All Lesson Combined:

TOPIC 1

Lesson 1: What & Why Data Engineering?

Objective:

Understand the basics and importance of data engineering.

1. Definition of Data Engineering:

Data Engineering involves designing, building, and maintaining systems and architecture that allow for the collection, storage, and analysis of data. It's about creating the infrastructure that data scientists and analysts need to perform their tasks effectively.

Imagine data engineering as the plumbing system of the data world. Just as a plumber lays down pipes to ensure water flows smoothly throughout a building, a data engineer sets up data pipelines to ensure data flows seamlessly from one place to another. This includes everything from extracting data from various sources, transforming it into a usable format, and loading it into a storage system where it can be easily accessed and analyzed.

2. Importance in Various Industries:

Data Engineering is crucial in various industries for several reasons:

- **Finance:**
  - **Fraud Detection:** By analyzing transaction data, banks can detect unusual patterns that may indicate fraudulent activity.
  - **Risk Management:** Financial institutions use data to assess and manage risk, ensuring they can make informed decisions about lending and investments.
  - **Market Analysis:** Data engineering enables the analysis of vast amounts of market data to identify trends and make strategic investment decisions.

Imagine a bank that needs to process millions of transactions daily. Without proper data engineering, it would be impossible to sift through this data to detect fraud or analyze market trends in real-time.

- **Healthcare:**
  - **Patient Care:** Hospitals use data to improve patient care by tracking medical histories, treatment plans, and outcomes.
  - **Medical Records Management:** Efficient data engineering ensures that patient records are accurately maintained and easily accessible.
  - **Operational Efficiency:** Data helps in streamlining hospital operations, from managing staff schedules to optimizing the supply chain for medical equipment.

Consider a hospital that uses data from various sources – patient records, lab results, and treatment plans – to create a comprehensive view of each patient's health. This holistic view enables doctors to make better-informed decisions about patient care.

- **Technology:**
    - **AI and Machine Learning:** Data engineering provides the large datasets needed to train and deploy AI and machine learning models.
    - **User Experience:** Tech companies use data to understand user behavior and improve their products and services.
    - **Innovation:** Data drives innovation by enabling companies to identify new opportunities and optimize existing processes.

Think of a tech company developing a new AI-powered recommendation system. Data engineering ensures that the system has access to the vast amounts of data it needs to learn and make accurate recommendations.

3. Support for Data Science, Machine Learning, and Business Intelligence:

Data Engineering provides the foundational support for various data-driven domains:

- **Data Science:** Supplies clean, structured data for building predictive models.
- **Machine Learning:** Facilitates the training and deployment of algorithms by providing large datasets.
- **Business Intelligence:** Ensures accurate and timely data is available for reporting and strategic decision-making.

Imagine a chef preparing a meal. Before the chef can start cooking, someone needs to gather all the ingredients, clean and prepare them, and ensure everything is organized and ready to use. In this analogy, the data engineer is the person who prepares and organizes the ingredients (data), making it easier for the chef (data scientist) to create a delicious meal (analysis or model).

Summary:

In this lesson, we introduced the foundational concepts of data engineering, highlighting its significance across various industries and its support for data science, machine learning, and business intelligence.

Lesson 2: The Role of a Data Engineer

Objective:

Learn about the responsibilities and skills required for a data engineer.

1. Typical Responsibilities of a Data Engineer:

Data Engineers play a critical role in managing and optimizing data systems. Their daily tasks often include:

- **Data Pipeline Development:** Creating and maintaining data pipelines that automate the process of moving data from source systems to data warehouses or data lakes.
    - **Example:** Developing a pipeline that extracts sales data from a company's transaction system, transforms it into a consistent format, and loads it into a data warehouse for analysis.
- **ETL Processes:** Designing and implementing ETL (Extract, Transform, Load) processes to ensure data is accurately and efficiently moved from one system to another.
    - **Example:** Extracting customer data from multiple sources, cleaning and standardizing it, and loading it into a centralized database.
- **Data Quality Assurance:** Ensuring the data is accurate, complete, and reliable by performing data validation, cleaning, and profiling.
    - **Example:** Regularly checking data for inconsistencies or missing values and implementing procedures to correct any issues.
- **Database Management:** Setting up and maintaining databases and data storage solutions, ensuring they are optimized for performance and scalability.
    - **Example:** Configuring a cloud-based data warehouse to store large volumes of data and ensure fast query performance.
- **Collaboration:** Working closely with data scientists, analysts, and other stakeholders to understand their data needs and provide the necessary support.
    - **Example:** Collaborating with data scientists to provide clean, preprocessed data for machine learning model training.

