**ETL Project Report**

***Comparing Data Analyst Positions in Glassdoor.com and Dice.com***

Group Members: Melodie Vines, Tiffany Teasley, Stefanee Richardson, and Mabel Duran-Sanchez

1. **Extract:**
   1. We utilized <https://www.kaggle.com/> to find two datasets on Data Analysts jobs, one from *glassdoor.com* and the other from *dice.com*.
      1. Glassdoor dataset: <https://www.kaggle.com/andrewmvd/data-analyst-jobs>
      2. Dice.com dataset: <https://www.kaggle.com/PromptCloudHQ/us-technology-jobs-on-dicecom>
   2. Both datasets were CSV files which we read into pandas using Jupyter Notebook.
2. **Transform:**
   1. As we loaded each of the datasets, we filtered the dataset from *glassdoor.com* to only show those columns that we deemed relevant to our analysis and that were also present in the *dice.com* dataset (Company Name, Job Description, Location, Job Title, Source, and Salary Estimate).
   2. To each dataset we added a “Source” column to ensure that even after merging the datasets, we could easily identify where each data point came from (either *glassdoor.com* or *dice.com*).
   3. We also cleaned the “Salary Estimate” columns from the *glassdoor.com* dataset and removed the repeated string “(Glassdoor est.)” since it was not relevant. We cleaned the “Company Name” column in the *glassdoor.com* dataframe by removing unnecessary characters (which followed this format “\n\*”) as well.
   4. From the *dice.com* dataset, we filtered the job title columns to any jobs containing “Data Analyst” in their title as this dataset was quite large and included different kinds of jobs within the Tech sector which were not relevant to our analysis which focuses exclusively on Data Analyst roles.
   5. We also renamed the columns so that the *dice.com* dataframe followed the same format as our *glassdoor.com* dataframe. This also involved adding a “Source” and “Salary Estimate” column.
   6. At this point, we were able to combine both datasets and create a larger comprehensive dataframe. In doing so, we also checked for duplicates.
      1. As we checked for duplicates, we quickly realized that many of those job titles which came up as duplicates often times had different salary ranges or even different job locations despite being within the same company. This led us to believe that they may in fact not be duplicates. However, we could also not guarantee 100% that they were not indeed duplicates.
      2. ***Limitations:*** We understand that this data may not tell the complete story and that there may be in fact duplicates. However, with the information these two datasets provided, there was not enough information to discern duplicates.
3. **Load:** 
   1. In an effort to reduce and/or eliminate data redundancy, we created normalized tables prior to loading data into the local database. This consisted of identifying fields which included duplicative information including the following 3 tables:
      1. Source: the job board in which the job posting was posted.
      2. Salary: salary range for each posting (it was noted that this information was only available for jobs posted within the Glassdoor source).
      3. Location: city and state in which the job posted.
   2. We then created the "Job" table to bring in the foreign keys of the normalized tables (source\_id, salary\_id, location\_id).
   3. We connected the 4 tables to the local database in PostgreSQL. We chose PostgresSQL in particular because of the ability to run complex SQL queries and the ability to work with lots of existing applications.