204320 - Database Management

Chapter 4

Basic SQL

Adapted for 204320

by Areerat Trongratsameethong

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 (SQL)**
  - Statements for data definitions, queries, and updates (both DDL and DML)
  - **Core specification:** SQL ประกอบด้วยส่วนที่เป็นคำสั่ง หรือฟังก์ชันหลักที่ทุก RDBMS จะต้องมี
  - Plus specialized **extensions:** ส่วนที่เพิ่มเติมจาก core specification แล้วแต่ผู้ผลิต RDBMS ที่จะเพิ่มเติมเพื่อเป็นจุดขาย

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** (การกำหนดสิทธิในการเข้าถึงข้อมูล) and **descriptors** for each element (การกำหนดคำอธิบายของแต่ละ 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';` → `Jsmith` มีสิทธิในการเข้าถึง `Company Schema`
- **Catalog**
  - Named collection of schemas in an SQL environment: ใน 1 environment สามารถมี Database Schema ได้หลายตัว ซึ่งถูกจัดเก็บไว้รวมกันที่ System Catalog
- SQL **environment**
  - Installation of an SQL-compliant RDBMS on a computer system

The CREATE TABLE Command in SQL

- Specify a new relation: การสร้าง Relational Schema
  - Provide name: กำหนดชื่อ
  - Specify attributes and initial constraints: กำหนด attribute ต่างๆ และข้อบังคับเริ่มต้น
- Can optionally specify schema:
  - `CREATE TABLE COMPANY.EMPLOYEE ...`
    or
  - `CREATE TABLE EMPLOYEE ...`
- **Base tables** (**base relations**)
  - Relation and its tuples are actually created and stored as a file by the DBMS
- **Virtual relations:** ไม่ได้จัดเก็บข้อมูลลงในฐานข้อมูล แต่ได้ข้อมูลมาจากการสืบค้นข้อมูลจาก base relation ต่างๆ รวมกัน
  - 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:** อ้างอิง attribute ตัวเดียวกัน เช่น emp_id กับ emp_id
    - **Or because they refer to a table that has not yet been created:** อ้างอิง attribute ในตารางที่ยังไม่ได้สร้าง

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*`)` `e.g. '10101'`
- **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 e.g. '2014-01-24'

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 e.g. '2014-01-24 12:30:47.648302'
    - 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

Example

```
X:BEGIN
   SELECT price FROM stocks WHERE name="Acme"
          COMPLETE BEFORE   CURRENT_TIMESTAMP + INTERVAL '30' SECOND;
   -- other computations
END X COMPLETE BEFORE CURRENT_TIMESTAMP + INTERVAL '1' MINUTE;
```

- The execution timing constraint on the SELECT statement species that it must complete execution within 30 seconds. เช่น ถ้าตอนนี้เวลา 19:10:15 น. ต้องเสร็จก่อนเวลา 19:10:45 น.
- The timing constraint on the compound statement species that it must complete execution within 1 minute. เช่น ถ้าตอนนี้เวลา 19:10:15 น. ตั้งแต่ต้นจนจบต้องเสร็จก่อนเวลา 19:11:45 น.

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 e.g **`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 (หากการ update FK ขัดแย้งกับกฎข้อบังคับ default คือ Reject) ทั้งนี้เราสามารถกำหนด referential trigger action เมื่อเกิดความขัดแย้งเกิดขึ้นได้ ดังแสดงด้านล่าง
  - 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`**: การกำหนด **`CONSTRAINT`**
  - Name a constraint: กำหนดชื่อ Constraint
  - Useful for later altering: เป็นประโยชน์เมื่อมีการใช้หลายที่ และหากมีการแก้ไขในอนาคต แก้ที่เดียว
- 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
  - ข้อมูลผลลัพธ์ในรูปของ record ที่ได้จากการใช้คำสั่ง SQL สามารถมี record ที่มีข้อมูลที่มีค่าเหมือนกันหลาย record ได้ ซึ่งแตกต่างจาก relational data model ที่ไม่อนุญาตให้มี tuple หลายๆ tuple ที่มีค่าข้อมูลเหมือนกันทุก attribute

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: attribute ที่ปรากฏในผลลัพธ์ (ที่ระบุไว้ต่อจาก SELECT)
- **Selection condition**
  - Boolean condition that must be true for any retrieved tuple: tuple ที่มีเงื่อนไขถูกต้องตามที่ระบุต่อจาก WHERE เท่านั้นที่จะปรากฏในผลลัพธ์

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

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

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)

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

(b)

Fname	Lname	Address
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)