2. Key Skills and Technologies:

Data Engineers need a diverse set of skills and familiarity with various technologies, including:

- **Programming Languages:**
    - **SQL:** Essential for querying and managing databases.
    - **Python:** Widely used for scripting, data manipulation, and building data pipelines.
    - **Java/Scala:** Often used for working with big data tools like Apache Spark.
- **Data Storage Solutions:**
    - **Relational Databases:** Such as MySQL, PostgreSQL, and SQL Server for structured data.
    - **NoSQL Databases:** Like MongoDB and Cassandra for unstructured or semi-structured data.
    - **Data Warehouses:** Such as Amazon Redshift, Google BigQuery, and Snowflake for large-scale data storage and analytics.
    - **Data Lakes:** Using platforms like Apache Hadoop and Amazon S3 for storing raw data in its native format.
- **Data Processing Tools:**
    - **Apache Spark:** For distributed data processing and large-scale data analytics.
    - **Apache Kafka:** For real-time data streaming and event-driven data pipelines.
    - **ETL Tools:** Such as Talend, Apache NiFi, and Informatica for designing and managing ETL processes.
- **Cloud Platforms:**
    - **AWS (Amazon Web Services):** Offering services like S3 for storage, Redshift for data warehousing, and EMR for big data processing.
    - **Azure:** With services like Azure Data Lake, Azure SQL Database, and Azure Synapse Analytics.

    - **GCP (Google Cloud Platform):** Featuring BigQuery for data warehousing, Cloud Storage for data lakes, and Dataflow for stream and batch processing.

3. Daily Tasks and Examples:

A day in the life of a data engineer can be quite dynamic, involving various tasks to ensure data systems are running smoothly and efficiently:

- **Morning:**
    - **Monitoring Data Pipelines:** Checking the status of overnight ETL jobs, ensuring they completed successfully without errors.
    - **Example:** Reviewing logs to verify that a nightly data ingestion pipeline processed all incoming data correctly.
- **Midday:**
    - **Collaborating with Teams:** Meeting with data scientists to discuss data requirements for a new machine learning project.
    - **Example:** Understanding the specific data transformations needed to prepare the dataset for model training.
- **Afternoon:**
    - **Developing New Pipelines:** Writing code to create a new data pipeline that integrates data from a new source system.
    - **Example:** Building a pipeline that extracts social media data, processes it to remove noise, and loads it into a data lake for analysis.
- **Evening:**
    - **Data Quality Checks:** Running scripts to profile data and identify any quality issues that need to be addressed.
    - **Example:** Identifying missing values in a customer dataset and implementing a process to fill in the gaps with appropriate data.

Summary:

In this lesson, we explored the role of a data engineer, highlighting their typical responsibilities, the key skills and technologies they use, and examples of their daily tasks. Data engineers are essential in creating and maintaining the infrastructure that enables organizations to leverage data for decision-making and innovation.

Lesson 3: Key Concepts and Terminologies

Objective:

Get familiar with essential data engineering concepts and terminology.

1. Critical Data Engineering Concepts:

Data engineering involves several key concepts that form the foundation of effective data management and processing. Understanding these concepts is crucial for anyone looking to work in this field.

