**Tuning report performance**

**Introduction**

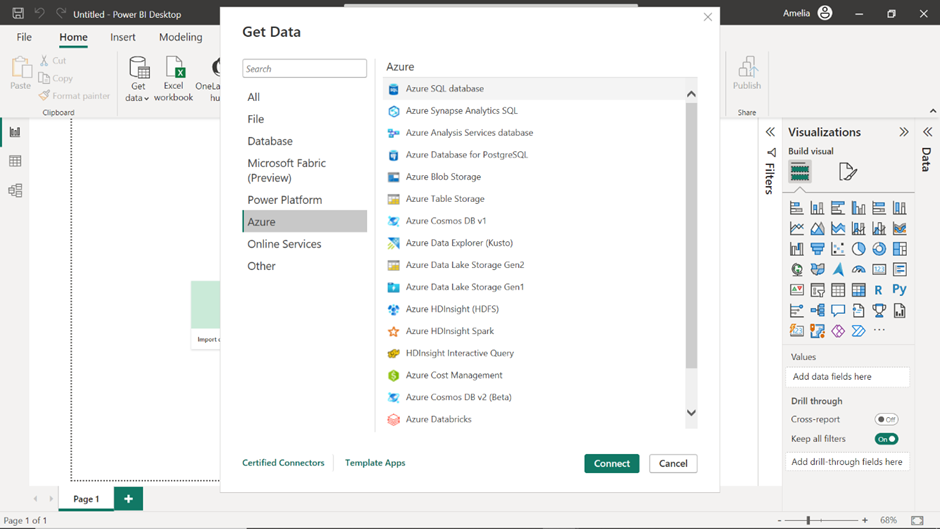
Imagine you're a data analyst who's just landed a role in the marketing department at Adventure Works – a multinational manufacturing company. You're excited to contribute and make a significant impact. You've assembled a report with a medley of visuals that unveil market segments, track sales trends, and highlight regional performance. But then reality strikes, the dashboard is slow. Not just inconveniently slow, but agonizingly, slow. As you attempt to refresh the data, the seconds stretch into minutes. That sense of frustration as you tap your fingers impatiently on your desk is exactly what you're here to address today – tuning report performance.

**What is Power BI performance tuning?**

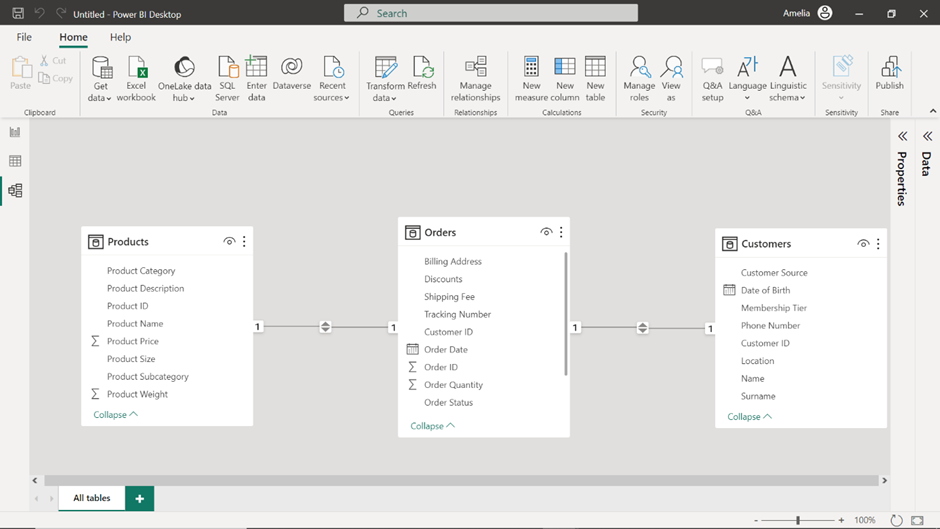
Performance tuning in Microsoft Power BI involves optimizing various aspects of your report so it runs faster and more efficiently. It's about making targeted changes to your data queries, data model, and visualizations to improve performance, thereby ensuring that your insights are delivered promptly and effectively. It's like fine-tuning a high-performance car; every cog, wheel, and engine part must work in perfect harmony for the entire machine to deliver peak performance.

Let’s dive into the key components of Performance Tuning in Power BI:

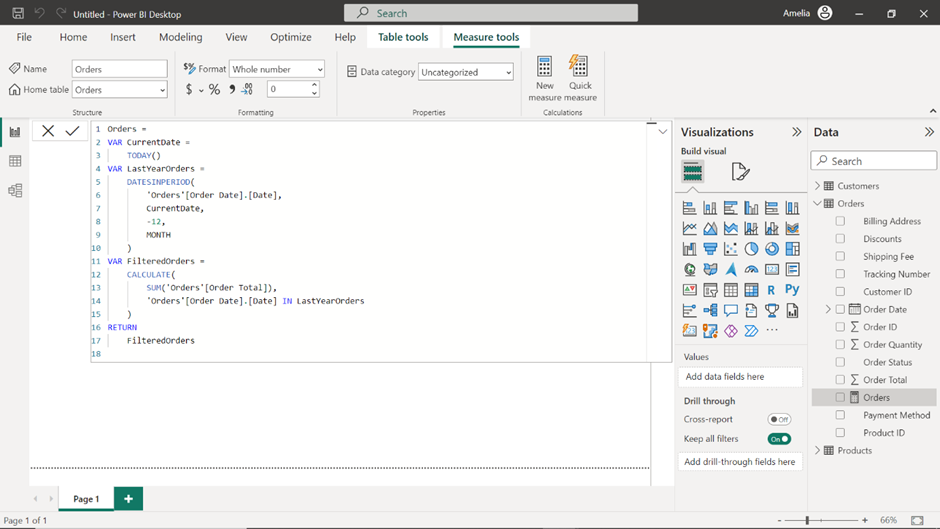
1. **Data Queries**: In Power BI, the journey begins with data extraction. You use queries to pull data from various sources, which could be Microsoft Azure SQL databases, Excel spreadsheets, or even web services. Your data queries, or the way you're interacting with your data sources, can be the first culprit of slow performance.



