MySql Project

By Joshua Nguyen

-- Simple and Join Queries

-- Skills used: Recursive functions, CTE's, Temp Tables, Joins, Aggregate Functions, Subqueries

```sql
-- A1. Return all nurses
SELECT *
FROM nurse;

-- A2. Return the number of patients who booked an appointment with at l
east one Physician.
SELECT count(distinct patient)
FROM appointment;

-- A3. Return the physicians who are the head of a department.
SELECT concat(phys.firstname, ' ' ,phys.surname) AS Physicians, dept.nam
e as "Department"
FROM physicians phys,
department dept
WHERE phys.EmployeeID = dept.PhysicianHead;

-- A4. Return all physicians and the departments they are affiliated wit
h using JOIN functions.
SELECT concat(phys.firstname, ' ', phys.surname) AS "Physician", dept.na
me AS "Department"
FROM physicians phys
JOIN affiliated_with aff
ON phys.EmployeeID =  aff.physicianID
JOIN department dept
ON aff.department = dept.departmentID;

-- A5. Return those physicians who have trained for special treatment.
SELECT concat(phys.firstname, ' ', phys.surname) AS "Physician", pro.nam
e AS "Procedure Name"
FROM physicians phys
JOIN trained_in tra
ON phys.EmployeeID = tra.physicianID
JOIN procedures pro
ON tra.treatmentcode = pro.code;

-- A6. Return those physicians who are not a specialized physician.
SELECT concat(phys.firstname, ' ', phys.surname) AS "Physician", phys.po
sition
FROM physicians phys
JOIN trained_in train
ON phys.EmployeeID = train.physicianID
WHERE train.treatmentcode IS NULL
ORDER BY phys.EmployeeID;
-- Lucky in our hopsital, all of our physiicans are trained in a procedu
re :)

-- A7. Return every patients and the number of apppointments they have t
aken with a physician in our hospital.
SELECT concat(ptn.firstname, ' ', ptn.surname) AS "Patient", count(app.p
```

```
atient) AS "Number of Appointments with a Physician in Our Hospital"
FROM patients ptn
JOIN appointment app
ON ptn.MRN = app.patient
GROUP BY ptn.MRN
HAVING count(app.patient) >=1;

-- A8. Return those patients who have a scheduled appointment in Novembe
r 2022.
SELECT *
FROM patients ptn
JOIN appointment appt
ON ptn.MRN = appt.patient
WHERE appt.date LIKE '2021-11%';

-- A9. Return every patient with their PCP and medications prescribed.
SELECT concat(ptn.firstname, ' ', ptn.surname) AS "Patient", concat(phys
.firstname, ' ', phys.surname) AS "PCP", pres.name AS "Prescribed This M
edication:"
FROM medications meds
JOIN patients ptn
ON meds.patient = ptn.MRN
JOIN physicians phys
ON meds.physicianID = phys.EmployeeID
JOIN prescription pres
ON meds.medication = pres.code
WHERE ptn.PCP = phys.EmployeeID;

-- A10. Return those physicians who performed a medical procedure, but t
hey are not certified to perform
SELECT concat(physicians.firstname, ' ', physicians.surname) AS "Physici
an"
FROM physicians
WHERE physicians.EmployeeID IN
(
SELECT undergoes.physician
FROM undergoes
INNER JOIN trained_in
ON undergoes.physician = trained_in.physicianID
AND undergoes.procedure = trained_in.treatmentcode
WHERE trained_in.treatmentcode IS NULL
); -- Luckily, all of our procedures have been performed by physicians t
rained in them...

-- A11. Return those physicians who completed a medical procedure their
 certification expiration date
SELECT concat(physicians.firstname, ' ', physicians.surname) AS "Physici
an", physicians.position
FROM physicians
```

```
JOIN undergoes
ON physicians.EmployeeID = undergoes.physician
JOIN trained_in
ON undergoes.procedure = trained_in.treatmentcode
AND undergoes.physician = trained_in.physicianID
WHERE undergoes.date > trained_in.certificationexpires
; -- Lucikly, all of our procedures have been performed by physicians wi
th certification

-- Using Subqueries.
SELECT concat(physicians.firstname, ' ', physicians.surname) AS "Physici
an", physicians.position
FROM physicians
WHERE physicians.EmployeeID IN
(
SELECT undergoes.physician
FROM undergoes
WHERE undergoes.date IN
(
SELECT trained_in.certificationexpires
FROM trained_in
JOIN undergoes
ON trained_in.physicianID = undergoes.physician
AND trained_in.treatmentcode = undergoes.procedure
)
);

-- A12. Return patients who have been undergone a procedure costing more
than $5,000 and the name of the physician who has carried out the surger
y.
