CS2102: Database Systems

Tutorial #2: *Relational Algebra & SQL (Part 1)*

Week 4 Guide

AY 2022/23 Sem 2

1 Discussions

The following questions are to be discussed during tutorial. All answers will be released with explanation.

1. **(RA Operator)** This question considers a binary relational algebra operator called the **division operator** denoted by /[1].

   Consider two relations R and S:

   - $R(A_1, \cdots, A_m, B_1, \cdots, B_n)$
   - $S(B_1, \cdots, B_n)$

   where $m \geq 1$ and $n \geq 1$. That is, the set of attributes in S is a *proper* subset of the set of attributes in R.

   Assume that the attributes that are in R but not in S are ordered as $(A_1, \cdots, A_m)$ in the schema of R and the schema of S is $(B_1, \cdots, B_n)$. Let L denote the list of attributes in the schema of R.

   The division of R by S (denoted by R/S) computes the largest set of tuples $Q \subseteq \pi_{[A1,\cdots,A_m]}(R)$ such that for every tuple $(a_1, \cdots, a_m) \in Q$,

   $$\pi_{[L]}(\{(a_1, \cdots, a_m)\} \times S) \subseteq R$$

   Q is also referred to as the **quotient** of R/S and its schema is $(A_1, \cdots, A_m)$. The following example illustrates R/S given two relations $R(A, B)$ and $S(B)$.

[1] It is a "division" operation because it kind of reverses the "multiplication" operation $\times$.

R

A	**B**
a	1
a	2
b	1
c	1
c	2
c	3
d	2
d	3

S

B
1
2

R/S

A
a
c

(a) Consider again Question 3 in Tutorial 1 (*Challenge*) to find the restaurants that sell all the pizzas that Maggie likes and don't sell any pizza that Ralph likes. Write a relational algebra expression for this query that uses the division and natural join operators.

(b) Given relations $R(A, B)$ and $S(B)$, write a relational algebra expression to compute the division of R by S using only the basic relational operators (*i.e.*, σ, π, ρ, $\times$, $\cup$, $\cap$ and $-$).

Suggested Guide:

If you noticed, R/S given $R(A, B)$ and $S(B)$ is exactly all the tuples with the same value on attribute A for which the attribute B contains all values in S (*i.e.*, the superset of S).

(a) In this case, finding the restaurant that sells all the pizzas that Maggie likes can be done using the division operator.

$$Q_1 := \pi_{[\texttt{rname,pizza}]}(\texttt{Sells})/\pi_{[\texttt{pizza}]}(\sigma_{[\texttt{cname='Maggie'}]}(\texttt{Likes}))$$

We then need to remove all the restaurants that sells any pizza that Ralph likes.

$$Q_2 := \pi_{[\texttt{rname}]}(\texttt{Sells} \bowtie \sigma_{[\texttt{cname='Ralph'}]}(\texttt{Likes}))$$

This gives us the final answer of

$$Q_{ans} := Q_1 - Q_2$$

(b) Let $\pi_A(R) = Q \cup Q'$ where $Q = R/S$ and Q' are the remaining tuples in $\pi_A(R)$ that are not in quotient of R/S.

The expression $\pi_{[A]}(R) \times S$ computes all the combinations of $\pi_{[A]}(R)$ and S.

Thus, $Q' = \pi_{[A]}((\pi_{[A]}(R) \times S) - R)$.

Therefore, $Q = \pi_{[A]}(R) - Q' = \pi_{[A]}(R) - \pi_{[A]}((\pi_{[A]}(R) \times S) - R)$.

2. **(SQL DDL)** Consider a relational database for a company that consists of the following two tables.

   - `Offices(office_id, building, floor, room_number, area)`
   - `Employees(emp_id, name, office_id, manager_id)`

   The database satisfies the following constraints:

   1. `Offices` stores information about the office rooms in the company. Each room has a unique identifier `office_id` (*which is the primary key of* `Offices`) and information on its building name, floor level, room number and floor area.
   2. {`building, floor, room_number`} is a *candidate key* of `Offices`.
   3. `Employees` stores information about the employees in the company.
   4. `emp_id` is the *primary key* of `Employees`.
   5. The name of each employee must be a non-`NULL` value.
   6. Each employee must be assigned to exactly one office identified by `office_id`.
   7. Each employee may be managed by at most one manager.
   8. If an employee is managed by someone, the `emp_id` of his/her manager is recorded in `manager_id`.
   9. A record in `Offices` cannot be removed if there's some employee assigned to that office.
   10. A manager in `Employees` cannot be removed if there's some other employee managed by that manager.
   11. Any modification to `office_id` in `Offices` is propagated to other database records.
   12. Any modification to `emp_id` in `Employees` is propagated to other database records.

   Write SQL statements to create the database schema with appropriate attribute domains and constraints.

