Database Systems

Lecture 10,11,12

Chapter # 6

Basic SQL

Chapter Outlines

- **SQL**
- **Attribute Data Types**
- **SQL Data Definition**
- **Specifying Constraints in SQL**
- **Basic Retrieval Queries in SQL**
- **INSERT, DELETE, and UPDATE Statements in SQL**
- **Additional Features of SQL**

Basic SQL(Structured Query language)

- SQL Actually comes from the word “SEQUEL” which was the original term used in the paper: “SEQUEL TO SQUARE” by Chamberlin and Boyce. Pronounced as S-Q-L or sometimes as See-Quell.
- SQL (Structured Query Language) is used to perform operations on the records stored in the database, such as:
  - Updating records
  - Inserting records
  - Deleting records
  - Creating and modifying database tables, views, etc.

Attribute Data Types

- Numeric data types

SMALLINT(size)	A small integer. Signed range is from -32768 to 32767. Unsigned range is from 0 to 65535. The size parameter specifies the maximum display width (which is 255)
INT	A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The size parameter specifies the maximum display width (which is 255)
INTEGER	Equal to INT(size)
FLOAT(p)	A floating point number. MySQL uses the p value to determine whether to use FLOAT or DOUBLE for the resulting data type. If p is from 0 to 24, the data type becomes FLOAT(). If p is from 25 to 53, the data type becomes DOUBLE()
DOUBLE PRECISION	
Number(p,s)	Exact numerical, precision p, scale s. The maximum precision depends on the DBMS.
DECIMAL(size, d)	An exact fixed-point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter. The maximum number for size is 65. The maximum number for d is 30. The default value for size is 10. The default value for d is 0.
DEC(size, d)	Equal to DECIMAL(size,d)

Attribute Data Types

- String Data Types

CHAR(size)	A FIXED length string (can contain letters, numbers, and special characters). The size parameter specifies the column length in characters - can be from 0 to 255. Default is 1
VARCHAR(size)	A VARIABLE length string (can contain letters, numbers, and special characters). The size parameter specifies the maximum column length in characters - can be from 0 to 65535
Varchar2(Size)	The VARCHAR2 datatype stores variable-length character strings. When you create a table with a VARCHAR2 column, you specify a maximum string length (in bytes or characters) between 1 and 4000 bytes for the VARCHAR2 column.

Date and Time Data Types

DATE	Represents a date. Format : dd-mm-yyyy, SYSDATE
TIMESTAMP	Format: dd-mm-yyyy hh:mm:ss, CURRENT_TIMESTAMP
TIMESTAMP WITH TIME ZONE	Represents a combination of DATE and TIME values separated by a space with time zone. Format : dd-mm-yy hh:mm:ss AT TIME ZONE -06:00.

Attribute Data Types

- **Binary large object Type:**

BINARY LARGE OBJECT (BLOB).	BLOB stores a long sequence of bytes. For storing large pictures and sound etc.

SQL Data Definition

- The **CREATE TABLE** command is used to specify a new relation by giving it a name and specifying its attributes and initial constraints.
- Syntax

```
CREATE TABLE table_name (
    column_name datatype constraint_name,
    column_name datatype constaint_name,
    column_name datatype constaint_name,
   ....
);
```

SQL Data Definition

- Base tables (base relations)
  - Relation and its tuples are actually created and stored as a file by the DBMS.
- Virtual relations (views)
  - Created through the CREATE VIEW statement. Do not correspond to any physical file.

Specifying Attribute Constraints and Attribute Defaults

- Relational Model has 3 basic constraint types that are supported in SQL:
  - Key constraint: A primary key column cannot have duplicate value
  - Entity Integrity Constraint: A primary key value cannot be null
  - Referential integrity constraints : The “foreign key “ must have a value that is already present as a primary key, or may be null.

