CMSC 424 – Database design
Lecture 8
SQL, constraints

Mihai Pop
Next:

Integrity constraints

Prevent semantic inconsistencies
IC’s

Predicates on the database
Must always be true (checked whenever db gets updated)

There are the following 4 types of IC’s:

*Key constraints (1 table)*
  e.g., 2 accts can’t share the same acct_no

*Attribute constraints (1 table)*
  e.g., accts must have nonnegative balance

*Referential Integrity constraints (2 tables)*
  E.g., bnames associated w/ loans must be names of real branches

*Global Constraints (n tables)*
  E.g., all loans must be carried by at least 1 customer with a svngs acct
Key Constraints

Idea: specifies that a relation is a set, not a bag

SQL examples:

1. Primary Key:
   ```sql
   CREATE TABLE branch(
     bname  CHAR(15)  PRIMARY KEY,
     bcity  CHAR(20),
     assets INT);
   ```
   or
   ```sql
   CREATE TABLE depositor(
     cname   CHAR(15),
     acct_no  CHAR(5),
     PRIMARY KEY(cname, acct_no));
   ```

2. Candidate Keys:
   ```sql
   CREATE TABLE customer ( 
     ssn    CHAR(9)  PRIMARY KEY,
     cname  CHAR(15),
     address CHAR(30),
     city   CHAR(10),
     UNIQUE (cname, address, city));
   ```
Key Constraints

Effect of SQL Key declarations

PRIMAR Y (A1, A2, .., An) or
UNIQUE (A1, A2, ..., An)

Insertions: check if any tuple has same values for A1, A2, .., An as any inserted tuple. If found, reject insertion

Updates to any of A1, A2, ..., An: treat as insertion of entire tuple

Primary vs Unique (candidate)

1. 1 primary key per table, several unique keys allowed.
2. Only primary key can be referenced by “foreign key” (ref integrity)
3. DBMS may treat primary key differently
   (e.g.: create an index on PK)

How would you implement something like this?
Attribute Constraints

Idea:

*Attach constraints to values of attributes*
*Enhances types system (e.g.: >= 0 rather than integer)*

In SQL:

1. **NOT NULL**
   
   e.g.: `CREATE TABLE branch(
          bname CHAR(15) NOT NULL,
          ....
       )`
   
   Note: declaring `bname` as primary key also prevents null values

2. **CHECK**
   
   e.g.: `CREATE TABLE depositor(
           ....
           balance int NOT NULL,
           CHECK( balance >= 0),
           ....
        )`

   affect insertions, update in affected columns
Attribute Constraints

Domains: can associate constraints with DOMAINS rather than attributes

e.g: instead of: 

```sql
CREATE TABLE depositor(

    ....
    balance INT NOT NULL,
    CHECK (balance >= 0)
)
```

One can write:

```sql
CREATE DOMAIN bank-balance INT ( 
    CONSTRAINT not-overdrawn CHECK (value >= 0), 
    CONSTRAINT not-null-value CHECK( value NOT NULL));
```

```sql
CREATE TABLE depositor ( 

    ....
    balance  bank-balance,
)
```

Advantages?
Attribute Constraints
Advantage of associating constraints with domains:

1. can avoid repeating specification of same constraint for multiple columns

2. can name constraints
e.g.: CREATE DOMAIN bank-balance INT (CONSTRAINT not-overdrawn CHECK (value >= 0), CONSTRAINT not-null-value CHECK( value NOT NULL));

allows one to:
1. add or remove:
   ALTER DOMAIN bank-balance
   ADD CONSTRAINT capped
   CHECK( value <= 10000)
2. report better errors (know which constraint violated)
## Joins

### Natural (inner) join

<table>
<thead>
<tr>
<th>account name</th>
<th>acc_no</th>
<th>loan name</th>
<th>loan_no</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>101</td>
<td>Bob</td>
<td>L1</td>
</tr>
<tr>
<td>Bob</td>
<td>102</td>
<td>Jane</td>
<td>L3</td>
</tr>
<tr>
<td>Jane</td>
<td>107</td>
<td>Harry</td>
<td>L4</td>
</tr>
<tr>
<td>Janice</td>
<td>109</td>
<td>Tom</td>
<td>L7</td>
</tr>
</tbody>
</table>

### Outer joins (left, right)

<table>
<thead>
<tr>
<th>account name</th>
<th>acc_no</th>
<th>loan name</th>
<th>loan_no</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>101</td>
<td>Bob</td>
<td>L1</td>
</tr>
<tr>
<td>Bob</td>
<td>102</td>
<td>Jane</td>
<td>L3</td>
</tr>
<tr>
<td>Jane</td>
<td>107</td>
<td>Harry</td>
<td>L4</td>
</tr>
<tr>
<td>Janice</td>
<td>109</td>
<td>Tom</td>
<td>L7</td>
</tr>
</tbody>
</table>
SQL Query Examples

- Movie(\textit{title, year, length, inColor, studioName, producerC#})
- StarsIn(movieTitle, movieYear, starName)
- MovieStar(\textit{name, address, gender, birthdate})
- MovieExec(name, address, \textit{cert#, netWorth})
- Studio(\textit{name, address, presC#})

- Queries:
  - Producer with maximum average length of movies
  - Find producer of Star Wars.
  - All producers of movies in which harrison ford stars
SQL Query Examples

- Movie(title, year, length, inColor, studioName, producerC#)
- StarsIn(movieTitle, movieYear, starName)
- MovieStar(name, address, gender, birthdate)
- MovieExec(name, address, cert#, netWorth)
- Studio(name, address, presC#)

Queries:
- Find movie titles that appear more than once
- Find number of people 3 hops away from Kevin Bacon