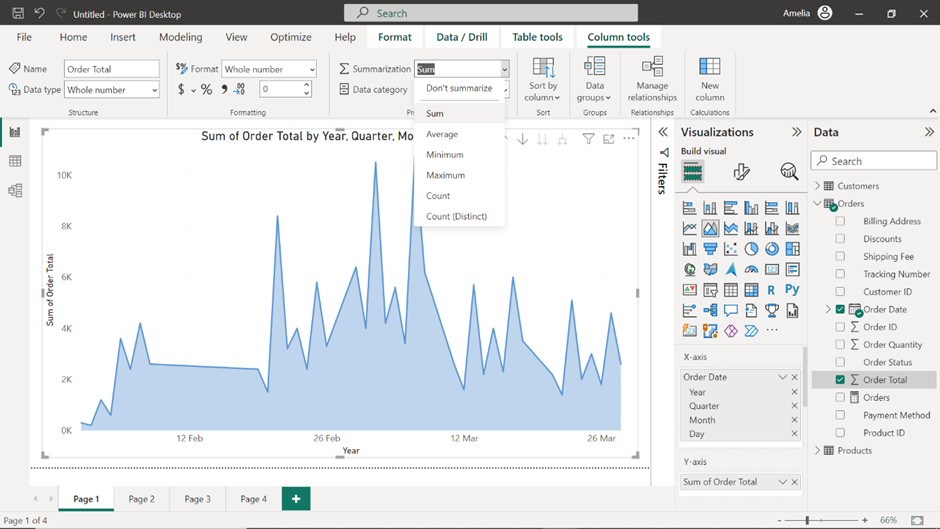
2. **Data Model**: After your data is pulled into Power BI, the next step is structuring it in a way that is both meaningful and efficient. Your data model includes tables, relationships between them, hierarchies, and calculated columns or measures. An optimized data model avoids redundancy and ensures that relationships are well-defined, which in turn, makes **DAX** calculations and data retrieval more efficient.



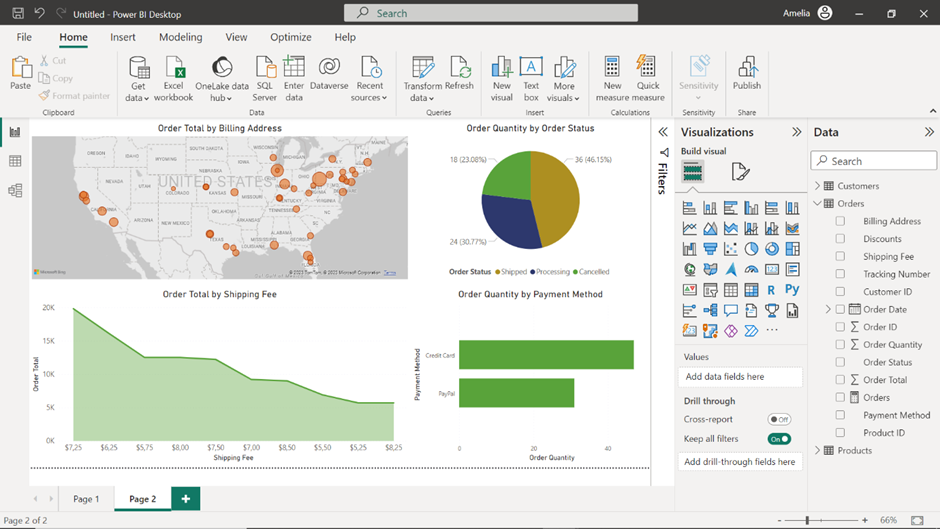
3. **DAX Calculations**: **DAX** is the formula language in Power BI used for calculations and data analysis. While powerful, it can be computationally intensive when not used correctly. Understanding the context in which **DAX** operates can help you write more efficient code. For instance, the **CALCULATE** function can modify filter contexts and can be inefficient in terms of performance. Understanding how to optimize its use can significantly speed up your reports.



4. **Visualizations**: Visualizations are the primary way through which your data speaks to end-users. However, complex visuals can be computationally expensive. The type of visualizations you choose, the amount of data they display, and the number of visuals on a page all affect performance.



5. **Report Layout**: While it may seem like a trivial part of the report, the layout can impact performance. Power BI has certain rendering behaviors that, when understood, can be used to your advantage. For example, Power BI renders visuals from top to bottom and considers the visibility of elements. Knowing this can help you place your most important and least computationally intensive visuals at the top of your report, ensuring they load first for a more responsive user experience.



6. **Backend Infrastructure**: The performance of your Power BI reports is also dependent on the backend infrastructure. Whether you're using Power BI Service in the cloud or Power BI Report Server on-premises, the capabilities of these servers can be a bottleneck.

