University
Mohammed VI
Polytechnic

College of Computing

Deliverable : Queries - SQL and Relation Algebra

Data Management Course

UM6P College of Computing

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

Team Information

Team Name	Sekel
Member 1	Anass Mahdad
Member 2	Rania Moujahed
Member 3	Haitam Laghmam
Member 4	Abdelmoughit Labrigui
Member 5	Youssef Ouazzani
Member 6	Mohammed Idriss Nana
Repository Link	`https://github.com/Data-Project-Hierarchy/`

---

1 Introduction

Writing queries in relational algebra and SQL helps learning how to use the database you've built. While the database design defines what data you have and how it's organized, queries are how you actually interact with it — asking questions, extracting information, and combining data from different tables. Relational algebra providing the theoretical foundation, and SQL turning these foundations to actual commands to manipulate data.

2 Requirements

For each query, provide RA and SQL:

1. Find the names of patients who have had at least one clinical activity handled by active staff.
2. Find Staff IDs of staff who are either `'Active'` or have issued at least one prescription.
3. Find Hospital IDs of hospitals located in `'Benguerir'` or having at least one department with the specialty `'Cardiology'`.
4. Find Hospital IDs of hospitals that have both `'Cardiology'` and `'Pediatrics'` departments.
5. Find staff members who have worked in every department of the hospital with `HID = 1`.
6. Find staff members who participated in every clinical activity of the department with `DEP_ID = 2`.
7. Find pairs of staff members (s_1, s_2) such that s_1 has handled more clinical activities than s_2.
8. Find Patient IDs of patients who had clinical activities with at least two different staff members.
9. Find CAIDs of clinical activities performed in September 2025 at hospitals located in `'Benguerir'`.
10. Find Staff IDs of staff who have issued more than one prescription.
11. List IIDs of patients who have scheduled appointments in more than one department.
12. Find Staff IDs who have no scheduled appointments on the day of the Green March holiday (November 6).
13. Find departments whose average number of clinical activities is below the global departmental average.

14. For each staff member, return the patient who has the greatest number of completed appointments with that staff member.

15. List patients who had at least 3 emergency admissions during the year 2024.

Refinement : Derive functional dependencies from the MNHS schema.

3 Implementation & Results

Question 1

SQL:

```
-- 1. Find the names of patients who have had at least one
   ↪ clinical
-- activity handled by active staff.

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

Relational Algebra:

$$R = \rho_p(\text{Patient}) \bowtie_{\text{ca.iid=p.iid}} \rho_{ca}(\text{ClinicalActivity})$$

$$\pi_{\text{FullName}}\Big(\sigma_{s.\text{Status="Active"}}\big((R) \bowtie_{s.\text{staff\_id=ca.staff\_id}} \rho_s(\text{Staff})\big)\Big)$$

Where:

- R: The join of patient on clinical activity on the same patient IDs.

Question 2

SQL:

```
-- 2. Find Staff IDs of staff who are either Active or have
   ↪ issued at least one prescription.

SELECT STAFF_ID FROM staff s
WHERE Status = 'Active'
 AND EXISTS(
 SELECT STAFF_ID
 FROM clinicalactivity c JOIN prescription p ON p.CAID = c.CAID
 WHERE c.STAFF_ID = s.STAFF_ID
 );
```

Relational Algebra:

$$\pi_{STAFF\_ID}((\sigma_{Status='Active'}(Staff)) \cup (Prescription \bowtie Clinicalactivies \bowtie Staff))$$

Question 3

SQL:

```sql
-- 3. Find Hospital IDs of hospitals located in 'Benguerir'
-- or having at least one department with the specialty '
   ↪ Cardiology'

SELECT DISTINCT H.HID
FROM Hospital H
WHERE H.City = 'Benguerir'
UNION
SELECT D.HID
FROM Department D
WHERE D.Specialty = 'Cardiology';
```

Relational Algebra:

$$R_1 = \pi_{HID}(\sigma_{City='Benguerir'}(Hospital))$$
$$R_2 = \pi_{HID}(\sigma_{Specialty='Cardiology'}(Department))$$
$$R = R_1 \cup R_2$$

Result: $\pi_{HID}(R)$

Where:

- R_1: The set of all hospitals located in the city Benguerir.
- R_2: The set of all the hospitals which have a department with the specialty Cardiology.
- R: The set of Hospital IDs of hospitals located in 'Benguerir' *or* having at least one department with the specialty 'Cardiology'.

Question 4

SQL:

