

INDEX - Course Lecture Out-Line

Chapter: 1 Introduction to Data Analytics.

Page No. 3 - 17

- Course Introduction
- Data Analytics Overview
- Dealing with Different Types of Data
- Data Visualization for Decision making
- Data Science, Data Analytics, and Machine Learning
- Data Science Methodology
- Data Analytics in Different Sectors
- Analytics Framework and Latest trends

Chapter: 2 Data Ingestion

Page No. 18 - 56

- Power BI Data Ingestion and Modelling
- Reading from MySQL
- Reading from HDF5
- Reading from The Web
- Reading from APIs
- ETC
- Reading from Other Sources
- Summary and close

Chapter: 3 Data Cleansing

Page No. 57 - 100

- we look at finding data and reading different file types.
- Getting and Cleaning Data! The primary goal is to introduce you to the most common data storage systems and the appropriate tools to extract data from web or from databases like MySQL.
- Introduction to Data Wrangling and Shaping
- Advanced Shaping
- Beyond Initial Shaping
- Advanced DAX
- Publishing and Testing Your Dataset
- Summarizing Data
- Creating New Variables

Chapter: 4 Data Modelling

Page No. 101 - 141

- What is Data Modelling?
- Data Modelling sometimes needs Data Analysis
- Star Schema
- Snowflake Schema
- Create a Relationship
- One to many, many to many relationships

Chapter: 5 Data Visualization

Page No. 142 - 204

- Overview of Power BI
- Advantages of using Power BI
- Chart Selection
- Getting Started
- Transforming Data
- Report Creation
- Different types of visualization charts
- Area Charts
- Line Charts
- Bar Charts
- Column Charts
- Combo Charts

Chapter: 6 Data Regression

Page No. 205 - 244

- Types of Regression Models
- Simple Linear Regression Model
- Simple Linear Regression Equation
- Equation for the best straight line
- Graph of the best straight line
- Interpreting the Results
- Measures of Variation

Course Assignment

Page No. 245 - 264

Assignment on Power BI report creation with Solution

CHAPTER 7 : INTRODUCTION	Page No: 265-274
CHAPTER 8 : DATABASE	Page No: 275-305
CHAPTER 9 : DATA ENGINEERING SERVICES	Page No: 306-329
CHAPTER 10 : S3 or STORAGE SERVICES	Page No: 330-343
CHAPTER 11 : DMS (Ingestion)	Page No: 344-363
CHAPTER 12 : REDSHIFT or CLOUD BASE DATA WAREHOUSE	Page No: 364-390
CHAPTER 13 : GLUE or DATA PIPELINING	Page No: 391-396
CHAPTER 14 : EMR PROCESSING	Page No: 397-404
CHAPTER 15 : SCHEDULING & MONITORING	Page No: 405-412

Chapter: 1

Introduction to Data Analytics

Scope:

- I. Course Introduction
- II. Data Analytics Overview
- III. Dealing with Different Types of Data
- IV. Data Visualization for Decision making
- V. Data Science, Data Analytics, and Machine Learning
- VI. Data Science Methodology
- VII. Data Analytics in Different Sectors
- VIII. Analytics Framework and Latest trends

1.1 Data Analytics:

Data analytics is the science of analyzing raw datasets in order to derive a conclusion regarding the information they hold. It enables us to discover patterns in the raw data and draw valuable information from them. Data analytics processes and techniques may use applications incorporating machine learning algorithms, simulation, and automated systems. The systems and algorithms work on the unstructured data for human use. These findings are interpreted and used to help organizations understand their clients better, analyze their promotional campaigns, customize content, create content strategies, and develop products. Data analytics help organizations to maximize market efficiency and improve their earnings.

1.2 Process of Data Analytics method:

Step 1: Determine the criteria for grouping the data

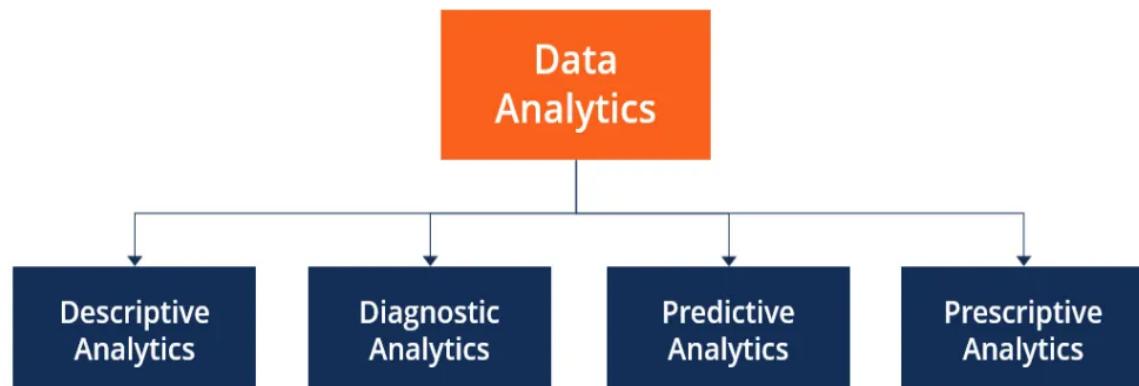
Step 2: Collecting the data

Step 3: Organizing the data

Step 4: Cleaning the data

1.3 Data Analytics Types:

The following are the four fundamental types of data analytics:



1. **Descriptive Analytics** describes the happenings over time, such as whether the number of views increased or decreased and whether the current month's sales are better than the last one.
2. **Diagnostic Analytics** focuses on the reason for the occurrence of any event. It requires hypothesizing and involves a much diverse dataset. It examines data to answer questions, such as "Did the weather impact the selling of beer?" or "Did the new ad strategy affect sales?"

3. **Predictive Analytics** focuses on the events that are expected to occur in the immediate future. Predictive analytics tries to find answers to questions like, what happened to the sales in the last hot summer season? How many weather forecasts expect this year's hot summer?

4. **Prescriptive Analytics** indicates a plan of action. If the chance of a hot summer calculated as the average of the five weather models is above 58%, an evening shift can be added to the brewery, and an additional tank can be rented to maximize the production.

1.4 Role of Data Analytics:

As the process of analyzing raw data to find trends and answer questions, the definition of data analytics captures its broad scope of the field. However, it includes many techniques with many different goals.

The data analytics process has some components that can help a variety of initiatives. By combining these components, a successful data analytics initiative will provide a clear picture of where you are, where you have been and where you should go.

- Generally, this process begins with *descriptive analytics*. This is the process of describing historical trends in data. Descriptive analytics aims to answer the question “what happened?” This often involves measuring traditional indicators such as return on investment (ROI). The indicators used will be different for each industry. Descriptive analytics does not make predictions or directly inform decisions. It focuses on summarizing data in a meaningful and descriptive way.
- The next essential part of data analytics is *advanced analytics*. This part of data science takes advantage of advanced tools to extract data, make predictions and discover trends. These tools include classical statistics as well as machine learning. Machine learning technologies such as neural networks, natural language processing, sentiment analysis and more enable advanced analytics. This information provides new insight from data. Advanced analytics addresses “what if?” questions.
- The availability of machine learning techniques, massive data sets, and cheap computing power has enabled the use of these techniques in many industries. The collection of big data sets is instrumental in enabling these techniques. Big data analytics enables businesses to draw meaningful conclusions from complex and varied data sources, which has been made possible by advances in parallel processing and cheap computational power.
- Data analysts exist at the intersection of information technology, statistics and business. They combine these fields in order to help businesses and organizations succeed. The primary goal of a data analyst is to increase efficiency and improve performance by discovering patterns in data.
- The work of a data analyst involves working with data throughout the data analysis pipeline. This means working with data in various ways. The primary steps in the data

analytics process are data mining, data management, statistical analysis, and data presentation. The importance and balance of these steps depend on the data being used and the goal of the analysis.

- Data mining is an essential process for many data analytics tasks. This involves extracting data from unstructured data sources. These may include written text, large complex databases, or raw sensor data. The key steps in this process are to extract, transform, and load data (often called ETL.) These steps convert raw data into a useful and manageable format. This prepares data for storage and analysis. Data mining is generally the most time-intensive step in the data analysis pipeline.
- Data management or data warehousing is another key aspect of a data analyst's job. Data warehousing involves designing and implementing databases that allow easy access to the results of data mining. This step generally involves creating and managing SQL databases. Non-relational and NoSQL databases are becoming more common as well.
- Statistical analysis allows analysts to create insights from data. Both statistics and machine learning techniques are used to analyse data. Big data is used to create statistical models that reveal trends in data. These models can then be applied to new data to make predictions and inform decision making. Statistical programming languages such as R or Python (with pandas) are essential to this process. In addition, open source libraries and packages such as TensorFlow enable advanced analysis.
- The final step in most data analytics processes is data presentation. This step allows insights to be shared with stakeholders. Data visualization is often the most important tool in data presentation. Compelling visualizations can help tell the story in the data which may help executives and managers understand the importance of these insights.

1.5 Types of Data Analytics:

Data analytics is a broad field. There are four primary types of data analytics: descriptive, diagnostic, predictive and prescriptive analytics. Each type has a different goal and a different place in the data analysis process. These are also the primary data analytics applications in business.

- Descriptive analytics helps answer questions about what happened. These techniques summarize large datasets to describe outcomes to stakeholders. By developing key performance indicators (KPIs,) these strategies can help track successes or failures. Metrics such as return on investment (ROI) are used in many industries. Specialized metrics are developed to track performance in specific industries. This process requires the collection of relevant data, processing of the data, data analysis and data visualization. This process provides essential insight into past performance.
- Diagnostic analytics helps answer questions about why things happened. These techniques supplement more basic descriptive analytics. They take the findings from descriptive analytics and dig deeper to find the cause. The performance indicators are

further investigated to discover why they got better or worse. This generally occurs in three steps:

- Identify anomalies in the data. These may be unexpected changes in a metric or a particular market.
- Data that is related to these anomalies is collected.
- Statistical techniques are used to find relationships and trends that explain these anomalies.
- Predictive analytics helps answer questions about what will happen in the future. These techniques use historical data to identify trends and determine if they are likely to recur. Predictive analytical tools provide valuable insight into what may happen in the future and its techniques include a variety of statistical and machine learning techniques, such as: neural networks, decision trees, and regression.
- Prescriptive analytics helps answer questions about what should be done. By using insights from predictive analytics, data-driven decisions can be made. This allows businesses to make informed decisions in the face of uncertainty. Prescriptive analytics techniques rely on machine learning strategies that can find patterns in large datasets. By analyzing past decisions and events, the likelihood of different outcomes can be estimated.

These types of data analytics provide the insight that businesses need to make effective and efficient decisions. Used in combination they provide a well-rounded understanding of a company's needs and opportunities.

1.6 Benefits of Data Analytics:

1. Decision making improves
2. Marketing becomes more effective
3. Customer service improves
4. The efficiency of operations increases

1.7 Data Visualization:

Data visualization is the practice of translating information into a visual context, such as a map or graph, to make data easier for the human brain to understand and pull insights from. The main goal of data visualization is to make it easier to identify patterns, trends and outliers in large data sets. The term is often used interchangeably with others, including information graphics, information visualization and statistical graphics.

Data visualization is one of the steps of the data science process, which states that after data has been collected, processed and modelled, it must be visualized for conclusions to be made. Data visualization is also an element of the broader data presentation architecture (DPA) discipline, which aims to identify, locate, manipulate, format and deliver data in the most efficient way possible.

1.7.1 Why is data visualization important?

Data visualization provides a quick and effective way to communicate information in a universal manner using visual information. The practice can also help businesses identify which factors affect customer behaviour; pinpoint areas that need to be improved or need more attention; make data more memorable for stakeholders; understand when and where to place specific products; and predict sales volumes.

Other benefits of data visualization include the following:

- the ability to absorb information quickly, improve insights and make faster decisions.
- an increased understanding of the next steps that must be taken to improve the organization.
- an improved ability to maintain the audience's interest with information they can understand.
- an easy distribution of information that increases the opportunity to share insights with everyone involved.
- eliminate the need for data scientists since data is more accessible and understandable; and
- an increased ability to act on findings quickly and, therefore, achieve success with greater speed and less mistakes.

1.8 Data Visualization for Decision making

The average person makes a stunning 35,000 choices per day. People have been creating, adapting, and refining decision-making techniques long before big data made its entrance. Scientists have also been studying these techniques for decades, trying to understand how we can make good decisions, and how we can avoid bad decisions. Research shows that we make decisions by forming opinions and choosing actions through "mental processes which are influenced by biases, reason, emotions, and memories." While we work hard to develop rational, logical conclusions, there are still several factors that can limit our ability to make the right decisions.

Here are four ways that data visualization can reduce these factors and improve your decision-making.

Increase Speed

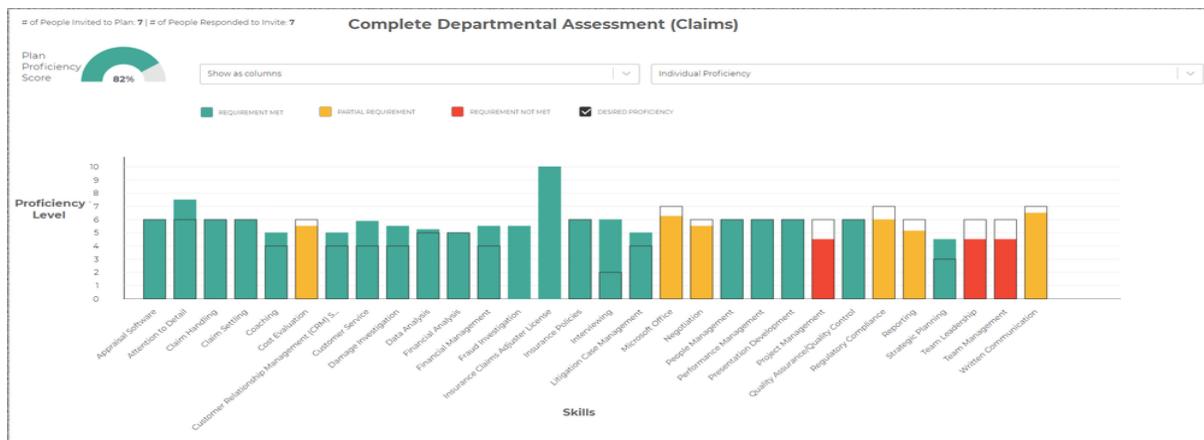
Information travels at the speed of light in our always-connected world. Due to the fast pace of business, managers are often expected to make critical decisions in short periods. If you are not timely, an opportunity may pass by, or a small problem may grow exponentially worse overnight. Data visualization can help you draw actionable insights from massive amounts of

data in a short amount of time. Even a simple visualization, like a bar graph, can present valuable insights in seconds.

Data Analytics

Student Material

Take a look at the example below:



Data collected from a corporate technology assessment is organized into this colorful bar graph. By glancing at the chart, an IT manager could immediately recognize which skills need improvement. From there, the manager could decide to allocate resources towards training and recruiting in these skill areas. All within a few minutes.

Improve Accuracy

Business leaders must make timely and *informed* decisions. However, collecting and efficiently reviewing all of the numbers is not always a reasonable option. What happens when you don't have all of the facts before making a decision? You fill in the blanks with assumptions and biases.

Simplify Communication

A decision is just words until it is carried out through people's actions. After you make a decision, you must effectively communicate your thoughts with the people who will carry out the subsequent steps. In the same way that data visualization simplifies data analysis, it can also streamline and objectify communication.

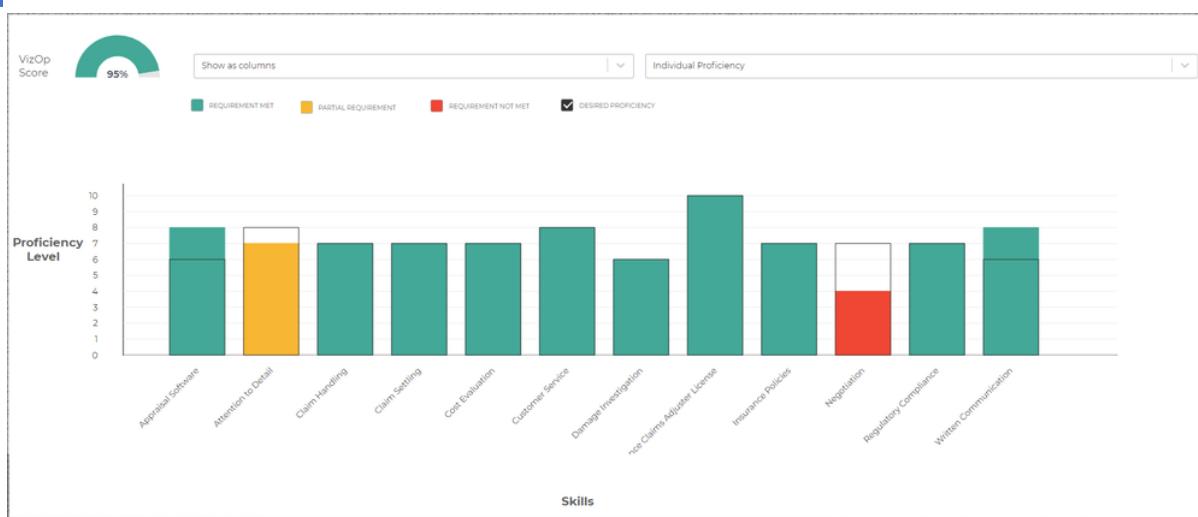
For example, a manager at an insurance company determines his lead adjuster needs to improve his negotiation skills (red column) and assigns him to training. The manager could just say this to his adjuster. However, now, the adjuster is left with several unanswered questions. Why does he need training? Where should his negotiation skills be right now? Does his manager dislike him?

While the decision may be apparent to the manager, it isn't communicated clearly to the adjuster. How can you expect people to take effective action when the message isn't clear in the first place?

Alternatively, the manager could use the graph below to clearly communicate why he is making this decision to the adjuster:

Data Analytics

Student Material



The chart clearly shows which skills do not meet the ideal proficiency levels and by how much those skills need to improve. By presenting his message in visual form, the manager can ensure the adjuster understands why he needs training and how he can gauge his progress. The visualization shifted the manager's message from unclear and subjective to concise and objective.

Empower Collaboration

Leaders should always seek the perspectives of others to enhance their decision-making process. Research shows that "being able to view the decision environment from multiple perspectives enhances the decision maker's ability to make better-informed choices."

1.9 Data Science

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from noisy, structured and unstructured data, and apply knowledge and actionable insights from data across a broad range of application domains. Data science is related to data mining, machine learning and big data.

Data science is a "concept to unify statistics, data analysis, informatics, and their related methods" in order to "understand and analyze actual phenomena" with data. It uses techniques and theories drawn from many fields within the context of mathematics, statistics, computer science, information science, and domain knowledge. However, data science is different from computer science and information science. Turing Award winner Jim Gray

imagined data science as a "fourth paradigm" of science (empirical, theoretical, computational, and now data-driven) and asserted that "everything about science is changing because of the impact of information technology" and the data deluge.

1.9.1 How Does Data Science Work?

Data science involves a plethora of disciplines and expertise areas to produce a holistic, thorough and refined look into raw data. Data scientists must be skilled in everything from

data engineering, math, statistics, advanced computing and visualizations to be able to effectively sift through muddled masses of information and communicate only the most vital bits that will help drive innovation and efficiency.

Data scientists also rely heavily on artificial intelligence, especially its subfields of machine learning and deep learning, to create models and make predictions using algorithms and other techniques.

Data science generally has a five-stage life cycle that consists of:

- Capture: Data acquisition, data entry, signal reception, data extraction
- Maintain: Data warehousing, data cleansing, data staging, data processing, data architecture
- Process: Data mining, clustering/classification, data Modelling, data summarization
- Communicate: Data reporting, data visualization, business intelligence, decision making
- Analyse: Exploratory/confirmatory, predictive analysis, regression, text mining, qualitative analysis

1.9.2 WHAT CAN DATA SCIENCE BE USED FOR?

- Anomaly detection (fraud, disease, crime, etc.)
- Automation and decision-making (background checks, credit worthiness, etc.)
- Classifications (in an email server, this could mean classifying emails as “important” or “junk”)
- Forecasting (sales, revenue and customer retention)
- Pattern detection (weather patterns, financial market patterns, etc.)
- Recognition (facial, voice, text, etc.)
- Recommendations (based on learned preferences, recommendation engines can refer you to movies, restaurants and books you may like)

1.10 Machine Learning:

Data analytics, also known as data analysis, is the process of cleaning, inspecting, modelling, and transforming data for finding valuable information, informing conclusions and enhancing the decision-making process. Data analytics focuses on generating valuable insights from the available data. Companies use data analytics to make better-informed decisions regarding various matters including marketing, production, etc. Data analytics helps you take raw data and extract helpful information from the same.

1.10.1 How machine learning works:

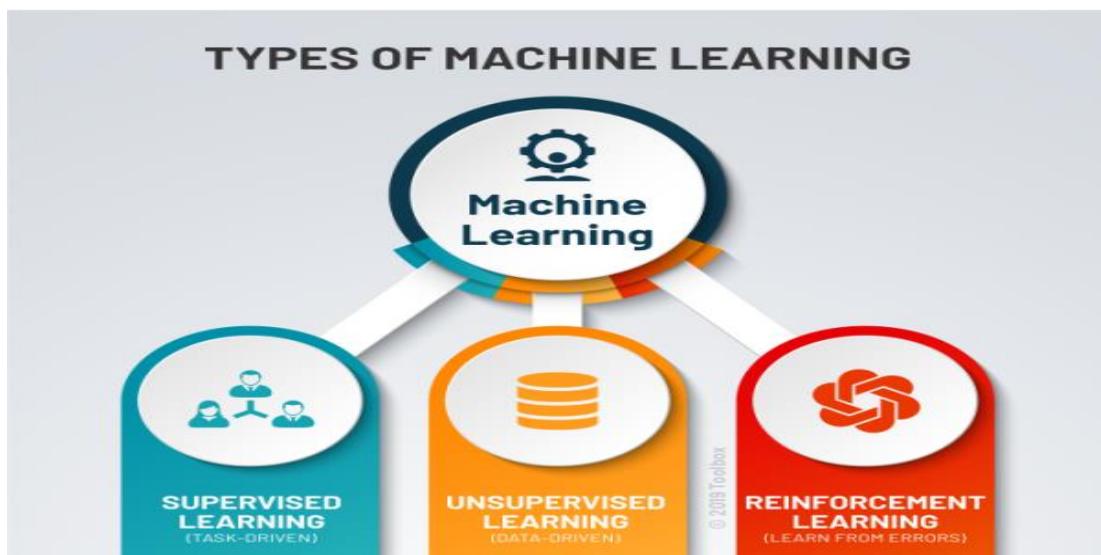
UC Berkeley (link resides outside IBM) breaks out the learning system of a machine learning algorithm into three main parts.

A Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labelled or unlabeled, your algorithm will produce an estimate about a pattern in the data.

An Error Function: An error function serves to evaluate the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

A Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this evaluate and optimize process, updating weights autonomously until a threshold of accuracy has been met.

1.10.2 Types of Machine Learning:



Supervised learning:

Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data. Even though the data needs to be labeled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.

In supervised learning, the ML algorithm is given a small training dataset to work with. This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with. The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labeled parameters required for the problem.

Unsupervised learning:

Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.

In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.

Reinforcement learning:

Reinforcement learning directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged, or ‘reinforced’, and non-favorable outputs are discouraged or ‘punished’. Based on the psychological concept of conditioning, reinforcement learning works by putting the algorithm in a work environment with an interpreter and a reward system. In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favorable or not. In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favorable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result. In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. The higher this percentage value is, the more reward is given to the algorithm. Thus, the program is trained to give the best possible solution for the best possible reward.

1.10.3 Real-world machine learning use cases:

Here are just a few examples of machine learning you might encounter every day:

- Speech recognition
- Customer service
- Computer vision
- Recommendation engines
- Automated stock trading

1.10.4 Challenges of machine learning:

As machine learning technology advances, it has certainly made our lives easier. However, implementing machine learning within businesses has also raised a number of ethical concerns surrounding AI technologies.

Some of these include:

- Technological singularity
- AI impact on jobs
- Privacy

- Bias and discrimination
- Accountability

1.10.5 Machine Learning vs. Deep Learning vs. Neural Networks:

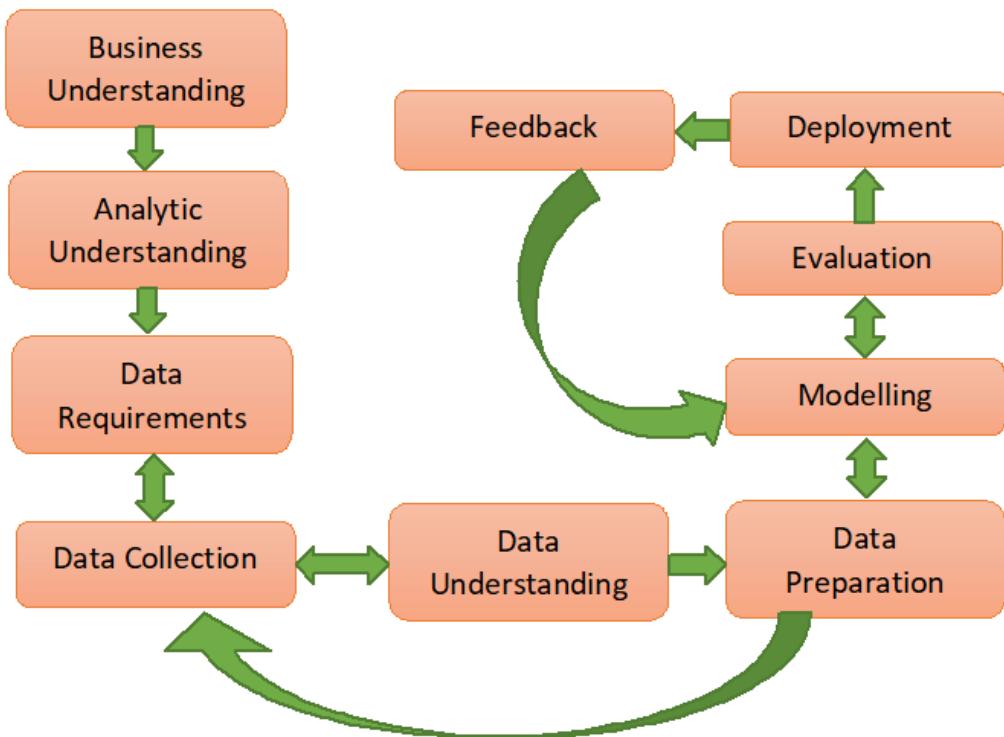
Since deep learning and machine learning tend to be used interchangeably, it's worth noting the nuances between the two. Machine learning, deep learning, and neural networks are all sub-fields of artificial intelligence. However, deep learning is actually a sub-field of machine learning, and neural networks is a sub-field of deep learning.

"Deep" machine learning can leverage labelled datasets, also known as supervised learning, to inform its algorithm, but it doesn't necessarily require a labelled dataset. It can ingest unstructured data in its raw form (e.g., text, images), and it can automatically determine the set of features which distinguish different categories of data from one another. Unlike machine learning, it doesn't require human intervention to process data, allowing us to scale machine learning in more interesting ways. Deep learning and neural networks are primarily credited with accelerating progress in areas, such as computer vision, natural language processing, and speech recognition.

Neural networks, or artificial neural networks (ANNs), are comprised of a node layer, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. The "deep" in deep learning is just referring to the depth of layers in a neural network. A neural network that consists of more than three layers—which would be inclusive of the inputs and the output—can be considered a deep learning algorithm or a deep neural network. A neural network that only has two or three layers is just a basic neural network.

1.11 Data Science Methodology

Data Science Methodology indicates the routine for finding solutions to a specific problem. This is a cyclic process that undergoes a critic behaviour guiding business analysts and data scientists to act accordingly.



1. Business Understanding
2. Analytic Understanding
3. Data Requirements
4. Data Collection
5. Data Understanding
6. Data Preparation
7. Modelling
8. Evaluation
9. Deployment
10. Feedback

1.12 Data Analytics in Different Sectors:

The concept of data analytics has such potential that its application should not be restricted within the IT sectors only. The main factor of data analytics is that it provides a new way to gain insights into the challenges businesses face every day. Previously, no company could collect and analyze vast quantities of data. The necessity of analyzing such a long array of data is to analyze the condition of the company and predict the future strategies that would bring success. At present, a business generates all sorts of data – from machine data associated with the servers and networks to emails to customer purchasing trends. If the company aims to stay competitive and lean, they need to leverage that data to help move the business forward and using tools of data analytics is the easiest way to do that.

Data Analytics

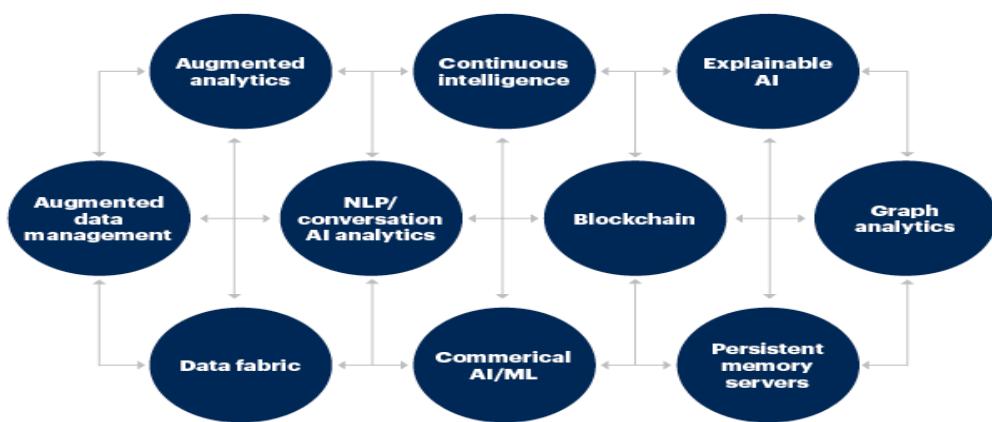
Student Material



1. Retail
2. Medicine
3. Banking and Finance
4. Construction
5. Transportation
6. Communications, Media, and Entertainment
7. Education
8. Manufacturing and Natural Resources
9. Government
10. Energy and Utilities
11. Outsourcing Industry

1.13 Analytics Framework and Latest trends:

Top 10 technology trends in data and analytics



Trend 1: Smarter and Scalable Artificial Intelligence

Trend 2: Agile and Composed Data & Analytics

Trend 3: Hybrid Cloud Solutions and Cloud Computing

Trend 4: Data Fabric

Trend 5: Edge Computing For Faster Analysis

Trend 6: Augmented Analytics

Trend 7: The Death of Predefined Dashboards

Trend 8: XOps

Trend 9: Engineered Decision Intelligence

Trend 10: Data Visualization

Chapter: 2

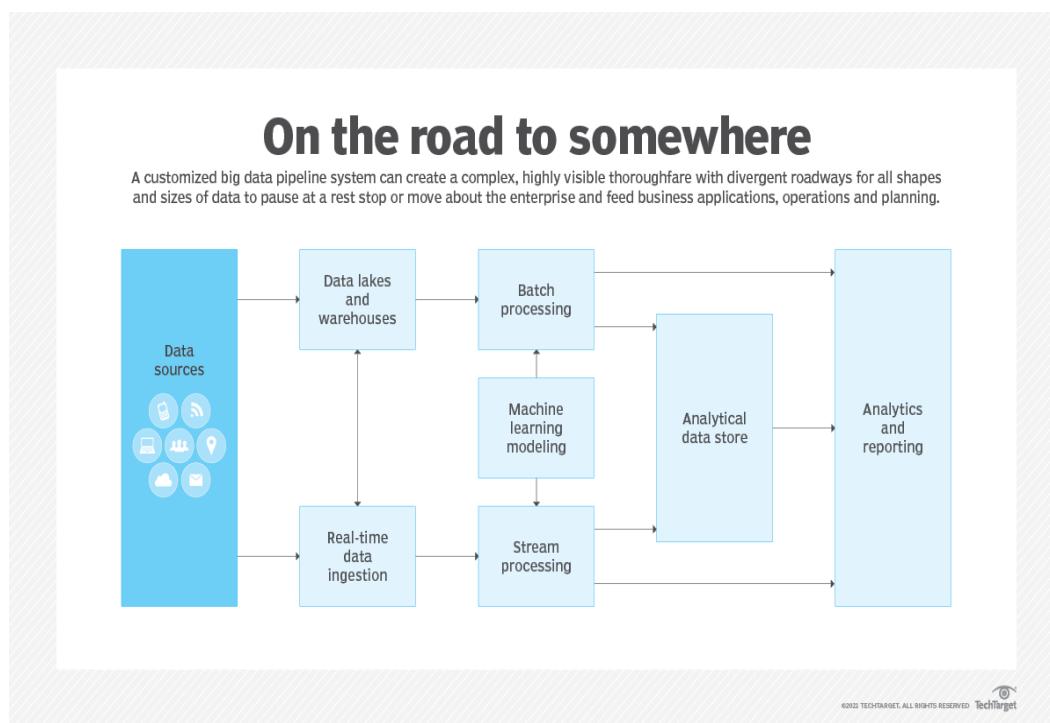
Data Ingestion

Scope:

- Power BI Data Ingestion and Modelling
- Reading from MySQL
- Reading from HDF5
- Reading from The Web
- Reading from APIs
- ETL
- Reading from Other Sources
- Summary and close

2.1 Data ingestion

It is the process of obtaining and importing data for immediate use or storage in a database. To ingest something is to take something in or absorb something. Data can be streamed in real time or ingested in batches. In real-time data ingestion, each data item is imported as the source emits it. When data is ingested in batches, data items are imported in discrete chunks at periodic intervals of time. The first step in an effective data ingestion process is to prioritize the data sources. Individual files must be validated and data items routed to the correct destinations. When numerous big data sources exist in diverse formats, the sources may number in the hundreds and the formats in the dozens. In this situation, it is challenging to ingest data at a reasonable speed and process it efficiently. To that end, vendors offer software to automate the process and tailor it to specific computing environments and applications. When data ingestion is automated, the software used may include data preparation capabilities. These features structure and organize data so it can be analyzed right away or later using business intelligence (BI) and business analytics programs.



Types of data ingestion

Batch processing. In batch processing, the ingestion layer collects data from sources incrementally and sends batches to the application or system where the data is to be used or stored. Data can be grouped based on a schedule or criteria, such as if certain conditions are

Data Analytics

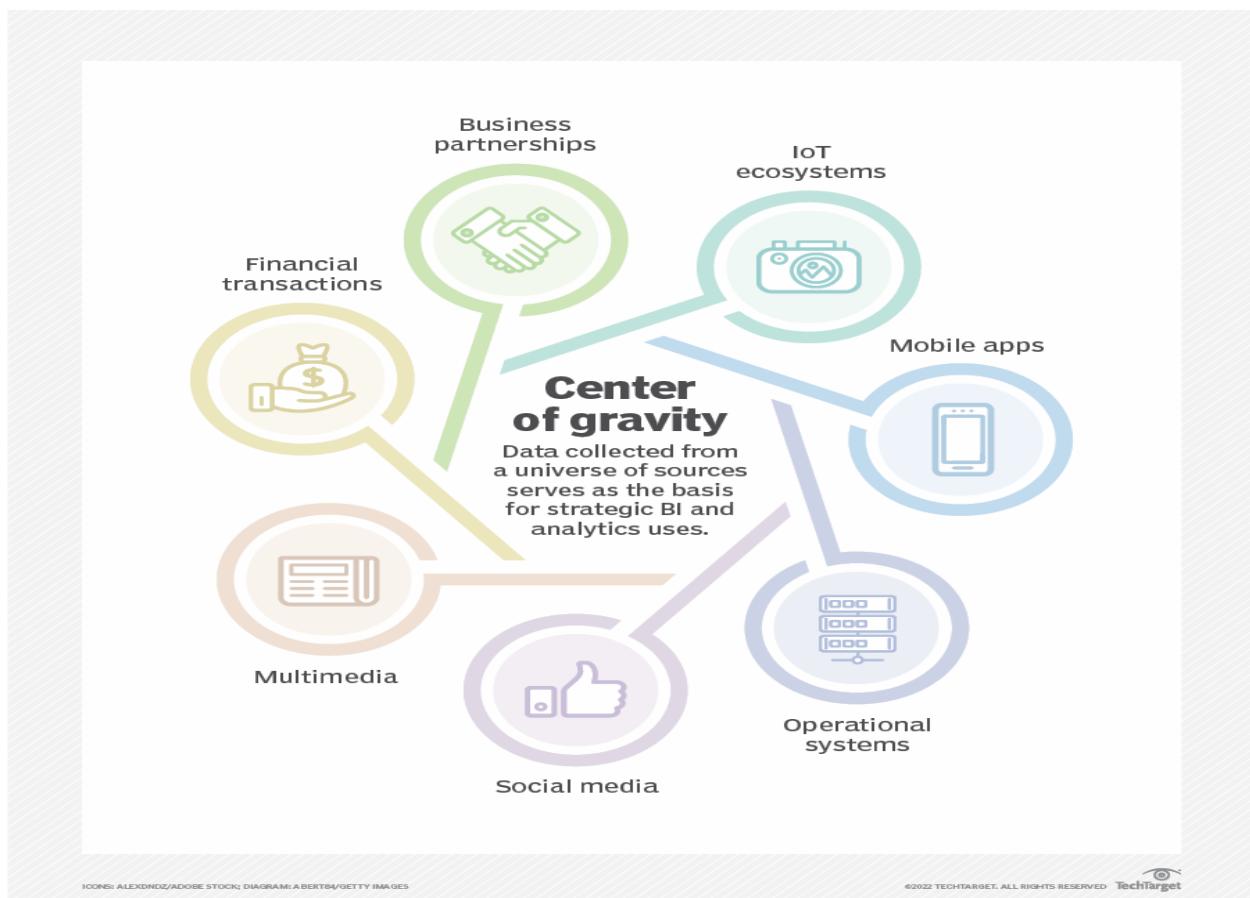
Student Material

triggered. This approach is good for applications that don't require real-time data. It is typically less expensive.

Real-time processing. This type of data ingestion is also referred to as stream processing. Data is not grouped in any way in real-time processing. Instead, each piece of data is loaded as soon as it is recognized by the ingestion layer and is processed as an individual object. Applications that require real-time data should use this approach.

Micro batching. This is a type of batch processing that streaming systems like Apache Spark Streaming use. It divides data into groups, but ingests them in smaller increments that make it more suitable for applications that require streaming data.

The components of an organization's data strategy and its business requirements determine the data ingestion method it uses. Organizations choose their model and data ingestion tools in part based on the data sources they use and how quickly they will need access to the data for analysis.



Data ingestion tools and features

Data ingestion tools come with a range of capabilities and features, including the following:

- Extraction.
- Processing.

- Data types.
- Data flow tracking and visualization.
- Volume.
- Security and privacy.

Benefits of data ingestion

Data ingestion technology offers the following benefits to the data management process:

- Flexibility.
- Simplicity.
- Analytics.
- Application quality.
- Availability.
- Decision-making.

Challenges of data ingestion and big data sets

Data ingestion also poses challenges to the data analytics process, including the following:

- Scale.
- Security.
- Fragmentation and data integration.
- Data quality.
- Costs.

What is Power BI?

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. Your data may be an Excel spreadsheet, or a collection of cloud-based and on-premises hybrid data warehouses. Power BI lets you easily connect to your data sources, visualize and discover what's important, and share that with anyone or everyone you want.

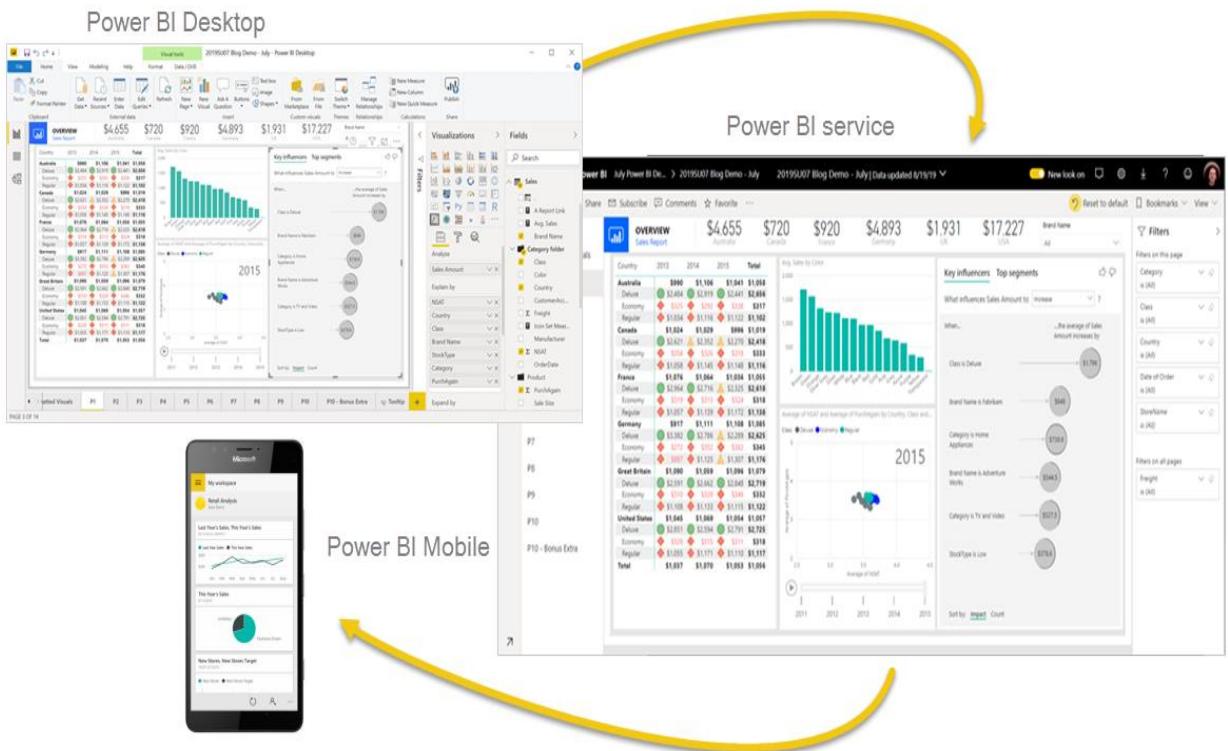
The parts of Power BI

Power BI consists of several elements that all work together, starting with these three basics:

- A Windows desktop application called Power BI Desktop.
- An online SaaS (Software as a Service) service called the Power BI service.
- Power BI mobile apps for Windows, iOS, and Android devices.

Data Analytics

Student Material



These three elements—Power BI Desktop, the service, and the mobile apps—are designed to let you create, share, and consume business insights in the way that serves you and your role most effectively.

Beyond those three, Power BI also features two other elements:

- Power BI Report Builder, for creating paginated reports to share in the Power BI service. Read more about paginated reports later in this article.
- Power BI Report Server, an on-premises report server where you can publish your Power BI reports, after creating them in Power BI Desktop. Read more about Power BI Report Server later in this article.

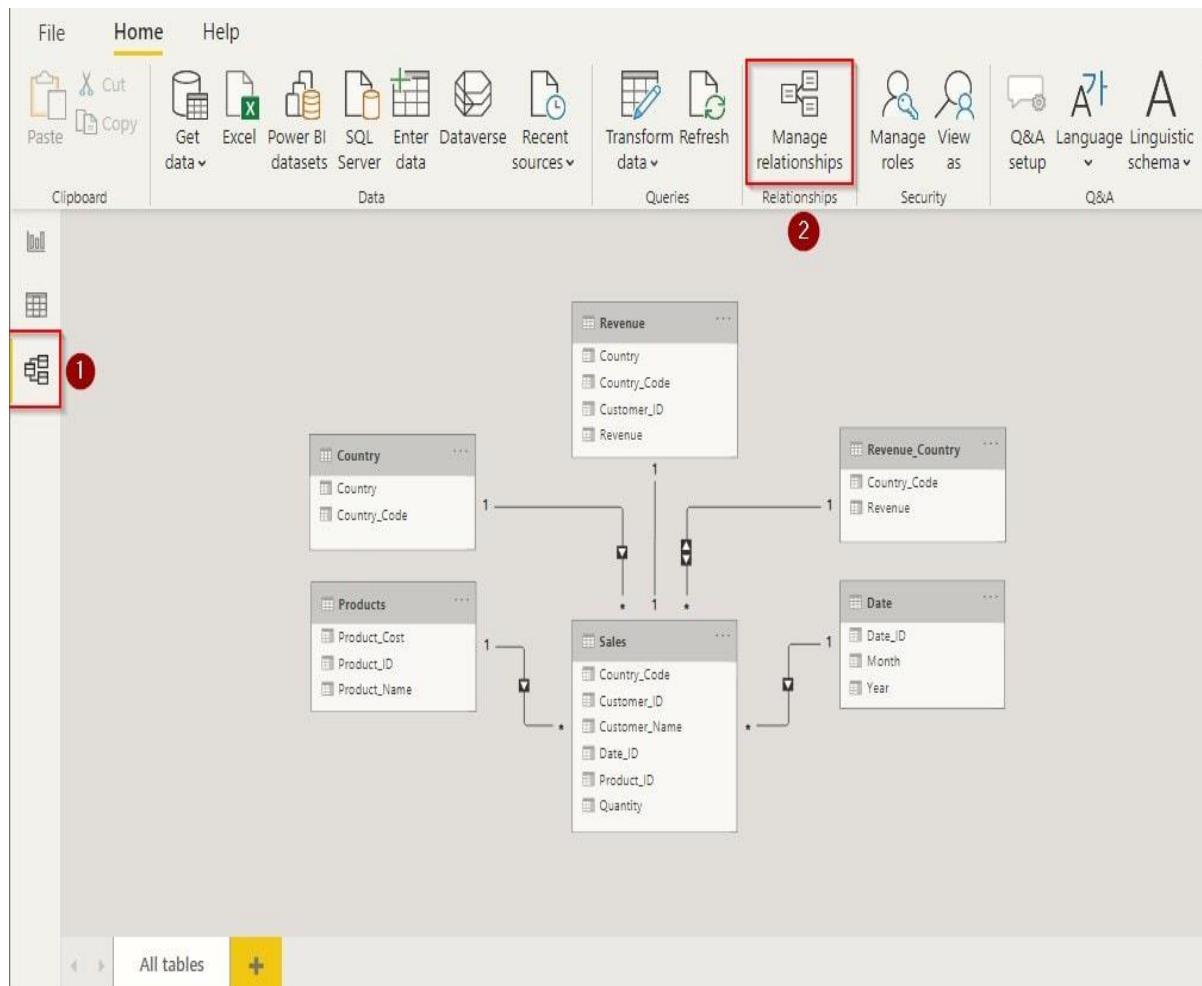
Data Modelling on Power BI

Data Models in Power BI:

Power BI automatically detect all the possible relations between different sets of data. Sometimes we need to create a relation between the data manually.

Data Analytics

Student Material



After the data import is complete, visit the 'Models' tab from the left side, as highlighted in the above image. Here you will see some automated relation created by Power BI. All the lines visible here in the 'Models' tab depicts the cardinality and direction of the relation from one table to another. You can create & modify this default relationship created by Power BI using the Manage Relationship tool given on the top menu as highlighted in the image.

Create and Manage Relationship

Manage relationships

Active	From: Table (Column)	To: Table (Column)
<input checked="" type="checkbox"/>	Sales (Country_Code)	Country (Country_Code)
<input checked="" type="checkbox"/>	Sales (Date_ID)	Date (Date_ID)
<input checked="" type="checkbox"/>	Sales (Product_ID)	Products (Product_ID)

At the bottom, there are four buttons labeled 1 through 4: New..., Autodetect..., Edit..., and Delete. A 'Close' button is also present.

Data Analytics

Student Material

After clicking on ‘Manage Relationship’, a similar screen will appear, as shown in the above image. You can see all the active relations here from one table to another. All the things you can do with these relations are explained below:

- New – This option will help you create a new relationship between tables.
- Autodetect – Using this option, Power BI automatically detects the relationship between data present in tables.
- Edit – This option will help you to edit your data relationship.
- Delete – It deletes the selected relationship between the tables.

Edit relationship

Select tables and columns that are related.

The screenshot shows the 'Edit relationship' dialog box. At the top, there are dropdown menus for selecting tables: 'Sales' (marked with a red circle 1) and 'Products' (marked with a red circle 2). Below each table is its respective data grid. Underneath the tables are two main configuration sections: 'Cardinality' (set to 'Many to one (:1)', marked with a red circle 3) and 'Cross filter direction' (set to 'Single', marked with a red circle 4). There are also two checkboxes: 'Make this relationship active' (checked) and 'Assume referential integrity' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons.

Now after clicking on ‘New’ or ‘Edit’, a similar screen will appear in front of you, as shown above. Here we will explain to you the purpose of all the options one by one.

Calculate and Measure Data

If you are familiar with Excel, you may have worked on the DAX (Data Analysis Expression) formula. If not, then no worries. I will explain in short and simple words. Like programming, DAX is a set of instructions used to calculate data from the tables. These expressions include commands for Addition, Multiplication, Average, Percentage and others with various filters.



Brought to you by:



Data Analytics

Student Material

Create Table

1 Revenue_Country = DISTINCT(Country[Country_Code])

1 2 3

Country_Code
IN
FR
US
IT
MY
UK
AU
TH
NZ
RU
JP
KW

Create Column

Click on 'New Column' from the top menu to create a calculated column.

1 Revenue = CALCULATE(SUM(Revenue[Revenue]), ALLSELECTED(Revenue[Country]))

1

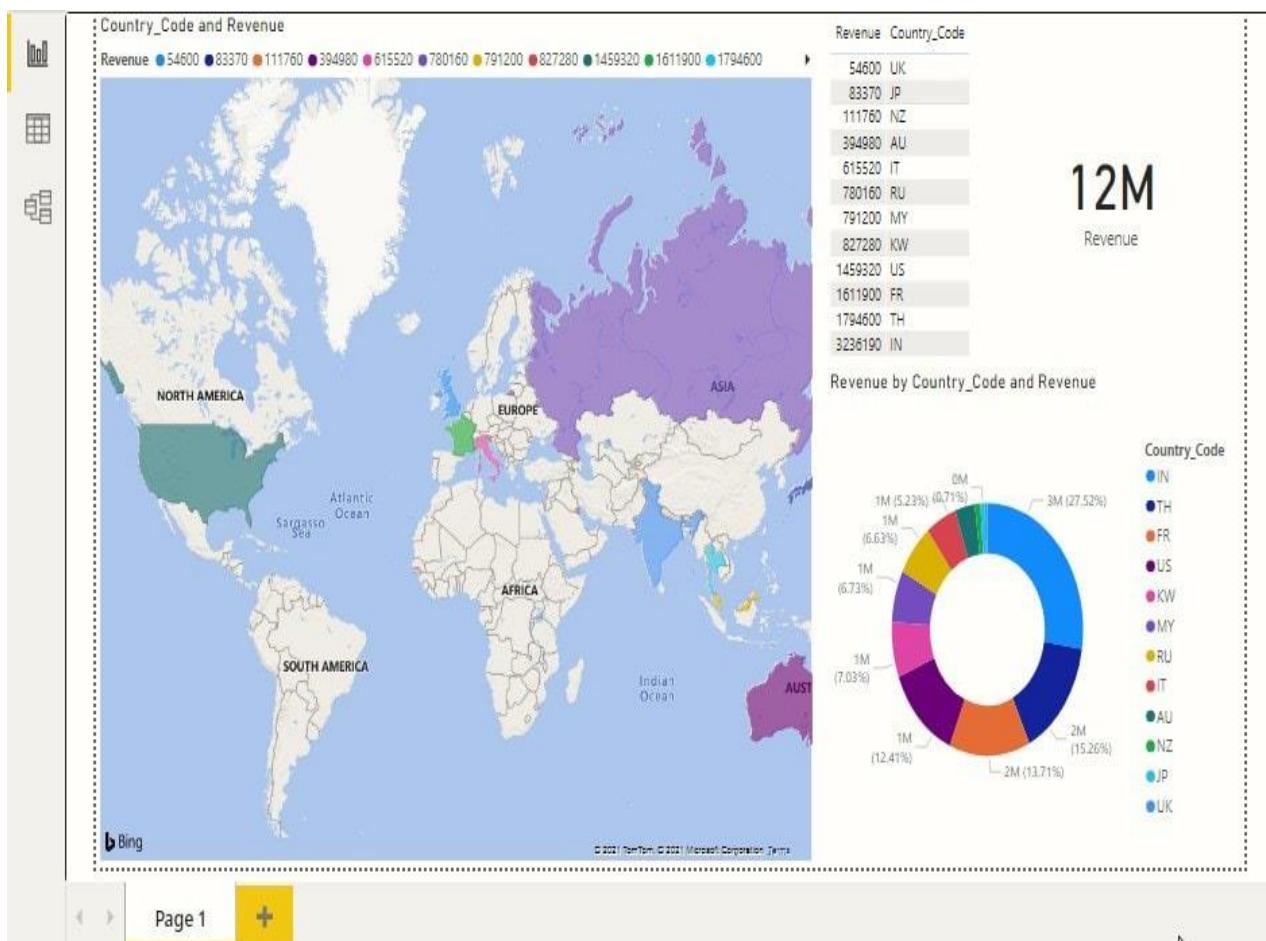
Country_Code	Revenue
IN	3236190
FR	1611900
US	1459320
IT	615520
MY	791200
UK	54600
AU	394980
TH	1794600
NZ	111760
RU	780160
JP	83370
KW	827280

The DAX expression will calculate all the revenue from the table 'Revenue' with filter as 'Country'. Without this expression, we might have spent hours calculating the individual revenue generated from the country.

Data Analytics

Student Material

Create Visualization

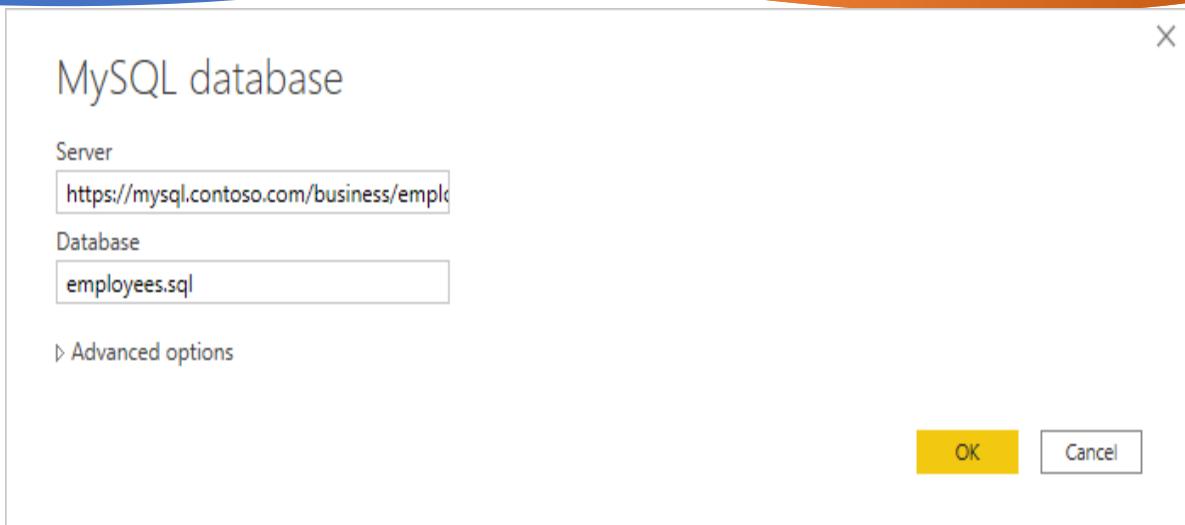


We only had sales, sales by country and product price in our imported data. Using the above methods, we created a separate table that contains the revenue generated from each country. Here we have shown the revenue from the country on the world map using different visualizations. Similarly, you can also create and manage data models in Power BI.

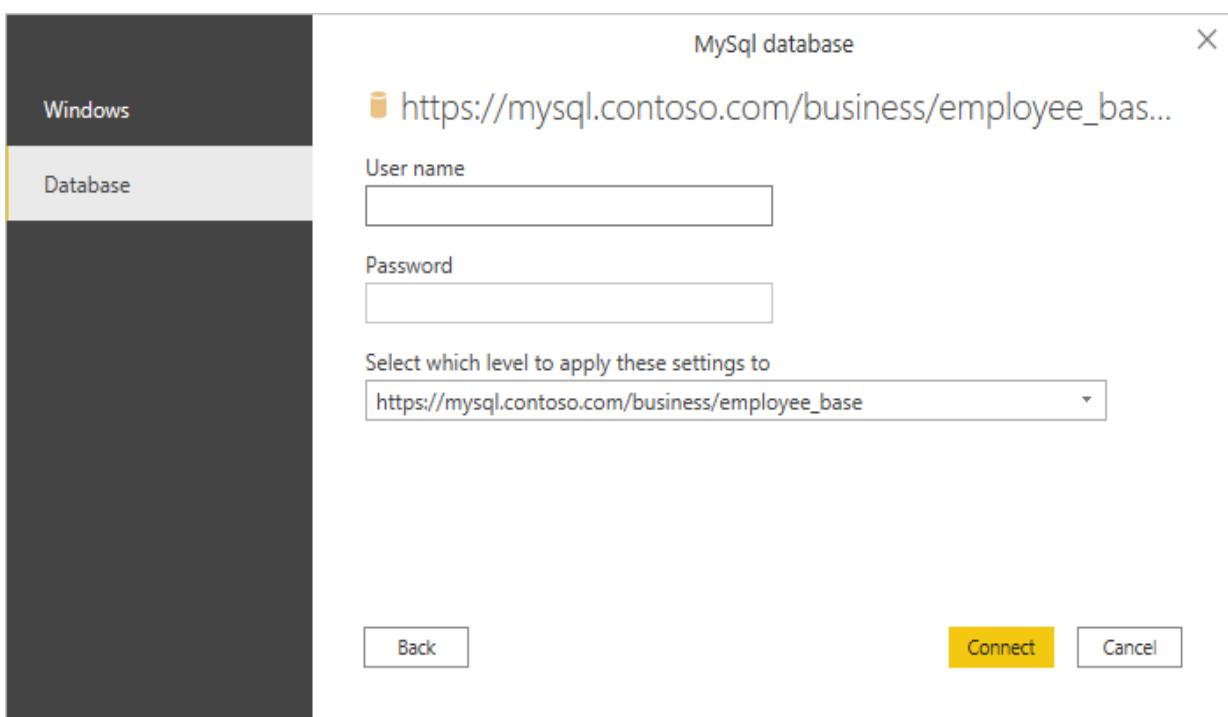
2.2 READING FROM MYSQL

To make the connection, take the following steps:

1. Select the **MySQL database** option in the connector selection.
2. In the **MySQL database** dialog, provide the name of the server and database.



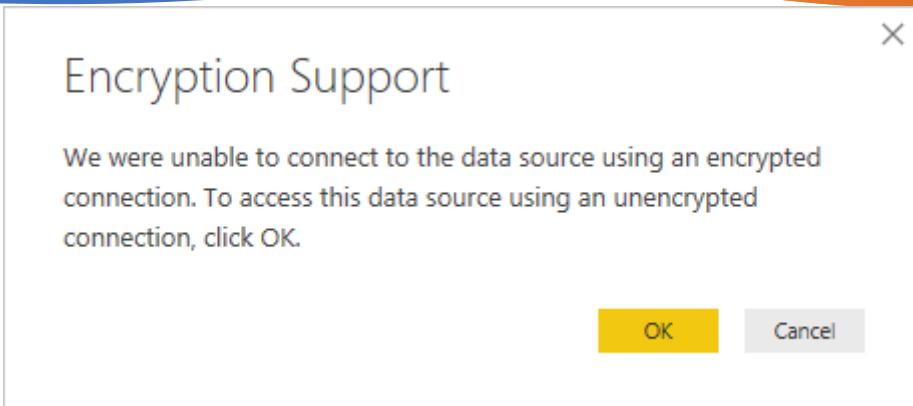
3. Select the **Database** authentication type and input your MySQL credentials in the **User name** and **Password** boxes.



4. Select the level to apply your credentials to.
5. Once you're done, select **OK**.

Note

If the connection is not encrypted, you'll be prompted with the following dialog.



Select **OK** to connect to the database by using an unencrypted connection, or follow the instructions to set up encrypted connections to SQL Server.

6. In **Navigator**, select the data you require, then either load or transform the data.

The screenshot shows the Power BI Navigator window. On the left, there's a tree view of tables from a MySQL database at the URL <https://mysql.contoso.com/business/employ...>. The 'employees.departments' table is selected, indicated by a checked checkbox next to it. On the right, a preview of the 'employees.departments' table is displayed in a grid format. The columns are: dept_no, dept_name, employees.dept_emp, and employees.dept_manager. The data rows are: d001 (Marketing, Table, Table), d002 (Finance, Table, Table), d003 (Human Resources, Table, Table), d004 (Production, Table, Table), d005 (Development, Table, Table), d006 (Quality Management, Table, Table), d007 (Sales, Table, Table), d008 (Research, Table, Table), and d009 (Customer Service, Table, Table). At the bottom of the window are buttons for 'Select Related Tables', 'Load', 'Transform Data', and 'Cancel'.

dept_no	dept_name	employees.dept_emp	employees.dept_manager
d001	Marketing	Table	Table
d002	Finance	Table	Table
d003	Human Resources	Table	Table
d004	Production	Table	Table
d005	Development	Table	Table
d006	Quality Management	Table	Table
d007	Sales	Table	Table
d008	Research	Table	Table
d009	Customer Service	Table	Table

Connect to MySQL database from Power Query Online

To make the connection, take the following steps:

1. Select the **MySQL database** option in the connector selection.
2. In the **MySQL database** dialog, provide the name of the server and database.

The screenshot shows the 'Connection settings' and 'Connection credentials' sections for a MySQL database. In the 'Connection settings' section, the 'Server' is set to https://mysql.contoso.com/business/employee_... and the 'Database' is set to 'employees.sql'. In the 'Connection credentials' section, the 'On-premises data gateway' is set to '[Admin] 10TestGateway' and the 'Authentication kind' is set to 'Basic'. There are empty fields for 'Username' and 'Password'. A checked checkbox labeled 'Use Encrypted Connection' is at the bottom.

3. If necessary, include the name of your on-premises data gateway.
4. Select the **Basic** authentication kind and input your MySQL credentials in the **Username** and **Password** boxes.
5. If your connection isn't encrypted, clear **Use Encrypted Connection**.
6. Select **Next** to connect to the database.
7. In **Navigator**, select the data you require, then select **Transform data** to transform the data in Power Query Editor.

2.3 READING FROM HDF5 FILES

HDF5 file stands for Hierarchical Data Format 5. It is an open-source file which comes in handy to store large amount of data. As the name suggests, it stores data in a hierarchical structure within a single file. So if we want to quickly access a particular part of the file rather than the whole file, we can easily do that using HDF5. This functionality is not seen in normal text files hence HDF5 is becoming seemingly popular in fact of being a new concept.

Let's now take a look at the most common problems that you may face if you do not clean the messy data (w.r.t to the above-mentioned types).

1. Data aggregation

Suppose you have null entries for a numeric column and you are calculating summary statistics (like mean, maximum, minimum values) on that column. The results will not get conveyed accurately in this case. Again consider the *Pima Indian Diabetes* dataset with the invalid zero entries. If you calculated summary statistics on the columns as mentioned before,

Data Analytics

Student Material

would you get the right results? Won't the results be erroneous? So, how to address this problem? There are several ways:

- Removing the entries containing missing/nullvalues(not recommended)
- Imputing the null entries with a numeric value(typically with mean or median of the respective column)

Let's now get hands-on with these problems and the second option for combating null values.

Consider the following PostgreSQL table named entries:

name text	weight_in_lbs double precision	age_in_years smallint
Christina	80.6	
Matthews		19
Gilbert	100.3	21

You can see two null entries in the above table. Suppose you want to get the average weight value from this table and you executed the following query:

```
select avg(weight_in_lbs) as average_weight_in_lbs from entries;
```

You got 90.45 as the output. Is this correct? So, what can be done? Let's fill the null entry with this average value with the help of the COALESCE() function.

Let's fill the missing values first with COALESCE() (remember that COALESCE() does not change the values in the original table, it just returns a temporary view of the table with the values changed):

```
select *, COALESCE(weight_in_lbs, 90.45) as corrected_weights from entries;
```

You should get an output like:

name text	weight_in_lbs double precision	age_in_years smallint	corrected_weights double precision
Christina	80.6		80.6
Matthews		19	90.45
Gilbert	100.3	21	100.3

Now you can apply the AVG() again:

```
select avg(corrected_weights) from
```

```
(select *, COALESCE(weight_in_lbs, 90.45) as corrected_weights from entries) as subquery;
```

This is a much more accurate result than the earlier one. Let's now study another problem that can take place if you have mismatches in the column data-types.

2. Table Joins

Consider you are working with the following tables student_metadata and department_details:

Data Analytics

Student Material

id smallint	name text	dept_id smallint
1	Rick	100
2	Twinkle	100
3	Steve	101

student_metadata

dept_id text	dept_name text	hod text
100	Computer Science	Chris Manning
101	Economics	Steve Harris

department_details

You can see in the student_metadata table, dept_id is of integer type, and in the department_details table, it is present in text type. Now, suppose, you want to join these two tables and want to produce a report which will contain the following columns:

1. id
2. name
3. dept_name

To do this, you run this query:

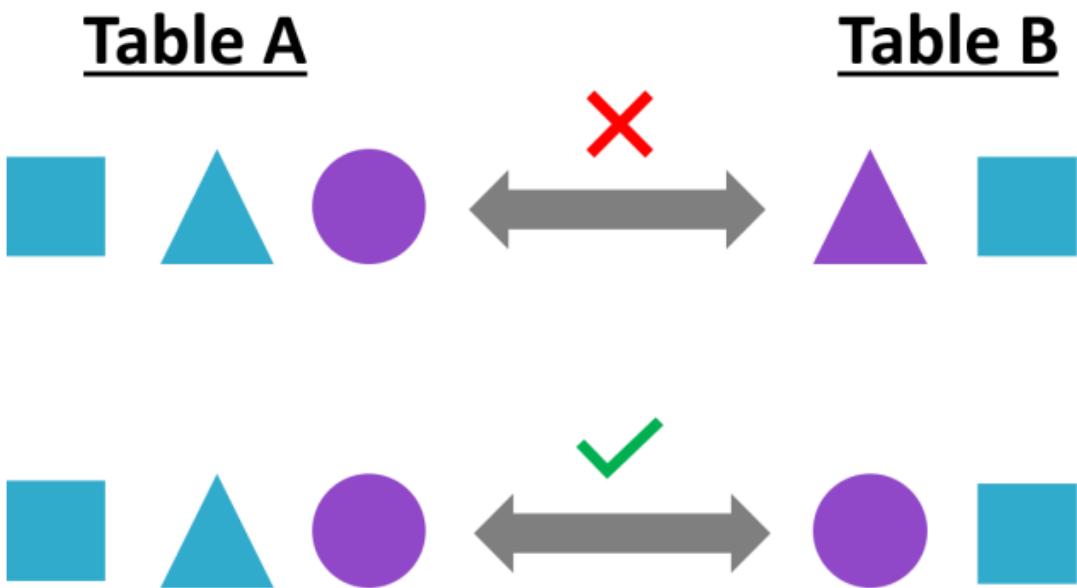
```
select id, name, dept_name from  
student_metadata s join department_details d
```

on s.dept_id = d.dept_id;

You will encounter this error, then:

ERROR: operator does not exist: smallint = text

Here's an amazing infographic which depicts this problem (from DataCamp's [Reporting in SQL](#) course):



Data Analytics

Student Material

This happening because the data-types are not getting matched while joining the two tables. Here, you can CAST the dept_id column in the department_details table to integer while joining the tables. Here's how to do that:

```
select id, name, dept_name from  
student_metadata s join department_details d  
on s.dept_id = cast(d.dept_id as smallint);
```

And you get your desired report:

id smallint	name text	dept_name text
1	Rick	Computer Science
2	Twinkle	Computer Science
3	Steve	Economics

Let's now discuss how strings can be present in messy forms, their problems and ways to deal with them.

Messy strings and cleaning them

String values are also very common. Let's start this section by looking at the values of a column dept_name (denoting department names) taken from a table named student_details:

id smallint	name text	dept_name text
1	Sayak	I.T
2	Hugo	Information Technology
2	Stephen	i.t
4	David	C.S.E
5	Andrew	C.S.E
6	Emily	E.C.E

String values like the above can cause a lot of unexpected problems. I.T, Information Technology, and i.t all mean the same department, i.e. *Information Technology* and suppose the specification document requires the values to be present as I.T only. Now, say, you want to count the number of students belonging to the department of I.T., and you run this query:

```
select dept_name, count(dept_name) as student_count  
from student_details  
group by dept_name;
```

And you get:

dept_name text	student_count bigint
Information Technology	1
i.t	1
C.S.E	2
I.T	1
E.C.E	1

Data Analytics

Student Material

Is this an accurate report? - No! So, how can you address this problem?

Let's first identify the problem in a more detailed way:

- You have Information Technology as a value which should be converted to I.T and
- You have i.t as another value which should be converted to I.T.

In the first case, you can REPLACE the value Information Technology to I.T, and in the second case, you convert the character to UPPER case. You can accomplish this in a single query though it is advised to address this kind of problems in a step-by-step fashion. Here's the query to address the problem:

```
select upper(replace(dept_name, 'Information Technology', 'I.T')) as dept_cleaned,  
count(dept_name) as student_count  
from student_details  
group by dept_cleaned;
```

And the report:

dept_cleaned text	student_count bigint
C.S.E	2
I.T	3
E.C.E	1

Messy dates and cleaning them

Consider you are working with a table named employees which contains a column called birthdate but not in an appropriate date type. Now, you want to execute queries with dedicated date functions like DATE_PART(). You will not be able to do that until and unless you CAST the birthdate column to date type. Let's see this in action.

Consider the birthdates to be in the YYYY-MM-DD format.

Here's what the employees table looks like:

employee_id smallint	employee_name text	dept_name text	birthdate text
10	Ramesh	Retail	1990-11-08
15	Ricky	Data & Analytics	1995-01-23
15	Richard	Customer Acquisition	1987-05-21

Now, you run the following query to extract the months from the birthdates:

```
select date_part('month', birthdate) from employees;
```

And you instantly get this error:

ERROR: function date_part(unknown, text) does not exist

Along with the error, you get a very good hint also:

HINT: No function matches the given name and argument types. You might need to add explicit type casts.

Let's follow the hint and CAST the birthdate to the appropriate date type and then apply DATE_PART():

```
select date_part('month', CAST(birthdate AS date)) as birthday_months from employees;
```

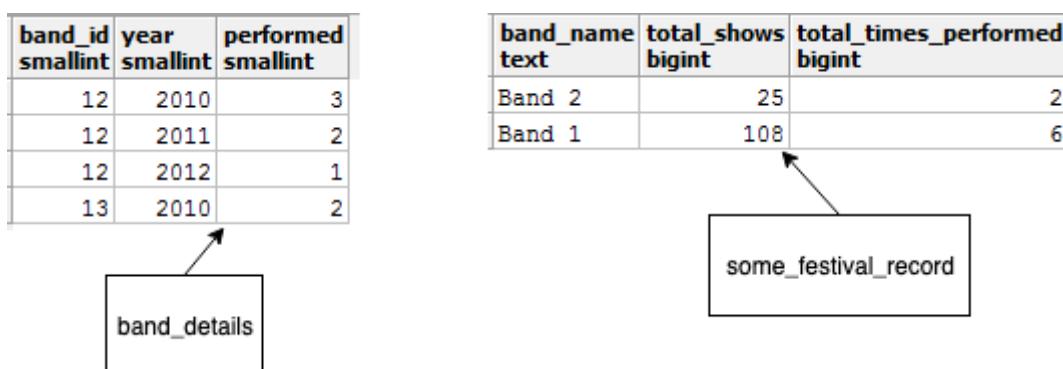
You should get the result:

birthday_months
double precision
11
1
5

Let's now proceed to the pre-concluding section of this tutorial where you will study the effects of data duplications and how you can tackle them.

Data duplications: Causes, effects, and solutions

In this section, you will be studying some of the most common causes which lead to data duplications. You will also see their effects and some of the ways using which prevent them. Consider the following two tables band_details and some_festival_record:



2.4 READING DATA FROM THE WEB

- The `urllib` module allows you to download data from web servers. Typically, you will download web-pages written in HTML that were designed for a web-browser to render (draw on-screen) for a human to read. This allows you to download useful data from web-pages, such as the current temperature, sports scores, item prices from web stores, and anything else you can find on the web.
- However, before learning how to do this, you should understand that some websites put legal (and ethical) restrictions on the data they provide. In some cases, a website intends the data to be seen by humans, but do not want a program to download the data automatically.

ETHICAL WEB SCRAPING

- If you are designing a program that downloads data from a website, you should ask the website owner for written permission to retrieve the data automatically. In many cases where a website wants you to have access to their data, they will have a web-based API that makes downloading the data easier than reading it

from an HTML page. Typically, web-based API's are formatted using JSON or XML to make the data returned easier to parse for a computer. (Although a computer can parse HTML, it can be complicated and take more work than you would like.)

- For example, the website PaperBackSwap.com allows users to trade paperback books online. They have a web accessible search form that allows you to search for a book by title, author, or ISBN number to see if a member has that book posted for swapping. You could write a web scraper to check if a book you want is available. However, it would have to parse the HTML page that PaperBackSwap.com returns, looking for the data it wants.

To load data from a web site with Power Query Desktop:

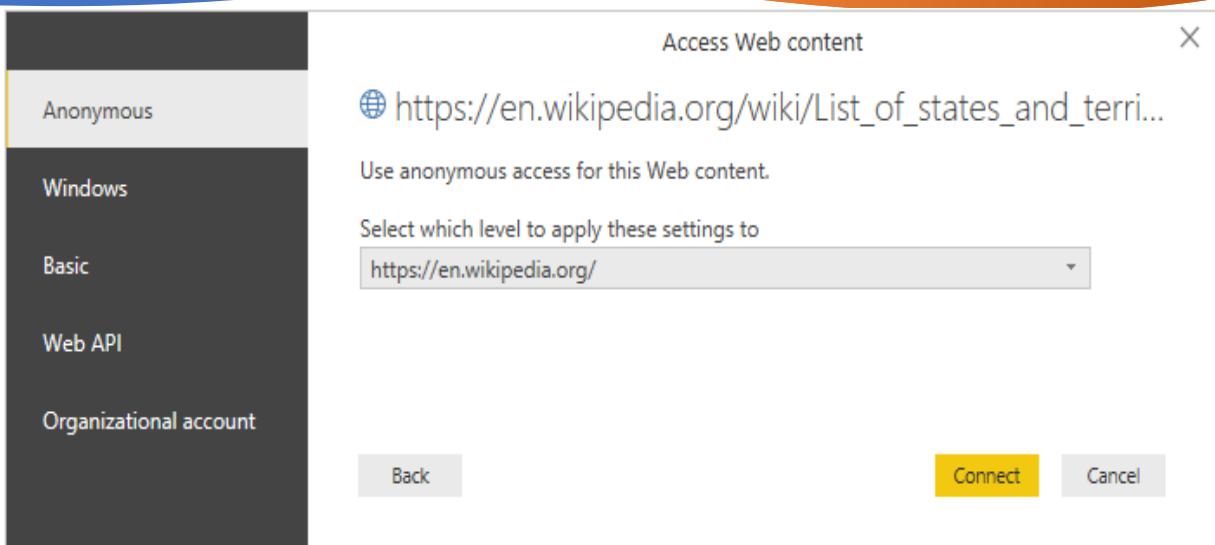
- Select **Get Data > Web** in Power BI or **From Web** in the **Data** ribbon in Excel.
- Choose the **Basic** button and enter a URL address in the text box. For example, enter https://en.wikipedia.org/wiki/List_of_states_and_territories_of_the_United_States. Then select **OK**.



If the URL address you enter is invalid, a  warning icon will appear next to the **URL** textbox.

If you need to construct a more advanced URL before you connect to the website, go to **Load Web data using an advanced URL**.

- Select the authentication method to use for this web site. In this example, select **Anonymous**. Then select the level to which you want to apply these settings to—in this case, <https://en.wikipedia.org/>. Then select **Connect**.



The available authentication methods for this connector are:

- **Anonymous:** Select this authentication method if the web page doesn't require any credentials.
- **Windows:** Select this authentication method if the web page requires your Windows credentials.
- **Basic:** Select this authentication method if the web page requires a basic user name and password.
- **Web API:** Select this method if the web resource that you're connecting to uses an API Key for authentication purposes.
- **Organizational account:** Select this authentication method if the web page requires organizational account credentials.

Note

When uploading the report to the Power BI service, only the **anonymous**, **Windows** and **basic** authentication methods are available.

The level you select for the authentication method determines what part of a URL will have the authentication method applied to it. If you select the top-level web address, the authentication method you select here will be used for that URL address or any subaddress within that address. However, you might not want to set the top URL address to a specific authentication method because different subaddresses could require different authentication methods. For example, if you were accessing two separate folders of a single SharePoint site and wanted to use different Microsoft Accounts to access each one.

Once you've set the authentication method for a specific web site address, you won't need to select the authentication method for that URL address or any subaddress again. For example, if you select the <https://en.wikipedia.org/> address in this dialog, any web page that begins with this address won't require that you select the authentication method again.

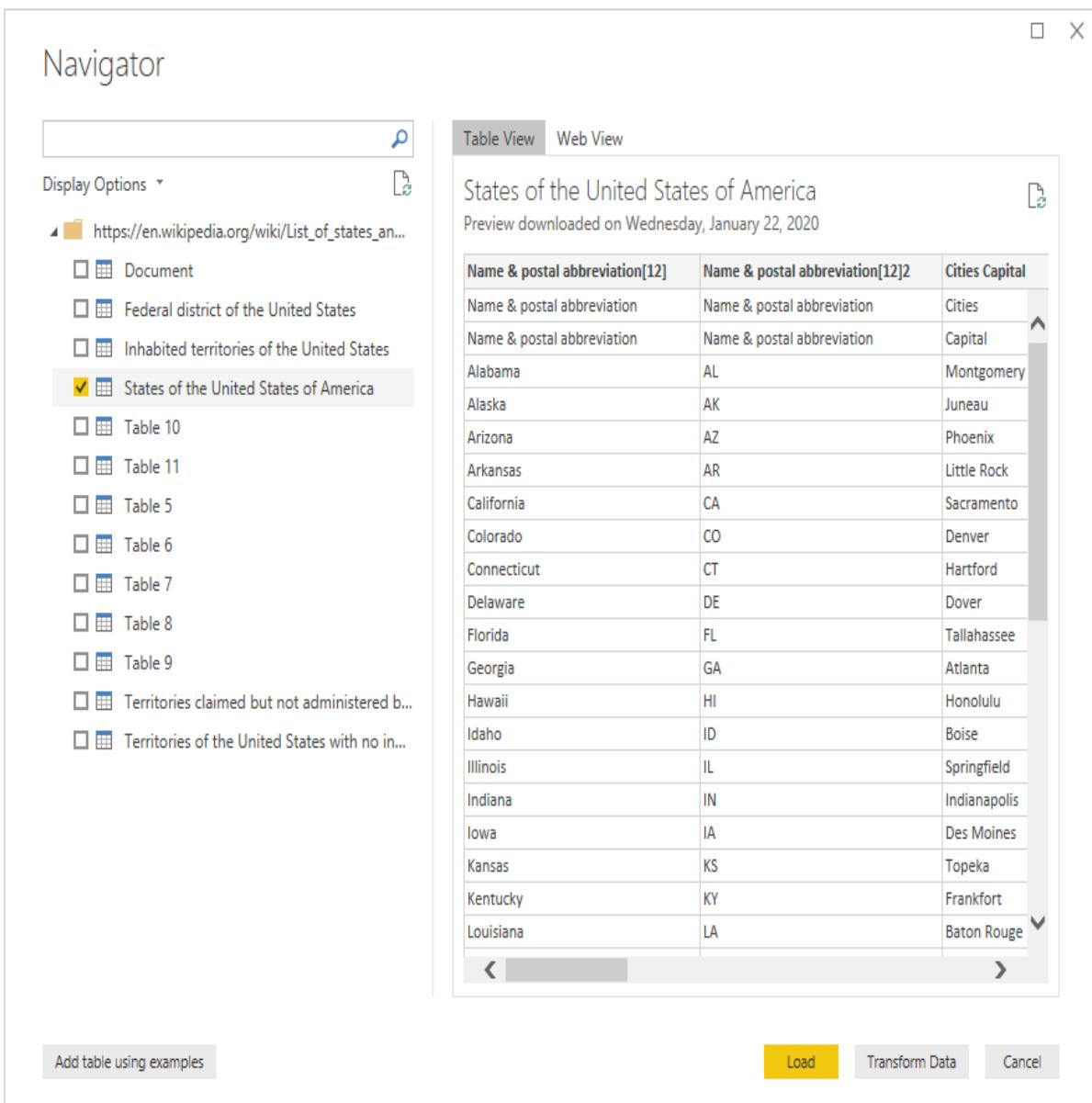
Data Analytics

Student Material

Note

If you need to change the authentication method later, go to **Changing the authentication method**.

- From the **Navigator** dialog, you can select a table, then either transform the data in the Power Query editor by selecting **Transform Data**, or load the data by selecting **Load**.



The right side of the **Navigator** dialog displays the contents of the table you select to transform or load. If you're uncertain which table contains the data you're interested in, you can select the **Web View** tab. The web view lets you see the entire contents of the web page, and highlights each of the tables that have been detected on that site. You can select the check box above the highlighted table to obtain the data from that table.

On the lower left side of the **Navigator** dialog, you can also select the **Add table using examples** button. This selection presents an interactive window where you can preview the content of the web page and enter sample values of the data you want to extract.

2.5 READING FROM APIs

API Documentation

There's no way around navigating API documentation. For working with Outlook, you go [here](#). If you are successful in following this tutorial and want to do more with the API, use that link for a reference.

Authorization

Most APIs, including the Microsoft Graph, require authorization before they'll give you an access token that will allow you to call the API. This is often the most difficult part for someone new to pulling data from APIs (it was for me).

Register your App

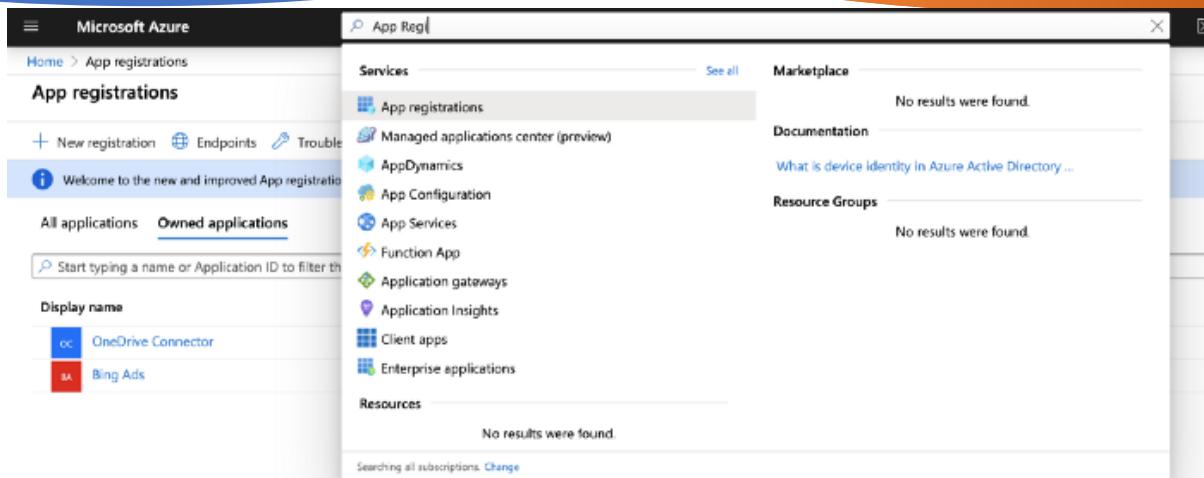
Follow the instructions here to register your app (these are pretty straightforward and use the Azure GUI) You'll need a school, work, or personal Microsoft account to register. If you use a work email, you'll need admin access to your Azure instance. If you're doing this for work, contact your IT department and they'll likely be able to register the app for you. Select 'web' as your app type and use a trusted URL for your redirect URI (one that only you have access to the underlying web data. I used *franklinsports.com* as only my team can access the web data for that site. If you were building an app, you would redirect to your app to a place where your app could pick up the code and authorize the user).

Enable Microsoft Graph Permissions

Once your app is registered, go to the App Registration portal:

Data Analytics

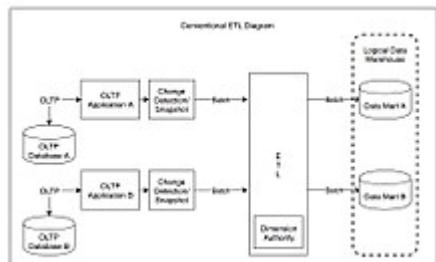
Student Material



The screenshot shows the Microsoft Azure portal's App Registrations page. At the top left, it says "Microsoft Azure" and "App Registrations". Below that, there are tabs for "All applications" and "Owned applications", with "Owned applications" currently selected. A search bar says "Start typing a name or Application ID to filter this list". Under "Display name", there are two entries: "OneDrive Connector" and "Bing Ads". On the right side, there are sections for "Services" (listing App registrations, Managed applications center (preview), AppDynamics, App Configuration, App Services, Function App, Application gateways, Application Insights, Client apps, and Enterprise applications), "Marketplace" (with a note "No results were found."), "Documentation" (with a link "What is device identity in Azure Active Directory..."), and "Resource Groups" (with a note "No results were found."). There is also a "See all" link at the top right of the services section.

2.6 ETL (extract, transform, load)

In computing, extract, transform, load (ETL) is a three-phase process where data is first extracted then transformed (cleaned, sanitized, scrubbed) and finally loaded into an output data container. The data can be collated from one or more sources and it can also be outputted to one or more destinations. ETL processing is typically executed using software applications but it can be also be done manually by system operators. ETL software typically automates the entire process and can be run manually or on reoccurring schedules either as single jobs or aggregated into a batch of jobs.



Conventional ETL diagram

Extract

A properly designed ETL system extracts data from source systems and enforces data type and data validity standards and ensures it conforms structurally to the requirements of the output. Some ETL systems can also deliver data in a presentation-ready format so that application developers can build applications and end users can make decisions.

Data extraction involves extracting data from homogeneous or heterogeneous sources; data transformation processes data by data cleaning and transforming it into a proper storage format/structure for the purposes of querying and analysis; finally, data loading describes the insertion of data into the final target database such as an operational data store, a data mart, data lake or a data warehouse.

ETL processing involves extracting the data from the source system(s). In many cases, this represents the most important aspect of ETL, since extracting data correctly sets the stage for

the success of subsequent processes. Most data-warehousing projects combine data from different source systems. Each separate system may also use a different data organization and/or format. Common data-source formats include relational databases, XML, JSON and flat files, but may also include non-relational database structures such as Information Management System (IMS) or other data structures such as Virtual Storage Access Method (VSAM) or Indexed Sequential Access Method (ISAM), or even formats fetched from outside sources by means such as web spidering or screen-scraping. The streaming of the extracted data source and loading on-the-fly to the destination database is another way of performing ETL when no intermediate data storage is required.

An intrinsic part of the extraction involves data validation to confirm whether the data pulled from the sources has the correct/expected values in a given domain (such as a pattern/default or list of values). If the data fails the validation rules, it is rejected entirely or in part. The rejected data is ideally reported back to the source system for further analysis to identify and to rectify the incorrect records.

Transform

In the data transformation stage, a series of rules or functions are applied to the extracted data in order to prepare it for loading into the end target.

An important function of transformation is data cleansing, which aims to pass only "proper" data to the target. The challenge when different systems interact is in the relevant systems' interfacing and communicating. Character sets that may be available in one system may not be so in others.

Selecting only certain columns to load: (or selecting null columns not to load). For example, if the source data has three columns (aka "attributes"), roll_no, age, and salary, then the selection may take only roll_no and salary. Or, the selection mechanism may ignore all those records where salary is not present (salary = null).

Translating coded values: (e.g., if the source system codes male as "1" and female as "2", but the warehouse codes male as "M" and female as "F")

Encoding free-form values: (e.g., mapping "Male" to "M")

Deriving a new calculated value: (e.g., sale_amount = qty * unit_price)

Sorting or ordering the data based on a list of columns to improve search performance

Joining data from multiple sources (e.g., lookup, merge) and deduplicating the data

Aggregating (for example, rollup — summarizing multiple rows of data — total sales for each store, and for each region, etc.)

Generating surrogate-key values

Transposing or pivoting (turning multiple columns into multiple rows or vice versa)

Splitting a column into multiple columns (e.g., converting a comma-separated list, specified as a string in one column, into individual values in different columns)

Disaggregating repeating columns

Looking up and validating the relevant data from tables or referential files

Applying any form of data validation; failed validation may result in a full rejection of the data, partial rejection, or no rejection at all, and thus none, some, or all of the data is handed over to the next step depending on the rule design and exception handling; many of the above transformations may result in exceptions, e.g., when a code translation parses an unknown code in the extracted data

Load

The load phase loads the data into the end target, which can be any data store including a simple delimited flat file or a data warehouse. Depending on the requirements of the organization, this process varies widely. Some data warehouses may overwrite existing information with cumulative information; updating extracted data is frequently done on a daily, weekly, or monthly basis. Other data warehouses (or even other parts of the same data warehouse) may add new data in a historical form at regular intervals — for example, hourly. To understand this, consider a data warehouse that is required to maintain sales records of the last year. This data warehouse overwrites any data older than a year with newer data. However, the entry of data for any one-year window is made in a historical manner. The timing and scope to replace or append are strategic design choices dependent on the time available and the business needs. More complex systems can maintain a history and audit trail of all changes to the data loaded in the data warehouse. As the load phase interacts with a database, the constraints defined in the database schema — as well as in triggers activated upon data load — apply (for example, uniqueness, referential integrity, mandatory fields), which also contribute to the overall data quality performance of the ETL process.

For example, a financial institution might have information on a customer in several departments and each department might have that customer's information listed in a different way. The membership department might list the customer by name, whereas the accounting department might list the customer by number. ETL can bundle all of these data elements and consolidate them into a uniform presentation, such as for storing in a database or data warehouse.

Another way that companies use ETL is to move information to another application permanently. For instance, the new application might use another database vendor and most likely a very different database schema. ETL can be used to transform the data into a format suitable for the new application to use.

An example would be an Expense and Cost Recovery System (ECRS) such as used by accountancies, consultancies, and legal firms. The data usually ends up in the time and billing system, although some businesses may also utilize the raw data for employee productivity reports to Human Resources (personnel dept.) or equipment usage reports to Facilities Management.

Real-life ETL cycle

The typical real-life ETL cycle consists of the following execution steps:

- Cycle initiation
- Build reference data
- Extract (from sources)

- Validate
- Transform (clean, apply business rules, check for data integrity, create aggregates or disaggregates)
- Stage (load into staging tables, if used)
- Audit reports (for example, on compliance with business rules. Also, in case of failure, helps to diagnose/repair)
- Publish (to target tables)
- Archive

Challenges

ETL processes can involve considerable complexity, and significant operational problems can occur with improperly designed ETL systems.

The range of data values or data quality in an operational system may exceed the expectations of designers at the time validation and transformation rules are specified. Data profiling of a source during data analysis can identify the data conditions that must be managed by transform rules specifications, leading to an amendment of validation rules explicitly and implicitly implemented in the ETL process.

Data warehouses are typically assembled from a variety of data sources with different formats and purposes. As such, ETL is a key process to bring all the data together in a standard, homogeneous environment.

Design analysis should establish the scalability of an ETL system across the lifetime of its usage — including understanding the volumes of data that must be processed within service level agreements. The time available to extract from source systems may change, which may mean the same amount of data may have to be processed in less time. Some ETL systems have to scale to process terabytes of data to update data warehouses with tens of terabytes of data. Increasing volumes of data may require designs that can scale from daily batch to multiple-day micro batch to integration with message queues or real-time change-data-capture for continuous transformation and update.

2.7 Reading Data From Other Sources

The **Other** category provides the following data connections:

- Web
- SharePoint list
- OData Feed
- Active Directory
- Microsoft Exchange
- Hadoop File (HDFS)
- Spark
- Hive LLAP
- R script
- Python script
- ODBC
- OLE DB

- Acterys : Model Automation & Planning (Beta)
- Amazon OpenSearch Service (Beta)
- Anaplan Connector Autodesk Construction Cloud (Beta)
- Solver
- BitSight Security Ratings
- BQE Core
- Bloomberg Data and Analytics
- Cherwell (Beta)
- Cognite Data Fusion
- Delta Sharing
- Eduframe (Beta)
- EQuIS (Beta)
- FactSet RMS (Beta)
- FHIR
- Google Sheets (Beta)
- Information Grid (Beta)
- Jamf Pro (Beta)
- Kognitwin
- MicroStrategy for Power BI
- OpenSearch Project (Beta)
- Paxata
- QubolePresto (Beta)
- Roamler (Beta)
- SIS-CC SDMX (Beta)
- Shortcuts Business Insights (Beta)
- SingleStore Direct Query Connector 1.0 (Beta)
- Siteimprove
- Socialbakers Metrics 1.1.0 (Beta)
- Starburst Enterprise
- SumTotal
- SurveyMonkey (Beta)
- Microsoft Teams Personal Analytics (Beta)
- Tenforce (Smart)List
- Uservcube (Beta)
- Vena
- Vessel Insight
- Zucchetti HR Infinity (Beta)
- Blank Query

The following image shows the **Get Data** window for **Other**.

Get Data

- All
- File
- Database
- Power Platform
- Azure
- Online Services
- Other

Other

- Web
- SharePoint list
- OData Feed
- Active Directory
- Microsoft Exchange
- Hadoop File (HDFS)
- Spark
- Hive LLAP (Beta)
- R script
- Python script
- ODBC
- OLE DB
- Acterys : Model Automation & Planning (Beta)
- Automation Anywhere (Beta)
- Solver
- Cherwell (Beta)

[Certified Connectors](#) | [Template Apps](#)

[Connect](#) [Cancel](#)

Note

At this time, it's not possible to connect to custom data sources secured using Azure Active Directory.

Template apps

You can find template apps for your organization by selecting the **Template Apps** link near the bottom of the **Get Data** window.

Get Data

Search

All
File
Database
Power Platform
Azure
Online Services
Other

Other

- Web
- SharePoint list
- OData Feed
- Active Directory
- Microsoft Exchange
- Hadoop File (HDFS)
- Spark
- Hive LLAP
- R script
- Python script
- ODBC
- OLE DB
- Acterys : Model Automation & Planning (Beta)
- Anaplan Connector (Beta)
- Solver
- Bloomberg Data and Analytics (Beta)

Certified Connectors | **Template Apps** | Connect | Cancel

Available Template Apps may vary based on your organization.

READING DIFFERENT FILE TYPES.

1. What is a file format?

A file format is a standard way in which information is encoded for storage in a file. First, the file format specifies whether the file is a binary or ASCII file. Second, it shows how the information is organized. For example, comma-separated values (CSV) file format stores tabular data in plain text.

Data Analytics

Student Material

To identify a file format, you can usually look at the file extension to get an idea. For example, a file saved with name “Data” in “CSV” format will appear as “Data.csv”. By noticing “.csv” extension we can clearly identify that it is a “CSV” file and data is stored in a tabular format.

CSV				JSON																																																																					
<table border="1"><thead><tr><th>A</th><th>B</th><th>C</th><th>D</th></tr></thead><tbody><tr><td>1 ID</td><td>Gender</td><td>City</td><td>Monthly_I</td></tr><tr><td>2 ID000002C</td><td>Female</td><td>Delhi</td><td>20000</td></tr><tr><td>3 ID000004E</td><td>Male</td><td>Mumbai</td><td>35000</td></tr><tr><td>4 ID000007F</td><td>Male</td><td>Panchkula</td><td>22500</td></tr><tr><td>5 ID000008I</td><td>Male</td><td>Saharsa</td><td>35000</td></tr><tr><td>6 ID000009J</td><td>Male</td><td>Bengaluru</td><td>100000</td></tr><tr><td>7 ID000010K</td><td>Male</td><td>Bengaluru</td><td>45000</td></tr><tr><td>8 ID000011L</td><td>Female</td><td>Sindhuduri</td><td>70000</td></tr><tr><td>9 ID000012N</td><td>Male</td><td>Bengaluru</td><td>20000</td></tr><tr><td>10 ID000013R</td><td>Male</td><td>Kochi</td><td>75000</td></tr><tr><td>11 ID000014C</td><td>Female</td><td>Mumbai</td><td>30000</td></tr><tr><td>12 ID000016C</td><td>Male</td><td>Mumbai</td><td>25000</td></tr><tr><td>13 ID000018S</td><td>Female</td><td>Surat</td><td>25000</td></tr><tr><td>14 ID000019T</td><td>Female</td><td>Pune</td><td>24000</td></tr><tr><td>15 ID000021V</td><td>Male</td><td>Bhubanes</td><td>27000</td></tr><tr><td>16 ID000022V</td><td>Female</td><td>Howrah</td><td>28000</td></tr></tbody></table>				A	B	C	D	1 ID	Gender	City	Monthly_I	2 ID000002C	Female	Delhi	20000	3 ID000004E	Male	Mumbai	35000	4 ID000007F	Male	Panchkula	22500	5 ID000008I	Male	Saharsa	35000	6 ID000009J	Male	Bengaluru	100000	7 ID000010K	Male	Bengaluru	45000	8 ID000011L	Female	Sindhuduri	70000	9 ID000012N	Male	Bengaluru	20000	10 ID000013R	Male	Kochi	75000	11 ID000014C	Female	Mumbai	30000	12 ID000016C	Male	Mumbai	25000	13 ID000018S	Female	Surat	25000	14 ID000019T	Female	Pune	24000	15 ID000021V	Male	Bhubanes	27000	16 ID000022V	Female	Howrah	28000	{ "Employee": [{ "id": "1", "Name": "Ankit", "Sal": "1000", }, { "id": "2", "Name": "Faizy", }] }	
A	B	C	D																																																																						
1 ID	Gender	City	Monthly_I																																																																						
2 ID000002C	Female	Delhi	20000																																																																						
3 ID000004E	Male	Mumbai	35000																																																																						
4 ID000007F	Male	Panchkula	22500																																																																						
5 ID000008I	Male	Saharsa	35000																																																																						
6 ID000009J	Male	Bengaluru	100000																																																																						
7 ID000010K	Male	Bengaluru	45000																																																																						
8 ID000011L	Female	Sindhuduri	70000																																																																						
9 ID000012N	Male	Bengaluru	20000																																																																						
10 ID000013R	Male	Kochi	75000																																																																						
11 ID000014C	Female	Mumbai	30000																																																																						
12 ID000016C	Male	Mumbai	25000																																																																						
13 ID000018S	Female	Surat	25000																																																																						
14 ID000019T	Female	Pune	24000																																																																						
15 ID000021V	Male	Bhubanes	27000																																																																						
16 ID000022V	Female	Howrah	28000																																																																						
				<?xml version="1.0"?><contact-info><name>Ankit</name><company>Analytics Vidhya</company><phone>+9187654321</phone></contact-info>																																																																					

2. Why should we understand different file formats?

Usually, the files you will come across will depend on the application you are building. For example, in an image processing system, you need image files as input and output. So you will mostly see files in jpeg, gif or png format.

As a data scientist, you need to understand the underlying structure of various file formats, their advantages and dis-advantages. Unless you understand the underlying structure of the data, you will not be able to explore it. Also, at times you need to make decisions about how to store data.

Comma-separated values

Comma-separated values file format falls under spreadsheet file format.

What is Spreadsheet File Format?

In spreadsheet file format, data is stored in cells. Each cell is organized in rows and columns. A column in the spreadsheet file can have different types. For example, a column can be of string type, a date type or an integer type. Some of the most popular spreadsheet file formats are Comma Separated Values (CSV), Microsoft Excel Spreadsheet (xlsx) and Microsoft Excel Open XML Spreadsheet (xlsx).

Each line in CSV file represents an observation or commonly called a record. Each record may contain one or more fields which are separated by a comma.

Sometimes you may come across files where fields are not separated by using a comma, but they are separated using tab. This file format is known as TSV (Tab Separated Values) file format.

The below image shows a CSV file which is opened in Notepad.

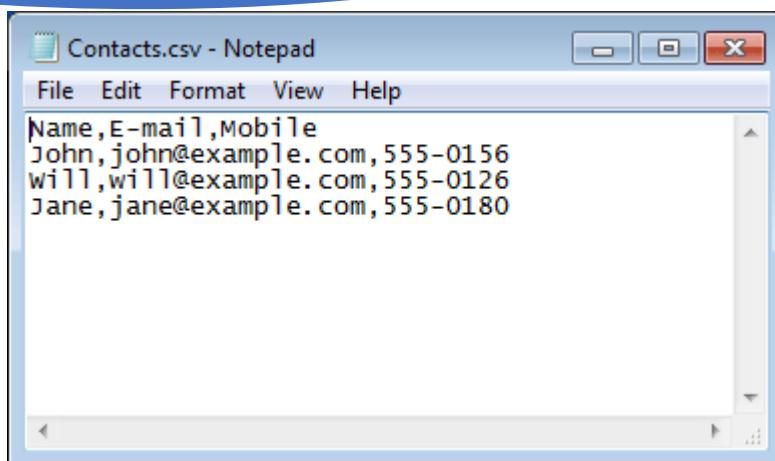
.....

Brought to you by:



Data Analytics

Student Material



XLSX files

XLSX is a Microsoft Excel Open XML file format. It also comes under the Spreadsheet file format. It is an XML-based file format created by Microsoft Excel. The XLSX format was introduced with Microsoft Office 2007.

In XLSX data is organized under the cells and columns in a sheet. Each XLSX file may contain one or more sheets. So a workbook can contain multiple sheets.

The below image shows a “xlsx” file which is opened in Microsoft Excel.

	A	B	C	D	E	F	G	H	I
1	Invoice ID	Billed To	Amount Billed in Loc	Country	Amount Billed in U	Billed By	Billed On	Paid On	
2	11270	39045213	8,738,600.640.00	Brazil	3,675,848,666.21	239185	10/5/04	10/18/04	
3	11271	18543489	11,063,836.00	USA	11,063,836.00	457232	10/5/04		
4	11273	19783482	252,148.50	USA	252,148.50	239185	10/6/04	11/11/04	
5	11276	14324742	1,934,460.00	USA	1,934,460.00	135673	10/6/04	10/20/04	
6	11278	14898029	1,400,825.00	USA	1,400,825.00	239185	10/6/04	10/19/04	
7	11280	39045213	8,738,600.640.00	Brazil	3,675,848,666.21	423286	10/7/04	10/20/04	
8	11282	19783482	252,148.50	USA	252,148.50	457232	10/7/04	10/25/04	
9	11285	38763919	2,234,301.30	Argentina	772,847.05	239185	10/10/04	11/30/04	
10	11286	43459747	14,836,604.08	Australia	11,366,352.06	423286	10/10/04		
11	11287	15432147	252,148.50	USA	252,148.50	457232	10/11/04	11/4/04	
12	12051	39045213	8,738,600.640.00	Brazil	3,675,848,666.21	457232	11/2/04		
13	12102	18543489	11,063,836.00	USA	11,063,836.00	239185	11/17/04		
14	12263	19783482	252,148.50	USA	252,148.50	423286	12/5/04		
15	12468	14898029	1,400,825.00	USA	1,400,825.00	135673	12/24/04	1/2/05	
16	12471	39045213	8,738,600.640.00	Brazil	3,675,848,666.21	457232	12/27/04		
17	12476	38763919	2,234,301.30	Argentina	772,847.05	135673	12/24/04		
18	12478	15432147	252,148.50	USA	252,148.50	423286	12/24/04	1/2/05	
19									
20									
21									

In above image, you can see that there are multiple sheets present (bottom left) in this file, which are Customers, Employees, Invoice, Order. The image shows the data of only one sheet – “Invoice”.

ZIP files

ZIP format is an archive file format.

What is Archive File format?

In Archive file format, you create a file that contains multiple files along with metadata. An archive file format is used to collect multiple data files together into a single file. This is done for simply compressing the files to use less storage space.

There are many popular computer data archive format for creating archive files. Zip, RAR and Tar being the most popular archive file format for compressing the data.

So, a ZIP file format is a lossless compression format, which means that if you compress the multiple files using ZIP format you can fully recover the data after decompressing the ZIP file. ZIP file format uses many compression algorithms for compressing the documents. You can easily identify a ZIP file by the .zip extension.

Plain Text (txt) file format

In Plain Text file format, everything is written in plain text. Usually, this text is in unstructured form and there is no meta-data associated with it. The txt file format can easily be read by any program. But interpreting this is very difficult by a computer program.

Let's take a simple example of a text File.

The following example shows **text file data** that contain **text**:

“world is full of idiots”

JSON file format

JavaScript Object Notation(JSON) is a text-based open standard designed for exchanging the data over web. JSON format is used for transmitting structured data over the web. The JSON file format can be easily read in any programming language because it is language-independent data format.

Let's take an example of a JSON file

Data Analytics

Student Material

The following example shows how a typical JSON file stores information of employees.

```
{  
  "Employee": [  
    {
```

```
      "id": "1",  
      "Name": "Ankit",  
      "Sal": "1000",  
    },  
    {
```

```
      "id": "2",  
      "Name": "Faizy",  
      "Sal": "2000",  
    },  
  ]  
}
```

XML file format

XML is also known as Extensible Markup Language. As the name suggests, it is a markup language. It has certain rules for encoding data. XML file format is a human-readable and machine-readable file format. XML is a self-descriptive language designed for sending information over the internet. XML is very similar to HTML, but has some differences. For example, XML does not use predefined tags as HTML.

Let's take the simple example of XML File format.

~~|~|~|~|~

Brought to you by:



The following example shows an xml document that contains the information of an employee.

```
<?xml version="1.0"?>  
<contact-info>  
  
<name>Ankit</name>  
  
<company>AnalyticsVidhya</company>  
  
<phone>+9187654321</phone>  
  
</contact-info>
```

The “`<?xml version="1.0"?>`” is a XML declaration at the start of the file (it is optional). In this deceleration, *version* specifies the XML version and *encoding* specifies the character encoding used in the document. `<contact-info>` is a tag in this document. Each XML-tag needs to be closed.

HTML files

HTML stands for Hyper Text Markup Language. It is the standard markup language which is used for creating Web pages. HTML is used to describe structure of web pages using markup. HTML tags are same as XML but these are predefined. You can easily identify HTML document subsection on basis of tags such as `<head>` represent the heading of HTML document. `<p>` “paragraph” paragraph in HTML. HTML is not case sensitive.

The following example shows an HTML document.

```
<!DOCTYPE html>  
<html>  
<head>  
<title>Page Title</title>  
</head>  
<body><h1>My First Heading</h1>  
<p>My first paragraph.</p></body>  
</html>
```

Each tag in HTML is enclosed under the angular bracket(<>). The <!DOCTYPE html> tag defines that document is in HTML format. <html> is the root tag of this document. The <head> element contains heading part of this document. The <title>, <body>, <h1>, <p> represent the title, body, heading and paragraph respectively in the HTML document.

Image files

Image files are probably the most fascinating file format used in data science. Any computer vision application is based on image processing. So it is necessary to know different image file formats.

Usual image files are 3-Dimensional, having RGB values. But, they can also be 2-Dimensional (grayscale) or 4-Dimensional (having intensity) – an Image consisting of pixels and meta-data associated with it.

Each image consists of one or more frames of pixels. And each frame is made up of two-dimensional array of pixel values. Pixel values can be of any intensity. Meta-data associated with an image, can be an image type (.png) or pixel dimensions.



PDF file format

PDF (Portable Document Format) is an incredibly useful format used for interpretation and display of text documents along with incorporated graphics. A special feature of a PDF file is that it can be secured by a password.

Here's an example of a pdf file.

Data Analytics

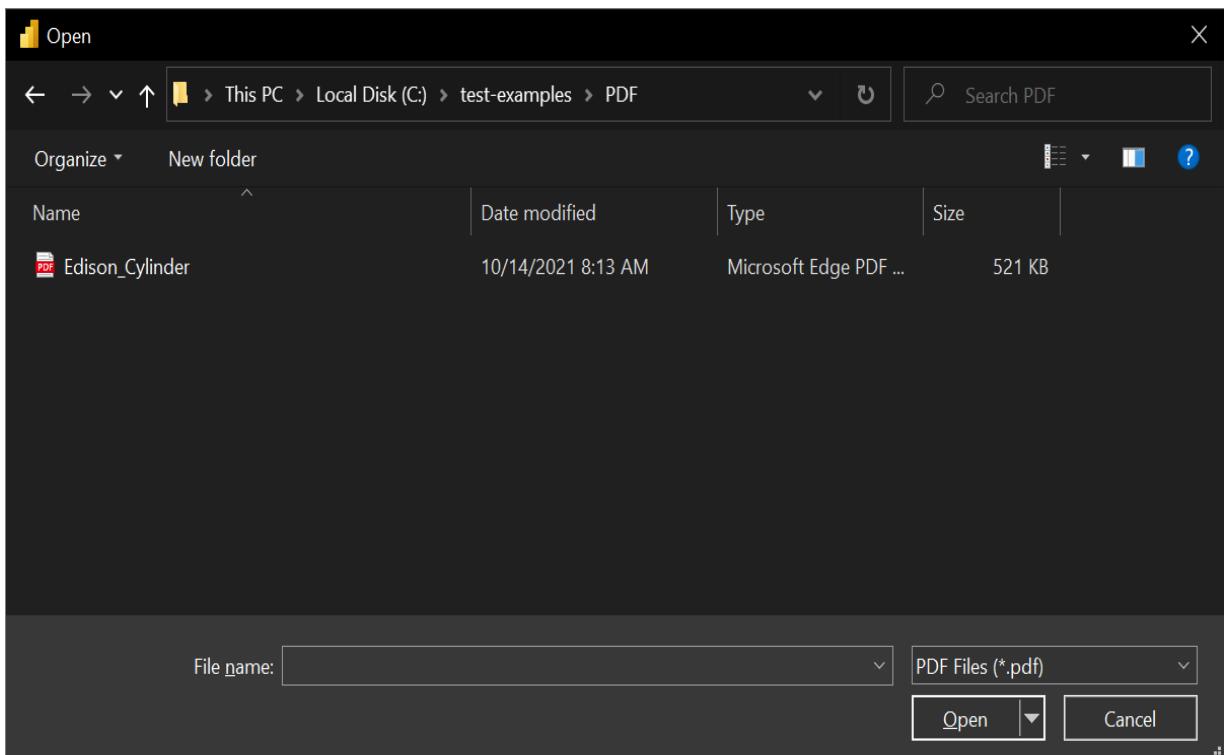
Student Material

The screenshot shows the Adobe Reader interface with the file 'DeepLearningBook.pdf' open. The left sidebar contains icons for file operations like Open, Save, Print, and Copy. The top menu bar includes File, Edit, View, Window, Help, Tools, Sign, and Comment. The main content area displays the book's table of contents. The title 'Deep Learning' is visible at the top of the page. The table of contents lists various chapters and their page numbers.

