Database Design and Modeling - Section 0101

Team Project Final Submission Report

5/12/2024

Team G3

Kieran M., Ali S., Justin H.

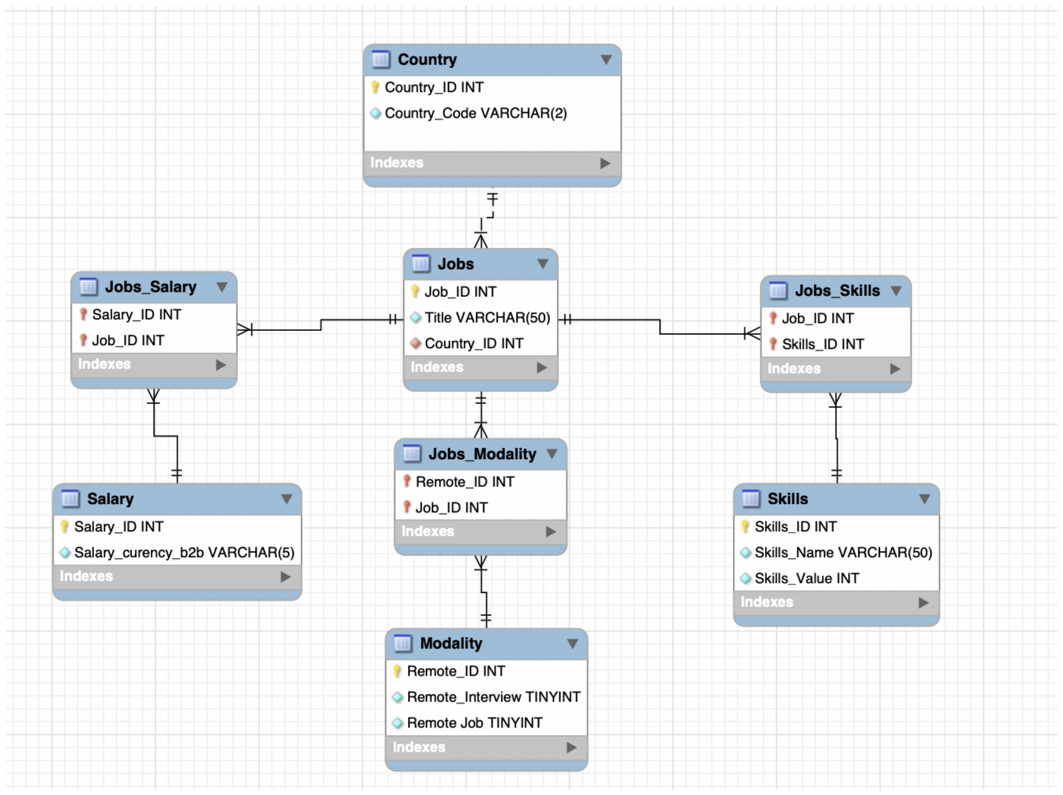
Introduction

Using the Polish IT job listings dataset, our team sought out to create a streamlined database that presents information about IT job listings not just in Poland, but other major European countries as well. We approached our database creation process with the intention to provide users with quick and useful information about these job listings, allowing them to discern the feasibility and capability of attaining a desired position. Initially, our scope was far too grand, with intentions to include information such as a specific company name, hardware accessabilities, and paid time off, among other things. Once our team was able to narrow down the capabilities that the initial dataset provides, we were able to focus on the potential information we could offer, and isolate that with which we found most useful for our project.

Once our team was able to isolate the information we desired for our database, it made every step of the design process much easier and effective. Focusing on the type of jobs available, skills required for said jobs, and the salary offered, among other information were instrumental in approaching our database creation as a whole. Because we initially struggled with determining what kinds of information should be included in our database, reaching a seven table minimum was a struggle. It was not until we spoke with our advisor that we were afforded new ideas that assisted us in our approach. A categorical table and salary table were instrumental in strengthening our database to our team's satisfaction, suggestions given to us that we implemented effectively.

Database Description

Logical Design



In this ERD, everything is a many-to-many relationship, besides jobs to the country. There are only many jobs for one country as indicated by the dotted line. For our tables, we originally had “Country,” “Jobs,” “Skills,” and a linking table “Job_Skills.” In order to meet the number of table requirements we created the “Salary” and “Modality" table, along with their linking tables “Jobs_Salary” and “Jobs_Modality.” All of the ID’s have data types INT being that it is an integer. All of the strings like “Skills_Name,” “Salary_currency_b2b” and “Title” have data type VARCHAR(x) to indicate their variable character length. Lastly, the “Remote_Interview” and “Remote_Job” are TINYINT to become binary 1's and 0’s in our created database.

Physical Database

We created a total of 5 tables with 3 linking tables. The way we imported the data was through table data import wizard. We encountered many errors while importing - such as formatting issues or errors that helped us realize that we had repeating columns in our sample. We then exported the data to a self-contained file so that other members can use it.

