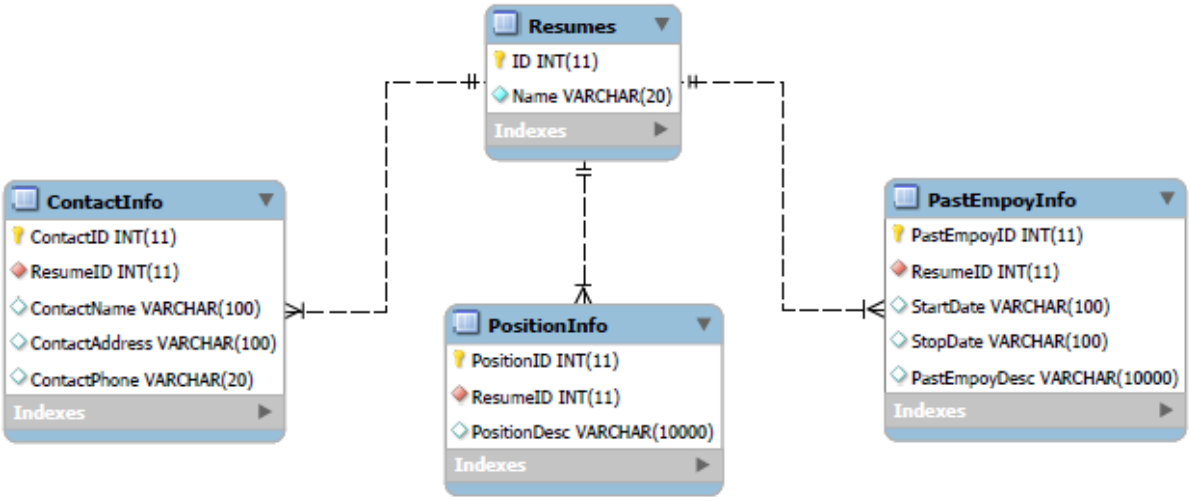
Jakub Szpunar
CS4540
PS4



Above we see the database datagram I prepared for PS4.

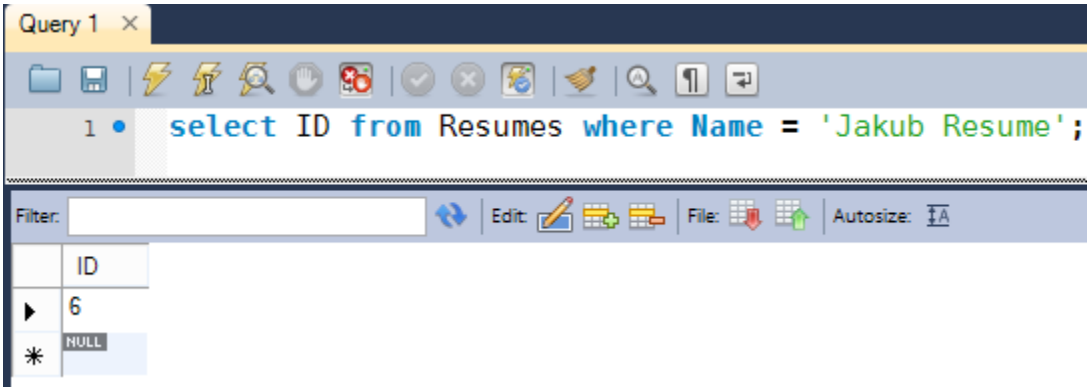
The Resume table has an ID and a Name. The ID is a primary key, non-null, unique, auto-incremented, and an integer. The ID is used to join data from multiple tables together. The Name field stores the Name of the resume. It is unique, non-null, and a varchar(20). This limits the length of the name to 20 characters and forces a unique name to be included with each resume.