Section	Page Number
Website	vii
Acknowledgments	viii
Notation	xi
1 Introduction	1
1.1 Who Should Read This Book?	8
1.2 Historical Trends in Deep Learning	11
2 Applied Math and Machine Learning Basics	29
2.1 Scalars, Vectors, Matrices and Tensors	31
2.2 Multiplying Matrices and Vectors	34
2.3 Identity and Inverse Matrices	36
2.4 Linear Dependence and Span	37
2.5 Norms	39
2.6 Special Kinds of Matrices and Vectors	40
2.7 Eigendecomposition	42
2.8 Singular Value Decomposition	44
2.9 The Moore-Penrose Pseudoinverse	45
2.10 The Trace Operator	46
2.11 The Determinant	47
2.12 Example: Principal Components Analysis	48
3 Probability and Information Theory	53
3.1 Why Probability?	54

To make the connection from Power Query Desktop:

1. Select the **PDF** option in the connector selection.
2. Browse for and select the PDF file you want to load. Then select **Open**.



If the PDF file is online, use the Web connector to connect to the file.

Data Analytics

Student Material

3. In **Navigator**, select the file information you want, then either select **Load** to load the data or **Transform Data** to continue transforming the data in Power Query Editor.

The screenshot shows the Microsoft Power BI Navigator window. On the left, there's a tree view of a PDF file named "Edison_Cylinder.pdf". The node "Table001 (Page 1)" is selected, indicated by a checked checkbox. Other nodes include "Table002 (Page 2)", "Table003 (Page 3)", "Table004 (Page 4)", "Table005 (Page 5)", "Table006 (Page 6)", and several "Page001" through "Page006" nodes. To the right of the tree view is a preview pane titled "Table001 (Page 1)". It contains a table with three columns: "Label No.", "Title", and "Performed by:". The table lists 42 entries, such as "3 Vocal Sextette From Lucia", "126 Cello Solo, Nina", and "422 Down In Turkey Hollow". At the bottom of the window are three buttons: "Load" (highlighted in yellow), "Transform Data", and "Cancel".

Connect to a PDF file from Power Query Online

To make the connection from Power Query Online:

1. Select the **PDF** option in the connector selection.
2. In the PDF dialog box that appears, either provide the file path or the URL to the location of the PDF file. If you're loading a local file, you can also select **Upload file (Preview)** to browse to the local file or drag and drop the file.

Data Analytics

Student Material

PDF File

Connection settings

Link to file Upload file (Preview) [?](#)

File path or URL *

C:\test-examples\PDF\Edison_Cylinders.pdf [Browse OneDrive...](#)

Connection credentials

Data gateway * [On-premises][Admin] TestGateway23 [?](#)

Authentication kind Windows

Username domain\alias

Password

3. If necessary, select an on-premises data gateway to access the PDF file.
4. If this is the first time you've accessed this PDF file, select the authentication kind and sign in to your account (if needed).
5. In **Navigator**, select the file information you want, and then select **Transform Data** to continue transforming the data in Power Query Editor.

Power Query - Choose data

Search

Display options [?](#)

PDF [12]

Table001 (Page 1)

Table002 (Page 2)

Table003 (Page 3)

Table004 (Page 4)

Table005 (Page 5)

Table006 (Page 6)

Page001

Page002

Page003

Page004

Page005

Page006

Table001 (Page 1)

Label No.	Title	Performed by:	Duration	Type	Take/Mold	Condition
3	Vocal Sextette From Lucia		Four minute	Black Wax	0.39	Good
126	Cello Solo, Nina	Kronold	Four minute	Black Wax	0.7	Good
179	Are You Coming Home Tonight	Anthony & Harrison	Four minute	Black Wax	.9	Good
217	Garden Melody, Violin	A. Spalding	Four minute	Black Wax	0.17	Poor
232	Dream Of The Tyrolienne	Venetian Trio	Four minute	Black Wax	0.11	Good
265	Superba Lancers - 1st And 2nd Figures	Band	Four minute	Black Wax	0.8	Good
266	Superba Lancers - 3rd And 4th Figures	Band	Four minute	Black Wax	0.7	Good
268	Petunia Quadrille - 1st And 2nd Figures	Band	Four minute	Black Wax	.10	Good
274	Dublin Daisies Two-Step	Band	Four minute	Black Wax	0.8	Good
275	Lucky Moon Three-Step	Band	Four minute	Black Wax	0.11	Poor
303	My Old Kentucky Home	Knickerbocker Quartet	Four minute	Black Wax	10	Good
321	He Was A Wonderful Man	Jones & Murray	Four minute	Black Wax	.12	Good
363	Medley Of Emmett's Yodle Songs	Watson	Four minute	Black Wax	.11	Good
395	That Mesmerizing Mendelssohn Tune	Collins & Harlan	Four minute	Black Wax	.9	Fair
422	Down In Turkey Hollow	Golden & Hughes	Four minute	Black Wax	.11	Good
553	Buck Dance Medley (Accordion)	Kimmble	Four minute	Black Wax	.9	Good

Back [Cancel](#) [Transform data](#)

Assignment

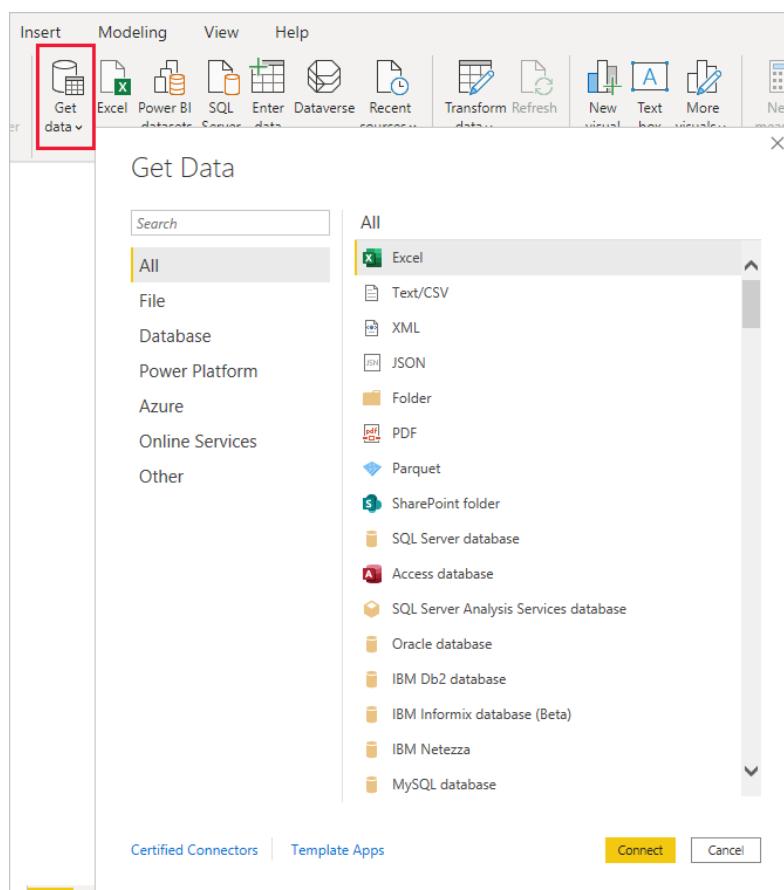
Connect excel data source with Power BI. (Get free sample database from online)

Solution :

Click on the “**Get data**” button to see the most common data types menu.

The “**Get Data**” dialog box categorizes the data types in the following way.

- **File:** Excel, CSV, XML, JSON, etc.
- **Database:** SQL Server, Oracle, MySQL, etc.
- **Power Platform:** Power BI datasets, Power BI dataflows, Dataverse, etc.
- **Azure:** MS Azure SQL Database, MS Azure Marketplace, etc.
- **Online Services:** Google Analytics, Zendesk, Asana, GitHub, Marketo, etc.
- **Other:** Web, R script, Python script, Spark, Google Sheets, etc.



Get Data

Power BI Desktop and Power Query allow users to automate the process of Power BI Data Ingestion. Let's discuss an example to understand this better. Before proceeding, make sure you've Power BI Desktop installed on your system. For the purposes of this demonstration,

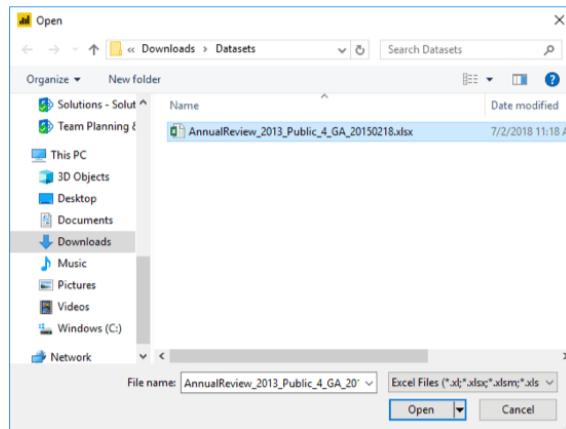
Data Analytics

Student Material

the General Aviation 2013 Excel dataset file (available for public use from Data.gov) has been used. Download the Excel file and save it in a local folder.

Follow the below-mentioned steps to load data into Power BI.

- Launch Power BI Desktop and click on the “**Get Data**” button from the Power BI toolbar. Now, choose the Excel connector and click on “**Connect**”.
- Locate the General Aviation Excel file you just downloaded and click on “**Open**”.



Chapter: 3

Data Cleansing

Scope:

- we look at finding data and reading different file types.
- Getting and Cleaning Data! The primary goal is to introduce you to the most common data storage systems and the appropriate tools to extract data from web or from databases like MySQL.
- Introduction to Data Wrangling and Shaping
- Advanced Shaping
- Beyond Initial Shaping
- Advanced DAX
- Publishing and Testing Your Dataset
- Summarizing Data
- Creating New Variables

DATA CLEANSING OVERVIEW

- Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.
- When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct.
- There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset. But it is crucial to establish a template for your data cleaning process, so you know you are doing it the right way every time.

What is the difference between data cleaning and data transformation?

Data cleaning is the process that removes data that does not belong in your dataset. Data transformation is the process of converting data from one format or structure into another. Transformation processes can also be referred to as data wrangling, or data munging, transforming and mapping data from one "raw" data form into another format for warehousing and analyzing. This article focuses on the processes of cleaning that data.

How do you clean data?

While the techniques used for data cleaning may vary according to the types of data your company stores, you can follow these basic steps to map out a framework for your organization.

Step 1: Remove duplicate or irrelevant observations

Remove unwanted observations from your dataset, including duplicate observations or irrelevant observations. Duplicate observations will happen most often during data collection. When you combine data sets from multiple places, scrape data, or receive data from clients or multiple departments, there are opportunities to create duplicate data. De-duplication is one of the largest areas to be considered in this process. Irrelevant observations are when you notice observations that do not fit into the specific problem you are trying to analyze. For example, if you want to analyze data regarding millennial customers, but your dataset includes older generations, you might remove those irrelevant observations. This can make analysis more efficient and minimize distraction from your primary target—as well as creating a more manageable and more performant dataset.

Step 2: Fix structural errors

Structural errors are when you measure or transfer data and notice strange naming conventions, typos, or incorrect capitalization. These inconsistencies can cause mislabeled

categories or classes. For example, you may find “N/A” and “Not Applicable” both appear, but they should be analyzed as the same category.

Step 3: Filter unwanted outliers

Often, there will be one-off observations where, at a glance, they do not appear to fit within the data you are analyzing. If you have a legitimate reason to remove an outlier, like improper data-entry, doing so will help the performance of the data you are working with. However, sometimes it is the appearance of an outlier that will prove a theory you are working on. Remember: just because an outlier exists, doesn't mean it is incorrect. This step is needed to determine the validity of that number. If an outlier proves to be irrelevant for analysis or is a mistake, consider removing it.

Step 4: Handle missing data

You can't ignore missing data because many algorithms will not accept missing values. There are a couple of ways to deal with missing data. Neither is optimal, but both can be considered.

- As a first option, you can drop observations that have missing values, but doing this will drop or lose information, so be mindful of this before you remove it.
- As a second option, you can input missing values based on other observations; again, there is an opportunity to lose integrity of the data because you may be operating from assumptions and not actual observations.
- As a third option, you might alter the way the data is used to effectively navigate null values.

Step 5: Validate and QA

At the end of the data cleaning process, you should be able to answer these questions as a part of basic validation:

- Does the data make sense?
- Does the data follow the appropriate rules for its field?
- Does it prove or disprove your working theory, or bring any insight to light?
- Can you find trends in the data to help you form your next theory?
- If not, is that because of a data quality issue?

Components of quality data

Determining the quality of data requires an examination of its characteristics, then weighing those characteristics according to what is most important to your organization and the application(s) for which they will be used.

Characteristics of quality data

- Accuracy. Ensure your data is close to the true values.
- Completeness. The degree to which all required data is known.
- Consistency. Ensure your data is consistent within the same dataset and/or across multiple data sets.

- Uniformity. The degree to which the data is specified using the same unit of measure.

Benefits of data cleaning

Having clean data will ultimately increase overall productivity and allow for the highest quality information in your decision-making. Benefits include:

- Removal of errors when multiple sources of data are at play.
- Fewer errors make for happier clients and less-frustrated employees.
- Ability to map the different functions and what your data is intended to do.
- Monitoring errors and better reporting to see where errors are coming from, making it easier to fix incorrect or corrupt data for future applications.
- Using tools for data cleaning will make for more efficient business practices and quicker decision-making.

3.1 WE LOOK AT FINDING DATA AND READING DIFFERENT FILE TYPES

Data Types

Data types generally fall into five categories:

1. Observational
 - Can't be recaptured, recreated or replaced
 - Examples: Sensor readings, sensory (human) observations, survey results
2. Experimental
 - Data collected under controlled conditions, in laboratory
 - Should be reproducible, but can be expensive
 - Examples: gene sequences, chromatograms, spectroscopy, microscopy
3. Derived or compiled
 - Reproducible, but can be very expensive
 - Examples: text and data mining, derived variables, compiled database, 3D models
4. Simulation
 - Results from using a model to study the behaviour and performance of an actual or theoretical system
 - Models and metadata, where the input can be more important than output data
 - Examples: climate models, economic models, biogeochemical models
5. Reference or canonical
 - Static or organic collection [peer-reviewed] datasets, most probably published and/or curated.
 - Examples: gene sequence databanks, chemical structures, census data, spatial data portals.

Data Formats

Research data comes in many varied formats: text, numeric, multimedia, models, software languages, discipline specific (e.g. crystallographic information file (CIF) in chemistry), and instrument specific.

Data Analytics

Student Material

Formats more likely to be accessible in the future are:

- Non-proprietary
- Open, documented standards
- In common usage by the research community
- Using standard character encodings (ASCII, UTF-8)
- Uncompressed (desirable, space permitting)

Data Analytics

Student Material

Geospatial data vector and raster data	ESRI Shapefile (essential: .shp, .shx, .dbf ; optional: MapInfo Interchange Format .prj, .sbx, .sbn) geo-referenced TIFF (.tif, .tfw) CAD data (.dwg) tabular GIS attribute data	ESRI Geodatabase format (.mdb)
Qualitative data textual	<ul style="list-style-type: none"> • eXtensible Mark-up Language (XML) text according to an appropriate Document Type Definition (DTD) or schema (.xml) • Rich Text Format (.rtf) • plain text data, UTF-8 (Unicode; .txt) 	<ul style="list-style-type: none"> • plain text data, ASCII (.txt) • Hypertext Mark-up Language (HTML) (.html) • widely-used proprietary formats, e.g. MS Word (.doc/.docx) • LaTeX(.tex)
Digital image data	TIFF version 6 uncompressed (.tif)	<ul style="list-style-type: none"> • JPEG (.jpeg, .jpg) • TIFF (other versions; .tif, .tiff) • JPEG 2000 (.jp2) • Adobe Portable Document Format (PDF/A, PDF) (.pdf)
Digital audio data	<ul style="list-style-type: none"> ➢ Free Lossless Audio Codec (FLAC) (.flac) ➢ Waveform Audio Format (WAV) (.wav) ➢ MPEG-1 Audio Layer 3 (.mp3) - spoken word audio only 	<ul style="list-style-type: none"> • MPEG-1 Audio Layer 3 (.mp3) • Audio Interchange File Format (AIFF) (.aif)
Digital video data	<ul style="list-style-type: none"> • MPEG-4 High Profile (.mp4) • motion JPEG 2000 (.jp2) 	JPEG 2000 (.mj2)
Documentation & Scripts	<ul style="list-style-type: none"> • Rich Text Format (.rtf) • Open Document Text (.odt) • HTML (.htm, .html) 	<ul style="list-style-type: none"> • plain text (.txt) • widely-used proprietary formats, e.g. MS Word (.doc/.docx) or MS Excel (.xls/ .xlsx) • XML marked-up text (.xml) according to an appropriate DTD or schema, e.g. XHMTL 1.0 • PDF/A or PDF (.pdf)

3.3 Data wrangling

Brought to you by:



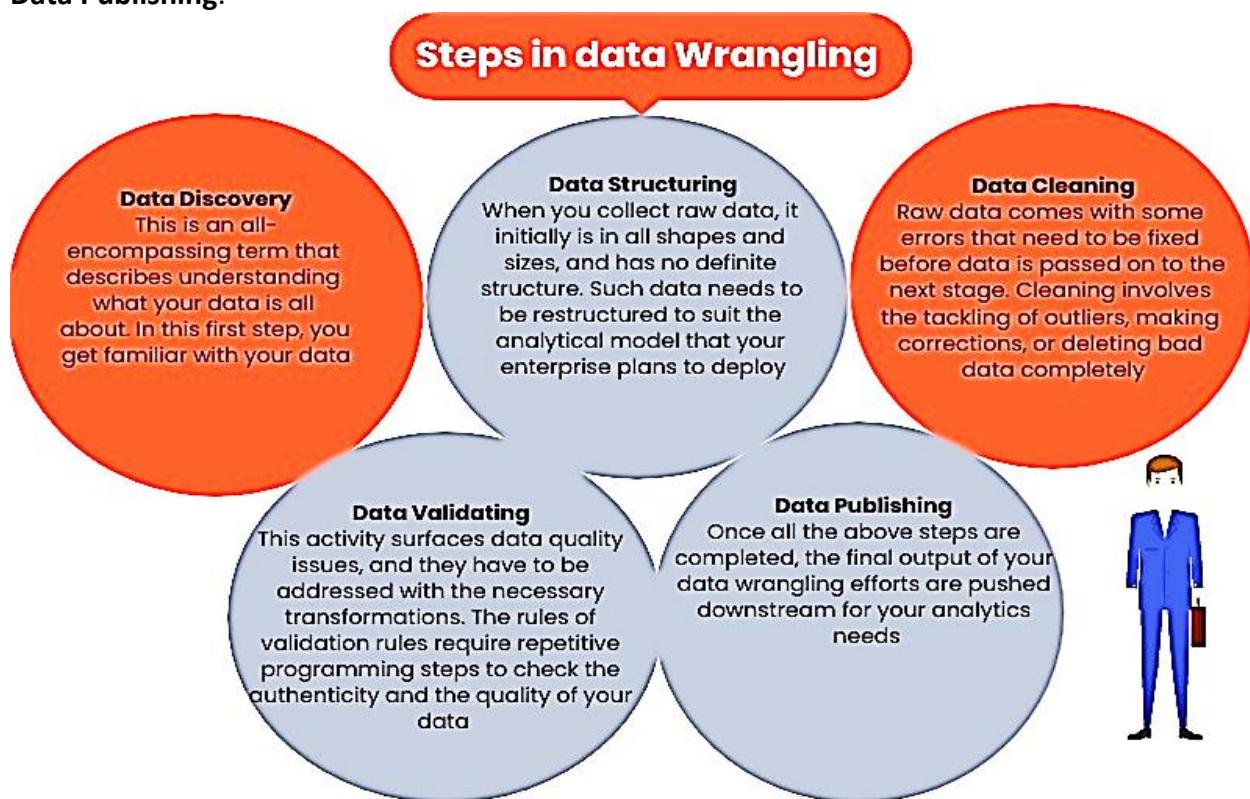
Data Analytics

Student Material

Data wrangling is the practice of converting and then plotting data from one “raw” form into another. The aim is to make it ready for downstream analytics. Often in charge of this is a data wrangler or a team of “mangers”.

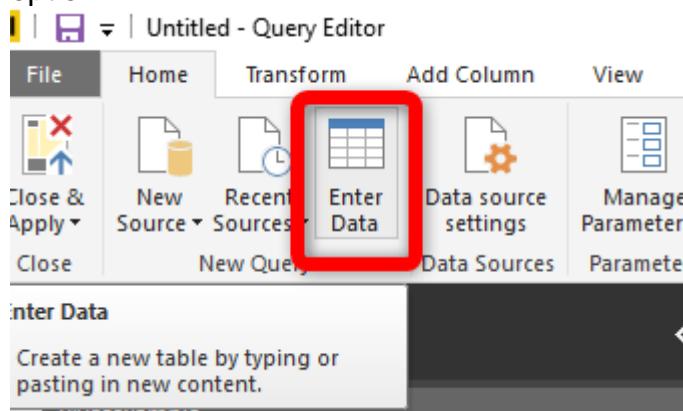
Steps in Data Wrangling:

1. **Data Discovery.**
2. **Data Structuring.**
3. **Data Cleaning.**
4. **Data Enriching.**
5. **Data Validating.**
6. **Data Publishing.**



Data Wrangling in Power BI

To pivot data using Power Query, we need to enter data using “Enter Data Manually” option.



Data Analytics

Student Material

By clicking on the “Enter Data”, you able to put data manually in Power Query.



We have the same data as we have in Python. Then we need to just click on the type column, Then click on the “Pivot Colum” in Transformation tab. The result of pivoting is shown in below picture.

Then you need to specify which column you are going to put a value and type.

Merging DataSet

in Power Query. I have created two different datasets “Data1” and “Data2”. After creating a dataset, you need to click on the “Merge” column to merge “Data1” with “Data2” .

Data Analytics

Student Material

In merge window, we able to specify which column we are going to merge based on and what type of merge we are interested. After merging, you need to specify what columns you want to show to users. Please follow steps explained in Figure 6-19. Then the output will be there and you able to see the results.

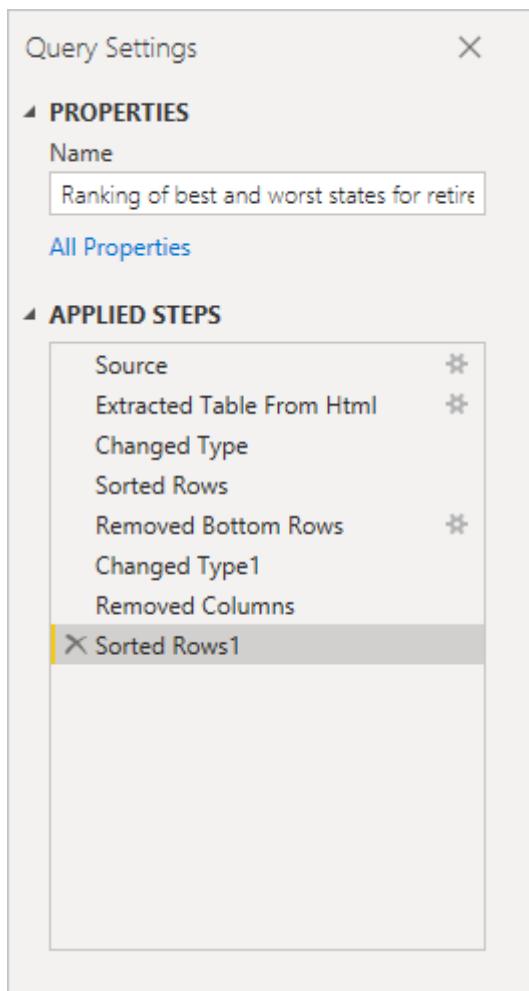
The figure consists of three vertically stacked screenshots from the Microsoft Power Query Editor:

- Merge Step:** Shows the "Merge" dialog with two tables selected: "Table1" (Type) and "Table2" (Value). A dropdown menu at the bottom shows various join types: Left Outer Join (selected), Left Inner Join, Right Outer Join, Right Inner Join, Full Outer Join, Left Only Matching Rows, Right Only Matching Rows, and Left/Right Join Only in Current.
- NestedJoin Dialog:** Shows the "NestedJoin" dialog for the "Type" column. It has a "Search Columns to Expand" input field, a radio button for "Expand" (selected), and a checkbox for "Use original column name as prefix". Under "Select All Columns", the "Value" checkbox is checked (indicated by a red circle).
- Final Table:** Shows the resulting table after the operation. It contains three rows with columns "Type" and "Value". Row 1: Type=a, Value=11, Data2.Value=20. Row 2: Type=b, Value=1, Data2.Value=null. Row 3: Type=c, Value=99, Data2.Value=null.

Data Shaping

When you shape data in Power Query Editor, you provide step-by-step instructions for Power Query Editor to carry out for you to adjust the data as it loads and presents it. The original data source isn't affected; only this particular view of the data is adjusted, or *shaped*.

The steps you specify (such as rename a table, transform a data type, or delete a column) are recorded by Power Query Editor. Each time this query connects to the data source, Power Query Editor carries out those steps so that the data is always shaped the way you specify. This process occurs whenever you use Power Query Editor, or for anyone who uses your shared query, such as on the Power BI service. Those steps are captured, sequentially, in the **Query Settings** pane, under **Applied Steps**. We'll go through each of those steps in the next few paragraphs.



From Getting Started with Power BI Desktop, let's use the retirement data, which we found by connecting to a web data source, to shape that data to fit our needs. We'll add a custom column to calculate rank based on all data being equal factors, and compare this column to the existing column, **Rank**.

1. From the **Add Column** ribbon, select **Custom Column**, which lets you add a custom column.

Data Analytics

Student Material

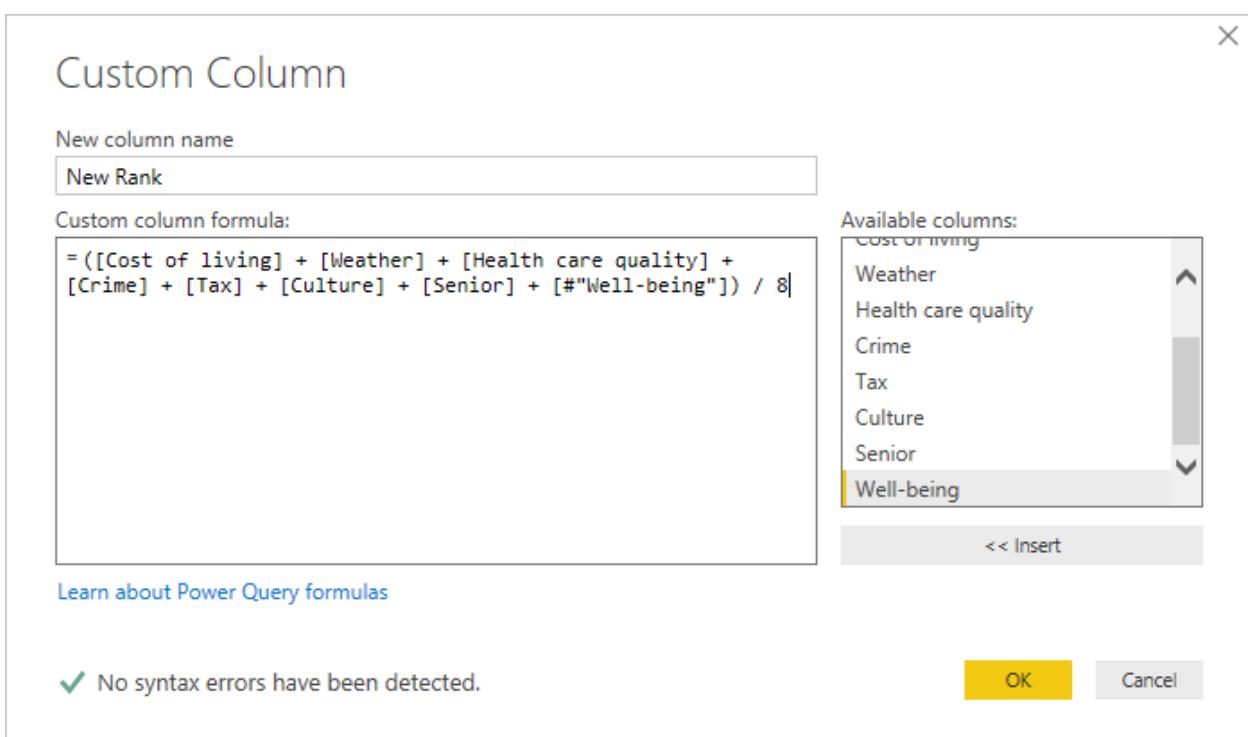
The screenshot shows the Microsoft Power Query Editor interface. The ribbon at the top has tabs: File, Home, Transform, Add Column, View, Tools, and Help. The 'Add Column' tab is selected. A red box highlights the 'Custom Column' icon in the 'General' section of the ribbon. Below the ribbon, the 'Queries [1]' list shows 'Ranking'. The main area displays a 'Custom Column' dialog with the title 'Custom Column' and the sub-instruction 'Create a new column in this table, based on a custom formula.' A formula bar contains the formula: '= Table.RenameColumns(#"Sorted Rows1",{{"Colu...". To the right is a preview table with three rows:

	1	2	3	Overall Rank
a				16
	2			48
		3		38

2. In the **Custom Column** window, in **New column name**, enter *New Rank*.
In **Custom column formula**, enter the following data:

([Cost of living] + [Weather] + [Health care quality] + [Crime] + [Tax] + [Culture] + [Senior] + [#"Well-being"]) / 8

1. Make sure the status message is *No syntax errors have been detected*, and select **OK**.

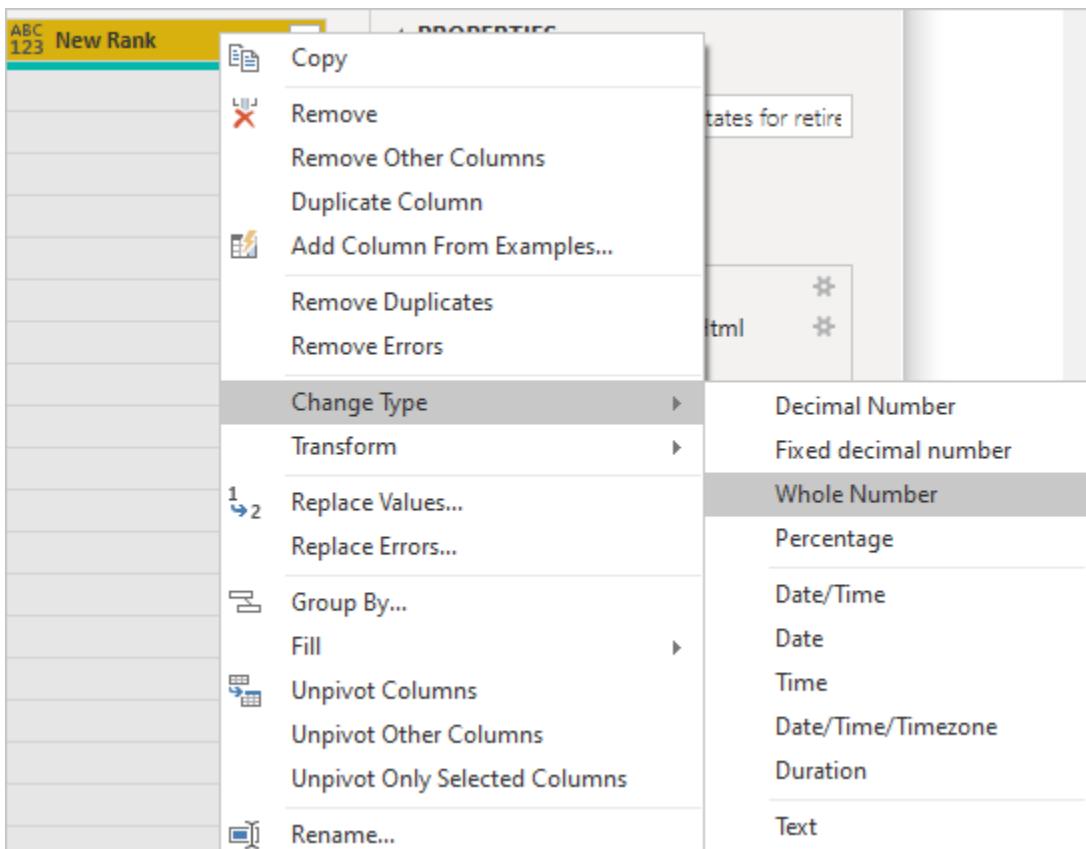


Data Analytics

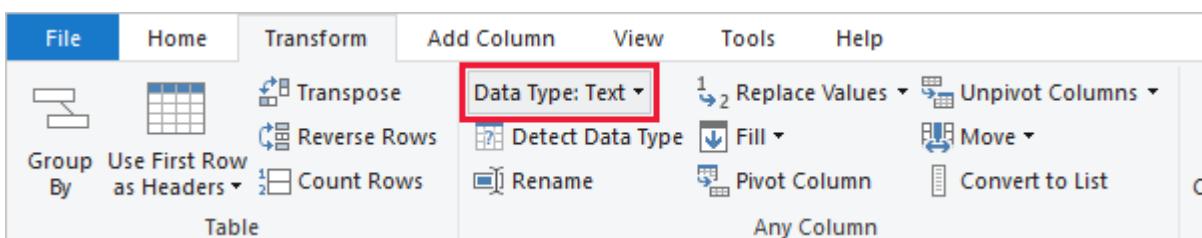
Student Material

- To keep column data consistent, transform the new column values to whole numbers. To change them, right-click the column header, and then select **Change Type > Whole Number**.

If you need to choose more than one column, select a column, hold down **SHIFT**, select additional adjacent columns, and then right-click a column header. You can also use the **CTRL** key to choose non-adjacent columns.



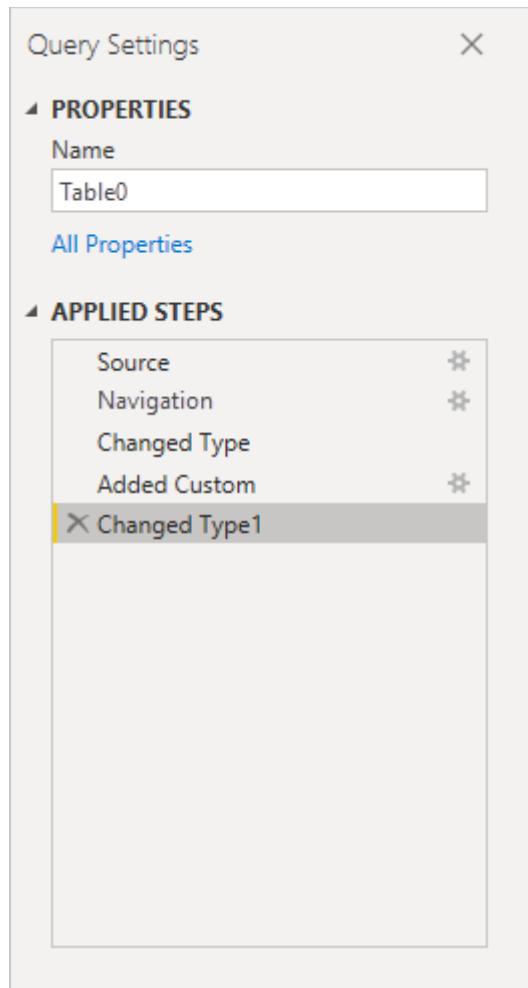
- To transform column data types, in which you transform the current data type to another, select **Data Type Text** from the **Transform** ribbon.



- In **Query Settings**, the **Applied Steps** list reflects any shaping steps applied to the data. To remove a step from the shaping process, select the **X** to the left of the step.

In the following image, the **Applied Steps** list reflects the added steps so far:

- **Source:** Connecting to the website.
- **Extracted Table from Html:** Selecting the table.
- **Changed Type:** Changing text-based number columns from *Text* to *Whole Number*.
- **Added Custom:** Adding a custom column.
- **Changed Type1:** The last applied step.



3.4 Advance shaping

Adjust data

Before we can work with this query, we need to make a few changes to adjust its data:

- Adjust the rankings by removing a column.
We've decided **Cost of living** is a non-factor in our results. After removing this column, we find that the data remains unchanged.
- Fix a few errors.

Data Analytics

Student Material

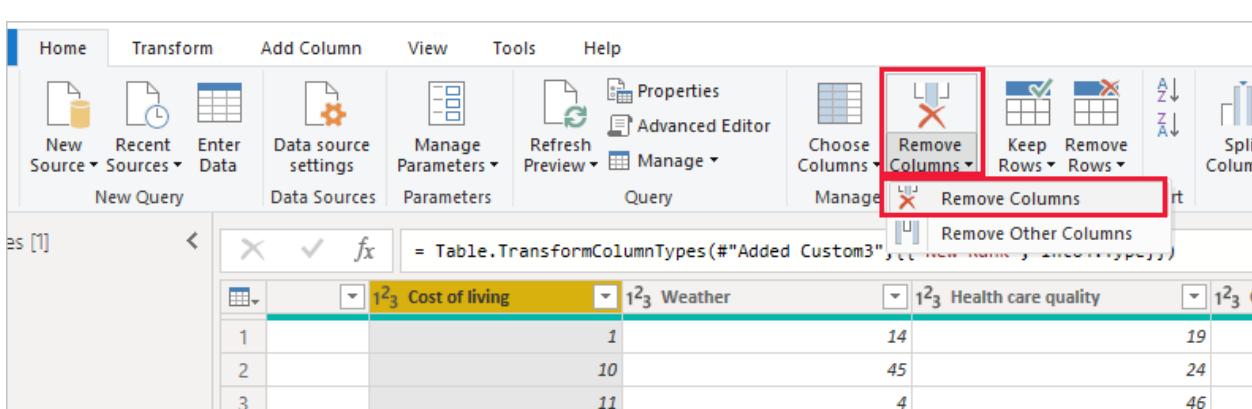
Because we removed a column, we need to readjust our calculations in the **New Rank** column, which involves changing a formula.

- Sort the data.
Sort the data based on the **New Rank** and **Rank** columns.

- Replace the data.
We'll highlight how to replace a specific value and the need of inserting an **Applied Step**.

- Change the table name.
Because **Table 0** isn't a useful descriptor for the table, we'll change its name.

1. To remove the **Cost of living** column, select the column, choose the **Home** tab from the ribbon, and then select **Remove Columns**.



The screenshot shows the Microsoft Power Query Editor interface. The ribbon at the top has the 'Home' tab selected. In the 'Manage' section of the ribbon, there is a dropdown menu for 'Columns'. Within this dropdown, the 'Remove Columns' option is highlighted with a red box. Below the ribbon, a table is displayed with three columns: 'Cost of living', 'Weather', and 'Health care quality'. The 'Cost of living' column is currently selected, indicated by a yellow background. The table contains several rows of data with numerical values.

	1 ² Cost of living	1 ² Weather	1 ² Health care quality
1		1	14
2		10	45
3		11	4
			19
			24
			46

Notice the **New Rank** values haven't changed, due to the ordering of the steps. Because Power Query Editor records the steps sequentially, yet independently, of each other, you can move each **Applied Step** up or down in the sequence.

2. Right-click a step. Power Query Editor provides a menu that lets you do the following tasks:
 - **Rename:** Rename the step.
 - **Delete:** Delete the step.
 - **Delete Until End:** Remove the current step, and all subsequent steps.
 - **Move before:** Move the step up in the list.
 - **Move after:** Move the step down in the list.
3. Move up the last step, **Removed Columns**, to just above the **Added Custom** step.

Data Analytics

Student Material

The screenshot shows the 'Query Settings' dialog box in Power BI. The 'APPLIED STEPS' pane lists several steps: 'Source', 'Extracted Table From Html', 'Changed Type', 'Added Custom', and 'Changed Type1'. A context menu is open over the 'Added Custom' step, with the 'Move before' option highlighted.

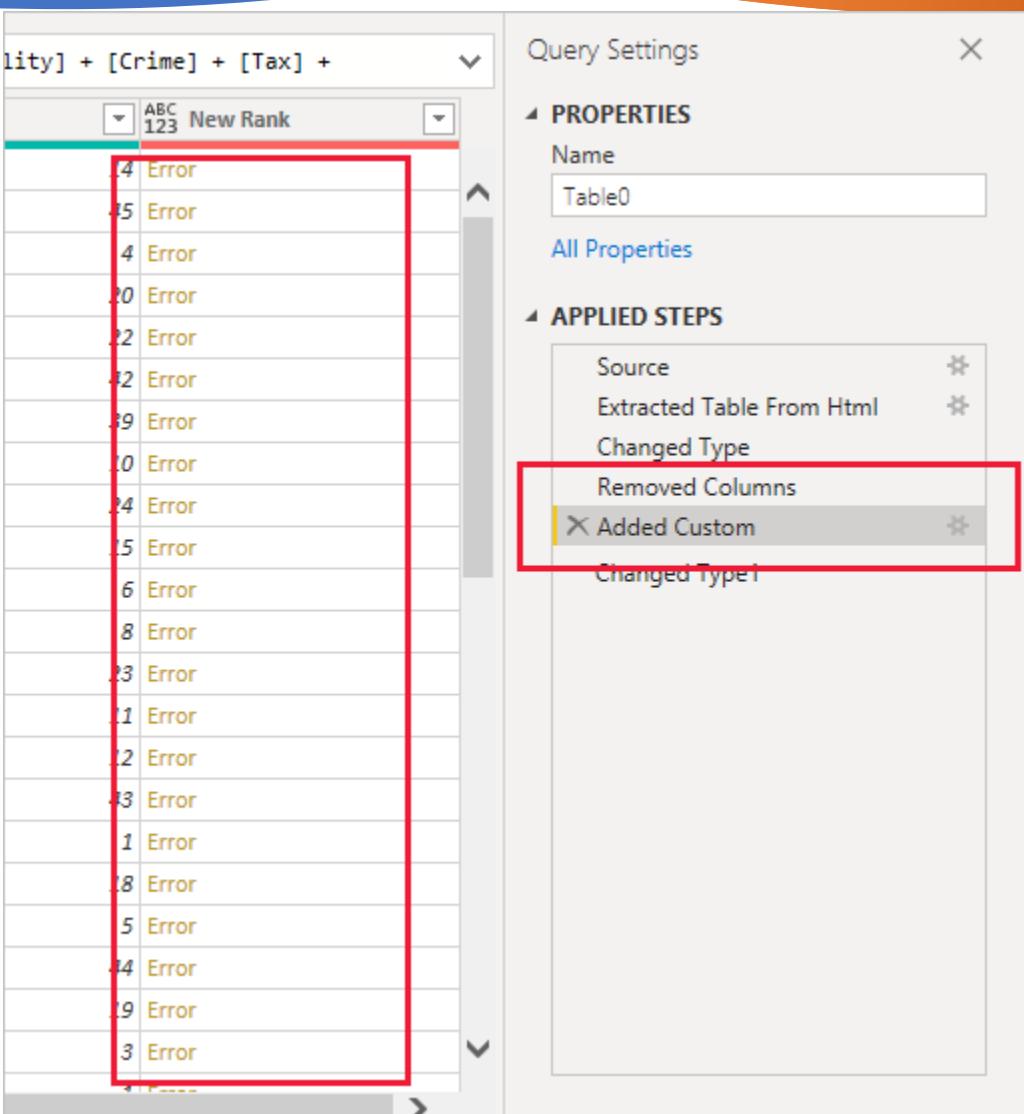
- Source
- Extracted Table From Html
- Changed Type
- Added Custom
- Changed Type1
- Removed Column⁻¹
 - Edit Settings
 - Rename
 - Delete
 - Delete Until End
 - Insert Step After
 - Move before
 - Move after
 - Extract Previous

4. Select the **Added Custom** step.

Notice the data now shows *Error*, which we'll need to address.

Data Analytics

Student Material



The screenshot shows the Power Query Editor interface. On the left is a preview pane displaying a table with two columns: 'State' and 'New Rank'. The 'New Rank' column contains numerous 'Error' entries. A red box highlights the first few rows of the 'New Rank' column. On the right is the 'Query Settings' pane, which includes a 'PROPERTIES' section where 'Name' is set to 'Table0', and an 'APPLIED STEPS' section. The 'Applied Steps' list shows several steps: 'Source', 'Extracted Table From Html', 'Changed Type', 'Removed Columns', and 'Added Custom'. The 'Added Custom' step is highlighted with a red box.

There are a few ways to get more information about each error. If you select the cell without clicking on the word *Error*, Power Query Editor displays the error information.



The screenshot shows the Power Query Editor with a specific error highlighted. The formula bar at the top shows the formula: = "#Added Custom3" {11} [New Rank]. Below it, a yellow tooltip provides detailed error information: 'Expression.Error: The field 'Cost of living' of the record wasn't found.' It lists the following details: State=Iowa, Weather=8, Health care quality=15, Crime=20, Tax=34, Culture=12, Senior=12, Well-being=8. A 'Go To Error' button is visible in the top right corner of the tooltip.

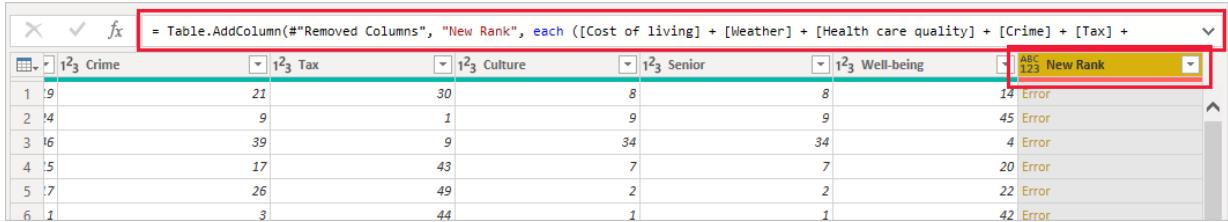
If you select the word *Error* directly, Power Query Editor creates an **Applied Step** in the **Query Settings** pane and displays information about the error.

- Because we don't need to display information about the errors, select **Cancel**.

Data Analytics

Student Material

6. To fix the errors, select the **New Rank** column, then display the column's data formula by selecting the **Formula Bar** checkbox from the **View** tab.



The screenshot shows the Power Query Editor interface. A red box highlights the formula bar at the top, which contains the formula: `= Table.AddColumn(#"Removed Columns", "New Rank", each ([Cost of living] + [Weather] + [Health care quality] + [Crime] + [Tax] + [Culture] + [Senior]) / 7)`. Below the formula bar is a table with six columns: Crime, Tax, Culture, Senior, Well-being, and New Rank. The New Rank column contains numerical values (e.g., 14, 45, 4, 20, 22, 42) followed by the word 'Error' in orange. The table has 6 rows, indexed 1 to 6.

	Crime	Tax	Culture	Senior	Well-being	New Rank
1	9	21	30	8	8	14 Error
2	14	9	1	9	9	45 Error
3	16	39	9	34	34	4 Error
4	5	17	43	7	7	20 Error
5	7	26	49	2	2	22 Error
6	1	3	44	1	1	42 Error

7. Remove the *Cost of living* parameter and decrement the divisor, by changing the formula as follows:

```
Table.AddColumn(#"Removed Columns", "New Rank", each ([Weather] + [Health care quality] + [Crime] + [Tax] + [Culture] + [Senior] + [#"Well-being"]) / 7)
```

8. Select the green checkmark to the left of the formula box or press **Enter**.

Power Query Editor replaces the data with the revised values and the **Added Custom** step completes with no errors.

Note

You can also select **Remove Errors**, by using the ribbon or the right-click menu, which removes any rows that have errors. However, we didn't want to do so in this tutorial because we wanted to preserve the data in the table.

9. Sort the data based on the **New Rank** column. First, select the last applied step, **Changed Type1** to display the most recent data. Then, select the drop-down located next to the **New Rank** column header and select **Sort Ascending**.

Data Analytics

Student Material

The screenshot shows the Power Query Editor interface. On the left, there's a 'Number Filters' pane with a search bar and a list of numbers from 16 to 30, each with a checked checkbox. On the right, the 'Query Settings' dialog is open, showing the 'PROPERTIES' section with 'Name' set to 'Table0' and the 'APPLIED STEPS' section which includes 'Changed Type1'.

PREVIEW DOWNLOADED ON WEDNESDAY

The data is now sorted according to **New Rank**. However, if you look at the **Rank** column, you'll notice the data isn't sorted properly in cases where the **New Rank** value is a tie. We'll fix it in the next step.

10. To fix the data sorting issue, select the **New Rank** column and change the formula in the **Formula Bar** to the following formula:

```
= Table.Sort(#"Changed Type1",{{"New Rank", Order.Ascending}, {"Rank", Order.Ascending}})
```

11. Select the green checkmark to the left of the formula box or press **Enter**.

The rows are now ordered in accordance with both **New Rank** and **Rank**. In addition, you can select an **Applied Step** anywhere in the list, and continue shaping the data at that point in the sequence. Power Query Editor automatically inserts a new step directly after the currently selected **Applied Step**.

Data Analytics

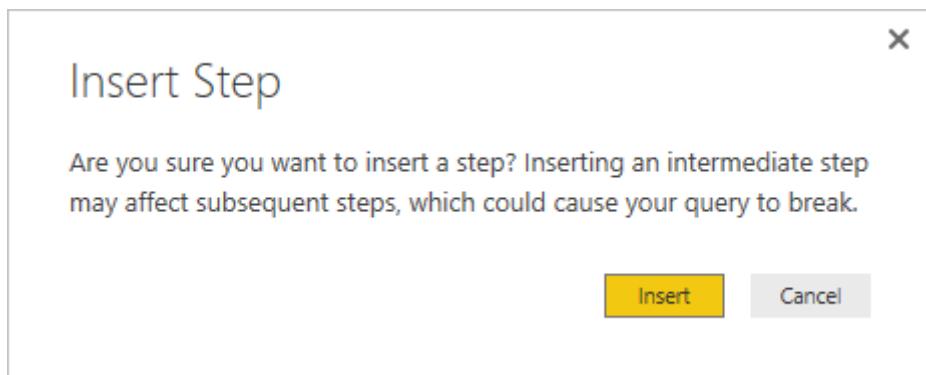
Student Material

12. In **Applied Step**, select the step preceding the custom column, which is the **Removed Columns** step. Here we'll replace the value of the **Weather** ranking in Arizona. Right-click the appropriate cell that contains Arizona's **Weather** ranking, and then select **Replace Values**. Note which **Applied Step** is currently selected.

	A ^B _C State	A ^B _C Cost of living	A ^B _C Weather	A ^B _C Health care quality	A ^B _C Crime
31	Indiana	29	3	27	41
32	Connecticut	30	46	7	8
33	Maine	31	35	3	1
34	Delaware	32	30	36	9
35	Colorado	33	36	32	22
36	Pennsylvania	34	28	13	15
37	Utah	35	21	21	47
38	Louisiana	36	29	48	48
39	New Mexico	37	26	49	38
40	Arizona	38	33	43	
41	Virginia	39	32	6	
42	Minnesota	40	31	14	
43	South Carolina	41	27	45	
44	New Jersey	42	48	5	
45	California	43	49	34	
46	Oregon	44	37	30	

13. Select **Insert**.

Because we're inserting a step, Power Query Editor warns us about the danger of doing so; subsequent steps could cause the query to break.



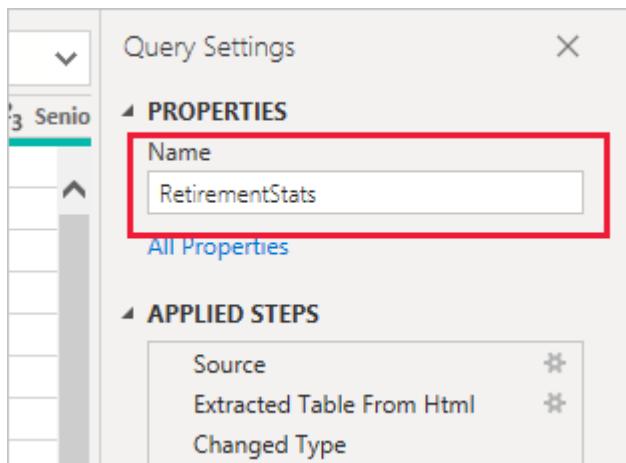
14. Change the data value to 51.

Power Query Editor replaces the data for Arizona. When you create a new **Applied Step**, Power Query Editor names it based on the action; in this case, **Replaced Value**. If you have more than one step with the same name in your query, Power Query Editor adds a number (in sequence) to each subsequent **Applied Step** to differentiate between them.

15. Select the last **Applied Step**, **Sorted Rows**.

Notice the data has changed regarding Arizona's new ranking. This change occurs because we inserted the **Replaced Value** step in the correct location, before the **Added Custom** step.

16. Lastly, we want to change the name of that table to something descriptive. In the **Query Settings** pane, under **Properties**, enter the new name of the table, and then select **Enter**. Name this table *RetirementStats*.



When we start creating reports, it's useful to have descriptive table names, especially when we connect to multiple data sources, which are listed in the **Fields** pane of the **Report** view.

We've now shaped our data to the extent we need to. Next let's connect to another data source, and combine data.

3.5 Beyond Initial Shaping

Combine data

The data about various states is interesting, and will be useful for building additional analysis efforts and queries. But there's one problem: most data out there uses a two-letter abbreviation for state codes, not the full name of the state. We need a way to associate state names with their abbreviations.

We're in luck; there's another public data source that does just that, but it needs a fair amount of shaping before we can connect it to our retirement table. To shape the data, follow these steps:

1. From the **Home** ribbon in Power Query Editor, select **New Source > Web**.
2. Enter the address of the website for state abbreviations, https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations, and then select **Connect**.

The Navigator displays the content of the website.

Data Analytics

Student Material

The screenshot shows the Microsoft Power BI Navigator interface. On the left, there's a search bar and a 'Display Options' dropdown. Below that is a tree view of 'HTML Tables [5]': 'Codes and abbreviations for U S states, fe...', 'Table 1', 'Matches between USPS and USCG cod...', 'Table 2', 'Table 3', and 'Suggested Tables [7]'. The 'Codes and abbreviations for U S states, fe...' node is selected, indicated by a yellow checkmark. On the right, there are two tabs: 'Table View' (selected) and 'Web View'. The 'Table View' tab shows a preview of a table with two columns: 'Column1' and 'Column2'. The table lists US states and their abbreviations, along with their status (e.g., Federal state, State). The preview includes a header row and several data rows. At the bottom of the preview are navigation arrows and scroll bars. Below the preview are 'OK' and 'Cancel' buttons.

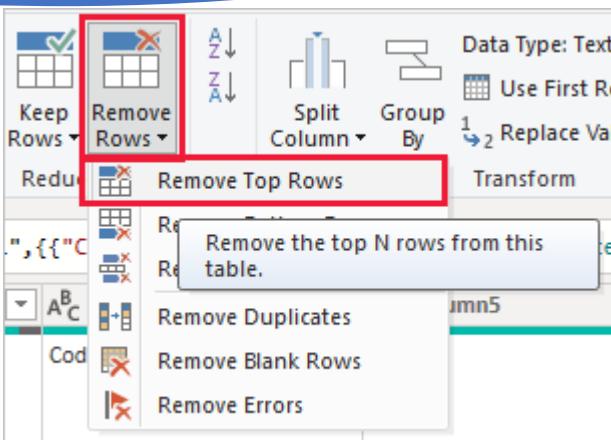
3. Select **Codes and abbreviations**.

Tip

It will take quite a bit of shaping to pare this table's data down to what we want. Is there a faster or easier way to accomplish the steps below? Yes, we could create a *relationship* between the two tables, and shape the data based on that relationship. The following steps are still good to learn for working with tables; however, relationships can help you quickly use data from multiple tables.

To get the data into shape, follow these steps:

1. Remove the top row. Because it's a result of the way that the web page's table was created, we don't need it. From the **Home** ribbon, select **Remove Rows > Remove Top Rows**.

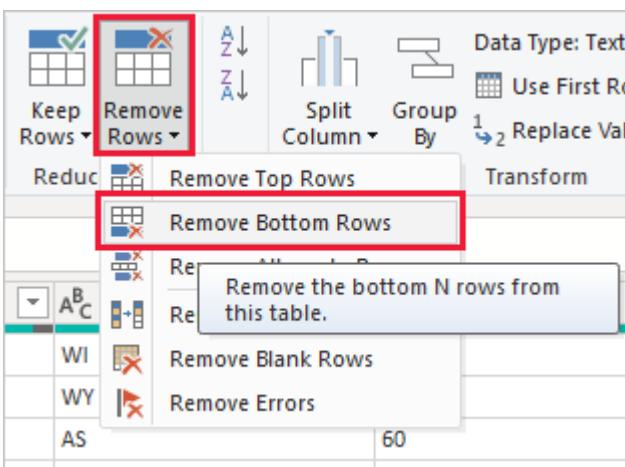


The **Remove Top Rows** window appears, letting you specify how many rows you want to remove.

Note

If Power BI accidentally imports the table headers as a row in your data table, you can select **Use First Row As Headers** from the **Home** tab, or from the **Transform** tab in the ribbon, to fix your table.

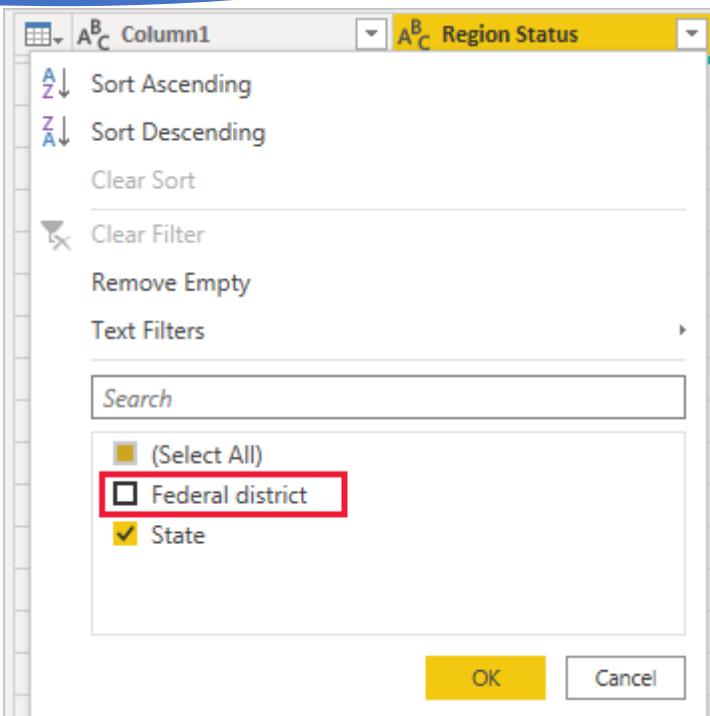
2. Remove the bottom 26 rows. These rows are U.S. territories, which we don't need to include. From the **Home** ribbon, select **Remove Rows > Remove Bottom Rows**.



3. Because the RetirementStats table doesn't have information for Washington DC, we need to filter it from our list. Select the **Region Status** drop-down, then clear the checkbox beside **Federal district**.

Data Analytics

Student Material



4. Remove a few unneeded columns. Because we need only the mapping of each state to its official two-letter abbreviation, we can remove several columns. First select a column, then hold down the **CTRL** key and select each of the other columns to be removed. From the **Home** tab on the ribbon, select **Remove Columns > Remove Columns**.

A screenshot of the Microsoft Power Query Editor showing the 'Home' tab of the ribbon. The 'Remove Columns' button is highlighted with a red box. A tooltip window appears, stating 'Remove the currently selected columns from this table.' Below the ribbon is a table with five columns: Column1, Region Status, Column3, Column4, and Column5. The data in the table is as follows:

	Column1	Region Status	Column3	Column4	Column5
1	Alabama	State	US-AL	AL	01
2	Alaska	State	US-AK	AK	02
3	Arizona	State	US-AZ	AZ	04
4	Arkansas	State	US-AR	AR	05

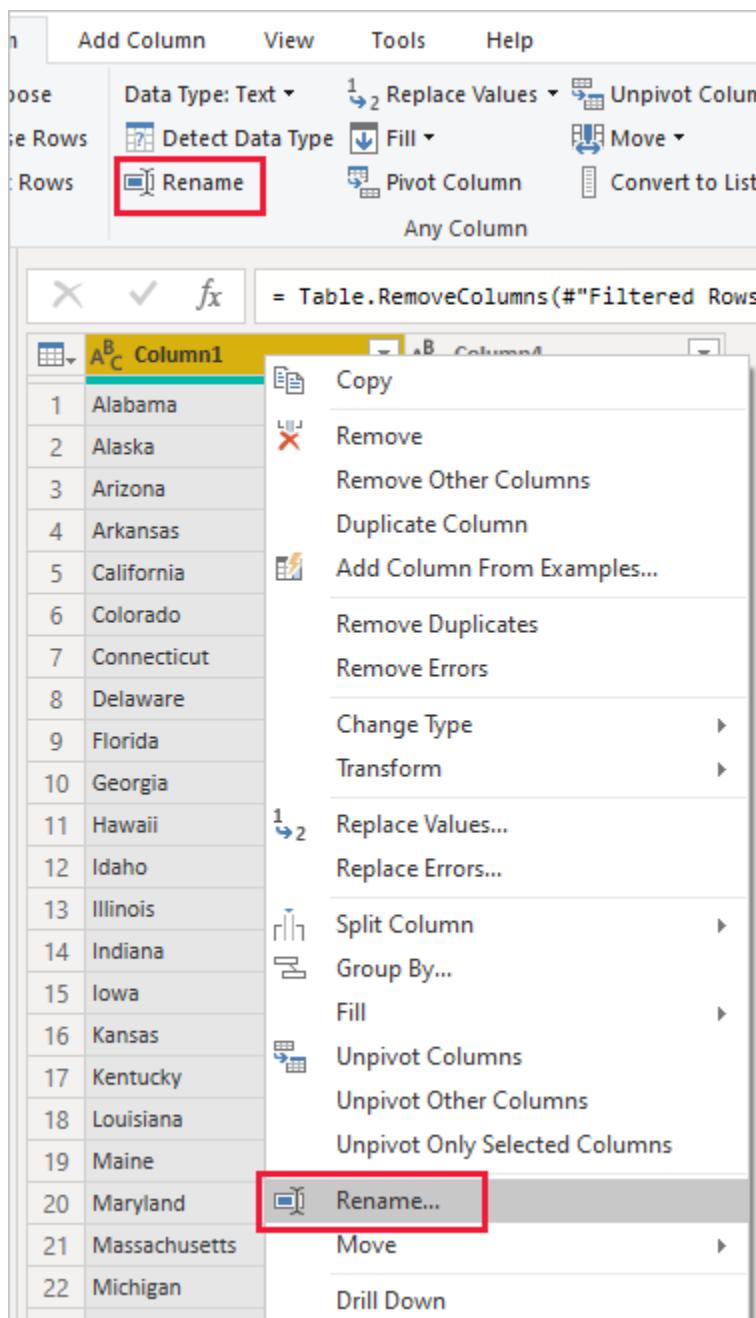
Note

This is a good time to point out that the *sequence* of applied steps in Power Query Editor is important, and can affect how the data is shaped. It's also important to consider how one step may impact another subsequent step; if you remove a step from the Applied Steps, subsequent steps may not behave as originally intended, because of the impact of the query's sequence of steps.

Note

When you resize the Power Query Editor window to make the width smaller, some ribbon items are condensed to make the best use of visible space. When you increase the width of the Power Query Editor window, the ribbon items expand to make the most use of the increased ribbon area.

5. Rename the columns and the table. There are a few ways to rename a column: First, select the column, then either select **Rename** from the **Transform** tab on the ribbon, or right-click and select **Rename**. The following image has arrows pointing to both options; you only need to choose one.



6. Rename the columns to *State Name* and *State Code*. To rename the table, enter the **Name** in the **Query Settings** pane. Name this table *StateCodes*.

3.6 DAX overview

Data Analysis Expressions (DAX) is a formula expression language used in Analysis Services, Power BI, and Power Pivot in Excel. DAX formulas include functions, operators, and values to perform advanced calculations and queries on data in related tables and columns in tabular data models.

DAX function

Aggregation functions - These functions calculate a (scalar) value such as count, sum, average, minimum, or maximum for all rows in a column or table as defined by the expression.

Date and time functions - These functions in DAX are similar to date and time functions in Microsoft Excel. However, DAX functions are based on the datetime data types used by Microsoft SQL Server.

Filter functions - These functions help you return specific data types, look up values in related tables, and filter by related values. Lookup functions work by using tables and relationships between them. Filtering functions let you manipulate data context to create dynamic calculations.

Financial functions - These functions are used in formulas that perform financial calculations, such as net present value and rate of return.

Information functions - These functions look at a table or column provided as an argument to another function and returns whether the value matches the expected type. For example, the ISERROR function returns TRUE if the value you reference contains an error.

Logical functions - These functions return information about values in an expression. For example, the TRUE function lets you know whether an expression that you are evaluating returns a TRUE value.

Math and Trig functions - Mathematical functions in DAX are similar to Excel's mathematical and trigonometric functions. However, there are some differences in the numeric data types used by DAX functions.

Other functions - These functions perform unique actions that cannot be defined by any of the categories most other functions belong to.

Parent and Child functions - These functions help users manage data that is presented as a parent/child hierarchy in their data models.

Relationship functions - These functions are for managing and utilizing relationships between tables. For example, you can specify a particular relationship to be used in a calculation.

Statistical functions - These functions calculate values related to statistical distributions and probability, such as standard deviation and number of permutations.

Table manipulation functions - These functions return a table or manipulate existing tables.

Text functions - With these functions, you can return part of a string, search for text within a string, or concatenate string values. Additional functions are for controlling the formats for dates, times, and numbers.

Time intelligence functions - These functions help you create calculations that use built-in knowledge about calendars and dates. By using time and date ranges in combination with aggregations or calculations, you can build meaningful comparisons across comparable time periods for sales, inventory, and so on.

DAX Statements

DEFINE (Keyword) Defines entities that exist only for the duration of a DAX query.

EVALUATE (Keyword) A statement required to execute a DAX query.

ORDER BY (Keyword) Defines one or more expressions used to sort results of a DAX query.

VAR (Keyword) Stores the result of an expression as a named variable, which can then be passed as an argument to other measure expressions.

DAX Calculations

DAX formulas are used in measures, calculated columns, calculated tables, and row-level security.

Measures

Measures are dynamic calculation formulas where the results change depending on context. Measures are used in reporting that support combining and filtering model data by using multiple attributes such as a Power BI report or Excel PivotTable or PivotChart. Measures are created by using the DAX formula bar in the model designer.

A formula in a measure can use standard aggregation functions automatically created by using the Autosome feature, such as COUNT or SUM, or you can define your own formula by using the DAX formula bar. Named measures can be passed as an argument to other measures.

When you define a formula for a measure in the formula bar, a Tooltip feature shows a preview of what the results would be for the total in the current context, but otherwise the results are not immediately output anywhere. The reason you cannot see the (filtered) results of the calculation immediately is because the result of a measure cannot be determined without context. To evaluate a measure requires a reporting client application that can provide the context needed to retrieve the data relevant to each cell and then evaluate the

expression for each cell. That client might be an Excel PivotTable or PivotChart, a Power BI report, or a table expression in a DAX query in SQL Server Management Studio (SSMS).

Regardless of the client, a separate query is run for each cell in the results. That is to say, each combination of row and column headers in a PivotTable, or each selection of slicers and filters in a Power BI report, generates a different subset of data over which the measure is calculated.

For example, using this very simple measure formula:

DAXCopy

Total Sales = SUM ([Sales Amount])

When a user places the Total Sales measure in a report, and then places the Product Category column from a Product table into Filters, the sum of Sales Amount is calculated and displayed for each product category.

Unlike calculated columns, the syntax for a measure includes the measure's name preceding the formula. In the example just provided, the name Total Sales appears preceding the formula. After you've created a measure, the name and its definition appear in the reporting client application Fields list and depending on perspectives and roles is available to all users of the model.

Calculated columns

A calculated column is a column that you add to an existing table (in the model designer) and then create a DAX formula that defines the column's values. When a calculated column contains a valid DAX formula, values are calculated for each row as soon as the formula is entered. Values are then stored in the in-memory data model.

For example, in a Date table, when the formula is entered into the formula bar:

DAXCopy

= [Calendar Year] & " Q" & [Calendar Quarter]

A value for each row in the table is calculated by taking values from the Calendar Year column (in the same Date table), adding a space and the capital letter Q, and then adding the values from the Calendar Quarter column (in the same Date table). The result for each row in the calculated column is calculated immediately and appears, for example, as 2017 Q1. Column values are only recalculated if the table or any related table is processed (refresh) or the model is unloaded from memory and then reloaded, like when closing and reopening a Power BI Desktop file.

Calculated tables

A calculated table is a computed object, based on a formula expression, derived from all or part of other tables in the same model. Instead of querying and loading values into your new table's columns from a data source, a DAX formula defines the table's values.

Calculated tables can be helpful in a role-playing dimension. An example is the Date table, as OrderDate, ShipDate, or DueDate, depending on the foreign key relationship. By creating a calculated table for ShipDate explicitly, you get a standalone table that is available for queries, as fully operable as any other table. Calculated tables are also useful when configuring a filtered rowset, or a subset or superset of columns from other existing tables. This allows you to keep the original table intact while creating variations of that table to support specific scenarios.

Calculated tables support relationships with other tables. The columns in your calculated table have data types, formatting, and can belong to a data category. Calculated tables can be named and surfaced or hidden just like any other table. Calculated tables are re-calculated if any of the tables it pulls data from are refreshed or updated.

Row-level security

With row-level security, a DAX formula must evaluate to a Boolean TRUE/FALSE condition, defining which rows can be returned by the results of a query by members of a particular role.

For example, for members of the Sales role, the Customers table with the following DAX formula:

```
DAXCo= Customers[Country] = "USA"
```

Members of the Sales role will only be able to view data for customers in the USA, and aggregates, such as SUM are returned only for customers in the USA. Row-level security is not available in Power Pivot in Excel.

When defining row-level security by using DAX formula, you are creating an allowed row set. This does not deny access to other rows; rather, they are simply not returned as part of the allowed row set. Other roles can allow access to the rows excluded by the DAX formula. If a user is a member of another role, and that role's row-level security allows access to that particular row set, the user can view data for that row.

Row-level security formulas apply to the specified rows as well as related rows. When a table has multiple relationships, filters apply security for the relationship that is active. Row-level security formulas will be intersected with other formulas defined for related tables.

Advanced DAX

The Advanced DAX course is designed to build on the fundamentals taught in the Introduction to DAX course. This course focuses on the advanced and difficult to comprehend DAX concepts like evaluation context, security, iterator functions and advanced DAX functions.

3.7 Publishing Data Set:

A training data set is a data set of examples used during the learning process and is used to fit the parameters (e.g., weights) of, for example, a classifier. For classification tasks, a supervised learning algorithm looks at the training data set to determine, or learn, the optimal combinations of variables that will generate a good predictive model. The goal is to produce a trained (fitted) model that generalizes well to new, unknown data. The fitted model is evaluated using “new” examples from the held-out datasets (validation and test datasets) to estimate the model’s accuracy in classifying new data. To reduce the risk of issues such as over-fitting, the examples in the validation and test datasets should not be used to train the model. Most approaches that search through training data for empirical relationships tend to over fit the data, meaning that they can identify and exploit apparent relationships in the training data that do not hold in general.

3.8 SUMMARIZING DATA

Why do we summarize? We summarize data to “simplify” the data and quickly identify what looks “normal” and what looks odd. The distribution of a variable shows what values the variable takes and how often the variable takes these values.

The two most useful ways of describing the distribution of data are:

- The typical: This describes the center—or middle—of the data. This way of describing the center is also called a “measure of central tendency”.
- The spread of the values around the center: This describes how densely the data is distributed around the center. This is also called a “measure of dispersion”.

These two ways of describing the data are also referred to as descriptive statistics.

1. The Center: What is Typical? (Central Tendencies)

The three common ways of looking at the center are average (also called mean), mode and median. All three summarize a distribution of the data by describing the typical value of a variable (average), the most frequently repeated number (mode), or the number in the middle of all the other numbers in a data set (median). In this module, we are going to focus on the average. The average is the most appropriate way to measure the center for interval/continuous data (e.g, numbers of registered voters). To calculate the average, we add up all the numbers for a variable and then divide by how many numbers there are. Put another way the average (mean) is the sum divided by the count.

Simple Example

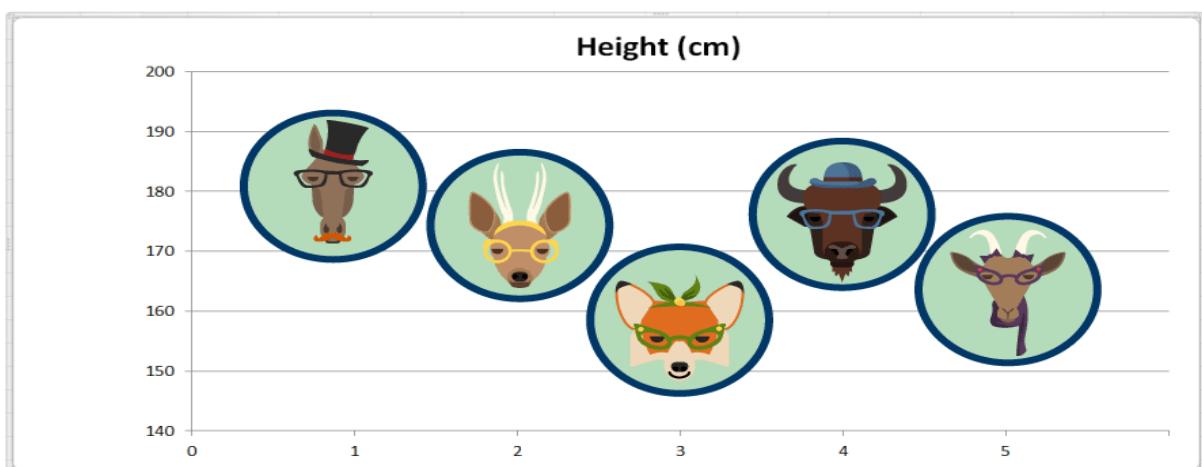
In the example dataset below, we have information about the names of some animals. We also have measurements of the height of each animal. The dataset has two variables – name and height – and five observations. Here is the dataset:

Data Analytics

Student Material

	A	B
1	Name	Height (cm)
2	Harry the Horse	181
3	Dana the Deer	175
4	Fran the Fox	159
5	Bob the Buffalo	177
6	Gracie the Goat	165

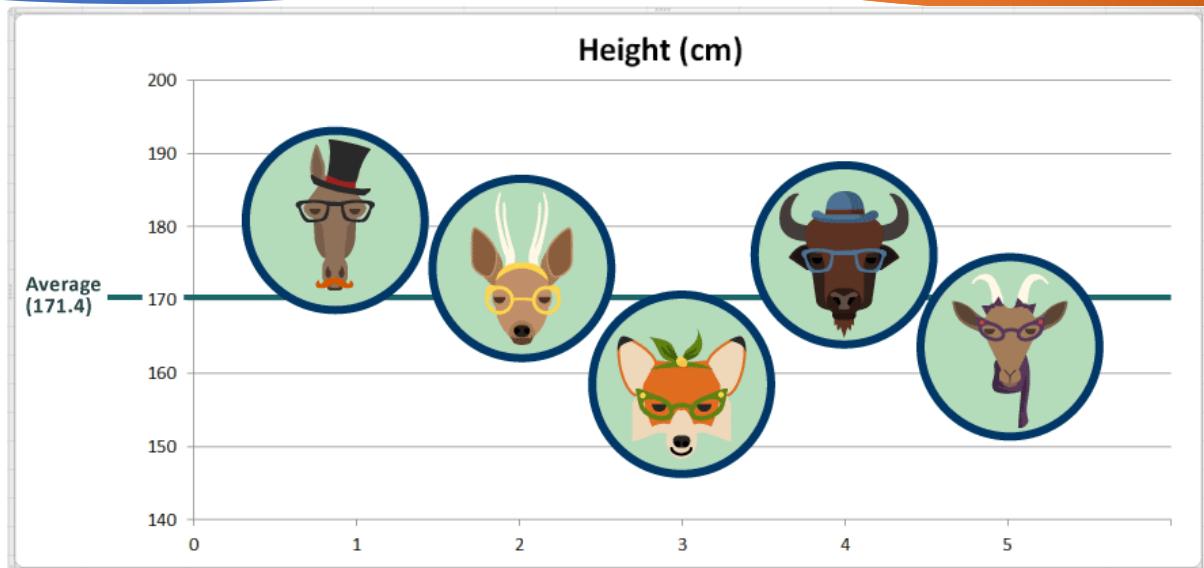
Here we've made a quick chart that plots the height of each animal:



To calculate the Average height (in cm) we sum up all the values and divide by the total count of observations:

$$\text{Average height} = (181 + 175 + 159 + 177 + 165) \div 5 = 857 \div 5 = 171.4$$

The average value for height is 171.4 centimeters. Here we have added a reference line marking the average on our chart so we can see how that looks:



2. The Spread: How is the data distributed around the center? (Measures of Dispersion)

Looking at the spread of the distribution of data tells us about the amount of variation, or diversity, within the data. The three measures of the spread of the data are the range, the standard deviation, and the variance.

Range

This is the difference between the largest and the smallest values. It is the distance between the extremes. To calculate the range, we take the maximum value and subtract the minimum value.

In our height dataset, what is the largest value (maximum)? 181 cm

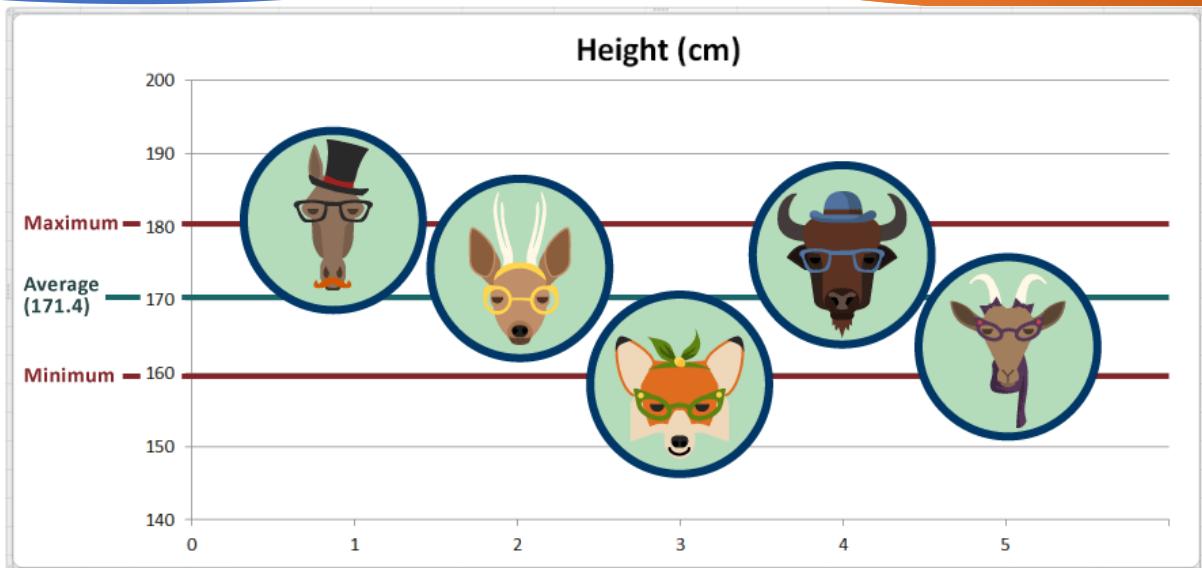
In the same example, what is the smallest value (minimum)? 159 cm

The range in our small dataset of heights is $181 - 159 = 22$ cm

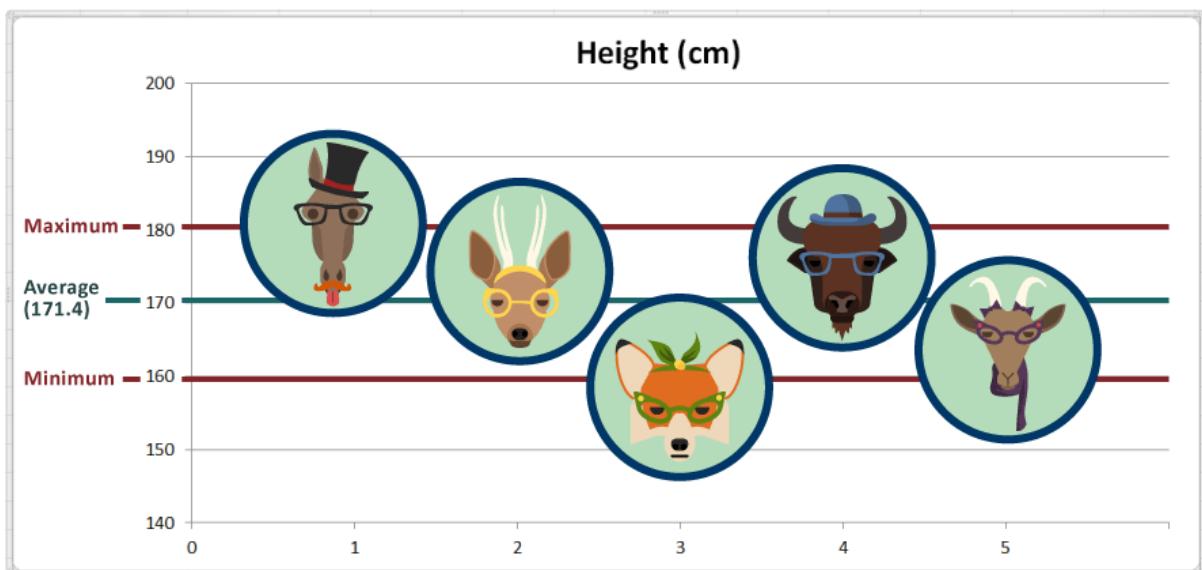
Here we added some reference lines on the chart to indicate the maximum and minimum:

Data Analytics

Student Material



In practical terms, the animal with the maximum value is the tallest, and the animal with the minimum value is the shortest. So Harry the Horse is the tallest, and Fran the Fox is the shortest.



While the range gives us the endpoints (i.e., extremes), it does not tell us anything about how tightly or loosely the data are distributed between those two endpoints. We also do not know whether more of the data is closer to the average, the maximum or the minimum. From our chart, it looks like just over half of the animals are tall (i.e., above the average height).

Two other related measures of dispersion—the variance and the standard deviation—can help us answer these questions. They provide a numerical summary of how much the data are scattered.

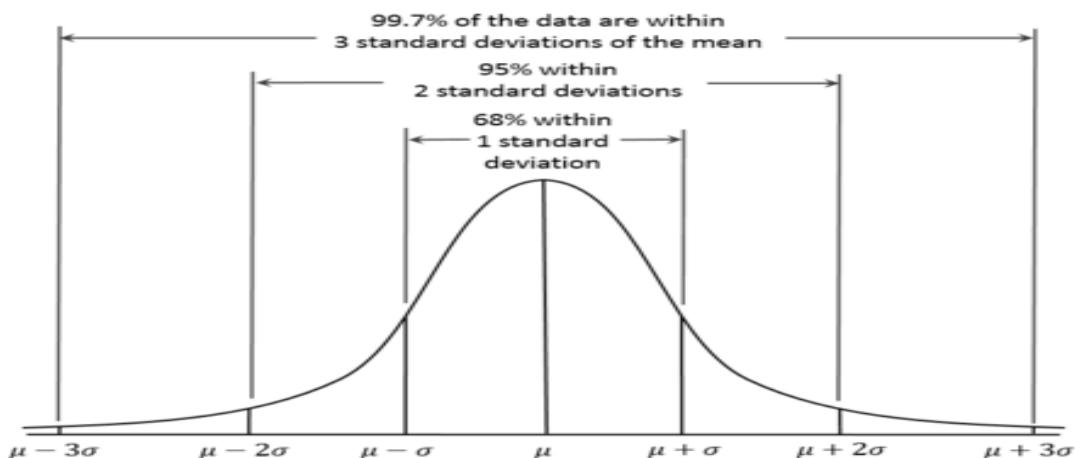
Data Analytics

Student Material

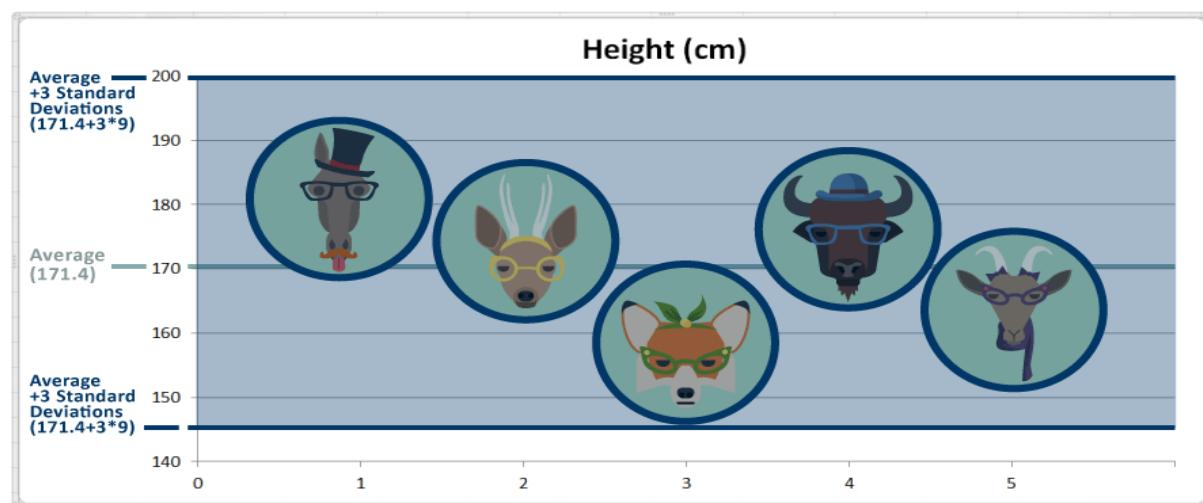
Standard Deviation

The standard deviation provides us with a standard way of knowing what is normal² given the average. A really helpful attribute of the standard deviation is that it is expressed in the same units as the data itself. The standard deviation is like an “index of variability,” because it is proportional to the scatter of the data. The standard deviation is larger for more diverse distributions (i.e., the data are widely scattered). The standard deviation is smaller for less diverse distributions (i.e., the data are clustered together).

The standard deviation is very useful for understanding the spread of a variable. For most “normally” distributed data, generally almost all of the values will be within three standard deviations of the average. In statistics, this is sometimes referred to as the 68-95-99.7 rule. About 68.27% of the values lie within 1 standard deviation of the average (mean). Similarly, approximately 95.45% of the values lie within 2 standard deviations of the mean. Nearly all (99.73%) of the values lie within 3 standard deviations of the mean.



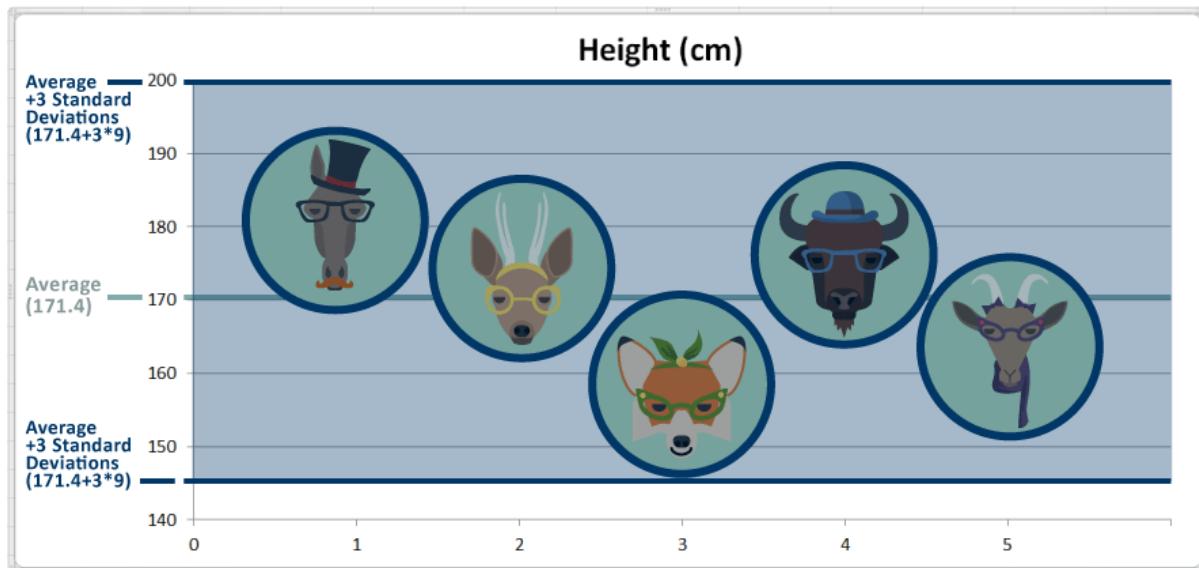
In the sample animal heights dataset, we've calculated the standard deviation for heights. It is 9.1 cm³. On the chart we have shaded the area to show what data is within three standard deviations (9.1 x 3) of the average. Any data within this range is “normal.”



Data Analytics

Student Material

The standard deviation gives us a standardized way of knowing what is normal, what is extra-large or what is extra small. We know that Fran the Fox is short. When we consider the standard deviation and that nearly all (99.73%) of all values are generally within 3 standard deviations, we can conclude that Fran is short but not abnormally short.



3.10 Creating New Variables

What is the DAX Variable?

In DAX calculation, we can use variables to make the calculations easier to understand. When you are writing any complex or **nested** expression using DAX functions, variables can help to break these **complex** calculations into **smaller**, more useful sections.

Why we need DAX variable



You can use variables in any form of DAX calculations and it includes calculated measures, columns and tables. A type of variable is nothing but the calculated object type in DAX

Structure of Variable

Structure

```
VAR varname = expression
```

```
RETURN expression
```

1. To close a declared variable scope, there **must** be a **RETURN** statement.

```
VAR varA=2  
RETURN varA * 2
```

2. You can declare **multiple** variables with the **same** layer of **scope** and a **single** RETURN statement.

```
VAR varA=2  
  
VAR varB=varA+2  
  
RETURN varB*3
```

3. Variables can be assigned once and **cannot be reassigned**. Give an error if you like below

```
VAR varA=2  
  
varA=varA+2
```

Get Data

For this case study, I consider the US Superstore dataset from [Kaggle](#).

- Let's start with the **Get Data** option under the **Home** tab. As this is a CSV file, select the **Text/CSV** option from the drop-down list
- Select the file named **US Superstore data.csv**
- After selecting the file, data will be displayed in the below format

Data Analytics

Student Material

The screenshot shows the Microsoft Power BI Data Editor window. The title bar says "US Superstore data.csv". The top menu has tabs for "Data", "Transform", "Visuals", and "Report". The "Data" tab is selected. In the center, there's a preview of the data with 20 rows. The columns are: Row ID, Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, Segment, Country, and City. The "Data Type Detection" section at the top right says "Based on first 200 rows". At the bottom right are three buttons: "Load" (yellow), "Transform Data", and "Cancel".

- Click on Load and save data.

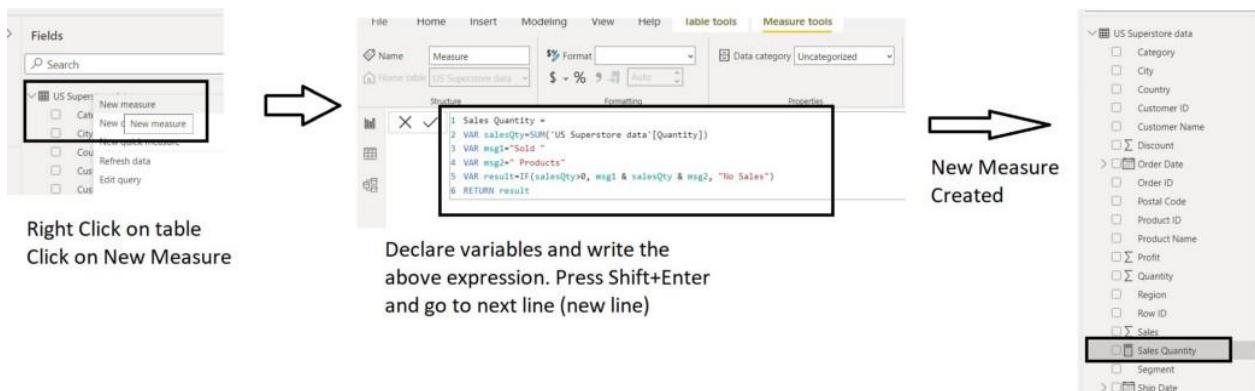
VARIABLES WITH TEXT

Variables can store numeric data as well as text. Using this feature, you can create a dynamic text-based measure that can be varied depending on filter selection. Let's see one example.

- Create one text-based calculated measure (Sales Quantity) using variable
- There are 4 variables. one for the direct measure, two for hard-coded messages and the final result variable where you are using IF clause condition and concatenate operator.
Here you are using the result as a return statement.
- To make the next line or new line, press Shift + Enter.

Data Analytics

Student Material



- 4 . Your first measure using variables has been ready for presentation.
5. Create one Slicer and one table to test your newly created measure.
6. For Slicer, add Order Date Hierarchy in the Field section.
7. For Table visual, add Order Date Hierarchy and Sales Quantity in the Values section.
8. Display report with some filter condition like 2015 as Year, Quarter as Qtr1, Month as January and observe the report result.

Data Analytics

Student Material

File Home Insert Modeling View Help

Cut Copy Format painter Get data Excel Power BI datasets SQL Server Enter data Dataverse Recent sources Transform R data

Clipboard Data Queries

Year, Quarter, Month

- 2014
- 2015
 - Qtr 1
 - January
 - February
 - March
 - Qtr 2
 - Qtr 3
 - Qtr 4
- 2016
- 2017

Year	Quarter	Month	Day	Sales Quantity
2015	Qtr 1	January	1	No Sales
2015	Qtr 1	January	2	Sold 29 Products
2015	Qtr 1	January	3	Sold 19 Products
2015	Qtr 1	January	4	Sold 22 Products
2015	Qtr 1	January	5	Sold 20 Products
2015	Qtr 1	January	6	Sold 7 Products
2015	Qtr 1	January	7	No Sales
2015	Qtr 1	January	8	No Sales
2015	Qtr 1	January	9	Sold 13 Products
2015	Qtr 1	January	10	Sold 4 Products
2015	Qtr 1	January	11	No Sales
2015	Qtr 1	January	12	Sold 16 Products
2015	Qtr 1	January	13	Sold 18 Products
2015	Qtr 1	January	14	No Sales
2015	Qtr 1	January	15	No Sales
2015	Qtr 1	January	16	No Sales
2015	Qtr 1	January	17	Sold 17 Products
2015	Qtr 1	January	18	No Sales
2015	Qtr 1	January	19	Sold 5 Products
2015	Qtr 1	January	20	No Sales
2015	Qtr 1	January	21	No Sales
2015	Qtr 1	January	22	No Sales
2015	Qtr 1	January	23	Sold 5 Products
2015	Qtr 1	January	24	Sold 5 Products
2015	Qtr 1	January	25	No Sales
2015	Qtr 1	January	26	Sold 9 Products
Total				Sold 236 Products

VARIABLES IN CALCULATED COLUMNS

When you are creating variables inside calculated columns, variables automatically get access to values in other columns from the same row.

It can be useful to display text-based data in some visuals.

1. Create your first calculated variable based column.
2. Please follow the below steps to create that.

The screenshot shows the Power BI Data Editor interface. On the left, a context menu is open over a table named 'US Superstore data'. The 'New column' option is selected. In the center, a dialog box titled 'New column' is displayed with the following code:

```

Name: Country Region
Data type: Text
Format: Text
Summarization: None
Data category: None

Structure:
1 State and Country =
2 VAR state='US Superstore data'[State]
3 VAR cntry='US Superstore data'[Country]
4 RETURN state & ", " & cntry
  
```

On the right, a 'Fields' pane lists various data items, and a 'Search' bar is present. Two arrows point from the text 'Create calculated variable based column.' towards the central dialog box.

Create calculated variable based column.

Click on New column

3. Create a Table visual to see the output.

The screenshot shows a Power BI report. On the left, a 'Table' visual displays data with three columns: 'State', 'Country', and 'State and Country'. The 'State and Country' column contains values like 'Alabama, United States', 'Arizona, United States', etc. On the right, the 'Visualizations' pane is open, showing various chart and report options. The 'Values' section under 'Visualizations' has 'State', 'Country', and 'State and Country' selected. The 'Drill through' section is set to 'Off'.

VARIABLES IN CALCULATED MEASURES

To make a calculated measure faster, you can use variables.

However, variables in calculated measures do not relate with individual rows that mean it cannot be assigned as a column-based value. This is the key difference with variables in calculated columns.

Let us see some example.

1. Create one calculated measure Sales > 500 with variable.
2. Change the number Formatting from the Whole Number to a Decimal Number with 2 decimal places.

The screenshot shows the Microsoft Power BI Data Editor interface. The top navigation bar includes File, Home, Insert, Modeling, View, Help, Table tools, and Measure tools. The Measure tools tab is selected. In the center, there is a code editor window displaying the DAX code for a calculated measure:

```
1 Sales > 500 =  
2 VAR sales_above_500=  
3 | FILTER('US Superstore data','US Superstore data'[Sales]>500)  
4 RETURN CALCULATE(SUM('US Superstore data'[Sales]),sales_above_500)
```

The 'Formatting' section of the ribbon shows a decimal number format with two decimal places. To the right, the 'Fields' pane lists various data fields like Country, Customer ID, etc., with the 'Sales > 500' measure selected. The bottom pane shows a placeholder for building visualizations with the text "Build visuals with your data".

3. Create your report with Customer Name, Order ID, Product ID, Sales and Sales > 500.
4. Now analyse the report. For example, Ed Braxton placed one order for two products. For one product sales value is above 500 and another one is below 500.

Data Analytics

Student Material

The screenshot shows the Microsoft Power BI Data Editor interface. On the left, there's a table view with columns: Customer Name, Order ID, Product ID, Sales, and Sales > 500. The table contains several rows of data. On the right, there are two panes: 'Visualizations' and 'Fields'. The 'Fields' pane is expanded, showing a list of fields with checkboxes next to them. Some fields are checked, such as 'Customer Name', 'Order ID', 'Product ID', 'Sales', and 'Sales > 500'. Other fields like 'Country', 'Postal Code', 'Profit', 'Quantity', 'Region', 'Row ID', 'Sales', 'Sales > 500', 'Segment', 'Ship Date', 'Ship Mode', and 'State' are not checked.

VARIABLES IN CALCULATED TABLES

Variables in the calculated table are useful when you are creating any summary or aggregated table from the existing table. It helps to improve the performance of the report page.

1. Create one calculated table “Summary” using a variable.

The screenshot shows the Microsoft Power BI Data Editor interface. In the 'Structure' tab, there is a code editor window containing DAX code:

```

1 Summary *
2 VAR tableGrouped=
3     SUMMARIZE('US Superstore data',
4             'US Superstore data'[Sub-Category],
5             "Total Sales by Sub Category",SUM('US Superstore data'[Sales])
6     )
7 RETURN tableGrouped
8

```

Below the code editor, there is a message: "Build visuals with your data" and "Select or drag fields from the Fields pane onto the report canvas." To the right of the code editor, there is a 'Visualizations' pane and a 'Fields' pane. The 'Fields' pane is expanded, showing a list of fields with checkboxes next to them. Some fields are checked, such as 'Summary', 'Sub-Category', and 'Total Sales by Sub Category'. Other fields like 'US Superstore data', 'Category', and 'City' are not checked.

2. Now create two reports and compare the results whether your calculated table is working as expected or not.

Report from Summary Table (Calculated Table using variable)

Sub-Category	Total Sales by Sub Category
Phones	330,007.05
Chairs	328,449.10
Storage	223,843.61
Tables	206,965.53
Binders	203,412.73
Machines	189,238.63
Accessories	167,380.32
Copiers	149,528.03
Bookcases	114,880.00
Appliances	107,532.16
Furnishings	91,705.16
Paper	78,479.21
Supplies	46,673.54
Art	27,118.79
Envelopes	16,476.40
Labels	12,486.31
Fasteners	3,024.28
Total	2,297,200.86

Report from Original Table

Sub-Category	Sales
Phones	330,007.05
Chairs	328,449.10
Storage	223,843.61
Tables	206,965.53
Binders	203,412.73
Machines	189,238.63
Accessories	167,380.32
Copiers	149,528.03
Bookcases	114,880.00
Appliances	107,532.16
Furnishings	91,705.16
Paper	78,479.21
Supplies	46,673.54
Art	27,118.79
Envelopes	16,476.40
Labels	12,486.31
Fasteners	3,024.28
Total	2,297,200.86

Assignment:

Connect external data source (excel) with Power BI, remove redundant columns and set appropriate column headers.

Solution:

For Data source connection please Refer to page 35

the Column Headers are actually in the second row, hence, the first row needs to be removed. Click on “Remove Rows” from the Power Query Editor toolbar. Then, click on “Remove Top Rows” and enter “1” in the “Number of rows” dialog box to remove the first row.

Data Analytics

Student Material

The screenshot shows the Power Query Editor interface. A dialog box titled "Remove Top Rows" is open, prompting the user to specify how many rows to remove from the top. The "Number of rows" input field contains the value "1". The "OK" button is highlighted in yellow.

After removing that row, click on the “Use First Row as Headers” option to use the new first row as the Column Headers.

The screenshot shows the Power Query Editor ribbon. The "Transform" tab is selected. The "Use First Row as Headers" checkbox is highlighted with a red box.

Now, the table is displaying the correct Column Headers.

	A _C _ntsb_no	I ² ₃ _aircraft_key	ev_date	A _C _latitude	A _C _longitude	A _C _ev_city	A _C _ev_state	A _C _ev_country	I ² ₃ _inj_tot_f	I ² ₃ _inj_tot_s	A _C _ev_highest_injury
1	WPR13CA079	I	1/1/2013 423428N	1215226W		Chiloquin	OR	USA	null	null	NONE
2	WPR13LA082	I	1/2/2013 361238N	1151140W		North Las Vegas	NV	USA	null	null	NONE
3	WPR13FA083	I	1/2/2013 350358N	1203706W		Oceano	CA	USA	I	null	FATL

Remove Redundant Columns

Here, you can remove all the columns that are not required for your analysis. Select the columns that you wish to remove while pressing the CTRL-key. Then, right-click and click on “Remove Columns” to eliminate columns.

The screenshot shows the Power Query Editor interface. A context menu is open over a column header. The "Remove Columns" option is highlighted with a red box.

Chapter: 4

Data Modelling

Scope:

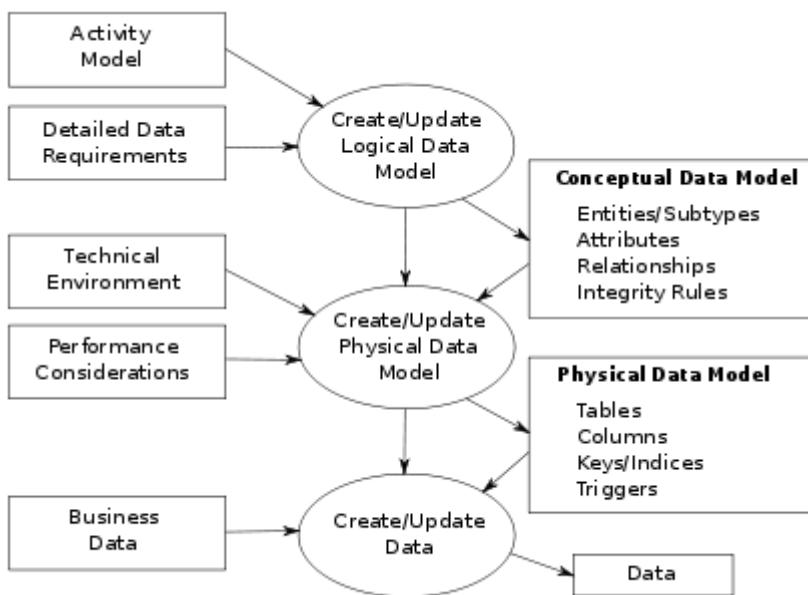
- What is Data Modelling?
- Data Modelling sometimes needs Data Analysis
- Star Schema
- Snowflake Schema
- Create a Relationship
- One to many, many to many relationships

Brought to you by:



4.1 Data Modelling:

Data Modelling is the process of analysing the data objects and their relationship to the other objects. It is used to analyse the data requirements that are required for the business processes. The data models are created for the data to be stored in a database. The Data Model's main focus is on what data is needed and how we have to organize data rather than what operations we have to perform.



The figure illustrates the way data models are developed and used today. A conceptual data model is developed based on the data requirements for the application that is being developed, perhaps in the context of an activity model. The data model will normally consist of entity types, attributes, relationships, integrity rules, and the definitions of those objects. This is then used as the start point for interface or database design.

Data modelling techniques and methodologies are used to model data in a standard, consistent, predictable manner in order to manage it as a resource. The use of data modelling standards is strongly recommended for all projects requiring a standard means of defining and analysing data within an organization, e.g., using data modelling:

- to assist business analysts, programmers, testers, manual writers, IT package selectors, engineers, managers, related organizations and clients to understand and use an agreed upon semi-formal model that encompasses the concepts of the organization and how they relate to one another
- to manage data as a resource
- to integrate information systems
- to design databases/data warehouses (aka data repositories)

4.2 Uses of Data Modelling Tools

Data Modelling is a process to formulate data in an information system in a structured format. Listed below are certain practical uses of the related tools in any sector or industry.

- Data Modelling helps create a robust design with a data model that can show an organization's entire data on the same platform.
- The data model makes sure that all the data objects required by the database are represented or not.
- The database at the logical, physical, and conceptual levels can be designed with the help data model.
- The relation tables, foreign keys, and primary keys can be defined with the data model's help.
- Data Modelling Tools help in the improvement of data quality.
- Data Model gives the clear picture of business requirements.
- Redundant data and missing data can be identified with the help of data models.
- In data models, all the important data is accurately represented. The chances of incorrect results and faulty reports decreased as the data model reduces data omission.
- The data models create a visual representation of the data. With the help of it, the data analysis gets improved. We get the data picture, which can then be used by developers to create a physical database.
- Better consistency can be qualified with the help of a data model across all the projects.
- The data model is quite a time consuming, but it makes the maintenance cheaper and faster.

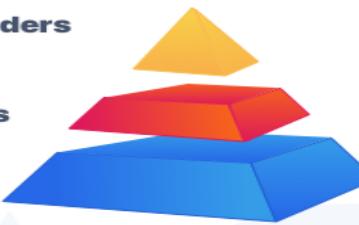
4.3 Three Perspectives of a Data Model

Data Modelling helps to create a conceptual model and create the relationship between the items. The basic data modelling techniques involve dealing with three perspectives of a data model.

3 Perspectives of a Data Model

Audience

- Business Stakeholders
- Data Architecture
Business Analytics
- DBAs Developers



Purpose

- Communication & Definition of Business Terms & Rules
- Clarification & Detail of Business Rules & Data Structures
- Technical Implementation on a Physical Database

1. Conceptual Model

The conceptual data model is a view of the data that is required to help business processes. It also keeps track of business events and keeps related performance measures. The conceptual model defines what the system contains. This type of Data Modelling focuses on finding the data used in a business rather than processing flow. The main purpose of this data model is to organize, define business rules and concepts. For example, it helps business people to view any data like market data, customer data, and purchase data.

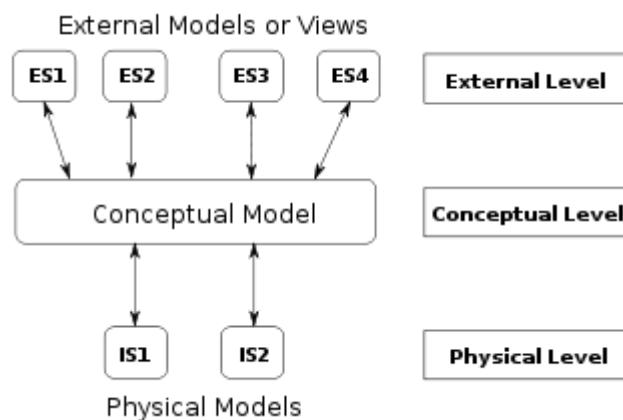
2. Logical Model

In the logical data model, the map of rules and data structures includes the data required, such as tables, columns, etc. Data architects and Business Analysts create the Logical Model. We can use the logical model to transform it into a database. This type of Data Modelling is always present in the root package object. This data model helps to form the base for the physical model. In this model, there is no secondary or primary key is defined.