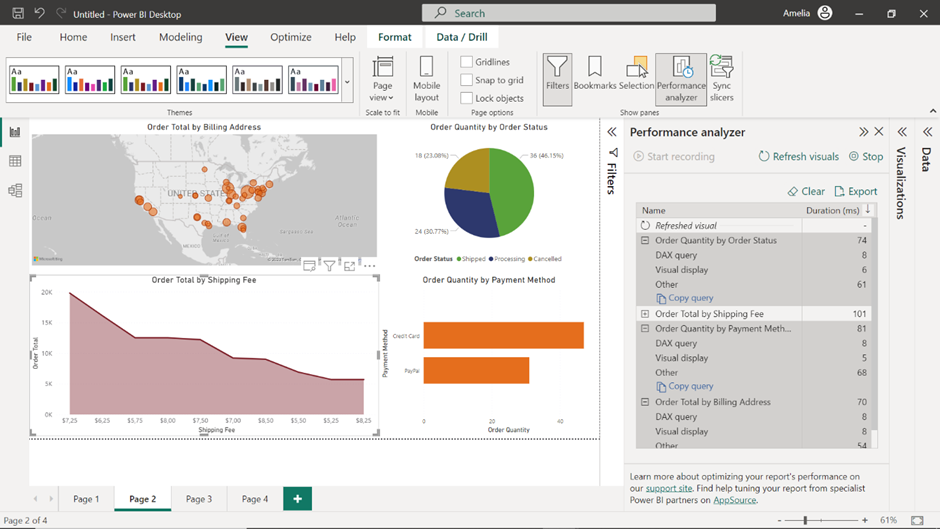
**Why do we need performance tuning?**

Performance tuning isn't a luxury; it's a necessity, particularly for data-heavy businesses like Adventure Works. The data it handles is not just voluminous but also incredibly dynamic, spanning sales, manufacturing, human resources, and much more. Now, let’s explore the consequences of poor performance if left unchecked:

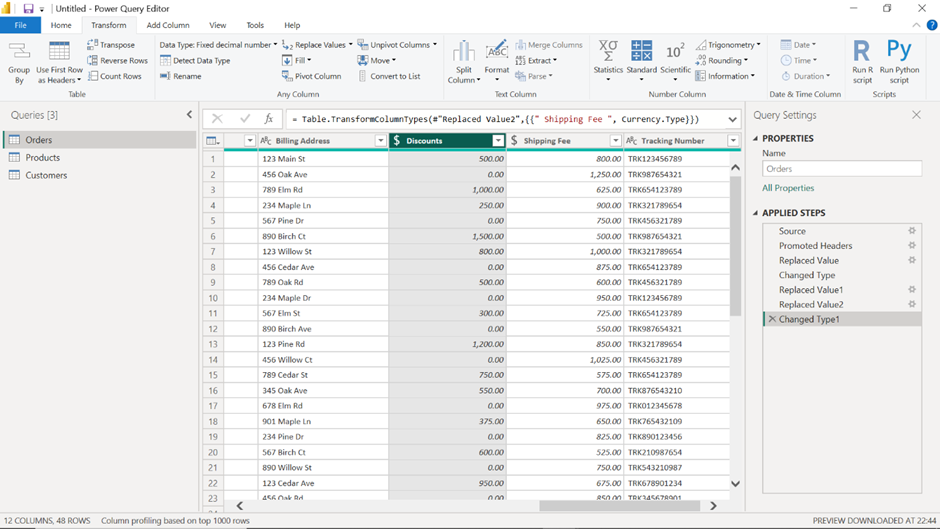
1. **Longer decision-making processes**: In a competitive marketplace, every minute counts. Slow report loading times mean that decisions are delayed, opportunities are missed, and threats go unmitigated.
2. **User dissatisfaction**: When employees face constant delays, they get frustrated. This frustration can lead to decreased engagement, leading to reduced effectiveness in job roles that depend on real-time data.
3. **Undermining data-driven initiatives**: One of the critical effects of poor performance is the erosion of faith in data-driven decision-making. If the reports meant to facilitate this are not up to the mark, there’s a risk of stakeholders reverting back to gut-feeling decisions, which can be highly unreliable.
4. **Resource inefficiency**: Slow performance doesn’t just waste time; it wastes computational resources. When queries are not optimized, or when data models are too complex, you may end up requiring more computing power than necessary, which could lead to increased operational costs.

Now that you understand performance tuning in Power BI and its importance, let’s explore the steps involved in the tuning process:

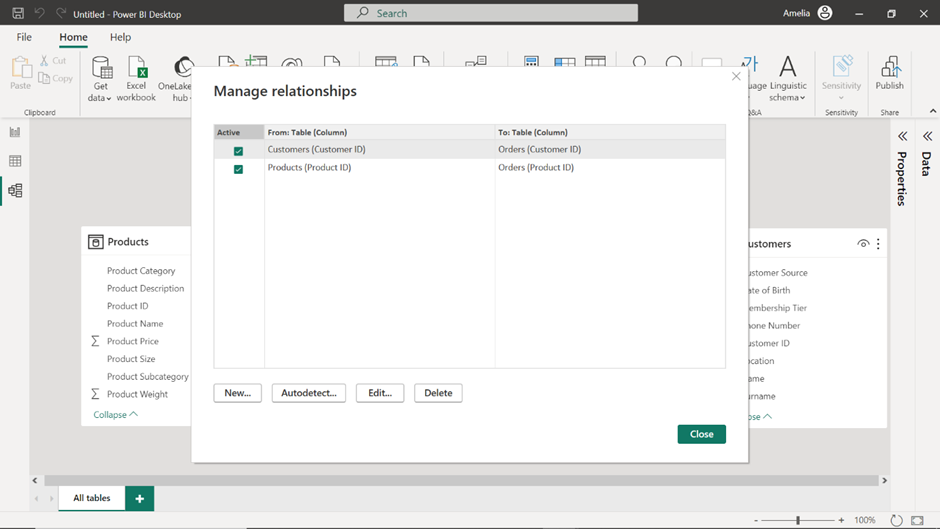
1. **Identify the problem areas:** The first step is to figure out what's causing the lag. Use the Power BI Performance Analyzer to identify which queries, visualizations, or data models are slow. Pinpoint problem areas so you know exactly where to direct your optimization efforts.