Specifying Attribute Constraints and Attribute Defaults

- **NOT NULL:** specified if NULL is not permitted for a particular attribute value.
- **CHECK:** Restricting values in tuples.
  - Check is a row-based constraint.
  - Applies to each row individually and are checked whenever a row is inserted or modified.
  - Dnumber INT NOT NULL CHECK (Dnumber > 0 AND Dnumber < 21);
- **DEFAULT:** default value is included in any new tuple if an explicit value is not provided for that attribute.
  - Specified using DEFAULT <value>

Specifying Key and Referential Integrity Constraints

- **UNIQUE:** Values in the column must be unique.
  - Specifies alternate (secondary) keys (called CANDIDATE keys in the relational model).
  - Dname VARCHAR(15) UNIQUE;
- **PRIMARY KEY:** specifies one or more attributes that make up the primary key of a relation.
  - For only one column as a Primary key in a table.
    - Dnumber INT PRIMARY KEY
  - For more than one column as Primary Key.
    - PRIMARY KEY (Dnumber, Dlocation)

Specifying Key and Referential Integrity Constraints

- **FOREIGN KEY:** Referential integrity is specified via the FOREIGN KEY.
- **Default action for integrity Violation**: Reject the update operation (Restrict)
- Alternative action to be taken by attaching a **referential triggered action** clause to any foreign key constraint.
- The options include
  - SET NULL
  - CASCADE, and
  - SET DEFAULT
- An option must be qualified with either ON DELETE or ON UPDATE.