- **Data Ingestion:**
    - **Definition:** The process of importing data from various sources into a database or data warehouse.
    - **Example:** Collecting data from transactional systems, sensors, social media, and external APIs.
    - **Analogy:** Imagine data ingestion like filling a water tank. The water comes from multiple streams (data sources), and the tank (database) needs to store it all.
- **Data Transformation:**
    - **Definition:** The process of converting data from its raw format into a more suitable format for analysis.
    - **Example:** Cleaning data, removing duplicates, standardizing formats, and enriching data with additional information.
    - **Analogy:** Think of data transformation like making a smoothie. You take raw ingredients (data), blend them together to create a consistent mixture, and add any necessary enhancements (enrichments).
- **Data Storage:**
    - **Definition:** Storing data in a structured and organized manner to facilitate easy access and analysis.
    - **Example:** Using relational databases, NoSQL databases, data warehouses, and data lakes.
    - **Analogy:** Data storage is like organizing a library. Books (data) are categorized and shelved in a way that makes them easy to find and use.
- **Data Governance:**
    - **Definition:** Establishing policies and procedures to ensure data quality, security, and compliance.
    - **Example:** Implementing access controls, data encryption, and data quality checks.
    - **Analogy:** Data governance is like managing a bank. It involves setting rules and protocols to ensure that money (data) is handled securely, accurately, and according to regulations.
- **Data Security:**
    - **Definition:** Protecting data from unauthorized access, breaches, and other security threats.
    - **Example:** Using encryption, firewalls, and secure access controls.
    - **Analogy:** Data security is like installing a security system in your house. It ensures that valuable possessions (data) are protected from theft or damage.

2. Explanation of Key Terminologies:

Understanding the terminology used in data engineering is essential for effective communication and collaboration within the field.

- **ETL (Extract, Transform, Load):**
    - **Definition:** A process that involves extracting data from source systems, transforming it into a usable format, and loading it into a destination system.
    - **Example:** Extracting sales data from a transactional system, cleaning and standardizing it, and loading it into a data warehouse for analysis.
    - **Analogy:** ETL is like making a smoothie. You extract ingredients (data) from different places, blend them together (transform), and pour the smoothie into a glass (load).
- **ELT:**

    - **Definition:** A process that involves extracting data from source systems, transforming it into a usable format, and loading it into a destination system.
    - **Example:** Extracting sales data from a transactional system, cleaning and standardizing it, and loading it into a data warehouse for analysis.
    - **Analogy:** ETL is like baking a cake. You collect the ingredients (extract), mix and bake them according to the recipe (transform), and then place the finished cake on a serving plate (load).
- **Data Warehouse:**
    - **Definition:** A centralized repository where data is stored for analysis and reporting.
    - **Example:** Using Amazon Redshift to store and analyze large volumes of structured data.
    - **Analogy:** A data warehouse is like a library. It stores vast amounts of information (data) organized in a way that makes it easy to find and use.
- **Data Lake:**
    - **Definition:** A storage system that holds a large amount of raw data in its native format until it is needed.
    - **Example:** Using Amazon S3 to store raw log files, images, and other unstructured data.
    - **Analogy:** A data lake is like a large storage room. It keeps everything as it is until someone comes in to organize and use the items.
- **Real-Time Processing:**
    - **Definition:** Processing data as it arrives, enabling immediate analysis and action.
    - **Example:** Using Apache Kafka to process streaming data from IoT devices in real-time.
    - **Analogy:** Real-time processing is like live broadcasting. Data is captured and processed instantly, allowing for immediate viewing and response.
- **Batch Processing:**
    - **Definition:** A method of processing data where transactions are collected over a period of time and processed together in a single batch.
    - **Example:** Using Apache Hadoop to process large volumes of log files overnight.
    - **Analogy:** Batch processing is like doing laundry. You gather dirty clothes over time (collect transactions), and then wash them all at once in a single load (process the batch).

- **Full Load:** with a full load, the entire dataset is dumped, or loaded, and is then completely replaced (i.e. deleted and replaced) with the new, updated dataset. No additional information, such as timestamps, is required. For example, take a store that uploads all of its sales through the ETL process in data warehouse at the end of each day. Let’s say 5 sales were made on a Monday, so that on Monday night a table of 5 records would be uploaded. Then, on Tuesday, another 3 sales were made which need to be added. So on Tuesday night,

assuming a full load, Monday's 5 records as well as Tuesday's 3 records are uploaded – an inefficient system, although relatively easy to set up and maintain. While this example is overly simplified, the principle is the same.