The ContactInfo table has a ContactID, ResumeID, ContactName, ContactAddress, and ContactPhone. The contactID has the same properties as the ID from the resume table; however, the contactID is not reference in any other tables. The ResumeID is unique, non-null, and an integer. The table has a foreign key constraint that forces the ResumeID to match an ID from the Resume table. The key is set up to cascade with any changes in the Resume table. That is, if a resume is removed from the resume table, any contact information matching the resume will also be removed automatically. The ContactName, ContactAddress, and ContactPhone fields are all varchar fields. They may be null in order to allow for incomplete data to be stored.

The PositionInfo and PastEmployInfo tables function nearly identically to the ContactInfo table. An ID field identifies each row uniquely. ResumeID is constrained to be set to an ID from the Resumes table. Similarly cascading is used for these foreign keys. In the PastEmpoyInfo, a varchar field instead of datetime was selected for the date columns in order to ease storing dates that have no formatting requirements in the php program.

Store command in website:

First check to see if a resume of that name exists:

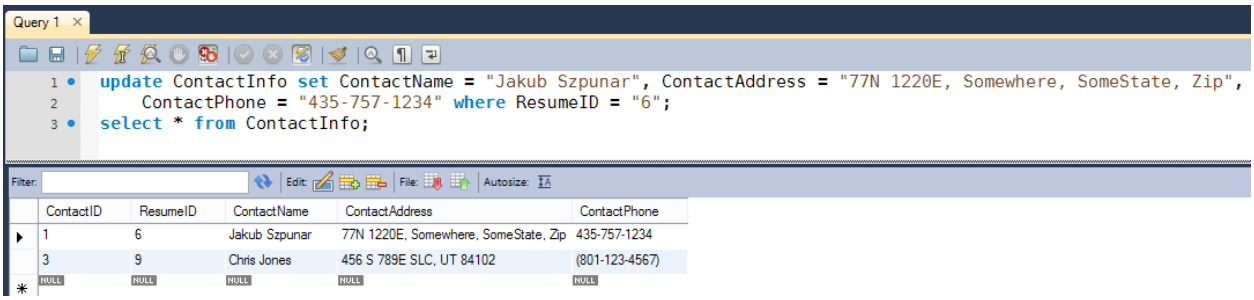


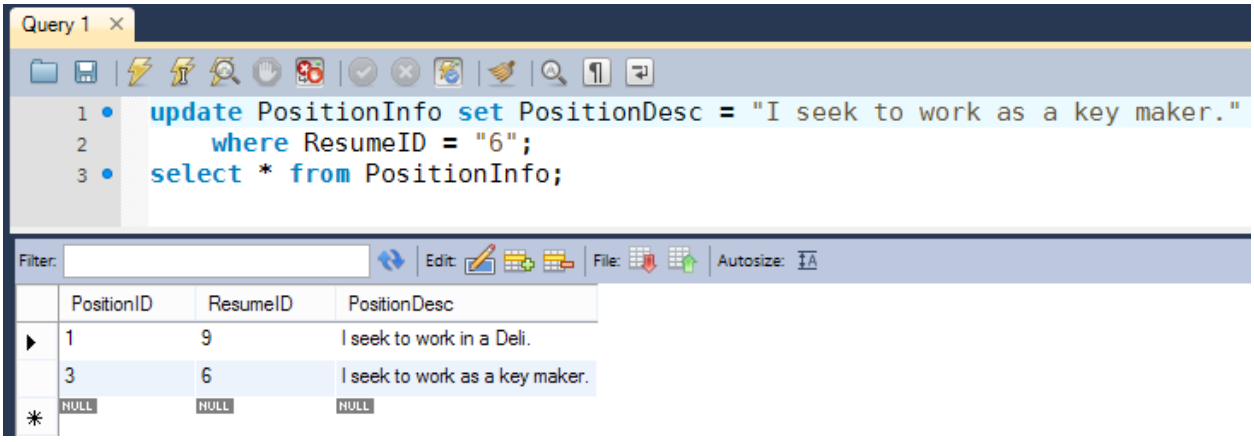
The name in quotes would of course be the entered name from the website. If an ID comes back, we know a resume of that name exists and we will update that resume. If the name does not exist, we will create a new resume.

