PHASE 3

DB Creation, Population, Business Questions

Florentina Vania Santosa

Database Structure

This SQL project utilizes a detailed dataset comprising over 10,000 job postings from LinkedIn, collected over two distinct days. Each posting is enriched with 27 attributes such as job title, description, salary, location, and work-types, along with supplementary files detailing benefits, skills, and industries. Additionally, most of these postings are linked to corresponding companies, with a separate CSV file detailing company-specific information like description, headquarters, employee count, and follower numbers from their LinkedIn page.
The dataset offers extensive analytical possibilities, including analysis of top-paying job titles, companies, and regions, salary and comparison of internship offerings and benefits across industries and companies which will deliver insights not only to the company how to attract best candidates but for job-seekers to understand the job advertisement trend nowadays.

Discussion of how to convert the datasets into tables

The conversion of datasets into database tables is a critical process in database design and data management. Below is the process on how I convert the datasets and by following these steps, the LinkedIn datasets were effectively converted into a structured relational database ready for querying and analysis.

- Understanding the Data Structure: The initial step involves reviewing the datasets to understand the data structure, including the relationships between different data points. For the LinkedIn dataset, this meant analyzing the CSV files to determine how job postings, companies, benefits, skills, and other attributes are related.

- Normalization: To minimize redundancy and enhance data integrity, the data is normalized. This involves organizing data into tables in such a way that each data element is stored in only one place. In the LinkedIn dataset, this meant creating separate tables for jobs, companies, job benefits, and so on.

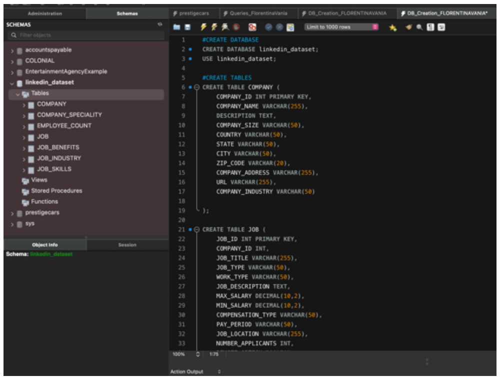
- Defining Tables: For each entity represented in the datasets, a table is created. The table structure is determined by the columns present in the CSV files. For example, the `JOB` table was created with columns for job ID, job type, job title, etc., based on the attributes found in the job postings dataset.

- Assign Primary Keys (PK): Each table is given a primary key, which uniquely identifies each record. For instance, `JOB_ID` is the primary key for the `JOB` table, ensuring each job posting is unique.

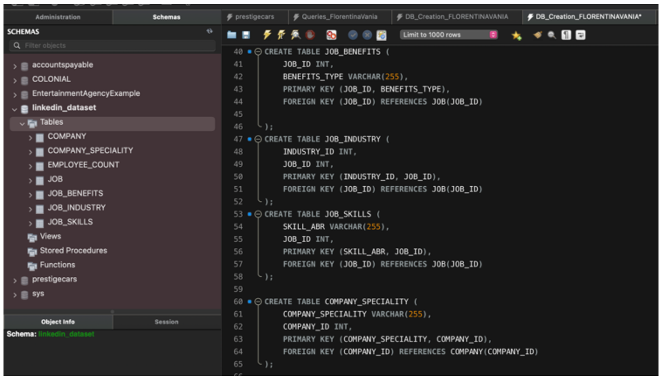
- Determine Foreign Keys (FK): Relationships between tables are established using foreign keys. A foreign key in one table points to a primary key in another table, creating a link between the two. For example, `COMPANY_ID` in the `JOB` table is a foreign key that references the primary key in the `COMPANY` table.

- Data Types and Sizes: Each field in the tables is assigned an appropriate data type and size, such as `INT` for numeric identifiers, `VARCHAR` for string-based fields, and `DECIMAL` for financial figures. The sizes are chosen based on the maximum length observed in the datasets to ensure all data fits while also maintaining efficiency.

