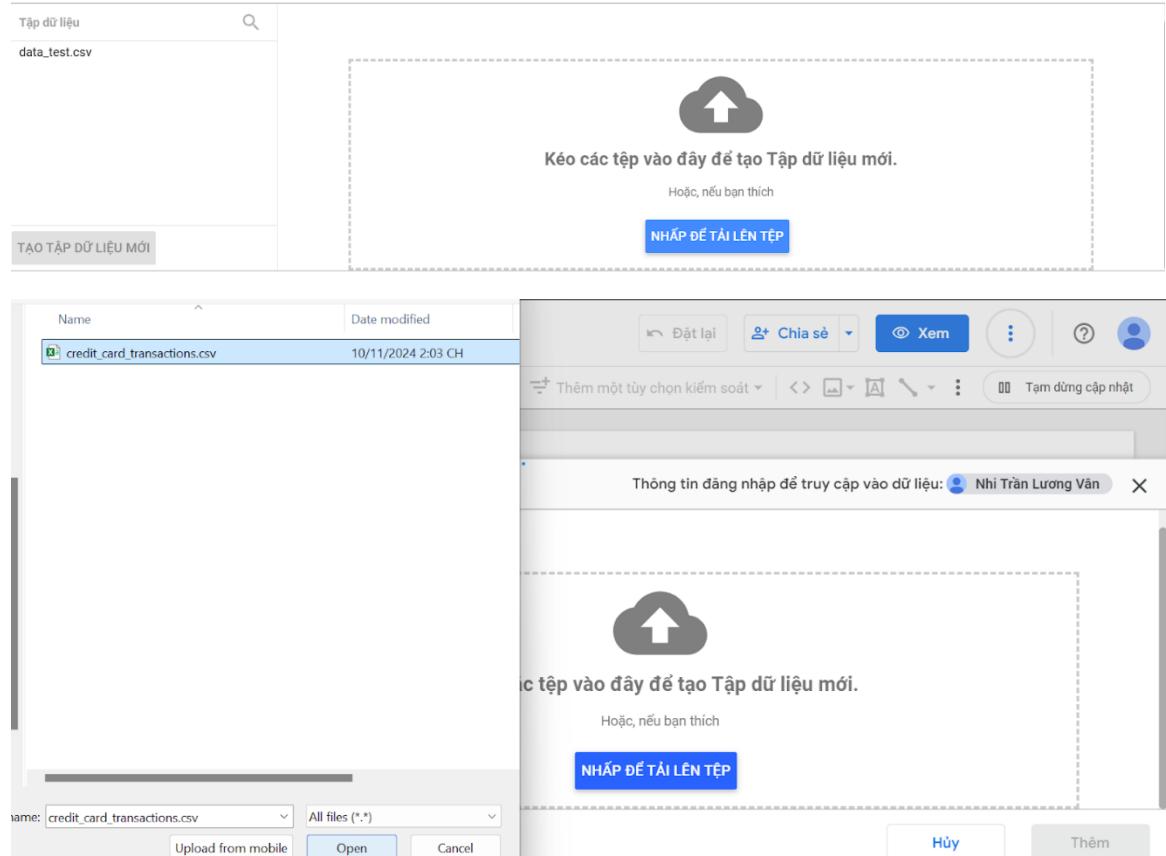
## 4.4 Thực hiện tạo báo cáo bằng Looker Studio

### 4.3.1 Quá trình upload dữ liệu vào Looker Studio

Bước 1: Tại giao diện của Looker Studio, nhấn vào Báo cáo trống

Bước 2: Nhấn chọn Tải lên

Bước 3: Chọn NHẤP ĐỂ TẢI TỆP LÊN, và chọn file .csv trong máy (lưu ý Looker không cho tải file nặng hơn 100MB)



Bước 4: Nhấn Thêm để hoàn thành bước thêm dữ liệu

This screenshot shows the 'Thêm Tệp' (Add File) dialog. At the top, it displays the file name 'credit\_card\_transactions.csv'. Below this are four status fields: 'TỔNG KÍCH THƯỚC TỆP' (Total file size) showing '7 MB (Đã sử dụng 8% trong số 100MB)', 'SỐ LƯỢNG TỆP' (Number of files) showing '1', 'NGÀY TẠO' (Created on) showing '11/17/24 9:47 PM', and 'NGÀY SỬA ĐỔI GẦN NHẤT' (Last modified on) showing '11/17/24 9:48 PM'. A 'XEM TỆP TRONG ĐÁM MÂY' (View file in the cloud) link is located below these fields. The main body of the dialog has a 'THÊM TỆP' (Add file) button, a note about file similarity, and a table with columns 'Tên tệp' (File name), 'Được tải lên lúc' (Uploaded at), 'Kích thước' (Size), and 'Trạng thái' (Status). The 'Hủy' (Cancel) and 'Thêm' (Add) buttons are at the bottom right. The entire dialog is set against a background of the Looker Studio interface.

Giao diện làm việc của Looker Studio

Báo cáo không có tiêu đề

Tệp Chính sửa Xem Chèn Trang Sắp xếp Tài nguyên Trợ giúp

Đặt lại Chia sẻ Xem

Thêm trang Thêm dữ liệu Thêm biểu đồ Thêm một tùy chọn kiểm soát Tạm dừng cập nhật

+ Thêm bộ lọc nhanh

Bắt đầu

Kéo một trường từ Bảng dữ liệu vào canvas để thêm một biểu đồ mới hoặc chọn một thành phần trên canvas báo cáo để chỉnh sửa thành phần đó.

Dữ liệu

Tim kiếm credit\_card\_transactions.csv

123 amount  
123 cardholder\_name  
123 category  
123 city  
123 day  
123 distance  
123 dob  
123 gender  
123 hour  
123 is\_fraud

+ Thêm trường

+ Thêm thông số

+ Thêm dữ liệu

#### 4.3.2 Báo cáo 1

Nội dung: Top 10 thành phố có giao dịch gian lận nhiều nhất, xếp theo thứ tự giảm dần

Bước 1: Chọn Thêm biểu đồ và chọn biểu đồ cột

Báo cáo không có tiêu đề

Tệp Chính sửa Xem Chèn Trang Sắp xếp Tài nguyên Trợ giúp

Đặt lại Chia sẻ Xem

Thêm trang Thêm dữ liệu Thêm biểu đồ Thêm một tùy chọn kiểm soát Tạm dừng cập nhật

+ Thêm bộ lọc nhanh

Bảng

Thẻ điểm

Total 1,168 Sessions 69.3K

Chuỗi thời gian

Biểu đồ thanh

Biểu đồ hình tròn

Google Maps

Biểu đồ địa lý

Bắt đầu

Rút một trường từ Bảng dữ liệu vào để thêm một biểu đồ mới hoặc chọn một thành phần trên báo cáo để chỉnh sửa thành phần đó.

Dữ liệu

Tim kiếm credit\_card\_transactions.csv

123 amount  
123 cardholder\_name  
123 category  
123 city  
123 day  
123 distance  
123 dob  
123 gender  
123 hour  
123 is\_fraud

+ Thêm trường

+ Thêm thông số

+ Thêm dữ liệu

Bước 2: Lựa chọn thuộc tính dùng để tính toán

**Biểu đồ**

**THIẾT LẬP**

**KIỂU**

Phương diện

ABC state

Xem chi tiết

Phương diện phân tích

123 is\_fraud

Chỉ số

CTD transaction\_id

Chỉ số tùy chọn

Thanh trượt chỉ số

Dữ liệu

credit\_card\_transactions.csv

- 123 amount
- ABC cardholder\_name
- ABC category
- city
- 123 day
- 123 distance
- dob
- gender
- hour
- 123 is\_fraud
- + Thêm trường
- + Thêm thông số
- + Thêm dữ liệu

### Bước 3: Tạo thêm bộ lọc và lựa chọn sử dụng

Tạo bộ lọc

Tên  
fraud

File  
credit\_card\_transactions.csv

Bao gồm  
123 is\_fraud

Bằng (=)  
1

HOẶC

VÀ

Bộ lọc này có 1 mệnh đề

LƯU

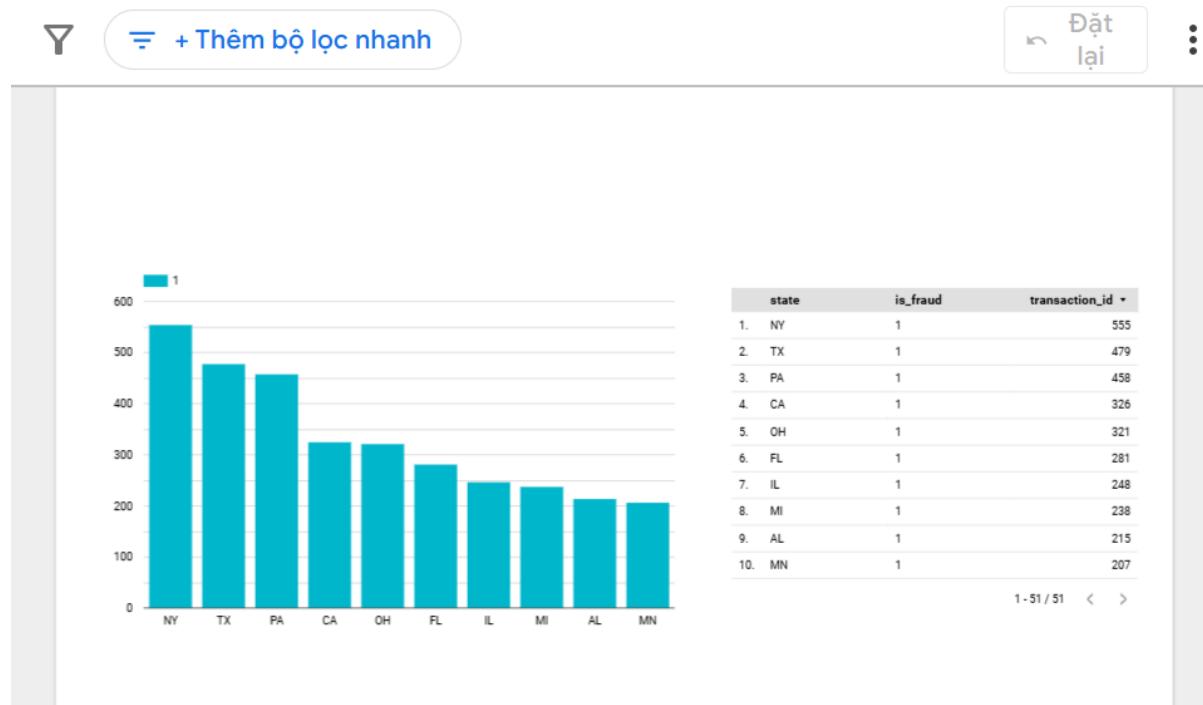
**Bộ lọc**

**Bộ Lọc Trên Chart Này**

fraud

+ THÊM BỘ LỌC

Bước 4: Có thể trình bày thêm báo cáo dưới dạng Table để có thể thấy số liệu rõ ràng hơn.



Bước 5: Xuất báo cáo dưới dạng PDF. Chọn Tệp→ Tải xuống dưới dạng→ PDF.

The screenshot shows a reporting interface. On the left, there is a sidebar with the following options:

- Chỉnh sửa
- Xem
- Chèn
- Trang
- Sắp xếp
- Tài nguyên
- Trợ giúp

Below these are some specific items:

- Chia sẻ...
- Giao diện và bố cục
- Cài đặt báo cáo
- Danh sách phiên bản
- Cài đặt đăng
- Báo cáo mới
- Tạo một bản sao...
- Tải xuống dưới dạng
- Nhúng báo cáo

In the center, there is a canvas area containing a bar chart and a table. The bar chart has the same data as in Step 4. The table is identical to the one in Step 4.