If the resume existed:

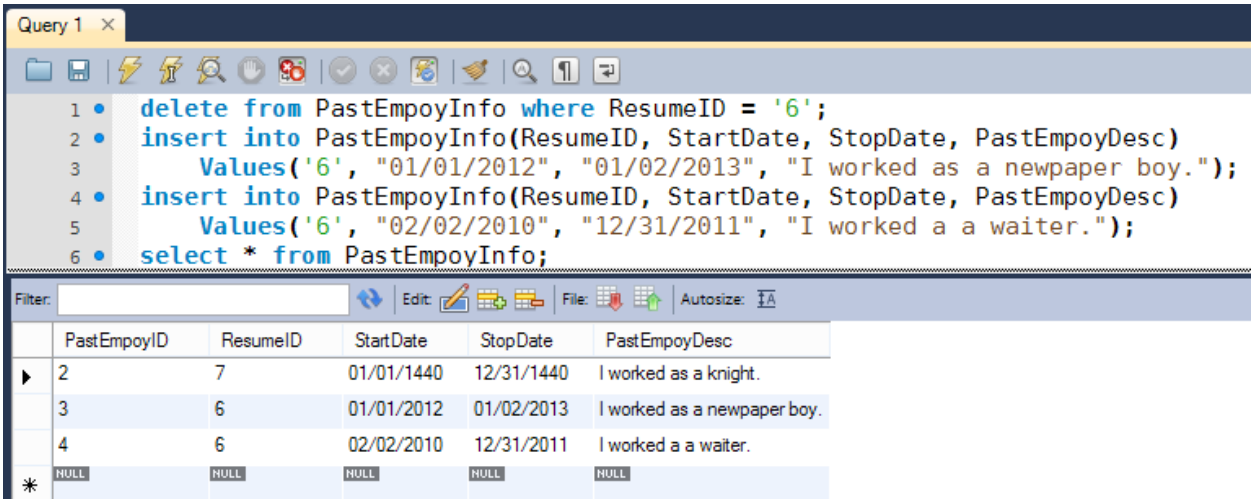
We got the ID from the previous command.

We now update the ContactInfo, and PositionInfo.



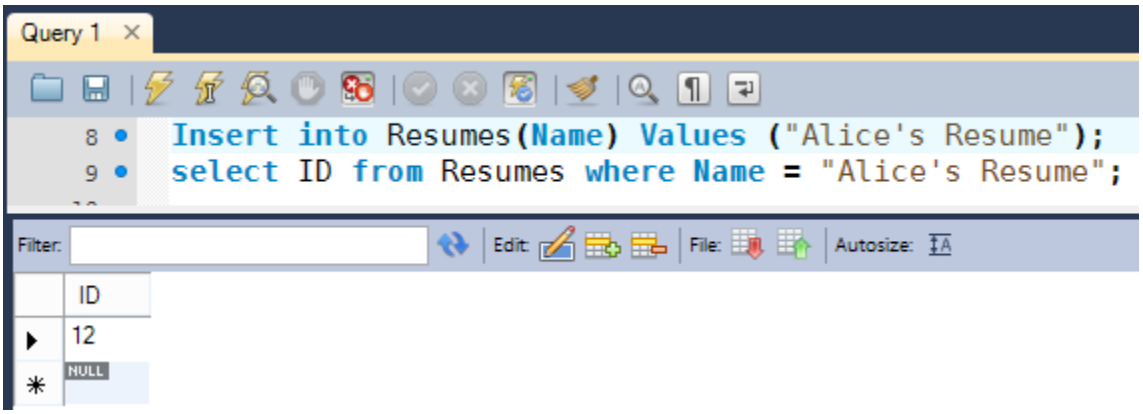


We also need to update PastEmpoyInfo. However, we cannot simply update fields as it is hard to match new information with old information as the number of past employment jobs may have changed. Instead it is easier to delete all the previous employment jobs and insert new ones from the session data.

