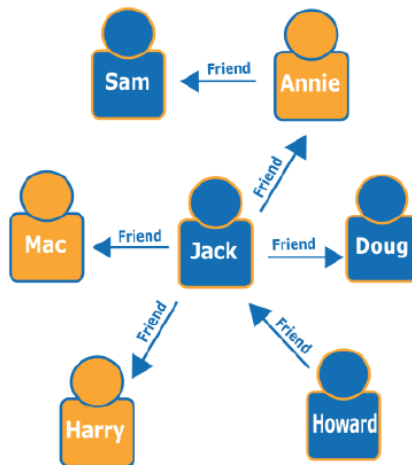


| | |
|-----------------------|--------------|
| Name | Divyesh Shah |
| UID no. | 2020300060 |
| Experiment No. | 9 |

| | |
|----------------------------|---|
| AIM: | Study GraphDB and implement CRUD operations on a Neo4J DB |
| GraphDB | |
| PROBLEM STATEMENT : | Create Neo4J database for a scenario displaying the relation between all people using a particular social media application. The database depicts the follower and following relation between all users. |
| Theory : | <p>What are Graph Databases?</p> <p>Graph databases are purpose-built to store and navigate relationships. Relationships are first-class citizens in graph databases, and most of the value of graph databases is derived from these relationships. Graph databases use nodes to store data entities, and edges to store relationships between entities. An edge always has a start node, end node, type, and direction, and an edge can describe parent-child relationships, actions, ownership, and the like. There is no limit to the number and kind of relationships a node can have.</p> <p>A graph in a graph database can be traversed along specific edge types or across the entire graph. In graph databases, traversing the joins or relationships is very fast because the relationships between nodes are not calculated at query times but are persisted in the database. Graph databases have advantages for use cases such as social networking, recommendation engines, and fraud detection, when you need to create relationships between data and quickly query these relationships.</p> <p>The following graph shows an example of a social network graph. Given the people (nodes) and their relationships (edges), you can find out who the "friends of friends" of a particular person are—for example, the friends of Howard's friends.</p> |



A Graph Database is essentially a type of Database Engine that models both Nodes and Edges as first-class entities within a relational Graph. This enables you to represent intricate interactions between your data in a more natural and realistic way. Graph Databases are frequently schema-less, which provides the flexibility of a Document or Key/Value Store database while supporting Relationships like a traditional Relational Database. However, this doesn't mean that there is no data model linked to the database. It just means that there is more room for defining it, which can speed up your projects. Although it's possible to achieve the same level of expressiveness in other database solutions, it may not be as elegant as in a Graph Database and may require link tables or nested documents. Graph Databases also enable the application of Graph Theory to your data in an efficient manner, allowing you to uncover connections that would be difficult to see otherwise. For instance, you can find minimal routes between nodes or disjoint sets within your data.

Need of Graph Databases

1. It solves Many-To-Many relationship problems.

If we have friends of friends and stuff like that, these are many to many relationships. Used when the query in the relational database is very complex.


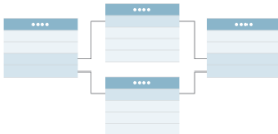
2. When relationships between data elements are more important.

For example- there is a profile and the profile has some specific information in it but the major selling point is the relationship between these different profiles that is how you get connected within a network. In the same way, if there is data element such as user data element inside a graph database there

could be multiple user data elements but the relationship is what is going to be the factor for all these data elements which are stored inside the graph database.

3. Low latency with large scale data.

When you add lots of relationships in the relational database, the data sets are going to be huge and when you query it, the complexity is going to be more complex and it is going to be more than a usual time. However, in graph database, it is specifically designed for this particular purpose and one can query relationship with ease.

| | Graph database | Relational database |
|-----------------|--|--|
| FORMAT | Nodes and edges | Tables with rows and columns |
| RELATIONSHIPS | Considered data, represented by edges between nodes | Related across tables, established using foreign keys between tables |
| COMPLEX QUERIES | Run quickly and do not require joins | Require complex joins between tables |
| TOP USE CASES | Relationship-heavy use cases, including fraud detection and recommendation engines | Transaction-focused use cases, including online transactions and accounting |
| EXAMPLE |  |  |

