IBI_Unit 1_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127445825

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:00 PM GMT+5:30

File Name

IBI_Unit 1_V3.docx

File Size

281.3 KB

23 Pages

5,323 Words

33,083 Characters

1% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

2 Not Cited or Quoted 1%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **2** Not Cited or Quoted 1%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Internet

www.access.run **<1%**

2 Internet

docplayer.net **<1%**

Unit 1: Fundamentals of Business Intelligence

Learning Objectives

1. Understand the practical implementation of Business Intelligence tools in real-world business scenarios.
2. Identify the key challenges organizations face when adopting BI systems.
3. Analyze how data-driven decision-making improves operational efficiency and competitive advantage.
4. Evaluate the role of BI in strategic planning and performance monitoring.
5. Examine the integration of BI tools with other enterprise systems (e.g., ERP, CRM).
6. Interpret BI dashboards and reports to extract actionable insights.
7. Apply BI knowledge to propose data solutions for specific business problems

Content

1.0 Introductory Caselet

“Ravi’s BI Challenge: Converting Retail Data Into Business Insights”

Background:

Ravi is an Area operations manager for “FashionFusion”,an expanding retail clothing chain with 100+ stores that span across several cities. Ravi is required to oversee the performance of the stores and its inventories, and support the management team in taking well-informed decisions.

FashionFusion has recently deployed a Business Intelligence (BI) solution that centralizes data from their point-of-sale systems, loyalty program, inventory database and online sales interface. The fact that the BI system exists, however, is little consolation to Ravi who can’t previously access the insights he is seeking. Store-level reports are strewn about like yesterday’s news and key performance indicators are usually stuck in some well-buried, overly complex spreadsheet.

For about a quarter of an hour, sales collapse in a number of shops - but Ravi can't figure out what's going on at first. He touches base with the central BI team asking for assistance. They provide him with an interactive dashboard of sales trend by product category, store location and customer segments. The dashboard shows one product line is underperforming in urban stores while it made its way to store shelves late and just received bad customer reviews online.

As a result, Ravi collaborates with the merchandising team to make changes in inventory allocations and runs targeted promos in the regions impacted. The next month, the sales rebound and customer sentiment improves.

This example shows us how BI can help you, when applied in the correct way, turn raw data collected from your operations into actionable insights that drive decision at different levels within the business.

Critical Thinking Question:

How would you, if you were Ravi, ensure that your team leverage the BI tools to make decisions on a daily basis? How can we increase data literacy in nontechnical personnel, and how will real-time analytics impact the way performance is managed across retail?

1.1 Introduction to Business Intelligence

Business Intelligence (BI) has become a necessity for organizations in the modern era. It allows organizations to “operate with data in real time, where it resides.” This section describes an overview of BI in terms of definition, characteristics and scope of BI and its strategic involvement for decision-making.

1.1.1 Definition of Business Intelligence (BI)

Definition of Business Intelligence (BI) It is the term meant for technologies systems practices and applications that are necessary for the collection integration analysis and presentation of business information. The goal of BI is to enable better decision making at all organizational levels.

BI technology processes large amounts of raw data generated from business transactions or other activities, then turns it into knowledge that companies can use to make sound business decisions. It covers historical analysis – Descriptive Analytics (what has happened) and predictive analytics (what will probably happen).

Put another way, BI enables companies to address important questions like:

- What's happening in the business?
- Why is it happening?
- What could happen next?
- What should we do about it?

1.1.2 Characteristics and Features of BI

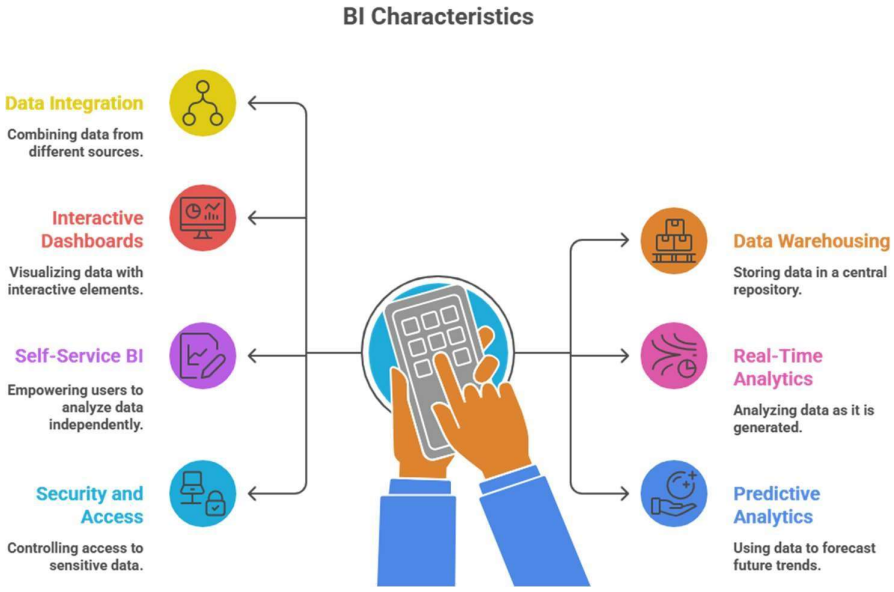


figure: **Characteristics and Features of BI**

Some salient features and attributes of Business Intelligence are:

Data integration – Bring data from various departments like sales, finance, marketing, HR and logistics in one place for analysis which most BI tools do.

Data Warehousing: BI frequently requires a data store (with relatively large amounts of structured information stored in one place) that is available for reporting and analysis.

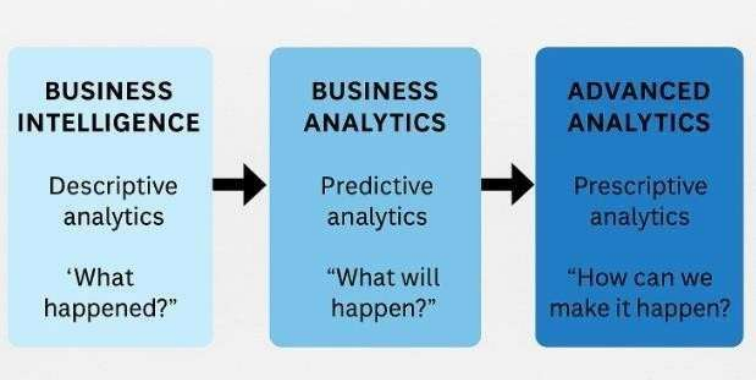
ERP Implementation And Integration: Visualisations such as charts, graphs, and tables are made available in BI platforms that make complex data easy to understand.

Real-Time Analytics: When you need to respond to something in real-time, (e.g., for a business decision), Real-Time analytics provided by modern BI tool can play a valuable role.

Self-Service BI: Business users can create their own reports and analyze data without IT intervention.

Predictive and Prescriptive Analytics: Predicting future outcomes and advising the best course of action, BI tools leverage mathematical calculations and machine learning.

Security and Access Control – BI systems offer secure access to data and allow organizations to limit who can view or alter sensitive details.



1.1.3 Scope of BI in Modern Organizations

BI is broad and expanding. It affects nearly all operations inside a company and is employed by employees at every level of the organization — the top rung to lower-rung staff. Uses There are several typical applications for BI :

• Sales and Marketing: Sales analysis, customers preferences, campaign effectiveness.

• Finance: Tracking expenditures, profits, forecasts and compliance reporting.

• Operations: Inventory management, logistics, the efficiency of supply chains.

• HR: Evaluation of performance of employees, hiring activities and rate of turnovers.

• Customer Service – addressing customer issues, feedback knowledge and service efficiency.

BI—even when used internally—can tell organizations what market trends look like, how competitors are faring, and other customer behavior that might require a change in strategy.

1.1.4 Importance of BI in Strategic Decision-Making

Strategic decisions are long-term, and affect the overall direction of an entire organization. BI supports this by:

Quality Information: Decisions must be based on information and not assumptions.

Trends Detection: Business intelligence can display long-term trends of customer behavior, sales or operational performance to allow managers to plan in the future.

Better Foretelling: BI relies on historic data and predictive algorithms to predict the future, making it possible for proactive planning.

Assisting Risk Management: With the knowledge of any potential threats or inefficiencies, companies can mitigate risks by taking action in a timely manner.

Innovating: BI data may uncover unsatisfied customer needs or market gaps that pave the way for new products and services.

Boosts Agility: When firms are able to swiftly make sense of data, they can respond more readily to disruptions in the market or within their own industry.

In conclusion, Business Intelligence is more than just a technology; it is an asset with strategic importance for any organization who needs to make faster, smarter and better decisions.

1.1.4 Importance of BI in Strategic Decision-Making

Strategic decision making is the process of considering long-term business goals and making strategic choices, which collectively lead to the overall direction of an organization. BI supports this by:

When You Know Facts Somehow, decisions that are made based on real numbers and real facts seem to be so much more reliable than those made from assumptions.

Spotting trends and patterns: BI tools can reveal long-term trends in customer behavior, sales or operating conditions, enabling managers to plan for the future.

Enhanced Forecasting: Historial data and weighted models are used to predict future trends, enabling proactive plans to be made.

Supporting Risk Management: Based on awareness of potential threats or inefficiencies, companies can be provided with early warnings to avoid risks.

Innovating: BI data may expose a gap in the market or needs that have not been satisfied, creating new opportunities for products and services.

Improving Agility: Being able to understand data in days vs weeks allows companies the flexibility to change direction or respond more effectively to changes in the market and the business.

Summing up, we can say that Business Intelligence is not a technological tool as it is a strategic asset of modern businesses by helping them make better, faster and more informed decisions.

1.2 Evolution of BI Tools

Business Intelligence tools have evolved in a number of ways—from basic spreadsheet analysis to complex, interconnected systems that provide real-time data as well as automated and predictive functionality. This part describes how the BI has disrupted over time, allowing companies to easily get and dispose of data for decision-making.

1.2.1 Early BI Practices (Spreadsheets and Excel)

Spreadsheets dominated data analysis during the early days of Business Intelligence (BI), particularly via Microsoft Excel. This was the prevailing paradigm of the '80s and '90s, with Excel being both accessible and familiar to business users. For a long time, spreadsheets were the first step for anyone trying to analyze structured data.

Early BI Tools (Excel-based) Key Features:

• Manual input and formula-based operations

• Simple analyses using pivot tables and charts

• Macros for automating repetitive tasks

Scarce resource (performance problem with big data)

Limitations:

• Problem of managing complex and large set of data.

• High likely hood of Formula mistakes/manual entry errors

• No live link or refreshes of real-time data

• Teams cannot work collaboratively and version control is limited

Spreadsheets continue to be common, if rudimentary, BI tools that just do not have the features or function- ality required for enterprise data management and more sophisticated analytics.

Evolution of Business Intelligence Tools

What we know as Business Intelligence today really arrived in the 90s, and didn't even go mainstream until businesses started to gravitate towards specialized software for data warehousing, visualizatio n and analytics. BI tools then advanced to offer real-time reporting, self-service dashboards and predictive analytics.

• Tableau: Launched in 2003, Tableau revolutionized BI by making data visualization and interactive dashboards the priority. Its drag and drop-interface allowed people without tech skills to visually look at data, democratizing BI for business managers and decision makers.

• Power BI: Microsoft introduced Power BI to complement Office 365 and Azure services in 2015. It has progressed through regular updates and versions, most recently delivering cloud-based BI with AI-driven insights and excellent Excel integration. The low overhead of Power BI and great integration with Microsoft, made it a go-to when considering enterprises large or small.

1.2.2 Rise of Visualization Tools (Tableau)

With companies having to cope with sophisticated and large data volumes, the demand for improved data visualization and interactive reporting also increased. New data visualization tools, likeTableau emerged out of this.

Tableau caught on and was hugely successful because it simplifies the process of turning raw data into interactive dashboards, reports and charts in visual form without you having to be a techie.

Key Features:

• Move and resize charts, maps, and graphs with the same drag-and-drop interface

• Multiple data sources (Excel,SQL databases, cloud based services) connectivity

• Live data updates and interactive dashboards

• Simple sharing and collaboration via Tableau Server or Tableau Public

Advantages:

• Simplifies complex data

• Helps in recognising trends and outliers Perform this step for both train and test data, but note that there is a much larger dataset available.

• Used by many in marketing, sales, finance, and operations to gain quick insights

Limitations:

• Not as ideal for extensive enterprise level reporting by/without integration.

• More directed towards visualising the data rather than deep analytics or detailed counterfactual modelling

Did You Know?

“Did you know: #Tableau can pull in a live stream from social media and plot Twitter hashtag trends using APIs?” This function enables organizations to track real-time brand sentiment and post-campaign performance. And it’s not just business data — journalists and researchers use Tableau for public interest stories as well.”

1.2.3 Self-Service BI and Enterprise Integration (Power BI)

With self-service BI tools that offer visualization, data integration and advanced analytics all integrated within the same platform, the next evolutionary path of BI is here. Microsoft’s Power BI is one of the best known tools in this space.

Power BI enables technical and non-technical users to analyze data, generate reports and develop dashboards without significant IT support.

Key Features:

• Integrate with Excel, SQL Server, Azure and other Microsoft services

• Natural language query (Q&A capability) to ask questions in plain English

• AI-driven capabilities for anomaly detection and forecasts

• Auto refresh data and deliver reports on a schedule

Enterprise Integration Capabilities:

• Scaling to whole organization

• Role-based access control for security

• Centralized data modelling and governance

Benefits:

• Brings together data preparation, analysis, and visualization on a single platform

• Promotes a data-driven culture and enables users at all levels

• Reduces dependency on IT departments

“Exercise: Exploring Self-Service BI Data with Power BI Desktop”

Instruction to Students:

Install Power BI Desktop (free from Microsoft). Use the CSV shared with you including employee performance data in various departments and regions. Complete the following:

You now load the CSV file into Power BI.

Create at least three visualizations:

o Pie chart displaying the employees count by department.

o Bar Graph compared performances across locations 2.

o A filtered table of employee's name, roles and scores.

Use the "Q&A" feature to ask questions about Power BI such as:

o What is a typical performance score for the sales department?

o "5 highest rating employees.

Export your dashboard to PDF and send it along with a short blurb whether or not you found the process of self-service analysis easy (or didn't need IT!)

1.2.4 Comparison and Use of BI Tools

The selection of the Business Intelligence (BI) tool is based on organizational requirements, including the size of the organization, user roles and responsibilities for data creation and analysis, complexity of data types and the business purposes. Here, for your entertainment pleasure, is a feature by feature comparison of the most popular BI reporting tools as of right now with a general idea on new features in their newest versions.

Feature / Tool	**Excel**	**Tableau – Latest 2025.2.1**	**Power BI – Latest v2.146.1133.0 (Aug 2025)**
Target Users	Individuals, Analysts	Analysts, Business Users	Enterprise Users, Analysts
Data Capacity	Limited	Moderate to High	High
Visualization	Basic Charts	Advanced, Interactive	Advanced, Interactive
Real-Time Analysis	No (manual refresh)	Yes (with connectors)	Yes (with auto-refresh and AI support)
Integration	Limited	Moderate	Strong (Microsoft ecosystem)
Ease of Use	High (familiar tool)	High (drag-and-drop interface)	High (user-friendly & integrated)

Cost	Low	Medium to High	Low to Medium (depends on usage)
Enterprise Features	None	Limited	Yes (security, governance, semantic modeling)

Key Insights:

- Excel remains a popular tool for quick, small-scale analysis but is limited for enterprise data management.
- Tableau (2025.2.1) is widely used for its advanced, interactive visualization capabilities.
- Power BI (v2.146.1133.0, Aug 2025) is favored for its balance of affordability, integration with Microsoft services, and enterprise features.

1.3 Applications of BI in Business Decision-Making

Business Intelligence is not just a technical solution; it is a powerful tool that supports decision-making at all levels of an organization. From day-to-day operations to long-term strategy, BI provides the data insights required to make better choices and improve business outcomes.

1.3.1 Operational Decision-Making with BI

Operational decisions are made on a daily basis and affect the short-term running of a company. These decisions usually involve routine activities like inventory management, staffing, logistics, and service delivery. BI tools support operational decision-making by providing real-time data and performance dashboards.

Examples of Operational BI Applications:

- Monitoring daily sales performance across different store locations
- Tracking inventory levels to avoid stockouts or overstocking
- Analysing customer service ticket volumes to manage support staff

- Identifying process bottlenecks in supply chain or production

figure: Streamlining Business Operations

By employing BI in operational decisions, companies are able to react more quickly to changes, minimize mistakes, and run day-to-day processes more efficiently.

“Project: Build a Daily Sales Dashboard in Google Sheets or Excel”

Instruction to Students:

Download sales data for a make-believe retail store over 30 days (sample data may be supplied). Do the following in either Google Sheets or Excel:

Summarize the sales information by date, product category and units sold.

Create a dashboard that includes:

o A graph with lines representing all sales over time.

o Bar graph to compare categories of product.

o A filter to report provide sales specific weeks.

Answer a quick note based on your dashboard :

o Which days were the best and worst selling?

o Which product categories performed best?

o What business decisions could be driven by this data (i.e. What can the company be doing based on this analysis)? Turn in your spreadsheet and brief, 200-word summary of what you found.

1.3.2 Tactical Applications: Market and Customer Insights

Tactical decisions are concerned with medium-term planning and are generally taken by middle level management to meet the needs of a well-defined strategy. BI tools aid in analysing customer behaviour and market conditions, internal performance data are used to make better tactical decisions.

Common BI Tactical Applications:

• Studying shopping habits of customers to create focused marketing campaigns

• Profiling customers through demographics, behavioral and location specific partitioning

• Measurement of promotional offers and sales strategies

• Keep an eye on competition and market trends

BI provides businesses a better understanding of why customers do what they do and how the market affects business, empowering organizations to predict more accurately and act decisively.

1.3.3 Strategic Applications: Forecasting and Competitive Analysis

Strategic decisions are forward-looking and define a company´s general course. The Business Calculation Developed to be used by CEOs and IT decision makers and vice presidents of finance, VAR Business (R) Formula VP is a comprehensive business intelligence product that helps provide senior-level executives with management information for investment planning, resource deployment, growth strategies and competitive positioning.

Strategic BI Applications Include:

• Prediction of Sales based on historical data and market considerations

• Spotting market trends as they materialize to develop new business opportunities

• Competitive benchmarking based on industry data and performance measures

• Analysing risk scenarios with predictive analytics

These solutions enable leadership to eliminate uncertainty and make data-driven decisions that benefit the organization’s vision and long-term goals.

Role of AI/ML in Strategic BI

Strategic BI has been greatly empowered by Artificial Intelligence (AI) and Machine Learning (ML). Where traditional BI is largely descriptive and diagnostic, AI/ML models can reveal hidden patterns, forecast with astounding accuracy, and more accurately predict as more data comes online.

AI-driven forecasting example:

A retailer applies machine learning models to historical sales, customers' behavior, social network trends and macroeconomic indicators. The AI system forecasts demand for products in the next quarter at a regional level, and this allows the firm to manage inventory better to avoid stockouts and align its marketing campaigns with the forecasted demand from customers.

1.3.4 Role of BI in Data-Driven Organizations

A data-driven organization makes decisions and takes action based on hard facts rather than intuition or guesses. Business intelligence (BI) is essential to facilitating that culture; it provides access, intelligibility and actionability of data.

Key Functions that BI Plays in Data-Centric Businesses:

• Promote the use of performance data in planning by all line departments

• Extending beyond data silos and bringing together information across functions

• Driving Transparency & Accountability through Trackable KPIs.

• Fostering innovation through insight and evidence-based experimentation

For companies who embrace data, BI isn't just a tool— it is the company's mind. People in every level of employment count on the business intelligence solutions for insights to carry out their tasks more efficiently.

Example of a Data-Driven Organization

You asked about the trends and well, one of them is definitely going to be data-driven companies such as Amazon. It applies BI and advanced analytics to personalize customer recommendations, improve supply chain performance and pricing agility, as well as forecast demand. Amazon is data driven in every aspect of their business, and it works: they are the most efficient company on earth.

Did You Know?

"Are you aware that data-driven organizations have a 23X chances of getting clients, and they are 6 times more likely to keep the customers?" said Abdulla. Becoming a data-driven company isn't only about the technology, it's also about transforming the culture so that all decision-making is grounded in data."

1.4 Case Studies and Real-World Examples

The advent of Business Intelligence has changed the course of businesses by enabling them to take more informed decisions thanks to data insights. BI is used differently in different industries, according to the types and quantities of data they have at their disposal.

1.4.1 BI in Retail (Walmart, Target)

High volume store retailers process millions of transactional and customer records. BI enables them to optimize the supply chain, track inventory, understand customer preferences and drive sales.

Example – Walmart:

Walmart runs one of the world's biggest BI systems handling information from its worldwide operations. It tracks:

- Inventory updated in real-time for 1000s of stores
- Buying habits to replenish fast-selling products
- Local promotional plans basis regional sales trends.

Example – Target:

2 Target Predicts Everything from Pregnancy to Blizzard Stockpiling How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did_Target uses computers to predict everything from blizzards to credit card fraud and teenage pregnancies. One of its most famous BI use cases was mining purchase data to determine shoppers that possibly bribing babies –based on their purchasing patterns– so they could offer promotions.

Impact:

- Reduced stockouts and overstocking
- Increased customer loyalty through personalization
- Optimized pricing and promotions

1.4.2 BI in E-commerce (v: g., Amazon, Flipkart)

Enormous streams of data are created minute by minute on e-commerce platforms. It can help customize the shopping experience, maintain product catalogs and detect fraud attacks, as well as logistics planning.

Example – Amazon:

Amazon uses BI extensively to:

• Update product recommendations in near real-time by customer browsing and purchasing behaviour

• Forecast demand for warehouse restocking and delivery planning

• Track and control supplier performance and delivery schedules

Example – Flipkart:

Flipkart leverages the BI tools to extract valuable insights from customer reviews, monitor user activity, and drive its flash sales model. It also relies on dashboards to track daily orders, cancellations and return rates.

Impact:

• Improved customer satisfaction through personalization

• Improvement in logistics and warehousing efficiencies

• Better fraud detection and security

1.4.3 BI in the Banking & Financial Service Industry

The Banking and Finance community leverages Business Intelligence (BI) to improve customer service, identify fraud, manage risks, and meet regulatory compliance.

Use Cases:

• Risk and credit rating based on transaction history of clients

• Real time fraud detection by detecting anomalous patterns

• To issue personalized financial products for customer segmentation

• Regulatory reporting and tools to create automatic dashboards and audit trails

A Case Study – ICICI Bank (India): Source:. 2.

Today ICICI Bank employs BI to a great extent in areas, such as customer behavior analysis and better product development, _ RouteServiceProvider::www.ajiips.ciips94.conf_info_net_cp_dir] $%&'()*+, from where it offers good quality of service [2].

enhance operational efficiency. Dashboards are implemented to monitor performance measures across departments and for enhancing efficiency of the service centres.

Impact:

• Increased Loan Approval Rates and Minimised Credit Risk

• Enhanced product architecture and cross selling opportunities.

• Enhanced fraud prevention systems

• Enhanced regulatory compliance due to automated reporting, more accurate records and improved adherence of Reserve Bank Guidelines (RBI)

1.4.4 BI in Healthcare and Pharmaceuticals

Healthcare Business Intelligence (BI) is an essential tool for patient care analytics, medical researches and pharmaceutical innovations, hospital management. Crticial to leveraging BI in healthcare, providers and pharma companies are enabled to make data-driven decisions that enhance effectiveness and efficiency.

Example – Hospitals:

What Hospitals Track with BI Dashboards There is a wide range of data that hospitals are tracking via BI dashboards:

• Patient admissions and discharges

• Average length of stay

• Equipment usage and resource planning

• Clinical outcomes and staff performance

Example – Pharmaceutical Companies:

Pharma companies leverage BI to process clinical trial data, assess drug safety and determine market demand for new products.

Example – Apollo Hospitals:

BI is utilised by Apollo Hospitals to enhance diagnostics, decrease patient waiting times and to streamline the process of treating patients.

Impact:

• Optimizing decisions to produce favorable patient outcomes using data-driven protocols

• Efficient resource utilization in hospitals

• Increased speed and safety of drug development

• Increased adherence to health legislation and reporting requirements

Data Privacy and Ethics Note:

Application of BI in healthcare also brings the important questions about privacy, security, and ethics. Patient pool data, needs to be protected for trust and regulatory

compliance. In this scenario, standards like HIPAA (Health Insurance Portability and Accountability Act) from the global perspective in the

S. and the General Data Protection Regulation (GDPR) in Europe are important compliance frameworks that inform us how patient data should be stored, shared and analyzed responsibly.

1.4.5 BI in Technology and Consulting Firms

Tech and consulting firms leverage BI to optimize project delivery, examine client requirements and foster innovation.

Example – Accenture:

Accenture leverages its own internal BI platforms to track performance of projects, team productivity and utilization of budget. It also offers industry-specific BI tools to its customers.

Example – Microsoft:

Microsoft uses Power BI to systematize key functions, as well as for providing BI capabilities in the form of software-as-a-service (SaaS) to its customers.

Use Cases:

• Dashboards for the project health to track performance

• Talent management analytics

• Client engagement and feedback analysis

Impact:

• Accelerated completion dates and workmanship for capital projects

• Higher client satisfaction

• Stronger business performance through analytics

Knowledge Check 1

Choose the correct option:

Which one of the options below BEST describes BI?

A) Method for hardware installations in the data center

B) A tool for building websites

C) Business data analysis aids - Tools and techniques to support business data analysis

D) A marketing campaign strategy

What is the name of a BI tool that is well known for creating visual dashboards with drag and drop interfaces?

A) Excel

B) Tableau

C) PowerPoint

D) SQL Server

What is the benefit of Self-service BI tool?

A) Only developers can use them

B) They increase data storage capacity

C) Business users do not need IT to generate reports

D) None of the data is validated.

Which phase of this new decision making influenced by BI is being used to enhance the day-to-day business activities, like staffing or logistics?

A) Strategic

B) Operational

C) Tactical

D) Forecasting

Which of the following is not one of the characteristics of today's BI tools?

A) Real-time data visualization

B) Predictive analytics

C) Automatic content writing

D) Integration with multiple data sources

1.5 Summary

ϖ This chapter described the key ideas and uses of Business Intelligence (BI) in today's companies. It started with a realistic Caselet which demonstrated how BI can bring shape to raw operational data and help you have insights for the business. BI's definition was focused on data collection, analysis (processing of information from raw

data to extract meaning), and presentation through visualization to enable decision-making. The journey of BI tools was mapped from legacy spreadsheets to modern platforms (like Tableau, Power BI etc.) demonstrating their increase in functionalities and user friendliness.

ϖ The importance of BI in operational, tactical and strategic decisions was illustrated by means of examples from the applications like inventory control, market analysis and forecasting. Real-life case studies from retail, e-commerce, banking, healthcare and consulting showed how BI solution can be implemented to add value to the businesses via improved performance, customer satisfaction and business operations efficacy. This chapter makes it clear that BI is not only a tool: It is also a strategic asset used to create an analytics-stirred organization.

1.6 Key Terms

Business Intelligence (BI) - Applications that help gather, process and analyze business data to make decision making better.

Data Warehouse: The centre of storage that collects the integrated data from multiple sources acting as a source for BI systems.

Dashboard: A visual interface that shows KPIs and other metrics to monitor business performance.

Self-Service BI – Business intelligence applications that enable non-technical users to create reports and analyze data without needing IT.

Data Visualization: Displaying data in a chart, graph or map to reveal trends and patterns.

Predictive Analytics - a BI approach based on statistical models and machine learning which is used for estimation of future outcome.

Operational BI – The use of BI tools while making operational decisions, like a purchase or customer service.

Strategic BI – Such type of BI is used to analyze long term objectives, market trends and forecasts for businesses in order to develop a business strategy.

1.7 Descriptive Questions

Define Business Intelligence. How does it assist with decision-making?

Describe the main attributes and functionality of today's BI tools.

How has the BI tools progressed from a basic spreadsheet to more sophisticated software such as Power BI and Tableau?

In an organization, how is BI applied to make operational decisions, tactical decisions and strategic ones?

Give two examples of BI applications in the industry.

What purpose does BI serve in becoming a data-driven organization?

Compare features, uses and enterprise applications of Excel, Tableau and Power BI.

Explain how BI can enhance the browsing experience for customers in an e-commerce setting.

Describe how healthcare providers can find value in BI tools.

What are the shortcomings of early BI practices, and how do the updated BI tools fix them?

1.8 References

Ranjan, J. (2009). Business Intelligence: Concepts, Components, Techniques and Benefits. Journal of Theoretical and Applied Information Technology, 60-70.

Turban, E., Sharda, R (2020). Supporting Decision Making and Business Intelligence (11th ed.). Pearson.

Power, D. J. (2013). DS [Decision Support], Analytics, and Business Intelligence. Business Expert Press.

Tableau Software. (n.d.). Case Studies and Resources. Retrieved from https://www.tableau.com

Microsoft Power BI. (n.d.). Business Intelligence Resources. Retrieved from https://powerbi.microsoft.com

Watson, H. J. (2009). How to create Business Intelligence: A look at the past, present and future. Communications of the AIS, 25(1), 39–54.

Knowledge Check 1

C) Tools and techniques for business data analysis

B) Tableau

C) Business users shall design reports with no IT intervention

B) Operational

C) Automatic content writing

1.9 Case Study

Introduction

Business Intelligence (BI) is changing the way businesses operate and make decisions using data in real time. The retail sector, for example, is an industry with continually changing supply and demand dynamic, where BI tools offer critical visibility of inventory management, customer behavior and sales prediction. The BI capabilities of retail clothing chain FashionFusion provides a timely case study for how BI solves problems in the real world and improves operational efficiencies. Through a comprehensive single source of truth, the business changed inventory planning and eliminated waste with happier customers.

Background

FashionFusion FashionFusion is a mid-market retail chain with 80+ stores present in metro and tier-2 cities. The company had long struggled to get inventory just right — some popular items often sold out, while others went unsold for months at a time. The result was lost sales and high inventory carrying costs. Also store managers did not have the accurate information as well in time to take stocking decisions and used their intuition most of times or even delayed reports.

The company phased in a Business Intelligence solution to unify Sales, Supply Chain and Customer Feedback datatables into one centralized, variety rich dashboard. The system, which relied on real-time analytics and visualization tools, helped managers make informed stocking decisions and helped the central warehouse forecast demand with better accuracy.

Problem 1: Inventory Disparity Among Stores

FashionFusion had supply and demand mismatched in all its stores. Some stores overstocked items with poor sales, while others frequently went out of stock on high-demand goods, impacting customer experience and revenue.

Solution:

The BI system combined the daily sales with store location, seasonality and demographic trends. Stock movement patterns were displayed through remote dashboards in real-time that suggested actions for automated

redistribution of inventory between stores. Demand became less ‘assumption of demand’, and weekly restocking was decided by numbers instead.

Problem 2: Implicit Need for Better Predictive Demand Forecasting Insights

Before implementing BI, demand projections were based on gut feel and history averages which frequently did not capture evolving customer habits or market changes.

Solution:

The BI solution was leveraging predictive analytics that moved out of historical data, seasonal-sales practices and customer preferences. Product-level demand for each region was predicted using machine learning algorithms. This allowed the purchasing team to get closer in advance for ordering, cutting excess and shortages.

Problem (3): Poor Visibility to the Store Manager Level

Managers had no way to track sales trends, inventory or customer preferences. This hampered ability of providers to be proactive about what the local demands were.

Solution:

The company rolled out BI dashboards accessible on mobile devices for store-level management. The dashboards showed performance indicators like number of sales per day, top-selling items, stock warnings and customer comments. This increased local flexibility and decentralized decision-making at store level.

MCQ 1:

The primary driver behind FashionFusion's decision to implement a BI tool for inventory control was:

A) To reduce employee workload

B) To automate payroll processing

C) To deal with discrepancies in stocks and enhance the prediction of demand

D) To improve social media presence

Answer: C) To reduce stock demand and to forecast a better demand.

Explanation: To resolve the main problem of overstocking and understocking in store locations, we have implemented BI system to give accurate realtime demand forecasting.

MCQ 2:

And how did the BI system better demand forecasting?

A) By having store managers predict future sales festivals.

B) Using predictive analytics on past and current history data

C) On the basis of sales of previous years only

D) Outsourcing demand planning to third parties

Answer: B) With predictive analytics on historical and real-time data

Explanation: The BI product leverages data from multiple sources and predictive models to accurately predict demand at the product level.

MCQ 3:

Question 7: What were the main characteristics of the BI dashboards available to store managers?

A) They only displayed head office level reports.

B) They were hard to use if you didn't know how to program

C) They gave instant access to information regarding sales, inventory and customer responses

D) They were utilized in isolated instances once a year

Answer: C) They gave real-time information on sales, stock and customer feedback "The dashboards were putting insights in front of store managers' eyes every single day, allowing them to make that decision much faster," DeBrun said.

IBI_Unit 2_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127444739

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 3:59 PM GMT+5:30

File Name

IBI_Unit 2_V3.docx

File Size

333.0 KB

22 Pages

5,288 Words

30,033 Characters

2% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

7 Not Cited or Quoted 2%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

2% Internet sources
0% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **7** Not Cited or Quoted 2%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

2% Internet sources
0% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
www.analyticsvidhya.com		<1%
2	Submitted works	
Queen's College on 2023-09-04		<1%
3	Internet	
www.coursehero.com		<1%
4	Publication	
Benjamin Kettner, Frank Geisler. "Pro Serverless Data Handling with Microsoft Az...		<1%
5	Internet	
initiumlearning.com		<1%
6	Internet	
www.classace.io		<1%

Unit 2: Getting Started with Power BI

Learning Objectives

1. Understand the Power BI Ecosystem, including its components such as Power BI Desktop, Power BI Service, Power BI Mobile, and Power BI Gateway.

2. Install and configure Power BI Desktop on their local systems, including setting up the environment for connecting to various data sources.

3. Navigate the Power BI user interface, including key sections like the Report View, Data View, Model View, and the Fields and Visualizations panes.

4. Explore and utilize core features of Power BI, such as importing data, creating visualizations, building dashboards, applying filters and slicers, and publishing reports to the Power BI Service.

5. Differentiate between data sources and data modeling tools within Power BI and understand how data transformation is managed using Power Query Editor.

6. Apply fundamental BI skills by building basic interactive reports and dashboards using real-world data sets.

7. Evaluate the advantages of Power BI in the context of self-service BI, enterprise integration, and real-time decision-making.

Content

2.8 References

2.9 Case Study

2.0 Introductory Caselet

"Ravi's Reporting Bottleneck: How to Unlock Data with Power BI"

Background:

Ravi is a regional sales manager at FreshMart, one of the fastest growing grocery retail chain with more than 150 stores in leading Indian cities. This involves monitoring store-wise sales performance, stock turn and analysing sales trends to help make decisions on promotion or supplier replenishments.

Ravi gets static reports in Excel from central IT once a week. These reports typically appear slow, non-interactive, and do not capture and present current events such as sales or stock "live on the wire." Consequently, Ravi is not able to react to stockouts on fast-moving items and can't get pricing or promotions updated. His team also struggles to compare store performance across regions without spending hours manually scrubbing and analyzing the data.

Having expressed these concerns in a management meeting, The leadership of the company agrees to deploy Microsoft Power BI so that their reporting system can be transformed. The BI team deploys Power BI Desktop to the analysts and starts publishing interactive dash boards to the Power BI Service, where sales managers like Ravi can access them.

For the first time ever, Ravi would be able to filter reports by date, product category or location; see performance through the previous day in real-time; and receive visual alerts for stock anomalies. He soon realizes that sales of diary products shoot up on weekends in retail shops at cities, and he collaborates with the procurement team to alter inventory levels accordingly.

In a matter of weeks, the organization is seeing this effect: less waste, higher product availability and faster decision-making at every level.

Critical Thinking Question:

As Ravi, what key things would you want to see in a Power BI dashboard to allow you to better steer your region? What kind of problems could interactive reporting be utilized to solve for daily operational issues that static reports are incapable of resolving?

2.1 Power BI Ecosystem

Power BI Ecosystem Power BI ecosystem is Microsoft's well-rounded tool-set that helps an organization in every step of data analysis journey starting from importing and transforming data to analyzing, visualizing, sharing, and finally accessing the insights out of it whenever or wherever needed. It serves a variety of users from the soloist to corporates by giving flexibility on platforms.