Sample Data

The way we filtered out rows from the original dataset is by date. Although we didn't utilize the date, or "published at" column, we initially drew our subset of data from rows that held relation in terms of their publishing date - any publishing date in the month of April and May. We did not include columns like "salary_from_mandate" and "salary_to_mandate" because those rows yield 0's in every row, so therefore there is not enough varying information to use. In our sample data tables, "Country", "Salary" and "Modality" do not contain 15 rows simply because the original dataset did not have this information. For example, the Salary table contained less than 15 worldwide recognized currencies, preventing the possibility for meeting a 15 row minimum.

Views/Queries

We created five views using our database. Our first view, data_only_pl, uses both the Country and Jobs_Salary tables. It uses a WHERE clause in order to parse through the Country_ID column to display data only relating to Poland.

Our second view, job_salary_outside_of_us, utilizes our Salary, Jobs_Salary, and Jobs tables. It parses the salary_currency_b2b column in order to display currency types for job listings that are outside of the United States within this database.

Our third view, job_skills_without_pl, is a much more involved query. It uses the Country, Jobs, Job_Skills, and the Skills table in order to provide data on job listings that are outside of Poland. In order to do so, we used a subquery that contains a WHERE AND clause for the purposes of displaying not only jobs that are outside of Poland, but those that also maintain a Job_ID between 1 and 9.

Our fourth view, remote_only_a_to_s, is our most involved query, fulfilling four out of the five requirements for a view. However, we only use three tables for this query, those being

Modality, Jobs_Modality, and Jobs. By using a COUNT clause in our SELECT statement, we were able to compile jobs that had both remote interviews and the actual ability to work remotely, naming this aggregated set of data “Fully Remote”. Then, by using BETWEEN in our WHERE clause, we restricted the search of fully remote jobs to those whose title starts between A and S. We did this in order for the COUNT to produce a subset of rows.

Our final query, skills_only_sql, makes use of the Skills, Jobs_Skills and Jobs tables. By using a COUNT function, we strip the quantifying value of a skill and only represent if a skill is necessary for a certain job. Then, by using a LIKE function within our WHERE clause, we search for jobs that only contain the keyword SQL, in order to return jobs in the dataset that hold SQL as a skill requirement.

View Name	**Req. A**	**Req. B**	**Req. C**	**Req. D**	**Req. E**
data_only_pl;	**X**	**X**			
job_salary_outside_of_us;	**X**	**X**			
job_skills_without_pl;	**X**	**X**			**X**
remote_only_a_to_s;	**X**	**X**	**X**	**X**	
skills_only_sql;	**X**		**X**		

Changes From Original Design

As stated in the introduction, when looking back to our project proposal, our initial scope of the database we intended to create was far too grand. In the beginning of our approach, our team desired to populate our database with potential entities such as requirements, schedules, and even provided hardware. Additionally, we sought to include attributes such as company names, raise potentials, and sick days, among many other columns. Many of these entities and attributes were not even available within the dataset we were pulling information from. This was from an initial misunderstanding about what capabilities this Polish IT job listings dataset could actually provide our team. Once we understood this erroneous approach to compiling information into our database, it became a stepping stone that paved the way for our team’s future success.

Narrowing down the scope of our database allowed us to approach its creation in a far more effective and streamlined manner.

With a renewed understanding of the dataset, it became much easier to both plan and implement entities and attributes into our database. Simple efficiency became the ethos for our new approach, with entities resigning themselves to only the most useful information the dataset provided. Tables such as country, skills, and salary were some of the most useful pieces of information we could transfer, and with a minimized necessity for column density, our team was capable of creating a database that displayed only the most key points of data for user simplicity.

Database Ethics Considerations

Throughout the semester while working on this database creation project, our team foresaw no potential breach of database ethics, whether that be involving diversity, equity, inclusion, data privacy, or even fair use. After the successful completion of our database, our team can confidently say that database ethics was not relevant to our project topic. Considering the information our database offers, and the public data we used, no ethics considerations were necessary. Being composed solely of information from potential employment opportunities within Europe, our database provides users with the information about job listings, the skills required for said opportunities, and the salaries that come with them. Especially when considering how minimal our database is, along with our use of strictly raw data, database ethics were not relevant to our project topic.

Lessons Learned

One of the major lessons our team learned involved our approach to the creation of entities. While we initially lacked the minimum requirement for entities, our team established a time to connect with a member of the instructional team, in order to assist us in creating additional tables. In doing so, our team was not only able to create enough entities required for our final project, but we also learned an effective method of searching for potential entities on our own for future endeavors. Another lesson we learned involved the creation of our ERD. Our team initially struggled with comprehending the ERD creation process, and we had to establish additional time in order to study and implement how the ERD creation process works. In doing so, our team was confidently able to create a functional, effective ERD which displays the framework of our database.

Potential Future Work

If our team were to continue creation of this database, an opportunity of extension would involve providing city locations from within whatever country a particular job offer is from. In doing so, our database would have an even more precise layer of information to provide our users with when searching for IT jobs. Another potential addition that would add substantial value to our database would be information on the potential of a salary increase for a certain job. In doing so, we would not only be adding another layer of depth to the data we provide, but this additional information would have allowed our team to use more intricate aggregation queries.

Citations

(n.d.). Just Join IT. https://justjoin.it/