DATA SOLUTIONS PLATFORM

DATA ANALYTICS

POWER BI DATA ANALYSIS

*With Power BI Expert…. **Benjamin Taiwo***

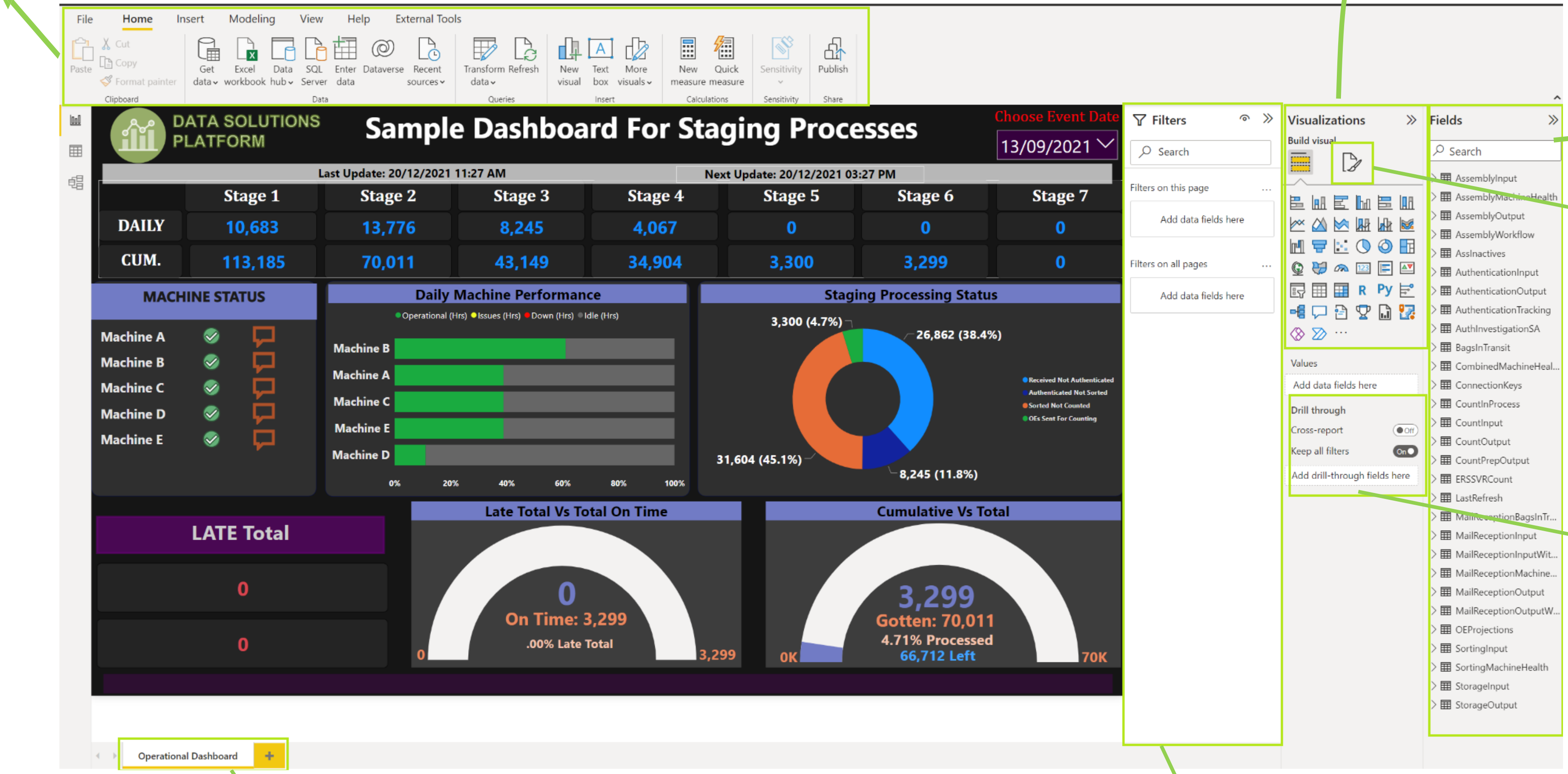
TODAY’S CONTENT

- Meet Power BI Desktop
- Power BI Workflow
- Options and Settings
- Model View Tabs
- Dimensional vs Fact Tables
- Star Schema and Snowflake Models
- Creating Table Relationships
- Relationship Cardinality
- Filter Flow
- Active and Inactive Relationships
- Organizing the tables
- Hiding fields from report view
- Automatic Date Tables
- Database Normalization
- Data Model Best Practices
- Data View Tabs
- DAX Introduction
- Calculated Columns
- Measures
- Quick Measures
- Measures and Calculated Columns
- Categories of common Dax Functions

MEET POWER BI DESKTOP

Menu/Home Options
Data, Queries, Insert, Calculations, Sensitivity, Share.

VISUALIZATION OPTIONS
Charts, Slicers, Maps, Matrices, etc.



Sample Dashboard For Staging Processes

Last Update: 20/12/2021 11:27 AM

Next Update: 20/12/2021 03:27 PM

	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7
DAILY	10,683	13,776	8,245	4,067	0	0	0
CUM.	113,185	70,011	43,149	34,904	3,300	3,299	0

- AssemblyInput
- AssemblyMachineHealth
- AssemblyOutput
- AssemblyWorkflow
- AssInactives
- AuthenticationInput
- AuthenticationOutput
- AuthenticationTracking
- AuthInvestigationSA
- BagsInTransit
- CombinedMachineHeal...
- ConnectionKeys
- CountInProcess
- CountInput
- CountOutput
- CountPrepOutput
- ERSSVRCount
- LastRefresh
- MailReceptionBagsInTr...
- MailReceptionInput
- MailReceptionInputWit...
- MailReceptionMachine...
- MailReceptionOutput
- MailReceptionOutputW...
- OEProjections
- SortingInput
- SortingMachineHealth
- StorageInput
- StorageOutput

FIELD LIST
Tables, Columns, Measures

FIELDS/FORMAT/ ANALYICTS PANE
Visual-specific configuration & formatting tools

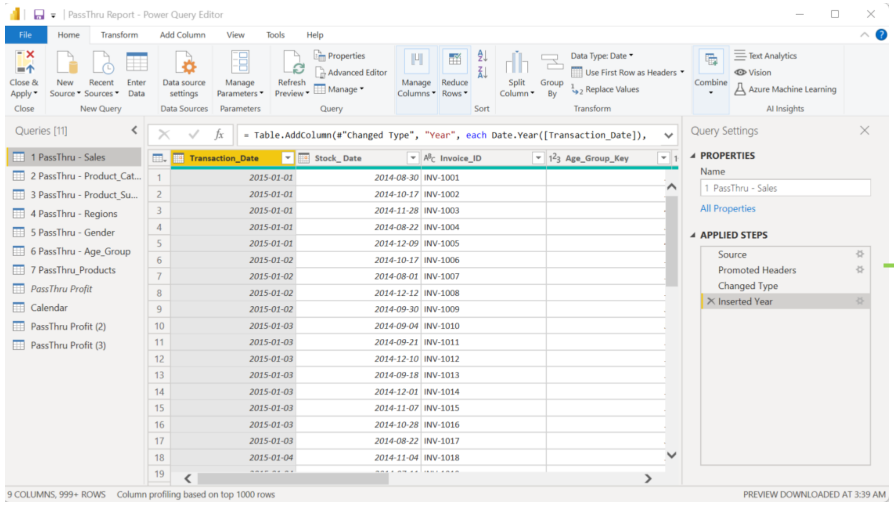
DRILL THROUGH FILTERS
Options for page-level drill through filters

REPORT PAGES
Similar to Excel tabs; each is a blank reporting canvas

FILTERS PANE
Visual-Level, Page-Level, and Report-Level Filters

THE POWER BI WORKFLOW

1. *Connect, load and transform dataset (Power Query)*



PassThru Report - Power Query Editor

Queries [11]

- 1 PassThru - Sales
- 2 PassThru - Product_Cat...
- 3 PassThru - Product_Su...
- 4 PassThru - Regions
- 5 PassThru - Gender
- 6 PassThru - Age_Group
- 7 PassThru_Products
- PassThru Profit
- Calendar
- PassThru Profit (2)
- PassThru Profit (3)

```
= Table.AddColumn(#"Changed Type", "Year", each Date.Year([Transaction_Date]),
```

	Transaction_Date	Stock_ Date	Invoice_ID	Age_Group_Key
1	2015-01-01	2014-08-30	INV-1001	
2	2015-01-01	2014-10-17	INV-1002	
3	2015-01-01	2014-11-28	INV-1003	
4	2015-01-01	2014-08-22	INV-1004	
5	2015-01-01	2014-12-09	INV-1005	
6	2015-01-02	2014-10-17	INV-1006	
7	2015-01-02	2014-08-01	INV-1007	
8	2015-01-02	2014-12-12	INV-1008	
9	2015-01-02	2014-09-30	INV-1009	
10	2015-01-03	2014-09-04	INV-1010	
11	2015-01-03	2014-09-21	INV-1011	
12	2015-01-03	2014-12-10	INV-1012	
13	2015-01-03	2014-09-18	INV-1013	
14	2015-01-03	2014-12-01	INV-1014	
15	2015-01-03	2014-11-07	INV-1015	
16	2015-01-03	2014-10-28	INV-1016	
17	2015-01-03	2014-08-22	INV-1017	
18	2015-01-04	2014-11-04	INV-1018	

Query Settings

PROPERTIES

Name: 1 PassThru - Sales

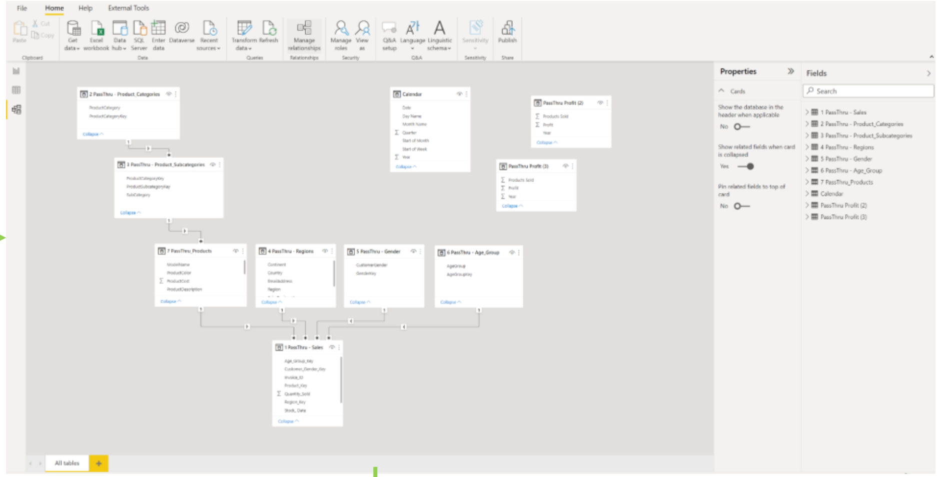
All Properties

APPLIED STEPS

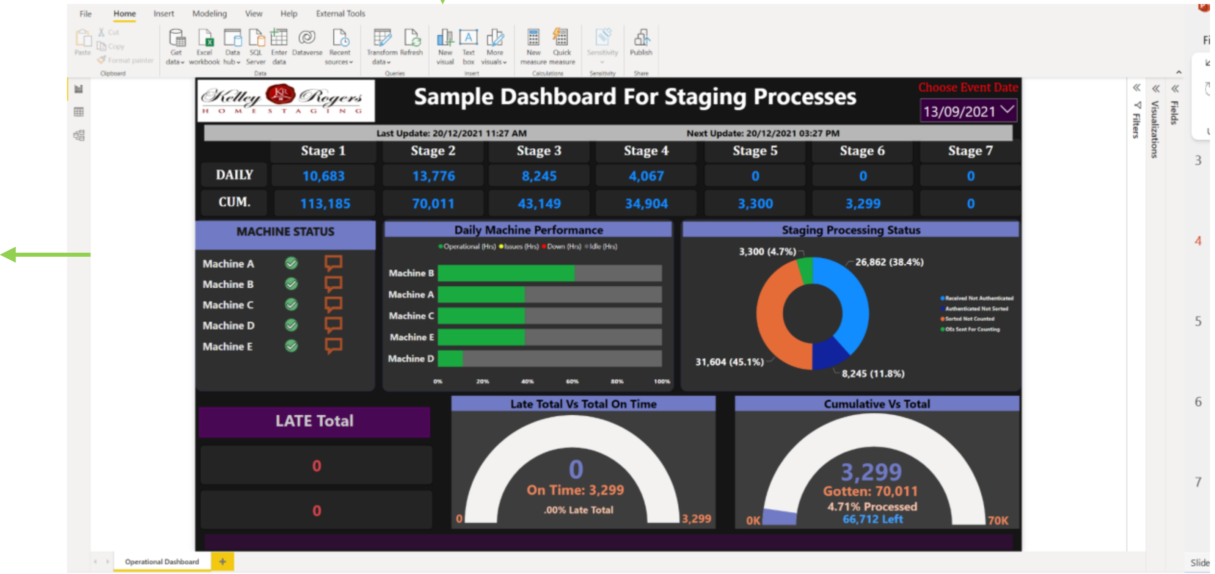
- Source
- Promoted Headers
- Changed Type
- Inserted Year

9 COLUMNS, 999+ ROWS   Column profiling based on top 1000 rows   PREVIEW DOWNLOADED AT 3:39 AM

2. *Build table relationships to create model for the cleaned dataset (PBI Desktop Model)*



3. *Develop interactive reports to understand and visualize the data (PBI Desktop Report)*



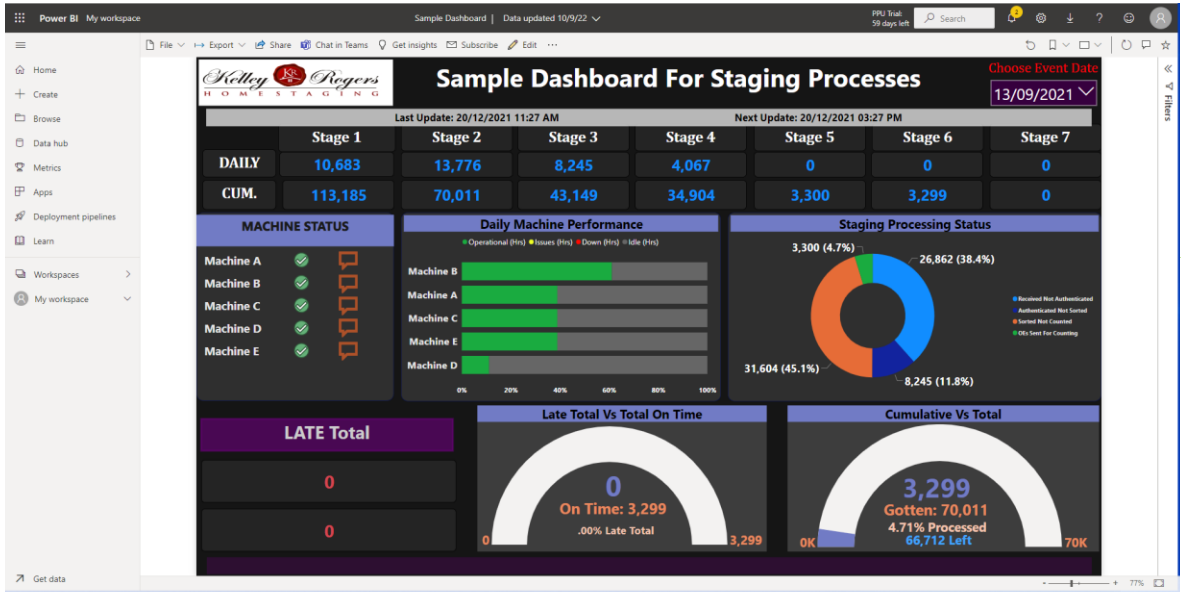
Sample Dashboard For Staging Processes

Choose Event Date 13/09/2021

Last Update: 20/12/2021 11:27 AM   Next Update: 20/12/2021 03:27 PM

	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7
DAILY	10,683	13,776	8,245	4,067	0	0	0
CUM.	113,185	70,011	43,149	34,904	3,300	3,299	0

