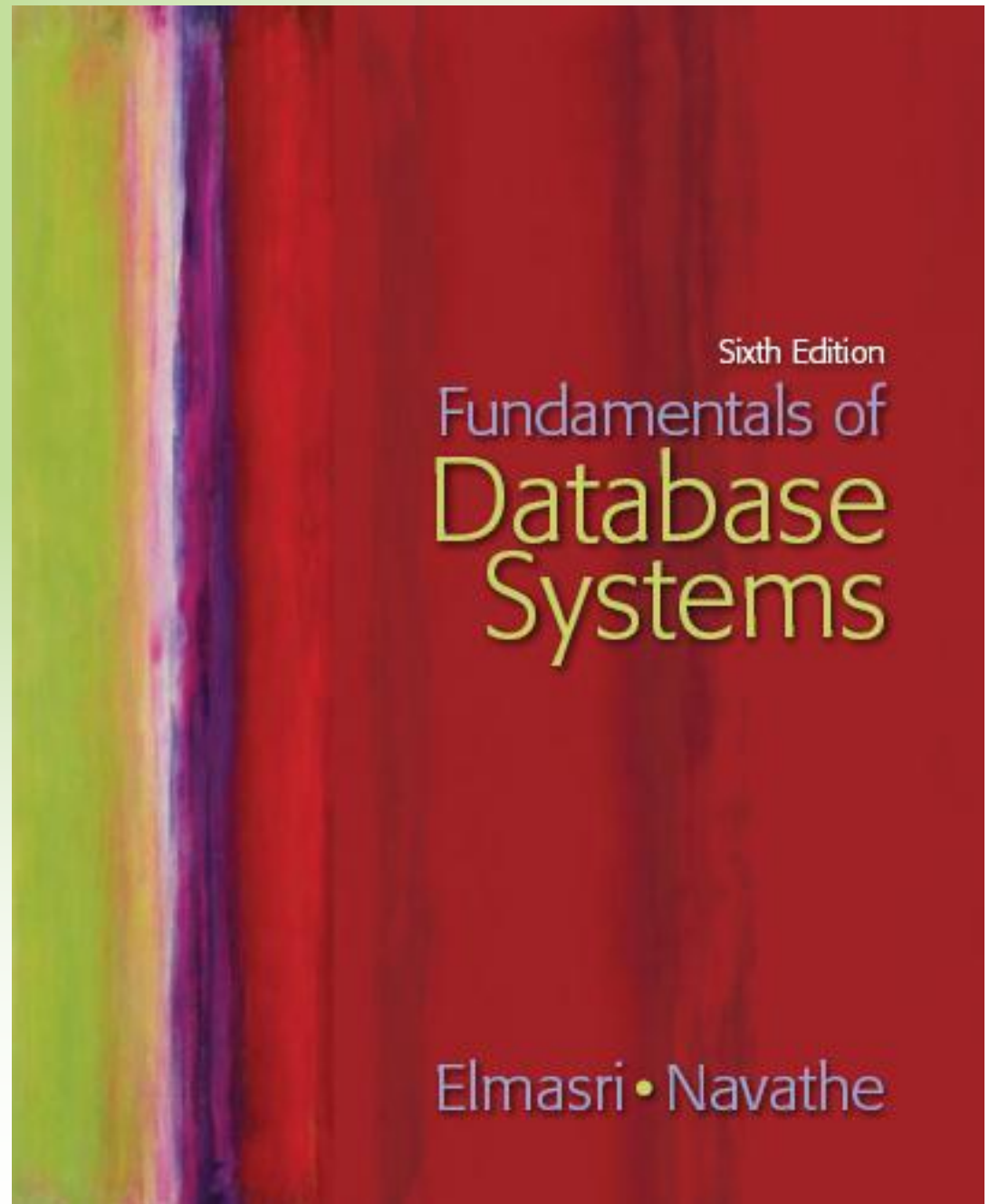


Chapter 4

Basic SQL



Sixth Edition

Fundamentals of Database Systems

Elmasri • Navathe

Addison-Wesley
is an imprint of

PEARSON

Chapter 4 Outline

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

Basic SQL

- SQL language
 - Considered one of the major reasons for the commercial success of relational databases
- **SQL**
 - **Structured Query Language**
 - Statements for data definitions, queries, and updates (both DDL and DML)
 - **Core specification**
 - Plus specialized **extensions**

