SQL Challenge - Homework 9

Tools Used

- Postregsql 4 - Type of SQL
- pgAdmin 4 - Database Admin tool
- QuickDBD - Quick Database Diagrams (Tool to create ERD - Entity Relationship Diagrams)
- Jupyter Notebooks - Presentation of work, graphing
- Python, Pandas, MatPlotLib, Numpy, SQL Alchemy - See Imports section

Data Modeling

Using the csv files provided, created an ERD in QuickDBD

- departments.csv
- employees.csv
- salaries.csv
- titles.csv
- dept_emp.csv
- dept_manager.csv

 title

Data Engineering

Next, starting from the ERD above, exported a table schema and edited it for specific data types and other constraints such as primary and foreign keys

Started with this (export from QuickDBD):

```
CREATE TABLE "Departments" (
    "dept_no" VARCHAR(4)   NOT NULL,
    "dept_name" VARCHAR(20)   NOT NULL,
    CONSTRAINT "pk_Departments" PRIMARY KEY (
        "dept_no"
     )
);

CREATE TABLE "Employees" (
    "emp_no" INTEGER   NOT NULL,
    "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,
    CONSTRAINT "pk_Employees" PRIMARY KEY (
        "emp_no"
     )
);

CREATE TABLE "Salaries" (
    "emp_no" INTEGER   NOT NULL,
    "salary" INTEGER   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,
    CONSTRAINT "pk_Salaries" PRIMARY KEY (
        "emp_no","from_date"
     )
);

CREATE TABLE "Titles" (
    "emp_no" INTEGER   NOT NULL,
    "title" VARCHAR(20)   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,
    CONSTRAINT "pk_Titles" PRIMARY KEY (
        "emp_no","from_date"
     )
);

CREATE TABLE "Dep_Emp" (
    "emp_no" INTEGER   NOT NULL,
    "dept_no" VARCHAR(4)   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,
    CONSTRAINT "pk_Dep_Emp" PRIMARY KEY (
        "emp_no","dept_no"
     )
);
```

```
CREATE TABLE "Dep_Manager" (
    "emp_no" INTEGER   NOT NULL,
    "dept_no" VARCHAR(4)   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,
    CONSTRAINT "pk_Dep_Manager" PRIMARY KEY (
        "emp_no","dept_no"
     )
);

ALTER TABLE "Salaries" ADD CONSTRAINT "fk_Salaries_emp_no" FOREIGN KEY("emp_no")
REFERENCES "Employees" ("emp_no");

ALTER TABLE "Titles" ADD CONSTRAINT "fk_Titles_emp_no" FOREIGN KEY("emp_no")
REFERENCES "Employees" ("emp_no");

ALTER TABLE "Dep_Emp" ADD CONSTRAINT "fk_Dep_Emp_emp_no" FOREIGN KEY("emp_no")
REFERENCES "Employees" ("emp_no");

ALTER TABLE "Dep_Emp" ADD CONSTRAINT "fk_Dep_Emp_dept_no" FOREIGN KEY("dept_no")
REFERENCES "Departments" ("dept_no");

ALTER TABLE "Dep_Manager" ADD CONSTRAINT "fk_Dep_Manager_emp_no" FOREIGN KEY("emp_n
o")
REFERENCES "Employees" ("emp_no");

ALTER TABLE "Dep_Manager" ADD CONSTRAINT "fk_Dep_Manager_dept_no" FOREIGN KEY("dept
_no")
REFERENCES "Departments" ("dept_no");
```

Then updated it to this:

```
CREATE TABLE "Departments" (
    -- Create fields
    "dept_no" VARCHAR(4)  NOT NULL,
    "dept_name" VARCHAR(20) NOT NULL,

    -- Add contstraints
    CONSTRAINT "pk_Departments" PRIMARY KEY ("dept_no")
);

CREATE TABLE "Employees" (
    -- Create fields
    "emp_no" INTEGER   NOT NULL,
    "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,

    -- Add constraints
    CONSTRAINT "pk_Employees" PRIMARY KEY ("emp_no")
);

CREATE TABLE "Salaries" (
    -- Create fields
    "emp_no" INTEGER   NOT NULL,
    "salary" INTEGER   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,

    -- Add constraints
    CONSTRAINT "fk_Salaries_emp_no" FOREIGN KEY("emp_no") REFERENCES "Employees" (
"emp_no"),
    CONSTRAINT "pk_Salaries" PRIMARY KEY ("emp_no","from_date")
);

CREATE TABLE "Titles" (
    -- Create fields
    "emp_no" INTEGER   NOT NULL,
    "title" VARCHAR(20)   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,

    -- Add constraints
    CONSTRAINT "fk_Titles_emp_no" FOREIGN KEY("emp_no") REFERENCES "Employees" ("em
p_no"),
    CONSTRAINT "pk_Titles" PRIMARY KEY ("emp_no","from_date")
);

CREATE TABLE "Dep_Emp" (
```

```
    -- Create fields
    "emp_no" INTEGER   NOT NULL,
    "dept_no" VARCHAR(4)   NOT NULL,
    "from_date" DATE   NOT NULL,
    "to_date" DATE   NOT NULL,

    -- Add constraints
    CONSTRAINT "fk_Dep_Emp_emp_no" FOREIGN KEY("emp_no") REFERENCES "Employees" ("e
```