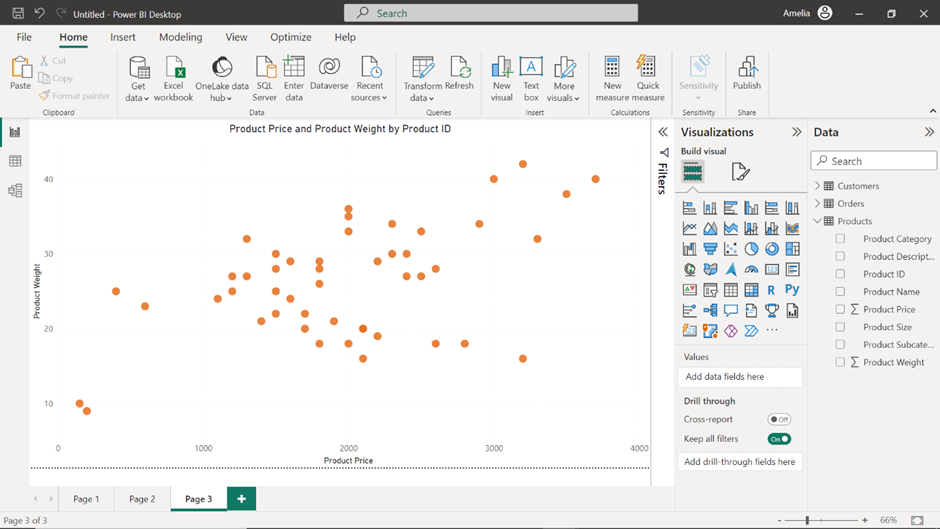
2. **Optimize data queries:** The next step focuses on slimming down your data queries. Trim down your data queries to fetch only the required fields. Remove any unnecessary calculations or transformations that are being performed during data import.



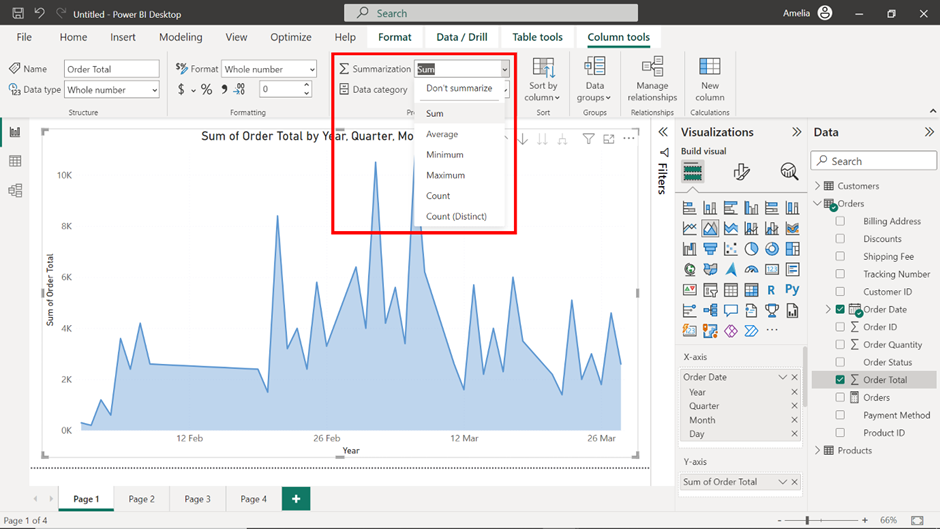
3. **Streamline the data model:** Once your queries are leaner, ensure that your data model is designed efficiently. Use appropriate indexing, relationships, and data types to optimize performance. Star schema designs and flattening tables where possible is a good practice to follow.



4. **Improve visualizations:** While a scatter plot or a 3D map might look impressive, they can be resource-intensive. Whenever possible, opt for simpler effective visuals that are less demanding on system resources. The goal is to convey information in the quickest and most effective way.



5. **Use aggregations and summarization:** Your report users may not need to see every single data point in detail. Instead of displaying all data, consider using summarized data and allow drill-downs to details. This reduces the initial data load, making your reports quicker to interact with.



6. **Caching and pre-loading data:** Power BI has excellent caching capabilities. You can leverage these to store queries that are often run, thus reducing the time it takes to fetch this data.

7. **Monitoring and iterating:** Finally, remember that performance tuning is an ongoing process. The data landscape and user requirements are always evolving, and your Power BI reports need to keep up. Regularly monitor your report’s performance metrics and adapt as needed.

**Conclusion**

Remember, the goal isn't just to have a functional Microsoft Power BI report; the goal is to deliver timely, valuable insights that empower teams and organizations to be more effective. By identifying problem areas with the Performance Analyzer and systematically addressing them, you can significantly improve your report's speed and usability. So, keep tuning and keep improving because in the world of data, speed, and efficiency are not just luxuries; they are necessities!

**Improving DAX performance**

**Introduction**

In the exercise *Improving DAX performance*, you were immersed in the world of Adventure Works, the multinational bicycle and accessories powerhouse, where you confronted a unique challenge: a slow Microsoft Power BI report. The mission in hand was to locate and optimize a DAX query that hindered the report’s performance, ensuring that data-driven decisions could be made efficiently and without delay.