SQL Data Definition and Data Types

- Terminology:
 - **Table**, **row**, and **column** used for relational model terms relation, tuple, and attribute
- CREATE statement
 - Main SQL command for data definition

Schema and Catalog Concepts in SQL

- **SQL schema**
 - Identified by a **schema name**
 - Includes an **authorization identifier** and **descriptors** for each element
- **Schema elements** include
 - Tables, constraints, views, domains, and other constructs
- Each statement in SQL ends with a **semicolon**

Schema and Catalog Concepts in SQL (cont'd.)

- **CREATE SCHEMA statement**
 - `CREATE SCHEMA COMPANY AUTHORIZATION 'Jsmith' ;`
- **Catalog**
 - Named collection of schemas in an SQL environment
- **SQL environment**
 - Installation of an SQL-compliant RDBMS on a computer system

The CREATE TABLE Command in SQL

- Specify a new relation
 - Provide name
 - Specify attributes and initial constraints
- Can optionally specify schema:
 - `CREATE TABLE COMPANY.EMPLOYEE ...`
 - or
 - `CREATE TABLE EMPLOYEE ...`

The CREATE TABLE Command in SQL (cont'd.)

- **Base tables (base relations)**
 - Relation and its tuples are actually created and stored as a file by the DBMS
- **Virtual relations**
 - Created through the `CREATE VIEW` statement

```

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),
  FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn),
  FOREIGN KEY (Dno) REFERENCES DEPARTMENT(Dnumber) );

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) );

```

Figure 4.1

SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 3.7.

```

CREATE TABLE DEPT_LOCATIONS
  ( Dnumber          INT          NOT NULL,
    Dlocation        VARCHAR(15)  NOT NULL,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );

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) );

```

Figure 4.1

SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 3.7.

The CREATE TABLE Command in SQL (cont'd.)

- Some foreign keys may cause errors
 - Specified either via:
 - Circular references
 - Or because they refer to a table that has not yet been created

Attribute Data Types and Domains in SQL

- **Basic data types**

- **Numeric data types**

- Integer numbers: `INTEGER`, `INT`, and `SMALLINT`
 - Floating-point (real) numbers: `FLOAT` or `REAL`, and `DOUBLE PRECISION`

- **Character-string data types**

- Fixed length: `CHAR (n)`, `CHARACTER (n)`
 - Varying length: `VARCHAR (n)`, `CHAR VARYING (n)`, `CHARACTER VARYING (n)`

Attribute Data Types and Domains in SQL (cont'd.)

- **Bit-string** data types
 - Fixed length: `BIT (n)`
 - Varying length: `BIT VARYING (n)`
- **Boolean** data type
 - Values of `TRUE` or `FALSE` or `NULL`
- **DATE** data type
 - Ten positions
 - Components are `YEAR`, `MONTH`, and `DAY` in the form `YYYY-MM-DD`

Attribute Data Types and Domains in SQL (cont'd.)

- Additional data types
 - **Timestamp** data type (`TIMESTAMP`)
 - Includes the `DATE` and `TIME` fields
 - Plus a minimum of six positions for decimal fractions of seconds
 - Optional `WITH TIME ZONE` qualifier
 - **INTERVAL** data type
 - Specifies a relative value that can be used to increment or decrement an absolute value of a date, time, or timestamp

Attribute Data Types and Domains in SQL (cont'd.)