- **Incremental load:** only the difference between the target and source data is loaded through the ETL process in data warehouse. There are 2 types of incremental loads, depending on the volume of data you're loading: streaming incremental load and batch incremental load. Following the previous example, the store that made 3 sales on Tuesday will load only the additional 3 records to the sales table, instead of reloading all records. This has the advantage of saving time and resources but increases complexity. Incremental loading is of course much faster than a full load. The main drawback to this type of loading is maintainability. Unlike a full load, with an incremental load you can't re-run the entire load if there's an error. In addition to this, files need to be loaded in order, so errors will compound the issue as other data queues up.

Summary:

In this lesson, we explored essential data engineering concepts and terminologies, including data ingestion, transformation, storage, governance, and security. We also explained key terms such as ETL, data pipelines, data warehouses, data lakes, and real-time processing. Understanding these concepts and terminologies is crucial for anyone working in the field of data engineering, as they form the foundation of effective data management and processing.

TOPIC 2

Today, we'll explore how data flows through a modern data system, examining the key components and their interactions. We'll also touch on the undercurrents that influence every stage of this process.

I. The Core Pipeline

Let's start with the heart of our lifecycle: the core pipeline.

1. Generation: Our journey begins with data generation. This is where data is born, originating from various sources such as user interactions, IoT devices, or business transactions. Think of this as the wellspring of our data river.
2. Ingestion: As our data flows from its source, it enters the ingestion phase. Here, we collect and import the raw data into our system. It's like a giant funnel, gathering information from multiple sources and preparing it for the next stage.
3. Storage: Once ingested, our data needs a home. The storage layer acts as a vast reservoir, holding both raw and processed data. This could be a data lake, a data warehouse, or a combination of storage solutions.
4. Transformation: Now comes the alchemical stage of our journey. In the transformation phase, raw data is refined and restructured. We clean it, enrich it, and shape it into a form that's useful for analysis. It's like turning ore into gold.
5. Serving: With our data transformed, it's time to serve it up. The serving layer makes our processed data available for consumption. It's the distribution center, packaging our data products for various uses.

II. Advanced Use Cases

From our serving layer, data flows into three key advanced use cases:

1. Analytics: Here, our data powers business intelligence, generating insights that drive decision-making. It's where we answer questions and uncover trends.
2. Machine Learning: Our refined data becomes the fuel for predictive models. In this stage, we're not just understanding the past, but predicting the future.
3. Reverse ETL: This is a fascinating newer concept. Reverse ETL takes our insights and predictions and feeds them back into operational systems. It's closing the loop, allowing our data-driven insights to directly influence business processes.

III. The Undercurrents

Flowing beneath our visible pipeline are six critical undercurrents. These influence every stage of our lifecycle:

1. Security: Ensuring our data remains protected and compliant at all times.
2. Data Management: Overseeing the quality, metadata, and governance of our data.

3. DataOps: Applying DevOps principles to create efficient, automated data workflows.
4. Data Architecture: Designing the overall structure of our data systems.
5. Orchestration: Coordinating and scheduling our various data processes.
6. Software Engineering: Applying robust development practices to our data engineering tasks.

Conclusion: The Data Engineering Lifecycle is a complex, interconnected system. Each component builds upon the last, creating a flowing river of data that powers our modern, data-driven world. The undercurrents ensure this river flows smoothly, securely, and efficiently.

As we progress through this course, we'll dive deeper into each of these components, exploring advanced concepts and practical applications. Remember, in the world of data engineering, everything is connected. Understanding these connections is key to mastering the field.

TOPIC 3

Lesson 1: What is Data Quality and Why is it Important?

Objective:

Understand the concept of data quality and its significance in data engineering.

1. Definition of Data Quality:

Data quality refers to the condition of a dataset that determines its fitness to serve its purpose in a given context. High-quality data is accurate, complete, reliable, and relevant.

Dimensions of Data Quality:

- **Accuracy:** Data accurately reflects the real-world entities or values it represents. This means there are no errors, distortions, or misrepresentations in the data, ensuring that it can be reliably used for decision-making and analysis.
- **Completeness:** All necessary data is available without any missing elements. This ensures that the dataset contains all required information, which is crucial for comprehensive analysis and reporting. Incomplete data can lead to incorrect conclusions and decisions.
- **Consistency:** Data remains uniform across different systems and formats, maintaining harmony in how information is presented and interpreted. Consistent data ensures that information is reliable and can be cross-referenced accurately, reducing the risk of discrepancies.