More specifically, you were required to:

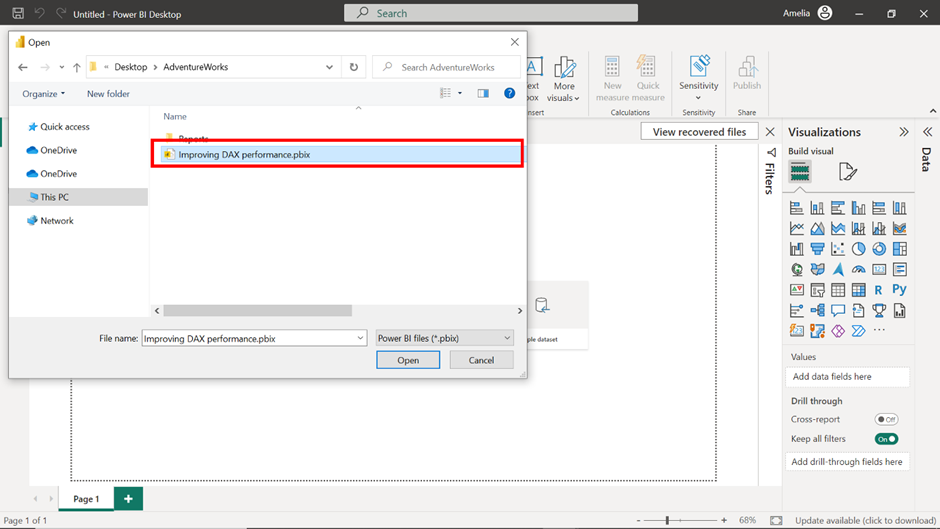
* Download the Adventure Works Power BI report named *Improving DAX performance.pbix* and open it in Power BI Desktop, setting the groundwork for your journey.
* Dive into the **Table view** to familiarize yourself with the dataset available in the report, laying the foundation for understanding the details of the data at hand.
* Harness the power of the **Performance Analyzer** to pinpoint the underperforming visualization and DAX formula in your report.
* Refine the DAX formula of the **Total Sales** field to optimize its performance, replacing the nested functions with a more streamlined version.
* Validate the modifications by re-running the **Performance Analyzer**, ensuring that the tweaks made indeed have a positive effect on the dashboard's efficiency.

This reading serves as a structured walkthrough, guiding you step by step, ensuring that you’re on the right track, and assisting you in comparing your efforts against a standard solution.

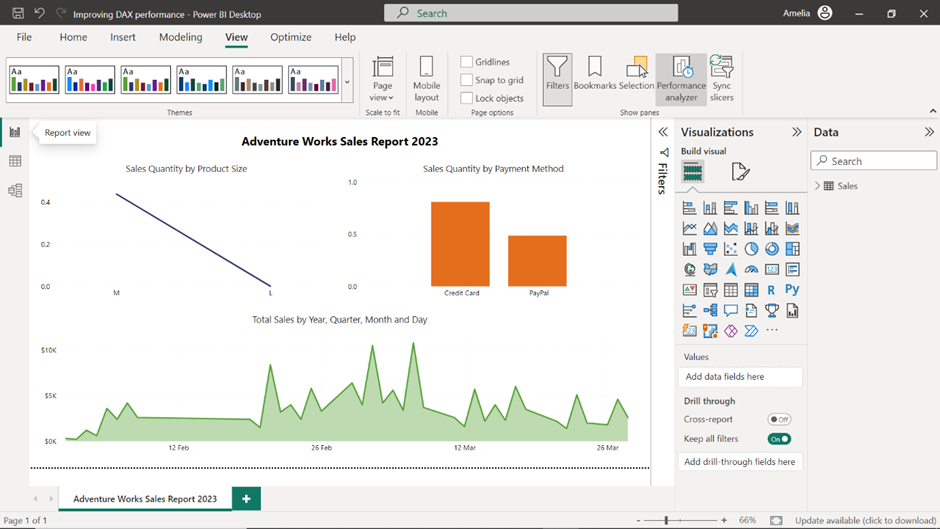
**Improving DAX performance**

**Step 1: Open your project**

1. Once you have Power BI Desktop open, in the top left corner, select the **File** menu.
2. Navigate through this to the location where your *Improving DAX performance* report file is saved.

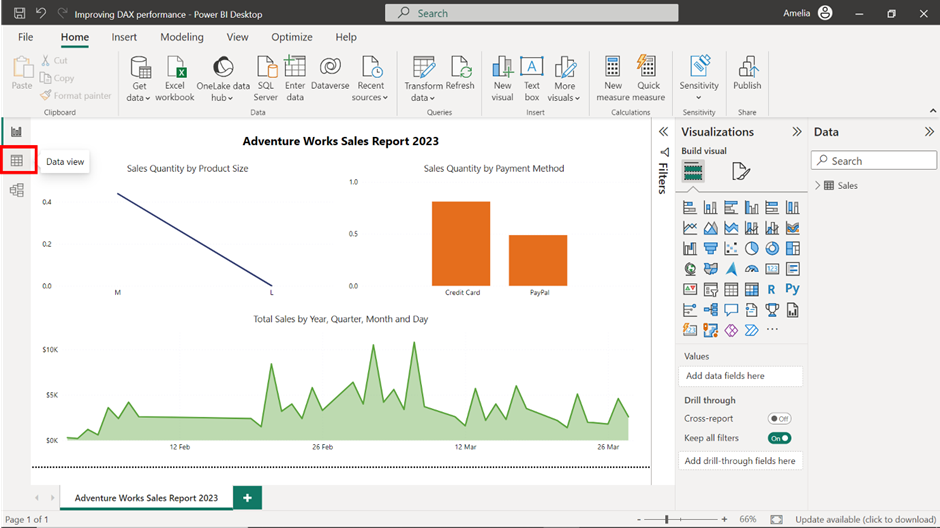


3. Select the file and select **Open** in the file explorer window. This action opens the saved project in the Power BI Desktop application.