4. *Build interactive dashboards and share insights (PBI Service)*



Sample Dashboard For Staging Processes

Choose Event Date 13/09/2021

Last Update: 20/12/2021 11:27 AM   Next Update: 20/12/2021 03:27 PM

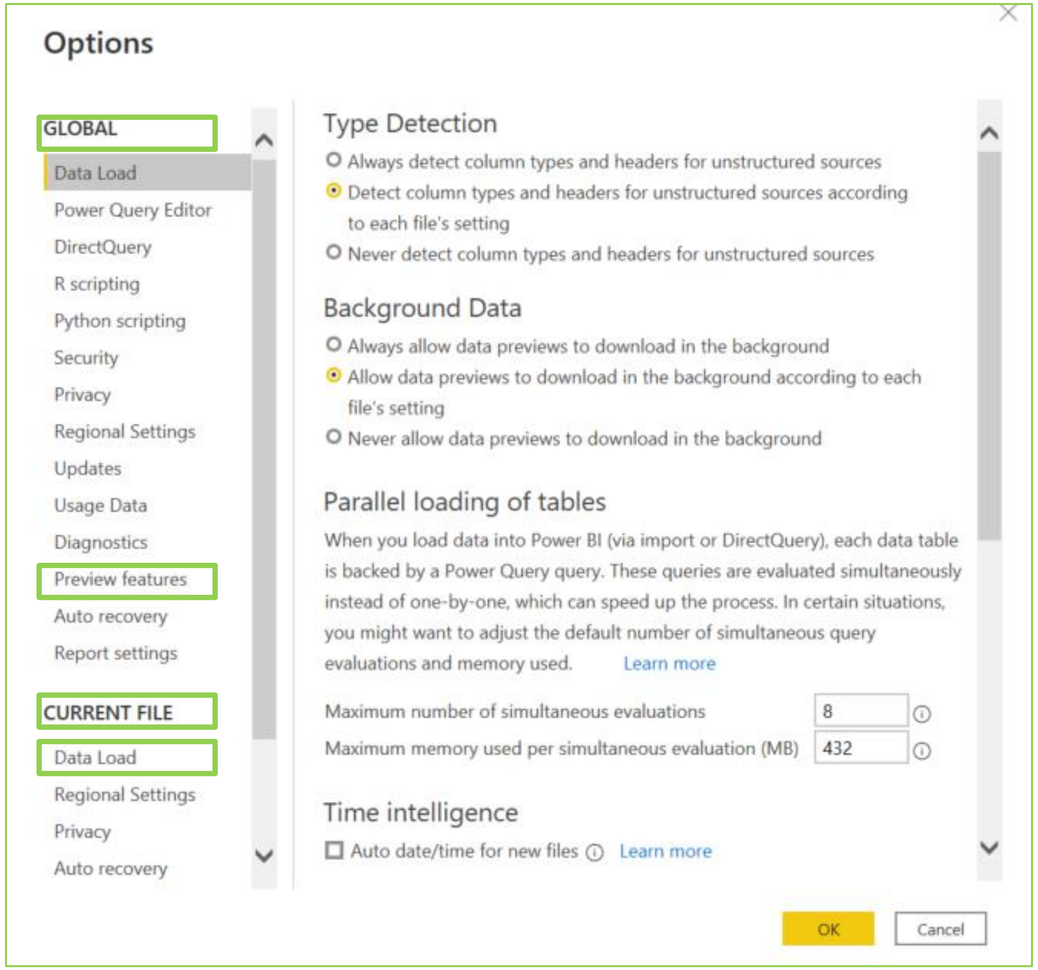
	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7
DAILY	10,683	13,776	8,245	4,067	0	0	0
CUM.	113,185	70,011	43,149	34,904	3,300	3,299	0

LATE Total

0

0

OPTIONS AND SETTINGS



Type Detection

○ Always detect column types and headers for unstructured sources

● Detect column types and headers for unstructured sources according to each file's setting

○ Never detect column types and headers for unstructured sources

Background Data

○ Always allow data previews to download in the background

● Allow data previews to download in the background according to each file's setting

○ Never allow data previews to download in the background

Parallel loading of tables

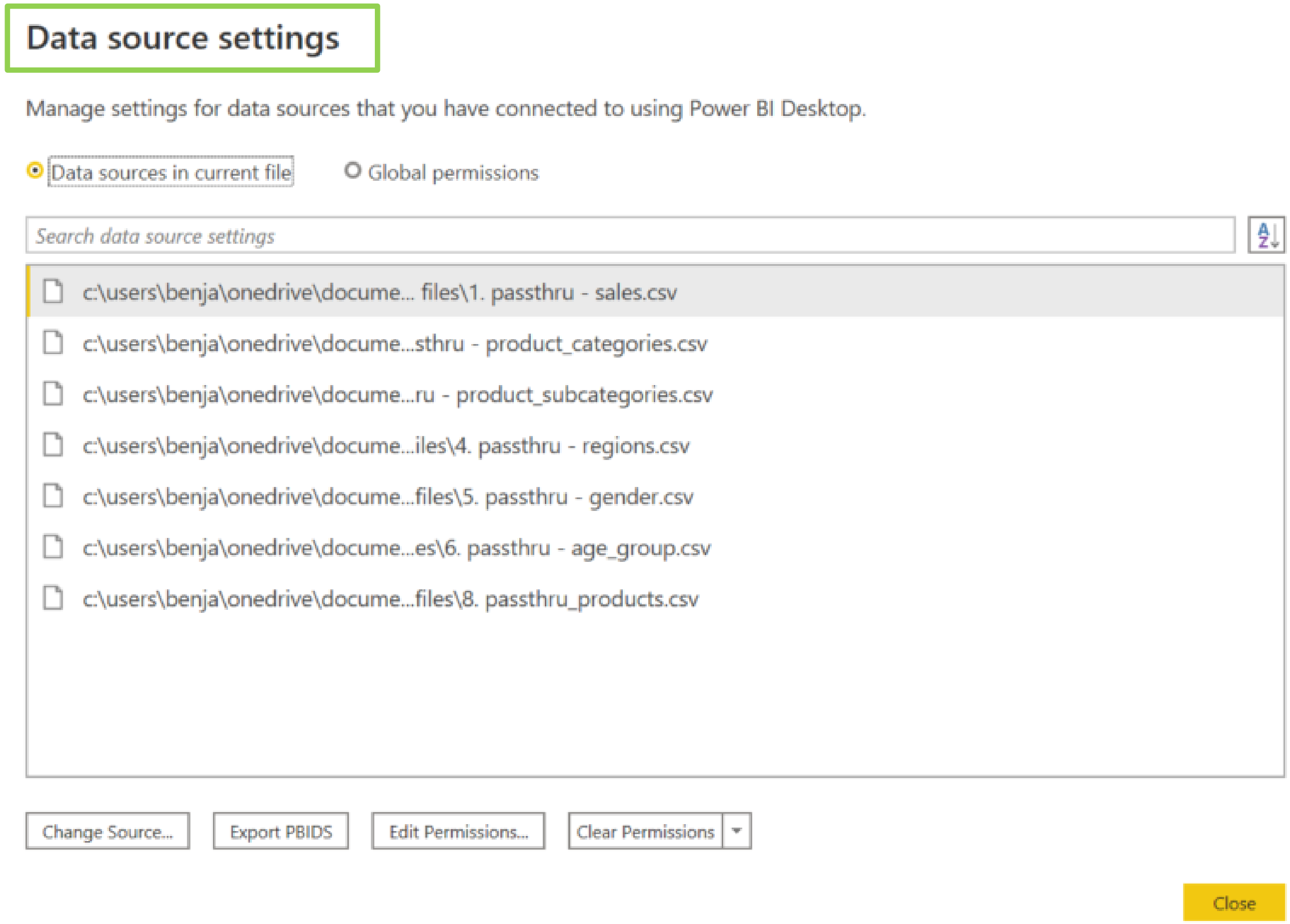
When you load data into Power BI (via import or DirectQuery), each data table is backed by a Power Query query. These queries are evaluated simultaneously instead of one-by-one, which can speed up the process. In certain situations, you might want to adjust the default number of simultaneous query evaluations and memory used. Learn more

Maximum number of simultaneous evaluations 8

Maximum memory used per simultaneous evaluation (MB) 432

Time intelligence

□ Auto date/time for new files Learn more



Data source settings

Manage settings for data sources that you have connected to using Power BI Desktop.

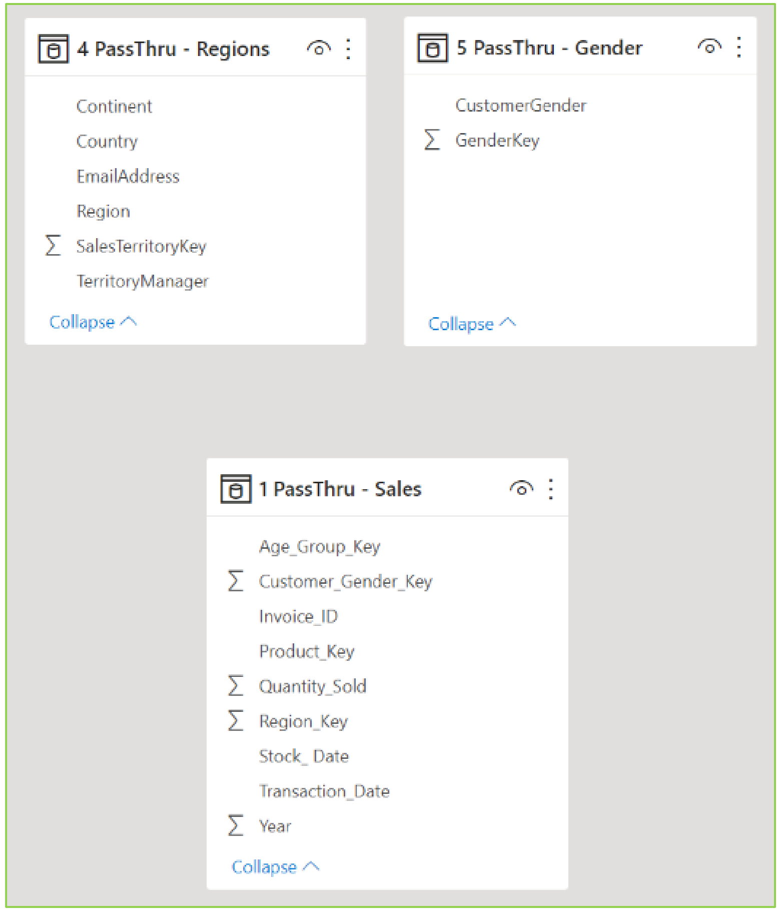
● Data sources in current file ○ Global permissions

Search data source settings

- c:\users\benja\onedrive\docume... files\1. passthru - sales.csv
- c:\users\benja\onedrive\docume...sthru - product_categories.csv
- c:\users\benja\onedrive\docume...ru - product_subcategories.csv
- c:\users\benja\onedrive\docume...iles\4. passthru - regions.csv
- c:\users\benja\onedrive\docume...files\5. passthru - gender.csv
- c:\users\benja\onedrive\docume...es\6. passthru - age_group.csv
- c:\users\benja\onedrive\docume...files\8. passthru_products.csv

WHAT’S A “DATA MODEL”?

This **IS NOT** a data model



4 PassThru - Regions

Continent
Country
EmailAddress
Region
∑ SalesTerritoryKey
TerritoryManager

5 PassThru - Gender

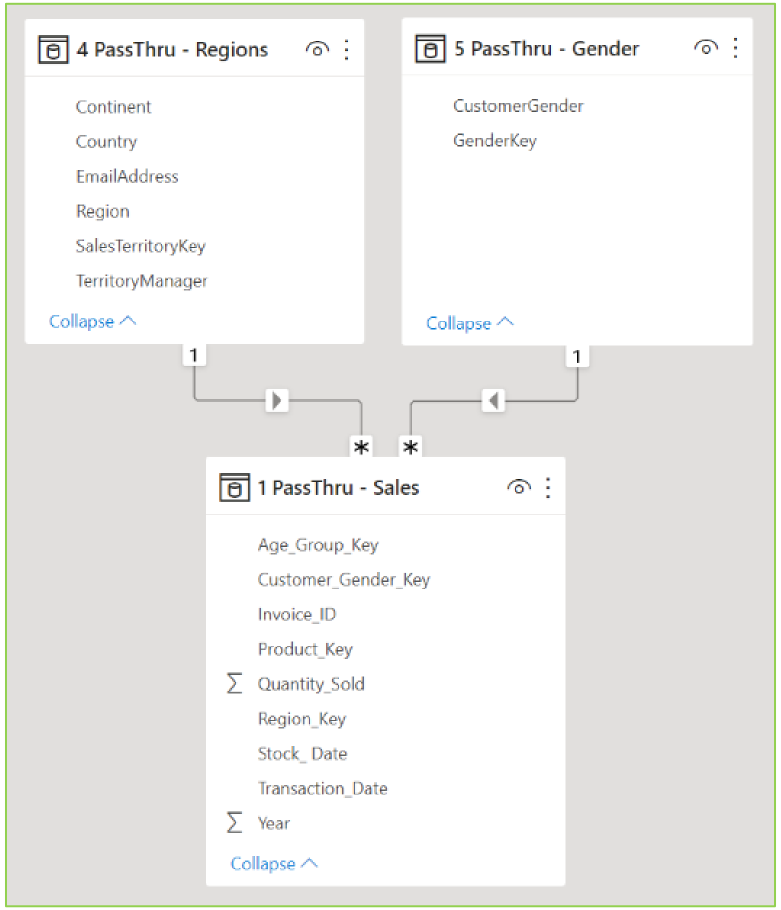
CustomerGender
∑ GenderKey

1 PassThru - Sales

Age_Group_Key
∑ Customer_Gender_Key
Invoice_ID
Product_Key
∑ Quantity_Sold
∑ Region_Key
Stock_ Date
Transaction_Date
∑ Year

No relationships with individual tables

This **IS** a data model



4 PassThru - Regions

Continent
Country
EmailAddress
Region
SalesTerritoryKey
TerritoryManager

5 PassThru - Gender

CustomerGender
GenderKey

1 PassThru - Sales

Age_Group_Key
Customer_Gender_Key
Invoice_ID
Product_Key
∑ Quantity_Sold
Region_Key
Stock_ Date
Transaction_Date
∑ Year

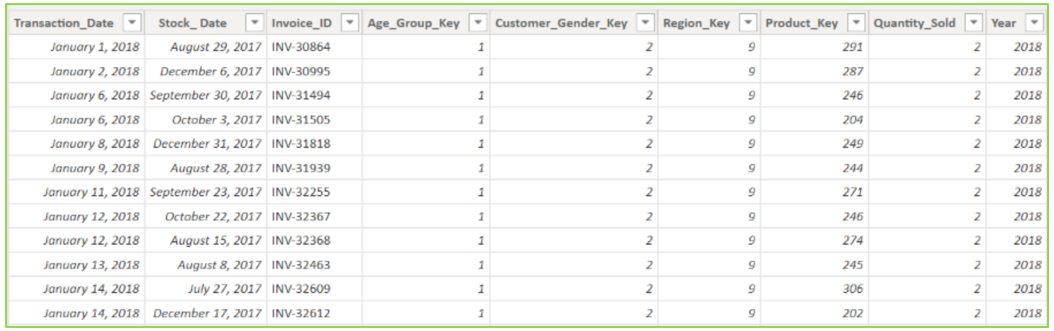
Relationships with individual tables using their common keys

DIMENSION VS FACT TABLES

Generally, data model contains two types of tables: **Fact** tables and **Dimension** tables.

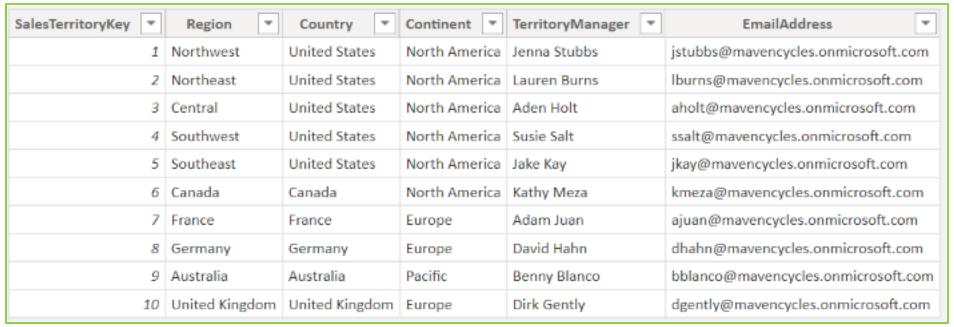
It is very important to understand the difference between these concepts.