- **Timeliness:** Data is up-to-date and available when needed. Timely data means that the information reflects the most current state of affairs, which is essential for making relevant and accurate decisions. Outdated data can lead to missed opportunities and incorrect assessments.
- **Validity:** Data conforms to the defined formats, rules, and standards. Valid data ensures that the information is in the correct format and follows established guidelines, making it usable and reliable. Invalid data can lead to errors in processing and interpretation.
- **Uniqueness:** Data is free from duplicates, ensuring that each piece of information is distinct. Unique data prevents redundancy and ensures that every entry is only recorded once. This is crucial for maintaining the integrity and efficiency of the database, as duplicates can lead to confusion and inefficiency.

2. The Impact of Poor Data Quality:

On Business Decisions:

Poor data quality can lead to incorrect business decisions, resulting in financial losses, missed opportunities, and reduced competitiveness.

On Operations:

Operational inefficiencies can arise from using poor-quality data, leading to increased costs, customer dissatisfaction, and compliance issues.

Examples of Data Quality Issues and Their Consequences:

- **Inaccurate Customer Data:** Leads to incorrect targeting in marketing campaigns, resulting in wasted resources and reduced ROI.
- **Incomplete Financial Data:** Causes errors in financial reporting and decision-making, potentially leading to regulatory penalties.
- **Inconsistent Product Data:** Results in supply chain inefficiencies and poor inventory management.

3. Importance of Data Quality in Data Engineering:

Ensuring Reliable Analytics:

High-quality data is essential for accurate analytics, machine learning models, and business intelligence. Reliable data leads to trustworthy insights and better decision-making.

Maintaining Data Integrity:

Ensuring data quality helps maintain the integrity of data systems, preventing errors and inconsistencies that can compromise data reliability.

Enhancing Customer Trust:

Providing high-quality data builds customer trust, as accurate and reliable information is critical for customer satisfaction and loyalty.

Summary:

In this lesson, we explored the concept of data quality, its dimensions, and its significance in data engineering. We discussed the impact of poor data quality on business decisions and operations and highlighted the importance of ensuring high-quality data for reliable analytics, data integrity, and customer trust.

Lesson 2: Ensuring Data Quality: Techniques and Tools

Objective:

Learn about various techniques and tools used to ensure data quality.

1. Data Profiling:

Definition and Purpose:

Data profiling involves analyzing data to understand its structure, content, and quality. It helps identify data quality issues and understand the characteristics of the dataset.

Tools for Data Profiling:

- **Power BI:** allows you to analyze data quality through features like column statistics and distribution visualizations within the Power Query Editor.
- **Informatica Data Quality:** Offers comprehensive data profiling capabilities to identify data anomalies and inconsistencies.

2. Data Cleansing:

Techniques for Cleaning Data:

- **Removing Duplicates:** Identifying and eliminating duplicate records to ensure uniqueness.
- **Correcting Errors:** Fixing inaccuracies, such as typos or incorrect values.
- **Standardizing Formats:** Ensuring data conforms to a consistent format (e.g., date formats, address formats).

3. Data Validation:

Importance of Validating Data:

Data validation ensures that data meets defined standards and business rules before it is used in analysis or decision-making.

Techniques for Data Validation:

- **Constraints:** Enforcing rules, such as unique keys or mandatory fields, to ensure data validity.
- **Rules:** Applying business rules to check for logical consistency (e.g., order date must be before delivery date).

4. Data Monitoring:

Ongoing Monitoring for Data Quality:

Continuous monitoring helps detect data quality issues in real-time, allowing for immediate corrective actions.

Tools for Data Monitoring:

- **Monte Carlo:** A data observability platform for monitoring data quality and reliability.
- **Datafold:** Provides data monitoring and validation capabilities to ensure data accuracy and consistency.

Summary:

In this lesson, we explored various techniques and tools used to ensure data quality, including data profiling, data cleansing, data validation, and data monitoring. Understanding and applying these techniques is crucial for maintaining high-quality data that supports reliable analytics and informed decision-making.