SELECT concat(patients.firstname, ' ', patients.surname) AS "Patient", c
oncat(physicians.firstname, ' ', physicians.surname) AS "Physician", pro
cedures.name, procedures.cost
FROM patients
JOIN undergoes ON patients.MRN = undergoes.patient
JOIN physicians ON physicians.EmployeeID = undergoes.physician
JOIN procedures ON procedures.code = undergoes.procedure
WHERE procedures.cost > 5000;

-- A13. Return patients and the physicians who are not their PCP but hav
e perscribed the patient at least one medication
SELECT concat(patients.firstname, ' ', patients.surname) AS "Patient", c
oncat(physicians.firstname, ' ', physicians.surname) AS "Physician", pre
scription.name, prescription.description
FROM patients
JOIN medications ON patients.MRN = medications.patient
JOIN physicians ON medications.physicianID = physicians.EmployeeID
JOIN prescription ON prescription.code = medications.medication
WHERE patients.PCP != physicians.EmployeeID;
```

```sql
-- Self-Verification
SELECT concat(patients.firstname, ' ', patients.surname) AS "Patient", c
oncat(physicians.firstname, ' ', physicians.surname) AS "PCP"
FROM patients
JOIN physicians
ON patients.PCP = physicians.EmployeeID
WHERE patients.firstname = 'Ruiz'; -- PCP == 'Vinicius Korhonen'

-- A14. Return list of patients who have undergone an Inguinal Hernia Re
pair
SELECT concat(patients.firstname, ' ', patients.surname) AS "Patient"
FROM patients
INNER JOIN undergoes
ON patients.MRN = undergoes.patient
INNER JOIN procedures
ON undergoes.procedure = procedures.code
WHERE procedures.name = 'Inguinal Hernia Repair';

-- A15/A16. Return the most and second most expensive medications avalia
ble in our hospital
-- Most Expensive
SELECT MAX(cost)
FROM prescription;

-- Second Most Expensive
SELECT *
FROM prescription
WHERE cost =
(
SELECT MAX(cost)
FROM prescription
WHERE cost <
(
SELECT MAX(cost)
FROM prescription
)
);

-- A17. Return list of patients who have undergone the procedure that co
sts the second lowest
SELECT * FROM patients;
SELECT * FROM undergoes;
SELECT * FROM procedures;

-- Query to obtain second lowest costing procedure
SELECT code
FROM procedures
WHERE cost =
```

```
(
SELECT MIN(cost)
FROM procedures
WHERE cost >
(
SELECT MIN(cost)
FROM procedures
)
);

-- Answering the Prompt
SELECT concat(ptns.firstname, ' ', ptns.surname) AS patient
FROM patients ptns
JOIN undergoes ug
ON ptns.MRN = ug.patient
WHERE ug.procedure =
(
SELECT code
FROM procedures
WHERE cost =
(
SELECT MIN(cost)
FROM procedures
WHERE cost >
(
SELECT MIN(cost)
FROM procedures
)
)
);

-- A18. Return count of patients whose surname starts with each letter o
f the alphabet
SELECT SUBSTR(surname,1,1) AS letter, count(*) AS count
FROM patients
GROUP BY letter
ORDER BY letter;

-- Aggregate Functions
-- B1. Return the number of primary physicians in each department
SELECT primaryaffiliation AS "department", count(distinct physicianID) F
ROM affiliated_with
GROUP BY primaryaffiliation;

-- B2. Compute and return the sum of patients seen in all departments
SELECT dept.name, count(undergoes.patient)
FROM department dept
JOIN affiliated_with aff
ON dept.departmentID = aff.primaryaffiliation
```

```sql
JOIN undergoes
ON undergoes.physician = aff.physicianID
GROUP BY dept.departmentID;

-- B3. Compute and return the average price of procedures done by physic
ians in their primary affiliated department
SELECT dept.name, AVG(pro.cost)
FROM department dept
JOIN affiliated_with aff
ON dept.departmentID = aff.primaryaffiliation
JOIN undergoes ug
ON aff.physicianID = ug.physician
JOIN procedures pro
ON ug.procedure = pro.code
GROUP BY dept.name;

-- B4. Return the number of procedures by the combination of date and ph
ysican while returning the date, physicianID and count in that order
SELECT undergoes.date, physician, count(*)
FROM undergoes
GROUP BY undergoes.date, undergoes.physician;

-- B5. Return the most expensive procedures performed by each physican
SELECT physicians.EmployeeID, MAX(procedures.cost)
FROM physicians
JOIN undergoes
ON physicians.EmployeeID = undergoes.physician
JOIN procedures
ON undergoes.procedure = procedures.code
GROUP BY physicians.EmployeeID;

-- B5. Return the most expensive procedures performed for each patient
SELECT patients.MRN, MAX(procedures.cost), procedures.name
FROM patients
JOIN undergoes
ON patients.MRN = undergoes.patient
JOIN procedures
ON undergoes.procedure = procedures.code
GROUP BY patients.MRN;

-- B6. Return total number of appointments made in each department accor
ding to the physican's primary affiliation in the year of 2021. Sort the
results by department name/ID and then month.
