Goals

- SQL: Data Definition Language
  - CREATE
  - ALTER
  - DROP
- SQL: Data Manipulation Language
  - INSERT
  - DELETE
  - UPDATE
  - SELECT

Relational Query Languages

- A major strength of the relational model:
  - supports simple, powerful querying of data
  - Queries can be written intuitively, and the DBMS is responsible for efficient evaluation.

SQL DDL and DML

- SQL statements can be divided into two categories:
  - Data definition language (DDL) statements
    - Used for creating and modifying tables, views, and other structures
    - CREATE, DROP, ALTER
  - Data manipulation language (DML) statements.
    - Used for queries and data modification
    - INSERT, DELETE, UPDATE, SELECT

Creating Tables

CREATE TABLE table_name(
  column_name1 column_type1 [constraints1],
  ...
  [CONSTRAINT constraint_name] table_constraint
)

Table constraints:
- NULL/NOT NULL
- PRIMARY KEY (columns)
- UNIQUE (columns)
- CHECK (conditions)
- FOREIGN KEY (local_columns) REFERENCES foreign_table (foreign_columns) [ON DELETE action_d ON UPDATE action_u]

Specify surrogate key in SQL Server:
  column_name int_type IDENTITY (seed, increment)
### Modifying Tables

- **ALTER TABLE** table_name clause

  Clauses:
  - `ADD COLUMN` column_name column_type [constraints]
  - `DROP COLUMN` column_name
  - `ALTER COLUMN / MODIFY` – DBMS specific!
  - `ADD CONSTRAINT` constraint
  - `DROP CONSTRAINT` constraint_name

### Removing Tables

- **DROP TABLE** table_name

  DROP TABLE Departments;

  If there are constraints dependent on table:
  - Remove constraints
  - Drop table

  ALTER TABLE Students
  DROP CONSTRAINT FK_Department;

  DROP TABLE Departments;

### SQL DDL and DML

- **Data definition language (DDL)** statements
  - Used for creating and modifying tables, views, and other structures
  - `CREATE`, `ALTER`, `DROP`

- **Data manipulation language (DML)** statements
  - Used for queries and data modification
  - `INSERT`, `DELETE`, `UPDATE`, `SELECT`
SQL DML

- Data manipulation language (DML) statements.
  - Used for queries and data modification
  - INSERT
  - DELETE
  - UPDATE
  - SELECT

INSERT Statement

```
INSERT INTO table_name [(column_list)] VALUES (data_values)
```

```
INSERT INTO table_name [(column_list)] select_statement
```

**INSERT command:**
```
INSERT INTO Students (StudentNumber, StudentLastName, StudentFirstName)
VALUES (190, 'Smith', 'John');
```

**Bulk INSERT:**
```
INSERT INTO Students VALUES(190, 'Smith', 'John', 'jsmith@usna.edu',
'410-431-3456')
```

**UPDATE Statement**

```
UPDATE table_name
SET column_name1 = expression1 [,column_name2 = expression2,... ]
WHERE search_condition
```

**UPDATE command:**
```
UPDATE Students
SET PhoneNumber = '410-123-4567'
WHERE StudentNumber = 673;
```

**Bulk UPDATE command:**
```
UPDATE Students
SET PhoneNumber = '410-123-4567'
WHERE StudentLastName = 'Doe';
```

**DELETE Statement**

```
DELETE FROM table_name
WHERE search_condition
```

**DELETE command:**
```
DELETE FROM Students
WHERE StudentNumber = 190;
```

If you omit the WHERE clause, you will delete every row in the table!

Another example:
```
DELETE FROM Departments
WHERE DepartmentName = 'ComSci'
```

Integrity constraints?!

