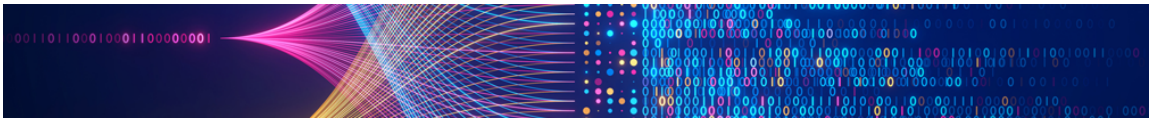
DX604 O1 Data Management at Scale (Spring 2025)

Welcome

This module provides a comprehensive introduction to the design and implementation of the data flows and the associated workflows that are meant to inform online and offline decision-making within large systems. Throughout the module, you will explore the data science lifecycle, including question formulation, data collection and cleaning (data wrangling), exploratory data analysis and visualization, and decision-making. The focus of the module is on tools and methods for data collection, retrieval, integration, processing, and interpretation, using relational (SQL), non-relational (noSQL) and big data tool paradigms to assemble analysis, optimization, and decision-making algorithms to track and scale processing of data as needed. Topics covered include: consolidation, synchronization, and summarization of multiple data streams; data maintenance and availability; optimization and analytics that can operate on large amounts of static or streaming data; and online and offline interactive visualization platforms for presenting and examining data.

Module Learning Outcomes

- Describe the data science lifecycle and various types of data.
- Identify, acquire, cleanse, stage, govern, and present data using ETL (extract, transform, load)/ELT (extract, load, transform) techniques.
- Use SQL for data manipulation and relational database management.
- Discuss the significance of NoSQL databases in contemporary data management.
- Develop strategies for properly distributing data and compute across disparate systems using modern technologies such as MapReduce with Hadoop, in-memory data processing with Spark, and examples of other compute options for data processing.
- Determine the best architectures for specific big data application requirements.
- Review and understand strategies for ensuring quality data with data governance.
- Develop skills to manage data responsibly and effectively.



Week 1 Overview:

This week, we explore the key concepts of the Data Science Lifecycle and how it enables the transformation of raw data into valuable insights. Starting with a review of simple data elements and their uses, we will dive into the data, information, knowledge, wisdom (DIKW) pyramid to understand how data evolve through these stages. You will also learn about the differences between structured, unstructured, and semi-structured data and how each can be framed and curated to add value. Finally, we'll delve into schema design and the essential role it plays in exploring data at deeper levels to extract meaningful insights. In preparation for homework assignments throughout this module, you will set up your Microsoft Azure account and begin exploring the platform.

Learning Objectives

At the end of this week, you will be able to:

- Identify stages of the Data Science Lifecycle
- Discuss simple data elements, uses, and value
- Determine the framing structure of data and various ways it can be curated for adding value
- Understand levels of data exploration with schema design
- Login and create a Microsoft Azure account

Topic Overview: Data Science Lifecycle

We begin by looking at the Data Science Lifecycle, a critical model for addressing business challenges through effective data management. This section explores how data is acquired based on business requirements, the importance of clearly defining those requirements, and ensuring ongoing alignment between data collection and evolving business needs.

Learning Objectives

- Identify stages of the Data Science Lifecycle

Introduction to the Data Science Lifecycle

To recap, the seven stages of the Data Science Lifecycle are as follows:

1. Business understanding
2. Data mining
3. Data cleaning

4. Data exploration
5. Feature engineering
6. Predictive modeling
7. Data visualization

Now that you have reviewed Data Science Lifecycle, there are a few key points to remember as you navigate this model.

Developing Business Understanding

It's essential to start with a clear understanding of the data required to meet the project objectives. Consider the question: What data will be required to satisfy the business (or project) objectives laid out before you? You should have a good idea of at least the base data you'll need for your analysis, but you may need to augment this with additional data for further exploration and development of the overall solution.

Data Mining for Exploring Useful Versus Useless Data

Once you've gathered your data, take the time to assess its relevance and quality. The data mining process is your chance to really understand the data you are working with in your project. For starters, you'll need to know what the data consists of and if it will be sufficient to satisfy your planned analysis. If it does not, you will either need to find some additional data or ask the stakeholder(s) you are working with for more.

The next stage is data cleaning. You'll want to understand the quality of the data and what it will require for cleaning and matching. If the data has many gaps or inconsistencies, you need to know up front that it may sacrifice the quality of the outcomes for your analysis.