Data Analysis

Phase 1 - SQL Queries

- Executed within pgAdmin4

NOTE: Throughout these queries it was unclear if we were to only use current employee data or include historical as well. When in doubt, it was run both ways noting that current employees have a to_date of 9999-01-01

1. List the following details of each employee: employee number, last name, first name, gender, and salary.

```
SELECT emp_no AS "Employee Number", last_name AS "Last Name", first_name AS "First
 Name", gender AS "Gender",
    (
        SELECT "Salaries".salary
        FROM "Salaries"
        WHERE "Employees".emp_no = "Salaries".emp_no
    ) AS "Salary"
FROM "Employees";
```

 title

2. List employees who were hired in 1986.

```
SELECT emp_no AS "Employee Number", last_name AS "Last Name", first_name AS "First
 Name", hire_date AS "Hire Date"
FROM "Employees"
WHERE EXTRACT(year FROM hire_date) = 1986;
```

 title

3. List the manager of each department with the following information:

- department number, department name, the manager's employee number, last name, first name, and start and end employment dates.

QUESTION: List the Complete History or just the current managers?

Here is the HISTORY

```
SELECT dpt.dept_no AS "Department Number", dpt.dept_name AS "Department Name", dptM
gr.emp_no AS "Manager Number",
    emp.last_name AS "Last Name", emp.first_name AS "First Name", dptMgr.from_date
AS "Start Date",
    dptMgr.to_date AS "End Date"
FROM "Departments" AS dpt, "Dep_Manager" AS dptMgr, "Employees" AS emp
WHERE dpt.dept_no = dptMgr.dept_no AND dptMgr.emp_no = emp.emp_no;
```

Here is just the current

```
SELECT dpt.dept_no AS "Department Number", dpt.dept_name AS "Department Name", dptM
gr.emp_no AS "Manager Number",
    emp.last_name AS "Last Name", emp.first_name AS "First Name", dptMgr.from_date
AS "Start Date",
    dptMgr.to_date AS "End Date"
FROM "Departments" AS dpt, "Dep_Manager" AS dptMgr, "Employees" AS emp
WHERE dpt.dept_no = dptMgr.dept_no AND dptMgr.emp_no = emp.emp_no AND EXTRACT(year
FROM dptMgr.to_date)=9999;
```

4. List the department of each employee with the following information: employee number, last name, first name, and department name.

QUESTION: Is this currently or ever?

Here is the History

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Departments" AS dpt, "Dep_Emp" AS dptEmp, "Employees" AS emp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no;
```

Here is current

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Departments" AS dpt, "Dep_Emp" AS dptEmp, "Employees" AS emp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no  AND EXTRACT(year
FROM dptEmp.to_date)=9999;
```

5. List all employees whose first name is "Hercules" and last names begin with "B."

```
SELECT emp_no AS "Employee Number", first_name AS "First Name", last_name AS "Last
 Name"
FROM "Employees"
WHERE first_name = 'Hercules' and last_name LIKE 'B%';
```

6. List all employees in the Sales department, including their employee number, last name, first name, and department name.

QUESTION: Is this currently or ever?

Here is the History

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Employees" AS emp, "Departments" AS dpt, "Dep_Emp" AS dptEmp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no AND dpt.dept_name
= 'Sales';
```

Here is the current

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Employees" AS emp, "Departments" AS dpt, "Dep_Emp" AS dptEmp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no AND dpt.dept_name
= 'Sales' AND EXTRACT(year FROM dptEmp.to_date)=9999;
```

7. List all employees in the Sales and Development departments, including their employee number, last name, first name, and department name

QUESTION: Is this currently or ever?