- **The fact or data table** contains measurements, metrics, and facts about a business process. These tables hold business facts to be analyzed and foreign keys**.**
- **Dimension or Lookup Tables** contains descriptive attributes to be used as query constraining. These tables contains dimensions of business fact



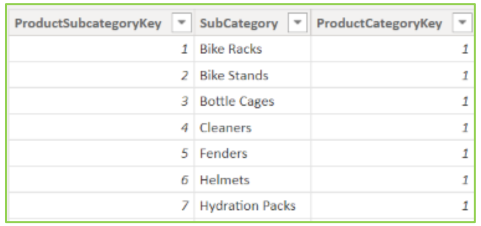
Transaction_Date	Stock_ Date	Invoice_ID	Age_Group_Key	Customer_Gender_Key	Region_Key	Product_Key	Quantity_Sold	Year
January 1, 2018	August 29, 2017	INV-30864	1	2	9	291	2	2018
January 2, 2018	December 6, 2017	INV-30995	1	2	9	287	2	2018
January 6, 2018	September 30, 2017	INV-31494	1	2	9	246	2	2018
January 6, 2018	October 3, 2017	INV-31505	1	2	9	204	2	2018
January 8, 2018	December 31, 2017	INV-31818	1	2	9	249	2	2018
January 9, 2018	August 28, 2017	INV-31939	1	2	9	244	2	2018
January 11, 2018	September 23, 2017	INV-32255	1	2	9	271	2	2018
January 12, 2018	October 22, 2017	INV-32367	1	2	9	246	2	2018
January 12, 2018	August 15, 2017	INV-32368	1	2	9	274	2	2018
January 13, 2018	August 8, 2017	INV-32463	1	2	9	245	2	2018
January 14, 2018	July 27, 2017	INV-32609	1	2	9	306	2	2018
January 14, 2018	December 17, 2017	INV-32612	1	2	9	202	2	2018

*This **Fact Table** contains “**quantity**” values, and connects to dimension tables through their primary keys*



SalesTerritoryKey	Region	Country	Continent	TerritoryManager	EmailAddress
1	Northwest	United States	North America	Jenna Stubbs	jstubbs@mavencycles.onmicrosoft.com
2	Northeast	United States	North America	Lauren Burns	lburns@mavencycles.onmicrosoft.com
3	Central	United States	North America	Aden Holt	aholt@mavencycles.onmicrosoft.com
4	Southwest	United States	North America	Susie Salt	ssalt@mavencycles.onmicrosoft.com
5	Southeast	United States	North America	Jake Kay	jkay@mavencycles.onmicrosoft.com
6	Canada	Canada	North America	Kathy Meza	kmeza@mavencycles.onmicrosoft.com
7	France	France	Europe	Adam Juan	ajuan@mavencycles.onmicrosoft.com
8	Germany	Germany	Europe	David Hahn	dhahn@mavencycles.onmicrosoft.com
9	Australia	Australia	Pacific	Benny Blanco	bblanco@mavencycles.onmicrosoft.com
10	United Kingdom	United Kingdom	Europe	Dirk Gently	dgently@mavencycles.onmicrosoft.com

*This **Regions Dimension** table provides additional attributes about each **region** (country, continent, quarter, etc.)*

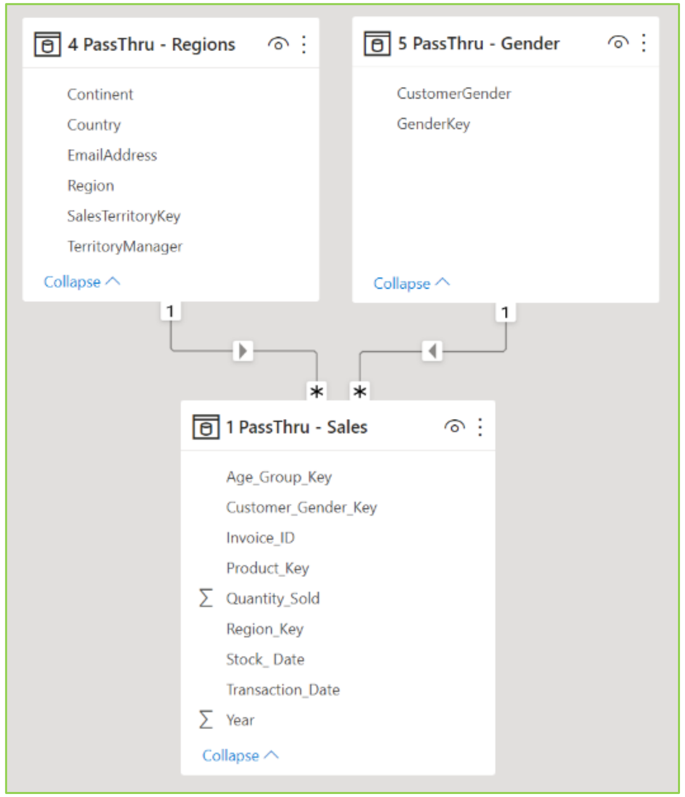


ProductSubcategoryKey	SubCategory	ProductCategoryKey
1	Bike Racks	1
2	Bike Stands	1
3	Bottle Cages	1
4	Cleaners	1
5	Fenders	1
6	Helmets	1
7	Hydration Packs	1

*This **Product Subcatrgories Dimension** table provides additional attributes about each **product subcategory***

STAR VS SNOWFLAKE SCHEMAS

Star Schema data model (Recommended)



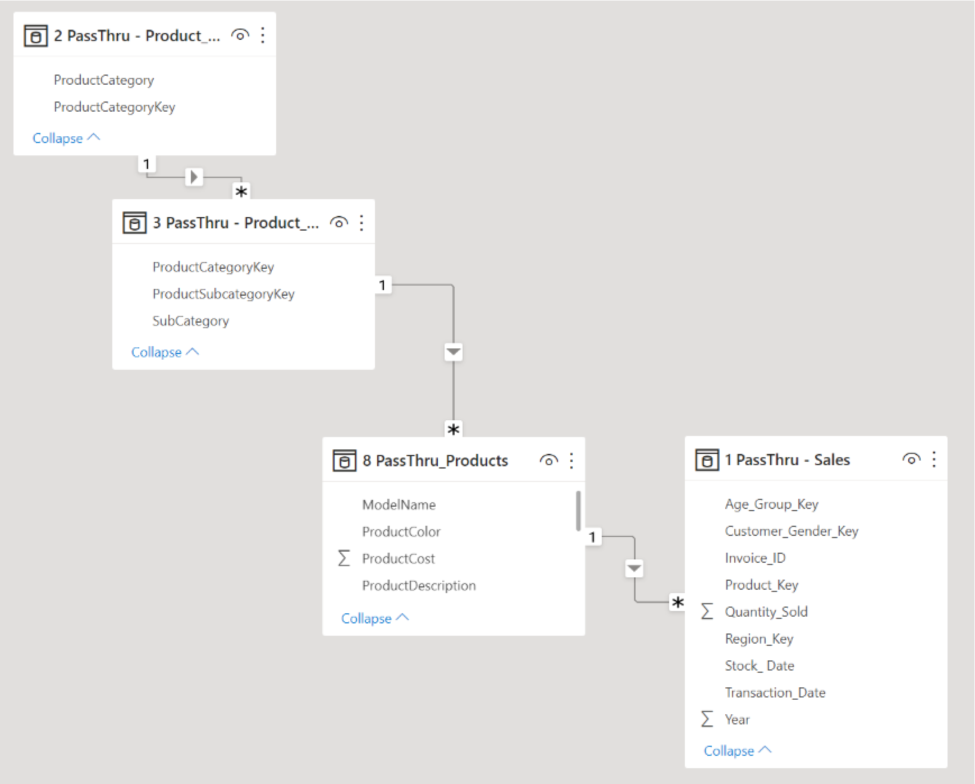
4 PassThru - Regions
- Continent
- Country
- EmailAddress
- Region
- SalesTerritoryKey
- TerritoryManager

5 PassThru - Gender
- CustomerGender
- GenderKey

1 PassThru - Sales
- Age_Group_Key
- Customer_Gender_Key
- Invoice_ID
- Product_Key
- ∑ Quantity_Sold
- Region_Key
- Stock_ Date
- Transaction_Date
- ∑ Year

This is a logical arrangement of a fact table surrounded by dimension tables. No dimension table has another dimension table connected to it.

Snowflake Schemadata model



2 PassThru - Product_...
- ProductCategory
- ProductCategoryKey

3 PassThru - Product_...
- ProductCategoryKey
- ProductSubcategoryKey
- SubCategory

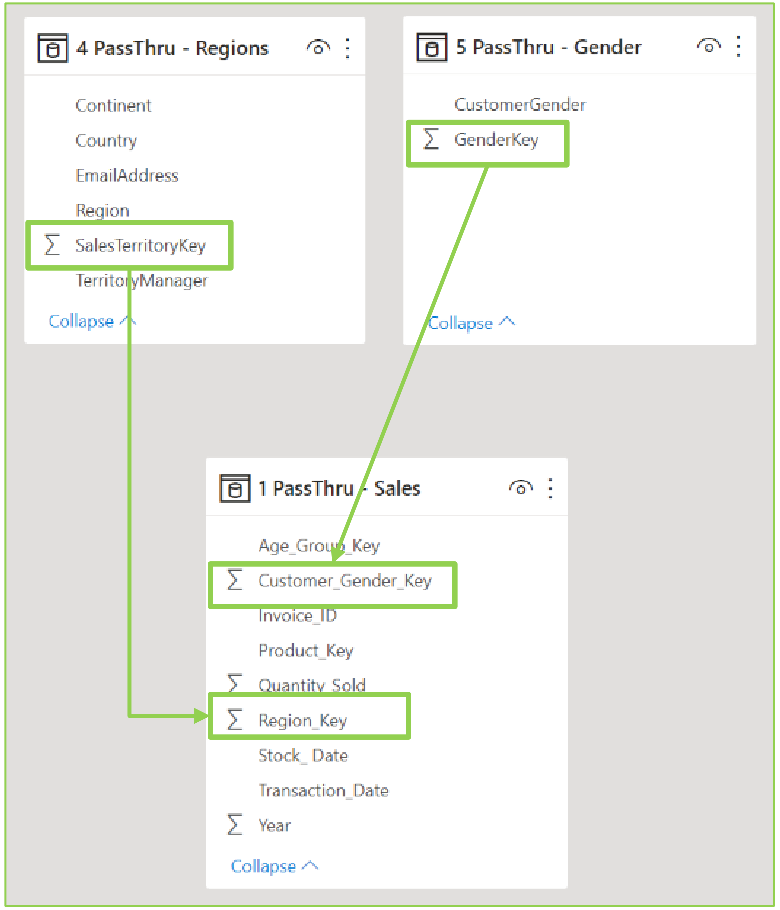
8 PassThru_Products
- ModelName
- ProductColor
- ∑ ProductCost
- ProductDescription

1 PassThru - Sales
- Age_Group_Key
- Customer_Gender_Key
- Invoice_ID
- Product_Key
- ∑ Quantity_Sold
- Region_Key
- Stock_ Date
- Transaction_Date
- ∑ Year

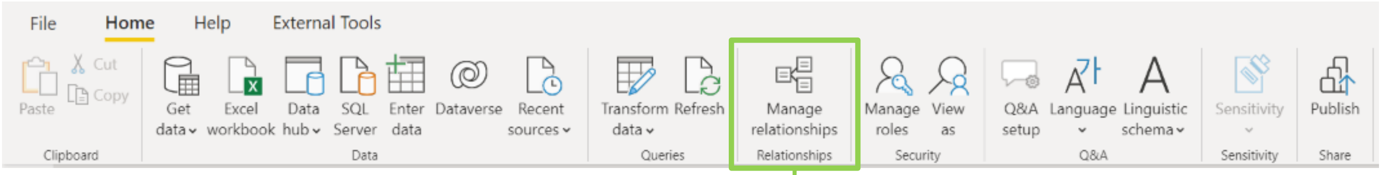
A logical arrangement of One fact table surrounded by dimension table which are in turn surrounded by dimension table

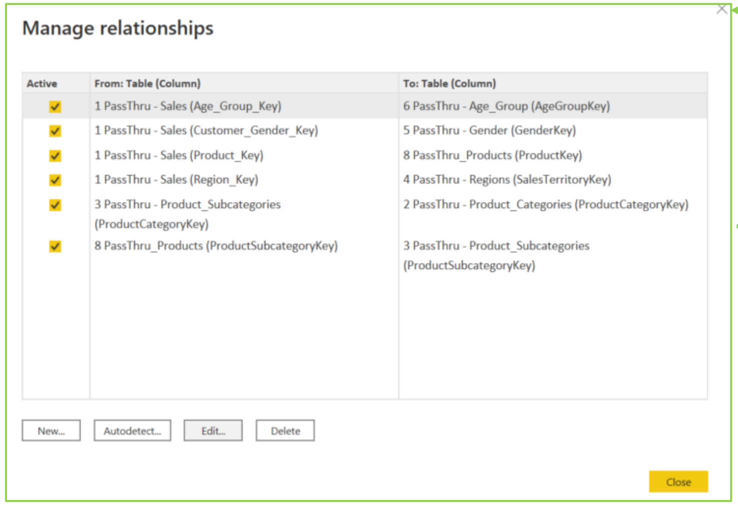
CREATE TABLE RELATIONSHIPS

Option 1: Click and drag to connect primary and foreign keys within the Modelpane



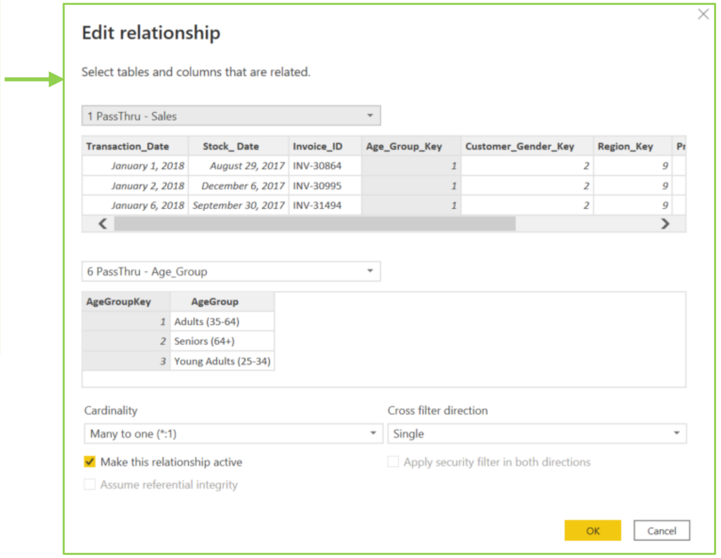
Option 2: Add or detect relationships using the "**Manage Relationships**" dialog box





Manage relationships

Active	From: Table (Column)	To: Table (Column)
☑	1 PassThru - Sales (Age_Group_Key)	6 PassThru - Age_Group (AgeGroupKey)
☑	1 PassThru - Sales (Customer_Gender_Key)	5 PassThru - Gender (GenderKey)
☑	1 PassThru - Sales (Product_Key)	8 PassThru_Products (ProductKey)
☑	1 PassThru - Sales (Region_Key)	4 PassThru - Regions (SalesTerritoryKey)
☑	3 PassThru - Product_Subcategories (ProductCategoryKey)	2 PassThru - Product_Categories (ProductCategoryKey)
☑	8 PassThru_Products (ProductSubcategoryKey)	3 PassThru - Product_Subcategories (ProductSubcategoryKey)



Edit relationship

Select tables and columns that are related.

