Schedule

• Today Oct. 23 (T) Constraints.
  ◆ Read Sections 7.1-7.2. Assignment 4 due.
• Project Part 3 due Oct. 24 (W).
• Oct. 25 (TH) More Constraints, Triggers.
  ◆ Read Sections 7.3-7.4.
• Oct. 30 (T) Embedded SQL.
  ◆ Read Section 8.1. Assignment 5 due. Project Part 4 due.
• Nov. 1 (TH) Midterm
  ◆ Covers material through Oct. 25 (TH) lecture and readings (Chapters 1-3, 5-7).
Project Part 3

• Extended deadline: Oct. 24 (W) midnight (because of power outages).
• Use submit (use submit -m for instructions)
  ✦ Doesn’t yet work on mandelbrot
  ✦ FTP your files to teach or learn and then submit in the meanwhile
• I have office hours today; I’ll arrange for some on Wednesday.
Constraints

Commercial relational systems allow much more “fine-tuning” of constraints than do the modeling languages we learned earlier.

- In essence: SQL programming is used to describe constraints.

Outline

1. Primary key declarations (already covered).
2. Foreign-keys = referential integrity constraints.
3. Attribute- and tuple-based checks = constraints within relations.
4. SQL Assertions = global constraints.
   - Not found in Oracle.
5. Oracle Triggers.
   - A substitute for assertions.
Foreign Keys

In relation $R$ a clause that “attribute $A$ references $S(B)$” says that whatever values appear in the $A$ column of $R$ must also appear in the $B$ column of relation $S$.

- $B$ must be declared the primary key for $S$.

Example

```sql
CREATE TABLE Beers (  
    name CHAR(20) PRIMARY KEY,  
    manf CHAR(20)  
);  

CREATE TABLE Sells (  
    bar CHAR(20),  
    beer CHAR(20) REFERENCES Beers(name),  
    price REAL  
);  
```
Alternative: add another element declaring the foreign key, as:

```
CREATE TABLE Sells (  
  bar CHAR(20),  
  beer CHAR(20),  
  price REAL,  
  FOREIGN KEY beer REFERENCES Beers (name)  
);  
```

• Extra element essential if the foreign key is more than one attribute.
What Happens When a Foreign Key Constraint is Violated?

- Two ways:
  1. Insert or update a Sells tuple so it refers to a nonexistent beer.
     - Always rejected.
  2. Delete or update a Beers tuple that has a beer value some Sells tuples refer to.
     a) Default: reject.
     b) Cascade: Ripple changes to referring Sells tuple.

Example

- Delete “Bud.” Cascade deletes all Sells tuples that mention Bud.
- Update “Bud” to “Budweiser.” Change all Sells tuples with “Bud” in beer column to be “Budweiser.”
c) *Set Null*: Change referring tuples to have NULL in referring components.

**Example**

- Delete “Bud.” Set-null makes all *Sells* tuples with “Bud” in the *beer* component have NULL there.
- Update “Bud” to “Budweiser.” Same change.
Selecting a Policy

Add `ON [DELETE, UPDATE] [CASCADE, SET NULL]` to declaration of foreign key.

Example

```
CREATE TABLE Sells (  
  bar CHAR(20),  
  beer CHAR(20),  
  price REAL,  
  FOREIGN KEY beer REFERENCES Beers(name)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE  
) ;
```

- “Correct” policy is a design decision.
  - E.g., what does it mean if a beer goes away? What if a beer changes its name?
Attribute-Based Checks

Follow an attribute by a condition that must hold for that attribute in each tuple of its relation.

• Form: CHECK (condition).
  ◆ Condition may involve the checked attribute.
  ◆ Other attributes and relations may be involved, but only in subqueries.
  ◆ Oracle: No subqueries allowed in condition.

• Condition is checked only when the associated attribute changes (i.e., an insert or update occurs).
Example

CREATE TABLE Sells (  
    bar CHAR(20),  
    beer CHAR(20) CHECK(  
        beer IN (SELECT name  
                FROM Beers)  
    ),  
    price REAL CHECK(  
        price <= 5.00  
    )  
);  

• Check on beer is like a foreign-key constraint, except:  
  • The check occurs only when we add a tuple or change the beer in an  
    existing tuple, not when we delete a tuple from Beers.
Tuple-Based Checks

Separate element of table declaration.

• Form: like attribute-based check.

• But condition can refer to any attribute of the relation.
  ◆ Or to other relations/attributes in subqueries.
  ◆ Again: Oracle forbids the use of subqueries.

• Checked whenever a tuple is inserted or updated.
**Example**

Only Joe's Bar can sell beer for more than $5.

```sql
CREATE TABLE Sells ( 
    bar CHAR(20),
    beer CHAR(20),
    price REAL,
    CHECK(bar = 'Joe''s Bar' OR 
          price <= 5.00)
);
```
SQL Assertions

• Database-schema constraint.
• Not present in Oracle.
• Checked whenever a mentioned relation changes.
• Syntax:
  CREATE ASSERTION < name>
  CHECK (<condition>) ;
Example

No bar may charge an average of more than $5 for beer.

\[
\text{Sells}(\text{bar, beer, price})
\]

\[
\text{CREATE ASSERTION NoRipoffBars}
\]

\[
\text{CHECK}(\text{NOT EXISTS (}}
\]

\[
\text{SELECT bar}
\]

\[
\text{FROM Sells}
\]

\[
\text{GROUP BY bar}
\]

\[
\text{HAVING } 5.0 < \text{AVG} (\text{price})
\]

\[
\text{)}
\]

• Checked whenever \text{Sells} changes.
Example

There cannot be more bars than drinkers.

Bars(name, addr, license)
Drinkers(name, addr, phone)

CREATE ASSERTION FewBar
CHECK(
    (SELECT COUNT(*) FROM Bars) <=
    (SELECT COUNT(*) FROM Drinkers)
);

• Checked whenever Bars or Drinkers changes.