A modal window titled "PDF" is open in the center, indicating that the report is being generated in PDF format.

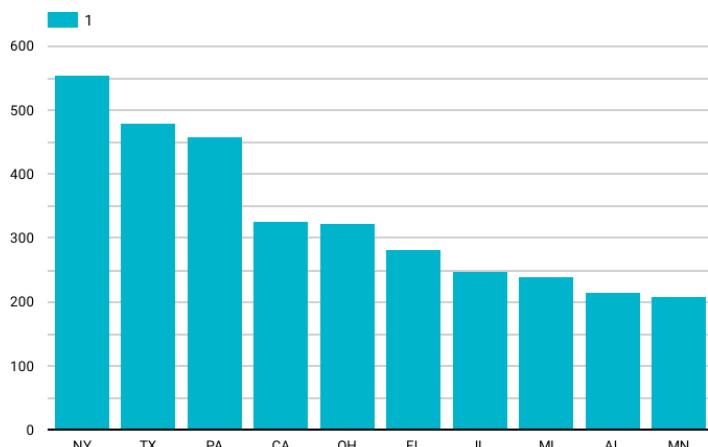
On the right, there is a section titled "Bắt đầu" with the following instructions:

Kéo một trường từ Bảng dữ liệu vào canvas để thêm một biểu đồ mới hoặc chọn một thành phần trên canvas báo cáo để chỉnh sửa thành phần đó.

Bước 6: Lựa chọn theo nhu cầu cá nhân (nếu có) và nhấn Tải xuống

The screenshot shows a data visualization interface. On the left, there is a bar chart with teal bars representing values for different states: NY, TX, PA, CA, OH, FL, IL, MI, AL, and MN. The y-axis ranges from 0 to 600. On the right, a modal dialog box titled "Tải xuống dưới dạng PDF" (Download as PDF) is displayed. It contains options for selecting pages (All pages or Selected pages), checkboxes for additional features like "Print background colors", "Add a link to the report", and "Report is password protected", and buttons for "HỦY" (Cancel) and "TẢI XUỐNG" (Download).

Báo cáo dưới dạng PDF cáo dưới dạng PDF



state	is_fraud	transaction_id
1. NY	1	555
2. TX	1	479
3. PA	1	458
4. CA	1	326
5. OH	1	321
6. FL	1	281
7. IL	1	248
8. MI	1	238
9. AL	1	215
10. MN	1	207

Nhận xét:

- Số lượng giao dịch gian lận của New York, Texas và Pennsylvania hơn hẳn so với các bang còn lại. Đây là một hiện tượng đáng chú ý, vì các bang này đều là trung tâm kinh tế lớn với quy mô dân số rất đông.

### 4.3.3 Báo cáo 2

Nội dung: Phân bố số lượng giao dịch của từng category

Bước 1: Chọn Thêm biểu đồ và chọn biểu đồ tròn

Bước 2: Lựa chọn thuộc tính và chỉ số dùng để tính toán

Bước 3: Có thể trình bày thêm báo cáo dưới dạng Table để có thể thấy số liệu rõ ràng hơn

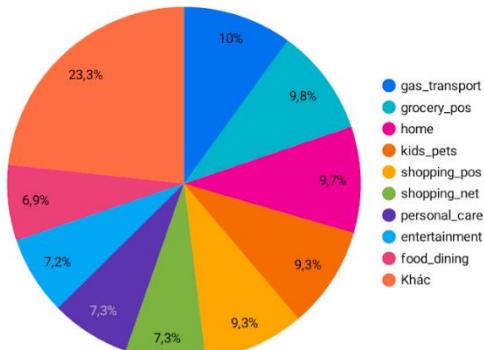
category	transaction count
1. gas_transport	4.992
2. grocery_pos	4.893
3. home	4.868
4. kids_pets	4.647
5. shopping_pos	4.626
6. shopping_net	3.644
7. personal_care	3.634
8. entertainment	3.594
9. food_dining	3.454
10. misc_pos	3.144

Bước 4: Xuất báo cáo dưới dạng PDF. Chọn Tệp→ Tải xuống dưới dạng→ PDF.

## Bước 5: Lựa chọn theo nhu cầu cá nhân (nếu có) và nhấn Tải xuống

The screenshot shows the Looker interface with a pie chart on the left. A modal window titled "Tải xuống dưới dạng PDF" (Download as PDF) is open in the center. It contains options for selecting all pages or specific pages, and checkboxes for customizing the report. At the bottom right of the modal is a blue "TẢI XUỐNG" (Download) button. To the right of the modal is a sidebar titled "Dữ liệu" (Data) which lists various CSV files and their columns.

Báo cáo dưới dạng PDF cáo dưới dạng PDF



category	transaction_id
1. gas_transport	4.992
2. grocery_pos	4.895
3. home	4.868
4. kids_pets	4.647
5. shopping_pos	4.626
6. shopping_net	3.644
7. personal_care	3.634
8. entertainment	3.594
9. food_dining	3.454
10. misc_pos	3.144

### 4.3.4 Báo cáo 3

Nội dung: Tổng giá trị giao dịch theo tháng của các nghề nghiệp có liên quan đến Engineer và IT

Bước 1: Chọn Thêm biểu đồ và chọn biểu đồ đường

The screenshot shows the Looker interface with a chart selection menu open. The menu includes options for Google Maps, various chart types like map, line, area, scatter, bar, bullet, and summary, and a total table. On the right side, there is a sidebar titled "Dữ liệu" (Data) listing CSV files and their columns.

Bước 2: Lựa chọn thuộc tính và chỉ số dùng để tính toán

THIẾT LẬP	KIỂU
Phương diện phạm vi ngày	
dob (Ngày)	
Phương diện	
month	
Xem chi tiết	<input checked="" type="checkbox"/>
Phương diện phân tích	
job	
Chỉ số	
SUM amount	

Bước 3: Chính sửa thêm thuộc tính Sắp xếp

Sắp xếp	
month	
<input type="radio"/> Giảm dần	
<input checked="" type="radio"/> Tăng dần	
Sắp xếp phụ	
SUM amount	
<input checked="" type="radio"/> Giảm dần	
<input type="radio"/> Tăng dần	

#### Bước 4: Tạo thêm bộ lọc và lựa chọn sử dụng

Tạo bộ lọc

... X ĐÓNG

Bao gồm  Chứa  HOẶC

Bao gồm  Chứa  HOẶC

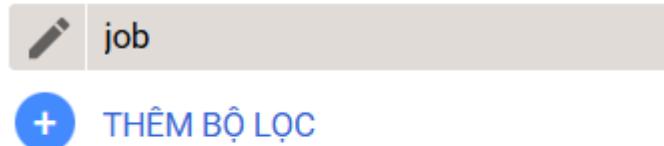
VÀ

Bộ lọc này có 2 mènh đề

LƯU

#### Bộ lọc

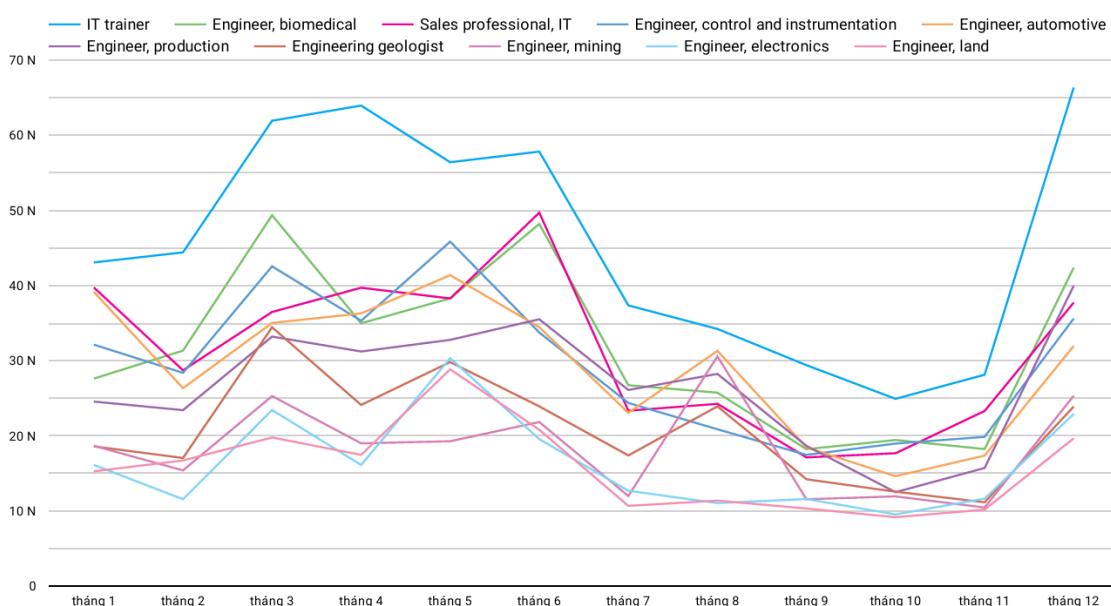
##### Bộ Lọc Trên Chart Nay



Bước 5: Xuất báo cáo dưới dạng PDF. Chọn Tệp → Tải xuống dưới dạng → PDF.

Bước 6: Lựa chọn theo nhu cầu cá nhân (nếu có) và nhấn Tải xuống

Báo cáo dưới dạng PDF cáo dưới dạng PDF



Nhận xét:

- Tổng giá trị giao dịch của các nghề đều có xu hướng tăng cao vào tháng 3 và tháng 12.
- Khoảng thời gian tháng 9 đến tháng 11 có xu hướng giao dịch thấp nhất.
- Nổi bật về tổng giá trị giao dịch so với các nghề là IT trainer, với mức giao dịch có thể đạt tới 65 000
- Những nghề nghiệp kỹ sư như Engineer Production, Engineer Electronic, Engineer Automotive... có tổng số giao dịch tương đối ổn định nhưng cũng không quá nổi bật.

## PHẦN 5: QUÁ TRÌNH KHAI PHÁ DỮ LIỆU (DATA MINING)

### 5.1 Khám phá dữ liệu và tiền xử lý dữ liệu

Bước 1: Nhập dữ liệu và kiểm tra các thuộc tính

```
columns_name = ['transaction_id', 'merchant_name', 'cardholder_name', 'job', 'gender', 'dob', 'city', 'category',
                 'state', 'hour', 'day', 'month', 'quarter', 'year', 'amount', 'distance', 'is_fraud']

df = pd.read_csv('/home/hien2706/school/nam3_hk1/IS217/data_mining/credit_card_transactions.csv', names=columns_name)

df.info()
✓ 2.0s

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1296675 entries, 0 to 1296674
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   transaction_id  1296675 non-null  int64  
 1   merchant_name   1296675 non-null  object  
 2   cardholder_name 1296675 non-null  object  
 3   job              1296675 non-null  object  
 4   gender            1296675 non-null  object  
 5   dob               1296675 non-null  object  
 6   city              1296675 non-null  object  
 7   category          1296675 non-null  object  
 8   state              1296675 non-null  object  
 9   hour              1296675 non-null  int64  
 10  day               1296675 non-null  int64  
 11  month             1296675 non-null  int64  
 12  quarter           1296675 non-null  int64  
 13  year              1296675 non-null  int64  
 14  amount             1296675 non-null  float64 
 15  distance           1296675 non-null  float64 
 16  is_fraud          1296675 non-null  int64  
dtypes: float64(2), int64(7), object(8)
memory usage: 168.2+ MB
```

Bước 2: Chuyển kiểu dữ liệu của thuộc tính dob từ object thành datetime cho phù hợp với dữ liệu ngày, tháng, năm

```
df['dob'] = pd.to_datetime(df['dob'])

✓ 0.0s
```

Bước 3: Bỏ các cột không sử dụng.

