SQL QUERIES WITH DISCUSSION AND ANALYSIS

After creating the schema diagram, we randomized our synthetic data through Excel and only filled up the tables that we needed for our analysis. We then imported these files into our schema and forward engineer it in order to get a running database. Once we successfully implemented our database, we were able to create queries that we needed for our analysis.

For this project, we ran three queries and performed analysis for the first two queries, as per project requirement. The following sections will cover the queries, queries' outcome along with their respective analysis and future works.

Query 1*:* Fetch all the attributes from the Hospital Database schema which influence the Doctor's Salary and arrange the output based on employeeID. Also, display the Doctor's first and last name along with their Department.

```
Select e.Employee_ID, e.Fname, e.Lname, e.Salary,
DATE_FORMAT(FROM_DAYS(DATEDIFF(NOW(), e.DOB)), '%Y') + 0 AS age,
e.Department_Dept_Code as dept,
d.speciality, d.yearsofexperience,
count(distinct a.patient_patient_id) as patient_count,
count(p.MEDICATION_Medication_Code) as medication_count
from EMPLOYEE as e, DOCTOR as d , APPOINTMENT  as a, Prescriptions_has_MEDICATION as p
where e.Employee_ID = d.EMPLOYEE_Employee_ID
and d.EMPLOYEE_Employee_ID = a.DOCTOR_EMPLOYEE_Employee_ID
and d.EMPLOYEE_Employee_ID = p.Prescriptions_DOCTOR_EMPLOYEE_Employee_ID
group by d.EMPLOYEE_Employee_ID;
```

We used a *SELECT* statement to retrieve employee_id, first_name, last_name, salary, and department from the Employee table. For age, we retrieved the date_of_birth from the employee table and used the datediff() function to fetch the age. We also fetched specialty and years_of_experience from the Doctor table with the condition that the employee_id in the employee table matches the employee_id in the doctor table. Using the count() function, we fetched the number of patients seen by each doctor as well as the medication given by the Doctor.

Output of Query 1:

Employee_ID	Fname	Lname	Salary	age	dept	speciality	yearsofexperien...	patient_cou...	medication_co...
1234	Bruce	Banner	150000	33	CARDIO	Cardiology	10	3	9
1235	Stephen	Strange	152000	36	A&E	Accident and emergency	15	1	1
1237	Leonard	McCoy	154000	32	DEN	Dental	13	1	1
1240	Doogie	Howser	164000	45	NEURO	Neurology	21	2	4
1241	Cristina	Yang	171000	47	SUR	Surgery	4	1	1
1243	Malcolm	Sayer	132000	52	A&E	Accident and emergency	7	1	1
1245	Hubert	Bombay	122000	34	INFECD	Infectious disease	10	1	1
1247	Erin	Mears	170000	37	A&E	Accident and emergency	11	2	4
1250	Richard	Kimble	154000	34	CARDIO	Cardiology	5	2	4
1251	Abraham	Helsing	122000	32	ICU	Intensive Care	7	1	1
1252	Frederick	Franke...	146000	39	ICU	Intensive Care	8	1	1
1253	Johann	Georg	164000	37	ONCO	Oncology	9	1	1
1254	Charles	McNider	163000	50	OPH	Ophthalmology	2	1	1
1255	Elliot	Tolliver	161000	49	PSY	Psychiatry	12	1	1