COMPANY relational database schema

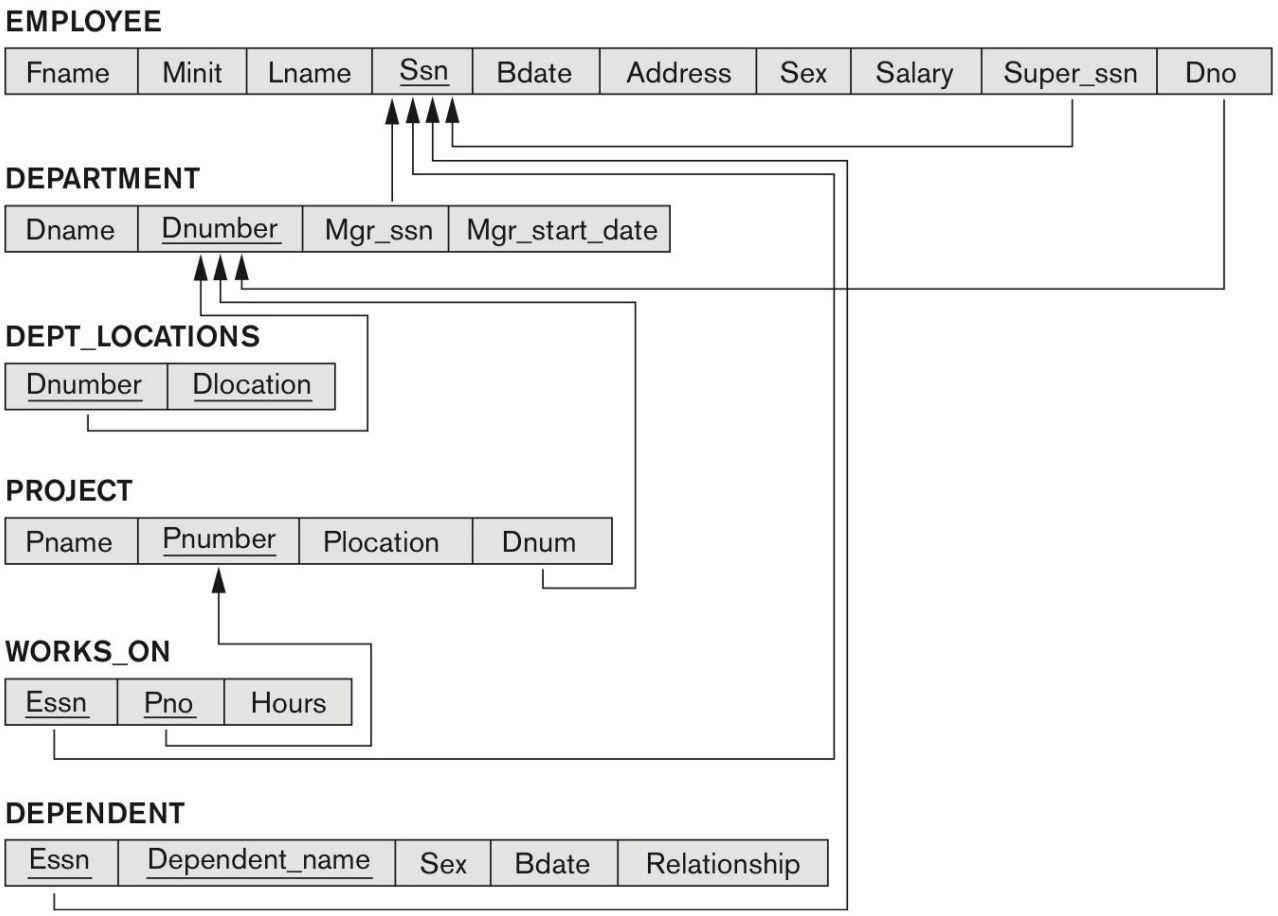

Figure 5.6

One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

```
CREATE TABLE EMPLOYEE
        ( Fname                 VARCHAR(15)             NOT NULL,
          Minit                 CHAR,
          Lname                 VARCHAR(15)             NOT NULL,
          Ssn                   CHAR(9)                 NOT NULL,
          Bdate                 DATE,
          Address               VARCHAR(30),
          Sex                   CHAR,
          Salary                DECIMAL(10,2),
          Super_ssn             CHAR(9),
          Dno                   INT                     NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)             NOT NULL,
          Dnumber               INT                     NOT NULL,
          Mgr_ssn               CHAR(9)                 NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                     NOT NULL,
          Dlocation             VARCHAR(15)             NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
```

SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 5.7 (Fig. 6.1)-continued

```
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)             NOT NULL,
          Pnumber               INT                     NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                     NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)                 NOT NULL,
          Pno                   INT                     NOT NULL,
          Hours                 DECIMAL(3,1)            NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)                 NOT NULL,
          Dependent_name        VARCHAR(15)             NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Create Table Using Another Table

- A copy of an existing table can also be created using CREATE TABLE.
- The new table gets the same column definitions. All columns or specific columns can be selected.
- Syntax:

```
CREATE TABLE new_table_name AS
    SELECT column1, column2,...
    FROM existing_table_name
    WHERE ....;
```

Example: Create Table Using Another Table

- Example

```
CREATE TABLE TestTable AS
SELECT customername, contactname
FROM customers;
```

SQL DROP TABLE Statement

- DROP TABLE table_name;
- Example:
  - DROP TABLE Persons;

Giving Names to Constraints

- Syntax:
- CONSTRAINT CONSTRAINT_NAME CONSTRAINT_TYPES
  - CONSTRAINT: Keyword
  - CONSTRAINT_NAME: Name of the constraint

```
CREATE TABLE EMPLOYEE
    ( … ,
      Dno          INT          NOT NULL       DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                    ON DELETE SET NULL          ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                    ON DELETE SET DEFAULT     ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( … ,
      Mgr_ssn CHAR(9)          NOT NULL       DEFAULT '888665555',
      … ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
      FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
                    ON DELETE SET DEFAULT     ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
    ( … ,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
                    ON DELETE CASCADE         ON UPDATE CASCADE);
```

Example

- Consider the following relations for a database that keeps track of business trips of salespersons in a sales office:
- **SALESPERSON(Ssn, Name, Start_year, Dept_no)**
- **TRIP(Ssn, From_city, To_city, Departure_date, Return_date, Trip_id)**
- **EXPENSE(Trip_id, Account#, Amount)**
- SSN can be upto four digits.
- Start year and Department number of Salesperson must not be null.
- From_city, To_city, Departure_date, Return_date must not be null.
- Trip_id and amount can be upto 6 digits.
- Account # must be of 14 digits.

Basic SQL Queries: The SELECT-FROM-WHERE Structure

SELECT <attribute list>

FROM <table list>

WHERE <condition>;

OR

SELECT <column1>, <column2>…..<column n> FROM <table1>;

- where
- <attribute list> is a list of attribute names whose values are to be retrieved by the query.
- <table list> is a list of the relation names required to process the query.
- <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query.

Basic SQL Queries: The SELECT-FROM-WHERE Structure

- Logical comparison operators in SQL
  - =, <, <=, >, >=, and <>
- Sample Query

Query 0. Retrieve the birth date and address of the employee(s) whose name is 'John B. Smith'.