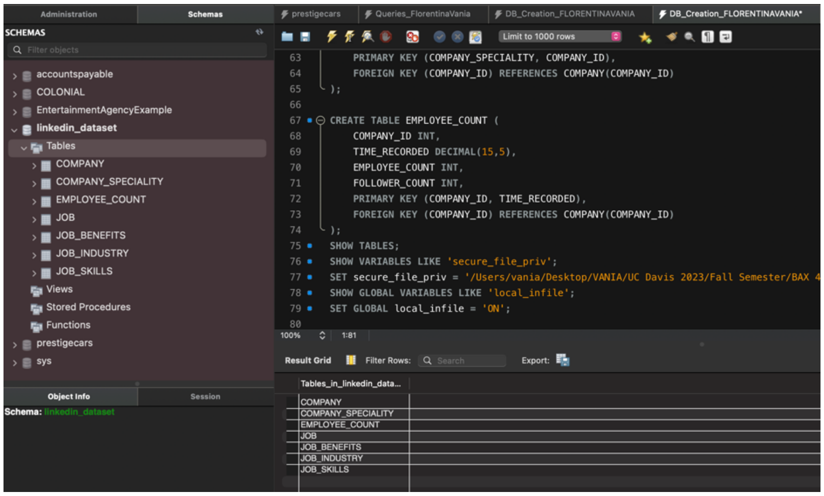
- Importing Data: After creating the tables with their respective constraints and data types, data from the CSV files is imported into the database. This is done using commands like `LOAD DATA INFILE`, which insert data into the correct tables while maintaining the relationships defined by the primary and foreign keys.



```sql
#CREATE DATABASE
CREATE DATABASE linkedin_dataset;
USE linkedin_dataset;

#CREATE TABLES
CREATE TABLE COMPANY (
    COMPANY_ID INT PRIMARY KEY,
    COMPANY_NAME VARCHAR(255),
    DESCRIPTION TEXT,
    COMPANY_SIZE VARCHAR(50),
    COUNTRY VARCHAR(50),
    STATE VARCHAR(50),
    CITY VARCHAR(50),
    ZIP_CODE VARCHAR(20),
    COMPANY_ADDRESS VARCHAR(255),
    URL VARCHAR(255),
    COMPANY_INDUSTRY VARCHAR(50)

);

CREATE TABLE JOB (
    JOB_ID INT PRIMARY KEY,
    COMPANY_ID INT,
    JOB_TITLE VARCHAR(255),
    JOB_TYPE VARCHAR(50),
    WORK_TYPE VARCHAR(50),
    JOB_DESCRIPTION TEXT,
    MAX_SALARY DECIMAL(10,2),
    MIN_SALARY DECIMAL(10,2),
    COMPENSATION_TYPE VARCHAR(50),
    PAY_PERIOD VARCHAR(50),
    JOB_LOCATION VARCHAR(255),
    NUMBER_APPLICANTS INT,
```



```sql
CREATE TABLE JOB_BENEFITS (
    JOB_ID INT,
    BENEFITS_TYPE VARCHAR(255),
    PRIMARY KEY (JOB_ID, BENEFITS_TYPE),
    FOREIGN KEY (JOB_ID) REFERENCES JOB(JOB_ID)

);
CREATE TABLE JOB_INDUSTRY (
    INDUSTRY_ID INT,
    JOB_ID INT,
    PRIMARY KEY (INDUSTRY_ID, JOB_ID),
    FOREIGN KEY (JOB_ID) REFERENCES JOB(JOB_ID)
);
CREATE TABLE JOB_SKILLS (
    SKILL_ABR VARCHAR(255),
    JOB_ID INT,
    PRIMARY KEY (SKILL_ABR, JOB_ID),
    FOREIGN KEY (JOB_ID) REFERENCES JOB(JOB_ID)
);

CREATE TABLE COMPANY_SPECIALITY (
    COMPANY_SPECIALITY VARCHAR(255),
    COMPANY_ID INT,
    PRIMARY KEY (COMPANY_SPECIALITY, COMPANY_ID),
    FOREIGN KEY (COMPANY_ID) REFERENCES COMPANY(COMPANY_ID)
);
```



```sql
    PRIMARY KEY (COMPANY_SPECIALITY, COMPANY_ID),
    FOREIGN KEY (COMPANY_ID) REFERENCES COMPANY(COMPANY_ID)
);

CREATE TABLE EMPLOYEE_COUNT (
    COMPANY_ID INT,
    TIME_RECORDED DECIMAL(15,5),
    EMPLOYEE_COUNT INT,
    FOLLOWER_COUNT INT,
    PRIMARY KEY (COMPANY_ID, TIME_RECORDED),
    FOREIGN KEY (COMPANY_ID) REFERENCES COMPANY(COMPANY_ID)
);
SHOW TABLES;
SHOW VARIABLES LIKE 'secure_file_priv';
SET secure_file_priv = '/Users/vania/Desktop/VANIA/UC Davis 2023/Fall Semester/BAX 4
SHOW GLOBAL VARIABLES LIKE 'local_infile';
SET GLOBAL local_infile = 'ON';
```