```sql
SELECT ca.STAFF_ID
FROM Prescription p
JOIN ClinicalActivity ca ON p.CAID = ca.CAID
```

```
GROUP BY ca.STAFF_ID
HAVING COUNT(p.PID) > 1;
```

Relational Algebra:

$$R_1 = Prescription \bowtie_{Prescription.CAID=ClinicalActivity.CAID} ClinicalActivity$$
$$R_2 = \gamma_{STAFF\_ID,\text{count}(PID)\rightarrow cnt}(R_1)$$
$$R = \sigma_{cnt>1}(R_2)$$
$$Result = \pi_{STAFF\_ID}(R)$$

Where:

- R_1: Jointure between Prescription and Clinical Activity on on CAID.
- R_2: Counting the number of prescriptions by STAFF_ID.
- R_3: Selecting staff with more than one prescription

Question 5

SQL:

```
-- 5. Find staff members who have worked in every department
--    of the hospital with HID = 1

SELECT s.STAFF_ID, s.FullName
FROM Work_in w
JOIN Department d ON d.DEP_ID = w.Dep_ID
JOIN Staff s  ON s.STAFF_ID = w.STAFF_ID
WHERE d.HID = 1
GROUP BY s.STAFF_ID, s.FullName
HAVING COUNT(DISTINCT w.Dep_ID) = (
    SELECT COUNT(*) FROM Department WHERE HID = 1
);
```

Relational Algebra :

$$R_1 = \pi_{\text{DEP\_ID}}\big(\sigma_{\text{HID}=1}(\text{Department})\big)$$
$$R_2 = \delta\big(\pi_{\text{STAFF\_ID,DEP\_ID}}(\text{Work\_in})\big)$$
$$R_3 = R_2 \div R_1$$
$$R_4 = R_3 \bowtie \text{Staff}$$
$$\text{Resultat} = \pi_{\text{Staff.STAFF\_ID, Staff.FullName}}(R_4)$$

Where:

- R_1: Departments belonging to the hospital with HID = 1.
- R_2: Pairs of (STAFF_ID, DEP_ID) showing where each staff member worked (deduplicated).

- R_3: Staff who worked in *all* departments of R_1 (division).
- R_4: Join of R_3 with Staff to retrieve names.
- **Resultat**: STAFF_ID and FullName of those staff members.

Question 6

SQL:

```
-- 6. Find staff members who participated in every clinical
   ↪ activity
    -- of the department with DEP_ID = 2.
SELECT S.STAFF_ID FROM Staff S
    JOIN clinicalactivity CA ON CA.STAFF_ID = S.STAFF_ID AND CA.
       ↪ Dep_ID = 2
    GROUP BY S.STAFF_ID
    HAVING COUNT(*) = (
        SELECT COUNT(*)
        FROM clinicalactivity
        WHERE Dep_ID = 2
    );
```

Relational Algebra:

$$R_1 = \pi_{CAID}(\sigma_{Dep\_ID=2}(ClinicalActivity))$$

$$R_2 = \pi_{STAFF\_ID,CAID}(\sigma_{Dep\_ID=2}(ClinicalActivity))$$

$$R = R_2 \div R_1$$

$$\text{Result: } \pi_{STAFF\_ID}(R)$$

Where:

- R_1: The set of all clinical activities belonging to department 2.
- R_2: The set of $(STAFF\_ID, CAID)$ pairs representing staff participation in department 2 activities.
- R: The set of staff members who participated in *every* clinical activity of department 2.

Question 7

SQL:

```
--  7. Find pairs of staff members (s1,s2) such that s1 has
   ↪ handled
-- more clinical activities than s2.