Query 1 Discussion: Statistical Analysis

For the first analysis, we aimed to predict the salary for doctors in this new hospital and understand what significant attributes might want to be considered in order to properly compensate them. The dataset was retrieved from the first query we ran, with its attributes consisting of employee id, first name, last name, salary, age, department, specialty, years of experience, count of patients treated, and count of medications prescribed. We ran this in Python as a multilinear linear regression model and chose to drop the employee id, first name, and last name variables since they are not relevant measurements to work ethic. We assigned salary as our dependent variable with age, department, specialty, years of experience, count of patients treated, and count of medications prescribed as our independent variables. Finally, we gave department and specialty categorical values through integer encoding. In doing so, we achieved an R-square value of 0.607.

```
lreg2 = smf.ols(formula = 'Salary ~  age  +   yearsofexperience  +   patient_count  +\
         medication_count  + dept + speciality', data = df_train).fit()
print(lreg2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Salary   R-squared:                       0.607
Model:                            OLS   Adj. R-squared:                  0.017
Method:                 Least Squares   F-statistic:                     1.030
Date:                Wed, 14 Dec 2022   Prob (F-statistic):              0.513
Time:                        12:29:35   Log-Likelihood:                -115.07
No. Observations:                  11   AIC:                             244.1
Df Residuals:                       4   BIC:                             246.9
Df Model:                           6
Covariance Type:            nonrobust
=====================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------
Intercept          6.758e+04   4.68e+04      1.444      0.222   -6.24e+04    1.98e+05
age                 801.7808   1006.003      0.797      0.470   -1991.332    3594.894
yearsofexperience  -485.8934   1121.405     -0.433      0.687   -3599.413    2627.626
patient_count      6.794e+04   5.61e+04      1.211      0.293   -8.78e+04    2.24e+05
medication_count    -1.6e+04   1.47e+04     -1.090      0.337   -5.67e+04    2.47e+04
dept              -9023.2513   1.02e+04     -0.889      0.424   -3.72e+04    1.92e+04
speciality         9996.0579   1.06e+04      0.943      0.399   -1.94e+04    3.94e+04
==============================================================================
Omnibus:                        0.539   Durbin-Watson:                   2.065
Prob(Omnibus):                  0.764   Jarque-Bera (JB):                0.563
Skew:                          -0.285   Prob(JB):                        0.755
Kurtosis:                       2.049   Cond. No.                         713.
==============================================================================
```

Lastly, we realized that some future works can be implemented. We noticed that the R-square was not that high but was fairly justifiable since we only randomized a small dataset for it. In addition, the p-values were not that high as well. Potential future work for this is considering a larger dataset with values that are more representative of the overall distribution of the doctors.

***Query 2**:* Obtain the Number of Patients and Sum of Cost that's Involved in Treatments for Each Procedure.

```
SELECT proc_id, proc_name, COUNT(DISTINCT pat_id) cnt_pat, SUM(proc_cost) sum_cost
FROM (
    SELECT  tp.TREATMENT_Treatment_ID trea_id,
            tp.PROCEDURE_Procedure_Code proc_id,
            p.Procedure_Name proc_name,
            p.Procedure_Cost proc_cost,
            t.PATIENT_Patient_ID pat_id,
            t.Treatment_Time trea_time
    FROM TREATMENT_contains_PROCEDURE tp, `PROCEDURE` p, Treatment t
    WHERE tp.PROCEDURE_Procedure_Code = p.Procedure_Code
    AND tp.TREATMENT_Treatment_ID = t.Treatment_ID
) t
GROUP BY proc_id;
```

In this case, we want to summarize the frequencies and revenue generated by different procedures and see the most common procedures in this hospital. We reached three tables, Treatment, Procedure, and their relationship table TREATMENT_contains_PROCEDURE, and then connected them by Treatment_ID and Procedure_Code using inner join. Putting this in the subquery, we used aggregate functions, COUNT and SUM to compute the number of patients and total cos, which are grouped by Procedure_ID.

