COSC 3380
Design of Database Systems

Basic Structured Query Language (SQL)

March 18, 2024

Basic SQL Retrieval Query Block

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

INSERT, DELETE, and UPDATE Statements

- Three commands used to modify the database:
  - **`INSERT`** typically inserts a tuple (row) in a relation (table)
  - **`UPDATE`** may update a number of tuples (rows) in a relation (table) that satisfy the condition
  - **`DELETE`** may also update a number of tuples (rows) in a relation (table) that satisfy the condition

INSERT Command

- In its simplest form, it is used to add one or more tuples to a relation
- Attribute values should be listed in the same order as the attributes were specified in the `CREATE TABLE` command
- Constraints on data types are observed automatically
- Any integrity constraints as a part of the DDL specification are enforced

INSERT Command

- Specify the relation name and a list of values for the tuple. All values including nulls are supplied.

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

- The variation below inserts multiple tuples where a new table is loaded values from the result of a query.

```
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;
```

INSERT Command

- Attributes with NULL allowed or DEFAULT values can be left out.

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

- Reject insert operation if constraints are violated

```
INSERT INTO    EMPLOYEE (Fname, Lname, Ssn, Dno)
VALUES         ('Robert', 'Hatcher', '980760540', 2);
```

```
INSERT INTO    EMPLOYEE (Fname, Lname, Dno)
VALUES         ('Robert', 'Hatcher', 5);
```

Bulk loading of Tables

- Another variation of **INSERT** is used for bulk-loading of several tuples into tables
- A new table TNEW can be created with the same attributes as T and using LIKE and DATA in the syntax, it can be loaded with entire data.

```
CREATE TABLE D5EMPS  LIKE  EMPLOYEE
  (SELECT    E.*
    FROM    EMPLOYEE AS E
    WHERE       E.Dno=5)
  WITH DATA;
```

DELETE Command

- Removes tuples from a relation
  - Includes a WHERE-clause to select the tuples to be deleted
  - The number of tuples deleted depends on the number of tuples in the relation that satisfy the WHERE-clause
  - A missing WHERE-clause specifies that *all tuples* in the relation are to be deleted; the table then becomes an empty table
  - Referential integrity should be enforced
  - Tuples are deleted from only *one table* at a time (unless CASCADE is specified on a referential integrity constraint)

DELETE Command

```
DELETE FROM     EMPLOYEE
WHERE           Lname='Brown';
```

```
DELETE FROM     EMPLOYEE
WHERE           Ssn='123456789';
```

```
DELETE FROM     EMPLOYEE
WHERE           Dno=5;
```

```
DELETE FROM     EMPLOYEE;
```

UPDATE Command

- Used to modify attribute values of one or more selected tuples
- WHERE-clause selects the tuples to be modified
- An additional SET-clause specifies the attributes to be modified and their new values
- Each command modifies tuples *in the same relation*
- Referential integrity specified as part of Schema specification is enforced

UPDATE Command

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

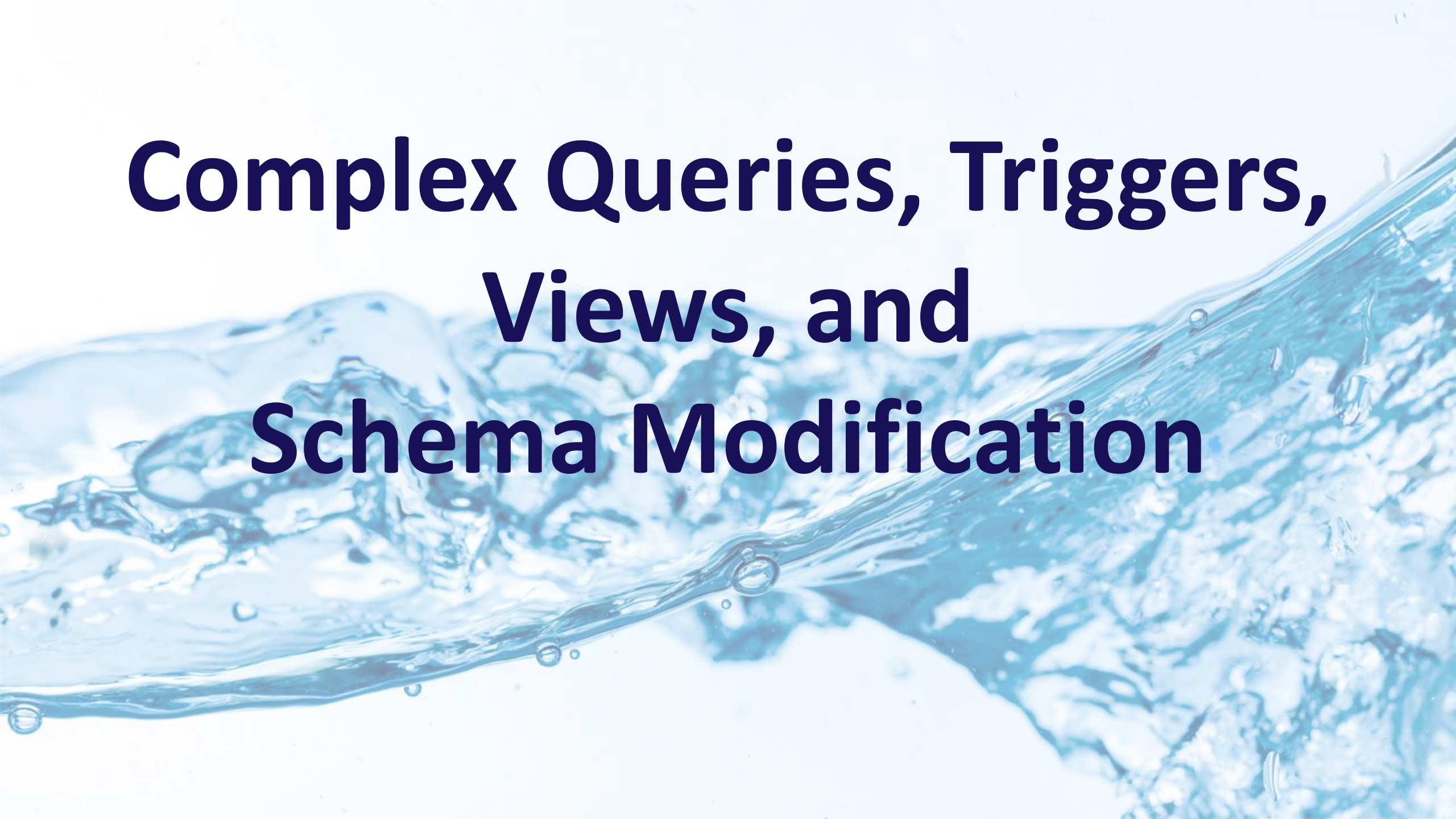
```
UPDATE      EMPLOYEE
SET         Salary = Salary * 1.1
WHERE       Dno = 5;
```