Pnumber	Dnum	Lname	Address	Bdate
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';
```

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

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

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

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 (คลุมเครือ, กำกวม)
  - ในกรณีที่มีชื่อ attribute เหมือนกันอยู่หลาย table และมีการอ้างอิง attribute ที่มีชื่อเหมือนกันแต่อยู่ต่าง table อาจทำให้เกิดความสับสนได้ ดังนั้นให้ระบุชื่อ table ตามด้วย . และตามด้วยชื่อ attribute

```
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:** การใช้ alias name สามารถใช้แทนชื่อ attribute หรือ ชื่อ table ได้
  - Declare alternative relation names E and S
  - `EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd, Addr, Sex, Sal, Sssn, Dno)`

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

equivalent

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


Unspecified WHERE Clause and Use of the Asterisk

- **Missing `WHERE` clause**
  - **Indicates no condition on tuple selection**
- **`CROSS PRODUCT`**
  - **All possible tuple combinations**
  - การที่ไม่ระบุ **WHERE** จะเกี่ยวข้องกับทุก **record**

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;
```

←Ssn ของทุก record ใน EMPLOYEE จะมาหมด

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

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

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

Q10: ทุก record ของ EMPLOYEE รวมกับ DEPARTMENT จะมาหมด
เช่น หาก EMPLOYEE มี 10 record และ DEPARTMENT มี 5 Record
ผลลัพธ์คือ 10 x 5 = 50 Record
โดยที่แต่ละ record ประกอบด้วย attribute ของ EMPLOYEE ต่อด้วย attribute ของ DEPARTMENT

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

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

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
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

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

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

→ ผลลัพธ์คือ ?

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

→ ผลลัพธ์คือ ?

Tables as Sets in SQL

- <u>SQL does not automatically eliminate duplicate tuples in query results</u>
- Use the keyword **DISTINCT** in the SELECT clause: หากต้องการให้ผลลัพธ์ที่ได้จากคำสั่ง SQL ไม่เกิด record ซ้ำ ให้ใช้ keyword DISTINCT ต่อจาก SELECT
  - Only distinct tuples should remain in the result: record ที่มีค่าข้อมูลซ้ำกันหลายๆ record จะเหลือเพียงแค่ 1 record

Salary
30000
40000
25000
43000
38000
25000
25000
55000

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;
```

Salary
30000
40000
25000
43000
38000
55000

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                                    ⇒ ผลลัพธ์คือ ?
       ( 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

Query 12. Retrieve all employees whose address is in Houston, Texas.

```
Q12:   SELECT    Fname, Lname
       FROM      EMPLOYEE
       WHERE     Address LIKE '%Houston,TX%';
```

Query 12A. Find all employees who were born during the 1950s.

```
Q12:   SELECT    Fname, Lname
       FROM      EMPLOYEE
       WHERE     Bdate LIKE '_ _ 5 _ _ _ _ _ _ _';
```

Query 13. Show the resulting salaries if every employee working on the 'ProductX' project is given a 10 percent raise.

```
Q13:   SELECT    E.Fname, E.Lname, 1.1 * E.Salary AS Increased_sal
       FROM      EMPLOYEE AS E, WORKS_ON AS W, PROJECT AS P
       WHERE     E.Ssn=W.Essn AND W.Pno=P.Pnumber AND
                 P.Pname='ProductX';
```

Query 14. Retrieve all employees in department 5 whose salary is between $30,000 and $40,000.

```
Q14:   SELECT    *
       FROM      EMPLOYEE
       WHERE     (Salary BETWEEN 30000 AND 40000) AND Dno = 5;
```

The condition (Salary **BETWEEN** 30000 **AND** 40000) in Q14 is equivalent to the condition ((Salary >= 30000) **AND** (Salary <= 40000)).

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

Query 15. Retrieve a list of employees and the projects they are working on, ordered by department and, within each department, ordered alphabetically by last name, then first name.

```
Q15:   SELECT    D.Dname, E.Lname, E.Fname, P.Pname
       FROM      DEPARTMENT D, EMPLOYEE E, WORKS_ON W,
                 PROJECT P
       WHERE     D.Dnumber= E.Dno AND E.Ssn= W.Essn AND
                 W.Pno= P.Pnumber
       ORDER BY  D.Dname, E.Lname, E.Fname;
```

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 (ให้สิทธิ) and revoking (ยกเลิกสิทธิ) of privileges to users
- Constructs for creating triggers
- Enhanced relational systems known as object-relational
- New technologies such as XML and OLAP

http://olap.com/olap-definition/

OLAP (Online Analytical Processing) is the technology behind many Business Intelligence (BI) applications. OLAP is a powerful technology for data discovery, including capabilities for limitless report viewing, complex analytical calculations, and predictive "what if" scenario (budget, forecast) planning.

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