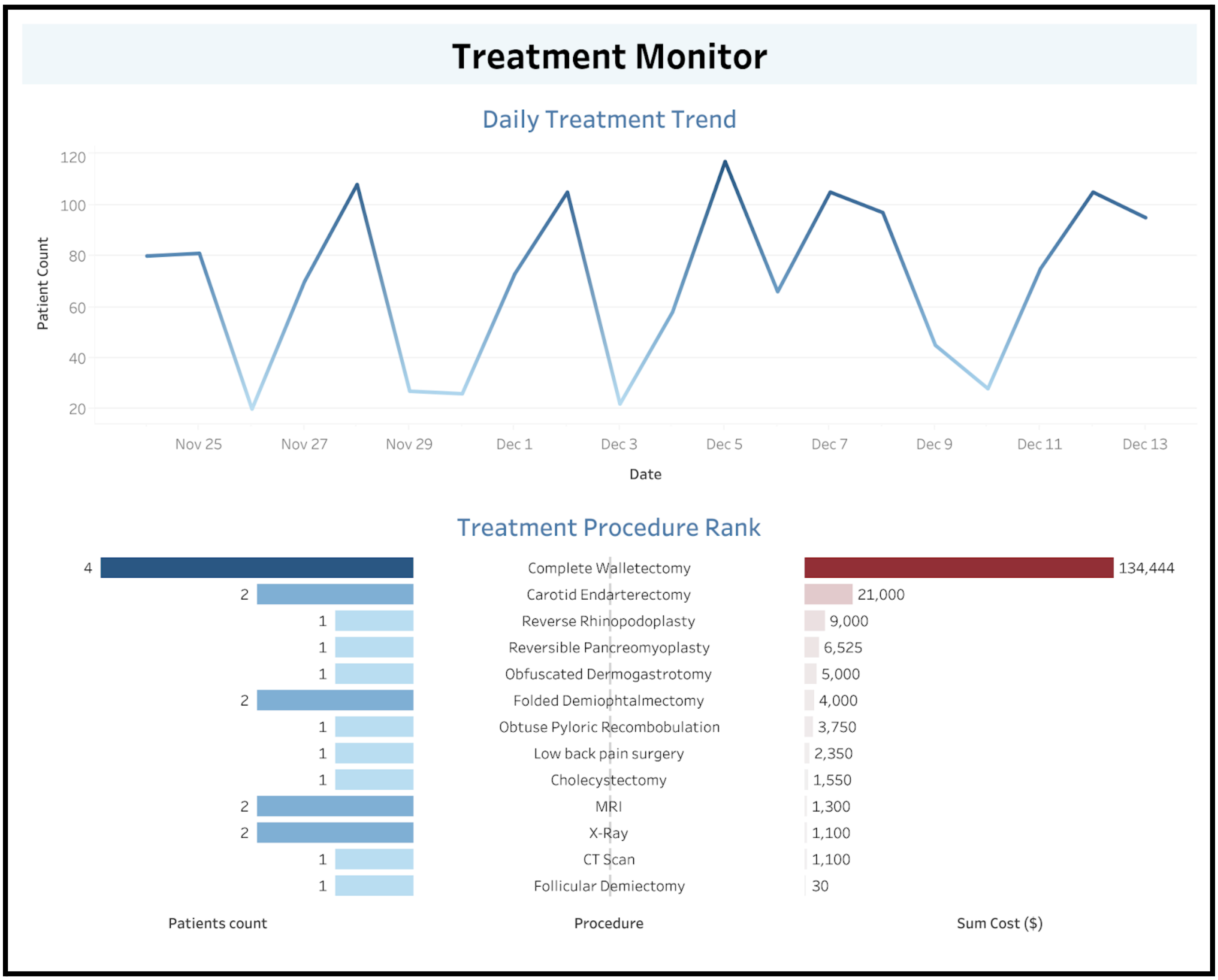
Output of Query 2:

proc_id	proc_name	cnt_pat	sum_cost
100	Reverse Rhinopodoplasty	1	9000
101	Obtuse Pyloric Recombobulation	1	3750
102	Folded Demiophtalmectomy	2	4000
103	Complete Walletectomy	4	134444
104	Obfuscated Dermogastrotomy	1	5000
105	Reversible Pancreomyoplasty	1	6525
106	Follicular Demiectomy	1	30
108	Carotid Endarterectomy	2	21000
110	Cholecystectomy	1	1550
111	Low back pain surgery	1	2350
112	X-Ray	2	1100
113	CT Scan	1	1100
114	MRI	2	1300

Query 2 Discussion: Business Intelligence Analysis

Based on the summary metrics about treatments and procedures, we can visualize them through graphics and build a dashboard to monitor the operations of treatments. For example, we built a mini dashboard in Tableau as follows, which is limited by our small sample database. The first graphic tracks the daily trend of patient amounts taking treatments. The second butterfly bar chart shows the most common procedure ordered by patient counts and their cost.