SELECT department.departmentID, department.name, EXTRACT(MONTH FROM appo
intment.date) as "Month Number", count(appointment.appointmentID) as "Nu
mber of Appointments"
FROM department
JOIN affiliated_with
ON department.departmentID = affiliated_with.primaryaffiliation
```

```
JOIN appointment
ON affiliated_with.physicianID = appointment.physician
WHERE EXTRACT(YEAR FROM appointment.date) = 2021
GROUP BY department.departmentID, EXTRACT(MONTH FROM appointment.date)
ORDER BY department.departmentID, EXTRACT(MONTH FROM appointment.date);

-- Subqueries Functions
-- C1. Return all physicians who have performed on a patient living in a
n address starting with the number 9
SELECT distinct undergoes.physician
FROM undergoes
WHERE undergoes.patient IN
(
SELECT MRN
FROM patients
WHERE address LIKE '9%'
);

-- C2. Return all procedures with costs greater than that of an Appendec
tomy
SELECT name, cost
FROM procedures
WHERE cost >
(
SELECT cost
FROM procedures
WHERE name = 'Appendectomy'
);

-- C3. Return all procedures that cost more than the average procedures
 cost done in the month of October in 2018.
SELECT name, cost
FROM procedures
WHERE procedures.cost > (
SELECT AVG(cost)
FROM procedures
WHERE procedures.code IN (
SELECT undergoes.procedure
FROM undergoes
WHERE undergoes.date LIKE '2018-10%'
)
);

-- C4. Return all physicians who only operated/performed on one or less
 patient
SELECT  * FROM undergoes;
SELECT concat(firstname, ' ', surname) AS "Physician"
FROM physicians
WHERE physicians.EmployeeID IN
```

```
(
SELECT physician
FROM undergoes
GROUP BY physician
HAVING count(patient) < 2
);

-- Alternative solution
SELECT physicians.firstname, physicians.surname
FROM physicians
WHERE 1 =
(
SELECT count(*)
FROM undergoes
WHERE undergoes.physician = physicians.EmployeeID
);

-- More Intensive Queries

-- D1. Return list of all medications brought by patients in our hospita
l of all time along with the total cost while remembering that patients
 who were prescribed a medication from a physician that is not their PCP
will cost twice as much. In our case, the cost of the medication is per
 single 1 unit dose.

-- List of patients and their PCP
SELECT MRN, PCP
FROM patients;

-- List of patients and MD where prescriptions were made by PCP
SELECT physicianID, patient
FROM medications
WHERE (patient, physicianID) IN
(
SELECT MRN, PCP
FROM patients
);

-- List of patients and MD where presctiptions were not made by PCP
-- I should be expecting a total of count(*) FROM medications minus size
of query above or 2 which is equal to 14 -2 = 12.
SELECT physicianID, patient
FROM medications
WHERE (patient, physicianID) NOT IN
(
SELECT MRN, PCP
FROM patients
);
```

```sql
-- Answering the Prompt
SELECT pres.name,
sum(med.dose * pres.cost *
    CASE
        WHEN
            (med.physicianID, med.patient) IN
                (SELECT physicianID, patient
                        FROM medications
                        WHERE (patient, physicianID) IN
                        (
                        SELECT MRN, PCP
                        FROM patients
                        )
            ) THEN 1
        ELSE 2
        END) AS total_cost
FROM prescription pres
LEFT JOIN medications med
ON pres.code = med.medication
GROUP BY pres.code
ORDER BY pres.code;
-- Notice that our ED medication made the most profit out of all medicat
ions.