IMPLEMENTATION

1. Install Neo4j for desktop

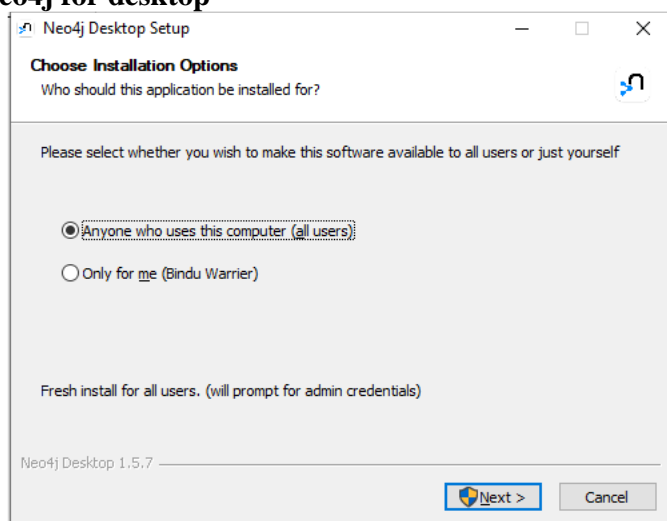


Figure 1: Install Neo4J DB

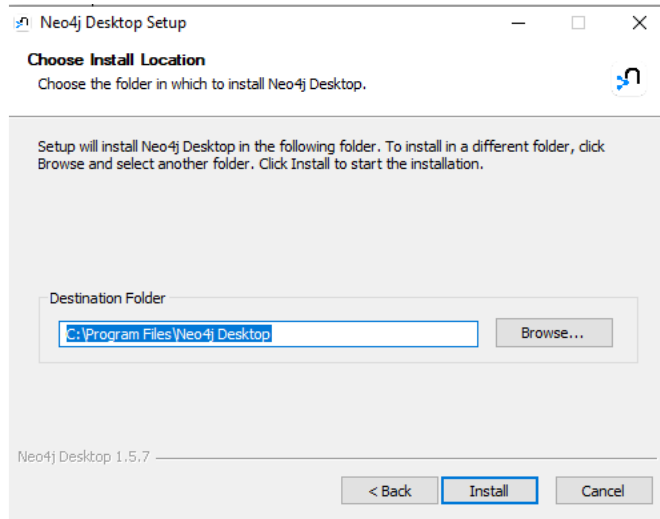


Figure 2: Provide directory for installation.

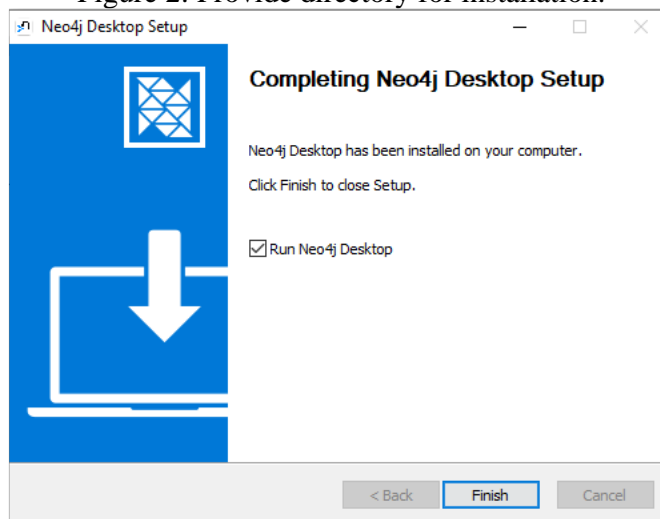
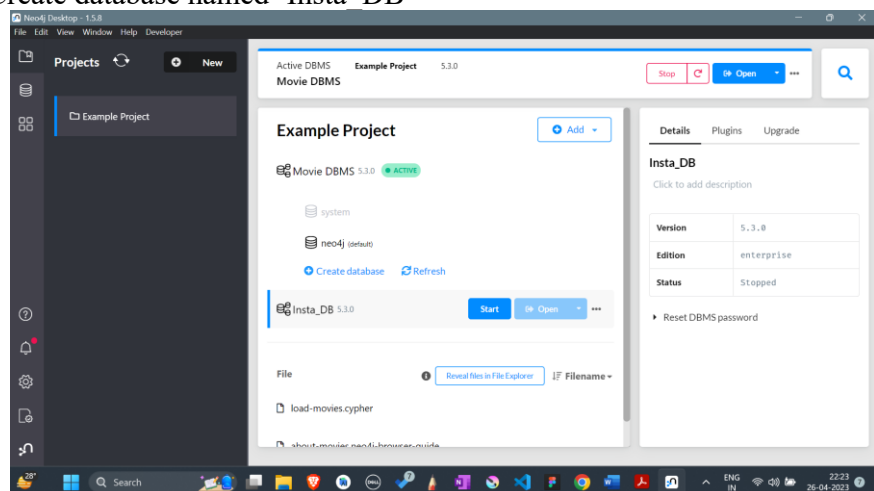
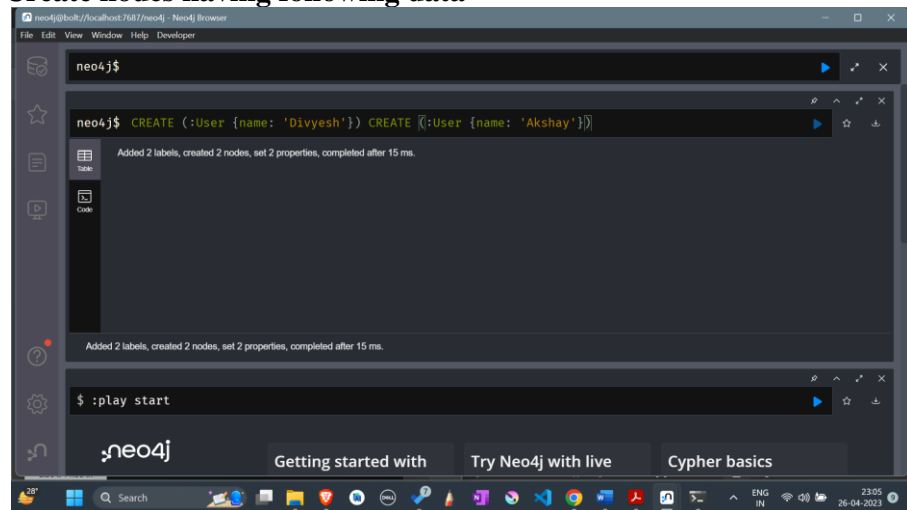


