CREATING A DATA MODEL

CREATING A DATA MODEL

In this section we'll cover **foundational data modeling topics** like normalization, fact and dimension tables, primary and foreign keys, relationship cardinality and filter flow

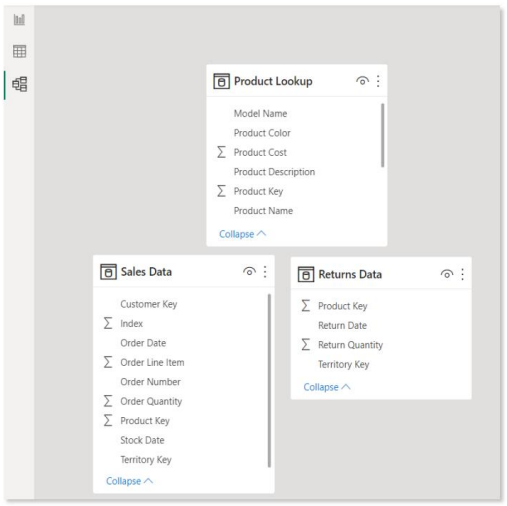
TOPICS WE'LL COVER:

- Data Modeling 101
- Normalization
- Facts & Dimensions
- Primary & Foreign Keys
- Cardinality
- Filter Flow
- Common Schemas
- Hierarchies

GOALS FOR THIS SECTION:

- Understand the basic principles of data modeling, including normalization, fact & dimension tables and common schemas
- Create table relationships using primary and foreign keys, and discuss different types of relationship cardinality
- Configure report filters and trace filter context as it flows between related tables in the model
- Explore data modeling options like hierarchies, data categories and hidden fields

WHAT IS A DATA MODEL?

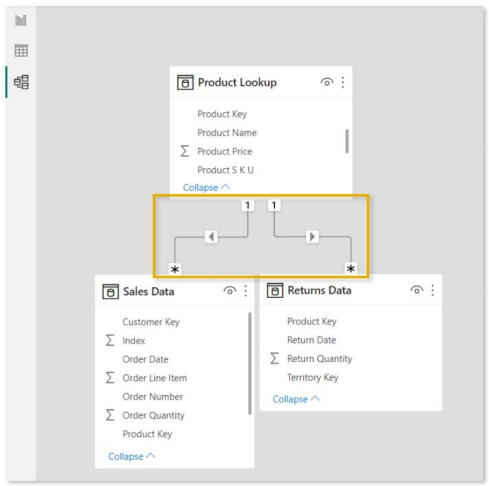


This **IS NOT** a data model ☹

- This is a collection of independent tables, which share no connections or relationshi
- If you tried to visualize **Orders** and **Return** by **Product**, this is what you'd get

ProductName	OrderQuantity	ReturnQuantity
All-Purpose Bike Stand	84,174	1,828
AWC Logo Cap	84,174	1,828
Bike Wash - Dissolver	84,174	1,828
Cable Lock	84,174	1,828
Chain	84,174	1,828
Classic Vest, L	84,174	1,828
Classic Vest, M	84,174	1,828
Classic Vest, S	84,174	1,828
Fender Set - Mountain	84,174	1,828
Total	**84,174**	**1,828**

WHAT IS A DATA MODEL?



This **IS** a data model! 😀

- The tables are connected via relationships, based on a common field (Product Key)
- Now **Sales** and **Returns** data can be filtered using fields from the **Product Lookup** table!

