Looking at the bar chart and summarized salary view, the data set looks suspicious and might not be correct. Following are the key reasons for the suspicions

1. The average salary of Senior Engineer is less than Engineer, which does not make sense.
2. The average salary of Manager is less than Staff and Senior Staff, which does not make sense.
3. The difference among average salaries of Assistant Engineer, Engineer, Senior Engineer and Technical Leader are less than 100$ (that is less than 1% of each other), which does not make sense.

Following is the summary of attached files

1. employee\_Ashok.sql is a combined SQL file to define the table schemas and querying the database. It has also the answers to all the SQL Related questions.
2. Ashok-Employees-ERD-1.rtf and Ashok-Employees-ERD-2.docx files are the ERD files. These files are created with the help from tools provided by quickdatabasediagrams. Ashok-Employees-ERD-1 is the direct import from quickdatabasediagrams and Ashok-Employees-ERD-2 has the screen shot view to show the pictorial relationship among the tables.
3. Ashok-Employees-Extra.ipynb has the Python/Pandas code to connect with the database and display the bar chart. It is for the bonus question. employee-barchart.png file has the output from the bar chart.
4. Ashok\_Summary\_EmployeeSQL.docs has the summary view, brief analysis of the derived data and part answers to the bonus question.
5. my\_password\_keys has the password. This file is not uploaded to the GitHub.

Following is the SQL Code

-- delete departments table, if needed

DROP TABLE departments;

-- create schema for departments table and import the corresponding data

CREATE TABLE departments (

dept\_no VARCHAR(10) NOT NULL PRIMARY KEY,

dept\_name VARCHAR(30) NOT NULL

);

-- show 5 entries of departments table after importing the data

SELECT \* FROM departments

LIMIT 5;

-- delete employees table, if needed

DROP TABLE employees;

-- create schema for employees table and import the corresponding data

CREATE TABLE employees (

emp\_no INT NOT NULL PRIMARY KEY,

birth\_date DATE NOT NULL,

first\_name VARCHAR(30) NOT NULL,

last\_name VARCHAR(30) NOT NULL,

gender CHAR(1) NOT NULL,

hire\_date DATE NOT NULL

);

-- show 5 entries of departments table after importing the data

SELECT \* FROM employees

LIMIT 5;

-- delete dept\_emp table, if needed

DROP TABLE dept\_emp;

-- create schema for dept\_emp table and import the corresponding data

CREATE TABLE dept\_emp (

emp\_no INT NOT NULL,

FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no),

dept\_no VARCHAR(10) NOT NULL,

FOREIGN KEY (dept\_no) REFERENCES departments(dept\_no),

from\_date DATE NOT NULL,

to\_date DATE NOT NULL

);

-- show 5 entries of dept\_emp table after importing the data

SELECT \* FROM dept\_emp

LIMIT 5;

-- delete dept\_manager table, if needed

DROP TABLE dept\_manager;

-- create schema for dept\_manager table and import the corresponding data

CREATE TABLE dept\_manager (

dept\_no VARCHAR(10) NOT NULL,

FOREIGN KEY (dept\_no) REFERENCES departments(dept\_no),

emp\_no INT NOT NULL,

FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no),

from\_date DATE NOT NULL,

to\_date DATE NOT NULL

);

-- show 5 entries of dept\_manager table after importing the data

SELECT \* FROM dept\_manager

LIMIT 5;

-- delete salaries table, if needed

DROP TABLE salaries;

-- create schema for salaries table and import the corresponding data

CREATE TABLE salaries (

emp\_no INT NOT NULL,

FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no),

salary INT NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE NOT NULL

);

-- show 5 entries of salaries table after importing the data

SELECT \* FROM salaries

LIMIT 5;

-- delete titles table, if needed

DROP TABLE titles;

-- create schema for titles table and import the corresponding data

CREATE TABLE titles (

emp\_no INT NOT NULL,

FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no),

title VARCHAR(30) NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE NOT NULL

);

-- show 5 entries of titles table after importing the data

SELECT \* FROM titles

LIMIT 5;

-- Answer to Question-1

SELECT e.emp\_no, e.last\_name, e.first\_name, e.gender, s.salary

FROM employees AS e

INNER JOIN salaries AS s ON

e.emp\_no = s.emp\_no;

