1. A **graph** is a mathematical structure that represents relationships between objects. It consists of **nodes** (vertices) and **edges** (connections).
2. A **knowledge graph** is a specialized type of graph used as a database, where information is stored as **nodes** (entities) and **relationships** (edges).
3. Both nodes and relationships can have **properties** (key–value attributes).
4. Nodes may carry one or more **labels** to group them by type.
5. Relationships always have a **type** and a **direction**, specifying how two nodes are connected.

Node Centrality Analysis

Node centrality measures the **importance** or **influence** of a node within a graph. It helps identify nodes that play central roles in the structure and flow of information. Some of the most common centrality measures are:

1. **Degree Centrality**
    - Counts the number of edges connected to a node.
    - Nodes with higher degree centrality are more connected and often more influential locally.
2. **Betweenness Centrality**
    - Measures how often a node lies on the shortest path between other nodes.
    - Nodes with high betweenness centrality act as **bridges** or **gateways** that control the flow of information between different parts of the graph.
3. **Closeness Centrality**
    - Quantifies how quickly a node can reach all other nodes in the graph.
    - Nodes with higher closeness centrality can **communicate more efficiently**, since they are closer (on average) to all others.

Graph Embeddings

Graph embeddings are **mathematical representations** of nodes, edges, or entire subgraphs in a **continuous vector space**. They capture both the **structural** and **relational** information of the graph, making complex graph data easier to work with.

1. **Representation**
   - Each node (or edge) is mapped to a vector of real numbers.
   - Similar nodes/edges in the graph tend to have similar vector representations.
2. **Captured Information**
   - **Structural features**: how a node is positioned relative to others.
   - **Relational features**: the types and strengths of connections.
3. **Applications**
   - **Node similarity**: identify nodes that play similar roles in the graph.
   - **Link prediction**: infer potential new relationships between nodes.
   - **Visualization**: reduce high-dimensional graph data into 2D or 3D space for easier interpretation.
   - **Downstream ML tasks**: embeddings can serve as input features for classification, clustering, or recommendation systems.