CS 440: Database Management Systems

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

Review of SQL (Structured Query Language)

- Example:
  
  ```sql
  CREATE SCHEMA company AUTHORIZATION jsmith;
  CREATE TABLE company.employee ...
  CREATE TABLE employee ...
  ```
Review of SQL (Structured Query Language)

CREATE TABLE employee
   ( FName VARCHAR(15) NOT NULL,
     MInit CHAR,
     LName VARCHAR(15) NOT NULL,
     SSN CHAR(9) NOT NULL,
     BDateDate,
     Address VARCHAR(30),
     Gender CHAR,
     Salary DECIMAL(10,2),
     SuperSSN VARCHAR(9),
     DNo INT NOT NULL,
   PRIMARY KEY (SSN),
   FOREIGN KEY (SuperSSN)
            REFERENCES employee(SSN),
   FOREIGN KEY (DNo) REFERENCES department(DNumber) );

Review of SQL (Structured Query Language)

- Attributes Data Types and Domains
  - 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);

Review of SQL (Structured Query Language)

- Basic constraints:
  - Key constraints
  - Referential integrity constraints
  - Domain constraints
  - NULL
  - Constraints on tuples within a relation
## Review of SQL (Structured Query Language)

### Key, referential integrity, and participation constraints:

```sql
CREATE TABLE employee

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>MInit</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>LName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>SSN</td>
<td>CHAR(9)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>BDate</td>
<td>DATE</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>VARCHAR(30)</td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>Salary</td>
<td>DECIMAL(10.2)</td>
<td></td>
</tr>
<tr>
<td>SuperSSN</td>
<td>VARCHAR(9)</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
<tr>
<td>DNo</td>
<td>INT</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
</tbody>
</table>

CONSTRAINT EMPPK PRIMARY KEY (SSN)

CONSTRAINT EMPSUPERFK FOREIGN KEY (SuperSSN) 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
```

### Attribute/Domain constraints:

```sql
CREATE DOMAIN Dnum_Type AS INTEGER NOT NULL CHECK (DNumber > 0 AND DNumber < 21);

CREATE TABLE department

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>Dnum</td>
<td>INT</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>Dnum</td>
<td>INT</td>
<td>NOT NULL</td>
</tr>
</tbody>
</table>

CONSTRAINT DEPTPK PRIMARY KEY (DName)

CREATE TABLE employee

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>MInit</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>LName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>SSN</td>
<td>CHAR(9)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>BDate</td>
<td>DATE</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>VARCHAR(30)</td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>Salary</td>
<td>DECIMAL(10.2)</td>
<td></td>
</tr>
<tr>
<td>SuperSSN</td>
<td>VARCHAR(9)</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
<tr>
<td>DNo</td>
<td>INT</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
</tbody>
</table>

CONSTRAINT EMPPK PRIMARY KEY (SSN)

FOREIGN KEY (SuperSSN) REFERENCES employee(SSN) ON DELETE SET NULL

FOREIGN KEY (DNo) REFERENCES department(DNumber) ON DELETE SET DEFAULT ON UPDATE CASCADE
```

```sql
CREATE TABLE department

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>Dnum</td>
<td>INT</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>Dnum</td>
<td>INT</td>
<td>NOT NULL</td>
</tr>
</tbody>
</table>

CONSTRAINT DEPTPK PRIMARY KEY (DName)

CREATE TABLE employee

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>MInit</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>LName</td>
<td>VARCHAR(15)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>SSN</td>
<td>CHAR(9)</td>
<td>NOT NULL</td>
</tr>
<tr>
<td>BDate</td>
<td>DATE</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>VARCHAR(30)</td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>CHAR</td>
<td></td>
</tr>
<tr>
<td>Salary</td>
<td>DECIMAL(10.2)</td>
<td></td>
</tr>
<tr>
<td>SuperSSN</td>
<td>VARCHAR(9)</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
<tr>
<td>DNo</td>
<td>INT</td>
<td>NOT NULL DEFAULT 1</td>
</tr>
</tbody>
</table>

CONSTRAINT EMPPK PRIMARY KEY (SSN)

FOREIGN KEY (SuperSSN) REFERENCES employee(SSN) ON DELETE SET NULL

FOREIGN KEY (DNo) REFERENCES department(DNumber) ON DELETE SET DEFAULT ON UPDATE CASCADE

```

### Attribute/Domain constraints:

```sql
CREATE DOMAIN DNumber INT NOT NULL CHECK (DNumber > 0 AND DNumber < 21);

CREATE DOMAIN Drum_Type AS INTEGER NOT NULL CHECK (DNumber > 0 AND DNumber < 21);
```

---

2/2/2012
Review of SQL (Structured Query Language

- Attribute default:

```sql
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),
    Gender CHAR,
    Salary DECIMAL(10,2),
    SuperSSN VARCHAR(9),
    DNo INT NOT NULL DEFAULT 1,
    PRIMARY KEY (SSN),
    FOREIGN KEY (SuperSSN) REFERENCES employee(SSN),
    FOREIGN KEY (DNo) REFERENCES department(DNumber)
);```

- Tuple constraints:

- `CHECK` clauses at the end of a `CREATE TABLE` statement
  - Apply to each tuple individually

```sql
CHECK (Dept_create_date <= Mgr_start_date);
```

- Some foreign keys may cause errors
  - Specified either via:
    - Circular references
    - Or because they refer to a table that has not yet been created
Review of SQL (Structured Query Language)

- **DROP**
  - Table
  - View
  - Constraints
  - Triggers
  - Assertions
  - Stored procedures
  
  ```
  DROP TABLE Dependent;
  ```

Review of SQL (Structured Query Language)

- **ALTER**
  
  ```
  ALTER TABLE Dependent ALTER MGRSSN DROP DEFAULT;
  ```

Review of SQL (Structured Query Language)

- **DML:**
  
  - Read operations:
    - Select (see what are currently there)
  
  - Write operations:
    - Insert (add new records)
    - Delete (remove records that are currently in the database)
    - Update (change certain attributes of records that are currently in the database)
Review of SQL (Structured Query Language)

