Database Lab Report 4

Course: Database Management
Lab Number: *Lab #4*
Date: *2025-02-11*
Name: *Ryan Munger*

1. Objective

To write some interesting SQL queries using subqueries and set operations. Please do not use joins; save them for the next assignment!

2. Lab Setup

A fresh CAP database.

3. Procedure

Part 1: *Subqueries*

Use subqueries to answer the following questions. Then, use AI and grade its responses.

1. Get all the People data for people who are customers.

```
-- Me
SELECT *
FROM People
WHERE pid IN (SELECT pid
              FROM Customers);
-- AI
SELECT *
FROM People
WHERE pid IN (SELECT pid FROM Customers);
```

Grade: 9.9/10: No indents!

2. Get all the People data for people who are agents.

```
-- Me
SELECT *
FROM People
WHERE pid IN (SELECT pid
              FROM Agents);
-- AI
SELECT *
```

```
FROM People
WHERE pid IN (SELECT pid FROM Agents);
```

Grade 9.9/10

3. Get all of People data for people who are both customers and agents.

```
-- Me - queryception, I know I could have used AND
SELECT *
FROM People
WHERE pid IN (SELECT pid
              FROM Customers
              WHERE pid IN (SELECT pid
                            FROM Agents));
-- AI
SELECT *
FROM People
WHERE pid IN (SELECT pid FROM Customers)
  AND pid IN (SELECT pid FROM Agents);
```

Grade: 10/10 but I had more fun.

4. Get all of People data for people who are neither customers nor agents.

```
-- Me
SELECT *
FROM People
WHERE pid NOT IN (SELECT pid FROM Customers)
  AND pid NOT IN (SELECT pid FROM Agents);
-- AI
SELECT *
FROM People
WHERE pid NOT IN (SELECT pid FROM Customers)
  AND pid NOT IN (SELECT pid FROM Agents);
```

Grade 10/10: ditto

5. Get the ID of customers who ordered either product p01 or p03 (or both). List the IDs in order from lowest to highest. Include each ID only once.

```
-- Me
SELECT DISTINCT custId
FROM Orders
WHERE prodId = 'p01' OR prodId = 'p03'
Order by custId ASC;
-- AI
SELECT DISTINCT custId
```

```
FROM Orders
WHERE prodId = 'p01' OR prodId = 'p03'
ORDER BY custId ASC;
```

Grade: 10/10, as good as human

6. Get the ID of customers who ordered both products p01 and p03. List the IDs in order from highest to lowest. Include each ID only once.

```
-- Me
SELECT DISTINCT custId
FROM Orders
WHERE custId IN (SELECT custId
                 FROM Orders
                 WHERE prodId = 'p01')
  AND custId IN (SELECT custId
                 FROM Orders
                 WHERE prodId = 'p03')
Order by custId DESC;
-- AI
SELECT custId
FROM Orders
WHERE prodId = 'p01'
INTERSECT
SELECT custId
FROM Orders
WHERE prodId = 'p03'
ORDER BY custId DESC;
```

Grade: 9/10 I would have preferred to see a subquery

7. Get the first and last names of agents who sold products p05 or p07 in order by last name from A to Z.

```
-- Me
SELECT firstName, lastName
FROM People
WHERE pid IN (SELECT pid
              FROM Agents
              WHERE pid IN (SELECT agentId
                            FROM Orders
                            WHERE prodId = 'p05' OR prodId = 'p07'))
Order by lastName ASC;
-- AI
SELECT firstName, lastName
FROM People
WHERE pid IN (SELECT agentId FROM Orders WHERE prodId = 'p05' OR prodId =
'p07')
ORDER BY lastName ASC;
```

Grade: 9.9/10 where are Alan's indents?

8. Get the home city and birthday of agents booking an order for the customer whose pid is 007, sorted by home city from Z to A.

```
-- Me
SELECT homeCity, DOB
FROM People
WHERE pid IN (SELECT pid
              FROM Agents
              WHERE pid IN (SELECT agentId
                            FROM Orders
                            WHERE custId = 7))
Order by homeCity DESC;
-- AI
SELECT homeCity, DOB
FROM People
WHERE pid IN (SELECT agentId FROM Orders WHERE custId = '007')
ORDER BY homeCity DESC;
```

Grade: 10/10: It did better than me as I did not need to go through the agents table.

9. Get the unique ids of products ordered through any agent who takes at least one order from a customer in Saginaw, sorted by id from highest to lowest. (This is not the same as asking for ids of products ordered by customers in Saginaw.)

```
-- Me
SELECT DISTINCT prodId
FROM Orders
WHERE agentId IN (SELECT agentId
                  FROM Orders
                  WHERE custId IN (SELECT pid
                                   FROM People
                                   WHERE homeCity = 'Saginaw'))
Order by prodId DESC;
-- AI
SELECT DISTINCT prodId
FROM Orders
WHERE agentId IN (SELECT pid FROM Agents WHERE pid IN (SELECT agentId FROM
Orders WHERE custId IN (SELECT pid FROM People WHERE homeCity =
'Saginaw')))
ORDER BY prodId DESC;
```

Grade: 10/10: So it goes through the Agents table as soon as I don't?

10. Get the last name and home city for all customers who place orders through agents in Regina or Pinner in order by last name from A to Z.

```
-- Me
SELECT lastName, homeCity
FROM People
WHERE pid IN (SELECT pid
              FROM Customers
              WHERE pid IN (SELECT custId
                            FROM Orders
                            WHERE agentId IN (SELECT pid
                                              FROM People
                                              WHERE homeCity = 'Regina' OR
homeCity = 'Pinner')))
Order by lastName ASC;
-- AI
SELECT lastName, homeCity
FROM People
WHERE pid IN (SELECT custId FROM Orders WHERE agentId IN (SELECT pid FROM
Agents WHERE pid IN (SELECT pid FROM People WHERE homeCity = 'Regina' OR
homeCity = 'Pinner')))
ORDER BY lastName ASC;
```

Grade: We both had unnecessary subqueries upon closer inspection. That's one long WHERE line though...

Overall Grade: 10/10. I am surprised that a generally clowned on model like Gemini is performing so well even as the queries become more complex.