Ensuring Satisfaction with Business Requirements

After identifying the data you need and ensuring it is clean and complete, the next step is data exploration. This can be time-consuming and should be thorough enough to satisfy the requirements laid out. The best way to build the analysis iteratively is to work with the data enough to have questions about the requirements from the stakeholder(s). The more communication and understanding you have, the more likely you are to have a successful project. You don't want to work in a vacuum where you emerge at the end and expect a perfect result. Be sure to have regular check-ins on your progress and ask difficult questions as they arise.

Beginning the Model Training Process

Although this course will primarily focus on stages 1 through 4, the Data Science Lifecycle has a few more steps as they apply to a complete machine learning solution. We will cover these steps in lesser detail; however, they are just as critical when building the solution as well as needing continual improvement.

When we have confidence that our data is clean and the data looks to be sufficient for the objectives laid before us, we start to look at feature engineering. In feature engineering, we use what has been learned about the data and specify what information can be used to improve the performance and accuracy of the model. This process can be challenging and requires quite a bit of trial and error. The assigned article for this section, "Understanding the Data Science Lifecycle," describes this in more detail as well as having multiple links providing even deeper explanations and strategies for feature engineering.

The next step is predictive modeling, where the appropriate model is based on algorithms and statistical analysis used to get the best results from the data. This step requires an evaluation of cost tradeoffs between performance and accuracy. It is likely more accurate models can be used; however, the additional accuracy may be too costly in terms of time and compute required. The objectives the project looks to satisfy may have an indicator of which is more important or what the proper mix might be along with the other analysis performed.

Finally, after confidence has grown in the quality of the data, the data that are being analyzed has been identified, and features and model have been selected, data visualization helps to illustrate the outcomes of all the work done to this point. The proper visualization techniques will also be dictated by business objectives, as well as

the most effective way of telling the “story” of the data used and models developed. An effective presentation of the data is a creative process and requires intimate knowledge of all components discussed for this project.

Reading | Agarwal, S. (2018, February 9). Understanding the data science lifecycle. Sudeep.co.

https://www.sudeep.co/data-science/2018/02/09/Understanding-the-Data-Science-Lifecycle.html

Topic 2: Simple Data Elements, Uses, and Value

Data can only be considered valuable if they can be used to satisfy what is required for analysis. In this section, we will discuss how to take raw, less-valuable data and increase its value by structuring the data and using it to assist in the decision-making process. We will explore the DIKW model, different uses of data, and identify the value of it.

Learning Objectives

- Discuss simple data elements, uses, and value

The DIKW Model (Data, Information, Knowledge, Wisdom)

- Base: Data - Raw Facts and Figures
- Information - Provides context, helps answer basic querstion like "who", "what", or "when".
- Data Value - Uniqueness, Timeliness, Accuracy

DIKW Pyramid

The DIKW pyramid, also known as the wisdom hierarchy, represents the relationships between data, information, knowledge, and wisdom, which are key concepts in information science and knowledge management. It organizes these concepts into a hierarchical structure that illustrates their progression in terms of meaning and value.

Overview of Levels

1. **Data**:
    - The foundation of the pyramid.
    - Consists of raw, unorganized facts and signals.
    - Lacks context or interpretation.
2. **Information**:

- Processed and organized data.
- Provides context, making data meaningful.
- Answers questions like "who," "what," "where," and "when."

3. **Knowledge**:
    - Synthesized and applied information.
    - Represents understanding gained from experience or education.
    - Answers "how" questions and facilitates decision-making.
4. **Wisdom**:
    - The pinnacle of the pyramid.
    - Involves deep insight and ethical judgment.
    - Answers "why" questions and focuses on the long-term consequences of decisions.

Application

The DIKW pyramid is widely used in fields like business intelligence, data analytics, and education. It highlights the transformation of raw data into actionable wisdom through contextualization, analysis, and interpretation.

Criticisms

The model has been criticized for oversimplifying complex concepts and for its lack of clarity regarding the transitions between levels. Critics also argue that it assumes a linear progression, which may not always be accurate.

For more detailed exploration, refer to the full Wikipedia article: DIKW Pyramid

2.2 Lesson: Data For Different Purposes

We are surrounded by data in our personal and work lives every minute of every day, unless you are one of the lucky ones who can go “off the grid” occasionally. This data comes in many forms and isn’t limited to just “digital” data:

- We have verbal data received from various sources, such as friends you talk to or other media content like podcasts, YouTube, or television.
- Consuming analog media such as books, newspapers, and magazines is still very common.
- Probably most of all, we receive constant alerts and updates on our computers and smartphones.