1 PassThru - Sales

Transaction_Date	Stock_ Date	Invoice_ID	Age_Group_Key	Customer_Gender_Key	Region_Key	Pr
January 1, 2018	August 29, 2017	INV-30864	1	2	9	
January 2, 2018	December 6, 2017	INV-30995	1	2	9	
January 6, 2018	September 30, 2017	INV-31494	1	2	9	

6 PassThru - Age_Group

AgeGroupKey	AgeGroup
1	Adults (35-64)
2	Seniors (64+)
3	Young Adults (25-34)

Cardinality: Many to one (*:1)

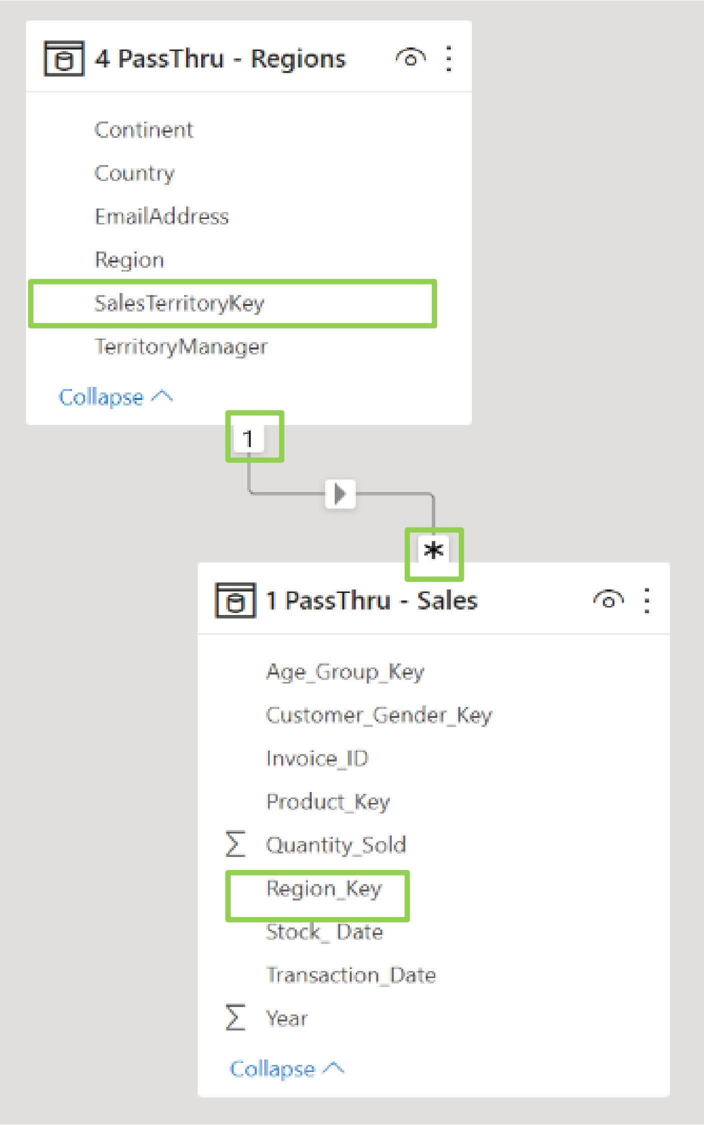
Cross filter direction: Single

☑ Make this relationship active

□ Apply security filter in both directions

□ Assume referential integrity

RELATIONSHIP CARDINALITY



Cardinality is the numerical relationship between rows of one table and rows in another.

In Power BI Modeling, there are three types

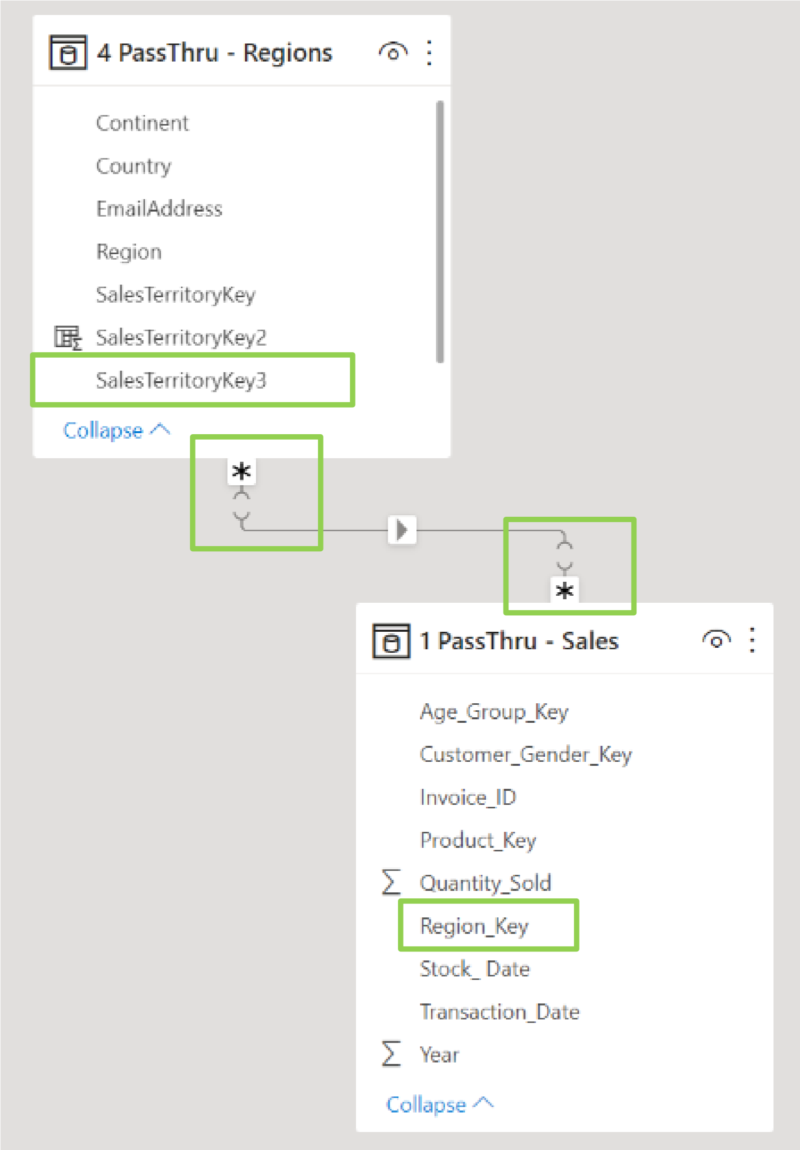
- **One to Many (Recommended)**
- One to One
- Many to Many

One to Many Cardinality means one row of a table (usually the primary key in the dimensional table) to many rows of another table (usually the foreign key in the fact table).

In this illustration, ONE row of each SalesTerritoryKey in the "PassThru – Regions" table (designated by the “1”), since each row contains attributes of a single region (Country, Continent…)

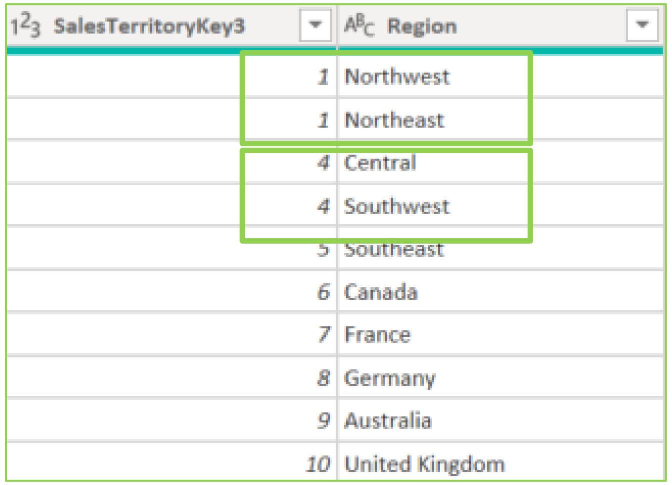
The MANY rows of each Region_Key in the Sales Fact table (designated by the asterisk *), since there are multiple sales associated with each region

RELATIONSHIP CARDINALITY (MANY – MANY)



Many to Many Cardinality means two or more rows of a table to many rows of another table. Not recommended at all.

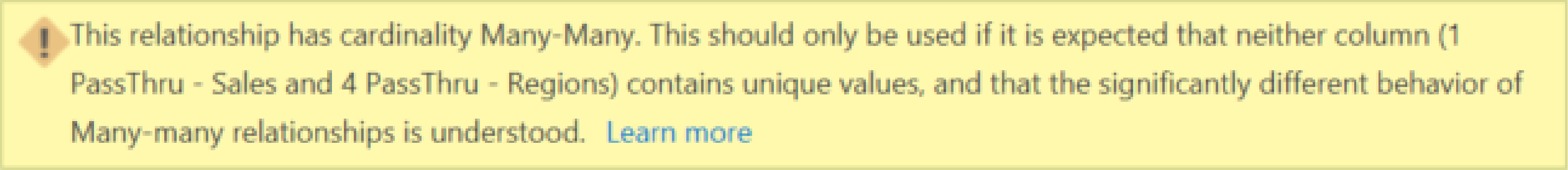
In this instance, we were able to connect the Region table (SalesTerritoryKey3) to the Sales (Region_Key). The problem is that there are many rows of each region ID in both tables



SalesTerritoryKey3	Region
1	Northwest
1	Northeast
4	Central
4	Southwest
5	Southeast
6	Canada
7	France
8	Germany
9	Australia
10	United Kingdom

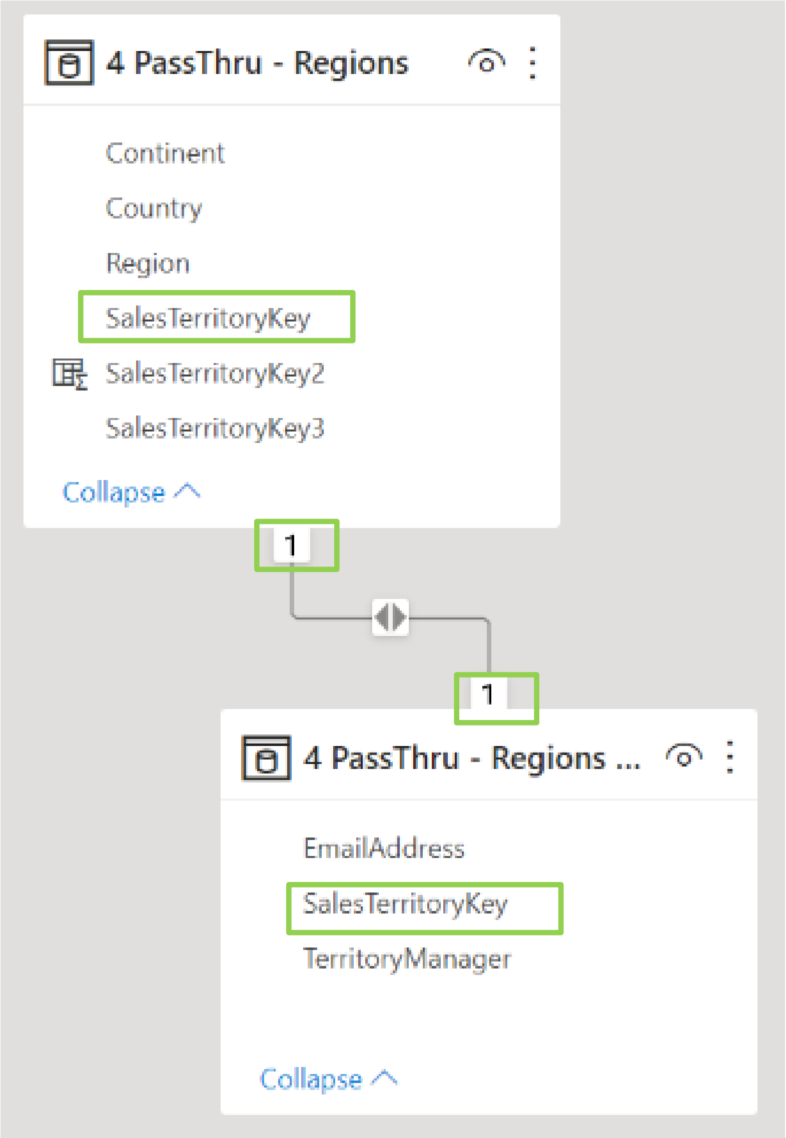
As you can see, there is no way to distinguish between Northwest and Northeast regions using the SalesTerritoryKey3. Similarly, we cannot distinguish between “Central” and “Southwest” regions.

Error displayed in the Edit Relationship Window

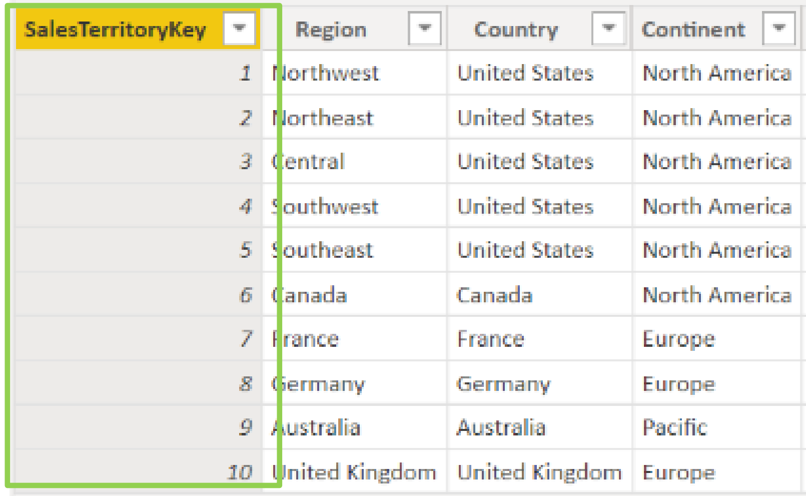


This relationship has cardinality Many-Many. This should only be used if it is expected that neither column (1 PassThru - Sales and 4 PassThru - Regions) contains unique values, and that the significantly different behavior of Many-many relationships is understood. Learn more

RELATIONSHIP CARDINALITY (ONE-ONE)

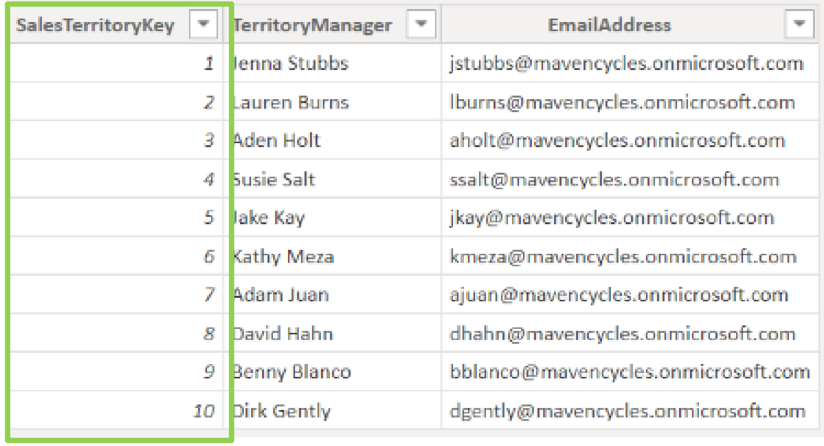


One to One Cardinality means one row of each key in a table connect to one row of similar key in another table. This is usually no efficient. It is better to merge the tables