3. Physical Data Model

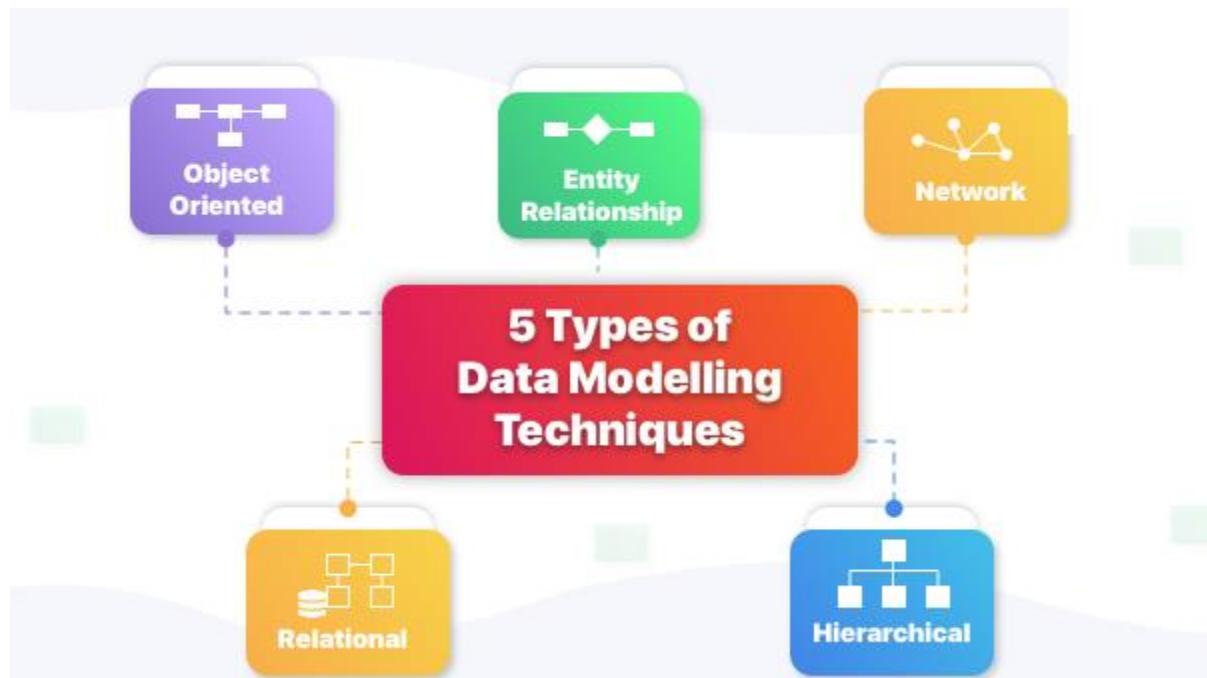
In a physical data model, the implementation is described using a specific database system. It defines all the components and services that are required to build a database. It is created by using the database language and queries. The physical data model represents each table, column, constraints like primary key, foreign key, NOT NULL, etc. The main work of the physical data model is to create a database. This model is created by the Database Administrator (DBA) and developers. This type of Data Modelling gives us the abstraction of the databases and helps to create the schema. This model describes the particular implementation of the data model. The physical data model helps us to have database column keys, constraints, and RDBMS features.

According to ANSI, this approach allows the three perspectives to be relatively independent of each other. Storage technology can change without affecting either the logical or the conceptual schema. The table/column structure can change without (necessarily) affecting the conceptual schema. In each case, of course, the structures must remain consistent across all schemas of the same data model.



The ANSI/SPARC three level architecture. This shows that a data model can be an external model (or view), a conceptual model, or a physical model. This is not the only way to look at data models, but it is a useful way, particularly when comparing models.

4.4 Data Modelling Techniques



Below given are 5 different types of techniques used to organize the data:

1. Hierarchical Technique

The hierarchical model is a tree-like structure. There is one root node, or we can say one parent node and the other child nodes are sorted in a particular order. But, the hierarchical model is very rarely used now. This model can be used for real-world model relationships.

2. Object-oriented Model

The object-oriented approach is the creation of objects that contains stored values. The object-oriented model communicates while supporting data abstraction, inheritance, and encapsulation.

3. Network Technique

The network model provides us with a flexible way of representing objects and relationships between these entities. It has a feature known as a schema representing the data in the form

Data Analytics

Student Material

of a graph. An object is represented inside a node and the relation between them as an edge, enabling them to maintain multiple parent and child records in a generalized manner.

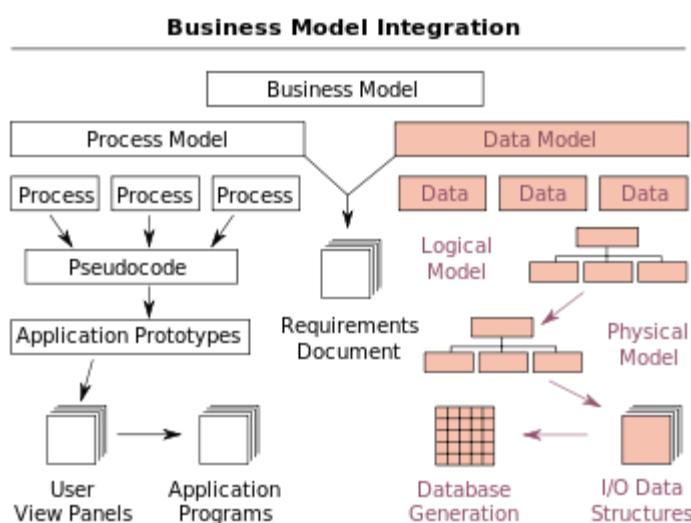
4. Entity-relationship Model

ER model (Entity-relationship model) is a high-level relational model which is used to define data elements and relationship for the entities in a system. This conceptual design provides a better view of the data that helps us easy to understand. In this model, the entire database is represented in a diagram called an entity-relationship diagram, consisting of Entities, Attributes, and Relationships.

5. Relational Technique

Relational is used to describe the different relationships between the entities. And there are different sets of relations between the entities such as one to one, one to many, many to one, and many to many.

4.5 Business process integration



The process of designing a database involves producing the previously described three types of schemas - conceptual, logical, and physical. The database design documented in these schemas are converted through a Data Definition Language, which can then be used to generate a database. A fully attributed data model contains detailed attributes (descriptions) for every entity within it. The term "database design" can describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term "database design" could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the Database Management System or DBMS.

4.6 Modelling methodologies

Data models represent information areas of interest. While there are many ways to create data models, according to Len Silverston (1997) only two Modelling methodologies stand out, top-down and bottom-up:

- Bottom-up models or View Integration models are often the result of a reengineering effort. They usually start with existing data structures forms, fields on application screens, or reports. These models are usually physical, application-specific, and incomplete from an enterprise perspective. They may not promote data sharing, especially if they are built without reference to other parts of the organization.
- Top-down logical data models, on the other hand, are created in an abstract way by getting information from people who know the subject area. A system may not implement all the entities in a logical model, but the model serves as a reference point or template.

4.7 Entity–relationship Model

There are several notations for data Modelling. The actual model is frequently called "entity–relationship model", because it depicts data in terms of the entities and relationships described in the data. An entity–relationship model (ERM) is an abstract conceptual representation of structured data. Entity–relationship Modelling is a relational schema database Modelling method, used in software engineering to produce a type of conceptual data model (or semantic data model) of a system, often a relational database, and its requirements in a top-down fashion.

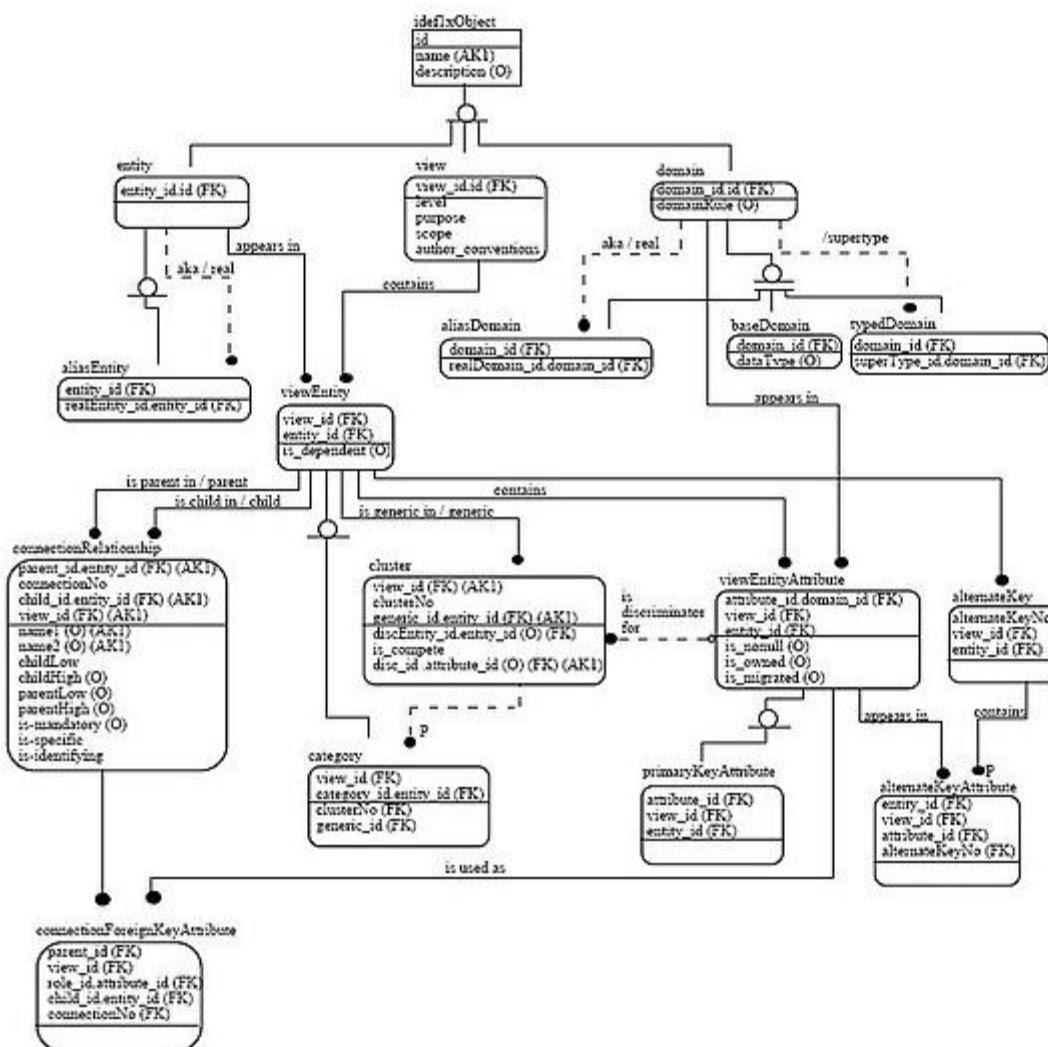
These models are being used in the first stage of information system design during the requirements analysis to describe information needs or the type of information that is to be stored in a database. The data Modelling technique can be used to describe any ontology (i.e. an overview and classifications of used terms and their relationships) for a certain universe of discourse i.e. area of interest.

Several techniques have been developed for the design of data models. While these methodologies guide data modelers in their work, two different people using the same methodology will often come up with very different results. Most notable are:

- Bachman diagrams
- Barker's notation
- Chen's notation
- Data Vault Modelling
- Extended Backus–Naur form
- IDEF1X
- Object-relational mapping
- Object-Role Modelling and Fully Communication Oriented Information Modelling
- Relational Model
- Relational Model/Tasmania

Data Analytics

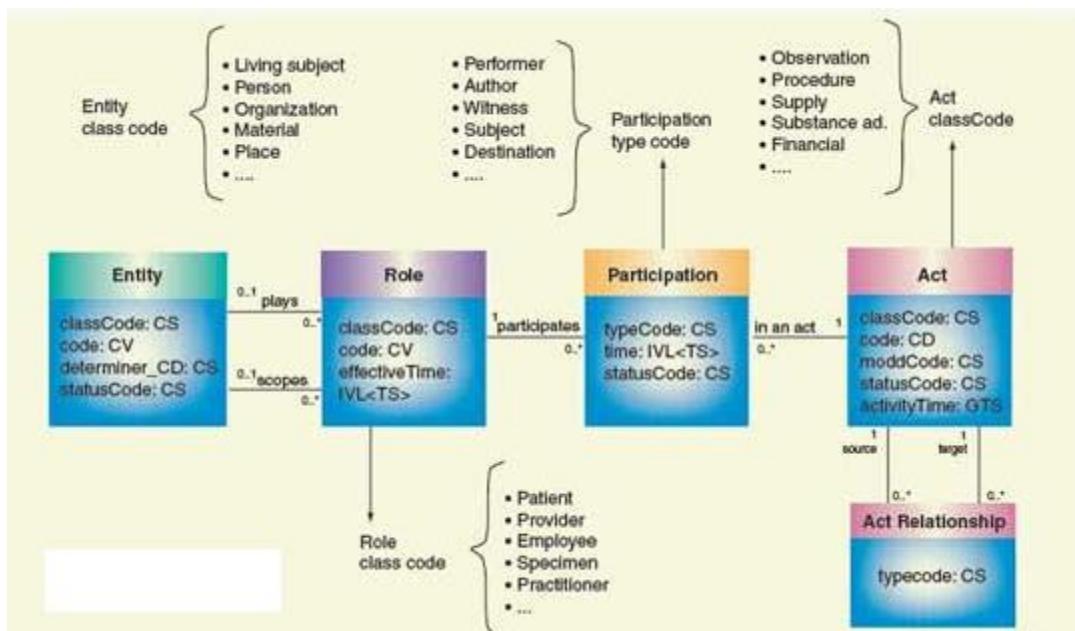
Student Material



Example of an IDEF1X entity–relationship diagrams

4.8 Generic data Modelling

Generic data models are generalizations of conventional data models. They define standardized general relation types, together with the kinds of things that may be related by such a relation type. The definition of generic data model is similar to the definition of a natural language. For example, a generic data model may define relation types such as a 'classification relation', being a binary relation between an individual thing and a kind of thing (a class) and a 'part-whole relation', being a binary relation between two things, one with the role of part, the other with the role of whole, regardless the kind of things that are related.



Given an extensible list of classes, this allows the classification of any individual thing and to specify part-whole relations for any individual object. By standardization of an extensible list of relation types, a generic data model enables the expression of an unlimited number of kinds of facts and will approach the capabilities of natural languages. Conventional data models, on the other hand, have a fixed and limited domain scope, because the instantiation (usage) of such a model only allows expressions of kinds of facts that are predefined in the model.

4.9 Semantic data Modelling

The logical data structure of a DBMS, whether hierarchical, network, or relational, cannot totally satisfy the requirements for a conceptual definition of data because it is limited in scope and biased toward the implementation strategy employed by the DBMS. That is unless the semantic data model is implemented in the database on purpose, a choice which may slightly impact performance but generally vastly improves productivity.



Semantic data models.

Therefore, the need to define data from a conceptual view has led to the development of semantic data Modelling techniques. That is, techniques to define the meaning of data within the context of its interrelationships with other data. As illustrated in the figure the real world, in terms of resources, ideas, events, etc., are symbolically defined within physical data stores. A semantic data model is an abstraction which defines how the stored symbols relate to the real world. Thus, the model must be a true representation of the real world.

A semantic data model can be used to serve many purposes, such as:

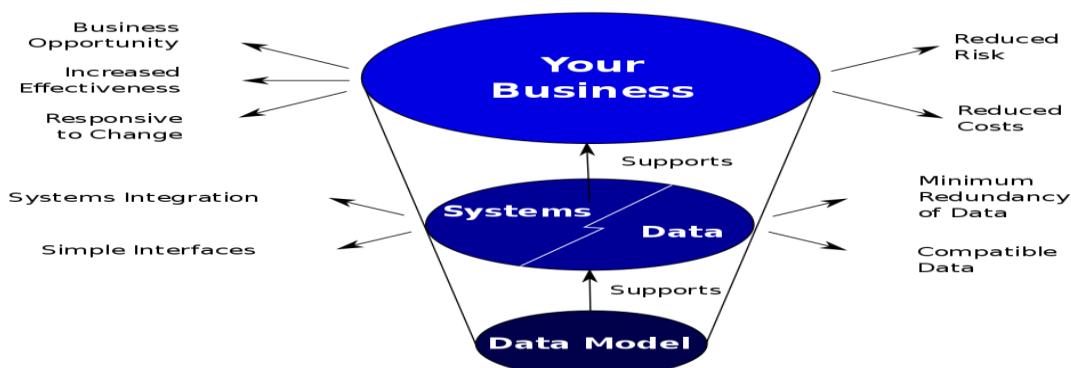
- planning of data resources
- building of shareable databases
- evaluation of vendor software
- integration of existing databases

4.10 What does a Data Modeller do?

- They create an entity relationship diagram to visualise relationships between key business concepts.
- They create a conceptual-level data dictionary to communicate data requirements that are important to business stakeholders.
- They create a data map to resolve potential data issues for a data migration or integration project.
- A data modeller would not necessarily query or manipulate data or become involved in designing or implementing databases or data repositories.

4.11 How data models deliver benefit

Data models provide a framework for data to be used within information systems by providing specific definition and format. If a data model is used consistently across systems then compatibility of data can be achieved. If the same data structures are used to store and access data then different applications can share data seamlessly. The results of this are indicated in the diagram. However, systems and interfaces are often expensive to build, operate, and maintain. They may also constrain the business rather than support it. This may occur when the quality of the data models implemented in systems and interfaces is poor.



4.12 Some common problems found in data models are:

- Business rules, specific to how things are done in a particular place, are often fixed in the structure of a data model. This means that small changes in the way business is conducted lead to large changes in computer systems and interfaces. So, business rules need to be implemented in a flexible way that does not result in complicated dependencies, rather the data model should be flexible enough so that changes in the business can be implemented within the data model in a relatively quick and efficient way.
- Entity types are often not identified, or are identified incorrectly. This can lead to replication of data, data structure and functionality, together with the attendant costs of that duplication in development and maintenance. Therefore, data definitions should be made as explicit and easy to understand as possible to minimize misinterpretation and duplication.
- Data models for different systems are arbitrarily different. The result of this is that complex interfaces are required between systems that share data. These interfaces can account for between 25-70% of the cost of current systems. Required interfaces should be considered inherently while designing a data model, as a data model on its own would not be usable without interfaces within different systems.
- Data cannot be shared electronically with customers and suppliers, because the structure and meaning of data has not been standardised. To obtain optimal value from an implemented data model, it is very important to define standards that will ensure that data models will both meet business needs and be consistent.

4.13 Data Modelling sometimes needs Data Analysis

BA's often need to analyse data as part of making data Modelling decisions, and this means that data Modelling can include some amount of data analysis. A lot can be accomplished with very basic technical skills, such as the ability to run simple database queries. This is why you may see a technical skill like SQL in a business analyst job description.

Many BA's succeed without knowing these more technical skills, instead, they rely on their ability to collaborate with technical professionals and other knowledgeable stakeholders to ensure the data is understood well enough to make the right modelling decisions.

The non-technical BA can also evaluate sample data, interview stakeholders to discover possible data-related issues, review current state database models, and analyse exception reports.

While data analysis skills are valuable for the business analyst, they are not essential. However, data modelling falls squarely within the business analyst's domain.

4.14 Star schema overview

Star schema is a mature modeling approach widely adopted by relational data warehouses. It requires modelers to classify their model tables as either *dimension* or *fact*.

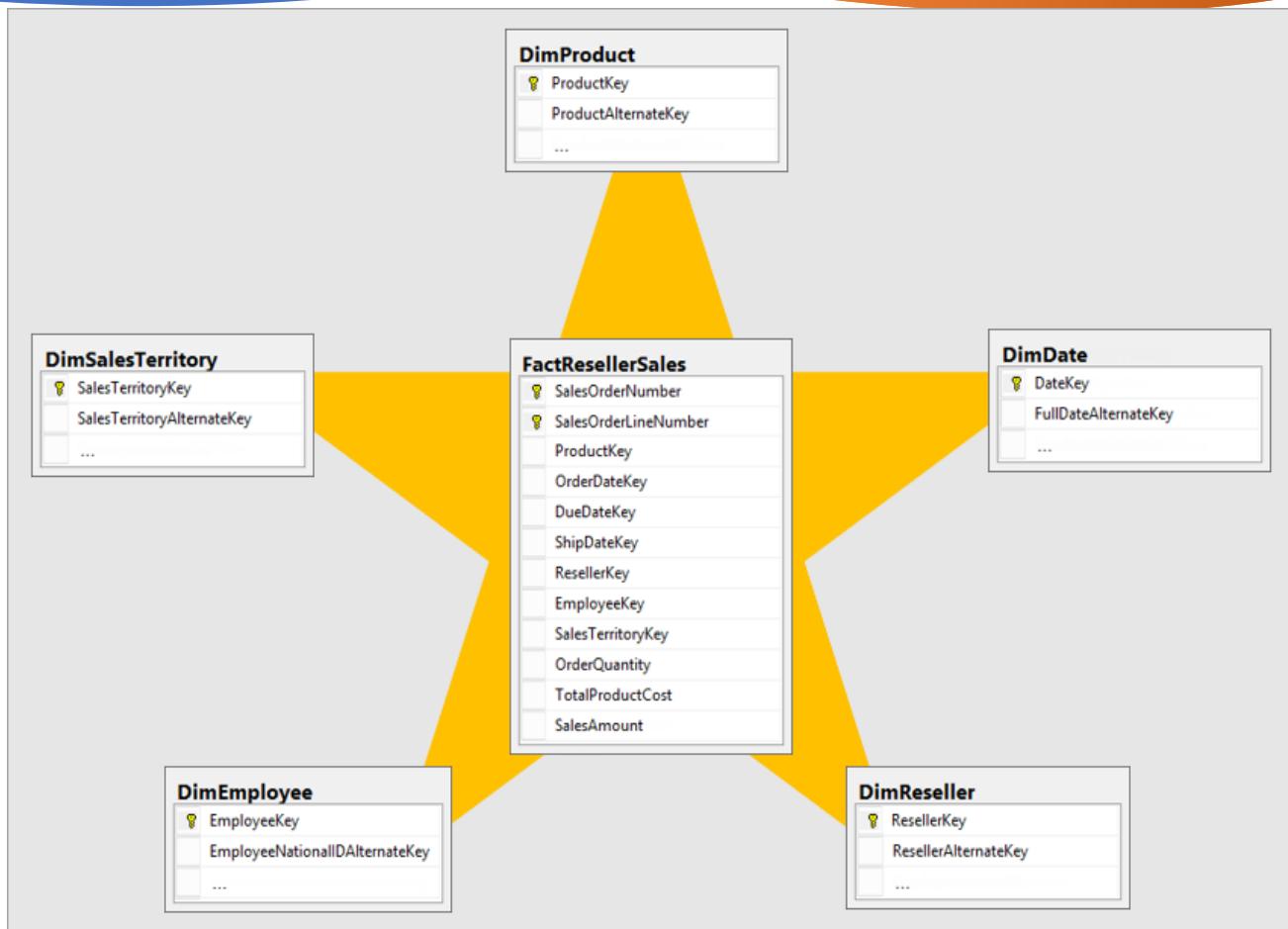
Dimension tables describe business entities—the *things* you model. Entities can include products, people, places, and concepts including time itself. The most consistent table you'll find in a star schema is a date dimension table. A dimension table contains a key column (or columns) that acts as a unique identifier, and descriptive columns.

Fact tables store observations or events, and can be sales orders, stock balances, exchange rates, temperatures, etc. A fact table contains dimension key columns that relate to dimension tables, and numeric measure columns. The dimension key columns determine the *dimensionality* of a fact table, while the dimension key values determine the *granularity* of a fact table. For example, consider a fact table designed to store sale targets that has two-dimension key columns **Date** and **ProductKey**. It's easy to understand that the table has two dimensions. The granularity, however, can't be determined without considering the dimension key values. In this example, consider that the values stored in the **Date** column are the first day of each month. In this case, the granularity is at month-product level.

Generally, dimension tables contain a relatively small number of rows. Fact tables, on the other hand, can contain a very large number of rows and continue to grow over time.

Data Analytics

Student Material



Normalization vs. denormalization

To understand some star schema concepts described in this article, it's important to know two terms: normalization and denormalization.

Normalization is the term used to describe data that's stored in a way that reduces repetitious data. Consider a table of products that has a unique key value column, like the product key, and additional columns describing product characteristics, including product name, category, color, and size. A sales table is considered normalized when it stores only keys, like the product key. In the following image, notice that only the **ProductKey** column records the product.

	SalesOrderNumber	OrderDate	ProductKey	ResellerKey	SalesAmount
1	SO69561	2020-05-31	594	546	226.00
2	SO69560	2020-05-30	513	100	218.45
3	SO69560	2020-05-30	594	100	113.00
4	SO69539	2020-05-28	243	529	858.90
5	SO69539	2020-05-28	378	529	1466.01
6	SO69541	2020-05-28	594	661	113.00
7	SO69542	2020-05-28	243	317	1717.80
8	SO69544	2020-05-28	243	666	3435.60
9	SO69545	2020-05-28	378	436	5864.04
10	SO69532	2020-05-27	594	312	113.00
11	SO69532	2020-05-27	513	312	436.90
12	SO69533	2020-05-27	594	476	226.00

Data Analytics

Student Material

If, however, the sales table stores product details beyond the key, it's considered *denormalized*. In the following image, notice that the **ProductKey** and other product-related columns record the product.

	SalesOrderNumber	OrderDate	ProductKey	Product	Category	Color	Size	ResellerKey	SalesAmount
1	SO69561	2020-05-31	594	Mountain-500 Silver, 48	Bikes	Silver	48	546	226.00
2	SO69560	2020-05-30	513	ML Mountain Frame-W - Silver, 46	Components	Silver	46	100	218.45
3	SO69560	2020-05-30	594	Mountain-500 Silver, 48	Bikes	Silver	48	100	113.00
4	SO69539	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	529	858.90
5	SO69539	2020-05-28	378	Road-250 Black, 52	Bikes	Black	52	529	1466.01
6	SO69541	2020-05-28	594	Mountain-500 Silver, 48	Bikes	Silver	48	661	113.00
7	SO69542	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	317	1717.80
8	SO69544	2020-05-28	243	HL Road Frame - Red, 44	Components	Red	44	666	3435.60
9	SO69545	2020-05-28	378	Road-250 Black, 52	Bikes	Black	52	436	5864.04
10	SO69532	2020-05-27	594	Mountain-500 Silver, 48	Bikes	Silver	48	312	113.00
11	SO69532	2020-05-27	513	ML Mountain Frame-W - Silver, 46	Components	Silver	46	312	436.90
12	SO69533	2020-05-27	594	Mountain-500 Silver, 48	Bikes	Silver	48	476	226.00

When you source data from an export file or data extract, it's likely that it represents a denormalized set of data. In this case, use Power Query to transform and shape the source data into multiple normalized tables.

As described in this article, you should strive to develop optimized Power BI data models with tables that represent normalized fact and dimension data. However, there's one exception where a snowflake dimension should be denormalized to produce a single model table.

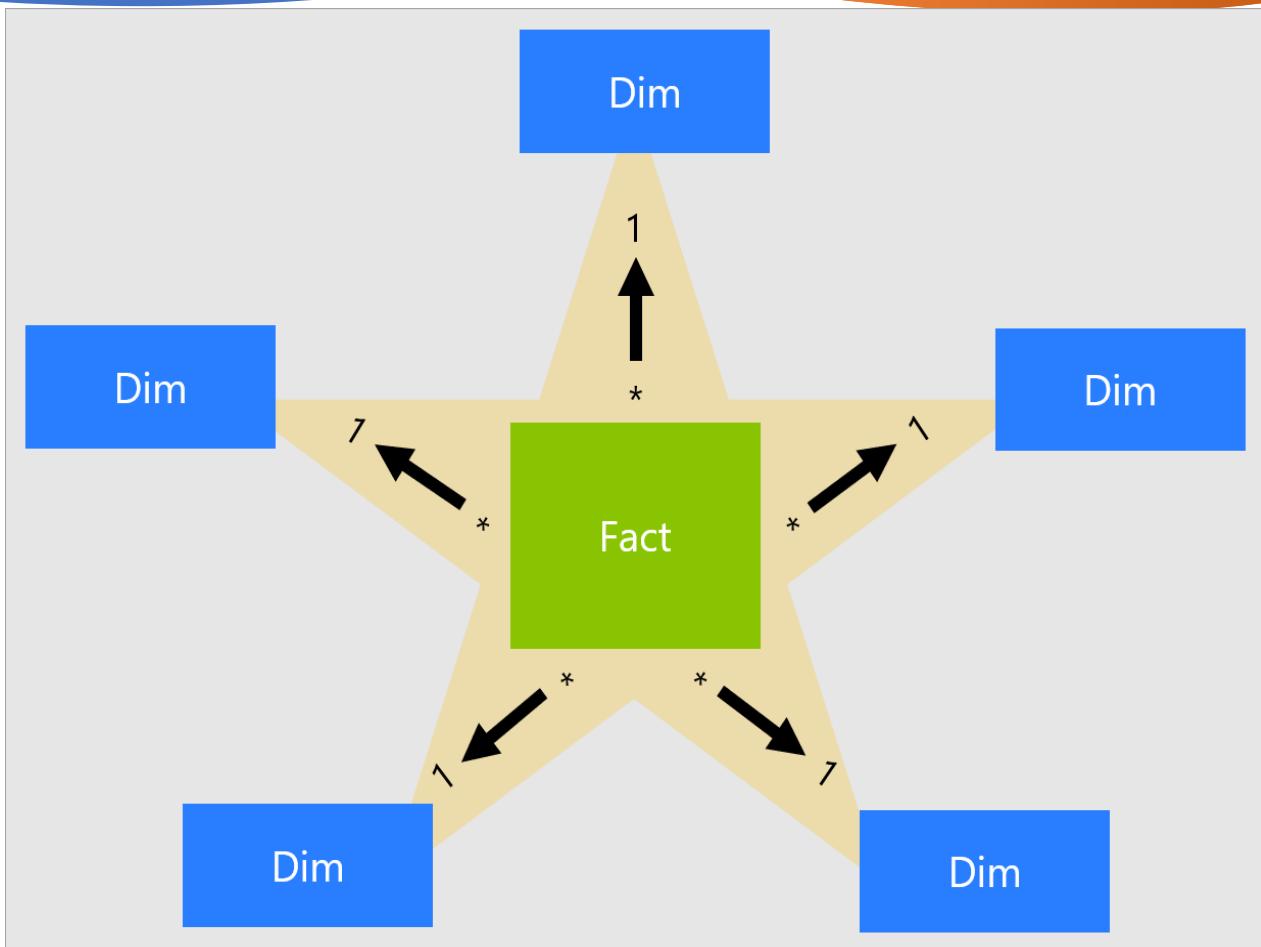
Star schema relevance to Power BI models

Star schema design and many related concepts introduced in this article are highly relevant to developing Power BI models that are optimized for performance and usability.

Consider that each Power BI report visual generates a query that is sent to the Power BI model (which the Power BI service calls a dataset). These queries are used to filter, group, and summarize model data. A well-designed model, then, is one that provides tables for filtering and grouping, and tables for summarizing. This design fits well with star schema principles:

- Dimension tables support *filtering* and *grouping*
- Fact tables support *summarization*

There's no table property that modelers set to configure the table type as dimension or fact. It's in fact determined by the model relationships. A model relationship establishes a filter propagation path between two tables, and it's the **Cardinality** property of the relationship that determines the table type. A common relationship cardinality is *one-to-many* or its inverse *many-to-one*. The "one" side is always a dimension-type table while the "many" side is always a fact-type table.



A well-structured model design should include tables that are either dimension-type tables or fact-type tables. Avoid mixing the two types together for a single table. We also recommend that you should strive to deliver the right number of tables with the right relationships in place. It's also important that fact-type tables always load data at a consistent grain.

Lastly, it's important to understand that optimal model design is part science and part art. Sometimes you can break with good guidance when it makes sense to do so.

There are many additional concepts related to star schema design that can be applied to a Power BI model. These concepts include:

- Measures
- Surrogate keys
- Snowflake dimensions
- Role-playing dimensions
- Slowly changing dimensions
- Junk dimensions
- Degenerate dimensions
- Factless fact tables

Measures

In star schema design, a **measure** is a fact table column that stores values to be summarized.

In a Power BI model, a **measure** has a different—but similar—definition. It's a formula written in Data Analysis Expressions (DAX) that achieves summarization. Measure expressions often leverage DAX aggregation functions like SUM, MIN, MAX, AVERAGE, etc. to produce a scalar value result at query time (values are never stored in the model). Measure expression can range from simple column aggregations to more sophisticated formulas that override filter context and/or relationship propagation. For more information, read the [DAX Basics in Power BI Desktop article](#).

It's important to understand that Power BI models support a second method for achieving summarization. Any column—and typically numeric columns—can be summarized by a report visual or Q&A. These columns are referred to as *implicit measures*. They offer a convenience for you as a model developer, as in many instances you do not need to create measures. For example, the Adventure Works reseller sales **Sales Amount** column could be summarized in numerous ways (sum, count, average, median, min, max, etc.), without the need to create a measure for each possible aggregation type.

The screenshot shows the 'Fields' pane in Power BI. Under the 'Reseller Sales' table, there are five columns listed with their respective DAX aggregation formulas:

- \sum Order Quantity
- \sum Sales (highlighted with a red arrow pointing to the text 'Measure')
- \sum Sales Amount (highlighted with a red arrow pointing to the text 'Summarizable column')
- \sum Total Product Cost
- \sum Unit Price

However, there are three compelling reasons for you to create measures, even for simple column-level summarizations:

- When you know your report authors will query the model by using **Multidimensional Expressions (MDX)**, the model must include *explicit measures*. **Explicit measures are defined by using DAX**. This design approach is highly relevant when a Power BI dataset is queried by using MDX, because MDX can't achieve summarization of column values. Notably, MDX will be used when performing Analyse in Excel, because PivotTables issue MDX queries.
Note: *MDX can be used to query against SSAS Multi-dimensional models, while DAX is used for SSAS Tabular models. If you use Power BI for your visualization needs, it can connect to SSAS Multi-dimensional models even though Power BI primarily uses Tabular models.*

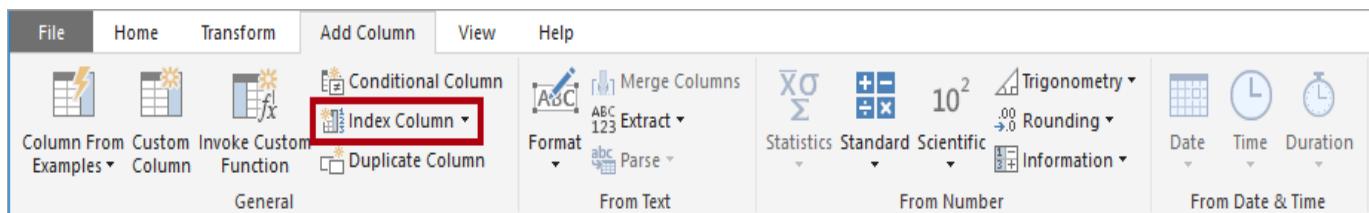
- When you know your report authors will create Power BI paginated reports using the MDX query designer, the model must include explicit measures. Only the MDX query designer supports server aggregates. So, if report authors need to have measures evaluated by Power BI (instead of by the paginated report engine), they must use the MDX query designer.
- When you need to ensure that your report authors can only summarize columns in specific ways. For example, the reseller sales **Unit Price** column (which represents a per unit rate) can be summarized, but only by using specific aggregation functions. It should never be summed, but it's appropriate to summarize by using other aggregation functions like min, max, average, etc. In this instance, the modeler can hide the **Unit Price** column, and create measures for all appropriate aggregation functions.

This design approach works well for reports authored in the Power BI service and for Q&A. However, Power BI Desktop live connections allow report authors to show hidden fields in the **Fields** pane, which can result in circumventing this design approach.

Surrogate keys

A **surrogate key** is a unique identifier that you add to a table to support star schema modeling. By definition, it's not defined or stored in the source data. Commonly, surrogate keys are added to relational data warehouse dimension tables to provide a unique identifier for each dimension table row.

Power BI model relationships are based on a single unique column in one table, which propagates filters to a single column in a different table. When a dimension-type table in your model doesn't include a single unique column, you must add a unique identifier to become the "one" side of a relationship. In Power BI Desktop, you can easily achieve this requirement by creating a Power Query index column.



You must merge this query with the "many"-side query so that you can add the index column to it also. When you load these queries to the model, you can then create a one-to-many relationship between the model tables.

Snowflake dimensions

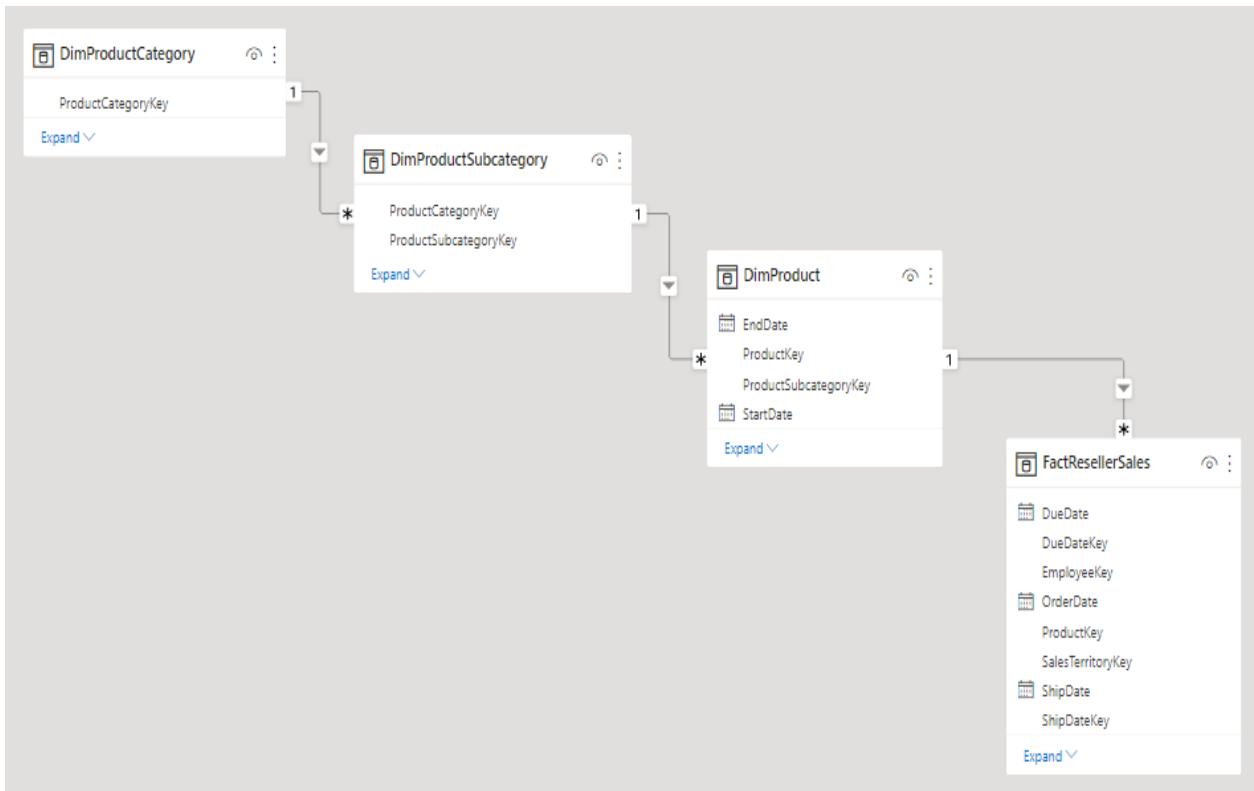
A **snowflake dimension** is a set of normalized tables for a single business entity. For example, Adventure Works classifies products by category and subcategory. Products are assigned to subcategories, and subcategories are in turn assigned to categories. In the Adventure Works relational data warehouse, the product dimension is normalized and

Data Analytics

Student Material

stored in three related tables: **DimProductCategory**, **DimProductSubcategory**, and **DimProduct**.

If you use your imagination, you can picture the normalized tables positioned outwards from the fact table, forming a snowflake design.



In Power BI Desktop, you can choose to mimic a snowflake dimension design (perhaps because your source data does) or integrate (denormalize) the source tables into a single model table. Generally, the benefits of a single model table outweigh the benefits of multiple model tables. The most optimal decision can depend on the volumes of data and the usability requirements for the model.

When you choose to mimic a snowflake dimension design:

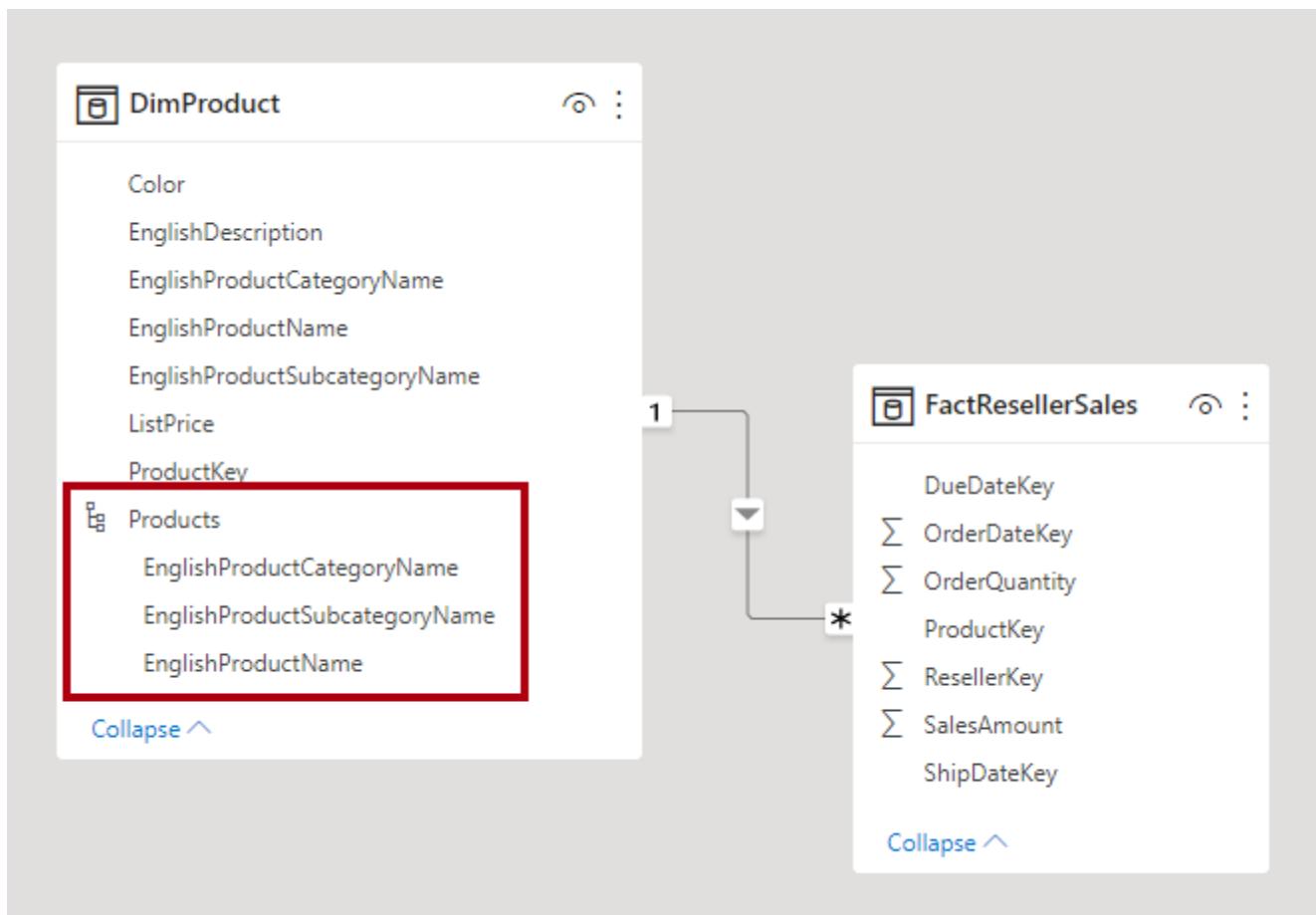
- Power BI loads more tables, which is less efficient from storage and performance perspectives. These tables must include columns to support model relationships, and it can result in a larger model size.
- Longer relationship filter propagation chains will need to be traversed, which will likely be less efficient than filters applied to a single table.
- The **Fields** pane presents more model tables to report authors, which can result in a less intuitive experience, especially when snowflake dimension tables contain just one or two columns.
- It's not possible to create a hierarchy that spans the tables.

When you choose to integrate into a single model table, you can also define a hierarchy that encompasses the highest and lowest grain of the dimension. Possibly, the storage of

Data Analytics

Student Material

redundant denormalized data can result in increased model storage size, particularly for very large dimension tables.



Slowly changing dimensions

A **slowly changing dimension** (SCD) is one that appropriately manages change of dimension members over time. It applies when business entity values change over time, and in an ad hoc manner. A good example of a *slowly* changing dimension is a customer dimension, specifically its contact detail columns like email address and phone number. In contrast, some dimensions are considered to be *rapidly* changing when a dimension attribute changes often, like a stock's market price. The common design approach in these instances is to store rapidly changing attribute values in a fact table measure.

Star schema design theory refers to two common SCD types: Type 1 and Type 2. A dimension-type table could be Type 1 or Type 2, or support both types simultaneously for different columns.

Type 1 SCD

A **Type 1 SCD** always reflects the latest values, and when changes in source data are detected, the dimension table data is overwritten. This design approach is common for columns that store supplementary values, like the email address or phone number of a customer. When a customer email address or phone number changes, the dimension table

updates the customer row with the new values. It's as if the customer always had this contact information.

A non-incremental refresh of a Power BI model dimension-type table achieves the result of a Type 1 SCD. It refreshes the table data to ensure the latest values are loaded.

Type 2 SCD

A **Type 2 SCD** supports versioning of dimension members. If the source system doesn't store versions, then it's usually the data warehouse load process that detects changes, and appropriately manages the change in a dimension table. In this case, the dimension table must use a surrogate key to provide a unique reference to a *version* of the dimension member. It also includes columns that define the date range validity of the version (for example, **StartDate** and **EndDate**) and possibly a flag column (for example, **IsCurrent**) to easily filter by current dimension members.

For example, Adventure Works assigns salespeople to a sales region. When a salesperson relocates region, a new version of the salesperson must be created to ensure that historical facts remain associated with the former region. To support accurate historic analysis of sales by salesperson, the dimension table must store versions of salespeople and their associated region(s). The table should also include start and end date values to define the time validity. Current versions may define an empty end date (or 12/31/9999), which indicates that the row is the current version. The table must also define a surrogate key because the business key (in this instance, employee ID) won't be unique.

It's important to understand that when the source data doesn't store versions, you must use an intermediate system (like a data warehouse) to detect and store changes. The table load process must preserve existing data and detect changes. When a change is detected, the table load process must expire the current version. It records these changes by updating the **EndDate** value and inserting a new version with the **StartDate** value commencing from the previous **EndDate** value. Also, related facts must use a time-based lookup to retrieve the dimension key value relevant to the fact date. A Power BI model using Power Query can't produce this result. It can, however, load data from a pre-loaded SCD Type 2 dimension table.

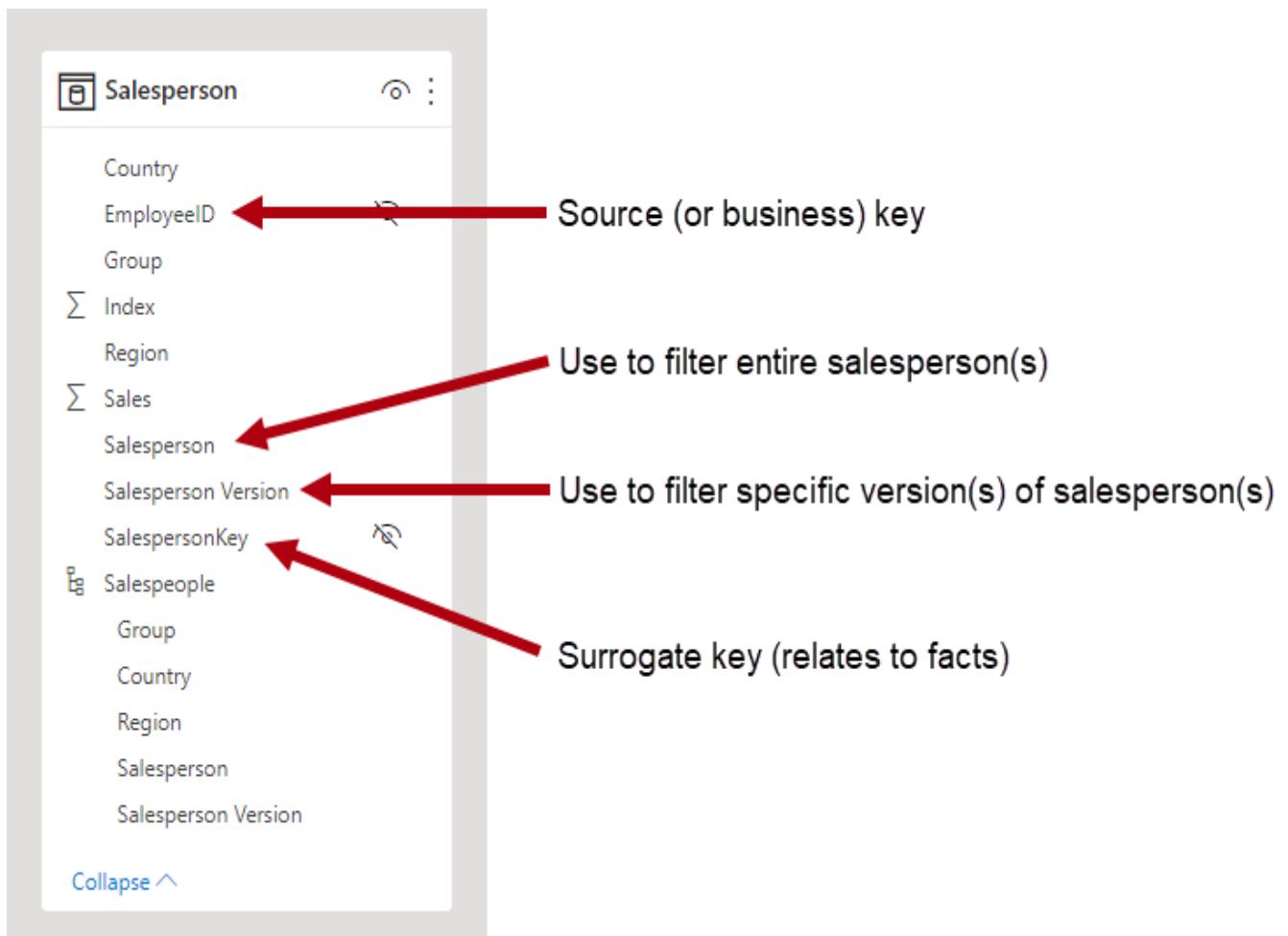
The Power BI model should support querying historical data for a member, regardless of change, and for a version of the member, which represents a particular state of the member in time. In the context of Adventure Works, this design enables you to query the salesperson regardless of assigned sales region, or for a particular version of the salesperson.

To achieve this requirement, the Power BI model dimension-type table must include a column for filtering the salesperson, and a different column for filtering a specific version of the salesperson. It's important that the version column provides a non-ambiguous description, like "Michael Blythe (12/15/2008-06/26/2019)" or "Michael Blythe (current)". It's also important to educate report authors and consumers about the basics of SCD Type 2, and how to achieve appropriate report designs by applying correct filters.

Data Analytics

Student Material

It's also a good design practice to include a hierarchy that allows visuals to drill down to the version level.



Group	Country	Region	Salesperson	Salesperson Version	Sales
North America	United States	Northeast	David Campbell	David Campbell (12/15/2008-06/26/2019)	\$52,307.11
				David Campbell (06/27/2019-Current)	\$119,807.56
				Total	\$172,114.67
			Pamela Ansman-Wolfe	Pamela Ansman-Wolfe (03/03/2007-Current)	\$222,087.01
				Total	\$222,087.01
			Tete Mensa-Annan	Tete Mensa-Annan (01/09/2005-Current)	\$23,098.4
				Total	\$23,098.4
				Total	\$417,300.08
			Total		\$417,300.08
			Total		\$417,300.08

Role-playing dimensions

A **role-playing dimension** is a dimension that can filter related facts differently. For example, at Adventure Works, the date dimension table has three relationships to the reseller sales facts. The same dimension table can be used to filter the facts by order date, ship date, or delivery date.

Brought to you by:

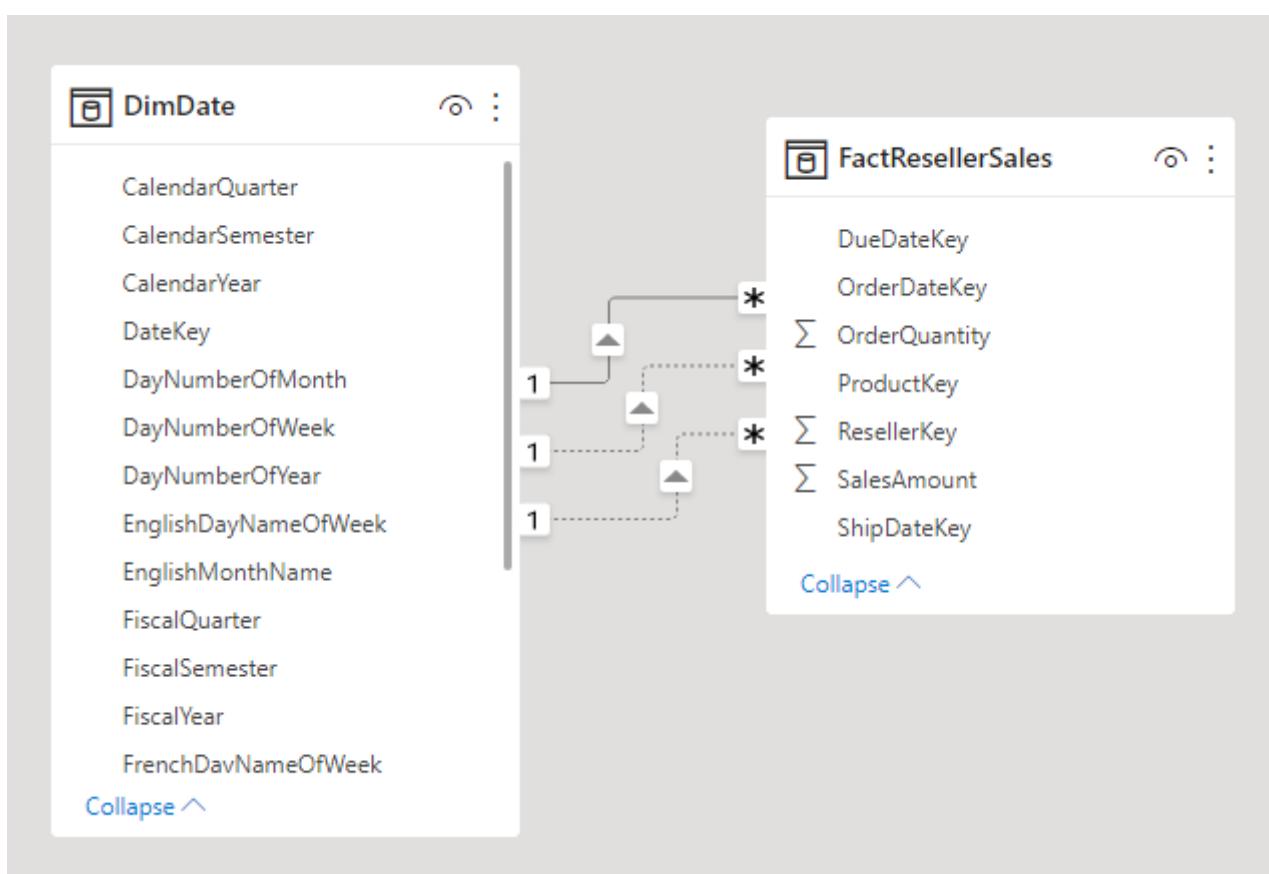


Data Analytics

Student Material

In a data warehouse, the accepted design approach is to define a single date dimension table. At query time, the "role" of the date dimension is established by which fact column you use to join the tables. For example, when you analyze sales by order date, the table join relates to the reseller sales order date column.

In a Power BI model, this design can be imitated by creating multiple relationships between two tables. In the Adventure Works example, the date and reseller sales tables would have three relationships. While this design is possible, it's important to understand that there can only be one active relationship between two Power BI model tables. All remaining relationships must be set to inactive. Having a single active relationship means there is a default filter propagation from date to reseller sales. In this instance, the active relationship is set to the most common filter that is used by reports, which at Adventure Works is the order date relationship.

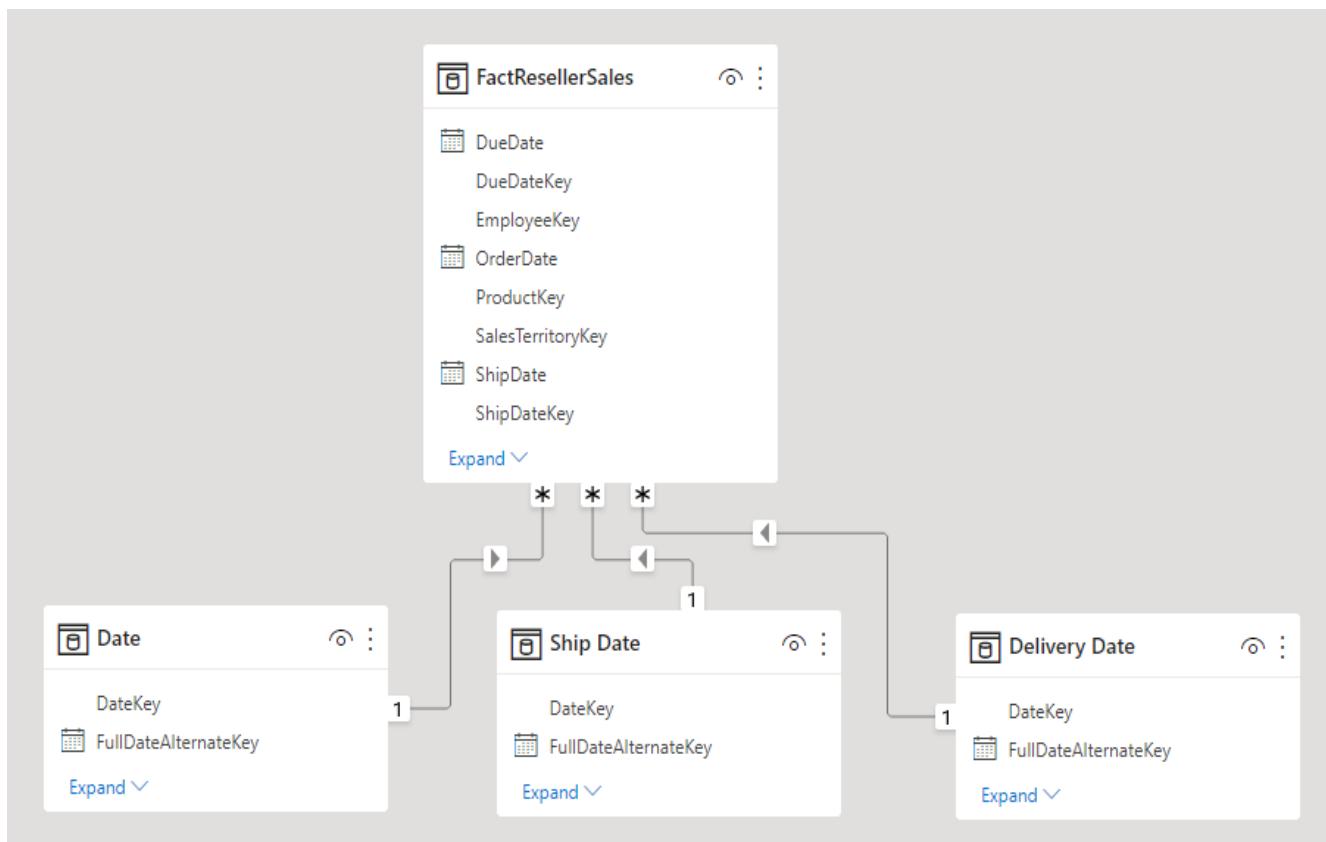


The only way to use an inactive relationship is to define a DAX expression that uses the USERELATIONSHIP function. In our example, the model developer must create measures to enable analysis of reseller sales by ship date and delivery date. This work can be tedious, especially when the reseller table defines many measures. It also creates **Fields** pane clutter, with an overabundance of measures. There are other limitations, too:

- When report authors rely on summarizing columns, rather than defining measures, they can't achieve summarization for the inactive relationships without writing a report-level measure. Report-level measures can only be defined when authoring reports in Power BI Desktop.

- With only one active relationship path between date and reseller sales, it's not possible to simultaneously filter reseller sales by different types of dates. For example, you can't produce a visual that plots order date sales by shipped sales.

To overcome these limitations, a common Power BI modeling technique is to create a dimension-type table for each role-playing instance. You typically create the additional dimension tables as calculated tables, using DAX. Using calculated tables, the model can contain a **Date** table, a **Ship Date** table and a **Delivery Date** table, each with a single and active relationship to their respective reseller sales table columns.



This design approach doesn't require you to define multiple measures for different date roles, and it allows simultaneous filtering by different date roles. A minor price to pay, however, with this design approach is that there will be duplication of the date dimension table resulting in an increased model storage size. As dimension-type tables typically store fewer rows relative to fact-type tables, it is rarely a concern.

Observe the following good design practices when you create model dimension-type tables for each role:

- Ensure that the column names are self-describing. While it's possible to have a **Year** column in all date tables (column names are unique within their table), it's not self-describing by default visual titles. Consider renaming columns in each dimension role table, so that the **Ship Date** table has a year column named **Ship Year**, etc.
- When relevant, ensure that table descriptions provide feedback to report authors (through **Fields** pane tooltips) about how filter propagation is configured. This clarity is

Data Analytics

Student Material

important when the model contains a generically named table, like **Date**, which is used to filter many fact-type tables. In the case that this table has, for example, an active relationship to the reseller sales order date column, consider providing a table description like "Filters reseller sales by order date".

Junk dimensions

A **junk dimension** is useful when there are many dimensions, especially consisting of few attributes (perhaps one), and when these attributes have few values. Good candidates include order status columns, or customer demographic columns (gender, age group, etc.).

The design objective of a junk dimension is to consolidate many "small" dimensions into a single dimension to both reduce the model storage size and also reduce **Fields** pane clutter by surfacing fewer model tables.

A junk dimension table is typically the Cartesian product of all dimension attribute members, with a surrogate key column. The surrogate key provides a unique reference to each row in the table. You can build the dimension in a data warehouse, or by using Power Query to create a query that performs full outer query joins, then adds a surrogate key (index column).



Order Status	ResellerSalesJunkKey	Order Status	Delivery Status
Quote	1	1 Quote	Not Delivered
Ordered	2	2 Quote	Delivered
Cancelled	3	3 Ordered	Not Delivered
	4	4 Ordered	Delivered
Delivery Status	5	5 Cancelled	Not Delivered
Not Delivered	6	6 Cancelled	Delivered
Delivered			

You load this query to the model as a dimension-type table. You also need to merge this query with the fact query, so the index column is loaded to the model to support the creation of a "one-to-many" model relationship.

Degenerate dimensions

A **degenerate dimension** refers to an attribute of the fact table that is required for filtering. At Adventure Works, the reseller sales order number is a good example. In this case, it doesn't make good model design sense to create an independent table consisting of just this one column, because it would increase the model storage size and result in **Fields** pane clutter.

In the Power BI model, it can be appropriate to add the sales order number column to the fact-type table to allow filtering or grouping by sales order number. It is an exception to the formerly introduced rule that you should not mix table types (generally, model tables should be either dimension-type or fact-type).

The screenshot shows the 'Fields' pane in Power BI. At the top is a search bar. Below it, under the 'Reseller Sales' table, the 'Order Number' field is selected, indicated by a red arrow pointing to its icon. Other fields listed include Order Quantity, Sales, Sales Amount, Total Product Cost, and Unit Price.

However, if the Adventure Works resellers sales table has order number *and* order line number columns, and they're required for filtering, a degenerate dimension table would be a good design. For more information, see One-to-one relationship guidance (Degenerate dimensions).

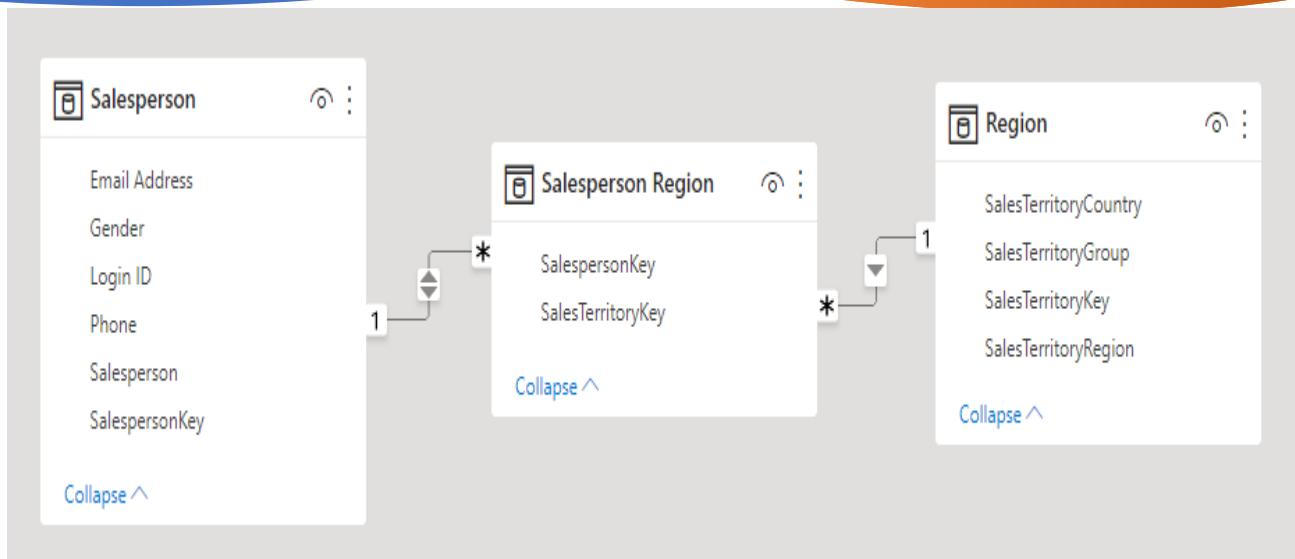
Factless fact tables

A **factless fact** table doesn't include any measure columns. It contains only dimension keys.

A factless fact table could store observations defined by dimension keys. For example, at a particular date and time, a particular customer logged into your web site. You could define a measure to count the rows of the factless fact table to perform analysis of when and how many customers have logged in.

A more compelling use of a factless fact table is to store relationships between dimensions, and it's the Power BI model design approach we recommend defining many-to-many dimension relationships. In a many-to-many dimension relationship design, the factless fact table is referred to as a *bridging table*.

For example, consider that salespeople can be assigned to one *or more* sales regions. The bridging table would be designed as a factless fact table consisting of two columns: salesperson key and region key. Duplicate values can be stored in both columns.



This many-to-many design approach is well documented, and it can be achieved without a bridging table. However, the bridging table approach is considered the best practice when relating two dimensions.

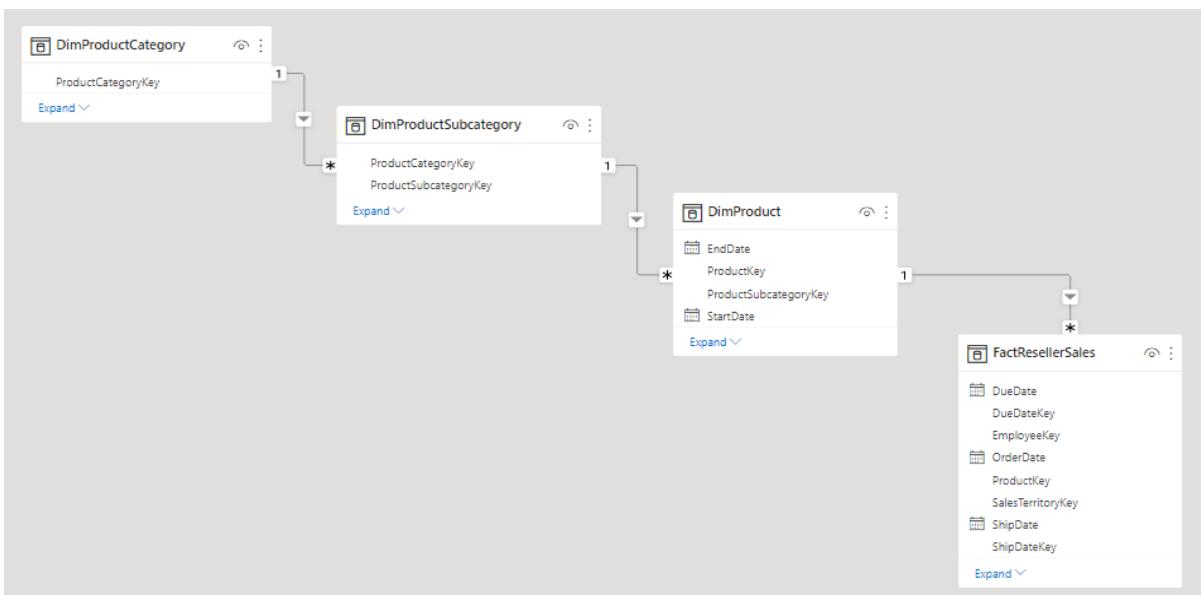
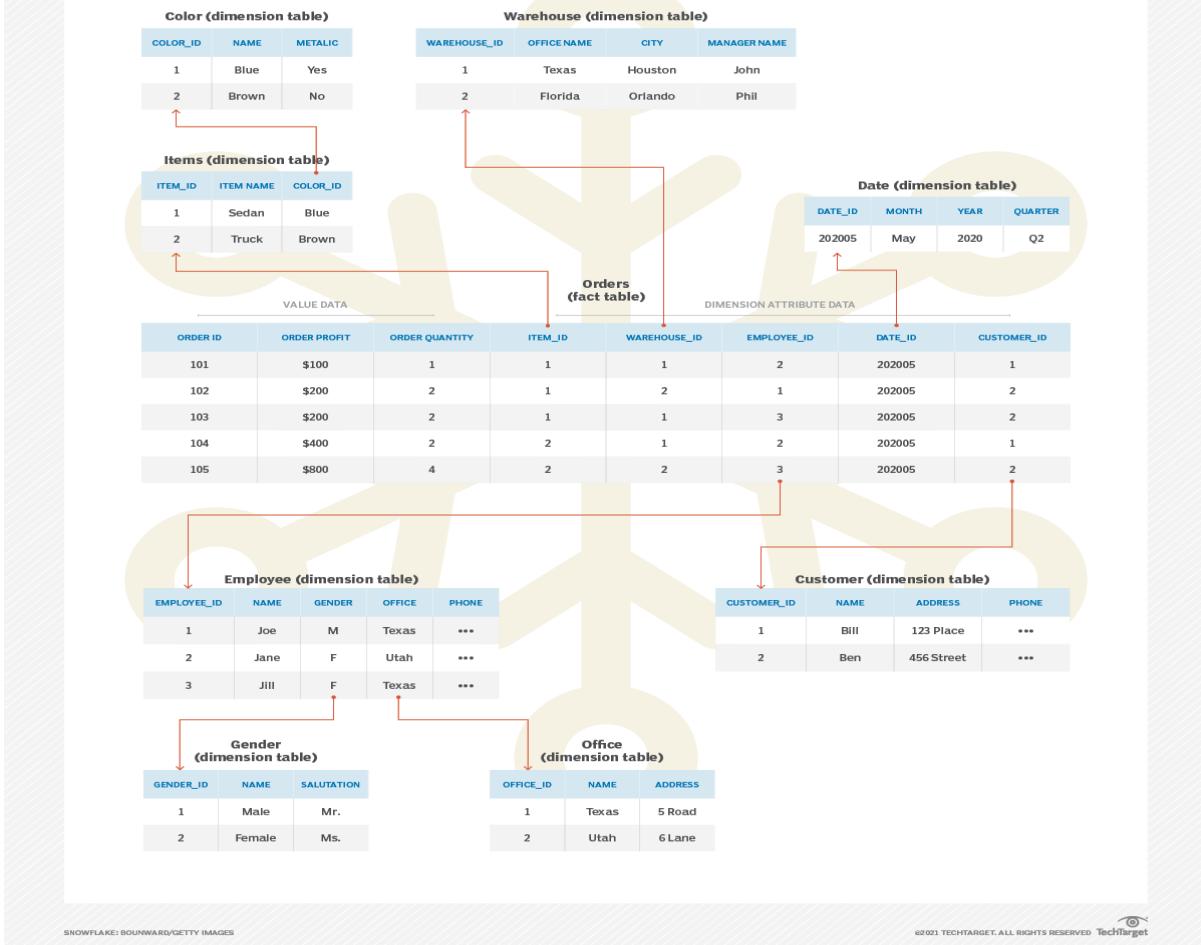
4.15 Snowflake Schema

A snowflake schema is a set of normalized tables for a single business entity. For example, Adventure Works classifies products by category and subcategory. Products are assigned to subcategories, and subcategories are in turn assigned to categories. In the Adventure Works relational data warehouse, the product dimension is normalized and stored in three related tables: Dim Product Category, Dim Product Subcategory, and Dim Product.

Data Analytics

Student Material

Snowflake schema

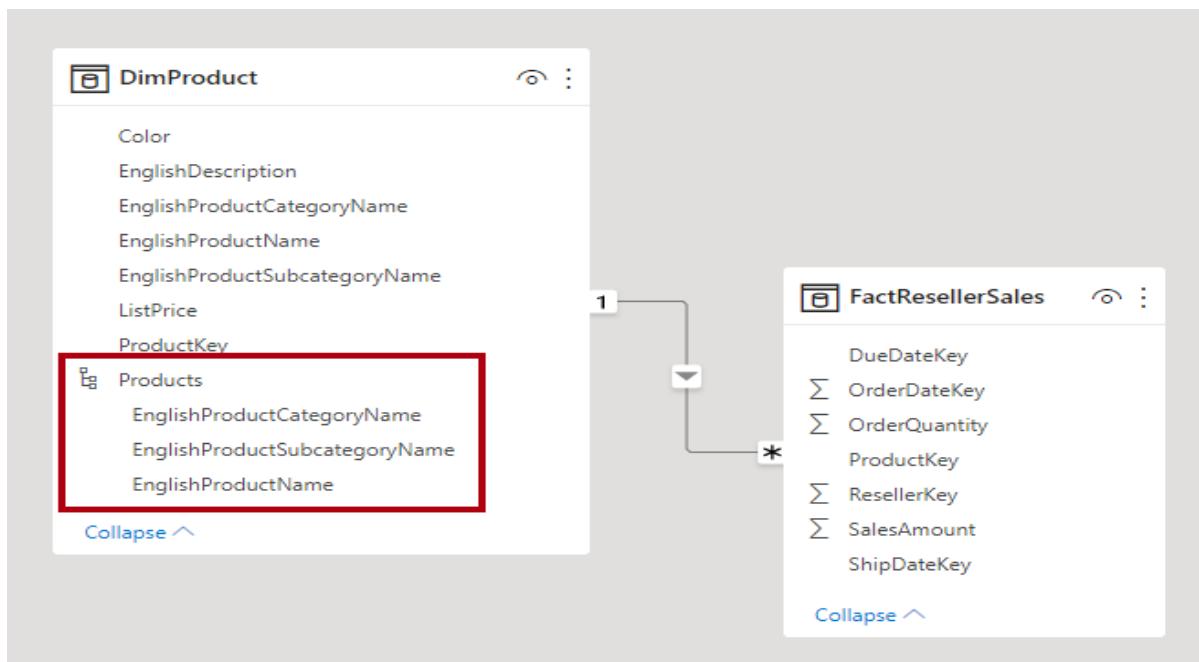


Brought to you by:



In Power BI Desktop, you can choose to mimic a snowflake dimension design (perhaps because your source data does) or integrate (denormalize) the source tables into a single model table. Generally, the benefits of a single model table outweigh the benefits of multiple model tables. The most optimal decision can depend on the volumes of data and the usability requirements for the model.

When you choose to integrate into a single model table, you can also define a hierarchy that encompasses the highest and lowest grain of the dimension. Possibly, the storage of redundant denormalized data can result in increased model storage size, particularly for very large dimension tables.



4.16 Star schema vs. snowflake schema

Star schema's dimension tables do not contain any foreign keys. That is, the dimension tables do not reference any other tables, nor do they have any "sub-dimension tables." They are generally denormalized because some information may be duplicated in the dimension tables. This allows star schema databases to be optimized for read and query performance along specific dimensions.

A snowflake schema database is similar to a star schema in that it has a single fact table and many dimension tables. However, for a snowflake schema, each dimension table might have foreign keys that relate to other dimension tables.

Data Analytics

Student Material

In a star schema, the record would contain information such as "male" and "Texas Office." This would be duplicated data also in other rows for employees with the same gender or branch office.

In a snowflake schema, the gender or branch office would contain a foreign key value to a gender dimension table and a branch office dimension table. These could contain information such as gender, name, salutation (Mr. or Ms.), branch office name, branch office address or branch manager.

4.17 Create and manage relationships

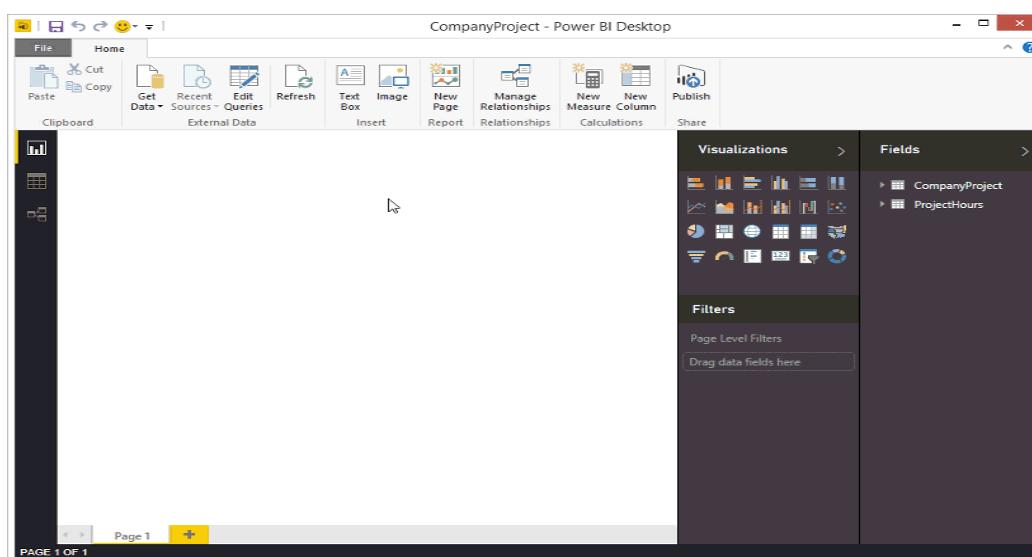
When you import multiple tables, chances are you'll do some analysis using data from all those tables. Relationships between those tables are necessary to accurately calculate results and display the correct information in your reports. Power BI Desktop makes creating those relationships easy. In fact, in most cases you won't have to do anything, the autodetect feature does it for you. However, sometimes you might have to create relationships yourself, or need to make changes to a relationship. Either way, it's important to understand relationships in Power BI Desktop and how to create and edit them.

Auto detect during load

If you query two or more tables at the same time, when the data is loaded, Power BI Desktop attempts to find and create relationships for you. The relationship options Cardinality, Cross filter direction, and Make this relationship active are automatically set. Power BI Desktop looks at column names in the tables you're querying to determine if there are any potential relationships. If there are, those relationships are created automatically. If Power BI Desktop can't determine with a high level of confidence there's a match, it doesn't create the relationship. However, you can still use the Manage relationships dialog box to manually create or edit relationships.

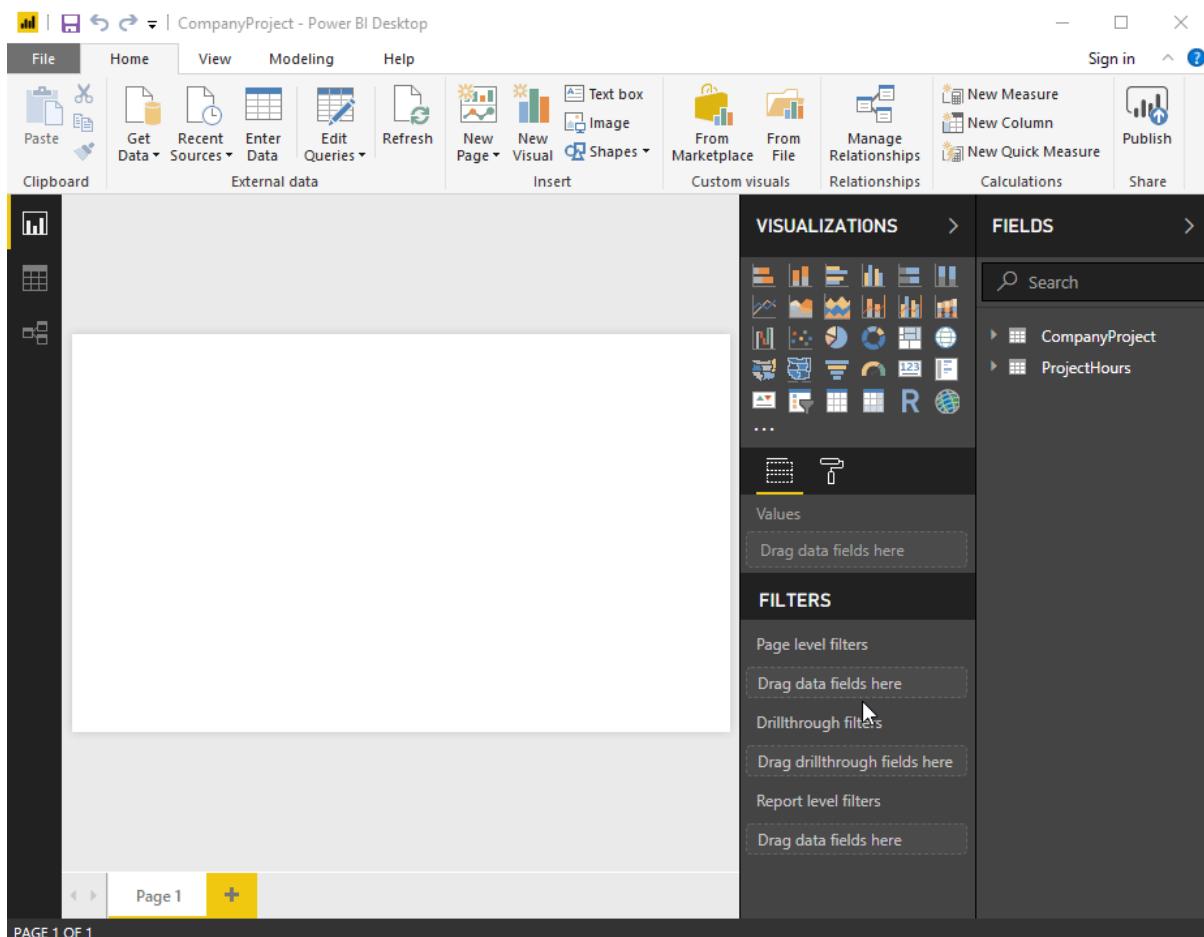
Create a relationship with auto detect:

On the Modelling tab, select Manage relationships > Auto detect.



Create a relationship manually:

- On the Modelling tab, select Manage relationships > New.
- In the Create relationship dialog box, in the first table drop-down list, select a table. Select the column you want to use in the relationship.
- In the second table drop-down list, select the other table you want in the relationship. Select the other column you want to use, and then select OK.



4.18 Types of relationships

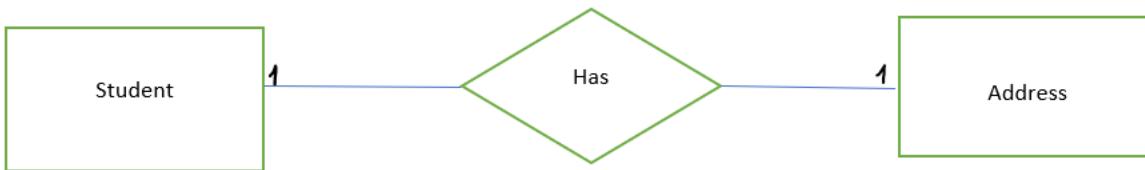
One to one (1:1):

One to one relationship means one entity in a table is related to just one entity in another table. It should be rare in any relational database design.

For instance, each student has only one address and each address represents one student.

Data Analytics

Student Material



A one-to-one relationship can be created when both tables each contain a column of common and unique values.

There are two scenarios that involve one-to-one relationships:

- Degenerate dimensions: You can derive a degenerate dimension from a fact-type table.
- Row data spans across tables: A single business entity or subject is loaded as two (or more) model tables, possibly because their data is sourced from different data stores. This scenario can be common for dimension-type tables. For example, master product details are stored in an operational sales system, and supplementary product details are stored in a different source.

It's unusual, however, that you'd relate two fact-type tables with a one-to-one relationship. It's because both fact-type tables would need to have the same dimensionality and granularity. Also, each fact-type table would need unique columns to allow the model relationship to be created.

Degenerate dimensions

When columns from a fact-type table are used for filtering or grouping, you can consider making them available in a separate table. This way, you separate columns used for filter or grouping, from those columns used to summarize fact rows.

This separation can:

- Reduce storage space
- Simplify model calculations
- Contribute to improved query performance
- Deliver a more intuitive Fields pane experience to your report authors

Consider a source sales table that stores sales order details in two columns.

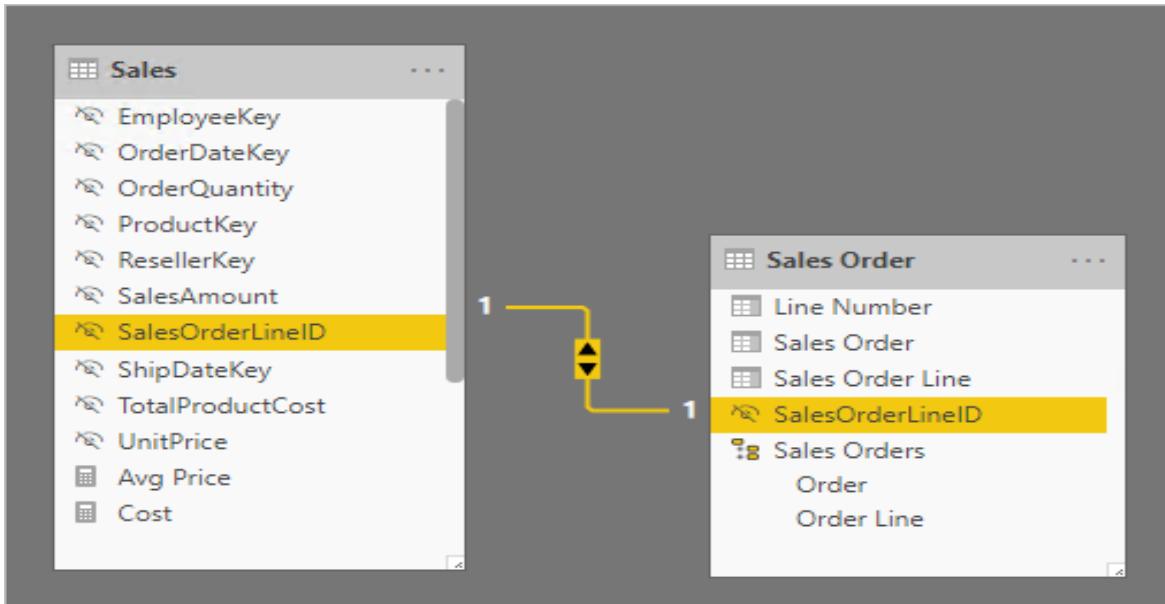
	OrderNumber	OrderLineNumber	OrderDate	DueDate	ShipDate	ProductID	CustomerID
1	43659	1	2018-12-29	2019-01-10	2019-01-05	349	676
2	43659	2	2018-12-29	2019-01-10	2019-01-05	350	676
3	43659	3	2018-12-29	2019-01-10	2019-01-05	351	676
4	43660	1	2018-12-29	2019-01-10	2019-01-05	326	117
5	43660	2	2018-12-29	2019-01-10	2019-01-05	319	117

The Order Number column stores the order number, and the Order Line Number column stores a sequence of lines within the order. In the following model diagram, notice that the

Data Analytics

Student Material

order number and order line number columns haven't been loaded to the Sales table. Instead, their values were used to create a surrogate key column named Sales Order Line ID. (The key value is calculated by multiplying the order number by 1000, and then adding the order line number.)



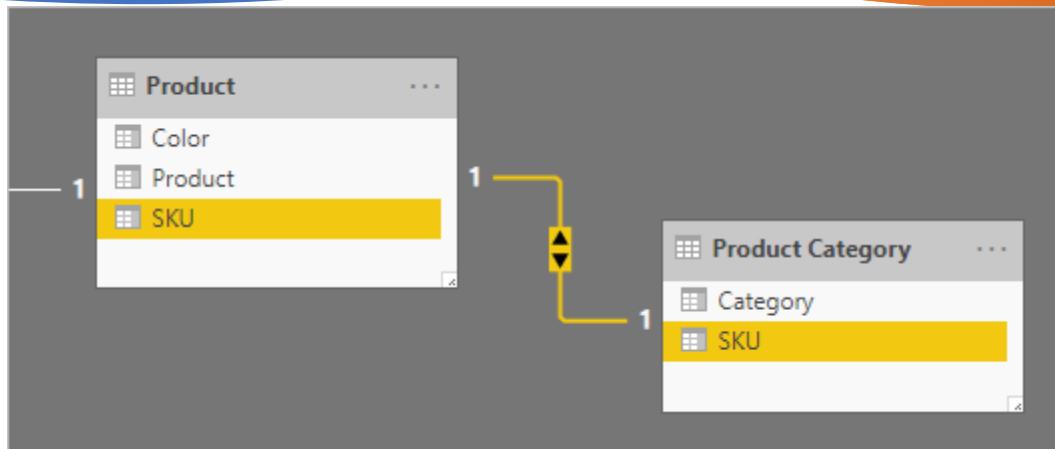
The Sales Order table provides a rich experience for report authors with three columns: Sales Order, Sales Order Line, and Line Number. It also includes a hierarchy. These table resources support report designs that need to filter, group by, or drill down through orders and order lines.

As the Sales Order table is derived from the sales data, there should be exactly the same number of rows in each table. Further, there should be matching values between each Sales Order Line ID column.

Row data spans across tables

Consider an example involving two one-to-one related dimension-type tables: Product, and Product Category. Each table represents imported data and has a SKU (Stock-Keeping Unit) column containing unique values.

Here's a partial model diagram of the two tables.



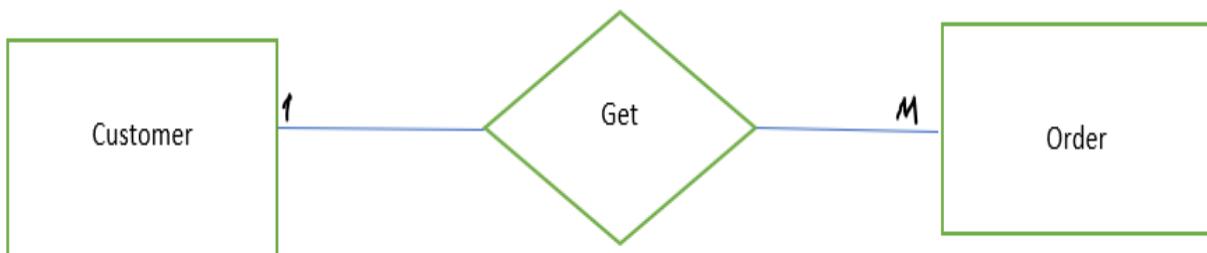
The first table is named Product, and it contains three columns: Color, Product, and SKU. The second table is named Product Category, and it contains two columns: Category, and SKU. A one-to-one relationship relates the two SKU columns. The relationship filters in both directions, which is always the case for one-to-one relationships.

To help describe how the relationship filter propagation works, the model diagram has been modified to reveal the table rows. All examples in this article are based on this data.

One to Many (1 : M) Relationship:

One to many relationship means one entity in a table is related to one or many entities in another table. A one to many (1 : M) relationship should be the norm in any relational database design and is found in all relational database environments.

For example, each customer has one or more orders.

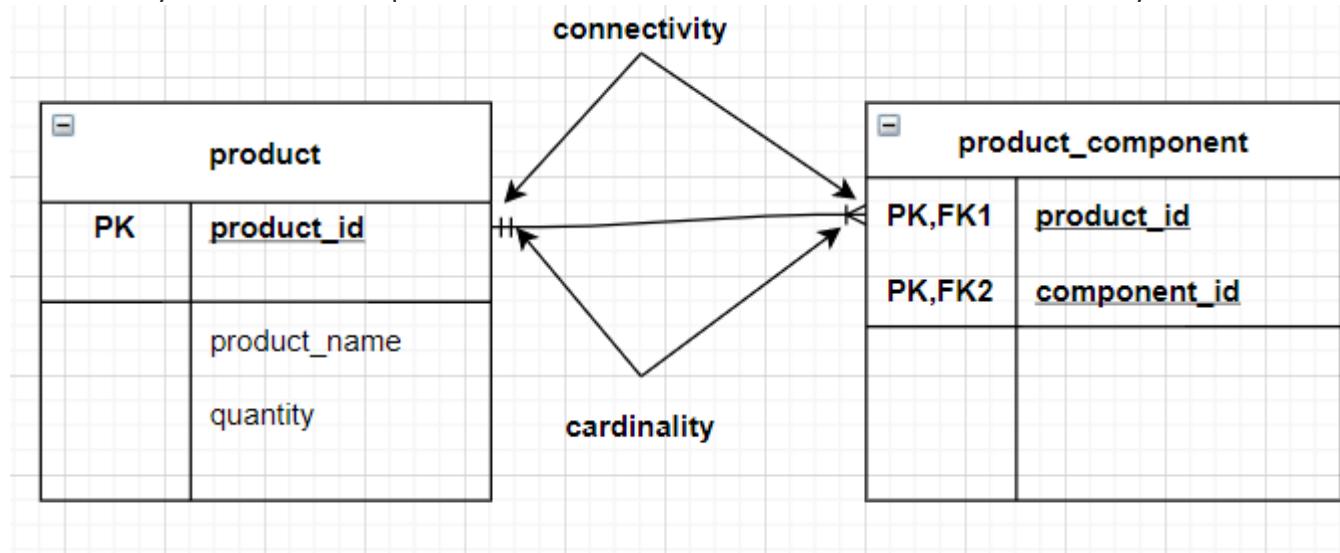


Before proceeding many to many relationships, let's look at the cardinality and connectivity. These are helpful to understand many to many relationships.

Cardinality and Connectivity

Cardinality describes the relationship between two data tables by expressing the minimum and maximum number of entity occurrences associated with one occurrence of a related entity. Either there is a relationship or not.

Connectivity is the relationship between two tables. It can be one to one or one to many.

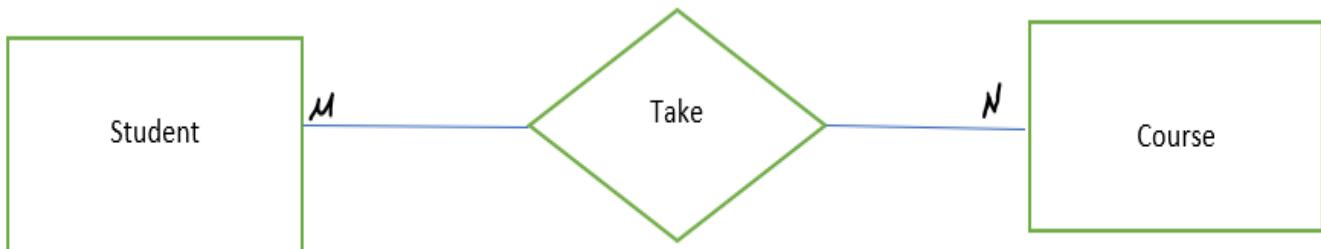


Many to one (*:1): A many-to-one relationship is the most common, default type of relationship. It means the column in a given table can have more than one instance of a value, and the other related table, often known as the lookup table, has only one instance of a value.

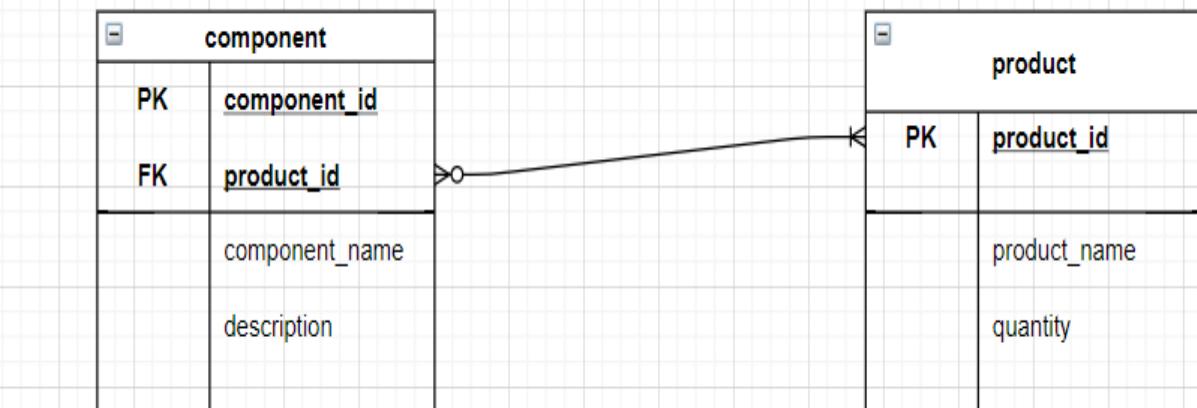
Many to Many (M : N) Relationship

Many to many relationship means multiple entities in a table can be associated with more than one entity in another table.

For instance, each student can attend many courses and, each course can consist of many students.



Many to many relationship causes duplications in database. Duplications cause false results from queries. For instance, a company produces products, which consist of several components. One product is made up of many components, and one component can be in more than one product.



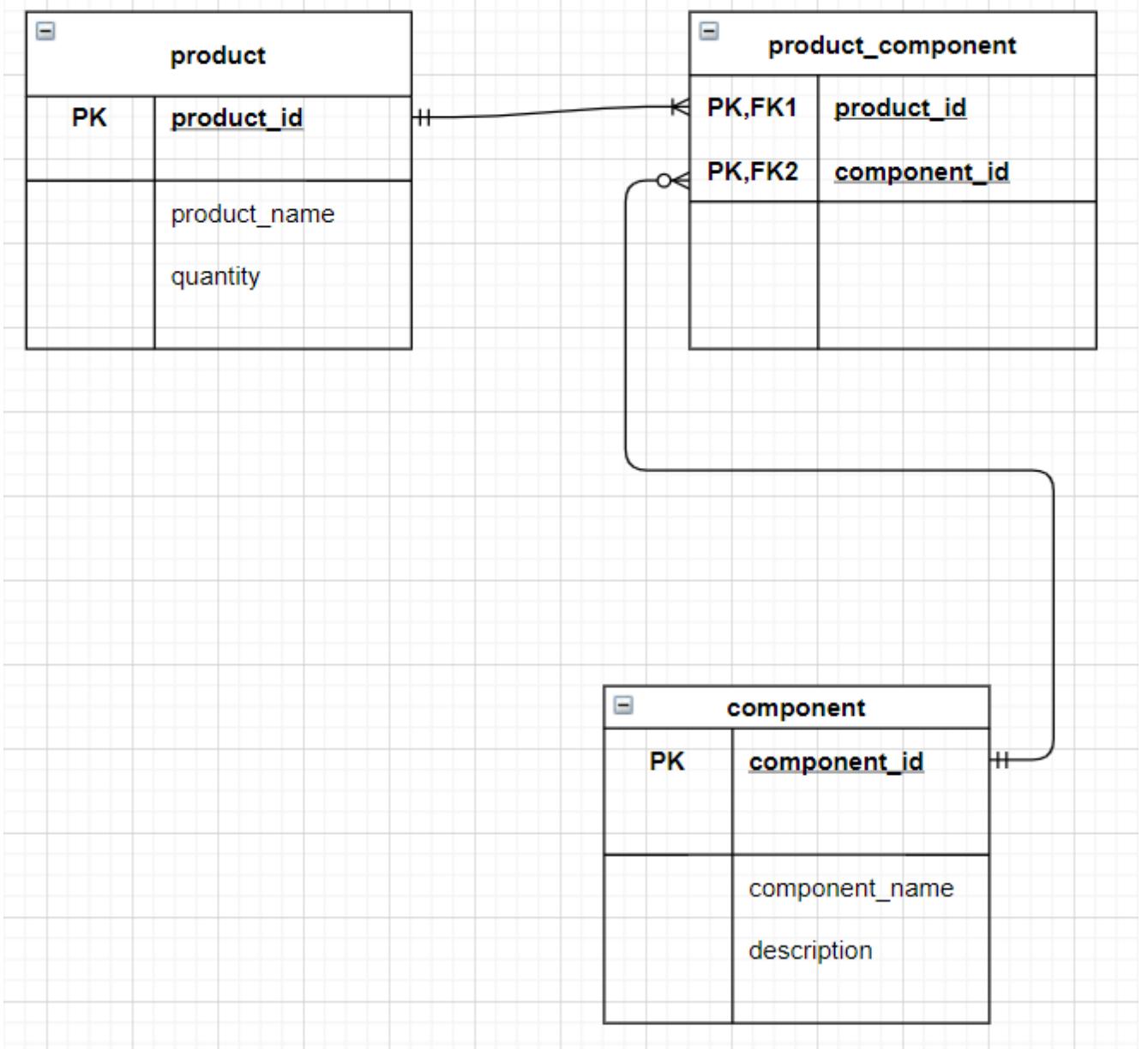
In the Entity-Relationship (ER) Diagram, “component_id” is specified as Primary Key in a component table, and Primary Key should be unique. Looking at the component table below, the primary key(component_id), component_name, and description are duplicated in order to record detailed information about each component.

Product			Component			
product_id	product_name	quantity	component_id	product_id	component_name	description
1 x		2		1	1 a	b
2 y		3		1	2 a	b
3 z		2		1	3 a	b
				2	1 d	e

Let's explain a way to prevent the duplication while setting many to many relationships. The solution is creating a totally new table. It's called as bridge table or join table.

Data Analytics

Student Material



Data Analytics

Student Material

Product			Component		
product_id	product_name	quantity	component_id	component_name	description
1	x	2	1	a	b
2	y	3	2	d	e
3	z	2			

Product_Component		
product_id	component_id	type
1	1	k
2	1	l
3	1	m
1	2	n

Bridge table consist of “product_id” , “component_id” from related tables’ Primary Keys. “product_id” and “component_id” are both Primary and Foreign Keys the bridge table. Also, new columns can be added that are not available in the product table and component table. Entity-Relationship (ER) Diagram helps to make and understand Entity Relationship (ER) Data Model. This site which is the below can be used for Entity Relationship (ER) Data Modelling with Entity-Relationship (ER) Diagram. This website helps to create and visualize diagram. The visualization makes it easier to understand the relationship between the tables. Also, this website is free and speedy. It has to many options to create diagram.

To sum up, relational database has ability to create meaningful information by joining tables. Joining tables help to understand the relationship between the data and tables. Besides, relational databases eliminate data redundancy. Relational database has three relationships between tables. Many to many relationship is harder to understand than other relationships. The bridge table makes the database and relationship easy to understand, and it prevents data redundancy.

Assignment:

Create data model relationship using Power BI

Solution:

Creating Model Relationships

- Open the Power BI Desktop application and click on the “File” ribbon.

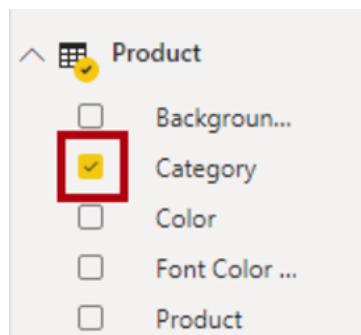
Data Analytics

Student Material

- Click on the “Open” option and import the data for making Power BI Data Model.
- For this tutorial, sample data from Sales Analysis will be used.
- Click on the “Model View” option from the left side of the Power BI, as shown in the image below.



- Here, you can view each table and relationship.
- In the “Fields” pane right-click on an empty area and click on the “Expand All” option to view all the table fields.
- Now, create a visual table by selecting the “Category” field inside the “Product” table, as shown in the image below.



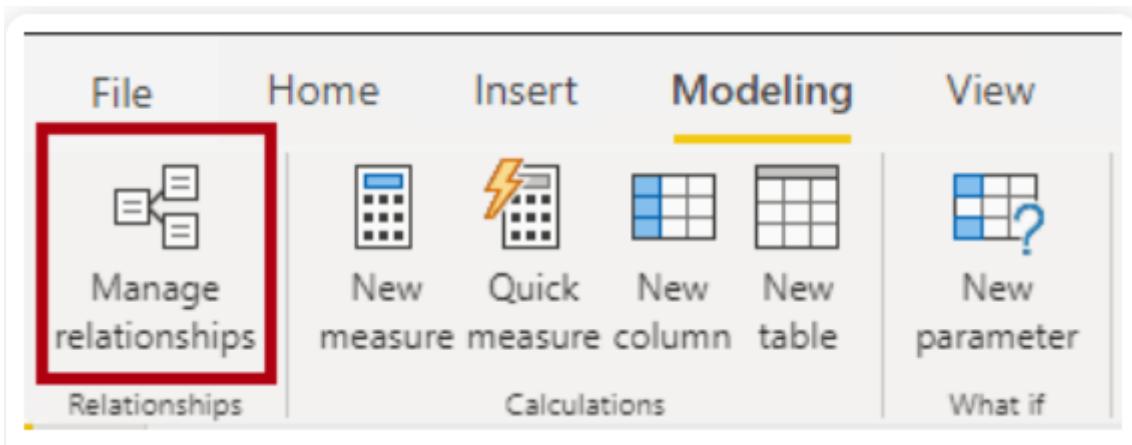
- For adding a column to the table, go to the “Fields” pane and select the “Sales” from the “Sales” table, as shown in the image below.

Category	Sales
Accessories	\$77,548,570.2
Bikes	\$77,548,570.2
Clothing	\$77,548,570.2
Components	\$77,548,570.2
Total	\$77,548,570.2

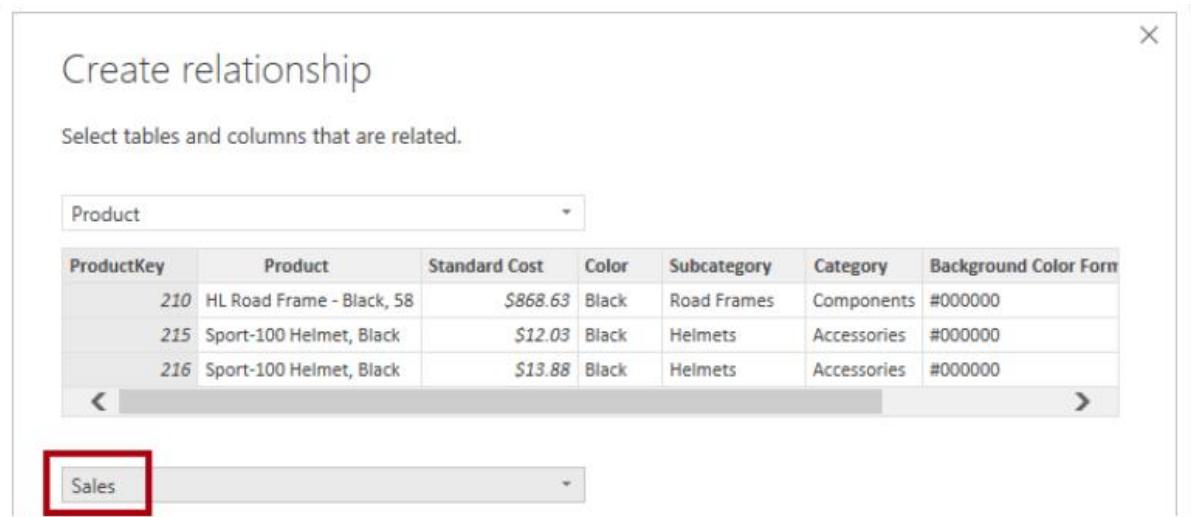
Data Analytics

Student Material

- There is no relationship between tables, so it will not filter the “Sales” table.
- Navigate to the “Modeling” ribbon, and from the “Relationships” group, select the “Manage Relationships” option, as shown in the image below.