TOPIC 4

Lesson 1: Introduction to Data Warehousing

Objective:

Understand the fundamentals of data warehousing.

1. Definition of a Data Warehouse:

A data warehouse is a centralized repository designed to store large volumes of structured data from multiple sources. It's like a giant organized archive specifically built for analyzing information. This information helps with business intelligence (BI) activities, such as uncovering trends, generating reports, and making data-driven decisions.

Key Characteristics:

- **Optimized for Querying and Analysis:** Think of it like a giant filing cabinet specifically designed for easy searching and analysis.
- **Integrated:** Data from various sources is brought together and organized consistently.
- **Subject-Oriented:** Focused on specific business topics, like sales or customer data.
- **Non-Volatile:** Data is not deleted or overwritten, allowing you to track changes over time.
- **Time-Variant:** Includes historical data alongside current information for trend analysis.
- **Designed for Decision Support:** Provides insights to inform better business choices.

3. Benefits of Using Data Warehouses:

Improved Data Quality:

Data warehouses consolidate data from various sources, apply data cleansing processes, and ensure data consistency and accuracy.

Enhanced Performance:

Data warehouses are optimized for read-heavy operations, enabling fast query performance and reducing the load on transactional systems.

Historical Insights:

Storing historical data allows organizations to analyze trends, perform time-series analysis, and generate forecasts.

Scalability:

Data warehouses are designed to handle large volumes of data, making them suitable for growing data needs.

Advanced Analytics:

Supports complex queries, data mining, machine learning, and other advanced analytics techniques.

Summary:

In this lesson, we introduced the concept of data warehousing, explaining its definition, role in data engineering, and benefits. Data warehouses are essential for integrating data from multiple

sources, ensuring data quality, enhancing performance, providing historical insights, and supporting advanced analytics.

Lesson 2: Data Warehousing Architectures

Objective:

Learn about different data warehousing architectures.

Similarities and Key Differences: Databases vs. Data Warehouses

While both store data, databases and data warehouses have distinct purposes. Databases manage data for various day-to-day transactions, like an ATM system. Data warehouses, on the other hand, handle large-scale data analysis. They store historical information for complex analytical queries, unlike databases that focus on real-time data.

Data Warehouse Structure: Three Layers Working Together

A data warehouse typically has a three-tier architecture:

- **Bottom Tier:** The foundation, usually a relational database system. Data is cleansed, transformed, and loaded here using back-end tools.
- **Middle Tier:** The Middle Tier features an OLAP server that facilitates rapid query processing. Within this tier, three distinct types of OLAP models are utilized - ROLAP, MOLAP, and HOLAP. The type of database system that is being used will determine which OLAP model is best.
- **Top Tier:** The user interface, where you access information. It provides tools for querying, analyzing, reporting, and data mining.

How Data Warehouses Function: Integration and Analysis

Data warehouses bring together information from various sources into a single, unified database. Imagine combining customer data from sales systems, mailing lists, websites, and feedback forms. It can even hold confidential employee data. Businesses leverage this combined data to gain customer insights.

Data mining is a powerful feature of data warehouses. It involves sifting through vast amounts of data to discover hidden patterns and trends. Businesses can then use these insights to develop strategies that boost sales and profits.

Types of Data Warehouses: Different Solutions for Different Needs

There are three main types of data warehouses:

- **Enterprise Data Warehouse (EDW):** This centralized data hub provides crucial information for decision-making across the entire organization. It offers a unified view of data, allowing for cross-departmental analysis and complex queries.
- **Operational Data Store (ODS):** This warehouse refreshes data in real-time, making it suitable for operational tasks like managing employee records. It's used when data warehouses can't meet a company's reporting needs.
- **Data Mart:** This is a focused subset of a data warehouse, catering to a specific department, region, or business unit. Each department might have its own data mart to store relevant data. Information from data marts is periodically uploaded to the ODS, which then feeds it into the EDW for long-term storage and analysis.