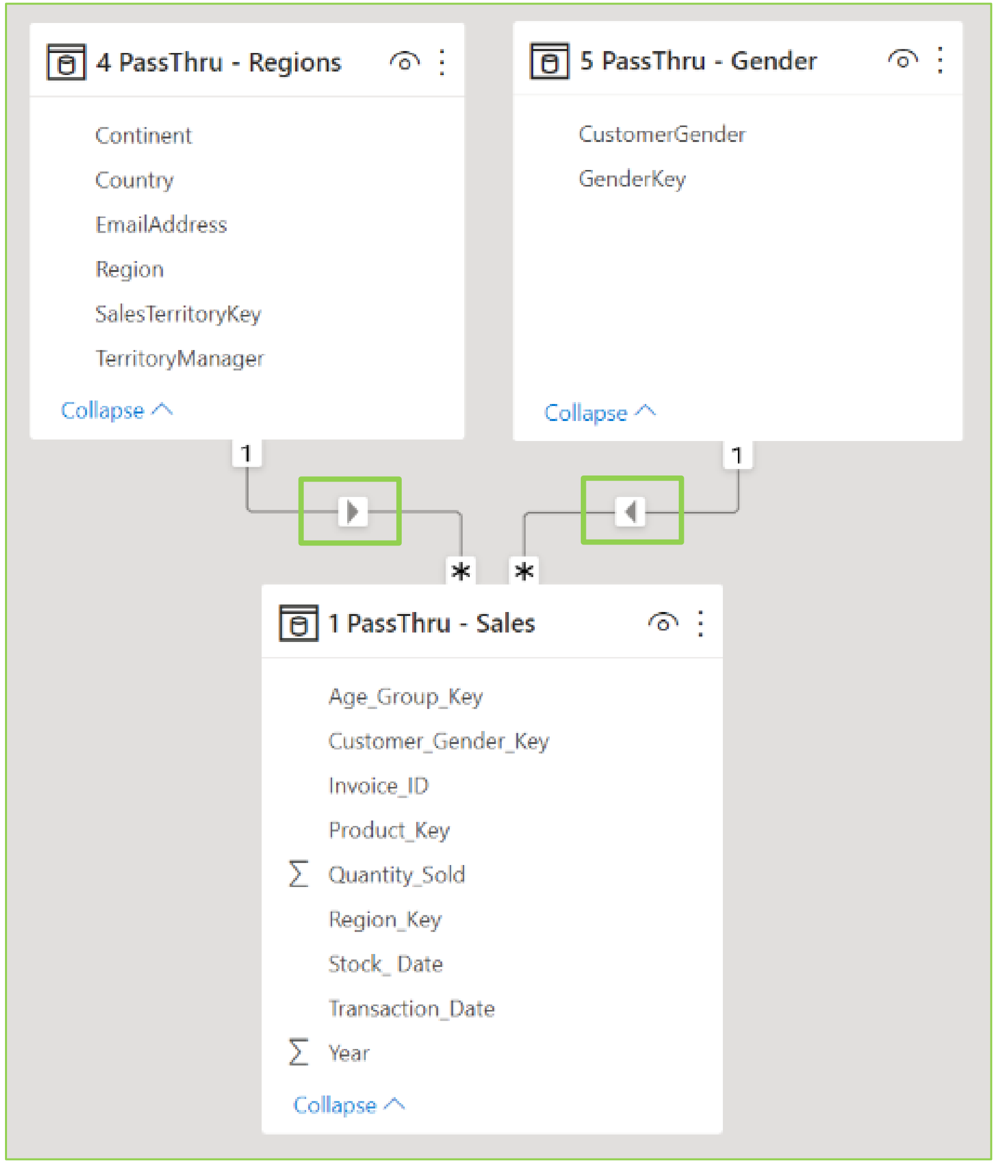
SalesTerritoryKey	Region	Country	Continent
1	Northwest	United States	North America
2	Northeast	United States	North America
3	Central	United States	North America
4	Southwest	United States	North America
5	Southeast	United States	North America
6	Canada	Canada	North America
7	France	France	Europe
8	Germany	Germany	Europe
9	Australia	Australia	Pacific
10	United Kingdom	United Kingdom	Europe

- *In this case, it is more efficient to merge the two tables into one single table*



SalesTerritoryKey	TerritoryManager	EmailAddress
1	Jenna Stubbs	jstubbs@mavencycles.onmicrosoft.com
2	Lauren Burns	lburns@mavencycles.onmicrosoft.com
3	Aden Holt	aholt@mavencycles.onmicrosoft.com
4	Susie Salt	ssalt@mavencycles.onmicrosoft.com
5	Jake Kay	jkay@mavencycles.onmicrosoft.com
6	Kathy Meza	kmeza@mavencycles.onmicrosoft.com
7	Adam Juan	ajuan@mavencycles.onmicrosoft.com
8	David Hahn	dhahn@mavencycles.onmicrosoft.com
9	Benny Blanco	bblanco@mavencycles.onmicrosoft.com
10	Dirk Gently	dgently@mavencycles.onmicrosoft.com

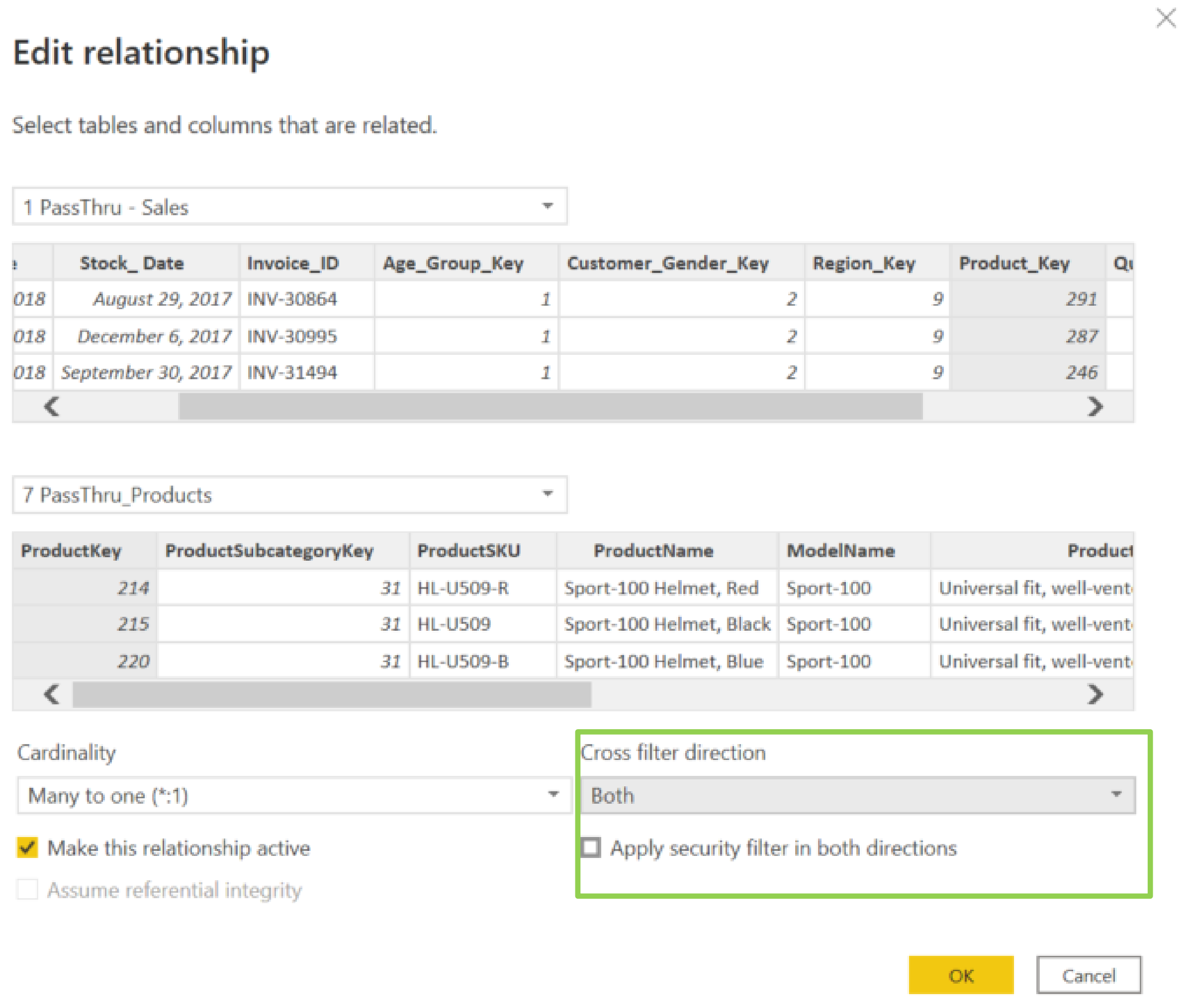
FILTER FLOW



- Filters usually flow in the direction of the arrow and not against it except when both directions are enabled.
- There are two types of Cross Filter direction: Single or Both
- **SINGLE** means filter flows in one direction only while **BOTH** implies filter flows in the two directions

Best Practice: Arrange your dimension tables above your fact tables in your model as a visual reminder that filters flow “downstream”.

FILTER FLOW (BOTH DIRECTIONS)



Edit relationship

Select tables and columns that are related.

1 PassThru - Sales

ː	Stock_ Date	Invoice_ID	Age_Group_Key	Customer_Gender_Key	Region_Key	Product_Key	Qu
018	August 29, 2017	INV-30864	1	2	9	291	
018	December 6, 2017	INV-30995	1	2	9	287	
018	September 30, 2017	INV-31494	1	2	9	246	

7 PassThru_Products

ProductKey	ProductSubcategoryKey	ProductSKU	ProductName	ModelName	Product
214	31	HL-U509-R	Sport-100 Helmet, Red	Sport-100	Universal fit, well-vent
215	31	HL-U509	Sport-100 Helmet, Black	Sport-100	Universal fit, well-vent
220	31	HL-U509-B	Sport-100 Helmet, Blue	Sport-100	Universal fit, well-vent

Cardinality: Many to one (*:1)

Cross filter direction: Both

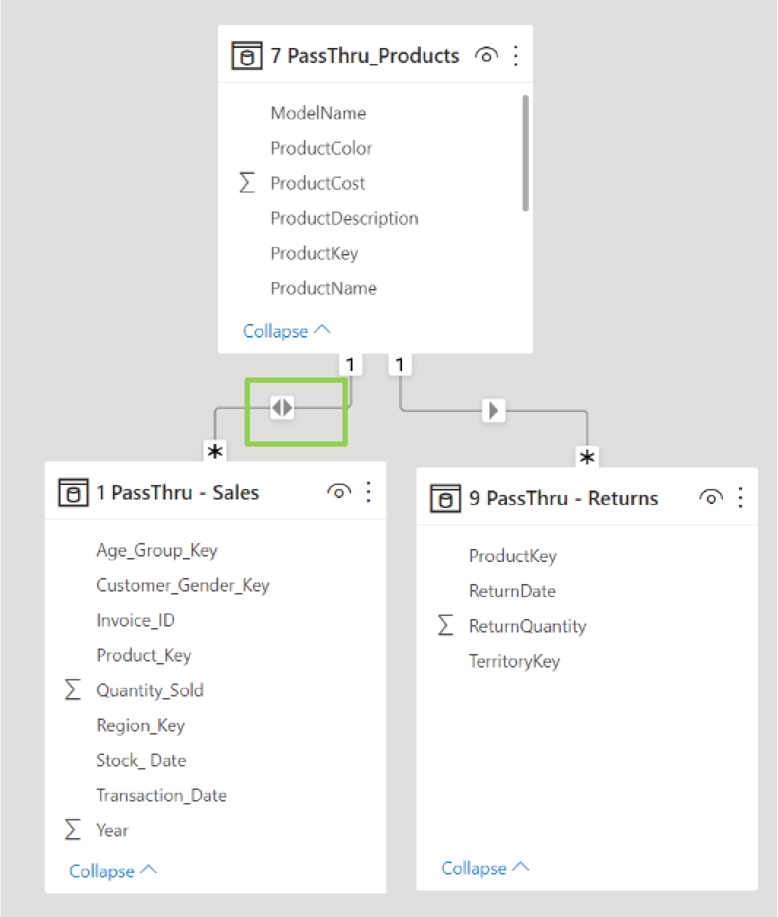
☑ Make this relationship active

□ Apply security filter in both directions

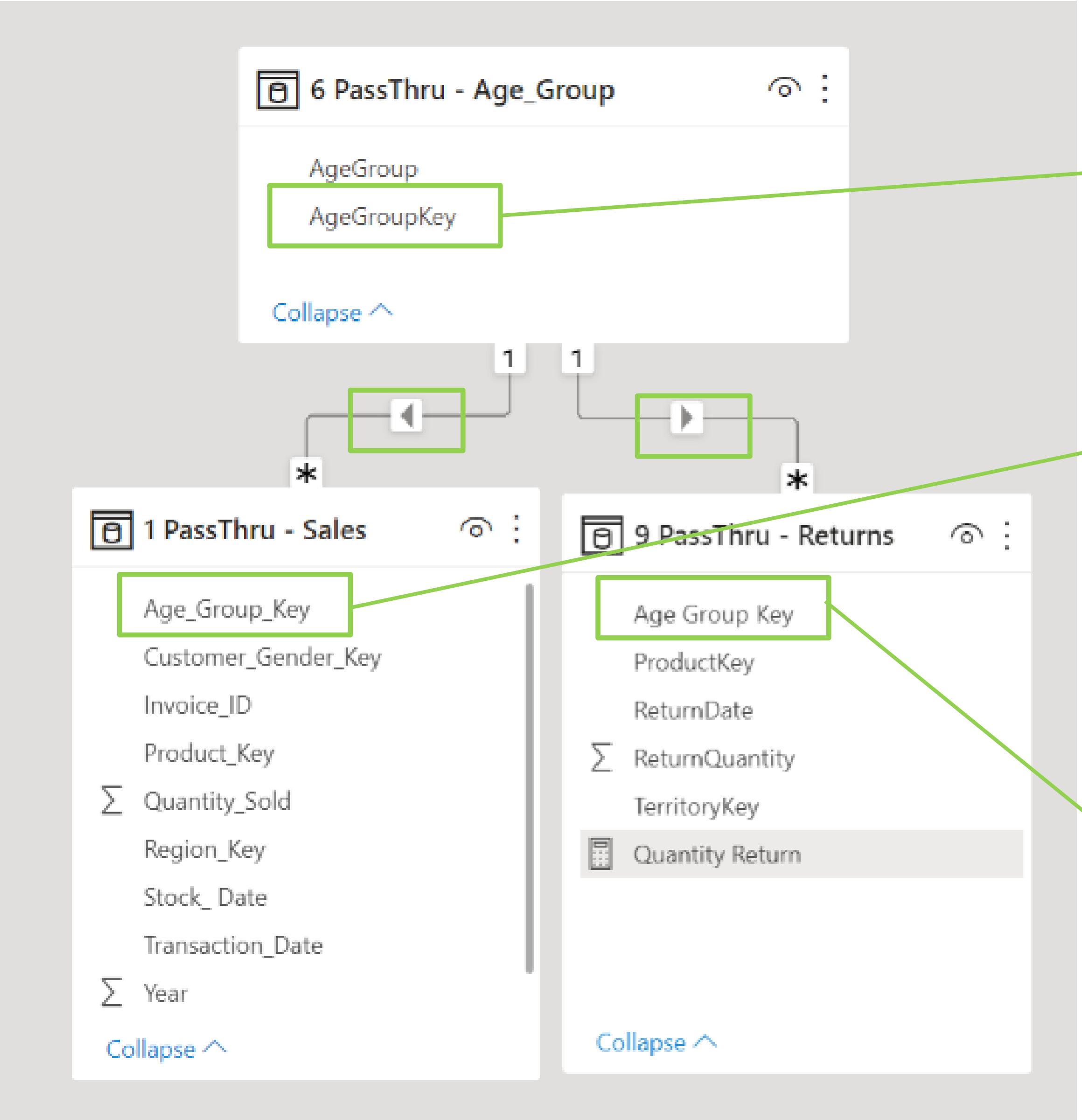
□ Assume referential integrity

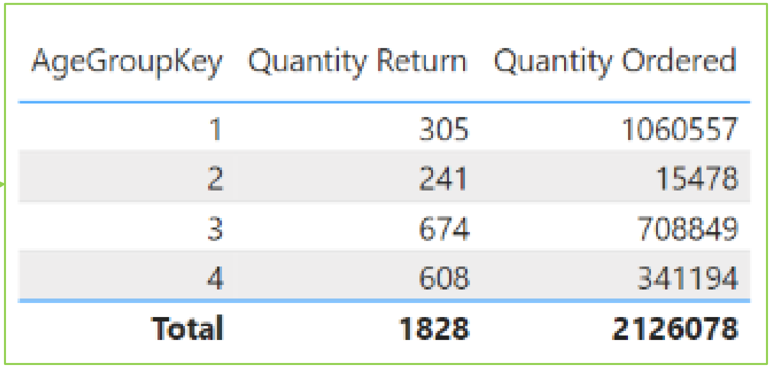
Under the Cross Filter direction, change the filter context direction from Single to Both. This implies that both tables can filter each other.

We will consider few examples of this concept.