- Cột carholder\_name: đây là dữ liệu định danh cá nhân, là thông tin nhạy cảm, và dễ overfit mô hình
- Cột transaction\_id: đây là khóa, không có ích cho việc huấn luyện mô hình

```
df.drop(['cardholder_name', 'transaction_id'], axis=1, inplace=True)
```

```
✓ 0.0s
```

Bước 4: Khám phá các thuộc tính số và phân loại trong bộ dữ liệu

```

def visualize_and_correlate(df):
    # Thiết lập giao diện đẹp cho biểu đồ
    sns.set(style='whitegrid')

    print('Khám phá dữ liệu các biến số')
    # Trực quan hóa các thuộc tính số
    numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
    for col in numeric_cols:
        distinct_values = df[col].nunique()
        if distinct_values < 7:
            # Nếu có ít hơn 7 giá trị, sử dụng barplot nằm ngang
            plt.figure(figsize=(10, 5))
            sns.countplot(y=df[col], order=df[col].value_counts().index)
            plt.title(f'Biểu đồ tần suất của cột {col}')
            plt.xlabel('Số lượng') # Đổi tên trục x thành 'Số lượng'
            plt.show()
        else:
            # Nếu nhiều giá trị hơn, sử dụng biểu đồ hộp
            plt.figure(figsize=(10, 5))
            sns.boxplot(x=df[col])
            plt.title(f'Biểu đồ phân phối theo dạng hộp của cột {col}')
            plt.show()

    # Tạo và trực quan hóa ma trận hệ số tương quan
    correlation_matrix = df[numeric_cols].corr()
    plt.figure(figsize=(10, 8))
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
    plt.title('Ma trận hệ số tương quan')
    plt.show()

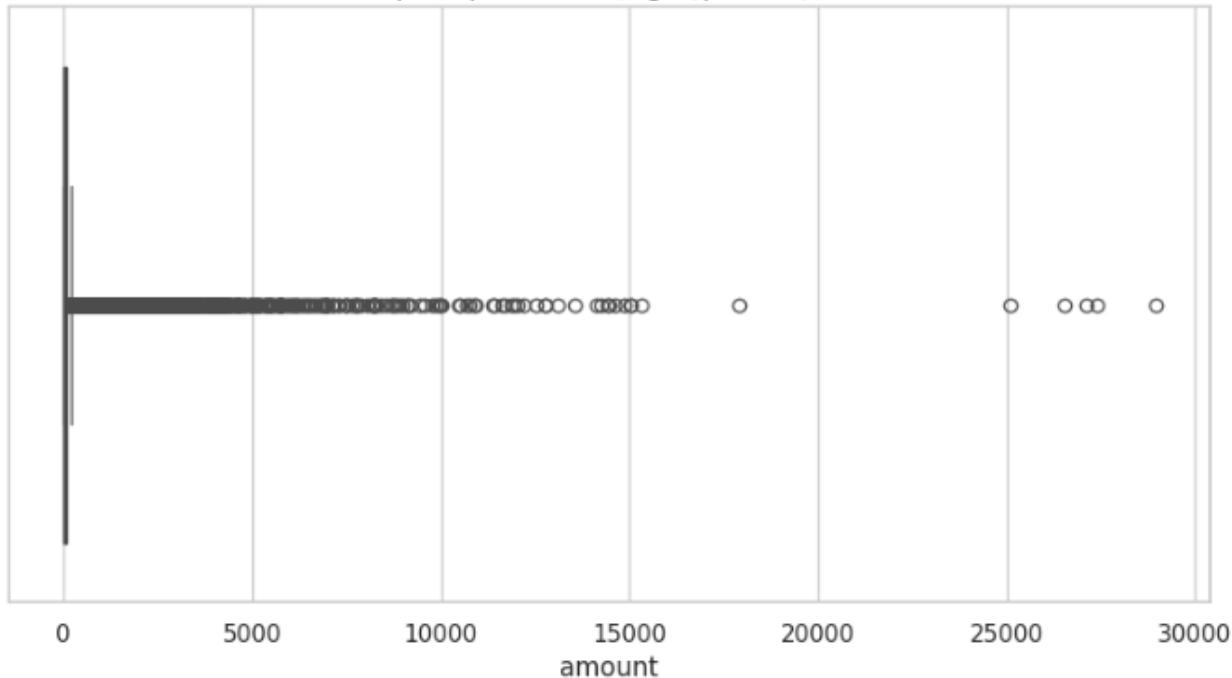
    print('Khám phá dữ liệu các biến phân loại')
    # Trực quan hóa các thuộc tính phân loại
    categorical_cols = df.select_dtypes(include=['object']).columns
    for col in categorical_cols:
        distinct_values = df[col].nunique()
        if distinct_values > 20:
            # Hiển thị top 20 giá trị phổ biến nhất bằng barplot nằm ngang
            top_20 = df[col].value_counts().nlargest(20)
            plt.figure(figsize=(10, 5))
            sns.barplot(y=top_20.index, x=top_20.values, orient='h')
            plt.title(f'Biểu đồ tần suất của cột {col} (Top 20)')
            plt.xlabel('Số lượng') # Đổi tên trục x thành 'Số lượng'
            plt.show()
        else:
            # Hiển thị tất cả các giá trị bằng barplot nằm ngang
            plt.figure(figsize=(10, 5))
            sns.countplot(y=df[col], order=df[col].value_counts().index)
            plt.title(f'Biểu đồ tần suất của cột {col}')
            plt.xlabel('Số lượng') # Đổi tên trục x thành 'Số lượng'
            plt.show()

```

✓ 0.0s

Nhìn chung kết quả trực quan của các cột đều hợp lý, nhưng còn tồn tại vấn đề ở cột amount và is\_fraud, và về ma trận tương quan giữa các thuộc tính.

Biểu đồ phân phối theo dạng hộp của cột amount

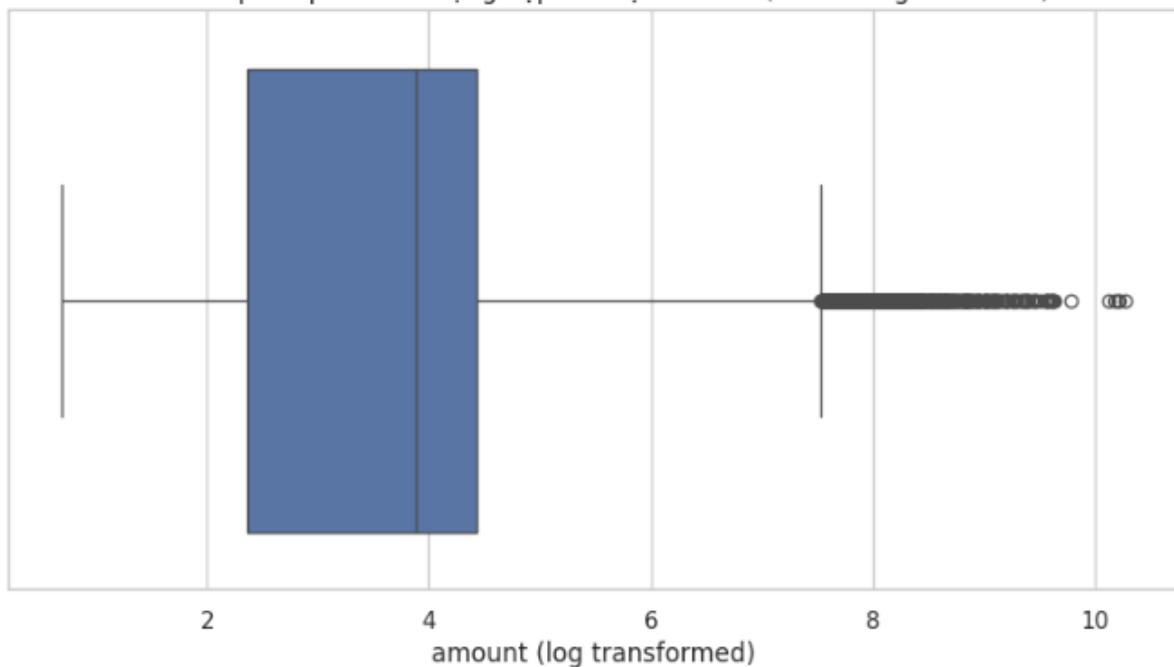


Bước 5: Cột amount có rất nhiều outlier, vì vậy tiến hành log transform cột này

```
df['amount'] = np.log1p(df['amount'])  
[✓] 0.0s
```

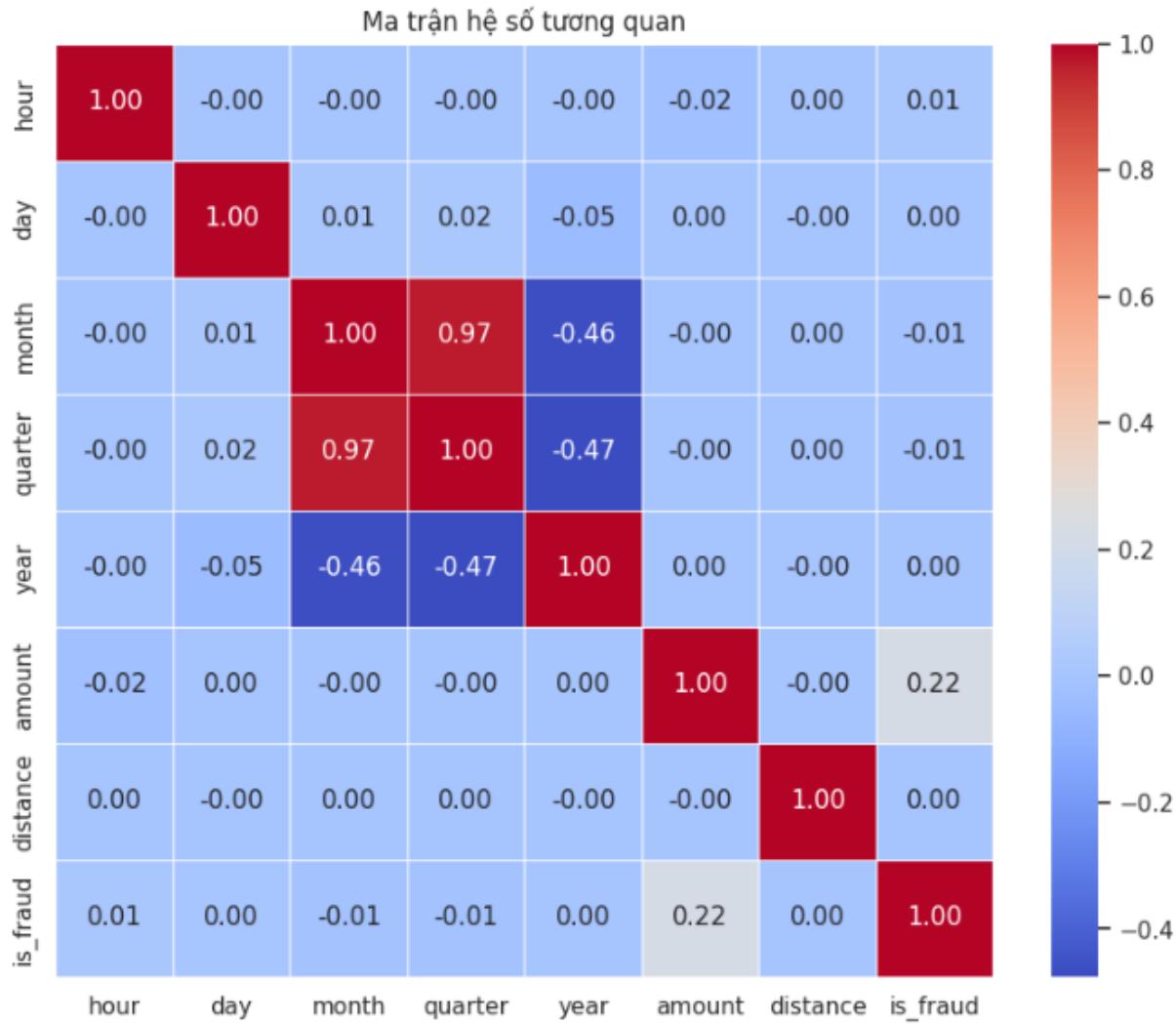
Kết quả sau khi xử lý cột amount

Biểu đồ phân phối theo dạng hộp của cột amount (sau khi log transform)