2.1.1 Power BI Desktop, Service, and Mobile Components

The Power BI ecosystem is broadly divided into three areas, and each area is designed to fulfil a specific role in the end-to-end analytics process:

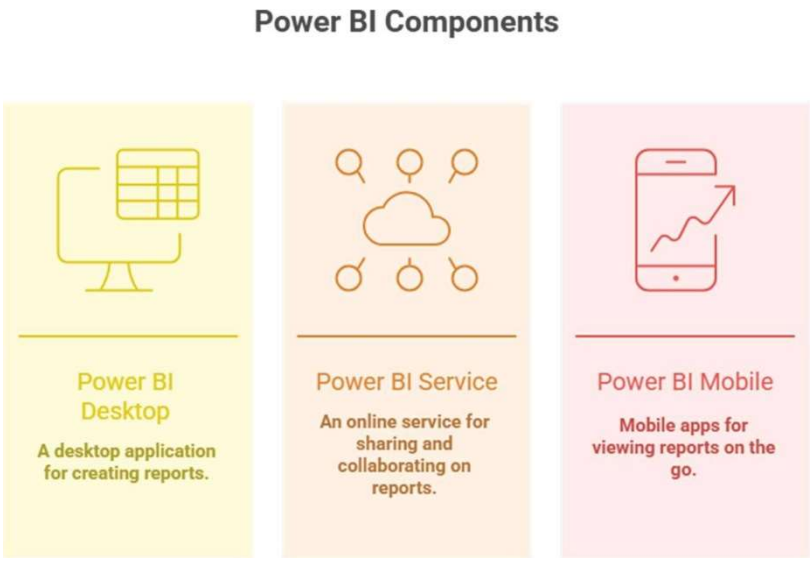


Figure: **Power BI Components**

Power BI Desktop

o A cost-free Windows application for designing reports and dashboards.

o Great tool for analysts and report writers.

o Connect to your Data Source, perform transformations and shape data relations. Design a variety of visualizations including tables, grids, bands etc.

Power BI Service (Power Bi. com)

o A cloud-based service for publishing, sharing and collaborating on reports and dashboards.

o Supports real-time dashboards, scheduled data refreshes and ensuring secure sharing with users authorized to use it.

o Offers capabilities for workspace, KPI alerting, distribution of insights across teams etc.

Power BI Mobile

o Apps for Android and iOS.

o Access reports and dashboards anywhere, anytime.

o Supports push notifications, drilling-down, as well as updating live to keep users informed regardless of location.

Collectively, these three pieces also enable the complete BI workflow: create (Desktop) → share (Service) → view and act (Mobile).

Note on Power BI Gateway

Power BI Gateway is an essential component for enterprise scenarios that help connect on-premises data sources with Power BI Service in a secure way. It enables scheduled refreshes and real-time data access while keeping sensitive content on premises. Businesses use gateways to secure data, enable compliance and ensure that their on-premises and Power BI cloud services are working together.

2.1.2 Role of Power BI Desktop in Data Modelling

The mind of any BI solution is the data model in Power BI Desktop. Data modelling is the process of structuring and arranging data ready for analysis.

Some of the data modelling features available in Power BI Desktop include:

• Data Import: Access data from various sources, including Excel, SQL Server, and cloud databases.

•Power Query Editor: Clean, actualize and shape the data before the modelling stage.

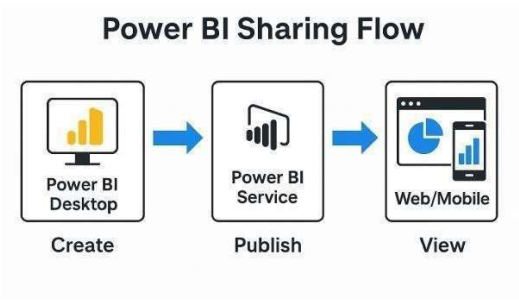
• Data Relationships: Establish relationship between tables using primary keys and foreign keys concept (as a relational database).

• Calculated Columns & Measures: Utilize DAX measurements to add new information for in depth analysis.

• Hierarchies and Formatting: Create Year, Quarter, and Month time hierarchies for decomposition and manage data navigation through grouping.

With rich, intelligent visuals comes in-depth and well-articulated analysis which can be done with Power BI Desktop– the brains behind the creation of Power BI ecosystem.

Visual Representation: Sharing Flow in Power



2.1.3 Power BI Service for Collaboration and Sharing

4 After you've created a report in Power BI Desktop, you can publish it to the Power BI Service so that everyone can read and interact with it across teams and organizations.

Functions of Power BI Service:

- Sharing Dashboards: Share visuals with stakeholders in minutes and experience the object of truth via workspaces, and apps.

3 • Schedule Data Refresh: Keep your data fresh by connecting to your on-premises data sources without the need to move the data.

• Role-Based Access: Permissions based, control what you are allowed to see or do.

o For example, a finance manager can view and edit financial reports but sales personnel has read-only access to sales dashboards. Managers could get summarized KPIs without viewing sensitive transactional data.

- Collaboration: Comment, tag users and get alerted on important metrics.

• App Workspaces: Establish a place where teams can nurture and develop datasets, reports, and dashboards together.

Power BI Service changes single reports into corporate assets by providing access to the data in a secure and collaborative environment.

Did You Know?

"Did you know that Power BI Service allows you to create data alerts on KPIs and numeric visuals? For instance, set up auto-email messages to be sent when sales plummet below a specified amount or stock descends beneath a certain threshold. This takes Power BI beyond being just a reporting tool, but also a proactive monitoring system."

2.1.4 Power BI Mobile for On-the-Go Access

Power BI Mobile allows business users to keep a pulse on their businesses remotely. It provides a safe and responsive mobility experience delivering insights in real time.

Features of Power BI Mobile:

• Interactive Reports: View, interact with and filter reports directly on your mobile device.

• Push Notifications: Ability to define alerts on data (e.g., sales are below a threshold).

• QR Code scanning - Get retail or manufacturing location-relevant reports.

• Offline Access: Download dashboards to see them even while you are offline.

• Consistent Experience: Your mobile reports look and feel just like any other report, using the aliases and expressions that you are familiar with on Desktop and Service.

With Power BI Mobile, you can make informed decisions using advanced insights with data as close to real time as possible. Power Bi Mobile UI Screenshot

2.2 Installation and Configuration of Power BI Desktop

Analysts and developers primarily use Power BI Desktop to create data models, design reports, and build dashboards. This section describes how to set up the tool, configure it for initial use and hook it up to data sources for analysing.

2.2.1 Requirements and Installation Process System Requirements

Before you begin You should ensure that your system meets the minimum necessary requirements to run Power BI Desktop. These prerequisites are shown in the following table:Above table is showing these prerequisite's.

Component	Requirement
Operating System	Windows 10 or later (Power BI Desktop is not supported on macOS)

.NET Framework	Version 4.7.2 or later
Memory (RAM)	Minimum: 4 GB; Recommended: 8 GB or more for large datasets
Disk Space	At least 2 GB of free space
Processor	1 GHz or faster; x64 architecture recommended
Display	Minimum screen resolution of 1440 × 900 pixels

These system requirements help ensure that Power BI Desktop installs and runs successfully, as well as performs optimally when working with your data.

Installation Process

Power BI Desktop installation is a step-by-step process as follows:

Get the Installer: download Power BI Desktop from Microsoft's official Power BI website or get it on the Microsoft Store. Installing Lovid is generally recommended through the Store for updates.

Execute the Wizard: Open the installer file and follow the instructions in the setup wizard on your screen.

Completely Installed: Now you can launch Power BI Desktop from the Start Menu or with a desktop shortcut.

Sign In (Optional) : Optionally sign in using a Microsoft or organizational account to use features like publish reports to Power BI service.

Overall, the installation procedure is pretty quick and takes only a few minutes, depending on your system resources.

Common Installation Errors and Resolutions

Although it may appear simple, sometimes installation problems do occur. Below is a table explaining errors and their remedies:

Error or Symptom	**Likely Cause**	**Resolution**
Missing ".NET Framework"	Required framework not installed	Download and install .NET Framework 4.7.2 or later from the Microsoft website
Installation process freezes or crashes	Insufficient system resources or conflicts	Restart the system, close unnecessary applications, and retry installation

Application fails to launch after installation	Corrupt installation or access restrictions	Reinstall Power BI Desktop; try running as administrator
Issues with Microsoft Store installation	Store cache errors or connectivity issues	Run wsreset.exe to reset the Store cache; check internet connectivity
Application crashes when handling large data	Low memory or CPU performance limitations	Upgrade memory to 8 GB or higher; reduce data size or optimize data sources

Troubleshooting Guidance

In the event that installation fails or the application behaves unexpectedly after installing, try the following:

• Turn your computer off and re-install it.

• Check that operating system updates are up to date.

• Temporarily turn off antivirus if it's blocking the installation.

• If your system is corporate or institution-managed,confirm with the administrator that you haven't been imposed one of the administrative restrictions / group policies.

• If you need any more technical support, feel free to contact Microsoft or check its official documentation and community forums.

2.2.2 Initial Setup and Configuration

So, after you get Power BI desktop downloaded and installed you are greeted with a clean interface asking what to do once installed.

Initial Setup Includes:

• Sign in (optional) – No requirement to sign in for report authoring, but it's needed to publish to the Power BI service.

• Language: Settings can be changed in options menu.

• Preview Features: Enabling beta features in the "Options" → "Preview Features" menu!

• Default file settings: New files can open in DirectQuery or Import mode by default.

Configuration at this level makes it possible for the software to match the users environment and preferences.

2.2.3 Connecting to Data Sources

Power BI features robust data connectors that allow users to connect, import, and unify data from diverse sources. Such versatility facilitates smooth connection to structured and unstructured data ranging from on-premises databases to cloud-based platforms.

Popular Data Sources

Power BI can connect to the following types of data sources:

Category	Examples
Flat Files	Excel, CSV, XML, JSON
Databases	SQL Server, MySQL, PostgreSQL, Oracle, Microsoft Access
Cloud Services	SharePoint, Azure, Google Analytics, Salesforce
Online Sources	Web pages (HTML tables), REST APIs

This extensive support makes Power BI an ideal tool for integrating disparate data systems within a unified analytical environment.

Connecting to a Data Source Step-by-Step Instructions

To connect to any supported Data Source the following steps are common:

Navigate to: Home → Get Data.

Choose the correct source of data here; you could pick from all your different connectors (Excel, SQL Server, Web).

Provide connection details such as file path, server address, auth type or API key.

Preview the data, and then select the tables, sheets, or data ranges that you want to connect to.

Load the data into your model directly or select Edit to open it in Power Query Editor.

With Power Query Editor, users cleanse and filter data, transform columns and change data types before loading their data into the analytical model in Power BI. This is done to ensure consistent and valid data, which is then visualized and reported.

Caselet: Connecting to Google Analytics

Use case: A digital marketing analyst has been asked to develop a performance dashboard which would be used to surface website traffic related stats such as sessions, bounce rate and user demographics. The information on this site is from Google Analytics.

Objective: Connect Power BI to Google Analytics and extract website performance data.

Steps:

Access the Connector:

o On the Home tab of Power BI Desktop, select Get Data → More.

o Navigate to the data connector dialog and choose Google Analytics.

Sign In to Google Account:

o The first time a user connects, Power BI will require that the user sign in using a valid Google account.

o Power BI must be allowed to connect with the Goggle Analytics account.

Select the Account and View:

o The list of Google Analytics accounts, properties, and views (available to the signer in) Power BI is shown after signing in.

o User chooses for the desired view (e. grievance, “All Website Data”).

Choose Metrics and Dimensions:

o A metrics and dimensions lists are presented (sessions, pageviews, bounce rate; source, device category, country).

o The user chooses those fields for analysis.

Load or Transform Data:

o The information can be previewed and then either loaded immediately or transformed within the Power Query Editor if you want to filter time periods, change a column name or merge with other data sets.

Model and Visualize:

o Once data has been loaded, you can use it to build visuals: line charts, tables and even dashboards showing how web traffic patterns have evolved and where users went.

Note: Google Analytics connector uses the Google Analytics Reporting API, which is subject to API quota limits on standard dimensions and metrics. API customization or third-party connectors may be necessary for custom dimensions or metrics.

"Task: Create a Product Sales Dashboard with Excel as the Data Source"

Instruction to Student:

You have been given an excel-sheet which has the monthly sales report of product in three different regions.

Launch Power BI Desktop and open the supplied Excel file.

First load the data using the function “Get Data”.

Use Power Query to:

o Remove null values

o Rename columns for clarity

o Filter data for this year

Feed the preprocessed data into your model.

On a report page, add the following visuals:

o Column chart explaining the sales region wise

o Pie chart for product category contribution

o A card showing total revenue

Deliverables:

Save your file as a PDF and upload it with a brief description (150-200 words) of how cleaning your data helped make the report more informative/interesting.

2.2.4 Import vs. DirectQuery Modes

Power BI has two basic ways of accessing external data. They are Import mode and DirectQuery Mode. There’s pros and cons to each mode so the choice comes down to what amount of data you are working with, how quickly you need to query it, and if you require real time updating.

Comparison of Data Access Modes

Feature	Import Mode	DirectQuery Mode
How It Works	Loads a snapshot of data into Power BI	Connects live to the data source without storing data
Performance	High performance; data is cached locally	Slightly slower; performance depends on the source system
Data Size Limits	Limited by local system memory (RAM)	No strict limits; relies on the capabilities of the backend
Offline Access	Fully available offline after data import	Requires continuous connection to the data source
Data Refresh	Requires scheduled or manual refresh	Queries data in real time on every interaction
Modeling Features	Full support for DAX, calculated columns, and	Some limitations in DAX functions and data modeling

	relationships	

Visual Comparison: Pros and Cons

To assist in users' selection of these two modes, the comparisons of each mode based on their advantages and disadvantages are listed.

Import Mode Pros:

• Quicker execution through the storing of data locally

• Complete access to DAX bringing advanced data modeling capabilities

• Offline mode usage once the data is loaded

• More appropriate for complex dashboards and transformations

Cons:

• Data may become outdated between refreshes

• Dependent on the local machine memory size

• No On-Demand Query: Need to schedule for data refresh regularly

DirectQuery Mode Pros:

• Auto Updated: Always kept up to date, immediate reflection of changes in input data

• No storage or memory limitations

• Good for very big datasets or enterprise systems

Cons:

• Relies on continuous connectivity

• Live query execution could make the result slow

• Some DAX functions and visual interactions are not fully supported

When to Use Each Mode

• Use Import Mode when:

o Your system's memory is enough to contain the data set

o Good speed and offline support is a must

o Need to do advanced modeling and calculated columns

• Use DirectQuery Mode when:

o Handling very large databases that cannot be managed locally

o Real-time updates are essential

o You do not wish to build up valuable locally stored sensitive data.

2.3 Navigating the Power BI Interface

Power BI Desktop is an intuitive and accessible software solution that will enable users to quickly load new information, build models on it, and create stunning visualizations and reports. This part of the article deconstructs the essentials of the interface, and describes how users make use of different elements when creating a report.

2.3.1 Home Screen and Ribbon Options

As soon as you open Power BI Desktop, the Home screen provides quick access to some of the most important tools that'll help you start working on a new report.

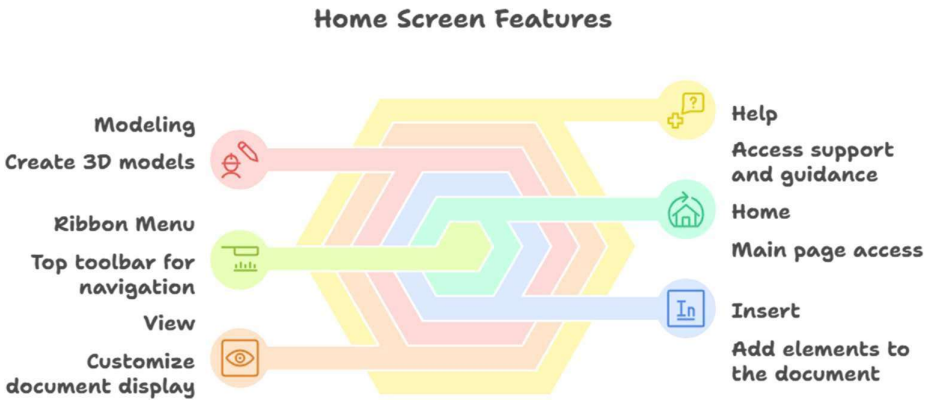


Figure: **Home Screen Features**

Home Screen Features:

• Start Page: Contains links like "Get Data," "Recent Sources" and "Open Report."

• Ribbon Menu (upper toolbar): Contains the following tabs:

o Home: To perform basics—Get Data, Transform Data, Manage Relationships, Publish.

o Wrap: Insert images, buttons or any type of graphic.

o Model: Build calculated columns and measures and create relationships.

o View: Switch on/off design elements (gridlines, snap-to-grid etc.).

o Help: Allows browsing of tutorials and documentation.

The ribbon is a set of toolbars at the top of Word, PowerPoint, and Excel that will eventually be in all Office applications to provide users with a global experience.

2.3.2 Report View, Data View and Model View

Power BI Desktop contains three main views that are in the left side of the window, arranged in a vertical panel. Each one has its own function in the report director process:

Report View (default view)

o Wherever you create and arrange your visuals (charts, graphs, KPIs).

o Drag and place fields on the canvas to design reports.

o Insert slicers, pictures, text boxes and buttons.

Data View

o You can check the real data processed in Power BI.

o Good for reviewing data transformations, and calculated columns.

o You see data in tables like Excel.

Model View

o Graphical relationships between the tables.

o Shows foreign key–primary key links.

o Allows users defining the c ar dinality and cross-filtering behavior as (one-to-many...) Switching between these views means you can go from raw data to polished dashboard with no friction.

2.3.3 Fields, Filters, and Visualizations Panes:

On the right side of Power BI Desktop, you will find three crucial side panels which govern the content of your report: 1. Fields Pane Table 1. Fields Pane o This displays all the tables and columns available in your data model. You are required to drag these fields onto the report canvas to generate visuals. o Calculated fields and measures also show up in this panel. 2. Visualizations Pane o This has icons for various chart types, including, but not limited to bar, pie, line, map, matrix. Also, allows for formatting of visuals: labels, colors, legends, tooltips. You can alter between visuals or change settings for any selected item on the canvas. 3. Filters Pane o This allows you to place filters at three different levels : • Visual-level filter : This is particular for one visual. • Page-level filter : This applies for the present report page. • Report-level filter : This applies across the entire report. Users can add fields here and set up the formalization (for instance: “Top N”, ranges, etc.). Combined, these three panels constitute the

command center for generating meaningful, filtered, and aesthetically appealing visual reports.

2.3.4 Customization and User Preferences:

Power BI enables users to modify the reporting experience and adjust visuals or settings distinctly for User Preferences Include: • Themes and Color Palettes o Pick from existing themes or upload personalized JSON themes. Implement corporate branding through fonts and color schemes. • Gridlines and Snap-to-Grid o Use gridlines for precision of visuals. “Snap-to-grid” assists you in placing the visuals accurately. • Default Settings o Set default summarization (i.e., Do not sum up numerical fields automatically). Change the default page size and resolution for exporting reports.• Q&A Setup o Train the Power BI natural language engine to discern business terms or synonyms for more accuracy. • Personalized Visual Interaction: o Customize how visuals react when the audience clicks on or filters an alternative visual. . equalTo or highlight These customization options improve both visual and user interaction by enabling reports to comply with organizational specifications or individual distinctions.

Did You Know?

“Power BI permits the customization of interactivity between visuals. For instance, instead of filtering, a visual can be set to highlight, take no action, or cross-filter another visual. This is because it enables customized storytelling and reduces the risk of false impressions.”

2.4 Overview of Core Power BI Features

Power BI is a collection of software services, apps, and connectors that work together to turn your unrelated sources of data into coherent, visually immersive, and interactive insights. The central technologies used for the Power BI capabilities are: power query, Power Pivot, and Power View as well as how they fit together to deliver a full solution.

2.4.1 Power Query: Data Extraction, Transformation, and Loading (ETL)

Power Query is the ETL tool part of Power BI. It offers a visual, no code environment to prepare data for analysis.

Key Functions:

• Import: Get access to data from anywhere – Excel, databases, cloud services, or the web.

• Transform:

o Remove or filter rows

o Rename columns

o Merge or split columns

o Replace values

o Unpivot data for normalization

• Load: After relevant transformations the cleaned data is loaded into the Power BI model to be analysed).

Behind the scenes Power Query utilizes a functional language, The M Language. However, most of what you do is going to be through the user interface.

One Use Case Example: This is a cleaned sales data where you can remove the null rows, convert date formats and merge different Monthly files into one table.

Did You Know?

This video also shows how Power Query can automatically identify and correct data types, combine several Excel files from a folder into one query, and keep track of each step involved in transforming a dataset in its “Applied Steps” pane. EDIT: You can EVEN reorder delete and edit steps through a versioning style editor WITH NO CODE!

2.4.2 Power Pivot: Data Modelling and Relationships

It is Power Pivot that provides the analysis engine for Power BI, where users can create models to represent data from different places (for example: T-SQL databases and Microsoft Azure tables) Table Power Pivot combines multiple tables into a single, uniform structure. New connections relationships, calculated columns, measures and hierarchies Table-based projects that form the basis for new relationships can now be created New queries using DAX on model data to enable robust forecasting scenarios submitted on the fly.

Core Functions of Power Pivot

Power Pivot improves Power BI data modeling by providing the following:

• Relationships between the tables (e.g., one-to-many, many-to-one) via primary and foreign keys

• Working with calculated columns and measures in DAX

• Creating a Hierarchical relationship (year --> quarter –> month) so that you can easily drill down.

• Filter and RLS (row-level security) for visibility / Access control of data

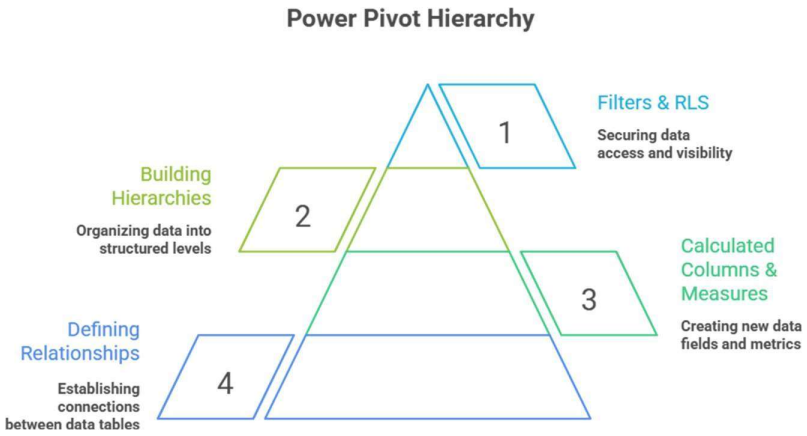


figure:**Power Pivot Hierarchy**

With these capabilities, Power Pivot lets your data professional sta create a semantic model where you then integrate the data from various systems and logically organize this information for fast retrieval and browsing.

DAX in Power Pivot

DAX (Data Analysis Expressions) is the formula language of Power Pivot which allows the creation of custom calculations. With DAX, users can apply time intelligence, build aggregations and define custom conditional logic between related tables.

The 3 most used DAX formulas:

Total Sales

Total Sales = SUM(Sales[Amount])

This measure is just summing up the values in the Sales[Amount] column.

Year-to-Date (YTD) Sales

YTD Sales = TOTALYTD(SUM(Sales[Amount]), Calendar[Date])

This formula is giving YTD, year-to-current-date, sales according to a calendar table.

Profit Margin

Profit Margin = DIVIDE(SUM(Sales[Profit]), SUM(Sales[Revenue]))

This equation can compute the profit margin, taking care of division by zero using the DIVIDE function.

These exemples show how Power Pivot uses DAX to introduce new and re-usable calculations in the data model.

Example Use Case

I've kept a common business problem of YTD sales and profit margins in a retail data. With Power Pivot, you can:

Import Sales Products Calendar as independent tables

Establish relationships between the tables (like Sales[ProductID] → Products[ProductID])

Create a YTD Sales measure and Profit Margin measure in DAX

See trends and KPIs with Power BI visuals, such as line chart or KPI indicators

This method guarantees repeatable computations and a scalable reporting structure, particularly when the underlying data is refreshed often.

Instruction to Student:

You have two datasets in an Excel format each:

• Customers Table: CustomerID, Name, Region

• Sales Table: OrderID, CustomerID, ProductName, Quantity, Revenue

Load the 2 tables into Power BI Desktop.

Navigate to Model View and link CustomerID between the two table.

With Power Pivot (Modeling tab) you can:

o Create a measure of Total Revenue with DAX: Total Revenue = SUM(Sales[Revenue])

o Add a measure for Total Quantity by using DAX: Total Quantity = SUM(Sales[Quantity])

Create a report page with:

o A bar chart for revenue by region

o A matrix of sum(quantity) by product and region

Deliverables:

Turn in your PBIX file and a brief description on how relationships enabled you to visualize insights from both of the tables.

2.4.3 Power View: Building Interactive Reports

Power View is the visualization engine behind Power BI, which enables users to create interactive, cross filterable reports with a number of different visuals. Although the name “Power View” doesn’t see as much light of day in the UI, its features are baked into Power BI’s Report View.

Key Visualization Features:

• Charts (bar, column, line, pie, waterfall, area)

• Tables and matrices

• Maps (with location-based data)

• Cards and KPIs

• Slicers and filters for interactivity

Especially pages can contain cross-filtering between visuals, drill-downs for deeper analysis and the conform system instead of tooltips.

Example Use Case: Making a sales dashboard, complete with monthly sales, regional performance, and top-selling products — Enables Users to interactively be able to traverse insights.

2.4.4 Features Integration for End-to-End BI Workflow (FI-BIW)

Power BI really shines in that it brings all of these together—Power Query, Power Pivot and Power View into a single tool allowing for seamless transition from one stage of the BI flow to another.

End-to-End Workflow Example:

Get Data from Excel and SQL Server using Power Query.

Process the data: clean and tidy it.

Now model the cleaned up data in Power Pivot including relationships and measures.

" "I'm not going to add new reports, just visualize my data with Power View to create dashboards and reports.

Publish and Share reports in Power BI Service – for collaborative consumption!

" Dale Bant, CTO - Identifor "Using Power BI means that we do not need to hire specialized report developers.

Knowledge Check 1

Choose the correct option:

Which of the following is NOT built into Power BI?

A) Power BI Desktop

B) Power BI Gateway

C) Power PivotTable

D) Power BI Mobile

What is that the primary function of Power Query on Power BI?

A) Designing charts and dashboards

B) Developing data model and DAX calculations

C) Extracting & transforming data before load.

D) Sharing reports across mobile devices

What functionality exist in Power BI Desktop to create relationships between tables and calculate measures?

A) Power View

B) Power Pivot

C) Data View

D) Q&A

In Power BI, where do users set a filter that applies on the whole report?

A) Report-level filter pane

B) Visualizations pane

C) Fields pane

D) Format tab

Which of the following most closely describes Power BI import mode?

A) Connects with live data directly at source

B) Data can continue to reside in its original database

C) Brings the data into Power BI for faster performance

D) Not admitting any transformation or modeling

2.5 Summary

23 ϖ This chapter was an overall introduction to the Power BI ecosystem, discussing its actual deployment and main functional blocks. Participants delved into the different components, such as Power BI -- Desktop, Service and Mobile -- and how they interact to enable organizations make data-driven decisions.

ϖ The chapter described the process of installing and running Power BI Desktop, which included managing views (e.g., Report, Data, Model), as well as how users could use panes such as Fields, Filters and Visualizations. Combine Power Query (ETL), Power Pivot (model) and Power View (reporting) to provide the most streamlined Business

Intelligence experience available. These tools combined make it simple for users to import, clean, model, visulalize and share data.

ϖ Empower users with Power BI at all levels of organizations to access and interact with data on the fly – in the office or on the go数据互交之旅：不限于任何一个场所，使用Power BI打开并与数据进行实时互动。

2.6 Key Terms

Power BI Desktop – A Windows installer application that provides data-connection and reporting and shaping features(Data Transformation and Modeling).

2 Power BI Service: cloud based service to publish, share and collaborate on reports and dashboards.

Power BI Mobile – A mobile app released for Android and iOS, enabling both dashboard access and management.

Power Query – A data extraction, transformation, and loading (ETL) tool.

Power Pivot- This is an option to do the data modeling, create the relationships and doing some advanced level of calculations using DAX.

Power View – The Power BI interface to report design for building interactive visualizations.

6 DAX (Data Analysis Expressions) – A formula language for Power BI used to create calculated columns and measures.

DirectQuery – This is a connection type that maintains the data in the source system and queries it live.

Import Mode – Requires that the data be loaded in Power BI, for optimal performance.

Visualization Pane—In Power BI Desktop, this is the area where users choose and arrange charts, maps, and visuals.

2.7 Descriptive Questions

Identify the elements of the Power BI ecosystem and describe how they interact.

Explain how to set up and configure Power BI Desktop.

What are the differences between Import and DirectQuery?

Discuss the function of Report View, Data View and Model View in Power BI.

Explain power query regarding ETL? Give an example.

How Power Pivot is related to the Modeling in Power BI?

Explain the use of filters in Power BI and distinguish types such as visual, page and report-level filter.

What are benefits of Power BI Mobile for business users?

How to tie together power query, Power Pivot and Power View into an end-to-end BI workflow?

Describe the role of user customization and preference in optimizing Power BI usage.

2.8 References

Microsoft Corporation. (n.d.). Power BI Documentation. Retrieved from https://learn.microsoft.com/power-bi

Ferrari, A., & Russo, M. (2019). Introducing Microsoft Power BI. Microsoft Press.

Chhabra, S. (2021). Mastering Power BI: Expert techniques for effective data analytics and business intelligence output. Packt Publishing.

Power BI Community Blog. (n.d.). Retrieved from https://community.powerbi.com

Pitre, R. (2020). Data Analysis and Visualization with Microsoft Power BI. BPB Publications.

Answers to Knowledge Check

Knowledge Check 1

1. C) Power PivotTable
2. C) Extracting and transforming data before loading
3. B) Power Pivot
4. A) Report-level filter pane
5. C) Loads data into Power BI for faster processing

IBI_Unit 3_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127444741

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 3:59 PM GMT+5:30

File Name

IBI_Unit 3_V3.docx

File Size

203.4 KB

17 Pages

3,768 Words

20,959 Characters

0% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

1 Not Cited or Quoted 0%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **1** Not Cited or Quoted 0%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Internet

www.wikihow.com **<1%**

Unit 3: Data Sources & Import

Learning Objectives

1. Connect to and access data from a variety of sources supported by Power BI, such as files, databases, cloud services and web data (section 3.1).
2. Analyze and compare Direct Query with Import mode under different use cases, including also a discussion on their pros and cons in terms of performance (Section 3.2).
3. Combine multiple data sources in a single report and model relationships between them successfully (Section 3.3).
4. Learn the basics of query creation applying power query, for example: loading and transforming data, sorting (Section 3.4).
5. Use fundamental ETL principles in Power BI through a no code interface to create clean, concise datasets for reporting.
6. Diagnose and resolve connectivity issues or data conflicts when working with live and off-line sources.
7. Show that you can transform raw data into a structured format which is compatible with display, by building queries yourself.

Content

3.0 Introductory Caselet

Background:

Arjun works as a business analyst at GreenLeaf Organics, a business with online e-commerce, retail stores and third-party distributor operations. The company employs various systems to track its data:

• Store sales have been stored in Excel files,

• Online sales are maintained in a SQL Server database and,

• Information of the partner distributor is recorded in a Google Sheet.

Arjun has been asked to create a Summary dashboard which shows aggregated sales performance from all the three channels. But each source is in a different shape, category and update frequency. Before Arjun used to manually combine this in excel taking up few hours/week and increase in risk of errors.

As they scaled the business, the leadership team required error-free reporting in real time. Looking for a way to automate the data integration and alleviate reporting delays, Arjun looked at Power BI.

To connect to each source, he made use of the “Get Data” functionality in Power BI: he imported Excel files, used DirectQuery for SQL Server and accessed Google Sheets via the Web connector. Then he 'extracted and cleansed' the data using Power Query to match up the sets. He would then establish relationships among them in the data model and displayed the synthesized insights via interactive dashboards.

At the end of the week, Arjun had managed to completely automate his data loading. The refreshed dashboard was loaded up to the Power BI Service, where management could view it at any time. The outcome: increased visibility, more informed decision-making and decreased manual grunt work.

Critical Thinking Question:

If you are in Arjun’s shoes, how would you prioritize live connectivity and imported Data models? How do you manage data from different sources? What are the challenges and how to make the process consistent and reliable for regeneration/refreshing?

3.1 Connecting to Various Data Sources

Power BI is a flexible Business Intelligence tool that works from many different sources. Such sources can be local files, relational databases servers, cloud storage providers or even live web data. The power to pull data from multiple systems means users can build robust and powerful reports.

3.1.1 Importing Data from Excel

Excel is still one of the most common data formats in the business world. Excel natively connects to Power BI quite easily.

How to Import Excel Data into Power BI:

Navigate to the Home tab in Power BI Desktop and select “Get Data” > “Excel.”

Click on Browse and select the Excel file (. xlsx or. xls).

The navigator window will appear, displaying worksheets and named ranges.

Where you want to import the sheet or range?

1 Click Load to load the data into Power BI or click Transform Data to open it in Power Query and clean the data.

Features:

• Power BI can import from multiple sheets.

•It will detect table formats and apply type detection to some extent.

• If the Excel source data file updates or changes (at the same location), you can refresh those changes in Power BI.

3.1.2 Importing Data from CSV Files

Introduction CSV (Comma-Separated Values) files are text-based data files that store tabular plain text form of information. These can be created on the basis of databases, web applications or third-party software.

How to Import CSV in Power BI:

Select “Get Data” and then “Text/CSV.”

Choose the. csv file from your machine or cloud.

A preview window will pop out with displayed delimiter type and column headers with data.

Could Load to import or transform to clean the data.

Key Points:

• CSV files are not formatted or formulized (as in Excel).

• Headers are generally in the first line, and delimiters may change (comma, semicolon etc.)

• Power BI will auto-detect the file encoding and delimiter or use it manually.

Application: Import sales logs, exported transactional data or CRM contact lists.

3.1.3 Connecting to SQL Databases

There are many enterprises with millions of records in SQL Server, MySQL, PostgreSQL or Oracle. Power BI can connect to those sources but not with DirectQuery mode.

How to Connect to a SQL Server Database:

Navigate to "Get Data" → "SQL Server."

Key in the server's name and the database name (if applicable).

Select the method of authenticating (windows, database credentials, etc.).

Choose whether to Import or use Direct Query (live connection).

Choose the target tables or directly input a SQL query.

Features:

• Import Mode: Power BI imports data and provides speedy performance and full modelling functionality.

• Direct Query: Accesses data directly from the server; allows real-time analytics but with certain restrictions on functionality and speed.

Tip For large-scale or frequent refresh scenarios, Direct Query is more applicable, and Import mode is better for deep modelling and performance.

3.1.4 Connecting to Web Data Sources and APIs

Power BI also supports accessing data over the web, whether located on public websites, in the cloud or REST APIs. This is particularly helpful for live data such as currency exchange rates, stock prices, the weather or even social media metrics.

Steps to Connect to Web Data:

Select "Get Data" → "Web."

Pass the URL of a data source (like for example a webpage with a table in HTML, a CSV file or an API output in JSON format).

APIs: These may need extra setup (authentication, parameters).

With Power Query, you can extract and clean tables from the returned content.

Examples of Web Data Sources:

• Wikipedia or official government websites. • Tables of Wikipedia and other governmental sites

COVID-19 data in JSON format

• Live financial data from public APIs

Note: API connection might need an understanding about the authentication (API key, oAuth) and data structure (JSON, XML).

Did You Know?

"Did you know that Power BI can connect to a public API and perform an auto-refresh of data from a live web feed (such as currency exchange rates or weather)? You can even pull in HTML tables directly from a website — such as Wikipedia or governmental sites — by putting the URL into the “Web” connector.”

3.2 Direct Query vs Import Mode

Power BI offers two primary data connection modes: Import and DirectQuery. Every mode has the unique features that makes affect on data storage, performance, modelling capacity and real time. Knowing the differences is really important to be able to choose the right mode according to data environment and business needs.