```
Q0:   SELECT   Bdate, Address
      FROM     EMPLOYEE
      WHERE    Fname='John' AND Minit='B' AND Lname='Smith';
```

Bdate	Address
1965-01-09	731Fondren, Houston, TX

Unspecified WHERE Clause and Use of the Asterisk

- Missing WHERE clause Indicates no condition on tuple selection, all the tuples are qualified for projection.
- If more than one relation is specified in the FROM clause and there is no WHERE clause, then the CROSS PRODUCT—all possible tuple combinations—of these relations is selected.

Queries 9 and 10. Select all EMPLOYEE Ssns (Q9) and all combinations of EMPLOYEE Ssn and DEPARTMENT Dname (Q10) in the databa

Q9:
```
SELECT  Ssn
FROM    EMPLOYEE;
```

Q10:
```
SELECT  Ssn, Dname
FROM    EMPLOYEE, DEPARTMENT;
```

(e)

E.Fname
123456789
333445555
999887777
987654321
666884444
453453453
987987987
888665555

(f)

Ssn	Dname
123456789	Research
333445555	Research
999887777	Research
987654321	Research
666884444	Research
453453453	Research
987987987	Research
888665555	Research
123456789	Administration
333445555	Administration
999887777	Administration
987654321	Administration
666884444	Administration
453453453	Administration
987987987	Administration
888665555	Administration
123456789	Headquarters
333445555	Headquarters
999887777	Headquarters
987654321	Headquarters
666884444	Headquarters
453453453	Headquarters
987987987	Headquarters
888665555	Headquarters

Unspecified WHERE Clause and Use of the Asterisk (cont'd.)

- Specify an asterisk (*) to retrieve all the attribute values of the selected tuples

```
Q1C:    SELECT    *
        FROM      EMPLOYEE
        WHERE     Dno=5;

Q1D:    SELECT    *
        FROM      EMPLOYEE, DEPARTMENT
        WHERE     Dname='Research' AND Dno=Dnumber;

Q10A:   SELECT    *
        FROM      EMPLOYEE, DEPARTMENT;
```

(g)

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-09-01	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

INSERT, DELETE, and UPDATE Statements in SQL

- INSERT: used to add a single tuple (row) to a relation (table).
- INSERT INTO Syntax:

  1. INSERT INTO table_name (col1,col2,col3,……)VALUES (value1, value2, value3, ...);

  2. INSERT INTO table_name VALUES (value1, value2, value3, ...);

Demo Database

Customer ID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland

INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');

Query Result

Customer ID	Customer Name	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

SQL INSERT INTO SELECT Statement

- The INSERT INTO SELECT statement copies data from one table and inserts it into another table.
- The INSERT INTO SELECT statement requires that the data types in source and target tables matches.
- **Copy all columns from one table to another table:**

```
INSERT INTO table2
SELECT * FROM table1
WHERE condition;
```

- **Copy only some columns from one table into another table:**

```
INSERT INTO table2 (column1, column2, column3, ...)
SELECT column1, column2, column3, ...
FROM table1
WHERE condition;
```

Demo Database

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

And a selection from the "Suppliers" table:

SupplierID	SupplierName	ContactName	Address	City	Postal Code	Country
1	Exotic Liquid	Charlotte Cooper	49 Gilbert St.	Londona	EC1 4SD	UK
2	New Orleans Cajun Delights	Shelley Burke	P.O. Box 78934	New Orleans	70117	USA
3	Grandma Kelly's Homestead	Regina Murphy	707 Oxford Rd.	Ann Arbor	48104	USA

Example

- INSERT INTO Customers (CustomerName, City, Country) SELECT SupplierName, City, Country FROM Suppliers;
- Example

  INSERT INTO Customers (CustomerName, City, Country)
  SELECT SupplierName, City, Country FROM Suppliers
  WHERE Country='Germany';

U2: **INSERT INTO** EMPLOYEE (Fname, Lname, Ssn, Dno)
VALUES ('Robert', 'Hatcher', '980760540', 2);
(U2 is rejected if referential integrity checking is provided by DBMS.)

U2A: **INSERT INTO** EMPLOYEE (Fname, Lname, Dno)
VALUES ('Robert', 'Hatcher', 5);
(U2A is rejected if NOT NULL checking is provided by DBMS.)

INSERT INTO SELECT

```
U3A:   CREATE TABLE    WORKS_ON_INFO
       ( Emp_name      VARCHAR(15),
         Proj_name     VARCHAR(15),
         Hours_per_week DECIMAL(3,1) );

U3B:   INSERT INTO     WORKS_ON_INFO ( Emp_name, Proj_name,
                       Hours_per_week )
       SELECT          E.Lname, P.Pname, W.Hours
       FROM            PROJECT P, WORKS_ON W, EMPLOYEE E
       WHERE           P.Pnumber = W.Pno AND W.Essn = E.Ssn;
```

The DELETE Command

- DELETE command removes tuples from a relation.