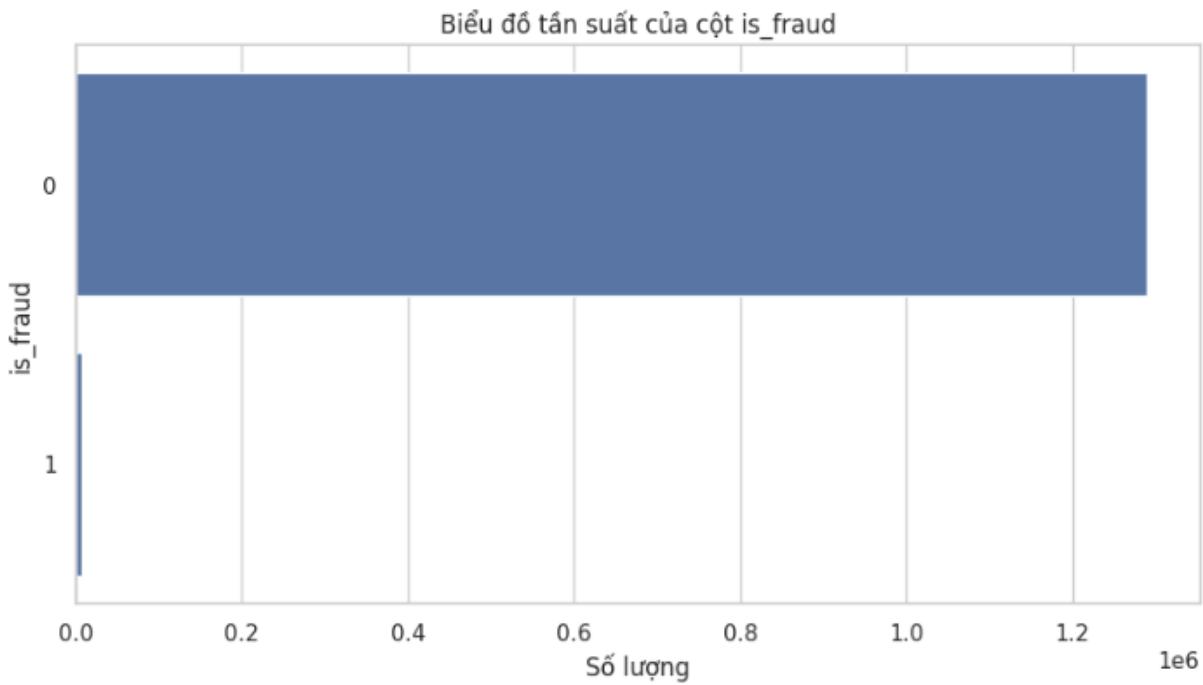


Ta có thể thấy tình trạng nhiều outlier đã được giảm đáng kể

Bước 6: Theo ma trận tương quan của các thuộc tính số, có thể thấy cột month và quarter có tương quan cao, vì vậy ta tiến hành bỏ cột quarter để làm đơn giản hóa mô hình.



Bước 7: Xử lý sự mất cân bằng nặng của biến mục tiêu is\_fraud

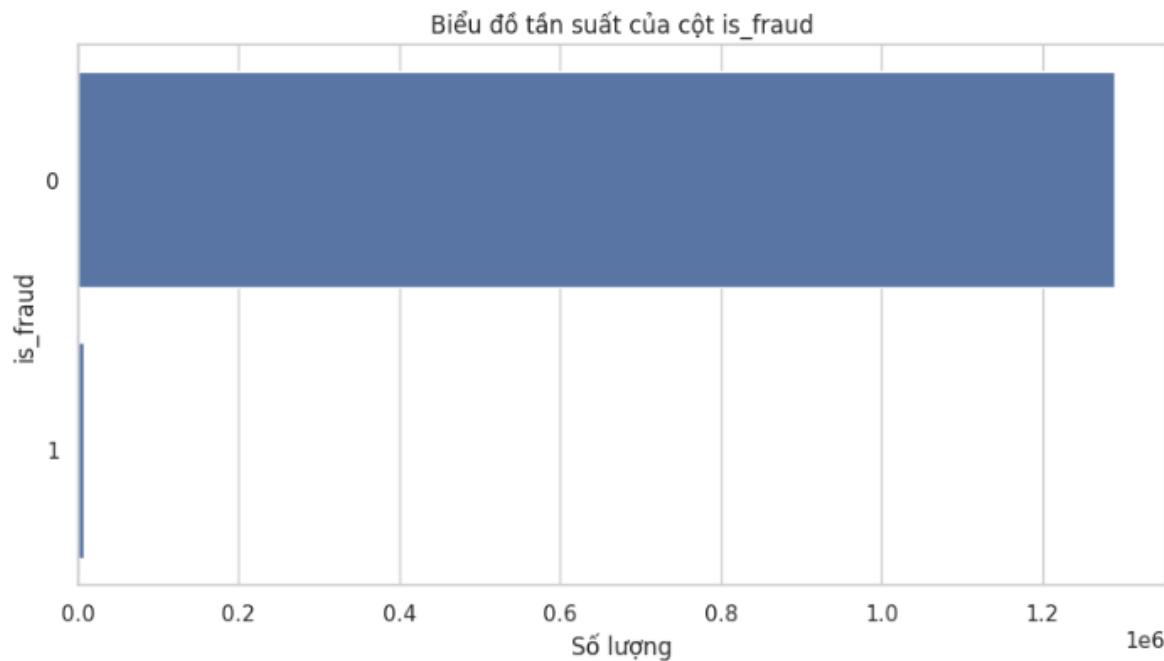


```
df['is_fraud'].value_counts()
```

✓ 0.0s

```
is_fraud
0    1289169
1      7506
Name: count, dtype: int64
```

Tỉ lệ biến 1 (có gian lận) so với biến 0 (không gian lận) là 1:172, vì vậy nhóm sẽ cân nhắc giảm thiểu dữ liệu của biến 0 cho phần sau.



Bước 8: Để xuất tạo ra các thuộc tính mới có thể giúp ích cho việc dự đoán của mô hình

- Thuộc tính age: Giúp mô hình hiểu được tác động của tuổi chủ thẻ đối với hành vi gian lận.
- Đặc trưng cuối tuần: chỉ ra liệu một giao dịch có xảy ra vào cuối tuần (thứ Bảy hoặc Chủ nhật) hay không. Các hoạt động gian lận có thể có các mẫu khác nhau vào cuối tuần so với các ngày trong tuần.
- Đặc trưng ban đêm: chỉ ra liệu một giao dịch có diễn ra vào ban đêm (ví dụ, từ 10 giờ tối đến 6 giờ sáng) hay không. Các hoạt động gian lận có thể liên quan đến các thời điểm giao dịch bất thường.
- Đặc trưng cuối tháng: để nhận diện các giao dịch diễn ra vào những ngày cuối tháng (ví dụ, sau ngày 25).

```

● # Tạo cột mới age từ cột dob
df['age'] = (pd.to_datetime('2020-06-01 00:00:00') - df['dob']).dt.days // 365
df.drop('dob', axis=1, inplace=True)
] ✓ 0.1s

# Tạo cột mới is_weekend
df['date'] = pd.to_datetime(df[['year', 'month', 'day']])
df['day_of_week'] = df['date'].dt.dayofweek # 0 = Monday, 6 = Sunday
df['is_weekend'] = df['day_of_week'].apply(lambda x: 1 if x >= 5 else 0) # 1 for Saturday, Sunday
df.drop(columns=['date', 'day_of_week'], inplace=True)
] ✓ 0.5s

# Tạo cột mới is_night từ cột hour
df['is_night'] = df['hour'].apply(lambda x: 1 if x < 6 or x > 22 else 0)
] ✓ 0.4s

df['is_end_of_month'] = df['day'].apply(lambda x: 1 if x >= 25 else 0)
] ✓ 0.4s

```

Bước 9: Tiến hành downsampling dữ liệu ở biến đa số của tập này, giảm lại còn tỉ lệ 1:50 cho tập dữ liệu đang bị mất cân bằng ở biến mục tiêu

```

df_majority = df[df['is_fraud'] == 0]
df_minority = df[df['is_fraud'] == 1]

# Downsample majority class to match a xx ratio
ratio = 50
n_samples_majority = len(df_minority) * ratio

# Downsample the majority class
df_majority_downsampled = resample(df_majority,
                                     replace=False,      # sample without replacement
                                     n_samples=n_samples_majority, # desired number of samples
                                     random_state=42) # for reproducibility

# Combine minority class with downsampled majority class
df_downsampled = pd.concat([df_minority, df_majority_downsampled])

# Shuffle the downsampled dataframe
df_downsampled = df_downsampled.sample(frac=1, random_state=42).reset_index(drop=True)

# Check the new balance
print(df_downsampled['is_fraud'].value_counts())
✓ 0.3s

is_fraud
0    375300
1     7506
Name: count, dtype: int64

```

## 5.2 Mã hóa và lựa chọn thuộc tính

### 5.2.1 Mã hóa thuộc tính

Đây là một bước quan trọng trong quy trình tiền xử dữ liệu để chuyển đổi các đặc trưng phân loại thành giá trị số mà các thuật toán học máy có thể hiểu được. Tùy thuộc vào loại và tính chất của các đặc trưng, nhóm áp dụng các kỹ thuật mã hóa khác nhau để biểu diễn dữ liệu một cách hiệu quả trong khi vẫn giữ nguyên thông tin vốn có. Mục tiêu chính là chuyển đổi các biến phân loại thành một định dạng mà mô hình có thể sử dụng, mà không tạo ra sự phức tạp không cần thiết hoặc làm giảm hiệu suất.

Frequency Encoding [4]: Kỹ thuật này thường được áp dụng cho các đặc trưng phân loại có nhiều giá trị khác nhau, ví dụ như *merchant\_name*, *job*, *city*, hoặc *state*. Frequency encoding giúp giữ lại thông tin quan trọng, đồng thời làm giảm kích thước dữ liệu, điều này cực kỳ hữu ích với những đặc trưng có độ đa dạng cao.

One-Hot Encoding [4]: Phương pháp này phù hợp với các đặc trưng phân loại có ít giá trị duy nhất, chẳng hạn như *gender* hoặc *category*. Mã hóa kiểu one-hot được sử dụng để giữ nguyên tính phân loại của các biến này mà không khiến dữ liệu trở nên "cồng kềnh".