Suggested Guide:

```sql
CREATE TABLE Offices (
  office_id   INTEGER,
  building    TEXT    NOT NULL,
  floor       INTEGER NOT NULL,
  room_number INTEGER NOT NULL,
  area        INTEGER,
  PRIMARY KEY (office_id),
  UNIQUE (building, floor, room_number)
);


CREATE TABLE Employees (
  emp_id      INTEGER,
  name        TEXT    NOT NULL,
  office_id   INTEGER NOT NULL,
  manager_id  INTEGER,
  PRIMARY KEY (emp_id),
  FOREIGN KEY (office_id) REFERENCES Offices (office_id)
    ON UPDATE CASCADE,
  FOREIGN KEY (manager_id) REFERENCES Employees (emp_id)
    ON UPDATE CASCADE
);
```

Note that constraints (6) and (7) are both enforced by the primary key constraint in Employees which ensure that there can't be two Employees records with the same primary key value and different values for office_id / manager_id.

Additionally, when you are writing a database project where you may change the table again and again, it is a good practice to start the file with deleting the existing tables.

```
DROP TABLE IF EXISTS Offices, Employees CASCADE;
```

3. **(SQL DDL)** Consider a relational database for an online shop that consists of the following five tables.

- Books(isbn, title, authors, year, edition, publisher, number_pages, price)
- Customers(cust_id, name, email)
- Carts(cust_id, isbn)
- Purchase(pid, purchase_date, cust_id)
- Purchased_items(pid, isbn)

The database satisfies the following constraints:

1. Books records information about the books available for sale in an online shop. Each book has a unique identifier isbn and information about its title, authors, publishers, publication year, edition, number of pages and selling price. The title and authors must have non-NULL values. The value of the edition must be non-NULL with one of the following values: *paperback*, *hardcover* or *ebook*. The selling price must have a positive value. If the number of pages in known, it must be a positive value.
2. Customers stores information about the shop's customers. Each customer has a unique identifier cust_id, a name and an email address. The name must have a non-NULL value.
3. Carts stores information about the books in customers' shopping carts. Each shopping cart record indicates a book that a customer is interested to purchase but hs not yet purchased.
4. When a customer decides to purchase their selected books, a new record for this purchase is recorded in the Purchase table which has the following information:
   (a) A unique identifier pid.
   (b) The date of the purchase.
   (c) The customer identifier.

   In addition, each book in the customers' shopping cart is added to the Purchased_items table and the customers' shopping cart is emptied.

(a) Write SQL statements to create a database schema with appropriate attribute domains and constraints.

(b) Suppose that the schema of Purchase is changed with purchase_date being replaced by purchase_timestamp, where each customer has at most one purchase at any timestamp. How would this change affect your answer for part (a)?

(c) For each of the following additional constraints on the database, state whether the constraint can be expressed using SQL constructs that you have learned. If it is possible, add this constraint to your database schema in part (a).

1. If a book is a hardcover edition, its selling price must be at least 30.
2. If a book has both hardcover and paperback editions (*for the same book title and authors*), the selling price for the hardcover edition must be higher than the selling price for the paperback edition.
3. If the number of pages in a book is more than 1000, the edition of the book must be an ebook or its price must be at least 100.
4. All the books published by 'Acme' from 2010 onwards have only ebook edition.

Suggested Guide:

Each table in a database schema must have a primary key.

(a) SQL statements

```
DROP TABLE IF EXISTS
    Books, Customers, Carts, Purchase, Purchased_items CASCADE;

CREATE TABLE Books (
  isbn          TEXT,
  title         TEXT NOT NULL,
  authors       TEXT NOT NULL,
  year          INTEGER,
  edition       TEXT NOT NULL
   CHECK (edition in ('hardcopy', 'paperback', 'ebook')),
  publisher     TEXT,
  number_pages INTEGER CHECK (number_pages > 0),
  price         NUMERIC NOT NULL CHECK (price > 0),
  PRIMARY KEY (isbn)
);

CREATE TABLE Customers (
  cust_id      INTEGER,
  name         TEXT NOT NULL,
  email        TEXT NOT NULL,
  PRIMARY KEY (cust_id)
);

CREATE TABLE Carts (
  cust_id      INTEGER,
  isbn         TEXT,
  PRIMARY KEY (cust_id, isbn),
  FOREIGN KEY (cust_id) REFERENCES Customers,
  FOREIGN KEY (isbn) REFERENCES Books
);

CREATE TABLE Purchase (
  pid             INTEGER,
  purchase_date   DATE NOT NULL,
  cust_id         INTEGER NOT NULL,
  PRIMARY KEY (pid),
  FOREIGN KEY (cust_id) REFERENCES Customers
);

CREATE TABLE Purchased_Items (
  pid    INTEGER,
  isbn   TEXT,
  PRIMARY KEY (pid, isbn),
  FOREIGN KEY (pid) REFERENCES Purchase,
```

```
  FOREIGN KEY (isbn) REFERENCES Books
);
```

(b) (`purchase_timestamp, cust_id`) is a candidate key of `Purchase`.

```
CREATE TABLE Purchase (
  pid                  INTEGER,
  purchase_timestamp   TIMESTAMP NOT NULL,
  cust_id              INTEGER NOT NULL,
  PRIMARY KEY (pid),
  FOREIGN KEY (cust_id) REFERENCES Customers,
  UNIQUE (purchase_timestamp, cust_id)
);
```

(c) The constraint of the form "$p \Rightarrow q$" is equivalent to "(not p) or q".