```
U4A:  DELETE FROM   EMPLOYEE
      WHERE         Lname = 'Brown';
U4B:  DELETE FROM   EMPLOYEE
      WHERE         Ssn = '123456789';
U4C:  DELETE FROM   EMPLOYEE
      WHERE         Dno = 5;
U4D:  DELETE FROM   EMPLOYEE;
```

The UPDATE Command

- UPDATE command is used to modify attribute values of one or more selected tuples.
- **Syntax:**

  UPDATE table_name

  SET column1 = value1, column2 = value2, ...

  WHERE condition;

```
U5:   UPDATE   PROJECT
      SET      Plocation = 'Bellaire', Dnum = 5
      WHERE    Pnumber = 10;
```

Update Case

```
CREATE TABLE project   (project_no   CHAR(4) NOT NULL,
 project_name CHAR(15) NOT NULL,
budget FLOAT NOT NULL);

insert into project values ('p1', 'Search Engine',       120000.00);
insert into project values ('p2', 'Programming',         95000.00);
insert into project values ('p3', 'SQL',                 186500.00);

UPDATE project SET budget = CASE
WHEN budget > 0 and budget < 100000 THEN budget* 1.2
WHEN budget > = 100000 and budget < 200000 THEN budget* 1.1
ELSE budget* 1.05
END;
```

Tables as Sets

- Use the keyword DISTINCT in the SELECT clause
  - To select rows or tuples with distinct values

Query 11. Retrieve the salary of every employee (Q11) and all distinct salary values (Q11A).

```
Q11:   SELECT   ALL Salary
       FROM     EMPLOYEE;

Q11A:  SELECT   DISTINCT Salary
       FROM     EMPLOYEE;
```

(a)

Salary
30000
40000
25000
43000
38000
25000
25000
55000

(b)

Salary
30000
40000
25000
43000
38000
55000

Tables as Sets in SQL

- Set operations
  - **UNION**, **EXCEPT** (difference), **INTERSECT**
  - Corresponding multiset operations: UNION ALL, EXCEPT ALL, INTERSECT ALL)
  - Type compatibility is needed for these operations to be valid

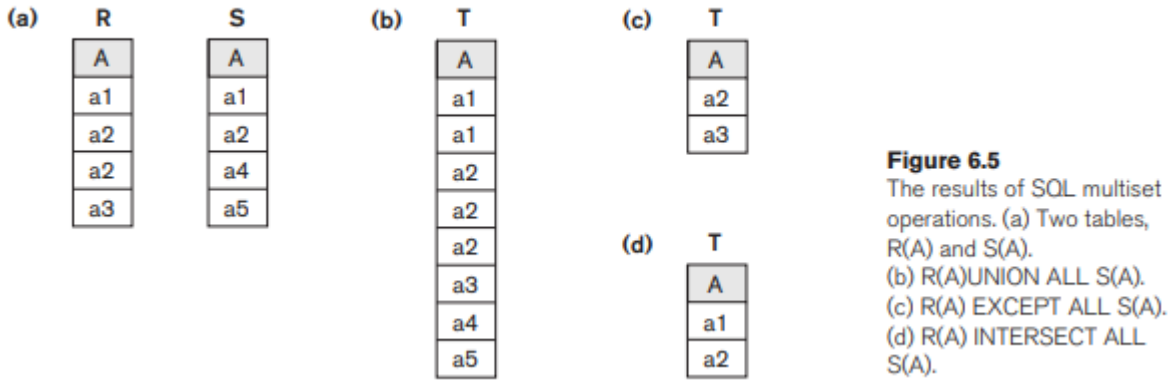

Figure 6.5
The results of SQL multiset operations. (a) Two tables, R(A) and S(A).
(b) R(A)UNION ALL S(A).
(c) R(A) EXCEPT ALL S(A).
(d) R(A) INTERSECT ALL S(A).