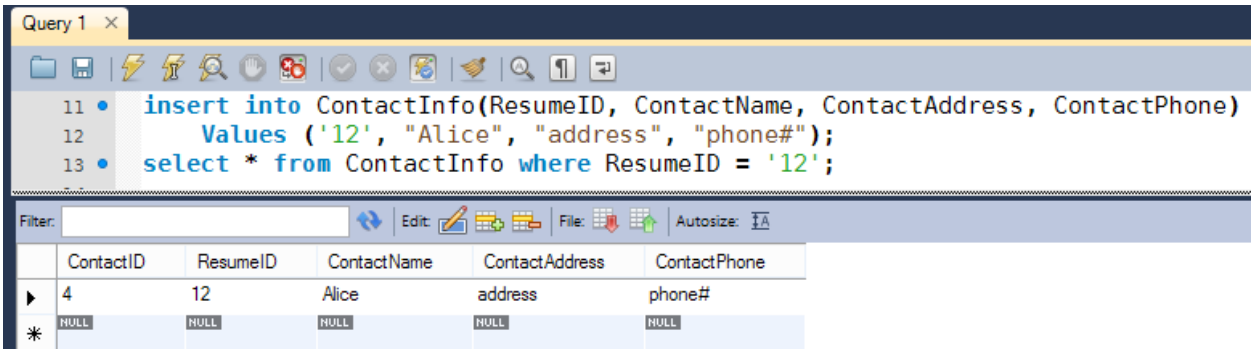


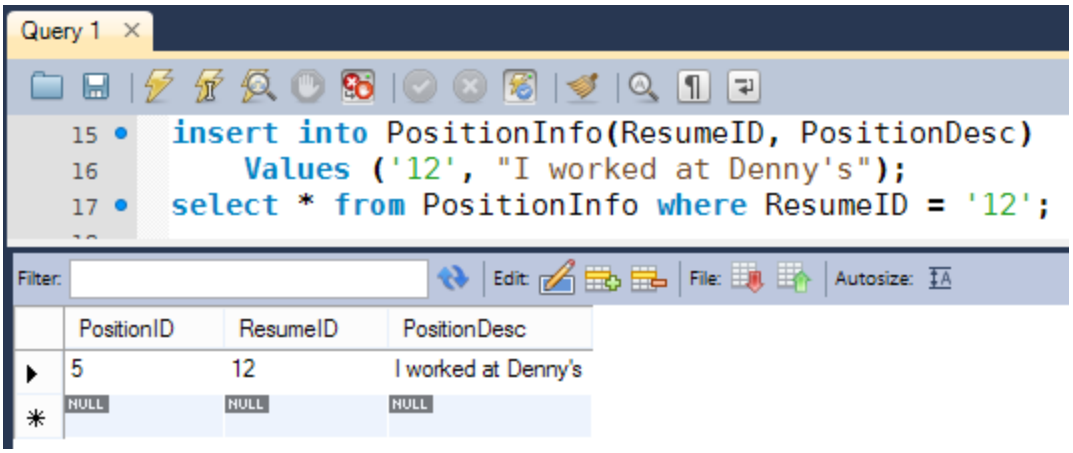
If the resume did not exist

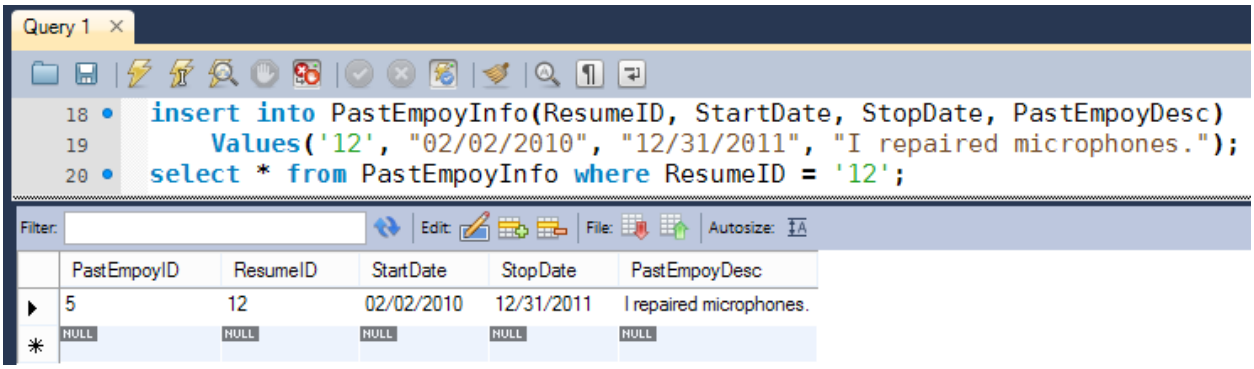
In this case we first create a new resume field in the table and get the ID for it.