Cyclical Encoding [5]: Đây là kỹ thuật thường được dùng cho những đặc trưng liên quan đến thời gian, như *hour* hay *day*. Vì các đặc trưng này mang tính chu kỳ (ví dụ: giờ 23 năm rất gần với giờ 0), kỹ thuật Cyclical Encoding sẽ chuyển đổi chúng thành các thành phần sin và cos để mô hình dễ xử lý hơn.

```
def frequency_encoding(df, columns):
    for column_name in columns:
        encoded_column_name = f"{column_name}_freq"
        df[encoded_column_name] = df[column_name].map(df[column_name].value_counts(normalize=True))
    df.drop(columns=[column_name], inplace=True)
    return df

def one_hot_encoding(df, columns, drop_first=True):
    df = pd.get_dummies(df, columns=columns, drop_first=drop_first)
    return df

✓ 0.0s
```

```
frequency_encoding_columns = ['merchant_name', 'job', 'city', 'state']
one_hot_encoding_columns = ['gender', 'category']
df = frequency_encoding(df, frequency_encoding_columns)
df = one_hot_encoding(df, one_hot_encoding_columns)

✓ 1.0s
```

```
df['hour_sin'] = np.sin(2 * np.pi * df['hour'] / 24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour'] / 24)

# Cyclical Encoding for 'day' (1 to 31)
df['day_sin'] = np.sin(2 * np.pi * df['day'] / 31)
df['day_cos'] = np.cos(2 * np.pi * df['day'] / 31)

# Optionally drop the original 'hour' and 'day' columns if they are no longer needed
df.drop(columns=['hour', 'day'], inplace=True)

✓ 0.1s
```

## 5.2.2 Lựa chọn thuộc tính

Nhóm đã quyết định sử dụng độ đo Information Value (IV) [6] để đánh giá xem các biến có sức mạnh dự đoán thế nào đối với biến mục tiêu. Đây là một cách khá hiệu quả để xem biến nào thực sự hữu ích trong việc dự đoán và biến nào không đáng được giữ lại.

Information Value (IV) giúp chúng ta đo lường mức độ liên quan giữa từng đặc trưng và biến mục tiêu. Ý tưởng là tính toán một giá trị phản ánh mối quan hệ giữa hai biến này, từ đó xác định được đặc trưng nào quan trọng nhất cho mô hình. Để tính IV, trước tiên chúng ta cần xác định Weight of Evidence (WoE) cho từng giá trị của đặc trưng. Sau đó, IV sẽ được tính bằng cách cộng tổng các tích giữa WoE và sự chênh lệch giữa tỷ lệ xảy ra và không xảy ra sự kiện (event) cho từng giá trị.

**Weight of Evidence (WoE)** là một khái niệm dùng để đánh giá mức độ tách biệt giữa các sự kiện "tốt" (không gian lận) và "xấu" (gian lận) dựa trên từng giá trị cụ thể của một đặc trưng. Công thức tính toán WoE được áp dụng để chuyển các đặc trưng phân loại thành giá trị số, giúp giảm bớt chiều dữ liệu trong khi vẫn giữ được mối liên hệ với biến mục tiêu.

$$WOE = \ln(\% \text{ of non-events} \div \% \text{ of events})$$

$$IV = \sum (\% \text{ of non-events} - \% \text{ of events}) * WOE$$

Bằng cách tính toán WoE, chúng ta có thể chuyển đổi các đặc trưng phân loại thành các giá trị số, duy trì mối quan hệ với biến mục tiêu đồng thời giảm bớt chiều dữ liệu.

Dựa trên giá trị IV, chúng ta có thể đánh giá khả năng dự đoán của một đặc trưng như sau:

- $IV < 0.02$ : Không có khả năng dự đoán
- $0.02 \leq IV < 0.1$ : Khả năng dự đoán yếu
- $0.1 \leq IV < 0.3$ : Khả năng dự đoán trung bình
- $0.3 \leq IV < 0.5$ : Khả năng dự đoán mạnh
- $IV \geq 0.5$ : Đáng ngờ hoặc tốt đến mức khó tin

Những đặc trưng có giá trị IV nhỏ hơn **0.02** sẽ bị loại bỏ, vì chúng không đóng góp gì cho khả năng dự đoán và có thể chỉ tạo thêm nhiễu cho mô hình.

Trước khi tính toán IV, các đặc trưng như amount và distance được chia thành các khoảng (bins) theo phân vị bằng cách sử dụng hàm pd.qcut() trong pandas. Phương pháp này tạo ra các khoảng rời rạc từ các đặc trưng liên tục bằng cách chia dữ liệu thành 10 phân vị, đảm bảo rằng mỗi khoảng chứa xấp xỉ số lượng quan sát bằng nhau. Việc phân khoảng giúp xử lý các biến liên tục một cách hiệu quả khi tính toán IV, cho phép quan sát phân phối của các sự kiện và không sự kiện trong các khoảng giá trị khác nhau của đặc trưng. Bằng cách tạo các khoảng, chúng ta giảm thiểu tác động của các giá trị ngoại lai và làm cho việc tính toán WoE trở nên ổn định và dễ hiểu hơn.

```
df_downsampled['amount_binned'] = pd.qcut(df_downsampled['amount'], q=10, duplicates='drop') # Bin 'amount' into 10 quantiles
df_downsampled['distance_binned'] = pd.qcut(df_downsampled['distance'], q=10, duplicates='drop') # Bin 'distance' into 10 quantiles
```

#### Hàm tính IV:

```
def calc_woe_iv(df, feature, target):
    lst = []
    for val in tqdm(df[feature].unique(), desc=f"Calculating WoE and IV for {feature}"):
        count = df[df[feature] == val].shape[0]
        count_event = df[(df[feature] == val) & (df[target] == 1)].shape[0]
        count_non_event = df[(df[feature] == val) & (df[target] == 0)].shape[0]

        event_rate = count_event / df[df[target] == 1].shape[0]
        non_event_rate = count_non_event / df[df[target] == 0].shape[0]

        # Avoid division by zero
        if event_rate == 0 or non_event_rate == 0:
            woe = 0
        else:
            woe = np.log(event_rate / non_event_rate)

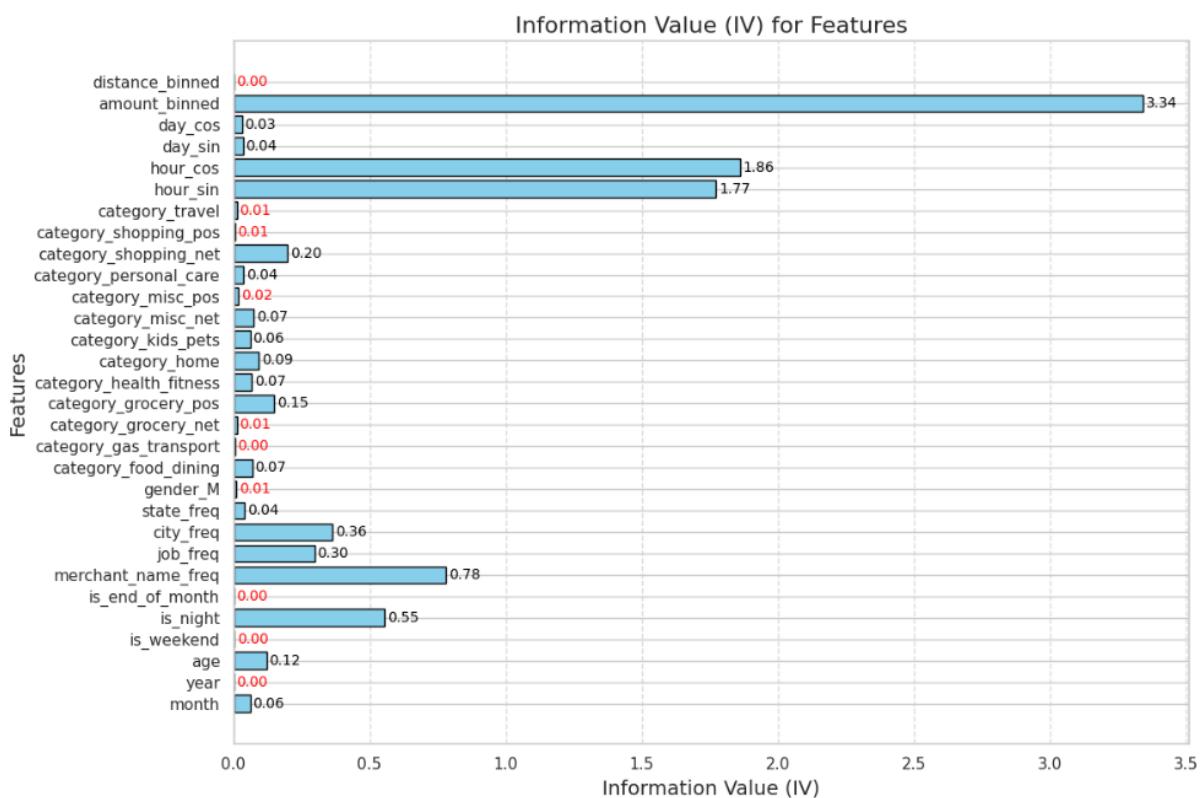
        iv = (event_rate - non_event_rate) * woe

        lst.append({'Value': val, 'WoE': woe, 'IV': iv})

    woe_iv_df = pd.DataFrame(lst)
    total_iv = woe_iv_df['IV'].sum()

    return woe_iv_df, total_iv
```

Kết quả:



Ta sẽ tiến hành loại bỏ các cột có giá trị IV < 0.02 vì chúng không có khả năng dự đoán.

## 5.3 Lựa chọn mô hình

### 5.3.1 Decision Tree

Decision Tree [2] là một thuật toán học máy khá phổ biến, được thiết kế dựa trên cấu trúc dạng cây để đưa ra quyết định. Đây là một phương pháp được ưa chuộng trong các bài toán phân loại và hồi quy nhờ tính đơn giản, dễ hiểu và khả năng xử lý tốt các tập dữ liệu phức tạp. Về cơ bản, Decision Tree hoạt động bằng cách liên tục chia nhỏ tập dữ liệu thành các nhóm nhỏ hơn, dựa trên một tiêu chí cụ thể (chẳng hạn như Gini impurity hoặc Entropy), nhằm tối ưu hóa sự đồng nhất trong từng nhóm.

#### 5.3.1.1 Các bước hoạt động của thuật toán Decision Tree:

Chọn điểm chia (Split Criteria):

Ở mỗi nút, Decision Tree chọn một đặc trưng và giá trị ngưỡng để phân chia dữ liệu sao cho giảm thiểu độ không thuần nhất (impurity). Tiêu chuẩn thường dùng gồm:

- + Gini Impurity: Đo lường mức độ không đồng nhất của một tập hợp.
- + Entropy: Đo lường mức độ không chắc chắn trong dữ liệu.

Phân nhánh dữ liệu:

Sau khi chọn tiêu chí, dữ liệu sẽ được phân chia thành hai tập con dựa trên giá trị ngưỡng đã xác định. Quá trình này được lặp lại liên tục cho đến khi đạt điều kiện dừng. Một số điều kiện dừng thường gặp:

- + Tất cả mẫu trong một nút thuộc cùng một lớp.
- + Khi đạt đến độ sâu tối đa được chỉ định.

Dự đoán:

Khi có một mẫu dữ liệu mới, Decision Tree sẽ phân nhánh mẫu đó qua các nút, dựa trên giá trị của các đặc trưng, cho đến khi đạt đến nút lá. Nút lá chứa nhãn phân loại hoặc giá trị dự đoán cuối cùng.