ProductName	OrderQuantity	ReturnQuantity
All-Purpose Bike Stand	234	8
AWC Logo Cap	4,151	46
Bike Wash - Dissolver	1,706	25
Classic Vest, L	182	4
Classic Vest, M	182	7
Classic Vest, S	157	8
Fender Set - Mountain	3,960	54
Half-Finger Gloves, L	840	18
Half-Finger Gloves, M	918	16
Total	**84,174**	**1,828**

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

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

date	product_id	quantity	product_brand	product_name	product_sku	product_weight
1/1/1997	869	5	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/7/1997	869	2	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/3/1997	1	4	Washington	Washington Berry Juice	90748583674	8.39
1/1/1997	1472	3	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/6/1997	1472	2	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/5/1997	2	4	Washington	Washington Mango Drink	96516502499	7.42
1/1/1997	76	4	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/1/1997	76	2	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/5/1997	3	2	Washington	Washington Strawberry Drink	58427771925	13.1
1/7/1997	3	2	Washington	Washington Strawberry Drink	58427771925	13.1
1/1/1997	320	3	Excellent	Excellent Cranberry Juice	36570182442	16.4

*Models that aren't normalized contain **redundant, duplicate data**. In this case, all of the product-specific fields could be stored in a separate table containing a unique record for each **product id***

This may not seem critical now, but minor inefficiencies can become major problems at scale!

FACT & DIMENSION TABLES

Data models generally contain two types of tables: **fact** ("data") tables, and **dimension** ("lookup") tables:

- **Fact tables** contain **numerical values** or metrics used for summarization *(sales, orders, transactions, pageviews, etc.)*
- **Dimension tables** contain **descriptive attributes** used for filtering or grouping *(products, customers, dates, stores, etc.)*

date	product_id	quantity
1/1/1997	869	5
1/1/1997	1472	3
1/1/1997	76	4
1/1/1997	320	3
1/1/1997	4	4
1/1/1997	952	4
1/1/1997	1222	4
1/1/1997	517	4
1/1/1997	1359	4
1/1/1997	357	4
1/1/1997	1426	5
1/1/1997	190	4
1/1/1997	367	4
1/1/1997	250	5
1/1/1997	600	4
1/1/1997	702	5

*This **Fact** table contains **quantity** values, along with **date** and **product_id** fields*

date	day_of_month	month	year	weekday	week_of_year	week_ending	month_name	quarter
1/1/1997	1	1	1997	Wednesday	1	1/5/1997	January	Q1
1/2/1997	2	1	1997	Thursday	1	1/5/1997	January	Q1
1/3/1997	3	1	1997	Friday	1	1/5/1997	January	Q1
1/4/1997	4	1	1997	Saturday	1	1/5/1997	January	Q1
1/5/1997	5	1	1997	Sunday	2	1/5/1997	January	Q1
1/6/1997	6	1	1997	Monday	2	1/12/1997	January	Q1

*This **Calendar Lookup** table contains attributes about each **date** (month, year, quarter, etc.)*

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight
1	Washington	Washington Berry Juice	90748583674	2.85	0.94	8.39
2	Washington	Washington Mango Drink	96516502499	0.74	0.26	7.42
3	Washington	Washington Strawberry Drink	58427771925	0.83	0.4	13.1
4	Washington	Washington Cream Soda	64412155747	3.64	1.64	10.6
5	Washington	Washington Diet Soda	85561191439	2.19	0.77	6.66
6	Washington	Washington Cola	29804642796	1.15	0.37	15.8
7	Washington	Washington Diet Cola	20191444754	2.61	0.91	18
8	Washington	Washington Orange Juice	89770532250	2.59	0.8	8.97

*This **Product Lookup** table contains attributes about each **product_id** (brand, SKU, price, etc.)*

PRIMARY & FOREIGN KEYS

Fact Table

date	product_id	quantity
1/1/1997	869	5
1/1/1997	1472	3
1/1/1997	76	4
1/1/1997	320	3
1/1/1997	4	4
1/1/1997	952	4
1/1/1997	1222	4
1/1/1997	517	4
1/1/1997	1359	4
1/1/1997	357	4
1/1/1997	1426	5
1/1/1997	190	4
1/1/1997	367	4
1/1/1997	250	5
1/1/1997	600	4
1/1/1997	702	5