3.2.1 Features of Import Mode

Import Mode is the standard and most typical way to connect in Power BI. It loads data into Power BI local memory storage to allow for fast and rich analysis.

Key Features:

• Data Storage: Data gets imported and saved into the Power BI file (. pbix).

• High Performance: Reports are performant as the data is in-memory and can run with complex logics.

• Full Power BI Support: All Power BI capabilities (DAX, modeling, relationships, calculated columns) is supported.

• Offline Capability: Run reports without internet.

• Automatic Update: Users can schedule automatic updates(daily, hourly) through the Power BI service.

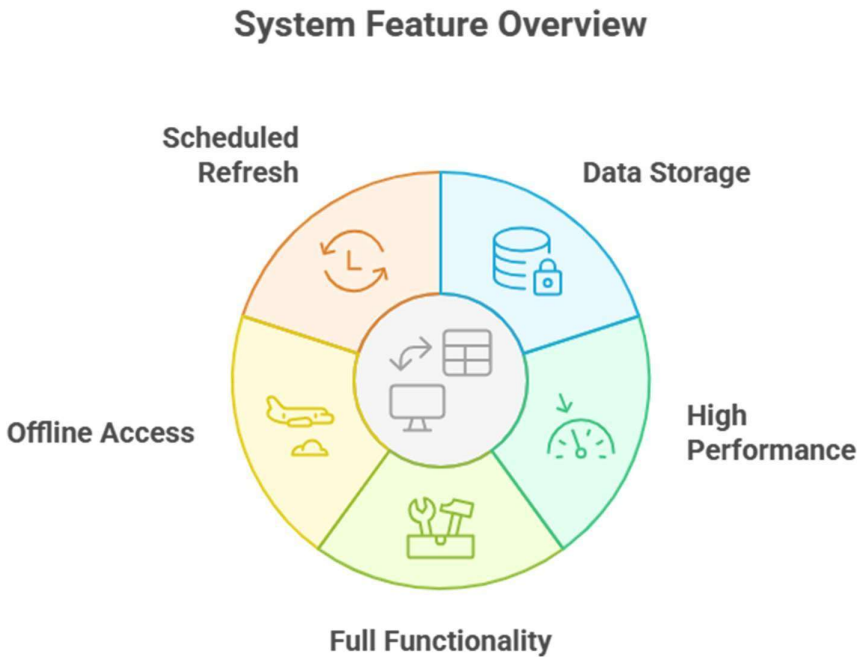


figure: **System Feature Overview**

Use Case: Perfect for when data is updated on a schedule (daily, weekly, etc.), but intermediate calculations or monstrous datasets are required.

3.2.2 Features of Direct Query Mode

Like Kathleen stated, Direct Query Mode is not actually downloading data to Power BI. Rather, it constructs a live connection to the source and makes queries on-the-fly whenever a user interacts with the report.

Key Features:

• Direct Query: Data doesn't import, and Power BI sends queries to the source.

• Real Time Analytics: Represents the most current data in your reports.

• PBIX File that's Lighter: The data is not kept inside the file, so the PBIX is light.

• Data Protection: No sensitive data redundancy; it will still be contained in its original source system.

• Limited Modelling - Some modeming and DAX capabilities such as are limited.

Use Case: Appropriate for very large enterprise databases that update frequently (for example, real-time dashboards in finance or logistics).

3.2.3 Advantages and Limitations of Each Mode

Aspect	Import Mode	DirectQuery Mode
Speed	Faster performance due to in-memory processing	Slower (depends on source system response time)
Real-Time Data	No (only after manual or scheduled refresh)	Yes (live connection to source)
Data Modeling	Full DAX, calculated tables, relationships supported	Limited DAX support; no calculated tables
File Size	Larger (data embedded in .pbix)	Smaller (no stored data)
Security	Requires proper handling of sensitive data	Higher security; data remains in source
Offline Use	Available	Not available (requires active connection)
Complex Queries	Supported and optimized	May result in slower performance or timeouts

3.2.4 Choosing the Right Mode for a Business Scenario

The choice between Import and DirectQuery depends on several factors such as data size, refresh frequency, source system performance, and analytical complexity.

Guidelines for Selection:

- Choose Import Mode when:
  - Performance is a priority.
  - You need advanced calculations, complex DAX, or modeling features.
  - The source system cannot handle frequent queries.
  - Daily or periodic refreshes are sufficient.
- Choose DirectQuery Mode when:
  - You need up-to-date data every time the report is used.
  - The dataset is too large to import into Power BI.
  - Data security policies prevent local storage of sensitive information.
  - You are working with a high-performance database designed for live querying.

Example:

A retail company may use Import Mode to analyze monthly sales trends using historical data, while a stock trading firm may use DirectQuery to visualize live market data updates in real time.

Did You Know?

“Did you know that you can use both Import and Direct Query in the same report using a feature called Composite Models? This hybrid model lets you balance performance and freshness—e.g., keep large, stable tables in Import mode and real-time tables in Direct Query.”

3.3 Handling Multiple Data Sources

Connecting, Merging and Databasing Power of the BI Perhaps one of power well, since we're on this note is “connecting things" combining data: from Excel files to Web APIs to cloud services right through relational databases. But then you start having to deal with the headache that is of structure, performance, consistency and modeling across diverse data sources. This section describes how Power BI copes with multi-source integration in a smart way.

3.3.1 Combining Data from Multiple Sources

Data often exists in a variety of formats and locations. You can import or connect to multiple sources in one report in Power BI and then combine (or merge) them if necessary.

Common Examples:

• Sales data in Excel

• Product listing in a SQL Server,

• Data of a marketing campaign from web-api

Combining Techniques:

• Append Query: Combine like structured tables (i.e., sales from additional branches).

• Merge Queries: Join tables on a common key (for example, join customer data to transactions).

• Staging Queries: Use as much or little intermediate steps for complex transformations.

A big part of that is Power Query, which gives you a place to do your data cleaning and aligning before you merge, join, or append.

3.3.2 Data Modelling Across Different Source Types

Once the data is loaded from various sources, Power BI users need to define relationships between the tables in order to develop the consolidated model.

Key Concepts in Cross-Source Modelling:

• Key Constraints: Primary and Foreign keys that are used to specify Oneto many or anyone relationships between relations.

• Star Schema: Model pattern favoured in Power BI (fact table + dimension tables).

• Auto-Detect Relationships: Power BI can Autodiscover relationships by column names and data types, albeit they might still require some manual adjustments.

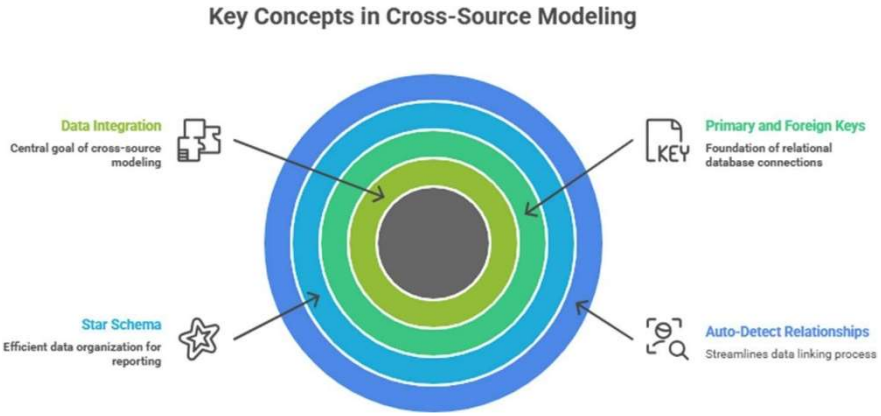


figure: Key Concepts in Cross-Source Modeling

Mixed Source Types Example:

• Merge imported Excel files with SQL Server data using enough DirectQuery tables.

• Generate sales data, extracted from a cloud CRM with customer information in on-premises database.

Important to note not all features in the model are available when you combine Direct Query with Import.

modes (referred to as com- posit models) and constraints may exist.

3.3.3 Resolving Data Conflicts and Inconsistencies

While merging various sources, inconsistencies or discrepancies may arise due to differences in data structure, formats or quality of data.

Typical Issues:

• Column names or data types that don't match up across the two data sources

• Duplicates or missing values

• uniformity of date format (like: DD/MM/YYYY OR MM/DD/YYYY for you pukey seppo's)

• Differences in business terminology (e.g., what a “customer” is – it may be an end user in one system and a reseller in another)

Resolution Strategies:

• The first steps can be applied in Power Query to get your format uniform (i.e. changing datatypes, cleaning nulls, renaming columns).

• Provide lookup/reference tables for mapping of diverse values into some standard form.

• Use calculated columns or DAX measures for an unified logic across all filters.

3.3.4 Performance Considerations with Multiple Sources

If you are connected to more than one data source--especially large or cloud-based ones -- they may affect report refresh time, query load, and performance.

Performance Challenges:

• Longer refresh intervals due to available queries from different sources

• The slow performance of Direct Query with multiple data sources

• Larger file size in Import mode with numerous data tables

Best Practices to Optimize Performance:

• Consider using Import Mode for heavy data if you don’t need to perform updates in real time.

• Exclude unused lines and columns before loading.

• Filter early in Power Query to decrease the amount of data.

• Do not use calculated columns if you don’t have to - use measures.

• Allow Query Folding: Try to push the transformations back to the data source.

Composite model: Try not to overdo the mix between Import and Direct Query, consider the performance and transformation compatibility.

3.4 Basics of Query Creation

The Power Query Editor in Power BI is the spending most of your time tool where you shape, clean and transform data. Creating a query is the act of establishing a connection to a data source and

doing transformations in front of the report. These transformations are represented as a sequence of applied steps and can be modified (reordered or deleted).

3.4.1 Introduction to Power Query Editor

The Power Query Editor is essentially a secondary window that appears when you select to transform your data from the “Get Data” screen. It provides a no-code, point-and-click interface for building queries — or instructions about how to shape your data.

Main Interface Components:

• Query Pane (Left) : A list of all queries (tables) in your report.

• Data Preview (mid): Shows a part of data in the process.

• Applied Steps Pane (on the right): Displays each of your transformations.

• Ribbon Menu: Offers transformation wizards (Split Column, Replace Values etc.).

Functionality:

• Preview data before loading

• Clean and format raw datasets

• Merge, append, and pivot tables

• Transform your code without touching the original source

Behind the scenes, Power Query utilizes the M language but we don’t need to write any code for most of us.

3.4.2 Applying Basic Transformations (Filter, Sort, Rename)

Power BI enables users to apply a variety of simple transformations with an easy-to-use graphical user interface.

Common Basic Transformations:

• Filter Rows

o Keep or delete rows by value/condition

o e.g. only show sales of the year 2011 or do not include empty lines

• Sort Data

o Alphabetically or numerically (ascending/descending)

o Eg, sort products by price or dates by most recent (or earliest)

• Rename Columns

o Makes the columns more user friendly when working with in visuals

o For example, change Column1 to Customer Name

You can see each of these steps listed in the Applied Steps pane and change them at any point. These operations will prep your data for analysis and presentation.

“Exercise: Cleaning and Preparing Sales Data with Power Query”

Instruction to the Student:

You get an ugly Excel with the name of “Raw_SalesData”. xlsx that contains:

• Blank rows

• Inconsistent column names (Such as “CustName” instead of “Customer Name”)

• Outliers in the Quantity Column (Values >1000)

Your task:

Load the Excel document into PowerBI through the “Excel” connector.

In Power Query:

o Remove blank rows

o Give the columns meaningful names

o Remove all the records where Quantity is 1000 above.

o Order the records by Region and then according to the Order Date. (Ascending)

Load your clean data to the model and generate a simple table view that displays:

o Customer Name

o Order Date

o Quantity

o Region

Deliverable:

Please provide your Power BI file and a screenshot of (Applied Steps pane) in Power Query Your Power BI!

3.4.3 Creating Custom Columns

Custom columns are employ to append newly computed fields derived from the underlying information. It allows for more flexibility in analysis without the need to change the source table.

Steps To Make Custom Columns:

Navigate to “Add Column” → “Custom Column”.

Write your logic with formula editor (Power Query uses the reduced M Language).

Example formula:

if [Sales] > 10000 then "High" else "Non-High"

Examples of Use:

• Classifying data (for example Sales Volume - Low, Medium, and High)

• Using date based flags (such as “Weekend” vs “Weekday”)

• Margin calculations (for example, Revenue/Cost).

Custom columns are evaluated prior to the data being consumed into the model, so they can be particularly beneficial for scoping and classifying data as well as transforming it.

3.4.4 Saving and Reusing Queries

The great thing is that you can also save your work and use the same queries for other reports - this let's you work faster, and stay consistent as well.

Key Features:

• Close & Apply: Saves your query and loads the transformed data into the data model.

• Reference Query: Use an existing query as the foundation for a new query (useful when you have multiple filtered views to create).

• Duplicate Query: Duplicates a query for modification, without changing the original one.

• Parameterization: Write custom queries and re-use them by adding parameters (e.g. file path, date filter).

• Extensibility: You can save queries as Power BI templates or reuse them through the Advanced Editor by copying M code.

Use Case:

So instead of recreating these transformation steps (sales data) every month, You will save a query template and just change the source file path or date.

Knowledge Check 1

Choose the correct option:

Which one of the following sources takes a URL and typically parses HTML, JSON or XML?

A) Excel

B) CSV

C) Web

D) SQL Server

What issue in DirectQuery mode cannot be solved in Import mode?

A1) It yields too many modeling degrees of freedom

B) It keeps all data in Power BI itself, without the need for any data saved externally

C) None of the data sources is supported

D) It limits a few of the DAX functions and performance capabilities

What is the Power BI feature to get rid of null,rename column and convert datatype prior to load?

A) Power View

B) Power Query Editor

C) DAX Editor

D) Power BI Service

What is the use of “Append Queri es” in power query?

A) For the extraction of one source data:

B) For matching two tables on a key

C) To join two or more identical tables

D) To generate images in the report view

When you have more than one source in a model, such as Excel, SQL, and JSON together in one modelwhat does lober content reference them together?

A) Power Pivot’s Relationships

B) Power View

C) Web Connector

D) Applied Steps Pane

3.5 Summary

ϖ In this chapter, you learned the basics of working with data in Power BI. It started by covering how to connect to different data sources such as Excel, CSV files, SQL databases and web APIs. Learners looked at the two fundamental types of connections: Import, and DirectQuery and their performance, modeling and refresh capabilities.

ϖ The chapter has also concentrated on the treatment of multiple sources of data, where it has been showed how one can merge, model and clean diverse datasets into a coherent background while dealing with inconsistencies and performance constraints. Finally, it introduced the

Power Query Editor to shape and transform data, using an easy drag-and-drop interface _ no coding required. Pupils acquired the ability to filter, sort and rename columns and create custom calculated measures as part of building up an understanding for how a template can be structured within Power BI for reuse against scaled reporting.

3.6 Key Terms

Import Mode - A mode of connection in which data is imported into the Power BI for faster, in-memory analysis.

DirectQuery- It is a live connection method using which Power BI queries the source (database tables) directly at run time.

Power Query Editor – The place in Power BI that can be used to shape, transform and clean data before putting it under the analytical magnifying glass.

Merge Queries – A Power Query transformation which joins two tables together using a common key.

Append Query - Appends rows to a table, row by row, based on the results of a query.

Derived Column - A custom which is created using expression language to convert or to classify the data.

Composite Model Import and DirectQuery sources combined in one report.

Applied Steps - A record of the transformations that have been applied to data in the Power Query Editor.

Query Folding - When the transformations are pushed back into the source system for efficient processing.

Reference Query - A query that's built from the steps and rationale of an existing query for reuse or variations.

3.7 Descriptive Questions

Explain how Power BI connects with different data sources. Provide one instance of a file-based, cloud and database source.

What is the differnece between Import mode and DirectQuery mode? Which one would you recommend for a real-time stock market dashboard and why?

Describe what aggregating data from several sources looks like within Power BI. What are some common obstacles to doing so?

How does Power Query Editor assist you in preparing your data? Write down and explain three of the basic transformations.

What are custom columns in Power BI? Give an example of a situation, where custom column could be handy?

What can be done to enhance performance while using multiple big data sources in Power BI?

Describe what you would do if you had to extract data from a variety of different sources.

How important is it to save queries in Power BI?

What is Query Folding and how does it help me in Power BI?

What is the back-and-forth about whether you should mix Import and DirectQuery sources in a Power BI model?

3.8 References

1. Microsoft Power BI Documentation. (n.d.). Retrieved from https://learn.microsoft.com/en-us/power-bi

2. Ferrari, A., & Russo, M. (2021). The Definitive Guide to DAX. Microsoft Press.

3. Chhabra, S. (2022). Power BI for Beginners. BPB Publications.

4. Power BI Community Blog. (n.d.). Retrieved from https://community.powerbi.com

5. Radacad. (n.d.). Articles on Power Query and Data Modeling. Retrieved from https://radacad.com/blog

Answers to Knowledge Check

Knowledge Check 1

1. C) Web
2. D) It restricts some DAX functions and performance features
3. B) Power Query Editor
4. C) To combine rows from two or more similar tables
5. A) Power Pivot’s Relationships

IBI_Unit 4_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127445824

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:00 PM GMT+5:30

File Name

IBI_Unit 4_V3.docx

File Size

300.0 KB

19 Pages

3,827 Words

21,251 Characters

0% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

1 Not Cited or Quoted 0%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **1** Not Cited or Quoted 0%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Publication

"ICT Analysis and Applications", Springer Science and Business Media LLC, 2025 **<1%**

Unit 4: Data Cleaning & Transformation

Learning Objectives

1. Explain the role and interface of the Power Query Editor in the Power BI data preparation process (4.1).
2. Identify and apply techniques to remove duplicate records, handle missing or null values, and ensure data quality and integrity (4.2).
3. Use data transformation tools such as Split Column, Merge Column, and Change Data Type effectively to prepare raw data for modeling (4.3).
4. Create custom columns using expressions, and develop a foundational understanding of M Language syntax and functions to support advanced transformations (4.4).
5. Utilize the Applied Steps pane to track and manage transformations, enhancing repeatability and reducing errors in data preparation workflows.
6. Develop structured, clean, and analysis-ready datasets using no-code and low-code tools in Power Query.
7. Improve analytical readiness of datasets by performing data wrangling operations aligned with specific business questions and reporting needs.

Content

4.9 Case Study

4.0 Introductory Caselet

" S neha's Data Cleaning Challenge: Preparing Sales data with Power Query"

Background:

Sneha works as a data analyst at UrbanKart, an e-commerce company in various Indian cities. You're assigned to analyze the monthly sales report coming from different branches, ie Regional Offices. But data comes from several Excels in dissimilar formats. Some sheets are repeating rows, some have empty columns, and many include dates and numbers that don't line up.

At first, Sneha attempted to clean the data manually using Excel, but it was arduous and repetitive. She required a faster, more consistent method of cleaning, transforming and tabulating data for reporting.

That's when she discovered Power Query Editor in Power BI.

Sneha utilized Power Query to pull in all Excel files, deleted duplicated data, replaced with some missing details and normalized the column names and data types too. She performed a number of actions, each of them was captured in the Applied Steps pane. And the best part: She could follow those same steps to new files every month, without starting over from scratch.

So, with her cleaned up data combination now in Power BI, Sneha created an interactive sales dashboard that would always stay current whenever new data was brought in.

Critical Thinking Question:

Where is Power Query over Excel? What is the role of the Applied Steps pane in facilitating consistent and efficient recurring data tasks?

1

4.1 Introduction to Power Query Editor

Power Query Editor is embedded data transformation tool within Power BI. It enables users to access multiple data sources, and then cleanse, shape and enhance the data prior to loading it into the Data Model.

Purpose of Power Query Editor

Data requires cleaning before it can be analyzed. Raw data may include:

- Duplicates
- Inconsistent formatting

• Missing values

• Extra rows or columns

• Mismatched data types

Power Query Editor offers a code-free environment for such transformations, allowing you to carry out the actions you take in Power Query as steps in a process that can be reused or refined later.

Key Components of the Interface

• Query Pane (Left) – Lists all of the loaded queries (tables)section (saved to file or dataset).

• Data Preview (Center): Shows a sample of the data you are working with

• Ribbon Toolbar (Upper): Tools like remove columns, filter rows, merge queries and so on.

• Applied Steps Panel (Right): Records each change in a sequence — just as you would follow a recipe-turned-dynamically.

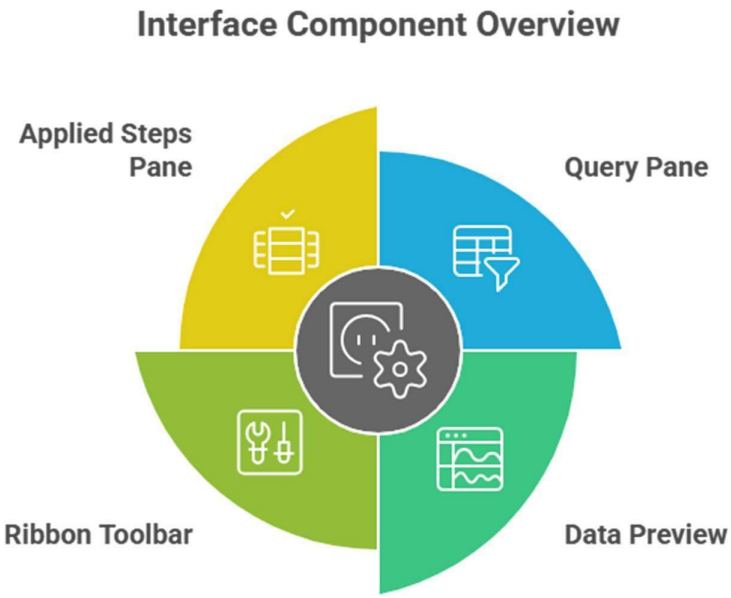


figure: **Interface Component Overview**

Functionality Highlights

• Connect to Excel, CSV, SQL and Web Data sources

• Remove duplicates or blank rows

• Convert data types (text to number or date)

• Filter and sort data

• Split or merge columns

• Add custom columns using expressions

• Combine multiple files or tables

All transformations done for Power Query are non destructive, therefore the source file will be left intact. Power Query doesn't replace your data with new data, but rather a custom view of the data tailored to fit your needs.

Why It Matters

Using Power Query:

• Saves some time for repetitive data cleaning tasks

• Improves data quality before reporting

• Makes your process open and auditable

• Automatic refresh of cleared data when the source is updated

It's particularly invaluable for those working with data from various sources, regular reporting routines or.regex.studio

messy datasets that require standardization.

4.1.1 Overview of Power Query Interface

In Power BI Desktop, when you click "Transform Data", the Power Query Editor window appears. Its user interface enables users to apply transformations without coding, but with the ability to use advanced scripting with the M language.

Main Sections of the Interface:

Interface Area	Description
Ribbon Toolbar (Top)	Contains commands for transformations (e.g., remove rows, split columns, group data).
Queries Pane (Left)	Lists all active queries (tables) in the report.
Data Preview (Center)	Displays a snapshot of the current query's data after each transformation step.
Applied Steps Pane (Right)	Shows a step-by-step record of every transformation applied to the query.
Formula Bar (Optional)	Displays the M code generated by each transformation (can be enabled via the View tab).

This sane structure ensures that users do not get lost in the transformation pipeline and to provide them with as much control over the manipulations applied on their data.

4.1.2 Loading Data into Power Query

The load process of the data into Power Query starts from the “Get Data” menu in Power BI Desktop.

Steps to Load Data:

Go to “Home → Get Data” and choose a source (like Excel, CSV, SQL Server).

A Navigator window will open with available tables/sheets.

Click on “Transform Data” to open the power query editor and load the selected table(s).

Begin applying transformations as needed. After loading data into Power Query:

• It is temporarily retained for conversion (since it’s not in the data model).

• Nothing written to the source file.

• You then click “Close & Apply” once you’ve shaped the data to load it into Power BI for analysis.

4.1.3 Applied Steps and Query Settings

The Applied Steps pane is one of the most beneficial tools in Power Query. Every transformation you make is kept as a step, so data processing becomes transparent and can be audited or reversed at any time.

Key Features:

• Each step (e.g., renaming a column, filtering rows) is listed chronologically.

• You can tap on any step to see your data while in that step.

• Steps can be edited, rearranged or deleted by gear or "X" icon.

• The Query Settings panel (top-right):

o the name of the query

o All applied steps

o the data source

This configuration allows for reusability since you can repeat the process in another file or if a dataset is updated and you don't have to complete manually inputs.

Did You Know?

“Did you know that every step you take in Power Query is automatically saved as M code under the hood? You can edit, or copy and paste this code from the Advanced Editor to make all of your transformation logic totally transparent, reusable (which is something Excel isn't so hot at)!”

4.1.4 Best Practices for Data Preparation

Using Power Query well is not just about using transformations, it’s also about following a methodology that ensures results are clean and consistent so it’s easy to report from.

Best Practices:

Remove Unused Columns Early

o Can reduce the size of data and enhances model performance.

Name Queries Clearly

o Use meaningful names like (Customer_Sales_2023) rather than names like (Table1).

Document Complex Steps

o Use comments in the M code or a step log (the message box that comes up on screen) for more complex jobs.

Use Data Types Consistently

o Apply appropriate types (Date, Decimal Number...) to prevent problems in visuals/ DAX

Minimize Applied Steps

o Group transformations into related ones and minimise redoing work.

Test with Sample Data

o Develop your queries against a subset of the data to ease testing.

Use Reference Queries

o If you are using the same data for branched logic, use reference queries so as not to repeat it.

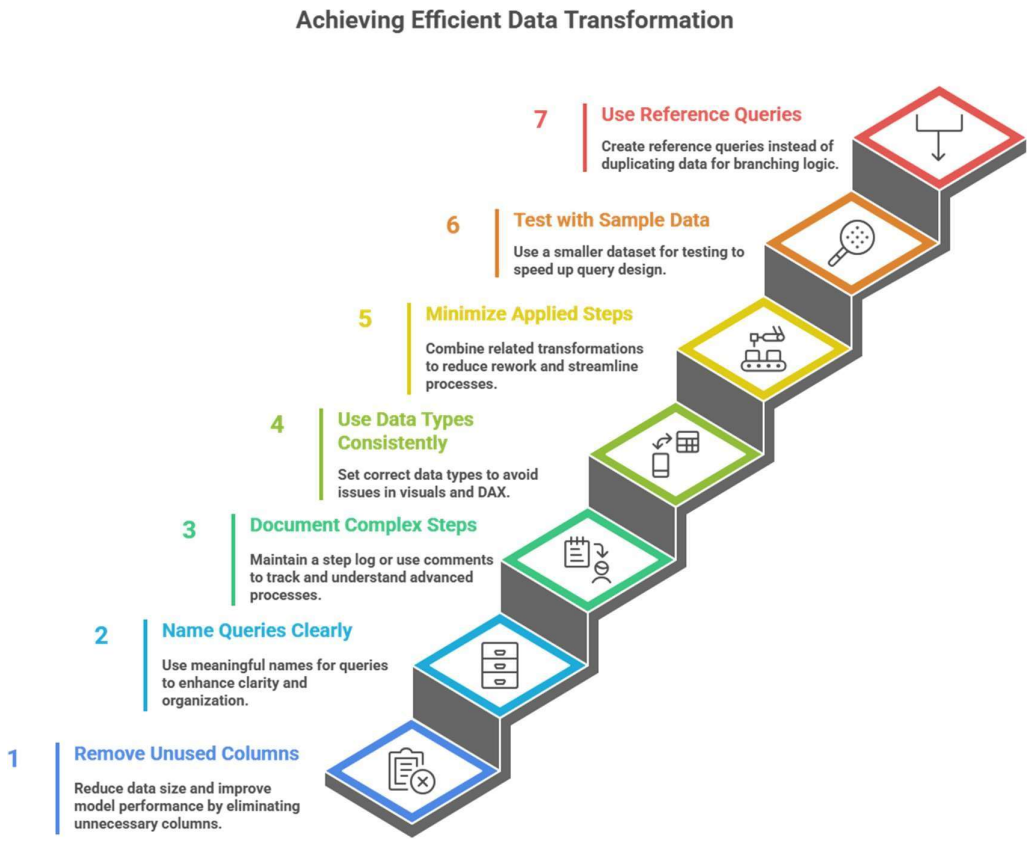


Figure: Achieving Efficient Data Transformation

These practices ensure that your data preparation is scalable, maintainable, and aligned with business reporting standards.

4.2.1 De-duplication and Missing Data De-duplication was done in the following way: we first removed all erroneous data (i.e., tagged as “Unknown”) from the train sets, and then dropped remained records with repeated values (the sum of records removed at this step was 506/1893 for task 1/task 2 based on its sorting order), resulting in two roughly equal halves.

One of the most fundamental things to be done while working with dirty data is handling duplicate values and missing value as this can impact accuracy of insights, analysis, etc. In Power BI, Power Query functionality to identify and remove (or fix) these data quality issues is a snap.

4.2.1 Identifying and Removing Duplicate Records

Duplicate records are created when we repeat a single row or parts of it. Duplicates occur as a result of erroneous data entry, system exports, or combination of datasets.

How to Remove Duplicates in Power Query:

Choose any single or multiple column(s) which define the uniqueness (e.g., Order ID).

From Home > Remove Rows > Remove Duplicates.

Power Query keeps the first and deletes all others.

Every remove is recorded as a step in the Applied Steps side pane.

Tip: Use “Keep Duplicates” to find duplicates without removal, for audit purposes.

Example: If using a system sync, deal with duplicate customers (i.e. the same ID appears more than once).

4.2.2 Handling Null and Missing Values

In Power BI, None is equivalent to Null which is not a value. These may appear due to:

• Incomplete data entry

• Errors during import

• Mismatches during merges or joins

Common Methods to Handle Nulls:

• Remove Null Rows:

Home → Remove Rows → Blank Rows 3.

• Replace Nulls with Defaults:

Transform → Replace Values → Replace “null” with some value (like “Unknown” or 0)

• Filter Out Nulls:

Apply filters to remove the rows with NULL values in key fields

• Conditional Replacement:

Use “Add Column” with if [Column] = null then "Default" else [Column]

Best Practice: Do not replace missing values where you don’t know why they’re missing (or even if–you want to be sure). If in doubt, it’s OK to flag or isolate them instead.

“Activity: Clean Up and Enter Missing Customer Data”

Instruction to the Student:

You have a dataset with the name Customer_Orders. csv, with customer names, regions, order amounts and review scores. Some entries have:

• Missing customer regions

• No feedback ratings Your job:

Using Power Query, load the file into Power BI.

Replace missing regions with “Unspecified”.

We will fill the missing ratings with average_ratings adjusted to one decimal.

Include new column of customer categories as:

o "Loyal" if rating ≥ 4

o "Average" is rating in [2, 3.9]

o “At Risk” if rating < 2 or is null

Deliverable:

Submit your. pbix file, a brief reflection (100-150 words) on how applying the steps improved data quality and facilitated more insightful analysis.

4.2.3 Data Imputation Techniques

When data are incomplete, we often need to impute (or estimate or complete) the missing items using either logical or statistical techniques. Simple imputation is possible with Power Query in Power BI.

Common Imputation Methods:

Technique	Description	Example
Default Value	Replace null with “N/A” or 0	For missing ratings, use “Not Rated”
Fill Down/Up	Use values from adjacent rows	Fill missing Region values down
Mean/Median Substitution	Use average of a column to replace nulls (requires custom column logic)	Replace null salary with average salary
Category-Based Imputation	Fill based on a related group	Replace missing age by average age in the same department

In Power Query, some of these can be implemented using:

- Fill → Down/Up
- Group By + Aggregate
- Custom Column with conditional logic

Did You Know?

“Truth: You can do grouping and conditional fill in PQ. (vectorized) calculations and assign e.g. average salary per department (and those skills), then fill in remaining

missing values, all of this without typing a single formula – just take advantage of the "Group By" feature and Merge Queries."

4.2.4 Ensuring Data Consistency

Data consistency ensures that alike values are presented and processed so as to be consistent. Even if one does deduplicate and address nulls, there still may be inconsistencies in the data such as typos, mixed cases or variable formatting.

Methods to Make Your Power Query More Consistent:

• Trim and Clean Text

Transform → Format → Trim (extra spaces), Clean (non-printable characters)

• Change Case

Use To Lower, Upper or Capitalize For Each Word to standarize the insert of text.

• Replace Values

Correct typos, etc. (e.g., replace "Banglore" by "Bangalore").

• Normalize Date/Number Formats

Assign correct data types; change text dates to an actual Date/Time.

• Create Reference Tables

For columns like region, department or category have a controlled look up table to standardise things.

Example: In a "State" column, with either "TN", "Tamilnadu" and joined by "(" without spaces, replaced with consistent "T.N".

4.3 Splitting, Merging, and Changing Data Types

You can use Power Query to clean and fine-tune your data by reshaping it, combining it with other data sources, and removing unwanted columns or rows. These operations are necessary for constructing clean, Table 7: Summary statistics of the GreenBook corpus analysis-ready corpora.

4.3.1 Splitting Columns by Delimiter and Number of Characters

A column split comes in handy when you have one field with two values, such as a first and last name or a city and state from an address.

Splitting by Delimiter:

• Use when you use characters like commas, spaces and slashes to delimit the data.

• To Perform: Transform → Split Column → By Delimiter

• Select the delimiter: (e.g., comma, space, anything you want)

• Specify how to split:

o At each occurrence

o At first occurrence or last only

Example:

“John Doe” → Split on white space → “John” and “Doe”

Splitting by Number of Characters:

• Use when all pieces have fixed length.

• Go to: Transform → Split Column → By Number of Characters

• Show character count (e.g., 4 for year in a date string)

Example:

→ Break into “2023” and “0415”, etc.

4.3.2 Merging Queries and Columns

It consolidates multiple columns into one—such as for a full address or full name.

• Use: Transform → Merge Columns

• Choose columns (First Name, Last Name etc.)

• Select separator (space, comma, custom)

• Converted: 1 column, con-cat code values

Example:

“Ravi Kumar” = “Ravi” + “Kumar” Sign and Symbol We use the following sign to denote concatenation.

Merging Queries:

This is to merge tables (queries) where you join them on a key field(like in SQL joins).

• Use: Home → Merge Queries

• Select two tables and a correlated column.

• Select the join type:

o Left, Right, Inner, Full Outer, Anti and so on.

• Unfold the combined table to show necessary columns

Use Case Example:

Joining “Orders” to “Customer Details” based on Customer ID.

“Activity: Join and Cleanse Product Sales Data”

Instruction to the Student:

You are given two files:

• Sales_Q1. xlsx consisting of Product ID, Quantity and Sales

• Product_Info. csv with columns Product ID, Product Name, and Category Task You should:

To do this, we will load both files in Power BI with Power Query.

Combine the two queries by Product ID (Left Join).

Unpivot the merged table for Product Name and Category.

Add an “Average Selling Price” column = Revenue / Quantity.

Get rid of unwanted columns and give the rest clear names.

Deliverable:

Submit your. pbix file with neat final table and a table visual of Product Name, Category, Revenue, Quantity and Average Selling Price.

4.3.3 Changing and Detecting Data Types

Correct data type in Power BI is so important for proper calculation, visualization and filtering.

Power BI Common Data Types:

• Text

• Whole Number

• Decimal Number

• Date/Time

• True/False (Boolean)

• Currency

Steps to Change Data Type:

• Change to: Transform → Data Type dropdown

• Choose the type that is right for the column

Auto Detection:

•Data type auto detection when loading to Power BI.

• You can turn this off or modify types yourself accordingly if you wish.

Tip: Always check the auto-detected type, especially for dates and for numbers that have been exported as text.

Example:

Change a column from Text to Date and use in a KPI time-based chart or slicer.

4.3.4 Transforming Text, Numbers, and Dates

Power Query provides a few built-in transformations to help cleanup and standardize text, numeric, and date/time data.

Text Transformations:

• Trim: Removes extra spaces

• Clean: Removes non-printable characters

• Upper/Lower/Capitalize: Adjusts text case

• Length: Measures the characters in a string

• View: Get few characters from a string (e.g., first 3letters)

Number Transformations:

• Round Up/Down: Adjust decimal places

• Absolute Value: Removes negative sign

• Normalize: Divide by standard deviation (for statistical purposes)

• Add/Subtract/Multiply Columns Date Transformations:

• Transform components such as Year, Month, Day, Weekday

• Add or subtract days

• Find the instant age based on a date or time period between two dates

Example Use Cases:

• Remove “15 April” and change to the year only: ”2023”.