1. Can be expressed using a table constraint on `Books`:
   `CHECK ((edition <> 'hardcover') OR (price >= 30))`
2. Can **NOT** be expressed using the constructs learned so far.
3. Can be expressed using a table constraint on `Books`:
   `CHECK ((number_pages <= 1000) OR (edition = 'ebook')`
   `OR (price >= 100))`
4. Can be expressed using a table constraint on `Books`:
   `CHECK ((publisher <> 'Acme') OR (year < 2010)`
   `OR (edition = 'ebook'))`

2 Challenge

The answers to the following questions is given without explanation. Please discuss them on Canvas.

1. **(SQL DDL)** This question refers to your solution for question 3.

   (a) Consider the following additional constraints on the database:

   1. If a customer is deleted from `Customers`, remove all the customers' records from `Carts` and `Purchase`.
   2. If a book is deleted from `Books`, remove all records from `Carts` that reference this book. Additionally, for each of the records in `Purchased_items` that reference this book, change its `isbn` value to the default value of 0.
   3. If a purchase is deleted from `Purchase`, remove all the records from `Purchased_items` that reference this purchase.
   4. Any modification of `cust_id` in `Customers` is propagated to other records in the database.
   5. Any modification of `isbn` in `Books` is propagated to other records in the database.
   6. Any modification of `pid` in `Purchase` is propagated to other records in the database.

   Add these additional constraints to your answer for question 3(a).

Suggested Guide:

(a) SQL statements

```sql
DROP TABLE IF EXISTS
    Books, Customers, Carts, Purchase, Purchased_items CASCADE;

CREATE TABLE Books (
  isbn          TEXT,
  title         TEXT NOT NULL,
  authors       TEXT NOT NULL,
  year          INTEGER,
  edition       TEXT NOT NULL
   CHECK (edition in ('hardcopy', 'paperback', 'ebook')),
  publisher     TEXT,
  number_pages  INTEGER CHECK (number_pages > 0),
  price         NUMERIC NOT NULL CHECK (price > 0),
  PRIMARY KEY (isbn)
);

CREATE TABLE Customers (
  cust_id      INTEGER,
  name         TEXT NOT NULL,
  email        TEXT NOT NULL,
  PRIMARY KEY (cust_id)
);

CREATE TABLE Carts (
  cust_id      INTEGER,
  isbn         TEXT,
  PRIMARY KEY (cust_id, isbn),
  FOREIGN KEY (cust_id) REFERENCES Customers
    ON DELETE CASCADE      ON UPDATE CASCADE,
  FOREIGN KEY (isbn) REFERENCES Books
    ON DELETE CASCADE      ON UPDATE CASCADE
);

CREATE TABLE Purchase (
  pid             INTEGER,
  purchase_date   DATE NOT NULL,
  cust_id         INTEGER NOT NULL,
  PRIMARY KEY (pid),
  FOREIGN KEY (cust_id) REFERENCES Customers
    ON DELETE CASCADE      ON UPDATE CASCADE
);

CREATE TABLE Purchased_Items (
  pid   INTEGER,
  isbn  TEXT DEFAULT '0',
  PRIMARY KEY (pid, isbn),
  FOREIGN KEY (pid) REFERENCES Purchase
    ON DELETE CASCADE      ON UPDATE CASCADE,
  FOREIGN KEY (isbn) REFERENCES Books
    ON DELETE SET DEFAULT  ON UPDATE CASCADE
);
```

For the "`ON DELETE SET DEFAULT`" action to work in `Purchased_items` when a referenced book in `Books` is deleted, there must exist a record in `Books` with `isbn = '0'`. Otherwise, the deletion operation will be rejected.

2. **(Cardinalities)** You are given 2 relations R and S, with m being the number of tuples in R (i.e., $|R| = m$) and n being the number of tuples in S (i.e., $|S| = n$). Assume that $m > n > 0$, R and S are union-compatible and both relations do not contain any *null* values.

   (a) $R \cup S$ [$\boxed{m}$, $\boxed{m+n}$]

   (b) $R \cap S$ [$\boxed{0}$, $\boxed{n}$]

   (c) $R - S$ [$\boxed{m-n}$, $\boxed{m}$]

   (d) $R \bowtie S$ [$\boxed{0}$, $\boxed{mn}$]

   (e) $R ⟗ S$ [$\boxed{m}$, $\boxed{mn}$]

3. **(Equivalence)** Select ALL (*strongly*) equivalences are true?

   (✓) $\sigma_c(E_1 - E_2) \equiv \sigma_c(E_1) - E_2$

   (b) $\pi_A(\pi_B(E)) \equiv \pi_A(E)$

   (✓) $\sigma_{c^1}(E_1 \bowtie_{c^2} E_2) \equiv \sigma_{c^2}(E_1 \bowtie_{c^1} E_2)$

   (d) $(E_1 \bowtie_{c^1} E_2) \bowtie_{c^2 \wedge c^3} E_3 \equiv E_1 \bowtie_{c^1 \wedge c^2} (E_2 \bowtie_{c^3} E_3)$

   (✓) $(E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3)$