Calendar Lookup

date	day_of_month	month	year	weekday	week_of_year	week_ending	month_name	quarter
1/1/1997	1	1	1997	Wednesday	1	1/5/1997	January	Q1
1/2/1997	2	1	1997	Thursday	1	1/5/1997	January	Q1
1/3/1997	3	1	1997	Friday	1	1/5/1997	January	Q1
1/4/1997	4	1	1997	Saturday	1	1/5/1997	January	Q1
1/5/1997	5	1	1997	Sunday	2	1/5/1997	January	Q1
1/6/1997	6	1	1997	Monday	2	1/12/1997	January	Q1

Product Lookup

product_id	product_brand	product_name	product_sku	product_retail_price	product_cost	product_weight
1	Washington	Washington Berry Juice	90748583674	2.85	0.94	8.39
2	Washington	Washington Mango Drink	96516502499	0.74	0.26	7.42
3	Washington	Washington Strawberry Drink	58427771925	0.83	0.4	13.1
4	Washington	Washington Cream Soda	64412155747	3.64	1.64	10.6
5	Washington	Washington Diet Soda	85561191439	2.19	0.77	6.66
6	Washington	Washington Cola	29804642796	1.15	0.37	15.8
7	Washington	Washington Diet Cola	20191444754	2.61	0.91	18
8	Washington	Washington Orange Juice	89770532250	2.59	0.8	8.97

These are **foreign keys** (FK)

*They contain multiple instances of each value, and relate to **primary keys** in dimension tables*

These are **primary keys** (PK)

*They uniquely identify each row of the table, and relate to **foreign keys** in fact tables*

RELATIONSHIPS VS. MERGED TABLES

*Can't I just merge queries or use lookup functions to **pull everything into one single table**?*

- Anonymous confused man

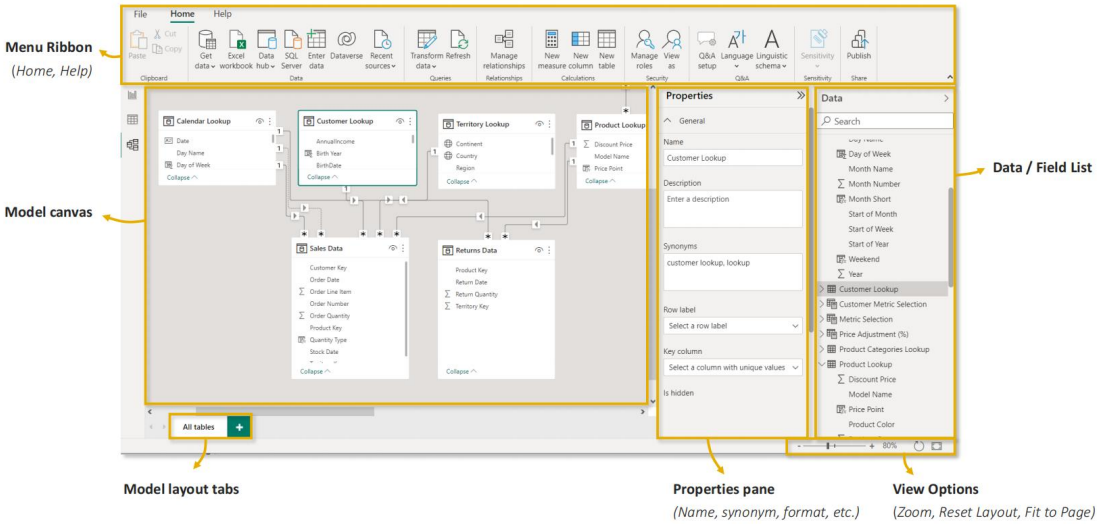
*Original **Fact Table** fields* | *Attributes from **Calendar Lookup** table* | *Attributes from **Product Lookup** table*

