Chapter 4
Basic SQL

Universiteit Maastricht

Chapter 4 Outline

- SQL Data Types
- Specifying Constraints in SQL
- Basic Retrieval Queries in SQL
- `INSERT`, `DELETE`, and `UPDATE` Statements in SQL

Basic SQL

- **SQL language**
  - Considered one of the major reasons for the commercial success of relational databases
  - Different DBMSs can support it, making conversion an easy matter
- **SQL**
  - **Structured Query Language**
  - Statements for data definitions, queries, and updates (both **DDL** and **DML**), views, security, authorization, constraints definition, transaction controls
  - Can be embedded in general-purpose programming languages

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
    - For schemas, tables and domains

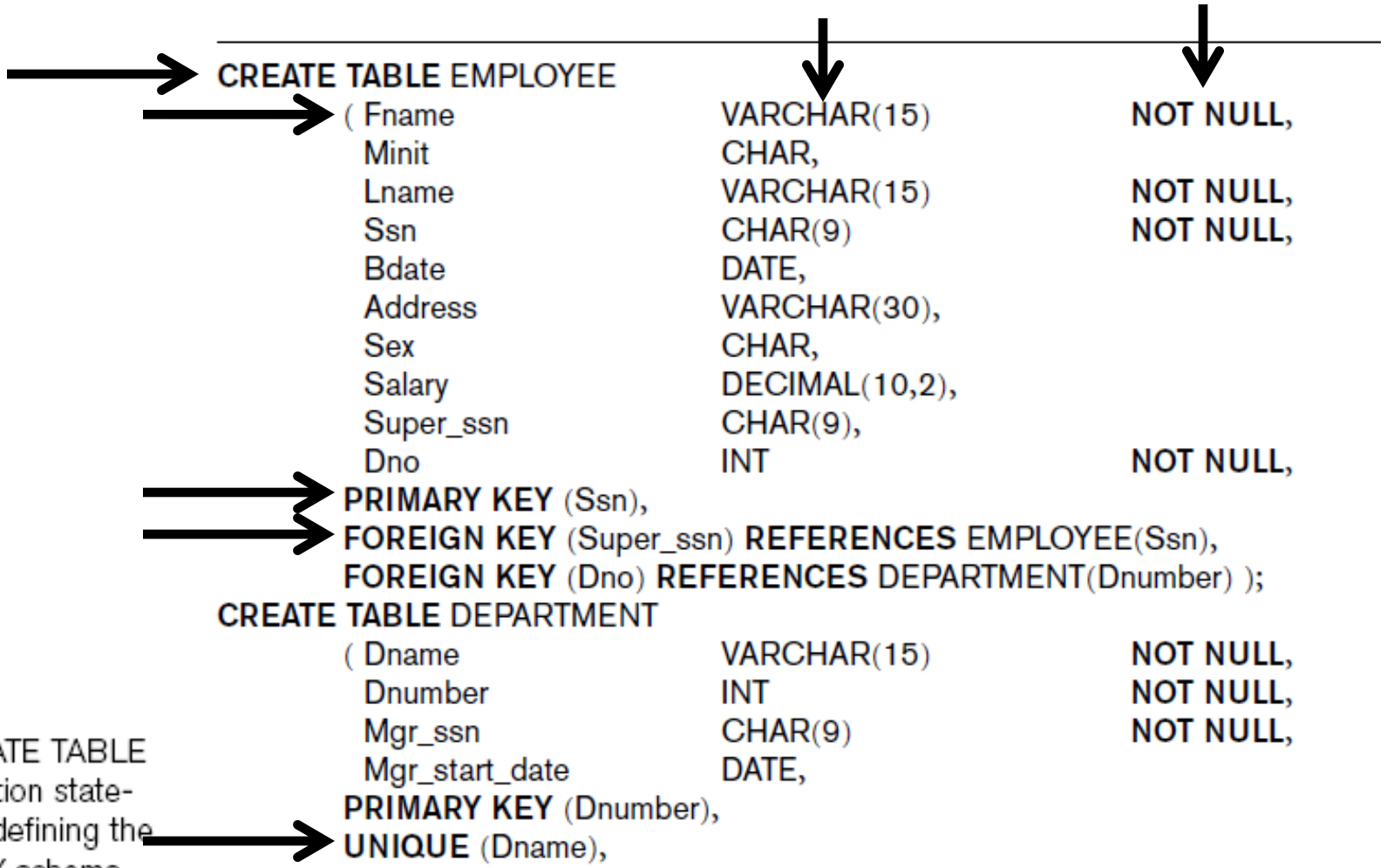
Schema and Catalog Concepts in SQL

- `CREATE Database COMPANY;`
- `CREATE SCHEMA COMPANY;`
- What is the difference between Schema and Database??

