**Table Shorthand:**

Users (UserID, username, password, date\_created, user\_type, user\_fname, user\_lname, user\_email)

Applicants (UserID, applicant\_address, applicant\_city, applicant\_state, applicant\_zip, applicant\_contact\_number, formID)

Managers (UserID, manager\_office\_phone, departmentID)

Forms (formID, highest\_degree\_earned, UserID)

Applicant Work History (formID, job\_title, start\_date, end\_date, description\_of\_duties, reason\_for\_leaving)

Applicant Education (formID, degree\_earned, school)

Additional Information (UserID, JobID, reason\_for\_applying, reference\_letter, date\_applied)

Job Posting (JobID, job\_description, departmentID, required\_experience, preferred\_experience, employment\_start\_date, salary, beginner\_review\_date, closing\_date)

Departments (DepartmentID, department\_location, department\_name, UserID)

**Normalization:**

The main need for normalization in this database came from the Forms table that applicants filled out. Being that is called for several different multivalued attributes (having multiple kinds of degrees or having multiple previous jobs), these had to be converted into their own tables with composite keys to accommodate. Now, each form will clearly identify each of the possible multiple degrees, as well as allow the database to track every previous job submitted by an applicant including dates and descriptions of each position. As for the rest, the only other composite key is a bridge entity connecting the M:N relationship between Applicants and Job Postings, eliminating any possibility of partial dependencies. There is a case to be made for a transitive property existing for applicants’ addresses, and we could certainly create another table specifically to hold address information, including state and zip codes. However, as we are only tracking address data for applicants, this step shouldn’t be necessary.

**Business Rules:**

It is assumed there will only be one manager for each department, with each manager only managing one department (1:1 relationship), and that manager will be the only one submitting job postings under his or her respective department. In addition, applicants will likewise only submit one form, with each form associated with a single applicant (also 1:1). As there are only 6 departments, there should also only ever be 6 managers in this database, as it is only concerned with the job posting aspect of their role; if a manager is moved from the position, he or she should be removed from the database and replaced with the new manager.

**SQL Commands**

* SELECT UsersID, user\_fname, user\_lname, applicant\_contact\_number, jobID, job\_description FROM `job postings` JOIN `additional information` ON `job postings`.JobID = `additional information`.`job postings\_jobID` JOIN applicants ON `additional information`.`Applicants\_Users\_UsersID` = applicants.Users\_UsersID JOIN users ON applicants.users\_usersID = users.usersID

WHERE jobID = 4; ***(jobID can be adjusted to change what job to search for)***

* SELECT \* FROM job\_search.`additional information` WHERE `date\_applied` > '2020-03-14';
* SELECT JobID, job\_description FROM `job postings` WHERE closing\_date > curdate() AND Departments\_departmentID = 3;
* SELECT jobID, job\_description, closing\_date, user\_fname, user\_lname FROM `job postings` JOIN `additional information` ON `job postings`.JobID = `additional information`.`job postings\_jobID` JOIN applicants ON `additional information`.`Applicants\_Users\_UsersID` = applicants.Users\_UsersID JOIN users ON applicants.users\_usersID = users.usersID

WHERE usersID = 2; ***(usersID here can be adjusted to change which applicant is being searched***  ***for)***

* SELECT jobID, job\_description, closing\_date, user\_fname, user\_lname FROM `job postings` JOIN departments ON `job postings`.departments\_departmentID = departments.departmentID JOIN managers ON departments.managers\_users\_usersID = managers.Users\_UsersID JOIN users ON managers.users\_usersID = users.usersID WHERE curdate() < closing\_date;