date	product_id	quantity	day_of_month	month	year	weekday	month_name	quarter	product_brand	product_name	product_sku	product_weight
1/1/1997	869	5	1	1	1997	Wednesday	January	Q1	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/7/1997	869	2	7	1	1997	Tuesday	January	Q1	Nationeel	Nationeel Grape Fruit Roll	52382137179	17
1/3/1997	1	4	3	1	1997	Friday	January	Q1	Washington	Washington Berry Juice	90748583674	8.39
1/1/1997	1472	3	1	1	1997	Wednesday	January	Q1	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/6/1997	1472	2	6	1	1997	Monday	January	Q1	Fort West	Fort West Fudge Cookies	37276054024	8.28
1/5/1997	2	4	5	1	1997	Sunday	January	Q1	Washington	Washington Mango Drink	96516502499	7.42
1/1/1997	76	4	1	1	1997	Wednesday	January	Q1	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/1/1997	76	2	1	1	1997	Wednesday	January	Q1	Red Spade	Red Spade Sliced Chicken	62054644227	18.1
1/5/1997	3	2	5	1	1997	Sunday	January	Q1	Washington	Washington Strawberry Drink	58427771925	13.1
1/7/1997	3	2	7	1	1997	Tuesday	January	Q1	Washington	Washington Strawberry Drink	58427771925	13.1
1/1/1997	320	3	1	1	1997	Wednesday	January	Q1	Excellent	Excellent Cranberry Juice	36570182442	16.4

You can, **but it's extremely inefficient!**

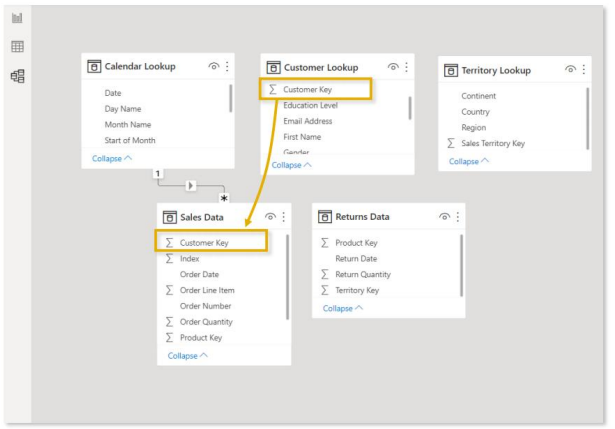
- Merging tables creates **redundancy** and often requires **significantly more memory and processing power** to analyze compared to a relational model with multiple small tables

THE MODEL VIEW

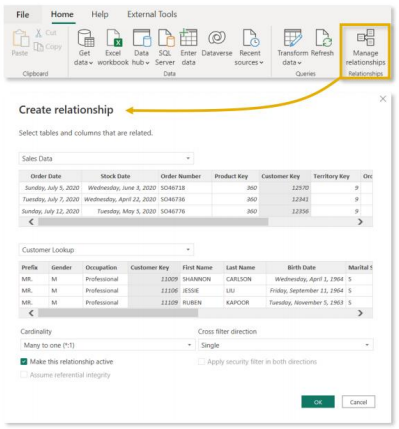


CREATING TABLE RELATIONSHIPS

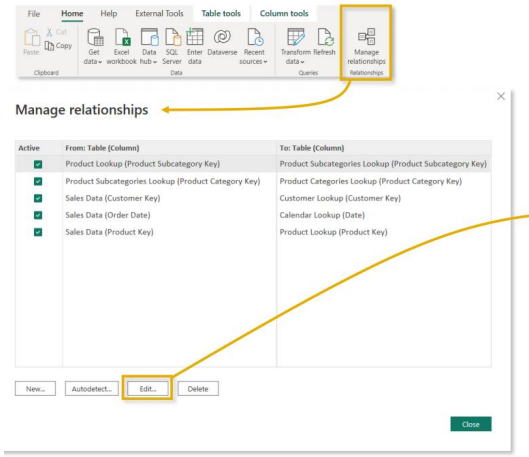
OPTION 1: Click and drag to connect primary and foreign keys within the **Model** view