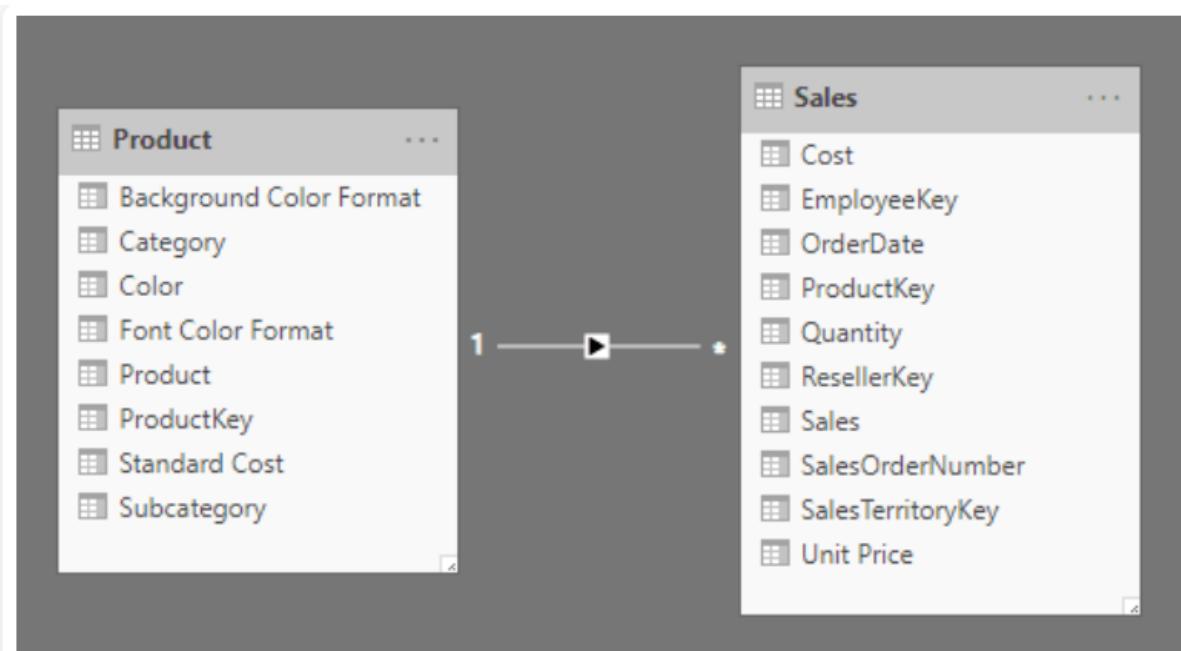
- Click on the “New” button to create a new relationship.
- It will open the “Create Relationship” window.
- Select the “Product” table from the first drop-down list, then select the “Sales” table from the second drop-down list, as shown in the image below.



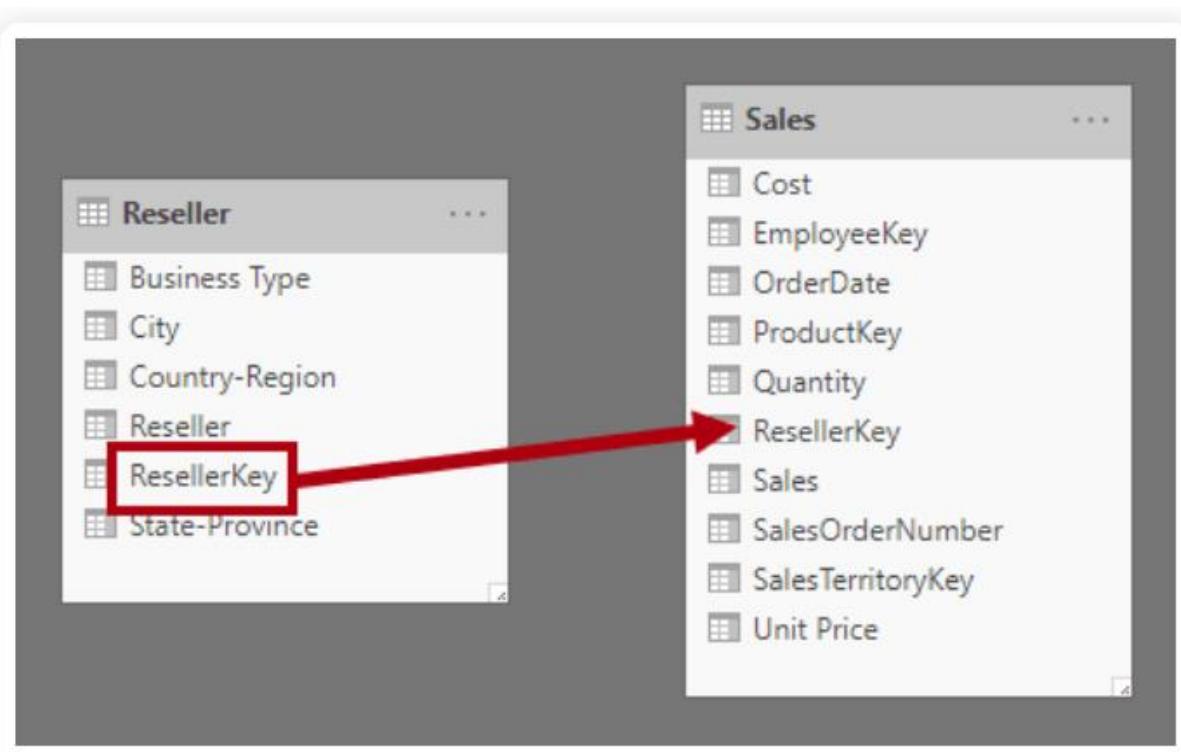
- The “ProductKey” columns from both the tables are automatically selected as common columns because it has the same data type and name.
- Click on the “Ok” button.
- Now notice that the table visual has been updated to display values for each product category.
- If you open the “Model View”, you will find a connector between the two tables that were not present before, as shown in the image below.

Data Analytics

Student Material



- Another way to create a relationship between tables is by dragging one column from one table to another.
- In this Model View, select the “**ResellerKey**” column from the “**Reseller**” table and drag it onto the “**Resellerkey**” column of the “**Sales**” table, as shown in the image below.



Data Analytics

Student Material

- Similarly, create the following relationships by dragging columns listed below:
 - “**SalesTerritoryKey**” column from the “**Region**” table to “**SalesTerritoryKey**” column of “**Sales**” table.
 - “**EmployeeKey**” column from the “**Salesperson**” table to the “**EmployeeKey**” column of the “**Sales**” table.
- Now save the Power BI file.

Chapter: 5

Data Visualization

Scope:

- Overview of Power BI
- Advantages of using Power BI
- Getting Started (Power BI Desktop)
- Transforming Data
- Report Creation
- Chart Selection
- Different types of visualization charts
- Area Charts
- Line Charts
- Bar Charts
- Column Charts
- Combo Charts

Data visualization Overview

It is the representation of data through use of common graphics, such as charts, plots, infographics, and even animations. These visual displays of information communicate complex data relationships and data-driven insights in a way that is easy to understand.

Data visualization can be utilized for a variety of purposes, and it's important to note that is not only reserved for use by data teams. Management also leverages it to convey organizational structure and hierarchy while data analysts and data scientists use it to discover and explain patterns and trends. Harvard Business Review (link resides outside IBM) categorizes data visualization into four key purposes: idea generation, idea illustration, visual discovery, and everyday dataviz. We'll delve deeper into these below:

Data visualization is commonly used to spur idea generation across teams. They are frequently leveraged during brainstorming or Design Thinking sessions at the start of a project by supporting the collection of different perspectives and highlighting the common concerns of the collective. While these visualizations are usually unpolished and unrefined, they help set the foundation within the project to ensure that the team is aligned on the problem that they're looking to address for key stakeholders.

Types of data visualizations

The earliest form of data visualization can be traced back the Egyptians in the pre-17th century, largely used to assist in navigation. As time progressed, people leveraged data visualizations for broader applications, such as in economic, social, health disciplines. Perhaps most notably, Edward Tufte published *The Visual Display of Quantitative Information* (link resides outside IBM), which illustrated that individuals could utilize data visualization to present data in a more effective manner. His book continues to stand the test of time, especially as companies turn to dashboards to report their performance metrics in real-time. Dashboards are effective data visualization tools for tracking and visualizing data from multiple data sources, providing visibility into the effects of specific behaviors by a team or an adjacent one on performance. Dashboards include common visualization techniques, such as:

Tables: This consists of rows and columns used to compare variables. Tables can show a great deal of information in a structured way, but they can also overwhelm users that are simply looking for high-level trends.

Pie charts and stacked bar charts: These graphs are divided into sections that represent parts of a whole. They provide a simple way to organize data and compare the size of each component to one other.

Line graphs and area charts: These visuals show change in one or more quantities by plotting a series of data points over time. Line graphs utilize lines to demonstrate these changes while area charts connect data points with line segments, stacking variables on top of one another and using color to distinguish between variables.

Histograms: This graph plots a distribution of numbers using a bar chart (with no spaces between the bars), representing the quantity of data that falls within a particular range. This visual makes it easy for an end user to identify outliers within a given dataset.

Scatter plots: These visuals are beneficial in revealing the relationship between two variables, and they are commonly used within regression data analysis. However, these can sometimes be confused with bubble charts, which are used to visualize three variables via the x-axis, the y-axis, and the size of the bubble.

Heat maps: These graphical displays are helpful in visualizing behavioral data by location. This can be a location on a map, or even a webpage.

Tree maps: which display hierarchical data as a set of nested shapes, typically rectangles. Tree maps are great for comparing the proportions between categories via their area size.

Open-source visualization tools

Access to data visualization tools has never been easier. Open source libraries, such as D3.js, provide a way for analysts to present data in an interactive way, allowing them to engage a broader audience with new data. Some of the most popular open source visualization libraries include:

D3.js: It is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. D3.js ([link resides outside IBM](#)) uses HTML, CSS, and SVG to create visual representations of data that can be viewed on any browser. It also provides features for interactions and animations.

ECharts: A powerful charting and visualization library that offers an easy way to add intuitive, interactive, and highly customizable charts to products, research papers, presentations, etc. Echarts ([link resides outside IBM](#)) is based in JavaScript and ZRender, a lightweight canvas library.

Vega: Vega ([link resides outside IBM](#)) defines itself as “visualization grammar,” providing support to customize visualizations across large datasets which are accessible from the web.

deck.gl: It is part of Uber's open source visualization framework suite. deck.gl ([link resides outside IBM](#)) is a framework, which is used for exploratory data analysis on big data. It helps build high-performance GPU-powered visualization on the web.

5.1 Overview of Power BI

Power BI gives the ability to analyze and explore data on-premise as well as in the cloud. Power BI provides the ability to collaborate and share customized dashboards and interactive reports across colleagues and organizations, easily and securely.

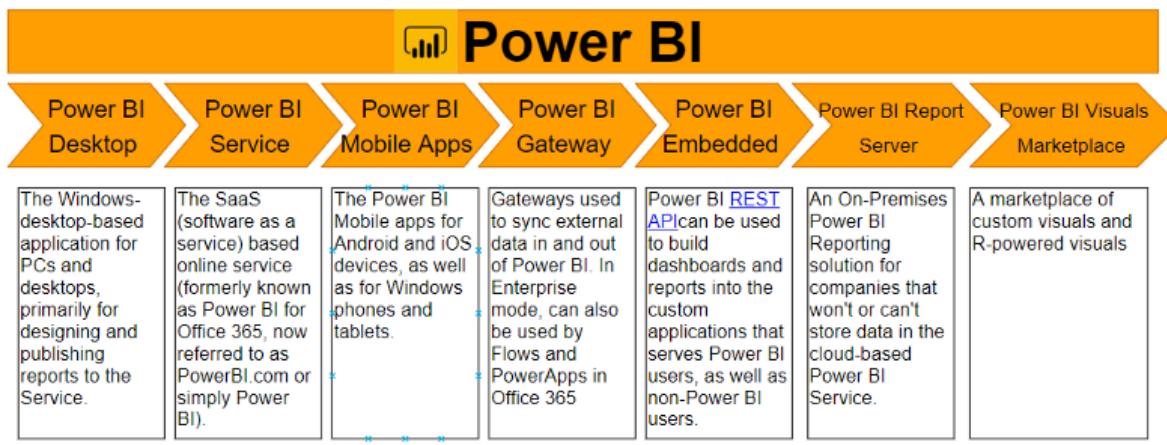
Data Analytics

Student Material



Power BI's Components:

Power BI consists of various components which are available in the market separately and can be used exclusively.



Power BI Desktop

The Windows desktop-based application for PCs and desktops, primarily for designing and publishing reports to the Service.

Power BI Service

The SaaS-based (software as a service) online service. This was formerly known as Power BI for Office 365, now referred to as PowerBI.com, or simply Power BI.

Power BI Mobile Apps

The Power BI Mobile apps for Android and iOS devices, as well as for Windows phones and tablets.

Power BI Gateway

Gateways are used to sync external data in and out of Power BI and are required for automated refreshes. In Enterprise mode, can also be used by Power Automate (previously called Flows) and PowerApps in Office 365.

Power BI Embedded

Power BI REST API can be used to build dashboards and reports into the custom applications that serve Power BI users, as well as non-Power BI users.

Power BI Report Server

An on-premises Power BI reporting solution for companies that won't or can't store data in the cloud-based Power BI Service.

Power BI Premium

Capacity-based offering that includes flexibility to publish reports broadly across an enterprise, without requiring recipients to be licensed individually per user. Greater scale and performance than shared capacity in the Power BI Service.

Power BI Visuals Marketplace

A marketplace of custom visuals and R-powered visuals.

Power BI Dataflow

A Power Query implementation in the cloud that can be used for data transformations to make a common Power BI Dataset that can be made available for several report developers through Microsoft's Common Data Service. It can be used as an alternative to for example doing transformations in SSAS and may ensure that several report developers use data that has been transformed in a similar way.

Power BI Dataset

A Power BI Dataset can work as a collection of data for use in Power BI reports, and can either be connected to or imported into a Power BI Report. A Dataset can be connected to and get its source data through one or more Data flows.

5.2 Advantages of using Power BI:

- **A quick start.** You'll be able to get insights quickly with an uncomplicated setup, no required training, and included dashboards for services such as Salesforce, Google Analytics, and Microsoft Dynamics.
- **Streamlined publication and distribution.** Instead of emailing large files or putting them on a shared drive, analysts upload reports and visualizations to the Power BI service, and their data is refreshed whenever the underlying dataset is updated.
- **Real-time information.** Dashboards update in real time, as data is pushed or streamed in, which gives viewers the ability to solve problems and identify opportunities quickly. Any report or dashboard can display and update real-time data and visuals. Sources of streaming data can be factory sensors, social media sources, or anything from which time-sensitive data can be collected or transmitted.
- **Ability to customize Power BI app navigation.** An "app navigation experiences" feature gives report developers the power to customize navigation to help viewers

find content quickly and understand the relationships between different reports and dashboards.

- **Ability to customize security features.** Report developers can set up row-level security (RLS) access filters to ensure that viewers see only data relevant to them, mitigating the risk of people seeing data they shouldn't.
- **Cortana integration.** Power BI works with Microsoft's digital assistant, Cortana. Users can verbally ask questions in natural language to access charts and graphs. This can be especially helpful for users with mobile devices.
- **Artificial Intelligence.** Power BI users can access image recognition and text analytics, create machine learning models, and integrate with Azure Machine Learning.

5.3 Chart Selection:

Area charts: Basic (Layered) and Stacked



The basic area chart is based on the line chart with the area between the axis and line filled in. Area charts emphasize the magnitude of change over time, and can be used to draw attention to the total value across a trend. For example, data that represents profit over time can be plotted in an area chart to emphasize the total profit.

Bar and column charts

Data Analytics

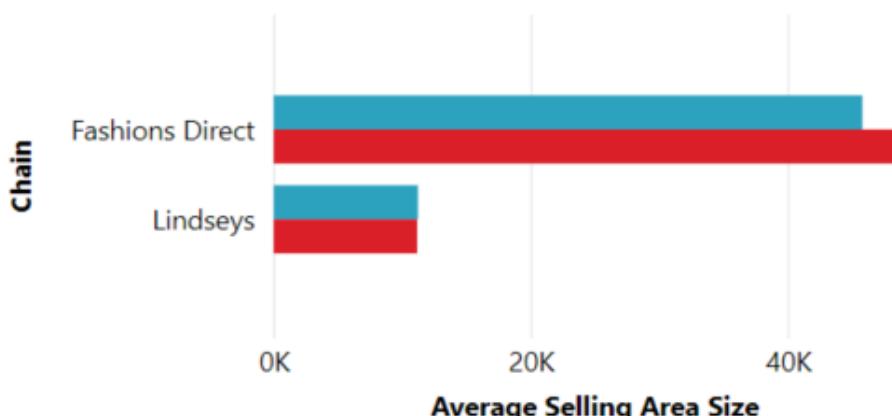
Student Material

Total Sales Variance % by FiscalMonth



Average Selling Area Size by Store Type, Chain

Store Type ● New Store ● Same Store



Bar charts are the standard for looking at a specific value across different categories.

Cards

Multi row

030-Kids
\$5.30
Average Unit Price

Multi row cards display one or more data points, one per row.

Single number

104

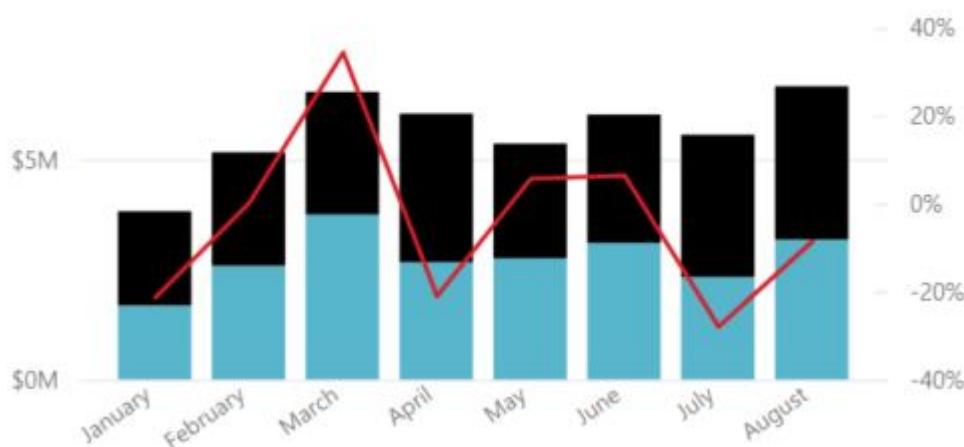
Total Stores

Single number cards display a single fact, a single data point. Sometimes a single number is the most important thing you want to track in your Power BI dashboard or report, such as total sales, market share year over year, or total opportunities.

Combo charts

This Year Sales, Last Year Sales and Total Sales Variance % by Month

● This Year Sales ● Last Year Sales ● Total Sales Variance %



A combo chart combines a column chart and a line chart. Combining the two charts into one lets you make a quicker comparison of the data. Combo charts can have one or two Y axes, so be sure to look closely.

Combo charts are a great choice:

- When you have a line chart and a column chart with the same X axis.
- To compare multiple measures with different value ranges.
- To illustrate the correlation between two measures in one visual.
- To check whether one measure meets the target which is defined by another measure.
- To conserve canvas space.

Data Analytics

Student Material

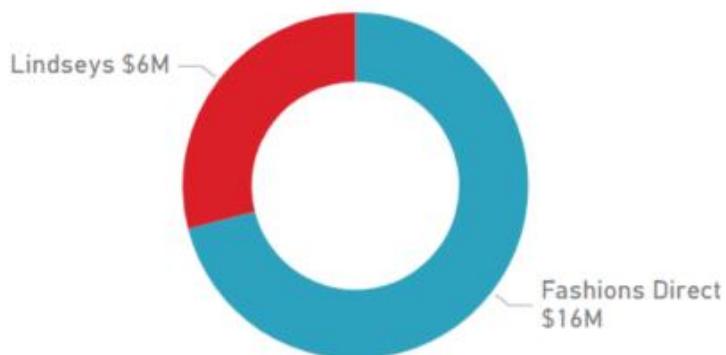
Decomposition tree



The decomposition tree visual lets you visualize data across multiple dimensions. It automatically aggregates data and enables drilling down into your dimensions in any order. It is also an artificial intelligence (AI) visualization, so you can ask it to find the next dimension to drill down into based on certain criteria. This makes it a valuable tool for ad hoc exploration and conducting root cause analysis.

Doughnut charts

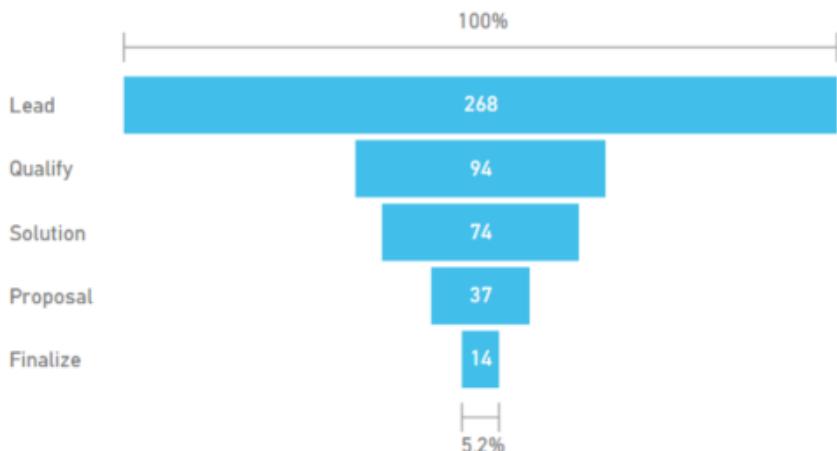
This Year Sales by Chain



Doughnut charts are similar to pie charts. They show the relationship of parts to a whole. The only difference is that the center is blank and allows space for a label or icon.

Funnel charts

Opportunity Count by Sales Stage



Funnels help visualize a process that has stages, and items flow sequentially from one stage to the next. One example is a sales process that starts with leads and ends with purchase fulfillment.

For example, a sales funnel that tracks customers through stages: Lead > Qualified Lead > Prospect > Contract > Close. At a glance, the shape of the funnel conveys the health of the process you're tracking. Each funnel stage represents a percentage of the total. So, in most cases, a funnel chart is shaped like a funnel -- with the first stage being the largest, and each subsequent stage smaller than its predecessor. A pear-shaped funnel is also useful -- it can identify a problem in the process. But typically, the first stage, the "intake" stage, is the largest.

Gauge charts

Average of Gross Sales



A radial gauge chart has a circular arc and displays a single value that measures progress toward a goal. The goal, or target value, is represented by the line (needle). Progress toward

Data Analytics

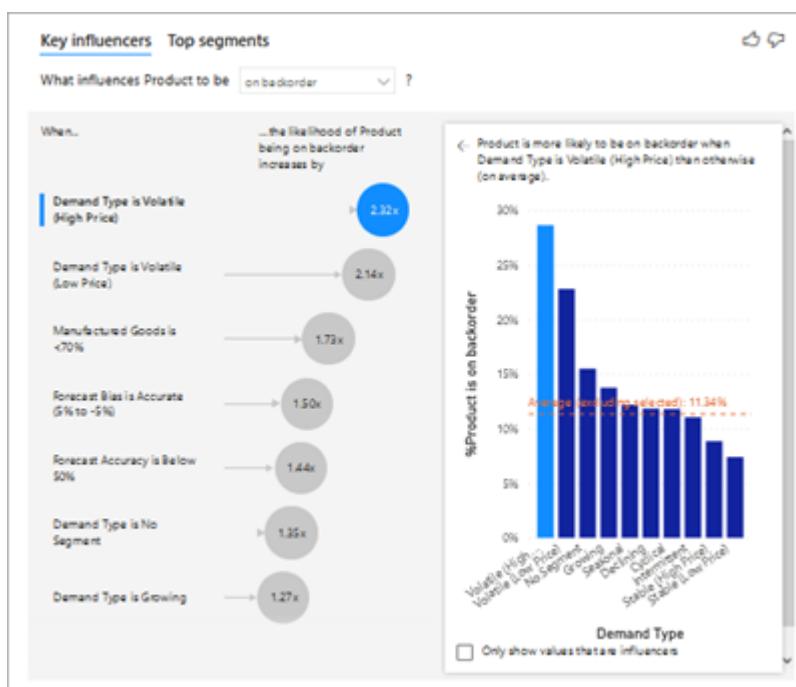
Student Material

that goal is represented by the shading. And the value that represents that progress is shown in bold inside the arc. All possible values are spread evenly along the arc, from the minimum (left-most value) to the maximum (right-most value).

Radial gauges are a great choice to:

- Show progress toward a goal.
- Represent a percentile measure, like a KPI.
- Show the health of a single measure.
- Display information that can be quickly scanned and understood.

Key influencers chart



A key influencer chart displays the major contributors to a selected result or value.

Key influencers are a great choice to help you understand the factors that influence a key metric. For example, *what influences customers to place a second order* or *why were sales so high last June*.

KPIs

Total Units This Year and Total Units Last Year by Month



A Key Performance Indicator (KPI) is a visual cue that communicates the amount of progress made toward a measurable goal.

KPIs are a great choice:

- To measure progress (what am I ahead or behind on?).
- To measure distance to a metric (how far ahead or behind am I?).

Line charts

This Year Sales and Last Year Sales by FiscalMonth



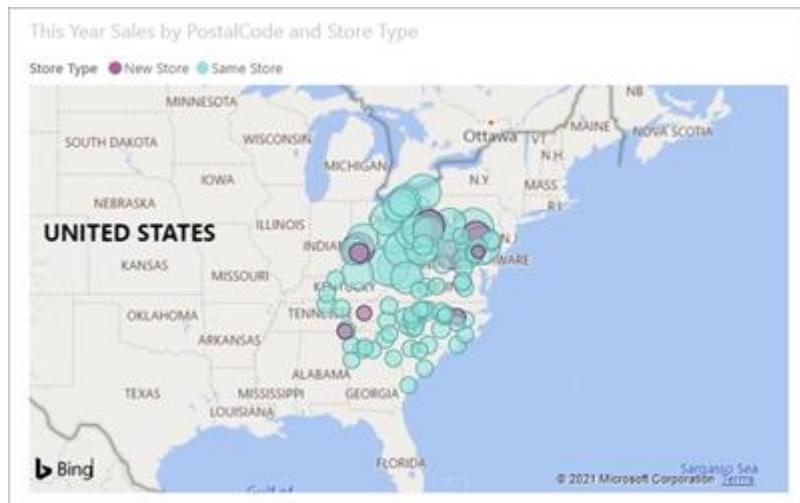
Line charts emphasize the overall shape of an entire series of values, usually over time.

Data Analytics

Student Material

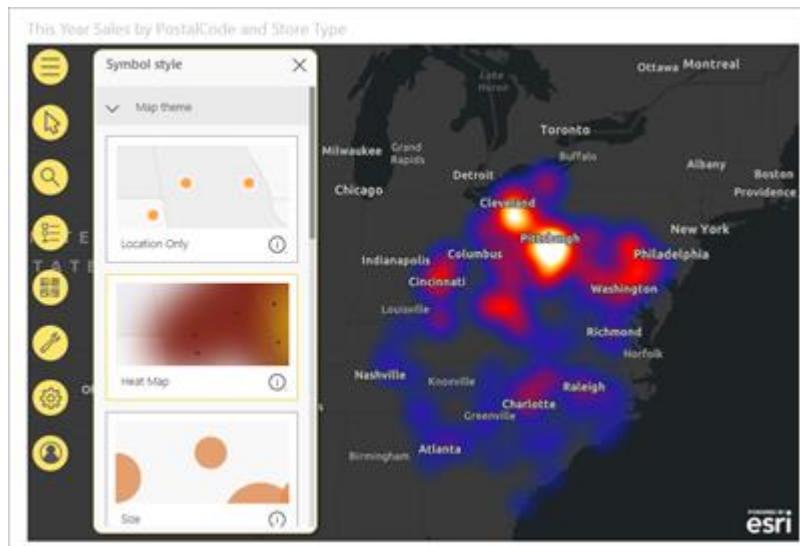
Maps

Basic map



Use a basic map to associate both categorical and quantitative information with spatial locations.

ArcGIS map

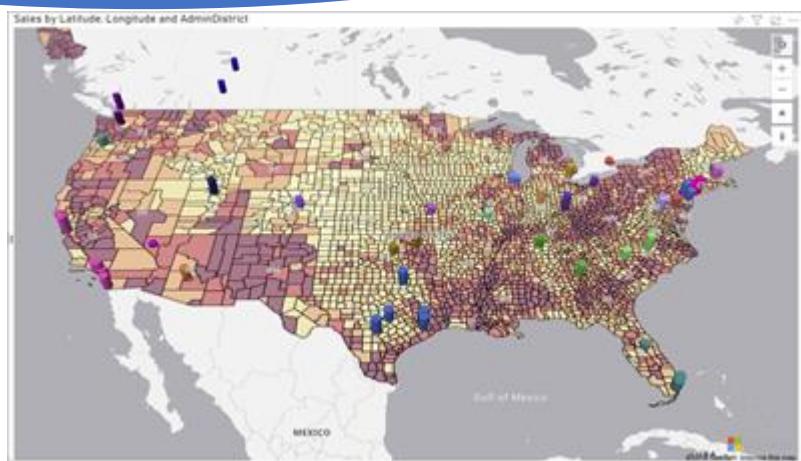


The combination of ArcGIS maps and Power BI takes mapping beyond the presentation of points on a map to a whole new level. The available options for base maps, location types, themes, symbol styles, and reference layers creates gorgeous informative map visuals. The combination of authoritative data layers (such as census data) on a map with spatial analysis conveys a deeper understanding of the data in your visual.

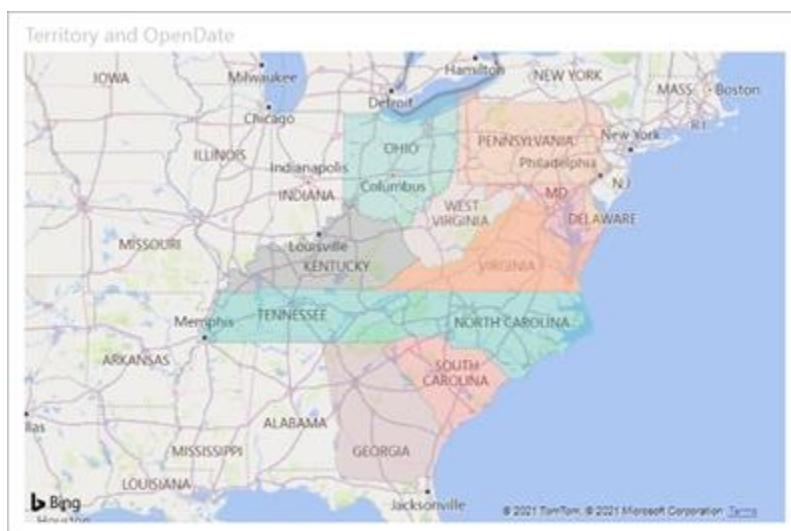
Azure map

Data Analytics

Student Material

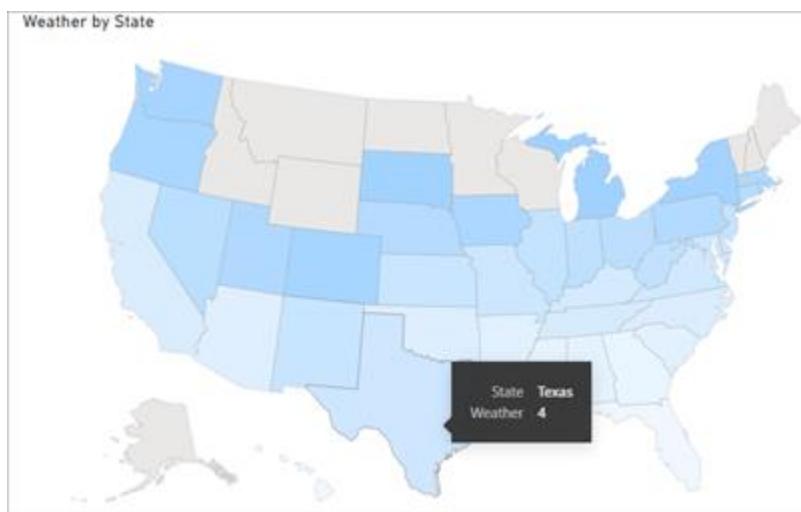


Filled map (Choropleth)



A filled map uses shading or tinting or patterns to display how a value differs in proportion across a geography or region. Quickly display these relative differences with shading that ranges from light (less-frequent/lower) to dark (more-frequent/more).

Shape map



Data Analytics

Student Material

Shape maps compare regions on a map using color. A shape map can't show precise geographical locations of data points on a map. Instead, its main purpose is to show relative comparisons of regions on a map by coloring them differently.

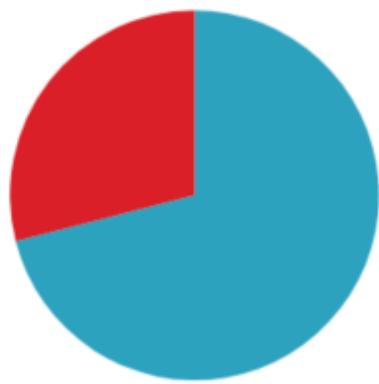
Matrix

Drill on Rows													
Region	Sales Stage	Central	East	West	Total	Central	East	West	Total	Central	East	West	Total
		Opportunity Count	Revenue	Opportunity Count	Revenue	Opportunity Count	Revenue						
Lead		102	\$307,574,417	114	\$473,887,937	52	\$206,198,114	268	\$1,237,621,368				
Quotation		28	\$111,715,461	30	\$185,692,154	18	\$52,442,363	94	\$359,849,978				
Solution		29	\$100,743,788	30	\$134,347,170	18	\$31,441,301	74	\$288,532,460				
Proposal		14	\$46,722,689	13	\$39,970,954	10	\$41,032,689	37	\$149,726,462				
Finalize		3	\$13,332,248	3	\$12,694,425	4	\$21,176,183	14	\$75,174,859				
Total		179	\$790,958,782	212	\$894,594,513	96	\$426,231,832	487	\$2,118,905,127				

The matrix visual is a type of table visual that supports a stepped layout. A table supports two dimensions, but a matrix makes it easier to display data meaningfully across multiple dimensions. Often, report designers include matrixes in reports and dashboards to allow users to select one or more element (rows, columns, cells) in the matrix to cross-highlight other visuals on a report page.

Pie charts

This Year Sales by Chain



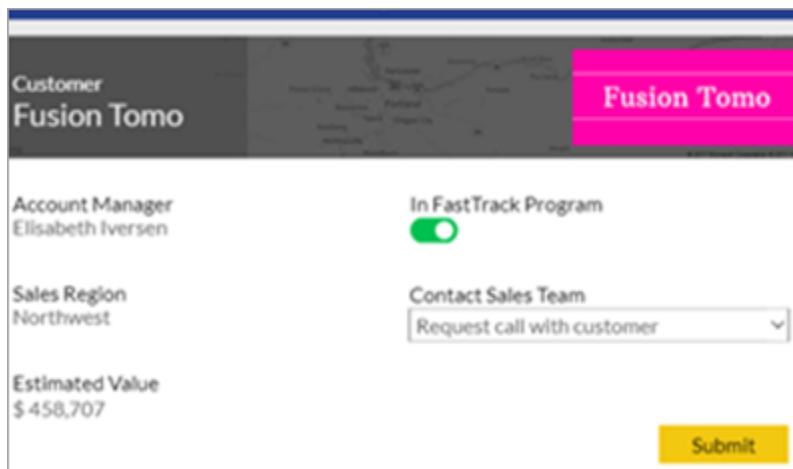
Chain ● Fashions Direct ● Lindseys

Pie charts show the relationship of parts to a whole.

Data Analytics

Student Material

Power Apps visual



Report designers can create a Power App and embed it into a Power BI report as a visual. Consumers can interact with that visual within the Power BI report.

Q&A visual

① Help Q&A understand people's questions better by adding synon... [Add synonyms now](#) X

Ask a question about your data

Try one of these to get started

what is the total store by city

what is the total sales by category

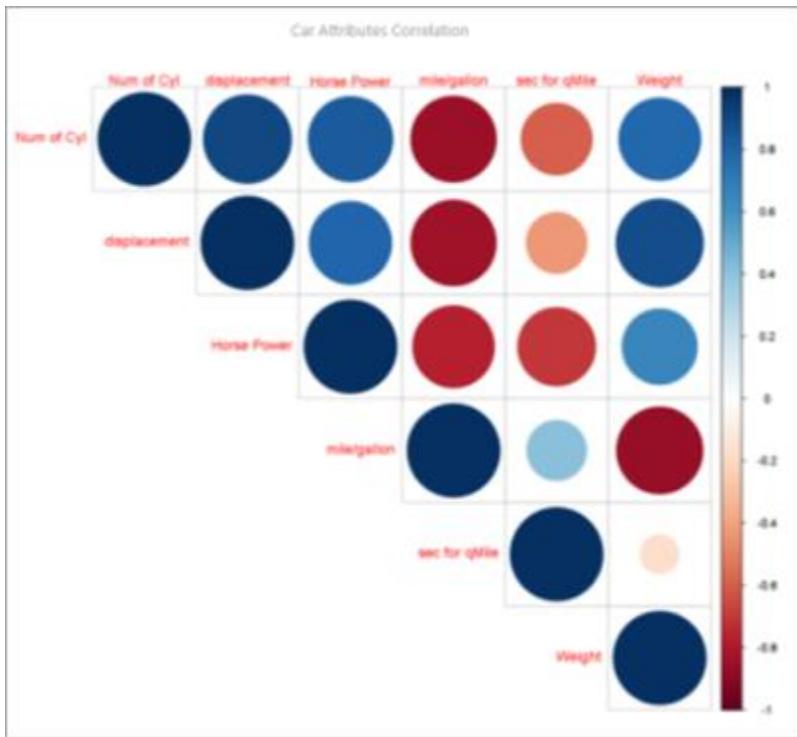
what is the total sales LY by category

Show all suggestions

Data Analytics

Student Material

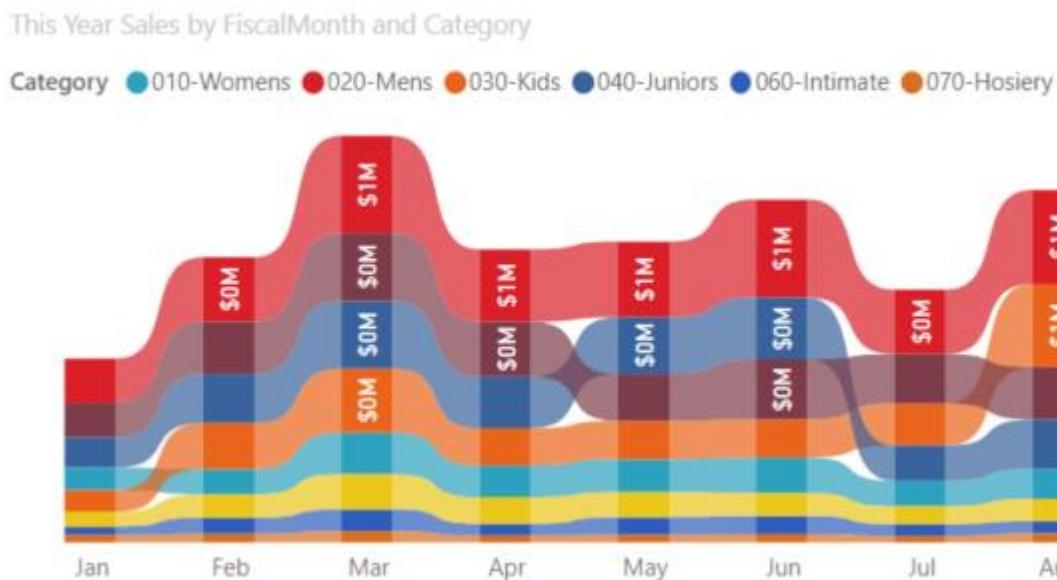
R script visuals



Tip

Visuals created with R scripts, commonly called *R visuals*, can present advanced data shaping and analytics such as forecasting, using the rich analytics and visualization power of R. R visuals can be created in Power BI Desktop and published to the Power BI service.

Ribbon chart

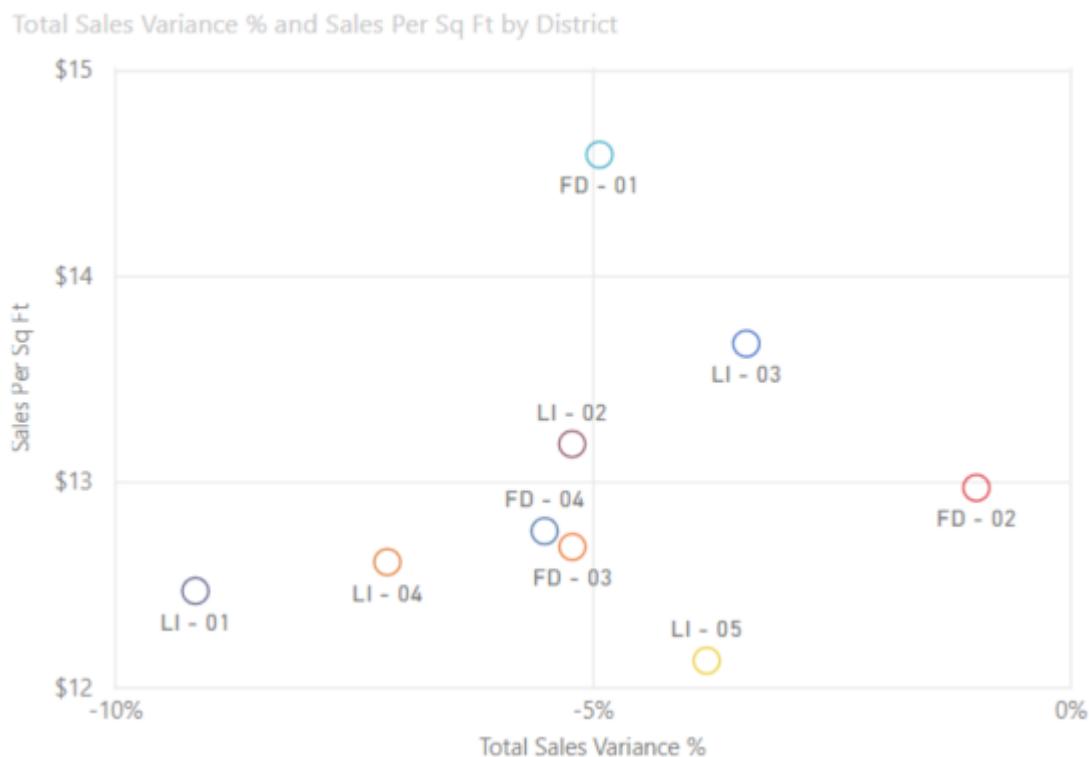


Ribbon charts show which data category has the highest rank (largest value). Ribbon charts are effective at showing rank change, with the highest range (value) always displayed on top for each time period.

Scatter

Scatter, bubble, and dot plot chart

A scatter chart always has two value axes to show one set of numerical data along a horizontal axis and another set of numerical values along a vertical axis. The chart displays points at the intersection of an x and y numerical value, combining these values into single data points. These data points may be distributed evenly or unevenly across the horizontal axis, depending on the data.

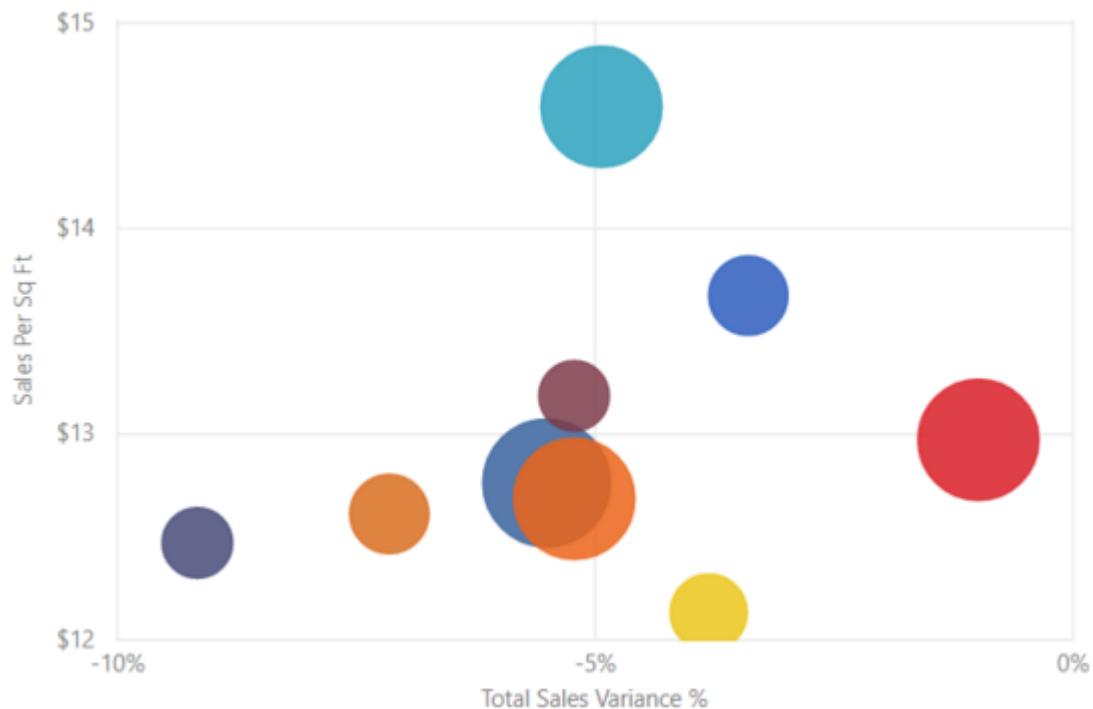


A bubble chart replaces data points with bubbles, with the bubble size representing an additional dimension of the data.

Data Analytics

Student Material

Total Sales Variance %, Sales Per Sq Ft and This Year Sales by District



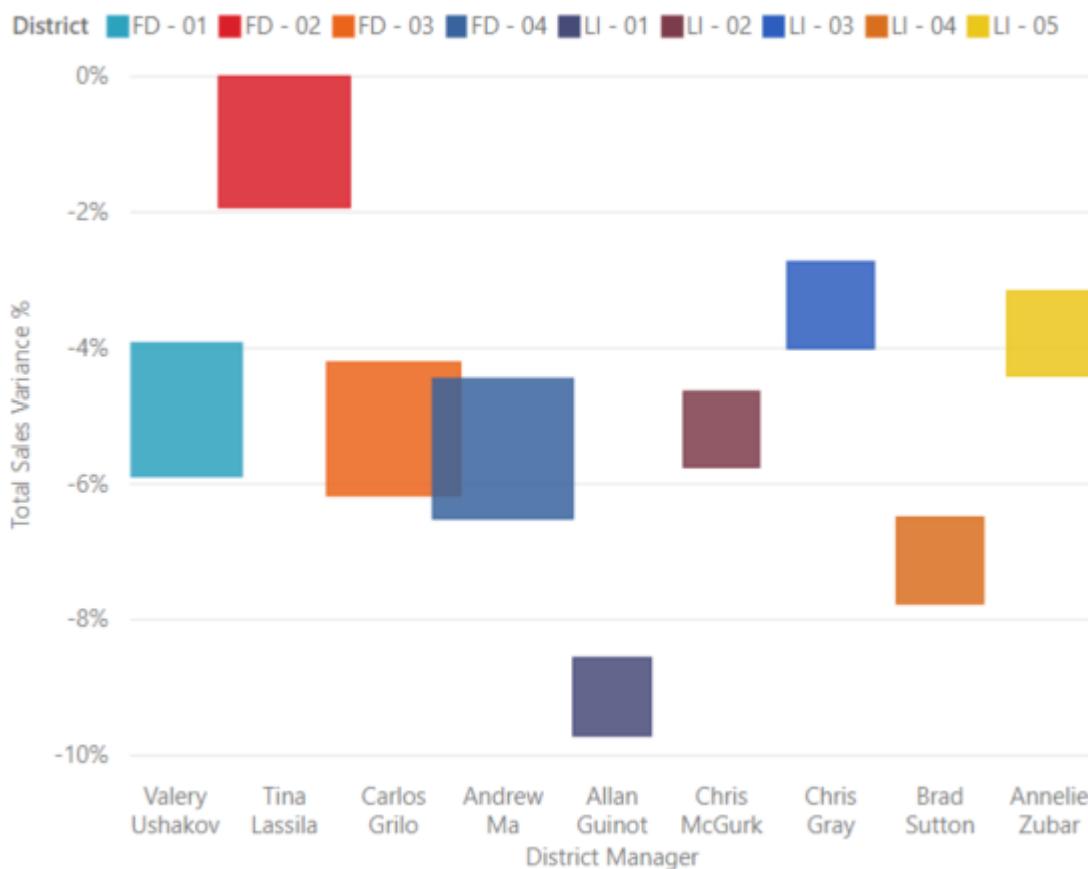
Both scatter and bubble charts can also have a play axis, which can show changes over time.

A dot plot chart is similar to a bubble chart and scatter chart except that it can plot numerical or categorical data along the X axis. This example happens to use squares instead of circles and plots sales along the X axis.

Data Analytics

Student Material

Total Sales Variance % and This Year Sales by District and District Manager



Scatter-high density

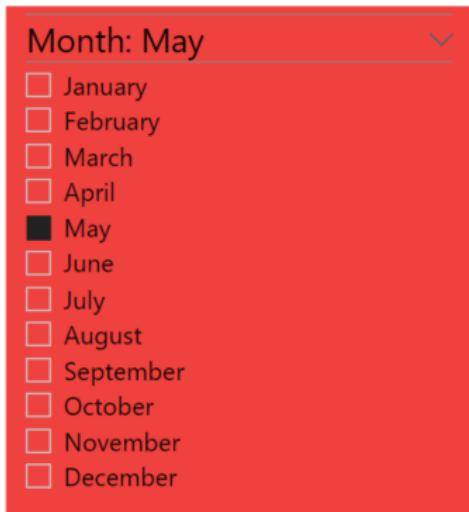


By definition, high-density data is sampled to create visuals reasonably quickly that are responsive to interactivity. High-density sampling uses an algorithm that eliminates

overlapping points, and ensures that all points in the data set are represented in the visual. It doesn't just plot a representative sample of the data.

This ensures the best combination of responsiveness, representation, and clear preservation of important points in the overall data set.

Slicers



A slicer is a standalone chart that can be used to filter the other visuals on the page. Slicers come in many different formats (category, range, date, etc.) and can be formatted to allow selection of only one, many, or all of the available values.

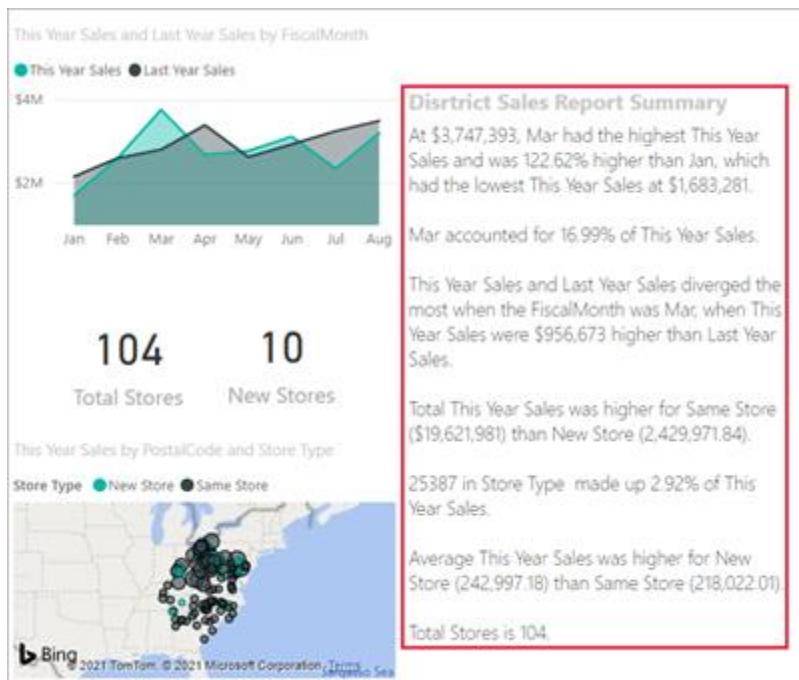
Slicers are a great choice to:

- Display commonly used or important filters on the report canvas for easier access.
- Make it easier to see the current filtered state without having to open a drop-down list.
- Filter by columns that are unneeded and hidden in the data tables.
- Create more focused reports by putting slicers next to important visuals.

Data Analytics

Student Material

Smart narrative



The Smart narrative adds text to reports to point out trends, key takeaways, and add explanations and context. The text helps users to understand the data and identify the important findings quickly.

Standalone images



A standalone image is a graphic that has been added to a report or dashboard.

Tables

Category	This Year Sales Status	Average Unit Price	Last Year Sales	This Year Sales	This Year Sales Goal
080-Accessories	●	\$4.84	\$1,273,096	\$1,379,259	\$1,273,096
090-Home	●	\$3.93	\$2,913,647	\$3,053,326	\$2,913,647
100-Groceries	●	\$1.47	\$810,176	\$829,776	\$810,176
020-Mens	●	\$7.12	\$4,453,133	\$4,452,421	\$4,453,133
030-Kids	●	\$5.30	\$2,726,892	\$2,705,490	\$2,726,892
050-Shoes	●	\$13.84	\$3,640,471	\$3,574,900	\$3,640,471
010-Womens	●	\$7.30	\$2,680,662	\$1,787,958	\$2,680,662
040-Juniors	●	\$7.00	\$3,105,550	\$2,930,385	\$3,105,550
060-Intimate	●	\$4.28	\$955,370	\$852,329	\$955,370
070-Hoslery	●	\$3.69	\$573,604	\$486,106	\$573,604
Total	●	\$5.49	\$23,132,601	\$22,051,952	\$23,132,601

A table is a grid that contains related data in a logical series of rows and columns. It may also contain headers and a row for totals. Tables work well with quantitative comparisons where

Data Analytics

Student Material

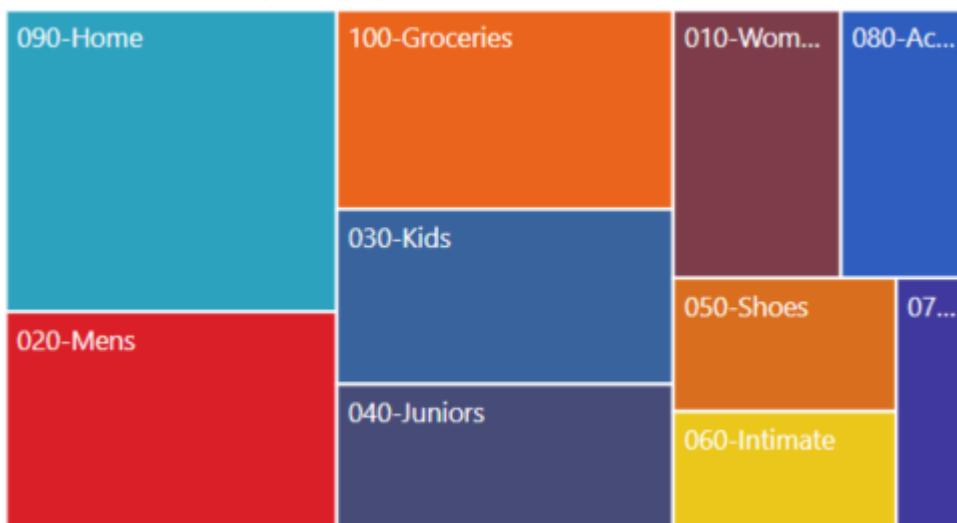
you are looking at many values for a single category. For example, this table displays five different measures for Category.

Tables are a great choice:

- To see and compare detailed data and exact values (instead of visual representations).
- To display data in a tabular format.
- To display numerical data by categories.

Treemaps

Total Units Last Year by Category



Treemaps are charts of colored rectangles, with size representing value. They can be hierarchical, with rectangles nested within the main rectangles. The space inside each rectangle is allocated based on the value being measured. And the rectangles are arranged in size from top left (largest) to bottom right (smallest).

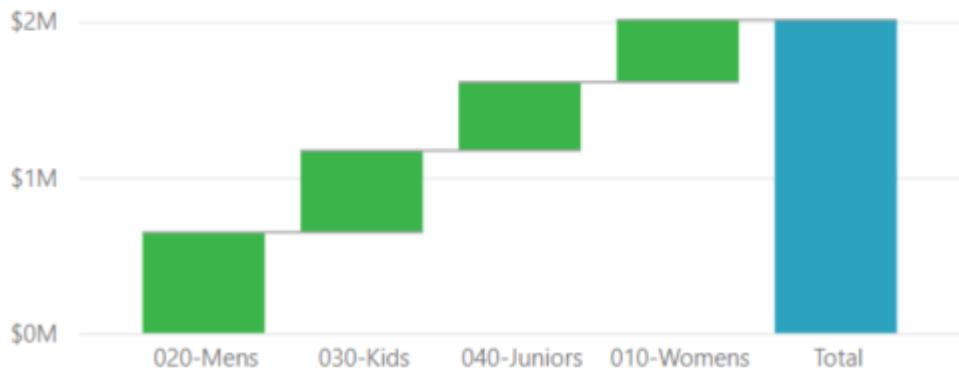
Treemaps are a great choice:

- To display large amounts of hierarchical data.
- When a bar chart can't effectively handle the large number of values.
- To show the proportions between each part and the whole.
- To show the pattern of the distribution of the measure across each level of categories in the hierarchy.
- To show attributes using size and color coding.
- To spot patterns, outliers, most-important contributors, and exceptions.

Waterfall charts

Total Units Last Year by Category

● Increase ● Decrease ● Total



A waterfall chart shows a running total as values are added or subtracted. It's useful for understanding how an initial value (for example, net income) is affected by a series of positive and negative changes.

The columns are color coded so you can quickly tell increases and decreases. The initial and the final value columns often start on the horizontal axis, while the intermediate values are floating columns. Because of this "look", waterfall charts are also called bridge charts.

Waterfall charts are a great choice:

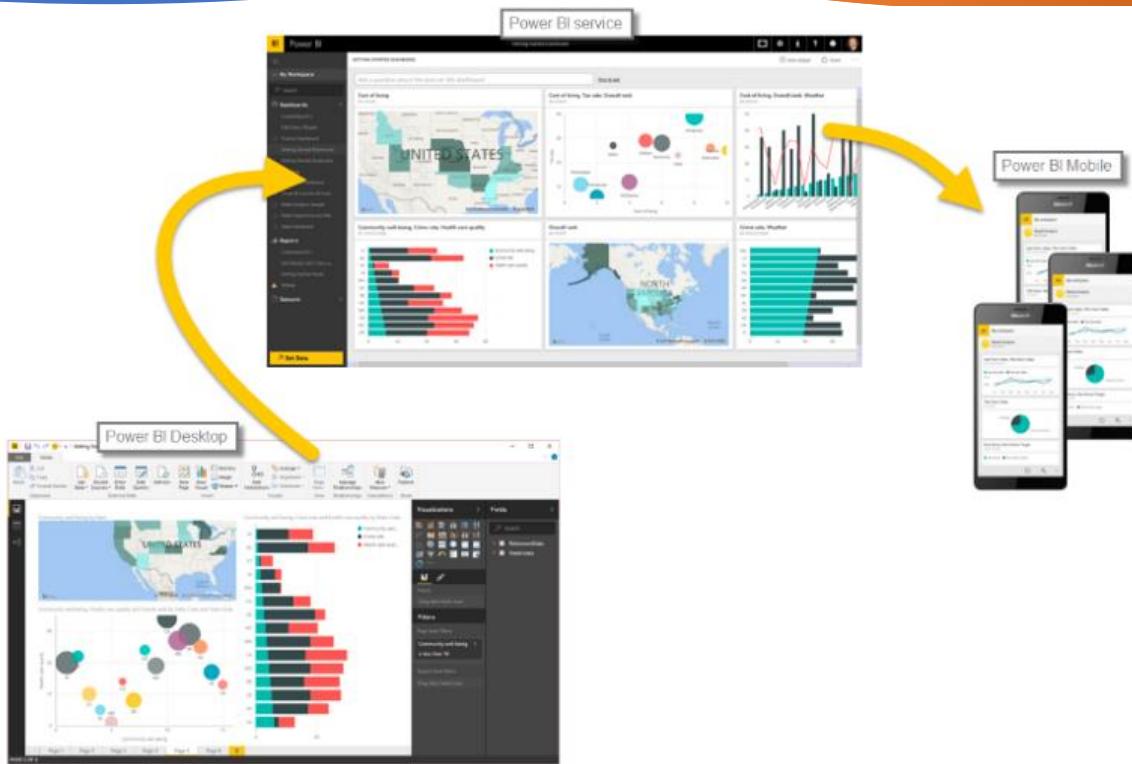
- When you have changes for the measure across time or across different categories.
- To audit the major changes contributing to the total value.
- To plot your company's annual profit by showing various sources of revenue and arrive at the total profit (or loss).
- To illustrate the beginning and the ending headcount for your company in a year.
- To visualize how much money you make and spend each month, and the running balance for your account.

5.4 Getting Started with Power BI Desktop:

Power BI is a free application that can be downloaded and installed on the system. It can be connected to multiple data sources. Usually, an analysis work begins in Power BI Desktop where report creation takes place. The report is then published to Power BI service from where it can be shared to the Power BI Mobile apps so that people can view the reports even on mobiles.

Data Analytics

Student Material



The most common uses for Power BI Desktop are as follows:

- Connect to data
- Transform and clean that data, to create a data model
- Create visuals, such as charts or graphs, that provide visual representations of the data
- Create reports that are collections of visuals, on one or more report pages
- Share reports with others by using the Power BI service

There are three views available in Power BI Desktop, which you select on the left side of the canvas.

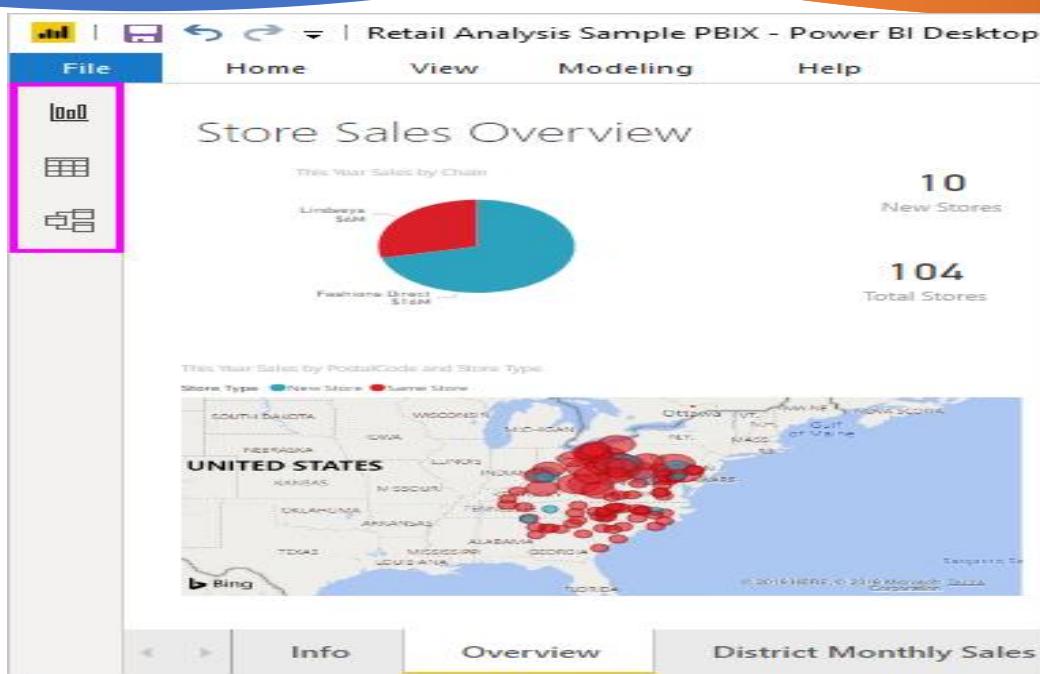
The views, shown in the order they appear, are as follows:

- **Report:** In this view, you create reports and visuals, where most of your creation time is spent.
- **Data:** In this view, you see the tables, measures, and other data used in the data model associated with your report and transform the data for best use in the report's model.
- **Model:** In this view, you see and manage the relationships among tables in your data model.

The following image shows the three views, as displayed along the left side of the canvas:

Data Analytics

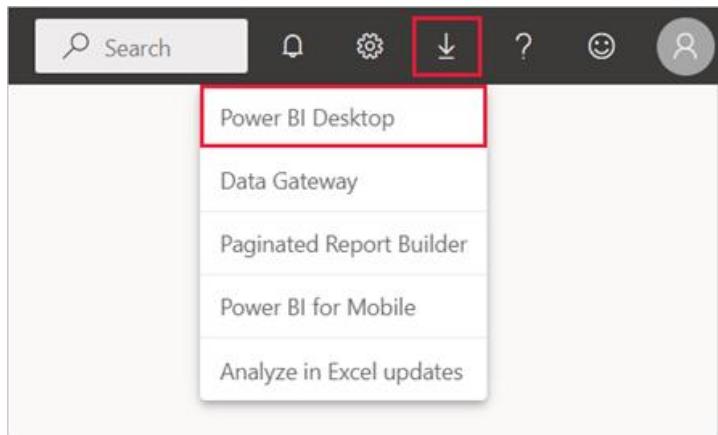
Student Material



Install and run Power BI Desktop

To download Power BI Desktop, go to the Power BI Desktop download page and select Download Free. Or for download options, select See download or language options.

You can also download Power BI Desktop from the Power BI service. Select the Download icon in the top menu bar, and then select Power BI Desktop.



On the Microsoft Store page, select Get, and follow the prompts to install Power BI Desktop on your computer. Start Power BI Desktop from the Windows Start menu or from the icon in the Windows taskbar.

The first time Power BI Desktop starts, it displays the Welcome screen.

From the Welcome screen, you can Get data, see Recent sources, open recent reports, Open other reports, or select other links. Select the close icon to close the Welcome screen.

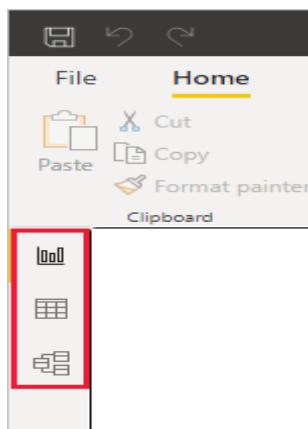
Data Analytics

Student Material

The screenshot shows the Power BI Desktop application window. On the left, there's a sidebar with three icons: Report (highlighted with a red box), Data, and Model. Below these are sections for 'Recent sources' and 'Recent reports'. The main area has a large circular placeholder icon. To the right, there's a yellow sidebar with sections for 'WHAT'S NEW', 'POWER BI BLOG', 'FORUMS', and 'TUTORIALS'. At the bottom of the main area, there's a 'Get started' button and a link to buy a license.

Along the left side of Power BI Desktop are icons for the three Power BI Desktop views: Report, Data, and Model, from top to bottom. The current view is indicated by the yellow bar along the left, and you can change views by selecting any of the icons.

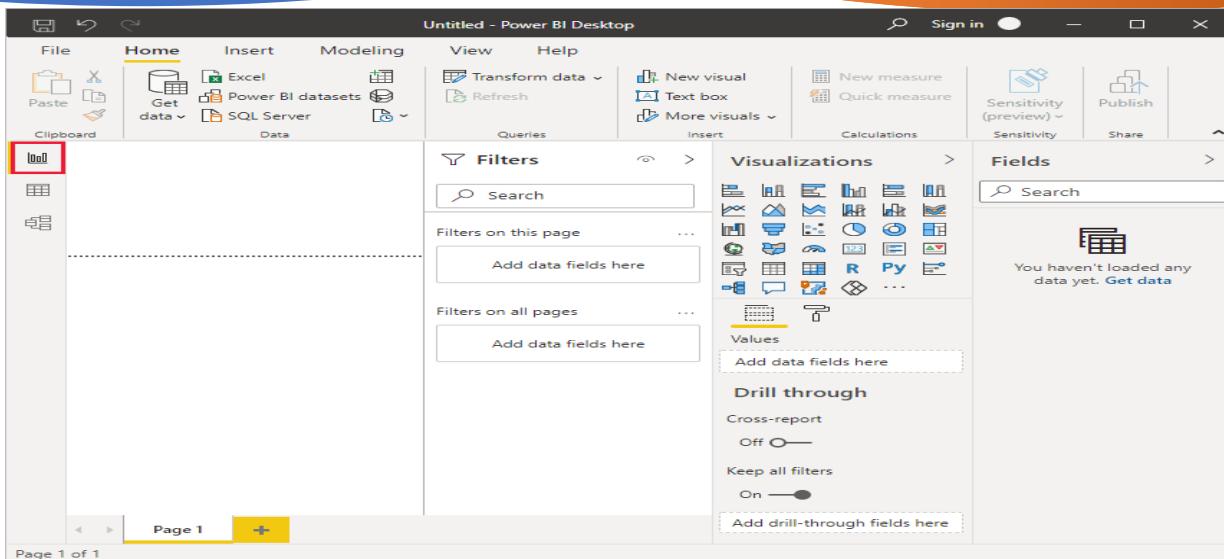
If you are using keyboard navigation, press Ctrl + F6 to move focus to that section of buttons in the window.



Report view is the default view.

Data Analytics

Student Material



Power BI Desktop also includes the Power Query Editor, which opens in a separate window. In Power Query Editor, you can build queries and transform data, then load the refined data model into Power BI Desktop to create reports.

Connect to data

To get started with Power BI Desktop, the first step is to connect to data. There are many different data sources you can connect to from Power BI Desktop.

1. To connect to data:

From the Home ribbon, select Get Data > More.

The Get Data window appears, showing the many categories to which Power BI Desktop can connect.

Get Data

Search

All

File

Database

Power Platform

Azure

Online Services

Other

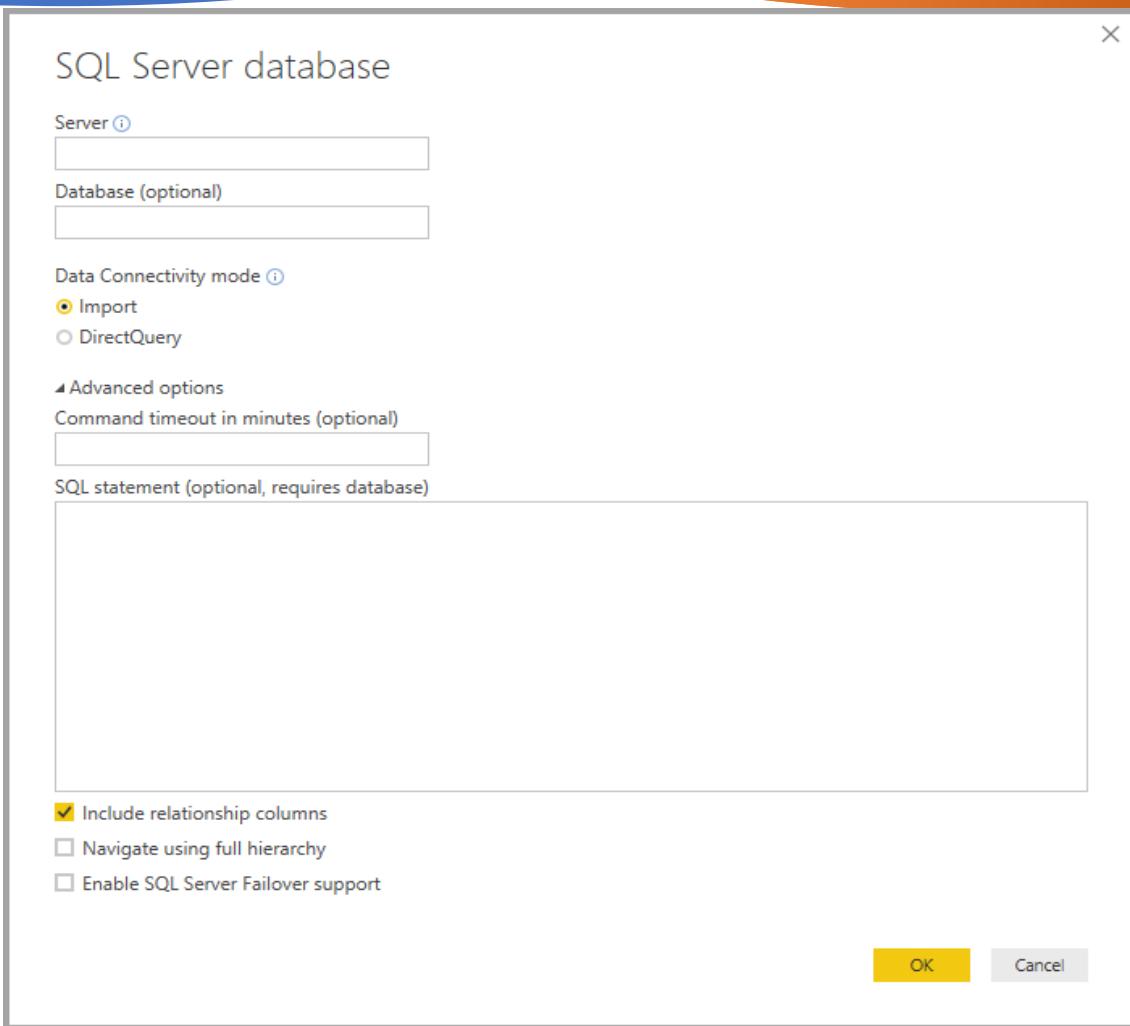
Online Services

- SharePoint Online List
- Microsoft Exchange Online
- Dynamics 365 (online)
- Dynamics NAV
- Dynamics 365 Business Central
- Dynamics 365 Business Central (on-premises)
- Microsoft Azure Consumption Insights (Beta)
- Azure DevOps (Beta)
- Azure DevOps Server (Beta)
- Salesforce Objects
- Salesforce Reports
- Google Analytics
- Adobe Analytics
- appFigures (Beta)
- Data.World - Get Dataset (Beta)
- Facebook

Certified Connectors

Connect Cancel

- When you select a data type, you're prompted for information, such as the URL and credentials, necessary for Power BI Desktop to connect to the data source on your behalf.



3. After you connect to one or more data sources, you may want to transform the data so it's useful for you.

5.5 Transform Data:

Transform and clean data, create a model

In Power BI Desktop, you can clean and transform data using the built-in Power Query Editor. With Power Query Editor, you make changes to your data, such as changing a data type, removing columns, or combining data from multiple sources. It's like sculpting: you start with a large block of clay (or data), then shave off pieces or add others as needed, until the shape of the data is how you want it.

To start Power Query Editor:

On the Home ribbon, in the Queries section, select Transform data.

Data Analytics

Student Material

The Power Query Editor window appears.

The screenshot shows the Microsoft Power Query Editor interface. The title bar reads "Untitled - Power Query Editor". The ribbon menu includes File, Home, Transform, Add Column, View, Tools, and Help. The Home tab is selected, showing various icons for operations like Close & Apply, New Query, Data Sources, Parameters, and Manage. The main area displays a table titled "Ranking of best and worst states for retire" with 15 rows. The columns are labeled "Column1" through "Column5". The "APPLIED STEPS" pane on the right lists the steps taken: "Source", "Extracted Table From Html", and "Changed Type".

	Column1	Column2	Column3	Column4	Column5
1	State	Overall rank	Affordability	Crime	Culture
2	Source: Bankrate's 2019 "Bes...				
3	Nebraska	1	14	19	21
4	Iowa	2	8	15	20
5	Missouri	3	1	42	33
6	South Dakota	4	17	23	12
7	Florida	5	25	29	13
8	Kentucky	6	9	9	46
9	Kansas	7	7	39	37
10	North Carolina	7	13	28	28
11	Montana	9	16	31	2
12	Hawaii	10	45	24	9
13	Arkansas	11	4	46	39
14	Wisconsin	12	20	15	17
15	North Dakota	13	22	17	26
16					

Each step you take in transforming data (such as renaming a table, transforming a data type, or deleting a column) is recorded by Power Query Editor. Every time this query connects to the data source, those steps are carried out so that the data is always shaped the way you specify.

The following image shows the **Power Query Editor** window for a query that has been shaped, and turned into a model.

Data Analytics

Student Material

Untitled - Power Query Editor

File Home Transform Add Column View Tools Help

CLOSE & APPLY NEW SOURCE SOURCES ENTER DATA DATA SOURCE SETTINGS MANAGE PARAMETERS REFRESH PREVIEW ADVANCED EDITOR PROPERTIES MANAGE QUERY MANAGE COLUMNS MANAGE ROWS SORT TRANSFORM

Queries [1]

	A _B State	Overall rank	A _B Affordability	A _B Weather
1	Hawaii	10	45	1
2	Florida	5	25	2
3	Louisiana	36	29	3
4	Texas	17	24	4
5	Georgia	28	19	5
6	Mississippi	19	6	6
7	Alabama	16	10	7
8	South Carolina	41	27	8
9	Arkansas	11	4	9
10	Arizona	38	33	10
11	Oklahoma	21	11	11
12	North Carolina	7	13	12
13	California	43	49	13
14	Tennessee	22	12	14
15	Kentucky	6	9	15
16	Delaware	32	30	16
17	Virginia	39	32	17
18	Maryland	50	47	18
19	Missouri	3	1	19
20	Kansas	7	7	20

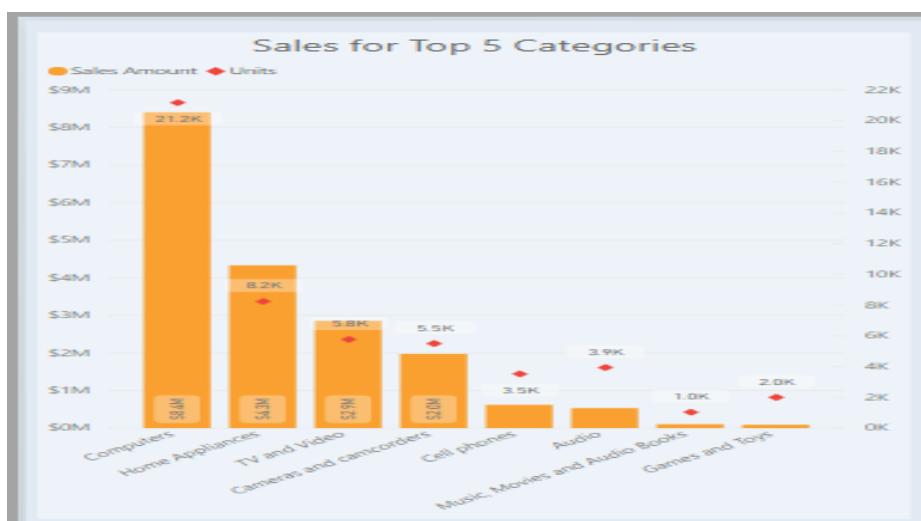
4 COLUMNS, 40 ROWS Column profiling based on top 1000 rows

PREVIEW DOWNLOADED AT 12:22 PM

Once your data is how you want it, you can create visuals.

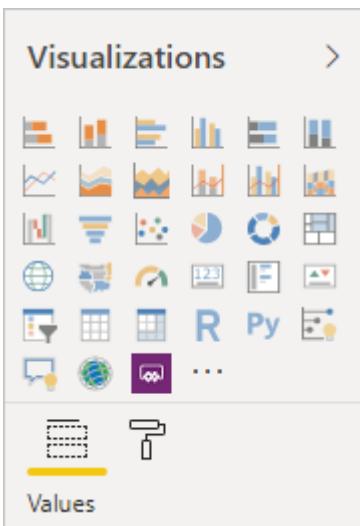
Create visuals

After you have a data model, you can drag fields onto the report canvas to create visuals. A visual is a graphic representation of the data in your model. There are many different types of visuals to choose from in Power BI Desktop. The following visual shows a simple column chart.



To create or change a visual:

- From the **Visualizations** pane, select the visual icon.



If you already have a visual selected on the report canvas, the selected visual changes to the type you selected.

If no visual is selected on the canvas, a new visual is created based on your selection.

5.6 Report Creation:

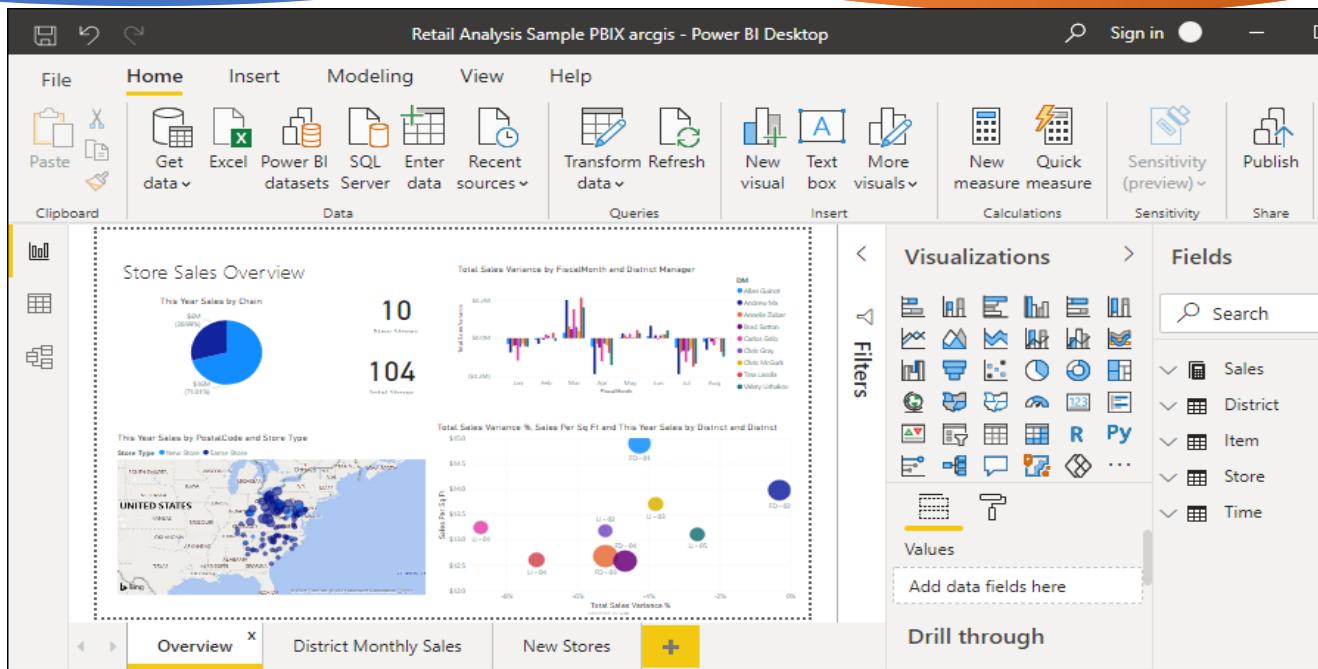
More often, you'll want to create a collection of visuals that show various aspects of the data you've used to create your model in Power BI Desktop. A collection of visuals, in one Power BI Desktop file, is called a *report*. A report can have one or more pages, just like an Excel file can have one or more worksheets.

With Power BI Desktop you can create complex and visually rich reports, using data from multiple sources, all in one report that you can share with others in your organization.

In the following image, you see the first page of a Power BI Desktop report, named Overview, as seen on the tab near the bottom of the image.

Data Analytics

Student Material

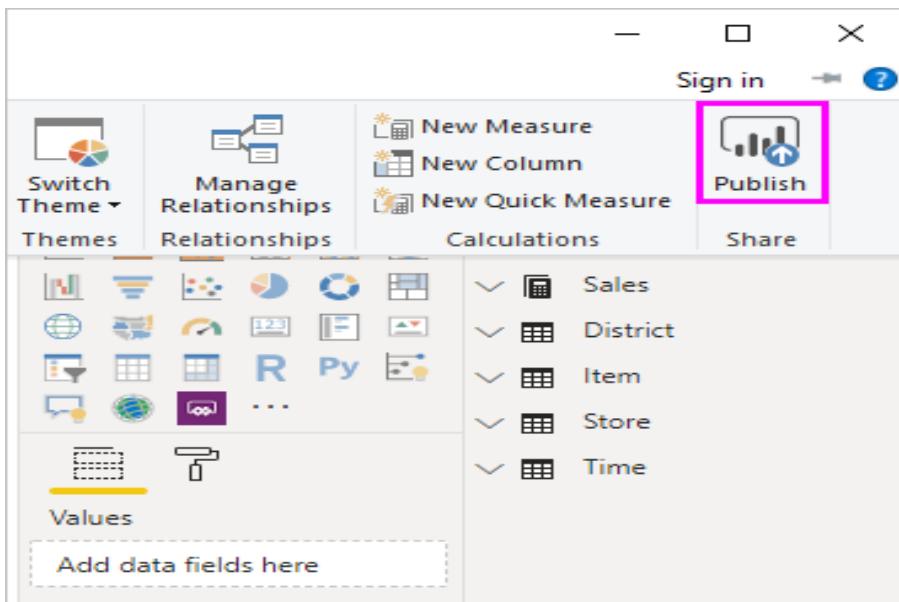


Share reports

After a report is ready to share with others, you can *publish* the report to the Power BI service, and make it available to anyone in your organization who has a Power BI license.

To publish a Power BI Desktop report:

1. Select **Publish** from the **Home** ribbon.



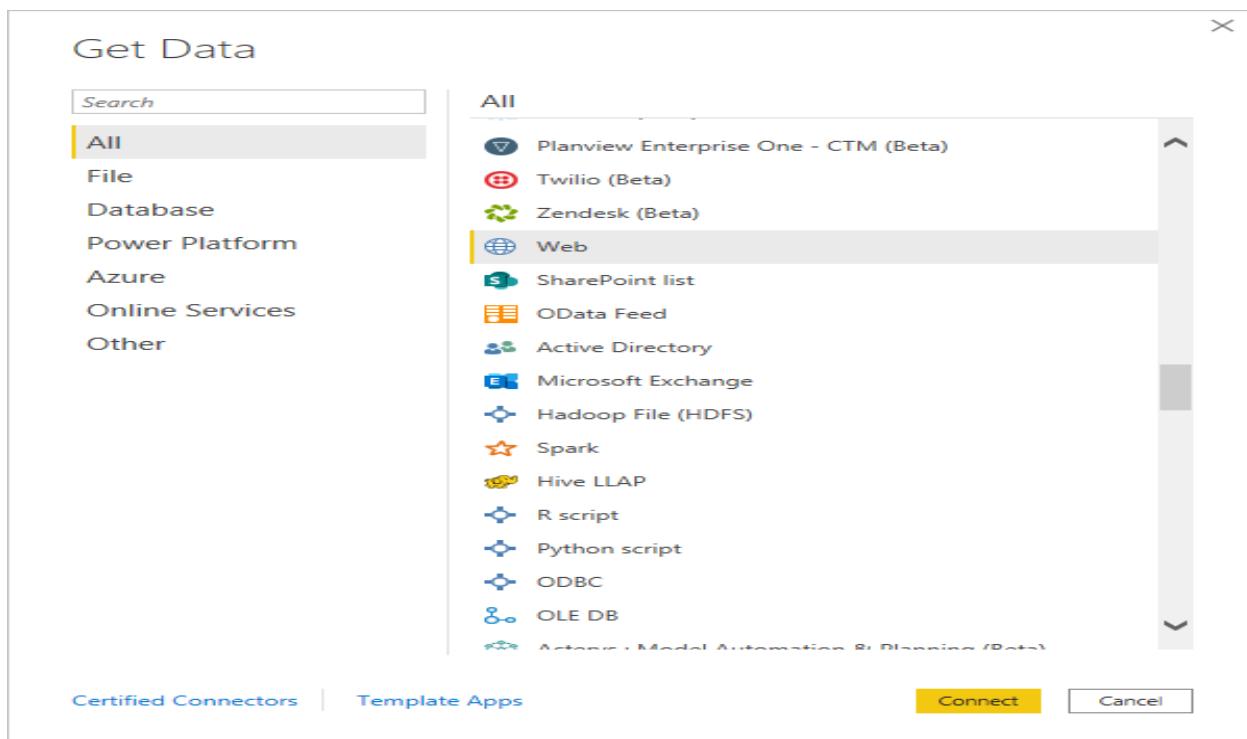
Power BI Desktop connects you to the Power BI service with your Power BI account.

- Power BI prompts you to select where in the Power BI service you'd like to share the report, such as your workspace, a team workspace, or some other location in the Power BI service.

You must have a Power BI license to share reports to the Power BI service.

Connect to data

With Power BI Desktop installed, you're ready to connect to the ever-expanding world of data. To see the many types of data sources available, select Get Data > More in the Power BI Desktop Home tab, and in the Get Data window, scroll through the list of All data sources. In this quick tour, you connect to a couple of different Web data sources.

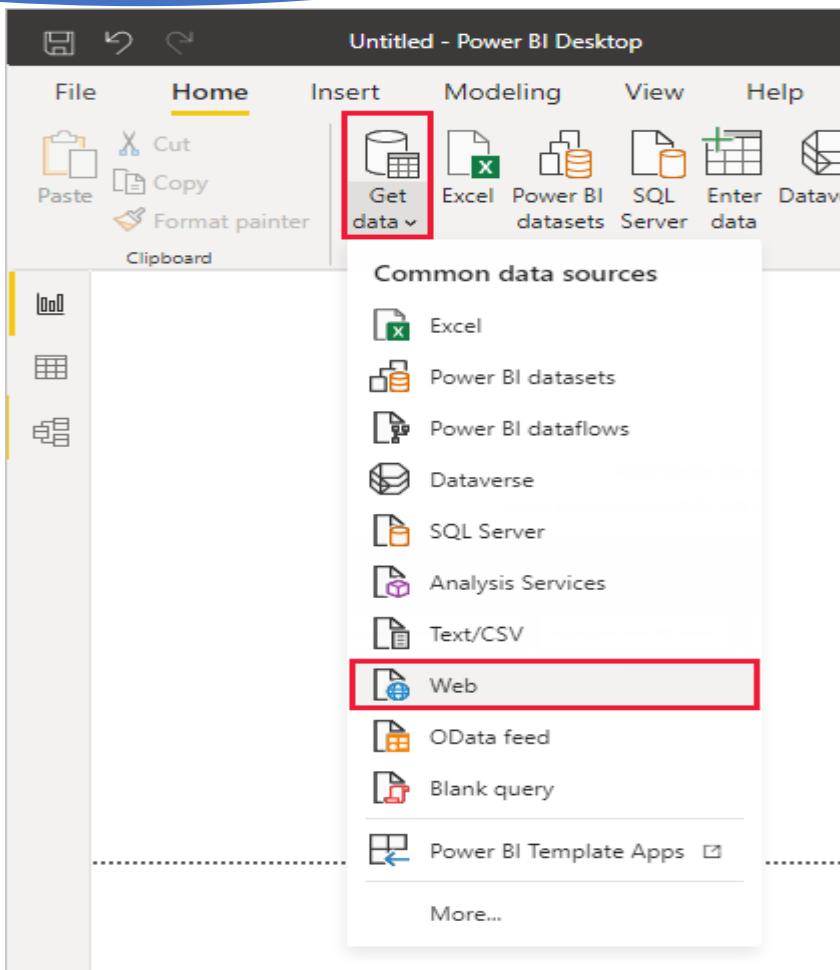


Imagine you're a data analyst working for a sunglasses retailer. You want to help your client target sunglasses sales where the sun shines most frequently. The Bankrate.com Best and worst states for retirement page has interesting data on this subject.

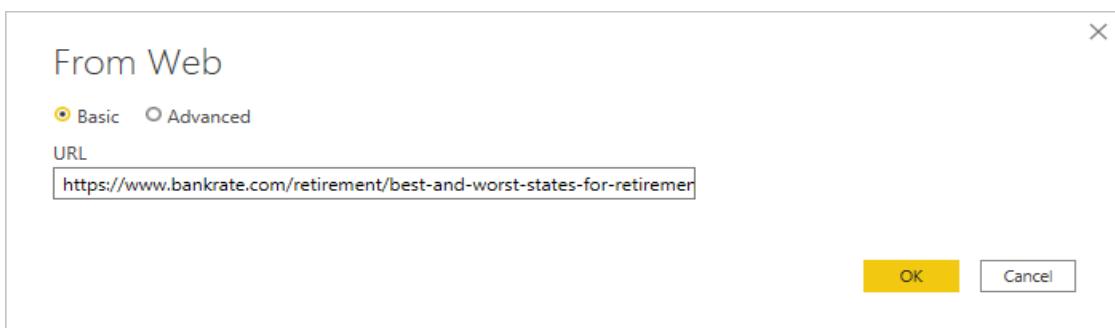
On the Power BI Desktop Home tab, select Get Data > Web to connect to a web data source.

Data Analytics

Student Material



In the From Web dialog box, paste the address <https://www.bankrate.com/retirement/best-and-worst-states-for-retirement/> into the URL field, and select OK.



If prompted, on the Access Web Content screen, select Connect to use anonymous access.

The query functionality of Power BI Desktop goes to work and contacts the web resource. The Navigator window returns what it found on the web page, in this case an HTML table called Ranking of best and worst states for retirement, and five other suggested tables. You're interested in the HTML table, so select it to see a preview.

Data Analytics

Student Material

At this point you can select Load to load the table, or Transform data to make changes in the table before you load it.

The screenshot shows the Microsoft Power BI Navigator interface. On the left, there's a sidebar with a search bar and a 'Display Options' dropdown. Below that are two sections: 'HTML Tables [1]' containing one item ('Ranking of best and worst states for retire...') and 'Suggested Tables [5]' containing five items labeled 'Table 1' through 'Table 5'. The main area is titled 'Ranking of best and worst states for retirement' and displays a table with two columns: 'State' and 'Overall rank'. The table lists 19 US states with their respective ranks. At the bottom of the interface are three buttons: 'Add Table Using Examples', 'Load' (highlighted in yellow), 'Transform Data' (with a red box around it), and 'Cancel'.

Column1	Column2
State	Overall rank
Source: Bankrate's 2019 "Best and worst states for retirement" study	Source: Bankrate'
Nebraska	1
Iowa	2
Missouri	3
South Dakota	4
Florida	5
Kentucky	6
Kansas	7
North Carolina	7
Montana	9
Hawaii	10
Arkansas	11
Wisconsin	12
North Dakota	13
Vermont	14
New Hampshire	15
Alabama	16
Texas	17
Idaho	18
Mississippi	19

When you select Transform data, Power Query Editor launches, with a representative view of the table. The Query Settings pane is on the right, or you can always show it by selecting Query Settings on the View tab of Power Query Editor.

Data Analytics

Student Material

The screenshot shows the Power Query Editor interface. The main area displays a table titled "Ranking of best and worst states for retire". The table has five columns: "State", "Overall rank", "Affordability", "Crime", and "Culture". The rows list states from 1 (Nebraska) to 15 (North Dakota). The "Applied Steps" pane on the right shows a single step: "Changed Type".

State	Overall rank	Affordability	Crime	Culture
1 Nebraska	1	14	19	21
2 Iowa	2	8	15	20
3 Missouri	3	1	42	33
4 South Dakota	4	17	23	12
5 Florida	5	25	29	13
8 Kentucky	6	9	9	46
9 Kansas	7	7	39	37
10 North Carolina	7	13	28	28
11 Montana	9	16	31	2
12 Hawaii	10	45	24	9
13 Arkansas	11	4	46	39
14 Wisconsin	12	20	15	17
15 North Dakota	13	22	17	26

Shape data

Shaping can mean transforming the data, such as renaming columns or tables, removing rows or columns, or changing data types. Power Query Editor captures these steps sequentially under Applied Steps in the Query Settings pane. Each time this query connects to the data source, those steps are carried out, so the data is always shaped the way you specify. This process occurs when you use the query in Power BI Desktop, or when anyone uses your shared query, such as in the Power BI service.

The screenshot shows the "Query Settings" dialog box. The "Properties" section shows the name "Ranking of best and worst states for retire". The "Applied Steps" section shows three steps: "Source", "Extracted Table From Html", and "Changed Type".

Data Analytics

Student Material

If you need to change a data type, select the column or columns to change. Hold down the Shift key to select several adjacent columns, or Ctrl to select non-adjacent columns. Either right-click a column header, select Change Type, and choose a new data type from the menu, or drop down the list next to Data Type in the Transform group of the Home tab, and select a new data type.

The screenshot shows the Power Query Editor interface. A context menu is open over 'Column1'. The 'Data Type' dropdown in the ribbon is set to 'Text'. The 'Change Type' option in the context menu is highlighted with a red box. The 'Text' option under 'Change Type' is also highlighted with a red box. The menu includes options like Copy, Remove, Duplicate Column, Add Column From Examples..., Remove Duplicates, Remove Errors, Change Type, Transform, Replace Values..., Replace Errors..., Split Column, Group By..., Fill, Unpivot Columns, Unpivot Other Columns, Unpivot Only Selected Columns, Rename..., Move, Drill Down, and Add as New Query. A preview pane on the right shows the data with the 'City' and 'Crime' columns.

The Power Query Editor in Power BI Desktop uses the ribbon or the right-click menus for available tasks. Most of the tasks you can select on the Home or Transform tabs of the ribbon are also available by right-clicking an item and choosing from the menu that appears.

For example, for sunglasses sales you're most interested in the weather ranking, so you decide to sort the table by the Weather column instead of by Overall rank. Drop down the arrow next to the Weather header, and select Sort ascending. The data now appears sorted by weather ranking, and the step Sorted Rows appears in Applied Steps.

Data Analytics

Student Material

The screenshot shows the Power BI Data View interface. On the left, there is a table titled "Weather" with columns "State" and "Score". The table has 14 rows, numbered 1 to 14. The "Score" column contains values such as 10, 5, 36, 17, 28, 19, and 16. To the right of the table is the "Query Settings" pane. Under the "APPLIED STEPS" section, the steps listed are: Source, Extracted Table From Html, Changed Type, Promoted Headers, Changed Type1, Removed Top Rows, and Changed Type2. A yellow box highlights the "Sorted Rows" step.

You're not very interested in selling sunglasses to the worst weather states, so you decide to remove them from the table. From the Home tab, select Reduce Rows > Remove Rows > Remove Bottom Rows. In the Remove Bottom Rows dialog box, enter 10, and then select OK.

The screenshot shows the Power Query Editor interface. The main area displays a table with columns "State", "Overall rank", and "Affordability". The rows show data for Hawaii, Florida, Louisiana, Texas, Georgia, Mississippi, and Alabama. On the ribbon, the "Transform" tab is selected. In the "Reduce Rows" dropdown menu, the "Remove Bottom Rows" option is highlighted with a red box. Below the table, a "Remove Bottom Rows" dialog box is open. It asks "Specify how many rows to remove from the bottom." A text input field contains the value "10", which is also highlighted with a red box. At the bottom of the dialog box are "OK" and "Cancel" buttons, with "OK" being highlighted with a yellow box.

Data Analytics

Student Material

The bottom 10 worst weather rows are removed from the table, and the step Removed Bottom Rows appears in Applied Steps.

You decide the table has too much extra information for your needs, and to remove the Affordability, Crime, Culture, and Wellness columns. Select the header of each column that you want to remove. Hold down the Shift key to select several adjacent columns, or Ctrl to select non-adjacent columns.

Then, from the Manage Columns group of the Home tab, select Remove Columns. You can also right-click one of the selected column headers and select Remove Columns from the menu. The selected columns are removed, and the step Removed Columns appears in Applied Steps.

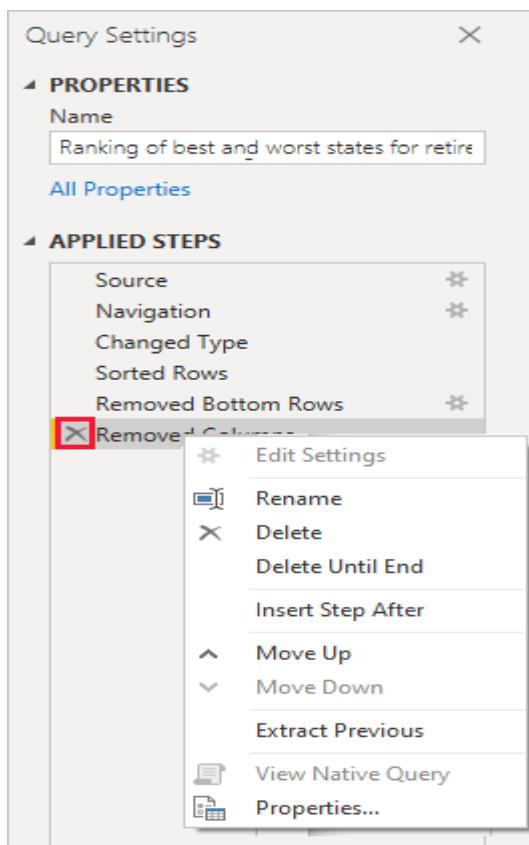
The screenshot shows the Power Query Editor interface. In the ribbon, the 'Home' tab is selected. Under the 'Transform' section, the 'Manage Columns' button is highlighted with a red box. A context menu is open over the 'Culture' column header, with the 'Remove Columns' option highlighted with a red box. The 'Remove Columns' option is also listed in the main 'Transform' dropdown menu on the right, which is also highlighted with a red box. The main area shows a table with columns: 'Affordability', 'Crime', 'Culture', 'Weather', and 'Wellness'. The 'Culture' column is currently selected. The status bar at the bottom left indicates '7 COLUMNS, 40 ROWS' and 'Column profiling based on top 1000 rows'. The status bar at the bottom right indicates 'PREVIEW DOWNLOADED AT 12:22 PM'.

On second thought, Affordability might be relevant to sunglasses sales after all. You'd like to get that column back. You can easily undo the last step in the Applied Steps pane by selecting the X delete icon next to the step. Now redo the step, selecting only the columns you want to delete. For more flexibility, you could delete each column as a separate step.

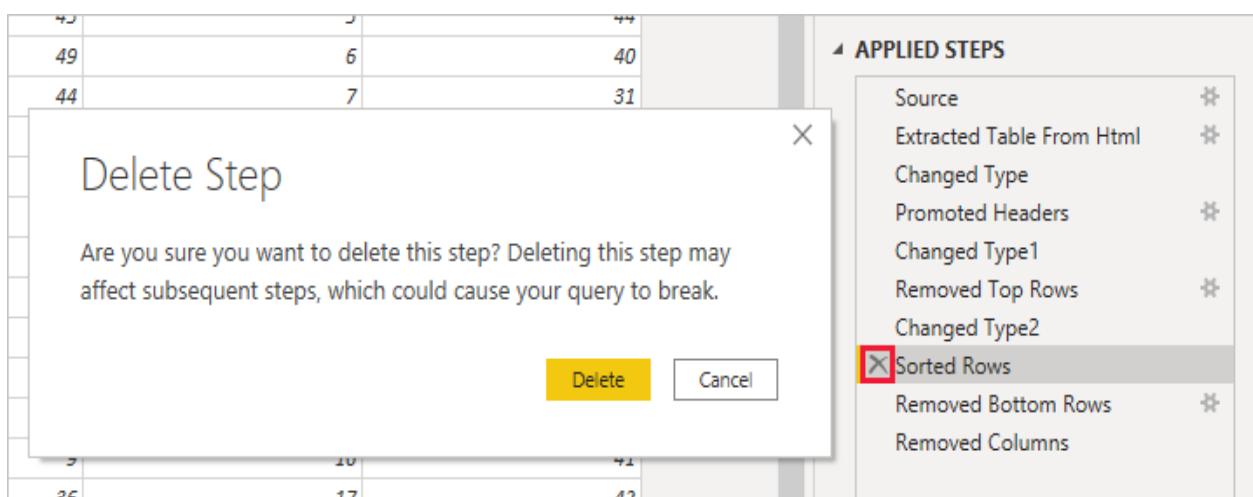
Data Analytics

Student Material

You can right-click any step in the Applied Steps pane and choose to delete it, rename it, move it up or down in the sequence, or add or delete steps after it. For intermediate steps, Power BI Desktop will warn you if the change could affect later steps and break your query.



For example, if you no longer wanted to sort the table by **Weather**, you might try to delete the **Sorted Rows** step. Power BI Desktop warns you that deleting this step could cause your query to break. You removed the bottom 10 rows after you sorted by weather, so if you remove the sort, different rows will be removed. You also get a warning if you select the **Sorted Rows** step and try to add a new intermediate step at that point.



Data Analytics

Student Material

Finally, you change the table title to be about sunglass sales instead of retirement. Under Properties in the Query Settings pane, replace the old title with Best states for sunglass sales.

The finished query for your shaped data looks like this:

The screenshot shows the Microsoft Power Query Editor interface. The main area displays a table with four columns: State, Overall rank, Affordability, and Weather. The rows list various US states with their respective values. The 'State' column contains abbreviations like 'Hawaii', 'Florida', 'Louisiana', etc. The 'Overall rank' column ranges from 1 to 50. The 'Affordability' and 'Weather' columns range from 1 to 20. The 'Properties' pane on the right shows the table is named 'Best states for sunglass sales'. The 'Applied Steps' pane lists the transformation steps: Source, Extracted Table From Html, Changed Type, Promoted Headers, Changed Type1, Removed Top Rows, Changed Type2, Sorted Rows, Removed Bottom Rows, and Removed Columns (which is currently selected).

State	Overall rank	Affordability	Weather
Hawaii	10	45	1
Florida	5	25	2
Louisiana	36	29	3
Texas	17	24	4
Georgia	28	19	5
Mississippi	19	6	6
Alabama	16	10	7
South Carolina	41	27	8
Arkansas	11	4	9
Arizona	38	33	10
Oklahoma	21	11	11
North Carolina	7	13	12
California	43	49	13
Tennessee	22	12	14
Kentucky	6	9	15
Delaware	32	30	16
Virginia	39	32	17
Maryland	50	47	18
Missouri	3	1	19
Kansas	7	7	20

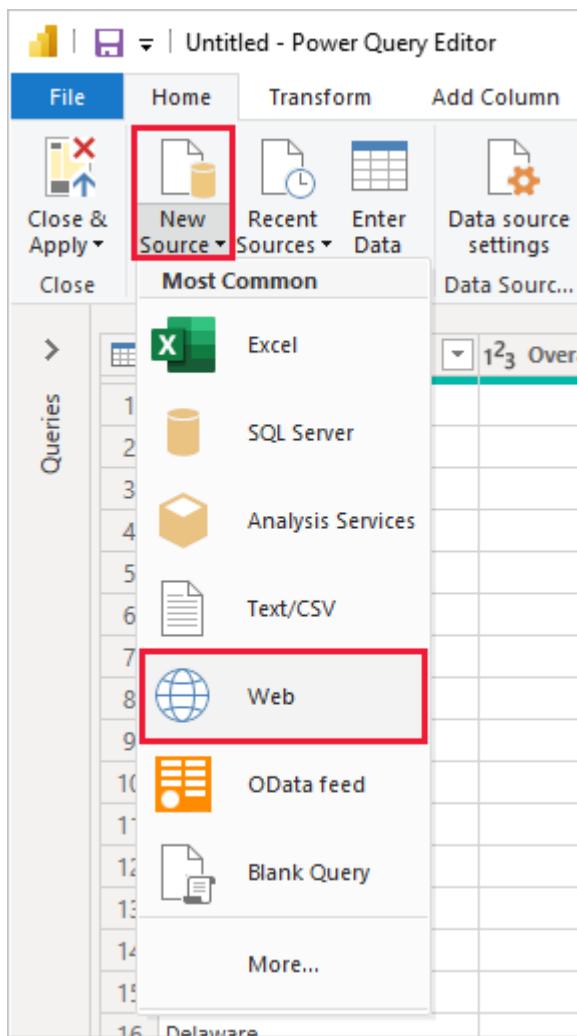
Combine data

The data about various states is interesting, and will be useful for building additional analysis efforts and queries. But there's one problem: most data out there uses two-letter abbreviations for state codes, not the full names of the states. To use that data, you need some way to associate your state names with their abbreviations.

Data Analytics

Student Material

To import the state abbreviations data into Power Query Editor, select New Source > Web from the New Query group on the Home tab of the ribbon.



In the Navigator window, select the table Codes and abbreviations for U.S. states, federal district, territories, and other regions, and then select OK. The table opens in Power Query Editor.

Remove all columns except for Name and status of region, Name and status of region, and ANSI. To keep only these columns, hold down Ctrl and select the columns. Then, either right-click one of the column headers and select Remove Other Columns, or, from the Manage Columns group of the Home tab, select Remove Other Columns.

Drop down the arrow next to the Name and status of region_1 column header, and select Filters > Equals. In the Filter Rows dialog box, drop down the Enter or select a value field next to equals and select State. Select OK.

Data Analytics

Student Material

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'Name and status of region_1' column header, with 'Text Filters' selected. A sub-menu is open under 'Text Filters' with 'Equals...' highlighted. A 'Filter Rows' dialog box is displayed, prompting the user to apply one or more filter conditions to the rows in the table. The condition being set is 'equals' for the 'State' value. The dialog includes options for 'Basic' and 'Advanced' filtering, and a radio button for 'And' or 'Or'. The 'OK' button is highlighted with a red border.

With extra values like Federal district and island removed, you now have a list of the 50 states and their official two-letter abbreviations. You can rename the columns to make more sense, for example State name, Status, and Abbreviation, by right-clicking the column headers and selecting Rename.

Note that all of these steps are recorded under Applied Steps in the Query Settings pane.

Your shaped table now looks like this:

	A ^B C State name	A ^B C Status	A ^B C Abbreviation
1	Alabama	State	AL
2	Alaska	State	AK
3	Arizona	State	AZ
4	Arkansas	State	AR
5	California	State	CA
6	Colorado	State	CO
7	Connecticut	State	CT
8	Delaware	State	DE
9	Florida	State	FL
10	Georgia	State	GA
11	Hawaii	State	HI
12	Idaho	State	ID
13	Illinois	State	IL
14	Indiana	State	IN
15	Iowa	State	IA
16	Kansas	State	KS
17	Kentucky	State	KY

Retitle the table to State codes in the Properties field of Query Settings.

Data Analytics

Student Material

With the State codes table shaped, you can combine these two tables into one. Since the tables you now have are a result of queries you applied to the data, they're also called queries. There are two primary ways of combining queries: merge and append.

When you have one or more columns you'd like to add to another query, you merge the queries. When you have additional rows of data you'd like to add to an existing query, you append the query.

In this case, you want to merge the State codes query into the Best states for sunglasses query. To merge the queries, switch to the Best states for sunglasses query by selecting it from the Queries pane on the left side of Power Query Editor. Then select Merge Queries from the Combine group in the Home tab of the ribbon.

In the Merge window, drop down the field to select State codes from the other queries available. Select the column to match from each table, in this case State from the Best states for sunglasses query and State name from the State codes query.

If you get a Privacy levels dialog, select Ignore privacy levels checks for this file and then select Save. Select OK.

Merge

Select a table and matching columns to create a merged table.