• A new column “Total Sales” is calculated by multiplying quantity and price

• Standard report: Trim and title case product names

4.4 Creating Custom Columns and M Language

Power BI Power Query gets you a step further than typical built-in transformations by letting you use custom columns. These columns can be calculations, IF logic and text transforms that aligns with business rules. You can define these custom transformations in Power Query by using the M Language, a similar data manipulation language.

4.4.1 Introduction to Custom Columns

A custom column is simply a new field that's created by performing operations with existing data in Power Query. This gives users the possibility to enrich the dataset with derived of categorized information.

How to add a new Custom Column:

Navigate to Add Column –> Custom Column.

Write the transformation expression in a formula builder.

Give a name to the new column.

OK to accept the changes and create the column.

Example:

[Sales] * [Quantity] → New column "Total Revenue".

Applications:

• Creating calculated metrics

• Categorizing values

• Formatting or combining fields

4.4.2 Conditional and Calculated Columns

As the name suggests, conditional columns in Power BI allows users to add logic-driven calculations and generate new values based on conditions.

Creating a Conditional Column:

Navigate to Add Column > Conditional Column.

Filter based on column value.

Specify results for each condition.

Example:

Score] >= 90 then "Excellent" elseif [Score] >= 75 then "Good" else "Needs Improvement"

This is important for grading students, bucketing ranges and enforcing business rules.

Calculated columns [in Power Query] can include math, formatted text (such as dates), and logic, like so:

- [Price] *1.18 → 18% tax will be added
- Text. Length([ProductName]) → count characters

Power BI precalculated these columns prior to loading data into the data model.

4.4.3 Basics of M Language Syntax

The M Language (Mashup) is the language behind every transformation step in Power Query. M code is automatically written when you apply changes doing something in Power BI, but you can also manually write or edit the M code.

M Language Characteristics:

- Case-sensitive
- A line of code for every step
- And leave the last output in the in statement
- Comments use // or /* */

Basic Syntax Structure:

let

Source = Excel. Workbook(File. Contents("Sales. xlsx")), FilteredRows = Table. SelectRows(Source, each [Region] = "South"),

RenamedColumns = Table. RenameColumns(FilteredRows, {{"OldName", "NewName"}})

in

RenamedColumns This example:

- Loads a workbook
- Filters it for the “South” region
- Renames a column

Each step has an ID and records a transformation.

Did You Know?

M Language is a functional language that is sensitive to case and it processes queries gradually, needing each query to be terminated by an in statement. “What this last in-

part after is wrong, its wasted time when you dont know how to order your steps and forget an in can ruin all your query."

Knowledge Check 1

Choose the correct option:

What do the "Applied Steps" in Power Query represent?

A) To view DAX measures

B) So that we can log every change to our data

C) To display visual charts

D) To show data model relationships

Which Power Query function will remove duplicates based on the selected column?

A) Remove Errors

B) Remove Nulls

C) Remove Duplicates

D) Filter Rows

What is the name of the feature in Power Query that lets you split a column by space or comma, etc.?

A) Merge Column

B) Replace Value

C) Group By

D) Split Column by Delimiter

What is the syntax to generate a derived column as a product of two columns say: 'Quantity' and 'Price'?

A) [Quantity] + [Price]

B) Quantity * Price

C) [Quantity] * [Price]

D) Sum(Quantity * Price)

Which of the following statement(s) is/are true regarding M Language in Power Query?

A) It is chart case-insensitive only

B) It is to be used for dashboard writing

C) It is sensitive to cases, and it closes with a in expression.

D) It is the same as SQL

4.5 Summary

In this chapter, you learned about the Power Query Editor—a must-have utility in Power BI for targeting data preparation tasks in raw data. The chapter looked into loading the data from disparate sources, performing important transformations such as eliminating the duplicates, handling the nulls and to make sure that there is consistency in your data.

ϖ Tools such as splitting and combining columns were provided to emphasize the significance of data structure, and to confirm presence of appropriate data types for analysis. Students also discovered how to transform data by adding new columns, both using predefined logic and with M Language in the Advanced Editor.

ϖ These skills combined aid in building clean, organized and analysis-ready datasets on which we can build powerful reports in Power BI.

4.6 Key Terms

Power Query Editor: This is the user interface within Power BI for cleaning and transforming data before it gets into the model.

Applied Steps - Ordered list of transformations on a query; viewable in the Power Query Editor.

Remove Duplicates Here is an awesome feature by which you can remove the repetitive rows with same value in a column or more.

Missing Values: Missing or no data records which must be dealt with or re-recorded to obtain valid results.

Imputation - Filling of blanks using predicted or default values.

Split Column - A split of the content of a column to one or more other columns.

Merge Queries - A feature for joining together sets of data from separate tables.

Data Types – Sorting data in types such as text, number, date/time etc to make it easier for analysis.

Custom columns – A new column derived from expressions/logic applied to existing data.

M Language - The functional programming language that operates behind the scenes in Power Query for transforming data.

4.7 Descriptive Questions

What is Power Query Editor in Power BI and why is it used during data preparation?

Describe how you would remove duplicates rows in a dataset with Power Query.

What do you do when the data in Power Query does not exist? Provide examples of imputation techniques.

Explain how a field can be separated by delimiter or length. Give an example of each.

Differentiate between columns merge and queries merge. When would you use one vs the other?

Why It Is Important To Detect And Correct Data Types In Power BI

What is conditional column in Power Query? How are they created?

Demonstrate the syntax of M language using a simple text transformation.

How does Advanced Editor assist in modifying Query Logic in Power BI?

What are the top 3 recommended practices when preparing your data using Power Query Editor?

4.8 References

1. Microsoft Documentation. (n.d.). Power Query Overview. Retrieved from: https://learn.microsoft.com/en- us/power-query

2. Ferrari, A., & Russo, M. (2020). The Definitive Guide to Power Query. SQLBI.

3. Chhabra, S. (2022). Power BI for Beginners. BPB Publications.

4. Power BI Community. (n.d.). Tips & Solutions. https://community.powerbi.com

5. Gil Raviv. (2018). Collect, Combine, and Transform Data Using Power Query in Excel and Power BI. Microsoft Press.

Answers to Knowledge Check

Knowledge Check 1

1. B) To track each transformation made to the data

2. C) Remove Duplicates

3. D) Split Column by Delimiter

4. C) [Quantity] * [Price]

5. C) It is case-sensitive and ends with an in statement

IBI_Unit 5_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127445826

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:00 PM GMT+5:30

File Name

IBI_Unit 5_V3.docx

File Size

230.7 KB

26 Pages

5,944 Words

35,016 Characters

1% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

4 Not Cited or Quoted 1%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **4** Not Cited or Quoted 1%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Submitted works
Liverpool John Moores University on 2025-12-02 <1%

2 Submitted works
Washington University of Science and Technology on 2026-02-02 <1%

3 Internet
www.coursehero.com <1%

4 Submitted works
Manipal University Jaipur Online on 2025-02-15 <1%

Unit 5: Data Modeling Concepts

Learning Objectives

1. Define the structure and functions of the money market, distinguishing it from capital markets.

2. Identify and describe the characteristics, participants, and instruments of the Indian money market.

3. Explain the features, maturity periods, and issuance process of Treasury Bills (T-Bills) and

Commercial Papers (CP).

4. Compare different short-term money market instruments such as Commercial Bills, Certificates of Deposit (CDs), and Call/Notice Money, focusing on liquidity, risk, and yield.

5. Illustrate how Collateralised Borrowing and Lending Obligations (CBLO) function in secured interbank lending, including the role of collateral.

6. Evaluate the suitability of different money market instruments for banks, corporates, and government entities in managing short-term funding requirements.

7. Apply knowledge of money market operations to interpret market trends and assist in short-term investment or borrowing decisions.

Content

5.7 Descriptive Questions

5.8 References

5.9 Case Study

5.0 Introductory Caselet

Background:

Ananya works as a business analyst at TechNova Solutions, working on implementing an executive dashboard that monitors sales, products and customer data. She is given 3 distinct data files from various departments:

• A Sales table with fields including Order ID, Product ID, Customer ID, Date and Revenue.

• A Customer Master table that contains the elements: Customer ID, Name, Region and Segment.

• A Product Master table with the fields Product ID, Product Name and Category.

When Ananya imports the data into Power BI, she generates visuals that display the total revenue by region and category. But the numbers don't add up and there are whole chunks of territory that somehow have been omitted. Upon inquiry, she discovers that she has not linked the tables together.

To address this, Ananya follows these steps in Power BI's data model view to:

• Establish one-to-many relationships from the Sales (fact) to each of the two dimension tables (Customer and Product).

• You should use the correct natural keys (Customer ID and Product ID).

• Ensure the correct cardinality and cross-filter direction are set up, otherwise filter on other report pages will not filter properly.

Ananya creates the relationships and updates the view, and everything is right where it should be. Her dashboard is now up-to-date, quick and easy to share.

Critical Thinking Question:

Why does proper relationship definition between tables in PowerBI matter? What dangers can come up with bad or non-existant relationships in a data model?

5.1 Tables, Keys, and Relationships

Aggregate your data at multiple granularities in Power BI using relationships between tables. A sound grasp of keys (primary and foreign) and their relationships is key to building dashboards that are meaningful/trustworthy.

5.1.1 Understanding Tables in Power BI

A table is a collection of related data in Power BI, that consists of rows and columns. You can import tables from multiple formats including Excel, SQL and online services.

Two main types of tables:

Table Type	Description	Example
Fact Table	Contains measurable, quantitative data (often transactional)	Sales, Orders, Revenue
Dimension Table	Contains descriptive attributes that provide context to facts	Customer Info, Products, Locations

In Power BI, tables often have a "star schema", where a single fact table is connected to several dimension tables.

Example: A "Sales" table stores the transactions, a "Product" table maps product names and categories. By linking them, you can display total sales by product category.

5.1.2 Primary Keys and Foreign Keys

When working with data from different tables in Power BI, it's important to understand how the Primary Key and Foreign Key in each table relate to provide a cohesive dataset. These keys are the basis for relations in a data model so that users can see relationships between tables.

3 2 A Primary Key is a column or combination of columns that uniquely identifies each row in the table. This guarantees that "ettbuv" is unique throughout the table, maintaining data integrity. A Foreign Key is, on the other hand, a field in one table that references to the Primary Key in another table. This reference joins the two tables so data from either can be used interchangeably.

You can think about a sample data set of two tables: Customers and Orders. This is a common pattern in businesses that have customers who repeatedly place orders. Here's how keys work, with primary and foreign keys in this situation:

• Customers Table (Dimension Table):

o Preserves information, including names, location/addresses and contact numbers of each individual customer.

o With a column Customer for each customer.

o Primary Key: Customer — Each row of this table has a distinct value in this column.

• Orders Table (Fact Table):

o Includes transactional information on every order, such as a date, product, quantity and customer related to the order.

o It also has a column Customer to hold the customer id that make the order.

o Foreign Key: Customer – This is the key in the Customers table and may be duplicated within this table to reflect orders by the same person.

This is the relationship that would enable you to answer questions like “how many orders do I have for each customer?” or “Who did not order anything this month?” by combining the information in both tables by linking the CustomerID.

Power BI tries to automatically detect and propose relationships during the import of data. It is heuristic including column name matching and data type. Still, it is crucial to manually check such recommendations since:

Keys are used correctly for relationships.

• The cardinality of the relationship (one-to-one, one-to-many) is properly determined.

• The relationship does match your analytical needs.

You can have bad/missing relationships that result in bad data models, which will then lead to false analytical conclusions. Thus, learning and getting the correct the primary foreign key setup is crucial when it comes to creating Power BI reports that will stand the test of time.

5.1.3 Types of Relationships (One-to-One, One-to-Many, Many-to-Many)

As we connect tables together, Power BI creates relationships based on cardinality (how records in one table correspond to and interact with those in another).

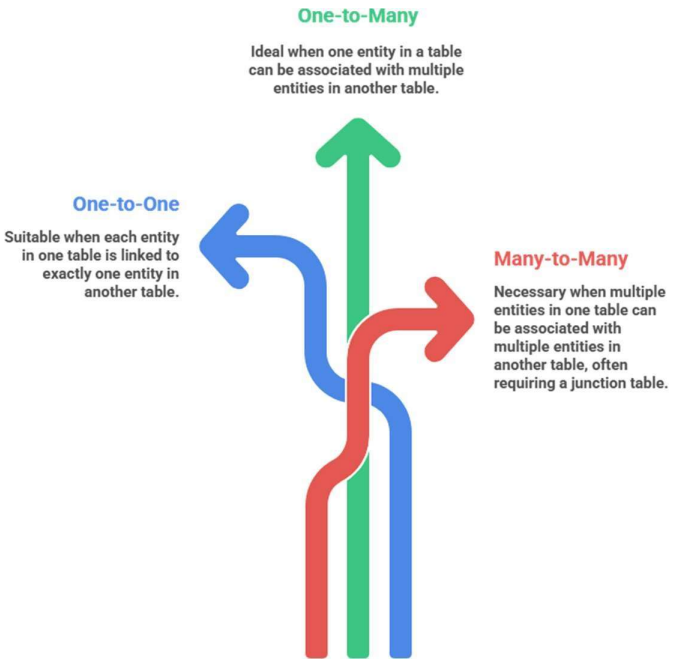


Figure: Types of Relationships (One-to-One, One-to-Many, Many-to-Many

4

One-to-One (1:1)

• Row in Table A directly corresponds to a row in Table B.

• Rare master to master table relationships only.

One-to-Many (1: or :1)

• The most common type.

• Several rows in a fact table can be related to one row in a dimension table.

Example: One item → multiple sales.

Many-to-Many (:)

• Applied to all cases when in both tables there are repeated values of the same property.

• Power BI does this with composite models or bridge tables.

Example: Multiple students in one course, one student in multiple courses.

For many-to-many relationships, performance could be affected and certain filters / DAX logic should be used.

Did You Know?

You're not one of those who didn't know that Power BI doesn't do referential integrity by default are you? This means you can have a relationship even if the foreign key does not

exist in the primary key table. This also has the potential to return null values in a visual so it's always nice to validate your data before this occurs."

5.1.4 Creating and Managing Relationships in Power BI Model View

Power BI has the Model View and allows you to establish and maintain connections between tables.

Steps to Create a Relationship:

Select Open Model View (diagram icon on left navigation menu).

Drag a field (such as Productid) from one table to a same-named field in another.

In Power BI there will be a line drawn between them — that's the relationship.

Relationship Properties:

• Cardinality: Set to One-to-Many, etc.

• Cross-filter direction:

o Single: Filter flows down from one table to another (best practice for most models).

o Both: Toggles between the possibility of filters flowing in both ways – can be used sparingly in complex models.

Managing Relationships:

• Edit relationships by double clicking the line representing the relationship.

• Disable or delete incorrect relationships.

• Utilize Manage Relationships (Home -> Manage Relationships) for tabular display and editing.

Best Practice: Relationships should be clear and simple. Use one-to - many whenever you can; it will give you the proper filtering flow and performance.

"Activity: Create and Test Relationships Between Tables"

Instruction to the Student:

In this dataset, you find three tables such as Sales, Products and Customers.

Import the dataset into Power BI.

Navigate to Model View.

Create relationships:

o Connect Sales[ProductID] to Products[ProductID]

o Connect Sales[CustomerID] to Customers[CustomerID]

Set the right cardinality with cross filter direction(new request) either one-to-many or nothing (don't cross-filter).

Create a report page that displays a table of Total Revenue by Customer Segment and Product Category.

Test filtering of these fields with a slicer.

Deliverable:

Submit a. pbix with model view snapshot and the report page which successfully interacting with filter.

5.2 Data Modeling Schemas

Data modeling schemas in Power BI (and other business intelligence tools) are used to standardize the structure of tables within a data model. These schemas are critical components of how data is exchanged between facts and dimensions, affecting query performance, ease of analysis, and maintainability.

5.2.1 Introduction to Data Schemas

A data schema is a conceptual model that describes the structure and how the data elements are related to each other in one or more tables. Schemas are essential for:

• Organizing large volumes of data

• Maintaining data consistency and referential constraints

• Enabling efficient querying and filtering

• Enabling support for scalable and reusable reports

Schema design in Power BI refers to how fact tables and dimension table are related and how they are organized. There are two popular schemas in Power BI:

• Star Schema

• Snowflake Schema

5.2.2 Star Schema – Design and Applications

There is the good old Star Schema - probably the most used data-modelling model in Power BI – you have a central fact table and surrounding it are dimension tables forming a star. This pattern also suits analytical workloads as we limit the relationships, increasing performance.

In a Star Schema, the dimension table is directly attached to the fact table via a one-to-many relationship (the many side resides in the fact table). This is a handy arrangement for slicing, filtering and aggregating data.

Structure:

DimCustomer

|

DimProduct — FactSales — DimDate

| DimRegion

Key Features:

• Simple and natural system: Easy to understand, easy to move around.

• Facts: This table has numerical data in the form of some business transactions like sales amounts, quantities and number of orders etc.

• Dimension tables - Contain descriptive attributes of an entity, meaning the categorical data like the product category, the customer description/attributes and time hierarchies.

E-Commerce Sales Example:

Take an online retailer that sells goods through its website. For performance analysis using Power BI, the company creates a Star Schema containing these tables:

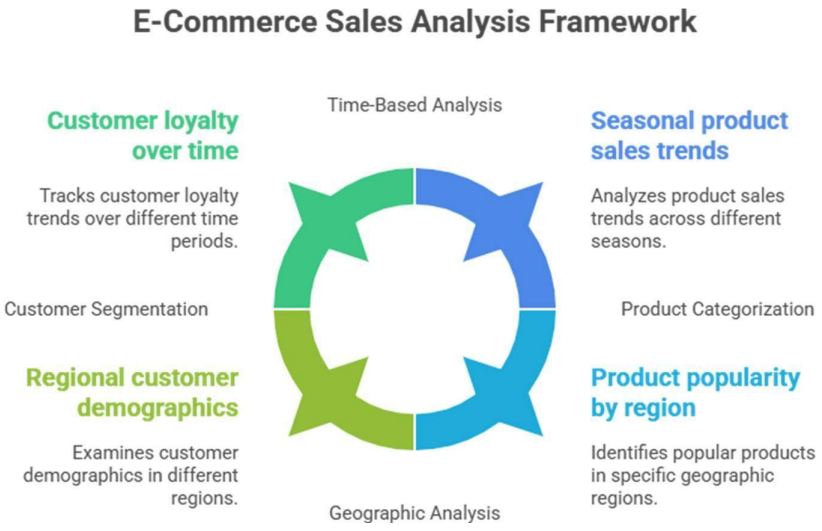


Figure: E-Commerce Sales Analysis Framework

• FactSales (Fact Table):

o Columns: SaleID, ProductID, CustomerID, DateID, RegionID, QuantitySold,SalesAmount,

Discount

o Records every click on the site.

• DimProduct:

o Fields: ProductID, ProductName, Category, Brand, Price

o Records information of the products that are sold over the internet.

• DimCustomer:

o Columns: CustomerID, FullName, Email, AgeGroup, MembershipLevel There are first five rows from above o Input Rows: 5 Describe the output with all columns and their types as shown below.

o Holds customer profiles to enable segmentation and targeting.

• DimDate:

o Columns: DateID, Date, Month, Quarter and Year

o Enables filtering and aggregating of sales over time.

• DimRegion:

o Columns: RegionID, RegionName, Country, Zone

o Geographical considerations for assessing regional performance.

This type of design enables analysts to address key business questions such as:

• What are the overall sales amounts by product category and region?

• How do PWPs purchase different from Regular purchases?

• How much have sales increased over the past four quarters?

Advantages:

• Improved performance for queries and visuals: Work on joines optimization and lower complexity improves report response.

• Easier to understand for report end-users: To avail the advantages of DAX those already using DAX can better understand the underlying model.

• Eases DAX measures and time intelligence: A clean structure for easy-to-understand DAX calculations such as YTD, QTD or filtering by dimensions.

Use Cases:

• Dashboards where fast response times are needed: Suitable for executive level summaries and live reporting.

• Reports that require ad hoc filtering and grouping: Makes interactions with slicers and visuals easier.

• Clean and de-normailzed data models: Appropriate for datasets with easy joins and no normalization.

This is applicable for most of your Power BI modeling best practices, if you are in data where we want to have performance and a good user experience.

5.2.3 Snowflake: Design and Applications

Snowflake Schema: the star schema can be forced into more normalization; this leads to what is known as a snowflake schema.

into sub-dimensions. This entails that not all dimension tables gets denormalized into one table.

Structure:

DimProductSubCategory — DimProductCategory

|

DimProduct — FactSales — DimDate

| DimCustomer

Key Features:

• Dimension tables may refer to other dimension tables.

• Enforces denormalized data (no redundance information)

• More Tables, but less data redundancy (also known as duplication).

Advantages:

• Saves storage by avoiding repetition

• Useful for preserving the integrity and structure of data in large, complicated datasets

• Relevant when source systems already normalise data

Use Cases:

• Data warehouses with enterprise data models containing dimensions that are large and hierarchically structured

• Use-cases in which storage density is valued more than the performance.

• Models maintained by expert (typically senior) developers or data engineers

Did You Know?

You can do that with a snowflake schema in Power BI, but performance may suffer if there are too many joins. It's a good idea to flatten your snowflakes out into star schemas as part of the ETL — Except when you really need your data to be highly-normalised (for accuracy or regulatory reasons).".

5.2.4 Star vs Snowflake: Contrast and Business Situations

When creating a data model in Power BI, or in any analytical tool that relies on relationships between tables, the lowest level of automatically generated relationship schema is one-to-many. There is no one-size-fits-all answer as to which structure to use for each need: data complexity, storage efficiency, user expertise, or query speed.

The below table shows a comparison between the two types of schema based on structure, advantages / disadvantages and proper business scenario for the same.

Criteria	Star Schema	Snowflake Schema
Structure	Flat, denormalized	Multi-level, normalized
Complexity	Simple and intuitive	More complex due to hierarchical dimensions
Performance	Generally faster because of fewer joins	Slightly slower as more joins are required
Ease of Use	Easier for business users and analysts to understand and work with	Requires deeper understanding of data relationships
Storage Efficiency	May include redundant data	Optimized for storage, avoids duplication
Data Integrity	Less strict; may allow inconsistencies	Maintains high data integrity through normalization
Best For	Speed, simplicity, interactive dashboards	Managing complex, hierarchical or relational data
Example Use Case	Retail sales dashboards, marketing campaign reporting	Enterprise HR systems, financial systems with multi-level departments

Business Scenario Comparison:

• Retail Sales Dashboard:

o In this case A Star Schema is a good choice. This allows for fast aggregations/filtering of metrics like sales total, revenue etc across product categories, date ranges, regions. It supports easy, real-time interaction and runs through dashboards without a hiccup.

• Enterprise HR Database:

o Snow flake Schema is suitable here. There might be several hierarchies in the HR data model: employee → department → division, hierarchies for payroll and benefits. The denormalized model eliminates redundancy and keeps referential integrity through several related entities.

5.3 Building Hierarchies

Hierarchies in Power BI are used to define a multi-level data, such as products organized by category and sub-category, and allow users to drill down on visualizations. The purpose of hierarchies is to make the exploration of data easier by classifying these into a structured set of levels through which users can easily drill.

5.3.1 Date Hierarchies (Year, Quarter, Month, Day)

Date hierarchies are also the most common and basic type of hierarchies used in the data model of Power BI. When you load a column of date values into Power BI, it automatically creates a default date hierarchy, which you then use to drill down on that data across all the different time periods. This integrated hierarchy reduces the complexity of time focused analysis without any manual setup.

Default Levels in a Date Hierarchy :

The auto-calculated date hierarchy that Power BI creates for you, contains these levels:

• Year: Aggregates data by year (e.g., 2023, 2024)

• Quarter: Splits each year into four (Q1, Q2, Q3, and Q4)

• Month: Further divide quarters into months (January through December)

• Day: Day is the primary level that represents each day / date and has the most granular information This multi-level structure helps the users to :

• View summaries (e.g., sum of revenue) by year, quarter, month or day

• Dive into trends beginning with an annual overview and drilling down to daily insights

Use time intelligence calculations like YTD (Year-to-Date), QTD (Quarter-to-Date) and period comparisons.

Month to Date (MTD) is useful when you need to compare the total through a current date with what was expected at this point of time.

Example from a Power BI Date Table:

Let's say you have a Power BI model with a Date table that looks like:

Date	Year	Quarter	Month	Day
2024-03-15	2024	Q1	March	15
2024-07-10	2024	Q3	July	10
2025-01-05	2025	Q1	January	5

For this instance, a visual (for example, a line or matrix table) can leverage the Date hierarchy as below:

• Drill-down route: Year Quarter Month Day

• A user could first look at Total Sales in 2024, then drill to Q3, and then into July, ending with analyzing sales on July 10, 2024.

Use Cases:

• Tracking revenue growth: Get a view of how overall revenue grows monthly or quarterly.

• Sales seasonality Comparison: Determine when are the high and low seasons for sales over the years.

• Performance metrics vs previous time periods: Compare your Q1 2023 to Q1 2024 or July 2023 to July 2024.

Custom Hierarchies:

There is no constraint to the users over the default hierarchy. In Power BI, you can create custom date hierarchies by adding calculated columns to the Date table for:

• Fiscal year and fiscal quarter

• Custom time periods (e.g. bi-weekly reporting)

• Week numbers or ISO weeks

These improvements allow for on the fly time bucketing analysis, particularly in institutions on non calendar fiscal times.

5.3.2 Geography Hierarchies (Country, State, City, etc.)

Geographic data frequently has levels of detail, so geography hierarchies are useful for location-based analysis.

Typical Geography Hierarchy Levels:

• Country

• State / Province

• City

• Region / Zone

You can arrange these fields in a hierarchy in Power BI so that, consumers of the report can:

• Drill down from national KPIs to local performance

• Filter and group visuals by geography levels

• Map graphics for multi-level exploration of the land

Use Cases:

• Regional sales comparisons

• Territory planning

• Analyzing customer distribution by location

Data types should be assigned properly to show the visualization.

5.3.3 Product Hierarchies (Category, Subcategory, Product)

Product hierarchies can be used for logically arranging product related information. It's a routine in retail, production or stock keeping.

Common Product Hierarchy Levels:

• Product Category (e.g., Electronics)

• Product Subcategory (e.g., Mobile Phones)

• Product Name/SKU These hierarchies keep the users:

• Block, create and analyse the sales or profit by store or product level

• Drill down to subcategories to find best or struggling products

• Visualize inventory or pricing structures

Use Cases:

• Sales analysis by product line

• Product performance dashboard

• Inventory breakdown by category

Typically, we create these hierarchies in the Product Dimension table that can be used inside visuals to continue to drill down.

5.3.4 Hierarchies in Reports and Visuals

You can then use hierarchies in Power BI with a variety of visuals such as tables, matrices, column charts and map visuals.

Advantages of Using Hierarchies in Visuals:

• Drill Down/ Drill Up – Users can drill from summarized data to detailed data (Ex: From Year to Month)

• Expand All: Expands all the hierarchy levels.

• Hover or Click Navigation: Interact with the data without making multiple visuals.

Using a Hierarchy in a visual:

Pull the entire structure into a viz (a barchart) in fact.

You drill down to the next level using on-screen buttons.

Mix and match hierarchies with filters, slicers and tooltips for more interactivity.

Best Practices:

• Double check to ensure that the hierarchy order is logical and correct.

•Don't create too many levels in one visual.

• Added the ability to customize axis labels when using hierarchy based charts for better clarity.

Did You Know?

"Fact: If I create a hierarchy in a Viz [visual], Power BI auto enables 'Drill Mode', where users can click and drill into level of details." This interaction is good for storytelling and user exploration, and no DAX writing or extra logic to be created as the hierarchy takes care of it."

5.4 Best Practices in Data Modeling

Getting a solid efficient scalable data model is key to having great Power BI reports. With the well-designed models, comes good performance, data fidelity and ease of use. This article includes best practices for designing and managing your data models.

5.4.1 Designing Efficient Data Models

A good data model doesn't overcomplicate things while allowing everything to be used. It helps to keep your data organised and searchable.

Key Practices:

• Use Star Schema whenever possible. Having multiple dimension tables associated with a single fact table.

• Avoid flat, wide tables. If possible normalize to avoid duplication.

• Load only the necessary data. Create non-used columns and tables during import.

• Use proper data types that use the least memory and are efficient.

• Name your tables and fields meaningfully to increase the readability.

Benefits:

• Faster loading times

•End user navigation through reports was simplified

• Easy DAX measures and calculated columns

5.4.2 Avoiding Circular and Ambiguous Relationships

Circular and ambiguous Power BI data model relationships can be extremely dangerous ones that cause reports to return incorrect results, corrupting your data model. These problems will usually be inherent from badly defined or unsuitable.relationships between tables.

conflicting loopy or confusing filtering. Power BI will not actively allow circular relationships, however we can be left with ambiguous relationship that lead to incorrect aggregations of the data or unexpected appearance of visuals.

Problems These Can Cause:

• Incorrect measure values: The measures either returns wrong value for the filters as a result of cascading effect or more than expected values.

• Filter collisions: When filters are applied from different directions, the result might be conflicting or unanticipated behaviour in visualisations.

• Modeling issues: In Power BI, when you're creating a model, circular paths in data (relationships between two or more lookup tables) aren't allowed.

Example of a Circular Relationship:

Let’s take an example of a simplified sales model that consists of the following three tables:

• Sales

o Columns: SaleID, ProductID, CustomerID, Amount

• Products

o Columns: ProductID, CategoryID, ProductName

• Customers

o Columns: CustomerID, RegionID, CustomerName

• Regions

o Columns: RegionID, RegionName

• Categories

o Columns: CategoryID, CategoryName Now assume:

• Sales is related to Products with a ProductID

• Sales is related with Customers using CustomerID from the Customers.EntityFrameworkCoreomidouScaffold\TemplateDatabase.

• Products connects to Categories

• Customers connects to Regions

This setup works fine. But: If the same someone also tries to hook up Products and Customers directly (e.g., by using a common RegionID, or perhaps some marketing mapping table), circular dependency would form:

Sales → Products → Customers → Sales

Power BI will recognize this circularity and deny the ability to create this relationship, because if filter were applied down a chain of 2 or more relationships, then there would be 2 or more paths from a column in one table to a different table.

Correction Technique: Bridge Table You will use a bridge table.

In this case, the best would be to add a bridge table that would resolve any conflicting paths. For example if using the previous example we actually want to analyse product popularity by customer region then instead of linking Customers directy with Products do this through a Customer-Product Bridge Table:

CustomerProductBridge

• Columns: CustomerID, ProductID

This table is a denormalized view of who interacted with what products (for ex, purchases, views or wishlists). The model now dodges a Products – Customers relationship, eradicating the circularity while retaining being able to analyse via explicit DAX Queries.

Prevention Techniques:

• Bring one-to-many relationships when you can - side is "one", and facts (e.g., Sales) are on the "many" side.

• Minimize bi-directional filtering: Use one directional filters as much as possible for clear and predictable filter flows.

• Bridge table usage: For cross-filtering between two tables in a many-to-many pattern, bridge table can be introduced to separate out the filter context.

• DAX for context control: In more complex cases, apply DAX functions such as TREATAS(), USERELATIONSHIP() or CALCULATE() to specify what filters should be and how they should behave.

When working with composite models: Be careful when using the model; for example, do not have importing and direct-query sources create overlapping filter paths or ambiguous relationships.

Good modeling design prevents not just circularity and ambiguity, but also faster report performance and ease of understanding for the end-users.

5.4.3 Reducing Redundancy and Ensuring Scalability

While (as I have demonstrated) iterative functions in DAX look quite neat, It's is very important to be accountable to metaphorically "hose down" your data models, getting rid of any features we do not need, using clean code which doesn't give you a headache when you come back after Christmas and the report has broken. A proper architecture model avoids redundancy as much as possible, performs radically higher without degrading to larger datasets.

Strategies to Reduce Redundancy:

• Eliminate unnecessary columns

Drop columns that don’t appear in visuals or aren’t used in any calculations, especially text based high cardinality like transaction descriptions, long notes or verbose addresses. These raise memory consumption and degrade model performance.

• Remove calculated columns where possible

Where ever possible, use measures instead of calculated columns for real time calculations. It computes measures on request and stores columns that are calculated, thus taking up more memory.

• Avoid duplicating dimension tables

Do not reduplicate Date, Product or Customer among over fact tables, and maintain a single Fact table for Data Analytics. This technique centralises filtering, making it independent to questions and thus keeping the model shallow.

For Scalability:

• Use parameters and query folding

Put Power Query parameters around it for partial data while developing and the ability to scale a filter (e.g., only pull in the last 12 months.) Make sure your transforms will be folded to the query, so nothing happens in Excel.

• Split the large tables at source

On SQL based sources (SQL Server, Azure) partitioning big fact tables by date or region can gain a lot of performances on the data refresh and slicing process.

• Optimize for refresh performance

Turn of auto date/time where it is not needed, reduce the amount of transformation steps in Power Query and use calendar tables.

Incremental refresh if applicable to minimize loads for full dataset.

Note on Denormalization vs. Normalization:

Normalizing is structuring the data to minimize redundancy by breaking the data into several related tables. This will fit well in more complex enterprise systems (for example, ERP or HR databases)and will reduce copying and ensure referrantial integrity.

• Related data can be flattened with denormalization into fewer tables to decrease the number of joins and increase query performance. This is more for analytics and reporting use-cases where fast aggregation is key, along with ease of use.

The decision to select one over the other is driven by business requirements: normalized are best for long-term data maintenance and integrity, whereas denormalized are ideal for high-performance visual reporting.

Business Example:

A big retailer with transactional data both in store and online. They first de-normalized their data by having two customer tables (on-line and in-store) and two Date tables (for different types of reports). This led to code repetition, non-uniform filters, and a bulky model.

To improve efficiency:

• They put all transactions in one table called FactSales.

• Established a common DimCustomer and shared Date table.

• Deleted profit margin calculated columns, and replaced them with DAX measures.

• Added support for incremental refresh of historical data partitions.

This reduced the model size by 40% and brought great improvement to report loading time, making sure we could sustainably scale as data volume would increase month after month.

5.4.4 Performance Optimization Tips

Performance tuning Power BI models is a necessary task to deliver high quality reports which are responsive and fast—even when working with large, complex data sources. Good performance needs to have efficient memory consumption, skinny data model, well-crafted DAX expressions and good handling of refresh. Power BI has a lot of great features and best practices so your models scale.

Key Tips for Performance Optimization:

• Use digits instead of letters

String keys (e.g., of customer names or product identifiers) take much more space than integers. Your table relationship should always be defined with numeric surrogate keys to use space more efficiently and make your joining faster.

• Avoid calculated columns unless necessary

Computed columns add to the in-memory storage, and more so if the tables are large. Wherever possible, replace them with DAX measures that are computed at query time and deliver both memory utilization efficiency and responsiveness.

• Enable data reduction techniques

o Filters - Restrict data at the time of import with query filters or Power Query parameters.

o Summarization: Summarize data at source or directly in Power BI by using grouped summaries.

o Row-Level Security (RLS): RLS not only secures data but also minimizes the size of visible data for every user which impacts rendering.

• Use the Performance Analyzer

Located at View → Performance Analyzer, you can:

o Track the performance of single visuals.

o Split up total load time by DAX Query, Visual Render and others.

o Pinpoint bottlenecks due to poor visuals or costly measures.

One great use for this is to debug slow visuals or see the impact of filters on query time.

Aggregations are important for large-scale datasets

Aggregations in Power BI enable you to precompute summarized or aggregates values of few measures at certain levels (week, month, quarter or year etc.) using more granular data available in detail level tables.

• Aggregation tables can have millions of records less than a detailed fact table when allowing the user to query data at a summary level and hence users get quick responses.

• Power BI's storage engine makes the decision of when to use the aggregation table or the detailed, based upon what a user is asking for.

• Imports can be brought in with details left on DirectQuery – hybrid is the key!

Example:

If you have a 100 million row FactSales table, then EntityStringAttribute (and the resultant hierarchy and dimension) is going to work fine. As mentioned earlier, you can create rolled up tables that summarize sales by Product, Year and Region. If a visual asks for top-level totals, Power BI knows to use this small table and the queries run much more quickly.