Now we can use that ID to add information to the ContactInfo, PositionInfo, and PastEmpoyInfo tables.



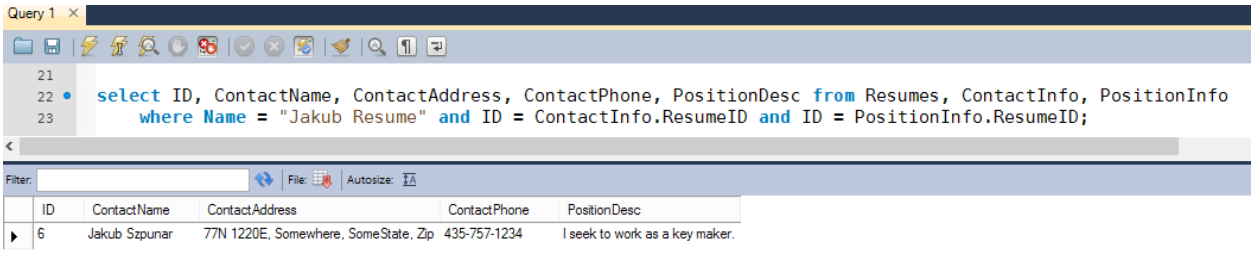




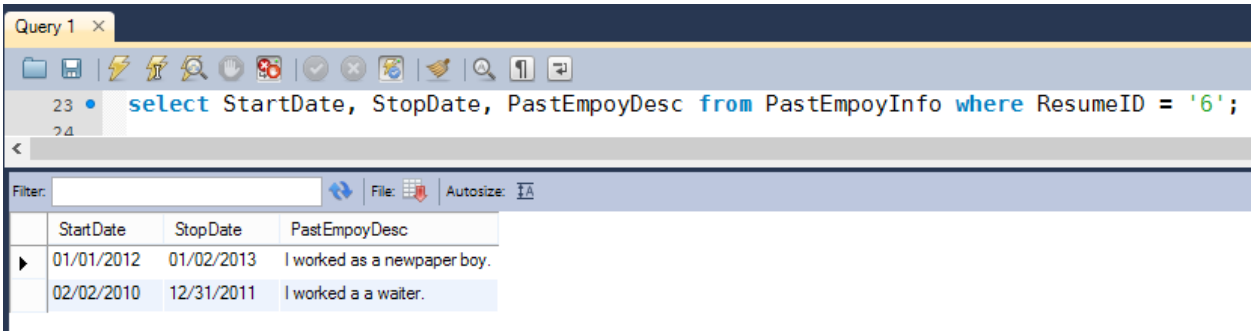
After all this, we have a new resume in the database. These commands inserted new rows into each table. The PastEmpoyInfo will be called for each past employment history added.

Load/View Command in Website:

We need to attempt to load/view a resume from the DB, let's first get the information from Resume, ContactInfo, and PositionInfo as we know there will be a single row of data in each table if the Resume exists. If a resume did not exist, we would not receive anything back. The queries for load and view are identical as they both require the same thing from the DB, getting the information out of it.