- Domain
 - Name used with the attribute specification
 - Makes it easier to change the data type for a domain that is used by numerous attributes
 - Improves schema readability
 - Example:
 - `CREATE DOMAIN SSN_TYPE AS CHAR(9);`

Specifying Constraints in SQL

- Basic constraints:
 - Key and referential integrity constraints
 - Restrictions on attribute domains and NULLs
 - Constraints on individual tuples within a relation

Specifying Attribute Constraints and Attribute Defaults

- NOT NULL
 - NULL is not permitted for a particular attribute
- Default value
 - **DEFAULT** <value>
- **CHECK** clause
 - `Dnumber INT NOT NULL CHECK (Dnumber > 0 AND Dnumber < 21) ;`

```

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);

```

Figure 4.2

Example illustrating how default attribute values and referential integrity triggered actions are specified in SQL.

Specifying Key and Referential Integrity Constraints

- **PRIMARY KEY** clause
 - Specifies one or more attributes that make up the primary key of a relation
 - `Dnumber INT PRIMARY KEY;`
- **UNIQUE** clause
 - Specifies alternate (secondary) keys
 - `Dname VARCHAR(15) UNIQUE;`

Specifying Key and Referential Integrity Constraints (cont'd.)

- **FOREIGN KEY** clause
 - Default operation: reject update on violation
 - Attach **referential triggered action** clause
 - Options include `SET NULL`, `CASCADE`, and `SET DEFAULT`
 - Action taken by the DBMS for `SET NULL` or `SET DEFAULT` is the same for both `ON DELETE` and `ON UPDATE`
 - `CASCADE` option suitable for “relationship” relations

Giving Names to Constraints

- **Keyword CONSTRAINT**
 - Name a constraint
 - Useful for later altering

Specifying Constraints on Tuples Using CHECK

- CHECK clauses at the end of a CREATE TABLE statement
 - Apply to each tuple individually
 - CHECK (Dept_create_date <= Mgr_start_date);

Basic Retrieval Queries in SQL

- `SELECT` statement
 - One basic statement for retrieving information from a database
- SQL allows a table to have two or more tuples that are identical in all their attribute values
 - Unlike relational model
 - Multiset or bag behavior

The SELECT-FROM-WHERE Structure of Basic SQL Queries

- Basic form of the `SELECT` statement:

```
SELECT    <attribute list>  
FROM      <table list>  
WHERE     <condition>;
```

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.

The SELECT-FROM-WHERE Structure of Basic SQL Queries (cont'd.)

- Logical comparison operators
 - =, <, <=, >, >=, and <>
- **Projection attributes**
 - Attributes whose values are to be retrieved
- **Selection condition**
 - Boolean condition that must be true for any retrieved tuple

Figure 4.3

Results of SQL queries when applied to the COMPANY database state shown in Figure 3.6. (a) Q0. (b) Q1. (c) Q2. (d) Q8. (e) Q9. (f) Q10. (g) Q1C.

(a)

<u>Bdate</u>	<u>Address</u>
1965-01-09	731Fondren, Houston, TX

(b)

<u>Fname</u>	<u>Lname</u>	<u>Address</u>
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

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';
```

Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

```
Q1:    SELECT    Fname, Lname, Address
        FROM      EMPLOYEE, DEPARTMENT
        WHERE     Dname='Research' AND Dnumber=Dno;
```

Figure 4.3

Results of SQL queries when applied to the COMPANY database state shown in Figure 3.6. (a) Q0. (b) Q1. (c) Q2. (d) Q8. (e) Q9. (f) Q10. (g) Q1C.

(c)

<u>Pnumber</u>	<u>Dnum</u>	<u>Lname</u>	<u>Address</u>	<u>Bdate</u>
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

Query 2. For every project located in ‘Stafford’, list the project number, the controlling department number, and the department manager’s last name, address, and birth date.