**Step 2: Explore the data in Table view**

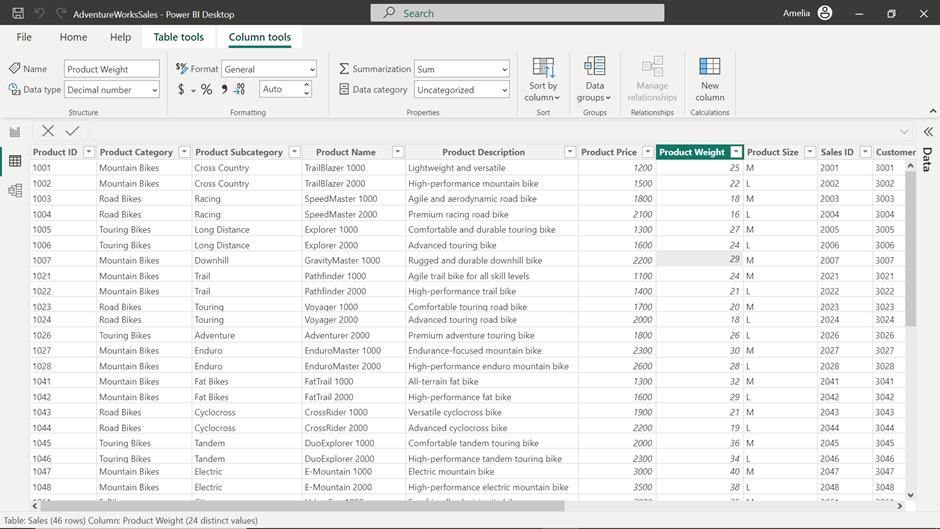
1. Once your report is loaded, on the left side of the Power BI interface, you'll find a vertical toolbar with different icons. The second icon from the top that resembles a table is the **Table view** icon. When you select this, Power BI takes you to the **Table view** which allows you to explore the data contained in your project.



2. Select the **Sales** dataset on the right of the screen and take a moment to observe the first 10 records.

3. Note which record has the highest **Product Weight** value. With a weight of 29 units, the **GravityMaster 1000** stands as the heaviest bicycle.

4. Then, note the **Day of Week** that has the highest frequency of sales. Monday stands out as the busiest day at Adventure Works–with three sales records, it has the highest frequency in the **Day of Week** column.



5. Finally, use the icons on the vertical toolbar on the left side of the Power BI interface to switch back to the **Report view**.

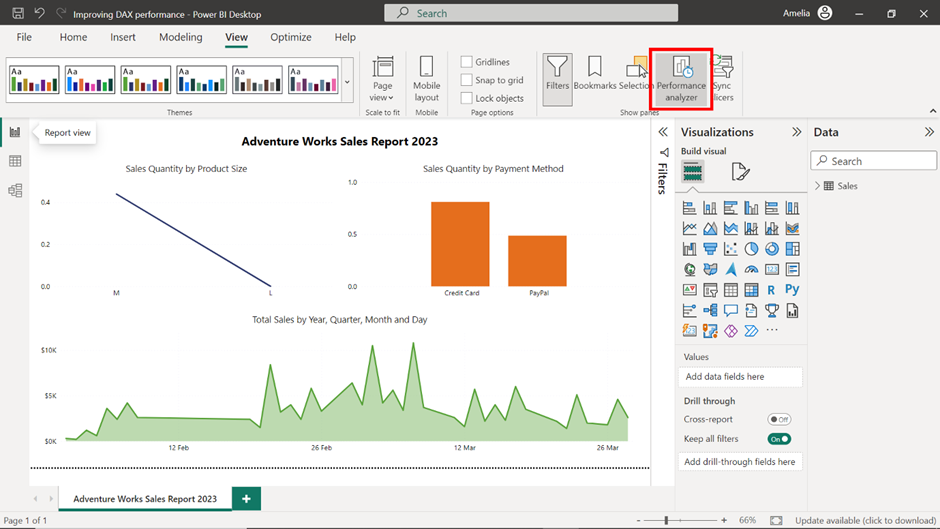
A screenshot of a computer

Description automatically generated

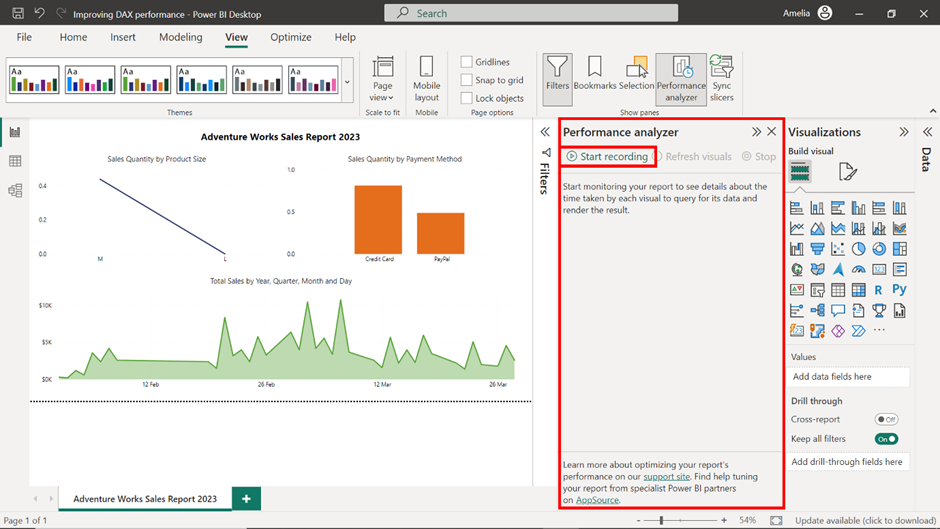
**Step 3: Access the Performance Analyzer**

1. Once you're in the **Report view**, you first need to open the **Performance Analyzer**. On the ribbon interface at the top of your Power BI report, locate and select the **View** tab.