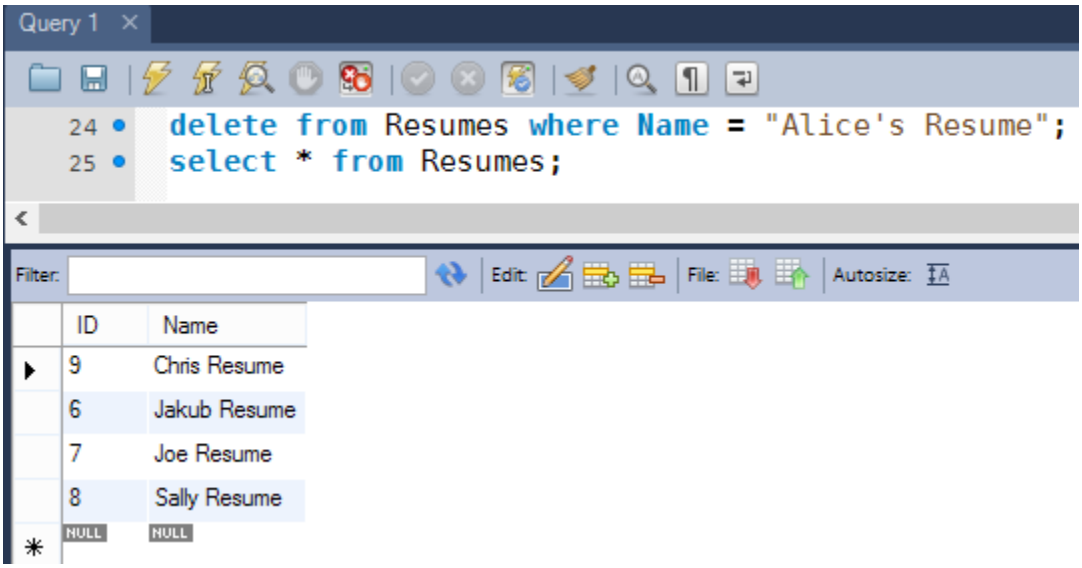


If we did get a row back, we can now get all the past employment history from the resume. We use the ID received back from the first SQL command. I decided to do this in two steps instead of one because if we did it in one step, we would receive a lot of duplicate information back if there was more than one past employment history. (Address/Phone/etc would be sent back for each employment history).

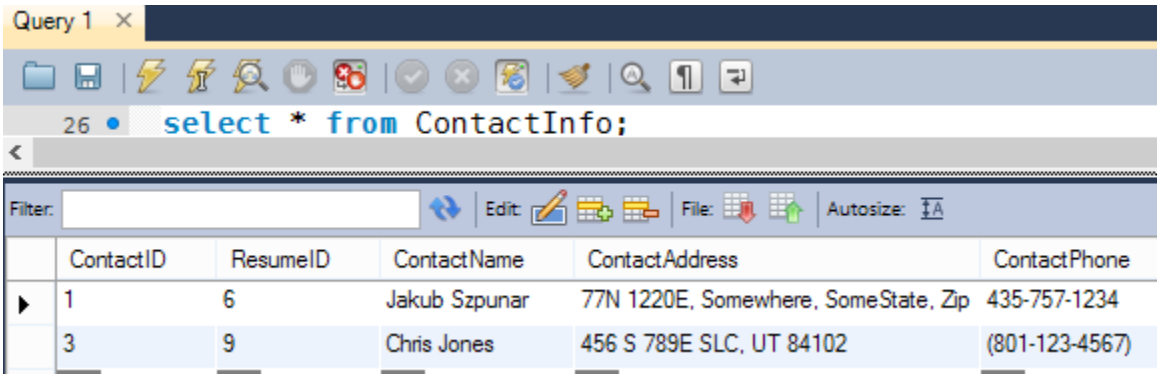


Delete Command in Website:

In order to delete something, we simply need to delete the row in the Resume table. All of the information in the other tables will be automatically deleted due to the Cascade foreign key constraints.



We can look in the other tables and see accompanying information is deleted automatically. Alice's unique key was 12. Let's look in ContactInfo for anything to do with Alice:



We see there is nothing to do with Alice, Everything was automatically deleted.