Deliverable #: Lab-4/ Relational Algebra,SQL and Functional Dependencies

Data Management Course

UM6P College of Computing

Professor: Karima Echihabi **Program:** Computer Engineering
Session: Fall 2025

Team Information

Team Name	StarsDB
Member 1	Nouha Talssi
Member 2	Mariam Said
Member 3	Tilamsane Wissal
Member 4	Alae Sabir
Member 5	Amina Sidki
Repository Link	`https://github.com/StarsDB`

1 -Introduction

This deliverable aims to elaborate on the relational algebra expressions of the queries provided in Lab4 and their SQL implementations. The focus will be on understanding how to translate queries into relational algebra operations and then into SQL implemented queries effectively.Additionally, we will refine our MNHS database schema based on the functional dependencies identified in this lab, ensuring that our database design adheres to normalization principles and optimizes data integrity and efficiency and reduce redundancy of information.

2 -Requirements

- Express all queries mentioned in this lab using Relational Algebra (RA), applying concepts learned from Week 4 lectures.
- After completing the RA expressions, translate each one into SQL queries based on the database schema provided in the previous lab solution.
- Analyze the MNHS schema from the previous lab to identify functional dependencies.
- Examine each table in the schema and discuss the relationships between attributes to determine all functional dependencies present in the database structure.

3 -Methodology

In this lab, we followed a systematic approach to address the requirements outlined. First, we reviewed the queries provided in Lab4 and devided them into five parts, so that each team member could focus on specific queries. Each member then expressed their assigned queries using relational algebra and implemented them using SQL, ensuring that they applied the concepts learned from Week 4 lectures. After completing the RA expressions and the SQL implementations, we reconvened as a team to discuss and verify the accuracy of our work. Next, we analyzed the MNHS schema from the previous lab to identify functional dependencies. Each team member examined the tables in the schema and discussed the relationships between attributes to determine all functional dependencies present in the database structure. Finally, we compiled our findings and results into this deliverable document, ensuring clarity and coherence in our presentation.

4 Implementation & Results

- **QUERY1 :** Find the names of patients who have had at least one clinical activity handled by active staff:
  - **Relational Algebra:**

$$\rho\Big(\text{R1},\ \sigma_{\text{status='active'}}(\text{Staff})\Big)$$
$$\rho\Big(\text{R2},\ \text{ClinicalActivity} \bowtie \text{R1}\Big)$$
$$\rho\Big(\text{R3},\ \text{Patients} \bowtie \text{R2}\Big)$$
$$\pi_{\text{FullName}}(\text{R3})$$

- **SQL Code:**

```sql
    SELECT DISTINCT p.FullName
    FROM Patient p
    JOIN ClinicalActivity ca ON p.IID = ca.IID
    JOIN Staff s ON ca.STAFF_ID = s.STAFF_ID
    WHERE s.Status = 'Active';
```

- **QUERY2 :** Find Staff IDs of staff who are either 'Active' or have issued at least one prescription:
  - **Relational Algebra:**

$$\pi_{\text{staff\_ID}}\,(\sigma_{\text{status=''active''}}(\text{Staff}))\cup\pi_{\text{staff\_ID}}\,((\pi_{\text{CAID}}(\text{Perscription}))\bowtie \text{ClinicalActivity})$$

  - **SQL Code:**

```sql
SELECT staff_ID
FROM Staff
WHERE status = "active"
UNION
SELECT staff_ID
FROM
(SELECT CAID
FROM ClinicalActivity
NATURAL JOIN
(SELECT CAID
FROM Perscription));
```

- **QUERY3 :** Find Hospital IDs of hospitals located in 'Benguerir' or having at least one department with the specialty 'Cardiology':
  - **Relational Algebra:**

$$\pi_{\text{HID}}(\sigma_{\text{City='Benguerir'}}(\text{Hospital})) \cup \pi_{\text{HID}}(\sigma_{\text{Specialty='Cardiology'}}(\text{Department}))$$

- **SQL Code:**

```sql
SELECT HID
FROM Hospital
WHERE City = 'Benguerir'

UNION

SELECT D.HID
FROM Department
WHERE Specialty = 'Cardiology';
```

- **QUERY4 :** Find Hospital IDs of hospitals that have both 'Cardiology' and 'Pediatrics' departments:

  - **Relational Algebra:**

$$\pi_{\text{HID}}(\sigma_{\text{specialty='pediatrics'}}(\text{Departement}))\cap\pi_{\text{HID}}(\sigma_{\text{specialty='cardiology'}}(\text{Departement}))$$

  - **SQL Code:**

```sql
      SELECT HID
      FROM Department
      WHERE specialty='pediatrics'
      INTERSECT
      SELECT HID
      FROM Department
      WHERE specialty='cardiology';
```