Figure 3: Run Neo4J Desktop

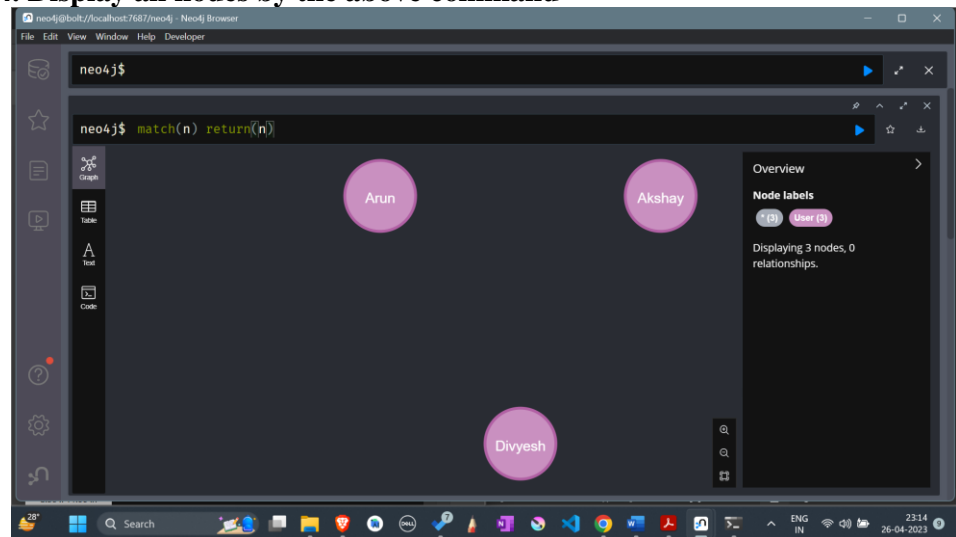
2. Create database named 'Insta DB'