2. Within the **View** tab, find and select the **Performance Analyzer** option.

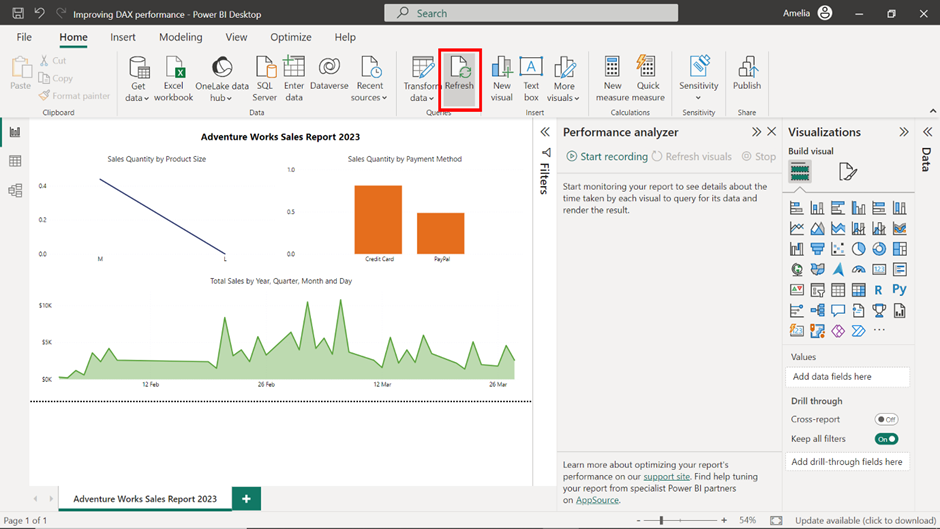


3. Upon selection, you’ll notice a pane unfold on the right side of the Power BI window. This **Performance Analyzer** pane is designed to showcase real-time performance metrics of your report visuals. Within the **Performance Analyzer** pane, locate and select the **Start Recording** button.



**Step 4: Refresh the report**

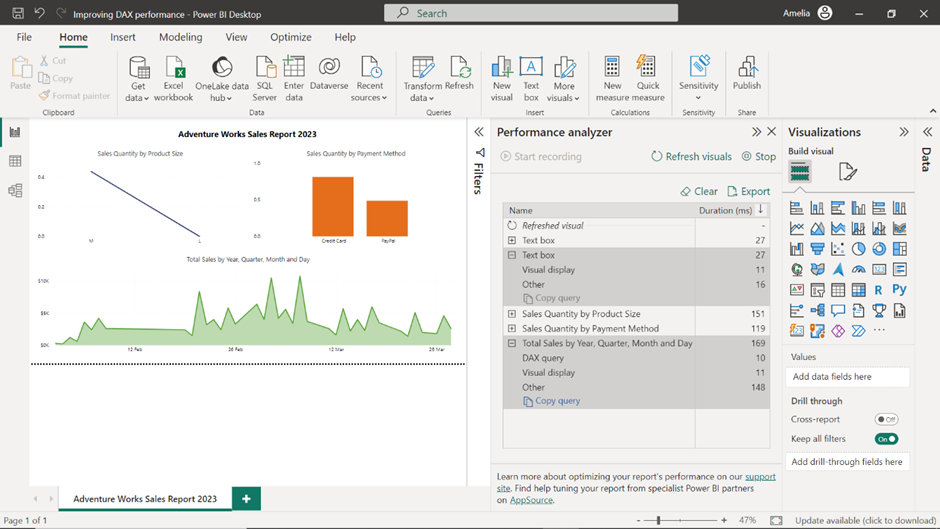
* Now, it's time to refresh your report. You can accomplish this in two ways: either by selecting the **Refresh** button situated in the **Home** tab of the ribbon interface or by directly interacting with the report.



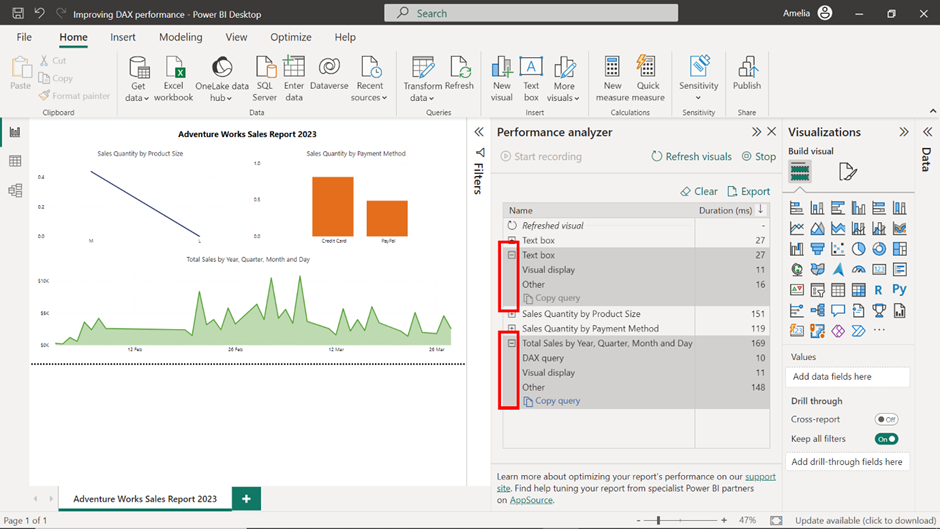
**Note:** As you interact with the report while the **Performance Analyzer** is recording, it will track and document the time taken to load each individual visual item. This data will be crucial for diagnosing performance issues.

