



Microsoft Power BI Data Analyst

Exam Ref PL-300

Daniil Maslyuk

Exam Ref PL-300 Microsoft Power BI Data Analyst

Daniil Maslyuk

6. After you load data from an Excel file, you encounter the error shown in Figure 1-39.

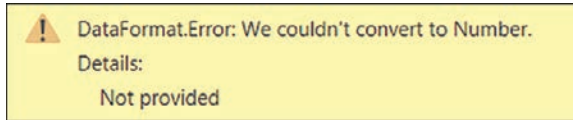


FIGURE 1-39 Error message

Which Power Query feature can you use to resolve the error? Your solution must retain all data from Excel.

- A. Replace errors
- B. Remove rows
- C. Change type (add new step)
- D. Change type (replace current)

Thought experiment answers

1. The answer is **A**. Using a parameter allows changing the source of all relevant queries at once, and it allows changing the source in the Power BI service. Although option B also allows changing the source of all affected queries, it does not allow changing the source in Power BI service. Option C requires you to change the source in each query separately, and you cannot change the source in the Power BI service this way.
2. The answer is **B** or **C**. Keeping top rows, if applied early in the query, will translate into SQL efficiently, resulting in fast data loading due to filtering taking place in the database. Using your own SQL statement with a TOP clause also filters data in the database, resulting in efficient data loading. Option A requires you to load all data first, which can take a long time.
3. The answer is **B** and **D**. Column distribution does not show the number of missing values, whereas column quality does. By default, column profiling is based on the first 1,000 rows; since there are over 10,000 rows in the CSV file, you need to perform column profiling on the entire dataset; otherwise, you risk missing some empty values at the end of the file.
4. The answer is **C**. The DirectQuery mode allows you to get the latest data as it arrives. Importing data requires refreshing the dataset, which would require Power BI Premium to refresh so frequently. Live Connection would require an Analysis Services model or a Power BI dataset, which are not available in Contoso.
5. The answer is **B**. Merge combines two tables by joining them based on a set of keys. Group by is for summarizing a single table. Transpose switches rows and columns of a single table. Append is best used to combine tables with mostly same columns.

6. The answer is **D**. The error is due to a text value being converted to a number, so changing the type to text instead of number would prevent the error from happening. Replacing an error will result in a number or null value, which would mean the original text value would be lost. Removing rows would also result in data loss. Changing type as a new step will not solve the problem as it will apply to the error value.

Model the data

In the previous chapter, we reviewed the skills necessary to get and transform data by using Power Query Editor—the process also known as *data shaping*. In this chapter, we examine the skills needed to model data.

Although Power BI allows you to analyze your data to some degree right after you load it, a strong understanding of data modeling allows you to perform sophisticated analysis using rich data modeling capabilities, which includes creating relationships, hierarchies, and various calculations to bring out the true power of Power BI. Previously in the Power Query Editor we used the M language; after we load the data into the model, we use data analysis expressions, more commonly referred to as DAX—Power BI’s native query language.

In this chapter, we review the skills necessary to design, develop, and optimize data models. Additionally, we look at DAX and how it can be used to enhance data models.

Skills covered in this chapter:

- Skill 2.1: Design a data model
- Skill 2.2: Develop a data model
- Skill 2.3: Create model calculations by using DAX
- Skill 2.4: Optimize model performance

Skill 2.1: Design a data model

A proper data model is the foundation of meaningful analysis. A Power BI data model is a collection of one or more tables and, optionally, relationships. A well-designed data model enables business users to understand and explore their data and derive insights from it. This step should be taken before you create any visuals by loading your data and defining the relationships between tables. Data modeling often occurs at the beginning phase of building a Power BI report so that you can create efficient measures that build upon your data model. In this section, we design a data model by focusing our attention on tables and their relationships.

This skill covers how to:

- Define the tables and design a data model that uses a star schema
- Configure table and column properties

- Design and implement role-playing dimensions
- Define a relationship's cardinality and cross-filter direction
- Create a common date table

Define the tables and design a data model that uses a star schema

Once a query is loaded, it becomes a table in a Power BI data model. Tables can then be organized into different data model types, also known as *schemas*. The three most common schemas in Power BI are:

- Flat (fully denormalized) schema
- Star schema
- Snowflake schema

There are other types of data models, though these three are the most common ones.

Flat schema

In the flat type of data model, all attributes are fully denormalized into a single table. Because there's only one table, there are no relationships, and in most cases there's no need for key.

In our Wide World Importers example, we have a single table that contains all columns from all tables, meaning that the Sale and Targets columns will be in the same table. Because the tables have different data granularity, you run into problems when comparing actuals and targets.

NOTE DATA GRANULARITY

We review the concept of data granularity later in this skill section.

From the performance point of view, flat schemas are very efficient, though there are downsides:

- A single table can be cumbersome and confusing to navigate.
- Columns and data can often be duplicated, leading to a comparatively large file size.
- Mixing facts of different grains results in more complex DAX formulas.

Flat schemas are often used when connecting to a single, simple source. However, for more complex data models, flat schemas should be avoided in Power BI as much as possible.

Star schema

When you use a star schema, tables are conceptually classified into two kinds:

- **Fact tables** These tables contain the metrics you want to aggregate. Fact tables have foreign keys, which are required in order to create relationships with dimensions, and

columns that you can aggregate. In our Wide World Importers example, the Sale and Targets tables are fact tables. Fact tables are sometimes also known as *data tables*.