-- D2. Return list of all medications brought by patients in our hospita
l of all time along with the total cost being less than 100,000 remember
ing that patients who were prescribed a medication from a physician that
is not their PCP will cost twice as much. In our case, the cost of the m
edication is per single 1 unit dose.
SELECT pres.name,
sum(med.dose * pres.cost *
    CASE
        WHEN
            (med.physicianID, med.patient) IN
                (SELECT physicianID, patient
                        FROM medications
                        WHERE (patient, physicianID) IN
                        (
                        SELECT MRN, PCP
                        FROM patients
                        )
            ) THEN 1
        ELSE 2
        END) AS Total_cost
FROM prescription pres
LEFT JOIN medications med
ON pres.code = med.medication
GROUP BY pres.code
HAVING Total cost < 100000
```

```
ORDER BY pres.code;

-- Sometimes physicians will refer patients to a different physician for
continuity of care. Lets create a new column in the physicians table nam
ed "referred_ptn_to_this_md" indicating a single unique physican referra
l for each physican in our hospital. Ideally, in real life, we should cr
eate a new table as referrals by definition do not have to be unique. Bu
t lets continue by making a rule that each physcian can only refer patie
nts to a single physician.

-- Adding New column
ALTER table physicians
ADD COLUMN refers_ptn_to_this_md INT;

-- Adding referrals
UPDATE physicians
SET refers_ptn_to_this_md = 300000005
WHERE EmployeeID = 300000006;

UPDATE physicians
SET refers_ptn_to_this_md = 300000001
WHERE EmployeeID = 300000003;

UPDATE physicians
SET refers_ptn_to_this_md = 300000008
WHERE EmployeeID = 300000007;

UPDATE physicians
SET refers_ptn_to_this_md = 300000002
WHERE EmployeeID = 300000008;

SELECT * FROM physicians;

-- Using Recursive, return the upward referral chain for physician 30000
0003, i.e., find which physician refers to this physican and the physica
n who refers to the former, ect.
WITH RECURSIVE referrals(Referral_from) AS
(
SELECT refers_ptn_to_this_md
    FROM physicians
        WHERE EmployeeID = 300000003
UNION ALL
SELECT physicians.refers_ptn_to_this_md
    FROM referrals
        INNER JOIN physicians
            ON physicians.EmployeeID = referrals.referral_from
)
SELECT referrals.Referral_from, concat(physicians.firstname, ' ' ,physic
ians.surname) AS physician name
```

```
    FROM referrals
        INNER JOIN physicians
            ON referrals.referral_from = physicians.EmployeeID
ORDER BY physicians.EmployeeID DESC;

-- Using Recursive, return the downward referral chain for physician 300
000002, i.e., find which physician refers to this physican and the physi
can who refers to the former, ect.
WITH RECURSIVE referrals(EmployeeID) AS
(
SELECT EmployeeID
    FROM physicians
        WHERE refers_ptn_to_this_md = 300000002
UNION ALL
SELECT physicians.EmployeeID
    FROM referrals
        INNER JOIN physicians
            ON referrals.EmployeeID = physicians.refers_ptn_to_this_md
)
SELECT referrals.EmployeeID, concat(physicians.firstname, ' ' ,physician
s.surname) AS physician_name
    FROM referrals
        INNER JOIN physicians
            ON referrals.EmployeeID = physicians.EmployeeID
ORDER BY referrals.EmployeeID;

-- Create a CTE that returns the upward referral chain for any physician
in our hospital. Test out the CTE for physicians 300000006 and 30000000
3.
WITH RECURSIVE referrals(Referral_from, EmployeeID) AS
(
SELECT refers_ptn_to_this_md, EmployeeID
    FROM physicians
UNION ALL
SELECT physicians.refers_ptn_to_this_md, physicians.EmployeeID
    FROM referrals
        INNER JOIN physicians
            ON physicians.EmployeeID = referrals.referral_from
)
SELECT referrals.EmployeeID, referrals.referral_from, concat(physicians.
firstname, ' ' ,physicians.surname) AS physician_name
    FROM referrals
        INNER JOIN physicians
            ON referrals.referral_from = physicians.EmployeeID
    WHERE referrals.EmployeeID = 300000006 or referrals.EmployeeID = 300
000003
ORDER BY referrals.EmployeeID ASC, referrals.referral_from DESC;
```