Best states for sunglass sales

State	Overall rank	Affordability	Weather
Hawaii	10	45	1
Florida	5	25	2
Louisiana	36	29	3
Texas	17	24	4
Georgia	28	19	5

State codes

State name	Status	Abbreviation
Alabama	State	AL
Alaska	State	AK
Arizona	State	AZ
Arkansas	State	AR
California	State	CA

Join Kind

Left Outer (all from first, matching from second)

Use fuzzy matching to perform the merge

Fuzzy matching options

The selection matches 40 of 40 rows from the first table.

OK Cancel

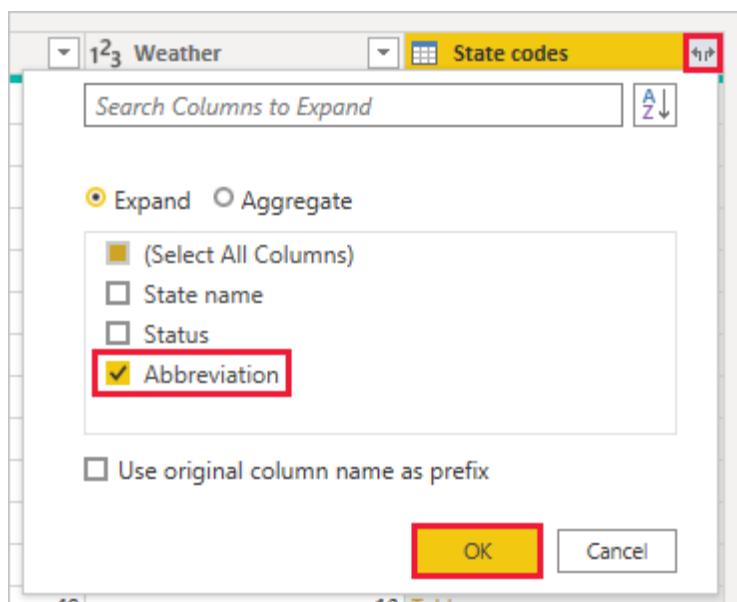
Data Analytics

Student Material

A new column called State codes appears on the right of the Best states for sunglass sales table. It contains the state code query that you merged with the best states for sunglass sales query. All the columns from the merged table are condensed into the State codes column. You can expand the merged table and include only the columns you want.

	A ^B State	1 ² Overall rank	1 ² Affordability	1 ² Weather	State codes
1	Hawaii	10	45	1	1 Table
2	Florida	5	25	2	2 Table
3	Louisiana	36	29	3	3 Table
4	Texas	17	24	4	4 Table
5	Georgia	28	19	5	5 Table
6	Mississippi	19	6	6	6 Table
7	Alabama	16	10	7	7 Table
8	South Carolina	41	27	8	8 Table
9	Arkansas	11	4	9	9 Table
10	Arizona	38	33	10	10 Table
11	Oklahoma	21	11	11	11 Table
12	North Carolina	7	13	12	12 Table
13	California	43	49	13	13 Table
14	Tennessee	22	12	14	14 Table
15	Kentucky	6	9	15	15 Table

To expand the merged table and select which columns to include, select the Expand icon in the column header. In the Expand dialog box, select only the Abbreviation column. Deselect Use original column name as prefix, and then select OK.



Data Analytics

Student Material

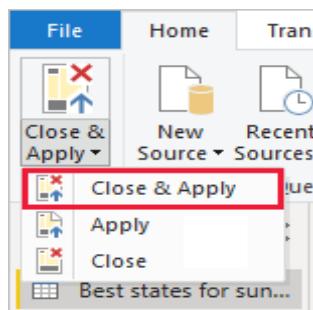
You now have a single query table that combines two data sources, each of which has been shaped to meet your needs. This query can serve as a basis for lots of additional, interesting data connections, such as demographics, wealth levels, or recreational opportunities in the states.

The screenshot shows the Power Query Editor interface with a table of data. The columns are labeled: Overall rank, Affordability, Weather, and Abbreviation. The data includes rows for various US states like Hawaii, Alabama, Florida, etc., with their respective ranks and abbreviations. The ribbon at the top has tabs for File, Home, Transform, Add Column, View, Tools, and Help. The Home tab is selected, showing icons for Close & Apply, New Source, Recent Sources, Enter Data, Data source settings, Manage Parameters, Refresh Preview, Advanced Editor, Manage Columns, Reduce Rows, Sort, Split Column, Group By, Replace Values, and Transform. The Transform tab is also visible. On the right, there's a 'Query Settings' pane with sections for Properties (Name: Best states for sunglass sales) and Applied Steps (listing various steps like Source, Extracted Table From Html, Changed Type, etc.).

	Overall rank	Affordability	Weather	Abbreviation
1	Hawaii	10	45	HI
2	Alabama	16	10	AL
3	Florida	5	25	FL
4	Louisiana	36	29	LA
5	Arizona	38	33	AZ
6	Arkansas	11	4	AR
7	Georgia	28	19	GA
8	California	43	49	CA
9	Mississippi	19	6	MS
10	Colorado	33	36	CO
11	Connecticut	30	46	CT
12	South Carolina	41	27	SC
13	Delaware	32	30	DE
14	Oklahoma	21	11	OK
15	North Carolina	7	13	NC
16	Illinois	47	40	IL
17	Indiana	29	3	IN
18	Kentucky	6	9	KY
19	Iowa	2	8	IA
20	Kansas	7	7	KS
21	Maryland	50	47	MD
22	Missouri	3	1	MO
23	New Mexico	37	26	NM
24	Massachusetts	23	43	MA
25				

5 COLUMNS, 40 ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED ON MONDAY

For now, you have enough data to create an interesting report in Power BI Desktop. Since this is a milestone, apply the changes in Power Query Editor and load them into Power BI Desktop by selecting Close & Apply from the Home tab of the ribbon. You can also select just Apply to keep the query open in Power Query Editor while you work in Power BI Desktop.

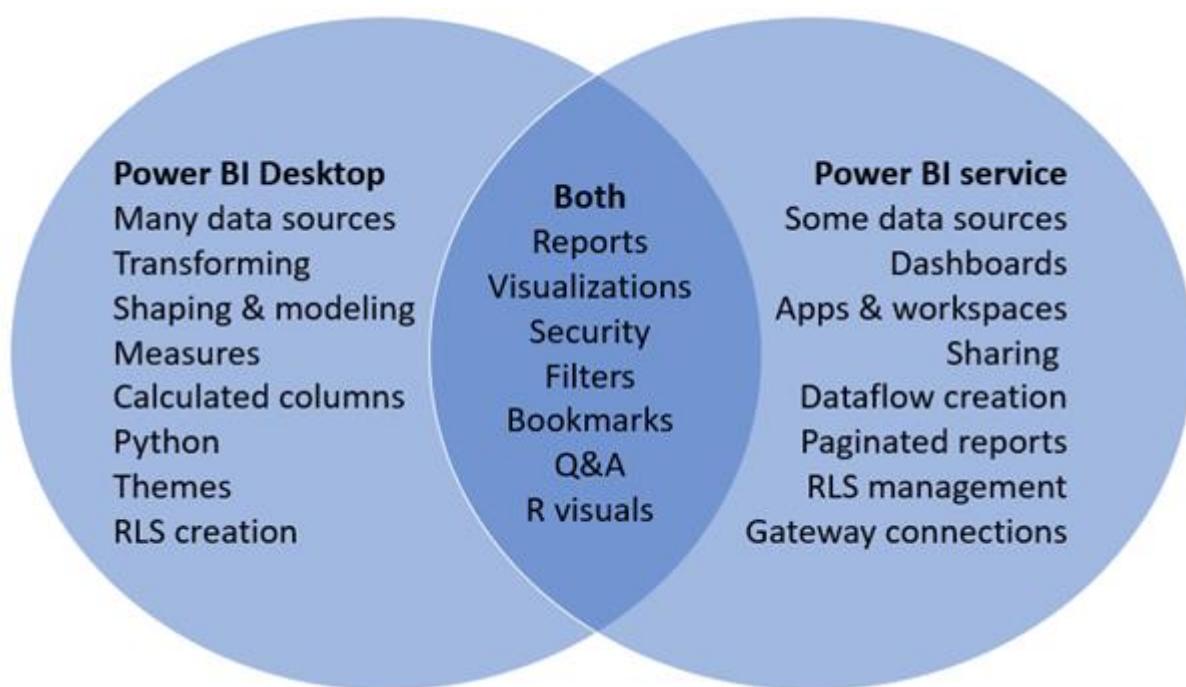


Comparing Power BI Desktop and the Power BI service

Power BI Desktop is an application that you download and install for free on your local computer. Desktop is a complete data analysis and report creation tool that is used to connect to, transform, visualize, and analyze your data. It includes the Query Editor, in which you can connect to many different sources of data, and combine them (often called Modelling) into a data model. Then you design a report based on that data model. Reports can be shared with others directly or by publishing to the Power BI service. The Power BI Desktop getting started guide walks through the process.

The Power BI service is a cloud-based service, or software as a service (SaaS). It supports report editing and collaboration for teams and organizations. You can connect to data sources in the Power BI service, too, but Modelling is limited. The Power BI service is used to do things such as creating dashboards, creating and sharing apps, analyzing and exploring your data to uncover business insights, and much more. What is the Power BI service details many of the capabilities of the Power BI service.

In a Venn diagram comparing Power BI Desktop and the Power BI service, the area in the middle shows some of the areas where the two overlap. Some tasks you can do in either Power BI Desktop or the service. The two outer sides of the Venn diagram show the features that are unique to either the Desktop application or to the Power BI service.



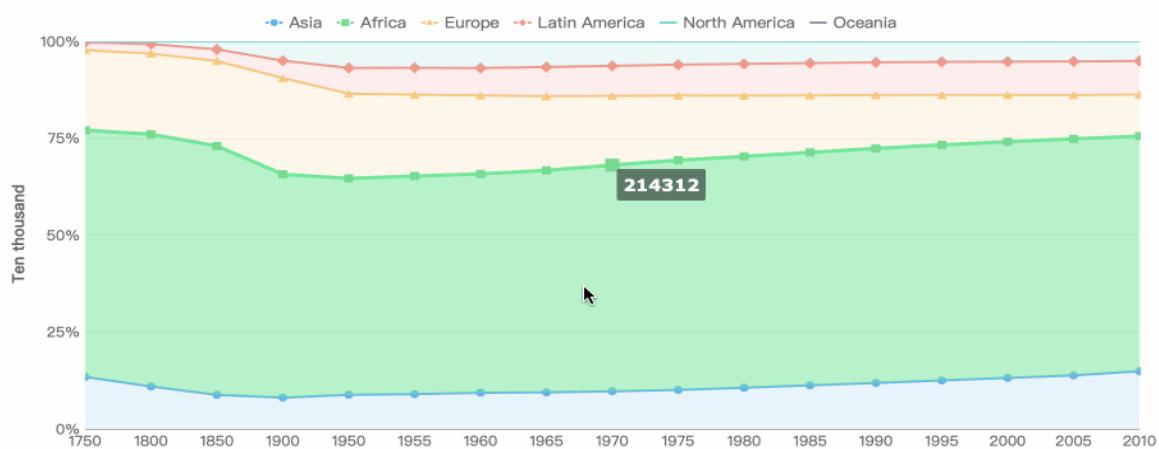
5.7 Different types of visualization charts

1. Column Chart
2. Bar Chart
3. Line Chart
4. Area Chart
5. Pie Chart
6. Scatter Plot
7. Bubble Chart
8. Gauge
9. Radar Chart
10. Frame Diagram
11. Rectangular Tree Diagram
12. Funnel Chart
13. Word Cloud Chart
14. Gantt Chart

5.8 Area Chart

The area chart is formed on the basis of the line chart. It fills the area between the polyline and the axis in the line chart with color. The filling of the color can better highlight the trend information.

The fill color of the area chart should have a certain transparency. The transparency can help the user to observe the overlapping relationship between different series. The area without transparency will cause the different series to cover each other.



Combo Charts: A combo chart is a combination of two column charts, two line graphs, or a column chart and a line graph.

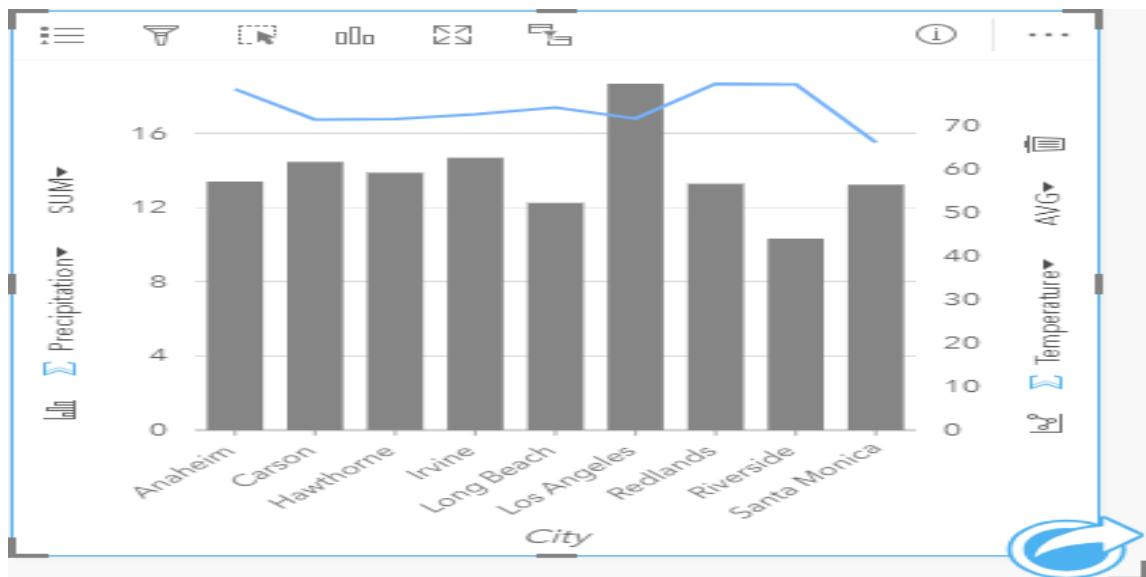
Example:

An environmental organization is tracking the drought conditions in Southern California and wants to compare temperatures and precipitation to determine which cities are most

Data Analytics

Student Material

vulnerable. The organization uses a combo chart to show both the total precipitation and the average temperature for each city in one chart.



5.9 Line Chart

A line chart is used to show the change of data over a continuous time interval or time span. It is characterized by a tendency to reflect things as they change over time or ordered categories.

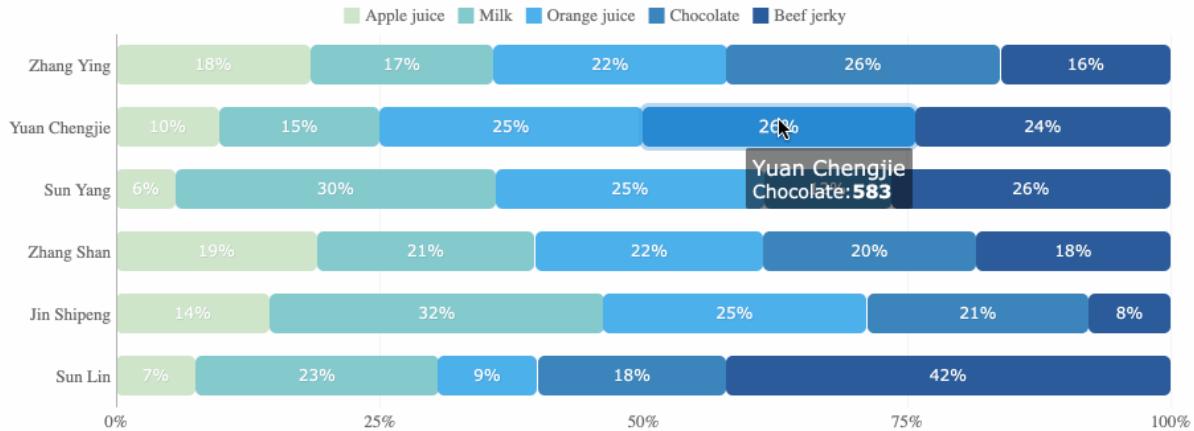
It should be noted that the number of data records of the line graph should be greater than 2, which can be used for trend comparison of large data volume. And it is better not to exceed 5 polylines on the same graph.



5.10 Bar Chart

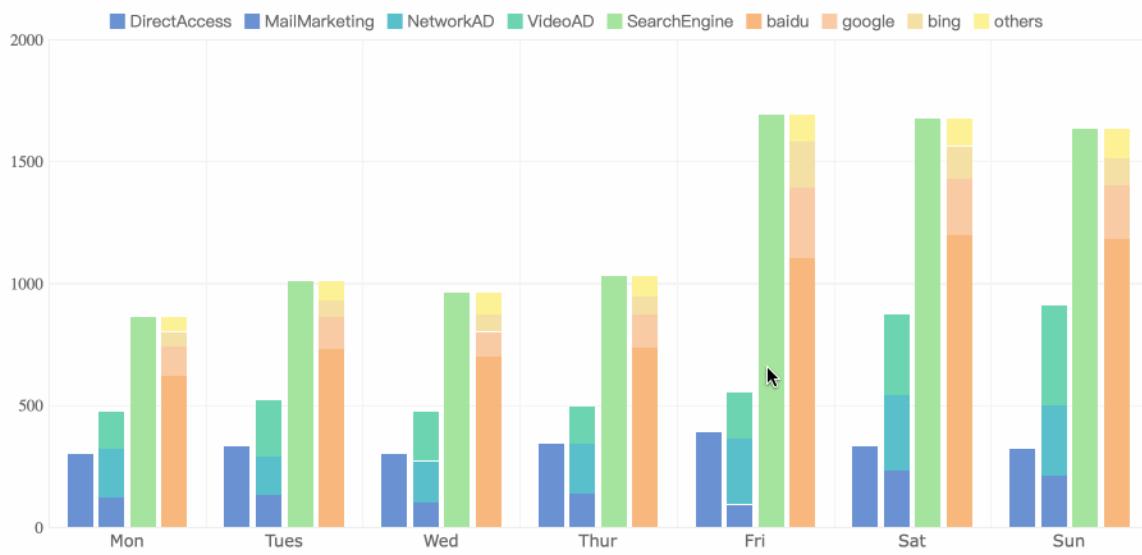
Bar charts are similar to column charts, but the number of bars can be relatively large. Compared with the column chart, the positions of its two axes are changed.

Percent Stacked Bar Chart



5.11 Column Chart

Column charts use vertical columns to show numerical comparisons between categories, and the number of columns should not be too large (the labels of the axis may appear incomplete if there are too many columns).

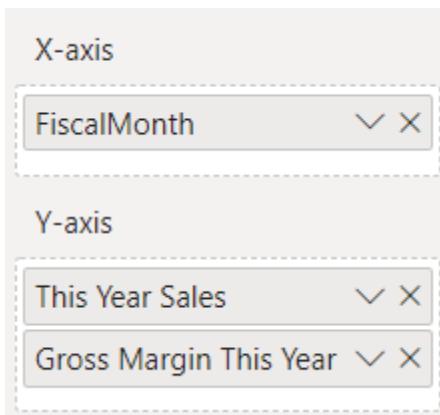


5.12 Combo Chart

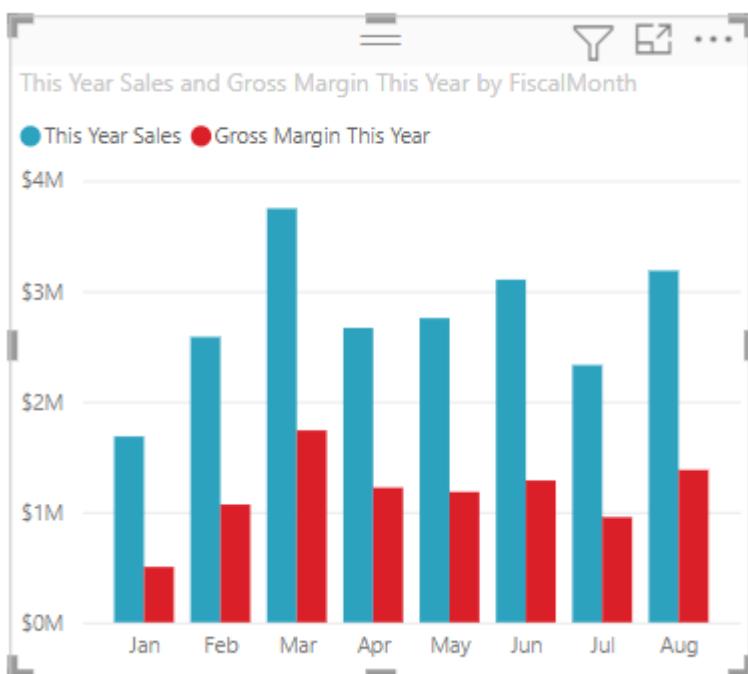
In Power BI, a combo chart is a single visualization that combines a line chart and a column chart. Combining the two charts into one lets you make a quicker comparison of the data.

Create a basic single-axis combo chart

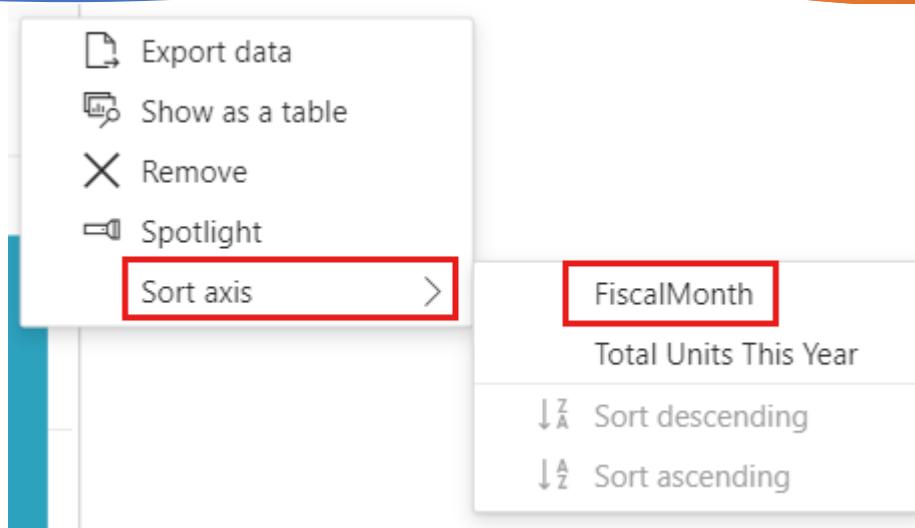
1. Start on a blank report page and create a column chart that displays this year's sales and gross margin by month.
 - a. From the Fields pane, select **Sales > This Year Sales > Value**.
 - b. Select **Sales > Gross Margin This Year** and drag it to the **Y-axis** well.
 - c. Select **Time > FiscalMonth** and drag it to the **X-axis** well.



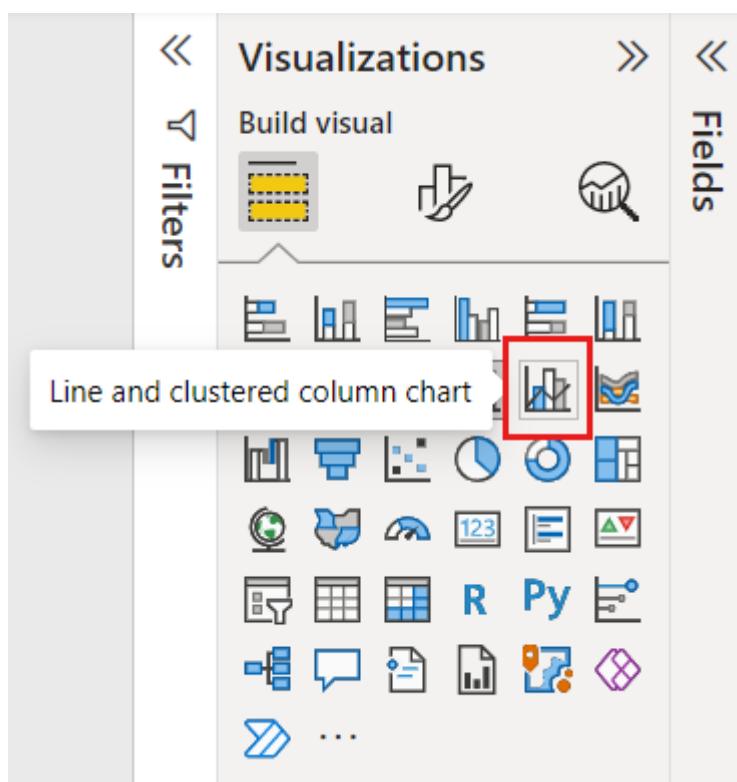
2. The visualization will be similar to this one.



3. In the upper-right corner of the visual, select the **More options** ellipsis (...) and select **Sort axis > FiscalMonth**.



4. Select the ellipsis again and choose **Sort axis > Sort ascending**.
5. Convert the column chart to a combo chart. There are two combo charts available: **Line and stacked column** and **Line and clustered column**. With the column chart selected, from the **Visualizations** pane select the **Line and clustered column chart**.



6. From the **Fields** pane, drag **Sales > Last Year Sales** to the **Line y-axis** bucket.

X-axis

FiscalMonth

Column y-axis

This Year Sales

Gross Margin This Year

Line y-axis

Last Year Sales

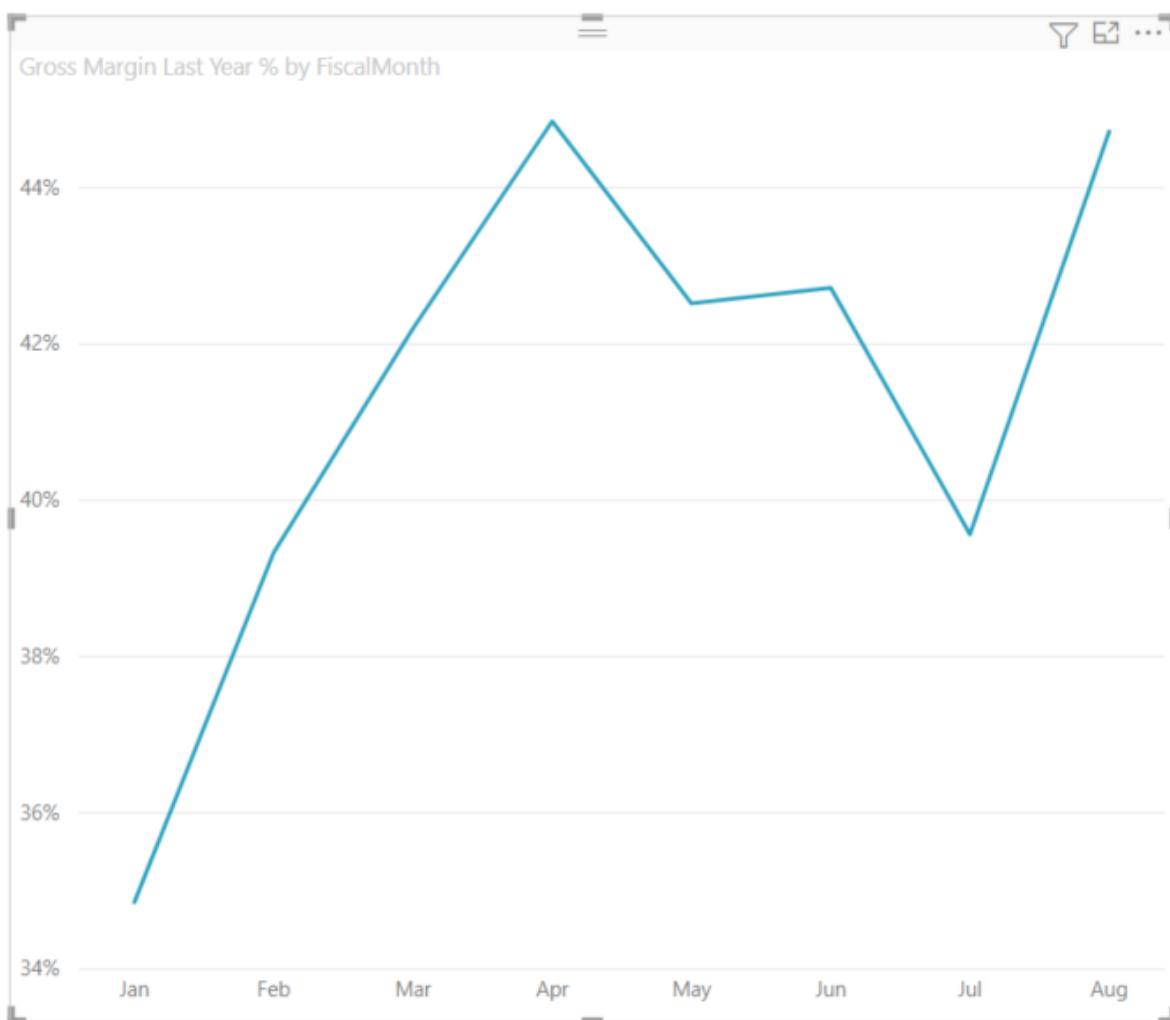
Your combo chart should look something like this:



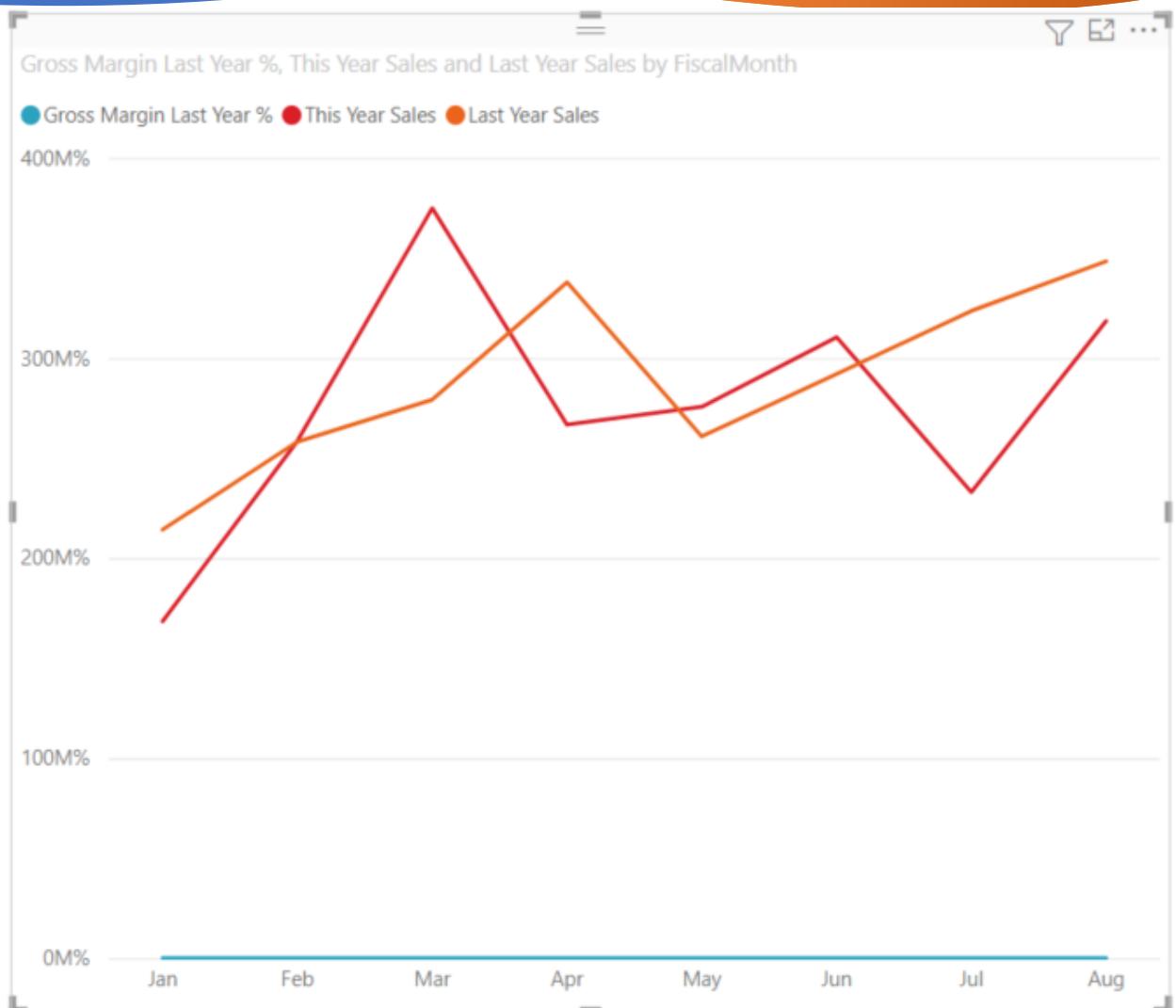
Create a combo chart with two axes

In this task, we'll compare gross margin and sales.

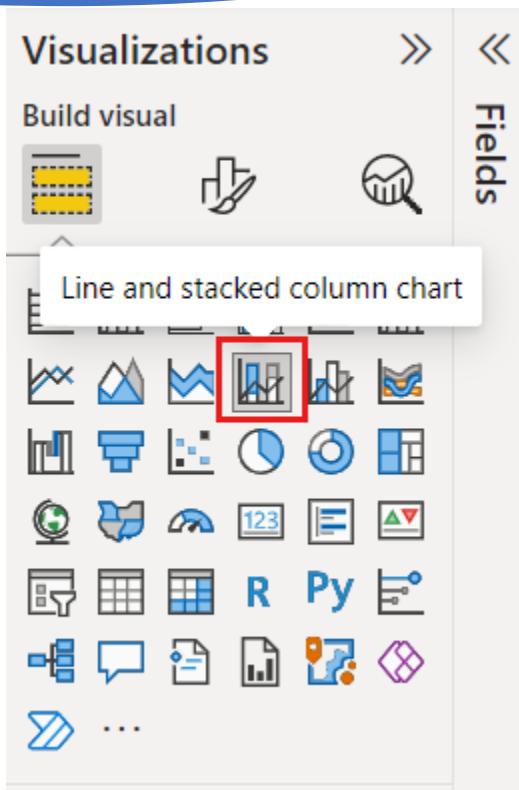
1. Create a new line chart that tracks **Gross Margin Last Year %** by **FiscalMonth**.
2. Select the ellipsis to sort it by **FiscalMonth**, then select the ellipsis again and choose **Sort axis > Sort ascending**.
3. In January GM% was 35%, peaked at 45% in April, dropped in July and peaked again in August. Will we see a similar pattern in sales last year and this year?



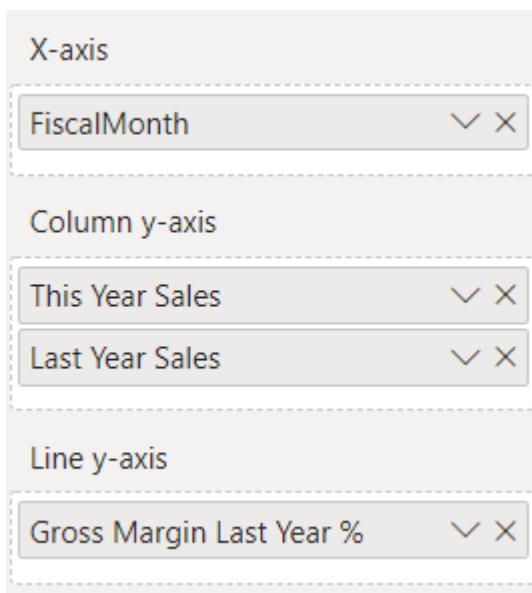
4. Add **This Year Sales > Value** and **Last Year Sales** to the line chart. The scale of **Gross Margin Last Year %** is much smaller than the scale of **Sales** which makes it difficult to compare.



5. To make the visual easier to read and interpret, convert the line chart to a **Line and stacked column chart**.



6. Drag **Gross Margin Last Year %** from **Column y-axis** into **Line y-axis**.

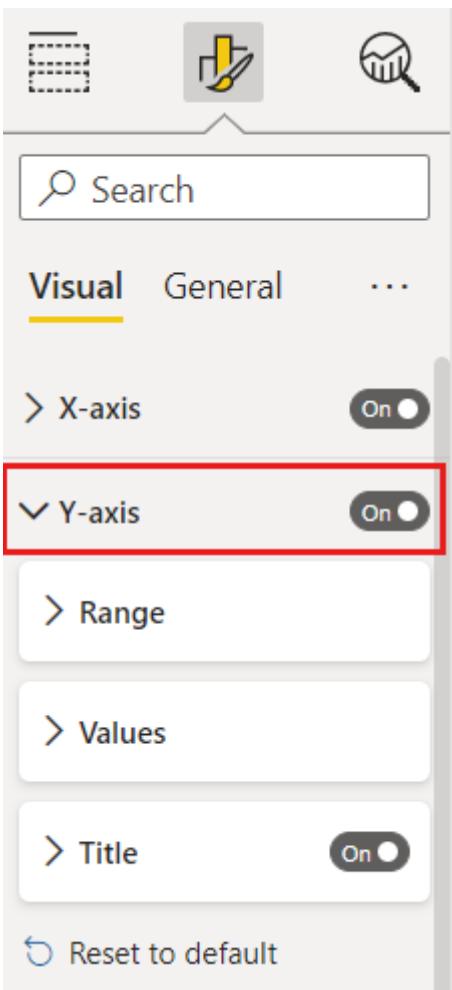


7. Power BI creates two axes, thus allowing the datasets to be scaled differently; the left measures sales dollars and the right measures percentage. And we see the answer to our question: yes, we do see a similar pattern.

Add titles to the axes



1. Select the paintbrush icon to open the **Formatting** pane.
2. Set **Y-axis** to **On**, then select the down arrow to expand the **Y-axis** options.
3. Set **Y-axis > Values > Display units** to **Millions**.
4. Set **Y-axis > Title** to **On**, then set **Style** to **Show title only**



5. Set **Secondary y-axis** to **On** to display options for formatting the line chart portion of the combo chart.

Data Analytics

Student Material

Secondary y-axis On

> Range

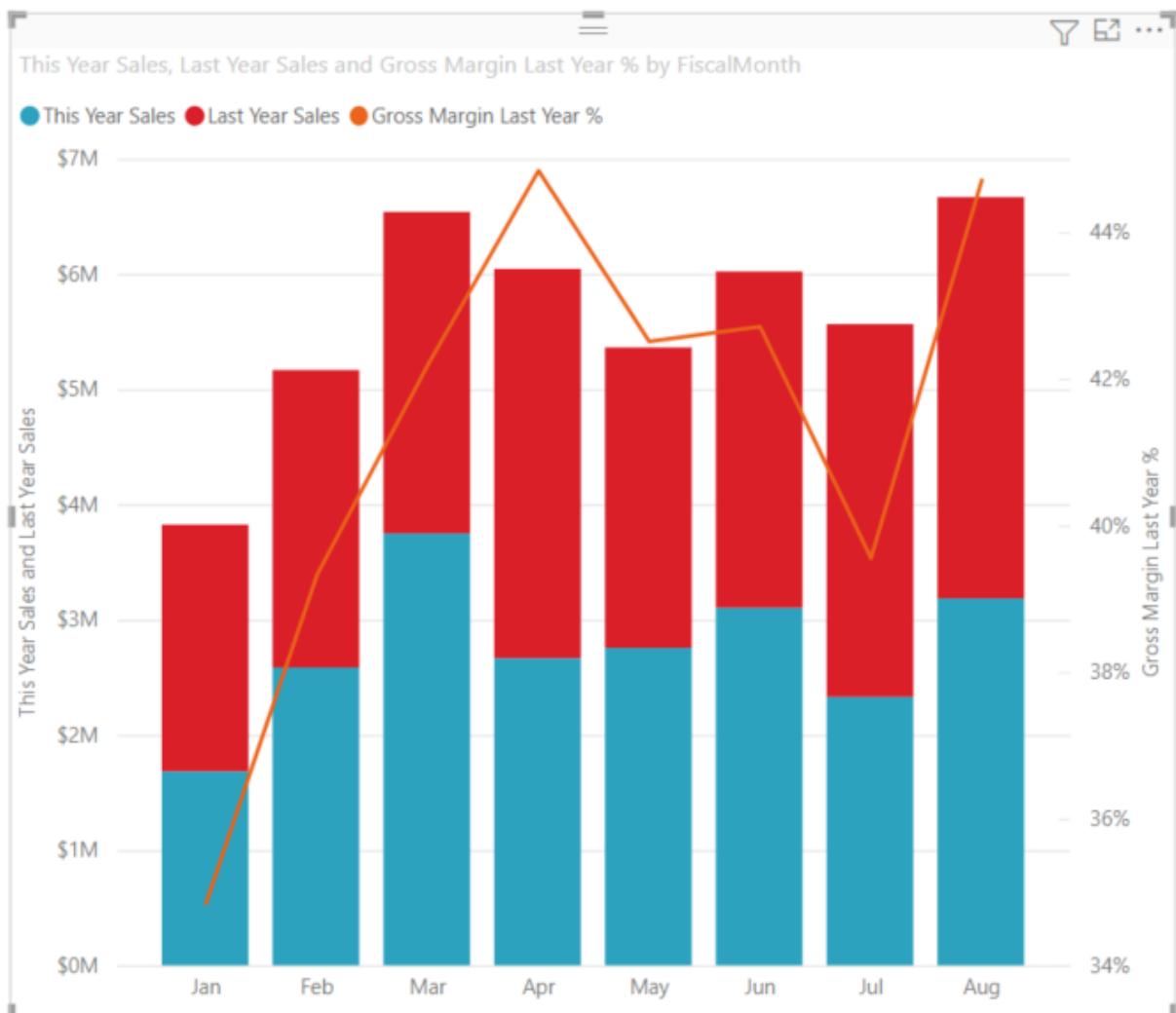
> Values

> Title

Reset to default

- Under **Secondary y-axis**, set **Title** to **On**.

Your combo chart now displays dual axes, both with titles.



Data Analytics

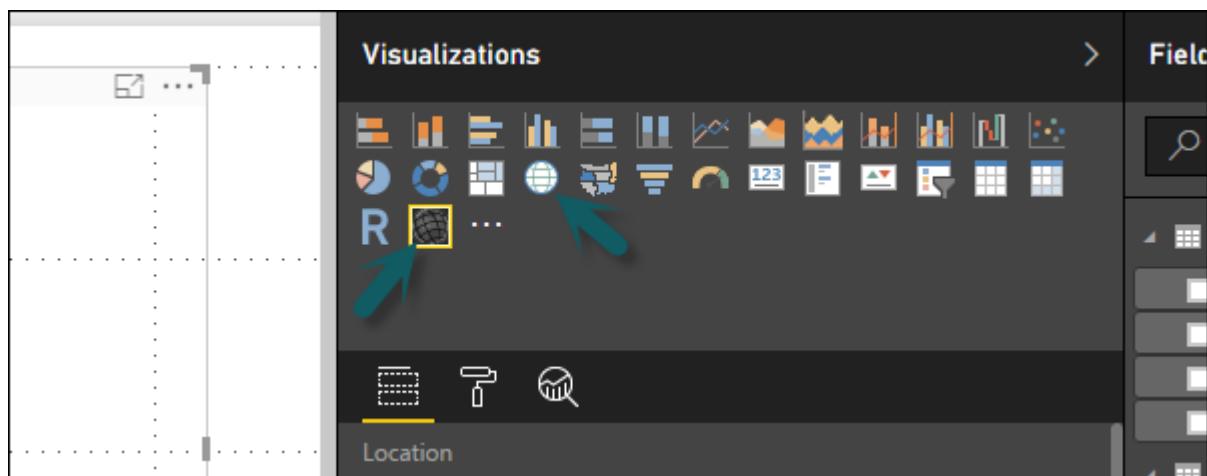
Student Material

7. Optionally, modify the text font, size, and color and set other formatting options to improve the display and readability of the chart.

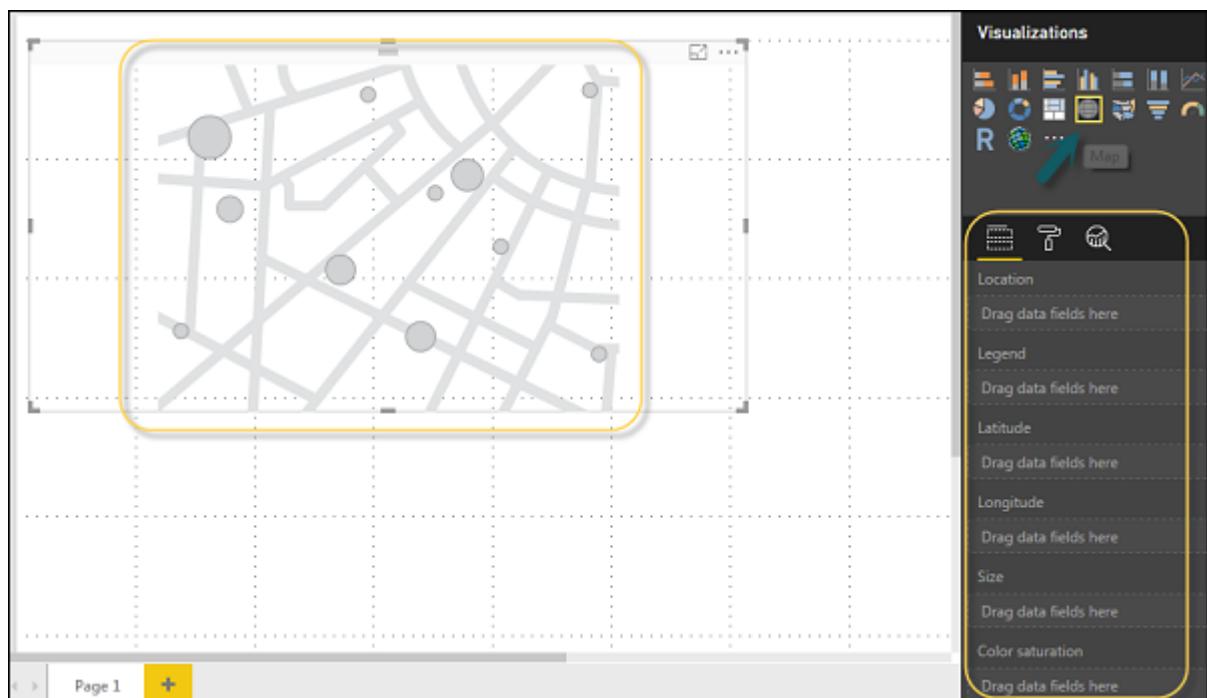
Assignment: Create map visualization using Power BI

Solution:

In Power BI, we have two types of map visualization - bubble maps and shape maps. If you want to create a bubble map, select the map option from the visualization pane.



To use a bubble map, drag the map from Visualizations to the Report Canvas. To display values, you have to add any location object to the axis.



Brought to you by:

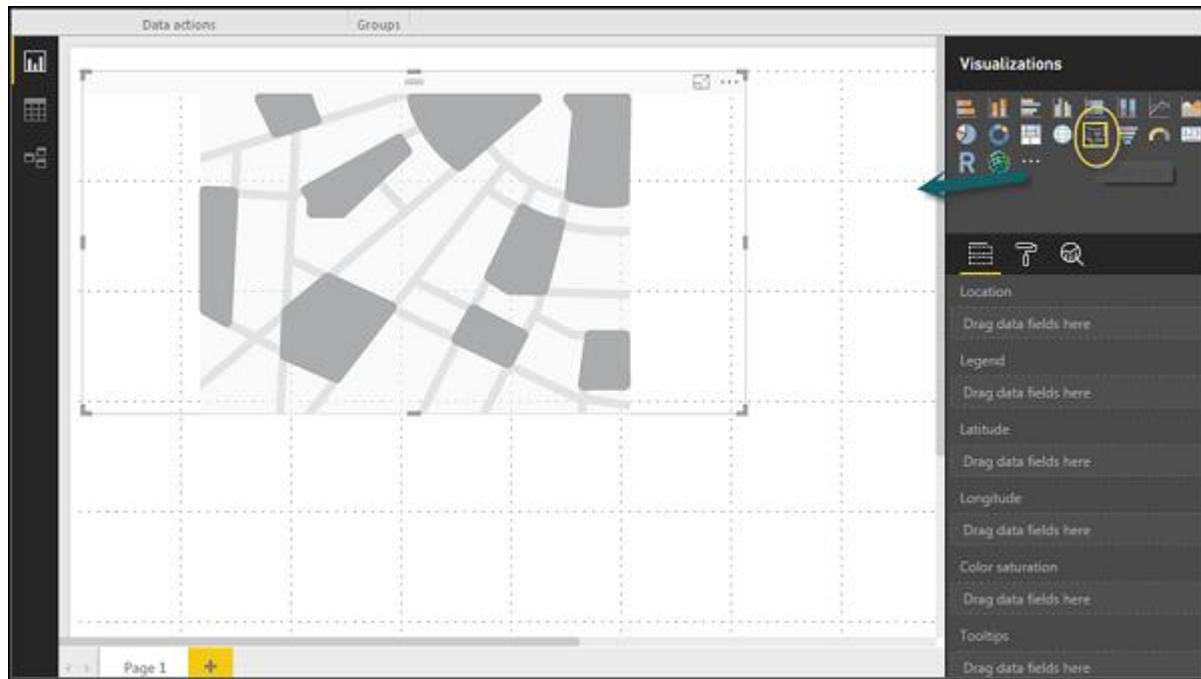


Data Analytics

Student Material

In the value fields, you can see that it accepts values axis such as City and State and or you can also add longitude and latitude values. To change the bubble size, you need to add a field to the value axis.

You can also use a filled map in data visualization, just by dragging the filled map to the Report Canvas.



Note – If you see a warning symbol on top of your map visualization, it means that you need to add more locations to your map chart.

Practice task:

Create combination chart using Power BI

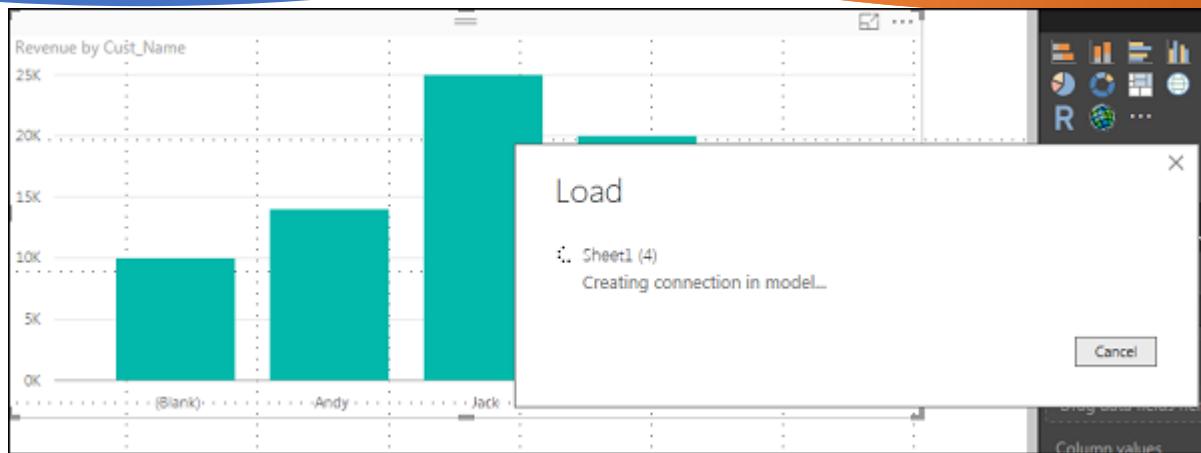
Solution:

In data visualization, it is also required to plot multiple measures in a single chart. Power BI supports various combination chart types to plot measure values. Let us say you want to plot revenue and unit_sold in one chart. Combination charts are the most suitable option for these kind of requirement.

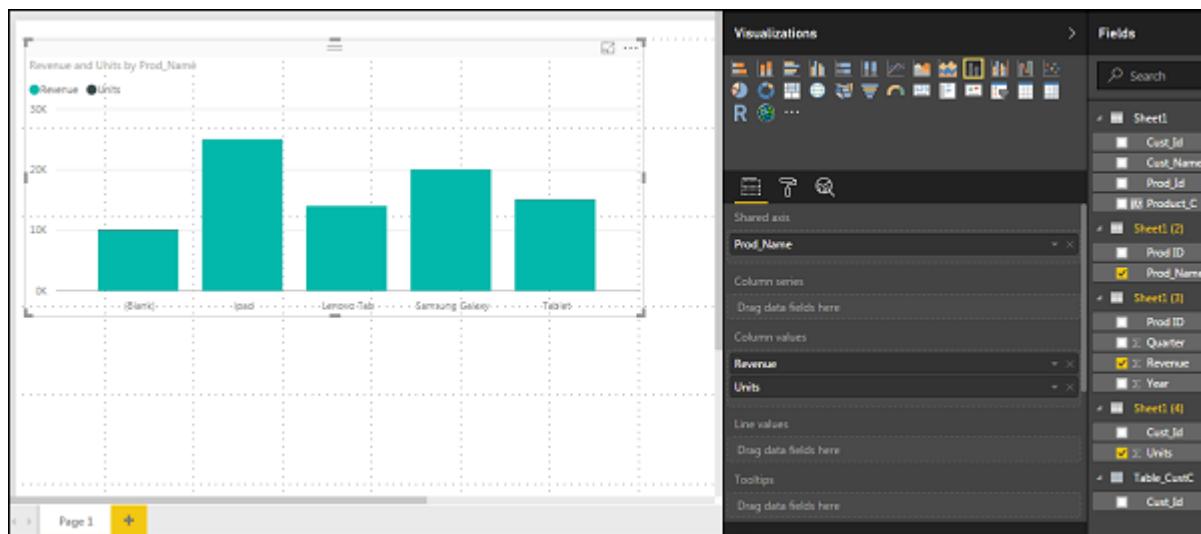
One of the most common Combination chart in Power BI is Line and Stacked column charts. Let us say we have a revenue field and we have added a new data source that contains customer-wise unit quantity and we want to plot this in our visualization.

Data Analytics

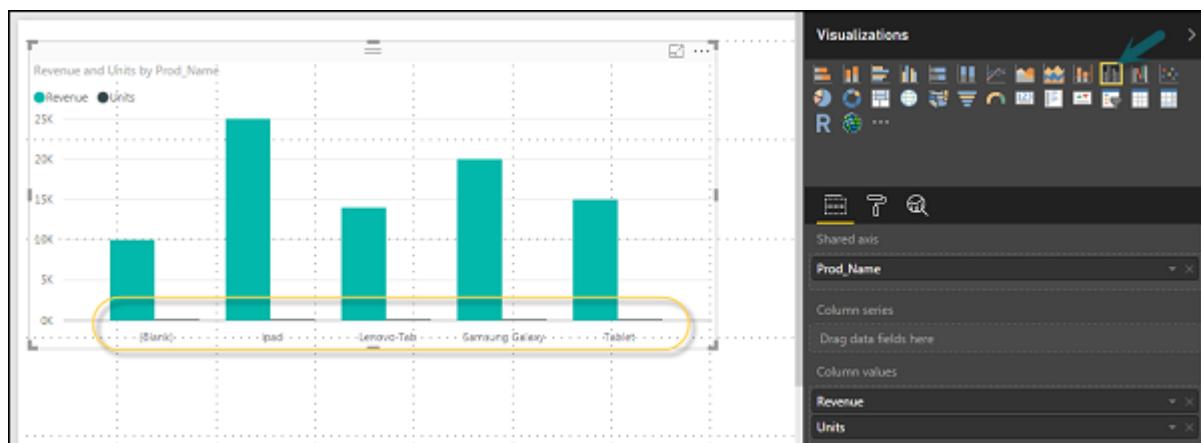
Student Material



Once you add a data source, it will be added to the list of fields on the right side. You can add units to the column axis as shown in the following screenshot.



You have other type of combine chart that you can use in Power BI - Line and Clustered Column.



Chapter: 6

Data Regression

Scope:

- Types of Regression Models
- Simple Linear Regression Model
- Simple Linear Regression Equation
- Equation for the best straight line
- Graph of the best straight line
- Interpreting the Results
- Measures of Variation

Regression Models:

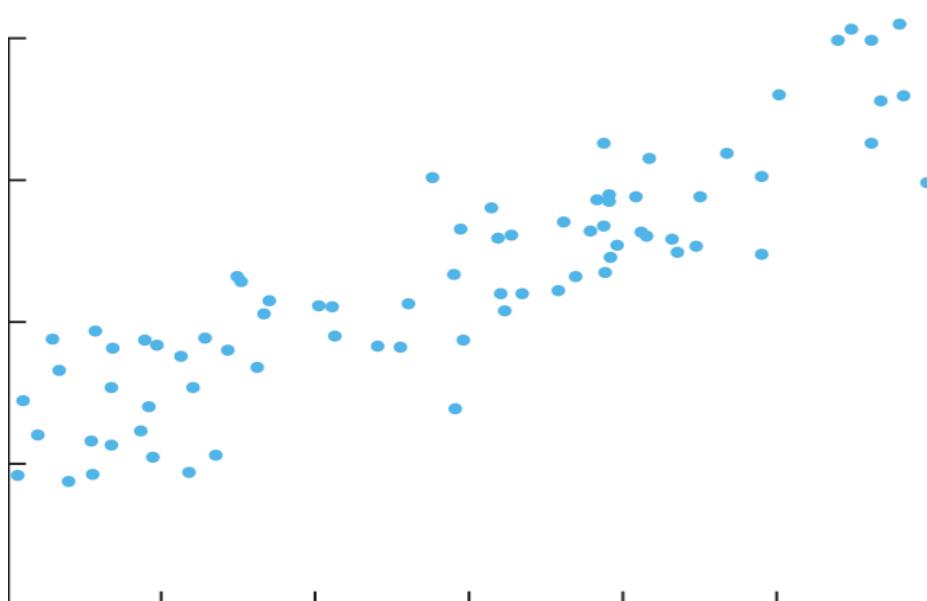
Regression analysis is a predictive modelling technique that analyzes the relation between the target or dependent variable and independent variable in a dataset. The different types of regression analysis techniques get used when the target and independent variables show a linear or non-linear relationship between each other, and the target variable contains continuous values. The regression technique gets used mainly to determine the predictor strength, forecast trend, time series, and in case of cause & effect relation.

Regression analysis is the primary technique to solve the regression problems in machine learning using data modelling. It involves determining the best fit line, which is a line that passes through all the data points in such a way that distance of the line from each data point is minimized.

How does it work?

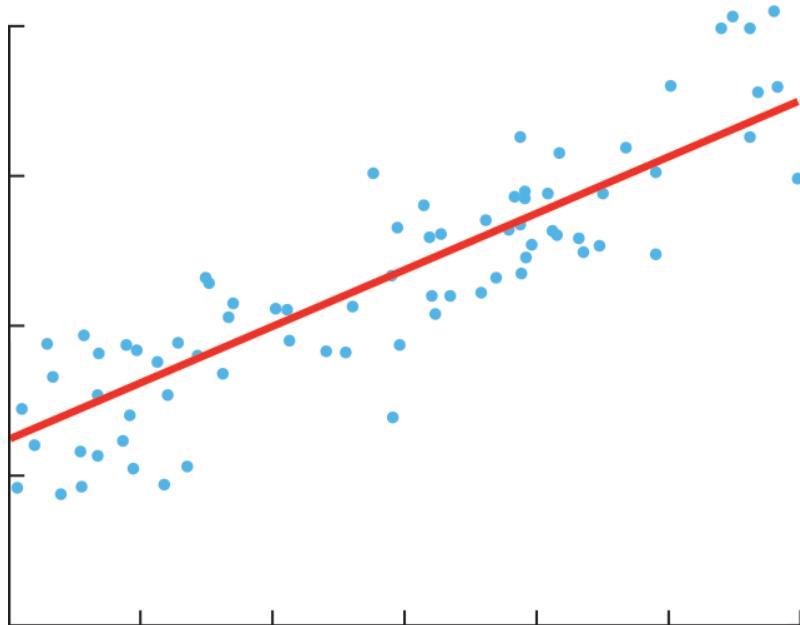
In order to conduct a regression analysis, you gather the data on the variables in question. (Reminder: you likely don't have to do this yourself, but it's helpful for you to understand the process your data analyst colleague uses.) You take all of your monthly sales numbers for, say, the past three years and any data on the independent variables you're interested in. So, in this case, let's say you find out the average monthly rainfall for the past three years as well.

Then you plot all of that information on a chart that looks like this:



The y-axis is the amount of sales (the dependent variable, the thing you're interested in, is always on the y-axis) and the x-axis is the total rainfall. Each blue dot represents one month's data—how much it rained that month and how many sales you made that same month.

Building a Regression model



This is called the regression line and it's drawn (using a statistics program like SPSS or STATA or even Excel) to show the line that best fits the data. In other words, explains Redman, "The red line is the best explanation of the relationship between the independent variable and dependent variable."

In addition to drawing the line, your statistics program also outputs a formula that explains the slope of the line and looks something like this:

$$Y = 200 + 5X + \text{error term}$$

Ignore the error term for now. It refers to the fact that regression isn't perfectly precise. Just focus on the model:

$$Y = 200 + 5X$$

What this formula is telling you is that if there is no “x” then Y = 200. So, historically, when it didn’t rain at all, you made an average of 200 sales and you can expect to do the same going forward assuming other variables stay the same. And in the past, for every additional inch of rain, you made an average of five more sales. “For every increment that x goes up one, y goes up by five,” says Redman.

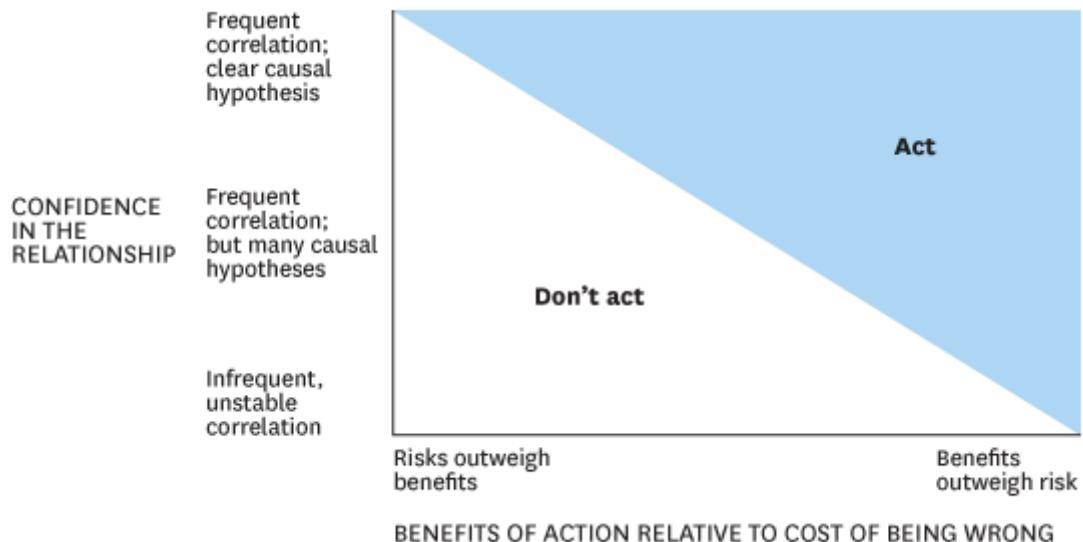
Now let’s return to the error term. You might be tempted to say that rain has a big impact on sales if for every inch you get five more sales, but whether this variable is worth your attention will depend on the error term. A regression line always has an error term because, in real life, independent variables are never perfect predictors of the dependent variables. Rather the line is an estimate based on the available data. So the error term tells you how certain you can be about the formula. The larger it is, the less certain the regression line.

The above example uses only one variable to predict the factor of interest — in this case rain to predict sales. Typically you start a regression analysis wanting to understand the impact of several independent variables. So you might include not just rain but also data about a competitor’s promotion. “You keep doing this until the error term is very small,” says Redman. “You’re trying to get the line that fits best with your data.” While there can be dangers to trying to include too many variables in a regression analysis, skilled analysts can minimize those risks. And considering the impact of multiple variables at once is one of the biggest advantages of regression.

What mistakes do people make when working with regression analysis?

First, don’t tell your data analyst to go out and figure out what is affecting sales. “The way most analyses go haywire is the manager hasn’t narrowed the focus on what he or she is looking for,” says Redman. It’s your job to identify the factors that you suspect are having an impact and ask your analyst to look at those. “If you tell a data scientist to go on a fishing expedition, or to tell you something you don’t know, then you deserve what you get, which is bad analysis,” he says. In other words, don’t ask your analysts to look at every variable they can possibly get their hands on all at once. If you do, you’re likely to find relationships that don’t really exist. It’s the same principle as flipping a coin: do it enough times, you’ll eventually think you see something interesting, like a bunch of heads all in a row.

Second, “analyses are very sensitive to bad data” so be careful about the data you collect and how you collect it, and know whether you can trust it. “All the data doesn’t have to be correct or perfect,” explains Redman but consider what you will be doing with the analysis. If the decisions you’ll make as a result don’t have a huge impact on your business, then it’s OK if the data is “kind of leaky.” But “if you’re trying to decide whether to build 8 or 10 of something and each one costs \$1 million to build, then it’s a bigger deal,” he says. The chart below explains how to think about whether to act on the data.



6.1 Types of Regression Models:

There are many types of regression analysis techniques, and the use of each method depends upon the number of factors. These factors include the type of target variable, shape of the regression line, and the number of independent variables.

Below are the different regression techniques:

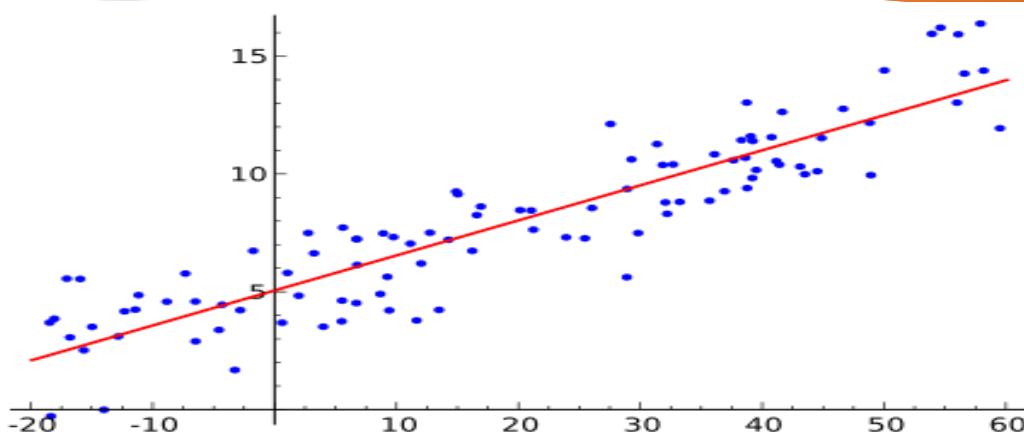
1. Linear Regression
2. Logistic Regression
3. Ridge Regression
4. Lasso Regression
5. Polynomial Regression
6. Bayesian Linear Regression

Linear Regression:

Linear regression is one of the most basic types of regression in machine learning. The linear regression model consists of a predictor variable and a dependent variable related linearly to each other. In case the data involves more than one independent variable, then linear regression is called multiple linear regression models.

The below-given equation is used to denote the linear regression model:
 $y=mx+c+e$

where m is the slope of the line, c is an intercept, and e represents the error in the model.



The best fit line is determined by varying the values of m and c. The predictor error is the difference between the observed values and the predicted value. The values of m and c get selected in such a way that it gives the minimum predictor error. It is important to note that a simple linear regression model is susceptible to outliers. Therefore, it should not be used in case of big size data.

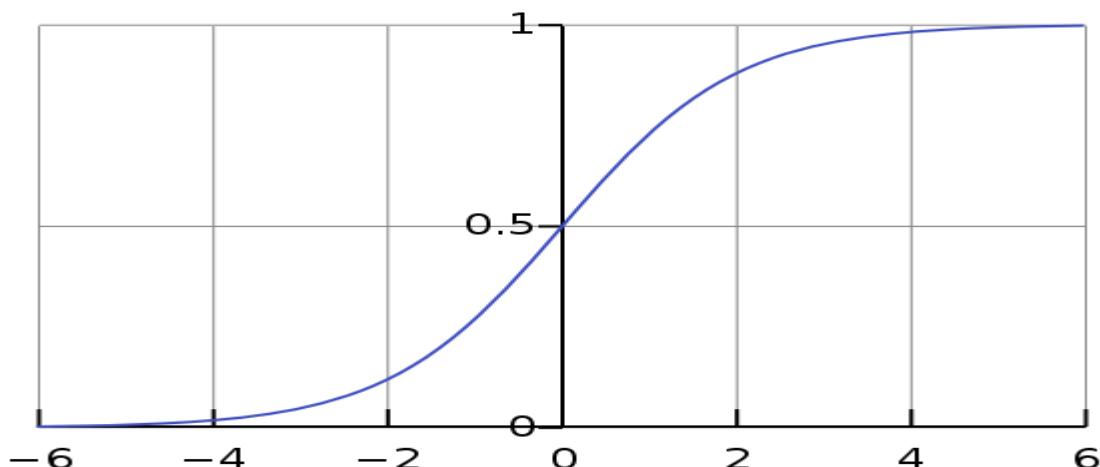
Logistic Regression:

Logistic regression is one of the types of regression analysis technique, which gets used when the dependent variable is discrete. Example: 0 or 1, true or false, etc. This means the target variable can have only two values, and a sigmoid curve denotes the relation between the target variable and the independent variable.

Logit function is used in Logistic Regression to measure the relationship between the target variable and independent variables. Below is the equation that denotes the logistic regression.

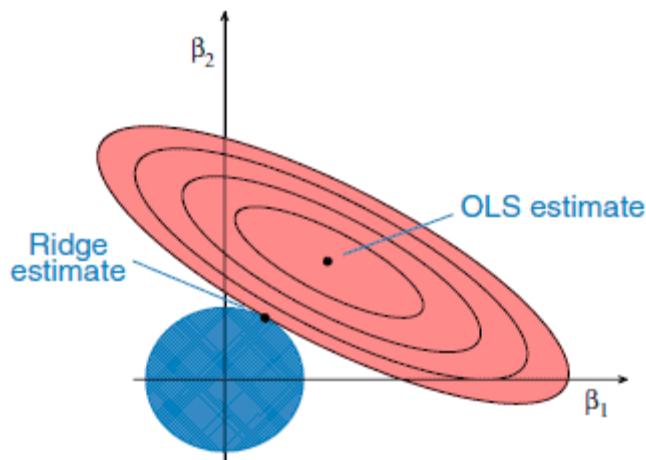
$$\text{logit}(p) = \ln(p/(1-p)) = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_3 + \dots + b_k X_k$$

where p is the probability of occurrence of the feature.



For selecting logistic regression, as the regression analyst technique, it should be noted, the size of data is large with the almost equal occurrence of values to come in target variables. Also, there should be no multicollinearity, which means that there should be no correlation between independent variables in the dataset.

Ridge Regression:



This is another one of the types of regression in machine learning which is usually used when there is a high correlation between the independent variables. This is because, in the case of multi collinear data, the least square estimates give unbiased values. But, in case the collinearity is very high, there can be some bias value. Therefore, a bias matrix is introduced in the equation of Ridge Regression. This is a powerful regression method where the model is less susceptible to overfitting.

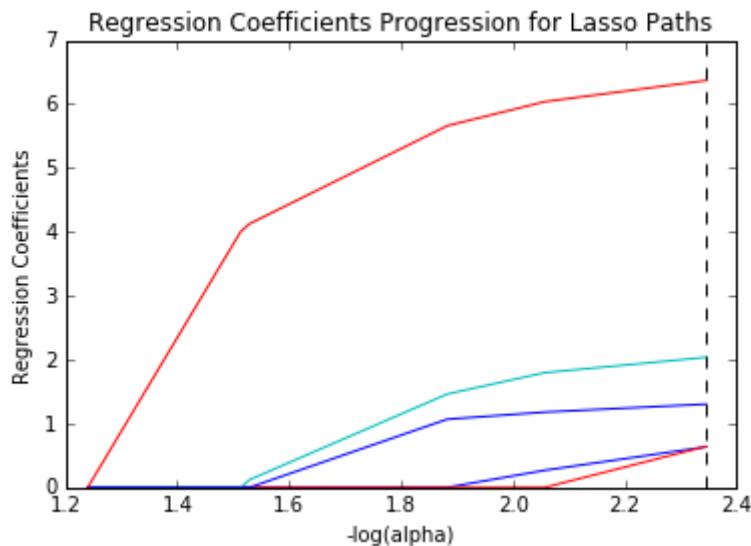
Below is the equation used to denote the Ridge Regression, where the introduction of λ (lambda) solves the problem of multicollinearity:

$$\beta = (X^T X + \lambda I)^{-1} X^T y$$

Lasso Regression:

Lasso Regression is one of the types of regression in machine learning that performs regularization along with feature selection. It prohibits the absolute size of the regression coefficient. As a result, the coefficient value gets nearer to zero, which does not happen in the case of Ridge Regression.

Due to this, feature selection gets used in Lasso Regression, which allows selecting a set of features from the dataset to build the model. In the case of Lasso Regression, only the required features are used, and the other ones are made zero. This helps in avoiding the over fitting in the model. In case the independent variables are highly collinear, then Lasso regression picks only one variable and makes other variables to shrink to zero.



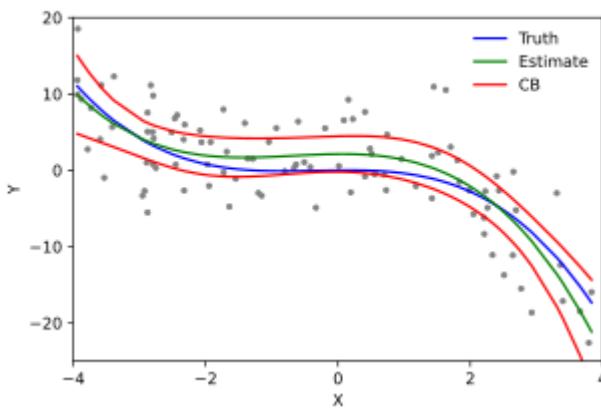
Below is the equation that represents the Lasso Regression method:

$$N^{-1} \sum_{i=1}^N f(x_i, y_i, \alpha, \beta)$$

Polynomial Regression:

Polynomial Regression is another one of the **types of regression analysis** techniques in machine learning, which is the same as Multiple Linear Regression with a little modification. In Polynomial Regression, the relationship between independent and dependent variables, that is X and Y, is denoted by the n-th degree.

It is a linear model as an estimator. Least Mean Squared Method is used in Polynomial Regression also. The best fit line in Polynomial Regression that passes through all the data points is not a straight line, but a curved line, which depends upon the power of X or value of n.



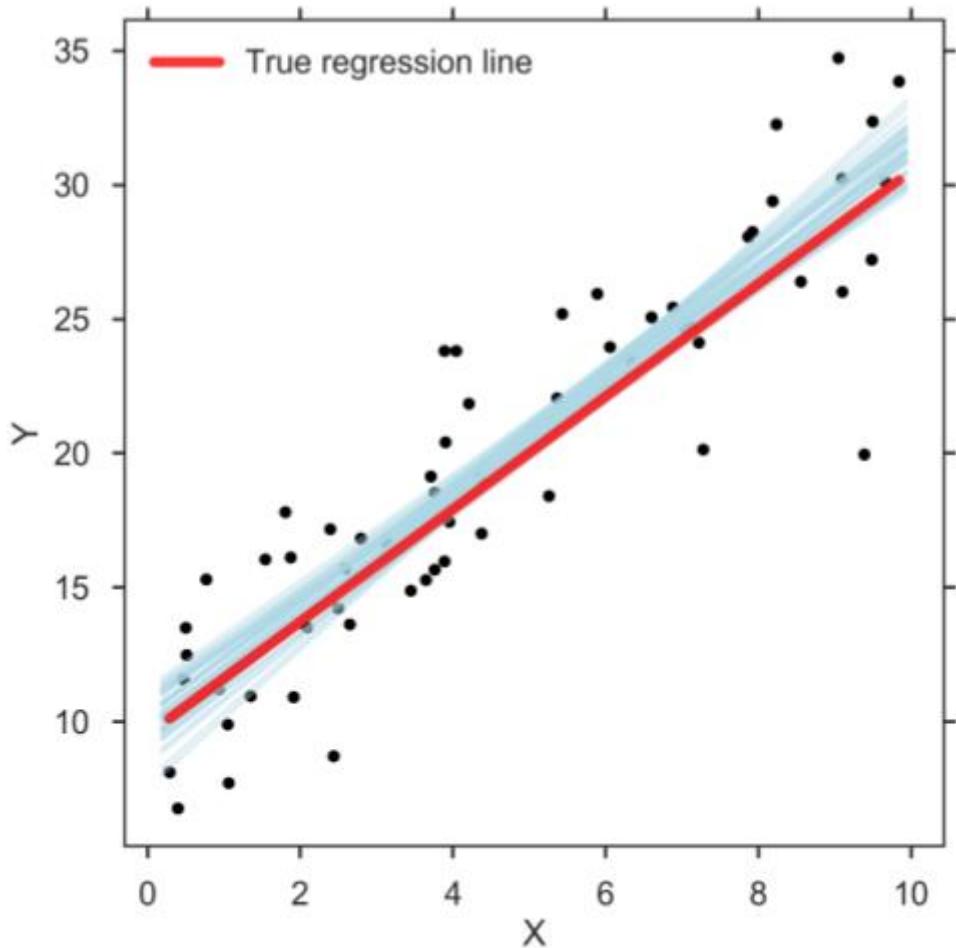
While trying to reduce the Mean Squared Error to a minimum and to get the best fit line, the model can be prone to overfitting. It is recommended to analyze the curve towards the end as the higher Polynomials can give strange results on extrapolation.

Below equation represents the Polynomial Regression:

$$I = \beta_0 + \beta_1 x_1 + \epsilon$$

Bayesian Linear Regression:

Bayesian Regression is one of the types of regression in machine learning that uses the Bayes theorem to find out the value of regression coefficients. In this method of regression, the posterior distribution of the features is determined instead of finding the least-squares. Bayesian Linear Regression is like both Linear Regression and Ridge Regression but is more stable than the simple Linear Regression.



Regression analysis is used to find equations that fit data. Once we have the regression equation, we can use the model to make predictions. One type of regression analysis is linear analysis. When a correlation coefficient shows that data is likely to be able to predict future outcomes and a scatter plot of the data appears to form a straight line, you can use simple linear regression to find a predictive function. If you recall from elementary algebra, the equation for a line is $y = mx + b$. This article shows you how to take data, calculate linear regression, and find the equation $y' = a + bx$. Note: If you're taking AP statistics, you may see the equation written as $b_0 + b_1x$, which is the same thing.

6.2 Simple Linear Regression Model

Regression models describe the relationship between variables by fitting a line to the observed data. Linear regression models use a straight line, while logistic and nonlinear regression models use a curved line. Regression allows you to estimate how a dependent variable changes as the independent variable(s) change. Simple linear regression is used to estimate the relationship between two quantitative variables.

Assumptions of simple linear regression:

Simple linear regression is a parametric test, meaning that it makes certain assumptions about the data. These assumptions are: Homogeneity of variance (homoscedasticity): the size of the error in our prediction doesn't change significantly across the values of the independent variable.

Independence of observations: the observations in the dataset were collected using statistically valid sampling methods, and there are no hidden relationships among observations.

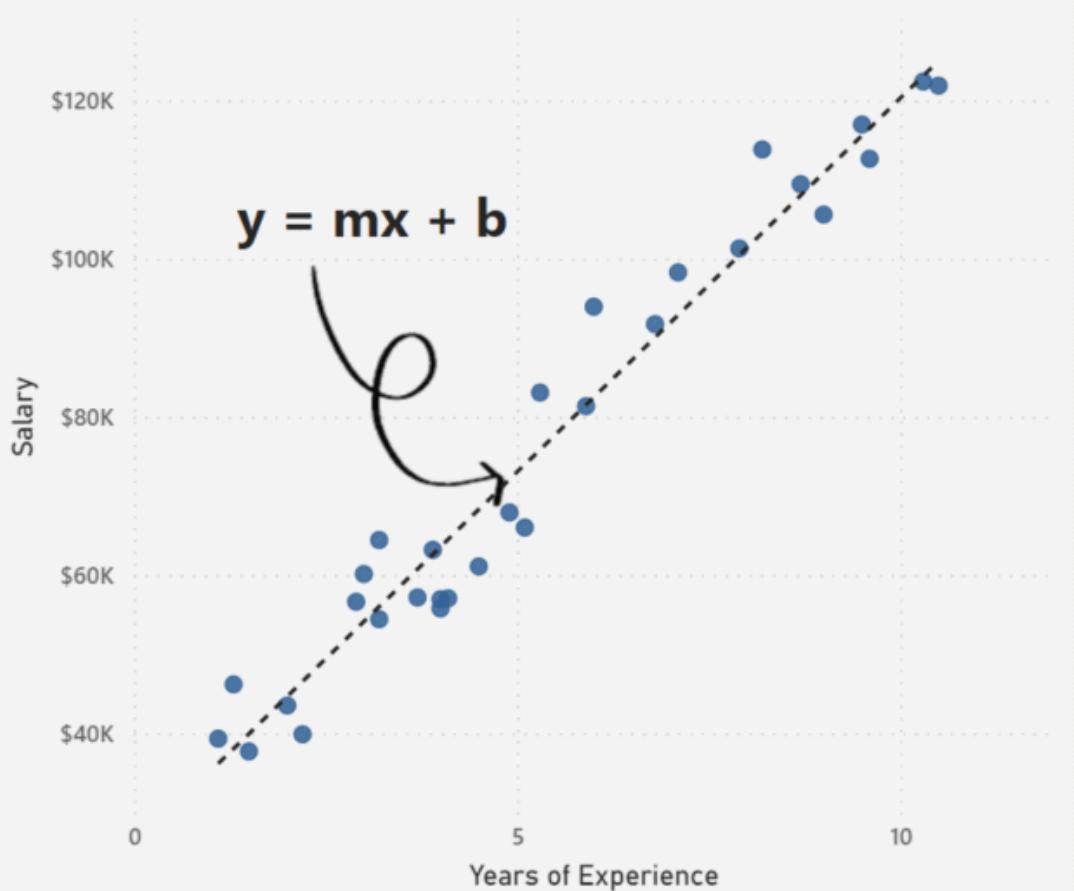
Normality: The data follows a normal distribution.

Linear regression makes one additional assumption: The relationship between the independent and dependent variable is linear: the line of best fit through the data points is a straight line (rather than a curve or some sort of grouping factor).

Example:

Consider this Scatterplot visual. Illustrated is the relationship between Years of Experience and Salary at a fictional company. By fitting a trend line to the Scatterplot, we can see that the more years of experience an employee has, the more they will get paid. Our y data points represent Salary in thousands, and the x data points represent Years of Experience.

Salary by Years of Experience



According to the data above, if an employee has 15 years of experience, then we have to find out how much will they be paid? What about 20 years of experience? 30 years?

Steps we need to take:

Create calculated columns and measures for the components of the equation of a line

Create a What-if parameter for the x-values

Create the measure of the equation of a line and insert a card visual with the measure you can practice using the Salary_data dataset from Kaggle.com.

Step 1: Create Calculated Columns and Measures

The formula for the equation of a line is $y = mx + b$.

Where:

y = How far up the y axis

m = Slope (Change in y divided by change in x) =

$$\frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

x = How far along the x axis

b = b-intercept (The value of y when $x = 0$) =

$$= \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

After finding out m and b with some calculations, we can *input any data point for x and the output will be y*.

We are going to create this formula using DAX calculated columns and measures.

These are the Calculated Columns we need to make and their DAX syntax:

Math Formula	DAX Formula
x^2	$xsq =$ Salary_Data[Years of Experience] ²
xy	$xy =$ Salary_Data[Years of Experience]*Salary_Data[Salary]

These are the Measures we need to make and their DAX syntax:

Math Formula	DAX Formula
n	$n =$ COUNTROWS(Salary_Data)
$\sum xy$	$xysum =$ SUM(Salary_Data[xy])
$\sum x$	$xsum =$ SUM(Salary_Data[Years of Experience])
$\sum y$	$ysum =$ SUM(Salary_Data[Salary])
$\sum x^2$	$xsqrsum =$ SUM(Salary_Data[xsq])

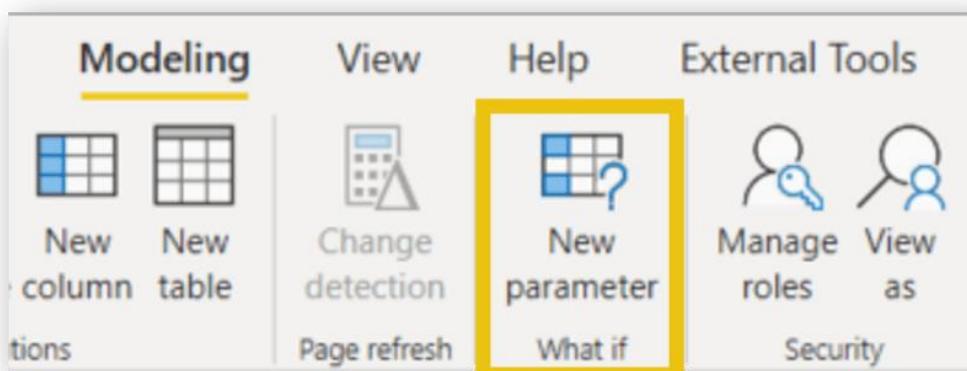
$(n(\sum xy) - (\sum x)(\sum y)) / (n(\sum x^2) - (\sum x)^2)$	$m \text{ (Slope)} =$ $\text{DIVIDE}($ $[n]*[xysum]-[Xsum]*[Ysum],$ $[n]*[xsqrsum]-[Xsum]^2,$ 0 $)$
$((\sum y)(\sum x^2) - (\sum x)(\sum xy)) / (n(\sum x^2) - (\sum x)^2)$	$b \text{ (Intercept)} =$ $\text{DIVIDE}($ $[ysum]*[xsqrsum]-$ $[Xsum]*[xysum],$ $[n]*[xsqrsum]-[Xsum]^2,$ 0 $)$

Step 2: Setting up a What-if parameter

Now that we have performed our calculations, for any number we insert into x, the formula will predict y.

To demonstrate this, we will be using a **What-if** parameter with a slicer.

A **What-if** parameter allows you to create an interactive slicer with a variable to visualize and quantify another value in your report.



After clicking the **What-if** parameter on the modeling ribbon, we can input these values:

What-if parameter



Name

Data type

Minimum

Maximum

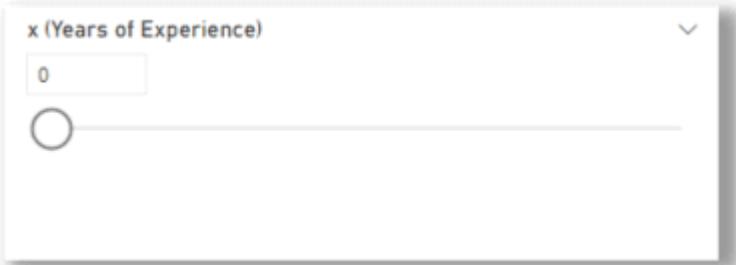
Increment

Default

Add slicer to this page

Name:	x	(Years	of	Experience)
Data	type:		Whole	Number
Minimum:				0
Maximum:				30
Increment:				1
Add slicer to this page:	Yes			

Once we press 'OK', the **What-if** parameter will have generated a series of x values and a slicer to use for our **What-if** scenarios.

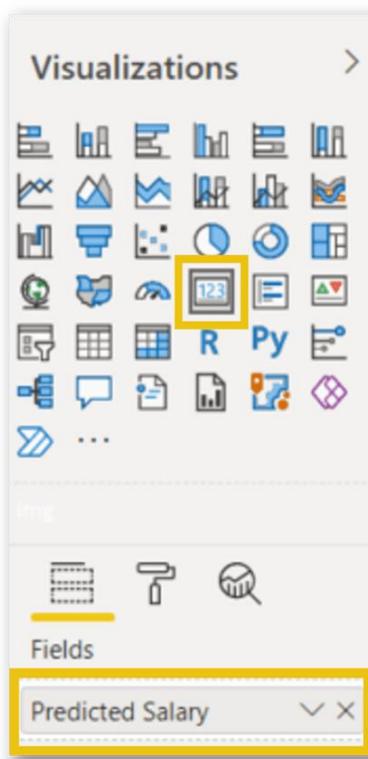


Step 3: Complete the measure for the equation of a line and visualize

Finally, we create the **Predicted Salary** measure that contains the complete formula for the regression equation of a line $y = mx + b$, using the components we have created previously.

Math Formula	DAX Formula
$y = mx + b$	Predicted Salary = ([m (Slope)]* 'x (Years of Experience)"[x (Years of Experience)Value]+ [b (Intercept)])

We can now insert a card visual in the report and select the measure **Predicted Salary** for the field.



We can test the calculation by toggling the slicer left and right. This action will interact with the Card visual and represent the predicted salary for the selected **Years of Experience**.

x (Years of Experience)

20



214.79K

Predicted Salary

Create awesome visualizations!

There are many ways to visualize the prediction that we have set up using Linear Regression. A toggle and a card provide enough information, but we can also integrate that calculation into a chart to visualize the existing data and the predicted value.



6.3 Simple Linear Regression Equation:

Linear regression is a way to model the relationship between two variables. You might also recognize the equation as the slope formula. The equation has the form $Y = a + bX$, where Y is the dependent variable (that's the variable that goes on the Y axis), X is the independent variable (i.e. it is plotted on the X axis), b is the slope of the line and a is the y -intercept.

$$a = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

The first step in finding a linear regression equation is to determine if there is a relationship between the two variables. This is often a judgment call for the researcher. You'll also need a list of your data in x - y format (i.e. two columns of data—independent and dependent variables).

How to Find a Linear Regression Equation:

Step 1: Make a chart of your data, filling in the columns in the same way as you would fill in the chart if you were finding the *Pearson's Correlation Coefficient*.

SUBJECT	AGE X	GLUCOSE LEVEL Y	XY	X^2	Y^2
1	43	99	4257	1849	9801
2	21	65	1365	441	4225
3	25	79	1975	625	6241
4	42	75	3150	1764	5625
5	57	87	4959	3249	7569
6	59	81	4779	3481	6561
Σ	247	486	20485	11409	40022

From the above table, $\Sigma x = 247$, $\Sigma y = 486$, $\Sigma xy = 20485$, $\Sigma x^2 = 11409$, $\Sigma y^2 = 40022$. n is the sample size (6, in our case).

Step 2: Use the following equations to find a and b.

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma xy)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$b = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$a = 65.1416$$

$$b = .385225$$

step-by-step instructions for solving this formula.

Find a:

- $((486 \times 11,409) - ((247 \times 20,485)) / 6 (11,409) - 247^2)$
- $484979 / 7445$
- $=65.14$

Find b:

- $(6(20,485) - (247 \times 486)) / (6 (11409) - 247^2)$
- $(122,910 - 120,042) / 68,454 - 247^2$
- $2,868 / 7,445$
- $= .385225$

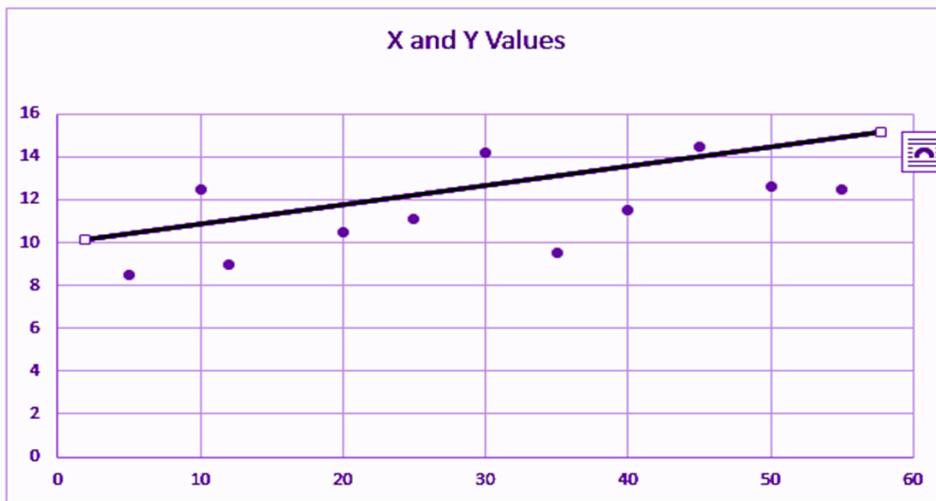
Step 3: Insert the values into the equation.

$$y' = a + bx$$

$$y' = 65.14 + .385225x$$

Least Square Regression Line:

The most popular method to fit a regression line in the XY plot is the method of least-squares. This process determines the best-fitting line for the noted data by reducing the sum of the squares of the vertical deviations from each data point to the line. If a point rests on the fitted line accurately, then its perpendicular deviation is 0. Because the variations are first squared, then added, their positive and negative values will not be cancelled.



Linear regression determines the straight line, called the least-squares regression line or LSRL, that best expresses observations in a bivariate analysis of data set.

Suppose Y is a dependent variable, and X is an independent variable, then the population regression line is given by;

$Y = B_0 + B_1 X$, Where

B_0 is a constant

B_1 is the regression coefficient

If a random sample of observations is given, then the regression line is expressed by;

$\hat{y} = b_0 + b_1 x$

where b_0 is a constant, b_1 is the regression coefficient, x is the independent variable, and \hat{y} is the predicted value of the dependent variable.

In the linear regression line, we have seen the equation is given by;

$Y = B_0 + B_1 X$, Where

B_0 is a constant

B_1 is the regression coefficient

Now, let us see the formula to find the value of the regression coefficient.

$$B_1 = b_1 = \frac{\sum [(x_i - \bar{x})(y_i - \bar{y})]}{\sum (x_i - \bar{x})^2}$$

Where x_i and y_i are the observed data sets.

And x and y are the mean value.

Data Analytics

Student Material

Nº i/o	X	Y	\hat{y}_x	$y - \hat{y}_x$
1	1	4,24	4,28	-0,04
2	2	5,0	5,01	-0,01
3	3	5,9	5,74	0,16
4	4	6,49	6,47	0,02
5	5	7,09	7,2	-0,11
Total	15	28,72	28,7	0,02
Average	3	5,74	5,74	0,004

Here and below: X – set number, Y – parameters.
number of errors, \hat{y}_x , $y - \hat{y}_x$ – additional values for finding linear regression

$$\begin{aligned}\hat{y}_1 &= 3,55 + 0,73 \cdot 1 = 4,28; \\ \hat{y}_2 &= 3,55 + 0,73 \cdot 2 = 5,01; \\ \hat{y}_3 &= 3,55 + 0,73 \cdot 3 = 5,74; \\ \hat{y}_4 &= 3,55 + 0,73 \cdot 4 = 6,47; \\ \hat{y}_5 &= 3,55 + 0,73 \cdot 5 = 7,2.\end{aligned}$$

6.4 EQUATION FOR THE BEST STRAIGHT LINE

A line of best fit is a straight line that is the best approximation of the given set of data.

It is used to study the nature of the relation between two variables. (We're only considering the two-dimensional case, here.)

A line of best fit can be roughly determined using an eyeball method by drawing a straight line on a scatter plot so that the number of points above the line and below the line is about equal (and the line passes through as many points as possible).

A more accurate way of finding the line of best fit is the least square method .

Use the following steps to find the equation of line of best fit for a set of ordered pairs $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$.

Step 1: Calculate the mean of the xx -values and the mean of the yy -values.

$$X^{\text{---}} = \sum_{i=1}^n x_i / n = \bar{x}$$

Step 2: The following formula gives the slope of the line of best fit:

$$m = \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) / \sum_{i=1}^n (x_i - \bar{x})^2$$

Step 3: Compute the yy -intercept of the line by using the formula:

$$b = \bar{y} - m\bar{x}$$

Step 4: Use the slope mm and the yy -intercept bb to form the equation of the line.

Example:

Use the least square method to determine the equation of line of best fit for the data. Then plot the line.

xx	88	22	1111	66	55	44	1212	99	66	11
----	----	----	------	----	----	----	------	----	----	----

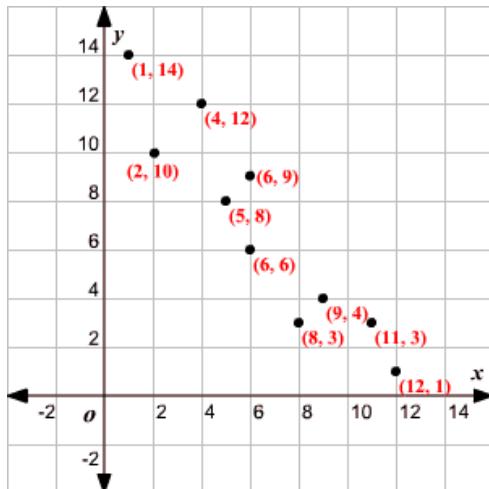
Data Analytics

Student Material

yy	33	1010	33	66	88	1212	11	44	99	1414
----	----	------	----	----	----	------	----	----	----	------

Solution:

Plot the points on a coordinate plane.



Calculate the means of the xx -values and the yy -values.

$$X^{\text{---}} = 8 + 2 + 11 + 6 + 5 + 4 + 12 + 9 + 6 + 110 = 6.4 \quad Y^{\text{---}} = 3 + 10 + 3 + 6 + 8 + 12 + 1 + 4 + 9 + 1410 =$$

$$7X^{\text{---}} = 8 + 2 + 11 + 6 + 5 + 4 + 12 + 9 + 6 + 110 = 6.4 \quad Y^{\text{---}} = 3 + 10 + 3 + 6 + 8 + 12 + 1 + 4 + 9 + 1410 = 7$$

Now calculate $xi - X^{\text{---}}$, $yi - Y^{\text{---}}$, $(xi - X^{\text{---}})(yi - Y^{\text{---}})$, and $(xi - X^{\text{---}})^2$ for each ii .

ii	x_i	y_i	$xi - X^{\text{---}}$	$yi - Y^{\text{---}}$	$(xi - X^{\text{---}})(yi - Y^{\text{---}})$	$(xi - X^{\text{---}})^2$
11	88	33	1.6	1.6	-4-4	-6.4-6.4
22	22	1010	-4.4	-4.4	33	-13.2-13.2
33	1111	33	4.6	4.6	-4-4	-18.4-18.4
44	66	66	-0.4	-0.4	-1-1	0.40.4
55	55	88	-1.4	-1.4	11	-1.4-1.4

Data Analytics

Student Material

66	44	1212	-2.4–2.4	55	-12–12	5.765.76
77	1212	11	5.65.6	-6–6	-33.6–33.6	31.3631.36
88	99	44	2.62.6	-3–3	-7.8–7.8	6.766.76
99	66	99	-0.4–0.4	22	-0.8–0.8	0.160.16
10 10	11	1414	-5.4–5.4	77	-37.8–37.8	29.1629.16
					$\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) = -131$ $n(\bar{x})(\bar{y}) = -131$	$\sum_{i=1}^n (x_i - \bar{x})^2 = 118$ $.4 \sum_{i=1}^n (x_i - \bar{x})^2 = 118$.4

Calculate the slope.

$$m = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{-131}{118} \approx -1.1$$

Calculate the yy -intercept.

Use the formula to compute the yy -intercept.

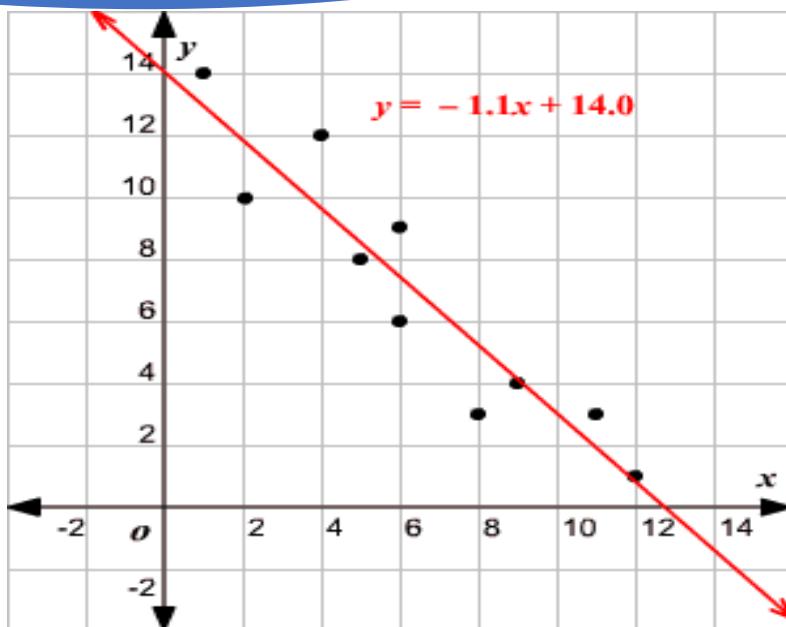
$$b = \bar{y} - m\bar{x} = 14.0 - (-1.1 \times 6.4) = 7 + 7.04 \approx 14.0$$

Use the slope and yy -intercept to form the equation of the line of best fit.

The slope of the line is -1.1 and the yy -intercept is 14.0.

Therefore, the equation is $y = -1.1x + 14.0$.

Draw the line on the scatter plot.

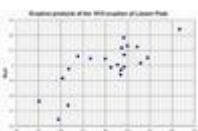


6.5 GRAPH OF THE BEST STRAIGHT LINE

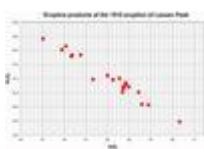
A best-fit line is meant to mimic the trend of the data. In many cases, the line may not pass through very many of the plotted points. Instead, the idea is to get a line that has equal numbers of points on either side. Most people start by eye-balling the data.

1. Take a look at the data and ask yourself these questions

Does the data look like a line? or a big blob? Try to approximate the general trend of the data with your mind (even if it's just a blob)



Does the trend of the points look positively



correlated (like they rise up to the right; click on image at right) or negatively correlated (like they start high near the x-axis and get lower as they approach the y-axis; see image to the left)? Your trendline (when you're finished with the next steps) should mimic those correlations.

If you blur your eyes, can you see a thick line trending in one direction or another? This is another way to visualize the trend of the data.

2. Now that you have an idea of the general trend of the data, there are two possible ways to construct a best-fit line by eye. You may use either of them; both are correct and

relatively easy ways to get a pretty accurate representation of a best-fit line. Pick the one that makes the most sense to you. The first method involves enclosing the data in an area:

The second method involves dividing data into two equal groups, approximating the center of each group and constructing a line between the two centers.

6.6 Interpreting the Results

Introduction

Once you've recorded the diagnostics you want to use, the next step is being able to understand what they say.

It's helpful to have a good understanding of what exactly each column in the query diagnostics schema means, which we're not going to repeat in this short tutorial.

In general, when building visualizations, it's better to use the full detailed table. Because regardless of how many rows it is, what you're probably looking at is some kind of depiction of how the time spent in different resources adds up, or what the native query emitted was.

As mentioned in our article on recording the diagnostics, I'm working with the OData and SQL traces for the same table (or nearly so)—the Customers table from Northwind. In particular, I'm going to focus on common ask from our customers, and one of the easier to interpret sets of traces: full refresh of the data model.

Building the visualizations

When you're going through traces, there are many ways you can evaluate them. In this article, we're going to focus on a two visualization split—one to show the details you care about, and the other to easily look at time contributions of various factors. For the first visualization, a table is used. You can pick any fields you like, but the ones recommended for an easy, high level look at what's going on are:

- Id
- Start Time
- Query
- Step
- Data Source Query
- Exclusive Duration (%)
- Row Count
- Category
- Is User Query
- Path

For the second visualization, one choice is to use a Stacked Column Chart. In the 'Axis' parameter, you might want to use 'Id' or 'Step'. If we're looking at the Refresh, because it doesn't have anything to do with steps in the Editor itself, we probably just want to look at 'Id'. For the 'Legend' parameter, you should set 'Category' or 'Operation' (depending on the granularity you want). For the 'Value', set 'Exclusive'

Data Analytics

Student Material

'Duration' (and make sure it's not the %, so that you get the raw duration value). Finally, for the Tooltip, set 'Earliest Start Time'.

Once your visualization is built, make sure you sort by 'Earliest Start Time' ascending so you can see the order things happen in.

Detailed Traces Table

Id	Start Time	Query	Step	Exclusive Duration (%)	Row Count	Category	Is User Query	Path	Data Source Query	
									A	B
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.01		Document Evaluator	False	0/1		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.00		Evaluation	False	0		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.08		Evaluator	False	0/2		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.00		Evaluator	False	0/3		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.01		Evaluator	False	0/3/4		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.00		Evaluator	False	0/3/5		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.03		Evaluator	False	0/3/5/6		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.03		Evaluator	False	0/3/5/7		
2.2	10/15/2019 12:28:13 PM	Customers (2)	Grouped Rows	0.00		Data Source	False	0/3/5/7/8/9/10		

Exclusive Duration and Earliest Start Time by Id and Category



While your exact needs might differ, this combination of charts is a good place to start for looking at numerous diagnostics files and for a number of purposes.

Interpreting the visualizations

As mentioned above, there's many questions you can try to answer with query diagnostics, but the two that we see the most often are asking how time is spent, and asking what the query sent to the source is.

Asking how the time is spent is easy, and will be similar for most connectors. A warning with query diagnostics, as mentioned elsewhere, is that you'll see drastically different capabilities depending on the connector. For example, many ODBC based connectors won't have an accurate recording of what query is sent to the actual back-end system, as Power Query only sees what it sends to the ODBC driver.

If we want to see how the time is spent, we can just look at the visualizations we built above.

Now, because the time values for the sample queries we're using here are so small, if we want to work with how Power BI reports time it's better if we convert the Exclusive Duration column to 'Seconds' in the Power Query editor. Once we do this this conversion, we can look at our chart and get a decent idea of where time is spent.

For my OData results, I see in the image that the vast majority of the time was spent retrieving the data from source—if I select the 'Data Source' item on the legend, it shows me all of the different operations related to sending a query to the Data Source.

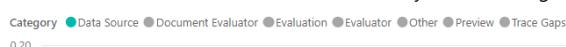
Data Analytics

Student Material

Detailed Traces Table

Exclusive Duration (%)	Path	Data Source Query
0.40	0/3/13/14/16	<p>Request:</p> <pre>GET https://services.odata.org/V4/Northwind/Northwind.svc/Customers?\$filter=ContactTitle eq 'Sales Representative'&\$select=CustomerID%20</pre> <p>Response:</p> <pre>https://services.odata.org/V4/Northwind/Northwind.svc/Customers?\$filter=ContactTitle eq 'Sales Representative'&\$select=CustomerID%20 HTTP/1.1 200 OK</pre>
0.35	0/3/5/7/8/9	<p>Request:</p> <pre>https://services.odata.org/V4/Northwind/Northwind.svc/ HTTP/1.1</pre> <p>Content-Type:</p> <pre>application/json;odata.metadata=minimal;q=1.0,application/json;odata=minimalmetadata;q=0.9,application/atomsvc+xml;q=0.8,application/av</pre>

Exclusive Duration and Earliest Start Time by Id and Category

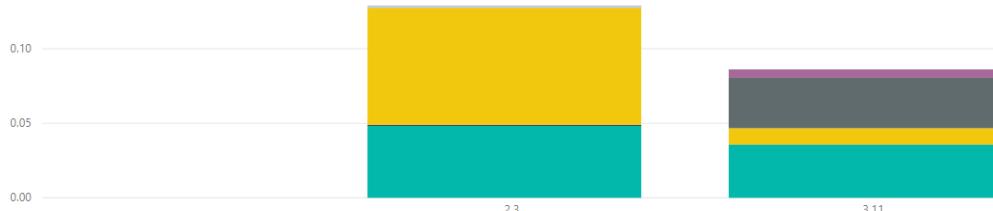


If we perform all the same operations and build similar visualizations, but with the SQL traces instead of the ODATA ones, we can see how the two data sources compare!

Detailed Traces Table

Id	Start Time	Query	Step	Exclusive Duration (%)	Row Count	Category	Is User Query	Path	Data Source Query
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.01		Document Evaluator	False	0/1	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.00		Evaluation	False	0	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.09		Evaluator	False	0/2	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.00		Evaluator	False	0/3	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.00		Evaluator	False	0/3/4	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.00		Evaluator	False	0/3/5	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.06		Evaluator	False	0/3/5/6	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.44		Evaluator	False	0/3/5/7	
2.3	10/15/2019 12:44:37 PM	Customers (3)	Grouped Rows	0.00		Data Source	False	0/3/5/7/8	

Exclusive Duration and Earliest Start Time by Id and Category



If we select the Data Source table, like with the ODATA diagnostics we can see the first evaluation (2.3 in this image) emits metadata queries, with the second evaluation actually retrieving the data we care about. Because we're retrieving small amounts of data in this case, the data pulled back takes a small amount of time (less than a tenth of a second for the entire second evaluation to happen, with less than a twentieth of a second for data retrieval itself), but that won't be true in all cases.

6.7 MEASURES OF VARIATION

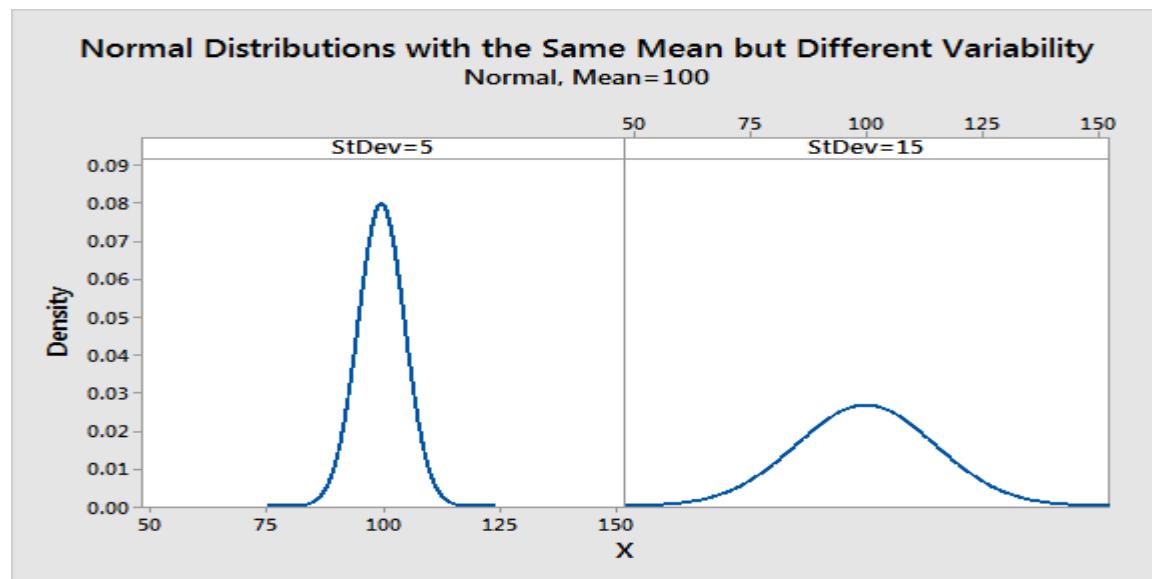
Data Analytics

Student Material

A measure of variability is a summary statistic that represents the amount of dispersion in a dataset. How spread out are the values? While a measure of central tendency describes the typical value, measures of variability define how far away the data points tend to fall from the center. We talk about variability in the context of a distribution of values. A low dispersion indicates that the data points tend to be clustered tightly around the center. High dispersion signifies that they tend to fall further away.