-- Answer to Question-2

SELECT \* FROM employees

WHERE EXTRACT(YEAR FROM employees.hire\_date) = 1986;

-- Answer to Question-3

SELECT m.dept\_no, d.dept\_name, m.emp\_no AS "mgr\_emp\_no", e.last\_name, e.first\_name, m.from\_date AS "employment start date", m.to\_date AS "employment end date"

FROM dept\_manager AS m

INNER JOIN departments AS d ON m.dept\_no = d.dept\_no

INNER JOIN employees AS e ON e.emp\_no = m.emp\_no

ORDER BY e.last\_name ASC;

-- Answer to Question-4

SELECT e.emp\_no, e.last\_name, e.first\_name, d.dept\_name

FROM employees as e

INNER JOIN dept\_emp AS de ON e.emp\_no = de.emp\_no

INNER JOIN departments AS d ON de.dept\_no = d.dept\_no

ORDER BY e.last\_name ASC;

-- Answer to Question-5

SELECT \* FROM employees

WHERE first\_name = 'Hercules' AND last\_name LIKE 'B%';

-- Answer to Question-6

SELECT e.emp\_no, e.last\_name, e.first\_name, d.dept\_name

FROM employees as e

INNER JOIN dept\_emp AS de ON e.emp\_no = de.emp\_no

INNER JOIN departments AS d ON de.dept\_no = d.dept\_no

WHERE d.dept\_name = 'Sales'

ORDER BY e.last\_name ASC;

-- Answer to Question-7

SELECT e.emp\_no, e.last\_name, e.first\_name, d.dept\_name

FROM employees as e

INNER JOIN dept\_emp AS de ON e.emp\_no = de.emp\_no

INNER JOIN departments AS d ON de.dept\_no = d.dept\_no

WHERE d.dept\_name = 'Sales' OR d.dept\_name = 'Development'

ORDER BY d.dept\_name DESC;

-- Answer to Question-8

SELECT last\_name as "Employees Last Name", COUNT(last\_name) AS "Frequency Count"

FROM employees

GROUP BY last\_name

ORDER BY "Frequency Count" DESC;

Following is the Python/Pandas Code for the bonus question

# Pandas

import pandas as pd

# import matplotlib and numpy

import matplotlib.pyplot as plt

import numpy as np

# SQL Alchemy

from sqlalchemy import create\_engine

# Import password

from my\_password\_keys import password\_key

# Create Engine and conenct to Employee Database

url\_firstpart = 'postgresql://postgres:'

url\_secondpart = '@localhost:5432/Employee'

query\_url = url\_firstpart + password\_key + url\_secondpart

engine = create\_engine(query\_url)

conn = engine.connect()

# Query All Records in salaries and titles table from the Employee Database

salary\_data = pd.read\_sql("SELECT emp\_no, salary FROM salaries", conn)

title\_data = pd.read\_sql("SELECT emp\_no, title FROM titles", conn)

# Merge two dataframes using an inner join

df = pd.merge(title\_data, salary\_data, on="emp\_no")

# only interested in title and salary columns

df = df[['title', 'salary']]

df = df.rename(columns = {"title" : "Job Title", "salary":"Average Salary"})

#group by title and get average of each title

df = df.groupby(["Job Title"]).agg('mean')

df = df.sort\_values(by = "Average Salary", ascending = True)

df['Average Salary'] = df['Average Salary'].map("{:.0f}".format)

df = df.reset\_index()

df

plt.bar(df['Job Title'], df['Average Salary'].apply(pd.to\_numeric), 0.5, color='r', alpha=0.5, align="center")

# Tell matplotlib where we would like to place each of our x axis headers

tick\_locations = [value for value in df['Job Title']]

plt.xticks(tick\_locations, df['Job Title'], size=5)

plt.yticks(size=8)

# Sets the x limits of the current chart

plt.xlim(-0.75, len(df['Job Title'])-0.25)

# Sets the y limits of the current chart

plt.ylim(int(min(df['Average Salary'])) - 10000, int(max(df['Average Salary'])) + 10000)

# Give our chart some labels and a tile

plt.title("Average Salaries for different Job Titles", size = 14)

plt.xlabel("Job Titles", size = 14)

plt.ylabel("Average Salaries in USD", size = 14)

plt.tight\_layout()

plt.savefig("employee-barchart.png")

plt.show()