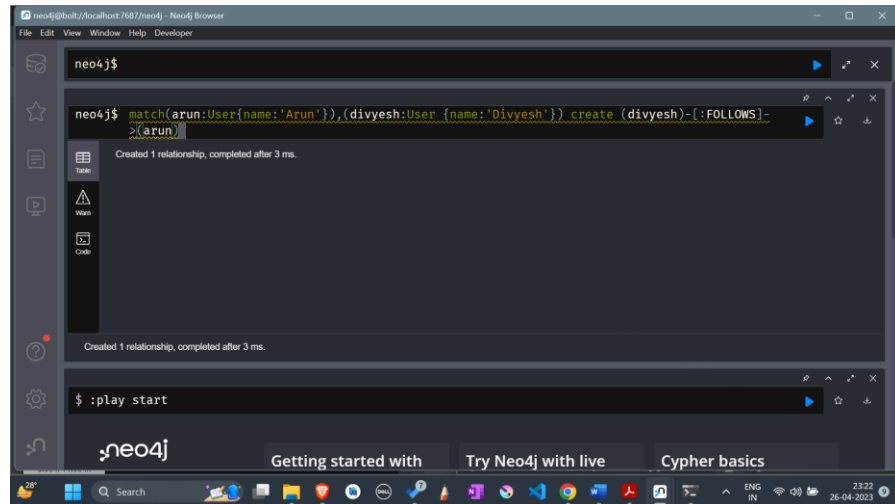
3. Create nodes having following data



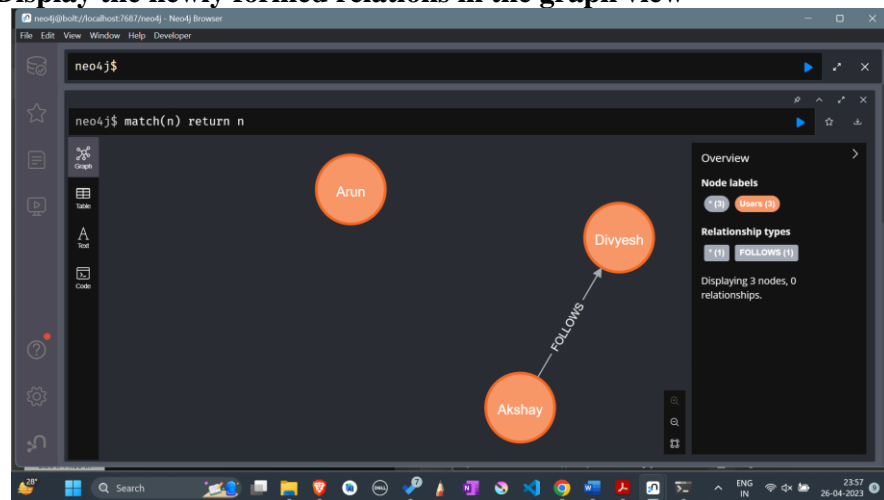
4. Display all nodes by the above command



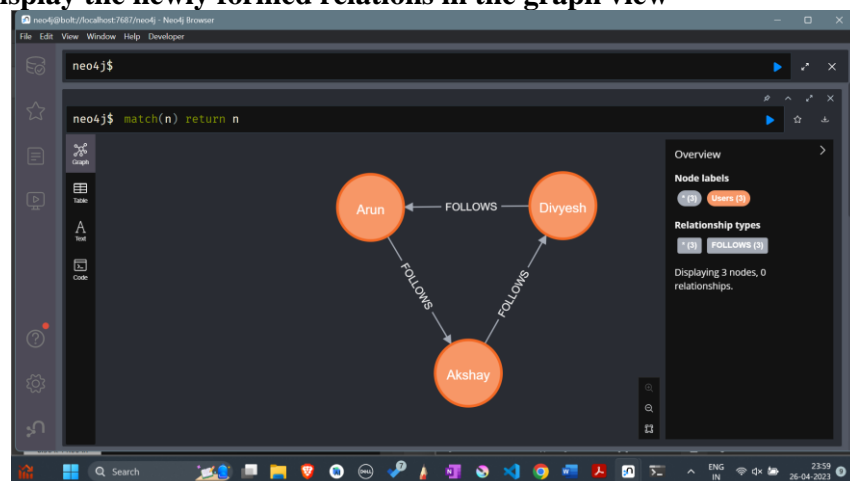
5. Create relation between alice and bob where alice is following bob