```
Q2:    SELECT    Pnumber, Dnum, Lname, Address, Bdate
        FROM      PROJECT, DEPARTMENT, EMPLOYEE
        WHERE     Dnum=Dnumber AND Mgr_ssn=Ssn AND
                Plocation='Stafford';
```

Ambiguous Attribute Names

- Same name can be used for two (or more) attributes
 - As long as the attributes are in different relations
 - Must **qualify** the attribute name with the relation name to prevent ambiguity

```
Q1A:  SELECT  Fname, EMPLOYEE.Name, Address
        FROM    EMPLOYEE, DEPARTMENT
        WHERE   DEPARTMENT.Name='Research' AND
                DEPARTMENT.Dnumber=EMPLOYEE.Dnumber;
```

Aliasing, Renaming, and Tuple Variables

- **Aliases or tuple variables**
 - Declare alternative relation names E and S
 - `EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd, Addr, Sex, Sal, Sssn, Dno)`

Unspecified WHERE Clause and Use of the Asterisk

- Missing WHERE clause
 - Indicates no condition on tuple selection
- CROSS PRODUCT
 - All possible tuple combinations

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

Q9: **SELECT** Ssn
 FROM EMPLOYEE;

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

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

- Specify an asterisk (*)
 - 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;
```

Tables as Sets in SQL

- SQL does not automatically eliminate duplicate tuples in query results
- Use the keyword **DISTINCT** in the `SELECT` clause
 - Only distinct tuples should remain in the result

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;
```

Tables as Sets in SQL (cont'd.)

- Set operations
 - UNION, **EXCEPT** (difference), **INTERSECT**
 - Corresponding multiset operations: UNION ALL, EXCEPT ALL, INTERSECT ALL)

Query 4. Make a list of all project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.

```
Q4A: (SELECT DISTINCT Pnumber
      FROM PROJECT, DEPARTMENT, EMPLOYEE
      WHERE Dnum=Dnumber AND Mgr_ssn=Ssn
           AND Lname='Smith' )

      UNION
      (SELECT DISTINCT Pnumber
      FROM PROJECT, WORKS_ON, EMPLOYEE
      WHERE Pnumber=Pno AND Essn=Ssn
           AND Lname='Smith' );
```

Substring Pattern Matching and Arithmetic Operators

- **LIKE** comparison operator
 - Used for string **pattern matching**
 - % replaces an arbitrary number of zero or more characters
 - underscore (_) replaces a single character
- Standard arithmetic operators:
 - Addition (+), subtraction (−), multiplication (*), and division (/)
- **BETWEEN** comparison operator

Ordering of Query Results

- Use **ORDER BY** clause
 - Keyword **DESC** to see result in a descending order of values
 - Keyword **ASC** to specify ascending order explicitly
 - `ORDER BY D.Dname DESC, E.Lname ASC, E.Fname ASC`

Discussion and Summary of Basic SQL Retrieval Queries

```
SELECT    <attribute list>  
FROM      <table list>  
[ WHERE   <condition> ]  
[ ORDER BY <attribute list> ];
```

INSERT, DELETE, and UPDATE Statements in SQL

- Three commands used to modify the database:
 - INSERT, DELETE, and UPDATE

The INSERT Command

- Specify the relation name and a list of values for the tuple

```
U1:  INSERT INTO  EMPLOYEE
      VALUES      ( 'Richard', 'K', 'Marini', '653298653', '1962-12-30', '98
                    Oak Forest, Katy, TX', 'M', 37000, '653298653', 4 );
```

```
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

- Removes tuples from a relation
 - Includes a `WHERE` clause to select the tuples to be deleted

```
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

- Modify attribute values of one or more selected tuples
- Additional **SET** clause in the `UPDATE` command
 - Specifies attributes to be modified and new values

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

Additional Features of SQL

- Techniques for specifying complex retrieval queries
- Writing programs in various programming languages that include SQL statements
- Set of commands for specifying physical database design parameters, file structures for relations, and access paths
- Transaction control commands

Additional Features of SQL (cont'd.)

- Specifying the granting and revoking of privileges to users
- Constructs for creating triggers
- Enhanced relational systems known as object-relational
- New technologies such as XML and OLAP

Summary

- SQL
 - Comprehensive language
 - Data definition, queries, updates, constraint specification, and view definition
- Covered in Chapter 4:
 - Data definition commands for creating tables
 - Commands for constraint specification
 - Simple retrieval queries
 - Database update commands