Tables_in_linkedin_data...
COMPANY
COMPANY_SPECIALITY
EMPLOYEE_COUNT
JOB
JOB_BENEFITS
JOB_INDUSTRY
JOB_SKILLS

Tables Dictionary in the Database

Table: JOB

JOB_ID	Unique identifier for each job, as seen on LinkedIn.
COMPANY_ID	Links the job posting to its corresponding company in the companies.csv file.
JOB_TITLE	The designated title for the job.
JOB_TYPE	Nature of employment (e.g., Full-time, Part-time, Contract).
WORK_TYPE	Specific work arrangement for the job.
JOB_DESCRIPTION	Detailed information about the job.
MAX_SALARY	Highest salary offered.
MIN_SALARY	Lowest salary offered.
COMPENSATION_TYPE	Nature of salary compensation.
PAY_PERIOD	Frequency of salary payment (e.g., Hourly/Monthly/Yearly).
JOB_LOCATION	Geographical location of the job.
NUMBER_APPLICANTS	Count of job applications received.
REMOTE_OPTION	Indicates if the job can be done remotely.
NUMBER_VIEWS	Total views of the job posting.

Table: COMPANY

COMPANY_ID	Unique identifier for each company, as listed on LinkedIn.
COMPANY_NAME	Name of the company.
DESCRIPTION	Overview of the company.
COMPANY_SIZE	Scale of the company based on employee count.
COUNTRY	Country where the company's headquarters is located.
STATE	State where the company's headquarters is situated.
CITY	City of the company's main office.
ZIP_CODE	Postal code for the company's headquarters.
COMPANY_ADDRESS	Full address of the company's main office.
URL	LinkedIn page of the company.
COMPANY_INDUSTRY	Type of the industry.

Table: JOB_BENEFIT

BENEFITS_TYPE	Kind of benefit offered (e.g., 401K, Medical Insurance).
JOB_ID	Identifier for the job related to the benefit.

Table: JOB_INDUSTRY

INDUSTRY_ID	Unique identifier for each company, as listed on LinkedIn.
JOB_ID	Identifier for the job related to the industry.

Table: JOB_SKILL

SKILLS_ABR	Skills required for the job.
JOB_ID	Identifier for the job related to the skills.

Table: EMPLOYEE_COUNT

COMPANY_ID	Unique identifier for each company.
TIME_RECORDED	Timestamp of when the data was recorded in Unix time.
FOLLOWER_COUNT	Number of followers the company has on LinkedIn.
EMPLOYEE_COUNT	Total number of employees in the company.

Table: COMPANY_SPECIALTY

COMPANY_SPECIALTY	Company’s strong point.
COMPANY_ID	Identifier for the company.

