**Student Name – Aritha Jayaratne.**

**Student ID – s4683873**

**Subject – Data mining & warehousing**

**Subject code – BCO7004**

**Lecturer – Dr Mahabub Ahmed**

**Possible relationships based on the ER diagram.**

**Company can have many Courses;** Many courses can be offered by a company. On the "Courses" side, a crow's foot represents the relationship. This indicates that the cardinality is 1:N, or one-to-many. A course is owned by a single company, even though a company may have many courses.

**Course can have many Qualifications:** A course may call for a variety of credentials. It's a many-to-many (M:N) relationship. A course may require more than one qualification, and multiple courses may require the same qualification.

**Candidate can have many Qualifications:** A candidate may possess a variety of Qualifications. It's a many-to-many (M:N) relationship. A candidate may possess multiple qualifications, and multiple candidates may hold the same qualification.

**Company can have many Candidates:** A company may receive many job applications. On the "Candidates" side, a crow's foot represents the relationship. This indicates that the cardinality is 1:N, or one-to-many. A candidate may apply to one or more companies, but they may only apply to one at a time.

**Company can have many Openings:** A company may have numerous job vacancies. On the "Openings" side, a crow's foot represents the relationship. This indicates that the cardinality is 1:N, or one-to-many. An opening can belong to one company, but a company can have many openings.

**Candidate can apply for many Openings (Placement):** A candidate is able to apply to multiple positions. On the "Placement" side, a crow's foot represents the relationship. This indicates that the cardinality is 1:N, or one-to-many. A placement, or application, is specific to a single candidate and one opening, although a candidate may apply for multiple openings.

**Mandatory and optional dependencies between the relationships described.**

Company - Courses:  
  
Mandatory dependency on courses (Each company need to have a job to be included in the database).  
Dependency of Courses on Company is optional; a company may or may not offer a course.  
  
Course - Qualifications:  
  
Dependency that is optional on both ends (a qualification may or may not be required for a course, and a course may or may not require a qualification).  
  
Candidate - Qualifications:  
  
Dependency that is optional for both parties (a candidate may or may not have qualifications, and candidates may or may not have qualifications).

Company - Candidates:  
  
Companies are required to rely on candidates (at least one candidate must apply for each company).  
Candidates' optional reliance on the company (a candidate is free to apply to a company or not).

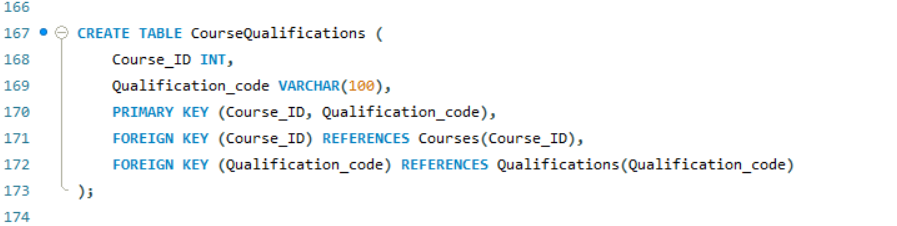
Company - Openings:   
  
Dependency of Company on Openings is required; every company needs to have at least one opening.  
Dependency of Openings on Company is optional; an opening may or may not be a part of a company.   
  
Candidate - Placement:   
  
Dependency of Candidates on placements  is Required (applicants must apply for a minimum of one opening).   
Dependency of Candidates on placements is required (at least one candidate must apply for each opening).

Resolving the Many to Many relationships Using Associative tables.

To resolve the many to many relationships, need to introduce associative tables to these relationships.

Course – Qualifications:

Create a new table named Coursequalification. This table has two foreign keys referencing the other two tables, which are Courses and Qualifications.



Candidate – Qualifications:

Create a table named CandidateQualifications to resolve many to many relationship between the two tables.

A computer screen shot of a code

Description automatically generated

Normalization concepts implemented on the designed database.

According to the Normalization concepts the database model the database checks all three of the normalization models. The following steps were taken, All the many to many relationships were normalized by adding new associative tables. And by adding primary keys to the tables to uniquely identify a column for each table.

**Scenario.**

In this scenario the current table is created to get the candidates who are eligible for job openings based on their qualification and job history.

The query selects distinct candidate IDs, candidate names, opening numbers and required qualifications. After selecting the distinct information created multiple JOIN operations to connect the Candidate, JobHistory, Placement, Opening, Courses, Course Qualifications, and Qualifications tables based on their respective relationships. Lastly the query adds the conditions to filter out the candidates based on their job history job end date to check if they are currently employed, and the condition will check if they have worked the sufficient hours to be eligible for the job openings. (Code is provided in the SQL script).