#### 5.3.1.2 Ưu điểm và nhược điểm của Decision Tree:

a. Ưu điểm:

- Dễ hiểu và trực quan: Decision Tree có cấu trúc dễ đọc, phù hợp với những bài toán cần sự minh bạch và giải thích rõ ràng từ mô hình.
- Khả năng xử lý dữ liệu mất cân bằng: Thuật toán có thể điều chỉnh tham số như `class_weight='balanced'` để xử lý tốt các dữ liệu bị mất cân bằng (như khi một lớp thiểu số chiếm tỷ lệ rất thấp).
- Tự động lựa chọn đặc trưng quan trọng: Mô hình sẽ tập trung vào các đặc trưng có sức mạnh dự đoán cao, giảm tác động của các đặc trưng không quan trọng.

b. Nhược điểm:

- Nguy cơ quá khớp: Nếu không kiểm soát tốt các tham số như độ sâu của cây hoặc số lượng mẫu tối thiểu tại mỗi nút, mô hình có thể bị overfitting (quá khớp với dữ liệu huấn luyện)
- Hiệu suất kém với dữ liệu phức tạp: Với các bài toán phức tạp hơn, các phương pháp hiện đại như Random Forest hoặc Gradient Boosting thường đạt hiệu quả cao hơn.

#### 5.3.1.3 Ứng dụng Decision Tree trong Fraud Detection

Trong bài toán phát hiện gian lận, Decision Tree có thể được sử dụng như một giải pháp hiệu quả để phân loại và phát hiện các giao dịch bất thường. Một số đặc điểm nổi bật của Decision Tree trong fraud detection gồm:

- Xử lý dữ liệu mất cân bằng: Với tham số `class_weight='balanced'`, mô hình có thể điều chỉnh trọng số của các lớp, giúp nhận diện tốt hơn lớp thiểu số (như các giao dịch gian lận).
- Tính minh bạch và dễ giải thích: Đây là một lợi thế lớn khi làm việc với các bài toán phát hiện gian lận, nơi việc hiểu rõ cách mô hình ra quyết định là điều rất quan trọng.
- Tập trung vào đặc trưng quan trọng: Decision Tree tự động chọn ra các đặc trưng đáng chú ý nhất ở từng bước, giảm tác động từ dữ liệu nhiễu hoặc các đặc trưng không hữu ích.
- Nhận diện dễ dàng các giao dịch bất thường: Trong các bài toán gian lận, giao dịch bất thường thường mang đặc điểm khác biệt so với các giao dịch hợp pháp. Decision Tree có thể nhận diện những giao dịch này hiệu quả nhờ cách phân chia dữ liệu dựa trên các tiêu chí cụ thể.

Decision Tree là một lựa chọn nhanh, gọn và phù hợp khi làm việc với các tập dữ liệu không quá phức tạp, hoặc trong trường hợp cần một mô hình dễ giải thích.

### 5.3.2 KNN

K-Nearest Neighbors [1] (KNN) là một thuật toán học máy không tham số và là một trong những thuật toán phân loại đơn giản nhưng rất hiệu quả. KNN thuộc nhóm các thuật toán học dựa trên trường hợp (Case-Based Learning), nơi việc phân loại của một đối tượng chưa biết được xác định dựa trên các trường hợp đã biết (lưu trữ trong bộ nhớ). Thuật toán KNN không có quá trình học trước mà chỉ thực hiện việc tìm kiếm và phân loại tại thời điểm truy vấn

Thuật toán KNN hoạt động dựa trên một ý tưởng rất đơn giản: xác định lớp của một điểm dữ liệu mới dựa trên "k" điểm dữ liệu gần nhất trong không gian đặc trưng của dữ liệu đã biết. Sự gần gũi của các điểm được xác định dựa trên một số phép đo khoảng cách hoặc độ tương tự, chẳng hạn như khoảng cách Euclid hoặc khoảng cách Manhattan.

#### 5.3.2.1 Các bước hoạt động của thuật toán KNN:

- Tính khoảng cách: Đầu tiên, thuật toán sẽ tính khoảng cách từ điểm cần phân loại đến tất cả các điểm dữ liệu trong tập huấn luyện. Một số phương pháp đo khoảng cách phổ biến gồm:

- + Khoảng cách Euclid (Euclidean Distance): Đây là cách đo chuẩn giữa hai điểm trong không gian. Nó phù hợp khi dữ liệu nằm trên một không gian liên tục.
- + Khoảng cách Manhattan (Manhattan Distance): Đây là khoảng cách tính theo trục X và trục Y, thường dùng khi di chuyển theo các trục chính có ý nghĩa, như trong không gian lưới.

- Chọn "k" hàng xóm gần nhất:

Sau khi tính toán khoảng cách, thuật toán sẽ chọn ra "k" điểm dữ liệu có khoảng cách nhỏ nhất. "k" là tham số quan trọng của KNN, cần được tối ưu để đạt hiệu suất tốt nhất. Nếu "k" quá nhỏ, mô hình dễ bị quá khớp (overfitting), còn nếu "k" quá lớn, mô hình có thể bị dưới khớp (underfitting).

- Bỏ phiếu đa số hoặc trung bình:

Sau khi chọn ra "k" hàng xóm gần nhất, thuật toán sẽ dự đoán lớp của điểm mới dựa trên lớp phổ biến nhất (đối với bài toán phân loại). Trong bài toán hồi quy, giá trị dự đoán sẽ là trung bình giá trị của các điểm hàng xóm.

### 5.3.2.2 Ưu điểm và nhược điểm của KNN:

#### a. Ưu điểm:

- Đơn giản và dễ sử dụng: KNN không yêu cầu quá trình huấn luyện phức tạp. Nó phù hợp với các bài toán mà việc giải thích đầu ra là quan trọng, vì chúng ta có thể dễ dàng thấy được những điểm dữ liệu nào ảnh hưởng đến quyết định phân loại.
- Không giả định về phân phối dữ liệu: Vì là một thuật toán phi tham số, KNN không yêu cầu các giả định cụ thể về phân phối dữ liệu.

#### b. Nhược điểm:

- Chi phí tính toán lớn: KNN yêu cầu tính toán khoảng cách từ điểm cần phân loại đến tất cả các điểm dữ liệu khác, làm cho chi phí tính toán tăng đáng kể khi dữ liệu lớn.
- Nhạy cảm với nhiễu: Nếu trong tập dữ liệu có các đặc trưng không liên quan hoặc nhiễu, KNN có thể cho kết quả không chính xác vì tất cả các đặc trưng đều đóng góp vào việc tính khoảng cách.

### 5.3.2.3 Ứng dụng trong Fraud Detection:

KNN là một thuật toán phù hợp cho bài toán phát hiện gian lận (fraud detection) nhờ khả năng dự đoán dựa trên sự tương đồng giữa các điểm dữ liệu. Trong các bài toán như vậy, mục tiêu là xác định liệu một giao dịch có khả năng gian lận hay không.

Ví dụ: Nếu một giao dịch nằm gần nhiều giao dịch được đánh dấu là gian lận trong không gian đặc trưng, thì giao dịch đó cũng có khả năng cao là gian lận. Bằng cách chọn giá trị "k" phù hợp và sử dụng phép đo khoảng cách thích hợp, KNN có thể giúp nhận diện các giao dịch bất thường dựa trên dữ liệu lịch sử.

Tuy nhiên, cần lưu ý rằng KNN hoạt động tốt nhất khi kích thước dữ liệu không quá lớn, hoặc dữ liệu đã được xử lý kỹ để loại bỏ nhiễu và các đặc trưng không cần thiết. Nếu dữ liệu quá lớn hoặc phức tạp, chi phí tính toán của KNN có thể trở thành một thách thức.

### 5.3.3 LightGBM

LightGBM [3] là một phiên bản cải tiến của mô hình Gradient Boosting Decision Tree (GBDT), nổi bật với tốc độ huấn luyện nhanh và khả năng xử lý dữ liệu lớn. Nhờ áp dụng hai kỹ thuật chính là Gradient-based One-Side Sampling (GOSS) và Exclusive Feature Bundling (EFB), LightGBM không chỉ tối ưu thời gian tính toán.

#### 5.3.3.1 Gradient Boosting Decision Tree (GBDT)

Trước khi tìm hiểu sâu về LightGBM, cần hiểu rõ hơn về GBDT, một thuật toán học máy phổ biến và cực kỳ mạnh mẽ, thường được sử dụng trong các bài toán như phân loại, hồi quy, dự đoán click, và học xếp hạng.

GBDT hoạt động bằng cách xây dựng một loạt cây quyết định, mỗi cây có nhiệm vụ cải thiện kết quả của cây trước đó bằng cách tối ưu hóa một hàm mất mát (loss function). Nhờ vào phương pháp boosting, GBDT thường đạt độ chính xác rất cao, đặc biệt trong các bài toán yêu cầu kết quả chính xác hàng đầu.

Tuy nhiên, nhược điểm lớn nhất của GBDT là hiệu suất tính toán khi xử lý các tập dữ liệu lớn. Điều này xuất phát từ việc mỗi đặc trưng phải được quét qua toàn bộ dữ liệu để tìm điểm chia (split point), khiến chi phí tính toán tăng theo cấp số nhân khi số lượng đặc trưng và mẫu dữ liệu tăng lên. Đây chính là vấn đề mà LightGBM được thiết kế để giải quyết.

### 5.3.3.2 Gradient-based One-Side Sampling (GOSS)

GOSS là một kỹ thuật độc đáo mà LightGBM sử dụng để tăng tốc độ huấn luyện. Thay vì sử dụng toàn bộ dữ liệu trong quá trình huấn luyện, GOSS chỉ giữ lại các mẫu dữ liệu quan trọng nhất, giúp giảm số lượng mẫu mà mô hình cần xử lý.

Nguyên tắc của GOSS rất thông minh:

- Các mẫu có gradient lớn thường chứa nhiều thông tin quan trọng hơn, vì chúng đại diện cho những trường hợp mà mô hình hiện tại chưa học tốt.
- Ngược lại, các mẫu có gradient nhỏ ít quan trọng hơn, vì mô hình đã xử lý tốt những trường hợp này.

Vì vậy, GOSS sẽ giữ lại tất cả các mẫu có gradient lớn và chỉ chọn ngẫu nhiên một phần nhỏ các mẫu có gradient nhỏ để giảm tải. Kỹ thuật này vừa đảm bảo mô hình không bị mất thông tin quan trọng, vừa tăng tốc độ huấn luyện một cách đáng kể.

### 5.3.3.3 Exclusive Feature Bundling (EFB)

Một vấn đề lớn khác mà LightGBM giải quyết là số lượng đặc trưng lớn trong các bài toán học máy. Trong nhiều trường hợp, không gian đặc trưng thường khá thừa – nghĩa là, có rất nhiều đặc trưng mà hiếm khi chúng đồng thời mang giá trị khác không. Điều này tạo ra cơ hội để giảm bớt số lượng đặc trưng mà không làm mất đi thông tin.

EFB hoạt động bằng cách "gộp" các đặc trưng gần nhau độc lập thành một đặc trưng duy nhất. Quá trình này có thể hình dung như sau:

- LightGBM coi mỗi đặc trưng là một đỉnh trong đồ thị, và nối một cạnh giữa hai đỉnh nếu hai đặc trưng đó không thể gộp chung (tức là không "độc lập").
- Sau đó, bài toán được chuyển thành một bài toán tô màu đồ thị, trong đó các đỉnh cùng màu sẽ được gộp vào một nhóm.

Kỹ thuật này giúp LightGBM giảm đáng kể số lượng đặc trưng phải xử lý, từ đó tăng tốc độ huấn luyện mà vẫn duy trì độ chính xác của mô hình.