Challenges Faced During Importing of The Data

When importing data into SQL, several challenges can arise due to the nature of the data and the requirements of the SQL database. Here’s a summary of challenges and solutions I have encountered during this step:

Challenge 1: Data Type Mismatches Mismatched data types between the source data and the SQL table schema can cause import failures.
Solution: Perform data profiling to understand the data types in the source files. Modified the table schema to match the source data types or convert the data to the appropriate types before importation. I converted data type INT become BIG INT because the content of several columns requires it.

Challenge 2: Combine Two Data and Treat Missing Values Source data may contain missing values that do not align with the not-null constraints of the database.
Solution: When I combine two files for JOB and Job Industry, I did VLOOKUP to insert the column Job Industry match with the JOB_ID in the table JOB and drop those Jobs who has no industry in JOB file, same goes with the Job Benefit.

Challenge 3: Inconsistent Formats Data like integers and strings may be in different formats that the database does not recognize.
Solution: Standardize the formats in the source data using transformation scripts or during the import process,

Challenge 4: Data Cleaning The source data may contain duplicates, outliers, or erroneous values that need to be cleaned.
Solution: Clean the data prior to import using data processing tools or languages Python to handle duplicates data.

Business Questions

Job Skills and Salary Analysis

Question: What are the most lucrative job skills for targeted job seeking or career guidance?
Query: Analyze average salaries associated with various job skills across industries by joining the JOB and JOB_SKILLS tables.
Insight: Identifying high-paying skills enables job seekers to target skill development for better career opportunities and higher salaries.

Benefit Type Trends by Company Size

Question: Which type of benefits are most offered by companies of different sizes?
Query: Combine COMPANY and EMPLOYEE_COUNT tables using subqueries to find the most recent employee count, then join with JOB and JOB_BENEFITS tables to aggregate the types of benefits offered, categorized by company size which defined by number of employees they have.
Insight: Understanding prevalent benefits offered by different-sized companies helps job seekers align expectations and target suitable employers.

Job Preferences: Remote vs. On-Site Positions

Question: What is the preference trend among job seekers for remote versus on-site positions?
Query: Analyze the relationship between remote versus on-site positions by analyzing data of the number applicants in the JOB table.
Insight: Insights into remote vs. on-site job preferences guide companies in structuring their job offers to attract the right candidates.

Remote Job Skill Requirements

Question: What are the common skill requirements for remote jobs?
Query: Investigate prevalent skills required for remote job positions by combining data from JOB_SKILLS and JOB tables, focusing on roles with a remote work option.
Insight: Knowledge of skill trends in remote jobs allows job seekers to tailor their skillsets to the growing remote work market.

Industry Trends in Job Posting Traffic

Question: Which industries are currently seeing the highest traffic in job postings?
Query: Explore which industries are experiencing the most activity in job postings by joining the JOB and COMPANY tables and examining job view counts.
Insight: Recognizing industries with high job posting traffic reveals current market demands and growth sectors for job seekers.

Company Popularity and Job Posting Volume

Question: How does a company's online presence impact its job posting engagement?
Query: Analyze the correlation between a company's follower count and its job posting volume by linking the most recent data from the EMPLOYEE_COUNT table with job postings in the JOB table. Using subqueries, we will be using the most

recent follower count based on timeseries we have in table EMPLOYEE_COUNT then we will join the table with JOB to aggregate the number of job posted for each company.

Insight: A correlation between online presence and job engagement informs companies on the impact of their digital footprint on recruitment whether company should allocate more budget/resource for their online presence or not.

Attractive Benefits for High-Applicant Jobs

Question: Which benefits are most attractive to applicants in job listings?

Query: Identify the most popular job benefits by linking JOB_BENEFITS with JOB and aggregating the number of applicants per benefit type.

Insight: Identifying benefits that draw the most applicants helps companies optimize their job listings to attract top talent.

Work Type Distribution and Job Salaries

Question: How do salaries vary across different work types?

Query: Calculate the average minimum salaries for various work types by utilizing WORK_TYPE and MIN_SALARY data from the JOB table.

Insight: Analyzing salary trends across different work types assists job seekers and employers in making informed salary negotiations and offers.