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, e.g., CREATE INDEX

Additional Features of SQL

- Transaction control commands – for concurrency control and recovery processes
- Specifying the granting and revoking of privileges to users (`GRANT` and `REVOKE`)
- Constructs for creating triggers – active database techniques
- SQL can interact with other technologies such as XML and OLAP



Complex Queries, Triggers, Views, and Schema Modification

Specifying Constraints as Assertions, Actions as Triggers

- Semantic Constraints
- **CREATE ASSERTION**
  - Specify additional types of constraints outside scope of built-in relational model constraints
- **CREATE TRIGGER**
  - Specify automatic actions that database system will perform when certain events and conditions occur

Specifying General Constraints as Assertions in SQL

- **`CREATE ASSERTION`**
  - Specify a query that selects any tuples that violate the desired condition
  - Use only in cases where it goes beyond a simple `CHECK` which applies to individual attributes and domains

```
CREATE ASSERTION SALARY_CONSTRAINT
CHECK ( NOT EXISTS  ( SELECT     *
                      FROM       EMPLOYEE E, EMPLOYEE M,
                                 DEPARTMENT D
                      WHERE      E.Salary>M.Salary
                                 AND E.Dno=D.Dnumber
                                 AND D.Mgr_ssn=M.Ssn ) );
```

Triggers in SQL

- **CREATE TRIGGER** statement
  - Used to monitor the database
- Typical trigger has three components which make it a rule for an “active database”:
  - **Event(s)**
  - **Condition**
  - **Action**

USE OF TRIGGERS

CREATE TRIGGER SALARY_VIOLATION
BEFORE INSERT OR UPDATE OF Salary, Supervisor_ssn **ON** EMPLOYEE

Event

FOR EACH ROW
WHEN (NEW.SALARY > **(** SELECT Salary FROM EMPLOYEE
WHERE Ssn = NEW. Supervisor_Ssn))
INFORM_SUPERVISOR (NEW.Supervisor.Ssn, New.Ssn)

Condition

Action

Comparisons Involving NULL and Three-Valued Logic

- Meanings of NULL
  - Unknown value
  - Unavailable or withheld value
  - Not applicable attribute
- Each individual NULL value considered to be different from every other NULL value
- SQL uses a three-valued logic:
  - TRUE, FALSE, and UNKNOWN (like Maybe)
- **NULL = NULL comparison is avoided**

Comparisons Involving NULL and Three-Valued Logic

(a)

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

(b)

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	UNKNOWN

(c)

NOT	
TRUE	FALSE
FALSE	TRUE
UNKNOWN	UNKNOWN

Comparisons Involving NULL and Three-Valued Logic

- SQL allows queries that check whether an attribute value is **NULL**
  - **IS NULL** or **IS NOT NULL**

```
SELECT    Fname, Lname
FROM      EMPLOYEE
WHERE     Super_ssn IS NULL;
```

Nested Queries, Tuples, Set/Multiset Comparisons

- **Nested queries**
  - Complete SELECT-FROM-WHERE blocks within WHERE clause of another query
  - Outer query and nested subqueries

- **Comparison operator IN**
  - Compares value *v* with a set (or multiset) of values *V*
  - Evaluates to TRUE if *v* is one of the elements in *V*

Nested Queries

```
SELECT    DISTINCT Pnumber
FROM      PROJECT
WHERE     Pnumber IN
          ( SELECT    Pnumber
            FROM      PROJECT, DEPARTMENT, EMPLOYEE
            WHERE     Dnum=Dnumber AND
                      Mgr_ssn=Ssn AND Lname='Smith' )
          OR
          Pnumber IN
          ( SELECT    Pno
            FROM      WORKS_ON, EMPLOYEE
            WHERE     Essn=Ssn AND Lname='Smith' );
```

Select project numbers of projects that have an employee with last name 'Smith' involved as manager.

Select project numbers of projects that have an employee with last name 'Smith' involved as a worker.