These describe our personal interactions with data, but data is being consumed all around us, from Wi-Fi to cellular networks as well as copper and fiber networks on our streets and buildings around us. Clearly it depends on “where you sit” that determines how these various sources of data impact you and what is used to begin to assign value to said data.

2.3 Lesson: Assigning Value to Various Types of Data

Processing data and assigning value to data are an essential part of a daily regimen for a data professional. Assigning value to data shapes how we make decisions, prioritize information, and take action. Processing data involves interpreting, organizing, and transforming raw information into something usable. Once processed, assigning value helps determine its importance, relevance, or usefulness based on your goals or context.

Think about the process of how you assign value to data you receive. Whether you get verbal information from a family member, friend or colleague, it requires mental processing, just like the digital data that may need to be analyzed for another outcome.

From there, you then decide if this information, such as gossip or casual updates, needs further consideration. For example, does the latest update on Taylor Swift and Travis Kelce’s relationship provide any real wisdom or value to your life? This illustrates the process of assigning value to our data, allowing you to decide what data is useful versus what data needs to be thrown away.

Assigning value to data enables us to navigate the overwhelming amount of information we encounter daily, helping us distinguish what is meaningful and relevant from what is not.

Topic Overview: Structure of Data

As discussed, data surrounds us constantly in our daily lives. After we have acquired data in any form, we have to determine the best way to use it by structuring the data for how it will be used. The next few lessons will take us through how the various types are structured and what needs to happen to get the data into a usable format. Next, we will discuss how the data will be used and, finally, assign its value when it is in the required format.

Learning Objectives Determine the structure of data and various ways it can be curated for adding value

3.1 Lesson: Structure, Semi-Structured, and Unstructured Data

Data is stored in many ways, but not all data is as easily acquired as others and will require varying degrees of manipulation to get it into the necessary state. From there, we can get value from the data we need. Let’s talk about it more as we discuss the various structures of data.

Introduction to Data Structures and Value Creation

In this video, we look at the three main types of data: structured, semi-structured, and unstructured data, reviewing examples of each. Then, we discuss what makes data valuable and how to enhance that value.

What Makes Data Valuable?

- Relevance
- Accuracy and Timeliness
-

Data Types

- Structured: Highly organized,stored in predefined formats like rows and columns
- Semi-Structured: Does not fit into strict table format, has some organizational elements (such as tags or metadata) tools must be used to use this data effectively.
  - Examples include JSON or XML data
- Unstructured: no predefined structure; documents, iamges, videos, audio, social media posts
  - referred to as "dark data".
  - Can be sued in social media sentiment analysis, video content evaluation, and customer reviews

What’s The Difference Between Structured, Semi-Structured And Unstructured Data?

When a conversation turns to analytics or big data, the terms structured, semi-structured and unstructured might get bandied about. These are classifications of data that are now important to understand with the rapid increase of semi-structured and unstructured data today as well as the development of tools that make managing and analysing these classes of data possible. Here’s what you need to know.

Structured Data

Data that is the easiest to search and organise, because it is usually contained in rows and columns and its elements can be mapped into fixed pre-defined fields, is known as

structured data. Think about what data you might store in an Excel spreadsheet and you have an example of structured data. Structured data can follow a data model a database designer creates – think of sales records by region, by product or by customer. In structured data, entities can be grouped together to form relations (‘customers’ that are also ‘satisfied with the service). This makes structured data easy to store, analyse and search and until recently was the only data easily usable for businesses. Today, most estimate structured data accounts for less than 20 percent of all data.

Often structured data is managed using Structured Query Language (SQL)—a programming software language developed by IBM in the 1970s for relational databases.

Structured data can be created by machines and humans. Examples of structured data include financial data such as accounting transactions, address details, demographic information, star ratings by customers, machines logs, location data from smart phones and smart devices, etc.

Unstructured Data

A much bigger percentage of all the data is our world is unstructured data. Unstructured data is data that cannot be contained in a row-column database and doesn’t have an associated data model. Think of the text of an email message. The lack of structure made unstructured data more difficult to search, manage and analyse, which is why companies have widely discarded unstructured data, until the recent proliferation of artificial intelligence and machine learning algorithms made it easier to process.

Other examples of unstructured data include photos, video and audio files, text files, social media content, satellite imagery, presentations, PDFs, open-ended survey responses, websites and call centre transcripts/recordings.

Instead of spreadsheets or relational databases, unstructured data is usually stored in data lakes, NoSQL databases, applications and data warehouses. The wealth of information in unstructured data is now accessible and can be automatically processed with artificial intelligence algorithms today. This technology has elevated unstructured data to an extremely valuable resource for organisations.