50 Whole model 2% Import + Direct QueryComposite models with Import and DirectQuery.setOnAction60.viewModel.onComplete().then(function(styles){ 70 let sum = Object.values(vm.processed).reduce((a, b) => a + b, 0); In this example the publisher has provided two alternative tools for building an application: (1) InterestKeeper to calculate interest and bank the payments or (2) CalculateInterestWithLedger that delegates the payment banking responsibility to another entity.

Composite models give you the choice of combining Import mode and DirectQuery mode in a single report. This enables you to:

• For better performance, keep smaller and more commonly used tables (e.g., Date, Product) to Import mode.

• Seamlessly connect large datasets (such as transactional data in SQL Server, or sales data in Dataverse) for powerful analysis using DirectQuery without importing into your Power BI solution and subsequently running the risk of reaching server's memory limits.

• Use Dual mode, which automatically switches between Import and DirectQuery depending on the query.

Best Practices with Composite Models:

• Be careful with the relationships of Import to DirectQuery tables–use one direction of filtering

to avoid ambiguous results.

• Restrict the visuals pulling from DirectQuery sources to prevent it from being slow while making interaction.

• Performance test your reports with Performance Analyzer after you implement Composite Models.

Practical DAX Example:

Instead of the calculated column as this: Sales[Revenue] = Sales[Quantity] * Sales[Price] Use a DAX measure:

Total Revenue = SUMX(Sales, Sales[Quantity] * Sales[Price]) That would be:

• Uses less memory (since the column isn't in the model).

• Evaluation is just in time and improves query performance.

• Enables to reuse over various visuals and minimizes data duplication.

"Task: Optimizing a Power BI Model for Speed"

Instruction to the Student:

You are provided with a. pbix file with already a ready and slow working model.

Open the file and identify:

o Unused columns in visuals

o Calculated columns which could be translated to DAX measures

o Text fields that act as relationship keys

Optimize the model by:

o Droping columns unused in all tables

o Substituting one or more calculated column by a DAX measure

o Converting relationship keys to numeric as much as possible

Analyze with Performance Analyzer this before and after loading times to compare.

Deliverable:

Provide a short report (150–200 words) describing your optimizations, and some screenshots from the Performance Analyzer.

Knowledge Check 1

Choose the correct option:

What is best described as a Star Schema in Power BI?

A) Each table connected to All the Other tables model.

B) One single flat table with all data joined together

C) One fact table connected to multiple dimension tables

D) A model including only one to one relationship.

Why do we have to create hierarchy in Power BI visuals?

A) To prevent end users from seeing fields.

B) To arrange the contents of a table in alphanumerical order.

C) In order that users can drill down through increasing levels of detail.

D) To increase data loading speed

What is a possible problem with cyclical relationships within the model of a Power BI workbook?

A) Enhanced report performance

B) Automatic drill-through functionality

C) Filter logic clashes and failures in models

D) Better DAX performance

What is the special feature Star schema and Snowflake schema?

A) Star schemas normalize dimension tables

B) Snowflake schemas do not use relationships C) Star schemas are always normalized.

C) Star schemas allow easier navigable relationships and improved performance.

D) Snowflake schemas are wide and do not need to be joined

In Power BI what does it mean by setting the cardinality of a relationship to “One-to-Many”?

A) Only the keys of both tables must be unique.

B) Many rows in a table correspond to many rows in another table

C) A row in the first table corresponds to multiple rows in the second table

D) There should be only on row for each table

5.5 Summary

ϖ In this chapter, students learned about the basics of data modeling in Power BI and how to build a good performing, accurate, and scalable model. The chapter started with tables, primary and foreign keys, types relationships; designing and using Star or Snowflake schema.

ϖ Different types of building hierarchy such as Date hierarchy, Geography hierarchy and Product based hierarchies were also explained which help in drill down the analysis in reports. Lastly, students learned about considerations for data modeling including eliminating circular relationships, minimizing redundancy, and maintaining optimal performance of the model.

ϖ These ideas make it so analyst can build better tables to get consistent, high performance, and easily maintain the Power BI solution no matter which Power BI Solution they are working on.

5.6 Key Terms

Fact Table - A table that holds data such as sales or revenue that can be aggregated.

Dimension Table - A table used to provide context around facts, such as products or customers.

Primary Key- The unique reference in the table that refers to each record.

Foreign Key - A field in a table that is the primary key of another table.

1 Star Schema - A data model consisting of one fact table and multiple dimension tables.

Snowflake Schema – A normalized form of a star schema in which dimension tables are broken down into additional sub-dimensions.

Hierarchy - Hierarchical relationship between dimensions (for example Year > Quarter > Month > Day), applicable for drill-downs.

Cardinality - Whether two tables are related with each other in the Database (One to Many etc).

8) AMBIGUOUS RELATIONSHIP: A relationship that connects one filter path to more than other, commonly resulting in an error.

Query Folding - When you send Power Query steps back to the data source for better performance.

5.7 Descriptive Questions

Explain about fact and dimension table? Provide examples.

Describe why you would want to use a primary or foreign key for establishing relationships.

List the components and applications of a star schema.

Distinguish between star schema and snowflake schema in respect of performance a complexity?

What are Power BI Hierarchies? How are they used in visuals?

Provide a scenario in which sales dashboard could be improved by using geographic hierarchy.

What is circular reference and how it can be avoided while working with Power BI?

What are 3 best practices for enhancing the performance of a Power BI data model?

Describe how cardinality works in relationships in Power BI.

What does Power BI Model View do for relationships?

5.8 References

1. Microsoft Docs. (n.d.). Relationships in Power BI Desktop. Retrieved from: https://learn.microsoft.com/en- us/power-bi/transform-model/desktop-relationships-overview

2. Russo, M. & Ferrari, A. (2020). The Definitive Guide to DAX. SQLBI.

3. Chhabra, S. (2021). Power BI Data Modeling Made Simple. BPB Publications.

4. Gil Raviv. (2018). Collect, Combine, and Transform Data Using Power Query. Microsoft Press.

5. Power BI Community Forum. https://community.powerbi.com

Answers to Knowledge Check

Knowledge Check 1

1. C – A central fact table linked to multiple dimension tables

2. C – To allow users to drill down through levels of detail
3. C – Conflicting filter logic and model errors
4. C – Star schemas offer simpler relationships and better performance
5. C – One row in the first table relates to many rows in the second table

IBI_Unit 6_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127445822

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:01 PM GMT+5:30

File Name

IBI_Unit 6_V3.docx

File Size

161.8 KB

32 Pages

6,724 Words

38,098 Characters

0% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

2 Not Cited or Quoted 0%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

1 Integrity Flag for Review

Hidden Text
23 suspect characters on 1 page
Text is altered to blend into the white background of the document.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **2** Not Cited or Quoted 0%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Publication

Mandeep Mehta. "Microsoft Excel Functions Quick Reference", Springer Science a... **<1%**

2 Submitted works

King's College on 2020-07-10 **<1%**

Unit 6: DAX (Data Analysis Expressions) Basics

Learning Objectives

1. Understand the purpose and role of DAX in Power BI for creating dynamic and calculated data fields.
2. Differentiate between Calculated Columns and Measures, and identify appropriate use cases for each.
3. Apply basic aggregation functions such as SUM, AVERAGE, COUNT, MIN, and MAX within DAX to perform numerical summaries.
4. Use logical functions in DAX (such as IF, AND, OR, SWITCH) to perform conditional evaluations and branching logic.
5. Write and interpret simple DAX expressions to manipulate and analyze data in Power BI.
6. Understand the concept of row context vs filter context, and how these affect DAX calculations.
7. Develop calculated fields that support key business metrics and improve the analytical depth of reports.

Content

6.9 Case Study

6.0 Introductory Caselet

"Ayesha's Reporting Challenge: Making Sales Insights Automation with DAX"

Background:

Ayesha is a business intelligence intern at UrbanTech Retail, which follows products sales across cities. She has been tasked to improve a sales report in Power BI, which the company's regional managers rely on.

It's a static report right now – it just displays the transaction data as it is without showing some KPIs like Total Revenue, Profit Margins or maybe Top Performing Products. And when management decides it wants to use some other metric, you have to manually calculate it in Excel or alter your SQL.

Ayesha starts learning DAX (Data Analysis Expressions), the formula language of Power BI, which she uses for dynamic calculation. She begins by adding calculated columns for "Profit" and measures for "Total Revenue" and "Average Order Value". After she creates additional measures with functions such as SUM, IF, and CALCULATE, her report becomes more interactive and informative.

Ayesha wraps up her week by delivering a robust dynamic dashboard, which auto-updates based on user filters and now supports comparison of profits across region as well as customer-wise segmentation—all using DAX.

Critical Thinking Question:

Why are DAX Measures better off than hardcoding calculations in data sources? What are the disadvantages of depending only on calculated columns?

6.1 Introduction to DAX

What is DAX?

DAX (Data Analysis Expressions) is a formula language that lets you create custom calculations and aggregations in Power BI, Excel Power Pivot, and tabular models on SQL Server Analysis Services.

DAX allows users to:

- Define metrics for the calculations – such as Total Sales, Average Profit and so on.
- Create calculated columns for calculated data, such as margin or customer segments

• Generate Filter Context and Row Context Logic On The Fly

• Support time intelligence, rank, conditional logic and more [!NOTE] > Multi-part collections don't rely on the same flow as a typical slot-filling prompt.

Key Features of DAX:

• Excel-like syntax but with more functionality designed for data models

• Works for both Row-Level (calculated columns) and Aggregate-Level (measures) logic

• Performance optimizations for data analysis with in-memory analytics engines (VertiPaq)

Basic DAX Syntax:

• A DAX formula always begins with an equal sign =

• Uses techniques such as SUM(), IF(), CALCULATE(), FILTER() etc.

Example 1 – Calculated Column:

Profit = Sales[Revenue] - Sales[Cost]

Example 2 – Measure:

Total Sales = SUM(Sales[Revenue])

Why DAX is Important in Power BI:

• DAX makes reports intelligent and flexible

• Creates interactive KPIs that react to slicers and filters

• Help business users to make sense and model data on the fly You will need DAX to unlock the power of your static information for real time analytical insights.

6.1.1 Purpose and Importance of DAX in Power BI

DAX serves to a make your reports and dashboards not just more powerful but also more intelligent by enabling you to derive values that are not directly stored in your data. For instance, you might have a sales table full of individual sales amounts, but if you want to calculate "year-to-date sales," or "average sales per customer," then it's DAX time.

So, the significance of DAX in Power BI is:

• Advanced Analytics: You can create measures like totals, average, growth percentages and rank.

• Dynamic Analysis: Results in DAX calculations react dynamically to filters and slicers applied in reports, as opposed to static reporting where numbers shown would remain the same regardless of filter or slicer selection.

•Data Modelling: DAX allows the definition of calculated columns and measures which enrich your data model with data not already in your original sources.

• Business Insights - It assists in translating raw data for decision making into actionable insights.

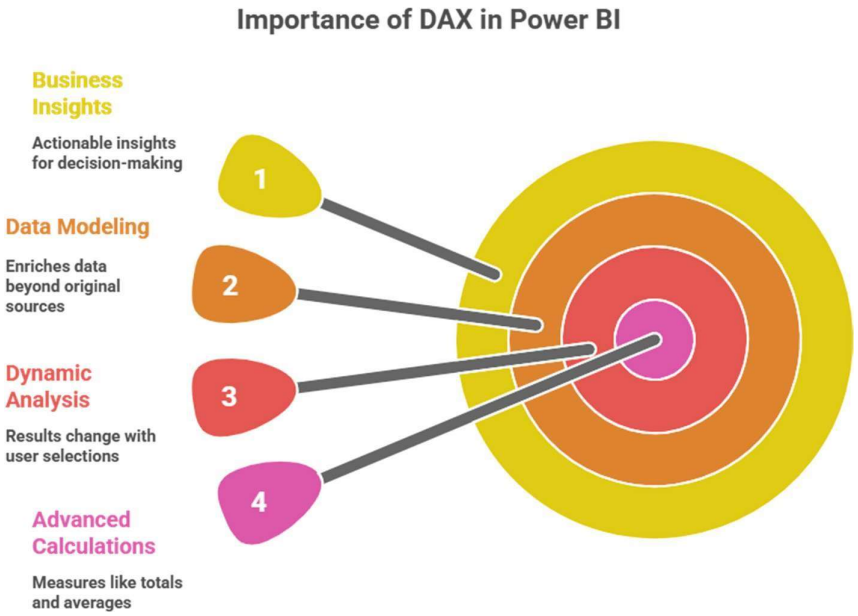


Figure: The importance of DAX in Power BI

6.1.2 Syntax and Structure of DAX Formulas

DAX formulas have a lot in common with Excel formulas, just as you might expect from something modeled on the functionality of Excel!

Starts with an equal sign (=)

Example: =SUM(Sales[Sales Amount])

Uses functions, operators, and values

o Functions: Existing formulas (e.g., SUM, AVERAGE, IF and CALCULATE).

o Operators: Arithmetic (+, -, *, /) and Relational (=,], Logical (AND, OR).

o Values: Columns, tables, or constants.

References to columns and tables

o Table Name['Column Name'] indicates the name of column.

o Example: =AVERAGE(Products [Price]) will average the price from the Products table.

Not case insensitive but you should try to write this consistently for readability.

In a nutshell the DAX syntax is Excel like, but it has special functions that are made to operate on relational data and filter contexts.

6.1.3 Data Types in DAX (Numeric, Text, Date/Time, Boolean)

Data types are an essential factor behind the powers of DAX that define how values interact with data and determine results of calculations in Power BI. Selecting appropriate data type is crucial to enable the valid calculations, performance optimizations and accurate visualization and measures.

DAX functions react differently to each data type - e.g.

• Aggregation functions like SUM, AVERAGE, MAX cannot be used on non-numeric data.

• Text-only functions like CONCATENATE or LEFT can only be used on text.

• Date intelligence functions like DATESYTD, MONTH or DATEDIFF expect Date/Time datatypes.

• Conditional statements like (IF, SWITCH, FILTER) tend to return or evaluate Booleans.

Table : Types of data in DAX with examples in Power BI

Data Type	Description	Example Value	Typical Use Cases
Numeric	Includes integers, decimals, and currency	100, 45.67, 2000.50	Calculations: totals, averages, percentages
Data Type	Description	Example Value	Typical Use Cases
Text	Sequence of characters (string values)	"Customer Name", "ABC123"	Labels, categories, filtering, concatenation
Date/Time	Combined date and time values	01/01/2023 12:30 PM	Time intelligence, date comparisons, period filters

Boolean	Logical values: TRUE or FALSE	TRUE, FALSE	Filters, conditionals, IF or SWITCH expressions
Blank	Special null-like value in DAX (not "null")	BLANK()	Missing data, optional conditions, default values
Variant*	Implicitly handled when types are mixed	Auto-converted	Used internally by DAX in dynamic type operations

Note: "Variant" is not an actual visible or selectable data type; it's simply that DAX has strong enough typing to do these dynamic things inside the expression.

Examples in Context:

• Numeric Example:

• Total Sales = SUM (Sales [Amount])

• Text Example:

• Full Name = CONCATENATE (Customer [FirstName], " ", Customer[LastName])

• Date/Time Example:

• Sales Year = YEAR (Sales [Order Date])

• Boolean Example:

• High Value Order = IF(Sales [Amount] > 1000, TRUE, FALSE)

• Blank Example:

• IF(Customer [Region] = "", BLANK (), Customer[Region])

6.1.4 Evaluation Context: Row Context vs Filter Context

In DAX, you are the context in which a formula is evaluated. It is not that DAX formulas are computed static to the model, when and how depend on where AND what you calculate.

There are two types of contexts in DAX:

Row Context

• Definition: Row context is the context that a DAX expression has when it's evaluated for each row of a table.

• Common Use: Measures (calculated columns), Running functions such as SUMX, FILTER, ADDColumns etc.

• Behaviour: The formula is aware of the row it's in, and can request other columns from the same row.

Example:

LineTotal = Sales [Quantity] * Sales[Price]

This formula is on a calculated column within the Sales table. It calculates Quantity * Price for each row, and caches the value as Total.

Filter Context

• Definition: Filter context is the set of all factors that determine what data is visible inside a formula—and therefore which rows contribute to a value calculation.

• Common Use: Functions that calculate totals and averages (SUM, COUNT, AVERAGE, etc.)

• Behaviour: The formula only computes results over the data subset which adheres to filters at the time of evaluation.

Example:

Total Sales = SUM(Sales[LineTotal])

If you add this measure to a visual filtered on Year = 2023, it will sum LineTotal's for sales in 2023 and nothing else.

Business Case: Row Context vs Filter Context at Work Company: Shop360 (E-commerce Company)

Objective: Determine sales per product and total by time filter.

Row Context Example – Column Formula:

Ravi, the analyst, would like to find out all values against every distinct Order in the Sales table. He creates a calculated column:

OrderValue = Sales[Quantity] * Sales[UnitPrice]

This involves row context--Power BI is going through the formula one row at a time, and multiplying quantity by price for each order.

This comes in handy especially if you want to do some deep diving into transaction level analysis or re-aggregation.

Filter Context Example – Measure:

The next requirement of Ravi is, he wants to show Total Sales for this year in a KPI card. He creates a measure:

Total Sales = SUM(Sales[Order Value])

If a slicer or visual filter contains Year = 2023, it means that Power BI considers this measure only for the rows where sales occurred in 2023.

This one-use filter context in that, depending on the selections made by users (region, year and customer segment) the formula automatically adjusts.

Did You Know?

“In DAX, whenever you do a calculation which has both row context and filter context applied to it simultaneously in the same calculation another thing takes place which is what Alberto refers to as a context transition. This happens for instance if you work with the CALCULATE function. It converts row context to filter context. If it were not for context transition, many of the most advanced DAX formulas would not produce correct results.

6.2 Calculated Columns vs Measures

I'm new to Power BI, and I see that in PBI there are calculated columns AND measures (all made via DAX functions), but what is the difference between both of them? Both apply logic and intelligence to your data model, but they work in different ways based on whether you are after row level calculations or aggregated values.

2

6.2.1 Definition and Use Cases of Calculated Columns

A calculated column is a new column that you add to an existing table in your Power Pivot workbook. Unlike measures, calculated columns are evaluated in a row context and the value that is generated is saved on the disk inside of your data model. After you create them, they are treated like any other column and you can use them in slicers, filters, visuals and relationships.

Key Characteristics:

• The values in a calculated column are calculated at data refresh time and not when the visual is interacted with.

• They reside in memory, so they can have an impact on a model's size.

• Calculated columns do have pull from the row context, perfect for calculation at a row level.

Use Cases:

Row-Level Calculations

When you want a value for each row that is based on other columns in the same table.

Example:

Profit = Sales [Revenue] - Sales[Cost]

This will add a new column Profit to the Sales table with value calculated for each row.

Creating Categorical Buckets or Labels

Well, calculated columns can be used to category or group rows which we can then use in slicers, visuals etc.

Example:

Sales Category = sales [revenue] > 1000? "High Sales" : "Low Sales"

This results in a column that can be either “High Sales” or “Low Sales” depending on the revenue of each sale.

Supporting Relationships or Filters

You can create the intermediate or filter column as a "bridge" field, say cleaned or extracted key.

Example:

YearMonth = FORMAT(Sales[OrderDate], "YYYY-MM")

This brings back a column we can use in visuals either as a group by, or slicer.

Performance Impact of Calculated Columns:

Although calculated columns are powerful, they can lead to performance and scalability issues if used indiscriminately:

• Because calculated columns physically materialized in memory, adding one increases the size of your data model.

• High cardinality columns (many unique values) are provided by greater memory consumption and less compression optimization.

• In many situations, the same logic can be performed more efficiently using measures (which are recalculated on demand, not kept).

Best Practice: You should use calculated columns only if the following rules apply:

• You need row-wise results.

• You need to build the categories/fields on relationships, filters, or grouping.

• The column is necessary for a visual item (like a slicer or legend).

Dataset Example: E-commerce Transactions

Assume a simplified Sales table:

Dataset Example: E-commerce Transactions

Assume a simplified Sales table:

OrderID	ProductID	Revenue	Cost	OrderDate
1001	P001	1500	1000	2023-01-10
Ordered	ProductID	Revenue	Cost	OrderDate
1002	P002	800	600	2023-01-12
1003	P003	2200	1500	2023-01-13

Calculated Columns:

• Profit = Revenue - Cost →Pluses: 500,200,700

• Sales Category = IF(Revenue > 1000, "High", "Low") → Include: High, Low, High

• Year Month = FORMAT(Order Date, "YYYY-MM") → Add: 2023-01, 2023-01, 2023-01

These columns are persisted in the model and can be used to create charts, slicers, groups.

6.2.2 Definition and Use Cases of Measures

Power bi measure is a type of dynamic calculation where you can write dax. Measures are also different from a Calculated column as there is no data for them to be stored in, they're performed at query time based on the filter context pushed down by slicers, visuals and user interaction.

Measures are perfect for KPIs, aggregations and performance metrics since they can be evaluated on demand, yet highly efficient in terms of indexing.

Key Points:

• Not stored in memory: Measures do not add to model size by being resident in memory.

• Dynamic and filter-sensitive: Results are dependent upon a work in progress filter context such as time period, region of the world, product category, etc.

• Performance driven: The measures use Power BI engine to compute the results at runtime in an optimized way.

Use Cases:

Aggregations: Totals, Averages, Ratios, Percentages

Aggregations by extension and basic level are the most typical usages of measures.

Example:

Total Revenue = SUM(Sales[Revenue])

This metric derives the total revenue by dynamically computing and refreshing aggregate view when users perform filtering, such as.

month, product category, or region.

Comparisons and Time Intelligence

Performance measures can be monitored over time with DAX time functions such as TOTALYTD, SAMEPERIODLASTYEAR or DATEADD.

Example:

YTD Sales = TOTALYTD(SUM(Sales[Revenue]), 'Date'[Date])

This allows users to calculate year-to-date sales, updating conveniently as people choose different date range or filters.

KPI Calculations for Dashboards

In Power BI, KPIs are most often used to measure some type of performance (e.g., sales), and need calculated measures that can be presented as tiles (or cards) on dashboards. Measures are the workhorses of these visualizations.

Example KPI Measure: Profit Margin

If you already have two measures:

Total Revenue = SUM(Sales[Revenue]) Total Cost =SUM(Sales[Cost]) • •

Profit Margin can be defined as:

Profit Margin = DIVIDE([Total Revenue] - [Total Cost], [Total Revenue]) This measure, calculates the profit of the segment as:

• Dynamically calculates percentage as profit margin based on inputs.

• Is reconciled for filters (margin by product/month/region etc).

• Data can be shown in KPI cards, bar charts, or conditional formatting.

6.2.3 Key Differences Between Columns and Measures

Understanding the difference between calculated columns and measures is essential for building efficient and scalable Power BI models. While both are created using DAX, they serve different purposes, operate under different contexts, and have different performance impacts.

Comparison Chart: Calculated Columns vs Measures

Feature	Calculated Column	Measure
Storage	Stored in the data model as part of the table	Not stored; calculated dynamically at query time
Feature	Calculated Column	Measure
Context Used	Uses row context – evaluates row by row	Uses filter context – evaluates based on filters and slicers
Granularity	Operates at the row level	Operates at the aggregated/report level
Performance Impact	Can increase memory usage, especially with high-cardinality data	More efficient – computed only when needed
Best For	Grouping, filtering, sorting, or creating relationships	Summarized calculations, KPIs, time intelligence, comparisons
Visual Use	Can be used in slicers, legends, filters, or axes of charts	Typically used in visuals like cards, matrices, or aggregated charts
Refresh Impact	Recalculated during data refresh	Evaluated during visual interaction

Examples Comparison:

Scenario	Calculated Column	Measure

Calculate profit per row	Profit = Sales[Revenue] - Sales[Cost]	— (not needed at row level)
Calculate total revenue	— (not row-specific)	Total Revenue = SUM(Sales[Revenue])
Label each row based on condition	SalesCategory = IF(Sales[Revenue] > 1000, "High", "Low")	—
Show overall profit margin	—	Profit Margin = DIVIDE([Total Revenue] - [Total Cost], [Total Revenue])
Use in a slicer (e.g., by product type)	ProductType = LEFT(Product[SKU], 3)	—
Display in a KPI card	—	Total Orders = COUNT(Sales[OrderID])

When to Use Which?

• Use Calculated Columns When:

o You need to categorize or rank the rows (e.g. SalesCategory as High/Low)

o You must relate tables to each other

o You need to perform the selection or and data on the new column.

• Use Measures When:

o You want to have your results (totals, averages, ratios) sumarized

o You're creating visuals for KPIs or summary metrics

o You are seeking to gain functionality that is slicer or filter aware

Did You Know?

"The size of your data model grows when you add a calculated column to it since it saves the values on a per row basis. A 'measure', on the contrary, doesn't jack up the size, because it will calculate just when you put that in a visual. That's why professional Power BI developers will often try to use measures wherever they can for better performance."

6.2.4 When to Use Columns vs Measures in BI Models

It's an important modeling decision to decide when you should use a calculated column or measure in Power BI. They are both constructed with DAX, but have different

purposes and affects your model differently: functionally, performance wise and in usage in visuals.

Use Calculated Columns When:

You need a value that has to be present at least once in each row

o Example: Determine the profit of individual transactions:

o Profit = Sales[Revenue] - Sales[Cost]

o This results in a new column that has the profit for each row of the Sales table.

You need to slice, filter, or set the axis on the new field

o For example, create a label field: “High Sales”, or “Low Sales”: for filtering/ segmenting :

o SalesCategory = IF(Sales[Revenue] > 1000?, “High”, “Low”)

The math is based on per row-level data

o Example: Sales by month, year based on order date for sales grouping:

o YearMonth = FORMAT(Sales[OrderDate], "YYYY-MM")

Use Measures When:

You want your calculated fields to be dynamic and react to filters

o Example: Total Sales for the Selected Region or Year:

o Total Revenue = SUM(Sales[Revenue])

You are looking for KPIs, Ratios or time intelligence

o For Example: Profit margin or Year-to-Date Sales amounted to:

o Profit Margin = DIVIDE([Total Revenue] - [Total Cost],[Total Revenue])

o YTD Sales = TOTALYTD(SUM(Sales[Revenue]), 'Date'[Date])

Performance and efficiency are important

o Measures are computed at query-time, not stored in-memory until being utilized; so it is efficient even for big or complicated data.

Real-World business use case: E-commerce Sales report For example, sheer figures of e-commerce sales are not helpful when we talk about sales done in Abu Dhabi or any other country.

Use case: Ravi is developing the Power BI report for Shop360, an e-commerce company. The two boxes in the top half of this diagram are for detailed, transaction-level analysis and management-summary type information respectively.

Ravi uses Calculated Columns to:

• Add a SalesCategory column:

• SalesCategory = IF(Sales[Revenue] > 1000, “High”, “Low”)

→ This field is a slicing for slicing the data.

• We will create a new profit column for each order:

• Profit = Sales[Revenue] - Sales[Cost]

→ This enables him to show PROFIT PER TRADE in intricate tablature.

• Add a YearMonth column:

• YearMonth = FORMAT(Sales[OrderDate], "YYYY-MM")

→ This is plotted on the x-axis of line charts to show time trends.

Ravi uses Measures to:

• Dynamically calculate key performance indicators such as Total Revenue and Average Order Value:

• Total Revenue = SUM(Sales[Revenue])

• Avg Order Value = AVERAGE(Sales[Revenue])

• Create Profit Margin KPI for executive dashboards:

• Profit Margin = DIVIDE ([Total Revenue] – [Total Cost], [Total Revenue])

• Show YTD Sales which dynamically changes as the year filter is changed: YTD Sales = TOTALYTD (SUM (Sales [Revenue]), 'Date'[Date])

6.3 Aggregation Functions in DAX

DAX aggregation functions are used to apply mathematical operations such as counting, sum, average, minimum and maximum over groups of rows. These functions are crucial in analysing large data sets as you can rapidly calculate sums, averages or other aggregations without having to sum all records manually.

6.3.1 SUM and SUMX Functions

In DAX, SUM and SUMX are formulas used to aggregate numeric data. But they behave differently, most notably with respect to how they treat expressions and context.

Knowing when and how to apply every function is crucial to writing correct DAX formulas.

SUM Function

• Explanation: Sum of all numbers in a single numeric column.

• Syntax:

• SUM(Table Name[Column Name])

• Example:

• SUM(Sales [Revenue])

This will sum up all the values from Revenue column of the Sales table.

• When to use:

o When you're looking to get a quick total from one column.

o Good for simple aggregations with no row-level logic.

SUMX Function

• Definition: Calculate a value for each record over the table and then sum those values.

• Syntax:

• SUMX (Table, Expression)

• Example:

• SUMX (Sales, Sales [Quantity] * Sales[Price])

First, it multiplies Quanty * Price row by row in Sales table and sum this all up.

• When to use:

o To be used when you have to calculate per row expression.

o Useful for calculated totals (for example: revenue, profit, weighted averages).

Worked Example: Understanding Row Context with SUM vs SUMX Sample Sales Table:

OrderID	Quantity	Price	Revenue
1001	2	500	1000
1002	3	200	600
1003	1	1000	1000

Using SUM:

Total Revenue (SUM) = SUM(Sales[Revenue])

- This sums the values in the Revenue column.

• Result:

• 1000 + 600 + 1000 = 2600

Using SUMX:

Revenue (SUMX) = SUMX(Sales, Sales[Quantity] * Sales[Price])

• This takes Quantity * Price for each row, and then adds those results.

• Row-by-row calculations:

o Row 1: 2 × 500 = 1000

o Row 2: 3 × 200 = 600

o Row 3: 1 × 1000 = 1000

• Result:

• 1000 + 600 + 1000 = 2600

In this example here, because the Revenue column already contains the product of Quantity and Price at each row level, two functions SUM and SUMX would return same value. But what if there was no Revenue column?

Without a Revenue column:

In such situation, for example with Quantity and Price present in table, using SUM(Sales[Revenue]) would not work as the column is not there. SUMX on the other hand, we sum with it dynamically using below:

SUMX(Sales, Sales[Quantity] * Sales[Price]) This is why SUMX is perfect for:

• Calculating values not already stored

• How to create dynamic totals which is based on more than one column

• How to deal with logic that varies per row (i.e. conditional multipliers, discount rates)

Did You Know?

“Whereas SUM can only operate on one column, SUMX will support the table aggregate if needed and also row context in a table before performing the operation. Infact, SUMX is one of the famous iterator function in DAX, and iterator functions will always induce row context internally.

6.3.2 COUNT, COUNTA, and COUNTROWS

COUNT

• Description: Returns total rows in a column that have numeric value (numbers only).

• Syntax:

• COUNT(TableName[ColumnName])

• Example:

COUNT(Sales[OrderID]) → will count the number of OrderIDs in the column that have numbers.

COUNTA

• What it does: Counts the rows in a column that do not contain blank cells (applies to text, numbers, or dates).

• Syntax:

• COUNTA(TableName[ColumnName])

• Example:

COUNTA(Customers[CustomerName]) → count of all customer names.

COUNTROWS

• Definition: It is a command to count the number of rows of a table, no matter what data it holds.

• Syntax:

• COUNTROWS(TableName)

• Example:

COUNTROWS(Sales) → calculates the sum of all Sales row values.

Difference:

- COUNT → counts numbers in a column.
- COUNTA → count all not empty values in one column.
- COUNTROWS → counts all rows across the whole table.

6.3.3 AVERAGE and AVERAGEX

Both AVERAGE and AVERAGEX are DAX functions to aggregate values in columns, but they do so in different ways depending on whether a row-by-row aggregation or a

simple average calculation is required. Deciding which one is better for you depends on the form of your data, and what sort of business logic you need.

AVERAGE

- Definition: Compute the average of all elements in a single column.

• Syntax:

• AVERAGE(Table Name[Column Name])

• Example:

• AVERAGE(Sales[Revenue])

This gives you the average of all values in the Revenue column.

• When to Use:

o If you simply want an average of available column values.

o Ideal for simple numeric fields (e.g., price, cost, quantity).

AVERAGEX

• Definition: Processes a custom expression for each row, and finally averages them.

• Syntax:

• AVERAGEX (Table, Expression)

• Example:

• AVERAGEX (Sales, Sales[Quantity] * Sales [Price])

This works out a sales quantity per row and takes an average for all rows.

• When to Use:

o When you need to average a calculated value not present in the data.

o Required if you have complex requirements, e.g., weighted averages, per-row measures or conditional calculation.

Key Difference

Function	Evaluates on…	Row-by-row logic	Use Case
AVERAGE	A single column	No	Average of one field (e.g., revenue)
AVERAGEX	A DAX expression	Yes	Average of a calculated value

Worked Business Example: Weighted Average Price The generality of the set-theoretic method is illustrated by its application to pricing in business.

Scenario:

Ravi, a data analyst at Shop360 is interested in calculating the average selling price per unit where this weighted by the number of

units sold. If you did just an AVERAGE(Sales[Price]), for example, this would treat all prices equally even if only 1 unit was sold of that price i.e. it would show average price across entire period which may not represent the true nature of your businesses pricing.

Sample Sales Table:

OrderID	Product	Quantity	Price (per unit)
1001	A	10	100
1002	B	5	200
1003	C	20	80

Wrong: Simple Average (Unweighted) Simple Avg Price = AVERAGE(Sales[Price]) Which then results in:

(100 + 200 + 80) / 3 = 126.67

But this doesn't take into account the fact that Product C (price = 80) sold 20 units and should have more influence.

Right: Weighted Average Price WITH AVERAGEX

Weighted Avg Price = DIVIDE(

SUMX(Sales, Sales[Quantity] * Sales[Price]), SUM(Sales[Quantity])

)

Explanation:

• Numerator: Amount (Quantity × Price per row)

• Denominator: Total quantity sold

• Result: Weighted average price per unit sold

Calculation:

The Total Revenue = (Price of the First Variety x Number Sold) + (Price of the Second Variety X Number Sold) + (Price of Finals Variety x Number sold) i.e. =(10×100) + (5×200) + (20×80)= 1000 + 1000+ 1600=3600

So the total number of = 10 + 5 +20=35

(1) Weighted Average Price = 3600 / 35 ≈102.86

This is much closer to an average price based on real sales volume.

“Activity”

Instruction to Student:

You are working with customer spend data from the Sales table consisting of columns CustomerID, Quantity, and Price.

Create a measure with AVERAGE(Sales[Price]) to compute the average per product unit price.

Create another measure as follows: AVERAGEX(Sales, Sales[Quantity] * Sales[Price]) to get the

average order value per transaction.

Either compare the two results in a card visual, or table.

Explain: Why do the two averages tell different stories? Which is more helpful for explaining customer spending habits?

6.3.4 MIN and MAX Functions

In DAX, MIN and MAX functions are used frequently to find out the minimum & maximum value in a column. They are very crucial for trend comparison, performance evaluation and outlier detection in datasets like sales, profits or inventory etc.

MIN Function

Definition : Min(Note: The prayer status can have only 2 possible values which are "completed" and "pending") Answer is get the minimum value of prayer_datetime where it has not been completed.

• Syntax:

• MIN(TableName[ColumnName])

• Example:

• MIN(Sales[Revenue])

Return: Returns the smallest sales in all rows in Sales table.

MAX Function

• Definition: Returns the highest, or maximum, value in a column of numbers.

• Syntax:

• MAX(Table Name[Column Name])

• Example:

• MAX(Sales[Revenue])

Returns the maximum value of sales in the Sales table.

Example Trend Analysis: When Are We Seeing Sales Extremes?

Business Scenario:

Ravi, a data analyst at Shop360 wants to determine: How is sales performance trending and changing over the months?

• A time of year when sales are at their slowest

• One with the most sales in a month

This helps management understand:

• Non-peak times where marketing pressure should be intensified

• Measurement of peak months with highest product demand.

Aggregate Revenue by Month

Staffing Create a measure to determine total revenue per month:

Monthly Revenue = SUM(Sales[Revenue])

Now drag a visualisation onto the canvas (e.g., Column chart) and place Date[Month] on the axis, Monthly Revenue as the value.

Step 2: Compute Revenue MIN and MAX

Now add two measures that would calculate min and max revenue considering all of the months Min Monthly Revenue = MINX(VALUES('Date'[Month]), [Monthly Revenue]) Max Monthly Revenue = MAXX(VALUES('Date'[Month]), [Monthly Revenue])

• MINX and MAXX loop over all distinct months, calculate the monthly revenue for each one, and return the minutely sum.

lowest or highest value.

