a)

Data Extraction:

When extracting data from various sources, we have several technological options:

1. For data stored in local or remote databases (e.g., MariaDB, GCP Cloud SQL), we can use structured query language (SQL) like MySQL, tailored to the specific database requirements.
2. Smaller datasets stored in files (e.g., .json, .xls) can be efficiently extracted using Python with libraries like Pandas or other programming languages.
3. Custom scripts are necessary for extracting data from scratch. In scenarios like real-time data extraction, cloud-based services such as GCP Pub/Sub, Amazon Kinesis, or Apache Kafka are viable options.

Data Transformation:

Data transformation aims to clean, enrich, and structure the data based on predefined business rules. In our context, rules may involve data validation (e.g., protection against injection attacks), cleansing (e.g., removing extra spaces), and enrichment for more insightful information. We also handle missing values and convert data to the required types.

Technologies for data transformation include Python (utilizing the Pandas library) or R to apply business rules. SQL queries are employed for updating data and exporting results in various file formats. Apache Camel and Apache NiFi provide versatile options for data transformation, while AWS Glue allows for the definition of Extract, Transform, and Load (ETL) processes.

The selection of technologies depends on the existing systems and standards within the company.

Data Loading:

When loading data, considerations involve determining the target system and the volume of data. Options range from local relational databases to high-scale databases like Google BigQuery, especially when data analysis and visualization using tools like Looker dashboards are required.

For smaller datasets, JSON, XML, or other text formats may suffice. ETL tools such as Apache NiFi, Talend, Informatica, and Microsoft SSIS offer visual interfaces for designing data workflows.

In the ABN environment, a tailored approach is necessary, and depending on the task, a combination of the aforementioned technologies may be required. Conducting a proof of concept can aid in assessing the efficiency and suitability of the selected technologies.

b)
Structured Data:

Structured data follows a well-defined, organized format and is commonly stored in databases with schemas. For this type of data, the technical specifications I would recommend include:

1. Relational Database Management Systems (RDBMS):
   - Utilize RDBMS systems such as MySQL for storing and managing structured data.
2. SQL:
   - Employ SQL for querying and extracting data from the database.
3. ETL Tools:
   - Implement ETL (Extract, Transform, Load) tools like Apache NiFi for seamless integration and transformation processes.

Semi-Structured Data:

Semi-structured data retains a level of organization, often in the form of key-value pairs. For handling semi-structured data, consider the following:

1. JSON or XML Parsing Libraries:
   - Use libraries specifically designed for parsing JSON or XML to extract and manipulate semi-structured data.
2. Data Storage and Processing:
   - Consider using file formats like Parquet, known for efficient data compression and encoding schemes, or Apache Avro, providing data serialization and exchange services for Apache projects.

Unstructured Data:

Unstructured data lacks a predefined data model or format and may include text, images, videos, or audio files. Dealing with unstructured data involves advanced techniques and storage solutions:

1. Machine Learning Algorithms:
   - Apply machine learning algorithms for tasks like feature extraction and content analysis, particularly for text, popular Natural Language Processing (NLP) techniques can be utilized.
2. Libraries for Multimedia Data:
   - Utilize libraries such as OpenCV for handling multimedia data.
3. Object Storage Systems:
   - Store unstructured data in object storage systems. For example, Azure Blob Storage in Microsoft Azure allows users to store large amounts of unstructured data efficiently.

In summary, the approach to handling structured, semi-structured, and unstructured data involves utilizing specific technologies and tools tailored to the characteristics of each data type. This ensures effective processing, storage, and analysis based on the nature of the data.

c)
Batch Processing

The method computers use to periodically complete high-volume, repetitive data jobs. Those include tasks such as backups, filtering, and sorting which can be computationally intensive and inefficient to run on individual data transactions.
To perform such tasks, the following technologies can be used:
Apache Hadoop MapReduce: which enables massive scalability across many servers for Hadoop clusters. Then, there is Apache Flink which enables stream processing with batch support.

Real-Time Processing

Real-time processing is a method of processing data at a near-instant rate, requiring a constant flow of data intake and output to maintain real-time insights.
Examples of such tools that can be used are firstly Apache Kafka Streams, enabling real-time stream processing. Secondly, there can be Apache Storm which can distribute stream processing.

Scalability

Scalable is a system that can grow to accommodate growth, modifications, or new demands. The methods, equipment, and strategies known as scaling allow an app to succeed. Scaling is a crucial part of developing distributed systems. Doing this distributes the work among many workers and processing units. Workers spread tasks among several processors or computers.

To enable scalability, the following two methodologies can be useful

1. Horizontal Scaling: Using tools working with containers such as Kubernets and Docker, as well as, auto-scaling groups in cloud platforms (AWS Auto Scaling)
2. Cloud Services: These can leverage cloud-native services (AWS Lambda and GCP).