Semi-Structured Data

Beyond structured and unstructured data, there is a third category, which basically is a mix between both of them. The type of data defined as semi-structured data has some defining or consistent characteristics but doesn’t conform to a structure as rigid as is expected with a relational database. Therefore, there are some organisational properties such as semantic tags or metadata to make it easier to organise, but there’s still fluidity in the data. mail messages are a good example. While the actual content is unstructured, it does contain structured data such as name and email address of sender and recipient,

time sent, etc. Another example is a digital photograph. The image itself is unstructured, but if the photo was taken on a smart phone, for example, it would be date and time stamped, geo tagged, and would have a device ID. Once stored, the photo could also be given tags that would provide a structure, such as ‘dog’ or ‘pet.’

A lot of what people would usually classify as unstructured data is indeed semi-structured, because it contains some classifying characteristics.

The Difference Between Structured, Unstructured, And Semi-Structured Data

To easily understand the differences between the classifications of data, let’s use this analogy to illustrate. When interviewing for a job, let’s say there are three different classifications of interviews: structured, semi-structured and unstructured.

In a structured interview, the interviewer follows a strict script that was defined by the human resources department and is followed for every candidate. Another form of interview is an unstructured interview. In an unstructured interview, it is entirely up to the interviewer to determine the questions and the order they will be asked (or even if they will be asked) for every candidate. A semi-structured interview takes elements from both structured and unstructured interview classifications. It uses the consistency and quantitative elements allowed with the structured interview but offers the freedom to customise based on the circumstances that are more in line with an unstructured interview.

So, for data, structured data is easily organizable and follows a rigid format; unstructured is complex and often qualitative information that is impossible to reduce to or organise in a relational database and semi-structured data has elements of both.

3.2 Lesson: Usng Each Type of Data

As we discussed in the video and reading describing structured, semi-structured, and unstructured data, each type has specific reasons for being structured the way they are, and there is a basis for each. Let’s review what we learned and look at some additional examples of each type.

Relational databases (**structured data**) are very efficient at storing data that nicely fit into rows and columns. Here, we think about business data such as data being stored behind an Enterprise Resource Planning (ERP) or Customer Relationship Management (CRM) system but also personal data, such as tracking your monthly expenses or keeping a family budget. These types of collection areas are held to a strict structure called a schema that helps with querying and using the data quantitatively. When we consider NoSQL, or “non-relational” databases, the data is largely schemaless (but could possibly contain a schema) and, in turn, allows for more flexibility in how the data

are structured. Typically, we would store semi-structured or unstructured data in a NoSQL database. This can lead to challenges when querying the data, but we will discuss that in a later topic in more detail.

Semi-structured data can have a more rigid structure by providing tags within the format to help at least categorize the data being used but isn't beholden to the formal structure of rows and columns as in a structured format. We commonly use semi-structured data when we want to exchange data between web-based systems or, in the case of XML, have structured underpinnings of documents needing formatting, such as in the case of email or Microsoft Word documents. HTTP(S) is the language of webpages, and with the tags used, we are able to apply a format for a website and use other, more modern languages within the tags for more dynamic interactions. Programming languages such as JavaScript and Cascading Style Sheets (CSS) are examples of embeddings in webpages that the HTTP tags will orchestrate for user interaction.

Finally, when we think of **unstructured** data, we have a big melting pot of how our data is stored. As a simple example, the files you store in your laptop, such as pictures, movies, and music files, would all be considered unstructured. Files stored in cloud storage, such as Microsoft OneDrive, Apple iCloud, and Google Drive, would all be considered unstructured as well. When we consider a business use beyond the files we just named, we might use Binary Large Objects (BLOB) storage. Here, we would be capturing backup files, log files, scans of documents, or other files that don't nicely fit in rows and columns.