Here is the History

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Employees" AS emp, "Departments" AS dpt, "Dep_Emp" AS dptEmp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no AND (dpt.dept_nam
e = 'Sales' OR dpt.dept_name = 'Development');
```

Here is the current

```
SELECT emp.emp_no AS "Employee Number", emp.last_name AS "Last Name", emp.first_nam
e AS "First Name",
    dpt.dept_name AS "Department Name"
FROM "Employees" AS emp, "Departments" AS dpt, "Dep_Emp" AS dptEmp
WHERE dpt.dept_no = dptEmp.dept_no AND dptEmp.emp_no = emp.emp_no AND (dpt.dept_nam
e = 'Sales' OR dpt.dept_name = 'Development') AND EXTRACT(year FROM dptEmp.to_date)
=9999;
```

8. In descending order, list the frequency count of employee last names, i.e., how

Phase 2 - Graphical Analysis

Imports

Pandas

Data manipulation and analysis

MatPlotLib Pyplot

2D plotting

Datetime

Dates and time

Numpy

Supports large, multi-dimensional arrays and matrix manipulation and high level mathematical functions on these arrays

SQLAlchemy

Database Import

In [1]:
```
import pandas as pd

import matplotlib.pyplot as plt

import matplotlib
from matplotlib import style
style.use('seaborn')

import datetime

import numpy as np

import sqlalchemy
from sqlalchemy.ext.automap import automap_base
from sqlalchemy.orm import Session
from sqlalchemy import create_engine, inspect, func

import psycopg2

# Import db pw
from keys import pw
```

Reusable References

In [2]:
```
BOLD = '\033[1m'
END = '\033[0m'
```

Create the Database Connection

In [3]:
```
# Create an engine
engine = create_engine('postgresql://katro:' + pw + '@localhost:5432/SQLChallenge')

# grab a connection
connection = engine.connect()
```

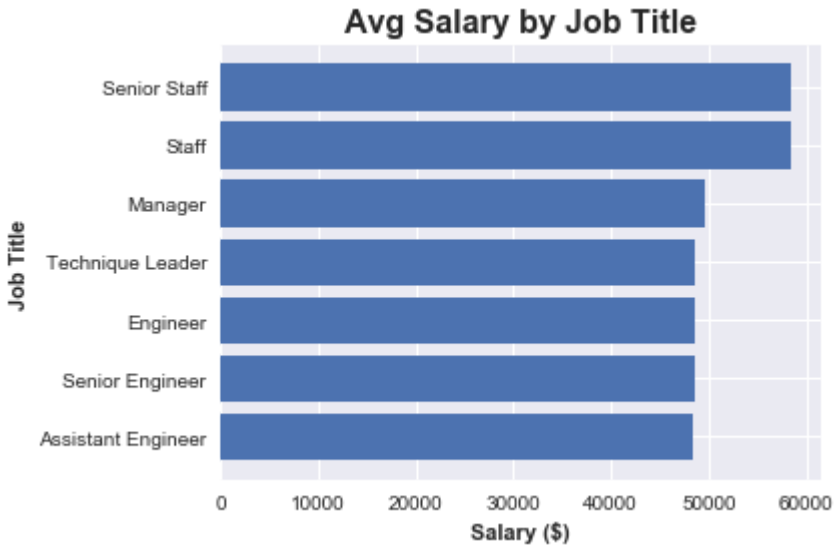
In [4]:
```
# Create the sql statement
sql_statement = 'SELECT t.title AS "Title", AVG(s.salary) AS "Avg Salary" FROM
"Titles" AS t, "Salaries" AS s WHERE t.emp_no = s.emp_no AND EXTRACT(year FROM
t.to_date)=9999 GROUP BY t.title ORDER BY "Avg Salary"'

# Grab the results into a dataframe and display
query_result = pd.read_sql_query(sql_statement, connection)
query_result
```

Out[4]:

	Title	Avg Salary
0	Assistant Engineer	48436.856187
1	Senior Engineer	48501.994322
2	Engineer	48532.428751
3	Technique Leader	48532.833762
4	Manager	49600.555556
5	Staff	58448.920003
6	Senior Staff	58511.960170

In [5]:
```
# Create a horizontal bar chart and plot the title on the y-axis and the salar
y on the x-axis
fig, ax = plt.subplots()
ax.barh(query_result.index, query_result['Avg Salary'])
ax.set_xlabel("Salary ($)", weight='bold')
ax.set_ylabel("Job Title", weight='bold')
ax.set_yticks(query_result.index)
ax.set_yticklabels(query_result['Title'])
ax.set_title("Avg Salary by Job Title", weight='bold', size=16)
fig.tight_layout()
plt.show()
```



Now, per the last item requested by the boss, look up employee id

Look up Employee ID = 499942

In [6]:
```
# Create the sql statement
sql_statement = 'SELECT * FROM "Employees" WHERE emp_no = 499942'

# Grab the results into a dataframe and display
my_info  = pd.read_sql_query(sql_statement, connection)
my_info
```

Out[6]:

	emp_no	birth_date	first_name	last_name	gender	hire_date
0	499942	1963-01-10	April	Foolsday	F	1997-02-10

NOTE the fact that suspicians were correct and the data is fake.