```
SELECT [DISTINCT | AS] [AS name, ...]
FROM table_name
WHERE [expression]...;
```

```
INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
[INTO] tbl_name [({col_name}, ...)]
VALUES [({expression | DEFAULT}, ...), ...]...
OR INSERT LOW_PRIORITY [LOW_PRIORITY, ...]
VALUES [({expression | DEFAULT}, ...), ...]...
```

```
DELETE [LOW_PRIORITY] [QUICK] FROM table_name
WHERE [expression]...
```

```
FROM {table_name | table_name.* | table_name[+] | table_name[-]}...
USING table_name...
```

Review of SQL (Structured Query Language)

UPDATE [LOW_PRIORITY] [IGNORE] tbl_name
SET col_name=expr1 [, col_name=expr2 ...]
WHERE where_definition
[ORDER BY ...]
[LIMIT row]

or

UPDATE [LOW_PRIORITY] [IGNORE] tbl_name [, tbl_name ...]
SET col_name=expr1 [, col_name=expr2 ...]
WHERE where_definition

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
Example Retrieval Queries...

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

\[ \pi_{\text{Birth, Address}}(\sigma_{\text{Name} = \text{"John B. Smith"}} \text{Employee}) \]

Q0: SELECT Birth, Address
    FROM EMPLOYEE
    WHERE Name = 'John' AND Initial = 'B' AND Lname = 'Smith';

Example Retrieval Queries...

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

\[ \pi_{\text{Name, Address}}(\sigma_{\text{Department} = \text{"Research"}} \text{Employee}) \]
\[ \text{OR} \]
\[ \pi_{\text{Name, Address}}(\sigma_{\text{Department} = \text{"Research"}} \text{Employee}) \]

Q1: SELECT Name, Address
    FROM EMPLOYEE, DEPARTMENT
    WHERE Department = 'Research';

Example Retrieval Queries...

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:

\[ \pi_{\text{Project, Dnum, Lname, Address, Birth}}(\sigma_{\text{Location = \text{"Stafford"}}} \text{Project, DEPARTMENT, EMPLOYEE}) \]

Q2: SELECT Project, Dnum, Lname, Address, Birth
    FROM PROJECT, DEPARTMENT, EMPLOYEE
    WHERE Dnum = Dnumber AND Mgr. assn = 'Yes' AND
    Location = '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

Example Retrieval Queries...

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

```
SEmpl ← P(\text{Employee}, \text{Employee}) \ (\pi_{\text{Fname, Lname, Fname, Lname}} (\text{Employee}))
\pi_{\text{Fname, Lname, Fname, Lname}} (\sigma_{\text{Super_ssn} = \text{Ssn}} (\text{Employee} \bowtie \text{SEmpl}))
```

```
Select E.Fname, E.Lname, S.Fname, S.Lname
From Employee As E, Employee As S
Where E.Super_ssn = S.Ssn;
```

Example Retrieval Queries...

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.

```
\pi_{\text{Pnumber}} (\sigma_{\text{Worker_ssn} = \text{Ssn} \text{ AND Ssn = Super_ssn}} (\text{ProjectWorks} \bowtie \text{Employee}))
\pi_{\text{Pnumber}} (\sigma_{\text{Manager_ssn} = \text{Ssn}} (\text{ProjectDepartment} \bowtie \text{Employee}))
```

```
QTA: SELECT DISTINCT Pnumber
FROM PROJECT, DEPARTMENT, EMPLOYEE
WHERE Dnum = Dnum AND Mgr_ssn = Ssn
AND Lname = 'Smith'
UNION
QTA: SELECT DISTINCT Pnumber
FROM PROJECT, WORKS, ON, EMPLOYEE
WHERE Pnumber = Pnumber AND Ename = Ename
AND Lname = 'Smith';
```
Pattern Matching

- Retrieve all employees whose address is in Houston, TX.
  
  \[
  \text{Select } \text{Fname, Lname} \\
  \text{From } \text{Employee} \\
  \text{Where Address Like } '%Houston,TX%';
  \]

- Find all employees who were born during the 1950s.
  
  \[
  \text{Select } \text{Fname, Lname} \\
  \text{From } \text{Employee} \\
  \text{Where Bdate Like } '_9_ _ _ _ _ _ _';
  \]

Ordering of Query Results

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

INSERT, DELETE, and UPDATE Statements in SQL

- Three commands used to modify the database:
  - \text{INSERT, DELETE, and UPDATE}
The INSERT Command

- Specify the relation name and a list of values

```
U1: INSERT INTO EMPLOYEE
VALUES
('Richard', 'K', 'Martin', '65359653', '7143-12-35', '98
Oak Forest, Katy, TX, M, 27000, 102298652', '31');

USB: INSERT INTO WORKS_ON_INFO
SELECT E.Name, P.Name, W.Hours
FROM PROJECT P, WORKS_ON W, EMPLOYEE E
WHERE P.Number = W.Project AND W.EmpNum = E.EmpNum;
```

The DELETE Command

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

```
U4: DELETE FROM EMPLOYEE
WHERE Name = 'Brown';

U5: DELETE FROM WORKS_ON
WHERE EmpNum = 123456789;

U6: DELETE FROM EMPLOYEE
WHERE Date = 5;

U7: 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

```
U8: UPDATE PROJECT
SET Plocation = 'Bellaire', Dnum = 5
WHERE Pnumber = 10;
```