WITH SActivityCount AS (
SELECT staff_id, COUNT(*) AS ac_count
```

```
FROM ClinicalActivity
GROUP BY staff_id)
SELECT s1.staff_id, ac1.ac_count AS s1_count, s2.staff_id, ac2.
   ↪ ac_count AS s2_count
FROM Staff s1
JOIN Staff s2 ON 1=1
JOIN SActivityCount ac1 ON ac1.staff_id = s1.staff_id
JOIN SActivityCount ac2 ON (ac2.staff_id = s2.staff_id AND ac2.
   ↪ ac_count < ac1.ac_count);
```

Relational Algebra:

$$\text{SActivityCount} = \gamma_{\text{staff\_id},\, \text{COUNT}(*)\rightarrow \text{ac\_count}}\,(\text{ClinicalActivity})$$

$$R1 = (\rho_{s1}(\text{Staff}) \times \rho_{s2}(\text{Staff})) \bowtie_{ac1.\text{staff\_id}=s1.\text{staff\_id}} \rho_{ac1}(\text{SActivityCount})$$

$$R2 = \rho_{ac2}(\text{SActivityCount})$$

$$\pi_{s1.\text{staff\_id},ac1.\text{ac\_count},s2.\text{staff\_id},ac2.\text{ac\_count}}\Big(\sigma\Big((R1) \bowtie_{ac2.\text{staff\_id}=s2.\text{staff\_id}\wedge(ac2.\text{ac\_count}<ac1.\text{ac\_count})} R2\Big)\Big)$$

Where:

- *SActivityCount*: A temporary table that contains staff IDs associated to the corresponding number of clinical activities.
- $R1$: The join of the pairs of staff members (cartesian product) on their respective number of clinical activities.
- $R2$: An alias on SActivityCount for a better readability.

Question 8

SQL:

```
-- 8 . Find Patient IDs of patients who had clinical activities
   ↪ with at least two different staff members

SELECT DISTINCT c1.IID FROM
clinicalactivity c1
JOIN
clinicalactivity c2
ON c1.IID = c2.IID
WHERE c1.STAFF_ID <> c2.STAFF_ID;
```

Relational Algebra:

$$\pi_{C1.IID}\big(\sigma_{C1.STAFF\_ID\neq C2.STAFF\_ID}\big(\rho_{C1}(\text{ClinicalActivity}) \bowtie_{\text{C1.IID=C2.IID}} \rho_{C2}(\text{ClinicalActivity})\big)\big)$$

Question 9

SQL:

```sql
-- 9. Find CAIDs of clinical activities performed in September
   ↪ 2025
-- at hospitals located in "Benguerir"

SELECT CAID
FROM ClinicalActivity CA
JOIN Department D ON D.Dep_ID = CA.Dep_ID
JOIN Hospital H ON D.HID = H.HID
WHERE H.City = 'Benguerir'
  AND CA.Date >= '2025-09-01'
  AND CA.Date < '2025-10-01';
```

Relational Algebra:

$$R_1 = \sigma_{City='Benguerir'}(Hospital)$$
$$R_2 = R_1 \bowtie Department$$
$$R_3 = R_2 \bowtie_{D.Dep\_ID=CA.Dep\_ID} ClinicalActivity$$
$$R_4 = \sigma_{Date\geq'2025\text{-}09\text{-}01'\wedge Date<'2025\text{-}10\text{-}01'}(R_3)$$
$$R = \pi_{CAID}(R_4)$$

Result: R

Where:

- R_1: The set of all the hospitals located in Benguerir.
- R_2: The set of all the departments that belong to hospitals located in Benguerir.
- R_3: The set of all the clinical activities linked to those departments.
- R_4: Selecting only the clinical activities that occured in the September of 2025.
- R: The CAIDs of these clinical activities.

Question 10

SQL:

```sql
SELECT DISTINCT HID
FROM department
WHERE department.Specialty='Cardiology' OR department.Specialty='
   ↪ Pediatrics';
```

Relational Algebra:

$$R = \sigma_{Specialty='Cardiology'\vee Specialty='Pediatrics'}(Department)$$
$$Result = \pi_{HID}(R)$$

Where:

- R: Selecting departents specialized in Cardiology or Pediatrics

Question 11

SQL:

```
-- 11. List IIDs of patients who have scheduled appointments
--     in more than one department

