### **Description**

These notes summarize the key concepts from an introductory video on Azure Data Engineering. They cover the prerequisites for learning, the core responsibilities of a data engineer illustrated through a practical example, the fundamental differences between batch and streaming data processing, and an overview of the typical Business Intelligence (BI) workflow.

### **1. Course & Exam (DP-203) Overview**

* **Primary Goal:** To help you become an Azure Data Engineer and prepare for the **DP-203 exam**.
* **Prerequisites:**
  + You need hands-on experience; watching alone is not enough. A **free Azure trial subscription** is sufficient to practice.
  + You are expected to have **basic knowledge of Azure fundamentals**.
* **Key Advice:** **Take notes!** Many Azure services have similar names (e.g., Data Lake, Data Factory, Data Bricks), which can be confusing.
* **Exam Scope:**
  + The official skills outline is available on the Microsoft Learn website for "DP-203".
  + **Important:** As of the video recording, **Microsoft Fabric is NOT** part of the DP-203 exam.

### **2. What is Data Engineering? A Real-World Example**

The speaker automated the process of checking his wife's book sales, which highlights common data engineering challenges.

* **The Problem:** The publisher's website showed total sales but had no API for easy data access. The goal was to automate checking for new sales and provide analysis.
* **Challenges & Solutions (The Core of Data Engineering):**
  + **Data Ingestion:**
    - **Challenge:** No API available.
    - **Solution:** Scraped the website by simulating a login and parsing the HTML content of the sales table.
  + **Data Transformation:**
    - **Challenge 1: Aggregated Data.** The site showed lifetime sales, not daily sales.
    - **Solution 1:** Calculated daily sales by subtracting yesterday's total from today's total: Salesdaily​=Totaltoday​−Totalyesterday​.
    - **Challenge 2: Non-Atomic Data.** Single columns contained multiple pieces of information (e.g., one column had "Sales Channel" and "Last Sale Date" together).
    - **Solution 2:** Parsed single columns into multiple, separate columns (e.g., [Book Title, Format, Type] -> [Book Title], [Format], [Type]) to enable proper filtering and analysis.
  + **Handling Multiple Data Sources:**
    - **Challenge:** Data existed in different locations and formats.
    - **Solution:** Integrated data from three distinct sources:
      1. The publisher's website (scraped data).
      2. A historical sales Excel file (different structure).
      3. Facebook Marketing API (to correlate ad clicks with sales).
  + **Data Volatility:**
    - **Challenge:** The main data source (the sales table on the website) would periodically reset to zero after an invoice was paid.
    - **Solution:** The system had to be designed to detect and handle this reset to avoid incorrect calculations. This shows that a "data source" can sometimes be a human-driven process.
* **Final Output:**
  + An automated email notification for new sales.
  + Power BI reports for visual analysis.
  + A common request from end-users: the ability to export data to Excel.

### **3. The Basic BI (Business Intelligence) Flow**

This is the typical lifecycle of data in a BI solution, which this course will follow.

#### **Batch vs. Streaming Processing**

* **Batch Processing:** Data is processed in large chunks (batches) on a schedule (e.g., once a day, every hour). This is common for generating daily reports.
* **Streaming Processing:** Data is processed continuously as it arrives, in near real-time. This is used for scenarios like IoT sensor data or hospital heart-rate monitors.
* The video (and this course) will first focus on **batch processing**.

#### **Stages of a Batch BI Solution**

1. **Data Sources:** The starting point. Can be anything from files (CSV, Excel), databases (SQL, Oracle), or APIs (Facebook).
2. **Ingest / Extract:** The process of getting data from the source and bringing it into your system.
3. **Store:** Saving the raw, ingested data in a flexible storage layer.
4. **Transform:** The "heavy lifting" of data engineering. This includes cleaning, parsing, joining datasets, and enriching the data to make it useful.
5. **Model & Serve:** Structuring the transformed data into a clean, optimized model (like a data warehouse) and making it available for reporting tools.
6. **Reports:** The final visualization layer (charts, KPIs, tables) that end-users interact with.

### **4. Roles & Responsibilities**

* **Data Engineer:** Responsible for the entire pipeline from **Data Sources** through **Model & Serve**. Their job is to build a reliable and efficient system that provides clean data.
* **Data Analyst:** Primarily responsible for the **Reports** stage. They take the data provided by the engineer and use it to build reports and find insights. (Note: In the real world, these roles often overlap).