3.3 Lesson: What Makes Data Valuable

When we talk about the "value" of data, it can be determined by its ability to drive business decisions, optimize operations, and provide insights that create a competitive advantage. The primary factors that make data valuable include:

- **Relevance:** Data is most valuable when it directly aligns with the specific needs and goals of an organization. If it helps answer critical business questions or supports key decision-making processes, its value increases significantly.
- **Accuracy & Quality:** As discussed, high-quality data that are accurate, consistent, complete, and timely is essential for reliable insights. Poor-quality data can lead to flawed conclusions and costly errors, diminishing their value.
- **Timeliness:** Data that are current and up to date have a higher value. Outdated data may not reflect current trends or circumstances, making them less relevant for decision-making.
- **Accessibility & Availability:** Data must be readily accessible to those who need it. If valuable data is locked away in silos or inaccessible due to system constraints, they lose their potential impact. When we discuss the concept of "dark data," these

are some of the challenges faced by organizations that don't have a good process of making the data available.
- **Granularity:** Depending on the level of detail we can acquire, the data we need will dictate the "grain" of the data. The more detailed and granular the data, the more valuable they become for deep analysis and understanding. Summarized monthly sales data are not as valuable as a transaction-by-transaction account because the amount of analysis that can be performed is limited.
- **Uniqueness:** Data that are unique or proprietary (e.g., a customer database with behavioral patterns) can offer a competitive edge for an organization and are therefore more valuable.

The value of data can be increased through several strategic approaches:

- **Data enrichment:** Enhancing existing data by adding external data sources (e.g., demographics, market trends) provides more context and depth, leading to richer insights. For example, combining internal sales data with market demographics can reveal new opportunities for targeting specific customer segments.
- **Data cleansing and quality improvement:** Regularly cleaning and validating data ensures they remain accurate and reliable. This includes removing duplicates, correcting errors, and standardizing formats. High-quality data increase confidence in decision-making and analysis.
- **Data integration and centralization:** Integrating data from disparate sources (e.g., CRM, ERP, and social media) into a centralized system, such as a data warehouse or data lake, creates a unified view. This integration enhances the ability to analyze data holistically and uncover cross-functional insights.
- **Data governance and security:** Implementing data governance frameworks ensures data integrity, security, and compliance. Establishing policies and standards increases trust in data and promotes its appropriate use across the organization.
- **Data analytics and advanced techniques:** Applying advanced analytics, such as machine learning and artificial intelligence (AI), can unlock new insights and predictions that were previously inaccessible. These techniques transform raw data into actionable knowledge, increasing its strategic value.
- **Data sharing and collaboration:** Sharing data across departments or with trusted partners (while maintaining privacy and security) can amplify its value. Collaborative use of data leads to shared insights and innovation that benefit the entire organization.

In summary, data becomes valuable when it is relevant, accurate, timely, and accessible, and its value can be increased through enrichment, integration, quality improvement, and strategic use of advanced analytics.

Topic 4: Levels of Data Exploration with Schema Design

Data exploration and schema design are essential processes of effectively managing and utilizing data in any organization. These concepts are typically broken down into three levels: conceptual, logical, and physical schemas. Each level serves a distinct purpose and progressively builds upon the previous one. Together, these levels of schema design facilitate a structured approach to data management, enabling organizations to create scalable, well-structured data systems that support efficient analysis and decision-making.

Learning Objectives

- Differentiate the schema types (conceptual, logical, and physical)

4.1 Lesson: Schema Types: Conceptual, Logical, Physical

Schemas are crucial for organizing and storing data efficiently at varying levels of focus, serving as blueprints for structuring information. In this video, we'll explore the three levels of schemas: conceptual, logical, and physical.

Schema Design Framework

- **Conceptual:** (or big-picture data organization) The conceptual schema offers a high-level, abstract view of the data within a system. This is the first step in designing any data solution. It defines key entities and relationships without getting into technical difficulties, like how the data is stored.
- **Logical:** Once the conceptual model is clear, the logical schema translates this into structured data elements within a database. It defines the tables, columns, data types, and relationships between entities. The focus is on structure, now yet how the data will be stored physically.
- In our CRM Example, the logical schema would include a table for customers, with columns for name, contact details, and order history.
    - It specifies the data types, (for example, string for Name or integers for orderID), and outlines relationships between the tables, such as how customer_orders are linked to products.
    - This stage ensures the data is organized logically to support business processes.
- **Physical:** Physical Scemas focus on optimizing data for performance. The physical schema brings everything down to the technical details of how data is stored and accessed.
    - It deals with storage formats, indexing, and performance optimization, ensuring the database can handle the required data load efficiently.
    - In our CRM example, the physical schema would define where customer and ordered data are stroed, on a cloud server or on-premises, and how it is

optimized for retrieval. It would also address techniques like indexing or partitioning to speed up queries. This stage is crucial for ensuring that the database can scale and perform well as data volumes grow.

Understanding the levels of schema design: conceptual, logical, and physical, enables us to build scalable databases.