• You can also use these measures within card or line charts to show how sales are behaving over time.

Step 3: Use in Dashboard

• Line chart: A graph that shows monthly revenue over time.

• Card visual: Use Min Monthly Revenue and Max Monthly Revenue to gain a quick insight.

• Conditional Formatting: Colour only the months which sales are equal to min or max.

Insights Provided:

Metric	Insight
Min Monthly Revenue	Identifies the **lowest-performing month**—useful for root-cause analysis
Max Monthly Revenue	Shows the **peak month**—helps in planning inventory and marketing strategy
Comparison (Max - Min)	Reveals the **range** or **volatility** in monthly sales

6.4 Logical Functions in DAX

DAX logical functions are useful to make decisions in Formula. They evaluate conditions (true or false) and then return answers on the basis of that evaluation. These are key functions when creating dynamic calculations, classifications and rules inside you your data model.

6.4.1 IF Function: Conditional Expressions

DAX has the same function, but its purpose is just like in Excel. It tests a logical condition and delivers one result if the condition is TRUE, and another if the condition is FALSE.

Syntax:

IF(Condition, ResultIfTrue, ResultIfFalse)

Example:

Sales Category = IF(Sales[Revenue] > 1000, “High”, “Low”)

• If Revenue >1000, then "High".

• Otherwise, the result is "Low".

This is a common pattern for applying business thresholds or statuses to rows.

Use Cases of IF in Power BI:

• Categorizing data

Example: Categorize sales as "High" and "Low" according to the amount of revenue they bring in

• Category = IF(Sales[Revenue] >= 500, "High", "Low")

• Creating binary flags

Example: Mark transactions that satisfy some condition

• HighValueFlag = IF(Sales[Revenue] > 1000,1,0)

• Custom labels for segmentation

Example:

• Result = IF(Customer[Segment] = "Enterprise", "Key Account", "Standard")

Note on nested IF's and performance:

Nesting multiple IF statements is nice for complex logic, but too much or improperly used nesting will:

• Degrade performance: Simulation of each IF is evaluated case by case, leading to excessive computational overheads.

• The readability and maintainability are horrible: you have multiple levels of nested if statements, which makes the code hard to read and debug.

• "Generate logic errors: you may forget conditions or make mistake as the comparing and fallback notations are mixed.

Example of Nested IF:

Rating =

IF(Sales[Revenue] > 2000, "Premium", IF(Sales[Revenue] > 1000, "Standard", "Basic"))
Example Using a DAX expression that looks into the Sales[Revenue] column and then returns either “Premium”, “Standard” or “Basic”.

• Evaluates in sequence:

o Revenue > 2000 → "Premium"

o Else, if Revenue > 1000 → "Standard"

o Else → "Basic"

Best Practices for Nested IFs:

• Use SWITCH() for many fixed conditions — it’s cleaner and faster for equality comparisons.

Example:

Grade = SWITCH(TRUE(), [Score] >= 90, "A",

[Score] >= 80, "B",

[Score] >= 70, "C", "F"

)

• For intermediate results, existing expressions can be saved within VAR sections:

Result =

VAR Revenue = Sales[Revenue]

RETURN IF(Revenue > 1000, "High", "Low")

6.4.2 SWITCH Function: Multiple Condition Processing Definition:

DAX SWITCH is not only more readable than nested IF-s and COPY-PASTE, it should be more efficient as well. So, provided a list of values to be matched against, it executes an expression in the above syntax pattern and returns the result of the first match found. If no match is found, it's possible to return an optional default value.

Syntax:

SWITCH(Expression, Value1, Result1, Value2, Result2,

...,

ElseResult

)

• Expression: The value to test.

• Value1, Value2: Possible matching values.

• Result1, Result2: Results to apply for each match.

• ElseResult: Optional default value when no match is found.

Example: Region Label Mapping Region Label = SWITCH(Sales[Region],

"North", "Region A",

"South", "Region B",

"East", "Region C",

"West", "Region D", "Other"

)

What you get from this is "Region A" for "North", "Region B" for "South", etc. If it does not find a value, that meets any of these, it returns "Other".

Use Cases:

• Mapping between values (e.g., codes to labels)./noopener

• Working as an alternative to long chains of IF statements.

• Categorization by regions, sizes, ratings or grade bands.

• Developing logic for dashboards or reports that is readable and maintainable.

SWITCH vs Nested IF: A Performance Comparison and the Case for Readability

Also referred to as a nested IF, you can use multiple conditions in several IF statements and act according to those conditions. But when you have multiple conditions, nested IF logic is difficult to read and maintain.

Example of Nested IF:

Category =

IF(Product[Price] > 1000, "Premium", IF(Product[Price] > 500, "Standard", OPP))))

IF(Product[Price] > 0, "Budget", "Unknown")

)

)

This will work but it starts to get very cumbersome when you have many branches.

Same approach with SWITCH and TRUE():

Category = SWITCH(TRUE(),

Product[Price] > 1000, "Premium", Product[Price] > 500, "Standard", Product[Price] > 0, "Budget","Unknown"

)

This is more clear compared to the previous version, and should be easier to maintain. It puts SWITCH(TRUE(), ...) to use, which lets you supply logical conditions just like IF but eliminates the deep nesting.

Performance Considerations:

• Performance could be an issue in nested IF statements on big models.

• SWITCH enables Power BI's DAX engine to apply efficient condition evaluation12.

• SWITCH will eliminate multiple evaluation of one expression making your code clearer and run faster.

6.4.3 RELATED Function: Getting Values from Related Tables Definition:

The RELATED function return values from another table in the data model, that share a relationship with the current table.

Syntax:

RELATED(TableName[ColumnName])

Example:

If the Sales table is related to the Products table on ProductID:
RELATED(Products[Category])

• This drags the category over from the Products table to the Sales table for each sale.

Use Cases:

• Pass through from look-up table some description data (e.g. Customer name, Product Category).

•Facts and dimensions in the star schema model.

6.4.4 Combining Logical and Aggregation Functions

In DAX using logical functions such as IF or SWITCH with aggregation functions like SUM, AVERAGE, SUMX allow you to create a various Calculation over the data and dynamically change which columns are being effected by the formula (or not at all). This technique is quite common in business logic to calculate scores, conditional totals and custom KPI calculations.

Key Concepts:

• Logical functions – Test conditions, return a TRUE or FALSE result.

• Aggregation Functions: Perform calculations (e.g., sum, average, count) across many lanes.

• You can apply them in tandem to do some calculations only when certain conditions are true.

Examples Recap:

Conditional Total Sales

TotalHighRevenue =

CALCULATE(SUM(Sales[Revenue]), Sales[Revenue] > 1000)

This measure computes the aggregate of Revenue for only those rows whose revenue amount is greater than 1000.

IF with Aggregation

Performance =

IF(AVERAGE(Sales[Revenue]) > 500, "Good Performance", "Needs Improvement") It calculates the average revenue and tags a performance according to this.

SWITCH with Aggregation

PerformanceRating =

SWITCH(TRUE(),

SUM(Sales[Revenue]) > 10000, "Excellent", SUM(Sales[Revenue]) > 5000, "Good", SUM(Sales[Revenue]) > 1000, "Average", Poor

)

In this example we are using SWITCH(TRUE() to compare total revenue against several thresholds and return a performance rating.

Expanded Example: IF and SUMX Business Scenario:

An e-commerce analysis needs to find the discounted revenue value for high volume orders; where only if the amount of quantity in a transaction is more than 5, it receives the discount. The effective revenue for each sale is Quantity Price Discount Rate, only on rows that satisfy the condition.

Sample Data:

OrderID	Quantity	Price	DiscountRate
101	3	100	0.95
102	6	200	0.90
103	8	150	0.85

Step-by-Step Calculation:

In the above, replace 5 with the measure you are using to determine if discount kicks in Recommendations: -Use an IF inside SUMX so that we apply the discount calculation when Quantity > 5 TotalDiscountedRevenue = Proud to be a Super User!

SUMX(

Sales,

IF(Sales[Quantity] > 5,

Sales[Quantity] Sales[Price] – 0 Sales[DiscountRate], 0

)

)

How it works:

• SUMX traverses line by line through the Sales table.

• It goes for each row in order, and then checks the Quantity > 5 part of the code.

o If so, then it computes the discounted revenue.

o If false, it returns 0.

• And finally, it tallies all of the results to arrive at the total.

Row-level Breakdown (based on data above):

- Order 101: Quantity = 3 → Fails condition → Returns 0
- Order 102: 6 × 200 × 0.90 = 1080
- Order 103: 8 × 150 × 0.85 = 1020
- Total = 1080 + 1020 = 2100

Knowledge Check 1

Choose the correct option:

Which of the following is true? Anser: A Calculated column 5) Which type of calculation best describes a calculated column?

a) the calculation for a stored value on a per-row basis of a table

b) An inference computed at query time only.

c) A pre-defined aggregation such as SUM or AVERAGE

d) Transient filter for use on a dataset

The behavior of the SUMX function is not the same as that of SUM because:

a) It works only with numeric columns

b) It takes expression one row at a time, evaluates it and then adds them together.

c) It is only function with relative tables.

d) It does not respect filters on the data set.

IF function is one of those familiar functions in DAX, yet I have seen several Dax beginner who still asks if there is a pre-built IF function in dax.

a) Grouping rows by category

b) Dealing with multiple conditions within the same formula

c)Glowing one value if a condition is True, and another value otherwise

d) Looping through the rows to add up standardUserDefaults on all keys

By filter context in DAX we mean:

a) The filters which are coming from dataset through slicers, visuals and DAX functions

b) the row by row operation on a table

c) Raising new tables automatically

d) Omission of missing data from a record.

6.5 Summary

 In this chapter, we explored the foundations of DAX (Data Analysis Expressions) in Power BI. DAX acts as the formula language that allows users to create new insights from existing data. We began with the importance of DAX, its syntax, data types, and the concept of evaluation contexts (row vs filter context).

 We then distinguished between calculated columns and measures, highlighting their use cases and differences in terms of storage, performance, and context. The discussion on aggregation functions (such as SUM, SUMX, COUNT, AVERAGE, MIN, and MAX) showed how DAX enables both simple and complex summarization of data.

 Logical functions like IF, SWITCH, and RELATED were also explained, illustrating how DAX supports decision-making logic and the integration of values across related tables. Finally, we examined how logical and aggregation functions can be combined to build more advanced business logic.

6.6 Key Terms

1. DAX (Data Analysis Expressions): A formula language for Power BI, Excel Power Pivot, and SSAS.

2. Calculated Column: A new column created with DAX that computes values row by row.

3. Measure: A dynamic calculation in DAX evaluated at query time, based on filter context.

4. Row Context: The context that applies to calculations at the row level.

5. Filter Context: The context created by filters, slicers, or DAX functions that affect which rows are considered in a calculation.

6. Aggregation Functions: Functions that summarize data (e.g., SUM, COUNT, AVERAGE).

7. Logical Functions: Functions that evaluate conditions and return results accordingly (e.g., IF, SWITCH).

8. RELATED: A DAX function that retrieves values from related tables.

6.7 Descriptive Questions

1. Explain the purpose of DAX in Power BI and describe its role in business intelligence.

2. Differentiate between row context and filter context with suitable examples.

3. What are calculated columns? Explain their use cases with examples.

4. Define measures in DAX. How do they differ from calculated columns?

5. Discuss the importance of aggregation functions in DAX. Compare SUM and SUMX with examples.

6. Write a DAX formula using IF that categorizes sales into “High” and “Low” groups.

7. How does the SWITCH function simplify multiple condition handling in DAX?

8. Explain the use of the RELATED function with an example from a sales-product model.

9. Why are measures generally preferred over calculated columns in large BI models?

10. Show how logical and aggregation functions can be combined to generate meaningful KPIs.

6.8 References

1. Microsoft Documentation: DAX Function Reference – https://learn.microsoft.com/power-bi/dax

2. Marco Russo & Alberto Ferrari (2020). The Definitive Guide to DAX: Business Intelligence for Microsoft Power BI, SQL Server Analysis Services, and Excel. Microsoft Press.

3. Gil Raviv (2018). Collect, Combine, and Transform Data Using Power Query in Excel and Power BI. Microsoft Press.

4. SQLBI Articles on DAX – https://www.sqlbi.com/articles/

5. Microsoft Power BI Community Forums – https://community.powerbi.com/

Answers to Knowledge Check

Knowledge Check 1

1. a – A stored value computed for each row of a table.

2. b – Evaluates an expression row by row before summing results.

3. c – Returns one value if true, another if false.

4. a – Filters applied via slicers, visuals, or DAX functions.

IBI_Unit 7_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127444740

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 3:59 PM GMT+5:30

File Name

IBI_Unit 7_V3.docx

File Size

195.8 KB

35 Pages

6,899 Words

41,482 Characters

0% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

0 Not Cited or Quoted 0%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **0** Not Cited or Quoted 0%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Unit 7: Advanced DAX & Time Intelligence

Learning Objectives

1. Explain the role of advanced DAX functions in solving complex analytical problems in Power BI.
2. Differentiate between basic and advanced DAX functions with relevant business examples.
3. Apply time intelligence functions such as YTD, MTD, QTD, SAMEPERIODLASTYEAR to perform period-over-period comparisons.
4. Utilize ranking, filtering, and iterator functions to answer business queries like “Top N” analysis and customer segmentation.
5. Optimize DAX formulas for performance by understanding query evaluation, avoiding inefficient patterns, and applying best practices.
6. Demonstrate the use of advanced DAX in real-world business cases, including profitability analysis, customer lifetime value, and regional performance tracking.

Content

7.0 Introductory Caselet

“Ravi’s Retail Dilemma: Applying DAX to Freeform Analysis and Insights”

Background:

Ravi is a data analyst at a rapidly growing chain of retail stores TrendyMart with presence in multiple cities in India. With the company growing so quickly, the executive team is increasingly leveraging data in order to help make operational and strategic decisions. We went through a demonstration of how to use Microsoft Power BI for interactive dashboards and reports. TrendyMart saw the value in going from raw transactional data to insights that can be used as actionable insights.

The company already measures basic metrics like sales, transactions and number of customers but Ravi was brought in to create more complex, sophisticated insights with DAX (Data Analysis Expressions). It is his responsibility to assist the management in knowing sales trend at different periods of time, customer purchasing pattern and profitability category wise.

One of the problems he has is to comparing YoY sales trends with filters by region and type of product. Also we need to calculate dynamically "The top 5 high-margin products" based on the selected time periods. To achieve these results, Ravi needs to move beyond simple aggregators and use a variety of DAX functions like CALCULATE, FILTER, ALL, DatesYTD, RANKX and REMOVEFILTERS.

By repeatedly solving problems, Ravi creates a portfolio of measurements that lets the business answer questions in real time with accuracy. It helps the company, plan better strategically but also allows his sales and marketing team the agility to respond to changes in the demand seasonality or marketplace conditions.

Critical Thinking Question:

As a data analyst, just like Ravi, what would you consider the tradeoff between model complexity and report performance when working with complex DAX? How would you deal with that to make sure the reports are insightful and still responsive as soon 1Gb grows?

7.1 Advanced DAX Functions

Complex calculations on data models in Power BI with Advanced DAX. These functions provide more than simple aggregations; they support dynamic, context-sensitive analytics. They are pretty important for creating KPIs, time intelligence reports and custom business logic.

7.1.1 Introducing complex measures in DAX

DAX complex measures are the formulas using a combination of functions, filters or modifiers to return a certain information from data Set.

They are destinated to adapt by themselves when the user interacts with one report, mainly filtered via slicers or visuals.

Complex measures differ from simple ones (e.g. SUM(Sales[Amount])) in that they calculate things like year-to-date totals, ranking, or metrics conditional on criteria.

For example, the business might ask "What are Sales for the last quarter but only premium customers and for North region."

We certainly can't achieve that with just a simple group by - we require logical functions, application of filters and measures that understand the context.

In DAX, it can be implemented on the fly by using complex functions such as CALCULATE, FILTER and ALL or with time intelligence actions.

7.1.2 CALCULATE Function: Definition of Syntax and Applications

The CALCULATE function in DAX is the magic key. It is commonly used to calculate the expression in a slighter modified filter context which allows analysts to change calculation condition.

Syntax

CALCULATE(, , ,...)

• : A DAX expression that returns a scalar value (most often, a measure or an aggregation such as SUM(Sales[Amount])).

• , , ...: One or more filters that will override the filter context. These expressions may be Boolean expressions, table expressions or filtering functions (e.g., ALL, FILTER).

Key Features

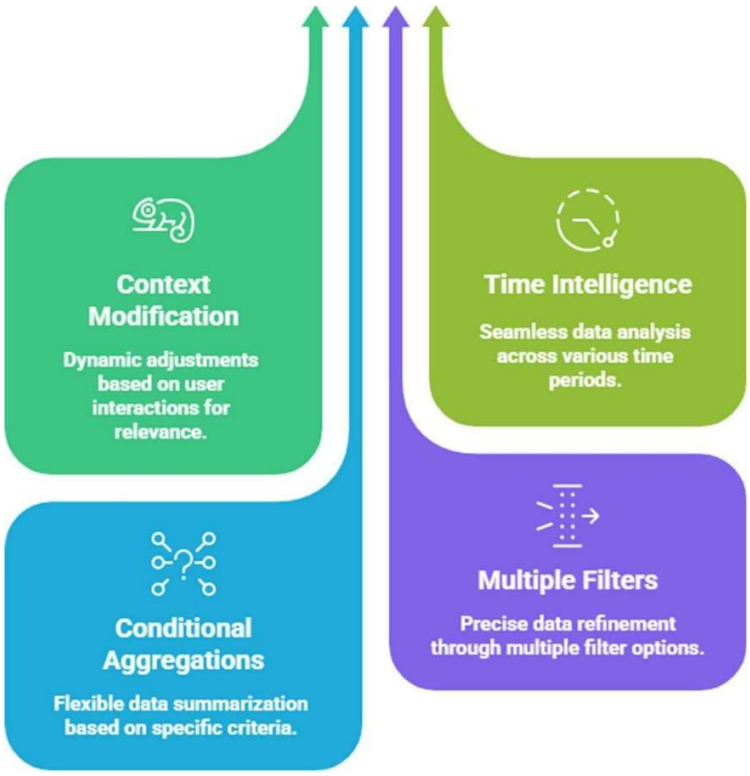


Figure: **Key Features**

Context Modification

CALCULATE transforms the filter context of the computation. This provide option to perform calculations withing the part of data model or conditions.

Integration with Time Intelligence

It is also used by time-based/calendrical calculations like YTD, QTD, MoM and indeedrolling averages.

Conditional Aggregations

Allows calculations at an industry/firm level of granularity using an arbitrary condition filter.

Supports Multiple Filters

More than one filter could be embedded in a CALCULATE statement allowing us to apply specific filters on the result with great flexibility.

Business Use Cases

Use Case 1: Regional Sales You work in sales, and you want to calculate the sales total for each region.

To implement the "North" region's overall sales: CALCULATE(SUM(Sales[Amount]), Region[Name] = "North")

Use Case: A sales manager wants to analyze Northern's performance while excluding the rest of the country.

Scenario 2 — Time Series – YTD Sales Let us look at another scenario using time-based statistics, which involve the 'YTD' and sales metrics.

CALCULATE(SUM(Sales[Amount]), DATESYTD(Date[Date]))

Scenario: Finance needs to see how total sales build up over the year, in order to measure performance relative to annual targets.

Scenario 3: Premium customer revenue Case Study 1 Here, however, premium pricing would write off a significant portion of customers—including those deemed to have the highest lifetime value.

CALCULATE(SUM(Sales[Amount]), Customer[Category] = "Premium")

Situation: You are running a test package and the marketing team is interested in seeing what Change in Evergreen Premium contribution we have made to overall revenue.

Use Case 4: Multi-Dimensional Filtering

CALCULATE(SUM(Sales[Amount]), Region[Name] = "East", Product[Type] = "Electronics")

Use Case: A Regional Product Manager wants to analyze all sales of electronic products in the Eastern region.

Scenario 5: Sales excluding Returns.

CALCULATE(SUM(Sales[Amount]), Sales[IsReturned] = FALSE)

Situation: Finance need to calculate Net Sales amount by eliminating the returns orders from Total Sales.

Comparison of Current Month Sales with Preceding Month(s) Example 6:

CALCULATE(SUM(Sales[Amount]), PARALLELPERIOD(Date[Date], -1, MONTH))

Example: The BI team is asked to create a MoM growth report for executive dashboards.

Use case 7: Dynamic segment analysis with user filters.

CALCULATE(SUM(Sales[Amount]), FILTER(Customer, Customer[Segment] = "Enterprise"))

For example: A self-service BI user creates ad hoc reports that filter on dynamic segment (slicer pick "Enterprise" customers).

7.1.3 Using FILTER with CALCULATE

Definition

The DAX FILTER function The aptly named FILTER function is actually an iterator, it iterates an internal calculated table resulting from some filtering condition applied to another table. It can be combined with CALCULATE to achieve more complex filter behavior than is typically possible using mere Boolean expressions and manipulate row context on-the-fly.

Syntax of FILTER

FILTER(,)

• : A table or table factor to be filtered.

• : A formdes which can be evaluated to TRUE or FALSE for each row. Rows whose condition is TRUE are retained.

Why Use FILTER with CALCULATE?

When to use FILTER with CALCULATEFILTER and CALCULATERequirement of FILTER....VALUES()FILTER – when used in a Data Model (DAX) expression, the function should not directly reference a column name, instead that column is to be defined using the ADDCOLUMNS() function as shown in example below.

- Row by row analysis is required.

• You are working with a lot of logical expressions.

• You want to ignore slicers or visual filters in your Power BI report.

• You have linked tables and you need to apply custom logic in filtering them.

Common Use Cases

Dynamic Row-Level Security Row Filtering CALCULATE(SUM(Sales[Amount]), FILTER(Sales, Sales[Quantity] > 10)) Securely retrieves the sum of sales for orders where quantity is greater than 10.

Multiple Logical Conditions

CALCULATE(

SUM(Sales[Amount]),

FILTER(Sales, Sales[Quantity] > 10 && Sales[Discount] <.05)

)

Sales returns having quantity greater than 10 and discount is less than 5%.

Overriding Context from Slicers / Visuals

FILTER discards the visual-level filter context and evaluates only the values of the expression directly, without reference to any explicit or implicit value of a slicer on the row.

Filtering on Related Tables

CALCULATE(

SUM(Sales[Amount]),

FILTER(Products, Products[Category] = "Furniture")

)

Filters the related Sales table by selecting information that only comes from Products, where ProductType is "Furniture".

Step-by-Step Worked Example Objective:

-- Calculate the sum of order sales with a quantity higher than 10 and discount lower than 5%.

Step 1: Sample Dataset

OrderID	Product	Quantity	Discount	Amount
1001	Laptop	5	0.10	50,000
1002	Monitor	15	0.03	30,000
1003	Keyboard	20	0.02	5,000
1004	Mouse	8	0.00	2,000
1005	Printer	12	0.06	18,000

Step 2: Define the DAX Expression

CALCULATE(

SUM(Sales[Amount]), FILTER(

Sales,

Sales[Quantity] > 10 && Sales[Discount] < 0.05

)

Step 3: Apply the Logic Row-by-Row

We evaluate the condition Quantity > 10 && Discount < 0.05:

OrderID	Quantity	Discount	Condition Met
1001	5	0.10	No
1002	15	0.03	Yes
1003	20	0.02	Yes
1004	8	0.00	No
1005	12	0.06	No

Rows 1002 and 1003 meet the condition.

Step 4: Calculate the Result

Sum the Total where rows match:

• Order 1002: 30,000

• Order 1003: 5,000

• Total = 35,000 Final Result:

CALCULATE(

SUM(Sales[Amount]), FILTER(

Sales,

Sales[Quantity] > 10 && Sales[Discount] <.05

)

) = 35,000

Performance Consideration

Although FILTER is excellent for row level filtering, its semantics are less efficient than simple expressions. So it should be used only sparingly, particularly on the large sets.

7.1.4 Context Transition and Row Context

Definition:

Context plays a great role in how DAX evaluation is performed. Context transition Context transition is the ability for relationships to automatically convert row context into filter context, usually when functions that require a filter are used (like CALCULATE) It's important to understand how "context" works when authoring DAX measures using Power BI.

Row Context

Row context occurs when a DAX expression is evaluated for each row of the input table, generally in the following situations:

• Calculated columns

• Iterative functions like SUMX, FILTER, ADDCOLUMNS For instance:

Sales[TotalCost] = Sales[Quantity] * Sales[UnitPrice]

Now the Quantity and UnitPrice each have their own row values, so that's why this measure works in a calculated column (it actually has row context).

Filter Context

Filter context When filters are applied to a table or visual, a filter context is created. This context can come from:

• Slicers or filters applied to the report

• The row and column headers on a matrix

• Filters that use DAX expressions directly (for example, within CALCULATE)

For instance, in a visual where we have SUM(Sales[Amount]) by Region, the Region is our filter context.

Context Transition

Context transition is simply when rows of context are converted to filter context so the calculations work off the current row.

This typically happens when:

A measure is being used in a row context (like withing SUMX)

• CALCULATE is called

Example:

CALCULATE(SUM(Sales[Amount]))

If used within a ROW or SUMX, CALCULATE will take the current row's cell value and place it in the filter context.

Illustrative Case:

SUMX(Customers, CALCULATE(SUM(Sales[Amount]))) In the above formula:

- SUMX is adding a row context and iterating the customer through each row

• CALCULATE changes that row context (e.g., one customer's) into a filter context, so it only sums sales for that customer

Why This Matters

• You may have totals when transitions of context are not taken into account or when filters won't work correctly.

• Understanding the row and filter context is also important when we write measures in DAX that utilise calculated columns, row level logic and aggregations.

Did You Know?

""Did you know that DAX automatically accelerates the Context Transition when measures are evaluated within a Row Context, through functions such as SUMX or FILTER? This covert move from rolling context to filtered context is the driver behind how complex calculations like customer wise sales rollups can work without thinking twice inside of CALCULATE. Without a transition like that, your measure could very well be returning wrong results or nothing at all."

7.2 Time Intelligence Functions

The time intelligence functions in DAX support calculations that can historically or to date for year events and natural language approach the syntax of the formulas encourage customers to understand analytical patterns over time such as a running balance of YTD sales, YTD comparisons even past years comparison shifting. These functions depend on a properly structured Date Table, and are heavily used in business reporting for trends, seasonality and growth analysis.

7.2.1 Understanding the Date Table in Power BI

A Date Table is nothing but a regular table with date in continuous range and additional columns like Year, Month, Quarter, Week.

It is the base date to Java time in all DAX Time intelligence calculations and it must behave correctly with time functions.

Data Table Requirements:

It need have only one column of unique, consecutive dates and no missing dates.

It needs to cover the full range of dates you want to analyze.

It has to be defined as a Date Table in Power BI (using "Mark as Date Table" option).

It should have a relationship with the fact table (e.g., Sales)., and be connected with date field one-way.

Example Columns in the Date Table:

• Date

• Year

• Month Name

• Month Number

• Quarter

• IsWeekend

• Fiscal Year

If you don't have a valid Date Table none of your time intelligence functions like TOTALYTD or SAMEPERIODLASTYEAR will work.

7.2.2 Year-to-Date (YTD) and Month-to-Date (MTD) Functions

TOTALYTD and TOTALMTD are the two DAX time intelligence functions that come built-in with the Power BI, used to calculate the YTD totals and MTD(Up to current day in filter context) totals from start of year or Month. They are particularly useful in business reporting where you need to compare how the data has evolved.

Function 1: TOTALYTD Syntax

TOTALYTD(, , [], [])

• : An aggregation (e.g., SUM(Sales[Amount]))

• : A column with dates (most likely 'Date'[Date])

• [] (optional): Additional filtering conditions

• [] (optional): If the fiscal year period is ending on a date other than Dec 31, this is used.

Example

Sales_YTD = TOTALYTD(SUM(Sales[Amount]), 'Date'[Date])

Use Case: Calculates total sales from the date of January 1 till a selected date.

Function 2: TOTALMTD Syntax

TOTALMTD(, , [])

• Like TOTALYTD but on the current month.

Example

Sales_MTD = TOTALMTD(SUM(Sales[Amount]), 'Date'[Date])

Use Case: Calculates aggregate sales from the 1st day of the current month to a chosen date.

Filter Context Behavior

Both functions use the current filter context even slicers, or visual filters or page filters.

Example:

If your report page has a slicer which selects April 15, 2024 then:

• Sales_YTD gets the sum from January 1 to April 15

• Sales_MTD will display the value from April 1 through April 15

Worked Examples of Business Scenarios: YTD Sales Example 1.

A retailer would like to monitor Year-to-Date (YTD) Sales so that they can ascertain whether cumulative sales are keeping up with those in the fiscal year. Our finance department referred a Power BI dashboard with a date slicer. They want a YTD measure that will actually respond to the selected date.

Sample Sales Data

Date	Sales Amount
Jan 5, 2024	5,000
Feb 10, 2024	8,000
Mar 20, 2024	12,000
Apr 5, 2024	10,000
Apr 15, 2024	6,000
May 1, 2024	7,500

DAX Measure

Sales_YTD = TOTALYTD(SUM(Sales[Amount]), 'Date'[Date])

Step-by-Step Calculation

1. Duration which the slicer is set: April 15, 2024

• TOTALYTD will aggregate sales from 1 January to 15 April

• Relevant lines: Jan 5, Feb 10, Mar 20, Apr 5, Apr 15

• Sum = 5,000 + 8,000 + 12,000 + 10,000 + 6,000 = 41,000

Final Result

Sales_YTD = 41,000

If the user selects a different date in the slicer, this value will update automatically.

"Exercise: Create and Compare YTD and MTD Sales Measures"

Instruction to Student:

You have been provided with a dataset which has the daily sales data of a retail company. Create a right Now, connect to the file you fixed, using Power BI.

Table and designate it as the date table. Then add two DAX measures:

YTD Sales using TOTALYTD()

MTD Sales using TOTALMTD()

Graph both on a line chart for the past 6 months and see if accumulation is doing Either way.

Task: Write a short reflection (200 words) Analyse the similarities and differences in the patterns. Explain when each measure would be most useful to use within business reporting.

Salaz, March..2:1) to date variables in calculations DAX TOTALQTD Function TOTALQTD (, , []) Summary Placed totals functions that yield dates are issued.

Use Case: Get the cumulative value from beginning of th e quarter to current date.

Sales_QTD = TOTALQTD(SUM(Sales[Amount]), 'Date'[Date])

Example Scenario:

So if the user selects May 10, and May is in Q2, then TOTALQTD will give you the total sales in q2 from April 1 to May 10.

Applications:

• Quarterly performance dashboards

• Cumulative metrics for business reviews

• Comparison vs forecast by quarter

7.2.4 SAMEPERIODLASTYEARand Parallel Period AnalysisOverview

In DAX, time-related functions are utilized such as SAMEPERIODLASTYEAR and PARALLELPERIOD to create comparative measures with respect to the previous periods. These functions enable businesses to compare current performance with equivalent periods in the past — like last year, last quarter or three months ago.

Function 1: SAMEPERIODLASTYEAR

Syntax

SAMEPERIODLASTYEAR()

- Returns a table of dates that are in the same period as the last year based on the current filter context.

Example Measure

Sales_LastYear = CALCULATE(SUM(Sales[Amount]), SAMEPERIODLASTYEAR('Date'[Date])

)

Use Case

If you filter the visual on March 2025, the results from SAMEPERIODLASTYEAR will return for March 2024. This makes life easy for an Year-over-Year (YoY) comparison.

Function 2: PARALLELPERIOD

Syntax

PARALLELPERIOD(, ,)

- : Column containing date values (eg 'Date'[Date])
- : A number specifying the period (negative: past, positive: future)
- : Should be one of MONTH, QUARTER or YEAR.

Example Measure

Sales_3MonthsAgo = CALCULATE(SUM(Sales[Amount]), PARALLELPERIOD('Date'[Date], -3, MONTH))

)

Use Case

Generates values for reporting on:

- “Sales three months ago”
- “Quarter-on-quarter comparisons”
- “Same quarter two years ago”

Key Differences

Feature	SAMEPERIODLASTYEAR	PARALLELPERIOD
Period Shift	Fixed to **1 year back**	Flexible (any number of months, quarters, or years)
Period Length Maintained	Yes	No
Supported Intervals	Only **1-year** offset	Month, Quarter, or Year
Common Use	Year-over-Year comparisons	Custom period shifts

Cons: A Date Table Must be Continuous It's Important to Mention :

Same as above but it applies for SimilarPeriodLastYear and PeriodsParallel. Need to make sure a Date Table is created which should be fully working without gaps throughout the time period within the SSAS Cube.

to function correctly.

They should take as arguments the date column to operate on:

• Contain no gaps in dates.

• Be drawn across the time scale required for analysis (e.g., years).

• Correctly appear as a Date Table in the model (as Mark as Date Table in Power BI).

If your date table is partial (or does not have every single day in it – only transaction dates for instance) then these functions will be returning the wrong result or blanks.

Did You Know?

"Did you know that SAMEPERIODLASTYEAR only functions as expected when you have a continuous series of dates in your date column (e.g. no gaps)? If the date table is do not include dates as missing days, such as weekends or holidays, then your comparison calculation might fail silently or produce partial result. This is why it is crucial to use the marked, continuous Date Table in time intelligence."

7.3 Performance Optimization with DAX

When dealing with large and complicated datasets, the performance of your DAX shines through in order to keep your reports fast and your models scalable.

This part concentrates on best patterns, mistakes not do and how to enhance and troubleshoot DAX formulas, where the same model is a big one.

7.3.1 Best Practices for Writing Efficient DAX

Good DAX means fast reporting and fewer computational efforts of the Power BI engine. Adhering to best practices is the key to creating models that are robust and capable of scaling with data.

Best Practices:

Use measures instead of calculated columns:

Measures: measures are computed at query time, which consumes small memory because of that; Calculated Columns - they are physically stored in the model (it bloats your file).

Avoid Repetition by Reusing Measures:

Do not duplicate logic: use the base measures within larger expressions. This improves maintainability and performance.

Total Sales = SUM(Sales[Amount])

Profit Margin = [Total Profit] / [Total Sales]

Avoid the Use of Nested CALCULATE or FILTER:

Complicated nested logic can drag down performance. Where you can, try factoring logic or moving filters to the model layer.

Pre-aggregate in Power Query:

Where feasible, it’s useful to do heavy transformation or summarization in Power Query prior the model receives data.

Taking Advantage of Variables (VAR) for Clean and Efficient DAX:

Some help improve readability so you're not recomputing things over and over.

VAR TotalCost = SUM(Sales[Cost])

VAR TotalSales = SUM(Sales[Amount])

RETURN DIVIDE(TotalSales - TotalCost, TotalSales)

7.3.2 Avoiding Common Pitfalls (Iterators, Row Context Misuse)

Most of the performance problems and mistakes in DAX can be attributed to ignorance about how context works and consequently bad usage of iterating function or calculated column. This section describes typical missteps, as well concrete examples of what to avoid and how to improve DAX code.

Key Pitfalls to Avoid

SUMX, AVERAGEX, etc.Common LispCommunityGtk-Sources 1.0 Overuse or Misuse of Iterators

Iterator functions such as SUMX and AVERAGEX compute the expression row by row, which may be inefficient for large dataset.

Misuse Example (Inefficient):

TotalRevenue = SUMX(Sales, [Quantity] * Sales[UnitPrice])

• This formula re-generates the revenue one row at a time, whenever we have an precomputed column (such as Sales [Amount]).

It consumes more memory and makes the model slower.

Optimized Alternative:

TotalRevenue = SUM(Sales[Amount])

• Assumes Sales[Amount] is a calculated column from loading/ETL.

• Aggregates precomputed values, improving performance.

Another Misuse Example (Redundant Use):

AverageQuantity = AVERAGEX(Sales, Sales[Quantity])

• This is logically equivalent to: AverageQuantity= AVERAGE(Sales[Quantity])

• Unless there is fancy row-wise logic to be performed, we favor built-in aggregation functions.

Misunderstanding Row Context vs Filter Context A common mistake is to make DAX calculations evaluate the row context instead of being able to switch from one evaluation context to another.