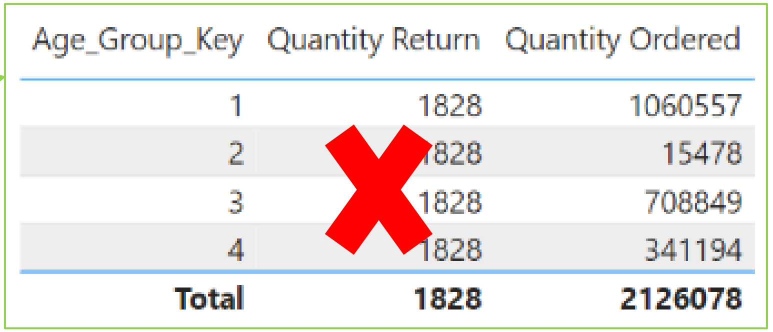


FILTER FLOW (BOTH DIRECTIONS)

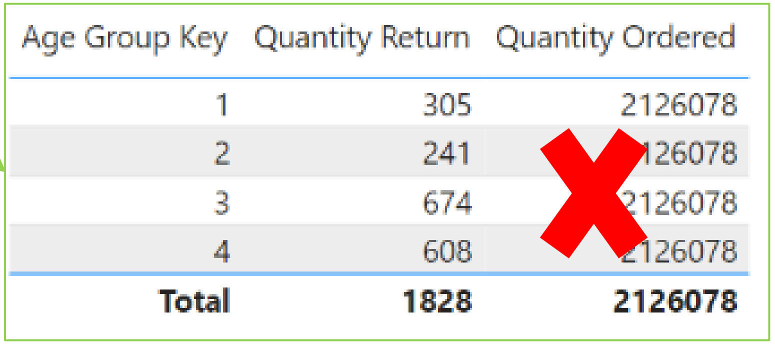




AgeGroupKey	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**

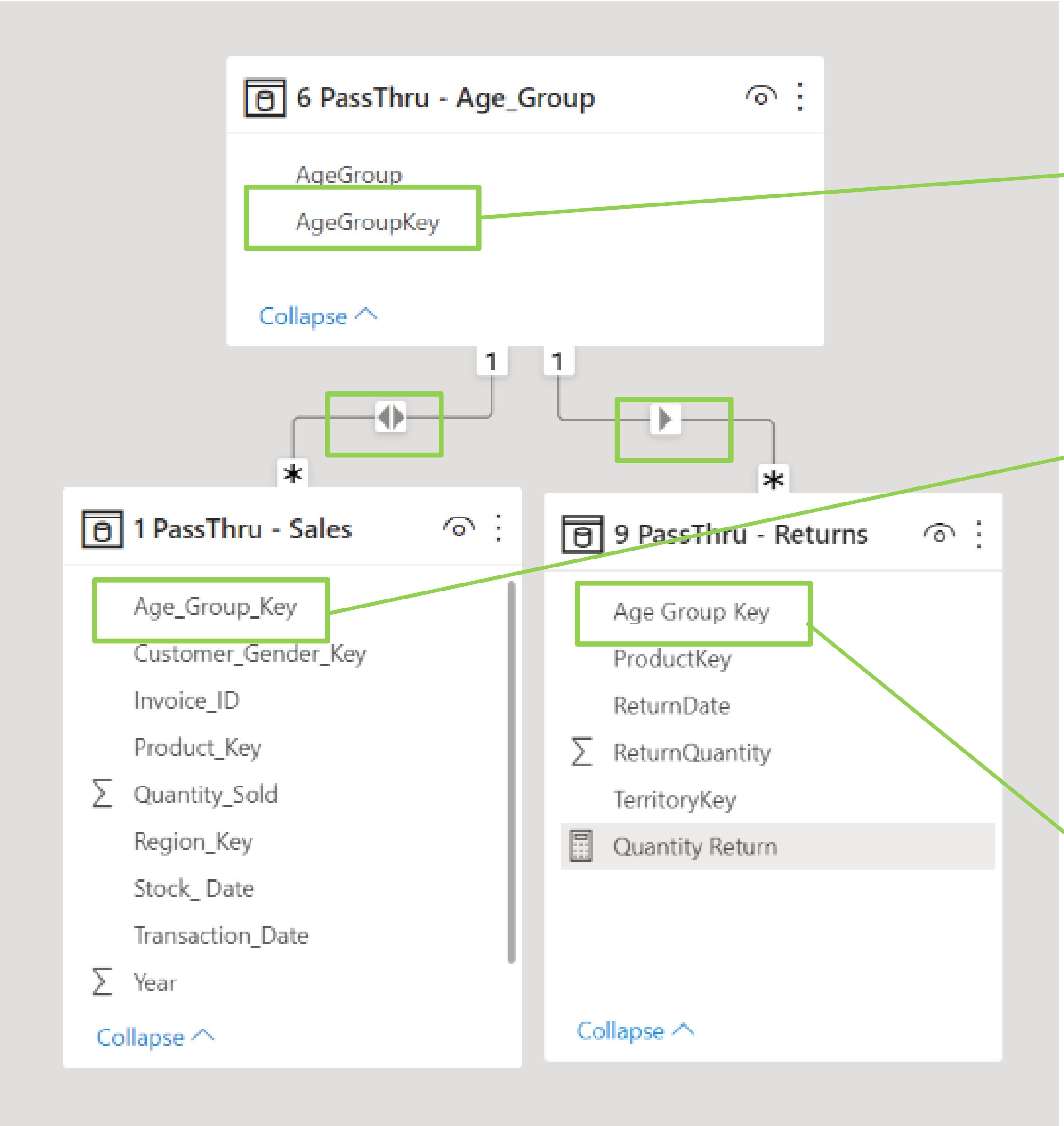


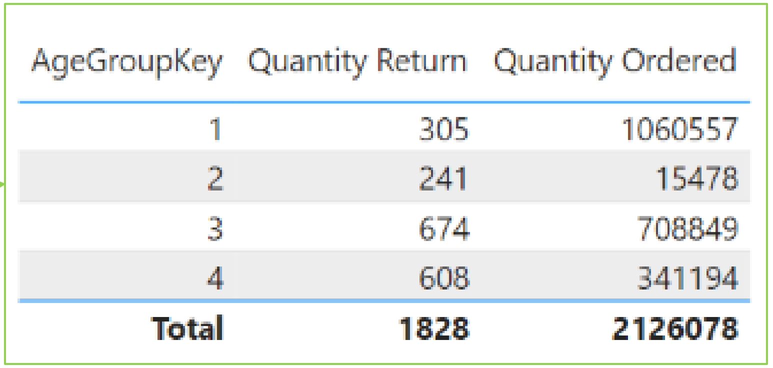
Age_Group_Key	Quantity Return	Quantity Ordered
1	1828	1060557
2	1828	15478
3	1828	708849
4	1828	341194
Total	**1828**	**2126078**



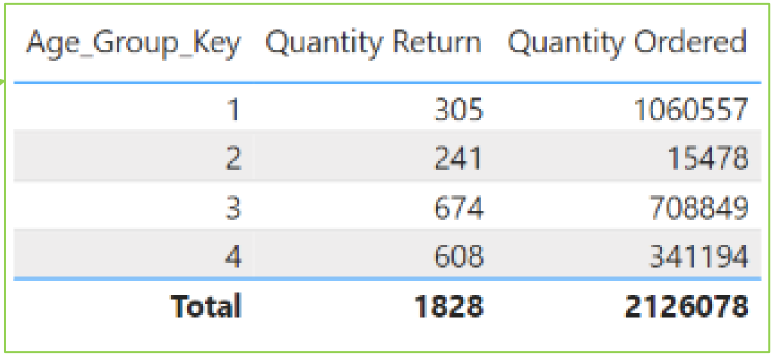
Age Group Key	Quantity Return	Quantity Ordered
1	305	2126078
2	241	2126078
3	674	2126078
4	608	2126078
Total	**1828**	**2126078**

FILTER FLOW (BOTH DIRECTIONS)

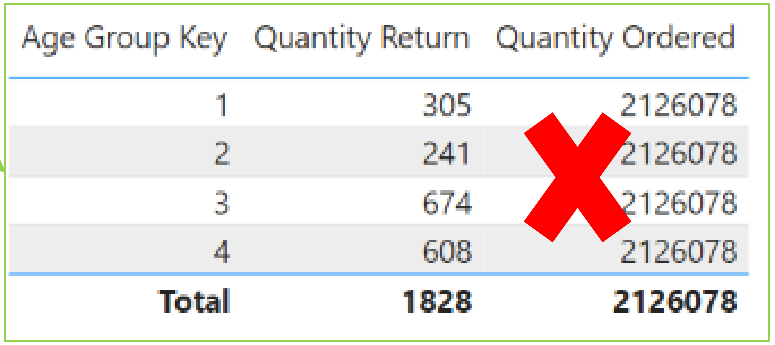




AgeGroupKey	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**

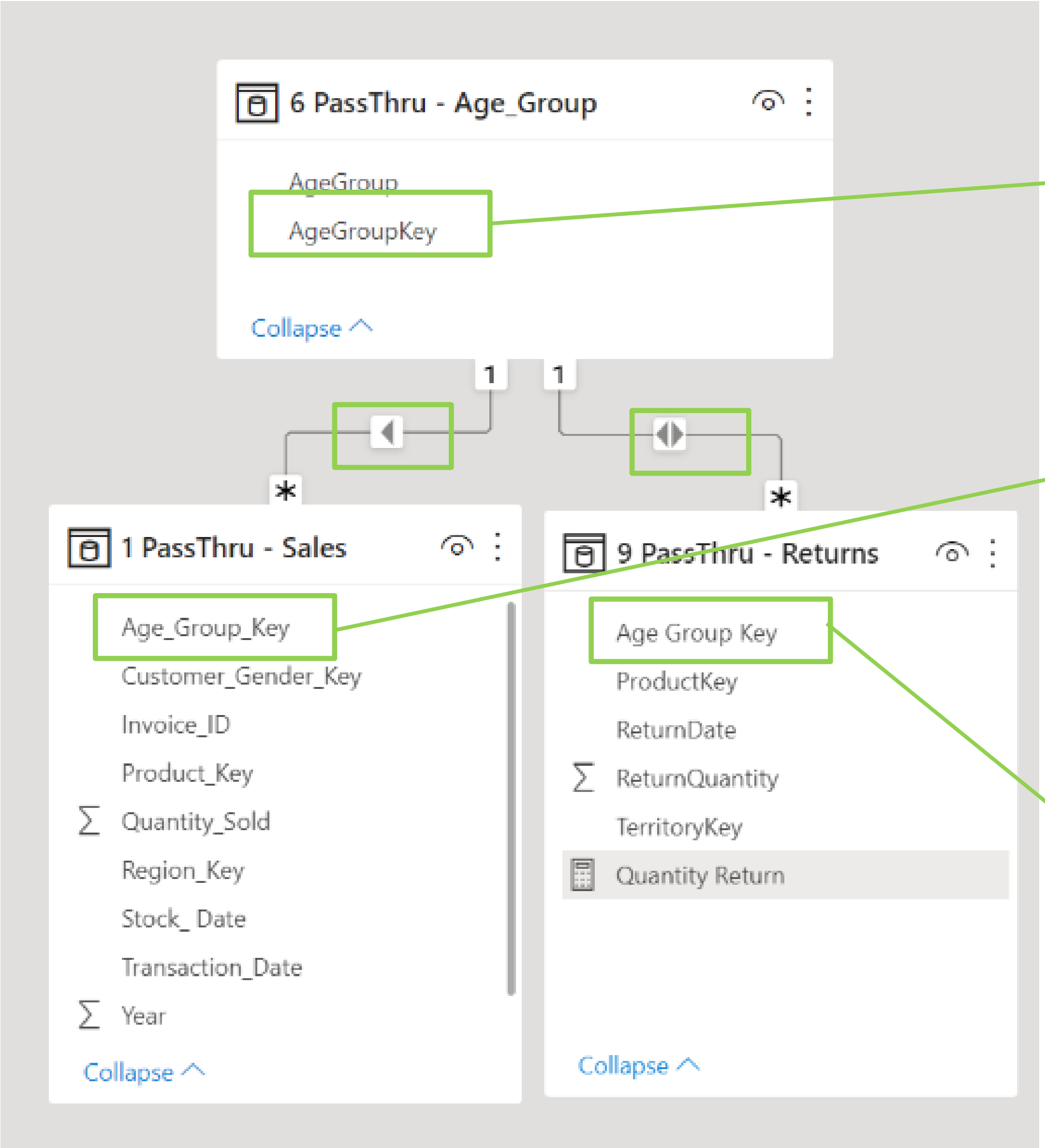


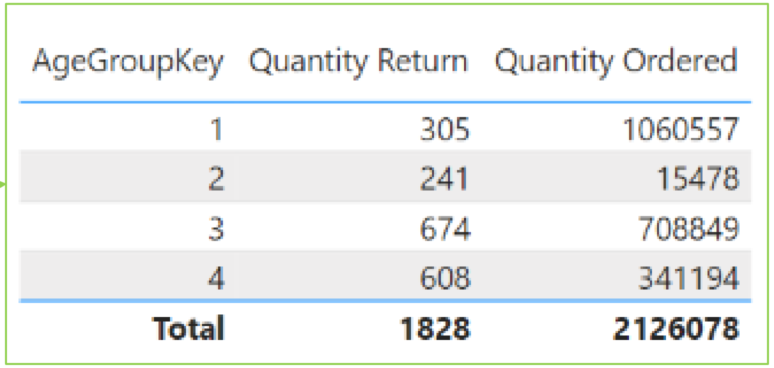
Age_Group_Key	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**



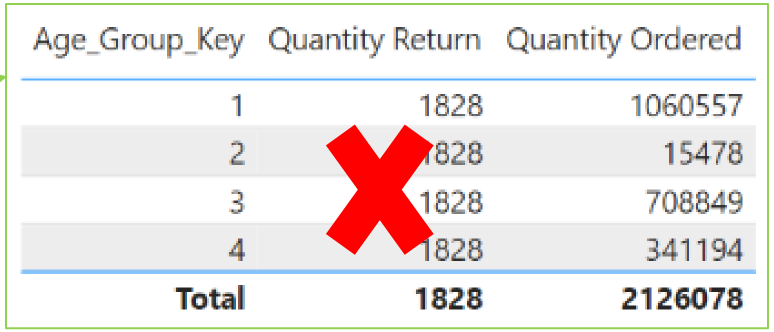
Age Group Key	Quantity Return	Quantity Ordered
1	305	2126078
2	241	2126078
3	674	2126078
4	608	2126078
Total	**1828**	**2126078**

FILTER FLOW (BOTH DIRECTIONS)

