



# THE LECTURE 6



DATABASE MODIFICATIONS

# 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

```
CREATE TABLE Apples (  
    name CHAR(20) PRIMARY KEY,  
    manf CHAR(20)  
);  
  
CREATE TABLE Sells (  
    shop CHAR(20),  
    apple CHAR(20) REFERENCES Apples(name),  
    price REAL  
);
```

## FOREIGN KEYS

Alternative: add another element declaring the foreign key, as:

```
CREATE TABLE Sells (  
    shop CHAR(20),  
    apple CHAR(20),  
    price REAL,  
    FOREIGN KEY name REFERENCES  
        Apples(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 `Apples` tuple that has an apple value some `Sells` tuples refer to.
    - a) Default: reject.
    - b) *Cascade*: Ripple changes to referring `Sells` tuple.

## Example

- Delete “Green” Cascade deletes all `Sells` tuples that mention Green.
- Update “Green” to “GreenWood” Change all `Sells` tuples with “Green” in apple column to be “GreenWood”

## SELECTING A POLICY

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

### Example

```
CREATE TABLE Sells (  
    shop CHAR(20),  
    apple CHAR(20),  
    price REAL,  
    FOREIGN KEY apple REFERENCES Apples(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 (  
    shop CHAR(20),  
    apple CHAR(20) CHECK(  
        apple IN (SELECT name  
            FROM Apples)  
    ),  
    price REAL CHECK(  
        price <= 5.00  
    )  
);
```

- Check on `apple` is like a foreign-key constraint, except:
  - The check occurs only when we add a tuple or change the apple in an existing tuple, not when we delete a tuple from `Apples`.



# 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 apple for more than \$5.

```
CREATE TABLE Sells (  
    shop CHAR(20),  
    apple CHAR(20),  
    price REAL,  
    CHECK (shop = 'Joe''s Shop' OR  
           price <= 5.00)  
);
```