SUMMARY:

Data warehouses, unlike databases, analyze historical data for trends. Their layered structure cleanses, stores, and analyzes data. Businesses can combine information from various sources and use data mining to find hidden patterns. There are different architectures for enterprise-wide insights, real-time tasks, and specific departments. Understanding these helps businesses choose the right tool to unlock data's potential.

Lesson 3: Data Warehouse Modeling

Objective: By the end of this lesson, you will understand the fundamental concepts of data warehouse modeling, including dimensions, facts, and common schema types. You'll learn how to design effective data structures for analysis and reporting.

1. Introduction to Dimensional Modeling

Dimensional modeling is a design technique specifically tailored for data warehouses. Its primary goal is to present data in a standard, intuitive framework that allows for high-performance access. Unlike normalized data models used in transactional systems, dimensional models are optimized for data retrieval and analysis.

The core principle of dimensional modeling is to structure data around business processes and their measurements. This approach results in a model that's both easy for business users to understand and efficient for query performance.

2. Facts: The Measure of Business Processes

Facts are the quantitative or measurable data in a business process. They represent what we're analyzing or measuring.

Key Characteristics of Facts:

- Numeric values that can be aggregated (summed, averaged, etc.)

- Represent business measurements or metrics
- Often additive across multiple dimensions

Examples of Facts:

- Sales amount
- Quantity sold
- Profit
- Number of website visits
- Customer service call duration

Facts are typically stored in fact tables, which form the center of our dimensional model.

3. Dimensions: The Context of Business Processes

Dimensions provide the context to facts. They are the "who, what, where, when, why, and how" of a business process.

Key Characteristics of Dimensions:

- Descriptive attributes that allow for filtering and grouping
- Provide the context for analyzing facts
- Often hierarchical in nature

Examples of Dimensions:

- Date (with hierarchies like Year > Quarter > Month > Day)
- Product (with hierarchies like Category > Subcategory > Product)
- Customer (with attributes like Name, Age, Location)
- Store Location (with hierarchies like Country > State > City)

Dimensions are stored in dimension tables, which surround and connect to the fact table.

4. Star Schema: The Fundamental Dimensional Model

The Star Schema is the most basic and widely used dimensional model.

Structure:

- A central fact table connected directly to multiple dimension tables
- Resembles a star shape, hence the name

Characteristics:

- Simple and intuitive design
- Faster query performance due to fewer joins

- Dimension tables are denormalized (redundant data allowed)

Example of a Star Schema for a Sales data warehouse: [Insert visual representation of a star schema with a central "Sales Fact" table connected to "Date", "Product", "Customer", and "Store" dimension tables]

5. Snowflake Schema: A Normalized Variation

The Snowflake Schema is a variation of the star schema where dimension tables are normalized into multiple related tables.

Structure:

- Similar to star schema, but dimension tables are split into multiple tables
- Resembles a snowflake shape

Characteristics:

- Reduces data redundancy through normalization
- More complex queries due to additional joins
- Can be more difficult to understand and maintain

Example of a Snowflake Schema for the same Sales data warehouse: [Insert visual representation of a snowflake schema, expanding on the star schema by normalizing dimensions like "Product" into separate "Category" and "Subcategory" tables]

6. Choosing Between Star and Snowflake Schemas

The choice between star and snowflake schemas depends on various factors:

Star Schema Benefits:

- Simpler queries and faster performance
- Easier for business users to understand
- Better suited for OLAP cubes

Snowflake Schema Benefits:

- Saves storage space through normalization
- Easier to maintain dimension hierarchies
- Better data integrity due to normalization

In practice, many data warehouses use a hybrid approach, snowflaking only the dimensions where it provides significant benefits.

Summary:

Dimensional modeling is a crucial skill for effective data warehouse design. By understanding facts, dimensions, and schema types, you can create data structures that balance performance, usability, and maintainability. The choice between star and snowflake schemas ensures that your data warehouse can provide accurate, historical analysis to support business decision-making.

TOPIC 5

Conceptualizing the Data pipeline process, with a study case of a “something” industry. From Data Ingestion, To Pass Data Quality check, using ETL tools. No deep dive yet, just a demo of how it works, practices.

Will deep dive in the next course which is a comprehensive study case and implementation and understand the tools.