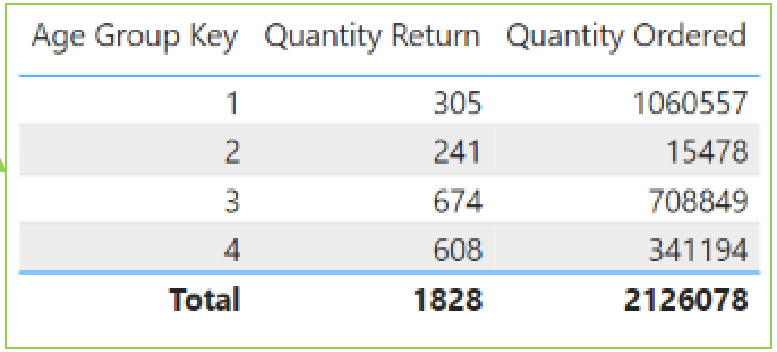




AgeGroupKey	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**

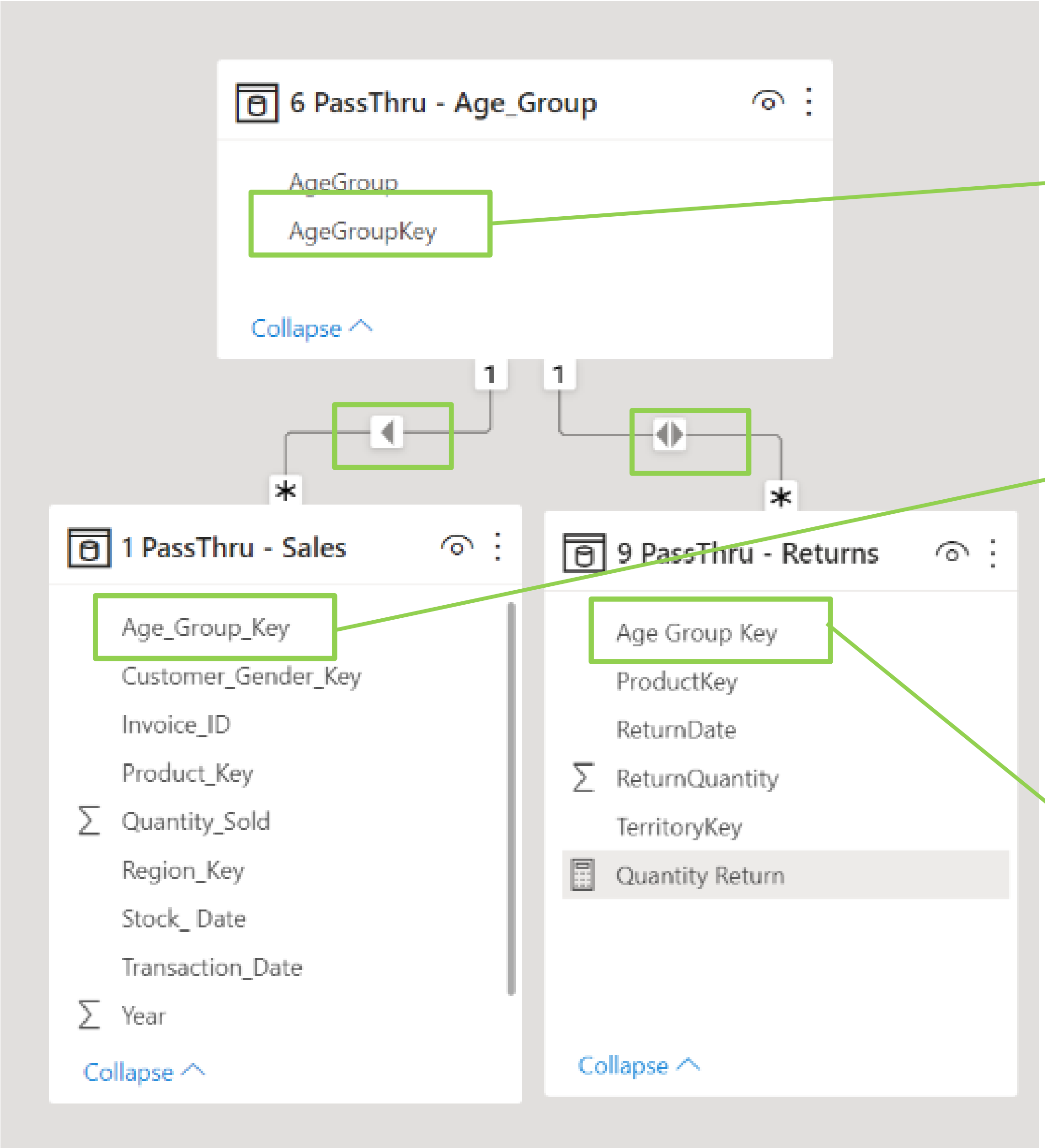


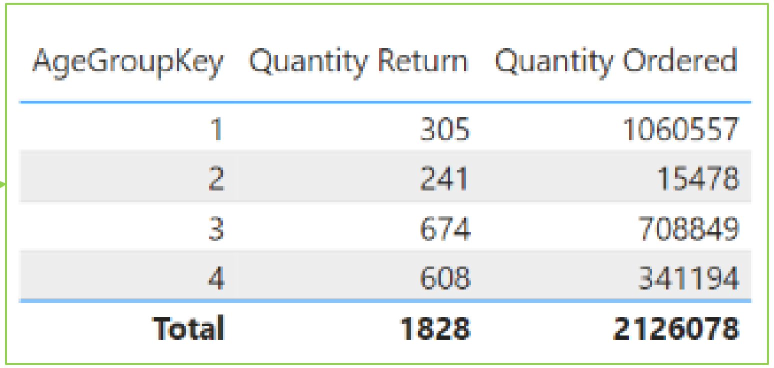
Age_Group_Key	Quantity Return	Quantity Ordered
1	1828	1060557
2	1828	15478
3	1828	708849
4	1828	341194
Total	**1828**	**2126078**



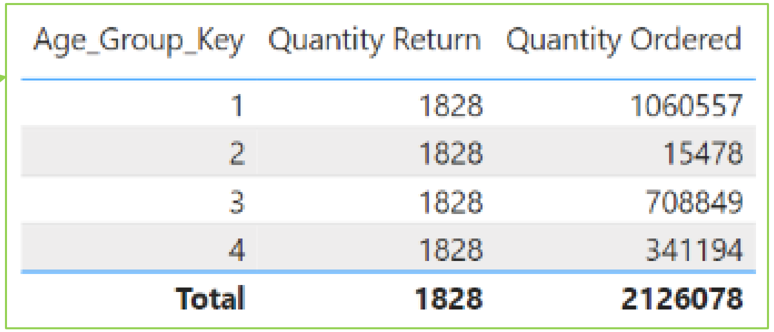
Age Group Key	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**

FILTER FLOW (BOTH DIRECTIONS)

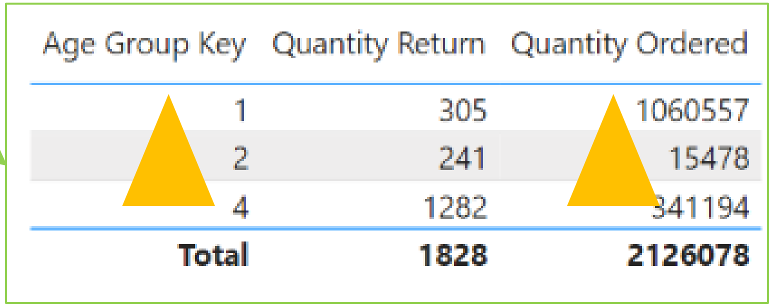




AgeGroupKey	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
3	674	708849
4	608	341194
Total	**1828**	**2126078**

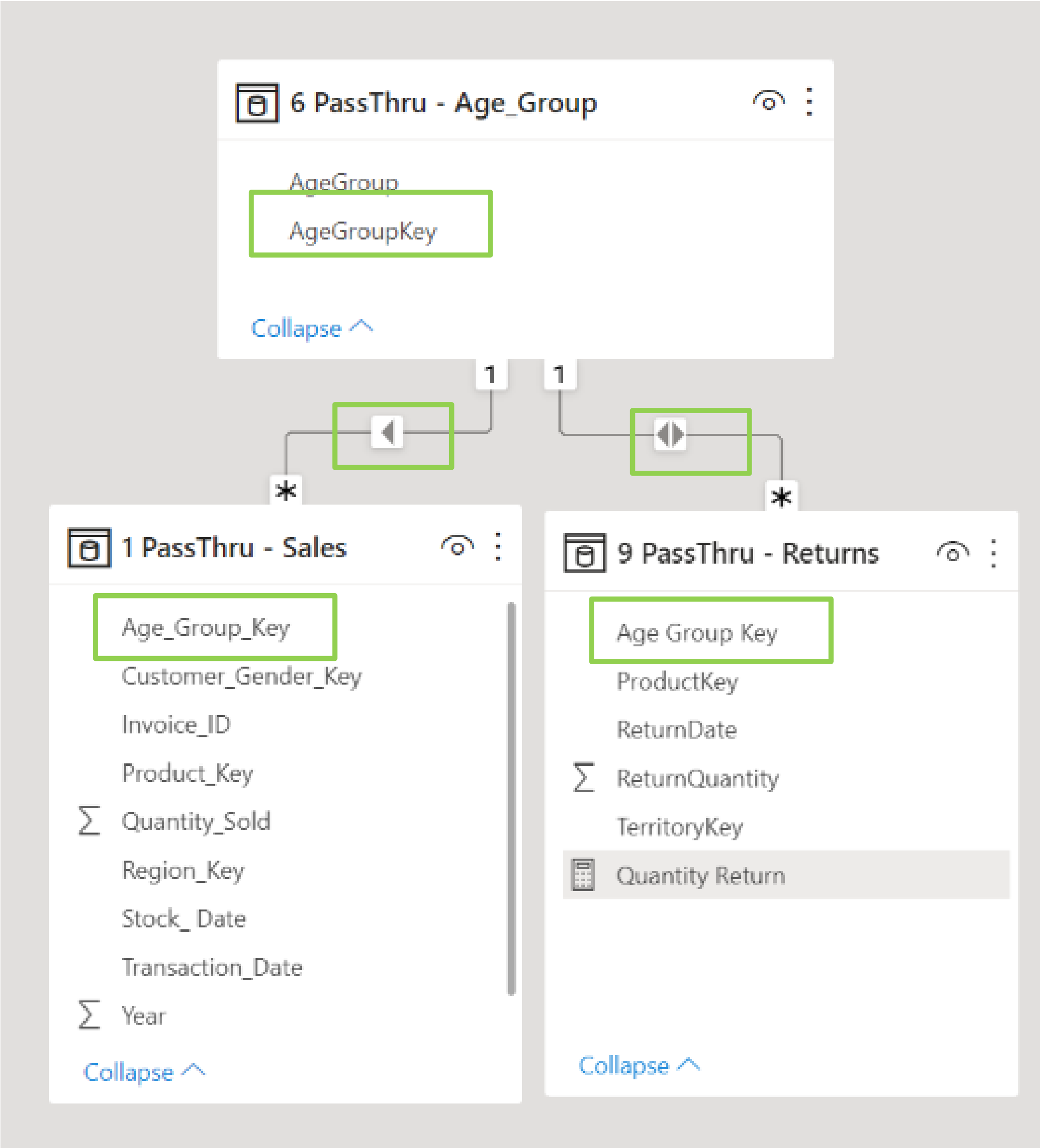


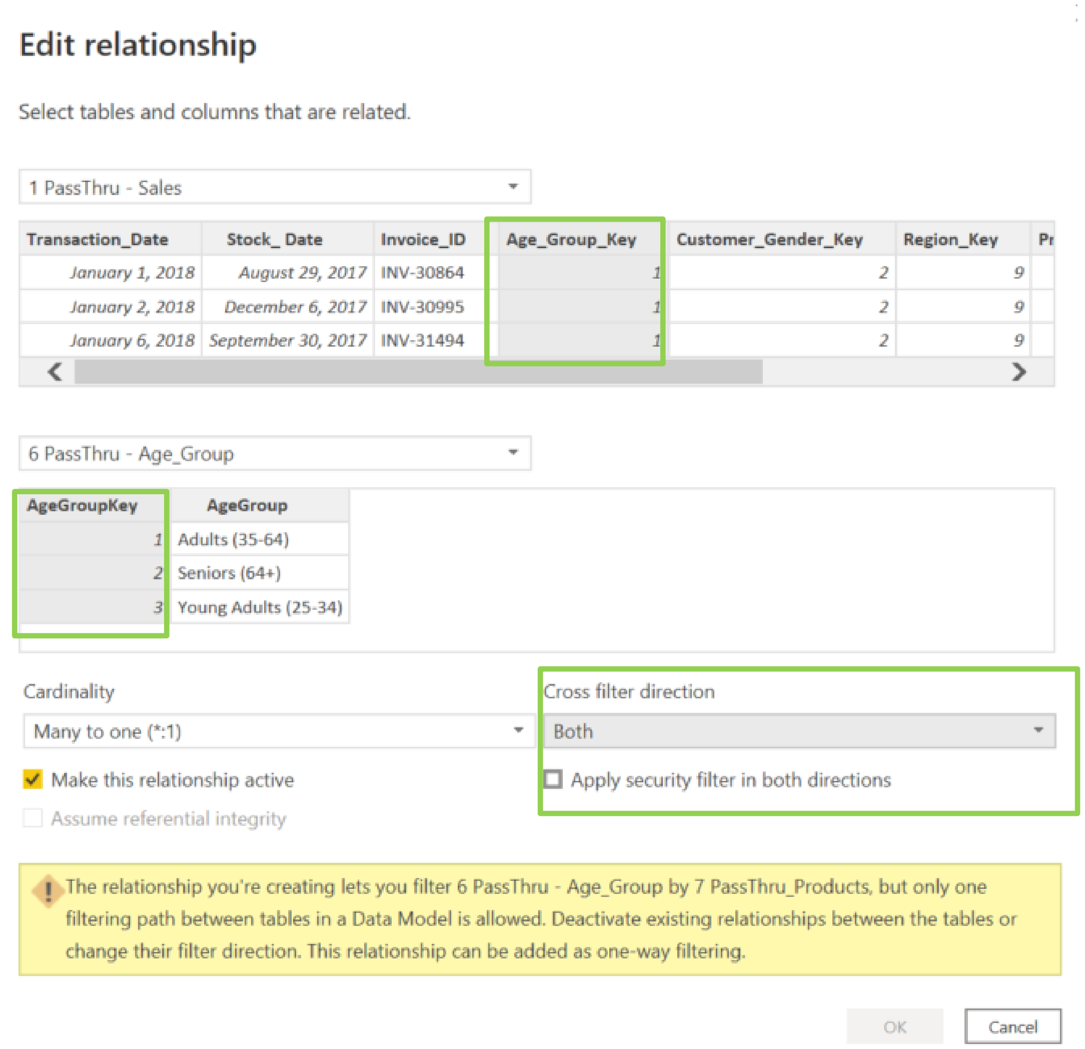
Age_Group_Key	Quantity Return	Quantity Ordered
1	1828	1060557
2	1828	15478
3	1828	708849
4	1828	341194
Total	**1828**	**2126078**



Age Group Key	Quantity Return	Quantity Ordered
1	305	1060557
2	241	15478
4	1282	341194
Total	**1828**	**2126078**

FILTER FLOW (BOTH DIRECTIONS)





Edit relationship

Select tables and columns that are related.

1 PassThru - Sales

Transaction_Date	Stock_ Date	Invoice_ID	Age_Group_Key	Customer_Gender_Key	Region_Key	Pr
January 1, 2018	*August 29, 2017*	INV-30864	*1*	*2*	*9*	
January 2, 2018	*December 6, 2017*	INV-30995	*1*	*2*	*9*	
January 6, 2018	*September 30, 2017*	INV-31494	*1*	*2*	*9*	

6 PassThru - Age_Group

AgeGroupKey	AgeGroup
1	Adults (35-64)
2	Seniors (64+)
3	Young Adults (25-34)

Cardinality: Many to one (*:1)

Cross filter direction: Both

☑ Make this relationship active

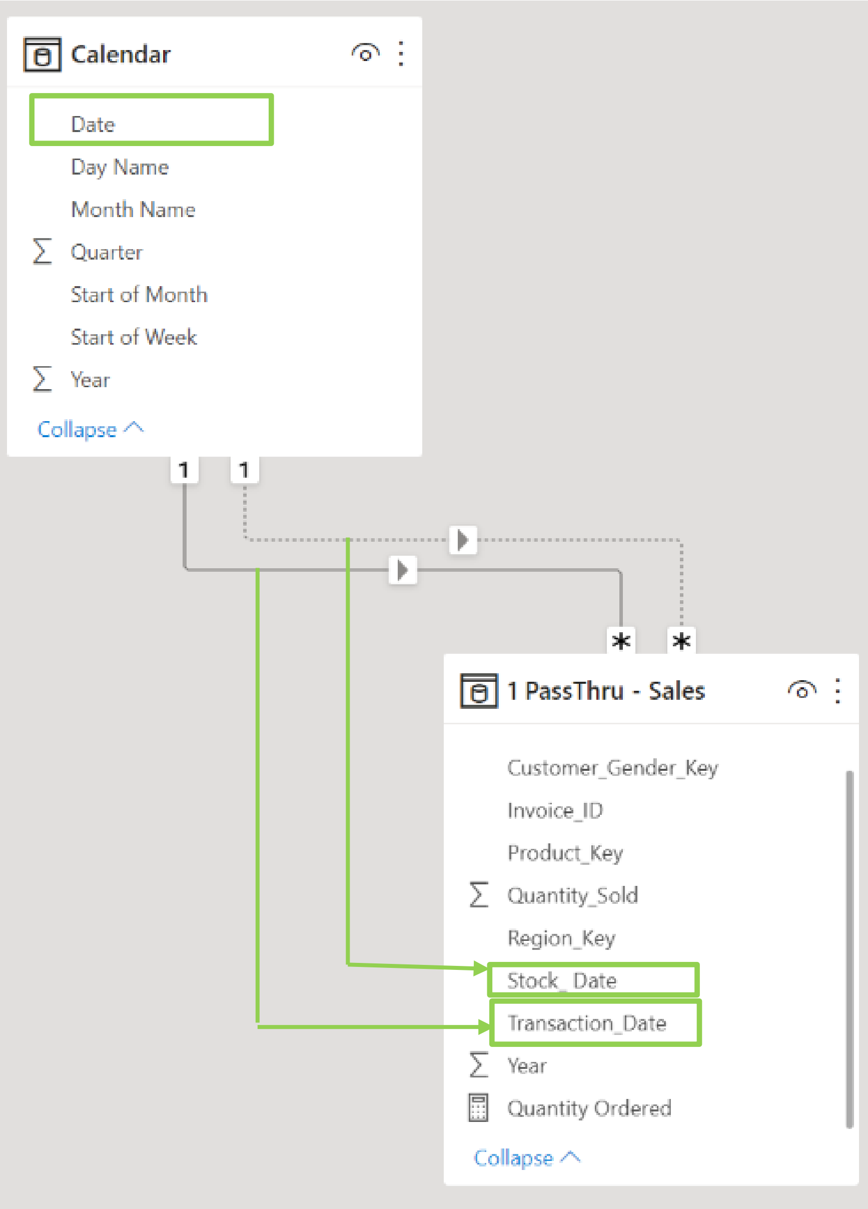
□ Apply security filter in both directions

□ Assume referential integrity

The relationship you're creating lets you filter 6 PassThru - Age_Group by 7 PassThru_Products, but only one filtering path between tables in a Data Model is allowed. Deactivate existing relationships between the tables or change their filter direction. This relationship can be added as one-way filtering.