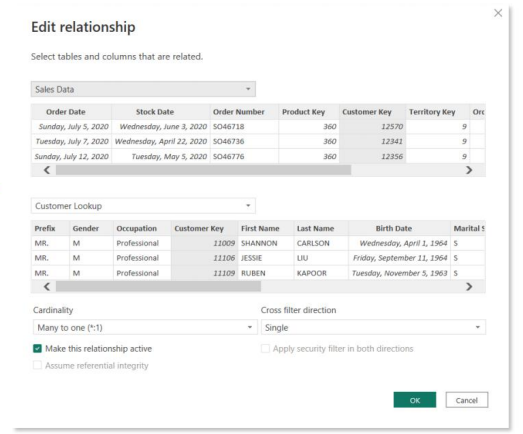
OPTION 2: Add or detect relationships using the **Manage Relationships** dialog box



MANAGING & EDITING RELATIONSHIPS

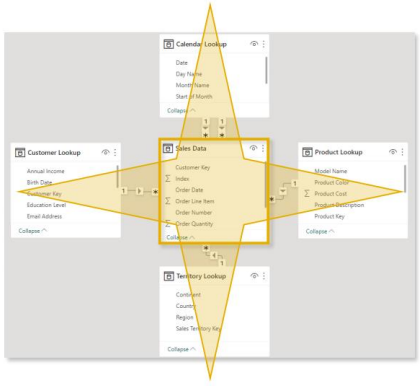


Launch the **Manage Relationships** dialog box or double-click a relationship to modify it

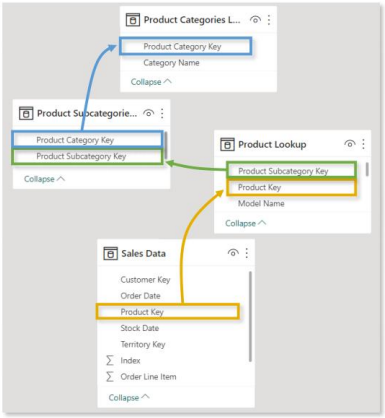


Editing tools allow you to **activate or deactivate** relationships and manage **cardinality** and **filter direction** – more on that soon!

STAR & SNOWFLAKE SCHEMAS



A **star schema** is the simplest and most common type of data model, characterized by a single fact table surrounded by related dimension tables



A **snowflake schema** is an extension of a star, and includes relationships between dimension tables and related sub-dimension tables

ASSIGNMENT: TABLE RELATIONSHIPS

NEW MESSAGE

From: **Dana Modelle** *(Analyst)*

Subject: **Need a favor...**

Hey there,

Ethan shared the data model you've been working on, and we might have an issue...

Last night I left my laptop open, and my cat Dennis somehow got his paws on our model. Now all the relationships are gone!

Could you please rebuild the model, including all three product tables? I owe you one!

-Dana

Reply Forward

Key Objectives

1. Delete all existing table relationships
2. Create a star schema by creating relationships between the Sales, Calendar, Customer, Product and Territories tables
3. Connect all three product tables (Product, Subcategory, Category) in a snowflake schema
4. Use the matrix visual to confirm that you can filter Order Quantity values using fields from each dimension table

SOLUTION: TABLE RELATIONSHIPS

NEW MESSAGE

From: **Dana Modelle** *(Analyst)*

Subject: **Need a favor...**

Hey there,

Ethan shared the data model you've been working on, and we might have an issue...

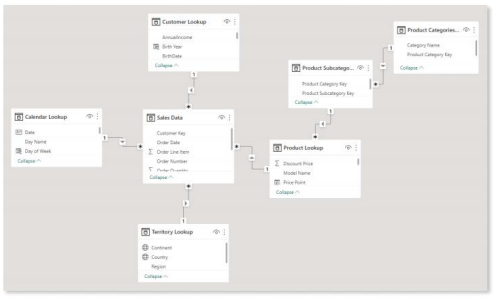
Last night I left my laptop open, and my cat Dennis somehow got his paws on our model. Now all the relationships are gone!