One common mistake is to expect that the row context apply filtering automatically, especially when working with related tables. This leads to incorrect results.

Misuse Example:

CustomerSales = SUM(Sales[Amount])

• This expression does not return sales by customer when used in a calculated column in the Customer table.

• It does not consider the association between Customer and Sales due to missing filter context.

Corrected Version Using CALCULATE:

CustomerSales = CALCULATE(SUM(Sales[Amount]))

• In a column definition in the Customer table, this establishes the correct filter context with respect to what customer record you are looking at.

Explanation:

• Row Context has access to current row values and it exists in both calculated columns and iterator functions.

• Aggregation over related tables needs Filter Context to be provided.

• CALCULATE changes row context into filter context for proper cross-table evaluation.

Calculated Columns where Measures would do the job

By employing DAX calculated columns to materialize intermediate results which are only requested by visuals, the memory used and model size often grow larger than necessary.

Misuse Example:

Profit = Sales[Amount] - Sales[Cost]

- This value stored as a calculated column take up memory in each row of the Sales table.

Better Alternative Using a Measure:

Profit = SUM(Sales[Amount]) - SUM(Sales[Cost])

• Measures are computed at query time and they do not take space in the data model.

Improper Use of EARLIER()

EARLIER() is employed in calculated columns to reference a value from an outer row context (e.g. from above the current row), however, it's frequently overused in complex nested calculations which create confusing and error prone code.

Misuse Example:

Rank = CALCULATE(

COUNTROWS(Sales),

FILTER(Sales, Sales[Amount] > EARLIER(Sales[Amount]))

)

- Hard to debug/maintain, especially with many levels of nesting.

Recommended Alternative Using VAR:

Rank =

VAR CurrentAmount = Sales[Amount] RETURN

CALCULATE(COUNTROWS(Sales),

FILTER(Sales, Sales[Amount] > CurrentAmount)

• The readability is enhanced, and the problem of nested context is avoided.

7.3.3 Optimizing Measures for Large Datasets

DAX expressions can be slow when you have large amount of data, even for simple DAX calculations. Reports scale nicely with optimization techniques.

Techniques:

Minimize Cardinality:

If columns have too many distinct value, please decrease the numbers of unique values. Like rounding currency to the nearest lower decimal or grouping dates by month.

Avoid Using DISTINCT and VALUES:

These can produce intermediate tables that are very large in memory. Replace it with any alternative logic or pre-aggregated data, if possible.

Filter Early, Not Late:

Use filters as early as possible in expressions, especially inside CALCULATE.

Optimized:

CALCULATE(SUM(Sales[Amount]), Sales[Region] = "North")

Replace VALUES by SELECTEDVALUE Where It Makes Sense:

SELECTEDVALUE(Customer[Segment], "All Segments")

is faster than a value exists in VALUE_VALUE table.

Employ Aggregated Tables for High-Level KPIs:

An alternative for very large datasets is to create pre-aggregated summary tables using Power Query or SQL and then use such created table while querying visuals.

Did You Know?

“Were you aware that high cardinality columns (e.g., a large number of unique values) are performance killers in DAX queries used as filter or inside visuals? That is, by substituting CustomerName instead of CustomerSegment you may experience increased Memory and time to compute the visual. One of the easiest and most efficient design patterns to optimize for large Power BI models is by reducing cardinality.

7.3.4 Debugging and Testing DAX Formulas

Testing and troubleshooting is an essential step to guarantee that DAX expressions always return the correct and expected results.

Techniques and Tools:

DEFINE MEASURE in DAX Studio is your friend:

Testing measures in isolation with performance traces can be done using DAX Studio. You can look at the query plan and memory usage.

Leverage RETURN with Intermediate Variables:

VAR TotalSales = SUM(Sales[Amount])

RETURN TotalSales

Use this for incremental logic elements.

Apply FORMAT() for Output Inspection:

During debug, cast your output to FORMAT() this way you can control precision and see what is going easily.

Utilize IF, ISBLANK and ERROR Checks:

Add logic to check, and respond to, the unexepected.

IF(ISBLANK([Total Sales]), 0, [Total Sales])

Tool Recommendations:

o DAX Studio: For performance tuning and tracing queries.

o Tabular Editor – Version control and clean, reusable measures.

o Performance Analyzer (in Power BI Desktop): In order to watch the load times on visuals and track down slow measures.

7.4 Business Use Cases using Complicated Formulation

I'm trying to highlight the fact that Power BI and DAX is not just some fictional technical jargon, but both age and intelligence are justified by them Business Analysts can spend days answered strategic questions with these advanced DAX formula whilst also showing trends of certain products and even make a business decision based on what the data is telling them.

This section will cover important use cases - such as sales growth, customer retention and financial forecasting among others – through practical examples in DAX.

7.4.1 Sales Growth Analysis

Objective:

To monitor and compare sales volume including both absolute and percentage increases.

Key Measures:

Current Period Sales:

Sales_Current = SUM(Sales[Amount])

Sales in the Preceding Period (e.g. Last Year):

Sales_PreviousYear = CALCULATE(SUM(Sales[Amount]),

SAMEPERIODLASTYEAR('Date'[Date]))

Sales Growth (Absolute):

Sales_Growth = [Sales_Current] - [Sales_PreviousYear]

Sales Growth (%):

Sales_GrowthPct = DIVIDE([Sales_Growth], [Sales_PreviousYear])

Business Impact:

These metrics also help reveal best-performing regions and seasonality, allowing both marketing and operational planning.

7.4.2 Customer Retention and Churn Measures

To know well is a business is multiplying its customer base with time and identify the churning trends in advance to target that customers on right time and take necessary actions such as loyalty campaigns, win-back campaigns or enhance the lifecycle of a customer.

Key Concepts

Term	Definition
Retained Customers	Customers who made purchases in **both** the current and previous periods.
Churned Customers	Customers who made purchases in the **previous period**, but **not** in the current one.
New Customers	Customers who made purchases for the **first time** in the current period.

Sample DAX Measures

1. Retained Customers

Retained_Customers = CALCULATE(

DISTINCTCOUNT(Sales[CustomerID]), FILTER(

```
ALL('Date'),

'Date'[Date] >= MIN('Date'[Date]) &&

'Date'[Date] <= MAX('Date'[Date]) && Sales[CustomerID] IN VALUES(PriorSales[CustomerID])
)

)
```

2. Churn Rate (%)

```
ChurnRate =

DIVIDE([Churned Customers], [Customers Last Period])
```

3. Retention Rate (%)

```
RetentionRate =

DIVIDE([Retained Customers], [Customers Last Period])
```

Note: [Customers Last Period] and [Churned Customers] are assumed to be separate supporting measures, defined using filters on prior time periods.

Worked Example: Dataset-Based Churn Analysis Scenario

A subscription-based e-commerce platform wants to evaluate monthly customer retention and churn to improve engagement strategies. Below is a simplified transaction log.

Sample Sales Data

C001 Jan 2025

C002 Jan 2025

C003 Jan 2025

C002 Feb 2025

C003 Feb 2025

C004 Feb 2025

C003 Mar 2025

C005 Mar 2025

Step-by-Step Analysis: February 2025

• THE PRESENT: FEB 2025 CUSTOMERS: C002, C003, C004

• Last period = Jan 2025Customers: C001, C002, C003

Step 1: Retained Customers

Users who appeared in Jan and Feb: C002, C003

→ Retained = 2

Step 2: Churned Customers

Customer in Jan but not Feb: C001

→ Churned = 1

Step 3: New Customers

Feb-only customers and not the Jan: C004

→ New = 1

Step 4: Customers Last Period

Customers in Jan: C001, C002, C003

→ Total Previous Period = 3 Calculated Rates

• Retention Rate = 2 / 3 = 66.67%

• Churn Rate = 1 / 3 = 33.33%

Visualization Summary (February 2025)

Metric	Value
Customers Last Month	3
Retained Customers	2
Churned Customers	1
New Customers	1
Retention Rate	66.67%
Churn Rate	33.33%

Business Impact

The following are the most important customer retention metrics you can use to gain key insights:

• Customer Lifetime Value (CLV) forecasting

• Marketing efficiency and segmentation strategy

• Customer health scoring for predictive analytics

• Drive reduction in CPAs and focus on loyalty and re-engagement

“Exercise: Compute and Inspect 2-Year Customer Retention”

Instruction to Student:

When you have the sales transaction table with customer IDs and dates, then do this in Power BI:

Create two DAX measures:

o One for those customers who have purchased this year

o One for people who bought from me last year

Add a third DAX measure for the retained customers, using INTERSECT().

Calculate the retention rate using DIVIDE().

Present these indicators in the card visuals and trend line (when applicable).

Task: Write a brief report (no more than 300 words) explaining what you think is happening. Respond to which season retention decreases or increases and why that changes.

7.4.3 Objective of Financial Forecasting and Variance Analysis

This is used to compare the real financial performance to forecasted or budgeted results, analyze how much in variance (deviation) and what can be done considering this, predict the future outcomes by analytic methods based on information from past. These methods are fundamental for planning, resource allocation, and financial control in contemporary business organizations.

Common Components

Component	Description
Actuals	Recorded financial results (e.g., sales, expenses, revenue).
Forecasts	Predicted values based on historical trends or business inputs.
Budgets	Planned targets set at the beginning of the fiscal year or quarter.
Variance	The difference between actual and forecasted/budgeted values.
Trend	Analysis of values over time to observe seasonality or directional shifts.

Key DAX Measures

Variance (Absolute)

Variance = [actual revenue] - [expected revenue]

Variance (%)

VariancePct = DIVIDE([Variance], [Forecast Revenue])

Rolling Forecast (Next 3 Months)

RollingForecast =

CALCULATE(

SUM(Forecast[Amount]), DATESINPERIOD('Date'[Date], TODAY(), 3, MONTH)

)

Forecasting Methods in Power BI

Manual and Automated Forecasts in Power BI You can perform both manual and automated forecasts with Power BI. All of which are used for different business types, availability of data, and forecasting horizon.

A. Manual Forecasting Using Forecast Tables

Business units will often come in and dump a forecast or budget into specialized tables. Those are then related to the main date table for compatibility.

Example Table Structure:

Date	Forecast Amount	Budget Amount
Jan 2025	120,000	115,000
Feb 2025	130,000	125,000
Mar 2025	150,000	145,000

These tables are only meant to create comparisons next to actuals with measures in DAX such as:
ActualVsForecast = [Actual Sales] - [Forecast Amount]

This method offers the most flexibility and control of our projection assumptions.

B. Automated Forecast via Built-In Analytics (Power BI Line Chart Forecast)

You can use the built-in time series forecasting tool in Power BI to quickly and easily forecast your line charts.

Key Features:

• Utilizes exponential smoothing algorithms.

• Predicts future values of historical data based on the trends.

• Supports confidence intervals (e.g., 95%).

How to Enable:

Use a line chart visual.

Drag a time-based field (for example, 'Date') to the Axis.

Drag a measure (for example, Revenue) to the Values.

In the Analytics pane, add a Forecast.

Define the time window length (3 months, for instance), confidence interval and seasonality.

Use Case Example:

• Predict the revenue for the next 6 months from last years monthly sales.

Limitation:

• This feature is available only for visualization, and it cannot be used in DAX expressions and measures.

• It assumes the time is evenly spaced and so it works best with daily or monthly data.

C. Custom Predictive Forecasting Using DAX and Historical Averages

You could just develop your own predictive model based on previous trends, like moving averages and growth rate, or.

year-over-year trends.

Example: 3-Month Moving Average Forecast

MovingAvgForecast = AVERAGEX(

DATESINPERIOD('Date'[Date], TODAY(), -3, MONTH),

[Actual Revenue]

)

Use Case: I need to forecast some numbers but we don't have a proper forecast yet so would like an estimate using historical data.

Example Variance Analysis Table

Month	Actual	Forecast	Variance	Variance (%)
Jan 2025	125,000	120,000	+5,000	+4.17%
Feb 2025	135,000	130,000	+5,000	+3.85%
Mar 2025	140,000	150,000	-10,000	-6.67%

This breakdown allows executives to:

- Identify over- or under-performance.
- Adjust future forecasts and budgets.
- Investigate root causes for deviations.

This breakdown allows executives to:

• Identify over- or under-performance.

• Adjust future forecasts and budgets.

• Investigate root causes for deviations.

Business Impact

Forecasting and variance analysis enable:

• Proactive financial planning

• Performance benchmarking against strategic goals

• Early warning mechanisms for cost over runs or underperformance.

• The ability to make agile decisions in a fast-moving market.

These are the insights that form the basis for CFO dashboards, quarterly board reports and annual planning cycles.

7.4.4 Comparative Analysis Across Time Periods

Objective:

To compare how a business has performed in different time frames (say, month-to-month, or for a quarter or year) looking for trends and strategic changes.

Types of Comparisons:

• Previous year's same period through SAMEPERIODLASTYEAR()

• Previous period using PARALLELPERIOD()

• Dynamic comparisons using DATEADD()

Sample DAX Measures:

Month-over-Month Growth:

Sales_LastMonth = CALCULATE(SUM(Sales[Amount]),

PARALLELPERIOD('Date'[Date], -1, MONTH))

MoM_Growth = [Sales_Current] - [Sales_LastMonth]

Dynamic Period Comparison (eg, with custom offset):

Sales_6MonthsAgo = CALCULATE(SUM(Sales[Amount]),

DATEADD('Date'[Date], -6, MONTH))

% Change Over Time:

GrowthPct =

DIVIDE([Sales_Current] - [Sales_LastPeriod], [Sales_LastPeriod])

Business Impact:

These KPIs enable to recognize seasonal behavior and long-term trends or performance declines that facilitate in strategic decisions and resource distribution.

Knowledge Check 1

Choose the correct option:

What is the most important DAX function for altering filter context of a calculation?

A) SUMX

B) FILTER

C) CALCULATE

D) VALUES

What formula calculates ytd totals from january to the selected date?

A) TOTALMTD

B) TOTALYTD

C) DATEADD

D) SAMEPERIODLASTYEAR

What will be the behavior if I use a row context in the CALCULATE?

A) The function returns blank.

B) It causes an error.

C) A row context is turned into a filter context. 1.4 Filters notionally to not affect the Data Model.

D) No calculation is performed.

Which XLCubed function is used to compare sales for the current month with that in the same month last year?

A) PARALLELPERIOD

B) SAMEPERIODLASTYEAR

C) EARLIER

D) RANKX

What's the standard in such cases as for performance optimization concerning large data sets?

A) Use many calculated columns

B) Avoid using variables

C) Minimize column cardinality

D) Always use nested FILTER functions these cases

7.5 Summary

It was a blast to learn more about Advanced DAX (Data Analysis Expressions), the powerful language in Power BI for creating custom, contextual analytics.

• In 7.1 you've learned about some advanced DAX functions like CALCULATE, FILTER and context transitions - fundamental principles for creating your own measures.

• 7.2 added Time Intelligence functions allowing YTD, MTD and comparison over time with a good date table.

• 7.3 - Performance Optimization We covered how to optimize for performance, design patterns and best practices, do's and don'ts Low down on tools for writing good DAX in larger models.

• 7.4 featured real-life business examples such as sales increase, customer attrition analysis, financial planning and period comparison.

ϖ These are the basic skills for all analysts working with data models, dashboards and enterprise reporting in Power BI.

7.6 Key Terms

DAX - A formula language, Power BI, Power Pivot and SSAS data modeling.

Measure - A live calculation that is computed at query time according to the context.

CALCULATE() – It changes the filter context of the expression.

FILTER() - Filters a table for a condition; similar to the WHERE clause in an SQL statement, and is used in CALCULATE.

Row Context – Context that being active when a formula is evaluated row by row.

Filter Context - The context generated by filters applied on visuals, slicer or DAX expressions.

Context Transition – Transformation of row context into the filter context, in particular within CALCULATE.

TOTALYTD(), MTD, QTD - Cumulative X pots that we saw in the last two examples.

SAMEPERIODLASTYEAR() – Returns the date period as the previous year!

PARALLELPERIOD() - Provides a parallel period date with the specified shift (back one year, for example).

Retention Rate - The percentage of customers retained over time.

Variance Analysis - This is when actual values are compared to projected or budgeted values.

7.7 Descriptive Questions

What is the distinction between row and filter context in DAX? Give an instance of a context switch.

Explain the syntax and how to use CALCULATE () functions? What is the diffirent between simple aggregations?

What is Date Table and its use in time intelligence functions? Why do I have to label it in Power BI?

Write a DAX expression to calculate YTD Sales and describe the solution parts.

3 Talk about the most common mistakes in DAX performance tuning.

How does SAMEPERIODLASTYEAR(), PARALLELPERIOD() work in comparative sales analysis?

Explain business scenario where you can use Customer Churn Analysis with DAX?

How do you optimize a DAX measure for large dataset?

Describe how VAR is used in DAX and how it can affect performance and readability.

Calculated Columns versus Measures in DAX This article describes how customer's data can be used to perform different types of calculations by using simple expressions that are comparable to the Excel formulas. When should each be used?

7.8 References

1. Microsoft Learn. (2023). DAX in Power BI. Retrieved from: https://learn.microsoft.com/en-us/power- bi/transform-model/desktop-dax-overview

2. Ferrari, A., & Russo, M. (2021). The Definitive Guide to DAX: Business Intelligence with Microsoft Excel, SQL Server Analysis Services, and Power BI. 2nd Edition.

3. SQLBI.com – Technical Articles and Blog by Marco Russo and Alberto Ferrari

4. Power BI Community Documentation. Retrieved from: https://community.powerbi.com

5. DataCamp & Coursera. (2022). Power BI DAX for Data Modeling and Analysis – Online Course Material

Answers to Knowledge Check

Knowledge Check 1

1. C) CALCULATE
2. B) TOTALYTD
3. C) The row context is converted to a filter context
4. B) SAMEPERIODLASTYEAR
5. C) Minimize column cardinality

7.9 Case Study

Strategic Business Intelligence at Nova Retail Corp

Introduction

The growing imperative for real-time insights and intelligence As companies expand in both geography and customer channels, the import of real-time knowledge becomes

increasingly essential to decision making. Nova Retail Corp, a mid-sized consumer electronics firm based in India jouey of managing business with data Nova had growing issues on tracking performance of the product across various regions and managing churn along with aligning sales made to forecasted budget.

The company had implemented Power BI for basic dashboards, but decision makers wanted deeper insights— to track Year-over-Year (YoY) sales growth and customer retention, perform variance analysis, compare performance across time periods.

This caselet looks at how advanced DAX formulas were incorporated into Nova's reporting mechanism. It covers common challenges with business intelligence and how to solve them without deviating from the best practices for DAX, including performance optimization of calculated columns and measures, using pattern-based calculations, enabling advanced Time Intelligence functions in Power BI desktop.

Background

The mission was simple – turn static sales reports into dynamic, smart dashboards with Nova's team of reporters at the helm under the guidance of a senior business analyst. Executive Leadership: The Business Another major obstacle the executive leadership team faced was: While Power BI visuals existed, these executives found it challenging to:

• No comparative benchmarks (e.g. last year/quarter)

• Lack of insight on sales trends in the past

• Cannot measure churned vs kept customers

• Manual comparison of plan vs. actuals

The analytics team quickly learned that advanced DAX functions, including CALCULATE, FILTER, SAMEPERIODLASTYEAR and TOTALYTD were needed to develop business-focused metrics that adjusted dynamically in the presence of slicers and filters.

Problem 1: You can't keep track of year over year, regional sales growths.

Nova's management had to take actual sales for the year in question and compare that amount with sales of the previous year. But regular DAX measures could not have this historical comparison in a visual and interactive way.

Solution:

The same thing the analyst did using SAMEPERIODLASTYEAR for dynamic year over year comparisons. This allowed executives to view which businesses were growing YoY with Region and Product Category filters.

Key Measure:

YoY Sales Growth % =

DIVIDE([Total Sales]- CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Date'[Date])), CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Date'[Date])))

Problem 2: Absence of transparency in customer retention and churn

There was no way for Nova to measure how many customers came back versus dropped off along the way.

Solution:

With dax and row context filters, the analyst constructed the all-time customer list as a base to compare customers over various date ranges for retained/churn/new.

Key Formula:

Retained Customers = CALCULATE(DISTINCTCOUNT(Sales[CustomerID]),

INTERSECT(

VALUES(Sales[CustomerID]), CALCULATETABLE(VALUES(Sales[CustomerID]),

DATEADD('Date'[Date], -1, YEAR))))

This enabled marketing and CRM teams to initiate focused win-back campaigns.

Problem 3: Variance in Forecast vs Actual Needed to be Monitored Manually

For department heads they required analysis of variances being delivered between the forecast sales and actuals by month and quarter.

Solution:

By bringing in some forecasting data and connecting it to the date table they had a nice set of dynamic variances with DIVIDE, CALCULATE, TOTALQTD etc.

Key Measure:

Sales Variance % =

DIVIDE([Actual Sales] - [Forecast Sales], [Forecast Sales])

Graphics were now showing over and under performing areas and products in real time.

PS4: Inadequate cross-time comparisons The third problem I mentioned was that it is naive to assume that people can be seen coming over the horizon.

Reports didn’t have the flexibility of comparing MoM/QoQ/rolling windows.

Solution:

The team utilized PARALLELPERIOD and DATEADD to build relative time measures.

Example:

MoM Sales Growth = [Sales of the Current Month] -

CALCULATE ([Current Month Sales], PARALLELPERIOD('Date'[Date],-1,MONTH))

Decision makers were now able to see KPIs trending over time in increments such as one month and so becoming more agile.

MCQ:

How can I calculate YoY (Year over Year) Sales Growth in Power BI using DAX?

A) SUMX with direct subtraction

B) SAMEPERIODLASTYEAR within CALCULATE

C) Generate with lagged values as calculated columns

D) Use LOOKUPVALUE to get last year's values

Answer: B) SAMEPERIODLASTYEAR within CALCULATE

Explanation: By using SAMEPERIODLASTYEAR we are sliding the date range by exactly one year and then comparing it to a filter of current-period.

Conclusion

Leveraging sophisticated DAX measures, Nova Retail Corp turned their dashboards from raw visuals to a decision support system. With time intelligence, customer segmentation, variance analysis and optimized formulas, Nova was able to enable its teams to analyze trends, react to market changes and better plan.

The successful transformation is a proof that not only learning the DAX functions for technical analytics is a key, but it's essential to deliver every day value thorough intelligence and automation.

IBI_Unit 8_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127444738

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:00 PM GMT+5:30

File Name

IBI_Unit 8_V3.docx

File Size

327.0 KB

16 Pages

3,433 Words

20,471 Characters

1% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

2 Not Cited or Quoted 1%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **2** Not Cited or Quoted 1%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Internet

www.coursehero.com <1%

2 Submitted works

Manipal University Jaipur Online on 2025-07-08 <1%

Unit 8: Visual Analytics & Report Development

Learning Objectives

1. Explain the core principles of effective data visualization for business reporting.
2. Distinguish between standard and custom visuals available in Power BI.
3. Apply appropriate chart types to represent different kinds of data insights.
4. Incorporate interactivity features such as slicers, filters, and drill-throughs into reports.
5. Use conditional formatting to highlight patterns, outliers, and key performance indicators.
6. Design storytelling dashboards that guide users through business narratives.
7. Evaluate visual design choices for clarity, accessibility, and decision support.

Content

8.0 Introductory Caselet

“Aarav’s Dashboard Dilemma: Sharing Insights Beyond Numbers”

Background:

Aarav is a junior business analyst at a logistics firm serving the entire country of India. Recently he has been asked to provide a performance dashboard each month to the executive team. While he can use Power BI to pull data and make charts, his initial report draft turns out confusing. Executives see the visuals as cluttered, find the color scheme inconsistent and have a hard time interpreting the insights.

His dashboard contains several different graphs displaying things like how many shipments are being made; problems with delivery time, fuel costs and customer complaints—and the executives can't tell him what it all means: "what are central performance trends and what matters process wise?"

Aarav quickly learns that good visuals are not about adding decoration but are instead about enhancing clarity, focus and storytelling. He is mentored by a senior data designer into the basic tenets of good visualization, such as selecting the right chart to fit the message, using color with meaning, reducing cognitive burden and visual consistency.

Using these techniques, Aarav refactors the dashboard, grouping KPIs in a natural way, drawing trends with lines on charts but protects to highlight critical areas and even adds tooltips and drill-downs for some interactivity. The new dashboard informs to the leadership for their strategy review meeting.

Critical Thinking Question:

How would you address complexity versus thoroughness in designing the report, based on your experience if you were Aarav? How can we structure the visuals both for explorative and explanatory purposes, from diverse perspectives of stakeholders?

8.1 Principles of Effective Visualization

Overview:

Good visualization is not just about prettying up reports; it's about clearly and efficiently communicating data to support decision making. Its about the translation of raw numbers into stories, using design and cognitive principles.

Key Principles

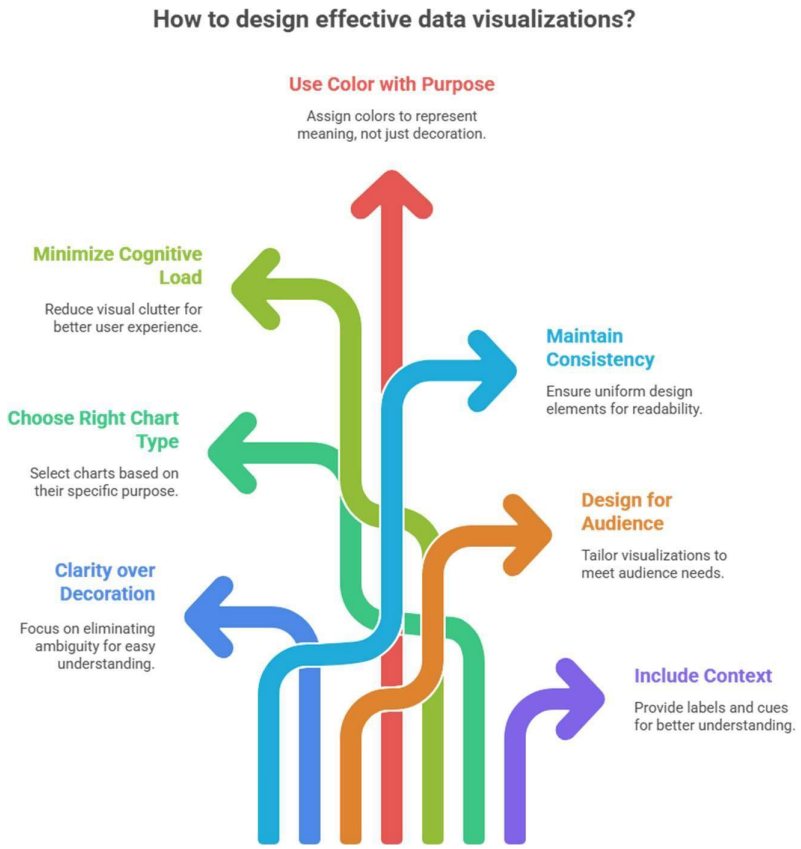


figure: **Key Principles**

Clarity over Decoration

Don't cram in extra graphics or 3-D effects. Data visualisations should make sense and be immediately comprehensible to the users.

Choose the Right Chart Type

Each visual does a different job:

2

o Use line charts for trends through time

o Use bar/column charts for comparisons

o Use pie/donut charts sparingly, for describing parts of a whole

o Use scatter plots for correlations

Minimize Cognitive Load

Keep report pages light on visuals. Don't overwhelm users with filters, colors and metrics. Group related information logically.

Use Color with Purpose

Colors should represent meaning:

o Red for dangers or negative values

o Green for goals or success

o Neutral background for supporting information Pretend colors do not exist.

Maintain Consistency

Maintain a uniform font style, layout and scale in visuals. 2. Consistency makes the code easier to read and less prone to misinterpretation.

Design for Your Audience

Executives want top-level guidance, but analysts may like to have deep-dive information. Customize the complexity of your dashboard and the patron interactivity to whom you are targeting.

Include Context and Labels

Give names to your axes, legends and data points wherever required. Give some sort of context, such as how it compares to targets or earlier periods.

8.1.1 Importance of Visual Storytelling in BI

BI is not just a numbers game, it's also about communicating discoveries. Visual storytelling is the sharing or communicating of a narrative with visual even moving graphic images. Rather than present stakeholders with raw data or isolated charts, a well-documented report can help guide an audience through the stages of understanding from problem to resolution.

• Storytelling guarantees that everything shown has a purpose to it: every chart is there to answer the business question.

• A story makes it easy to connect the dots between data, visuals and decisions—e.g., sales are down → thereby customer churn → which brings out regions for remediation.

• By mixing visual and contextual cues (titles, annotation, KPIs), storytelling empowers decision makers to interpret insights rapidly, and make decisions just as fast.

In BI, storytelling changes dashboard from a static report to a decision support system.

Did You Know?

“Human brains process visuals 60,000 times faster than text and more than 80 percent of the information we retain is visual.” That’s why dashboards that apply storytelling methodologies are much more memorable and actionable than text-heavy reports.

8.1.2 Choosing the Right Chart for the Right Data

Selecting the Appropriate Chart for the Appropriate Data

Choosing the right visualization type is key to good reporting. Each block serves a different analytical question:

• Change over time: Line and area charts

• Category comparisons: Bar charts, also known as column charts

•Part-to-whole relationships: pie charts, donut charts (use judiciously, use stacked bar/column chart)

• Describing values: Distribution of the values (Histograms, box-plots)

• Relationships and correlations: Scatter plot, bubble chart

• Hierarchies or breakdowns: Treemaps, waterfall charts etc.

For instance, if a pie chart were used to visualize sales over several years it wouldn't add up for the audience but with a line graph they could visually see trends up or down. As a rule it should be fit the chart to the question, not vice versa.

8.1.3 Design Principles: Clarity, Simplicity, and Relevance

There are three basic principles of effective visualization design:

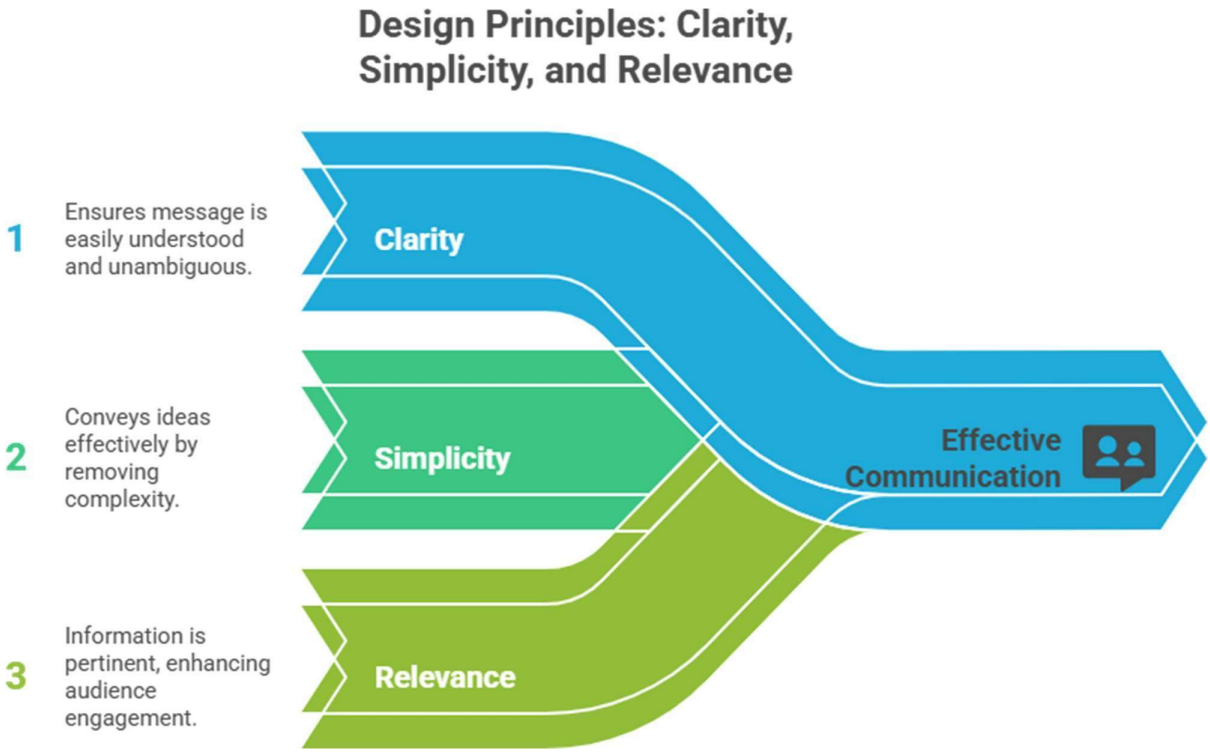


figure: Design Principles: Clarity, Simplicity, and Relevance

Clarity – The meaning of the chart should be obvious at the first glance. Axis labels, titles, and legends should be unambiguous. Keep illustrative extras to a minimum, ones that are taken for granted and which do not distract from the main idea.

Simplicity – Less is more. Keep visuals to a minimum per page, steer clear of gratuitous effects like 3D or gradients and tamp down that color overload. Simple design improves readability.

Pertinence – All contents of the dashboard should have a business reason to be included. KPIs, charts and filters should be driven by the objectives of the report's audience.

This will make sure that dashboards are actionable rather than decorative.

8.1.4 Avoiding Common Visualization Mistakes

Even expert analysts are susceptible to visualization traps. Some common mistakes include:

• Overwhelming dashboards with too many visuals - Overloading users with information and making it difficult to focus on the most important insights.

• Incompatible scales and axes – The alignment of the various axes among the charts can mislead interpretation.

• Ineffective color use: Too many colors and mixed schemes can make copy hard to read or detract from the overall message.

• Inappropriate chart type – (e.g. a Pie chart with 10 categories is unreadable.)

Context wanting – reporting figures without benchmarks/targets or priors renders the data useless.

Avoiding such mistakes requires discipline — always test your visuals by asking, "Does this chart give a clear and truthful answer to the business question?"

8.2 Standard and Custom Visuals

In Power BI, visuals are the primary means of expressing insights. You can use built-in standard chart types, as well as batty advanced or custom visuals available from the Power BI marketplace. Each performs a different type of analysis that allows users to explore, compare, and interpret data efficiently.

8.2.1 Standard Charts (Bar, Line, Pie, Column)

• Are used to compare magnitudes of categories.

• Vertical bar charts are more suitable for long category names, whereas horizontal column charts are suggested with short categories.

• Example: Monthly sales comparisons of various product categories.

Line Charts

• Good for comparing trends over time.

• Multi-series can be displayed to enable trend comparison.

• Example: Monitoring increase in revenue over the years.

Pie and Donut Charts

• Show parts to the whole but only work well with a few categories (3~5).

• It is overused, or used with so many categories that it becomes meaningless.

• Examples could include: -On a graph, displaying % of Sales by Region.

Key Principle:

Always select the simplest chart which gets your message across clearly.

8.2.2 KPIs and Scorecards

KPI (Key Performance Indicator) Visuals

• Show progress toward an objective or goal.

• Generally display current value, target value, indication of status (e.g., red/green arrows).

• Example: Amount of revenue reached vs. target revenue.

Scorecards

• Display a few KPIs in an organized, concise format.

• Enable business leads to view at a glance how well the business is performing on various metrics.

Illustration: Showing sales, expenses, customer satisfaction and churn rate in a single page of a dashboard.

Key Principle:

KPIs and scorecards should also focus on the organization's objectives, not just presenting raw data.

"Practical Work: Creating KPI Indicators in Power BI"

Instruction to Student:

You have a data consisting of monthly sales, profit and customer satisfaction scores. Do the following by setting up in Power BI:

Create three KPI visuals:

o Sales vs. Sales Target

o Profit Margin vs. Benchmark (15%)

o Customer Satisfaction vs. Target (85%)

Add conditional formatting: green for met, red for not.

Format the KPIs as a scoresheet on one report page.

Task Submit a screenshot of the scorecard and provide a brief commentary (approximately 150-200 words) on which KPIs are not doing well and what management should focus on.

8.2.3 Geospatial Graphics (Maps, Filled Maps) Map Graphics

- Geocode data points to show on a latitude/longitude or location based map.

• Well-suited for studying regional sales, customer coverage or distribution networks. Filled Maps (Choropleth Maps)

• Shade areas (states, districts etc) by value allowing for comparisons across geographies.

• Example: Displaying sales volume by state grouped as a fare place where the darker areas represent more expected sales.