In statistics, variability, dispersion, and spread are synonyms that denote the width of the distribution. Just as there are multiple measures of central tendency, there are several measures of variability. In this blog post, you'll learn why understanding the variability of your data is critical. Then, I explore the most common measures of variability—the range, interquartile range, variance, and standard deviation. I'll help you determine which one is best for your data.

The two plots below show the difference graphically for distributions with the same mean but more and less dispersion. The panel on the left shows a distribution that is tightly clustered around the average, while the distribution in the right panel is more spread out.

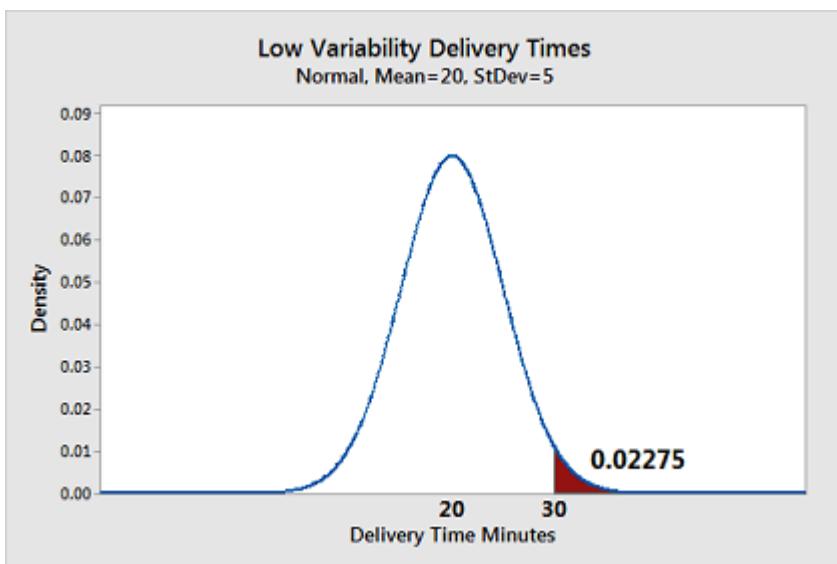
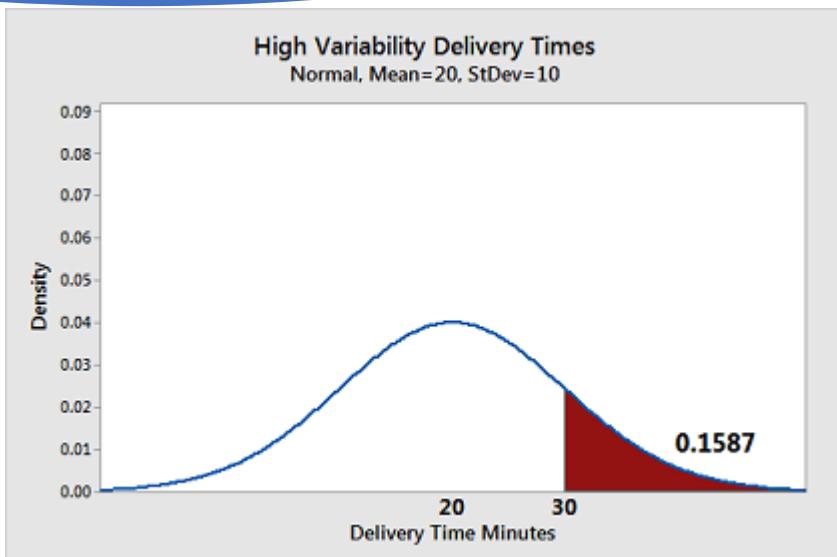


Example of Different Amounts of Variability

Let's take a look at two hypothetical pizza restaurants. They both advertise a mean delivery time of 20 minutes. When we're ravenous, they both sound equally good! However, this equivalence can be deceptive! To determine the restaurant that you should order from when you're hungry, we need to analyze their variability.

Suppose we study their delivery times, calculate the variability for each place, and determine that their variabilities are different. We've computed the standard deviations for both restaurants—which is a measure that we'll come back to later in this post. How significant is this difference in getting pizza to their customers promptly?

The graphs below display the distribution of delivery times and provide the answer. The restaurant with more variable delivery times has the broader distribution curve. I've used the same scales in both graphs so you can visually compare the two distributions.



In these graphs, we consider a 30-minute wait or longer to be unacceptable. We're hungry after all! The shaded area in each chart represents the proportion of delivery times that surpass 30 minutes. Nearly 16% of the deliveries for the high variability restaurant exceed 30 minutes. On the other hand, only 2% of the deliveries take too long with the low variability restaurant. They both have an average delivery time of 20 minutes, but I know where I'd place my order when I'm hungry!

As this example shows, the central tendency doesn't provide complete information. We also need to understand the variability around the middle of the distribution to get the full picture. Now, let's move on to the different ways of measuring variability!

Range

Let's start with the range because it is the most straightforward measure of variability to calculate and the simplest to understand. The range of a dataset is the difference between

the largest and smallest values in that dataset. For example, in the two datasets below, dataset 1 has a range of $20 - 38 = 18$ while dataset 2 has a range of $11 - 52 = 41$. Dataset 2 has a broader range and, hence, more variability than dataset 1.

Dataset 1	Dataset 2
20	11
21	16
22	19
25	23
26	25
29	32
33	39
34	46
38	52

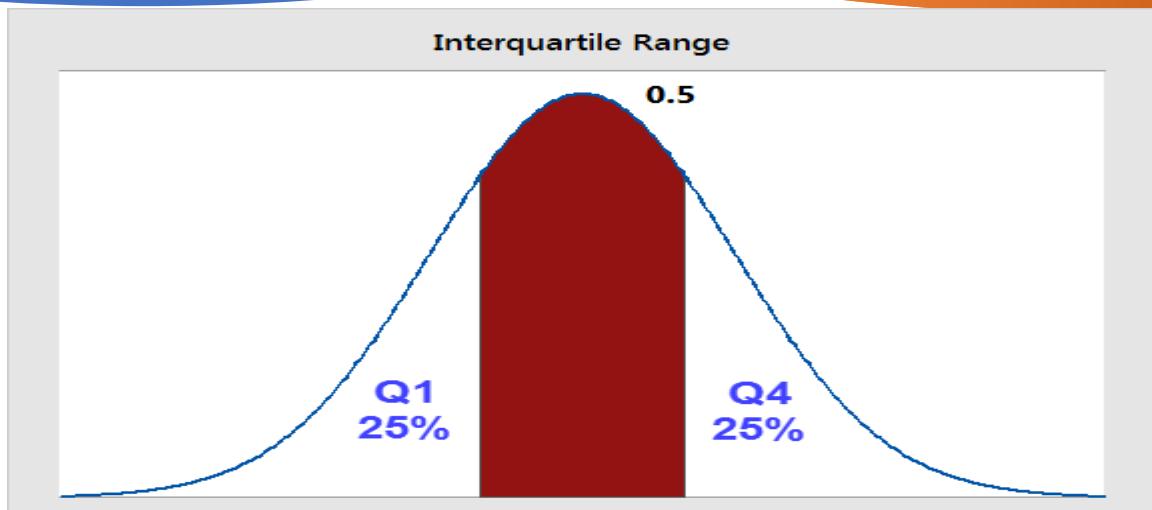
While the range is easy to understand, it is based on only the two most extreme values in the dataset, which makes it very susceptible to outliers. If one of those numbers is unusually high or low, it affects the entire range even if it is atypical.

Additionally, the size of the dataset affects the range. In general, you are less likely to observe extreme values. However, as you increase the sample size, you have more opportunities to obtain these extreme values. Consequently, when you draw random samples from the same population, the range tends to increase as the sample size increases. Consequently, use the range to compare variability only when the sample sizes are similar.

The Interquartile Range (IQR) . . . and other Percentiles

The interquartile range is the middle half of the data. To visualize it, think about the median value that splits the dataset in half. Similarly, you can divide the data into quarters. Statisticians refer to these quarters as quartiles and denote them from low to high as Q1, Q2, and Q3. The lowest quartile (Q1) contains the quarter of the dataset with the smallest values. The upper quartile (Q4) contains the quarter of the dataset with the highest values. The interquartile range is the middle half of the data that is in between the upper and lower

quartiles. In other words, the interquartile range includes the 50% of data points that fall between Q1 and Q3. The IQR is the red area in the graph below.



The interquartile range is a robust measure of variability in a similar manner that the median is a robust measure of central tendency. Neither measure is influenced dramatically by outliers because they don't depend on every value. Additionally, the interquartile range is excellent for skewed distributions, just like the median. As you'll learn, when you have a normal distribution, the standard deviation tells you the percentage of observations that fall specific distances from the mean. However, this doesn't work for skewed distributions, and the IQR is a great alternative.

I've divided the dataset below into quartiles. The interquartile range (IQR) extends from the low end of Q2 to the upper limit of Q3. For this dataset, the range is $39 - 20 = 19$.

IQR
11
13
16
19
20
21
23
25
26
29
33
34
36
38
39
46
52
55
58

Using other percentiles

When you have a skewed distribution, I find that reporting the median with the interquartile range is a particularly good combination. The interquartile range is equivalent to the region

between the 75th and 25th percentile ($75 - 25 = 50\%$ of the data). You can also use other percentiles to determine the spread of different proportions. For example, the range between the 97.5th percentile and the 2.5th percentile covers 95% of the data. The broader these ranges, the higher the variability in your dataset.

Variance

Variance is the average squared difference of the values from the mean. Unlike the previous measures of variability, the variance includes all values in the calculation by comparing each value to the mean. To calculate this statistic, you calculate a set of squared differences between the data points and the mean, sum them, and then divide by the number of observations. Hence, it's the average squared difference.

There are two formulas for the variance depending on whether you are calculating the variance for an entire population or using a sample to estimate the population variance. The equations are below, and then I work through an example in a table to help bring it to life.

Population variance

The formula for the variance of an entire population is the following:

$$\sigma^2 = \frac{\sum(X-\mu)^2}{N}$$

In the equation, σ^2 is the population parameter for the variance, μ is the parameter for the population mean, and N is the number of data points, which should include the entire population.

Sample variance

To use a sample to estimate the variance for a population, use the following formula. Using the previous equation with sample data tends to underestimate the variability. Because it's usually impossible to measure an entire population, statisticians use the equation for sample variances much more frequently.

$$s^2 = \frac{\sum(X-M)^2}{N-1}$$

In the equation, s^2 is the sample variance, and M is the sample mean. $N-1$ in the denominator corrects for the tendency of a sample to underestimate the population variance.

Standard Deviation

The standard deviation is the standard or typical difference between each data point and the mean. When the values in a dataset are grouped closer together, you have a smaller standard deviation. On the other hand, when the values are spread out more, the standard deviation is larger because the standard distance is greater.

Data Analytics

Student Material

Conveniently, the standard deviation uses the original units of the data, which makes interpretation easier. Consequently, the standard deviation is the most widely used measure of variability. For example, in the pizza delivery example, a standard deviation of 5 indicates that the typical delivery time is plus or minus 5 minutes from the mean. It's often reported along with the mean: 20 minutes (s.d. 5).

The standard deviation is just the square root of the variance. Recall that the variance is in squared units. Hence, the square root returns the value to the natural units. The symbol for the standard deviation as a population parameter is σ while s represents it as a sample estimate. To calculate the standard deviation, calculate the variance as shown above, and then take the square root of it. Voila! You have the standard deviation!

In the variance section, we calculated a variance of 201 in the table.

Therefore, the standard deviation for that dataset is 14.177.

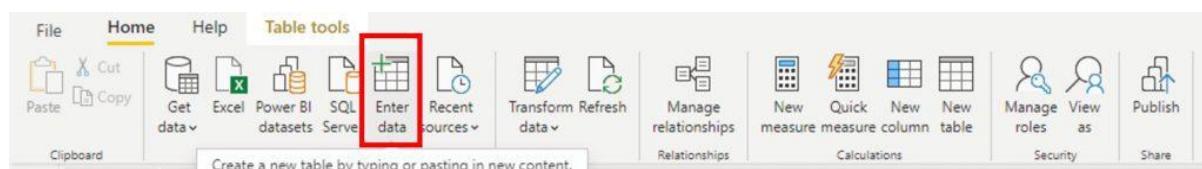
The standard deviation is similar to the mean absolute deviation. Both use the original data units and they compare the data values to mean to assess variability. However, there are differences.

Assignment:

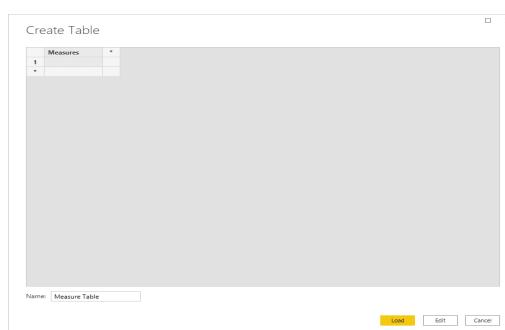
Create Measure Table using Power BI.

Solution:

In the Home tab of the Desktop ribbon, click **Enter Data**. This will pop up a screen that allows you to manually enter data. You can actually paste in data that you copied from Excel to this screen, but we're not going to use that functionality right now.

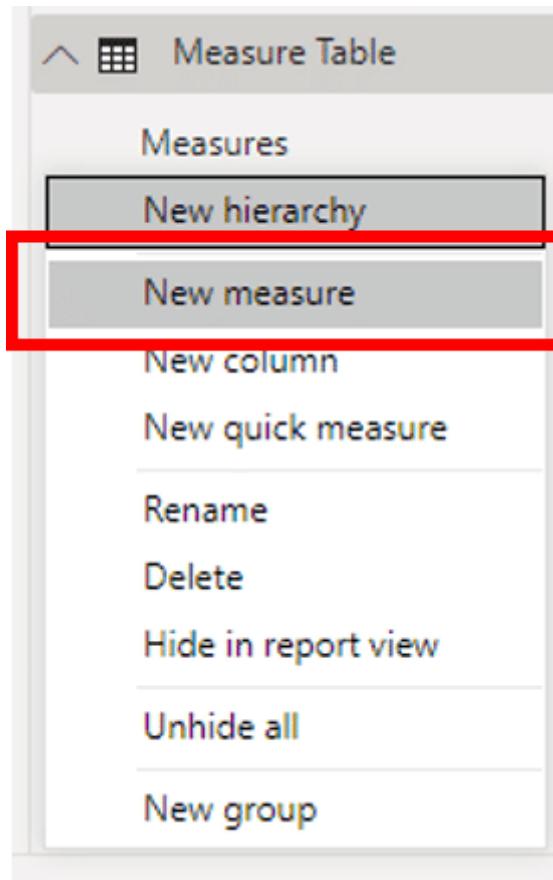


For now, we are going to rename the column to Measures and then rename the table name at the bottom to Measure Table.

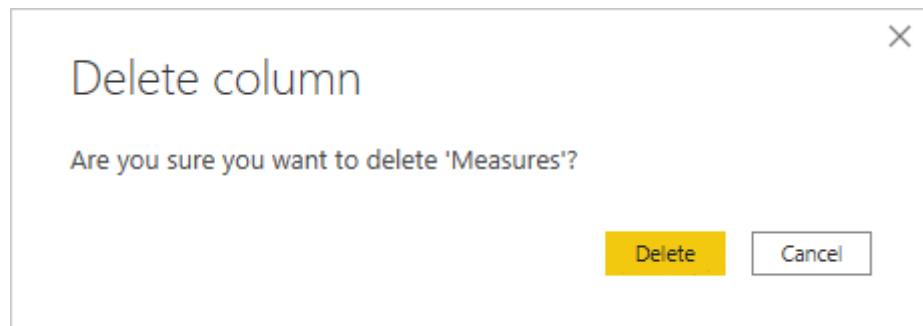


Now you have a new table with one blank measure in it named Measures. Right click in your new table and click **New measure**. Enter in the calculation that you want to add to your report.

All measures that you create under this Measure Table can reference other tables in your data model. Since measures aren't actually adding data to your model, only the measure itself lives here allowing you to separate out your measures from your other tables.



we named the column Measures, so go ahead and delete that Measures column from the Measure Table.



now you have a table specifically for storing your measures separate from the other tables in your data model.

The screenshot shows the 'Fields' pane in Power BI Desktop. At the top is a search bar with a magnifying glass icon. Below it is a tree view of data assets:

- 1. Measure Table
 - Avg Price of Order Measure
- Category Totals DIM
- Dates Table
- Distributor DIM
- Grocery FACT
- Grocery FACT (2)
- Measure Names

We do it because it adds simplicity to our interactions with Power BI Desktop. This way, you don't have to scroll through seven tables and hundreds of columns to find measures. While we could easily use the Search field, this approach makes our model more organized.

On the flip side, we understand how people would like the measures to be in the tables that they refer to. This would help people understand what tables measures were referring to without looking through the DAX.

Practice task:

How to find Linear regression in Power BI?

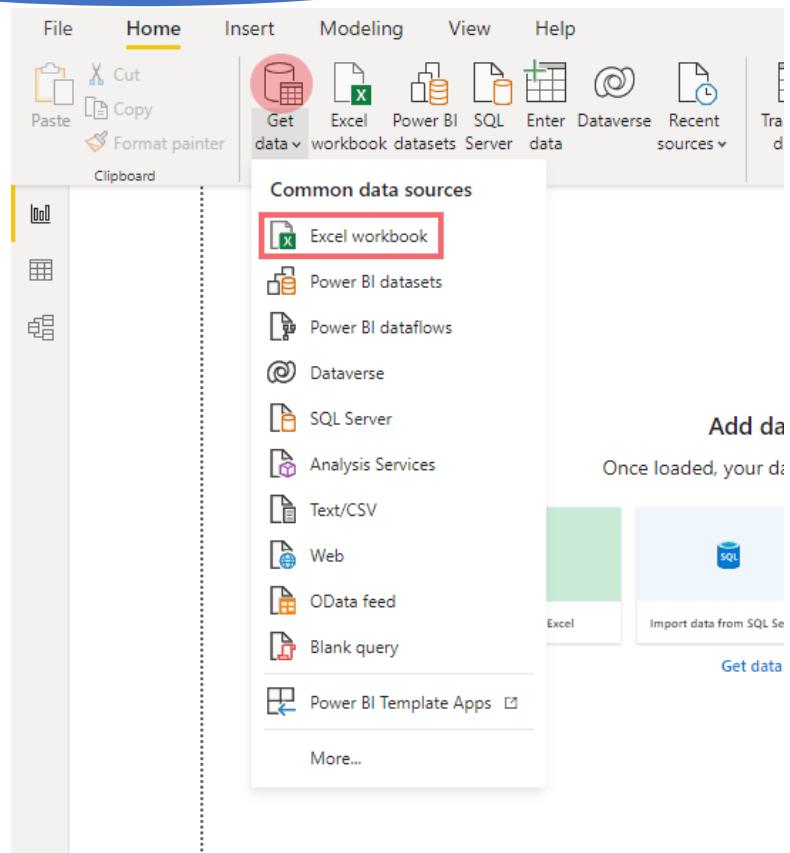
Step1:

The first step is to get your dataset into Power BI.

Dataset can be on SQL Server, Oracle, Power BI datasets, Power BI dataflows, MySQL database, Text/CSV, PDF, Access, XML, JSON, or on any other source. Our is in Excel. So, referring to that, go and click on the Get Data tab and select Excel workbook.

Data Analytics

Student Material

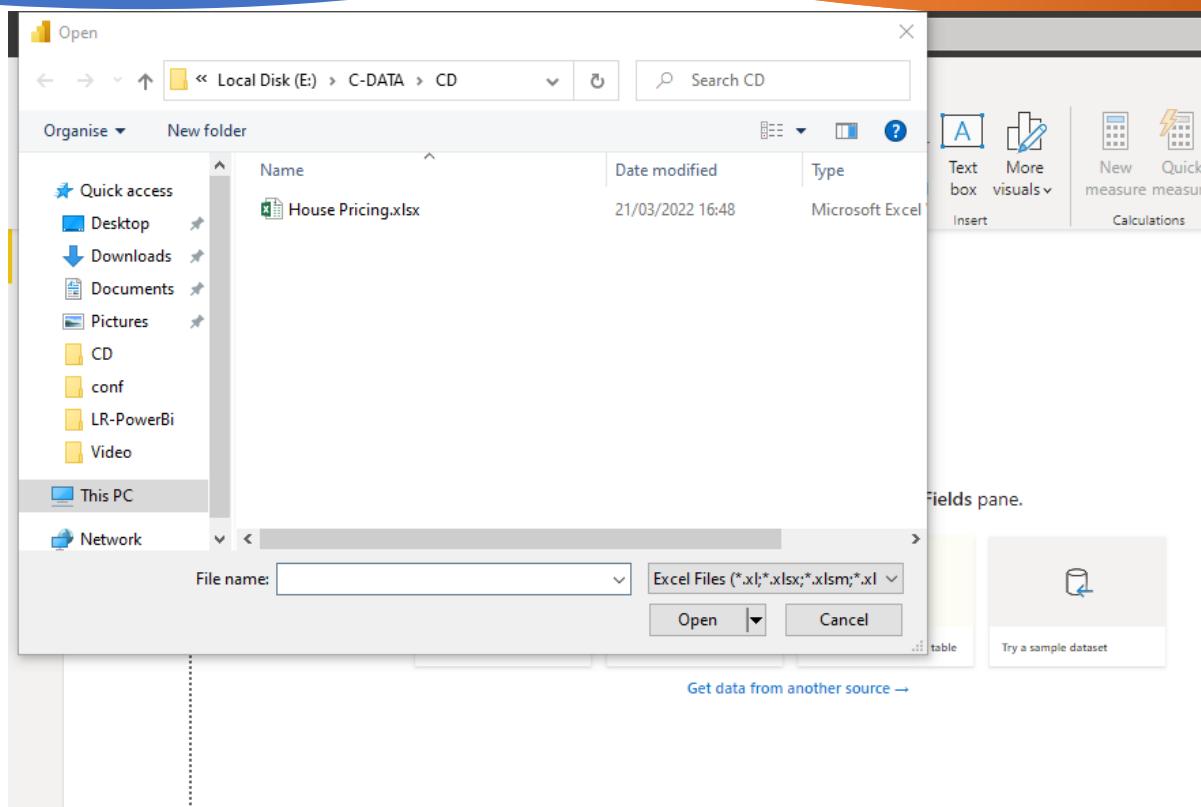


Dataset Selection

A dialogue box will open. Reach your destination, **select, and open your Excel file** which in this example is “House Pricing.xlsx”.

Data Analytics

Student Material



Dataset Selection in Power BI

Now **select a sheet** in your workbook having your dataset and load the dataset into Power BI.

Area in Square Feet	Price in USD	Column3	Column4	Column5	Column6
2900	102000	null	null	null	
7200	744000	null	null	null	
2300	321000	null	null	null	
8300	879000	null	null	null	
9600	1070000	null	null	null	
4200	336000	null	null	null	
4900	563000	null	null	null	
7900	732000	null	null	null	
4400	424000	null	null	null	
700	142000	null	null	null	
5400	525000	null	null	null	
5200	442000	null	null	null	
4200	437000	null	null	null	
2900	58000	null	null	null	
2400	352000	null	null	null	
5400	535000	null	null	null	
1600	87000	null	null	null	
1300	149000	null	null	null	
4900	606000	null	null	null	
3700	429000	null	null	null	
7800	740000	null	null	null	
3900	342000	null	null	null	
9100	864000	null	null	null	

Sheet selection

Step 2: Creating Scatter Plot for Linear Regression

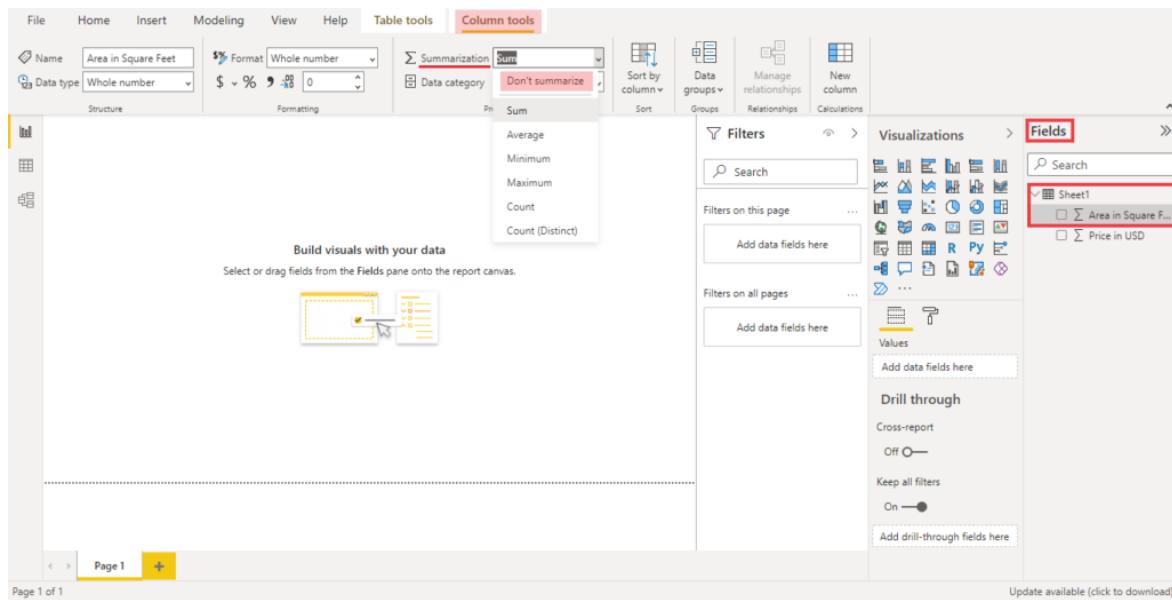
Now dataset has been loaded into Power BI and the second step would be to design a scatter plot for linear regression analysis. For this, Go into **Fields**, click on your **sheet**, and then on a **column** to select it and set a measure. In

Data Analytics

Student Material

this example, we first selected the column “Area in Square Feet”.

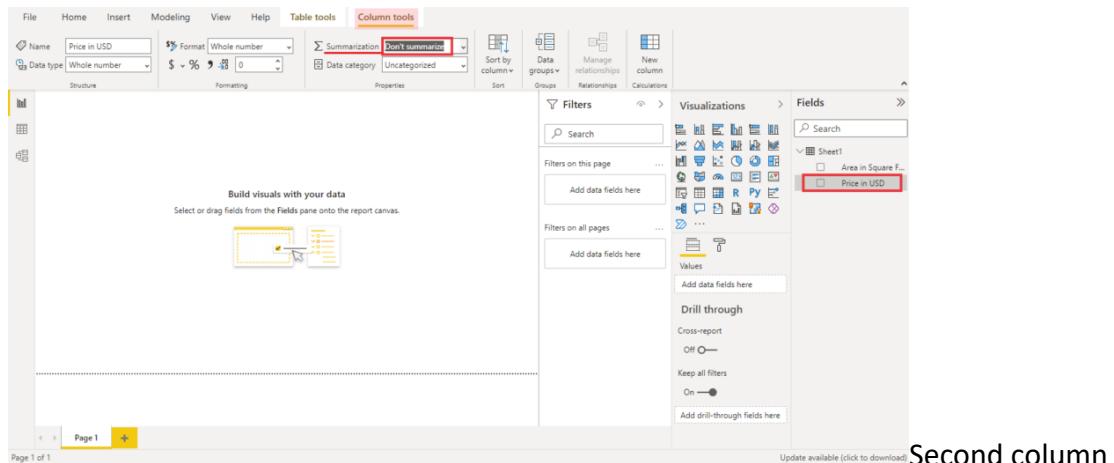
After selecting a column, go to the “**Column tools**” tab and then to “**Properties**” in the ribbon. After that, set the measure for “**Summarization**” from “sum” to “**Don’t Summarize**”.



The screenshot shows the Microsoft Power BI desktop interface. The ribbon at the top has the 'Column tools' tab selected. In the 'Structure' section, the 'Name' is 'Area in Square Feet' and 'Data type' is 'Whole number'. The 'Formatting' section shows a 'Format' dropdown set to 'Whole number'. The 'Summarization' dropdown is highlighted with a red box and set to 'Don't summarize'. The 'Fields' pane on the right shows the 'Sheet1' node expanded, with 'Area in Square Feet' selected. The report canvas below is empty, and the status bar at the bottom left says 'Page 1 of 1'.

First column selection and setting its measure

Repeat this step for your other variable. In this case, the other column i.e., “Price in USD” represents the other variable. We selected that column and set the measure to “Don’t Summarize” for this too.



The screenshot shows the Microsoft Power BI desktop interface. The ribbon at the top has the 'Column tools' tab selected. In the 'Structure' section, the 'Name' is 'Price in USD' and 'Data type' is 'Whole number'. The 'Formatting' section shows a 'Format' dropdown set to 'Whole number'. The 'Summarization' dropdown is highlighted with a red box and set to 'Don't summarize'. The 'Fields' pane on the right shows the 'Sheet1' node expanded, with 'Price in USD' selected. The report canvas below is empty, and the status bar at the bottom left says 'Page 1 of 1'.

selection and setting its measure

Now, we have selected our columns of variables and now we have to select for **SCATTER PLOT**. For this, go to the “**Visualizations**” and select “**SCATTER PLOT**”

Data Analytics

Student Material

The screenshot shows the Microsoft Power BI desktop interface. The ribbon at the top has tabs like File, Home, Insert, Modeling, View, Help, Format, Data / Drill, and others. The 'Home' tab is selected. In the center, there is a scatter plot with several gray circles of varying sizes. To the right of the plot is a 'Visualizations' pane with various chart icons, one of which is highlighted with a red box. Below the visualizations are sections for Filters, Fields, and Details. The 'Fields' section shows a search bar and a list of fields from a sheet named 'Sheet1', including 'Area in Square Feet' and 'Price in USD'. The 'Details' section includes options for X Axis, Y Axis, and Size.

Selecting scatter plot from visualizations

Now select your independent and dependent variable for your x and y-axis (in the plot) either from “**fields**” or from the “**Details**” section below the “**Visualizations**”. In this example, the factor “**Area**” is an independent variable and is set at the x-axis while the factor “**Price**” being the dependent variable is set on the y-axis.

This screenshot shows the same Power BI desktop interface as above, but with a linear regression trend line added to the scatter plot. The plot is titled 'Linear Regression House Area Vs Price'. The x-axis is labeled 'Area in Square Feet' and the y-axis is labeled 'Price in USD'. The 'Visualizations' pane on the right shows the selected chart type. The 'Details' section is highlighted with red boxes around the 'X Axis' and 'Y Axis' dropdowns, which are set to 'Area in Square Feet' and 'Price in USD' respectively. Other settings like 'Filters' and 'Fields' are also visible.

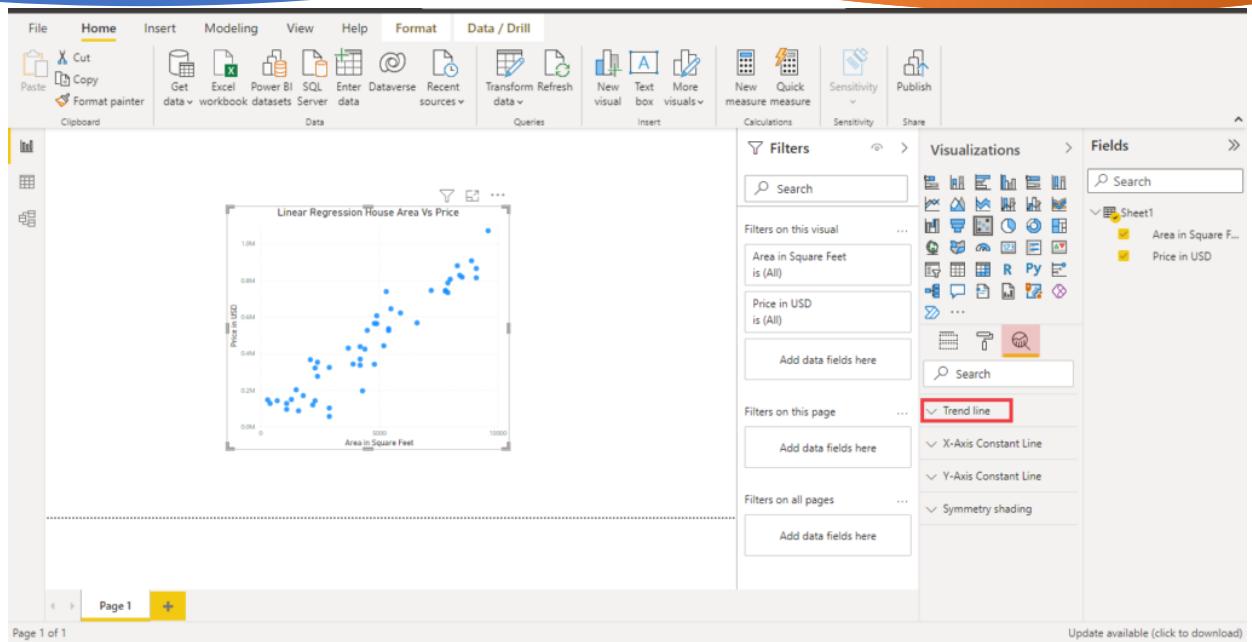
Specifying variables on the X and Y-axis

Step 3: Adding Trend Line in Scatter Plot for linear regression

After specifying, the x and y-axis, the next step is to add a trend line. Now, we are going to add a trend line to our plot. For this, click on “**Analytics**” and then on “**Trend Line**”.

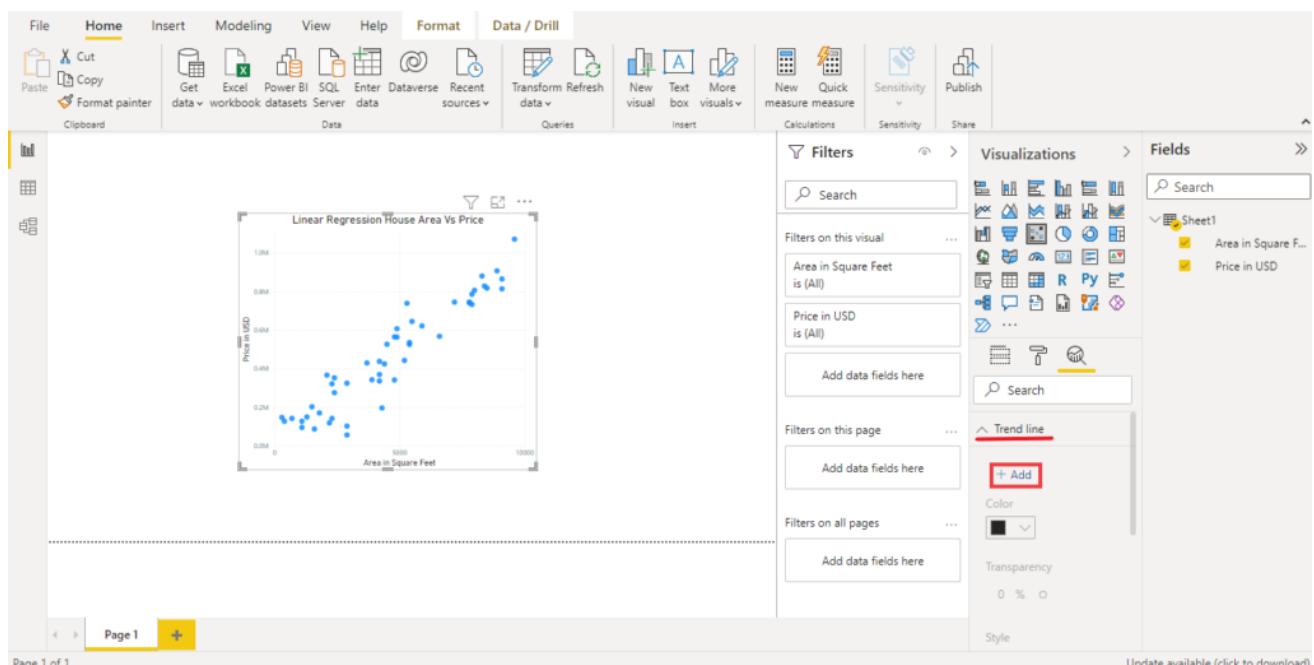
Data Analytics

Student Material



Adding trend line in scatter plot

In the “Trend Line” click on “ADD”

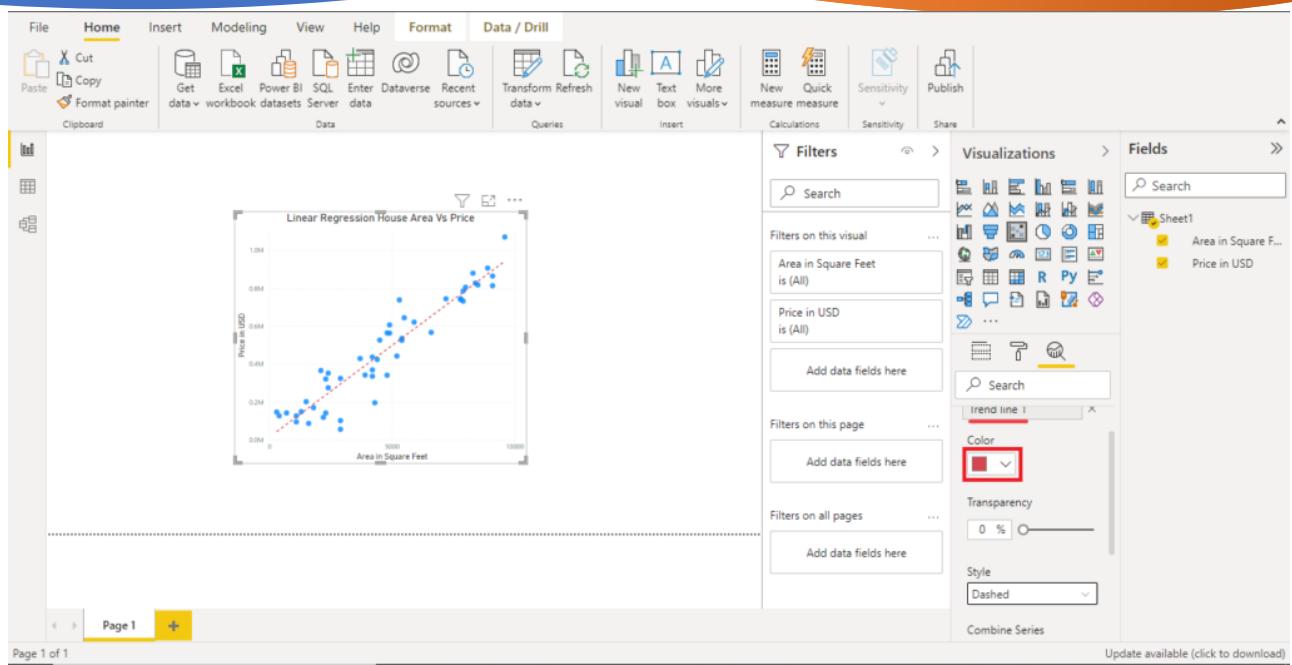


Setting trend line colour

And select the colour of your choice. In this example, we set the colour of the trend line to red.

Data Analytics

Student Material



Linear regression of House Area (in sq. ft) Vs Price (in USD)

This trend line is showing the relation i.e., the correlation between two values which is positive and implies as the independent variable i.e., Area of House (in ft^2) increases, the dependent variable i.e., Price (in USD), also increases.

Course Assignment:

Create a report from Excel file using Power BI desktop

Solution:

1st, download the sample:

You can download the sample workbook directly.

1. Download sample file from this link: [Financial sample Excel workbook](#).
2. Open Power BI Desktop.
3. In the **Data** section of the **Home** ribbon, select **Excel**.
4. Navigate to where you saved the sample workbook, and select **Open**.

Next, Prepare your data:

In **Navigator**, you have the option to *transform* or *load* the data. The Navigator provides a preview of your data so you can verify that you have the correct range of data. Numeric data types are italicized. If you need to make changes, transform your data before loading. To make the visualizations easier to read later, we do want to transform the data now. As you do each transformation, you see it added to the list under **Query Settings** in **Applied Steps**

1. Select the **Financials** table, and choose **Transform Data**.

The screenshot shows the Power BI Navigator window. On the left, there's a tree view of the 'Financial Sample.xlsx' file, with 'financials' checked. The main area displays the 'financials' table with the following data:

Segment	Country	Product	Discount (%)
Government	Canada	Carretera	Nc
Government	Germany	Carretera	Nc
Midmarket	France	Carretera	Nc
Midmarket	Germany	Carretera	Nc
Midmarket	Mexico	Carretera	Nc
Government	Germany	Carretera	Nc
Midmarket	Germany	Montana	Nc
Channel Partners	Canada	Montana	Nc
Government	France	Montana	Nc
Channel Partners	Germany	Montana	Nc
Midmarket	Mexico	Montana	Nc
Enterprise	Canada	Montana	Nc
Small Business	Mexico	Montana	Nc
Government	Germany	Montana	Nc

At the bottom, there are three buttons: 'Load' (yellow), 'Transform Data' (red box), and 'Cancel'.

Data Analytics

Student Material

2. Select the **Units Sold** column. On the **Transform** tab, select **Data Type**, then select **Whole Number**. Choose **Replace current** to change the column type.

The top data cleaning step users do most often is changing data types. In this case, the units sold are in decimal form. It doesn't make sense to have 0.2 or 0.5 of a unit sold, does it? So let's change that to whole number.

The screenshot shows a data analysis interface with a toolbar at the top. The 'Data Type' button is open, displaying a list of options: Decimal Number, Fixed decimal number, Whole Number, Percentage, Date/Time, Date, Time, Date/Time/Timezone, Duration, Text, True/False, and Binary. The 'Whole Number' option is highlighted with a red box. Below the toolbar, a table is visible with a yellow header row containing the text '(financials_Table,{{"Segment", t}'. The 'Units Sold' column is selected, showing values like 1618.5, 1321, 2178, 888, 2470, 1513, and 921.

(financials_Table,{{"Segment", t	1.2 Units Sold	123
	1618.5	
	1321	
	2178	
	888	
	2470	
	1513	
	921	

3. Select the **Segment** column. We want to make the segments easier to see in the chart later, so let's format the Segment column. On the **Transform** tab, select **Format**, then select **UPPERCASE**.

Data Analytics

Student Material

The screenshot shows the Microsoft Power BI Data Editor interface. The top navigation bar includes 'Transform', 'Add Column', 'View', 'Tools', and 'Help'. The 'Transform' tab is selected. Below the ribbon, there are several icons for data manipulation: 'Transpose', 'Reverse Rows', 'Count Rows', 'Data Type: Text' (set to 'Text'), 'Detect Data Type', 'Rename', 'Split Column' (selected), and 'Format'. A context menu is open over the 'Segment' column header, listing options: 'lowercase', 'UPPERCASE' (highlighted with a red box), 'Capitalize Each Word', 'Trim', 'Clean', 'Add Prefix', and 'Add Suffix'. The main data grid shows four rows of data under the 'Segment' column: GOVERNMENT, GOVERNMENT, MIDMARKET, and MIDMARKET.

- Let's shorten the column name from **Month Name** to just **Month**. Double-click the **Month Name** column, and rename to just **Month**.

The screenshot shows the Power BI Data Editor with a table containing four rows. The first column is labeled 'Month Name' and is highlighted with a red box. The second column contains numerical values (1, 1, 6, 6) and the third column contains month names ('January', 'January', 'June', 'June').

	Month Name	
1	January	
1	January	
6	June	
6	June	

- In the **Product** column, select the dropdown and clear the box next to **Montana**.

We know the Montana product was discontinued last month, so we want to filter this data from our report to avoid confusion.

The screenshot shows a software interface for data analysis. At the top, there's a toolbar with icons for sorting and filtering. A dropdown menu is open under the 'Product' column header, showing options like 'Sort Ascending', 'Sort Descending', and 'Clear Filter'. Below this is a 'Text Filters' section with a search bar labeled 'Search'. A list of categories is displayed, each with a checkbox. The categories are: '(Select All)', 'Amarilla' (checked), 'Carretera' (checked), 'Montana' (unchecked and highlighted with a red border), 'Paseo' (checked), 'Velo' (checked), and 'VTT' (checked). At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

6. You see that each transformation has been added to the list under **Query Settings** in **Applied Steps**.

Query Settings X

◀ **PROPERTIES**

Name
financials

[All Properties](#)

◀ **APPLIED STEPS**

- Source ⚙️
- Navigation ⚙️
- Changed Type
- Uppercased Text
- Renamed Columns
- Filtered Rows** ⚙️

7. Back on the **Home** tab, select **Close & Apply**. Our data is almost ready for building a report.

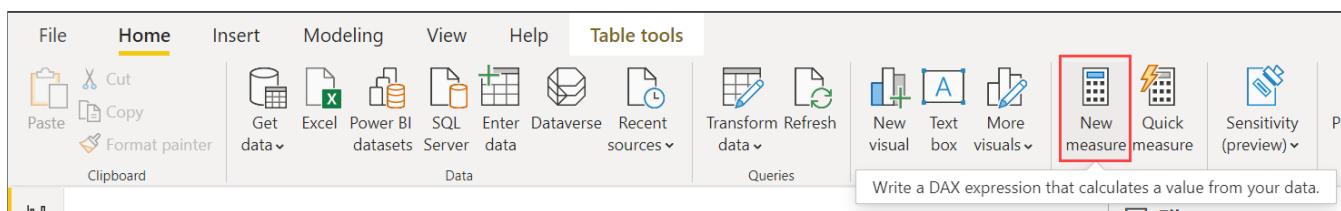
You see the Sigma symbol in the Fields list? Power BI has detected that those fields are numeric. Power BI also indicates the date field with a calendar symbol.

The screenshot shows the 'Fields' pane in Power BI. A search bar at the top contains the text 'Search'. Below it, a tree view shows a folder named 'financials' expanded. Inside, several items are listed: 'Sales' (highlighted with a red box), 'COGS' (highlighted with a red box), 'Country', 'Date' (highlighted with a red box), 'Discount Ba...', 'Discounts' (highlighted with a red box), 'Gross Sales' (highlighted with a red box), 'Manufactur...' (highlighted with a red box), and 'Month'.

Extra credit: Write an expression in DAX

Writing *measures* and creating *tables* in the *DAX* formula language is super powerful for data modeling. There's lots to learn about DAX in the Power BI documentation. For now, let's write a basic expression and join two tables.

1. On the **Home** ribbon, select **New measure**.



2. Type this expression to add all the numbers in the Units Sold column.

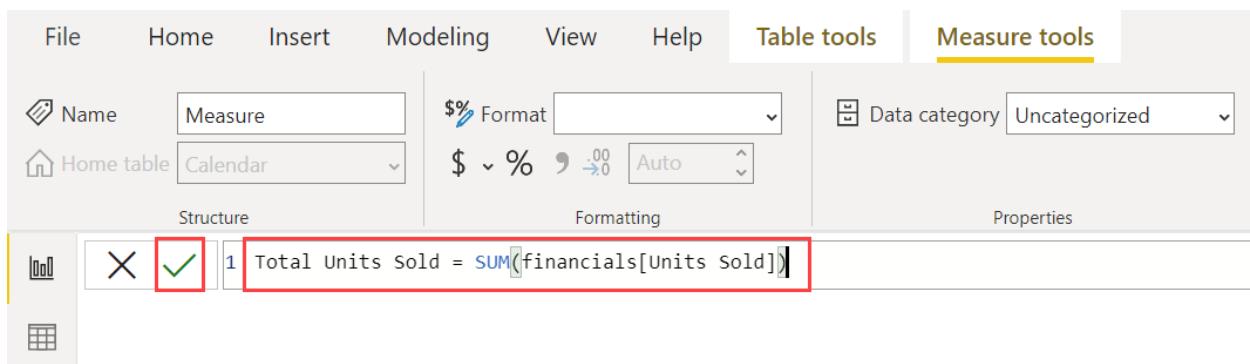
DAX:

Total Units Sold = SUM(financials[Units Sold])

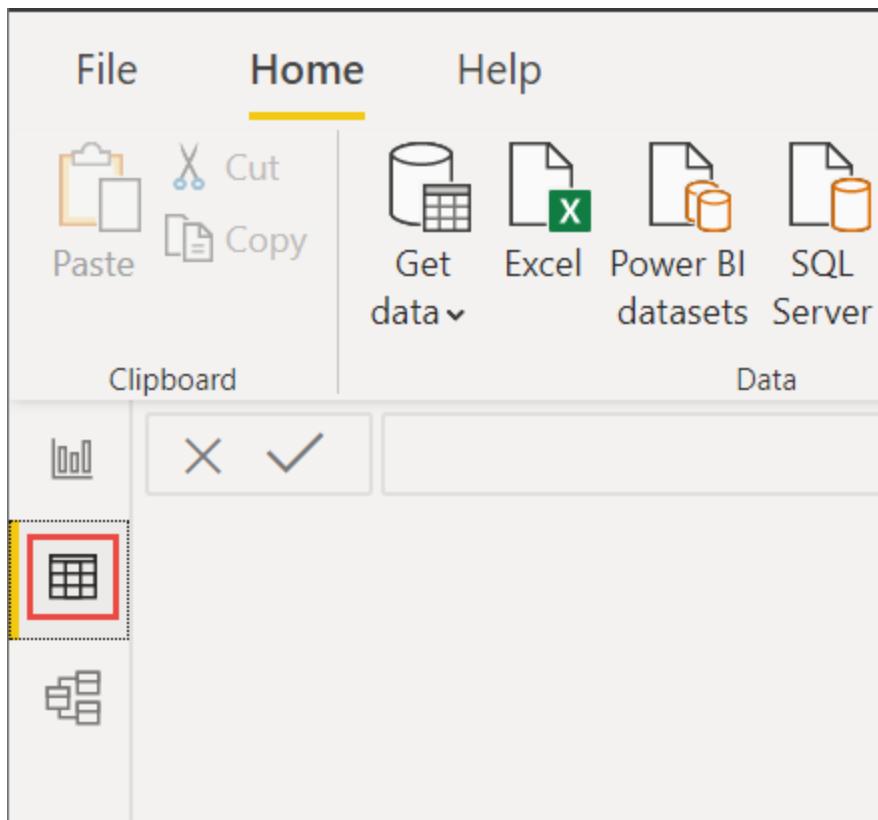
Data Analytics

Student Material

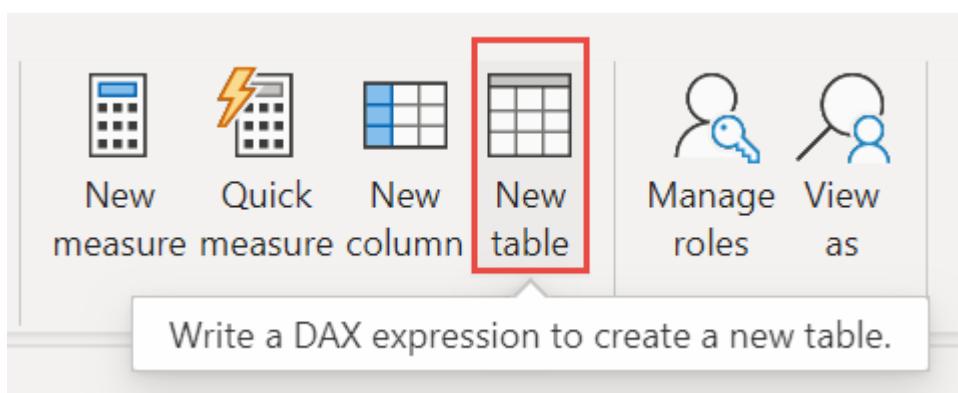
3. Select the check mark to commit.



4. Now select the **Data** view on the left.



5. On the **Home** ribbon, select **New table**.



Data Analytics

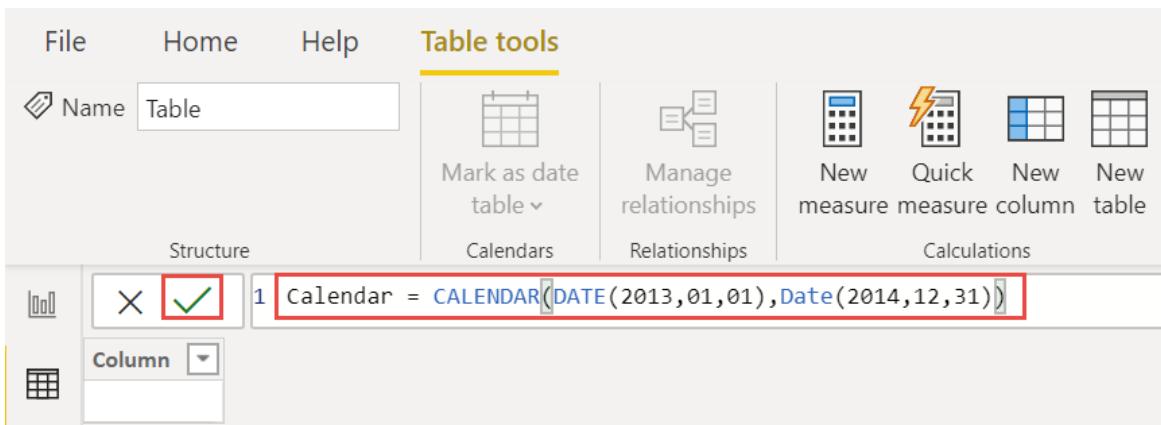
Student Material

- Type this expression to generate a Calendar table of all dates between January 1, 2013, and December 31, 2014.

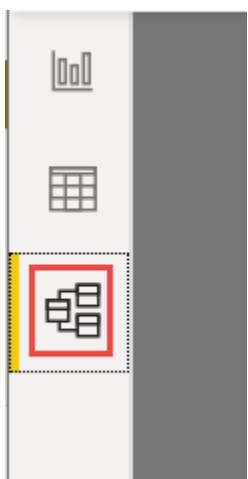
DAX:

```
Calendar = CALENDAR(DATE(2013,01,01),Date(2014,12,31))
```

- Select the check mark to commit.



- Now select **Model** view on the left.



- Drag the **Date** field from the financials table to the **Date** field in the Calendar table to join the tables, and create a *relationship* between them.

Data Analytics

Student Material

The screenshot shows the Power BI Fields pane on the left and a Calendar pane on the right. The Fields pane lists various financial metrics like Sales, COGS, and Manufacturing Price, along with Date and Month fields. A relationship is established between the Date field in the Fields pane and the Date field in the Calendar pane, indicated by a yellow line and a number '1'.

Now the main part, Build your report

Now that you've transformed and loaded your data, it's time to create your report. In the Fields pane on the right, you see the fields in the data model you created.

The final report, one visual at a time will look like this following image



Data Analytics

Student Material

Visual 1: Add a title

1. On the **Insert** ribbon, select **Text Box**. Type “Executive Summary – Finance Report”.
2. Select the text you typed. Set the **Font Size** to 20 and **Bold**.



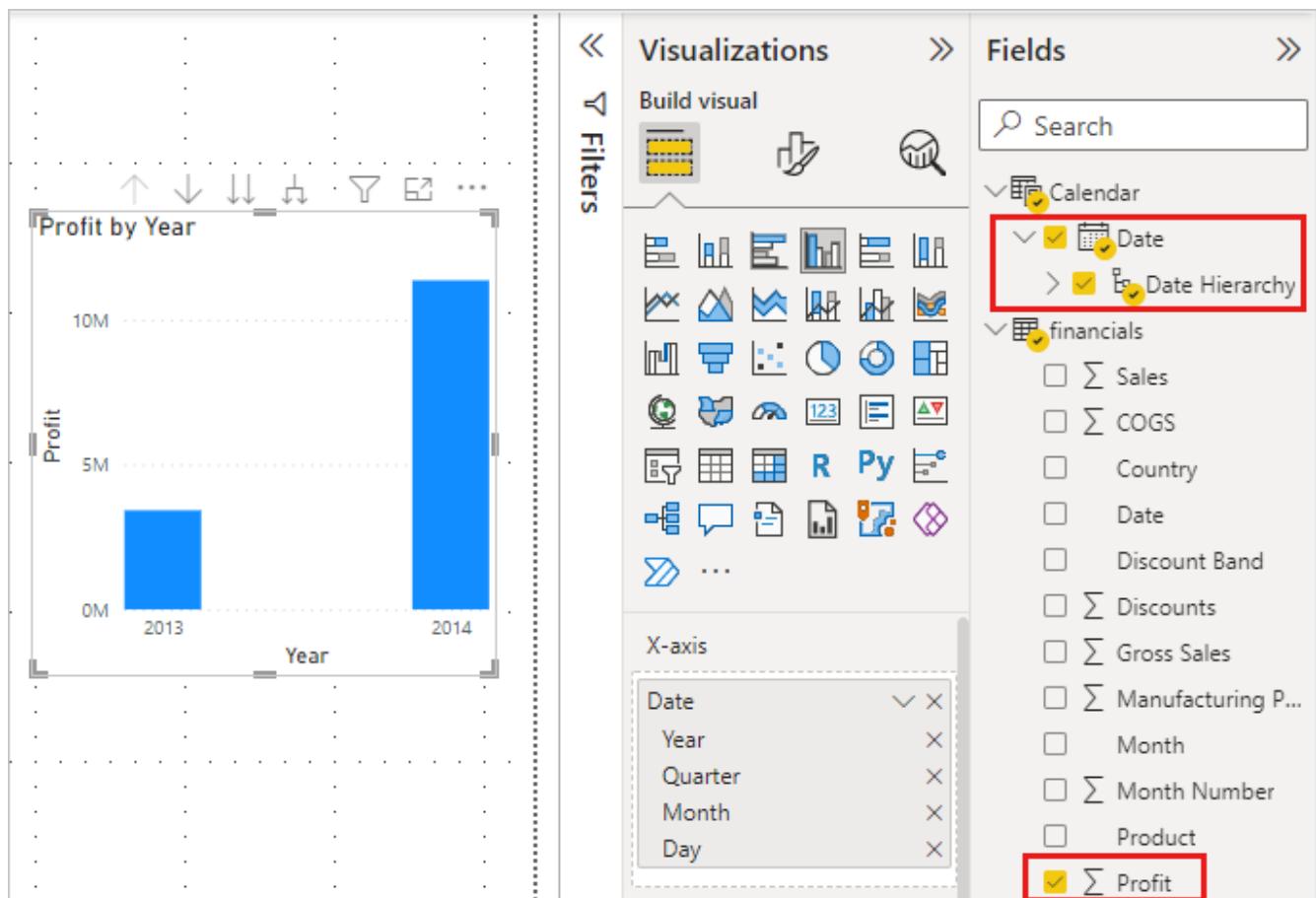
3. Resize the box to fit on one line.

Visual 2: Profit by Date

Now, you create a line chart to see which month and year had the highest profit.

1. From the Fields pane, drag the **Profit** field to a blank area on the report canvas. By default, Power BI displays a column chart with one column, Profit.
2. Drag the **Date** field to the same visual.

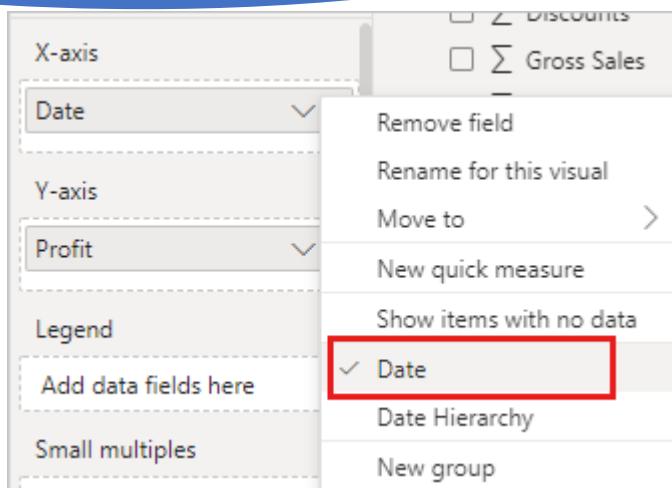
Power BI updates the column chart to show profit by the two years.



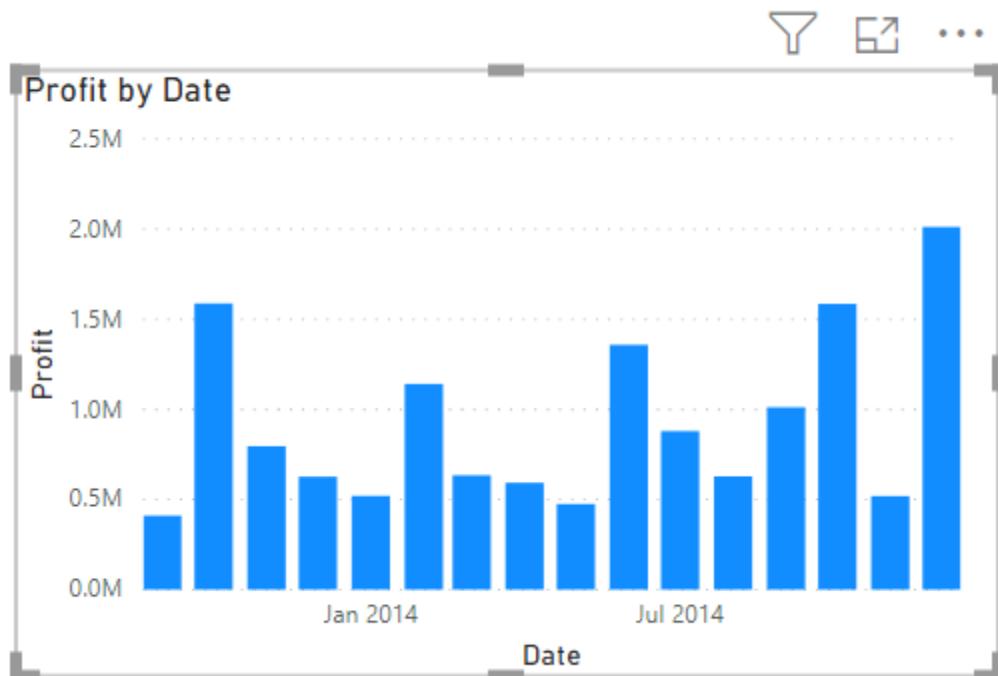
3. In the **Fields** section of the Visualizations pane, select the drop-down in the **X-axis** value. Change **Date** from **Date Hierarchy** to **Date**.

Data Analytics

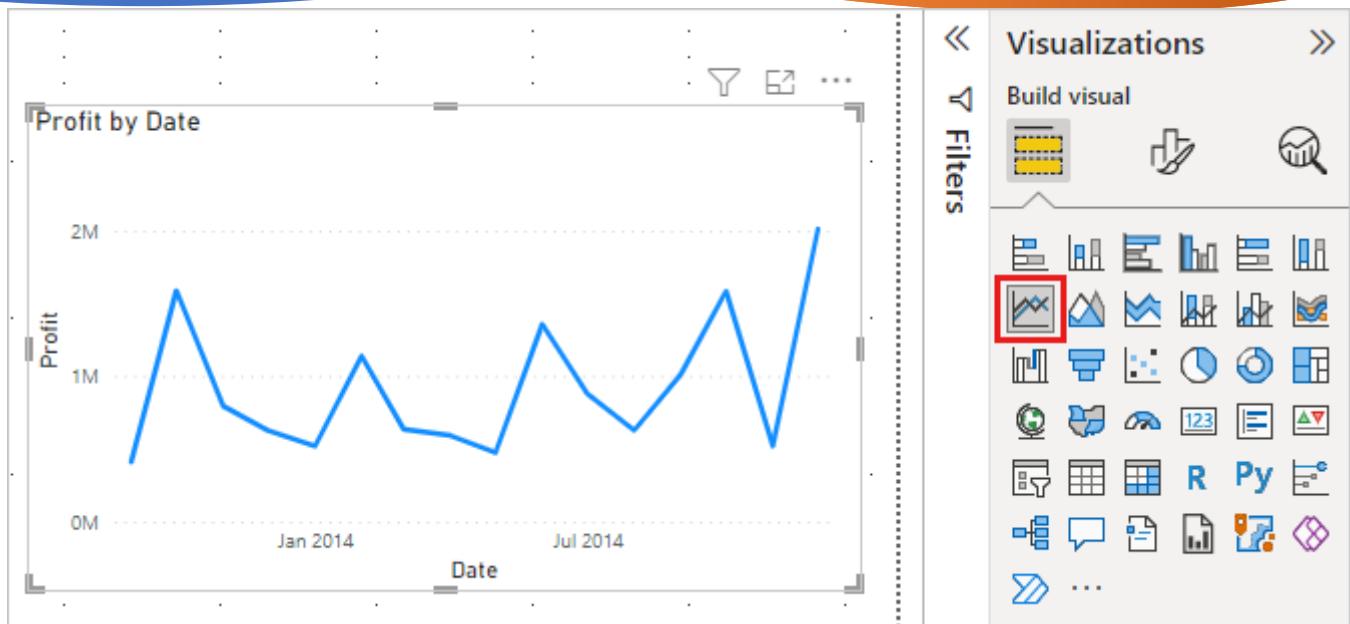
Student Material



Power BI updates the column chart to show profit for each month.



4. In the Visualizations pane, change the visualization type to **Line chart**.



Now you can easily see that December 2014 had the most profit.

Visual 3: Profit by Country/Region

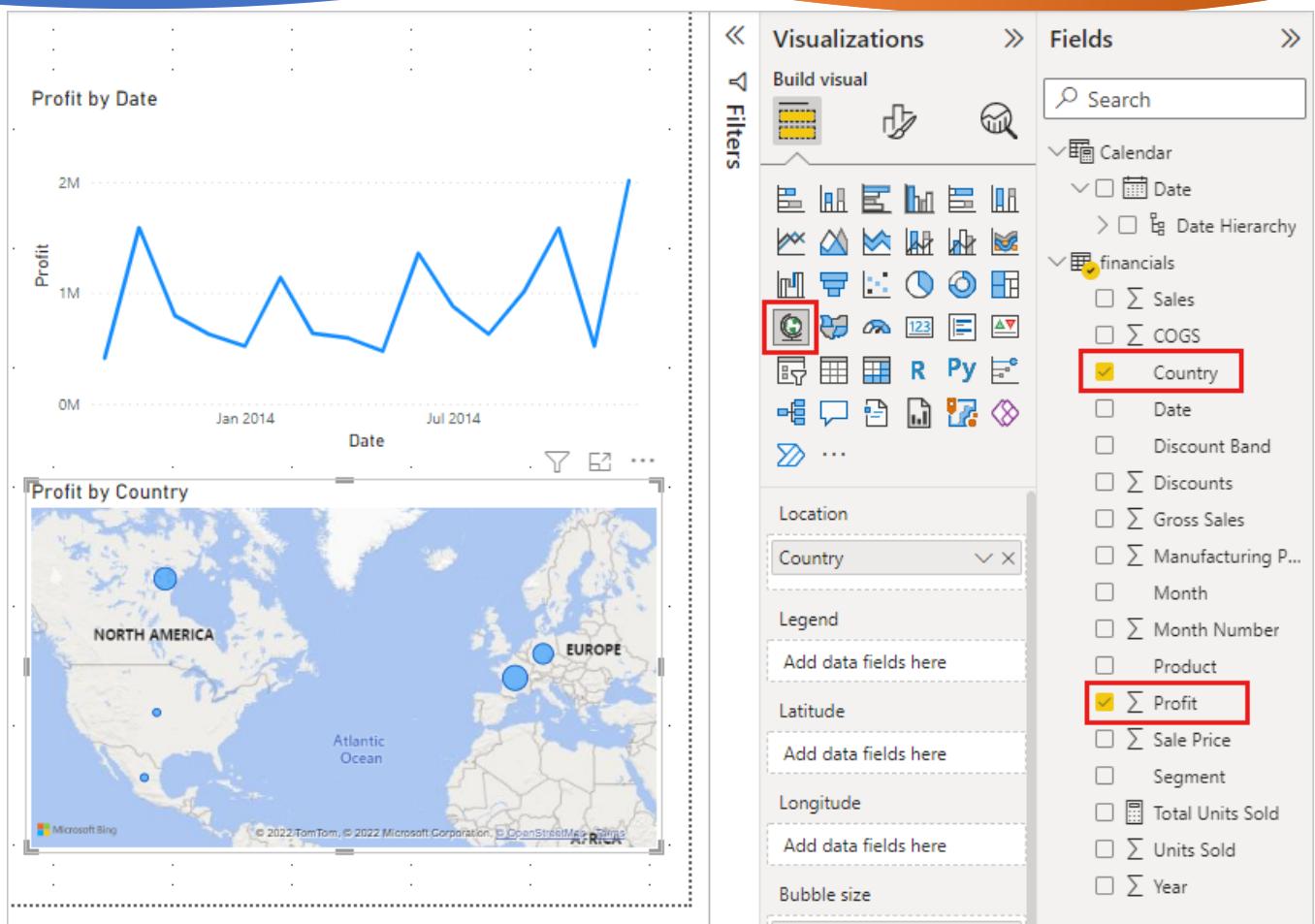
Create a map to see which country/region had the highest profits.

1. From the Fields pane, drag the **Country** field to a blank area on your report canvas to create a map.
2. Drag the **Profit** field to the map.

Power BI creates a map visual with bubbles representing the relative profit of each location.

Data Analytics

Student Material



Europe seems to be performing better than North America.

Visual 4: Sales by Product and Segment

Create a bar chart to determine which companies and segments to invest in.

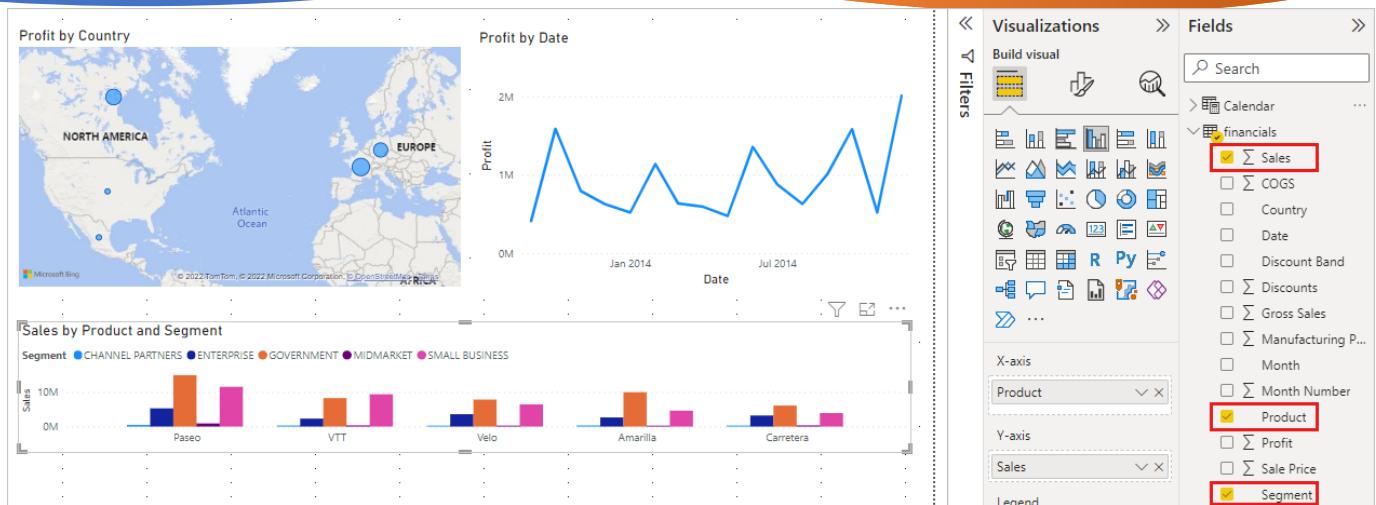
1. Drag the two charts you've created to be side by side in the top half of the canvas. Save some room on the left side of the canvas.
2. Select a blank area in the lower half of your report canvas.
3. In the Fields pane, select the **Sales**, **Product**, and **Segment** fields.

Power BI automatically creates a clustered column chart.

4. Drag the chart so it's wide enough to fill the space under the two upper charts.

Data Analytics

Student Material



Looks like the company should continue to invest in the Paseo product and target the Small Business and Government segments.

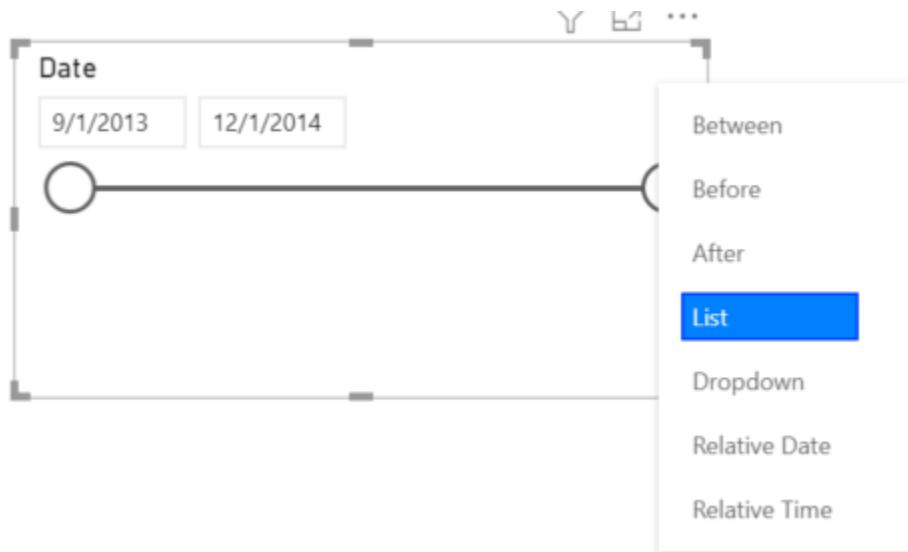
Visual 5: Year slicer

Slicers are a valuable tool for filtering the visuals on a report page to a specific selection. In this case, we can create two different slicers to narrow in on performance for each month and year. In which one slicer uses the date field in the original table.

Date slicer using the original table

1. In the Fields pane, select the **Date** field in the Financials table. Drag it to the blank area on the left of the canvas.
2. In the Visualizations pane, choose **Slicer**.

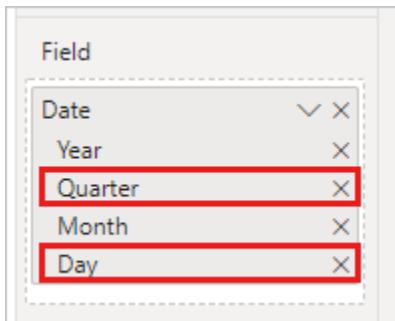
Power BI automatically creates a numeric range slicer.



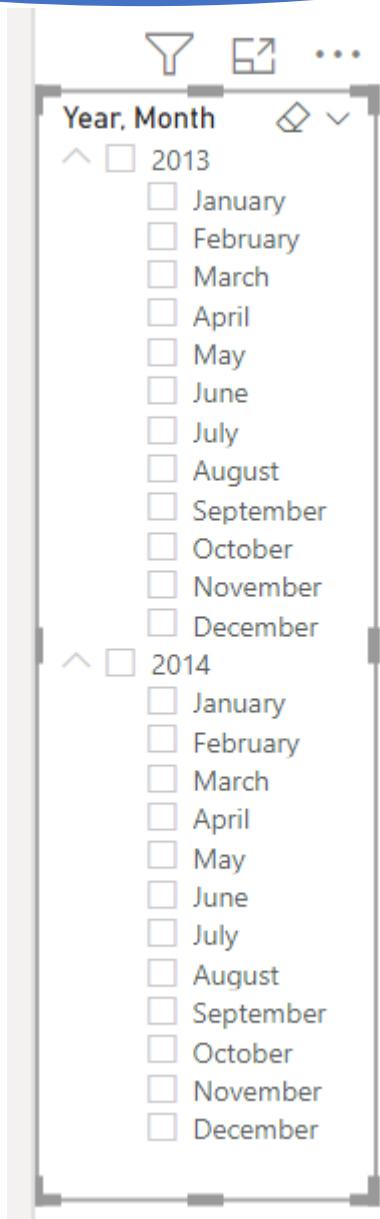
3. You can drag the ends to filter, or select the arrow in the upper-right corner and change it to a different type of slicer.

Date slicer using the DAX table

1. In the Fields pane, select the **Date** field in the Calendar table. Drag it to the blank area on the left of the canvas.
2. In the Visualizations pane, choose **Slicer**.
3. In the Fields section of the Visualizations pane, select the drop-down in **Fields**. Remove Quarter and Day so only Year and Month are left.



4. Expand each year and resize the visual, so all months are visible.



We'll use this slicer in the finished report.

Now if your manager asks to see just 2013 data, you can use either slicer to select years, or specific months of each year.

Extra credit: Format the report

If you want to do some light formatting on this report to add more polish, here are a few easy steps.

Theme

- On the **View** ribbon, change the theme to **Executive**.

Data Analytics

Student Material

File

Home

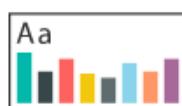
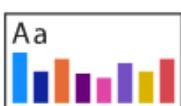
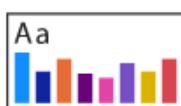
Insert

Modeling

View

Help

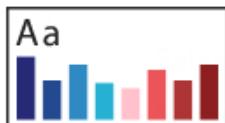
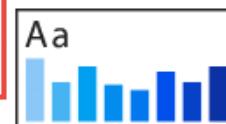
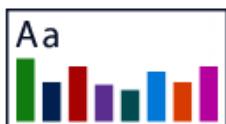
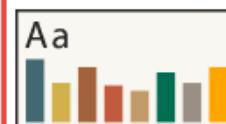
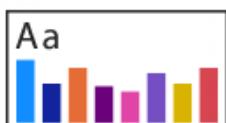
Format



This report

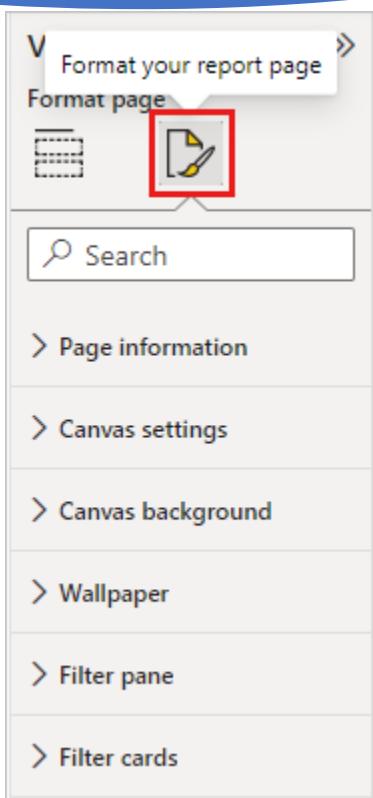


Power BI



Spruce up the visuals

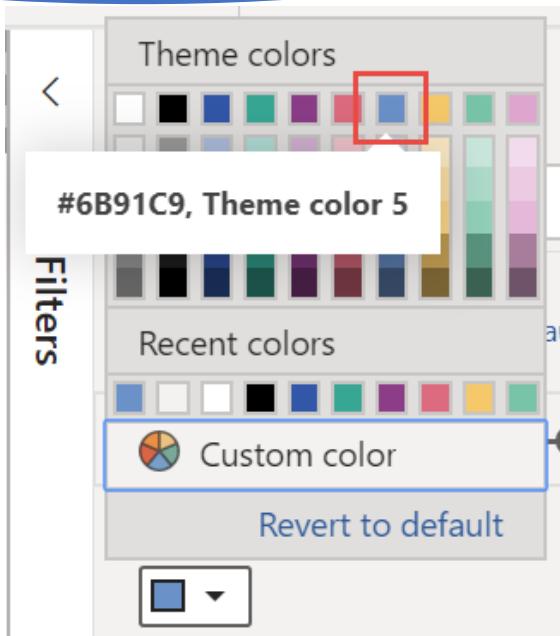
Make the following changes on the **Format** tab in the Visualizations pane.



1. Select Visual 2. In the **Title** section, change **Title text** to “Profit by Month and Year” and **Text size** to **16 pt**. Toggle **Shadow** to **On**.
2. Select Visual 3. In the **Map styles** section, change **Theme** to **Grayscale**. In the **Title** section, change title **Text size** to **16 pt**. Toggle **Shadow** to **On**.
3. Select Visual 4. In the **Title** section, change title **Text size** to **16 pt**. Toggle **Shadow** to **On**.
4. Select Visual 5. In the **Selection controls** section, toggle **Show "Select all"** option to **On**. In the **Slicer header** section, increase **Text size** to **16 pt**.

Add a background shape for the title

1. On the **Insert** ribbon, select **Shapes > Rectangle**. Place it at the top of the page, and stretch it to be the width of the page and height of the title.
2. In the **Format shape** pane, in the **Outline** section, change **Transparency** to **100%**.
3. In the **Fill** section, change **Fill color** to **Theme color 5 #6B91C9 (blue)**.



4. On the **Format** tab, select **Send backward > Send to back**.
5. Select the text in Visual 1, the title, and change the font color to **White**.

Add a background shape for visuals 2 and 3

1. On the **Insert** ribbon, select **Shapes > Rectangle**, and stretch it to be the width and height of Visuals 2 and 3.
2. In the **Format shape** pane, in the **Outline** section, change **Transparency** to **100%**.
3. In the **Fill** section, set the color to **White, 10% darker**.
4. On the **Format** tab, select **Send backward > Send to back**.

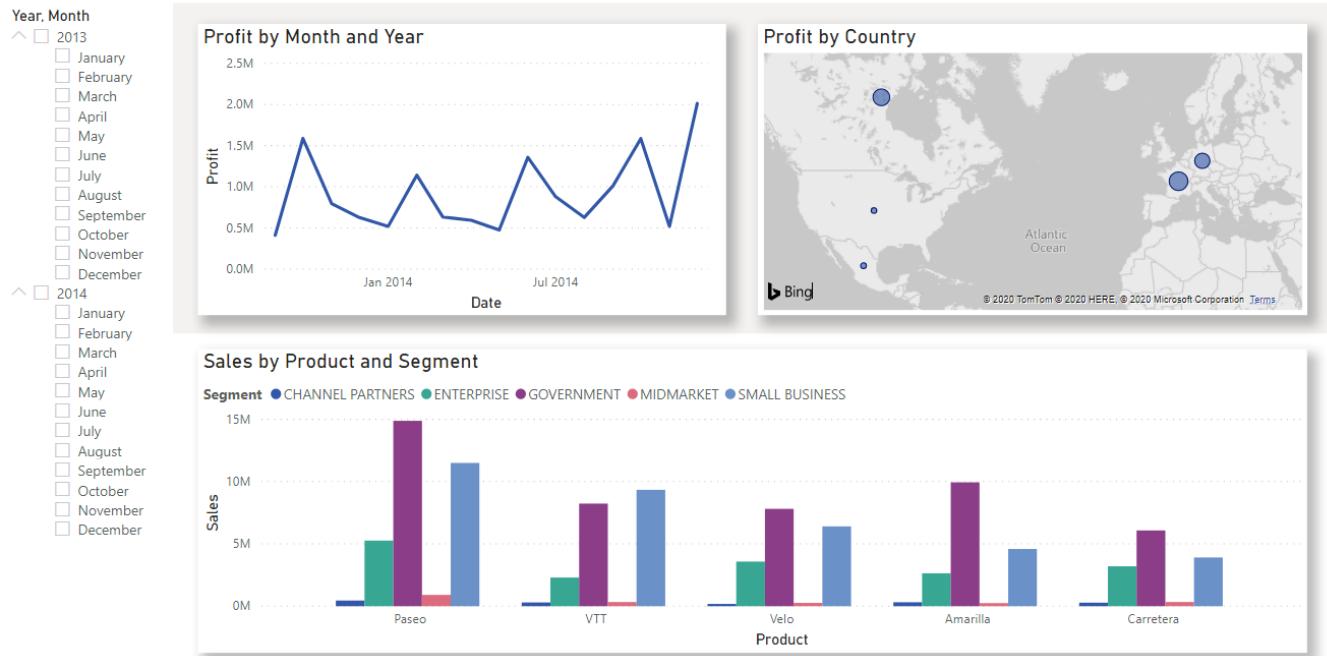
Finished report

Here's how your final polished report will look:

Data Analytics

Student Material

Executive Summary – Finance Report



In summary, this report answers these top questions:

- Which month and year had the most profit?
December 2014
- Which country/region is the company seeing the most success in?
In Europe, specifically France and Germany.
- Which product and segment should the company continue to invest in?
The company should continue to invest in the Paseo product and target the Small Business and Government segments.

Save your report

- On the **File** menu, select **Save**.

INTRODUCTION

Nowadays, every successful organization has a presence over the internet for which they require a data center irrespective of their size. The traditional data center is on-premises, meaning that all its functionality is carried out in a physical site within enterprise office space. A data center might be a few computers under a desk, a climate-controlled room filled with blade servers, or a whole building. It is managed by an in-house IT team employed and paid for by the enterprise which owns the data center.

Now, imagine when you want to create a simple website or digitize whole business processes, apart from building applications, you also need to create a data center for an organization. This means you will have to buy hardware (servers), software, and its licenses, create a network, build infrastructure, and hire a team of experts to manage or maintain this data center. Before setting up the data center, you need to do an assessment on how many servers you need, there will be chances that you might over-provision or under-provision hardware.

What if you don't have to worry about setting up your own data center and just focus on building applications? This is where Cloud Computing comes into the picture. Let us now understand what Cloud Computing is with examples in this article. Also, you can go for Cloud Computing training to get a deeper knowledge of Cloud Computing.

What is Cloud Computing?

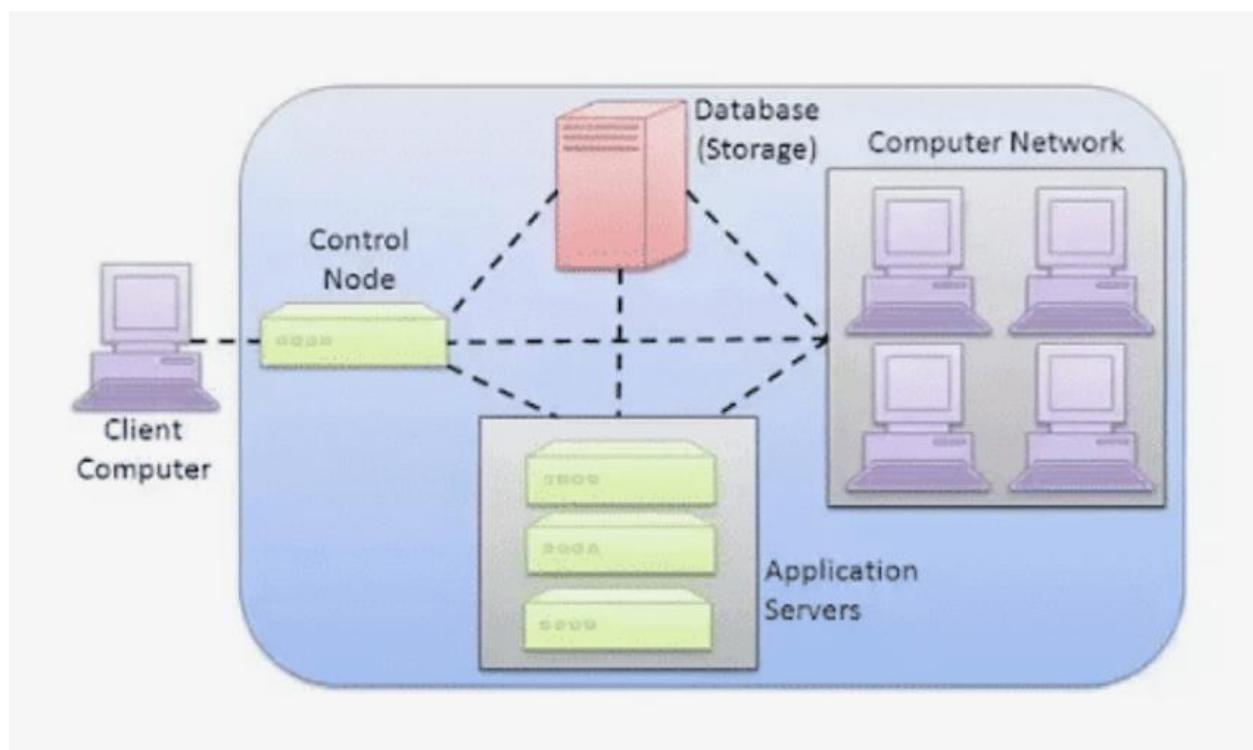
Cloud Computing is the delivery of computing services such as servers, data storage, databases, networking, software, analytics, and intelligence over the internet ("cloud") to offer flexible resources, faster innovation, and economies of scale. In simpler terms, instead of owning data centers, organizations can rent access to someone else's infrastructure like storage, computing servers, and databases from a Cloud Computing service provider and only pay for resources that they use.

You only need to pay for the cloud services that you use, which helps lower your operating costs, run infrastructure more efficiently, and scale your applications as per business needs.

Who Uses Cloud Computing?

Organizations of different types, sizes, and industries are using the cloud for a wide variety of use cases, such as building customer-facing web applications, data backup, sending email/SMS notifications, virtual desktops, software development and testing, big data analytics, and disaster recovery. For example, Telecom companies are using cloud services to connect with their customers by sending different types of communications. Financial services companies are using the cloud to power real-time fraud detection and prevention.

How Does Cloud Computing Work?



To understand how Cloud Computing works, let us divide it into two sections- Front end and Back end. The front end consists of the client's computer or computer network. The front end consists of the client's computer or computer network. The back end consists of various computers, servers and data storage systems that make up the cloud. They are connected to each other through a network, usually the Internet. The front end is the side of the computer user or client. The back end is 'the cloud' section of the system.

Cloud Computing Deployment Models

The deployment models specify different types of clouds. Every organization has different needs, they need to determine which cloud deployment model will work for them. There are mainly three cloud deployment models:

1. Public Cloud

The public cloud is a set of hardware, networking, storage, services, applications, and interfaces owned and operated by a third party for use by other companies or individuals. You access these services and manage your account using a web browser. These commercial providers create a highly scalable data center that hides the details of the underlying infrastructure from the consumer.

2. Private Cloud

A private cloud is a set of hardware, networking, storage, services, applications, and interfaces owned and operated by an organization for the use of its employees, partners, or customers. A private cloud can be created and managed by a third party for the exclusive use of one enterprise. The private cloud is a highly controlled environment not open for public consumption. It is essentially just another way of running an on-premises data center.

3. Hybrid Cloud

A hybrid cloud is a combination of a private cloud combined with the use of public cloud services where the two cloud environments work together to solve business problems. By allowing data and applications to move between private and public clouds, a hybrid cloud gives your business greater flexibility, more deployment options and helps optimize your existing infrastructure, security, and compliance.

The goal is to create a hybrid cloud environment that can combine services and data from a variety of cloud models to create a unified, automated, and well-managed computing environment.

Types of Cloud Services

Cloud Computing services are divided into three classes, according to the abstraction level of the capability provided and the service model of providers:

1. Infrastructure as a Service (IaaS in Cloud Computing)

The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

2. Platform as a Service (PaaS in Cloud Computing)

The capability provided to the consumer is to deploy onto the Cloud infrastructure consumer-created or acquired applications using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying Cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

3. Software as a Service (SaaS in Cloud Computing)

The capability provided to the consumer is their applications running on a Cloud infrastructure. The capability provided to the consumer is to use the provider's applications running on a Cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based e-mail), or a program interface.

The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, apart from limited user-specific application configuration settings.

Cloud Computing Examples and Use Cases

If you use an online service to send email, edit documents, watch movies or TV (like Netflix), listen to music, play games, or store pictures and other files, it is likely that you are part of cloud eco-system, as Cloud Computing is making it all possible behind the scenes. There are many use cases of Cloud Computing, few are mentioned below:

1. Test and Development

IT Companies uses cloud services for software development environment. DevOps teams can quickly spin up development, testing and production environments. This includes an automated provisioning of physical and virtual machines.

2. Big Data Analytics

There is massive amount of data collected each day from cloud applications, IoT devices and the users who interact with them. Cloud Computing allows organization to leverage the computing power of Cloud Computing.

3. Cloud Storage

Cloud data storage enables files to be automatically saved to the cloud, and then they can be accessed, stored, and retrieved from any device with an internet connection. Instead of maintaining data centers for storage, organizations can only pay for cloud storage they are use and do so without the worries of overseeing the daily maintenance of the storage infrastructure.

Benefits of Cloud Computing

There are several benefits of using Cloud Computing over traditional way. Few benefits are mentioned below:

1. Cost Saving

You pay only for the services you use; this eliminates capital expenses of buying hardware, software, setting up datacenters and operating cost for the same.

2. Rapid Elasticity

Cloud providers pool large number of resources from their data centers and make them easily accessible. A service provider can easily expand its services to large scale to handle rapid increase in service demands.

3. Global Scale

With the cloud, you can expand your business to new geographic locations and deploy applications globally within minutes. Many cloud providers give

services in lot of countries, deploying applications closer proximity to end users reduces latency and hence improves customer experience.

4. Reliability

Cloud services provides high availability with their robust infrastructure. You can easily make data backup, disaster recovery, which makes business continuity easier and less expensive as data can be mirrored at different geographic locations on the cloud provider's network.

5. Speed

Cloud Computing services are mostly self service and on demand. You can easily provision any number of resources within minutes, that gives a lot of flexibility to businesses to scale their application at right time and at right location.

Disadvantages of Cloud Computing

Although there are many benefits of Cloud Computing, there are few disadvantages as well which you need to be aware of, such as:

1. Security and Privacy

Although cloud service providers implement the best security standards and industry certifications, storing data and important files on external service providers always opens risks. Any discussion involving data must address security and privacy, especially when it comes to managing sensitive data. You must understand the shared responsibility model of your cloud provider. You will still be liable for what occurs within your network and in your product.

2. Vulnerability to Attack

In Cloud Computing, every component is online, which exposes potential vulnerabilities. Even the best teams suffer severe attacks and security breaches from time to time.

3. Limited Control

Cloud services run on remote servers that are owned and managed by cloud service providers, which makes it hard for the companies to have the level of control over cloud infrastructure.

Future of Cloud Computing and Emerging Technologies

Lot of companies are already using Cloud services to grow their business and make their global presence. Cloud Computing will be the most common deployment model for companies in future because of its benefits. The future of Cloud Computing is bright and will provide benefits to both the host and the customer. There are many technologies that are emerging with Cloud Computing such as:

1. Internet Of Thing (IoT)

The internet of things is one of the leading Technology, it comes with continuous innovation in real time Data Analytics and Cloud Computing. We can do it easily with the help of Cloud Computing.

2. Serverless Computing

Serverless architecture is the next evolution from monolithic application architecture after service-oriented architecture and micro-services architectures.

3. Artificial Intelligence (AI)

Artificial Intelligence is the next-generation technology solution set to present the technology world in a different view. However, building AI applications are complex for many businesses as it requires high computing power machines. Companies are looking at Cloud solutions for machine learning and other deep learning tools. Because of its vast computing and storage options, cloud-based AI is emerging as the most-sought solution for businesses of any size in realizing their AI efforts.

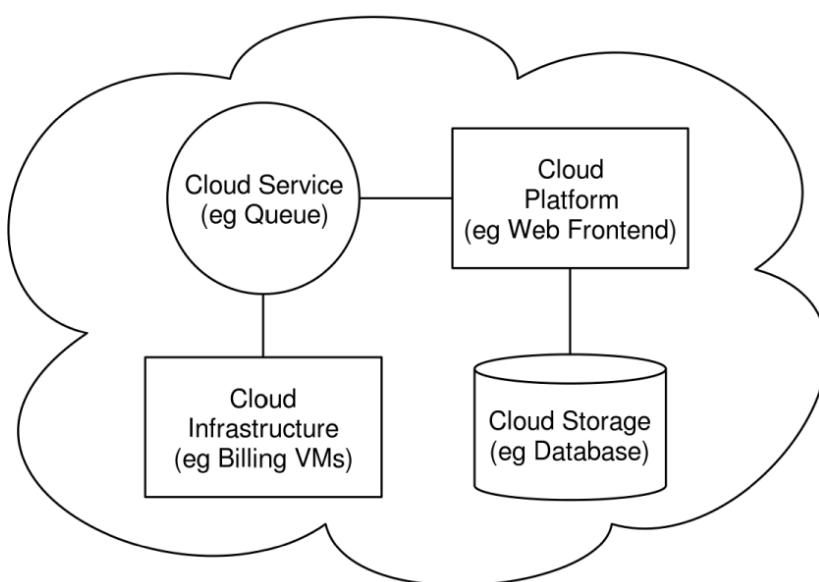
Cloud Computing Architecture

Architecture is how different components combine to create a cloud system, which every employee of an organization can use for data-related operations. Talking of cloud computing architecture, virtualization

technology pools different resources and software components in one place, calling it a cloud. There are various components in cloud architecture:

- A front-end device from which the user accesses the network.
- The backend platform is like the servers that store the information for the user to access.
- A delivery model that decides how the network will work and how the user will be able to access data.
- A trusted network on which the storage and sharing of information will happen.

Putting all these components together forms a cloud platform, which an organization and its employees can use. This system helps the users in plenty of ways. First, it reduces the risk of data theft as online servers are much more secure than physical ones. Secondly, it makes remote working possible for everyone as there will be no issue with data sharing.



Different Architectures

Although no two cloud models are similar, the basics of a few models are alike. Therefore, an organization can use the model, which they think would stand tall on their expectations, make changes, and enjoy all the perks that this virtual data storage and sharing platform offers.

Cloud Computing vs Traditional IT Infrastructure

Below are the differences between Cloud Computing and Traditional IT Infrastructure:

Parameters	Traditional IT Infrastructure	Cloud Computing
Position	Traditional computing is done on physical server provisioned and managed by organization. Companies own these servers	Cloud Computing run on outsider servers facilitated by third-party hosting organizations such as Microsoft AZURE, Amazon AWS, Google GCP
Cost Effective	In traditional computing companies must spend up-front expenses on hardware	Cloud Computing works on pay as you go model, you pay only for number of resources you use
Security	In traditional computing companies had to spend lot of efforts securing their infrastructure, they had to hire security experts to protect their data	In Cloud Computing the Cloud service providers are specialized in guarding the data and their primary responsibility is security data
Flexibility	In traditional computing you need to do proper estimations in order to buy hardware upfront	Cloud Computing is flexible in this regard, you can use resource as per your need

Conclusion

Data Analytics

Student Material

One of the many advantages of Cloud Computing are to reduce the time to market applications that need to scale dynamically. Cloud Computing brings lot of benefits in terms of cost, agility, scalability, resiliency and many more. Considering these benefits many organizations are utilizing cloud services, for creating highly scalable and resilient applications.

DATABASE

What is Data?

Data is a collection of a distinct small unit of information. It can be used in a variety of forms like text, numbers, media, bytes, etc. it can be stored in pieces of paper or electronic memory, etc.

Word 'Data' is originated from the word 'datum' that means 'single piece of information.' It is plural of the word datum.

In computing, Data is information that can be translated into a form for efficient movement and processing. Data is interchangeable.

What is Database?

A **database** is an organized collection of data, so that it can be easily accessed and managed.

You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.

Database handlers create a database in such a way that only one set of software program provides access of data to all the users.

The **main purpose** of the database is to operate a large amount of information by storing, retrieving, and managing data.

There are many **dynamic websites** on the World Wide Web nowadays which are handled through databases. For example, a model that checks the availability of rooms in a hotel. It is an example of a dynamic website that uses a database.

There are many **databases available** like MySQL, Sybase, Oracle, MongoDB, Informix, PostgreSQL, SQL Server, etc.

Modern databases are managed by the database management system (DBMS).

SQL or Structured Query Language is used to operate on the data stored in a database. SQL depends on relational algebra and tuple relational calculus.

A cylindrical structure is used to display the image of a database.

Evolution of Databases

The database has completed more than 50 years of journey of its evolution from flat-file system to relational and objects relational systems. It has gone through several generations.

The Evolution

File-Based

1968 was the year when File-Based database were introduced. In file-based databases, data was maintained in a flat file. Though files have many advantages, there are several limitations.

One of the major advantages is that the file system has various access methods, e.g., sequential, indexed, and random.

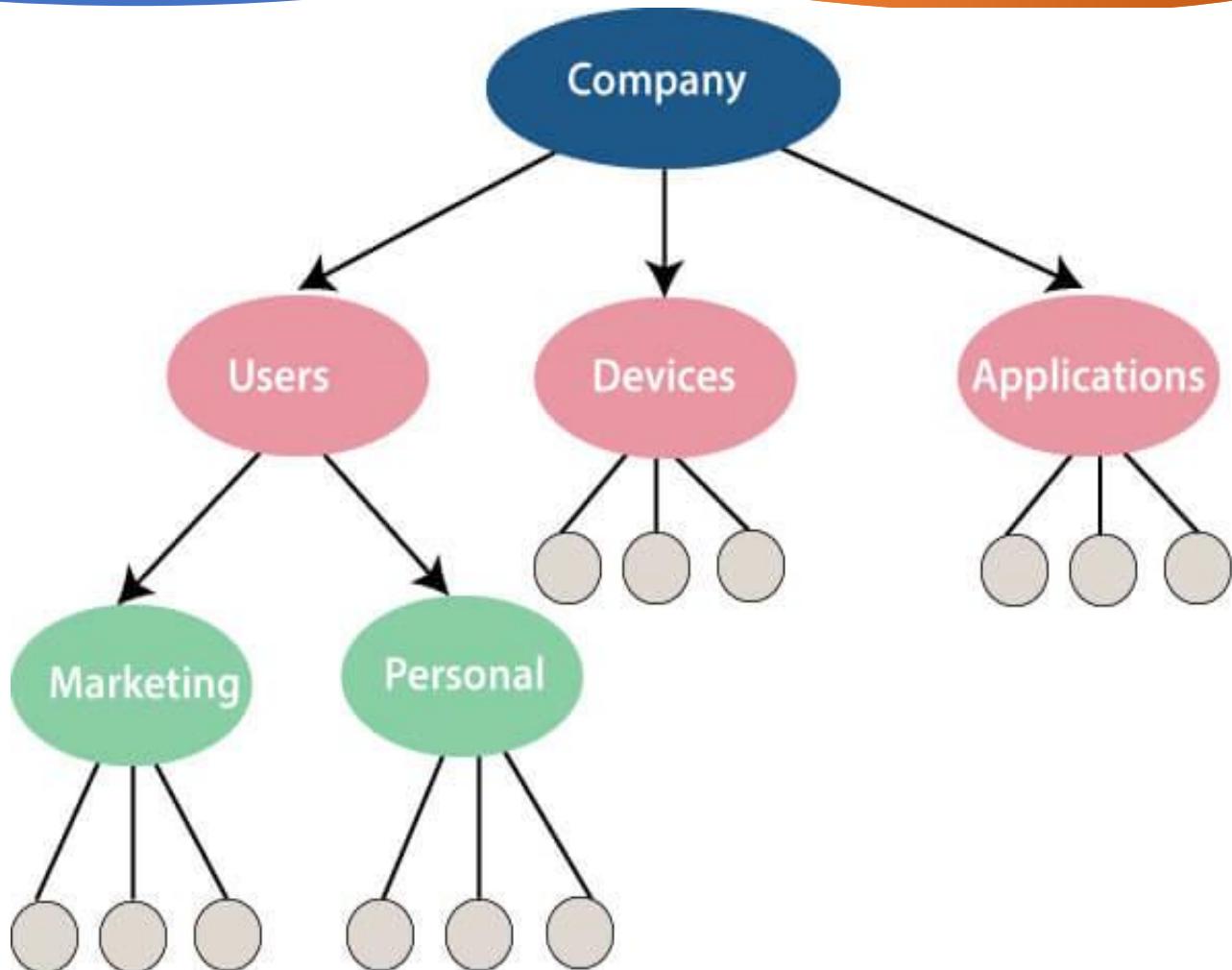
It requires extensive programming in a third-generation language such as COBOL, BASIC.

Hierarchical Data Model

1968-1980 was the era of the Hierarchical Database. Prominent hierarchical database model was IBM's first DBMS. It was called IMS (Information Management System).

In this model, files are related in a parent/child manner.

Below diagram represents Hierarchical Data Model. Small circle represents objects.



Like file system, this model also had some limitations like complex implementation, lack structural independence, can't easily handle a many-many relationship, etc.

Network data model

Charles Bachman developed the first DBMS at Honeywell called Integrated Data Store (IDS). It was developed in the early 1960s, but it was standardized in 1971 by the CODASYL group (Conference on Data Systems Languages).

In this model, files are related as owners and members, like to the common network model.

Network data model identified the following components:

- Network schema (Database organization)
- Sub-schema (views of database per user)
- Data management language (procedural)

This model also had some limitations like system complexity and difficult to design and maintain.

Relational Database

1970 - Present: It is the era of Relational Database and Database Management. In 1970, the relational model was proposed by E.F. Codd.

Relational database model has two main terminologies called instance and schema.

The instance is a table with rows or columns

Schema specifies the structure like name of the relation, type of each column and name.

This model uses some mathematical concept like set theory and predicate logic.

The first internet database application had been created in 1995.

During the era of the relational database, many more models had introduced like object-oriented model, object-relational model, etc.

Cloud database

Cloud database facilitates you to store, manage, and retrieve their structured, unstructured data via a cloud platform. This data is accessible over the Internet. Cloud databases are also called a database as service (DBaaS) because they are offered as a managed service.

Some best cloud options are:

- AWS (Amazon Web Services)
- Snowflake Computing
- Oracle Database Cloud Services
- Microsoft SQL server
- Google cloud spanner

Advantages of cloud database

Lower costs

Generally, company provider does not have to invest in databases. It can maintain and support one or more data centers.

Automated

Cloud databases are enriched with a variety of automated processes such as recovery, failover, and auto-scaling.

Increased accessibility

You can access your cloud-based database from any location, anytime. All you need is just an internet connection.

NoSQL Database

A NoSQL database is an approach to design such databases that can accommodate a wide variety of data models. NoSQL stands for "not only SQL." It is an alternative to traditional relational databases in which data is placed in tables, and data schema is perfectly designed before the database is built.

NoSQL databases are useful for a large set of distributed data.

Some examples of NoSQL database system with their category are:

- MongoDB, CouchDB, Cloudant (**Document-based**)
- Memcached, Redis, Coherence (**key-value store**)
- HBase, Big Table, Accumulo (**Tabular**)

Advantage of NoSQL

High Scalability

NoSQL can handle an extensive amount of data because of scalability. If the data grows, NoSQL database scale it to handle that data in an efficient manner.

High Availability

NoSQL supports auto replication. Auto replication makes it highly available because, in case of any failure, data replicates itself to the previous consistent state.

Disadvantage of NoSQL

Open source

NoSQL is an open-source database, so there is no reliable standard for NoSQL yet.

Management challenge

Data management in NoSQL is much more complicated than relational databases. It is very challenging to install and even more hectic to manage daily.

GUI is not available

GUI tools for NoSQL database are not easily available in the market.

Backup

Backup is a great weak point for NoSQL databases. Some databases, like MongoDB, have no powerful approaches for data backup.

The Object-Oriented Databases

The object-oriented databases contain data in the form of objects and classes. Objects are the real-world entity, and types are the collection of objects. An object-oriented database is a combination of relational model features with objects oriented principles. It is an alternative implementation to that of the relational model.

Object-oriented databases hold the rules of object-oriented programming. An object-oriented database management system is a hybrid application.

The object-oriented database model contains the following properties.

Object-oriented programming properties

- Objects
- Classes

- Inheritance
- Polymorphism
- Encapsulation

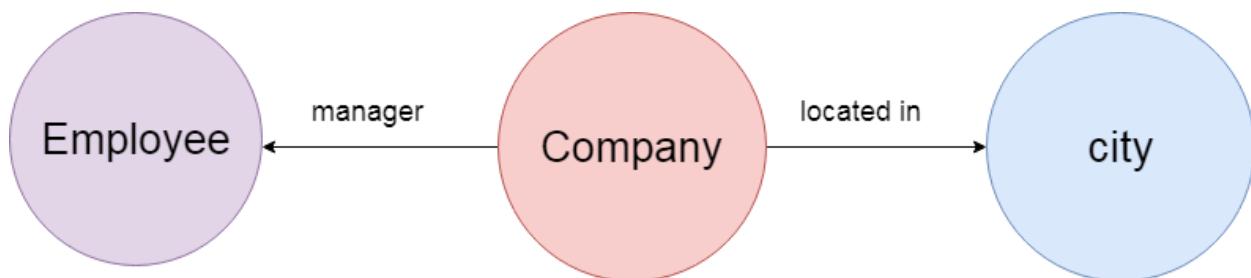
Relational database properties

- Atomicity
- Consistency
- Integrity
- Durability
- Concurrency
- Query processing

Graph Databases

A graph database is a NoSQL database. It is a graphical representation of data. It contains nodes and edges. A node represents an entity, and each edge represents a relationship between two edges. Every node in a graph database represents a unique identifier.

Graph databases are beneficial for searching the relationship between data because they highlight the relationship between relevant data.



Graph databases are very useful when the database contains a complex relationship and dynamic schema.

