CSE 180 - Lab 3

Details, Transactions, and Foreign Key constraints

Dev

Nov 2 2023, Section 6

Getting Started

- We provide you the create file
- New data loading file
- Goals
    1. Perform SQL to “combine data” from two tables
    2. Add foreign key constraints
    3. Add general constraints
    4. Write unit tests for constraints
    5. Create and query a view
    6. Create an index

What to turn in?

- combine.sql : Combine the new table ModifyReservations into Reservations
- foreign.sql: Add new foreign key constraints
- general.sql: Add new general constrains
- unittests.sql: Add unit tests 9, 2 each for all the constraints + 3 for foreign keys
- createview.sql: create a view
- queryviews.sql: Query across the view, delete data and re-query
- createindex.sql: Create an index

Combining Data

- There is a new table

```
CREATE TABLE NewReadArticles(
    subscriberPhone INT,
    editionDate DATE,
    articleNum INT,
    readInterval INTERVAL,
    PRIMARY KEY (subscriberPhone, editionDate, articleNum,
   readInterval),
    FOREIGN KEY subscriberPhone REFERENCES Subscribers,
    FOREIGN KEY (editionDate, articleNum) REFERENCES Articles
);
```

Merging into Read Articles:

- If (subscriberPhone, editionDate, articleNum values) is **not** in ReadArticles:
  - Insert it into ReadArticles
- If (subscriberPhone, editionDate, articleNum values) **is** in ReadArticles:
  - Update ReadInterval by adding the new interval

Transaction Syntax

```
BEGIN TRANSACTION ISOLATION LEVEL SERIALIZABLE;
Statement 1;
Statement 2;
Statement 3;
COMMIT TRANSACTION;
```

Transaction Isolation levels

Table 13.1. Transaction Isolation Levels

Isolation Level	Dirty Read	Nonrepeatable Read	Phantom Read	Serialization Anomaly
Read uncommitted	Allowed, but not in PG	Possible	Possible	Possible
Read committed	Not possible	Possible	Possible	Possible
Repeatable read	Not possible	Not possible	Allowed, but not in PG	Possible
Serializable	Not possible	Not possible	Not possible	Not possible

https://www.postgresql.org/docs/current/transaction-iso.html

Order of operations

- Consider carefully the order in which statements are laid out within a transaction
- Delete first, Update first, or insert first?
- Potential for deadlocks
- What happens when you try to update something that you just deleted?
- What happens when you try to update something that was just inserted?
- What happens when an update fails?

Example of a combine

- Let's say we have updated information on table bars with our table newbars:

-
```
CREATE TABLE Bars (
  bar VARCHAR(30),
  addr VARCHAR(50),
  license VARCHAR(50),
  PRIMARY KEY (bar)
);
```

```
CREATE TABLE newBars (
  bar VARCHAR(30),
  addr VARCHAR(50),
  license VARCHAR(50),
  PRIMARY KEY (bar)
);
```

Bars Table and NewBars Table

```
dpuranda=# SELECT * FROM Bars;
      bar       |       addr        | license
----------------+-------------------+---------
 Joes           | 123 Any Street    | B7462A
 Sues           | 456 My Way        | C5473S
 The Red Room   | 213 Front St.     | NM334D
 515 Cocktails  | 515 Front St.     | S112F1
(4 rows)
```

```
dpuranda=# SELECT * FROM NewBars;
      bar       |        addr          | license
----------------+----------------------+---------
 Mountain       |  328 Wallaby Way     | C5073S
 The Red Room   | 213 Front St.        | NM344D
(2 rows)
```

Writing the Insert and the Update

```
INSERT INTO Bars
SELECT nb.bar, nb.addr, nb.license
FROM newbars nb
WHERE NOT EXISTS ( SELECT *
                   FROM Bars b
                   WHERE b.bar = nb.bar);
```

Writing the Insert and the Update

```
UPDATE Bars b
SET addr = nb.addr, license = nb.license
FROM newbars nb
WHERE b.bar = nb.bar;
```

Putting Together into a Transaction

```
BEGIN TRANSACTION ISOLATION LEVEL SERIALIZABLE;
UPDATE Bars b
SET addr = nb.addr, license = nb.license
FROM newbars nb
WHERE b.bar = nb.bar;
INSERT INTO Bars
SELECT nb.bar, nb.addr, nb.license
FROM newbars nb
WHERE NOT EXISTS ( SELECT *
        FROM Bars b
        WHERE b.bar = nb.bar);
COMMIT TRANSACTION;
```

Adding Foreign key constraints

```
ALTER TABLE tablename
ADD FOREIGN KEY key
REFERENCES table(key)
  ON UPDATE action
  ON DELETE action;
```

Types of actions

- NO ACTION: Produces an error on referential integrity violation
- RESTRICT: Same as NO ACTION but cannot be DEFERRED
- CASCADE: Delete/Update any rows referring to deleted/updated rows
- SET NULL: Set referencing columns to NULL
- SET DEFAULT: Set referencing columns to their default values

`https://www.postgresql.org/docs/current/sql-createtable.html`

Hints and getting started

- Use the new create and loading data
- Your foreign key constraints and other checks could affect the state of tables
- Create tables and load data again before working on different parts of these assignments
- Carefully consider the order of operations in a transaction
- Get help early!
- This lab is easier than lab2!