### 5.3.3.4 Ứng dụng LightGBM trong Fraud Detection

LightGBM được xem là một trong những lựa chọn hàng đầu cho bài toán phát hiện gian lận, đặc biệt trong lĩnh vực như giao dịch thẻ tín dụng. Một số lý do khiến LightGBM rất phù hợp cho bài toán này gồm:

- **Khả năng xử lý dữ liệu lớn và mất cân bằng:** Trong bài toán phát hiện gian lận, số lượng giao dịch hợp pháp thường vượt trội so với giao dịch gian lận, gây ra tình trạng mất cân bằng dữ liệu. Nhờ GOSS, LightGBM có thể tập trung vào các giao dịch quan trọng nhất (những giao dịch có khả năng là gian lận), giúp cải thiện độ chính xác của mô hình.
- **Tốc độ huấn luyện nhanh:** Với các kỹ thuật như GOSS và EFB, LightGBM tối ưu hóa thời gian huấn luyện mà vẫn đảm bảo chất lượng. Điều này rất hữu ích khi cần cập nhật mô hình thường xuyên với dữ liệu mới.
- **Xử lý tốt không gian đặc trưng phức tạp:** Các bài toán phát hiện gian lận thường liên quan đến nhiều đặc trưng khác nhau, như thời gian, địa điểm, loại giao dịch, giá trị giao dịch,...

Nhờ kỹ thuật EFB, LightGBM có thể xử lý hiệu quả các đặc trưng đa chiều và thưa thớt, đảm bảo mô hình tận dụng tối đa thông tin từ dữ liệu.

#### 5.4 Độ đo sử dụng

Trong bài toán phát hiện gian lận, việc đánh giá hiệu quả của mô hình là rất quan trọng để đảm bảo rằng mô hình có thể phân loại đúng các giao dịch gian lận. Các độ đo thường dùng trong bài toán này bao gồm Accuracy, Precision, Recall, và F1 Score [7]. Dưới đây là lý thuyết chi tiết và công thức tính toán của từng độ đo:

##### 5.4.1 Accuracy (Độ chính xác)

Định nghĩa: Accuracy là tỉ lệ giữa số lượng dự đoán chính xác và tổng số lượng dự đoán. Đây là độ đo được sử dụng phổ biến để đánh giá mô hình phân loại.

Công thức:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

TP (True Positives): Số lượng mẫu gian lận được dự đoán đúng.

TN (True Negatives): Số lượng mẫu không gian lận được dự đoán đúng.

FP (False Positives): Số lượng mẫu không gian lận bị dự đoán sai thành gian lận.

FN (False Negatives): Số lượng mẫu gian lận bị bỏ sót (dự đoán sai thành không gian lận).

Nhưng cần lưu ý rằng trong ngữ cảnh bài toán nhóm đang thực hiện, Accuracy có thể không phù hợp trong bài toán mất cân bằng vì số lượng lớn giao dịch hợp pháp có thể làm che khuất các giao dịch gian lận.

##### 5.4.2 Precision (Độ chính xác khi dự đoán dương)

Định nghĩa: Precision đo lường tỉ lệ giữa số lượng dự đoán đúng là dương (gian lận) trên tổng số lượng dự đoán dương. Precision cho biết trong số những giao dịch được dự đoán là gian lận, có bao nhiêu giao dịch thực sự là gian lận.

Công thức:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Precision cao có nghĩa là ít có giao dịch hợp pháp bị nhầm lẫn thành gian lận, do đó rất hữu ích khi muốn giảm thiểu False Positives.

##### 5.4.3 Recall (Độ nhạy)

Định nghĩa: Recall đo lường tỉ lệ giữa số lượng dự đoán đúng là dương trên tổng số lượng mẫu thực sự là dương. Nó cho biết khả năng mô hình phát hiện được các trường hợp gian lận.

Công thức:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall cao đảm bảo rằng hầu hết các giao dịch gian lận đều được phát hiện, giảm thiểu False Negatives, điều này rất quan trọng trong việc phát hiện gian lận vì mục tiêu chính là không bỏ sót các giao dịch gian lận.

#### 5.4.4 F1 Score

Định nghĩa: F1 Score là trung bình điều hòa giữa Precision và Recall, dùng để cân bằng giữa hai độ đo này. F1 Score đặc biệt hữu ích khi tập dữ liệu mất cân bằng, như trong trường hợp phát hiện gian lận (khi số lượng giao dịch hợp pháp lớn hơn rất nhiều so với số lượng giao dịch gian lận).

Công thức:

$$F1 = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1 Score giúp xác định sự cân bằng giữa việc không bỏ sót các giao dịch gian lận (Recall) và giảm thiểu việc nhầm lẫn giao dịch hợp pháp thành gian lận (Precision).

### 5.5 Chiến thuật tối ưu hóa tham số

#### 5.5.1 RandomizedSearchCV

Nhóm sử dụng RandomizedSearchCV với StratifiedKFold (5 folds). Kỹ thuật này sẽ chọn ngẫu nhiên 20 bộ tham số và huấn luyện mô hình để đánh giá. Kết quả cuối cùng sẽ chọn bộ tham số có F1 Score tốt nhất làm tham số tối ưu

Tại sao F1 Score phù hợp? Trong bài toán phát hiện gian lận, dữ liệu thường bị mất cân bằng – số lượng giao dịch hợp pháp lớn hơn rất nhiều so với giao dịch gian lận. Điều này khiến Accuracy không phản ánh đúng hiệu quả của mô hình. F1 Score là trung bình điều hòa giữa Precision và Recall, giúp cân bằng giữa khả năng phát hiện các giao dịch gian lận (Recall) và giảm nhầm lẫn các giao dịch hợp pháp thành gian lận (Precision)

Trong bài toán phát hiện gian lận, việc giảm thiểu False Negatives (bỏ sót giao dịch gian lận) là ưu tiên hàng đầu, nhưng cũng cần giảm False Positives (dự đoán nhầm giao dịch hợp pháp). Vì vậy, F1 Score là độ đo lý tưởng để đánh giá hiệu quả tổng thể của mô hình.

```
def randomize_grid_search(models, param_distributions, X_train, y_train):
    opt_models = {}
    stratified_kfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    for name, model in models.items():
        print(f"Performing Randomized Search for {name}...")
        random_search = RandomizedSearchCV(model, param_distributions[name], n_iter=20, scoring='f1', cv=stratified_kfold, verbose=0, random_state=42, n_jobs=-1)
        random_search.fit(X_train, y_train)
        opt_models[name] = random_search.best_estimator_
        print(f"Best parameters for {name}: {random_search.best_params_}")
        print("-----")
    return opt_models
```

Python

#### 5.5.2 Các tham số tối ưu

Các tham số tối ưu:

```

● param_distributions = {
    'Decision Tree': {
        'max_depth': [5, 10, 20, None],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 5],
        'criterion': ['gini', 'entropy'],
        'class_weight': ['balanced', None]
    },
    'K-Nearest Neighbors': {
        'n_neighbors': [3, 5, 7, 9],
        'weights': ['uniform', 'distance']
    },
    'LightGBM': {
        'n_estimators': [50, 100, 200],
        'max_depth': [3, 5, 10],
        'learning_rate': [0.01, 0.05, 0.1],
        'num_leaves': [20, 31, 50],
        'min_child_samples': [10, 20, 30],
        'class_weight': ['balanced', None]
    }
}

```

### 5.5.2.1 Decision Tree:

#### Max\_depth (Độ sâu tối đa của cây):

- Giới hạn số mức của cây quyết định
- Đối với dữ liệu đơn giản hoặc nhỏ, giá trị nhỏ (ví dụ: 5, 10) có thể giúp tránh tình trạng **overfitting**.
- Nếu đặt là None (không giới hạn), cây có thể phát triển rất sâu, dễ dẫn đến overfitting, đặc biệt trên dữ liệu phức tạp.

#### Min\_samples\_split (Số lượng mẫu tối thiểu để thực hiện tách một node)

- Giá trị nhỏ (ví dụ: 2) cho phép chia nhanh hơn nhưng có nguy cơ overfitting.
- Giá trị lớn hơn (ví dụ: 10) giúp hạn chế việc chia nhỏ, làm cho cây tổng quát hơn.

#### Min\_samples\_leaf (Số lượng mẫu tối thiểu phải có tại một node lá):

- Node lá là điểm cuối cùng của cây
- Giá trị nhỏ hơn (ví dụ: 1) linh hoạt nhưng có nguy cơ overfitting.
- Giá trị lớn hơn (ví dụ: 5) giúp cây bớt phức tạp và tổng quát hóa tốt hơn.

#### Criterion (Tiêu chí đánh giá điểm chia):

- 'gini': Gini impurity (độ không đồng nhất).
- 'entropy': Đo lường bằng Entropy (Information Gain), dựa trên entropy, để đo lường chất lượng của điểm chia.

#### Class\_weight (Trọng số của các lớp):

- 'balanced': Tự động điều chỉnh trọng số dựa trên tỷ lệ các lớp, giúp xử lý dữ liệu mất cân bằng (lớp thiểu số rất nhỏ).
- None: Không áp dụng trọng số.

### 5.5.2.2 K-Nearest Neighbors (KNN)

#### N\_neighbors (Số hàng xóm gần nhất):

- Giá trị nhỏ (ví dụ: 3) cho phép tập trung vào các điểm gần nhất, nhưng có thể nhạy cảm với

nhiều

- Giá trị lớn hơn (ví dụ: 9) giúp làm mượt kết quả, phù hợp với dữ liệu ít nhiễu.

#### **Weights (trọng số cho các điểm lân cận):**

- 'uniform': Tất cả hàng xóm có trọng số bằng nhau.
- 'distance': Các hàng xóm gần hơn có trọng số cao hơn, giúp tăng độ chính xác trong một số trường hợp.

#### **5.5.2.3 LightGBM**

##### **N\_estimators (Số vòng lặp boosting):**

- Giá trị lớn ((ví dụ: 200) cải thiện hiệu suất nhưng sẽ tăng thời gian huấn luyện.

##### **Max\_depth (Độ sâu tối đa của mỗi cây):**

- Giá trị nhỏ (ví dụ: 3) giúp tránh overfitting, phù hợp với dữ liệu nhỏ.
- Giá trị lớn hơn (ví dụ: 10) phù hợp với dữ liệu phức tạp.

##### **learning\_rate (Tốc độ học của mô hình):**

- Giá trị nhỏ (ví dụ: 0.01) làm cho mô hình học từ từ, tốt hơn cho tổng quát hóa.
- Giá trị lớn (ví dụ: 0.1) học nhanh hơn nhưng có nguy cơ overfitting.

##### **Num\_leaves (Số lượng lá tối đa trong mỗi cây):**

- Giá trị nhỏ (ví dụ: 20) hạn chế phức tạp, giảm nguy cơ overfitting.
- Giá trị lớn (ví dụ: 50) cho phép cây học nhiều hơn, phù hợp với dữ liệu phức tạp.

##### **Min\_child\_samples (Số mẫu tối thiểu cần thiết để một node lá được chia):**

- Giá trị nhỏ (ví dụ: 10) làm cho cây linh hoạt hơn, nhưng có thể dẫn đến overfitting.
- Giá trị lớn (ví dụ: 30) hạn chế việc chia nhỏ không cần thiết, giúp cây ổn định hơn

##### **Class\_weight (Trọng số lớp):**

- 'balanced': Xử lý mất cân bằng dữ liệu bằng cách tự động tính trọng số cho các lớp.
- None: Không áp dụng trọng số.