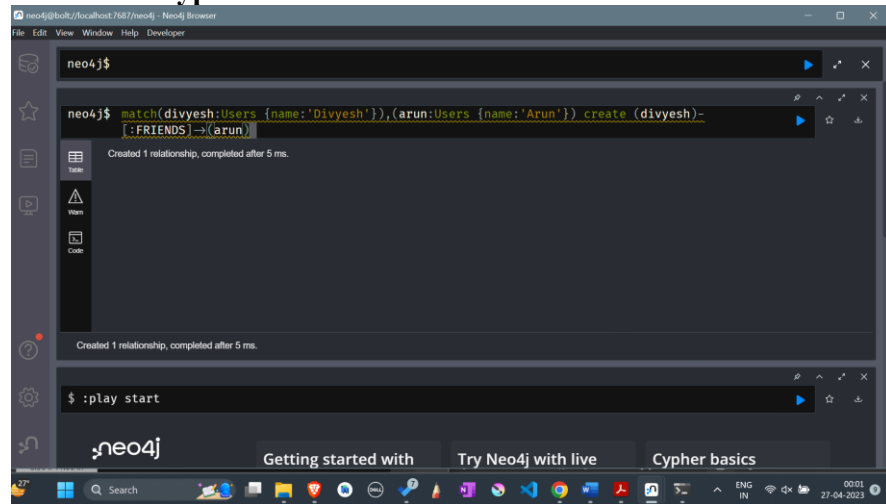
8. Display the newly formed relations in the graph view



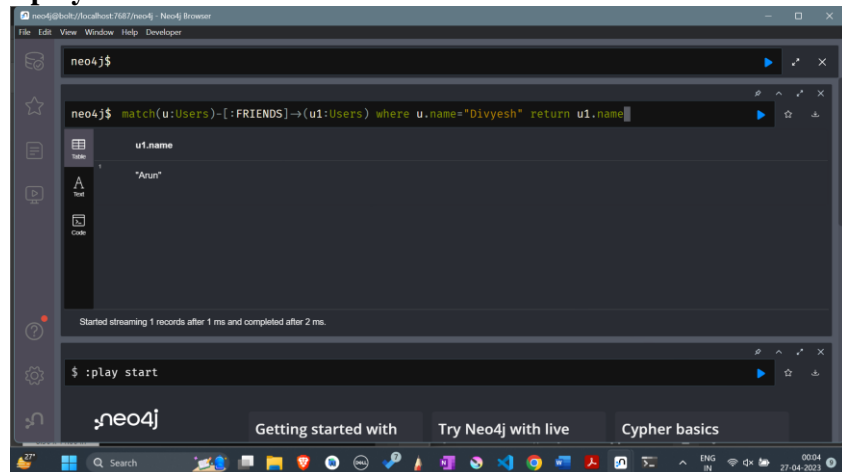
9. Display the newly formed relations in the graph view



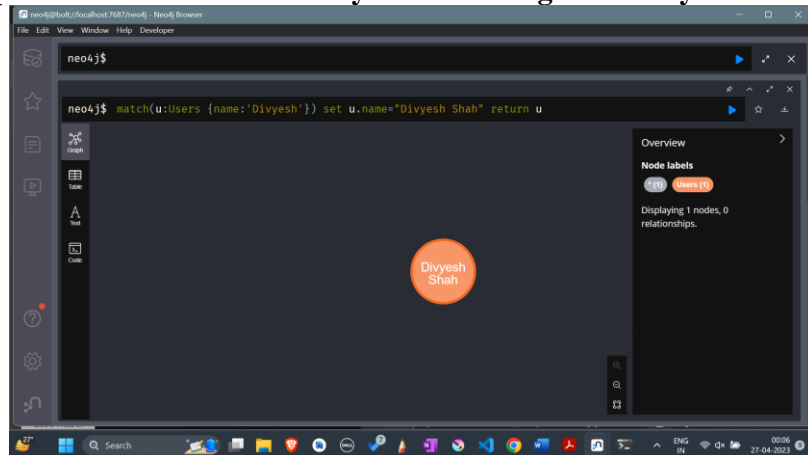
10. Create new type of relation called 'Friends'



