Chapter 9

Database Integrity

Introduction

- Integrity refers to the correctness or accuracy of data in the database
- For examples:
  - In Supplier-Part-Project database, the status values might have to be in the range of 1 to 100
  - Shipment quantities might have to be a multiple of 50
  - Red parts might have to be stored in London
- Database needs to be informed of such constraints and needs to be enforced

Introduction

Status values must be in the range 1 to 100

```
CONSTRAINT SC3
    IS_EMPTY (S WHERE STATUS < 1 OR STATUS >100)
```

- This constraint will registered in the system catalog under that name
- Any attempt to violate the constraint will be rejected

Introduction

- Enforcement of integrity
  - Declarative approach
  - Stored or trigger procedure approach (a precompiled procedures)
- Constraint classification
  - Type constraints
  - Attribute constraints
  - RELVAR constraints
  - Database constraints

Type Constraints

- An enumeration of the legal values of the type

  TYPE WEIGHT POSSREP (RATIONAL)

  CONSTRAINT THE_WEIGHT (WEIGHT) > 0.0

  WEIGHT can be represented in rational number and is greater than zero.

Attribute Constraints

- A declaration to the effect that a specified attribute is of a specified type.

```
VAR S BASE RELATION
   { S#              S#,
     SNAME           NAME,
     STATUS          INTEGER,
     CITY            CHAR };
```

RELVAR Constraints

- A constraint on an individual relvar

```
CONSTRAIN SC5
    IS_EMPTY ( S WHERE CITY = 'London' AND
      STATUS <> 20 )
```

(Suppliers in London must have status 20)

```
CONSTRAIN SCK
    COUNT (S) = COUNT (S {S#} )
```

(Supplier numbers are unique)

Relvar constraints are always checked immediately.

Database Constraints

- A constraint that interrelates two or more distinct relvars

```
CONSTRAINT DBC1
        IS_EMPTY ( ( S JOIN P )
                WHERE STATUS < 20 AND QTY > QTY(500) )
```

No supplier with status less than 20 can supply any part in a quantity greater that 500.

CONSTRAINT DBC2 SPJ (S#) <= S{S#}

Every supplier number in the shipments relvar also exists in the supplier relvar.

Database constraints are checked at COMMIT time.

KEYS

- Candidate keys
  - Let K be a set of attributes of relvar R. The K is a candidate key for if and only if it possesses both of the following properties:

    a). Uniqueness: No legal value of R ever contains two distinct tuples with the same values for K

    b). Irreducibility: No proper subset of K has the uniqueness property

{ S#, CITY}

{ S#} is a candidate key

Candidate Keys

TUTORIAL D: KEY {<attribute name commalist>}

```
VAR SPJ BASE RELATION
   {     S#       S#,
         P#       P#,
         J#       J#,
         QTY      QTY}
         KEY {S#, P#, J#};          // Composite candidate key

VAR ELEMENT BASE RELATION {    NAME     NAME,
                               SYMBOL CHAR,
                               ATOMIC# INTEGER }
                               KEY { NAME }
                               KEY { SYMBOL }
                               KEY { ATOMIC# };
```

Candidate Keys

```
VAR MARRIAGE BASE RELATION {   HUSBAND        NAME,
                               WIFE           NAME,
                               DATE           DATE}
/* assume no polyandry, no polygyny, and no husband and wife marry
     each other more than once */
                               KEY { HUSBAND, DATE}
                               KEY { DATE, WIFE }
                               KEY { WIFE, HUSBAND};
```

Candidate Keys

- Candidate keys provide the basic tuple-level addressing mechanism in the relation model

  For example:

  S WHERE S# = S#('S3')

  is guaranteed to yield at most one tuple.

  S WHERE CITY = 'Paris'

  yields an unpredictable number of tuples.