The SQL SELECT Statement

- Basic SQL Query:
```
SELECT [DISTINCT] column_name(s) | * FROM table_name(s) [WHERE conditions]
```

**SELECTing All Columns:**

```
The Asterisk (*) Keyword

SELECT *
FROM Students;
```

```
<table>
<thead>
<tr>
<th>StudentNumber</th>
<th>StudentLastName</th>
<th>StudentFirstName</th>
<th>Email</th>
<th>PhoneNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Smith</td>
<td>John</td>
<td><a href="mailto:smith@usna.edu">smith@usna.edu</a></td>
<td>410-431-3456</td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td>Jane</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td></td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td>Bob</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>443-451-7865</td>
</tr>
<tr>
<td>443-451-7865</td>
<td>410-431-3456</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

```
```
Specific Columns and Rows from One Table

```
SELECT StudentNumber, StudentLastName, StudentFirstName
FROM Students
WHERE MajorDeptName = 'ComSci';
```

<table>
<thead>
<tr>
<th>Student Number</th>
<th>Student LastName</th>
<th>Student FirstName</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Smith</td>
<td>John</td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td>Jane</td>
</tr>
</tbody>
</table>

The DISTINCT Keyword

```
SELECT SName
FROM Students;
```

```
SELECT DISTINCT SName
FROM Students;
```

Students, Courses, Enrolled

Find the names of students enrolled in IT420

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.Snb = E.SNb AND E.Cid = 'IT420'
```

<table>
<thead>
<tr>
<th>S.Nb</th>
<th>SName</th>
<th>Email</th>
<th>Cid</th>
<th>CName</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>IT420</td>
<td>Database</td>
<td>ComSci</td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>IT340</td>
<td>Networks</td>
<td>ComSci</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>SM121</td>
<td>Calculus</td>
<td>Math</td>
</tr>
</tbody>
</table>

```
SELECT - Conceptual Evaluation Strategy
```

- Semantics of an SQL query defined in terms of the following conceptual evaluation strategy:
  - Compute the cross-product of table_names
  - Discard resulting rows if they fail condition
  - Delete columns that are not in column_names
  - If DISTINCT is specified, eliminate duplicate rows
  - This strategy is probably the least efficient way to compute a query!
  - An optimizer will find more efficient strategies to compute the same answers.

Example Conceptual Evaluation

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.Snb = E.SNb AND E.Cid = 'IT420'
```

<table>
<thead>
<tr>
<th>S.Nb</th>
<th>SName</th>
<th>Email</th>
<th>E.SNb</th>
<th>Cid</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>190</td>
<td>IT340</td>
<td>Spring06</td>
</tr>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>312</td>
<td>IT420</td>
<td>Fall05</td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>190</td>
<td>IT340</td>
<td>Spring06</td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>312</td>
<td>IT420</td>
<td>Fall05</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>190</td>
<td>IT340</td>
<td>Spring06</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>312</td>
<td>IT420</td>
<td>Fall05</td>
</tr>
</tbody>
</table>
Example Conceptual Evaluation

```sql
SELECT SName
FROM Students S, Enrolled E
WHERE S.SnB = E.SnB AND E.Cid = ‘IT420’
```

<table>
<thead>
<tr>
<th>SnB</th>
<th>SName</th>
<th>Email</th>
<th>SnB</th>
<th>Cid</th>
<th>Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td>312</td>
<td>IT340</td>
<td>Spring2006</td>
</tr>
<tr>
<td>190</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>312</td>
<td>IT340</td>
<td>Spring2006</td>
</tr>
</tbody>
</table>

Modified Query

```sql
SELECT SnB
FROM Students S, Enrolled E
WHERE S.SnB = E.SnB AND E.Cid = ‘IT420’
```

- Would the result be different with DISTINCT?

LIKE and Wildcards

```sql
SELECT *
FROM Students
WHERE SnB LIKE ‘_9_%’
```

- SQL 92 Standard (SQL Server, Oracle, etc.):
  - _ = Exactly one character
  - % = Any set of one or more characters

- MS Access:
  - ? = Exactly one character
  - * = Any set of one or more characters

WHERE Clause Options

- AND, OR
- IN, NOT IN, BETWEEN

```sql
SELECT SnB
FROM Students S, Enrolled E
WHERE S.SnB = E.SnB AND E.Cid NOT IN (‘ComSci’, ‘Math’)```
Class Exercise

- Students(SNb, SName, Email)
- Courses(Cid, CName, Dept)
- Enrolled(SNb, Cid, Semester)

- Find the student number and name for each student enrolled in ‘Spring2007’ semester
- Find the names of all students enrolled in ‘ComSci’ courses