## **5.6 Các bước thực hiện**

Bước 1: Chia dataset đã xử lý thành 2 tập train và test với tỉ lệ 8:2

```
X = cleaned_df.drop(columns=['is_fraud'])
y = cleaned_df['is_fraud']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

Bước 2: Chuẩn hóa các biến số bằng hàm StandardScaler

```
columns_to_scale = [
    'amount', 'age', 'merchant_name_freq', 'job_freq', 'city_freq',
    'state_freq', 'hour_sin', 'hour_cos', 'day_sin', 'day_cos'
]

# Instantiate the scaler
scaler = StandardScaler()

# Fit and transform the training set, and transform the test set
X_train[columns_to_scale] = scaler.fit_transform(X_train[columns_to_scale])
X_test[columns_to_scale] = scaler.transform(X_test[columns_to_scale])
```

Ta thực hiện huấn luyện mô hình và xuất ra các độ đo với hàm sau

```
def train_models(X_train, y_train, X_test, y_test, models):
    results = []
    for name, model in models.items():
        # Record start time
        start_time = time.time()
        # Train the model
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        # Record end time
        end_time = time.time()
        # Evaluate the model
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, pos_label=1)
        recall = recall_score(y_test, y_pred, pos_label=1)
        f1 = f1_score(y_test, y_pred, pos_label=1)
        roc_auc = roc_auc_score(y_test, y_pred)
        # Store the results in the list
        results.append({
            'Model': name,
            'Accuracy': accuracy,
            'Precision': precision,
            'Recall': recall,
            'F1 Score': f1,
            'Time Taken (s)': end_time - start_time
        })
    # Print evaluation metrics
    print(f"{name} Evaluation on Test Set:")
    print(classification_report(y_test, y_pred))
    print(f"ROC AUC Score: {roc_auc:.4f}")
    print(f"Time taken for {name}: {end_time - start_time:.2f} seconds")
    print("-----")
    # Plot confusion matrix
    cm = confusion_matrix(y_test, y_pred)
    disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[0, 1])
    disp.plot(cmap='Blues')
    plt.title(f"Confusion Matrix for {name}")
    plt.show()
    # Convert results to a DataFrame
    results_df = pd.DataFrame(results)
    return results_df

models = {
    'Decision Tree': DecisionTreeClassifier(random_state=42),
    'K-Nearest Neighbors': KNeighborsClassifier(n_neighbors=5),
    'LightGBM': LGBMClassifier(random_state=42)
}

results = train_models(X_train, y_train, X_test, y_test, models)
✓ 25.3s
```

Bước 3: Thực hiện parameter tuning bằng cách dùng hàm RandomizedSearchCV để tìm ra tổ hợp tham số đạt kết quả hiệu quả nhất theo f1-score

```
def randomize_grid_search(models, param_distributions, X_train, y_train):
    opt_models = {}
    stratified_kfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    for name, model in models.items():
        print(f"Performing Randomized Search for {name}...")
        random_search = RandomizedSearchCV(model, param_distributions[name], n_iter=20, scoring='f1', cv=stratified_kfold, verbose=0, random_state=42, n_jobs=-1)
        random_search.fit(X_train, y_train)
        opt_models[name] = random_search.best_estimator_
        print(f"Best parameters for {name}: {random_search.best_params_}")
        print("-----")
    return opt_models
✓ 0.0s
```

```

param_distributions = {
    'Decision Tree': {
        'max_depth': [5, 10, 20, None],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 5],
        'criterion': ['gini', 'entropy'],
        'class_weight': ['balanced', None]
    },
    'K-Nearest Neighbors': {
        'n_neighbors': [3, 5, 7, 9],
        'weights': ['uniform', 'distance']
    },
    'LightGBM': {
        'n_estimators': [50, 100, 200],
        'max_depth': [3, 5, 10],
        'learning_rate': [0.01, 0.05, 0.1],
        'num_leaves': [20, 31, 50],
        'min_child_samples': [10, 20, 30],
        'class_weight': ['balanced', None]
    }
}

opt_models = randomize_grid_search(models, param_distributions, X_train, y_train)
✓ 12m 34.9s

```

Tham số sau khi được tối ưu:

```

{'Decision Tree': DecisionTreeClassifier(max_depth=10, min_samples_split=5, random_state=42),
 'K-Nearest Neighbors': KNeighborsClassifier(n_neighbors=3, weights='distance'),
 'LightGBM': LGBMClassifier(max_depth=5, n_estimators=200, random_state=42)}

```

Các tham số này sẽ được cho train lại với tập test một lần nữa để ra kết quả.

## 5.7 Đánh giá và so sánh mô hình

### 5.7.1 Kết quả:

Kết quả huấn luyện ban đầu

	Model	Accuracy	Precision	Recall	F1 Score	Time Taken (s)
0	Decision Tree	0.993900	0.846980	0.840773	0.843865	2.480708
1	K-Nearest Neighbors	0.989655	0.884074	0.543638	0.673267	21.219540
2	LightGBM	0.995742	0.927895	0.848767	0.886569	0.895253

Kết quả huấn luyện sau khi được tối ưu hóa tham số

	Model	Accuracy	Precision	Recall	F1 Score	Time Taken (s)
0	Decision Tree	0.994893	0.926923	0.802798	0.860407	1.938858
1	K-Nearest Neighbors	0.990818	0.850000	0.645570	0.733813	19.370515
2	LightGBM	0.996291	0.947756	0.858095	0.900699	1.294518

### 5.7.2 So sánh kết quả trước và sau khi tối ưu hóa tham số

Trong các bài toán phân loại với dữ liệu mất cân bằng, Accuracy (độ chính xác) thường không phải là thước đo phù hợp. Lý do là khi một lớp chiếm đa số áp đảo (ví dụ: lớp 0 trong trường hợp này), mô hình có thể đạt độ chính xác cao chỉ bằng cách dự đoán hầu hết các mẫu thuộc lớp chiếm đa số. Điều này làm cho Accuracy không phản ánh đúng khả năng của mô hình trong việc nhận diện lớp thiểu số (lớp 1). Vì vậy, các thước đo như Precision, Recall, và F1 Score được ưu tiên sử dụng để đánh giá hiệu quả của mô hình. Dưới đây là so sánh chi tiết trước và sau khi tối ưu hóa tham số.

### 5.7.2.1 Decision Tree

Trước tối ưu hóa, mô hình đạt F1 Score là 0.8439, với Precision là 0.8470 và Recall là 0.8408. Kết quả này cho thấy Decision Tree hoạt động khá tốt nhưng chưa tối ưu, đặc biệt trong việc cân bằng giữa Precision và Recall. Thời gian xử lý là 2.48 giây.

Sau khi tối ưu hóa, F1 Score tăng lên 0.8604, chủ yếu nhờ cải thiện lớn ở Precision (lên 0.9269). Tuy nhiên, Recall giảm nhẹ xuống 0.8028, nghĩa là mô hình đã bỏ sót nhiều hơn một số mẫu của lớp thiểu số. Mặt khác, thời gian xử lý giảm còn 1.94 giây, cho thấy hiệu quả về tốc độ đã được cải thiện.

### 5.7.2.2 K-Nearest Neighbors (KNN)

Trước tối ưu hóa, KNN là mô hình có hiệu suất thấp nhất trước khi tối ưu hóa, với F1 Score chỉ đạt 0.6733. Điều này chủ yếu do Recall rất thấp (0.5436), cho thấy mô hình gặp khó khăn lớn trong việc nhận diện lớp thiểu số. Precision đạt 0.8500, thể hiện mô hình ít nhầm lẫn lớp đa số thành lớp thiểu số. Thời gian xử lý tương đối cao, 21.22 giây.

Sau khi tối ưu hóa, F1 Score cải thiện đáng kể lên 0.7338 nhờ Recall tăng lên 0.6456, dù Precision giảm nhẹ xuống 0.8500. Điều này cho thấy mô hình đã nhận diện tốt hơn các mẫu của lớp thiểu số, nhưng sự đánh đổi này là hợp lý. Thời gian xử lý cũng giảm nhẹ xuống 19.37 giây, giúp tăng hiệu quả tổng thể

### 5.7.2.3 LightGBM

LightGBM là mô hình tốt nhất ngay từ đầu, với F1 Score đạt 0.8866. Kết quả này nhờ vào Precision cao (0.9279) và Recall tốt (0.8488). Thời gian xử lý rất nhanh, chỉ 0.90 giây, làm cho LightGBM vượt trội so với các mô hình khác.

Sau tối ưu hóa, F1 Score tăng lên 0.9007, một mức cải thiện nhỏ nhưng ý nghĩa. Precision tăng lên 0.9478, trong khi Recall vẫn giữ ở mức cao (0.8581). Mặc dù thời gian xử lý tăng nhẹ lên 1.29 giây, LightGBM vẫn là mô hình nhanh nhất và hiệu quả nhất trong cả ba mô hình.

## 5.7.3 Nhận xét tổng quan và đề xuất mô hình

Với dữ liệu mất cân bằng, các thước đo Precision, Recall, và F1 Score là những yếu tố quan trọng nhất để đánh giá hiệu quả của mô hình. Những thước đo này không bị ảnh hưởng bởi sự lệch lớp và tập trung vào khả năng nhận diện lớp thiểu số (lớp gian lận).

Decision Tree: Sau tối ưu hóa, Decision Tree có sự cải thiện nhẹ ở F1 Score và tốc độ xử lý. Tuy nhiên, Recall giảm đã làm giảm khả năng nhận diện lớp thiểu số, khiến mô hình chưa phải là lựa chọn tốt nhất.

KNN: Mô hình này có sự cải thiện rõ rệt về Recall và F1 Score sau tối ưu hóa, nhưng vẫn không đạt hiệu suất cao bằng LightGBM. Thời gian xử lý của KNN cũng lâu hơn so với các mô hình khác, đặc biệt trên dữ liệu lớn.

LightGBM: Đây là mô hình có hiệu suất vượt trội nhất. F1 Score sau tối ưu hóa đạt 0.9007, cao nhất trong cả ba mô hình, nhờ vào sự cân bằng tốt giữa Precision (0.9478) và Recall (0.8581). Mặc dù thời gian xử lý tăng nhẹ, LightGBM vẫn nhanh hơn đáng kể so với Decision Tree và KNN, làm cho nó trở thành lựa chọn lý tưởng.

Kết luận: LightGBM là mô hình phù hợp nhất cho bài toán phát hiện gian lận. Nó không chỉ đạt hiệu suất cao mà còn xử lý nhanh, phù hợp với yêu cầu thực tế, đặc biệt khi cần cập nhật mô hình thường xuyên.

## TÀI LIỆU THAM KHẢO

- [1] Pádraig Cunningham & Sarah Jane Delany (2020). k-Nearest Neighbour Classifiers 2nd Edition (with Python examples)
- [2] J.R. QUINLAN (1986). Induction of Decision Trees. *Machine Learning 1: 81-106*
- [3] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye & Tie-Yan Liu (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, 3149 – 3157*
- [4] Kuhn, M., & Johnson, K. (2019). Feature Engineering and Selection: A Practical Approach for Predictive Models. Chapman and Hall/CRC.
- [5] T. Mahajan, G. Singh, G. Bruns, G. Bruns, T. Mahajan, & G. Sing (2021) An Experimental Assessment of Treatments for Cyclical Data, *σto Proceedings of the 2021 Computer Science Conference for CSU Undergraduates, Virtual, τ. 6.*
- [6] GABRIELA S. RÊMA, BENEDITO D. BONATTO, RAFAEL M. SOARES & ANTONIO C. S. LIMA (2024). Data Analysis Methodology Utilizing the Statistical Metrics Weight of Evidence (WoE) and Information Value (IV) to Assist in Asset Management of Power Transformers. *IEEE Access, Volume: 12, 165322 – 165334*
- [7] Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. *Journal of Machine Learning Technologies, 2(1), 37–63.*