Could you please rebuild the model, including all three product tables? I owe you one!

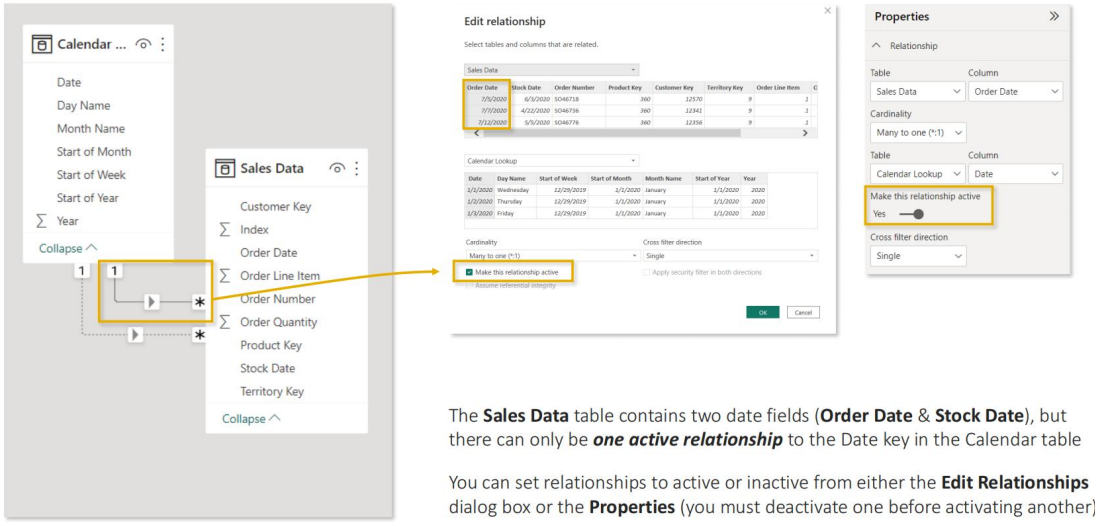
-Dana

Reply Forward

Solution Preview



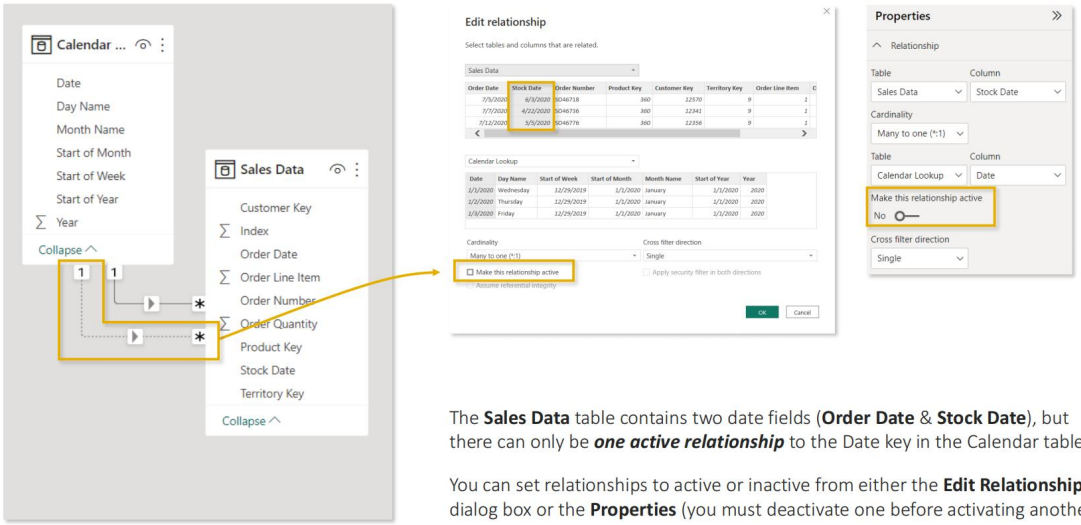
PRO TIP: ACTIVE & INACTIVE RELATIONSHIPS