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 ...`



```
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

- Could you guess what kind of problems you may face when creating foreign keys?
  - Some foreign keys may cause errors because they refer to a table that has not yet been created. (Employee and Department)
  - To deal with this type of problem, these constraints can be left out of the initial CREATE TABLE statement, and then added later using the ALTER TABLE statement

Attribute Data Types and Domains in SQL

- Basic **data types** (https://www.postgresql.org/docs/9.3/static/datatype.html)
  - **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

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

Attribute Data Types and Domains in SQL

- Additional data types
  - **`TIMESTAMP`** data type
    - Includes the `DATE` and `TIME` fields
    - Plus positions for decimal fractions of seconds
    - Optional `WITH TIME ZONE` qualifier

Attribute Data Types and Domains in SQL

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

Giving Names to Constraints

- Keyword **`CONSTRAINT`**
  - Name a constraint
    - Giving names to constraints is optional
    - Useful for later altering (e.g. dropping and replacing)

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 DOMAIN D_NUM AS INTEGER
    CHECK (D_NUM>0 AND D_NUM<21)

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;`
  - Or, after attribute definition:

    `PRIMARY KEY(Dnumber)`
- **`UNIQUE`** clause
  - Specifies alternate (secondary) keys
  - `Dname VARCHAR(15) UNIQUE;`
  - Or, after attribute definition:

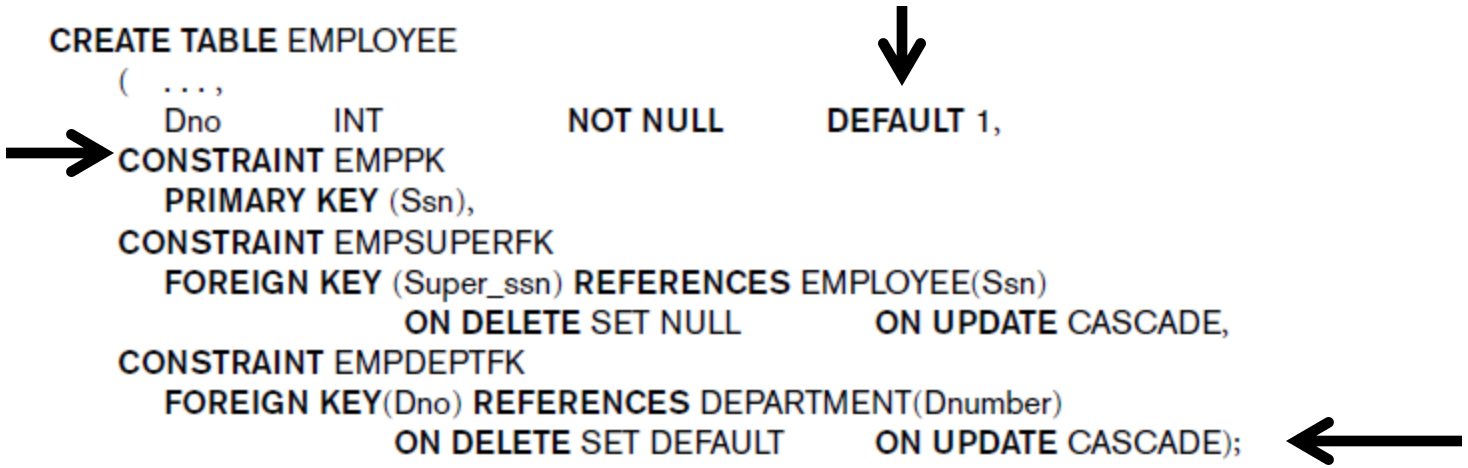
    `UNIQUE (Dname)`

Specifying Key and Referential Integrity Constraints

- **`FOREIGN KEY`** clause
  - Default operation: reject update on violation
  - Attach **referential triggered action** clause
    - Ex: FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber) ON DELETE SET DEFAULT ON UPDATE CASCADE);
      - Options include `SET NULL`, `CASCADE`, and `SET DEFAULT`

Specifying Constraints on Tuples Using CHECK

- `CHECK` clauses at the end of a `CREATE TABLE` statement
  - Apply to each tuple individually when entering or modifying tuple attributes
  - `CHECK (Dept_create_date <= Mgr_start_date);`



```
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.

Basic Retrieval Queries in SQL

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

The SELECT-FROM-WHERE Structure of Basic SQL Queries

- `SELECT` statement
  - One basic statement for retrieving information from a database
- 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

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

Figure 3.6
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

Figure 3.6
One possible database state for the COMPANY relational database schema.

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

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
      FROM
      WHERE
```

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

```
Q1:   SELECT
      FROM
      WHERE
```

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
      FROM
      WHERE
```

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

Query 8. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.

```
Q8:   SELECT   E.Fname, E.Lname, S.Fname, S.Lname
      FROM     EMPLOYEE AS E, EMPLOYEE AS S
      WHERE    E.Super_ssn=S.Ssn;
```

  - Declare alternative relation & attribute names

```
EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd, Addr, Sex, Sal, Sssn, Dno)
```

Unspecified WHERE Clause

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

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

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

Use of the Asterisk

- 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
  - Duplicate elimination is an expensive operation. One way to implement it is to sort the tuples first and then eliminate duplicates.
  - The user may want to see duplicate tuples in the result of a query.
  - When an aggregate function is applied to tuples, in most cases we do not want to eliminate duplicates.

Tables as Sets in SQL

- 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;
```

(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`**
    - Resulting relations are sets of tuples (duplicates are eliminated)
    - Apply only to union-compatible relations

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:

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

Substring Pattern Matching and Arithmetic Operators

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

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

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

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

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 DESC, E.Lname ASC, E.Fname ASC
```

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 );
```

- Attributes with NULL allowed or specified DEFAULT value can be left out

```
U1A:  INSERT INTO   EMPLOYEE (Fname, Lname, Dno, Ssn)
      VALUES        ('Richard', 'Marini', 4, '653298653');
```

The INSERT Command

- Check for integrity constraints. Why are these INSERT operations rejected?

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

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

INSERT data from other table

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

```
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

- 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
- It is also possible to specify NULL or DEFAULT as the new attribute value.
- Notice that each UPDATE command explicitly refers to a single relation only. To modify multiple relations, we must issue several UPDATE commands.

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

U6:
```
UPDATE   EMPLOYEE
SET      Salary = Salary * 1.1
WHERE    Dno = 5;
```

Next Time!

Additional Features of SQL

- Techniques for specifying complex retrieval queries (nested queries, aggregate functions, grouping, joined tables, outer joins, and recursive queries; SQL views, triggers, and assertions; and commands for schema modification.)