Superkeys

- A superset of a candidate key is a superkey

  {S#, CITY} is a superkey for relval S

  {S#, SNAME, CITY} is a superkey for relval S

Primary Keys & Alternate Keys

- Exactly one of the candidate keys is chosen as primary key
- And other candidate keys are called alternate keys
- Choice of primary key from candidate keys is arbitrary
- Every base RELVAR has always a primary key

Foreign Keys

- A foreign key is a set of attributes of one relvar R2 whose values are required to match values of some candidate key of some relvar R1.
  - For example: S# in SPJ
- FORMAL DEFINITION:
  - Let R2 be a relvar. Then a foreign key in R2 is a set of attributes of R2, day FK such that

    a). There exists a relvar R1 with a candidate key CK, and

    b). For all time, each value of FK in the current value of R2 is identical to the value of CK in some tuple in the currrent value of R1

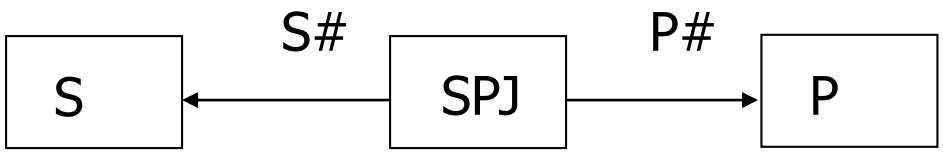
Notes about Foreign Keys

- Every value of a given foreign key appears as a value of the matching candidate keys. The converse is not a requirement.
- A foreign key is simple or composite depending on the candidate key it matches
- Each attribute of a given foreign key must have the same name and type as the matching candidate key

Referential Integrity

- The problem of ensuring that the database does not include any invalid foreign key values is known as REFERENTIAL INTEGRITY
- Referential Constraint:
  - Values of a given foreign key must match values of the corresponding candidate key.
- Referencing relvar: The RELVAR contains foreign key
- Referenced relvar: The RELVAR contains candidate key

Referential Diagram



R3 → R2 →R1

R(n) → R(n-1) →… → R2 → R1

R(n) → R(n-1) →… → R2 → R1 →R(n)

Referential Diagram

TABLE EMP {SSN, MGR_EMP} /* self-referencing */

R(n) → R(n-1) →… → R2 → R1 →R(n) /* referential cycle */

Referential Actions

DELETE S WHERE S# = S#('S1')

Assume:

- The database includes some shipments for supplier S1
- No shipments for supplier S1 are deleted
- When referential constraint from shipment to supplier is checked, a violation will be found

Referential Actions

```
VAR SPJ BASE RELATION {…}
        FOREIGN KEY {S# } REFERENCE S
                ON DELETE CASCADE;
```

- CASCADE: automatically delete shipments for supplier S1 when deleting S1 in S
- RESTRICT: DELETE operations are "restricted" to the cases where there are no matching shipments, otherwise they are rejected.
- NO ACTION: Perform as requested
- Rules are applied to UPDATE operation

SQL Facilities

- Candidate Keys
  - UNIQUE (<column name commalist>) OR
  - PRIMARY KEY(<column name commalist>)

```
CREATE TABLE SP (
    S# S# NOT NULL, P# P# NOT NULL, QTY QTY NOT NULL,
    PRIMARY KEY (S#, P#),
    FOREIGN KEY (S#) REFERENCE S ON DELETE CASCADE
                                    ON UPDATE CASCADE,
    FOREIGN KEY (P#) REFERENCE P ON DELETE CASCADE
                                    ON UPDATE CASCADE,
    CHECK (QTY >0 AND QTY < 5001));
```

SQL Facilities

- Assertions
  - General constraints

  - Every supplier has status at least five:

```
CREATE ASSERTION IC13 CHECK
  ( (SELECT MIN(STATUS) FROM S) > 4)
```

  - All red parts must be stored in London

```
CREATE ASSERTION IC99 CHECK
  ( NOT EXISTS ( SELECT * FROM P
                 WHERE COLOR = 'Red'
                 AND CITY <> 'London'
```