**Step 5: Observe the results**

1. Once the report has finished refreshing, review the **Performance Analyzer** pane. A list of all the visual items in your report and their respective load times is available. Pay special attention to any visual items that take a significantly longer time to load compared to others. This ensures that you're not wasting time optimizing calculations that are already performing well.



2. For these slower visuals, drill down into the details by selecting the **+** symbol beside the visual item’s name. This will provide a detailed breakdown of the DAX query time and the visual rendering time, helping you understand where the bottleneck lies. If the DAX query time is high, then your efforts should be directed towards optimizing the DAX measures.

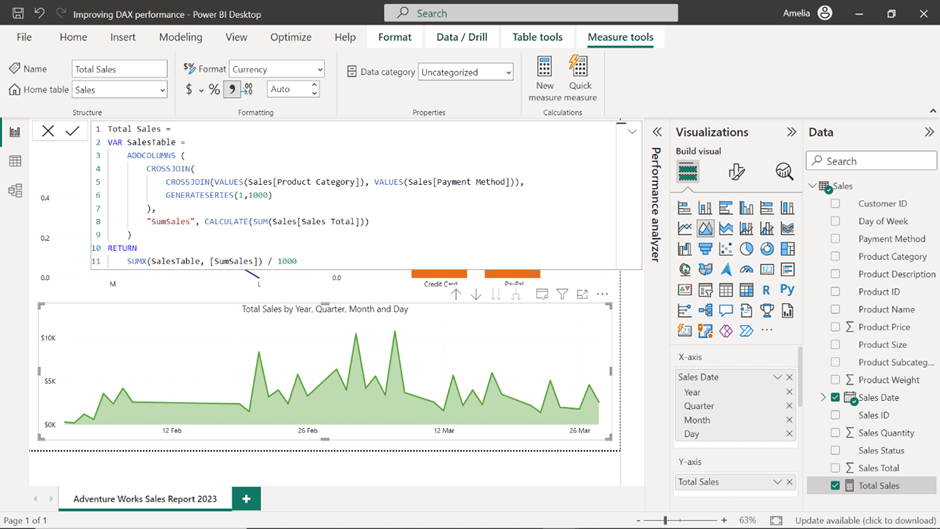


3. In this case, it appears that the **Total Sales by Year, Quarter, Month and Day** area chart is slowing down the report performance, as it has a considerably larger DAX loading time compared to other visuals.

**Step 6: Improve DAX performance**

Now that you have identified that the problematic visual is **Total Sales by Year, Quarter, Month and Day**, the next step is to refine and optimize. This might involve rewriting certain parts of the DAX formula for efficiency, eliminating unnecessary calculations, or simplifying complex ones. The goal is to reduce the computational load on the Power BI engine.

1. Locate the **Total Sales** field from the **Table** view on your right and select it to view the underlying DAX formula. This DAX formula inflates data with the nested **CROSSJOIN** operations, creating a much larger table. For each row, it recalculates the **Total Sales** using **CALCULATE**, a resource-intensive operation. Aggregating this massive table again with **SUMX** further strains performance, making the entire computation slow.



2. To simplify the DAX formula, eliminate the nested **CROSSJOIN** and **GENERATESERIES** functions. Instead use the **SUMX** function enclosed with a single **CROSSJOIN**:

Total Sales =

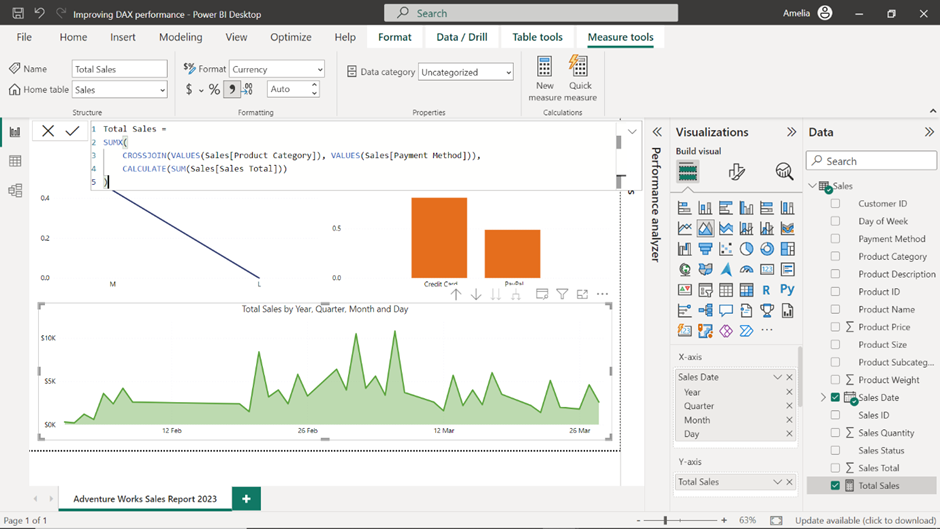
SUMX(

    CROSSJOIN(VALUES(Sales[Product Category]), VALUES(Sales[Payment Method])),

    CALCULATE(SUM(Sales[Sales Total]))

)

3. The **SUMX** function iterates over each unique combination of **Product Category** and **Payment Method** and calculates the **Total Sales** for that specific combination using **CALCULATE (SUM (Sales [Sales Total]))**. After calculating the sales for each combination, **SUMX** then sums up these individual results to produce the final **Total Sales**.



**Step 7: Test and review**

* Finally, re-run the **Performance Analyzer** to test if the optimization was successful. Optimizing DAX is often an iterative process. Once changes are made, it's vital to test and review the impact. Sometimes, what seems like an optimization might not have the desired effect, or there could be unintended side effects. Regular testing ensures that you're moving in the right direction.