Note: Design your models with 1-to-Many cardinality and one-way filters , except more complex relationships are necessary

ACTIVE & INACTIVE RELATIONSHIPS



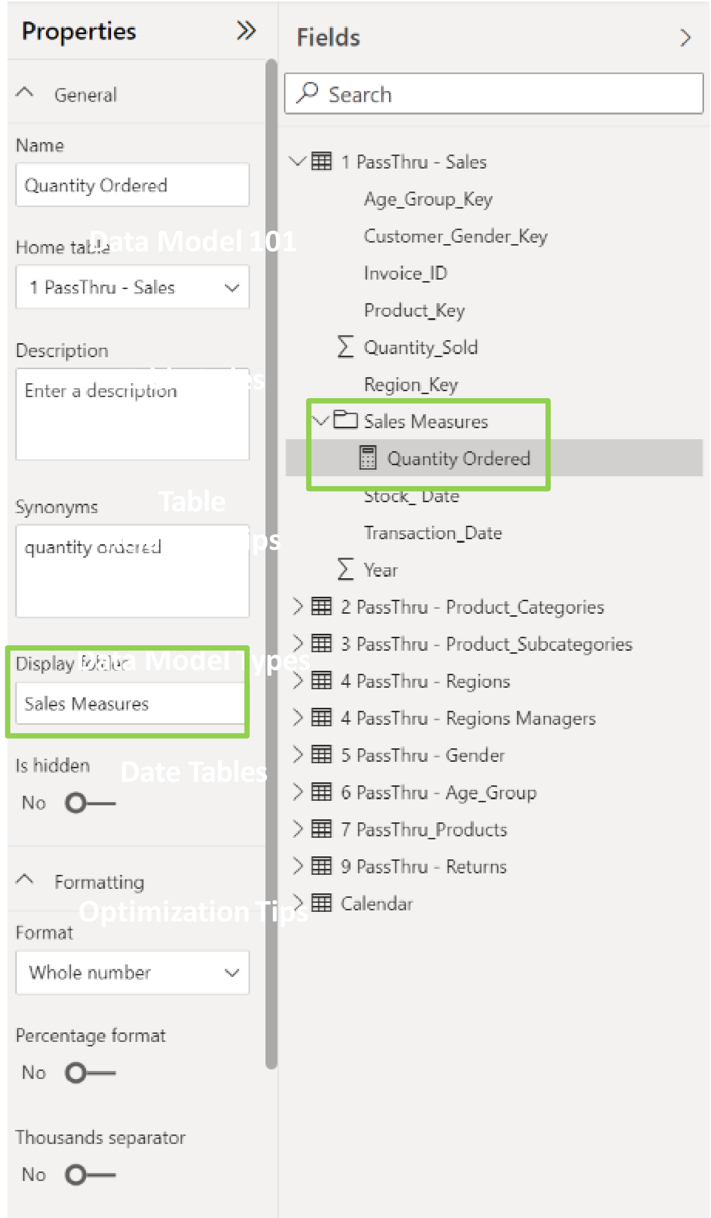
Two main types of physical table relationships: **Active** & **Inactive**

Active relationships propagate filters to other tables.

Inactive relationship only propagate filters when a DAX expression activates (uses) the relationship. The DAX functions include RELATED, RELATEDTABLE OR USERELATIONSHIP.

Note: Cross filter relationships can be enabled using DAX expression like CROSSFILTER.

ORGANIZE THE FIELDS IN THE TABLES



Properties

General

Name: Quantity Ordered

Home table: 1 PassThru - Sales

Description: Enter a description

Synonyms: quantity ordered

Display folder: Sales Measures

Is hidden: No

Formatting

Format: Whole number

Percentage format: No

Thousands separator: No

Fields

Search

- 1 PassThru - Sales
  - Age_Group_Key
  - Customer_Gender_Key
  - Invoice_ID
  - Product_Key
  - ∑ Quantity_Sold
  - Region_Key
  - Sales Measures
    - Quantity Ordered
  - Stock_ Date
  - Transaction_Date
  - ∑ Year
- 2 PassThru - Product_Categories
- 3 PassThru - Product_Subcategories
- 4 PassThru - Regions
- 4 PassThru - Regions Managers
- 5 PassThru - Gender
- 6 PassThru - Age_Group
- 7 PassThru_Products
- 9 PassThru - Returns
- Calendar

Use the Display folder to segmentize your fields for proper organization. This is very useful for grouping measures for easy identification.

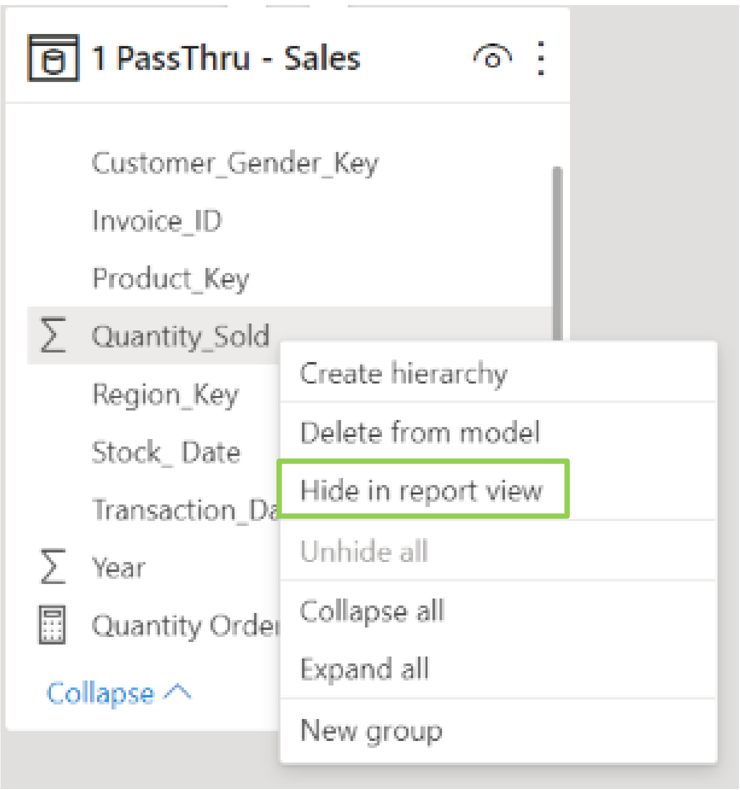
In this example, the Quantity Ordered was added to the Sales Measures folder.

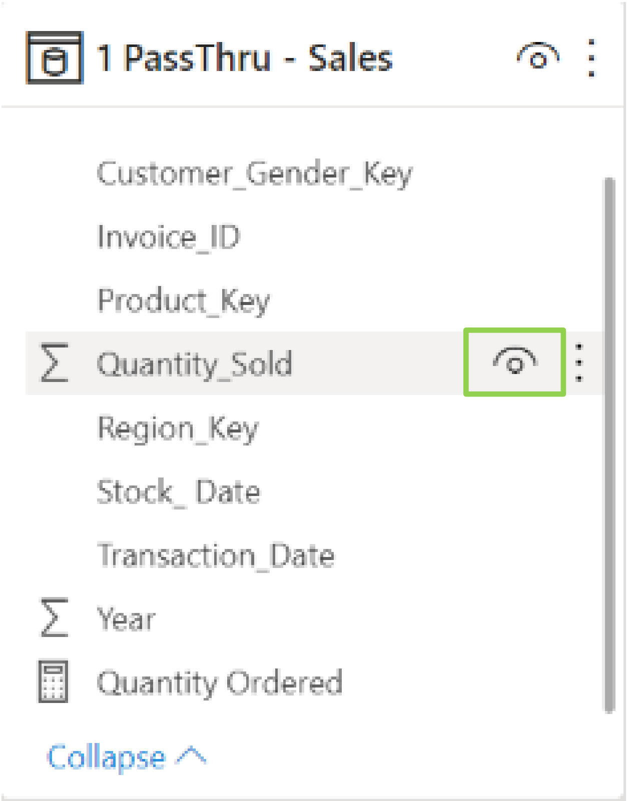
The process involve

- Select the column you want to add to a folder
- Type the name of the folder in the display folder

HIDE FIELDS FROM REPORT VIEW

Two ways to hide fields in report view





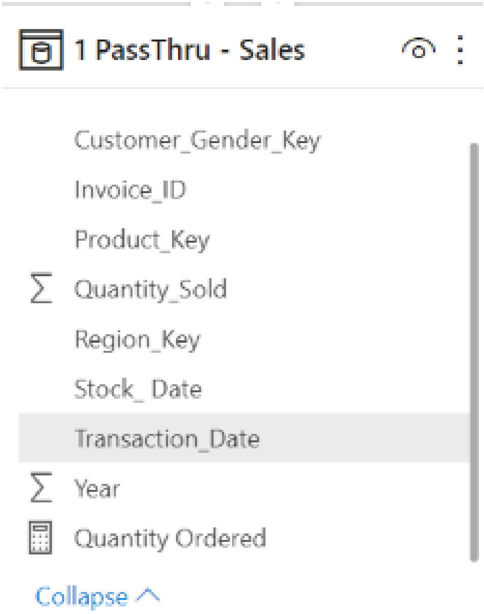
Hiding fields from Report View makes them inaccessible from the Report tab (*although they can still be accessed within the Data or Relationships views*)

This is commonly used to prevent users from filtering using invalid fields, or to hide irrelevant metrics from view

Note:
Hide the foreign key columns in your fact tables to force users to filter using the primary keys in the lookup tables

AUTOMATIC DATE TABLES

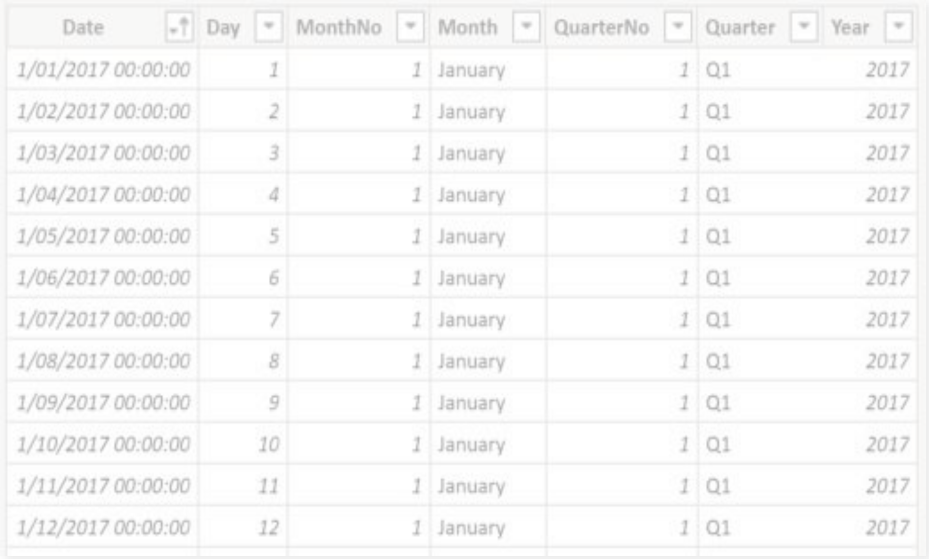
- By default, Power BI automatically creates a hidden date table for any table that contains a Date or DateTime column on the one-side of a relationship
- Auto-generated calendars include all dates through the end of the year, regardless of the actual date range in the table



1 PassThru - Sales

- Customer_Gender_Key
- Invoice_ID
- Product_Key
- ∑ Quantity_Sold
- Region_Key
- Stock_ Date
- Transaction_Date
- ∑ Year
- Quantity Ordered

Collapse ^



Date	Day	MonthNo	Month	QuarterNo	Quarter	Year
1/01/2017 00:00:00	1	1	January	1	Q1	2017
1/02/2017 00:00:00	2	1	January	1	Q1	2017
1/03/2017 00:00:00	3	1	January	1	Q1	2017
1/04/2017 00:00:00	4	1	January	1	Q1	2017
1/05/2017 00:00:00	5	1	January	1	Q1	2017
1/06/2017 00:00:00	6	1	January	1	Q1	2017
1/07/2017 00:00:00	7	1	January	1	Q1	2017
1/08/2017 00:00:00	8	1	January	1	Q1	2017
1/09/2017 00:00:00	9	1	January	1	Q1	2017
1/10/2017 00:00:00	10	1	January	1	Q1	2017
1/11/2017 00:00:00	11	1	January	1	Q1	2017
1/12/2017 00:00:00	12	1	January	1	Q1	2017

*Automatically creates a **hidden** date table containing all these columns*

AUTOMATIC DATE TABLES (CONTINUES)

ADVANTAGES:

- Automatically generated
- Enables (some) time intelligence functionality by default
- Simplifies data model creation and management
- Does not require an advanced understanding of DAX

DISADVANTAGES:

- Hidden from view, cannot be modified/customized
- Generated for every date field across every lookup/dimension table *(**bloats model size**)*
- Can’t be enabled or disabled at the table-level
- Hierarchies aren’t automatically generated *(if grouped by month, would summarize that month across ALL years)*
- Each automatic date table can *only* filter the table it corresponds to *(cannot traverse table relationships)*

Note: Turn OFF the auto date/time feature in Power BI Desktop and either import a date dimension table or create your own using CALENDAR functions

DATE TABLE REQUIREMENTS

If you import or create your own date table, ensure it must meet these requirements:

- Must contain all the days for all years represented in your fact tables
- Must have at least one field set as a Date or DateTime datatype
- Cannot contain duplicate dates or datetime values
- If using a time component within a date column, all times must be identical
- Should be marked as a date table
- If Time is present in your date field, split the time component into a new column (this adheres to relationship requirements and decreases column cardinality)

DATABASE NORMALIZATION

Normalization is the process of organizing the tables and columns in a relational database to reduce redundancy and preserve data integrity. It’s commonly used to:

- Eliminate redundant data to decrease table sizes and improve processing speed & efficiency
- Minimize errors and anomalies from data modifications (inserting, updating or deleting records)
- Simplify queries and structure the database for meaningful analysis

TIP: In a normalized database, each table should serve a ***distinct*** and ***specific*** purpose (*i.e product information, dates, transaction records, customer attributes, etc.*)

BEST PRACTICES: DATA MODELING

- **Focus on building a normalized model from the start**

  Make sure that each table in your model serves a single, distinct purpose

  Use relationships vs. merged tables; long & narrow tables are better than short & wide

- **Organize lookup tables *above* data tables in the diagram view**

  This serves as a visual reminder that filters flow “downstream”

- **Avoid complex cross-filtering unless absolutely necessary**

  Don’t use two-way filters when 1-way filters will get the job done

- **Hide fields from report view to prevent invalid filter context**

  Recommend hiding foreign keys from data tables, so that users can only access valid fields

THANK YOU FOR LISTENING

Q & A