- Conceptual databases provide the big picture
- Logical databaes give structure
- Physical Schemas ensure optimal performance

To review, by adhering to the schema design that is appropriate for our use case, we provide a structured approach to designing the approach we will take:

- The **conceptual schema** offers a high-level, abstract view of the data, focusing on what data elements exist and how they relate to one another, without delving into technical details. This level is crucial for aligning business objectives with data requirements.
- The **logical schema** refines this structure, providing a detailed blueprint of how the data is organized within the database, including entities, attributes, and relationships, but without specifying storage or access methods.
- Finally, the **physical schema** translates this logical design into the technical implementation, detailing how data are stored, accessed, and optimized for performance.

When discussing the three schemas, we should specify the level of detail needed to ensure avoiding miscommunications by choosing the correct level to satisfy requirements.

Topic 5: Introduction to Microsoft Azure

Microsoft Azure is a cloud computing platform that will give you hands-on experience managing data at-scale and applying the concepts learned throughout this course. Azure provides a wide range of services to help you build, manage, and deploy applications and data on a global scale.

In this module, you'll use Azure as the environment for completing your homework. It offers tools for data storage, processing, and analytics, enabling you to work with large datasets, experiment with different data structures, and explore various levels of data.

Introduction to Data Engineering on Azure

Microsoft Azure provides a comprehensive platform for data engineering; but what is data engineering? Complete this module to find out.

Learning objectives In this module you will learn how to:

- Identify common data engineering tasks
- Describe common data engineering concepts
- Identify Azure services for data engineering

Introduction:

In most organizations, a data engineer is the primary role responsible for integrating, transforming, and consolidating data from various structured and unstructured data systems into structures that are suitable for building analytics solutions. An Azure data engineer also helps ensure that data pipelines and data stores are high-performing, efficient, organized, and reliable, given a specific set of business requirements and constraints.

What is data engineering?

The data engineer will often work with multiple types of data to perform many operations using many scripting or coding languages that are appropriate to their individual organization.

Types of Data

there are three primary types of data that a data engineer will work with:

Structured

Structured data primarily comes from table-based source systems such as a relational database or from a flat file such as a comma separated (CSV) file. The primary element of a structured file is that the rows and columns are aligned consistently throughout the file.

Semi-Structured

Semi-structured data is data such as JavaScript object notation (JSON) files, which may require flattening prior to loading into your source system. When flattened, this data doesn't have to fit neatly into a table structure.

Unstructured

Unstructured data includes data stored as key-value pairs that don't adhere to standard relational models and Other types of unstructured data that are commonly used include portable data format (PDF), word processor documents, and images.

Data Operations

As a data engineer some of the main tasks that you'll perform in Azure include *data integration, data transformation, and data consolidation*.

Data Integration

Data Integration involves establishing links between operational and analytical services and data sources to enable secure, reliable access to data across multiple systems. For example, a business process might rely on data that is spread across multiple systems, and a data engineer is required to establish links so that the required data can be extracted from all of these systems.

Data Transformation

Operational data usually needs to be *transformed* into suitable structure and format for analysis, often as part of an *extract, transform, and load* (ETL) process; though

increasingly a variation in which you *extract, load, and transform* (ELT) the data is used to quickly ingest the data into a data lake and then apply "big data" processing techniques to transform it. Regardless of the approach used, the data is prepared to support downstream analytical needs.

Data Consolidation

Data consolidation is the process of combining data that has been extracted from multiple data sources into a consistent structure - usually to support analytics and reporting. Commonly, data from operational systems is extracted, transformed, and loaded into analytical stores such as a data lake or data warehouse.

Common Languages

Data Engineers must be proficient with a range of tools and scripting languages - in particular SQL and Python, and potentially others.

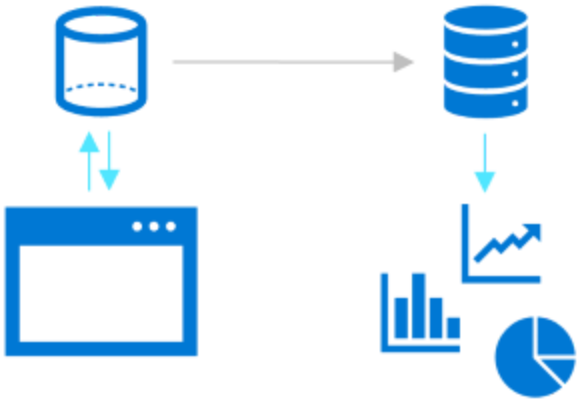
- SQL - One of the most common languages data engineers use is SQL, or Structured Query Language, which is a relatively easy language to learn. SQL uses queries that include SELECT, INSERT, UPDATE, and DELETE statements to directly work with the data stored in tables.
- Python - Python is one of the most popular and fastest growing programming languages in the world. It's used for all sorts of tasks, including web programming and data analysis. It has emerged as the language to learn for machine learning, and is increasing in popularity in data engineering with the use of notebooks.
- Others - Depending upon the needs of the organization and your individual skill set, you may also use other popular languages within or outside of notebooks including R, Java, Scala, .NET, and more. The use of notebooks is growing in popularity, and allows collaboration using different languages within the same notebook.