It is mostly used in **supply chain management**, identifying the source of **IP telephony**.

DBMS (Data Base Management System)

Database management System is software which is used to store and retrieve the database. For example, Oracle, MySQL, etc.; these are some popular DBMS tools.

- DBMS provides the interface to perform the various operations like creation, deletion, modification, etc.
- DBMS allows the user to create their databases as per their requirement.
- DBMS accepts the request from the application and provides specific data through the operating system.
- DBMS contains the group of programs which acts according to the user instruction.
- It provides security to the database.

Advantage of DBMS

Controls redundancy

It stores all the data in a single database file, so it can control data redundancy.

Data sharing

An authorized user can share the data among multiple users.

Backup

It provides Backup and recovery subsystem. This recovery system creates automatic data from system failure and restores data if required.

Multiple user interfaces

It provides a different type of user interfaces like GUI, application interfaces.

Disadvantage of DBMS

Size

It occupies large disk space and large memory to run efficiently.

Cost

DBMS requires a high-speed data processor and larger memory to run DBMS software, so it is costly.

Complexity

DBMS creates additional complexity and requirements.

RDBMS (Relational Database Management System)

The word RDBMS is termed as 'Relational Database Management System.' It is represented as a table that contains rows and column.

RDBMS is based on the Relational model; it was introduced by E. F. Codd.

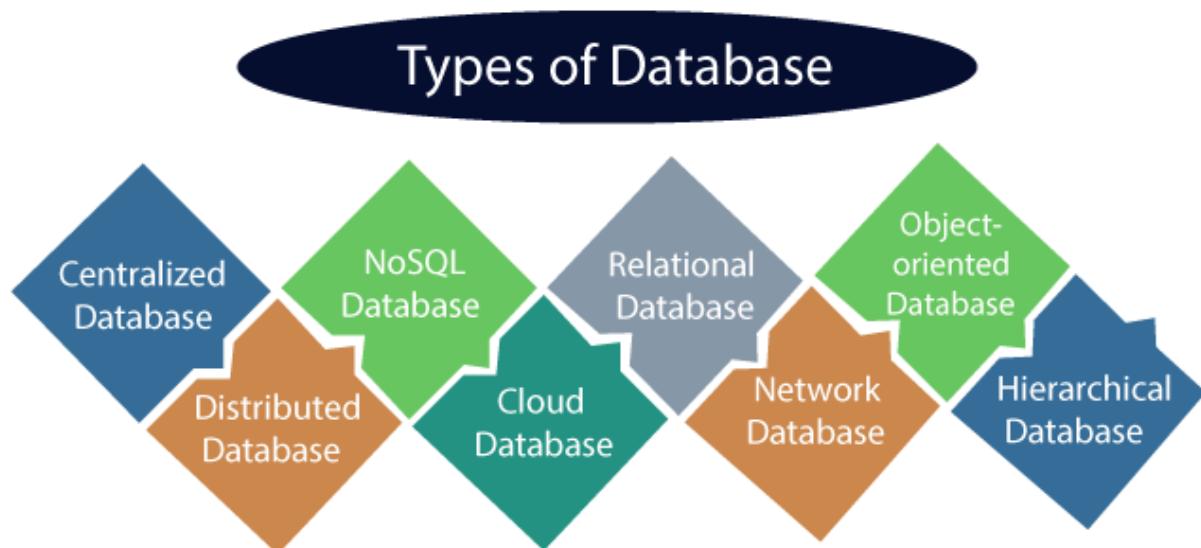
A relational database contains the following components:

- Table
- Record/ Tuple
- Field/Column name /Attribute
- Instance
- Schema
- Keys

An RDBMS is a tabular DBMS that maintains the security, integrity, accuracy, and consistency of the data.

Types of Databases

There are various types of databases used for storing different varieties of data:



1) Centralized Database

It is the type of database that stores data at a centralized database system. It comforts the users to access the stored data from different locations through

several applications. These applications contain the authentication process to let users access data securely. An example of a Centralized database can be Central Library that carries a central database of each library in a college/university.

Advantages of Centralized Database

- It has decreased the risk of data management, i.e., manipulation of data will not affect the core data.
- Data consistency is maintained as it manages data in a central repository.
- It provides better data quality, which enables organizations to establish data standards.
- It is less costly because fewer vendors are required to handle the data sets.

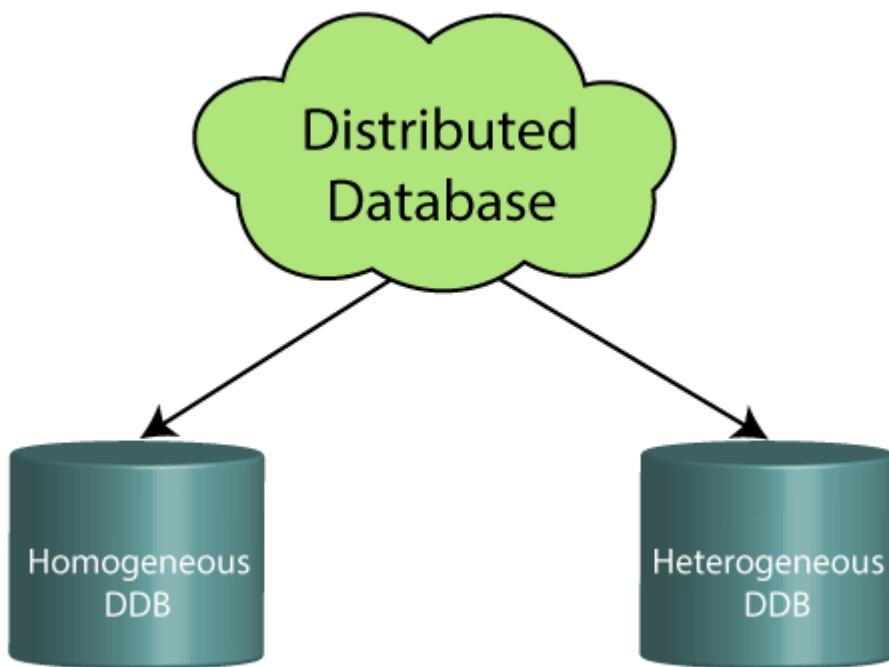
Disadvantages of Centralized Database

- The size of the centralized database is large, which increases the response time for fetching the data.
- It is not easy to update such an extensive database system.
- If any server failure occurs, entire data will be lost, which could be a huge loss.

2) Distributed Database

Unlike a centralized database system, in distributed systems, data is distributed among different database systems of an organization. These database systems are connected via communication links. Such links help the end-users to access the data easily. **Examples** of the Distributed database are Apache Cassandra, HBase, Ignite, etc.

We can further divide a distributed database system into:



- **Homogeneous DDB:** Those database systems which execute on the same operating system and use the same application process and carry the same hardware devices.
- **Heterogeneous DDB:** Those database systems which execute on different operating systems under different application procedures, and carries different hardware devices.

Advantages of Distributed Database

- Modular development is possible in a distributed database, i.e., the system can be expanded by including new computers and connecting them to the distributed system.
- One server failure will not affect the entire data set.

3) Relational Database

This database is based on the relational data model, which stores data in the form of rows(tuple) and columns(attributes), and together forms a table(relation). A relational database uses SQL for storing, manipulating, as well as maintaining the data. E.F. Codd invented the database in 1970. Each table in the database carries a key that makes the data unique from others. **Examples of Relational databases are MySQL, Microsoft SQL Server, Oracle, etc.**

Properties of Relational Database

There are following four commonly known properties of a relational model known as ACID properties, where:

A means Atomicity: This ensures the data operation will complete either with success or with failure. It follows the 'all or nothing' strategy. For example, a transaction will either be committed or will abort.

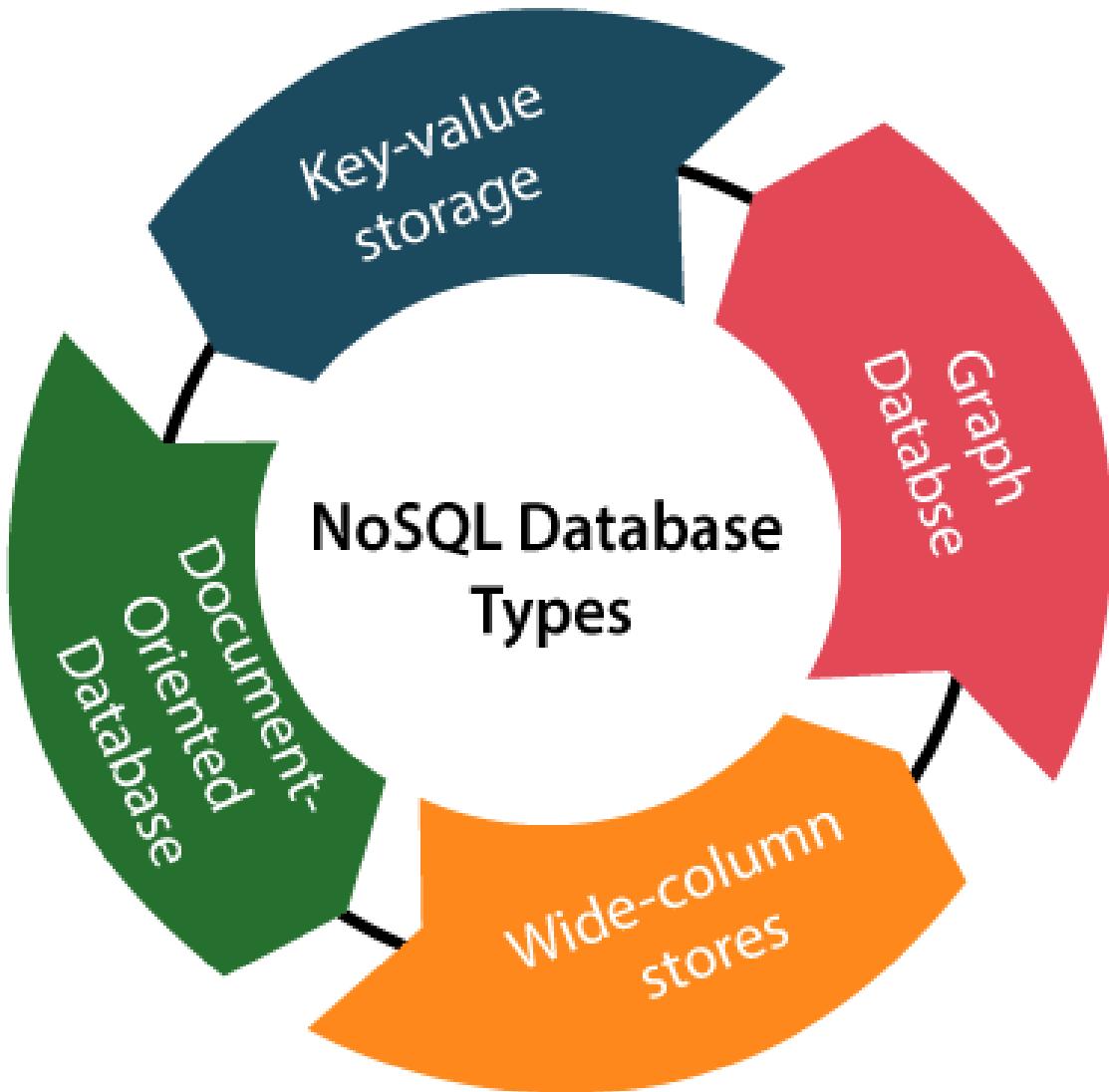
C means Consistency: If we perform any operation over the data, its value before and after the operation should be preserved. For example, the account balance before and after the transaction should be correct, i.e., it should remain conserved.

I means Isolation: There can be concurrent users for accessing data at the same time from the database. Thus, isolation between the data should remain isolated. For example, when multiple transactions occur at the same time, one transaction effects should not be visible to the other transactions in the database.

D means Durability: It ensures that once it completes the operation and commits the data, data changes should remain permanent.

4) NoSQL Database

Non-SQL/Not Only SQL is a type of database that is used for storing a wide range of data sets. It is not a relational database as it stores data not only in tabular form but in several different ways. It came into existence when the demand for building modern applications increased. Thus, NoSQL presented a wide variety of database technologies in response to the demands. We can further divide a NoSQL database into the following four types:



- a. **Key-value storage:** It is the simplest type of database storage where it stores every single item as a key (or attribute name) holding its value, together.
- b. **Document-oriented Database:** A type of database used to store data as JSON-like document. It helps developers in storing data by using the same document-model format as used in the application code.
- c. **Graph Databases:** It is used for storing vast amounts of data in a graph-like structure. Most commonly, social networking websites use the graph database.
- d. **Wide-column stores:** It is similar to the data represented in relational databases. Here, data is stored in large columns together, instead of storing in rows.

Advantages of NoSQL Database

- It enables good productivity in the application development as it is not required to store data in a structured format.
- It is a better option for managing and handling large data sets.
- It provides high scalability.
- Users can quickly access data from the database through key-value.

5) Cloud Database

A type of database where data is stored in a virtual environment and executes over the cloud computing platform. It provides users with various cloud computing services (SaaS, PaaS, IaaS, etc.) for accessing the database. There are numerous cloud platforms, but the best options are:

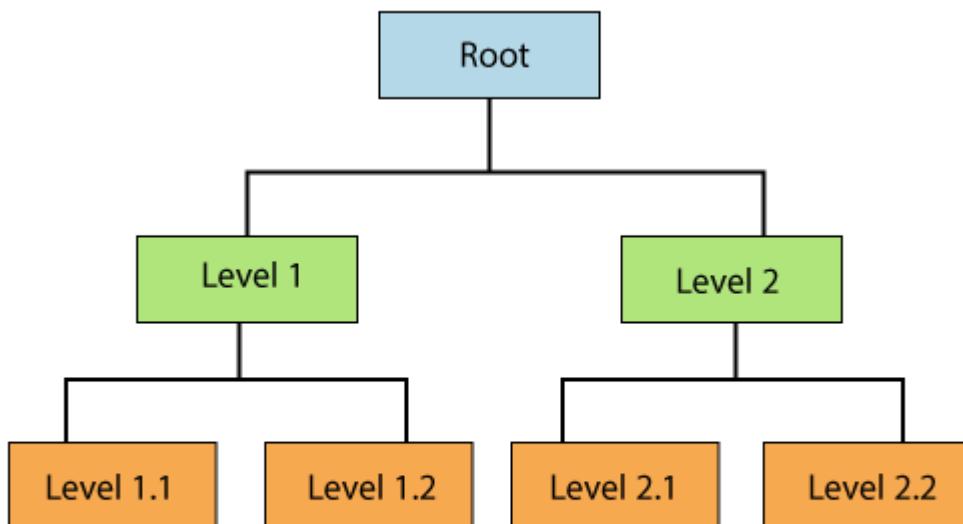
- Amazon Web Services(AWS)
- Microsoft Azure
- Kamatera
- PhonixNAP
- ScienceSoft
- Google Cloud SQL, etc.

6) Object-oriented Databases

The type of database that uses the object-based data model approach for storing data in the database system. The data is represented and stored as objects which are similar to the objects used in the object-oriented programming language.

7) Hierarchical Databases

It is the type of database that stores data in the form of parent-children relationship nodes. Here, it organizes data in a tree-like structure.



Hierarchical Database

Data get stored in the form of records that are connected via links. Each child record in the tree will contain only one parent. On the other hand, each parent record can have multiple child records.

8) Network Databases

It is the database that typically follows the network data model. Here, the representation of data is in the form of nodes connected via links between them. Unlike the hierarchical database, it allows each record to have multiple children and parent nodes to form a generalized graph structure.

9) Personal Database

Collecting and storing data on the user's system defines a Personal Database. This database is basically designed for a single user.

Advantage of Personal Database

- It is simple and easy to handle.
- It occupies less storage space as it is small in size.

10) Operational Database

The type of database which creates and updates the database in real-time. It is basically designed for executing and handling the daily data operations in

several businesses. For example, An organization uses operational databases for managing per day transactions.

11) Enterprise Database

Large organizations or enterprises use this database for managing a massive amount of data. It helps organizations to increase and improve their efficiency. Such a database allows simultaneous access to users.

Advantages of Enterprise Database:

- Multi processes are supportable over the Enterprise database.
- It allows executing parallel queries on the system.

What is RDBMS (Relational Database Management System)

RDBMS stands for *Relational Database Management System*.

All modern database management systems like SQL, MS SQL Server, IBM DB2, ORACLE, My-SQL, and Microsoft Access are based on RDBMS.

It is called Relational Database Management System (RDBMS) because it is based on the relational model introduced by E.F. Codd.

How it works

Data is represented in terms of tuples (rows) in RDBMS.

A relational database is the most commonly used database. It contains several tables, and each table has its primary key.

Due to a collection of an organized set of tables, data can be accessed easily in RDBMS.

Brief History of RDBMS

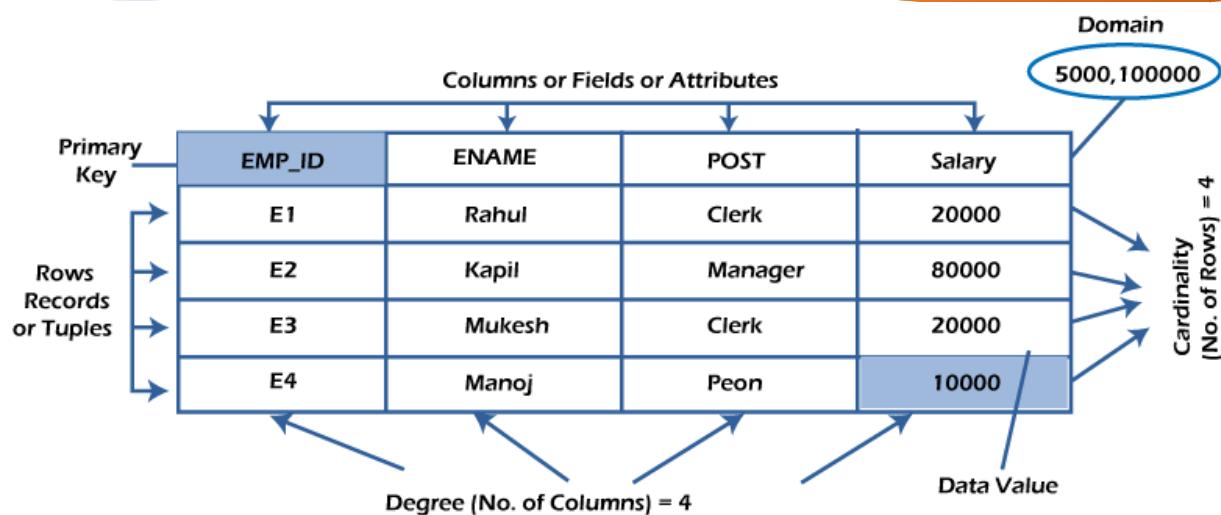
From 1970 to 1972, E.F. Codd published a paper to propose using a relational database model.

RDBMS is originally based on E.F. Codd's relational model invention.

Following are the various terminologies of RDBMS:

Data Analytics

Student Material



What is table/Relation?

Everything in a relational database is stored in the form of relations. The RDBMS database uses tables to store data. A table is a collection of related data entries and contains rows and columns to store data. Each table represents some real-world objects such as person, place, or event about which information is collected. The organized collection of data into a relational table is known as the logical view of the database.

Properties of a Relation:

- Each relation has a unique name by which it is identified in the database.
- Relation does not contain duplicate tuples.
- The tuples of a relation have no specific order.
- All attributes in a relation are atomic, i.e., each cell of a relation contains exactly one value.

A table is the simplest example of data stored in RDBMS.

Let's see the example of the student table.

ID	Name	AGE	COURSE
1	Ajeet	24	B.Tech
2	aryan	20	C.A
3	Mahesh	21	BCA

4	Ratan	22	MCA
5	Vimal	26	BSC

What is a row or record?

A row of a table is also called a record or tuple. It contains the specific information of each entry in the table. It is a horizontal entity in the table. For example, The above table contains 5 records.

Properties of a row:

- No two tuples are identical to each other in all their entries.
- All tuples of the relation have the same format and the same number of entries.
- The order of the tuple is irrelevant. They are identified by their content, not by their position.

Let's see one record/row in the table.

ID	Name	AGE	COURSE
1	Ajeet	24	B.Tech

What is a column/attribute?

A column is a vertical entity in the table which contains all information associated with a specific field in a table. For example, "name" is a column in the above table which contains all information about a student's name.

Properties of an Attribute:

- Every attribute of a relation must have a name.
- Null values are permitted for the attributes.
- Default values can be specified for an attribute automatically inserted if no other value is specified for an attribute.
- Attributes that uniquely identify each tuple of a relation are the primary key.

Name

Ajeet

Aryan

Mahesh

Ratan

Vimal

What is data item/Cells?

The smallest unit of data in the table is the individual data item. It is stored at the intersection of tuples and attributes.

Properties of data items:

- Data items are atomic.
- The data items for an attribute should be drawn from the same domain.

In the below example, the data item in the student table consists of Ajeet, 24 and Btech, etc.

ID	Name	AGE	COURSE
1	Ajeet	24	B.Tech

Degree:

The total number of attributes that comprise a relation is known as the degree of the table.

For example, the student table has 4 attributes, and its degree is 4.

ID	Name	AGE	COURSE
1	Ajeet	24	B.Tech

Data Analytics

Student Material

2	aryan	20	C.A
3	Mahesh	21	BCA
4	Ratan	22	MCA
5	Vimal	26	BSC

Cardinality:

The total number of tuples at any one time in a relation is known as the table's cardinality. The relation whose cardinality is 0 is called an empty table.

For example, the student table has 5 rows, and its cardinality is 5.

ID	Name	AGE	COURSE
1	Ajeet	24	B.Tech
2	aryan	20	C.A
3	Mahesh	21	BCA
4	Ratan	22	MCA
5	Vimal	26	BSC

Domain:

The domain refers to the possible values each attribute can contain. It can be specified using standard data types such as integers, floating numbers, etc. **For example,** An attribute entitled Marital_Status may be limited to married or unmarried values.

NULL Values

The NULL value of the table specifies that the field has been left blank during record creation. It is different from the value filled with zero or a field that contains space.

Data Integrity

There are the following categories of data integrity exist with each RDBMS:

Entity integrity: It specifies that there should be no duplicate rows in a table.

Domain integrity: It enforces valid entries for a given column by restricting the type, the format, or the range of values.

Referential integrity specifies that rows cannot be deleted, which are used by other records.

User-defined integrity: It enforces some specific business rules defined by users. These rules are different from the entity, domain, or referential integrity.

Difference between DBMS and RDBMS

Although DBMS and RDBMS both are used to store information in physical database but there are some remarkable differences between them.

The main differences between DBMS and RDBMS are given below:

No.	DBMS	RDBMS
1)	DBMS applications store data as file .	RDBMS applications store data in a tabular form .
2)	In DBMS, data is generally stored in either a hierarchical form or a navigational form.	In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables.
3)	Normalization is not present in DBMS.	Normalization is present in RDBMS.
4)	DBMS does not apply any security with regards to data manipulation.	RDBMS defines the integrity constraint for purpose of ACID (Atomocity, Consistency, Isolation and Durability) property.
5)	DBMS uses file system to store data, so there will be no relation between the tables .	in RDBMS, data values are stored in the form of tables, so a relationship between these data values will be stored in the form of a table as well.

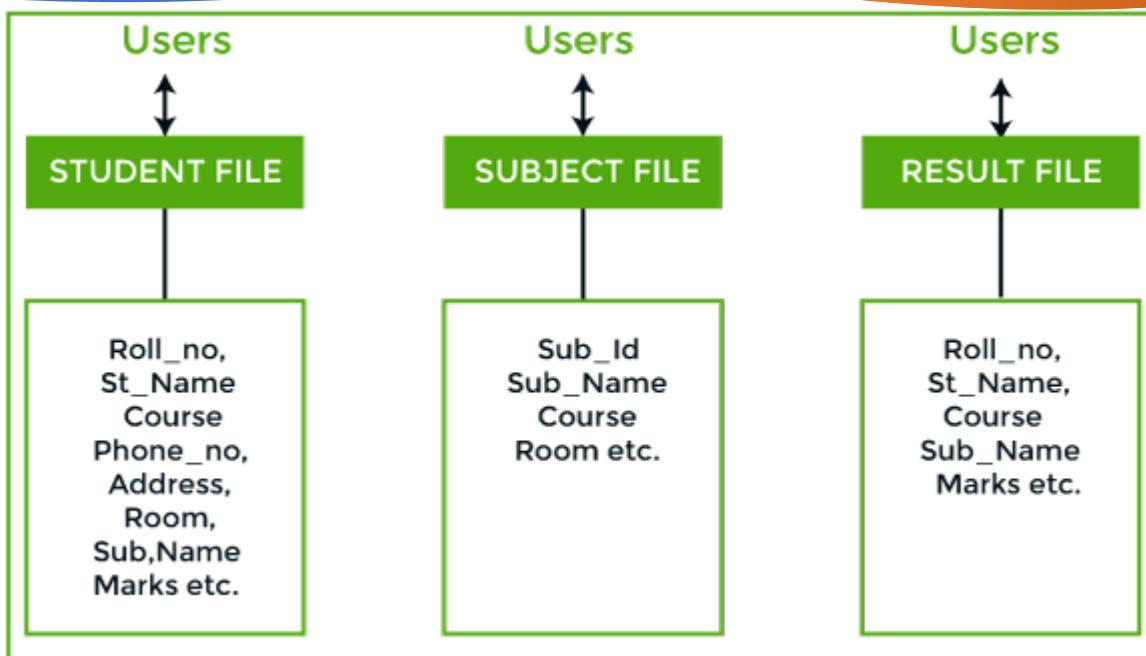
6)	DBMS has to provide some uniform methods to access the stored information.	RDBMS system supports a tabular structure of data and a relationship between them to access stored information.
7)	DBMS does not support distributed database.	RDBMS supports distributed database.
8)	DBMS is meant to be for small organization and deal with small data . it supports single user .	RDBMS is designed to handle large amount of data it supports multiple users .
9)	Examples of DBMS are file systems, xml etc.	Example of RDBMS are mysql, postgresql, server, oracle etc.

After observing the differences between DBMS and RDBMS, you can say that RDBMS is an extension of DBMS. There are many software products in the market today who are compatible for both DBMS and RDBMS. Means today a RDBMS application is DBMS application and vice-versa.

Difference between File System and DBMS

File System Approach

File based systems were an early attempt to computerize the manual system. It is also called a traditional based approach in which a decentralized approach was taken where each department stored and controlled its own data with the help of a data processing specialist. The main role of a data processing specialist was to create the necessary computer file structures, and also manage the data within structures and design some application programs that create reports based on file data.



In the above figure:

Consider an example of a student's file system. The student file will contain information regarding the student (i.e. roll no, student name, course etc.). Similarly, we have a subject file that contains information about the subject and the result file which contains the information regarding the result.

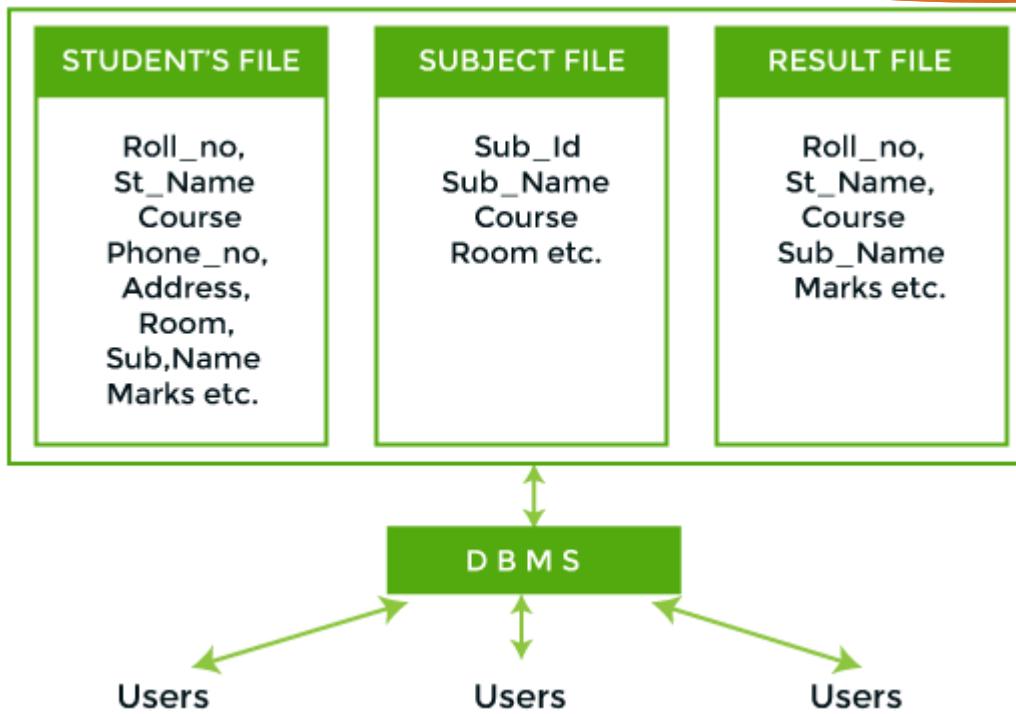
Some fields are duplicated in more than one file, which leads to data redundancy. So to overcome this problem, we need to create a centralized system, i.e. DBMS approach.

DBMS:

A database approach is a well-organized collection of data that are related in a meaningful way which can be accessed by different users but stored only once in a system. The various operations performed by the DBMS system are: Insertion, deletion, selection, sorting etc.

Data Analytics

Student Material



In the above figure,

In the above figure, duplication of data is reduced due to centralization of data.

There are the following differences between DBMS and File systems:

Basis	DBMS Approach	File System Approach
Meaning	DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	The file system is a collection of data. In this system, the user has to write the procedures for managing the database.
Sharing of data	Due to the centralized approach, data sharing is easy.	Data is distributed in many files, it may be of different formats, so it isn't easy to share data.
Data Abstraction	DBMS gives an abstract view of data that hides the details.	The file system provides the detailed view of the data representation and storage of data.

Data Analytics

Student Material

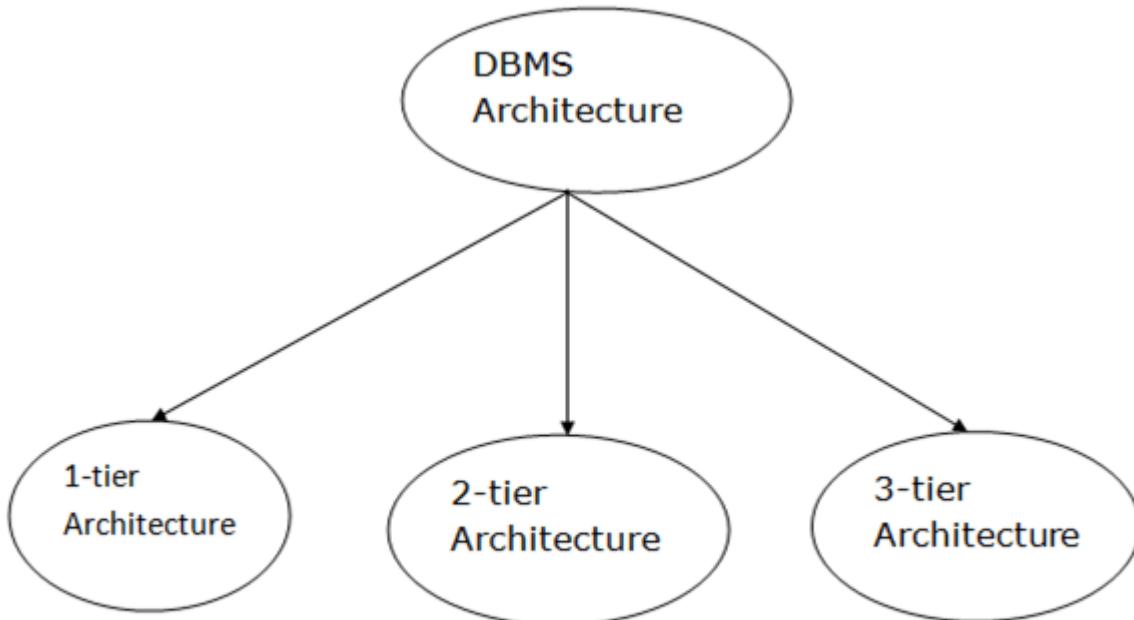
Security and Protection	DBMS provides a good protection mechanism.	It isn't easy to protect a file under the file system.
Recovery Mechanism	DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from system failure.	The file system doesn't have a crash recovery mechanism, i.e., if the system crashes while entering some data, then the content of the file will be lost.
Manipulation Techniques	DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	The file system can't efficiently store and retrieve the data.
Concurrency Problems	DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while deleting some information or updating some information.
Where to use	Database approach used in large systems which interrelate many files.	File system approach used in small systems which interrelate many files.
Cost	The database system is expensive to design.	The file system approach is cheaper to design.
Data Redundancy and Inconsistency	Due to the centralization of the database, the problems of data redundancy and inconsistency are controlled.	In this, the files and application programs are created by different programmers so that there exists a lot of duplication of data which lead to inconsistency.
Structure	The database structure is complex to design.	The file system approach has a simple structure.
Data Independence	In this system, Data Independence exists, and it can be of two types.	In the File system approach, there exists no Data Independence.

	<ul style="list-style-type: none"> ○ Logical Data Independence ○ Physical Data Independence 	
Integrity Constraints	Integrity Constraints are easy to apply.	Integrity Constraints are difficult to implement in file system.
Data Models	<p>In the database approach, 3 types of data models exist:</p> <ul style="list-style-type: none"> ○ Hierarchical data models ○ Network data models ○ Relational data models 	In the file system approach, there is no concept of data models exists.
Flexibility	Changes are often a necessity to the content of the data stored in any system, and these changes are more easily with a database approach.	The flexibility of the system is less compared to the DBMS approach.
Examples	Oracle, SQL Server, Sybase etc.	Cobol, C++ etc.

DBMS Architecture

- The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
- The client/server architecture consists of many PCs and a workstation which are connected via the network.
- DBMS architecture depends upon how users are connected to the database to get their request done.

Types of DBMS Architecture



Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like: **2-tier architecture** and **3-tier architecture**.

1-Tier Architecture

- In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
- Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
- The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

2-Tier Architecture

- The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC, JDBC** are used.
- The user interfaces and application programs are run on the client-side.

- The server side is responsible to provide the functionalities like: query processing and transaction management.
- To communicate with the DBMS, client-side application establishes a connection with the server side.

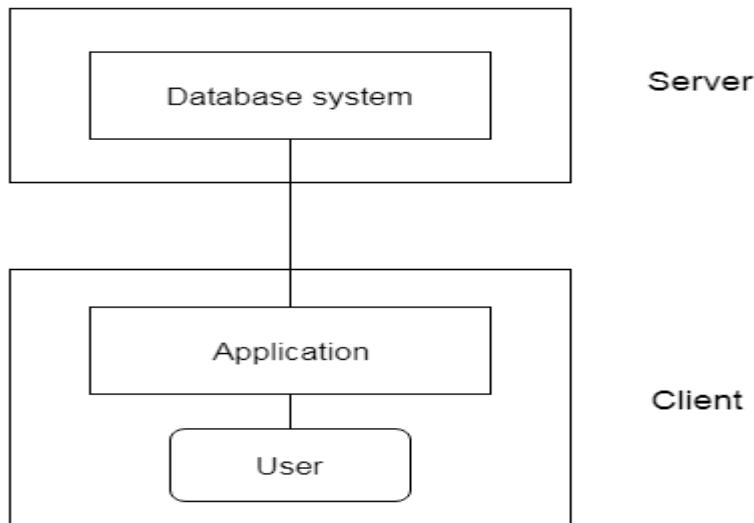


Fig: 2-tier Architecture

3-Tier Architecture

- The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
- The application on the client-end interacts with an application server which further communicates with the database system.
- End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
- The 3-Tier architecture is used in case of large web application.

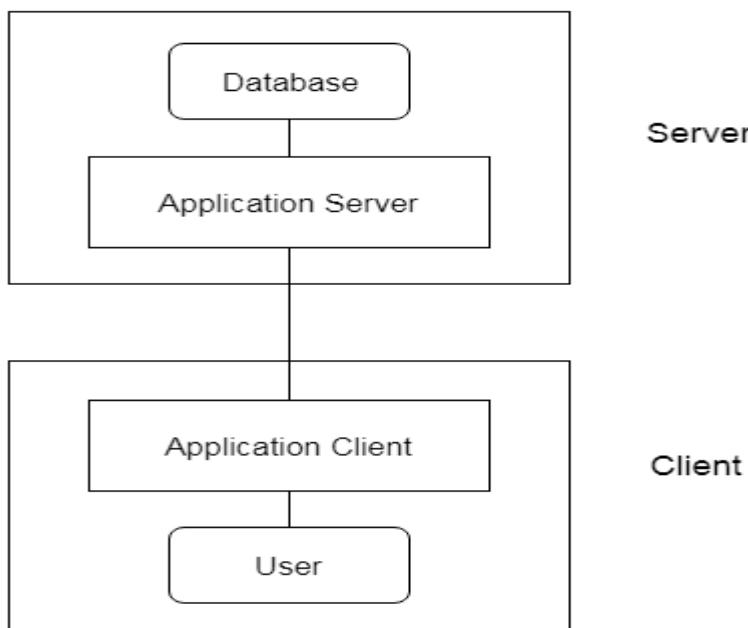
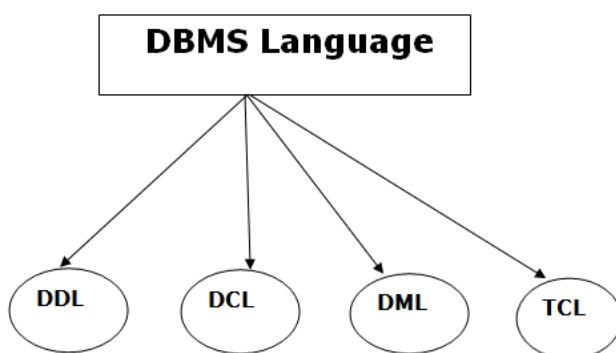


Fig: 3-tier Architecture

Database Languages in DBMS

- A DBMS has appropriate languages and interfaces to express database queries and updates.
- Database languages can be used to read, store and update the data in the database.



Types of Database Languages

1. Data Definition Language (DDL)

- **DDL** stands for **Data Definition Language**. It is used to define database structure or pattern.

- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Here are some tasks that come under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.
- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.
- **Comment:** It is used to comment on the data dictionary.

These commands are used to update the database schema that's why they come under Data definition language.

2. Data Manipulation Language (DML)

DML stands for **Data Manipulation Language**. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.
- **Insert:** It is used to insert data into a table.
- **Update:** It is used to update existing data within a table.
- **Delete:** It is used to delete all records from a table.
- **Merge:** It performs UPSERT operation, i.e., insert or update operations.
- **Call:** It is used to call a structured query language or a Java subprogram.
- **Explain Plan:** It has the parameter of explaining data.
- **Lock Table:** It controls concurrency.

3. Data Control Language (DCL)

- **DCL** stands for **Data Control Language**. It is used to retrieve the stored or saved data.

- The DCL execution is transactional. It also has rollback parameters.
(But in Oracle database, the execution of data control language does not have the feature of rolling back.)

Here are some tasks that come under DCL:

- **Grant:** It is used to give user access privileges to a database.
- **Revoke:** It is used to take back permissions from the user.

There are the following operations which have the authorization of Revoke:

CONNECT, INSERT, USAGE, EXECUTE, DELETE, UPDATE and SELECT.

4. Transaction Control Language (TCL)

TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

- **Commit:** It is used to save the transaction on the database.
- **Rollback:** It is used to restore the database to original since the last Commit.

DATA ENGINEERING SERVICES

Sharing top billing on the list of data science capabilities, machine learning and artificial intelligence are not just buzzwords: Many organizations are eager to adopt them. But prior to building intelligent products, you need to gather and prepare data, that fuels AI. A separate discipline — data engineering, lays the necessary groundwork for analytics projects. Tasks related to it occupy the first three layers of the data science hierarchy of needs suggested by Monica Rogati.

THE DATA SCIENCE HIERARCHY OF NEEDS

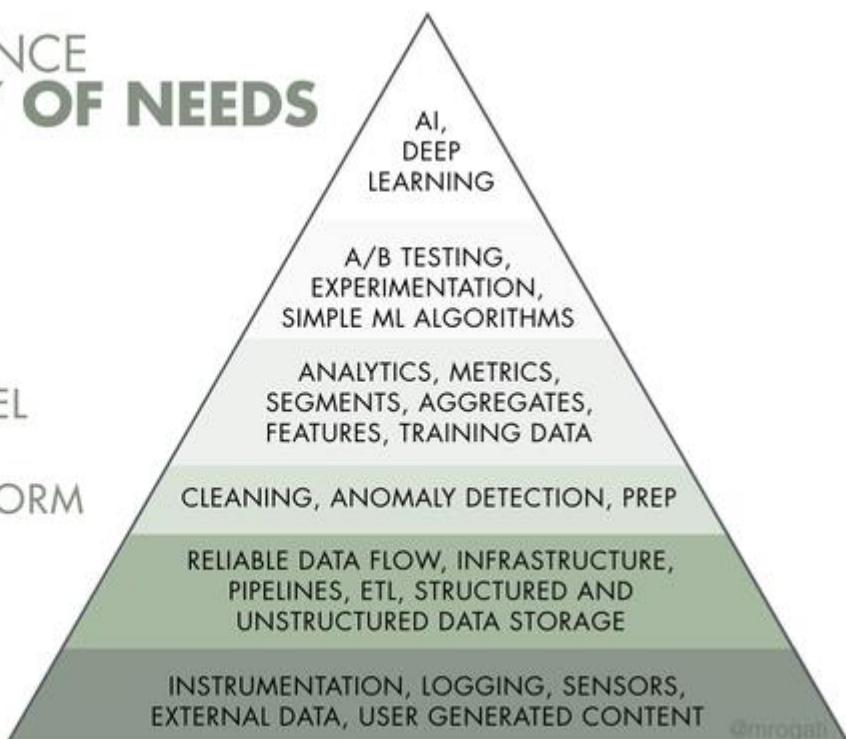
LEARN/OPTIMIZE

AGGREGATE/LABEL

EXPLORE/TRANSFORM

MOVE/STORE

COLLECT



What is data engineering?

Data engineering is a set of operations to make data available and usable to data scientists, data analysts, business intelligence (BI) developers, and other specialists within an organization. It takes dedicated experts – data engineers – to design and build systems for gathering and storing data at scale as well as preparing it for further analysis.

You may watch our video explainer on data engineering:

Within a large organization, there are usually many different types of operations management software (e.g., ERP, CRM, production systems, etc.), all containing databases with varied information. Besides, data can be stored as separate files or pulled from external sources — such as IoT devices — in

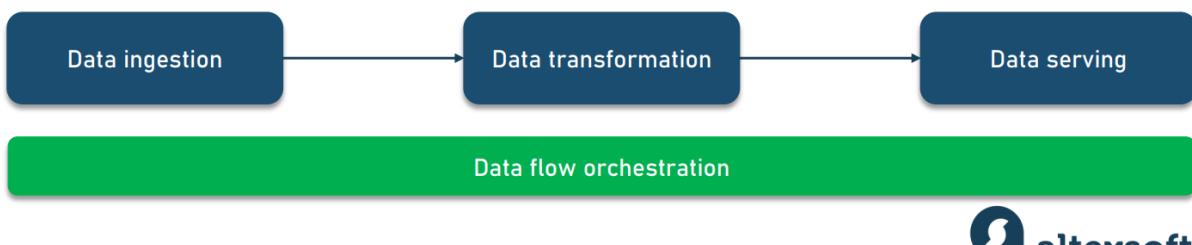
real time. Having data scattered in different formats prevents the organization from seeing a clear picture of its business state and running analytics.

Data engineering addresses this problem step by step.

Data engineering process

The data engineering process covers a sequence of tasks that turn a large amount of raw data into a practical product meeting the needs of analysts, data scientists, machine learning engineers, and others. Typically, the end-to-end workflow consists of the following stages.

DATA ENGINEERING PROCESS



Data ingestion (acquisition) moves data from multiple sources — SQL and NoSQL databases, IoT devices, websites, streaming services, etc. — to a target system to be transformed for further analysis. Data comes in various forms and can be both structured and unstructured.

Data transformation adjusts disparate data to the needs of end users. It involves removing errors and duplicates from data, normalizing it, and converting it into the needed format.

Data serving delivers transformed data to end users — a BI platform, dashboard, or data science team.

Data flow orchestration provides visibility into the data engineering process, ensuring that all tasks are successfully completed. It coordinates and continuously tracks data workflows to detect and fix data quality and performance issues.

The mechanism that automates ingestion, transformation, and serving steps of the data engineering process is known as a data pipeline.

Data engineering pipeline

A **data pipeline** combines tools and operations that move data from one system to another for storage and further handling. Constructing and maintaining data pipelines is the core responsibility of data engineers. Among other things, they write scripts to automate repetitive tasks – *jobs*.

Commonly, pipelines are used for

- data migration between systems or environments (from on-premises to cloud databases);
- data wrangling or converting raw data into a usable format for analytics, BI, and machine learning projects;
- data integration from various systems and IoT devices; and
- copying tables from one database to another.

To learn more, read our detailed explanatory post — Data Pipeline: Components, Types, and Use Cases. Or stay here to briefly explore common types of data pipelines.

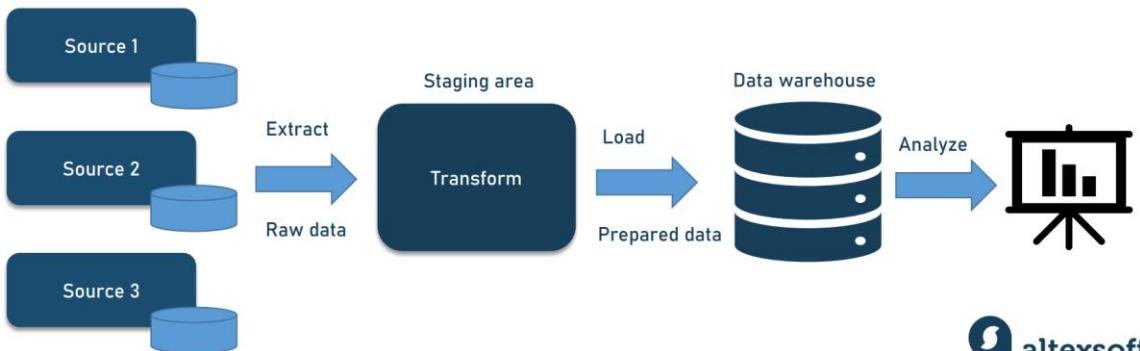
ETL pipeline

ETL (Extract, Transform, Load) pipeline is the most common architecture that has been here for decades. It's often implemented by a dedicated specialist — an ETL developer.

As the name suggests, an ETL pipeline automates the following processes.

1. Extract — retrieving data. At the start of the pipeline, we're dealing with raw data from numerous sources — databases, APIs, files, etc.
2. Transform — standardizing data. Having data extracted, scripts transform it to meet the format requirements. Data transformation significantly improves data discoverability and usability.
3. Load — saving data to a new destination. After bringing data into a usable state, engineers can load it to the destination, typically a database management system (DBMS) or data warehouse.

ETL PIPELINE



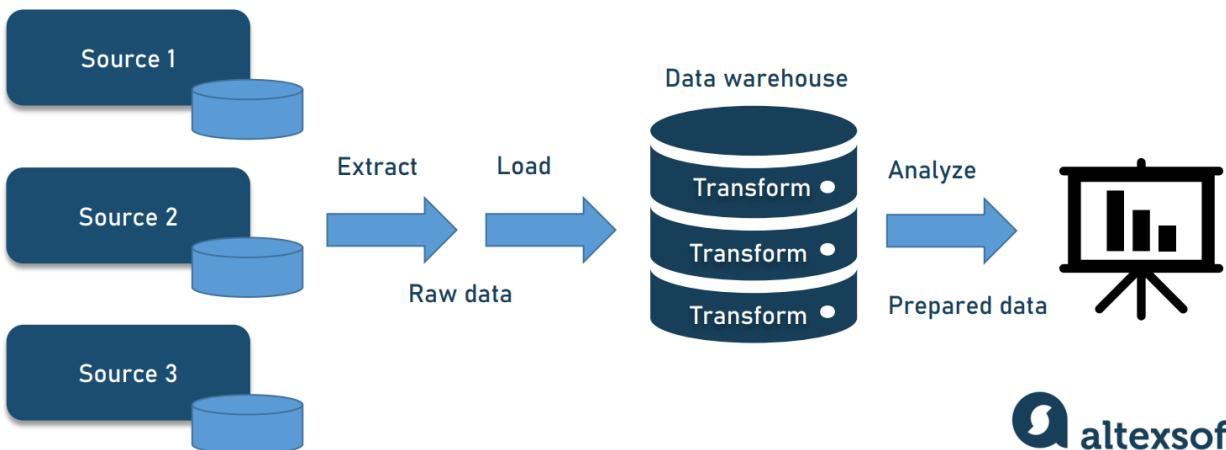
 altexsoft

Once the data is transformed and loaded into a centralized repository, it can be used for further analysis and business intelligence operations, i.e., generating reports, creating visualizations, etc. The specialist implementing ETL pipelines

ELT pipeline

An ELT pipeline performs the same steps but in a different order — Extract, Load, Transform. Instead of transforming all the collected data, you place it into a data warehouse, data lake, or data lakehouse. Later, you can process and format it fully or partially, once or numerous times.

ELT PIPELINE



 altexsoft

ELT pipelines are preferable when you want to ingest as much data as possible and transform it later, depending on the needs arising. Unlike ETL, the ELT architecture doesn't require you to decide on data types and formats in advance. In large-scale projects, two types of data pipelines are often

combined to enable both traditional and real-time analytics. Also, two architectures can be involved to support Big Data analytics.

Read our article ETL vs ELT: Key differences to dive deeper into the subject.

Data pipeline challenges

Setting up secure and reliable data flow is challenging. Many things can go wrong during data transportation: Data can be corrupted, hit bottlenecks causing latency, or data sources may conflict, generating duplicate or incorrect data. Getting data into one place requires careful planning and testing to filter out junk data, eliminating duplicates and incompatible data types to obfuscate sensitive information while not missing critical data.

Juan De Dios Santos, an experienced practitioner in the data industry, outlines two major pitfalls in building data pipelines:

- lacking relevant metrics and
- underestimating data load.

"The importance of a healthy and relevant metrics system is that it can inform us of the status and performance of each pipeline stage while underestimating the data load, I am referring to building the system in such a way that it won't face any overload if the product experiences an unexpected surge of users," elaborates Juan.

Besides a pipeline, a data warehouse must be built to support and facilitate data science activities. Let's see how it works.

Data warehouse

A **data warehouse** (DW) is a central repository storing data in queryable forms. From a technical standpoint, a data warehouse is a relational database optimized for reading, aggregating, and querying large volumes of data. Traditionally, DWs only contained structured data or data that can be arranged in tables. However, modern DWs can also support unstructured data (such as images, pdf files, and audio formats).

Without DWs, data scientists would have to pull data straight from the production database and may report different results to the same question or cause delays and even outages. Serving as an enterprise's single source of

truth, the data warehouse simplifies the organization's reporting and analysis, decision-making, and metrics forecasting.

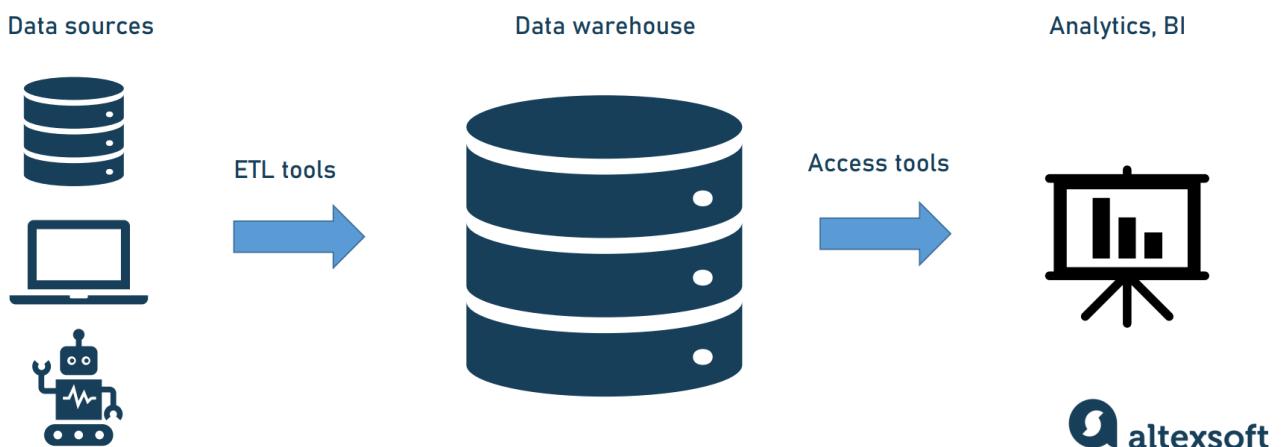
Surprisingly, DW isn't a regular database. How so?

First of all, they differ in terms of data structure. A typical database normalizes data excluding any data redundancies and separating related data into tables. This takes up a lot of computing resources, as a single query combines data from many tables. Contrarily, a DW uses simple queries with few tables to improve performance and analytics.

Second, aimed at day-to-day transactions, standard transactional databases don't usually store historical data, while for warehouses, it's their primary purpose, as they collect data from multiple periods. DW simplifies a data analyst's job, allowing for manipulating all data from a single interface and deriving analytics, visualizations, and statistics.

Typically, a data warehouse doesn't support as many concurrent users as a database designed for a small group of analysts and decision-makers.

DATA WAREHOUSE ARCHIRECTURE



To construct a data warehouse, four essential components are combined.

Data warehouse storage. The foundation of data warehouse architecture is a database that stores all enterprise data allowing business users to access it to draw valuable insights.

Data architects usually decide between on-premises and cloud-hosted DWs noting how the business can benefit from this or that solution. Although the cloud environment is more cost-efficient, easier to scale up or down, and isn't limited to a prescribed structure, it may lose to on-prem solutions regarding querying speed and security. We're going to list the most popular tools further on.

A data architect can also design collective storage for your data warehouse – multiple databases running in parallel. This will improve the warehouse's scalability.

Metadata. Adding business context to data, metadata helps transform it into comprehensible knowledge. Metadata defines how data can be changed and processed. It contains information about any transformations or operations applied to source data while loading it into the data warehouse.

Data warehouse access tools. These instruments vary in functionality. For example, *query and reporting tools* are used for generating business analysis reports. And *data mining tools* automate finding patterns and correlations in large amounts of data based on advanced statistical modeling techniques.

Data warehouse management tools. Spanning the enterprise, the data warehouse deals with a number of management and administrative operations. Dedicated data warehouse management tools exist to accomplish this.

For more detailed information, visit our dedicated post — Enterprise Data Warehouse: EDW Components, Key Concepts, and Architecture Types.

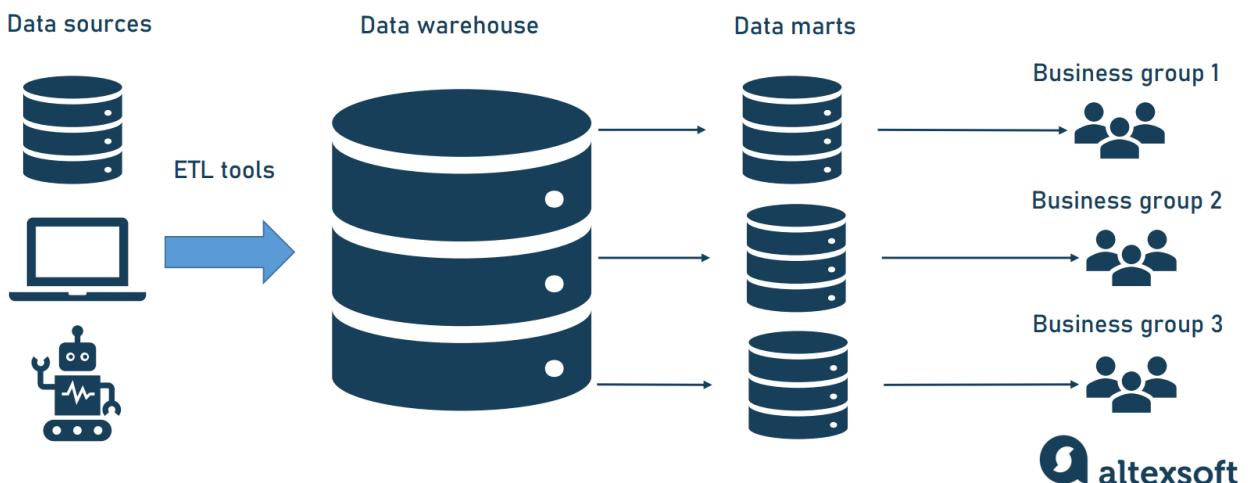
Data warehouses are a significant step forward in enhancing your data architecture. However, DWs can be too bulky and slow to operate if you have hundreds of users from different departments. In this case, data marts can be built and implemented to increase speed and efficiency.

Data marts

Simply speaking, a **data mart** is a smaller data warehouse (their size is usually less than 100Gb.). They become necessary when the company and the amount of its data grow and it becomes too long and ineffective to search for information in an enterprise DW. Instead, data marts are built to allow

different departments (e.g., sales, marketing, C-suite) to access relevant information quickly and easily.

DATA INFRASTRUCTURE WITH DATA MARTS



There are three main types of data marts.

Dependent data marts are created from an enterprise DW and used as a primary source of information (also known as a top-down approach).

Independent data marts are standalone systems that function without DWs extracting information from various external and internal sources (it's also known as a top-down approach).

Hybrid data marts combine information from both DW and other operational systems.

So, the main difference between data warehouses and data marts is that a DW is a large repository that holds all company data extracted from multiple sources, making it difficult to process and manage queries. Meanwhile, a data mart is a smaller repository containing a limited amount of data for a particular business group or department.

If you want to learn more, read our comprehensive overview — [Data Marts: What They Are and Why Businesses Need Them](#).

While data marts allow business users to quickly access the queried data, often just getting the information is not enough. It has to be efficiently processed and analyzed to get actionable insights that support decision-making. Looking at your data from different perspectives is possible thanks to OLAP cubes. Let's see what they are.

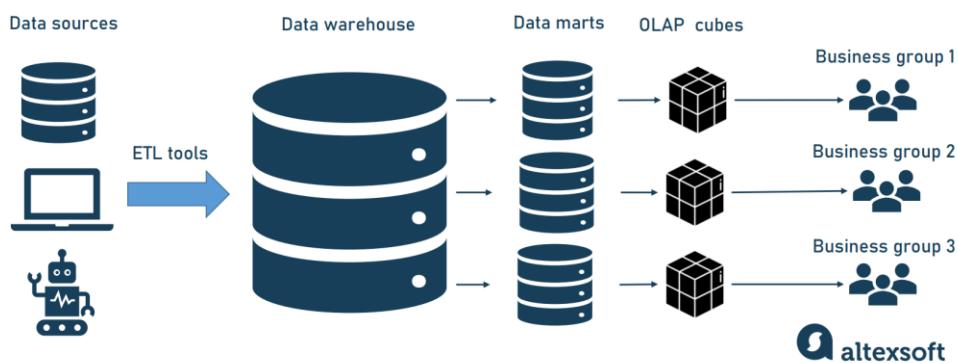
OLAP and OLAP cubes

OLAP or Online Analytical Processing refers to the computing approach allowing users to analyze multidimensional data. It's contrasted with *OLTP or Online Transactional Processing*, a simpler method of interacting with databases, not designed for analyzing massive amounts of data from different perspectives.

Traditional databases resemble spreadsheets, using the two-dimensional structure of rows and columns. However, in OLAP, datasets are presented in multidimensional structures — **OLAP cubes**. Such structures enable efficient processing and advanced analysis of vast amounts of varied data. For example, a sales department report would include such dimensions as product, region, sales representative, sales amount, month, and so on.

Information from DWs is aggregated and loaded into the OLAP cube, where it gets precalculated and is readily available for user requests.

DATA INFRASTRUCTURE WITH DATA MARTS AND OLAP CUBES



Within OLAP, data can be analyzed from multiple perspectives. For example, it can be drilled down/rolled up if you need to change the hierarchy level of data representation and get a more or less detailed picture. You can also slice information to segment a particular dataset as a separate spreadsheet or dice

it to create a different cube. These and other techniques enable finding patterns in varied data and creating a wide range of reports.

It's important to note that OLAP cubes must be custom-built for every report or analytical query. However, their usage is justified since, as we said, they facilitate advanced, multidimensional analysis that was previously too complicated to perform.

Read our article [What is OLAP: A Complete Guide to Online Analytical Processing](#) for a more detailed explanation.

Big data engineering

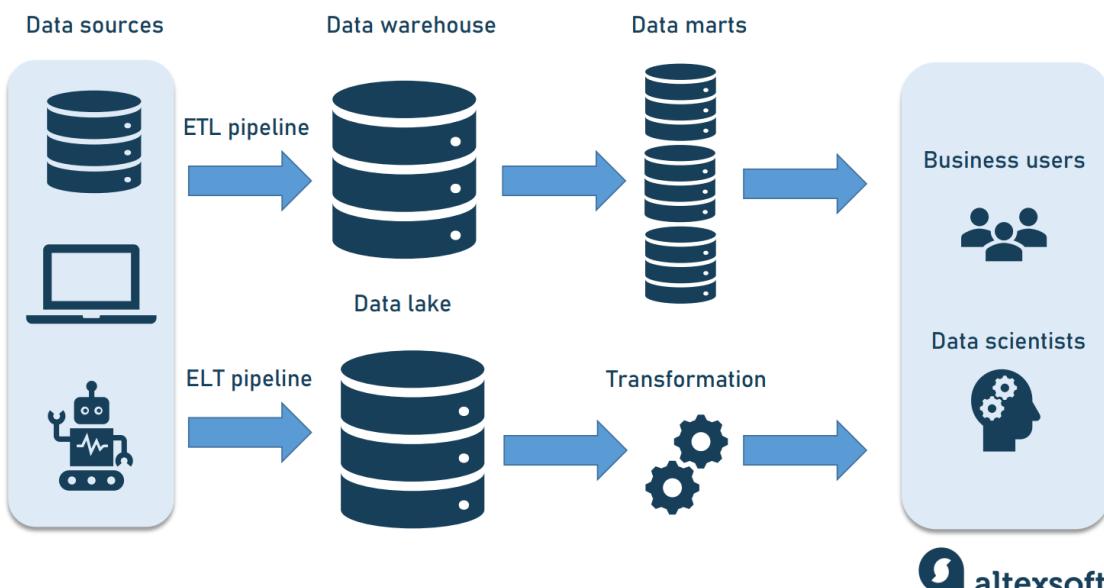
Speaking about data engineering, we can't ignore Big Data. Grounded in the four Vs – volume, velocity, variety, and veracity – it usually floods large technology companies like YouTube, Amazon, or Instagram. Big Data engineering is about building massive reservoirs and highly scalable and fault-tolerant distributed systems.

Big data architecture differs from conventional data handling, as here we're talking about such massive volumes of rapidly changing information streams that a data warehouse can't accommodate. That's where a data lake comes in handy.

Data lake

A **data lake** is a vast pool for saving data in its native, unprocessed form. It stands out for its high agility as it isn't limited to a warehouse's fixed configuration.

BIG DATA INFRASTRUCTURE WITH DATA LAKE



A data lake uses the ELT approach and starts data loading immediately after extracting it, handling raw — often unstructured — data.

A data lake is worth building in those projects that will scale and need a more advanced architecture. Besides, it's very convenient when the purpose of the data hasn't been determined yet. In this case, you can load data quickly, store it, and modify it as necessary.

Data lakes are also a powerful tool for data scientists and ML engineers, who would use raw data to prepare it for predictive analytics and machine learning. Read more about data preparation in our separate article or watch this 14-minute explainer.

Lakes are built on large, distributed clusters that would be able to store and process masses of data. A famous example of such a data lake platform is Hadoop.

Hadoop and its ecosystem

Hadoop is a large-scale, Java-based data processing framework capable of analyzing massive datasets. The platform facilitates splitting data analysis jobs across various servers and running them in parallel. It consists of three components:

Data Analytics

Student Material

- Hadoop Distributed File System (HDFS) capable of storing Big Data,
- a processing engine MapReduce, and
- a resource manager YARN to control and monitor workloads.

Also, Hadoop benefits from a vast ecosystem of open-source tools that enhance its capabilities and address various challenges of Big Data.

HADOOP ECOSYSTEM TIMELINE



Hadoop Core	2008-2009	2010-2012	2013 and later
hadoop MapReduce Processing engine	APACHE HBASE NoSQL database	Apache Kafka Event streaming	APACHE DRILL SQL on Hadoop
hadoop HDFS File system	APACHE ZooKeeper Coordination	Log data ingest ingesting	APACHE Spark Real-time data analytics
hadoop YARN Resource management	Apache Pig SQL-like scripting	HIVE SQL-like scripting	APACHE PHOENIX SQL on HBase
	mahout Machine learning	OOZIE Scheduling	APACHE GIRAPH Graph processing
	SQOOP RDBMS to HDFS data transfer	Flink Stream and batch processing	APACHE STORM Stream processing

Some popular instruments within the Hadoop ecosystem are

- HBase, a NoSQL database built on top of HDFS that provides real-time access to read or write data;
- Apache Pig, Apache Hive, Apache Drill, and Apache Phoenix to simplify Big Data exploration and analysis when working with HBase, HDFS, and MapReduce; and
- Apache Zookeeper and Apache Oozie to coordinate operations and schedule jobs across a Hadoop cluster.

Read about the advantages and pitfalls of Hadoop in our dedicated article [The Good and the Bad of Hadoop Big Data Framework](#).

Streaming analytics instruments

Tools enabling streaming analytics form a vital group within the Hadoop ecosystem. These include

- Apache Spark, a computing engine for large datasets with near-real-time processing capabilities;
- Apache Storm, a real-time computing system for unbounded streams of data (those that have a start but no defined end and must be continuously processed);
- Apache Flink processing both unbounded and bounded data streams (those with a defined start and end); and
- Apache Kafka, a streaming platform for messaging, storing, processing, and integrating large volumes of data.

All these technologies are used to build real-time Big Data pipelines. You can get more information from our articles [Hadoop vs Spark: Main Big Data Tools Explained](#) and [The Good and the Bad of Apache Kafka Streaming Platform](#).

Enterprise data hub

When a big data pipeline is not managed correctly, data lakes quickly become data swamps – a collection of miscellaneous data that is neither governable nor usable. A new data integration approach called a data hub emerged to tackle this problem.

Enterprise data hubs (EDHs) are the next generation of data architecture aiming at sharing managed data between systems. They connect multiple sources of information, including DWs and data lakes. Unlike DWs, the data hub supports all types of data and easily integrates systems. Besides that, it can be deployed within weeks or even days while DW deployment can last months and even years.

At the same time, data hubs come with additional capabilities for data management, harmonizing, exploration, and analysis — something data lakes lack. They are business-oriented and tailored for the most urgent organization's needs.

To sum it all up,

- a *data warehouse* is constructed to deal mainly with structured data for the purpose of self-service analytics and BI;
- a *data lake* is built to deal with sizable aggregates of both structured and unstructured data to support deep learning, machine learning, and AI in general; and
- a *data hub* is created for multi-structured data portability, easier exchange, and efficient processing.

An EDH can be integrated with a DW and/or a data lake to streamline data processing and deal with these architectures' everyday challenges. Read our article [What is Data Hub: Purpose, Architecture Patterns, and Existing Solutions Overview](#) to learn more.

Role of data engineer

Now that we know what data engineering is concerned with, let's delve into the role that specializes in creating software solutions around big data –a data engineer.

Juan De Dios Santos, a data engineer himself, defines this role in the following way: "*In a multidisciplinary team that includes data scientists, BI engineers, and data engineers, the role of the data engineer is mostly to ensure the quality and availability of the data.*" He also adds that a data engineer might collaborate with others when implementing or designing a data-related feature (or product) such as an A/B test, deploying a machine learning model, and refining an existing data source.

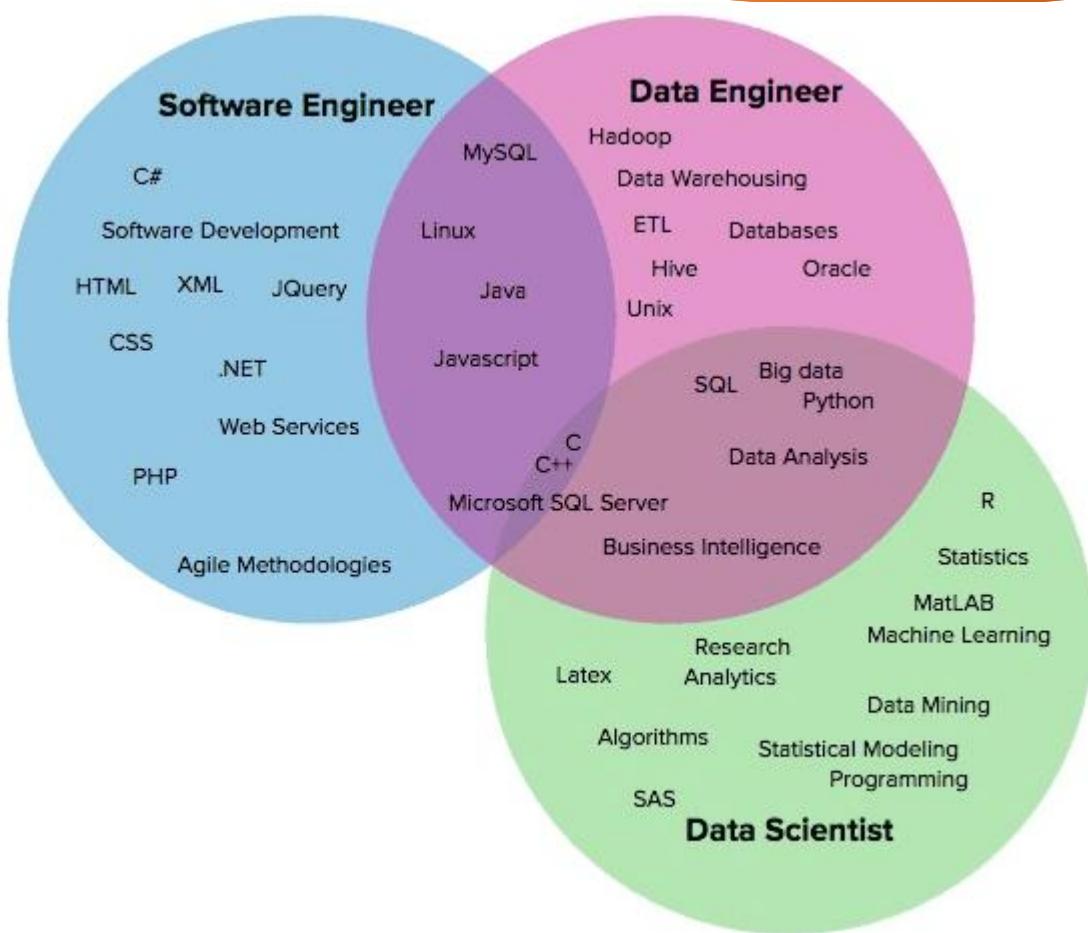
We have a separate article explaining what a data engineer is, so here we'll only briefly recap.

Skills and qualifications

Data engineering lies at the intersection of software engineering and data science, which leads to skill overlapping.

Data Analytics

Student Material



Software engineering background. Data engineers use programming languages to enable reliable and convenient access to data and databases. Juan points out their ability to work with the complete software development cycle, including ideation, architecture design, prototyping, testing, deployment, DevOps, defining metrics, and monitoring systems. Data engineers are experienced programmers in at least Python or Scala/Java.

Data-related skills. “A data engineer should have knowledge of multiple kinds of databases (SQL and NoSQL), data platforms, concepts such as MapReduce, batch and stream processing, and even some basic theory of data itself, e.g., data types, and descriptive statistics,” underlines Juan.

Systems creation skills. Data engineers need experience with various data storage technologies and frameworks they can combine to build data pipelines.

Toolkit

Data Analytics

Student Material

A data engineer should have a deep understanding of many data technologies to be able to choose the right ones for a specific job.

Airflow. This Python-based workflow management system was developed by Airbnb to rearchitect its data pipelines. Migrating to Airflow, the company reduced their experimentation reporting framework (ERF) run-time from 24+ hours to about 45 minutes. Among the Airflow's pros, Juan highlights its operators: "*They allow us to execute bash commands, run a SQL query or even send an email.*" Juan also stresses Airflow's ability to send Slack notifications, complete and rich UI, and the overall maturity of the project. On the contrary, Juan dislikes that Airflow only allows for writing jobs in Python.

Read our article [The Good and the Bad of Apache Airflow Pipeline Orchestration](#) to learn more.

Cloud Dataflow. A cloud-based data processing service, Dataflow is aimed at large-scale data ingestion and low-latency processing through fast parallel execution of the analytics pipelines. Dataflow is beneficial over Airflow as it supports multiple languages like Java, Python, SQL, and engines like Flink and Spark. It is also well maintained by Google Cloud. However, Juan warns that Dataflow's high cost might be a disadvantage.

Other popular instruments are the Stitch platform for rapidly moving data and Blendo, a tool for syncing various data sources with a data warehouse.

Warehouse solutions. Widely used on-premise data warehouse tools include Teradata Data Warehouse, SAP Data Warehouse, IBM db2, and Oracle Exadata. The most popular cloud-based data warehouse solutions are Amazon Redshift and Google BigQuery. Be sure to check our detailed comparison of the top cloud warehouse software.

Big data tools. Technologies that a data engineer should master (or at least know of) are Hadoop and its ecosystem, Elastic Stack for end-to-end big data analytics, data lakes, and more.

Data engineer vs data scientist

It's not rare that a data engineer is confused with a data scientist. We asked Alexander Konduforov, a data scientist with over ten years of experience, to comment on the difference between these roles.

Brought to you by:



"Both data scientists and data engineers work with data but solve quite different tasks, have different skills, and use different tools," Alexander explained, *"Data engineers build and maintain massive data storage and apply engineering skills: programming languages, ETL techniques, knowledge of different data warehouses and database languages. Whereas data scientists clean and analyze this data, get valuable insights from it, implement models for forecasting and predictive analytics, and mostly apply their math and algorithmic skills, machine learning algorithms and tools."*

Alexander stresses that accessing data can be a difficult task for data scientists for several reasons.

- Vast data volumes require additional effort and specific engineering solutions to access and process them in a reasonable amount of time.
- Data is usually stored in lots of different systems and formats. It makes sense first to take data preparation steps and move information to a central repository like a data warehouse makes sense. This is typically a task for data architects and engineers.
- Data repositories have different APIs for accessing them. Data scientists need data engineers to implement the most efficient and reliable pipeline for getting data.

As we can see, working with storage systems built and served by data engineers, data scientists become their "*internal clients*." That's where their collaboration takes place.

Consider that you manage an online store that is always open. Every minute or second, users can place orders and pay for the items. Your website will be processing a large number of customer transactions in real-time, including user IDs, credit card numbers, order IDs, and product and delivery details.

This all happens in the blink of an eye, thanks to online data processing systems, which include databases and platforms like e-commerce, CRM, and payment gateways.

In addition to carrying out regular tasks with data processing systems, your system should also be available to assess your business performance. For instance, you may want to analyze the sales of a particular product and compare it against the preceding month.

It implies you need to collect transactional data, process it using data processing tools, and then transfer it to the data warehouse so that analysts and other team members with access to BI interfaces can visualize the sales data.

Wait a minute... How do you transport relevant data quickly and reliably too? That is what data engineering is all about. To create a fast, productive, and efficient data pipeline, you need to have the necessary gear, software, and infrastructure built for it.

Before we go deep-dive into the processes, let's take a step back and first understand why and how data engineering systems became an integral part of our digital ecosystem.

Human-operated to System-managed Data Pipelines

A Data Pipeline is a system that captures, organizes, and routes the data to different systems, to be utilized further to perform analysis. Within a data pipeline architecture, ETL (Extract, Transform, and Load) and ELT (Extract, Load, and Transfer) are subprocesses.

In many traditional industries, the data pipeline is still manually operated by data personnel deployed to update tables on daily-basis for business analysis. It leads to numerous human errors and data breaches, especially for organizations dealing with sensitive information like banks and insurance companies.

The transactional data were typically uploaded every night to ensure that the data is available on the next day and ready for use. However, this approach in data engineering soon became a bottleneck, especially for urgent transactions where businesses and users had to wait until the following day to act on their data.

With the changing landscape of the customer ecosystem and the increasing need for speed and convenience, immediate data access has become an integral aspect of every competitive business.

The volume and the variability of the data also expanded, pushing many companies to opt for cloud storage and computing, for scaling and cost optimization. The data's engineers are also involved in constructing data pipelines to move data from on-premise data centers to cloud environments.

Today's business ETL systems must also be able to ingest, enrich, and manage transactions as well as support both structured and unstructured data in real time from any source, whether on-premises or in the cloud.

Now that we have established the need for a proper pipeline, let's take a close look into the ETL process.

What is an ETL Pipeline?

ETL, or extract, transform, and load, is a method used by data engineers to gather data from various sources, transform it into a reliable and usable resource, and then load it into the systems that end users may access and utilize later to address business-related issues.

ETL was introduced at the same time as databases gained popularity in the 1970s. The data pipeline was a way of integrating and loading data for computation and analysis. It eventually took over as the main technique for processing data for business specific data warehousing projects.

ETL process cleans and arranges data through a set of business rules in a way that satisfies particular business intelligence requirements, from as a simple monthly business report, to more advanced analytics that can enhance the back-end procedures or end-user experiences.

Extract

The initial stage in this process is to extract data from the target sources, which are typically structured and semi-structured data sets of the transaction databases, business systems, APIs, sensor data, marketing tools, and more.

Structured and Unstructured data types

Many of the day-to-day business data types are structured such as point-of-sale data, loyalty program data, transaction data, etc.

Others are semi-structured JSON server logs and unstructured data.

Data that does not cleanly fit into a normal database structure is referred to as unstructured data. Typically, it is inconsistently sourced, badly structured, and practically on Freeform. Unstructured data are frequently extracted from emails, social media posts, customer care call notes, and chats with customers on messaging platforms.

Although unstructured data is a very valuable source of pertinent data, it cannot be handled by standard data storage and regular ETL processing techniques. Separate data extraction techniques are used in the staging area before it could be stored, transformed, and joined with an already-existing structured data to start the data pipeline processing.

Transform

The second step consists of transforming the (structured and semi-structured and unstructured) data. It needs to be extracted from the sources into a format that can be used by either Business Intelligence (BI) tools or migrate the data to a repository, or replicate data as a backup. In this phase, data is cleaned, mapped, and transformed—often to a specific schema—so that it satisfies operational requirements.

To start the transformation of the unstructured data, the data engineer needs to define the associations and correlations between the information they've extracted. For example, to analyze multiple rows and columns of data, you will need to begin by matching the current table rows and columns with the rows and columns of the other table data it needs to be associated with.

In order for data sets to be what is called “transcodeable”, it needs to align enough so that it can be sensibly and clearly compared and contrasted. It doesn't have to include the exact same number of rows and columns, but it needs to at least share a basic common structure.

The fact that the data sets must be correct and full is another challenge. For example, in a healthcare use case, a precise dataset is necessary to ascertain how many staff members are available throughout the course of a given month to provide patient care. When the forecasting system tries to read a field that is blank or has falsified data, errors will be generated. The healthcare data repositories may rapidly get enmeshed in a “data swamp” if the data sets are not proper.

Moreover, the transformation process entails several types of changes that ensure the quality and integrity of data. Data is not usually loaded directly into the target data source, but instead it is common to have it uploaded initially into a staging database.

This step ensures a quick roll back in case something does not go as planned. During this stage, you have the possibility to generate audit reports for regulatory compliance or diagnose and repair any data issues.

Load

The process of writing converted data from a staging area to a target database—which may or may not have existed before—is known as the load. This procedure could be relatively straightforward or extremely complicated, depending on the requirements of the application. These processes can all be completed using ETL tools or by using custom code.

The loading procedure corresponds to importing your converted data into your data warehouse or another convenient location for your company.

You can load it all at once, which is known as full loading, or you can load data incrementally or in real-time.

Batch loading is often preferred, as it's easier to manage and doesn't put too much strain on your data warehouse or the data lake. Additionally, it also avoids data duplication by verifying the database before creating new records for any incoming data.

Challenges of ETL Pipeline

Fundamentally, data engineers created ETL tools for people who can understand and operate data tools. As a result, the methods for data transformation are often only available to IT staff and not to data consumers because they need extensive skills to install, implement, and manage.

ETL tools are doing the “power lifting” to get data into a standardized warehouse and the data consumers have access to the finished product in the Data Warehouse.

Although ETL tools function reasonably well in business specific data warehouse setups, it creates a bottleneck in processing requests from users outside the enterprise and in adding external data sources. As we'll see below, there are more challenges which ETL technique particularly struggles with.

Big Data Processing led to ETL burnout

It is estimated that around 80% of the data collected by organizations are in unstructured format and is growing every year. While the data collection

increases, only around 27% of the data is actually used for any analytics or decision-making projects.

The capacity, resources, cost, and timeliness of the conventional ETL were severely tested to the breaking point by the three V's of big data (velocity, volume, and variety). In essence, ETL simply couldn't keep up.

Limited Memory for Storage

ETL tools have limited CPU or memory. Hence it creates obstruction in data processing.

Struggle with Governing Data

ETL significantly contributes to the accessibility of enterprise data sources. But unlike the conventional, business specific data warehouse controlled by data engineers, achieving adequate data governance is a significant difficulty when working with a diversity of external data sources.

ETL lacks Normalization

ETL tools have trouble normalizing data in your data warehouse or data lake. Data consumers seek information that can be used immediately.

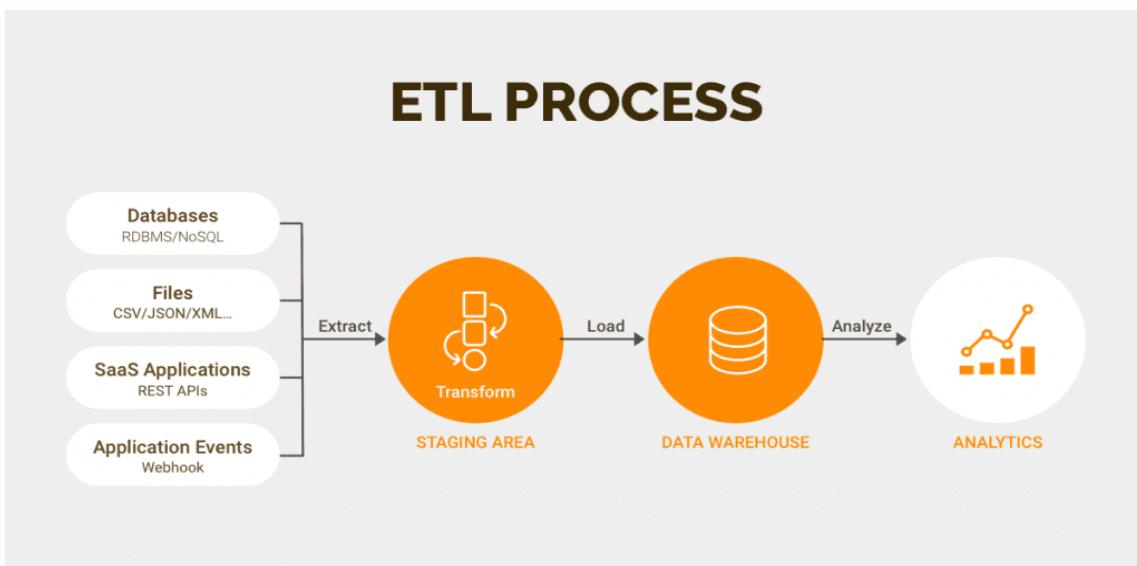
This information needs to offer a “one version of the truth” that data consumers can rely on in order to be acceptable. Data normalization should be used in addition to ETL rather than as a stand-alone method to achieve this level of precision.

Getting the transition process just right is not always simple. Poorly programmed mappings will inevitably result in a number of problems, including missing values and useless data. Garbage in, garbage out is a term used in data engineering to basically state that if you have terrible data, you'll receive bad insights.

In short, ETL is best suited for processing smaller enterprise-wide data sets that demand intricate transformations and have already been identified as pertinent to the analytical aims.

On the other hand, ELT is highly suited for processing both structured and unstructured big data since it can handle any size or type of data.

What is an ELT Pipeline?



Another method of integrating data is called ELT, or “Extract, Load, Transform,” and it functions similarly to ETL, or “Extract, Transform, Load.” This procedure transfers unprocessed data from a source system to a resource, like a data warehouse.

ELT is designed for big data and cloud repositories, with clear benefits in scalability, performance and cost, but with some risks in data governance. ELT tools, in contrast to ETL, are created with the data consumer in mind.

They are made to offer selected, simple-to-understand modifications that enable transparent, on-demand access to several data sources in any storage format. The idea behind ELT tools is that they are facilitators of data access.

Extract & Load

Unlike ETL, ELT allows for the extraction of unstructured data (along with structured and semi structured data) from a source system and load it onto a target system. There is no requirement for data staging because the extracted, unstructured data is made available to business intelligence systems.

Fundamental data transformations, such as data validation and duplicate data removal, are carried out by ELT employing data warehousing. These processes are real-time updated and used for massive amounts of unprocessed data.

Transform

The extracted data is loaded into the target system or data repository, and transformations are applied in the data repository. The destination system could be a data warehouse or data lake. This stage can involve the following:

- Filtering, cleansing, de-duplicating, validating, and authenticating the data.
- Utilizing the raw data to do computations, translations, data analysis, or summaries. This could involve anything from consistent row and column header changes to currency or measurement unit conversions, text string altering, adding or averaging values, and anything else required to meet the organization's unique BI or analytical needs.
- Deleting, encrypting, concealing, or otherwise safeguarding data that is subject to governmental or commercial laws.
- Transforming the data into tables or connected tables in accordance with the warehouse's deployed schema.

ELT is a relatively new technology in data engineering powered by cloud technologies. It helps to process large sets of data. Moreover, ELT has an advantage over ETL since the raw data is delivered directly to the destination system rather than a staging environment. This shortens the cycle between extraction and delivery.

Conclusion

The role and scope of data engineering services has expanded with the complexity, volume, and the need for compliance with privacy regulations. Today, data governance and data science initiatives greatly influence the role and structure of data engineering projects.

The earlier methods of ETL, however, were unable to keep up with the needs for quick processing, as data volume has increased tremendously. ELT (is actually ELT-L) is now more preferred over ETL, to address the complexity of the big data landscape.

The future of intelligent data pipelines would aim to enhance its ability to get high-quality, trusted data to the right data consumer at the right time in order to facilitate more timely and accurate decisions.

S3 or Storage Services (Modes & Process)

What is Amazon S3?

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

Features of Amazon S3

Storage classes

Amazon S3 offers a range of storage classes designed for different use cases. For example, you can store mission-critical production data in S3 Standard for frequent access, save costs by storing infrequently accessed data in S3 Standard-IA or S3 One Zone-IA, and archive data at the lowest costs in S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, and S3 Glacier Deep Archive.

You can store data with changing or unknown access patterns in S3 Intelligent-Tiering, which optimizes storage costs by automatically moving your data between four access tiers when your access patterns change. These four access tiers include two low-latency access tiers optimized for frequent and infrequent access, and two opt-in archive access tiers designed for asynchronous access for rarely accessed data.

Storage management

Amazon S3 has storage management features that you can use to manage costs, meet regulatory requirements, reduce latency, and save multiple distinct copies of your data for compliance requirements.

- S3 Lifecycle – Configure a lifecycle configuration to manage your objects and store them cost effectively throughout their lifecycle. You can transition

objects to other S3 storage classes or expire objects that reach the end of their lifetimes.

- **S3 Object Lock** – Prevent Amazon S3 objects from being deleted or overwritten for a fixed amount of time or indefinitely. You can use Object Lock to help meet regulatory requirements that require *write-once-read-many (WORM)* storage or to simply add another layer of protection against object changes and deletions.
- **S3 Replication** – Replicate objects and their respective metadata and object tags to one or more destination buckets in the same or different AWS Regions for reduced latency, compliance, security, and other use cases.
- **S3 Batch Operations** – Manage billions of objects at scale with a single S3 API request or a few clicks in the Amazon S3 console. You can use Batch Operations to perform operations such as **Copy**, **Invoke AWS Lambda function**, and **Restore** on millions or billions of objects.

Access management and security

Amazon S3 provides features for auditing and managing access to your buckets and objects. By default, S3 buckets and the objects in them are private. You have access only to the S3 resources that you create. To grant granular resource permissions that support your specific use case or to audit the permissions of your Amazon S3 resources, you can use the following features.

- **S3 Block Public Access** – Block public access to S3 buckets and objects. By default, Block Public Access settings are turned on at the bucket level. We recommend that you keep all Block Public Access settings enabled unless you know that you need to turn off one or more of them for your specific use case. For more information, see [Configuring block public access settings for your S3 buckets](#).
- **AWS Identity and Access Management (IAM)** – IAM is a web service that helps you securely control access to AWS resources, including your Amazon S3 resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.
- **Bucket policies** – Use IAM-based policy language to configure resource-based permissions for your S3 buckets and the objects in them.
- **Amazon S3 access points** – Configure named network endpoints with dedicated access policies to manage data access at scale for shared datasets in Amazon S3.

- Access control lists (ACLs) – Grant read and write permissions for individual buckets and objects to authorized users. As a general rule, we recommend using S3 resource-based policies (bucket policies and access point policies) or IAM user policies for access control instead of ACLs. Policies are a simplified and more flexible access control option. With bucket policies and access point policies, you can define rules that apply broadly across all requests to your Amazon S3 resources. For more information about the specific cases when you'd use ACLs instead of resource-based policies or IAM user policies, see [Access policy guidelines](#).
- S3 Object Ownership – Take ownership of every object in your bucket, simplifying access management for data stored in Amazon S3. S3 Object Ownership is an Amazon S3 bucket-level setting that you can use to disable or enable ACLs. By default, ACLs are disabled. With ACLs disabled, the bucket owner owns all the objects in the bucket and manages access to data exclusively by using access-management policies.
- IAM Access Analyzer for S3 – Evaluate and monitor your S3 bucket access policies, ensuring that the policies provide only the intended access to your S3 resources.

Data processing

To transform data and trigger workflows to automate a variety of other processing activities at scale, you can use the following features.

- S3 Object Lambda – Add your own code to S3 GET, HEAD, and LIST requests to modify and process data as it is returned to an application. Filter rows, dynamically resize images, redact confidential data, and much more.
- Event notifications – Trigger workflows that use Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), and AWS Lambda when a change is made to your S3 resources.

Storage logging and monitoring

Amazon S3 provides logging and monitoring tools that you can use to monitor and control how your Amazon S3 resources are being used. For more information, see [Monitoring tools](#).

Automated monitoring tools

- Amazon CloudWatch metrics for Amazon S3 – Track the operational health of your S3 resources and configure billing alerts when estimated charges reach a user-defined threshold.
- AWS CloudTrail – Record actions taken by a user, a role, or an AWS service in Amazon S3. CloudTrail logs provide you with detailed API tracking for S3 bucket-level and object-level operations.

Manual monitoring tools

- Server access logging – Get detailed records for the requests that are made to a bucket. You can use server access logs for many use cases, such as conducting security and access audits, learning about your customer base, and understanding your Amazon S3 bill.
- AWS Trusted Advisor – Evaluate your account by using AWS best practice checks to identify ways to optimize your AWS infrastructure, improve security and performance, reduce costs, and monitor service quotas. You can then follow the recommendations to optimize your services and resources.

Analytics and insights

Amazon S3 offers features to help you gain visibility into your storage usage, which empowers you to better understand, analyze, and optimize your storage at scale.

- Amazon S3 Storage Lens – Understand, analyze, and optimize your storage. S3 Storage Lens provides 29+ usage and activity metrics and interactive dashboards to aggregate data for your entire organization, specific accounts, AWS Regions, buckets, or prefixes.
- Storage Class Analysis – Analyze storage access patterns to decide when it's time to move data to a more cost-effective storage class.
- S3 Inventory with Inventory reports – Audit and report on objects and their corresponding metadata and configure other Amazon S3 features to take action in Inventory reports. For example, you can report on the replication and encryption status of your objects. For a list of all the metadata available for each object in Inventory reports, see Amazon S3 Inventory list.

Strong consistency

Amazon S3 provides strong read-after-write consistency for PUT and DELETE requests of objects in your Amazon S3 bucket in all AWS Regions. This behavior

applies to both writes of new objects as well as PUT requests that overwrite existing objects and DELETE requests. In addition, read operations on Amazon S3 Select, Amazon S3 access control lists (ACLs), Amazon S3 Object Tags, and object metadata (for example, the HEAD object) are strongly consistent. For more information, see [Amazon S3 data consistency model](#).

How Amazon S3 works

Amazon S3 is an object storage service that stores data as objects within buckets. An *object* is a file and any metadata that describes the file. A *bucket* is a container for objects.

To store your data in Amazon S3, you first create a bucket and specify a bucket name and AWS Region. Then, you upload your data to that bucket as objects in Amazon S3. Each object has a *key* (or *key name*), which is the unique identifier for the object within the bucket.

S3 provides features that you can configure to support your specific use case. For example, you can use S3 Versioning to keep multiple versions of an object in the same bucket, which allows you to restore objects that are accidentally deleted or overwritten.

Buckets and the objects in them are private and can be accessed only if you explicitly grant access permissions. You can use bucket policies, AWS Identity and Access Management (IAM) policies, access control lists (ACLs), and S3 Access Points to manage access.

Buckets

A bucket is a container for objects stored in Amazon S3. You can store any number of objects in a bucket and can have up to 100 buckets in your account. To request an increase, visit the [Service Quotas console](#).

Every object is contained in a bucket. For example, if the object named photos/puppy.jpg is stored in the DOC-EXAMPLE-BUCKET bucket in the US West (Oregon) Region, then it is addressable by using the URL <https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg>. When you create a bucket, you enter a bucket name and choose the AWS Region where the bucket will reside. After you create a bucket, you cannot change the name of the bucket or its Region.

Buckets also:

- Organize the Amazon S3 namespace at the highest level.
- Identify the account responsible for storage and data transfer charges.
- Provide access control options, such as bucket policies, access control lists (ACLs), and S3 Access Points, that you can use to manage access to your Amazon S3 resources.
- Serve as the unit of aggregation for usage reporting.

Objects

Objects are the fundamental entities stored in Amazon S3. Objects consist of object data and metadata. The metadata is a set of name-value pairs that describe the object. These pairs include some default metadata, such as the date last modified, and standard HTTP metadata, such as Content-Type. You can also specify custom metadata at the time that the object is stored.

An object is uniquely identified within a bucket by a key (name) and a version ID (if S3 Versioning is enabled on the bucket). For more information about objects, see [Amazon S3 objects overview](#).

Keys

An *object key* (or *key name*) is the unique identifier for an object within a bucket. Every object in a bucket has exactly one key. The combination of a bucket, object key, and optionally, version ID (if S3 Versioning is enabled for the bucket) uniquely identify each object. So you can think of Amazon S3 as a basic data map between "bucket + key + version" and the object itself.

Every object in Amazon S3 can be uniquely addressed through the combination of the web service endpoint, bucket name, key, and optionally, a version. For example, in the URL <https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg>, DOC-EXAMPLE-BUCKET is the name of the bucket and photos/puppy.jpg is the key.

S3 Versioning

You can use S3 Versioning to keep multiple variants of an object in the same bucket. With S3 Versioning, you can preserve, retrieve, and restore every version of every object stored in your buckets. You can easily recover from both unintended user actions and application failures.

Version ID

When you enable S3 Versioning in a bucket, Amazon S3 generates a unique version ID for each object added to the bucket. Objects that already existed in the bucket at the time that you enable versioning have a version ID of null. If you modify these (or any other) objects with other operations, such as [CopyObject](#) and [PutObject](#), the new objects get a unique version ID.

Bucket policy

A bucket policy is a resource-based AWS Identity and Access Management (IAM) policy that you can use to grant access permissions to your bucket and the objects in it. Only the bucket owner can associate a policy with a bucket. The permissions attached to the bucket apply to all of the objects in the bucket that are owned by the bucket owner. Bucket policies are limited to 20 KB in size.

Bucket policies use JSON-based access policy language that is standard across AWS. You can use bucket policies to add or deny permissions for the objects in a bucket. Bucket policies allow or deny requests based on the elements in the policy, including the requester, S3 actions, resources, and aspects or conditions of the request (for example, the IP address used to make the request). For example, you can create a bucket policy that grants cross-account permissions to upload objects to an S3 bucket while ensuring that the bucket owner has full control of the uploaded objects. For more information, see [Bucket policy examples](#).

In your bucket policy, you can use wildcard characters on Amazon Resource Names (ARNs) and other values to grant permissions to a subset of objects. For example, you can control access to groups of objects that begin with a common [prefix](#) or end with a given extension, such as .html.

S3 Access Points

Amazon S3 Access Points are named network endpoints with dedicated access policies that describe how data can be accessed using that endpoint. Access Points are attached to buckets that you can use to perform S3 object operations, such as [GetObject](#) and [PutObject](#). Access Points simplify managing data access at scale for shared datasets in Amazon S3.

Each access point has its own access point policy. You can configure [Block Public Access](#) settings for each access point. To restrict Amazon S3 data access to a private network, you can also configure any access point to accept requests only from a virtual private cloud (VPC).

Access control lists (ACLs)

You can use ACLs to grant read and write permissions to authorized users for individual buckets and objects. Each bucket and object has an ACL attached to it as a subresource. The ACL defines which AWS accounts or groups are granted access and the type of access. ACLs are an access control mechanism that predates IAM. For more information about ACLs, see [Access control list \(ACL\) overview](#).

S3 Object Ownership is an Amazon S3 bucket-level setting that you can use to both control ownership of the objects that are uploaded to your bucket and to disable or enable ACLs. By default, Object Ownership is set to the bucket owner enforced setting, and all ACLs are disabled. When ACLs are disabled, the bucket owner owns all the objects in the bucket and manages access to them exclusively by using access-management policies.

A majority of modern use cases in Amazon S3 no longer require the use of ACLs. We recommend that you keep ACLs disabled, except in unusual circumstances where you need to control access for each object individually. With ACLs disabled, you can use policies to control access to all objects in your bucket, regardless of who uploaded the objects to your bucket. For more information, see [Controlling ownership of objects and disabling ACLs for your bucket](#).

Regions

You can choose the geographical AWS Region where Amazon S3 stores the buckets that you create. You might choose a Region to optimize latency, minimize costs, or address regulatory requirements. Objects stored in an AWS Region never leave the Region unless you explicitly transfer or replicate them to another Region. For example, objects stored in the Europe (Ireland) Region never leave it.

Amazon S3 data consistency model

Amazon S3 provides strong read-after-write consistency for PUT and DELETE requests of objects in your Amazon S3 bucket in all AWS Regions. This behavior applies to both writes to new objects as well as PUT requests that overwrite existing objects and DELETE requests. In addition, read operations on Amazon S3 Select, Amazon S3 access controls lists (ACLs), Amazon S3 Object Tags, and object metadata (for example, the HEAD object) are strongly consistent.

Updates to a single key are atomic. For example, if you make a PUT request to an existing key from one thread and perform a GET request on the same key from a second thread concurrently, you will get either the old data or the new data, but never partial or corrupt data.

Amazon S3 achieves high availability by replicating data across multiple servers within AWS data centers. If a PUT request is successful, your data is safely stored. Any read (GET or LIST request) that is initiated following the receipt of a successful PUT response will return the data written by the PUT request. Here are examples of this behavior:

- A process writes a new object to Amazon S3 and immediately lists keys within its bucket. The new object appears in the list.
- A process replaces an existing object and immediately tries to read it. Amazon S3 returns the new data.
- A process deletes an existing object and immediately tries to read it. Amazon S3 does not return any data because the object has been deleted.
- A process deletes an existing object and immediately lists keys within its bucket. The object does not appear in the listing.

Note

- Amazon S3 does not support object locking for concurrent writers. If two PUT requests are simultaneously made to the same key, the request with the latest timestamp wins. If this is an issue, you must build an object-locking mechanism into your application.
- Updates are key-based. There is no way to make atomic updates across keys. For example, you cannot make the update of one key dependent on the update of another key unless you design this functionality into your application.

Bucket configurations have an eventual consistency model. Specifically, this means that:

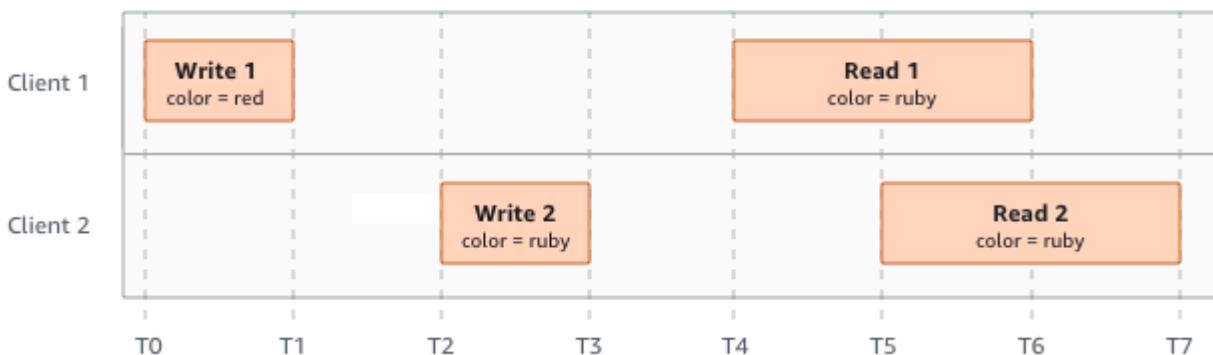
- If you delete a bucket and immediately list all buckets, the deleted bucket might still appear in the list.
- If you enable versioning on a bucket for the first time, it might take a short amount of time for the change to be fully propagated. We recommend that you wait for 15 minutes after enabling versioning before issuing write operations (PUT or DELETE requests) on objects in the bucket.

Concurrent applications

This section provides examples of behavior to be expected from Amazon S3 when multiple clients are writing to the same items.

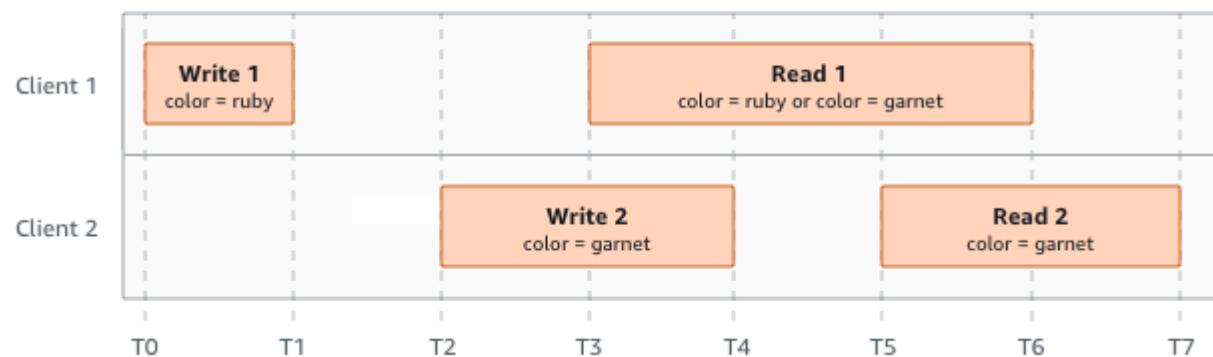
In this example, both W1 (write 1) and W2 (write 2) finish before the start of R1 (read 1) and R2 (read 2). Because S3 is strongly consistent, R1 and R2 both return color = ruby.

Domain = MyDomain, Item = StandardFez



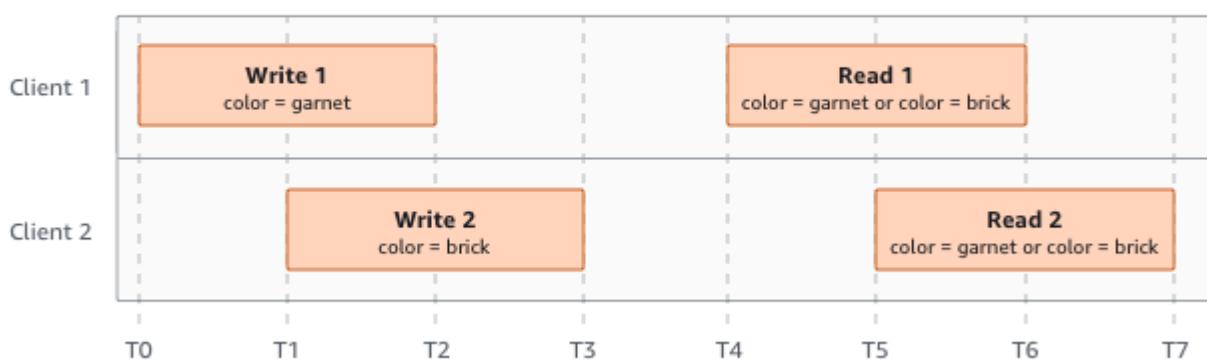
In the next example, W2 does not finish before the start of R1. Therefore, R1 might return color = ruby or color = garnet. However, because W1 and W2 finish before the start of R2, R2 returns color = garnet.

Domain = MyDomain, Item = StandardFez



In the last example, W2 begins before W1 has received an acknowledgment. Therefore, these writes are considered concurrent. Amazon S3 internally uses last-writer-wins semantics to determine which write takes precedence. However, the order in which Amazon S3 receives the requests and the order in which applications receive acknowledgments cannot be predicted because of various factors, such as network latency. For example, W2 might be initiated by an Amazon EC2 instance in the same Region, while W1 might be initiated by a host that is farther away. The best way to determine the final value is to perform a read after both writes have been acknowledged.

Domain = MyDomain, Item = StandardFez



Related services

After you load your data into Amazon S3, you can use it with other AWS services. The following are the services that you might use most frequently:

- **Amazon Elastic Compute Cloud (Amazon EC2)** – Provides secure and scalable computing capacity in the AWS Cloud. Using Amazon EC2 eliminates your need to invest in hardware upfront, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage.
- **Amazon EMR** – Helps businesses, researchers, data analysts, and developers easily and cost-effectively process vast amounts of data. Amazon EMR uses a hosted Hadoop framework running on the web-scale infrastructure of Amazon EC2 and Amazon S3.
- **AWS Snow Family** – Helps customers that need to run operations in austere, non-data center environments, and in locations where there's a lack of consistent network connectivity. You can use AWS Snow Family devices to

locally and cost-effectively access the storage and compute power of the AWS Cloud in places where an internet connection might not be an option.

- **AWS Transfer Family** – Provides fully managed support for file transfers directly into and out of Amazon S3 or Amazon Elastic File System (Amazon EFS) using Secure Shell (SSH) File Transfer Protocol (SFTP), File Transfer Protocol over SSL (FTPS), and File Transfer Protocol (FTP).

Accessing Amazon S3

You can work with Amazon S3 in any of the following ways:

AWS Management Console

The console is a web-based user interface for managing Amazon S3 and AWS resources. If you've signed up for an AWS account, you can access the Amazon S3 console by signing into the AWS Management Console and choosing **S3** from the AWS Management Console home page.

AWS Command Line Interface

You can use the AWS command line tools to issue commands or build scripts at your system's command line to perform AWS (including S3) tasks.

The [AWS Command Line Interface \(AWS CLI\)](#) provides commands for a broad set of AWS services. The AWS CLI is supported on Windows, macOS, and Linux. To get started, see the *AWS Command Line Interface User Guide*. For more information about the commands for Amazon S3, see [s3api](#) and [s3control](#) in the *AWS CLI Command Reference*.

AWS SDKs

AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, and so on). The AWS SDKs provide a convenient way to create programmatic access to S3 and AWS. Amazon S3 is a REST service. You can send requests to Amazon S3 using the AWS SDK libraries, which wrap the underlying Amazon S3 REST API and simplify your programming tasks. For example, the SDKs take care of tasks such as calculating signatures, cryptographically signing requests, managing errors, and retrying requests.

automatically. For information about the AWS SDKs, including how to download and install them, see [Tools for AWS](#).

Every interaction with Amazon S3 is either authenticated or anonymous. If you are using the AWS SDKs, the libraries compute the signature for authentication from the keys that you provide. For more information about how to make requests to Amazon S3, see [Making requests](#).

Amazon S3 REST API

The architecture of Amazon S3 is designed to be programming language-neutral, using AWS-supported interfaces to store and retrieve objects. You can access S3 and AWS programmatically by using the Amazon S3 REST API. The REST API is an HTTP interface to Amazon S3. With the REST API, you use standard HTTP requests to create, fetch, and delete buckets and objects.

To use the REST API, you can use any toolkit that supports HTTP. You can even use a browser to fetch objects, as long as they are anonymously readable.

The REST API uses standard HTTP headers and status codes, so that standard browsers and toolkits work as expected. In some areas, we have added functionality to HTTP (for example, we added headers to support access control). In these cases, we have done our best to add the new functionality in a way that matches the style of standard HTTP usage.

If you make direct REST API calls in your application, you must write the code to compute the signature and add it to the request. For more information about how to make requests to Amazon S3, see [Making requests](#).

Note

SOAP API support over HTTP is deprecated, but it is still available over HTTPS. Newer Amazon S3 features are not supported for SOAP. We recommend that you use either the REST API or the AWS SDKs.

Paying for Amazon S3

Pricing for Amazon S3 is designed so that you don't have to plan for the storage requirements of your application. Most storage providers require you to purchase a predetermined amount of storage and network transfer capacity. In this scenario, if you exceed that capacity, your service is shut off or

Data Analytics

Student Material

you are charged high overage fees. If you do not exceed that capacity, you pay as though you used it all.

DMS (Ingestion)

To use AWS Database Migration Service (AWS DMS) most effectively, see this section's recommendations on the most efficient way to migrate your data.