- **QUERY5 :** Find staff members who have worked in every department of the hospital with Hid=1:

  - **Relational Algebra:**

$$\rho\Big(\text{dep\_hpt\_1}, \pi_{\text{DEP\_ID}}\big(\sigma_{\text{HID}=1}(\text{Departement})\big)\Big)$$

$$\pi_{\text{STAFF\_ID}}(\text{Work\_in}) \div \text{dep\_hpt\_1}$$

  - **SQL Code:**

```sql
SELECT DEP_ID
FROM (
    SELECT *
    FROM Departement
    WHERE HID = 1
) AS dep_hpt_1;
SELECT w.STAFF_ID
FROM Work_in w
JOIN dep_hpt_1 dh ON w.DEP_ID = dh.DEP_ID
GROUP BY w.STAFF_ID
HAVING COUNT(DISTINCT w.DEP_ID) = (SELECT COUNT(*) FROM
    dep_hpt_1);
```

- **QUERY6 :** Find staff members who participated in every clinical activity of the department with DEP ID = 2:
  - **Relational Algebra:**

$$\rho\Big(\mathrm{R},\ \pi_{\mathrm{Staff\_id,\ CAID}}(\sigma_{\mathrm{dep\_id=2}}(\mathrm{ClinicalActivity})\Big)$$
$$\rho\Big(\mathrm{S},\ \pi_{\mathrm{CAID}}(\mathrm{R})\Big)$$
$$(\mathrm{R} \div \mathrm{S}) \bowtie \mathrm{Staff}$$

  - SQL Code:

```sql
SELECT s.STAFF_ID, s.FullName
FROM Staff s
WHERE NOT EXISTS (
    SELECT ca.CAID
    FROM ClinicalActivity ca
    WHERE ca.DEP_ID = 2
    AND ca.CAID NOT IN (
        SELECT ca2.CAID
        FROM ClinicalActivity ca2
        WHERE ca2.STAFF_ID = s.STAFF_ID
        AND ca2.DEP_ID = 2
    )
);
```

- **QUERY7 :** Find pairs of staff members (s1, s2) such that s1 has handled more clinical activities than s2:
  - **Relational Algebra:**

$$\rho\Big(\mathrm{R},\ \gamma_{\mathrm{staff\_ID}}\ (\mathrm{COUNT(CAID)} \rightarrow \mathrm{nbCA})\ (\mathrm{ClinicalActivity})\Big)$$
$$\rho(\mathrm{S1}, R)$$
$$\rho(\mathrm{S2}, R)$$
$$\pi_{\mathrm{S1.staff\_ID,S2.staff\_ID}}\ (\sigma_{\mathrm{S1.nbCA>S2.nbCA}}\ (\mathrm{S1} \times \mathrm{S2}))$$

  - **SQL Code:**

```sql
WITH R AS(
SELECT Staff_ID, COUNT(CAID) AS nbCA
FROM ClinicalActivity
GROUP BY Staff_ID)
SELECT S1.staff_ID AS Staff_ID1 , S2.staff_ID AS
   Staff_ID2
FROM R S1 , R S2
WHERE S1.nbCA > S2.nbCA;
```

- **QUERY8 :** Find Patient IDs of patients who had clinical activities with at least two different staff members:
  - **Relational Algebra:**

$$\rho(R1, ClinicalActivity)$$
$$\rho(R2, ClinicalActivity)$$
$$\pi_{R1.IID}(\sigma_{R1.IID=R2.IID \wedge R1.STAFF\_ID \neq R2.STAFF\_ID}(R1 \times R2))$$

  - **SQL Code:**

```sql
SELECT DISTINCT CA1.IID
FROM ClinicalActivity CA1, ClinicalActivity CA2
WHERE CA1.STAFF_ID <> CA2.STAFF_ID and CA1.IID = CA2.IID;
```

- **QUERY9 :**Find CAIDs of clinical activities performed in September 2025 at hospitals located in “Benguerir” :
  - **Relational Algebra:**

$$\rho\Big(R_1, (ClinicalActivity \bowtie (Department \bowtie_{Department.HID=Hospital.HID} Hospital)))$$

$$\pi_{CAID}(\sigma_{Hospital.city='Benguerir' \wedge Date\ \text{BETWEEN}\ '2025-09-01'\ \text{AND}\ '2025-09-30'}(R_1))$$

  - **SQL Code:**

```sql
SELECT ca.CAID
FROM ClinicalActivity ca
JOIN Department d ON ca.DEP_ID = d.DEP_ID
JOIN Hospital h ON d.HID = h.HID
WHERE h.City = 'Benguerir'
AND ca.Date BETWEEN '2025-09-01' AND '2025-09-30';
```