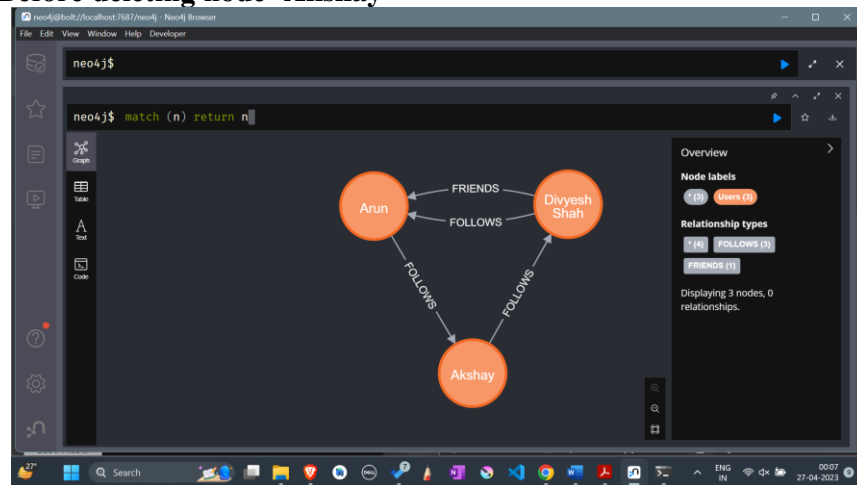
11. Display the users who are friends of Alice



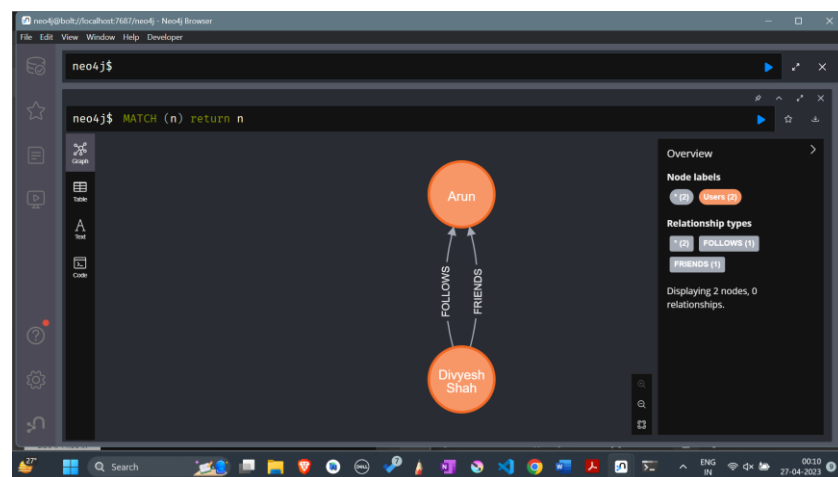
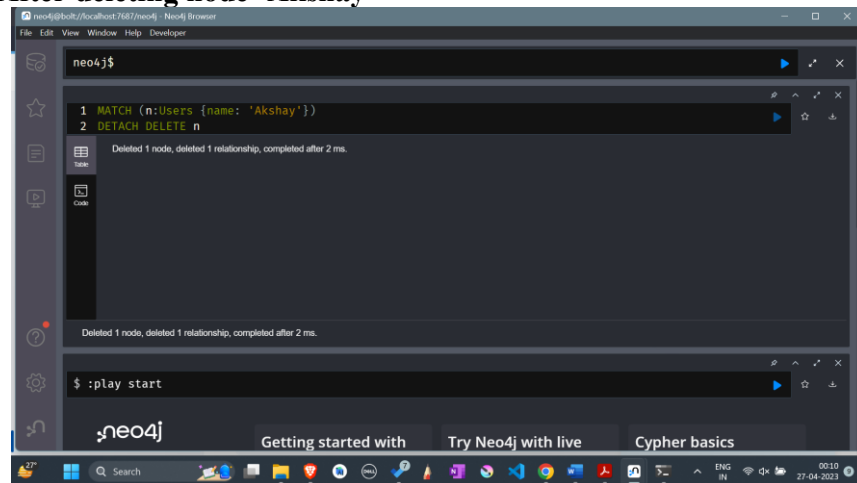
12. Update the name of node Divyesh and change it to Divyesh Shah



13. Before deleting node 'Akshay'



14. After deleting node 'Akshay'



Conclusion:

1. To sum up, I gained knowledge about the practical uses of Graph Databases, their superiority over conventional SQL databases, and when GraphDB is applicable.
2. I carried out CRUD operations and gained hands-on experience on working with a Graph database. I executed multiple queries in Cypher Query language and recorded the corresponding results in the report.
3. This project provided me with valuable insights into a new realm of databases and equipped me with the skills to work with Graph databases for real-time applications.

References:

1. <https://www.techtarget.com/searchdatamanagement/feature/Graph-database-vs-relational-database-Key-differences>
2. <https://neo4j.com/developer/graph-database/>
3. <https://www.geeksforgeeks.org/introduction-to-graph-databases/>
4. <https://www.compose.com/articles/introduction-to-graph-databases/>