SELECT ca.IID
FROM ClinicalActivity ca
JOIN Appointment a ON a.CAID = ca.CAID
WHERE a.Status = 'Scheduled'
GROUP BY ca.IID
HAVING COUNT(DISTINCT ca.DEP_ID) > 1;
```

Relational Algebra (using Aggregation):

$$R_1 = \sigma_{\text{Status}='Scheduled'}(\text{Appointment})$$
$$R_2 = \text{ClinicalActivity} \bowtie_{\text{CAID}} R_1$$
$$R_3 = \pi_{\text{IID,DEP\_ID}}(R_2)$$
$$R_4 = \gamma_{\text{IID; dcount:=COUNT\_DISTINCT(DEP\_ID)}}(R_3)$$
$$\text{Resultat} = \pi_{\text{IID}}\big(\sigma_{\text{dcount}>1}(R_4)\big)$$

Where:

- R_1: Appointments whose status is *Scheduled*.
- R_2: Join between ClinicalActivity and scheduled appointments.
- R_3: Pairs of (patient IID, department DEP_ID).
- R_4: Count of distinct departments per patient.
- **Resultat**: IIDs of patients with appointments in more than one department.

Question 12

SQL:

```
-- 12. Find Staff IDs who have no scheduled appointments
    -- on the day of the Green March holiday (November 6).
SELECT S.STAFF_ID FROM Staff S
    WHERE (
        SELECT COUNT(*) FROM clinicalactivity CA
            JOIN appointment A ON CA.CAID = A.CAID
            WHERE CA.STAFF_ID = S.STAFF_ID
                AND A.Status = 'Scheduled'
                AND MONTH(CA.Date) = 11
                AND DAY(CA.Date) = 6
    ) = 0;
```

Relational Algebra:

$$M = \rho(CA, ClinicalActivity) \bowtie_{CA.CAID=A.CAID} \rho(A, Appointment)$$
$$R_{busy} = \pi_{CA.STAFF\_ID}\big(\sigma_{A.Status='Scheduled' \wedge MONTH(CA.Date)=11 \wedge DAY(CA.Date)=6}(M)\big)$$
$$R_{all} = \pi_{STAFF\_ID}(Staff)$$

$$\text{Result: } R := R_{all} - R_{busy}$$

Where:

- M: The join between `ClinicalActivity` and `Appointment`, relating activities to their appointments.
- R_{busy}: The set of staff members who have at least one scheduled appointment on November 6.
- R_{all}: The set of all staff members.
- R: The resulting set of staff with *no scheduled appointments* on November 6 (i.e., $R_{all} - R_{busy}$).

Question 13

SQL:

```sql
--  13. Find departments whose average number of clinical
   ↪ activities
-- is below the global departmental average.

WITH DActivityCount AS (
SELECT dep_id, COUNT(*) AS ac_count
FROM ClinicalActivity
GROUP BY dep_id),
GlobalAvg AS (
SELECT AVG(ac_count) AS ca_avg FROM DActivityCount)

SELECT d.dep_id, ac.ac_count, ga.ca_avg
FROM Department d
JOIN DActivityCount ac ON ac.dep_id = d.dep_id
JOIN GlobalAvg ga ON 1=1
WHERE ac.ac_count < ga.ca_avg;
```

Relational Algebra:

$$DActivityCount = \gamma_{\text{dep\_id},\ \text{COUNT}(*)\rightarrow\text{ac\_count}}(\text{ClinicalActivity})$$

$$GlobalAvg = \gamma_{\text{AVG(ac\_count)}\rightarrow\text{ca\_avg}}(DActivityCount)$$
$$R = (\rho_d(\text{Department}) \bowtie_{ac.\text{dep\_id}=d.\text{dep\_id}} \rho_{ac}(DActivityCount)) \times \rho_{ga}(GlobalAvg)$$

$$\pi_{d.\text{dep\_id},\ ac.\text{ac\_count},\ ga.\text{ca\_avg}} = \left(\sigma_{ac.\text{ac\_count} < ga.\text{ca\_avg}}(R)\right)$$

Where:

- *DActivityCount*: A temporary table that contains department IDs associated to the corresponding number of clinical activities.
- *GlobalAvg*: A temporary table that contains the average value of the number of clinical activities in all the departments.
- *R*: The cartesian product (because it should be displayed for each row) of the global average with a join of department on the corresponding activity count for each one.

Question 14

SQL:

```sql
-- 14 . For each staff member, return the patient who has the
   ↪ greatest number of completed appointments with that staff
   ↪ member.