Key Considerations:

• It is necessary to have a nicely organized hierarchy of the locations (Country → State → City).

- Not to be over-utilized for non-geographic data, as it distorts interpretation.

8.2.4 Custom Visuals from Power BI Market Place

There is also a Power BI marketplace which gives people the ability to use 3rd Party Custom Visuals to do even more with reporting than what they can do with out-of-the-box visuals. These can be imported, and used as if you were using native charts.

Examples of Custom Visuals:

• Bullet Charts: A feature that presents performance vs. target in a concise manner.

- Hierarchy Slicer: Enable to filter hierarchy more than default slicers.

• Word Clouds: Show most common terms in the text data.

• Heatmaps: Present the density or intensity of values in a matrix area or on maps.

Advantages:

• Customized for different industries (financial, healthcare, marketing).

• Enhance storytelling by providing different visual angles.

Cautions:

• Only use visuals from reputable publishers to mitigate the risk of performance or security issues.

• Keep it simple—use this visual if there is no other way to get the desired point across.

Did You Know?

“Do you know that Power BI custom visuals get certified by Microsoft before they are released as “Certified Visuals”. Certified visuals are vetted for performance and security and are therefore safer to use in enterprise wide deployments."

8.3 Interactivity in Reports

Reports in Power BI is not static, they are meant to be dynamic so that data can be explored interactively. Users can interact with the data, drilling down from aggregated views to specific data items, apply filters and annotating connections among the data points. These capabilities make reporting so much more than your average dashboard.

8.3.1 Drill-Down and Drill-Up Techniques

• Drill-Down: Refers to the process of drilling from a higher level summary (for example – annual sales) to lower-level detail (for examples – quarterly, monthly, and daily).

• Drill-Up: Returns to the aggregated or higher-level view.

Example:

For example, a year total column chart can be drilled to view quarter sales and then further drilled to month sales.

Key Benefits:

• Enables hierarchical exploration of data.

• Context can be provided, without crowding the report with several visuals.

Best Practice:

With grid views, makes obvious the nesting order (Year → Quarter → Month) so users have an understanding of how data is drilled down.

8.3.2 Drill-Through for Detailed Analysis

• Drill-through is used to create a special page for detailed information about a single data point.

• Right click and drill through to a new page with detail metrics of a value (e.g., sales for specific region).

Example:

A user might navigate through a summary of sales data to details for a customer—including his or her transactions, contact information and purchase history in that region.

Key Benefits:

• Reduces clutter on main dashboards.

• Enables concentrated details for strategic review.

Best Practice:

Keep your drill-through pages visualizations focused to relevant visuals and don't re-use those higher level charts on the report page.

8.3.3 Using Slicers and Filters Effectively

• Slicers are visual, on-page buttons that can filter data dynamically (such as data by year, quarter or month, etc.)

• Filters on visual, page or report level - gives you freedom to scope the analysis.

Example:

A slicer for Region whereby users can toggle between North, South, East and West to get performance at a certain level.

Best Practices:

• Slicers on the most relevant dimensions for decision makers (ie date, geography, product line).

• Don’t put so many slicers on dashboards.

• Utilize dropdown slicers when you have a large number of categories and need to save space.

8.3.4 Cross-Filtering and Highlighting

• Cross-filter Separately: Yes, when selecting a value in one visual it only filters the connected visuals.

• Highlighting: It highlights the relevant data in other visuals without stripping off context.

Example:

When an end user selects “Electronics” from a sales bar chart, all other visuals (maps, KPIs, trend lines) will also filter so they include only Electronics data.

Key Benefits:

• Promotes data exploration by revealing insights between disparate visuals.

• Facilitates rapid pattern and correlation discovery.

Best Practices:

• Makes sure relationships in the data model are fully defined or else you may experience unexpected results from filters.

• Tell end-users when to filter (to remove unrelated information) and highlight (to emphasize some information, but none is removed).

Did You Know?

“Did you know that Cross-filtering in Power BI is determined by the type of relationship defined in the data model (single vs. bi-directional)? If the relationships are not established correctly, selecting a value in one visual may result in other visuals not being filtered as expected."

8.4 Conditional Formatting and Storytelling Reports

Power BI allows analysts to not just show their data, but also to emphasize key insights with conditional formatting and explain the story behind it all with narrative-driven reports. This dual focus on data and storytelling make the reports both analytical and communicative.

8.4.1 Conditional Formatting in Tables and Charts

• Definition: Condition formatting is visual representation of data (through color, icons and data bars) applying certain condition or value.

• Application in Tables:

o Hue variation (e.g., from dark green to light green) that represents a high/low value.

o Arrows (▲ ▼) for increase and decrease.

o Data bars to use visual cues next to numbers.

• Application in Charts:

Categories passing vs. failing a condition (e.g., sales above/below target) in different colors.

Example:

" In a sales table, we could identify all products that have lower than 10% product margin in red within seconds.

8.4.2 Dynamic Visuals with Rules and Thresholds

• Changing visuals automatically : Dynamic formatting rules enable visuals to change automatically when they cross a threshold.

• These criteria can be either static (absolute values) or dynamic (criteria).

• Examples:

o If revenue is less than 80% of target KPI cards will turn red shell.

o The strip graph features green lines for sales that are above their target and orange lines for sales below targets.

Best Practice:

• Set clear thresholds and goals that align with the business.

• Don’t get too fancy with visuals and lay on too many rules, because that can just become confusing.

8.4.3 Designing Multi-Page Reports

• Use case: multi-page reports that separate different insights logically (for example, Overview, Sales Analysis, Customer Trends, Financial Performance).

• Structure:

o Snapshot Page: Summary details of the KPI’s and high-level executive views.

o Detail Pages: For further analysis of regions, product lines or customer segments.

o Drill-through Pages: Analysis to a particular entity (e.g.,customer profile).

• Slide navigation: Navigate slides with buttons, bookmarks, or page tabs to permit instinctive movement.

Example:

A retail report may have an opening dashboard and separate pages for “Sales Trends,” “Customer Retention” and “Expenses.”

“Activity: Building an executive dashboard across multiple pages”

Instruction to Student:

You are provided a retail dataset containing sales, expenses etc. along with customer details. Create a three-page report using Power BI in the following order:

Page 1 (Summary): The main KPIs (sales, expenses and profit margin and customers).

Page 2 (Sales Analysis): Sales trend by Region and Category to date drill down to daily detail.

Page 3 (Customer Insights): New v returning customers and churn trend.

Task: Submit your Power Bi file and write a brief explanation (200-250 words) of how each page helps tell the story of business performance.

8.4.4 Storytelling through BI Spreadsheet Dashboards

• Narrative Design: The dashboard should present a story and lead clients through data insights one by one.

• Elements of Storytelling in BI:

Context: Give the context (e.g., sales in Q3 were down).

Insight: Provide visuals on proof points (e.g. drop is low retention led).

Action: Recommend next actions (including ramping up customer loyalty efforts).

• Storytelling Techniques:

o Use annotations, labels and callouts to describe trends.

o Conditional formatting to show exceptions or variants.

o Develop dashboards to have a natural reading order (left-right, top-bottom).

Key Principle:

Storytelling dashboards aren't simply to display data, but — when used effectively — help convince and direct decisions.

Knowledge Check 1

Choose the correct option:

1 Which type of chart is best for displaying trends over time?

A) Bar chart

B) Line chart

C) Pie chart

D) Scatter plot

What are the objectives we want to achieve with a KPI visual in Power BI?

A) To show detailed transactions

B) To compare multiple categories

C) To see how far we are from a goal

D) To display geographic data

What is the feature in Power BI through which you navigate to a separate detail page from a piece of selected data?

A) Drill-down

B) Drill-through

C) Cross-highlighting

D) Bookmarks

Which of the following is a benefit to Power BI custom visuals?

A) They never require internet access

B) They could potentially offer you insights unique to your industry

C) They substitute for all normal images

D) They will always be faster than "ingame" visuals

A Power BI conditional formatting is usually used in order to:

A) Minimize total visuals per page

B) Emphasize by the rules or thresholds significant values

C) Remove outliers from a dataset

D) Merge multiple tables into one

8.5 Summary

This module focuses on the design and delivery of reports in Power BI:

• 8.1 outlined the fundamentals of successful visualization, stressing clarity, simplicity and relevance.

• 8.2 described standard and custom visuals, which included bar and line charts as well as KPIs, maps and marketplace extensions.

• 8.3 showcased the drill-down, drill-through, and cross-filtering features that facilitate interactive exploration of data.

• 8.4 showed how conditional formatting and storytelling could turn dashboards into decision-making tools by helping to focus attention and guide a story.

ϖ Combined, these abilities guide an analyst not only to go beyond static reports, but create informative, interactive and persuasive dashboards all in line with business objective.

8.6 Key Terms

Data Visualization – Use of graphical representation of data; helps to make sense and act on insights.

Drill-Down/Up – Move from a summary view to detailed information.

Drill-Through – Show detail page for a data point.

Slicer – How to display filters for Vendors in a column?

Cross-Filtering – When one visualization is selected, it filters other visualizations.

Conditional format – Use formatting to highlight data with your own color scheme, icons, and more.

Storytelling Dashboard – Story-based report: context → insight → action.

Custom visuals – Third party visuals from Power BI marketplace.

KPI Visual – KPI to target tracking.

Filled Map – Chart that shows a geospatial map; colors regions by measures.

8.7 Descriptive Questions

What is the significance of visual storytelling in BI? How is it different than traditional reporting?

Bring in data types and when bar charts, line charts, pie chart, etc. is appropriate to use?

What are the pros and cons of custom visuals available on Power BI marketplace?

Explain what you mean by either drill-down or drill-through. Illustrate when each would be the appropriate choice.

Talk about when and how slicers and filters can be implemented without making the end user's head explode.

Give three examples of conditional formatting rules that can add value to reports by making decision-making easier.

Describe Multi-page report structure in Power BI. Why do I care about flow in my storytelling dashboards?

What are some pitfalls in the creation of visualizations and how can they be avoided?

In what ways can interactivity increase the use of dashboards by stakeholders?

Why is it important to be consistent with design (color, layout) and scales while creating your visualization?

8.8 References

1. Microsoft Learn. (2023). Power BI Documentation. Retrieved from: https://learn.microsoft.com/power-bi

2. Few, S. (2012). Show Me the Numbers: Designing Tables and Graphs to Enlighten. Analytics Press.

3. Knaflic, C. N. (2015). Storytelling with Data: A Data Visualization Guide for Business Professionals. Wiley.

4. SQLBI.com – Technical articles and blogs on Power BI visualization best practices.

5. Power BI Community (2023). Forums and Learning Resources. Retrieved from: https://community.powerbi.com

Answers to Knowledge Check

Knowledge Check 1

1. B) Line chart

2. C) To measure progress against a target

3. B) Drill-through

4. B) They can provide unique industry-specific insights

5. B) Highlight important values using rules or thresholds

IBI_Unit 9_V3.docx

Business Intelligence using Power BI_BBA_3

Business Intelligence using Power BI_BBA_3

ATLAS SkillTech University

Document Details

Submission ID

trn:oid:::3618:127445823

Submission Date

Feb 3, 2026, 3:45 PM GMT+5:30

Download Date

Feb 3, 2026, 4:00 PM GMT+5:30

File Name

IBI_Unit 9_V3.docx

File Size

149.8 KB

19 Pages

4,216 Words

25,171 Characters

0% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 15 words)

Match Groups

1 Not Cited or Quoted 0%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **1** Not Cited or Quoted 0%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
0% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1 Internet

businessdocbox.com **<1%**

Unit 9: Dashboards, Publishing & Governance

Learning Objectives

1. Design interactive dashboards that integrate multiple visuals and KPIs for effective decision-making.
2. Apply storytelling techniques using bookmarks and navigation to create guided report experiences.
3. Demonstrate the process of publishing reports and dashboards to the Power BI Service.
4. Explain methods of sharing and collaboration within Power BI workspaces.
5. Evaluate governance practices to ensure secure and compliant use of Power BI.
6. Configure security roles and row-level security (RLS) for protecting sensitive data.
7. Manage scheduled data refresh and monitor dataset performance in the Power BI Service.

Content

9.0 Introductory Caselet

Anita's Enterprise Dashboard Rollout: Finding the Right Balance of Access, Security & Collaboration

Background:

Anita is a senior BI (business intelligence) analyst for a major financial services company. After several months of his team's work, she has an advanced Power BI dashboard combining information across systems on customer transactions, branch operations and risk indicators. The prototype in Power BI Desktop was nice for going through department reviews, but now the work focuses on productionizing it for use throughout the organization.

Executives are looking for quick, top-line KPIs, managers want to drill down into specifics of the reports by geography and on a larger scale compliance officers wanting to make sure data is secure and the governance around it is in place. Anita learns that the formation of enterprise level ready solution is far beyond better visuals but also involves an active design, secure publishing, effective sharing and strong governance.

The Challenge:

Anita encountered a few problems with it when she first posted the dashboard to the Power BI Service:

• There was no holistic navigation and storytelling guidance for users to follow.

• Regional managers could access all company data, which was a security concern.

• Data wasn't being automatically renewed, leading to executives making decisions based on stale information.

• Documents were shared via ad hoc links, creating challenges with governance and version control.

The Turning Point:

Anita and her colleagues tackled these problems by:

• Remodeled dashboards including KPIs, drill down and slicers for interactivity.

• Navigated using bookmarks and navigation menus to walk executives through insights in a step-by-step fashion.

• Added Row-Level Security (RLS) to control users access into other regions data.

• Set up an on premises data gateway for scheduled refresh.

• Operationalized governance policies by establishing certified data sets and publishing curated dashboards using

Power BI Apps.

Outcome:

The two new dashboards empowered various stakeholders on different levels:

• Summary KPI apps:Executives saw summary KPIs through curated app(s).

• Regional managers safely drilled into their data.

• There were no confidence issues with governance and audit trails for compliance officers.

The critical importance of Design, Storytelling, Publishing & Sharing, Governance and Security is illustrated in the case of Anita.

in helping Power BI dashboards become not only informative but enterprise ready.

Critical Thinking Question:

Now if you were Anita, how would you balance end-user convenience consideration with stringent security and governance needs? What are the dangers associated with one of these areas being too heavily emphasized at the expense of the other?

9.1 Designing Interactive Dashboards

An interactive dashboard in Power BI is not just a series of visuals bundled together, it's a decision support tool that puts the critical KPIs and the most relevant insights at your fingertips (quite literally), allowing you to further explore the data by interacting with it. Good dashboards use design principles, the right visuals and interactive features to make sure they are clear, usable and relevant.

9.1.1 Principles of Dashboard Design (Clarity, Usability, KPIs)

figure: **Clarity, Usability, KPIs**

• Uncertainty: Dashboards need to portray (or not) confidence. Avoid clutter, including too many colors 3-D graphs or gratuitous graphics. Titles, legends and labels must directly describe what is being measured.

• Usefulness: The design of the dashboard should allow users to get what they want quickly and easily, with clear sections for visuals, a consistent visual formatting and low cognitive load. Readers should be able to understand the report without prior training.

• KPIs (Key Performance Indicators): Dashboards are intended to emphasize the organization's top level metrics. The best practice is to keep KPIs on the top of your dashboard for a snapshot or summary view, allowing execs to understand performance at a high level before drilling down.

Example: For a retail sales dashboard, you may have KPIs like total revenue, profit margin, and customer satisfaction represented followed by the sales breakdown into categories.

9.1.2 Combining Multiple Visuals into a Dashboard

• Dashboards consolidate several types of visuals — charts, tables, maps, key performance indicators — all in one view.

• Each visual should have a specific analytical job it is doing:

o Trends → Line chart

o Comparisons → Bar/column chart

o Geographic insights → Map

o Part-to-whole → Pie or donut chart (not too many!)

• Visuals should be additive rather than redundant.

Best Practices:

• Place your imagery in harmony – It's about achieving artistic balance.

• Avoid cramming too many images onto a single page (For the optimal experience, aim for 6-8).

• Use the same color palette for categories in all visuals.

9.1.3 Using Tiles, Cards, and KPIs for Summary View

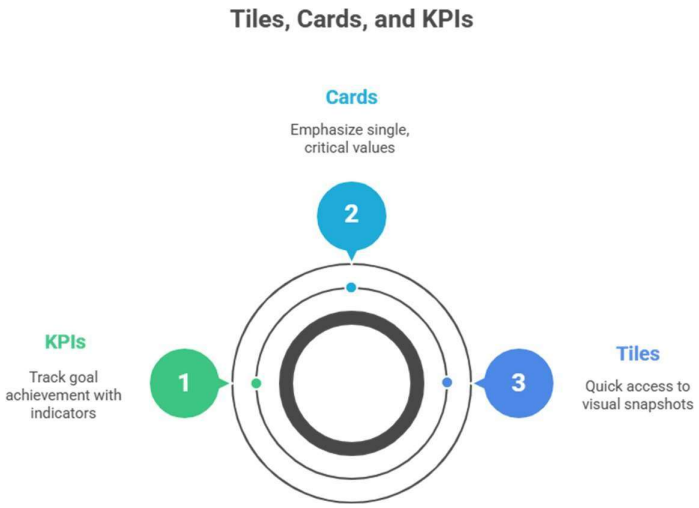


figure: **Tiles, Cards, and KPIs**

• Tiles: Tiles are snapshots of visuals pinned to dashboards in Power BI Service. They also serve for fast access to the most important metrics without running full reports.

• Cards show single values (total revenue, active customers etc). Best for emphasizing critical figures.

• KPIs: Display actual value vs. target + indicators (eg red/green arrows). KPIs show whether or not a business is achieving its objectives.

Example:

In a financial dashboard, at the head of each page cards can be used to display "Total Sales = ₹120 Cr" and "Profit Margin = 18%", while KPI visuals can show "Sales vs Target" with status indicator.

9.1.4 Enriching Interactivity with Drill-Down and Filters

• Drill-Down/Drill-Up: Allows to navigate across levels of data hierarchy (Year → Quarter → Month → Day). This also cuts down on clutter and gives more information if you want it.

• Filters and Slicers: Enable users to interactively choose subgroups of data (such as by geographical area, product, or time period).

• Cross-Filtering: Simply clicking on a visual can filter the rest of the visuals on your report and it will pin that value to every other measure.

Example:

Choose "South Region" on a map in a sales dashboard, for example, and bar charts, KPIs and trend lines will filter down automatically to show South's performance alone. Users can then further narrow down to "South → Karnataka → Bengaluru" for more in-depth analysis.

9.2 Storytelling with Bookmarks and Navigation

The feature Storytelling in Power BI adds a complicating factor as it can turn dashboards into guided narratives that pull users through insights and ideas, like so. Using bookmarks, navigation buttons and layouts, analysts can produce interactive dashboards that mimic presentations allowing stakeholders to wander the pages to see the evidence, context and conclusions.

9.2.1 What is Bookmarks in Power BI?

• Definition: Bookmarks record the current state of a report page, including filters and slicers, drill-down levels, visibility of visuals.

• Use case: They enable report developers to save and retrieve specific views, effectively serving as “checkpoints” in a story.

• Use Cases:

o Highlighting a particular region’s sales.

o Display pre-filtered data for the customer segment selected.

o Tour the dashboards via a step-by-step guide that moves between set views.

For example, a sales dashboard might have bookmarks that switch between “Overall Sales Performance” and “Top 10 Products.”

Did You Know?

“DID YOU KNOW: bookmarks in Power BI are able to store more than just the visual layout, they also store filter states, slicer selections, drill levels and hidden visuals? All of this make these an incredibly powerful tool for telling stories, making presentations or navigation menus that fit the need.”

9.2.2 Using Selection Pane and Custom Views

• Selection Pane: It allows one to decide which visuals are visible or not in a page. It also goes hand-in-hand with bookmarks to help create custom views.

• Custom Views: Create different points of focus in the same report page by hiding or showing visuals with the selection pane.

Example:

Another bookmark can represent a demographic profile on one customer dashboard, then with visual visibility toggling be made to represent purchase behavior on the same page.

Best Practice: Clearly identify visuals in the Selection Pane to know which elements are impacted by bookmarks.

Instruction to Student:

You are given a sales, customer and expense analysis in a multi-page Power BI report. Perform the following tasks:

Hide and show visuals for different states of your report with the Selection Pane.

Add three bookmarks for the Sales Overview, Customer Insights, and Expense Breakdown views.

Paste navigation buttons (such as shapes or icons), and then link each button to a bookmark.

Try the navigation, go from view to view as if you were pitching a story to some executive. Task: Submit your power bi file, along with a brief summary (150 - 200 words) which explains the added value of navigation in presentation compared to standard page tabs.

9.2.4 Design of Storytelling Dashboards With Guided Insights

• Guided Insights: Storytelling dashboards Each storytelling dashboard is built following a narrative structure:

Context – Provide background information.

Insight -Visualize insights |---------- Use some kind of visual to represent findings.

Action – Propose some courses of action or recommendations.

• Bookmarks + Navigation- Pairing bookmarks with the new navigation experience lets you create interactive read-only reports that can be used as presentations.

• Use Case: A financial report could walk a director level through general revenue trends → departmental analysis → expense slices → profitableness findings.

Best Practices:

• Keep the order logical and audience-centered.

• Highlight important findings using annotations and callouts.

• Don’t deploy too many animations or toggles that detract from the story.

9.3 Publishing and Sharing to Power BI Service

After you create dashboards and reports in Power BI Desktop, the next step is to publish them so that stakeholders can access them. Cloud service for publishing, sharing and collaborating on reports Insights all through the right channels The Power BI Service is a

cloud-based solution with which users can access their dashboards and reports from anywhere, anytime.

9.3.1 Introduction to Power BI Service and Workspaces

• Power BI Service is the place in cloud where reports and dashboards are hosted, consumed and shared.

• Workspaces are shared spaces in the service where you and your users can create, test, and publish reports.

• Types of Workspaces:

o My Workspace: Personal space which is based on the implementation of individual users.

o App Workspaces- Shared spaces teams can work together, appoint roles and redistribute content.

Key Benefits:

•Single access to dashboards and datasets.

• Collaboration based on role (Admin, Member, Contributor, Viewer).

• Integrations with Microsoft Teams and other productivity software.

9.3.2 Publishing Reports from Desktop to Service

• A report that you create in Power BI Desktop can be published directly to the Power BI Service.

• Steps to Publish:

Sign in to Power BI Desktop using your organization's sign-in authority.

Press the Publish button in the Home ribbon.

Choose the destination workspace in Power BI Service.

• Adopters of the report can use the online version, which is editable, shareable and refresh-able.

Best practice: Verify that the sensitive information is secure via local testing before reporting.

Activity: Publish and share reports in the Power BI service

Instruction to Student:

You build a sales performance report in Power BI Desktop. Perform the following:

Log into Power BI Desktop with your organization account.

Choose to publish the report to Power BI Service (to a workspace).

Pin at least three visuals as tiles to Service to create a dashboard.

Open the sharing settings and share your dashboard with two people (or groups).

Task: Upload screenshots of your published report, dashboard and the sharing configuration. Write a brief (150-200 word) summary of the benefits of Power BI Service compared to sending static reports by email.

9.3.3 Users and Groups in the Dashboard Syndication

• Do share the reports and dashboards in Power BI Service with individual users or security groups.

• Sharing Options:

o Share directly with email addresses.

o Share to Azure Active Directory groups for simpler management.

o Permissions control: allowing the recipients to reshare, build on datasets or export the data.

• Limitations:

o For sharing, both sender and recipient must have a Power BI Pro license.

o Recipients see a live, working version but cannot edit unless given access to the workspace. Example: A sales manager can let the entire regional sales team in on a performance dashboard, so that everyone always has access to that same set of data.

9.3.4 Power BI Apps and Collaboration Features

• Power BI Apps: Grouping of dashboards and reports created in and published from a workspace.

o People install apps to reach published, read-only content quickly.

o Apps hide the chaos and organize a series of dashboards, reports by combining with similar ones.

• Collaboration Features:

o Reflecting on articles to discussion them in context.

o Collaboration with Microsoft Teams for decision making.

o Data Alerts on KPIs for real-time monitoring.

Key Benefit: Mobile applications and collaboration tools make it possible not only to SHARE the insights they produce but also to ACT upon them in an organized and secure way.

9.4 Governance, Security, and Data Refresh

When Power BI is rolled out for enterprise, it's about more than just building dashboards but architecture, governance, security and data management. These aspects help business intelligence solutions to be reliable, compliant and secure for providing accurate insights.

9.4.1 Governance Policies and Access Management

• Governance in Power BI means setting policies regarding how data, reports, and dashboards are created, shared and by whom.

• Access Management:

o Workspace Role based permissions -- Admin, Member, Contributor, Viewer.

o Permissions to share, (limit any unauthorized resharing/export).

• Governance Policies include:

o Ways to name datasets and reports.

o Version control and change management.

o Auditing and monitoring report usage.

Example: One common policy enforced could be that certified sources (datasets) are the only ones that can be used for financial reports to maintain consistency and a single source of truth.

9.4.2 Row-Level Security (RLS) in Power BI

• Goal: RLS attempts to limit the access of one table only for a specific user.

• Function: Users are shown the data subset appropriate to their role or location.

• Implementation Steps:

Define roles and rules in Power BI Desktop (for example, by setting Region = "North").

Map users/groups to roles in Power BI Service.

• Dynamic RLS: Utilize USERNAME() or USERPRINCIPALNAME() functions to auto-apply filters at the row level based on the logged-in user.

Example: A Sales Dashboard may give Northern Manager a sight of sales in North, and Southern Manager can see only data from South from same report.

9.4.3 Scheduled Data Refresh and Gateway Configurations

• Standard Data Refresh: Keep reports in the Power BI Service up to date with data from the sources you connect to.

• Gateway Configuration:

o On-premises data gateway: Only required if the cloud-based reports need to connect using on-premise data sources (e.g. SQL server, ERP systems).

o Gateways securely move data between on-premises and Power BI Service systems.

• Refresh Options:

o Scheduled refresh (daily, hourly).

o Manual refresh (on demand).

o Real-time streaming datasets.

Key ConcepTion: The Refresh Frequency must be an equilibrium of business requirements and system performance/capacity.

9.4.4 Best Practice of Governance and Compliance

Consistent Data Sources: New and established referencing agencies maintain updated lists of Original Article URLs.

Use Layers: Use both the permissions from workspace and RLS.

Monitor and Audit Usage: Use Power BI audit logs to monitor sharing and access.

Documentation policies: Make sure to keep listing roles and reload schedules, access guidelines.

Adherence to Laws: Align Power BI with frameworks like GDPR, HIPAA, or ISO 27001 by industry.

Result: Robust governance and compliance policies build confidence in reports, safeguard sensitive data and help keep business running.

Knowledge Check 1

Choose the correct option:

What is the top principle you should keep in mind when you want to create an executive dashboard in Power BI?

A) Including as many images as we can

B) Using animations for engagement

C) Transparency and use of KPIs

D) Using 3D charts for aesthetics

What is the scope of a bookmark in PowerBI?

A) Only the current visual layout

B) Only the applied filters

C) How it is Visually layedOut, filters, slicers and drill level

D) Only hidden visuals

What is the main purpose of a workspace in Power BI Service?

A) To store Excel sheets only

B) Support to report building and sharing in an agreeable manner

C) To replace on-premises gateways

D) To export dashboards as PDFs

What feature allows a regional manager to view only the data from their region in a shared report?

A) Navigation buttons

B) Drill-through

C) Row-Level Security (RLS)

D) Custom visuals

What is the purpose of an on premises data gateway in Power BI?

A) Enable the secure transfer of data from on-premises to Power BI Service

B) Creating custom visuals for dashboards

C) To remove slicers and filters from reports

D) For datasets to be made available via the internet

9.5 Summary

ϖ This module focused on dashboards going beyond visualization and out into story-telling, work together and governance with Power BI:

• 9.1 introduced ideas of dashboard design, merging visuals, KPIs and interactivity.

• 9.2 brought to you storytelling on bookmarks and navigation, so you can tell the readers to read it this way.

• 9.3 addressed publishing and sharing in the Power BI Service, including workspaces, apps and collaboration tools.

• 9.4 emphasized governance, security and data refresh (access management row-based security and compliance).

ϖ With these parts combined, analysts are able to create dashboards that aren't just effective but also built.

secure, collaborative, and enterprise-ready.

9.6 Key Terms

Dashboard – Visuals and KPI's Gather in one view to look at the overall site performance.

Bookmark – A saved report state (contains filters & visibile visuals).

Selection Pane – A utility to show/hide visuals for custom views.

Drill-Through – Go to the detailed page for what was clicked on.

Workspace – Shared space for creating and sharing reports.

Power BI App – To package the dashboard/reports for distribution.

Row-Level Security – Limits access to rows in a database table based on the user logged into the application.

Gateway The secure on-premises bridge to Power BI Service.

Data Refresh – Brings the datasets up to date with the latest data.

Governance – Truth and use policies to secure compliance for BI.

9.7 Descriptive Questions

Describe the effective design principles of dashboards in Power BI. How do the clarity and usability of KPIs influence decision making?

Explain drill-down, drill-up and drill-through operations. Provide one example for each.

How do bookmarks and buttons change dashboards into a presentation tool?

What are workspaces in the Power BI Service and how do they enable collaboration?

Describe how sharing dashboards are different to published Power BI Apps.

Define Row-Level Security (RLS). How does it enhance the governance and the privacy of data?

Explain when you would use the Power BI Gateway. Why is it important in the context of schedules refresheing?

Provide three best practices for governance and compliance in Power BI deployments.

Give us an example of how publishing and sharing can help facilitate collaboration between disparate departments.

Why is it so important to manage data refresh in large BI environments?

9.8 References

1. Microsoft Learn. (2023). Power BI Documentation. Retrieved from: https://learn.microsoft.com/power-bi

2. Russo, A., & Ferrari, M. (2020). Power BI Governance and Deployment Approaches. SQLBI.

3. Kamat, R. (2022). Power BI Service Administration and Governance. Apress.

4. Power BI Community (2023). Discussion forums and tutorials. Retrieved from: https://community.powerbi.com

5. Whitepaper: Row-Level Security in Power BI. Microsoft Corporation.

Answers to Knowledge Check

Knowledge Check 1

1. C) Clarity, usability, and focus on KPIs

2. C) Visual layout, filters, slicers, and drill levels

3. B) To provide a collaborative environment for building and sharing reports

4. C) Row-Level Security (RLS)

5. A) To allow secure data transfer from on-premises sources to Power BI Service

9.9 Case Study

Using Power BI Service to create a secure and scalable Corporate Sales Dashboard

Case study 1: Power BI Deployment in an International Manufacturing Firm Background

Rajiv, as a BI Manager at a Multi National corporation for manufacturing had created Power BI Desktop interactive dashboards. Although his prototypes were popular among department heads, transitioning from individual desktop reporting to companywide

distribution proved more challenging. A number of important aspects were addressed, e.g., data security, access control, freshness and governance.

Background

The company's reporting requirements were wide-ranging:

• Executives needed high-level KPIs

*Regional managers wanted gated region-specific views.

Compliance officers believed that we were a burden.

When reports reached hundreds of users, Rajiv discovered that the Power BI Service was necessary to simplify sharing and apply enterprise standards.

Problem 1: No Centralized Access

Reports were emailed as static objects, resulting in versioning conflicts and data silos.

Solution:

He uploaded power bi service reports to the Power BI Service via shared workspaces. It enabled departments to work together in real time, sharing a single source of truth.

Problem 2: Security of the regional data

Some regional managers were able to see data from other locations, which is a breach of internal policy.

Solution:

Rajiv used Row-Level Security (RLS) to make sure Edgar could see only data for his region.

Problem Statement 3: Data Description and Its Freshness

The dashboards ran on-premises ERP systems. Report lag was the result of manually uploading data.

Solution:

Rajiv set up an on-premises data gateway and scheduled automated refreshes so that the reports always showed the most current metric values, without needing any manual intervention.

Issue 4: Governance and Compliance

Executives voiced worrry about undocumented sharing and use of data.

Solution:

Rajiv intends to create a reporting governance model by:

• Certifying datasets

• Assigning role-based permissions

• Documenting report logic and purpose

• Deploying dashboards from Power BI Apps

Outcome

Within three months:

•Curated KPIs were securely accessed by executives through Power BI Apps

• Performance was drilled into regional managers with no ability to see what others were doing

• Administrative teams were more comfortable with how data was managed and if it was the correct version

• IT cost of overhead was cut through automation of refresh and centralization management.

Problem 3: Refresh data problems

The dashboards were fed by on- premises ERP systems. Manual data submission led to delayed reporting.

Solution:

Rajiv used an on-premises data gateway and set up refresh schedules, so that everybody could always see reports with current numbers without any additional effort.

Issue #4: Governing and Compliant Applications

Executives expressed fears about unauthorized sharing, and lack of documentation.

Solution:

Rajiv established a model for report governance by:

• Certifying datasets

• Assigning role-based permissions

• Documenting report logic and purpose

• Sharing dashboards in Power BI apps

Outcome

Within three months:

• Execs securely consumed curated KPIs with Power BI Apps

• Managers at the regional level only looked into their own performance and had no access to anyone elses data

• Governance teams became comfortable with the data and versioning

• Automated refresh and centralized management lowered IT overhead

MCQ

What was Rajiv in his struggle to have managers only see the data that was meant for them?

A) Publishing reports as static PDFs

B) Applying Row-Level Security (RLS)

C) Using bookmarks for each region

D) Sharing Excel extracts individually

Answer:

B) Applying Row-Level Security (RLS)

Explanation:

Row level security Row level security (RLS) is used to limit table data rows that a user can access.

Case study 2 –Corporate sales dashboard : Publishing, Securing and Sharing using power BI Service

Introduction

Ritika, a Power BI Developer working for a corporate retail chain was given an assignment to create and implement the Corporate Sales Dashboard which had to serve multiple roles -Executives, Territory managers, Analysts!

The dashboard was developed in Power BI Desktop, and required secure sharing, real time insight and meet internal governance requirements.

Background

The retailer had more than 300 stores across 5 territories. Dashboard requirements included:

• KPIs (Sales vs. Target, Growth, Revenue)

• Views personalized to each territory manager

• Proactive Distribution for protecting from data was taken out (Data loss prevention).

• Auto refresh for daily sales data

The dashboard was decided to be scaled up for over 150 users using the Power BI Service with control and performance in mind.

Problem Statement 1: Inconsistent Distribution

Multiple copies of the dashboard were sent to users' inboxes, and breaks them down by outlook versions.

Solution:

Ritika shared the report into Power BI Service Workspace and disseminated access to everyone through Power BI Apps that way she knew everybody was looking at the same certified version.

Problem Statement 2: Access Control over Regions

In the beginning, all users were able to view sales from any region, which is a breach of the data security rules.

Solution:

Ritika mapped RLS roles to roles using a UserRegion map table filtering access per your log in. This way, only the respective data were seen by each manager.

Problem Description 3: SQL Server Cached Data Stale Rate

Sales data was sitting in an on-premises SQL Server, and there were manual weekly imports.

Solution:

She stood up the On-Premises data gateway and set up scheduled daily refreshes, achieving near real-time reporting on sales.

Issue 4: No User Training and Documentation

A lot of users found it difficult to apply filters properly, or do drill-through analysis and so on, which was a barrier for them to actually get around the system.

Solution:

Ritika made a user-guide on the dashboard with buttons and tool-tips. A quick training video was announced as well working with Microsoft Stream integration.

Outcome

The new corporate sales dashboard:

• Provided role specific insights to the user

• Daily data refreshes from the SQL Server were scheduled to occur automatically.

• Secured sharing using Power BI Apps and RLS

• More interaction with integrated user help

The implementation optimized sales analysis time, decreased manual reporting by 80% and increased decision making activity at a regional level.

MCQ

How did Ritika make sure that every regional manager could only see the data regarding his/her own sales?

A) Leveraged page level filters in Power BI

B) Created region specific dashboards

C) Use of Row Level Security with mapping table

D) Emailed screened Excel files for each region

Answer:

C) Row-Level Security using a mapping table

Explanation:

We introduced a dynamic RLS model by creating user-to-region mapping table, by which a single dashboard could securely meet all the users requirements.

Conclusion

Power BI dashboards cannot be considered simply a design requirement treatment but require specialized skills for security, governance, automation and user adoption when you scale them across the enterprise. Leveraging the Power BI Service, RLS, Data Gateways, and Power BI Apps can turn these static reports into scalable secure interactive reporting ecosystems.