The hospital staff can check the workload of treatments on a regular basis and optimize their operations on equipment, rooms, employee deployment, materials, etc. To realize such a more comprehensive and detailed monitoring in the further work, the client can expand their database and add more dimensions into the tables for visualization, such as departments, date, room, etc. In this way, the dashboard can include a variety of filters based on these dimensions and achieve interactive functions for users to get their customized analysis at will.



Query 3*:* The Hospital is trying to restock on medication and would like to know the usage of Medication drugs for Medication_Code from 9 to 15

```sql
select m.Medication_Code, m.Drug_Generic_Name, m.Brand_Name, m.Description,
count(m.Medication_Code) as Medication_Count
from MEDICATION as m, Prescriptions_has_MEDICATION as pm
where m.Medication_Code = pm. MEDICATION_Medication_Code
and pm. MEDICATION_Medication_Code in
    (select MEDICATION_Medication_Code
    from Prescriptions_has_MEDICATION
    where MEDICATION_Medication_Code between 9 and 15)
group by m.Medication_Code;
```

We used the **nested** *SELECT* statement to retrieve medication_code, drug_generic_name, brand_name, description and usage of medication from the Medication and Prescriptions_has_Medication table. To get the usage, we used the count() function and group by medication_code with 2 conditions. Firstly, the medication_Code in Medication matches the medication_Code from Prescriptions_has_Medication table. Secondly, we used another *SELECT* statement to filter the drug_generic_name based on the medication_code between 9 and 15.

Output of Query 3

Medication_Code	Drug_Generic_Name	Brand_Name	Description	Medication_Count
9	Survivin	IEOR Labs	Helps in anxiety attacks	1
10	Passin	IEOR Labs	Helps from failing in life and feel less depressed	1
11	PlsgivgudmaRX	S.Liu Pharm	Helps in passing all things in life	2
13	Wemadeit	IEOR Pharm	Relieves all financial burden	1
14	Manalytx	IEOR Pharm	Helps for studying for comprehesive exams	1
15	CurveurmaRX	S.Liu Pharm	Helps solve all patients problem	1

NORMALIZATION ANALYSIS

Given the “Treatment scenario” below, we completed normalization in three versions: First Normal Form (1NF), Second Normal Form (2NF), as well as Third Normal Form (3NF).

Each treatment receives a treatment ID when it is given by a doctor and assigned to a patient. Each doctor has a unique doctor_ID just as each patient gets a unique patient_ID. The doctor decides the treatment for the patient and arranges a room for the treatment during the appointment, with each appointment recorded with its own unique appointment ID. The hospital keeps a record of which doctor and patient attend each appointment.

The doctor records the starting time of the treatment once the appointment finishes.

Relation:
R(Treatment_ID, Doctor_ID, Doctor_Name, Patient_ID, Patient_Name, Appointment_ID, Treatment_Start_Time, Treatment_Room)

Functional Dependency (FD):
Appointment_ID → {Doctor_ID, Patient_ID}
Doctor_ID → {Doctor_Name}
Patient_ID → {Patient_Name}
{Treatment_ID} → {Treatment_Room, Treatment_Start_Time}

Discussion and Analysis:
From the above information, we can see that the two attributes both depend on treatment_ID, so the relation violates the 2NF, specifically because we disallow partial dependencies for non-prime attributes. The Doctor_Name and Patient_Name are both dependent on Doctor_ID and Patient_ID and both can be derived from the Appointment_ID. Therefore, the relation violates the 3NF, because transitive dependencies are not allowed.

$\{\text{Treatment\_ID}, \text{Appointment\_ID}\}^{+F}$ = {Treatment_ID, Doctor_ID, Doctor_Name, Patient_ID, Patient_Name, Appointment_ID, Treatment_Start_Time, Treatment_Room}

1. Appointment (Appointment_ID, Doctor_ID[2], Patient_ID[3])
2. Doctor (Doctor_ID, Doctor_Name)
3. Patient (Patient_PD, Patient_Name)
4. Treatment (Treatment_ID, Appointment_ID[1],Treatment_Room, Treatment_Start_Time)