SELECT
    P.FullName As PatientName
FROM
    Appointment A
    JOIN ClinicalActivity CA ON A.CAID = CA.CAID
    JOIN Staff S ON CA.STAFF_ID = S.STAFF_ID
    JOIN Patient P ON CA.IID = P.IID
WHERE
    A.Status = 'Completed'
GROUP BY
    S.STAFF_ID, P.IID
HAVING
    COUNT(*) = (
        SELECT MAX(PatientCount)
        FROM (
            SELECT CA2.IID, COUNT(*) AS PatientCount
            FROM Appointment A2
            JOIN ClinicalActivity CA2 ON A2.CAID = CA2.CAID
            WHERE A2.Status = 'Completed'
              AND CA2.STAFF_ID = S.STAFF_ID
            GROUP BY CA2.IID
        ) AS StaffPatients
    );
```

Relational Algebra:

$$A_c := \sigma_{\text{Status}='Completed'}(\text{Appointment})$$

$$T := \gamma_{\text{STAFF\_ID, IID; cnt=COUNT}(*)}(A_c \bowtie_{A_c.\text{CAID}=\text{ClinicalActivities.CAID}} \text{ClinicalActivities})$$

$$M := \gamma_{\text{STAFF\_ID; maxcnt=MAX(cnt)}}(T)$$

$$\pi_{\text{FullName}}\Big((T \bowtie_{T.\text{STAFF\_ID}=M.\text{STAFF\_ID} \,\wedge\, T.\text{cnt}=M.\text{maxcnt}} M) \bowtie_{T.\text{IID}=\text{Patient.IID}} \text{Patient})\Big)$$

Question 15

SQL:

```sql
-- 15. List patients who had at least 3 emergency admissions
-- during the year 2024.