- **QUERY10 :** Find staff ids of staff who have issued in more that one prescription :

- **Relational Algebra:**

$$\rho\Big(\pi_{\text{CAID}}(\text{Prescription}) \bowtie \text{ClinicalActivity}\Big)$$
$$\rho\Big(\gamma_{\text{STAFF\_ID}}(\text{COUNT}(\text{CAID}) \to \text{nbp})(R1)\Big)\Big)$$

$$\pi_{\text{STAFF\_ID}}\Big(\sigma_{\text{nbp}>1}(R)\Big)$$

- **SQL Code:**

```sql
SELECT STAFF_ID
FROM (
    SELECT ca.STAFF_ID, COUNT(DISTINCT p.CAID) AS nbp
    FROM Prescription p
    NATURAL JOIN ClinicalActivity ca
    GROUP BY ca.STAFF_ID
) AS R
WHERE nbp > 1;
```

- **QUERY11 :** List IIDs of patients who have scheduled appointements in more than one department :

  - Relational Algebra:

$$\rho\Big(R, \pi_{\text{CAID}}(\sigma_{\text{Status=''Scheduled''}}(\text{Appointment})) \bowtie \text{ClinicalActivity})$$

$$\rho\Big(R\_1, \gamma_{\text{IID;COUNT(DEP\_ID)}\to\text{depnum}}(R)\Big)$$

$$\pi_{\text{IID}}\left(\sigma_{\text{depnum}>1}(R1)\right)$$

  - **SQL Code:**

```sql
SELECT IID
FROM (
    SELECT IID, COUNT(DISTINCT DEP_ID) AS depnum
    FROM (
        SELECT CAID
        FROM Appointement
        WHERE Status = 'Scheduled'
    ) AS scheduled_appointments
    NATURAL JOIN ClinicalActivity
    GROUP BY IID
) AS R1
WHERE depnum > 1;
```

- **QUERY12 :** Find Staff IDs who have no scheduled appointments on the day of the Green March holiday (November 6):
  - **Relational Algebra:**

$$\rho\Big(R_1, \pi_{\text{staff\_ID}}(\sigma_{\text{Appointment.Status='Scheduled'}\wedge\text{Date='2025-11-06'}}(\text{Appointment} \bowtie \text{ClinicalActivity}))\Big)$$

$$\pi$$

  - **SQL Code:**

```sql
SELECT Staff_ID
FROM Staff
EXCEPT
SELECT Staff_ID
FROM Appointment
WHERE Date='2025-11-06';
```

- **QUERY13 :**Find departments whose average number of clinical activities is below the global departmental average:
  - **Relational Algebra:**

$$\rho\Big(DeptAvg, \gamma_{\text{DEP\_ID, count(CAID)}\rightarrow\text{DeptCount}}(\text{ClinicalActivity})\Big)$$

$$\rho\Big(GlobalAvg, \gamma \text{ avg(DeptCount)} \rightarrow \text{GlobalAvg}(DeptAvg)\Big)$$

$$\pi_{\text{DEP\_ID}}\left(\sigma_{\text{DeptCount<GlobalAvg}}(DeptAvg \times GlobalAvg)\right)$$

  - **SQL Code:**

```sql
SELECT D.DEP_ID, D.Name
FROM Department D
JOIN (
    SELECT DEP_ID, COUNT(CAID) AS DeptCount
    FROM ClinicalActivity
    GROUP BY DEP_ID
) AS DeptAvg ON D.DEP_ID = DeptAvg.DEP_ID
WHERE DeptAvg.DeptCount <
(
    SELECT AVG(DeptCount)
    FROM (
        SELECT DEP_ID, COUNT(CAID) AS DeptCount
        FROM ClinicalActivity
        GROUP BY DEP_ID
    ) AS AllDeptAvg
);
```

- **QUERY14 :**For each staff member, return the patient who has the greatest number of completed appointments with that staff member:

  – **Relational Algebra:**

$$\rho\Big(\mathrm{R},\ \pi_{\mathrm{CAID}}\left(\sigma_{\mathrm{status=''completed''}}\left(\mathrm{Appointment}\right)\right)\Big)$$
$$\rho\Big(S, \gamma_{\mathrm{staff\_ID,\ IID}}\left(\mathrm{COUNT(CAID)} \rightarrow \mathrm{nbApp}\right)\left(R \bowtie \mathrm{ClinicalActivity}\right)\Big)$$
$$\rho\Big(M, \gamma_{\mathrm{staff\_ID}}\left(\mathrm{MAX(nbApp)} \rightarrow \mathrm{maxApp}\right)(S)\Big)$$
$$\pi_{\mathrm{staff\_ID,IID}}\left(M \bowtie S\right)$$