Important data engineering concepts

There are some core concepts with which data engineers should be familiar. These concepts underpin many of the workloads that data engineers must implement and

support.

Operational and analytical data



Operational data is usually transactional data that is generated and stored by applications, often in a relational or non-relational database. *Analytical* data is data that has been optimized for analysis and reporting, often in a data warehouse.

One of the core responsibilities of a data engineer is to design, implement, and manage solutions that integrate operational and analytical data sources or extract operational data from multiple systems, transform it into appropriate structures for analytics, and load it into an analytical data store (usually referred to as ETL solutions).

Streaming data

Streaming data refers to perpetual sources of data that generate data values in real-time, often relating to specific events. Common sources of streaming data include internet-of-things (IoT) devices and social media feeds.

Data engineers often need to implement solutions that capture real-time stream of data and ingest them into analytical data systems, often combining the real-time data with other application data that is processed in batches.

Data Pipelines

Data pipelines are used to orchestrate activities that transfer and transform data. Pipelines are the primary way in which data engineers implement repeatable extract, transform, and load (ETL) solutions that can be triggered based on a schedule or in response to events.

Data Lakes

A data lake is a storage repository that holds large amounts of data in native, raw formats. Data lake stores are optimized for scaling to massive volumes (terabytes or petabytes) of data. The data typically comes from multiple heterogeneous sources, and may be structured, semi-structured, or unstructured.

The idea with a data lake is to store everything in its original, untransformed state. This approach differs from a traditional data warehouse, which transforms and processes the data at the time of ingestion.

Data Warehouses

A data warehouse is a centralized repository of integrated data from one or more disparate sources. Data warehouses store current and historical data in relational tables that are organized into a schema that optimizes performance for analytical queries.

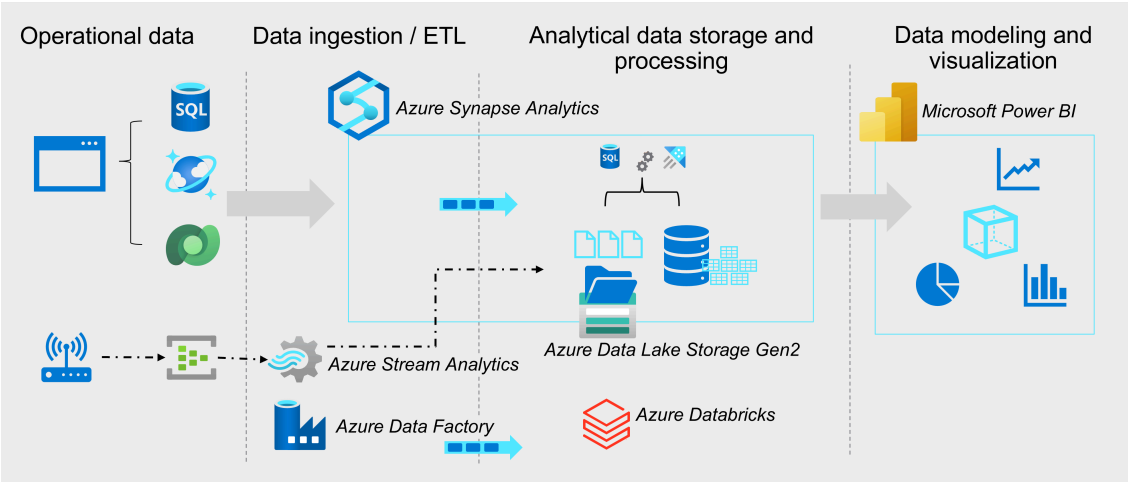
Data engineers are responsible for designing and implementing relational data warehouses, and managing regular data loads into tables.

Apache Spark

Apache Spark is a parallel processing framework that takes advantage of in-memory processing and a distributed file storage. It's a common open-source software (OSS) tool for big data scenarios.

Data engineers need to be proficient with Spark, using notebooks and other code artifacts to process data in a data lake and prepare it for modeling and analysis.