- **Dimension tables** These tables contain the descriptive attributes that help you slice and dice your fact tables. A dimension table has a unique identifier—a key column—and descriptive columns. In our Wide World Importers example, the City, Customer, Date, Employee, and Stock Item are dimension tables. Dimension tables are also sometimes known as *lookup tables*.

In a star schema, fact tables are surrounded by dimensions, as shown in Figure 2-1.

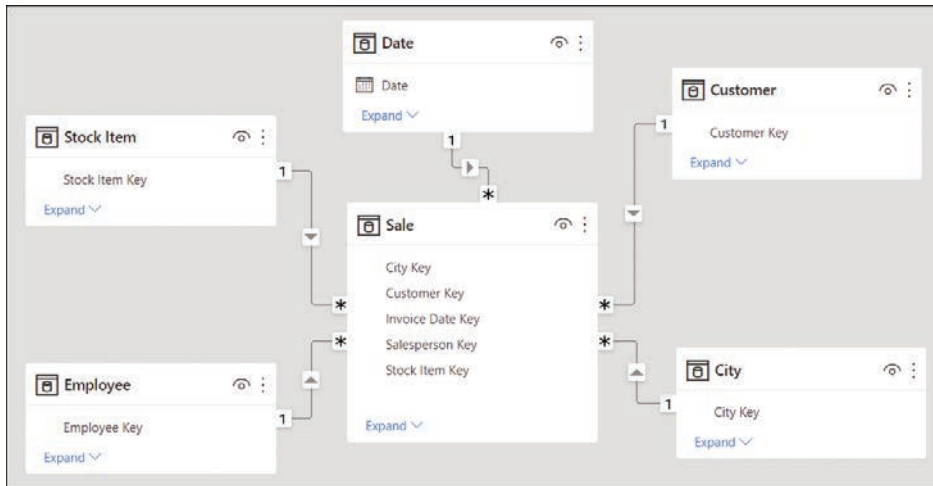


FIGURE 2-1 Star schema with Sale as the only fact table

The star schema has its name because it resembles a star, with the fact table in the center and dimension tables as the star points. It's possible to have more than one fact table in a star schema, and it will still be a star schema.

NOTE RELATIONSHIPS

The lines that connect tables in Figure 2-1 represent relationships. We cover the relationships in more detail later in this section.

In most cases, the star schema is the preferred data modeling approach in Power BI. It addresses the shortcomings of the flat schema:

- Fields are logically grouped, making the model easier to understand.
- There is less duplication of data, which results in more efficient storage.
- You don't need to write overly complex DAX formulas to work with fact tables that have a different grain.

Snowflake schema

The snowflake schema is similar to the star schema, except it can have some dimensions that “snowflake” from other dimensions. You can see an example in Figure 2-2.

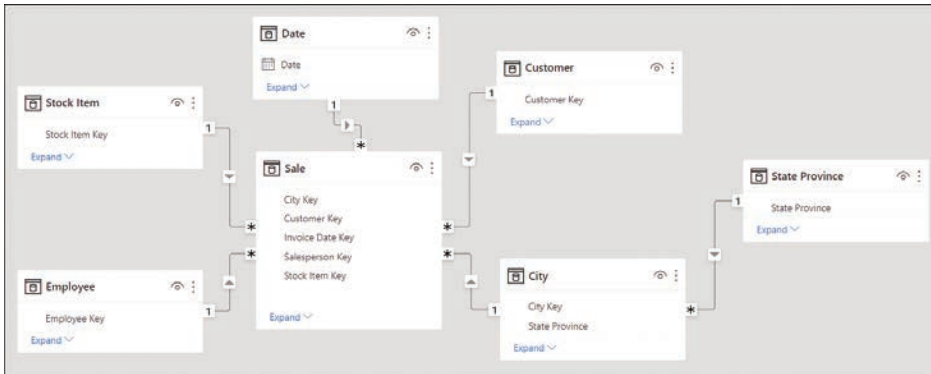


FIGURE 2-2 Snowflake schema with State Province snowflaking from the City table

In the Wide World Importers example, if we loaded the State Province query, the data model could be a snowflake schema. This is because the State Province table is related to the City dimension table, which in turn is related to the Sale fact table.

Snowflake schemas can be beneficial when there are fact tables that have different grains.

NEED MORE REVIEW? DIMENSIONAL MODELING

In addition to fact and dimension tables, there are other types of tables such as factless facts, junk, and degenerate dimensions. For more information, see “Understand star schema and the importance for Power BI” at <https://docs.microsoft.com/en-us/power-bi/guidance/star-schema>.

Configure table and column properties

Both tables and columns have various properties you can configure, and you can do it in the **Model** view. To see the properties of a column or a table, select an object, and you will see its properties in the **Properties** pane.

Table properties

For tables, depending on the storage mode, you can configure the following properties:

- **Name** Enter the table name.
- **Description** This property allows you to add a description of the table that will be stored in the model’s metadata. It can be useful when building reports because you can see the description when you hover over the table in the Fields pane.
- **Synonyms** These are useful for the Q&A feature of Power BI, which we review in the next skill section. You can add synonyms so that the Q&A feature can understand that you’re referring to a specific table even if you provide a different name for it.