  – **SQL Code:**

```
WITH S AS (
SELECT Staff_ID , IID, COUNT(CAID) AS nbApp
FROM ClinicalActivity
NATURAL JOIN
(SELECT CAID
FROM Appointment
WHERE status="completed")
GROUP BY Staff_ID , IID )
SELECT Staff_ID , IID
FROM S
NATURAL JOIN
(SELECT Staff_ID , IID, MAX(nbApp) AS maxApp
FROM S
GROUP BY Staff_ID );
```

- **QUERY15 :** List patients who had at least 3 emergency admissions during the year 2024 :

  – **Relational Algebra:**

$$\rho\Big(\mathrm{E1},\ \sigma_{\mathrm{Date}=2024}(\mathrm{ClinicalActivity} \bowtie \mathrm{Emergency})\Big)$$
$$\rho\Big(\mathrm{R},\ \gamma_{\mathrm{IID}}(\ \mathrm{COUNT(CAID)} \rightarrow \mathrm{count(E1)})\Big)$$
$$\rho\Big(\mathrm{T},\ \sigma_{\mathrm{count}\geq 3}(\mathrm{R})\Big)$$
$$\mathrm{T} \bowtie \mathrm{Patients}$$

  – **SQL Code:**

```
SELECT p.IID, p.FullName, COUNT(e.CAID) as
   emergency_count
```

```sql
FROM Patients p
JOIN ClinicalActivity ca ON p.IID = ca.IID
JOIN Emergency e ON ca.CAID = e.CAID
WHERE YEAR(ca.Date) = 2024
GROUP BY p.IID, p.FullName
HAVING COUNT(e.CAID) >= 3;
```

- **Functional Dependencies List:**
  - Patient:
    - IID → CIN, FullName,Birth, Sex, BloodGroup,Phone
    - CIN → FullName, Birth, Sex
  - Hospital:
    - HID→ Name, City , Region
    -City → Region
  - Department:
    - DEP_ID → HID, Name , Specialy
  - Staff:
    - STAFF_ID → FullName, Status
  - Clinical Activity:
    - CAID → IID, STAFF_ID, DEP_ID , Date, Time
  - Emergency :
    - CAID → TriageLevel, Outcome
  - Insurance:
    - InsID → Type
  - Expenses:
    - ExpID → InsID, CAID, Total
  - Medication:
    - MID → Name, Form, Strength , ActiveIngredient , TherapeuticalClass, Manufacturer
    - Name → Manufacturer, ActiveIngredient, TherapeuticalCalss
  - Stock:
    - HID → MID, StockTimeStamp , UnitPrice , Qty, ReorderLevel
  - Perscription:
    - PID → CAID , DateIssued
    - CAID → DateIssued
  - Includes:
    - PID , MID → Dosage, Duration
  - Contact Location:
    - CLID → City, Province , Street, Number , PostalCode, PhoneLocation
    - PhoneLocation → City, Province , Street, PostalCode
    - PostalCode → Province
    - City → province , PostalCode

5 Discussion

During this lab, our group encountered several conceptual and technical challenges that required discussion and teamwork to resolve. One of the main difficulties arose with Query 13, where we had to find departments whose average number of appointments was lower than the global average(average number of appointments for all departments). We struggled to interpret what "average per department" actually meant — whether it referred to the average number of appointments per day, per doctor, or across the entire dataset. This ambiguity made it difficult to decide how to structure the query correctly. We also ran into uncertainty with Queries 6 and 15, which asked us to list patients and staff. At first, we weren't sure if the goal was to display only names or IDs, or to return the full tuples containing all their details, which led to inconsistent interpretations within the group. Another interesting discussion emerged around query optimization: we debated whether using a cross product was more computationally expensive than relying on GROUP BY with aggregate functions such as COUNT or MAX, realizing that understanding efficiency is just as important as correctness. Additionally, we found the division operation in SQL particularly confusing, both in terms of syntax and logic, since it's rarely used and not directly supported in standard SQL. Finally, deciding which type of join to use for certain queries — especially when multiple tables were involved — also caused some hesitation, as the choice of join type could drastically affect the output and meaning of the results.

6 Conclusion

In this deliverable, we successfully translated complex queries into relational algebra expressions and SQL code, demonstrating our understanding of database querying techniques. We addressed various challenges, including interpreting ambiguous requirements and optimizing query performance. Through collaboration and discussion, we refined our approach to ensure accurate and efficient results. This lab has enhanced our skills in database management and query formulation, preparing us for more advanced topics in data handling and analysis.