The **Sales Data** table contains two date fields (**Order Date** & **Stock Date**), but there can only be ***one active relationship*** to the Date key in the Calendar table

You can set relationships to active or inactive from either the **Edit Relationships** dialog box or the **Properties** (you must deactivate one before activating another)

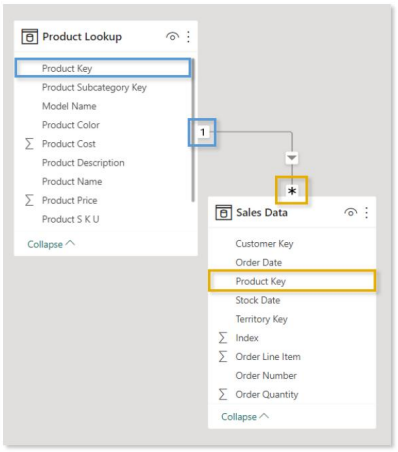
PRO TIP: ACTIVE & INACTIVE RELATIONSHIPS



The **Sales Data** table contains two date fields (**Order Date** & **Stock Date**), but there can only be ***one active relationship*** to the Date key in the Calendar table

You can set relationships to active or inactive from either the **Edit Relationships** dialog box or the **Properties** (you must deactivate one before activating another)

RELATIONSHIP CARDINALITY



Cardinality refers to the uniqueness of values in a column

- Ideally, all relationships in the data model should follow a **one-to-many** cardinality: **one** instance of each primary key, and **many** instances of each foreign key

*In this example there is only **ONE instance of each Product Key** in the Product table (noted by a "**1**"), since each row contains **attributes of a single product** (name, SKU, description, price, etc.)*

*There are **MANY instances of each Product Key** in the Sales table (noted by an asterisk *****), since there are **multiple sales for each product***

EXAMPLE: MANY-TO-MANY CARDINALITY

Product Lookup

product_id	product_name	product_sku
4	Washington Cream Soda	64412155747
4	Washington Diet Cream Soda	81727382373
5	Washington Diet Soda	85561191439
7	Washington Diet Cola	20191444754
8	Washington Orange Juice	89770532250

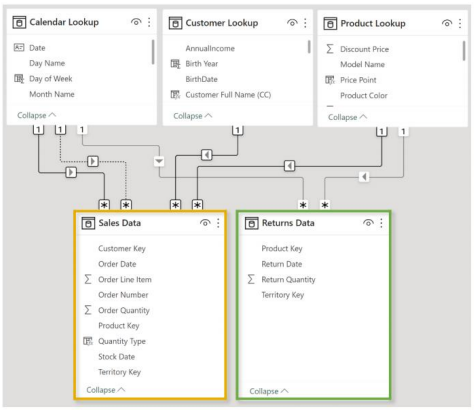
Sales

date	product_id	transactions
1/1/2017	4	12
1/2/2017	4	9
1/3/2017	4	11
1/1/2017	5	16
1/2/2017	5	19
1/1/2017	7	11

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

- If we try to connect the tables above using **product_id**, we'll get a **many-to-many relationship** warning since there are multiple instances of product_id in both tables
- Even if we force this relationship, how would we know which product was actually sold on each date – **Cream Soda** or **Diet Cream Soda**?

CONNECTING MULTIPLE FACT TABLES



This model contains two fact tables: **Sales Data** and **Returns Data**

- Since there is no primary/foreign key relationship, we can't connect them directly to each other
- But we *can* connect each fact table to related lookups, which allows us to filter both sales and returns data **using fields from any shared lookup tables**
- We can view orders and returns by product since both tables relate to Product Lookup, but we can't view returns by customer since no relationship exists

HEY THIS IS IMPORTANT!
Generally speaking, fact tables should **connect through shared dimension tables, not directly to each other**