SELECT FullName
FROM Patient P
JOIN ClinicalActivity CA ON CA.IID = P.IID
JOIN Emergency E ON E.CAID = CA.CAID
WHERE CA.Date >= '2024-01-01'
  AND CA.Date < '2025-01-01'
GROUP BY P.IID
HAVING COUNT(E.CAID) >= 3;
```

Relational Algebra:

$$R_1 = \sigma_{Date \geq '2024\text{-}01\text{-}01' \wedge Date < '2025\text{-}01\text{-}01'}(ClinicalActivity)$$

$$R_2 = R_1 \bowtie Emergency$$

$$R_3 = \pi_{IID, E.CAID}(R_2)$$

$$R = \pi_{FullName}\big(\sigma_{num\_emergency \geq 3}(\gamma_{P.IID,\, COUNT(E.CAID) \rightarrow num\_emergency}(R_3 \bowtie Patient))\big)$$

Result: R

Where:

- R_1: Clinical activities performed in 2024.
- R_2: Emergency activities joined with those clinical activities.
- R_3: The patient IID with the clinical activity id of R2 (which contain all the emergencies performed in 2024).
- R: The full name of all the patients that the number of emergencies is more than 3 (after grouping the emergencies per patient using γ).

Functional Dependencies in MNHS Database

In this section, we list the key functional dependencies (FDs) identified in the Moroccan National Health Services (MNHS) database.

We do not write the **trivial functional dependencies**, such as:

- $R \to R$
- All attributes $\to$ all attributes

because they do not provide new information about the structure of the database.

We also use **Armstrong's axioms** to infer additional dependencies from the given ones. The main axioms are:

1. **Reflexivity:** If $Y \subseteq X$, then $X \to Y$. *Example:* $\{A, B\} \to \{A\}$.
2. **Augmentation:** If $X \to Y$, then $XZ \to YZ$. *Example:* If $A \to B$, then $AC \to BC$.
3. **Transitivity:** If $X \to Y$ and $Y \to Z$, then $X \to Z$. *Example:* If $A \to B$ and $B \to C$, then $A \to C$.

These axioms allow us to infer non-trivial FDs from the given ones and to reason about keys, superkeys, and normalization.

Functional Dependencies by Relation

Patients

$$I \to ICSFBSEP \text{ (PK)}, \quad C \to ICSFBSEP \text{ (CK)}$$

Where:

- I: Patient ID (IID)
- C: CIN
- S: Sex
- F: FullName
- B: Birth
- E: BloodGroup
- P: Phone

Hospital

$$H \rightarrow HNCR \text{ (PK)}, \quad C \rightarrow R \text{ (CK)}, \quad NC \rightarrow HNCR$$

(no pair of hospitals with same city and name)

Where:

- H: Hospital ID (HID)
- N: Name
- C: City
- R: Region

Department

$$D \rightarrow DHNS \text{ (PK)}, \quad HN \rightarrow DHNS$$

(no hospital with a pair of department with the same name)

Where:

- D: Department ID (DEP_ID)
- H: Hospital ID (HID)
- N: Name
- S: Specialty

Staff

$$S \rightarrow SFSt \text{ (PK)}$$

Where:

- S: Staff ID (STAFF_ID)
- F: FullName
- St: Status

Work_in

$$SD \rightarrow SD \text{ (PK)}$$

Where:

- S: Staff ID
- D: Department ID

ClinicalActivity

$$C \rightarrow CISDDtT \text{ (PK)}, \quad ISTD \rightarrow C$$

(no staff member performing a ca in a patient in 2 different departments at the same time)

Where:

- C: ClinicalActivity ID (CAID)
- I: Patient ID (IID)
- S: Staff ID
- D: Department ID
- Dt: Date
- T: Time

Appointment

$$C \rightarrow CRS \text{ (PK)}$$

Where:

- C: ClinicalActivity ID (CAID)
- R: Reason
- S: Status

Emergency

$$C \rightarrow CTO \text{ (PK)}$$

Where:

- C: ClinicalActivity ID (CAID)
- T: TriageLevel
- O: Outcome

Insurance

$$I \rightarrow IT \text{ (PK)}$$

Where:

- I: Insurance ID
- T: Type

Expense

$$E \to EICT \text{ (PK)}, \quad C \to EICT \text{ (CK)}$$

Where:

- E: Expense ID
- I: Insurance ID
- C: ClinicalActivity ID
- T: Total

Medication

$$M \to MNFSATMf \text{ (PK)}, \quad N \to NATMf, \quad A \to T$$

(no pair of medication with same NATMf but maybe different forms of strengh so id -eg: Doliprane)
(A determine a big part of the effect)

Where:

- M: Medication ID (MID)
- N: Name
- F: Form
- S: Strength
- A: ActiveIngredient
- T: TherapeuticClass
- Mf: Manufacturer

Stock

$$MHS \to HMSUQR \text{ (PK)}$$

Where:

- M: Medication ID
- H: Hospital ID
- S: StockTimestamp
- U: UnitPrice
- Q: Quantity
- R: ReorderLevel

Prescription

$$P \rightarrow PCD \text{ (PK)}, \quad C \rightarrow PCD \text{ (CK)}$$

Where:

- P: Prescription ID
- C: ClinicalActivity ID
- D: DateIssued
- PC: Patient/Clinical details

Includes

$$PM \rightarrow PMDDu \text{ (PK)}$$

Where:

- P: Prescription ID
- M: Medication ID
- D: Dosage
- Du: Duration

ContactLocation

$$C \rightarrow CCiPSNPoPh \text{ (PK)}, \quad Po \rightarrow Ci, \quad P \rightarrow Ci$$

(the postal code or province determine the city)

Where:

- C: ContactLocation ID (CLID)
- Ci: City
- P: Province
- S: Street
- N: Number
- Po: PostalCode
- Ph: Phone

Have

$$IC \rightarrow IC \text{ (PK)}$$

Where:

- I: Patient ID
- C: ContactLocation ID

4 Discussion

Challenges faced

- Writing queries in Relational Algebra and SQL was sometimes tricky, especially when dealing with complex joins, nested queries, and aggregation functions.
- Identifying functional dependencies required paying attention to details and understanding how the different data components interact with each other.

Observations

- This step helped clarify parts of the database schema that were ambiguous; writing queries forced us to think carefully about keys and relationships.
- Functional dependencies can guide you to better understanding and optimization of the database to avoid redundancy or incorrect results.

Lessons learned

- Understanding relationships between entities is crucial before trying to write queries.
- Choosing the right attributes for selections, projections, and join conditions is key to producing accurate results.
- Writing queries and identifying functional dependencies helps uncover subtle aspects of the schema that impact both correctness and efficiency.

5 Conclusion

In this lab, we successfully translated instruction into concrete queries using Relational Algebra and SQL. We practiced writing queries to retrieve, manipulate, and combine data while respecting the structure of tables, keys, and relationships defined in the schema. During this process, we also identified functional dependencies giving a first idea of future optimizations, understanding how the data actually interact with itself. Overall, this lab highlighted how writing queries and analyzing functional dependencies are crucial steps in making a database not only theoretically correct, but also practically usable to answer real-world questions.