Data engineering in Microsoft Azure



Microsoft Azure includes many services that can be used to implement and manage data engineering workloads.

The diagram displays the flow from left to right of a typical enterprise data analytics solution, including some of the key Azure services that may be used. Operational data is generated by applications and devices and stored in Azure data storage services such as Azure SQL Database, Azure Cosmos DB, and Microsoft Dataverse. Streaming data is captured in event broker services such as Azure Event Hubs.

This operational data must be captured, ingested, and consolidated into analytical stores; from where it can be modeled and visualized in reports and dashboards. These tasks represent the core area of responsibility for the data engineer. The core Azure technologies used to implement data engineering workloads include:

- Azure Synapse Analytics
- Azure Data Lake Storage Gen2
- Azure Stream Analytics
- Azure Data Factory
- Azure Databricks

The analytical data stores that are populated with data produced by data engineering workloads support data modeling and visualization for reporting and analysis, often using sophisticated visualization tools such as Microsoft Power BI

Introduction to Azure Data Lake Storage Gen2

Many organizations have spent the last two decades building data warehouses and business intelligence (BI) solutions based on relational database systems. Many BI solutions have lost out on opportunities to store unstructured data due to cost and complexity in these types of data in databases.

Data lakes have become a common solution to this problem. A data lake provides file-based storage, usually in a distributed file system that supports high scalability for massive volumes of data. Organizations can store structured, semi-structured, and unstructured files in the data lake and then consume them from there in big data processing technologies, such as Apache Spark.

Azure Data Lake Storage Gen2 provides a cloud-based solution for data lake storage in Microsoft Azure, and underpins many large-scale analytics solutions built on Azure.

Understand Azure Data Lake Storage Gen2

A data lake is a repository of data that is stored in its natural format, usually as blobs or files. Azure Data Lake Storage is a comprehensive, massively scalable, secure, and cost-effective data lake solution for high performance analytics built into Azure.

Azure Data Lake Storage combines a file system with a storage platform to help you quickly identify insights into your data. Data Lake Storage builds on Azure Blob storage capabilities to optimize it specifically for analytics workloads. This integration enables analytics performance, the tiering and data lifecycle management capabilities of Blob storage, and the high-availability, security, and durability capabilities of Azure Storage.

Benefits

Data Lake Storage is designed to deal with this variety and volume of data at exabyte scale while securely handling hundreds of gigabytes of throughput. With this, you can use Data Lake Storage Gen2 as the basis for both real-time and batch solutions.

Hadoop Campataible Access

A benefit of Data Lake Storage is that you can treat the data as if it's stored in a Hadoop Distributed File System (HDFS). With this feature, you can store the data in one place and access it through compute technologies including Azure Databricks, Azure HDInsight, and Azure Synapse Analytics without moving the data between environments. The data engineer also has the ability to use storage mechanisms such as the parquet format, which is highly compressed and performs well across multiple platforms using an internal columnar storage.

Security

Data Lake Storage supports access control lists (ACLs) and Portable Operating System Interface (POSIX) permissions that don't inherit the permissions of the parent directory. In fact, you can set permissions at a directory level or file level for the data stored within the data lake, providing a much more secure storage system. This security is configurable through technologies such as Hive and Spark or utilities such as Azure Storage Explorer, which runs on Windows, macOS, and Linux. All data that is stored is encrypted at rest by using either Microsoft or customer-managed keys.

Performance

Azure Data Lake Storage organizes the stored data into a hierarchy of directories and subdirectories, much like a file system, for easier navigation. As a result, data processing requires less computational resources, reducing both the time and cost.

Data redundancy

Data Lake Storage takes advantage of the Azure Blob replication models that provide data redundancy in a single data center with locally redundant storage (LRS), or to a secondary region by using the Geo-redundant storage (GRS) option. This feature ensures that your data is always available and protected if catastrophe strikes.

> #### Tip
>
> Whenever planning for a data lake, a data engineer should give thoughtful consideration to structure, data governance, and security. This should include consideration of factors that can influence lake structure and organization, such as:
>
> - Types of data to be stored
> - How the data will be transformed
> - Who should access the data
>
> -What are the typical access patterns
>
> This approach will help determine how to plan for access control governance across your lake. Data engineers should be proactive in ensuring that the lake doesn't become the proverbial data swamp which becomes inaccessible and non-useful to users due to the lack of data governance and data quality measures. Establishing a baseline and following best practices for Azure Data Lake will help ensure a proper and robust implementation that will allow the organization to grow and gain insight to achieve more.

In [ ]: