CSE 4/560 Fall 2024

Databases and Query Languages Homework 1

Name: Sahithya Arveti Nagaraju

Person number: 50559752

Problem 1 [5X1=5 points] Define the following terms: relation, relation schema, database, database schema, Domain.

Solution:

Relation: A relation is a table that consists of rows and columns. Each row (or tuple) represents a unique record, and each column (or attribute) represents a specific piece of data about that record. Relations are a fundamental concept in relational databases.

Relation schema: This refers to the structure of a relation, defining the attributes (columns) of the table and their data types. It describes what data can be stored in the relation, including constraints like primary keys and foreign keys.

Database: A database is a collection of data, typically related and describing activities of an organization. A database is a collection of interrelated data that models some aspect of the real world.

Database schema: This is the overall structure of a database, describing how data is organized and how the relationships between different data entities are defined. It includes the definitions of tables, the relationships between them, constraints, and sometimes the views or stored procedures associated with the database.

Domain: In a database context, a domain refers to the set of valid values that an attribute can take. Domains help ensure data integrity by enforcing constraints on the values that can be stored in the database.

Problem 2 [5 points] What is a foreign key constraint? Why are such constraints important?

Solution: A foreign key constraint is a rule that establishes a relationship between two tables in a relational database. Specifically, it ensures that the values in a column (or a set

of columns) in one table (the child table) correspond to values in a column (or set of columns) of another table (the parent table). The foreign key in the child table refers to the primary key in the parent table. This means every value of the foreign key must either be null or match a value in the parent table's primary key.

Importance:

1. Data Integrity: They ensure that relationships between tables remain consistent.
2. Referential Integrity: They prevent actions that would leave orphaned records.
3. Logical Organization: They help in logically organizing data by linking related information across different tables.
4. Performance Benefits: Properly indexed foreign keys can improve query performance by allowing the database engine to quickly locate related data.

Problem 3 [5+5X2=15 points] Consider the Movie database example.

Movies (title , year, length , genre, studioName, producerC#)

StarsIn (movieTitle, movieYear, starName)

MovieStar (name, address, gender, birthdate)

MovieExec(name, address, cert# , netWorth)

Studio(name, address, presC#)

a. Identify the primary keys for each relations.

Solution:

1. Movies:
   - Primary Key: (title, year)
     (The combination of title and year uniquely identifies a movie.)
2. StarsIn:
   - Primary Key: (movieTitle, movieYear, starName)
     (This combination uniquely identifies the record of a star in a specific movie.)
3. MovieStar:
   - Primary Key: (name, birthdate)
     (Assuming star name with birthdate are unique.)
4. MovieExec:

- Primary Key: cert#
  (Assuming certification numbers are unique.)

5. Studio:
   - Primary Key: *presC#*
     (Assuming each studio has a unique *presC#*.)

b. Write the following queries in SQL

1. Find the address of MGM studios.

Solution:

```
SELECT address
FROM Studio
WHERE name = 'MGM';
```

2. Find Sandra Bullock's birthdate.

Solution:

```
SELECT birthdate
FROM MovieStar
WHERE name = 'Sandra Bullock';
```

3. Find all the stars that appeared either in a movie made in 1980 or a movie with "Love" in the title.

Solution:

```
SELECT DISTINCT starName
FROM StarsIn
WHERE movieYear = 1980 OR movieTitle LIKE '%Love%';
```

4. Find all executives worth at least $10,000,000.

Solution:

```
SELECT name
FROM MovieExec
WHERE netWorth >= 10000000;
```

5. Find all the stars who either are male or live in Malibu (have string Malibu as a part of their address).

Solution:

```
SELECT name
FROM MovieStar
WHERE gender = 'Male' OR address LIKE '%Malibu%';
```

Fig:1 Bank database

---

branch(*branch_name*, *branch_city*, *assets*)
customer (*ID*, *customer_name*, *customer_street*, *customer_city*)
loan (*loan_number*, *branch_name*, *amount*)
borrower (*ID*, *loan_number*)
account (*account_number*, *branch_name*, *balance*)
depositor (*ID*, *account_number*)

Problem 4 [3X5=15] Consider the bank database of Figure 1 Give an expression in the relational algebra for each of the following queries:

a. Find each loan number with a loan amount greater than $10000.

Solution: $\pi_{loan\_number}(\sigma_{amount > 10000}(loan))$

b. Find the ID of each depositor who has an account with a balance greater than $6000.

Solution: $\pi_{ID}(\sigma_{balance > 6000}(account \bowtie depositor))$

c. Find the ID of each depositor who has an account with a balance greater than $6000 at the "Uptown" branch. Answer each of the following questions briefly. The questions are based on the following relational schema:

Solution: $\pi_{ID}(\sigma_{balance > 6000 \wedge branch\_name = 'Uptown'}(account \bowtie depositor))$

Explanation of the expressions:

- **π:** Projection operator, selects specific attributes.
- **σ:** Selection operator, filters tuples based on conditions.
- **⋈:** Join operator, combines tuples from two relations based on a common attribute.

Problem 5 [5X5=25] Consider the following relations containing airline flight information. Write the following queries in relational algebra.

Flights(*flno:* integer, *from:* string, *to:* string,
distance: integer, *departs:* time, *arrives:* time)
Aircraft(*aid:* integer, *aname:* string, *cruisingrange:* integer)
Certified(*eid:* integer, *aid:* integer)
Employees(*eid:* integer, *ename:* string, *salary:* integer)

a) Find the eids of pilots certified for some Boeing aircraft.

Solution: $\pi_{eid}(\sigma_{aname='Boeing'}(\text{Aircraft} \bowtie \text{Certified}))$

b) Find the names of pilots certified for some Boeing aircraft.

Solution: $\pi_{ename}(\sigma_{aname='Boeing'}(\text{Aircraft} \bowtie \text{Certified} \bowtie \text{Employees}))$

c) Find the aids of all aircraft that can be used on non-stop flights from Bonn to Madras.

Solution: $\pi_{aid}((\sigma_{distance<=cruisingrange \wedge from = 'Bonn' \wedge to = 'Madras'}(\text{Aircraft} \bowtie \text{Flights})))$

d) Identify the flights that can be piloted by every pilot whose salary is more than $100,000.

Solution: $\pi_{flno}(\sigma_{salary>100000 \wedge distance<cruisingrange}(\text{Employees} \bowtie \text{Certified} \bowtie \text{Aircraft} \bowtie \text{Flights}))$

e) Find the names of pilots who can operate planes with a range greater than 3,000 miles but are not certified on any Boeing aircraft.

Solution: $\pi_{ename}(\sigma_{aname \neq 'Boeing' \wedge cruisingrange>3000}(\text{Aircraft} \bowtie \text{Certified} \bowtie \text{Employees}))$

Problem 6: [5X2=10] Consider the instance of the Sailor relation shown in Figure 5.1.

sid	*sname*	*rating*	*age*
18	jones	3	30.0
41	jonah	6	56.0
22	ahab	7	44.0
63	moby	*null*	15.0

Figure 5.1 An Instance of Sailors

a. Write SQL queries to compute the average rating, using AVG; the sum of the ratings, using SUM; and the number of ratings using COUNT.

Solution:

-- Average rating

SELECT AVG(rating) AS average_rating FROM Sailors;

-- Sum of ratings

SELECT SUM(rating) AS sum_ratings FROM Sailors;

-- Number of ratings

SELECT COUNT(rating) AS number_ratings FROM Sailors;

b. If you divide the sum just computed by the count, would the result be the same as the average? How would your answer change if these steps were carried out with respect to the age field instead of rating?

Solution:

1. **Rating Calculation**:
    - **Average**: SELECT Avg(rating) FROM Sailor; returns approximately **5.333**.
    - **Manual Calculation**: SELECT Sum(rating) / Count(rating) FROM Sailor; yields **5**.
    - **Discrepancy**: The difference suggests that null values in the rating field affect the count, leading to a lower manual average.
2. **Age Calculation**:
    - **Average**: SELECT Avg(age) FROM Sailor; gives **36.25**.
    - **Manual Calculation**: SELECT Sum(age) / Count(age) FROM Sailor; also results in **36.25**.
    - **Consistency**: This indicates no null values in the age field, confirming that all records are included.

Discrepancies in averages can indicate null values affecting counts. Consistent results show all relevant data is accounted for. Always consider the impact of nulls on aggregate calculations in SQL.

Problem 7: [5X3=15] Consider the following relational schema. An employee can work in more than one department; the pct_time field of the Works relation shows the percentage of time that a given employee works in a given department. Write triggers to ensure each of the following requirements is considered independently.

Emp(*eid:* integer, *ename:* string, *age:* integer, *salary:* real)
Works(*eid:* integer, *did:* integer, *pct_time:* integer)
Dept(*did:* integer, *dname:* string, *budget:* real, *managerid:* integer)

a. Define a table constraint on Emp that will ensure that every employee makes at least $10,000.

Solution:

```
ALTER TABLE Emp

ADD CONSTRAINT chk_min_salary

CHECK (salary >= 10000);
```

b. Define a table constraint on Dept that will ensure that all managers are age > 30.

Solution:

```
ALTER TABLE Dept

ADD CONSTRAINT chk_manager_age

CHECK((SELECT E.age FROM Emp E, Dept D WHERE E.eid=D.managerid)>30);
```

c. Create a trigger to ensure that whenever an employee is given a raise, the manager's salary must be increased to at least as much.

Solution:

```
CREATE TRIGGER manager_salary_raise AFTER UPDATE OF salary ON Emp

FOR EACH ROW

WHEN (NEW.salary > OLD.salary)

DECLARE  v_managerid INTEGER;

BEGIN

   SELECT managerid INTO v_managerid
```

```
    FROM Dept
    WHERE did IN (
        SELECT did
        FROM Works
        WHERE eid = OLD.eid
    );
    UPDATE Emp
    SET salary = NEW.salary
    WHERE eid = v_managerid AND salary < NEW.salary;
END;
```

Problem 8: [5X2=10] Discuss the strengths and weaknesses of the trigger mechanism.

Solution:

Strengths:

- Enforce Data Integrity: Triggers can help maintain data consistency and integrity by automatically enforcing rules and constraints.
- Automation: They automate tasks that would otherwise require manual intervention, reducing the risk of human error.
- Complex Rule Enforcement: Triggers can implement complex business rules that are difficult or impossible to enforce using constraints alone.
- Event-Based Actions: Triggers can be used to initiate actions based on specific events, such as data changes or insertions.

Weaknesses:

- Performance Overhead: Triggers can introduce performance overhead, especially if they execute complex logic or access large amounts of data.
- Complexity: Triggers can be difficult to write and maintain, especially for complex business rules.
- Unexpected Behavior: If not carefully designed and tested, triggers can lead to unexpected behavior or even errors.

- Limited Scope: Triggers are typically limited to the scope of a single transaction or database operation.