Migration planning for AWS Database Migration Service

When planning a database migration using AWS Database Migration Service, consider the following:

- To connect your source and target databases to an AWS DMS replication instance, you configure a network. Doing this can be as simple as connecting two AWS resources in the same virtual private cloud (VPC) as your replication instance. It can range to more complex configurations such as connecting an on-premises database to an Amazon RDS DB instance over a virtual private network (VPN). For more information, see Network configurations for database migration.
- **Source and target endpoints** – Make sure that you know what information and tables in the source database need to be migrated to the target database. AWS DMS supports basic schema migration, including the creation of tables and primary keys. However, AWS DMS doesn't automatically create secondary indexes, foreign keys, user accounts, and so on, in the target database. Depending on your source and target database engine, you might need to set up supplemental logging or modify other settings for a source or target database. For more information, see Sources for data migration and Targets for data migration.
- **Schema and code migration** – AWS DMS doesn't perform schema or code conversion. You can use tools such as Oracle SQL Developer, MySQL Workbench, and pgAdmin III to convert your schema. To convert an existing schema to a different database engine, you can use the AWS Schema Conversion Tool (AWS SCT). It can create a target schema and can generate and create an entire schema: tables, indexes, views, and so on. You can also use the tool to convert PL/SQL or TSQL to PgSQL and other formats. For more information on the AWS SCT, see the AWS SCT User Guide.
- **Unsupported data types** – Make sure that you can convert source data types into the equivalent data types for the target database. For more information on supported data types, see the source or target section for your data store.
- **Diagnostic support script results** – When you plan your migration, we recommend that you run diagnostic support scripts. With the results from

these scripts, you can find advance information about potential migration failures.

If a support script is available for your database, download it using the link in the corresponding script topic in the following section. After verifying and reviewing the script, you can run it according to the procedure described in the script topic in your local environment. When the script run is complete, you can review the results. We recommend running these scripts as a first step of any troubleshooting effort. The results can be useful while working with an AWS Support team.

Converting schema

AWS DMS doesn't perform schema or code conversion. If you want to convert an existing schema to a different database engine, you can use AWS SCT. AWS SCT converts your source objects, table, indexes, views, triggers, and other system objects into the target data definition language (DDL) format. You can also use AWS SCT to convert most of your application code, like PL/SQL or TSQL, to the equivalent target language.

You can get AWS SCT as a free download from AWS. For more information on AWS SCT, see the AWS SCT User Guide.

If your source and target endpoints are on the same database engine, you can use tools such as Oracle SQL Developer, MySQL Workbench, or PgAdmin4 to move your schema.

Reviewing the AWS DMS public documentation

We highly recommended that you go through the AWS DMS public documentation pages for your source and target endpoints before your first migration. This documentation can help you to identify the prerequisites for the migration and understand the current limitations before you begin. For more information, see Working with AWS DMS endpoints.

During migration, the public documentation can help you to troubleshoot any issues with AWS DMS. Troubleshooting pages in the documentation can help you to resolve common issues using both AWS DMS and selected endpoint databases.

Running a proof of concept

To help discover issues with your environment in early phases of your database migration, we recommend that you run a small test migration. Doing this can also help you to set a more realistic migration time line. In addition, you might need to run a full-scale test migration to measure whether AWS DMS can handle the throughput of your database over your network. During this time, we recommend to benchmark and optimize your initial full load and ongoing replication. Doing this can help you to understand your network latency and gauge overall performance.

At this point, you also have an opportunity to understand your data profile and how large your database is, including the following:

- How many tables are large, medium, and small in size.
- How AWS DMS handles data type and character-set conversions.
- How many tables having large object (LOB) columns.
- How long it takes to run a test migration.

Improving the performance of an AWS DMS migration

A number of factors affect the performance of your AWS DMS migration:

- Resource availability on the source.
- The available network throughput.
- The resource capacity of the replication server.
- The ability of the target to ingest changes.
- The type and distribution of source data.
- The number of objects to be migrated.

You can improve performance by using some or all of the best practices mentioned following. Whether you can use one of these practices depends on your specific use case. You can find some limitations following:

Provisioning a proper replication server

AWS DMS is a managed service that runs on an Amazon EC2 instance.

This service connects to the source database, reads the source data, formats the data for consumption by the target database, and loads the data into the target database.

Most of this processing happens in memory. However, large transactions might require some buffering on disk. Cached transactions and log files are also written to disk. In the following sections, you can find what to consider when you choose your replication server.

CPU

AWS DMS is designed for heterogeneous migrations, but it also supports homogeneous migrations. To perform a homogeneous migration, first convert each source data type to its equivalent AWS DMS data type. Then convert each AWS DMS type data to the target data type. You can find references for these conversions for each database engine within the *AWS DMS User Guide*.

For AWS DMS to perform these conversions optimally, the CPU must be available when the conversions happen. Overloading the CPU and not having enough CPU resources can result in slow migrations, which can also cause other side effects.

Replication instance class

Some of the smaller instance classes are sufficient for testing the service or for small migrations. If your migration involves a large number of tables, or if you intend to run multiple concurrent replication tasks, consider using one of the larger instances. A larger instance can be a good idea because the service consumes a fair amount of memory and CPU.

T2 type instances are designed to provide moderate baseline performance and the capability to burst to significantly higher performance, as required by your workload. They are intended for workloads that don't use the full CPU often or consistently, but that occasionally need to burst. T2 instances are well suited for general purpose workloads, such as web servers, developer environments, and small databases. If you're troubleshooting a slow migration and using a T2 instance type, check the CPU Utilization host metric. It can show you if you're bursting over the baseline for that instance type.

The C4 instance classes are designed to deliver the highest level of processor performance for computer-intensive workloads. They achieve significantly higher packet per second (PPS) performance, lower network jitter, and lower network latency. AWS DMS can be

CPU-intensive, especially when performing heterogeneous migrations and replications such as migrating from Oracle to PostgreSQL. C4 instances can be a good choice for these situations.

The R4 instance classes are memory optimized for memory-intensive workloads. Ongoing migrations or replications of high-throughput transaction systems using AWS DMS can, at times, consume large amounts of CPU and memory. R4 instances include more memory per vCPU.

AWS DMS support for R5 and C5 instance classes

The R5 instance classes are memory-optimized instances that are designed to deliver fast performance for workloads that process large data sets in memory. Ongoing migrations or replications of high-throughput transaction systems using AWS DMS can, at times, consume large amounts of CPU and memory. R5 instances deliver 5 percent additional memory per vCPU than R4 and the largest size provides 768 GiB of memory. In addition, R5 instances deliver a 10 percent price per GiB improvement and a ~20% increased CPU performance over R4.

The C5 instance classes optimized for compute-intensive workloads and deliver cost-effective high performance at a low price per compute ratio. They achieve significantly higher network performance. Elastic Network Adapter (ENA) provides C5 instances with up to 25 Gbps of network bandwidth and up to 14 Gbps of dedicated bandwidth to Amazon EBS. AWS DMS can be CPU-intensive, especially when performing heterogeneous migrations and replications such as migrating from Oracle to PostgreSQL. C5 instances can be a good choice for these situations.

Storage

Depending on the instance class, your replication server comes with either 50 GB or 100 GB of data storage. This storage is used for log files and any cached changes that are collected during the load. If your source system is busy or takes large transactions, you might need to increase your storage. If you're running multiple tasks on the replication server, you might also need a storage increase. However, the default amount is usually sufficient.

All storage volumes in AWS DMS are GP2 or General-Purpose solid-state drives (SSDs). GP2 volumes come with a base performance of three I/O operations per second (IOPS), with abilities to burst up to 3,000 IOPS on a credit basis. As a rule of thumb, check the ReadIOPS and WriteIOPS metrics for the replication instance. Make sure that the sum of these values doesn't cross the base performance for that volume.

Multi-AZ

Choosing a Multi-AZ instance can protect your migration from storage failures. Most migrations are transient and aren't intended to run for long periods of time. If you use AWS DMS for ongoing replication purposes, choosing a Multi-AZ instance can improve your availability should a storage issue occur.

When using a single AZ or Multi-AZ replication instance during a FULL LOAD and a failover or host replacement occurs, the full load task is expected to fail. You can restart the task from the point of failure for the remaining tables that didn't complete, or are in an error state.

Loading multiple tables in parallel

By default, AWS DMS loads eight tables at a time. You might see some performance improvement by increasing this slightly when using a very large replication server, such as a dms.c4.xlarge or larger instance. However, at some point, increasing this parallelism reduces performance. If your replication server is relatively small, such as a dms.t2.medium, we recommend that you reduce the number of tables loaded in parallel.

To change this number in the AWS Management Console, open the console, choose **Tasks**, choose to create or modify a task, and then choose **Advanced Settings**. Under **Tuning Settings**, change the **Maximum number of tables to load in parallel** option.

To change this number using the AWS CLI, change the MaxFullLoadSubTasks parameter under TaskSettings.

Using parallel full load

You can use a parallel load from Oracle, Microsoft SQL Server, MySQL, Sybase, and IBM Db2 LUW sources based on partitions and subpartitions. Doing this can improve overall full load duration. In addition, while running an AWS DMS migration task, you can

accelerate the migration of large or partitioned tables. To do this, split the table into segments and load the segments in parallel in the same migration task.

To use a parallel load, create a table mapping rule of type table-settings with the parallel-load option. Within the table-settings rule, specify the selection criteria for the table or tables that you want to load in parallel. To specify the selection criteria, set the type element for parallel-load to one of the following settings:

- partitions-auto
- subpartitions-auto
- partitions-list
- ranges
- none

Working with indexes, triggers, and referential integrity constraints

Indexes, triggers, and referential integrity constraints can affect your migration performance and cause your migration to fail. How these affect migration depends on whether your replication task is a full load task or an ongoing replication (change data capture, or CDC) task.

For a full load task, we recommend that you drop primary key indexes, secondary indexes, referential integrity constraints, and data manipulation language (DML) triggers. Or you can delay their creation until after the full load tasks are complete. You don't need indexes during a full load task, and indexes incur maintenance overhead if they are present. Because the full load task loads groups of tables at a time, referential integrity constraints are violated. Similarly, insert, update, and delete triggers can cause errors, for example if a row insert is triggered for a previously bulk loaded table. Other types of triggers also affect performance due to added processing.

If your data volumes are relatively small and the additional migration time doesn't concern you, you can build primary key and secondary indexes before a full load task. Always turn off referential integrity constraints and triggers.

For a full load plus CDC task, we recommend that you add secondary indexes before the CDC phase. Because AWS DMS uses logical

replication, make sure that secondary indexes that support DML operations are in place to prevent full table scans. You can pause the replication task before the CDC phase to build indexes and create referential integrity constraints before you restart the task.

You should enable triggers right before the cutover.

Turn off backups and transaction logging

When migrating to an Amazon RDS database, it's a good idea to turn off backups and Multi-AZ on the target until you're ready to cut over. Similarly, when migrating to systems other than Amazon RDS, turning off any logging on the target until after cutover is usually a good idea.

Use multiple tasks

Sometimes using multiple tasks for a single migration can improve performance. If you have sets of tables that don't participate in common transactions, you might be able to divide your migration into multiple tasks. Transactional consistency is maintained within a task, so it's important that tables in separate tasks don't participate in common transactions. Also, each task independently reads the transaction stream, so be careful not to put too much stress on the source database.

You can use multiple tasks to create separate streams of replication. By doing this, you can parallelize the reads on the source, the processes on the replication instance, and the writes to the target database.

Optimizing change processing

By default, AWS DMS processes changes in a transactional mode, which preserves transactional integrity. If you can afford temporary lapses in transactional integrity, you can use the batch optimized apply option instead. This option efficiently groups transactions and applies them in batches for efficiency purposes. Using the batch optimized apply option almost always violates referential integrity constraints. So we recommend that you turn these constraints off during the migration process and turn them on again as part of the cutover process.

Using your own on-premises name server

Usually, an AWS DMS replication instance uses the Domain Name System (DNS) resolver in an Amazon EC2 instance to resolve domain endpoints. However, you can use your own on-premises name server to resolve certain endpoints if you use the Amazon Route 53 Resolver. With this tool, you can query between on-premises and AWS using inbound and outbound endpoints, forwarding rules, and a private connection. The benefits of using an on-premises name server include improved security and ease of use behind a firewall.

If you have inbound endpoints, you can use DNS queries that originate on-premises to resolve AWS-hosted domains. To configure the endpoints, assign IP addresses in each subnet that you want to provide a resolver. To establish connectivity between your on-premises DNS infrastructure and AWS, use AWS Direct Connect or a virtual private network (VPN).

Outbound endpoints connect to your on-premises name server. The name server only grants access to IP addresses included in an allow list and set in an outbound endpoint. The IP address of your name server is the target IP address. When you choose a security group for an outbound endpoint, choose the same security group used by the replication instance.

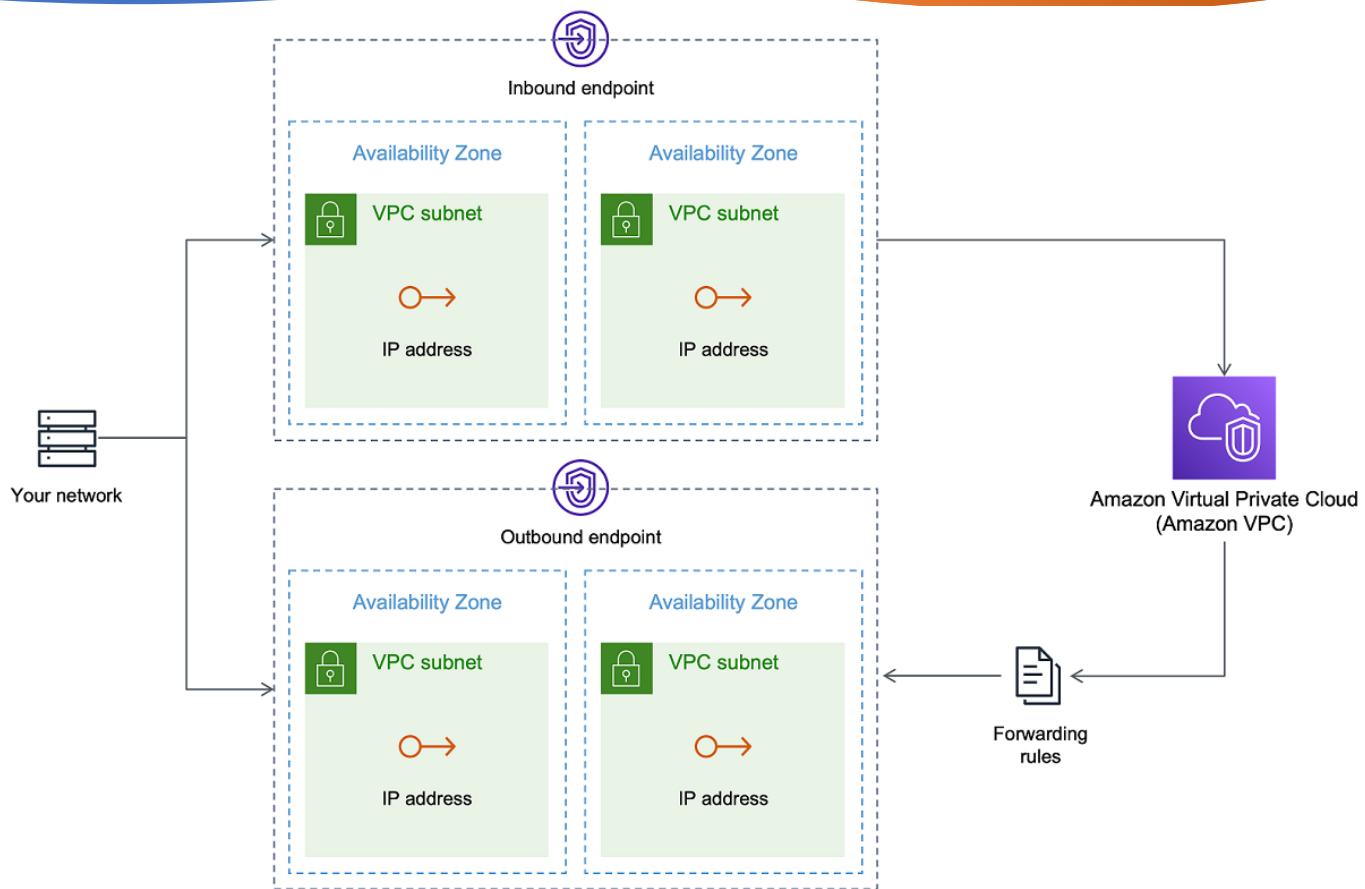
To forward select domains to the name server, use forwarding rules. An outbound endpoint can handle multiple forwarding rules. The scope of the forwarding rule is your virtual private cloud (VPC). By using a forwarding rule associated with a VPC, you can provision a logically isolated section of the AWS Cloud. From this logically isolated section, you can launch AWS resources in a virtual network.

You can configure domains hosted within your on-premises DNS infrastructure as conditional forwarding rules that set up outbound DNS queries. When a query is made to one of those domains, rules trigger an attempt to forward DNS requests to servers that were configured with the rules. Again, a private connection over AWS Direct Connect or VPN is required.

The following diagram shows the Route 53 Resolver architecture.

Data Analytics

Student Material



Using Amazon Route 53 Resolver with AWS DMS

You can create an on-premises name server for AWS DMS to resolve endpoints using [Amazon Route 53 Resolver](#).

To create an on-premises name server for AWS DMS based on Route 53

1. Sign in to the AWS Management Console and open the Route 53 console at <https://console.aws.amazon.com/route53/>.
2. On the Route 53 console, choose the AWS Region where you want to configure your Route 53 Resolver. The Route 53 Resolver is specific to a Region.
3. Choose the query direction—inbound, outbound, or both.
4. Provide your inbound query configuration:
 - a. Enter an endpoint name and choose a VPC.
 - b. Assign one or more subnets from within the VPC (for example, choose two for availability).
 - c. Assign specific IP addresses to use as endpoints, or have Route 53 Resolver assign them automatically.
5. Create a rule for your on-premises domain so that workloads inside the VPC can route DNS queries to your DNS infrastructure.

6. Enter one or more IP addresses for your on-premises DNS servers.
7. Submit your rule.

When everything is created, your VPC is associated with your inbound and outbound rules and can start routing traffic.

Migrating large binary objects (LOBs)

In general, AWS DMS migrates LOB data in two phases:

1. AWS DMS creates a new row in the target table and populates the row with all data except the associated LOB value.
2. AWS DMS updates the row in the target table with the LOB data.

This migration process for LOBs requires that, during the migration, all LOB columns on the target table must be nullable. This is so even if the LOB columns aren't nullable on the source table. If AWS DMS creates the target tables, it sets LOB columns to nullable by default. In some cases, you might create the target tables using some other mechanism, such as import or export. In such cases, make sure that the LOB columns are nullable before you start the migration task.

This requirement has one exception. Suppose that you perform a homogeneous migration from an Oracle source to an Oracle target, and you choose **Limited Lob mode**. In this case, the entire row is populated at once, including any LOB values. For such a case, AWS DMS can create the target table LOB columns with not-nullable constraints, if needed.

Using limited LOB mode

AWS DMS uses two methods that balance performance and convenience when your migration contains LOB values:

1. **Limited LOB mode** migrates all LOB values up to a user-specified size limit (default is 32 KB). LOB values larger than the size limit must be manually migrated. **Limited LOB mode**, the default for all migration tasks, typically provides the best performance. However, ensure that the **Max LOB size** parameter setting is correct. Set this parameter to the largest LOB size for all your tables.

2. **Full LOB mode** migrates all LOB data in your tables, regardless of size. **Full LOB mode** provides the convenience of moving all LOB data in your tables, but the process can have a significant impact on performance.

For some database engines, such as PostgreSQL, AWS DMS treats JSON data types like LOBs. Make sure that if you chose **Limited LOB mode**, the **Max LOB size** option is set to a value that doesn't cause the JSON data to be truncated.

AWS DMS provides full support for using large object data types (BLOBs, CLOBs, and NCLOBs). The following source endpoints have full LOB support:

- Oracle
- Microsoft SQL Server
- ODBC

The following target endpoints have full LOB support:

- Oracle
- Microsoft SQL Server

The following target endpoint has limited LOB support. You can't use an unlimited LOB size for this target endpoint.

- Amazon Redshift
- Amazon S3

For endpoints that have full LOB support, you can also set a size limit for LOB data types.

Improved LOB performance

While migrating LOB data, you can specify the following different LOB optimization settings.

Inline LOB settings

When you create an AWS DMS task, the LOB mode determines how LOBs are handled.

With full LOB mode and limited LOB mode, each has its own benefits and disadvantages. Inline LOB mode combines the advantages of both full LOB mode and limited LOB mode.

You can use inline LOB mode when you need to replicate both small and large LOBs, and most of the LOBs are small. When you choose this option, during full load the AWS DMS task transfers the small LOBs inline, which is more efficient. The AWS DMS task transfers the large LOBs by performing a lookup from the source table.

During change processing, both small and large LOBs are replicated by performing a lookup from the source table.

When you use inline LOB mode, the AWS DMS task checks all of the LOB sizes to determine which ones to transfer inline. LOBs larger than the specified size are replicated using full LOB mode. Therefore, if you know that most of the LOBs are larger than the specified setting, it's better not to use this option. Instead, allow an unlimited LOB size.

You configure this option using an attribute in task settings, `InlineLobMaxSize`, which is only available when `FullLobMode` is set to true. The default value for `InlineLobMaxSize` is 0 and the range is 1 – 102400 kilobytes (100 MB).

For example, you might use the following AWS DMS task settings. Here, setting `InlineLobMaxSize` to a value of 5 results in all LOBs smaller than or equal to 5,000 being transferred inline.

Improving performance when migrating large tables using row filtering

To improve the performance when migrating a large table, break the migration into more than one task. To break the migration into multiple tasks using row filtering, use a key or a partition key. For example, if you have an integer primary key ID from 1 to 8,000,000, you can create eight tasks using row filtering to migrate 1 million records each.

To apply row filtering in the console:

1. Open the AWS Management Console.
2. Choose **Tasks**, and create a new task.
3. Choose the **Table mappings** tab, and expand **Selection rules**.
4. Choose **Add new selection rule**. You can now add a column filter with either a less than or equal to, greater than or equal to, equal to, or a range condition between two values..

If you have a large partitioned table that is partitioned by date, you can migrate data based on date. For example, suppose that you have a table

partitioned by month, and only the current month's data is updated. In this case, you can create a full load task for each static monthly partition and create a full load plus CDC task for the currently updated partition.

If your table has a single-column primary key or unique index, you can have your AWS DMS task segment the table using a parallel load of the ranges type to load the data in parallel.

Ongoing replication

AWS DMS provides ongoing replication of data, keeping the source and target databases in sync. It replicates only a limited amount of data definition language (DDL) statements. AWS DMS doesn't propagate items such as indexes, users, privileges, stored procedures, and other database changes not directly related to table data.

If you plan to use ongoing replication, set the **Multi-AZ** option when you create your replication instance. By choosing **Multi-AZ**, you get high availability and failover support for the replication instance. However, this option can have an impact on performance and can slow down replication while applying changes to the target system.

Before you upgrade your source or target databases, we recommend that you stop any AWS DMS tasks that are running on these databases. Resume the tasks after your upgrades are complete.

During ongoing replication, it's critical to identify the network bandwidth between your source database system and your AWS DMS replication instance. Make sure that the network doesn't cause any bottlenecks during ongoing replication.

It's also important to identify the rate of change and archive log generation per hour on your source database system. Doing this can help you to understand the throughput that you might get during ongoing replication.

Reducing the load on your source database

AWS DMS uses some resources on your source database. During a full load task, AWS DMS performs a full table scan of the source table for each table processed in parallel. Also, each task that you create as part of a migration queries the source for changes as part of the CDC process. For AWS DMS to perform CDC for some sources, such as Oracle, you might need to increase the amount of data written to your database's change log.

If you find that you're overburdening your source database, reduce the number of tasks or tables for each task for your migration. Each task gets source changes independently, so consolidating tasks can decrease the change capture workload.

Reducing the bottlenecks on your target database

During the migration, try to remove any processes that compete for write resources on your target database:

- Turn off unnecessary triggers.
- Turn off secondary indexes during initial load and turn them back on later during ongoing replication.
- With Amazon RDS databases, it's a good idea to turn off backups and Multi-AZ until the cutover.
- While migrating to non-RDS systems, it's a good idea turn off any logging on the target until the cutover.

Using data validation during migration

To ensure that your data was migrated accurately from the source to the target, we highly recommend that you use data validation. If you turn on data validation for a task, AWS DMS begins comparing the source and target data immediately after a full load is performed for a table.

Data validation works with the following databases wherever AWS DMS supports them as source and target endpoints:

- Oracle
- PostgreSQL
- MySQL
- MariaDB

- Microsoft SQL Server
- Amazon Aurora MySQL-Compatible Edition
- Amazon Aurora PostgreSQL-Compatible Edition
- IBM Db2 LUW

Monitoring your AWS DMS tasks using metrics

You have several options for monitoring metrics for your tasks using the AWS DMS console:

Host metrics

You can find host metrics on the **CloudWatch metrics** tab for each particular replication instance. Here, you can monitor whether your replication instance is sized appropriately.

Replication task metrics

Metrics for replication tasks, including incoming and committed changes, and latency between the replication host and source/target databases can be found on the **CloudWatch metrics** tab for each particular task.

Table metrics

You can find individual table metrics on the **Table statistics** tab for each individual task. These metrics include these numbers:

- Rows loaded during the full load.
- Inserts, updates, and deletes since the task started.
- DDL operations since the task started.

Events and notifications

AWS DMS uses Amazon SNS to provide notifications when an AWS DMS event occurs, for example the creation or deletion of a replication instance. You can work with these notifications in any form supported by Amazon SNS for an AWS Region. These can include email messages, text messages, or calls to an HTTP endpoint.

Using the task log to troubleshoot migration issues

In some cases, AWS DMS can encounter issues for which warnings or error messages appear only in the task log. In particular, data truncation issues or row rejections due to foreign key violations are only written in the task log. Therefore, be sure to review the task log when migrating a database. To view the task log, configure Amazon CloudWatch as part of task creation.

Troubleshooting replication tasks with Time Travel

To troubleshoot AWS DMS migration issues, you can work with Time Travel. For more information about Time Travel, see [Time Travel task settings](#).

When you work with Time Travel, be aware of the following considerations:

- To avoid overhead on a DMS replication instance, turn on Time Travel only for tasks that need debugging.
- When you use Time Travel to troubleshoot replication tasks that might run for several days, monitor replication instance metrics for resource overheads. This approach applies especially in cases where high transaction loads run on source databases for extended periods of time. For more details, see [Monitoring AWS DMS tasks](#).
- When the Time Travel task setting EnableRawData is set to true, the task memory usage during DMS replication might be higher than when Time Travel isn't turned on. If you turn on Time Travel for extended periods of time, monitor your task.
- Currently, you can turn on Time Travel only at the task level. Changes to all tables are logged in Time Travel logs. If you are troubleshooting for specific tables in a database with high transaction volume, create a separate task .

Changing the user and schema for an Oracle target

When you use Oracle as a target, AWS DMS migrates the data to the schema owned by the target endpoint's user.

For example, suppose that you're migrating a schema named PERFDATA to an Oracle target endpoint, and that the target endpoint user name is MASTER.

AWS DMS connects to the Oracle target as MASTER and populates the MASTER schema with database objects from PERFDATA.

To override this behavior, provide a schema transformation. For example, to migrate the PERFDATA schema objects to a PERFDATA schema at the target endpoint, use the following transformation.

```
{  
    "rule-type": "transformation",  
    "rule-id": "2",  
    "rule-name": "2",  
    "object-locator": {  
        "schema-name": "PERFDATA"  
    },  
    "rule-target": "schema",  
    "rule-action": "rename",  
    "value": "PERFDATA"  
}
```

Changing table and index tablespaces for an Oracle target

When using Oracle as a target, AWS DMS migrates all tables and indexes to the default tablespace in the target. For example, suppose that your source is a database engine other than Oracle. All of the target tables and indexes are migrated to the same default tablespace.

To override this behavior, provide corresponding tablespace transformations. For example, suppose that you want to migrate tables and indexes to table and index tablespaces in the Oracle target that are named after the schema in the source. In this case, you can use transformations similar to the following. Here, the schema in the source is named INVENTORY and corresponding table and index tablespaces in the target are named INVENTORYTBL and INVENTORYIDX.

```
{  
    "rule-type": "transformation",  
    "rule-id": "3",  
    "rule-name": "3",  
    "rule-action": "rename",  
    "rule-target": "table-tablespace",  
    "object-locator": {  
        "schema-name": "INVENTORY",  
        "table-name": "%",  
        "table-tablespace-name": "%"  
    },  
    "value": "INVENTORYTBL"  
},  
{  
    "rule-type": "transformation",  
    "rule-id": "4",  
    "rule-name": "4",  
    "rule-action": "rename",  
    "rule-target": "index-tablespace",  
    "object-locator": {  
        "schema-name": "INVENTORY",  
        "table-name": "%",  
        "index-tablespace-name": "%"  
    }  
}
```

```
},  
    "value": "INVENTORYIDX"  
}
```

For more information about transformations, see Specifying table selection and transformations rules using JSON.

When Oracle is both source and target, you can preserve existing table or index tablespace assignments by setting the Oracle source extra connection attribute enableHomogenousTablespace=true. For more information, see Endpoint settings when using Oracle as a source for AWS DMS.

Upgrading a replication instance version

AWS periodically releases new versions of the AWS DMS replication engine software, with new features and performance improvements. Each version of the replication engine software has its own version number. It's critical to test the existing version of your AWS DMS replication instance running a production work load before you upgrade your replication instance to a later version. For more information on available version upgrades, see AWS DMS release notes.

Understanding your migration cost

AWS Database Migration Service helps you migrate databases to AWS easily and securely at a low cost. You only pay for your replication instances and any additional log storage. Each database migration instance includes storage sufficient for swap space, replication logs, and data cache for most replications and inbound data transfer is free.

You might need more resources during initial load or during peak load time. You can closely monitor replication instance resource utilization using cloud watch metrics. You can then scale up and scale down replication instance size based on usage.

Redshift or Cloud base data warehouse

What is Redshift?

- Redshift is a fast and powerful, fully managed, petabyte-scale data warehouse service in the cloud.
- Customers can use the Redshift for just \$0.25 per hour with no commitments or upfront costs and scale to a petabyte or more for \$1,000 per terabyte per year.

OLAP

OLAP is an **Online Analytics Processing System** used by the **Redshift**.

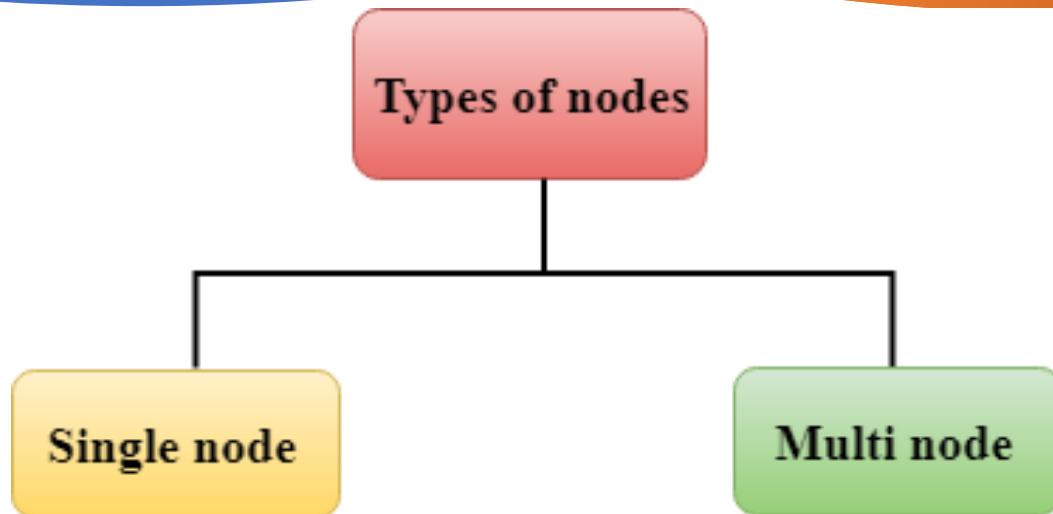
OLAP transaction Example:

Suppose we want to calculate the Net profit for EMEA and Pacific for the Digital Radio Product. This requires to pull a large number of records. Following are the records required to calculate a Net Profit:

- Sum of Radios sold in EMEA.
- Sum of Radios sold in Pacific.
- Unit cost of radio in each region.
- Sales price of each radio
- Sales price - unit cost

The complex queries are required to fetch the records given above. Data Warehousing databases use different type architecture both from a database perspective and infrastructure layer.

Redshift Configuration



Redshift consists of two types of nodes:

- **Single node**
- **Multi-node**

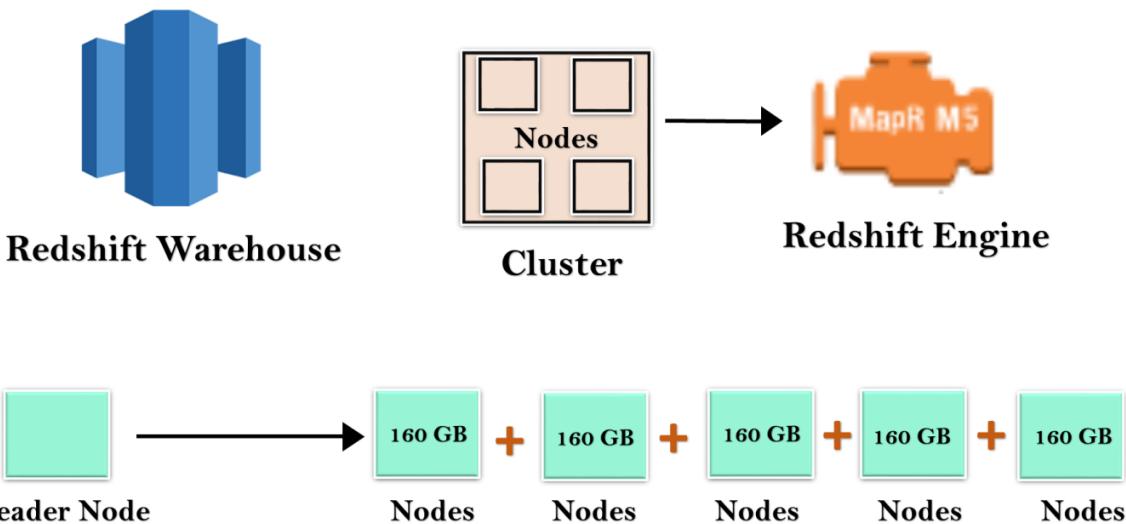
Single node: A single node stores up to 160 GB.

Multi-node: Multi-node is a node that consists of more than one node. It is of two types:

- **Leader Node**
It manages the client connections and receives queries. A leader node receives the queries from the client applications, parses the queries, and develops the execution plans. It coordinates with the parallel execution of these plans with the compute node and combines the intermediate results of all the nodes, and then return the final result to the client application.
- **Compute Node**
A compute node executes the execution plans, and then intermediate results are sent to the leader node for aggregation before sending back to the client application. It can have up to 128 compute nodes.

Let's understand the concept of leader node and compute nodes through an example.

Nodes



Massive Parallel Processing

Redshift warehouse is a collection of computing resources known as nodes, and these nodes are organized in a group known as a cluster. Each cluster runs in a Redshift Engine which contains one or more databases.

When you launch a Redshift instance, it starts with a single node of size 160 GB. When you want to grow, you can add additional nodes to take advantage of parallel processing. You have a leader node that manages the multiple nodes. Leader node handles the client connection as well as compute nodes. It stores the data in compute nodes and performs the query.

Why Redshift is 10 times faster

Redshift is 10 times faster because of the following reasons:

- **Columnar Data Storage**

Instead of storing data as a series of rows, Amazon Redshift organizes the data by column. Row-based systems are ideal for transaction processing while column-based systems are ideal for data warehousing

and analytics, where queries often involve aggregates performed over large data sets. Since only the columns involved in the queries are processed and columnar data is stored in a storage media sequentially, column-based systems require fewer I/Os, thus, improving query performance.

- **Advanced Compression**

Columnar data stores can be compressed much more than row-based data stores because similar data is stored sequentially on disk. Amazon Redshift employs multiple compression techniques and can often achieve significant compression relative to traditional relation data stores.

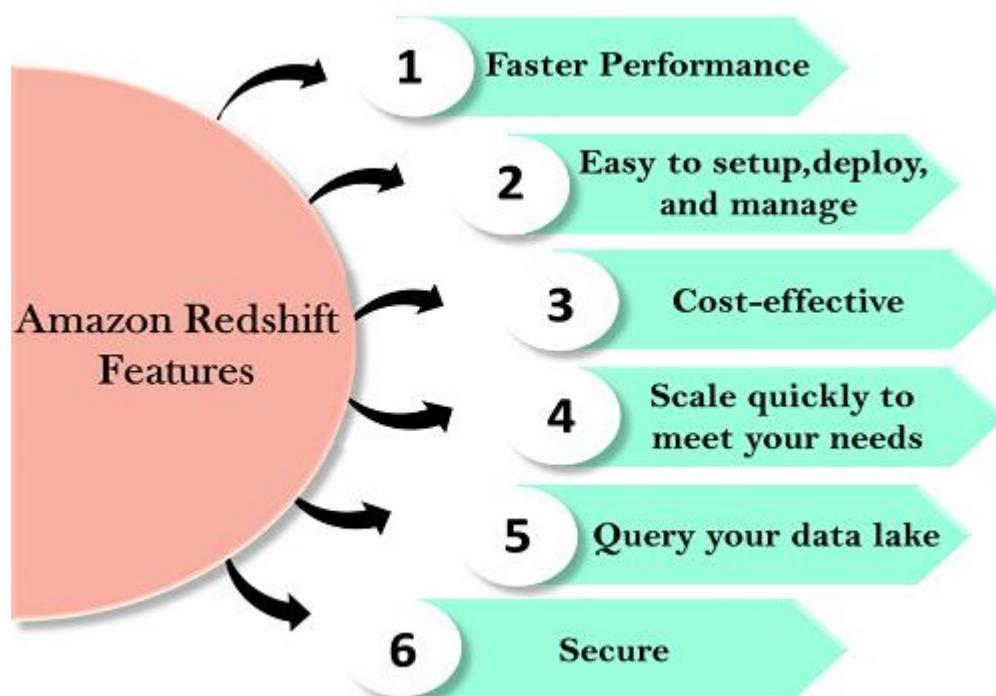
Amazon Redshift does not require indexes or materialized views so, it requires less space than traditional relational database systems. When loading a data into an empty table, Amazon Redshift samples your data automatically and selects the most appropriate compression technique.

- **Massively Parallel Processing**

Amazon Redshift automatically distributes the data and loads the query across various nodes. An Amazon Redshift makes it easy to add new nodes to your data warehouse, and this allows us to achieve faster query performance as your data warehouse grows.

Redshift features

Features of Redshift are given below:



- **Easy to setup, deploy and manage**

- **Automated Provisioning**

Redshift is simple to set up and operate. You can deploy a new data warehouse with just a few clicks in the AWS Console, and Redshift automatically provisions the infrastructure for you. In AWS, all the administrative tasks are automated, such as backups and replication, you need to focus on your data, not on the administration.

- **Automated backups**

Redshift automatically backups your data to S3. You can also replicate the snapshots in S3 in another region for any disaster recovery.

- **Cost-effective**

- **No upfront costs, pay as you go**

Amazon Redshift is the most cost-effective data warehouse service as you need to pay only for what you use.

Its costs start with \$0.25 per hour with no commitment and no upfront costs and can scale out to \$250 per terabyte per year.

Amazon Redshift is the only data warehouse service that offers On

Demand pricing with no up-front costs, and it also offers Reserved instance pricing that saves up to 75% by providing 1-3 year term.

- **Choose your node type.**

You can choose either of the two nodes to optimize the Redshift.

- **Dense compute node**

Dense compute node can create a high-performance data warehouses by using fast CPUs, a large amount of RAM, and solid-state disks.

- **Dense storage node**

If you want to reduce the cost, then you can use Dense storage node. It creates a cost-effective data warehouse by using a larger hard disk drive.

- **Scale quickly to meet your needs.**

- **Petabyte-scale data warehousing**

Amazon Redshift automatically scales up or down the nodes according to the need changes. With just a few clicks in the AWS Console or a single API call can easily change the number of nodes in a data warehouse.

- **Exabyte-scale data lake analytics**

It is a feature of Redshift that allows you to run the queries against exabytes of data in Amazon S3. Amazon S3 is a secure and cost-effective data to store unlimited data in an open format.

- **Limitless concurrency**

It is a feature of Redshift means that the multiple queries can access the same data in Amazon S3. It allows you to run the queries across the multiple nodes regardless of the complexity of a query or the amount of data.

- **Query your data lake**

Amazon Redshift is the only data warehouse which is used to query the Amazon S3 data lake without loading data. This provides flexibility by storing the frequently accessed data in Redshift and unstructured or infrequently accessed data in Amazon S3.

- **Secure**

With a couple of parameter settings, you can set the Redshift to use SSL

to secure your data. You can also enable encryption, all the data written to disk will be encrypted.

- **Faster performance**

Amazon Redshift provides columnar data storage, compression, and parallel processing to reduce the amount of I/O needed to perform queries. This improves query performance.

What is SQS?

- SQS stands for **Simple Queue Service**.
- SQS was the first service available in AWS.
- Amazon SQS is a web service that gives you access to a message queue that can be used to store messages while waiting for a computer to process them.
- Amazon SQS is a distributed queue system that enables web service applications to quickly and reliably queue messages that one component in the application generates to be consumed by another component where a queue is a temporary repository for messages that are awaiting processing.
- With the help of SQS, you can send, store and receive messages between software components at any volume without losing messages.
- Using Amazon sqs, you can separate the components of an application so that they can run independently, easing message management between components.
- Any component of a distributed application can store the messages in the queue.
- Messages can contain up to 256 KB of text in any format such as json, xml, etc.
- Any component of an application can later retrieve the messages programmatically using the Amazon SQS API.
- The queue acts as a buffer between the component producing and saving data, and the component receives the data for processing. This means that the queue resolves issues that arise if the producer is producing work faster than the consumer can process it, or if the producer or consumer is only intermittently connected to the network.

Data Analytics

Student Material

- If you got two EC2 instances which are pulling the SQS Queue. You can configure the autoscaling group if a number of messages go over a certain limit. Suppose the number of messages exceeds 10, then you can add additional EC2 instance to process the job faster. In this way, SQS provides elasticity.

Let's understand through an example.



Let's look at a website that generates a Meme. Suppose the user wants to upload a photo and wants to convert into Meme. User uploads a photo on a website and website might store a photo in s3. As soon as it finished uploads, it triggers a Lambda function. Lambda analyzes the data about this particular image to SQS, and this data can be "what the top of the meme should say", "what the bottom of the meme should say", the location of the S3 bucket, etc. The data sits inside the SQS as a message. An EC2 instance looks at the message and performs its job. An EC2 instance creates a Meme and stores it in S3 bucket. Once the EC2 instance completed its job, it moves back to the SQS. The best thing is that if you lose your EC2 instance, then also you would not lose the job as the job sits inside the S3 bucket.

Let's look at another example of SQS, i.e., Travel Website.

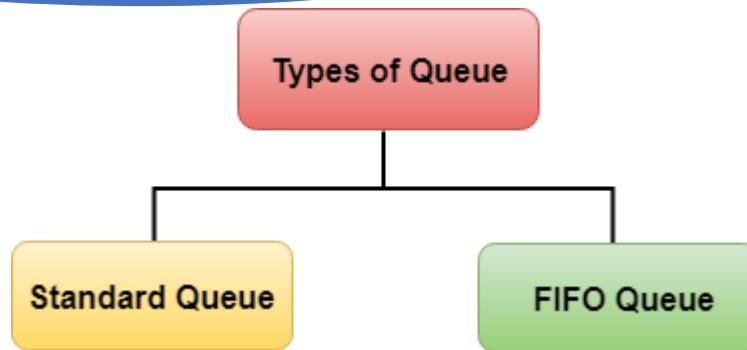


Suppose the user wants to look for a package holiday and wants to look at the best possible flight. A User types a query in a browser, it then hits the EC2 instance. An EC2 instance looks "What the user is looking for?", it then puts the message in a queue to the SQS. An EC2 instance pulls queue. An EC2 instance continuously pulling the queue and looking for the jobs to do. Once it gets the job, it then processes it. It interrogates the Airline service to get all the best possible flights. It sends the result to the web server, and the web server sends back the result to the user. A User then selects the best flight according to his or her budget.

If we didn't have SQS, then what happened?

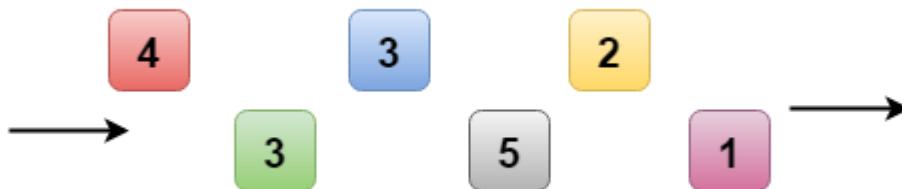
A web server passes the information to an application server and then application server queried an Airline service. If an Application server crashes, then a user loses its query. One of the great thing about SQS is that data is queued in the SQS even if the application server crashes, the message in the queue is marked as an invisible in a timeout interval window. When the timeout runs out, message reappears in the queue; then a new EC2 instance can use this message to perform its job. Therefore, we can say that SQS removes the application server dependency.

Queue Types



There are two types of Queue:

- **Standard Queues (default)**
 - **FIFO Queues (First-In-First-Out)**
 - **Standard Queue**
-



- SQS offers a standard queue as the default queue type.
 - It allows you to have an unlimited number of transactions per second.
 - It guarantees that a message is delivered at least once. However, sometime, more than one copy of a message might be delivered out of order.
 - It provides best-effort ordering which ensures that messages are generally delivered in the same order as they are sent but it does not provide a guarantee.
 - **FIFO Queue**
-



- The FIFO Queue complements the standard Queue.
- It guarantees ordering, i.e., the order in which they are sent is also received in the same order.

- The most important features of a queue are FIFO Queue and exactly-once processing, i.e., a message is delivered once and remains available until consumer processes and deletes it.
- FIFO Queue does not allow duplicates to be introduced into the Queue.
- It also supports message groups that allow multiple ordered message groups within a single Queue.
- FIFO Queues are limited to 300 transactions per second but have all the capabilities of standard queues.

SQS Visibility Timeout

- The visibility timeout is the amount of time that the message is invisible in the SQS Queue after a reader picks up that message.
- If the provided job is processed before the visibility time out expires, the message will then be deleted from the Queue. If the job is not processed within that time, the message will become visible again and another reader will process it. This could result in the same message being delivered twice.
- The Default Visibility Timeout is 30 seconds.
- Visibility Timeout can be increased if your task takes more than 30 seconds.
- The maximum Visibility Timeout is 12 hours.

Important points to remember:

- SQS is pull-based, not push-based.
- Messages are 256 KB in size.
- Messages are kept in a queue from 1 minute to 14 days.
- The default retention period is 4 days.
- It guarantees that your messages will be processed at least once.

From simple mechanisms for holding data like punch cards and paper tapes to real-time data processing systems like Hadoop, data storage systems have come a long way to become what they are now. For over 30 years, data

warehouses have been a rich business-insights source. Is it still so? With all the transformations in the sphere of cloud and information technologies, it may seem as if data warehousing has lost its relevance. Quite the opposite. Though there are countless options for storing, analyzing, and indexing data, data warehouses have remained relevant.

When reviewing BI tools, we described several data warehouse tools. In this article, we'll take a closer look at the top cloud warehouse software, including Snowflake, BigQuery, and Redshift. We'll review all the important aspects of their architecture, deployment, and performance to help you make an informed decision.

Before jumping into a comparison of available products, it's a good idea to first get acquainted with data warehousing basics.

What is a data warehouse?

A **data warehouse** is defined as a centralized repository where a company stores all valuable data assets integrated from different channels like databases, flat files, applications, CRM systems, etc. A data warehouse is often abbreviated as DW or DWH. You may also find it referred to as an enterprise data warehouse (EDW). It is usually created and used primarily for data reporting and analysis purposes. Thanks to the capability of data warehouses to get all data in one place, they serve as a valuable business intelligence (BI) tool, helping companies gain business insights and map out future strategies.

According to Bill Inmon, the data warehousing pioneer, there are several defining features of a DW:

Subject-oriented signifies that the data information in the warehouse revolves around some subject as compared to a data lake. It means that a warehouse doesn't contain all company data ever but only subject matters of interest. As an illustration, a particular warehouse can be built to track information about sales only.

Integrated means that the data warehouse has common standards for the quality of data stored. For instance, any organization may have a few business systems that track the same information. A data warehouse acts as a single source of truth, providing the most recent or appropriate information.

Data Analytics

Student Material

Time-variant relates to data warehouse consistency during a particular period when data is carried into a repository and stays unchanged. For example, companies can work with historical data to know what sales were like 5 or 10 years ago in contrast to current sales.

Non-volatile implies that once the data flies into a warehouse, it stays there and isn't removed with new data entries. As such, it is possible to retrieve old archived data if needed.

Summarized touches upon the fact the data is used for data analytics. Often, it is aggregated or segmented in data marts, facilitating analysis and reporting as users can get information by unit, section, department, etc.

Data warehouse architecture

The architecture of a data warehouse is a system defining how data is presented and processed within a repository. Warehouses can be divided into those following a traditional approach to storing and processing data versus modern cloud-based ones. Cloud systems are designed to fill in the gaps of legacy databases and address modern data management challenges. Let's go through the architectural components of both.

Traditional data warehouse architecture

Traditional or on-premise data warehouses have three standard approaches to constructing their architecture layers: single-tier, two-tier, and three-tier architectures. The most common one is the three-tier model, composed of the bottom, middle, and top tiers.

The bottom tier is represented by systems of report, usually relational database systems. A variety of back-end tools make it possible to extract, clean, transform, and load data into this layer. There are two different approaches to loading data into a data warehouse: ETL and ELT. Both processes include the Extract, Load, Transform functions but with a different sequence.

The middle tier serves as a mediator between the database and the end-user. It is a home for an OLAP (online analytical processing) server that converts data into a form more suitable for analysis and querying.

Brought to you by:



The top tier is referred to as the front-end or client layer. It contains an API (Application Programming Interface) and tools designed for data analysis, reporting, and data mining (the process of detecting patterns in large datasets to predict outcomes).

Cloud data warehouse architecture

Being relatively new, cloud warehouses more commonly consist of three layers such as compute, storage, and client (service). The compute layers comprise multiple compute clusters with nodes processing queries in parallel. Compute clusters are the sets of virtual machines grouped to perform computation tasks. These clusters are sometimes called virtual warehouses. In the storage layers, data is organized in partitions to be further optimized and compressed. The client layers are responsible for management activities. Although cloud-based data warehouse vendors often use slightly different approaches to constructing their architectures.

Cloud data warehouses can be categorized in multiple ways.

By the type of deployment:

- Cloud-based data warehouses that use the computation power and space presented by cloud providers.
- Hybrid cloud data warehouses adopt cloud capabilities while still allowing the use of on-premises solutions.

By generation:

- The 1st generation data warehouses are built using the SMP architecture or Symmetric Multiprocessing where multiple processors are attached to a single node.
- The 2nd generation Data Warehouses separate the compute and storage layers, relying on the MPP (Multiple Parallel Processing) architecture with multiple nodes processing different queries at the same time.
- The 3rd generation data warehouses add more computing choices to MPP and offer different pricing models.

By the level of back-end management involved:

- Serverless data warehouses get their functional building blocks with the help of serverless services, meaning they are fully-managed by third-party vendors. The term “serverless” doesn’t imply that there are no servers at all, but rather that a person or a company using the system doesn’t buy or rent any physical or virtual servers.
- DWs with servers, meaning a person or a company can rent, set up, and manage virtual servers for warehousing.

While traditional data warehouses are still alive and kicking, especially for storing sensitive data or working with close integrations of related structured data types, they lag behind modern cloud solutions big time. The variety of data explodes and on-premises options fail to handle it. Apart from the lack of scalability and flexibility offered by modern databases, the traditional ones are costly to implement and maintain.

Modern cloud solutions, on the other hand, cover the needs of high performance, scalability, and advanced data management and analytics. At the moment, cloud-based data warehouse architectures provide the most effective employment of data warehousing resources.

How to choose cloud data warehouse software: main criteria

Data storage tends to move to the cloud and we couldn’t bypass reviewing some of the most advanced data warehouses in the big data arena. To provide the most relevant information and unbiased opinions, all the data warehouse solutions are going to be compared based on the same criteria.

Architecture. Cloud-based data warehouses differ, and so do their approaches to organizing architectural components. And yet, pretty much all of them rely on massively parallel processing or MPP. There are shared-nothing or shared disk MPP architectures.

- In a **shared-nothing architecture**, nodes (processors) work independently and don’t share disk space. Different data is processed in parallel on different nodes.
- In a **shared-disk**, the processing is also performed on different nodes, but, as the name suggests, they are connected by a single memory disk, meaning all cluster nodes have access to all data.

Shared-nothing solutions are great if you are a large-scale company, with millions of users to serve daily. They also require a more intricate partitioning strategy and, ideally, avoiding operations that span across multiple storages. In many cases, a shared-disk architecture may be enough. However, in today's world, there are warehouse designs that can use both approaches simultaneously, like Snowflake.

Performance and data processing speed. Better performance reduces expenses as quite a few providers charge for the time you use their compute powers. So, you may want to find out how much time it takes for each solution to run queries. Another important factor resides in whether a product supports concurrent querying that speeds up performance.

Scalability opportunities. Scaling allows companies to take control of their system performance and storage bandwidth, which optimizes the usage of resources and saves money. Depending on the type of scaling used by vendors (horizontal or vertical), you will either add more machines to your pool of resources or add more power such as CPU or RAM to your existing system to meet your needs.

Offered security measures. From keeping a data warehouse compliant with the required data protection regulations to providing advanced user access management, a vendor you choose must employ all the needed measures to protect sensitive data.

Third-party integration support. Data warehouses often integrate with other systems. It is a good idea to check if it is possible to integrate a particular DW with your existing software tools as it can speed up the process of data migration into the cloud.

Data loading. It's a good thing to learn what types of data injection and what access methods the solution provides. Will it be possible to ingest data in real-time or in batches? Since we are comparing top providers on the market, they all have powerful data loading capabilities, including streaming data.

Support for data backup and recovery. To minimize worries about your data, it is better to ask your vendor what disaster recovery and data backup measures they provide upfront.

Deployment scenarios. Based on business needs and budget, companies have to decide which deployment option suits them best: on-premise, cloud, or hybrid. What's more, the ease of data warehouse deployment and its further use also should be taken into account.

Implementation process. Each company has its own needs and available resources. Consequently, the service implementation process is one of the determining factors for a business. What configuration opportunities does each solution offer? Is it easy or difficult to implement? Is it time-consuming? What specialists are required to handle a data warehouse and what would their expertise level have to be? Remember that all of the warehouse products available require some technical expertise to run, including data engineering and, in some cases, DevOps.

Price. Find out if a vendor offers a free trial to test the waters before a purchase. Is it a flat-rate or on-demand model? Also, be attentive as some packages don't include all the functionality.

Snowflake: for corporations in search of the easy deployment and configuration

Initially built on top of AWS (Amazon Web Services), Snowflake is an all-inclusive **cloud** data warehouse for structured and semi-structured data provided as Software-as-a-Service (SaaS). As a customer, you don't need to select, install, or manage any virtual or physical hardware, except for configuring the size and number of compute clusters. The rest of the maintenance duties are carried by Snowflake, which makes this solution practically serverless. Unlike traditional warehousing offerings, Snowflake provides more flexible, faster, and easier-to-use data storage and analytic solutions.

Architecture. Snowflake's architecture is natively designed for the cloud and combined with an innovative SQL query engine. Combining the characteristics of traditional shared-disk and shared-nothing database architectures,

Snowflake comprises three core layers such as database storage, query processing, and cloud services. There is a centralized data repository for a single copy of data that can be accessed from all independent compute nodes.

Performance. Thanks to the concept of separate compute and storage, Snowflake allows for concurrent workloads, meaning users can run multiple queries at a time. The workloads won't impact each other, which results in faster performance (according to one of the benchmarks, Snowflake is capable of processing 6 to 60 million rows of data in from 2 seconds to 10 seconds).

Scalability. Snowflake enables seamless, non-disruptive scaling (both horizontal and vertical) powered by multi-cluster shared data architecture. It doesn't require the involvement of a database operator or admin to scale as the software handles all the scaling automatically per business demand. That's a huge advantage for smaller companies with limited resources.

Security. Snowflake is highly secure and fully compliant with most data protection standards, including SOC 1 Type 2, SOC 2 Type 2 for all Snowflake editions and PCI DSS, HIPAA, HITRUST for the Business Critical Edition or higher. The system also provides controlled access management and high-level data security (all data and files are automatically encrypted).

Integrations. The system provides native connectivity with multiple data integration, BI, and analytics tools, including IBM Cognos, Azure Data Factory, Oracle Analytics Cloud, Fivetran, and Google Cloud, to name a few.

Data loading. Snowflake supports the ELT and ETL data integration approaches, meaning data transformation can happen during or after loading. The ELT approach helps with capturing raw data and then finding the best use case for it. It's important if you plan on designing machine learning models. The files can be loaded from cloud storage like Microsoft Azure or Amazon S3. Snowflake is also a good choice for data streaming. But keep in mind that all providers we compare here support data streaming. So, it's not going to be a differentiating feature in this article.

Data backup and recovery. The platform uses fail-safe instead of backup. Fail-safe offers a 7-day period during which Snowflake recovers data that may have been damaged or lost due to system failures.

Implementation. Snowflake is considered one of the most intuitive and simplest to use data warehouse products and it evokes a serverless experience. Having inherited a lot of relational database features and combined them with cloud principles, the service promises a quick and easy start. The platform allows for creating and maintaining multiple data warehouses using one account. The sizes of the compute cluster per warehouse can be configured in detail. All of this sounds great, but configuring Snowflake still requires solid SQL knowledge and skills as well as a good understanding of data warehouse architecture. To help you out, Snowflake provides explicit documentation as well as opportunities to become a certified Snowflake expert.

Price. Snowflake provides tiered pricing that is tailored to customer requirements and needs. There are on-demand and pre-purchase pricing plans. As the usages of storage and compute are separated, the latter is billed separately on a per-second basis (minimum 60 seconds).

Suitable for: Companies looking for an easily deployed DWH with nearly unlimited, automatic scaling and top-notch performance will benefit from using Snowflake.

Amazon Redshift: enterprise data warehouse tool

Part of Amazon's cloud-computing platform, Redshift is a **cloud-based** data warehouse software for enterprises. The platform enables fast processing of massive data sets. Not only is it fit for quality data analytics, it also provides automatic concurrency querying as per workload demand. Regardless of some managed features that we describe in the implementation section, Redshift is a more self-managed solution meaning that engineers will have to spend time on resource and server management.

Architecture. Redshift is designed with the shared-nothing MPP architecture. It comprises data warehouse clusters with compute nodes split up into node slices. Individual compute nodes are assigned with the code by the leader node. The system communicates with client applications by using industry-standard JDBC and ODBC drivers. The technology can be integrated with most

existing SQL-based client applications, ETL, BI, data analytics, and data mining tools.

Performance. While the performance is fine overall on most data types, it is quite low when using semi-structured data (e.g. JSON files). For optimal performance, it is recommended that users opt for the concept of distribution keys. These are columns used to define a database segment storing a particular row of data.

Scalability. Redshift allows for horizontal and vertical scaling. It is pre-configured to support 500 concurrent connections and up to 50 concurrent queries to be run at the same time in a cluster. That means up to 500 users can execute up to 50 queries at any given time in one cluster. In case you need to process more concurrent read queries, Redshift provides the concurrency scaling feature that automatically adds the capacity of another cluster. Different clusters can be used for different use cases while accessing the same data.

Security. With Redshift, you share security responsibilities with AWS: Security of the cloud is taken care of by AWS while security in the cloud is your responsibility. As security is of the highest priority to AWS, access to Redshift resources is controlled thoroughly at all levels. Redshift provides compliance with security standards, including ISO, PCI, HIPAA BAA, and SOC 1,2,3.

Integrations. The software tool provides robust integration opportunities with the whole AWS ecosystem, including Amazon S3, Amazon RDS, Amazon DynamoDB, Amazon EMR, AWS Glue, and AWS Data Pipeline. Redshift partners with a huge number of other platforms.

Data loading. While supporting both ELT/ETL approaches and standard DML (data manipulation language) commands including INSERT, Redshift claims that the most effective way to load data into your tables is to use their COPY command. With this command, it is possible to simultaneously work with multiple data streams and read data from multiple data files. The platform also offers real-time data streaming capabilities.

Data backup and recovery. Amazon Redshift uses the advanced system of both automated and manual snapshots – point-in-time backups of a cluster. They are stored in Amazon S3 by means of an encrypted SSL connection.

Implementation. Redshift's query engine resembles the PostgreSQL interface. Therefore, if your company has a team of data analysts who previously dealt with PostgreSQL or similar SQL-based management systems, it will be easy for you to start creating and processing queries in Redshift for your BI. Also, Redshift automates some of the most time-consuming cluster configuration responsibilities such as data backups, patching, replications, and facilitating the administration of a warehouse. One way or another, there should be a certain level of data engineering and server management expertise to work with Redshift: A DevOps or data engineer must configure clusters and define memory, compute, and storage allocation. For these reasons, Redshift falls more to the self-managed side of the spectrum.

Price. Redshift offers various pricing plans. With on-demand pricing, charges are set per hour. While it starts at only \$0.25 an hour, the final cost is calculated based on the number of nodes in a cluster. With managed storage pricing, users pay for the volume of data per month.

Suitable for: Redshift is initially made for big data warehousing. If your company deals with large scale data and your queries need quick responses, you should definitely give serious consideration to Redshift. The platform is also a good fit for businesses that are looking for a data warehouse with a transparent pricing model and little to no administrative overhead costs.

Google BigQuery fits corporations with varied workloads

Developed by Google, BigQuery does exactly what the name suggests – provides opportunities for querying large data sets. This is a cost-effective **multi-cloud** data warehouse technology possessing machine learning capabilities.

Architecture. BigQuery possesses a serverless architecture where storage and compute are separated. The main component of BigQuery architecture is called Dremel. This is a massively parallel query engine with the functionality to read thousands of rows in seconds. Data is stored in replicated, distributed storage, and processed in compute clusters consisting of nodes. This structure provides vast flexibility and differs from traditional on-premise or node-based

cloud data warehouse technologies. With such an approach under the hood, various users can put their data into the data warehouse and start analyzing that data using Standard SQL.

Performance. BigQuery supports partitioning, resulting in improved query performance. The data can be easily queried with either SQL or through Open Database Connectivity (ODBC). According to the Fivetran benchmark, Google BigQuery shows good but not top-tier performance – the average runtime of 99 TPC-DS queries (each TPC-DS consists of 24 tables with the largest one containing 4 million rows of data) is 11.18 seconds. Redshift and Snowflake showed 8.24 and 8.21 seconds respectively.

Scalability. Similar to Snowflake, BigQuery sets apart compute and storage, enabling users to scale processing and memory resources based on their needs. The tool obtains high vertical and horizontal scalability and executes real-time queries on petabytes of data relatively fast.

Security. To secure sensitive data, BigQuery offers column-level security allowing for creating policies and checking access status, Cloud DPL (cloud data loss prevention), and encryption keys management. As a part of the Google Cloud environment, BigQuery provides a huge number of compliance offerings such as HIPAA, FedRAMP, PCI DSS, ISO/IEC, SOC 1,2,3, to name a few.

Integrations. Apart from operational databases, the system allows for integration with a wide array of data integration tools, BI, and AI solutions. It also integrates with Google Cloud Platform systems, which makes it a great choice as quite a few companies these days use Google Workspace, previously known as G Suite.

Data loading. Along with traditional ETL/ELT batch data loading by the means of a standard SQL dialect, BigQuery allows for data streaming – ingesting data row-by-row in real-time using the streaming API (insertAll).

Data backup and recovery. BigQuery services include data backup and disaster recovery. Users are allowed to query point-in-time snapshots from 7 days of data changes.

Implementation. As far as the usability scale, BigQuery ranks high owing to the fully-managed nature of its data warehouses, meaning the BigQuery engineering team handles maintenance and updates and takes a lot of weight off your shoulders. But that doesn't mean that you don't need a data science team at all. The platform requires the knowledge of SQL commands and ETL tools. With the right experts, the processes of setup and configuration aren't time-consuming and you can start working with BigQuery quite quickly.

Price. As for BigQuery pricing, the platform offers on-demand and flat-rate subscriptions. Although you will be charged for data storage (\$0.020 per GB per month) and querying (\$5 per TB), things like exporting, loading, and copying data are free.

Suitable for: BigQuery will be the best fit for corporations with varied workloads and those interested in effective data mining. If you don't work with very big data volumes and query response time of up to several minutes isn't critical when you use that data to run queries, then BigQuery can be a great candidate.

Teradata: perfect for businesses needing deployment flexibility

Teradata has gained remarkable recognition as a powerful enterprise data warehouse that has provided scalable storage facilities and robust data analytics for over 35 years. It is one of the most efficient hybrid cloud data warehousing tools for processing huge volumes of data. Teradata offers deployment flexibility meaning a DW can be deployed **on-premises**, in a private **cloud**, in a public cloud, or within a hybrid cloud setting. While Teradata is more of a self-managed solution, their private cloud, called Teradata IntelliCloud, offers fully-managed services.

Architecture. Just like some other data warehouse products in the list, the Teradata engine is built on the shared-nothing MPP architecture, meaning that users can run multiple concurrent queries at a time. The architecture has four components: the parsing engine (the communicator between the client system and the AMPs), BYNET (the networking layer), AMPs (access module processors), and Disks (storage space).

Performance. With Teradata, businesses can count on high performance. The optimized performance is also reached thanks to the employment of in-memory processing (the processing of data using RAM or flash memory). The system divides data into hot and cold where hot refers to more frequently used data.

Scalability. The system has the capability of scaling from 100 gigabytes to over 100+ petabytes of data without compromising established performance. With Teradata, the system does both vertical and horizontal scaling, meaning users can add to or remove nodes from the system as well as add or remove CPU and memory per their needs.

Security. Teradata obtains various information security features from user-level security controls to network traffic encryption. The platform is fully compliant with strict security regulations from HIPAA, GDPR, PCI to ISO 27001, FISC, and SOC 1 and 2.

Integrations. A Teradata data warehouse can be integrated with Informatica, Google Cloud, Microsoft Azure, and other leading cloud providers.

Data loading. Teradata has a self-service solution for ingesting and administering data streams in real-time called Listener. It also supports both ELT and ETL approaches.

Data backup and recovery. Teradata provides complete data protection with its robust Backup and Restore solution that ensures an automated backup process and rapid recovery in case of operational failures.

Implementation. With Teradata, it's pretty easy and quick to get a data warehouse up and running. Most users concur that the system is overall user-oriented and convenient. While it requires a background in the use of SQL syntax, working with database management systems, and reading architectural descriptions, finding data specialists with fitting expertise is a no-brainer both onshore and offshore. The self-managed nature of Teradata on-prem solutions should be taken into account because configuration tasks rest on the shoulders of your DevOps and data engineers. The system provides documentation for all the configuration steps.

Price. The platform offers blended (a combination of flat and on-demand pricing) and consumption (on-demand) pricing models to help businesses use their budget efficiently.

Suitable for: Teradata is the only option from the list that offers deployment flexibility. So, it will be beneficial for businesses that need both on-premises and cloud solutions, or either of the two. Those who are looking for advanced business analytics and decision making will also reap benefits by using Teradata. The system can be used to analyze huge amounts of data and provide real-time analytics.

Azure Synapse by Microsoft (formerly SQL Data Warehouse)

Created by the technology giant Microsoft, Azure Synapse data warehouse serves as a **cloud** relational database with the opportunity of loading and processing petabytes of data as well as real-time reporting. Owing to seamless integration with the Microsoft SQL Server, the solution is a perfect match for organizations that seek easy transitioning to cloud-based warehouse technology.

Architecture. The system is built on nodes and a shared-disk massively parallel processing architecture, providing users with the capabilities of running more than a hundred parallel queries simultaneously. The Control node gets T-SQL commands from applications connected to it, prepares queries for concurrent processing, and sends operations to Compute nodes.

Performance. The architecture allows for concurrent query processing. As such, users can extract and visualize business insights quite fast. If we take the GigaOm benchmark, we'll see that Azure shows the best performance as it took the least aggregate time (2,996 seconds) to execute 103 queries.

Scalability. Azure is known for being an extremely scalable and elastic cloud service, which can be scaled up and down (horizontally and vertically) fast per requirements.

Security. Azure provides a variety of comprehensive data protection services for workloads in the cloud and on-premises. They include information security, access management, network security, threat protection, and data protection. The Microsoft service presents more than 90 compliance certifications such as HITRUST, HIPAA, ISO, NIST CSF, and many more.

Integrations. Azure has a set of integration tools such as API Management, Logic Apps, Service Bus, and Event Grid to connect with a wide array of third-party services. The system natively integrates with operational databases, BI, and ML software.

Data loading. Azure allows for ingesting huge volumes of data from over 95 connectors by creating ETL/ELT processes in a code-free environment.

Data backup and recovery. Microsoft offers a built-in solution used to back up and restore data resources called Azure Backup. It scales based on your backup storage needs.

Implementation. Azure excels at ease-of-use. Users who know Microsoft products will experience little to no headaches when implementing a data warehouse. But keep in mind that those who aren't used to the Microsoft environment might find Azure a bit difficult to learn. For analytic workloads, a company will need specialists who have hands-on experience using SQL and Spark. While Azure Synapse allows for leveraging serverless capabilities for database resources on-demand, the solution is referred to as more self-managed. Consequently, you will need experienced data engineers to configure the warehouse. The integrated development environment called Synapse Analytics Studio is nearly code-free. Azure provides clear instructions, well-organized documentation, and useful tutorials. Not to mention an intuitive interface.

Price. As far as the pricing of Azure SQL data warehouse, it is divided into a compute charge and a storage charge. When paused, you will not be charged for compute, only for storage. There are no upfront costs and termination fees.

Suitable for: Azure Synapse is a perfect match for enterprise DWHs as it offers a great price/performance ratio. It is also a win-win solution for companies that already use Microsoft products and need seamless integrations.

Reasons to choose modern cloud data warehouse products

The business environment has become highly competitive. To stay up to the minute, organizations increasingly turn from traditional on-premises platforms to more advanced cloud-based data warehouses. Here's why:

- There's no need to buy expensive physical hardware and hire an in-house team of specialists to maintain it.
- Cloud warehouses are much easier to set up and run.
- The absence of capital expenditure and low operational expenses are attractive features.
- Great scalability opportunities come at more affordable prices.
- The ability of modern warehouse architectures to perform complex analytical queries at a much faster pace due to their use of massively parallel processing (MPP) is an excellent option.

All of the above data warehouse vendors are worthy candidates if you need to migrate your data into the cloud. The checklist we provided here and the comparison of available products should help you assess whether or not a certain end-to-end solution fills the bill. And yet, some companies may find it difficult to decide which solution suits their workflows best or to figure out how to implement cloud DWHs by themselves. If this is the case, it is better to opt for the professional help of third-party providers with hands-on experience in data warehousing implementation and consulting.

Glue or Data Pipelining (Cleansing)

On the Amazon Web Services (AWS) Cloud, AWS Glue is a fully managed serverless environment where you can extract, transform, and load (ETL) data at scale. With AWS Glue, you can categorize data, clean it, enrich it, and move it reliably across various data stores and streams in a cost-effective manner. AWS Glue is serverless, so you don't have to worry about provisioning or managing servers. With AWS Glue, you pay only for the resources you use, and you can scale up or down as needed. AWS Glue consists of the following components:

- AWS Glue ETL – AWS Glue ETL provides batch and streaming options to extract, transform, and load data from one source to another.
- AWS Glue Data Catalog – Data Catalog is a central repository for organizing the metadata of all your data assets. Data Catalog provides a unified interface where you can search, discover, and share data assets across data analytics services.
- AWS Glue DataBrew – DataBrew is a no-code data preparation tool that you can use to visually explore, clean, and transform data. You can choose from more than 250 prebuilt transformations to automate data preparation tasks without writing any code. This guide provides a high-level introduction to AWS Glue, including how it works and how you can get started using it. It covers the key concepts that you need to know before authoring AWS Glue jobs, such as automation, monitoring, and integrating with other AWS services.

The Next steps (p. 12) section will get you up to speed with writing code in AWS Glue. If you already have some experience using AWS Glue, the Best practices (p. 9) section will help you fill in any gaps in your knowledge. By the end of this guide, you will be equipped with the knowledge and resources you need to start using AWS Glue effectively. AWS Glue ETL AWS Glue ETL supports extracting data from various sources, transforming it to meet your business needs, and loading it into a destination of your choice. This service uses the Apache Spark engine to distribute big data workloads across worker nodes, enabling faster transformations with in-memory processing. AWS Glue supports a variety of data sources, including Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, and Amazon Relational Database Service (Amazon RDS). To learn more about supported data sources, see Connection types and options for ETL in AWS Glue. Authoring in AWS Glue ETL AWS Glue provides multiple ways to author ETL jobs, depending on your experience and use-case:

- Python shell jobs are designed for running basic ETL scripts written in Python. These jobs run on a single machine, and are better suited for small or medium-sized datasets.
- Apache Spark jobs can be written in either Python or Scala. These jobs use Spark to horizontally scale workloads across many

worker nodes, so that they can handle large datasets and complex transformations. • AWS Glue streaming ETL uses the Apache Spark Structured Streaming engine to transform streaming data in microbatch jobs using exactly-once. You can author AWS Glue streaming jobs in either Python or Scala. • AWS Glue Studio is a visual boxes-and-arrows style interface to make Spark-based ETL accessible to developers who are new to Apache Spark programming. Data processing units AWS Glue uses data processing units (DPUs) to measure the compute resources allocated to an ETL job and calculate cost. Each DPU is equivalent to 4 vCPUs and 16 GB memory. DPUs should be allocated to your AWS Glue job depending on its complexity and data volume. Allocating the appropriate amount of DPUs will allow you to balance performance needs with cost constraints. AWS Glue provides four worker types that are optimized for various workloads:

- G.1X (for memory-intensive workloads)
- G.2X (for workloads with ML transforms)
- G.025X (for low volume and sporadic data streams)
- Standard (for AWS Glue versions 1.0 or earlier; not recommended for later versions of AWS Glue)

Using Python shell For a Python shell job, you can use either 1 DPU to use 16 GB of memory or 0.0625 DPU to use 1 GB of memory. Python shell is intended for basic ETL jobs with small or medium-sized datasets.

Comparing AWS Glue worker types The following table shows the different AWS Glue worker types for batch, streaming, and AWS Glue Studio ETL workloads using the Apache Spark environment.

	G.1X	G.2X	G.025X	Standard	vCPU	4	8	2	4	Memory	16 GB	32 GB	4 GB	16 GB	Disk space	64 GB	128 GB	64 GB	50 GB	Executor per worker	1	1	1	2	DPU	1	2	0.25	1
--	------	------	--------	----------	------	---	---	---	---	--------	-------	-------	------	-------	------------	-------	--------	-------	-------	---------------------	---	---	---	---	-----	---	---	------	---

Note that the Standard worker type is not recommended for AWS Glue version 2.0 and later. With AWS Glue Studio, you can use only G.1X and G.2X worker types. The G.025X worker type is available only for AWS Glue version 3.0 streaming jobs.

AWS Glue Data Catalog The AWS Glue Data Catalog is a centralized metadata repository for all your data assets across various data sources. It provides a unified interface to store and query information about data formats, schemas, and sources. When an AWS Glue ETL job runs, it uses this catalog to understand information about the data and ensure that it is transformed correctly. The Data Catalog is composed of the following components:

- Databases and tables
- Crawlers and classifiers
- Connections

Schema Registry AWS Glue databases and tables The AWS Glue Data Catalog is organized into databases and tables to provide a logical structure for storing and managing metadata. This structure supports precise data access control at a table or database level by using AWS Identity and Access Management (IAM) policies. An AWS Glue database can contain many tables, and each table must be associated with a single database. These tables contain references to the actual data, which can be stored in any of the various data sources that AWS

Glue supports. AWS Glue tables also store essential metadata such as column names, data types, and partition keys. There are several different methods for creating a table in AWS Glue:

- AWS Glue crawler
- AWS Glue ETL job
- AWS Glue console
- CreateTable operation in the AWS Glue API
- AWS CloudFormation template
- AWS Cloud Development Kit (AWS CDK)

• A migrated Apache Hive metastore AWS Glue crawlers and classifiers An AWS Glue crawler automatically discovers and extracts metadata from a data store, and then it updates the AWS Glue Data Catalog accordingly. The crawler connects to the data store to infer the schema of the data. It then creates or updates tables within the Data Catalog with the schema information that it discovered. A crawler can crawl both file-based and table-based data stores. To learn more about supported data stores, see Which data stores can I crawl? The crawler uses classifiers to accurately recognize the format of data and determine how it should be processed. By default, the crawler uses a set of common built-in classifiers provided by AWS Glue, but you can also write custom classifiers to handle specific use-cases.

AWS Glue connections

You can use AWS Glue connections to define connection parameters that enable AWS Glue to connect to various data sources. Adding connections centralizes and simplifies the configuration required to connect to these sources. When defining a connection, you specify the connection type, the connection endpoint, and any required credentials. After a connection is defined, it can be reused by multiple AWS Glue jobs and crawlers. Using connections with AWS Glue reduces the need to repeatedly enter the same connection information, such as login credentials or virtual private cloud (VPC) IDs.

AWS Glue Schema Registry

The AWS Glue Schema Registry provides a centralized location for managing and enforcing data stream schemas. It enables disparate systems, such as data producers and consumers, to share a schema for serialization and deserialization. Sharing a schema helps these systems to communicate effectively and avoid errors during transformation. The Schema Registry ensures that downstream data consumers can handle changes made upstream, because they are aware of the expected schema. It supports schema evolution, so that a schema can change over time while maintaining compatibility with previous versions of the schema. The Schema Registry integrates with many AWS services, including Amazon Kinesis Data Streams, Amazon Kinesis Data Firehose, and Amazon Managed Streaming for Apache Kafka. For examples of use cases and integrations, see Integrating with AWS Glue Schema Registry.

Important features and concepts

Logging and monitoring AWS Glue has several logging and monitoring options. By default, AWS Glue sends logs to the awsglue log group in Amazon CloudWatch. These logs include information such as start and end time, configuration settings, and

any errors or warnings that might have occurred. Additionally, AWS Glue Spark ETL jobs provide the following options, which must be enabled for advanced monitoring:

- Job metrics report job-specific metrics to the AWS Glue namespace in CloudWatch every 30 seconds. These job-specific metrics, such as processed records, total input/output data size, and runtime, provide insights into a job's performance. They can help identify bottlenecks or opportunities to optimize configurations.
- Continuous logging streams real-time Apache Spark job logs to the /aws-glue/jobs/logs-v2 log group in CloudWatch. By using real-time logs, you can dynamically monitor AWS Glue jobs while they are running.
- Spark UI provides a Spark history server web interface for viewing information about the Spark job, such as the event timeline of each stage, a directed acyclic graph, and job environment variables. The persisted Spark UI event logs are stored in Amazon S3, and you can use them in real time or after the job is complete.
- Job run insights simplifies job debugging and optimization by listening for common Spark exceptions, performing root cause analysis, and providing recommended actions to fix issues. The insights are stored in CloudWatch.

AWS Glue provides two main ways for you to automate ETL jobs: triggers and workflows.

AWS Glue triggers When fired, AWS Glue triggers start specified jobs and crawlers. A trigger can be fired on demand, based on a predefined schedule, or based on specific events. You can use triggers to design a chain of dependent jobs and crawlers. For more information, see AWS Glue triggers.

AWS Glue workflows For more complex workloads, you can use AWS Glue workflows to create directed acyclic graphs and to build dependencies between separate AWS Glue entities (triggers, crawlers, and jobs). Workflows also provide a unified interface where you can share parameters, monitor progress, and troubleshoot issues across associated entities. Setting up many associated entities within AWS Glue workflows can grow increasingly complex. Developers can create AWS Glue blueprints for sharing complex data pipelines with data scientists and business analysts. These templates allow for the consistent and repeatable creation of AWS Glue workflows, abstracting away the technical details. To learn more about AWS Glue blueprints and workflows, see Performing complex ETL activities using blueprints and workflows in AWS Glue.

Orchestrating AWS Glue jobs with other AWS services For more automation options, AWS Glue integrates with other AWS services, such as AWS Lambda, AWS Step Functions, and Amazon Managed Workflow for Apache Airflow (Amazon MWAA). To compare the different orchestration methods for AWS Glue ETL jobs, see Building an operationally excellent data pipeline.

AWS Glue DataBrew AWS Glue DataBrew is a fully managed visual data preparation service for cleaning, normalizing, and transforming data. It differs from AWS

Glue ETL in that you don't have write code to work with it. DataBrew provides more than 250 built-in transformations, with a visual point-and-click interface for creating and managing data transformation jobs. DataBrew is available in a separate console view from AWS Glue. It is natively integrated with several AWS services and supports many different file formats. For more information, see Product and service integrations. DataBrew is based on the following six core concepts:

- Project** The entire data preparation workspace in DataBrew
- Dataset** A collection of structured or semi-structured data
- Recipe** A set of data transformation steps; each step can contain many actions
- Job** A set of instructions to run a recipe or a data profile job
- Data lineage** The tracking of data in a visual interface to identify its origin
- Data profile** A summary view of the shape of your data

To get started with DataBrew, use the AWS Glue DataBrew sample project tutorial. Optimize memory management Memory management is crucial when writing AWS Glue ETL jobs because they run on the Apache Spark engine, which is optimized for in-memory processing. The blog post Optimize memory management in AWS Glue provides details on the following memory management techniques:

- Amazon S3 list implementation of AWS Glue
- Grouping
- Excluding irrelevant Amazon S3 paths and storage classes
- Spark and AWS Glue read partitioning
- Bulk inserts
- Join optimizations
- PySpark user-defined functions (UDFs)
- Incremental processing

Use efficient data storage formats When authoring ETL jobs, we recommend outputting transformed data in a column-based data format. Columnar data formats, such as Apache Parquet and ORC, are commonly used in big data storage and analytics. They are designed to minimize data movement and maximize compression. These formats also enable splitting data to multiple readers for increased parallel reads during query processing. Compressing data also helps reduce the amount of data stored, and it improves read/write operation performance. AWS Glue supports multiple compression formats natively. For more information about efficient data storage and compression, see Building a performance efficient data pipeline.

Use the appropriate type of scaling Understanding when to scale horizontally (change the number of workers) or scale vertically (change the worker type) is important for AWS Glue because it can impact the cost and performance of the ETL jobs. Generally, ETL jobs with complex transformations are more memory-intensive and require vertical scaling (for example, moving from the G.1X to the G.2X worker type). For compute-intensive ETL jobs with large volumes of data, we recommend horizontal scaling because AWS Glue is designed to process that data in parallel across multiple nodes. Closely monitoring AWS Glue job metrics in Amazon CloudWatch helps you determine whether a performance bottleneck is caused by a lack of memory or compute. For more

information about AWS Glue worker types and scaling, see Best practices to scale Apache Spark jobs and partition data with AWS Glue.

Analysis

It is evident from the results that the AWS Glue has outstanding features that enable the efficient integration of data and facilitation of workflow. In view of the theory of constraints, some of the weaknesses that characterize the service include the inability to enhance the performance of other application or system, unlike other tools. Notably, the inclusion or modification of this feature would most certainly increase the usability and relevance of the AWS Glue services. Also, the results unveil that the AWS Glue does not allow users to create their own components and use them to advance the data integration process and improve the output significantly. There is no doubt that the lack of this feature may be an inconvenience for the users who would prefer customizing their workflow.

EMR : Processing (Process Engineering)

Amazon EMR is the industry-leading cloud big data platform for processing vast amounts of data using open source tools such as Apache Spark, Apache Hive, Apache HBase, Apache Flink, Apache Hudi, and Presto. Amazon EMR makes it easy to set up, operate, and scale your big data environments by automating time-consuming tasks like provisioning capacity and tuning clusters. With EMR you can run petabyte-scale analysis at less than half of the cost of traditional on-premises solutions and over 3x faster than standard Apache Spark. You can run workloads on Amazon EC2 instances, on Amazon EKS clusters, or on-premises using EMR on AWS Outposts.

Amazon EMR does all the work involved with provisioning, managing, and maintaining the infrastructure and software of a Hadoop cluster. Amazon EMR now supports Amazon EC2 M6g instances to deliver the best price performance for cloud workloads. Amazon EC2 M6g instances are powered by AWS Graviton2 processors that are custom designed by AWS, utilizing 64-bit Arm Neoverse cores. Amazon EMR provides up to 35% lower cost and up to 15% improved performance for Spark workloads on Graviton2-based instances versus previous generation instances.

Ideal usage patterns

Amazon EMR's flexible framework reduces large processing problems and data sets into smaller jobs and distributes them across many compute nodes in a Hadoop cluster. This capability lends itself to many usage patterns with big data analytics. Here are a few examples:

- Log processing and analytics
- Large ETL data movement
- Risk modeling and threat analytics
- Ad targeting and click stream analytics
- Genomics
- Predictive analytics
- Ad hoc data mining and analytics

Cost model

With Amazon EMR, you can launch a persistent cluster that stays up indefinitely, or a temporary cluster that ends after the analysis is complete. In either scenario, you pay only for the hours the cluster is up.

Amazon EMR supports a variety of Amazon EC2 instance types (standard, high CPU, high memory, high I/O, and so on) and all Amazon EC2 pricing options (On-Demand, Reserved, and Spot). When you launch an Amazon EMR cluster (also called a "job flow"), you choose how many and what type of Amazon EC2 instances to provision. The Amazon EMR price is in addition to the Amazon EC2 price.

Performance

Amazon EMR performance is driven by the type of EC2 instances on which you choose to run your cluster, and how many you chose to run your analytics. You should choose an instance type suitable for your processing requirements, with sufficient memory, storage, and processing power. With the introduction of Graviton2 instances, you can see improved performance of up to 15% relative to equivalent previous generation instances. For more information about EC2 instance specifications, see [Amazon EC2 Instance Types](#).

An important consideration when you create an EMR cluster is how you configure Amazon EC2 instances. EC2 instances in an EMR cluster are organized into node types.

The node types in Amazon EMR are as follows:

- **Primary node** — A node that manages the cluster by running software components to coordinate the distribution of data and tasks among other nodes for processing. The primary node tracks the status of tasks and monitors the health of the cluster. Every cluster has a primary node, and it's possible to create a single-node cluster with only the primary node.
- **Core node** — A node with software components that run tasks and store data in the Hadoop Distributed File System (HDFS) on your cluster. Multi-node clusters have at least one core node.
- **Task node** — A node with software components that only runs tasks and does not store data in HDFS. Task nodes are optional.

Durability and availability

By default, Amazon EMR is fault tolerant for core node failures and continues job execution if a dependent node goes down. Amazon EMR will also provision a new node when a core node fails. However, Amazon EMR will not replace nodes if all nodes in the cluster are lost.

You can monitor the health of nodes and replace failed nodes with Amazon CloudWatch. When you launch an Amazon EMR cluster, you can choose to have one or three primary nodes in your cluster. Launching a cluster with three primary nodes is only supported by Amazon EMR version 5.23.0 and later. EMR can take advantage of EC2 placement groups to ensure primary nodes are placed on distinct underlying hardware to further improve cluster availability.

Scalability and elasticity

With Amazon EMR, it's easy to resize a running cluster. You can add core nodes which hold the HDFS at any time to increase your processing power and increase the HDFS storage capacity (and throughput). Additionally, you can use Amazon S3 natively, or using EMRFS along with or instead of local HDFS, which enables you to decouple your memory and compute from your storage providing greater flexibility and cost efficiency.

You can also add and remove task nodes at any time which can process Hadoop jobs, but do not maintain HDFS. Some customers add hundreds of instances to their clusters when their batch processing occurs, and remove the extra instances when processing completes. For example, you may not know how much data your clusters will be handling in six months, or you may have spiky processing needs.

With Amazon EMR, you don't need to guess your future requirements or provision for peak demand because you can easily add or remove capacity at any time.

You can add all new clusters of various sizes and remove them at any time with a few clicks in the console or by a programmatic API call.

Additionally, you can configure instance fleets for a cluster to choose a wide variety of provisioning options for EC2 instances. With instance fleets you can

specify target capacities on On-Demand Instances, and Spot Instances within each fleet. Amazon EMR tries to provide the capacity you need with the best mix of capacity and price based on your selection of Availability Zones.

While a cluster is running, if Amazon EC2 reclaims a Spot Instance because of a price increase, or an instance fails, Amazon EMR tries to replace the instance with any of the instance types that you specify. This makes it easier to regain capacity if a node is lost for any reason.

Interfaces

Amazon EMR supports many tools on top of Hadoop that can be used for big data analytics and each has their own interfaces. Here is a brief summary of the most popular options:

Hive

Hive is an open source data warehouse and analytics package that runs on top of Hadoop. Hive is operated by Hive QL, a SQL-based language, which enables users to structure, summarize, and query data. Hive QL goes beyond standard SQL, adding first-class support for map/reduce functions and complex extensible user-defined data types like JSON and Thrift. This capability allows processing of complex and unstructured data sources such as text documents and log files.

Hive allows user extensions via user-defined functions written in Java. Amazon EMR has made numerous improvements to Hive, including direct integration with DynamoDB and Amazon S3. For example, with Amazon EMR you can load table partitions automatically from S3, you can write data to tables in S3 without using temporary files, and you can access resources in S3, such as scripts for custom map and/or reduce operations and additional libraries.

For more information, see Apache Hive in the *Amazon EMR Release Guide*.

Pig

Pig is an open-source analytics package that runs on top of Hadoop. Pig is operated by Pig Latin, an SQL-like language which enables users to structure, summarize, and query data. Pig Latin also adds first-class support for map and reduce functions and complex extensible user-defined data types. This

capability allows processing of complex and unstructured data sources such as text documents and log files.

Pig allows user extensions via user-defined functions written in Java. Amazon EMR has made numerous improvements to Pig, including the ability to use multiple file systems (normally, Pig can only access one remote file system), the ability to load customer JARs and scripts from S3 (such as “REGISTER s3://my-bucket/piggybank.jar”), and additional functionality for String and DateTime processing.

Spark

Spark is an open-source data analytics engine built on Hadoop with the fundamentals for in-memory MapReduce. Spark provides additional speed for certain analytics and is the foundation for other power tools such as Shark (SQL driven data warehousing), Spark Streaming (streaming applications), GraphX (graph systems) and MLlib (machine learning).

EMR features Amazon EMR runtime for Apache Spark, a performance-optimized runtime environment for Apache Spark that is active by default on Amazon EMR clusters. Amazon EMR runtime for Apache Spark can be over 3x faster than clusters without the EMR runtime, and has 100% API compatibility with standard Apache Spark. This improved performance means your workloads run faster and saves you compute costs, without making any changes to your applications.

HBase

HBase is an open-source, non-relational, distributed database modeled after Google's Bigtable. It was developed as part of Apache Software Foundation's Hadoop project and runs on top of Hadoop Distributed File System (HDFS) to provide BigTable-like capabilities for Hadoop. HBase provides you a fault-tolerant, efficient way of storing large quantities of sparse data using column-based compression and storage. In addition, HBase provides fast lookup of data, because data is stored in-memory instead of on disk.

HBase is optimized for sequential write operations, and it is highly efficient for batch inserts, updates, and deletes. HBase works seamlessly with Hadoop, sharing its file system and serving as a direct input and output to Hadoop jobs. HBase also integrates with Apache Hive, enabling SQL-like queries over HBase

tables, joins with Hive-based tables, and support for Java Database Connectivity (JDBC). With Amazon EMR, you can back up HBase to Amazon S3 (full or incremental, manual or automated) and you can restore from a previously created backup.

Presto

Presto is an open-source distributed SQL query engine optimized for low-latency, ad hoc analysis of data. It supports the ANSI SQL standard, including complex queries, aggregations, joins, and window functions. Presto can process data from multiple data sources, including HDFS and Amazon S3.

Hudi

Apache Hudi is an open-source data management framework used to simplify incremental data processing and data pipeline development by providing record-level insert, update, upsert, and delete capabilities. *Upsert* refers to the ability to insert records into an existing dataset if they do not already exist or to update them if they do. By efficiently managing how data is laid out in S3, Hudi allows data to be ingested and updated in near-real-time. Hudi carefully maintains metadata of the actions performed on the dataset to help ensure that the actions are atomic and consistent. Hudi is integrated with Apache Spark, Apache Hive, and Presto. In Amazon EMR release versions 6.1.0 and later, Hudi is also integrated with Trino (PrestoSQL).

Kinesis Connector

The Kinesis Connector enables EMR to directly read and query data from Kinesis Data Streams. You can perform batch processing of Kinesis streams using existing Hadoop tools such as Hive, Pig, MapReduce, Hadoop Streaming, and Cascading. Some use cases enabled by this integration are:

- **Streaming log analysis** — You can analyze streaming web logs to generate a list of top ten error types every few minutes by Region, browser, and access domains.
- **Complex data processing workflows** — You can join Kinesis stream with data stored in Amazon S3, Dynamo DB tables, and HDFS. You can write queries that join clickstream data from Kinesis with advertising campaign information stored in a DynamoDB table to identify the most effective categories of ads that are displayed on particular websites.

- **Ad-hoc queries** — You can periodically load data from Kinesis into HDFS and make it available as a local Impala table for fast, interactive, analytic queries.

Other third-party tools

Amazon EMR also supports a variety of other popular applications and tools in the Hadoop system, such as R (statistics), Mahout (machine learning), Ganglia (monitoring), Accumulo (secure NoSQL database), Hue (user interface to analyze Hadoop data), HCatalog (table and storage management), and more.

Additionally, you can install your own software on top of Amazon EMR to help solve your business needs. AWS provides the ability to quickly move large amounts of data from S3 to HDFS, from HDFS to S3, and between S3 buckets using Amazon EMR's S3DistCp, an extension of the open source tool DistCp that uses MapReduce to efficiently move large amounts of data.

You can optionally use the EMR File System (EMRFS), an implementation of HDFS which allows Amazon EMR clusters to store data on S3. You can enable S3 server-side and client-side encryption.

EMR Studio

EMR Studio is an integrated development environment (IDE) that makes it easy for data scientists and data engineers to develop, visualize, and debug data engineering and data science applications written in R, Python, Scala, and PySpark.

EMR Studio provides fully managed Jupyter Notebooks, and tools like Spark UI and YARN Timeline Service to simplify debugging. Data scientists and analysts can install custom kernels and libraries, collaborate with peers using code repositories such as GitHub and BitBucket, or run parameterized notebooks as part of scheduled workflows using orchestration services like Apache Airflow or Amazon Managed Workflows for Apache Airflow. Administrators can set up EMR Studio such that analysts can run their applications on existing EMR clusters, or create new clusters using pre-defined AWS CloudFormation templates for EMR.

Anti-patterns

Amazon EMR has the following anti-pattern:

- **Small datasets** – Amazon EMR is built for massive parallel processing; if your data set is small enough to run quickly on a single machine, in a single thread, the added overhead to map and reduce jobs may not be worth it for small datasets that can easily be processed in memory on a single system. Amazon EMR or a relational database running on Amazon EMR may be a better option for workloads with stringent requirements.

Scheduling and Monitoring

In recent years, Cloud Computing has emerged as a major technology for delivering on-demand IT (Information Technology) services to consumers across the globe [1]. It provides IT resources based on a pay-as-you model and offers a rich set of services and tools. The Cloud Computing stack consists of three layers – Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a service (SaaS). IaaS offers hardware resources (computation, network, and storage) as virtualized cloud services. PaaS provides software services (appliances) and programming tools, whereas SaaS provides ready to use application stacks. In order to ensure bespoke application performance (e.g., minimize response time, maximize throughput); each layer of the Cloud stack requires optimum resource Scheduling, Monitoring, and Configuration-management (SMC). Numerous studies have addressed SMC by proposing various frameworks, models, and algorithms. Most of these studies however have assumed traditional, non-programmable Cloud DataCentre (CDC) networks. In this article, we discuss how resource scheduling, monitoring, and configuration-management becomes a challenging task when considering Software Defined Networking (SDN), and Network Function Virtualisation (NFV) performance parameters as part of overall cloud application scheduling process. The recent technology revolution of so-called Software Defined Networking (SDN) [2] has given full control and programmability of CDC network. Network Function Virtualization (NFV), on other hand, is a recent networking concept often presented with SDN, delivering network services using software processes hosted on CDC-based virtual machines (VMs), instead of proprietary dedicated hardware.

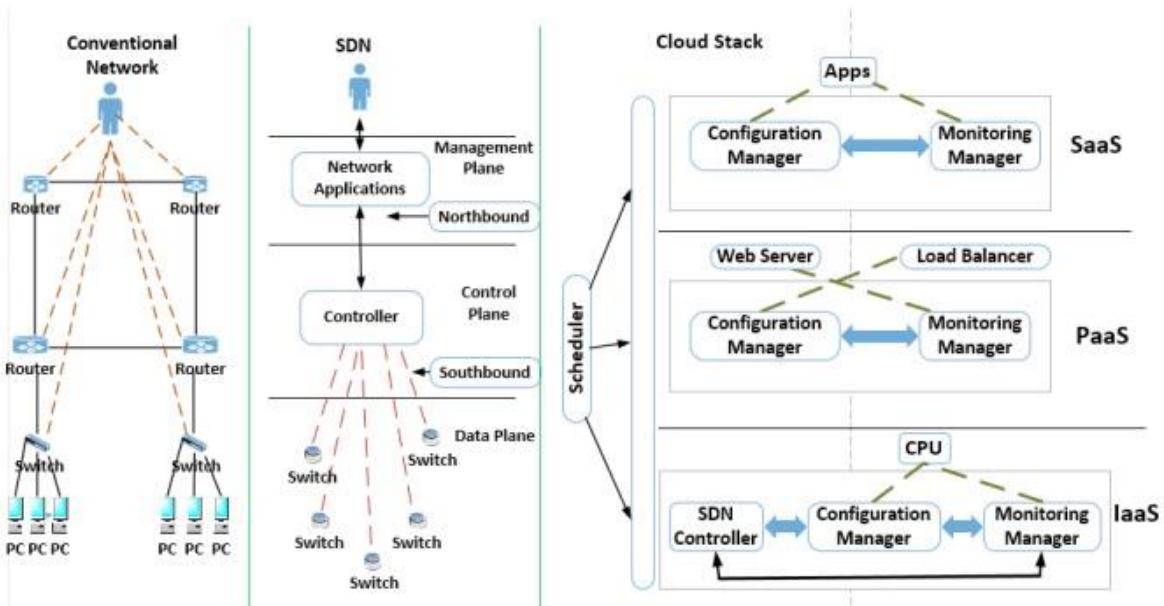


Figure 1: Conventional Network vs SDN network along with Cloud Scheduler, Monitor, Configuring Manager Architecture.

As shown in Figure 1, SDN consists of three layers: The Management plane, the Control plane, and the Dataplane. The Control plane (decision maker) is abstracted away from networking devices (e.g., switches and routers) to a central logic controller, providing a global-network view, programmability, and finer-grained control. It communicates with the data and management planes via two interfaces: southbound and northbound, respectively. The Data plane receives traffic policies from the control plane (e.g., via OpenFlow protocol), and forwards traffic accordingly. The Management plane consists of software services (e.g., via APIs) used by administrators to control and configure network devices.

Scheduling Challenges

Scheduling (a well-known NP-complete problem) is an essential mechanism for cloud computing aiming at managing application performance driven diverse goals including minimizing response times, maximizing resource utilization, and maximizing throughput [3, 4, 5]. It plays an important role at each layer of the Cloud stack where operations are mapped appropriately to intended components based on given Quality of Service (QoS) parameters and scenarios. Numerous studies have contributed to cloud scheduling in terms of workflows and VM, but none of them consider SDN based scheduling along

with the application and VM (mult-level scheduling). SDN can provide the cloud scheduler with critical factors (e.g., link availability, link utilization, and link failure at run time), resulting in smarter scheduling decisions. However, integrating inputs from the three layers is a non-trivial task making scheduling decisions a difficult task. The key challenging issues in SDN based scheduling is that there are too many parameters from each of three layers that need to be considered and integrated. The application parameters (first layer) include response time, processing time, rate of failure, throughput, and priorities. The VMs' parameters (second layer) include total number of VMs, total number of idle VMs, total number of VM reservation requests, and cost. For each VM, the scheduler also need to retrieve CPU capacity and utilization, memory capacity and utilization, and storage capacity and utilization. The SDN's parameters (third layer) include topology (re)configuration based on changes in real time, routing mechanisms based on packets/flows in real time, and link capacity, link failure, link reservation, and bandwidth utilization. The scheduler needs to retrieve simultaneously all the parameters from the three layers of the cloud stack at a particular time instance. The second issue is how the design of decision making model that can integrate these parameters each of which might be giving a conflicting picture. Clearly, designing a scheduler while addressing these issues is undoubtedly a daunting task, requiring further investigation of new schemes, models, and algorithms. Scheduling related research can be categorized into three areas based on above mentioned the problem: application, VM, and SDN scheduling. To the best of our knowledge, no attempt has been made for designing a scheduler that is able to operate according the parameters generated from the proposed architecture. Nevertheless, there are research work addressing the problem considering the application layer or/and VM layer parameters. For example, Sukhpal et al. [3, 4] proposed techniques to provision resources based on QoS requirements (e.g., CPU utilization) resulting in less execution time and cost. Warneke et al. [5] presented a Nephele framework with the principle of job scheduling and task execution where a job manager divides a given job's process into a number of tasks and then assigns VM(s) accordingly. Still, research on designing a multi-level scheduler that is Application-VM-SDN-Agnostic has not been solved yet.

Monitoring challenges

Monitoring can be defined as the process of observing or checking the quality of something over a period of time. Similarly, monitoring in Cloud computing is the process of auditing and managing the operational workflow, and the

different processes within a cloud-based IT asset or infrastructure. Cloud monitoring is an important part of cloud security and management , and it can be implemented within Cloud infrastructure through automated software providing central access and control over the cloud infrastructure. There are many applications of cloud monitoring, such as:

1. Troubleshooting: Monitoring allows analysis of network protocols and behaviour. It also helps in finding network traffic problems.
2. Security: Monitoring mechanisms help with sampling data to look for network-based attacks.
3. Performance: Monitoring helps in optimizing the performance of the CDC resources and network. In SDN, there can be many problems that can remain undetected in the network, thus requiring process monitoring process.

Monitoring in SDN also helps in checking the utilization of CPU, latency and the total request count. Additional activities can also be checked such as memory usage and data volume. Due to these features, monitoring plays an important role in SDN data centres . Network monitoring and visibility has become increasingly challenging since IT infrastructure must support network, server, and storage virtualization, as well as user access to cloud-based applications. Integration of service context and dynamic or real time change are among the hard challenges of the monitoring process. While end-to-end monitoring is important for all type of Cloud applications (multi-tier web, content delivery network, etc.), it is much more critical in context of big data analytics application where network performance determines (e.g. network throughput and latency) performance of virtual machine hosted analytics software component (e.g., mappers and reducers in context of Apache Hadoop) In big data analytics applications, SDN can be leveraged to program switches in order to provide optimal data flow paths between data analytics software components on the fly, during each stage of data analysis. SDN enables better QoS between the virtual machines by dedicating more cross-links between them depending on the type of the analysis process . However, this approach brings several challenges, such as: 1) enabling all vendor technologies to implement both SDN controller integration with virtual machine schedulers. and 2) how to instrument SDN devices with monitoring agent. The implementation of Network Function Virtualization (NFV) can decrease the amount of hardware required to launch and operate network services. However, NFV also brings some challenges of its own, such as:

- 1) performance degradation,

2) the elasticity of service provisioning may require the consolidation and migration of VNFs based on traffic load and user demand.

3) Implementation of NFV brings a new set of security concerns, as virtual applications running within data centres might not be owned by network operators. Monitoring a network is a top concern within IT departments. This is especially the case as monitoring efforts are ubiquitously leveraged to meet network security and performance goals. Monitoring can be effected by the unavailability of useful tools and alerting capabilities. In different networks, it is necessary to capture, store and analyse the vast amount of monitoring data. In short, we can say that monitoring is quite difficult and challenging due to different problems like Integration of service context and Dynamic or real time change. An ideal scenario would be one where the capabilities of SDN and NFV are combined to provide a holistic monitoring solution. SDN has capacity of building and managing large IP/Ethernet networks by separating the network's control, whereas NFV (with the purpose of reducing deployment costs) has capability of virtualizing network functions and migrate them to generic servers [13]. There are several commercial monitoring tools available for big data monitoring, they are: Monitis, RevealCloud, LogicMonitor, Nimsoft, Nagios, SPAE by SHALB, CloudWatch, OpenNebula, CloudHarmony, Windows Azure FC etc. These tools can provide a report in the form of a graph or chart about the load on servers and other information. Each of these tools have their own advantages and drawbacks. For example "RevealCloud" can only provide information for the last 30 days. These tools use different communication protocols for performing tasks, which may or may not work perfectly with SDN and/or NFV communication protocols.

Configuration Management Challenges

Configuration management is the detailed recording and updating of information that describes an enterprise's hardware and software. In a Cloud computing context, configuration management supports the management of services by providing information about how the services are being assembled or bundled together. This information is crucial to the other service management processes, especially for change management, incident management, or problem management. It is also crucial to ensure meeting all agreed-to service levels. Many tools are made for automating the cloud applications configuration management. Among these tools, CFEngine, Puppet, Chef, Ansible are well known. Different cloud-based services (e.g., IaaS, PaaS, or SaaS), will require different levels of configuration management. In the IaaS layer, the consumer of IaaS services usually has control over the configuration aspects of the resources, such as which operating system to run on a virtual

machine, or how to utilize the storage resources, or how to assign IP address to a provisioned virtual machine. In the PaaS layer, configuration management could be performed on the individual components of the platform, such as the automated integration between database and application server layer in context of multi-tier web applications. In the SaaS layer, configuration management operations include configuration end user's account and access credentials with the CDC-hosted SaaS application. To allow information flow between the management plane and other planes (Figure 1), configuration management interfaces need to be in place. These interfaces allow settings to be enforced across network devices, and information to be sent back into the management plane, for example, for reporting to a network administrator [9]. In traditional networks, the logic of the data and control planes is confined inside each network device according to a well-defined set of standardized protocols. With the separation of planes, as SDN promotes, the need to bootstrap the communication between the data and control planes becomes a basic requirement. Configuring this communication can be particularly complex, considering that both planes can operate under protocols defined by software. Moreover, changes in any plane may affect such communication directly. Ideally, a new management interface is required than can manage configuration properly in SDN. Given the need to deploy and setup novel data forwarding devices into a traditional network, administrators must configure these devices in order to have them operational. In the most basic and common scenario, the administrator would interact with a command line interface on the device and have it configured by performing many intricate proprietary commands. In the case of SDN data forwarding devices, there are still a few parameters to be configured, such as the communication policy with the control plane (e.g., drop every packet when the control plane is unavailable or follow the last valid set of rules). In response to these necessities, the Open Networking Foundation (ONF) proposed the OpenFlow Management and Configuration Protocol (OF-Config). This protocol allows operational staff to assign controllers to switches, set ports up/down, configure queues, assign certificates for the communication with the control plane, set up tunnels, handle versioning, and retrieve device capabilities. OF-Config is already a significant step toward tackling dataplane-related management requirements. On the other hand, this protocol is targeted specifically to OpenFlow networks; therefore, it is tied to the limited view of SDN employed by this technology (e.g., fixed dataplane and logically centralized control plane). The challenge of configuration management in SDN is to automate the management of configurations in each plane (application plane, control plane, data plane) by an administrator via management interfaces. SDN and its most known

realization, OpenFlow, has enabled widespread and vendor-neutral programmability of the control plane of the network environment. However, despite such proposals from research and standardization bodies, the management plane still lacks a comparable interface and protocol for capability discovery, device management and monitoring [10]. Therefore, network management is still heavily human-centred with minor autonomous behaviour. The Organization for the Advancement of Structured Information Standards (OASIS) is working on a Topology and Orchestration Specification for Cloud Applications (TOSCA), to enable the creation of portable cloud applications and the automation of their deployment and management. This standard provides a concept named management plan, which makes the management of complex enterprise applications automated, repeatable, traceable, and less error prone. However, the abstraction focuses on higher-level network services such as database management systems (DBMS) and their relationship to other entities, not on details of the individual forwarding elements. In terms of Network Function Virtualisation (NFV), the agile and automated management of virtualized network functions (VNFs) throughout their lifecycles becomes a foremost objective [11]. TOSCA operationalizes the deployment of VNFs and triggers their initial configuration. A TOSCA template is a file-based description of VNF-related things and a set of execution guidelines for deploying and operating them as a single entity on cloud infrastructure. Each TOSCA-defined node has interfaces through which it can be manipulated by an initial configuration application or script, which is referenced in the TOSCA template. However, each VNF instance needs to be further configured at runtime to fulfil the specific needs of a consumer and service. Runtime configuration is beyond the scope of a TOSCA template.

Conclusion

Cloud computing will affect large part of computer industry including Software companies, Internet service providers. Cloud computing makes it very easy for companies to provide their products to end-user without worrying about hardware configurations and other requirements of servers. The cloud computing and virtualization are distinguished by the fact that all of the control plane activities that center around creation, management, and maintenance of the virtual environment, are outsourced to an automated layer that is called as an API and other management servers for the cloud management.

In simple words, the virtualization is a part of cloud computing where manual management is done for interacting with a hypervisor. On the other hand, in cloud computing, the activities are self-managing where an API (Application Program Interface) is used so that the users can self-consume the cloud service.