SY306 Web and Databases for Cyber Operations

SQL: Structured Query Language

SQL - The Language of Databases

- Developed by IBM in the 1970s
- Create and process database data
- SQL programming is a critical skill !!!
Relational Query Languages

- A major strength of the relational model:
  - supports simple, powerful querying of data
- Ad-hoc queries
- High-level (declarative) languages
  - Queries can be written intuitively
  - DBMS is responsible for efficient evaluation.

The SQL SELECT Statement

- Basic SQL Query:
  
  SELECT [DISTINCT] column_name(s) | *
  FROM table_name(s)
  [WHERE conditions]
  [ORDER BY column_name1 [ASC|DSC], …]
Selecting All Columns: 
The Asterisk (*) Keyword

**SELECT**  *  
**FROM**  Students;

<table>
<thead>
<tr>
<th>Student Number</th>
<th>Student LastName</th>
<th>Student FirstName</th>
<th>Email</th>
<th>PhoneNumber</th>
<th>Major</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
<td>Smith</td>
<td>John</td>
<td><a href="mailto:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>410-431-3456</td>
<td>SCY</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>
<td>SCY</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td>Jane</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>443-451-7865</td>
<td>Math</td>
</tr>
</tbody>
</table>

Specific Columns and Rows from One Table

**SELECT**  StudentNumber,  
           StudentLastName,  
           StudentFirstName  
**FROM**  Students  
**WHERE**  Major = ‘SCY’;

<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

```sql
SELECT StudentLastName FROM Students;
```

<table>
<thead>
<tr>
<th>StudentLastName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
</tr>
<tr>
<td>Doe</td>
</tr>
<tr>
<td>Doe</td>
</tr>
</tbody>
</table>

```sql
SELECT DISTINCT StudentLastName FROM Students;
```

<table>
<thead>
<tr>
<th>StudentLastName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
</tr>
<tr>
<td>Doe</td>
</tr>
</tbody>
</table>

Class Exercise

- Department(`DeptName`, `ChairName`, `WebAddress`, `DivName`)

- Find the name of the Chair of the ‘Math’ Department
WHERE Clause Options

- **AND, OR**
- **IN, NOT IN, BETWEEN**
- **LIKE**
  
  Wild cards:
  - SQL-92 Standard (SQL Server, Oracle, etc.):
    - `_` = Exactly one character
    - `%` = Any set of characters (zero or more)
  - MS Access
    - `?` = Exactly one character
    - `*` = Any set of characters (zero or more)

**Example:**
Students(SNb, SName, Email, Major)
Find alpha and name of SCY or SCS students with SNb starting with ‘16’

```
SELECT SNb, SName
FROM Students
WHERE SNb LIKE '16%' AND
      Major IN ('SCY', 'SCS')
```

Sorting the Results

```
SELECT [DISTINCT] column_name(s) | *
FROM table_name(s)
[WHERE conditions]
[ORDER BY column_name(s) [ASC/DESC]]
```

**Example:**
Students(SNb, SName, Email, Major)

```
SELECT SNb, SName
FROM Students
ORDER BY SName ASC, SNb DESC
```
Summary (partial)

- **SELECT** `[DISTINCT] column_name(s)`
  FROM `table_name`
  WHERE `conditions`
  ORDER BY `column_name(s)` [ASC/DESC]

---

**SELECT from Two or More Tables**

Find the names of students enrolled in SY306

SELECT SName
FROM Students S, Enrolled E
WHERE S.Snb = E.SNb AND E.Cid = 'SY306'

<table>
<thead>
<tr>
<th>Students</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SNb</td>
<td>SName</td>
<td>Email</td>
<td></td>
</tr>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td></td>
</tr>
<tr>
<td>673</td>
<td>Doe</td>
<td><a href="mailto:jdoe@usna.edu">jdoe@usna.edu</a></td>
<td></td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Courses</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cid</td>
<td>CName</td>
<td>CDept</td>
<td></td>
</tr>
<tr>
<td>SY306</td>
<td>WebDbScy</td>
<td>ComSci</td>
<td></td>
</tr>
<tr>
<td>SY301</td>
<td>Data Structures</td>
<td>ComSci</td>
<td></td>
</tr>
<tr>
<td>SM121</td>
<td>Calculus1</td>
<td>Math</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enrolled</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SNb</td>
<td>Cid</td>
<td>Semester</td>
</tr>
<tr>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>312</td>
<td>SY306</td>
<td>Spring2015</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

```sql
SELECT SName
FROM Students S, Enrolled E
WHERE S.SnB = E.SnB AND E.Cid = 'SY306'
```

<table>
<thead>
<tr>
<th>S.SnB</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:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</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>SY301</td>
<td>Fall2015</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>SY306</td>
<td>Spring2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</td>
</tr>
</tbody>
</table>
Example Conceptual Evaluation

```sql
SELECT SName
FROM Students S, Enrolled E
WHERE S.Snb = E.SNb AND E.Cid = 'SY306'
```

<table>
<thead>
<tr>
<th>S.SNb</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:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</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>SY301</td>
<td>Fall2015</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>SY306</td>
<td>Spring2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</td>
</tr>
</tbody>
</table>

Example Conceptual Evaluation

```sql
SELECT SName
FROM Students S, Enrolled E
WHERE S.Snb = E.SNb AND E.Cid = 'SY306'
```

<table>
<thead>
<tr>
<th>SName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doe</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S.SNb</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:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>190</td>
<td>Smith</td>
<td><a href="mailto:jsmith@usna.edu">jsmith@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</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>SY301</td>
<td>Fall2015</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>SY306</td>
<td>Spring2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>190</td>
<td>SY301</td>
<td>Fall2015</td>
</tr>
<tr>
<td>312</td>
<td>Doe</td>
<td><a href="mailto:jdoe2@usna.edu">jdoe2@usna.edu</a></td>
<td>312</td>
<td>SY306</td>
<td>Spring2015</td>
</tr>
</tbody>
</table>
Modified Query

SELECT S.SNb
FROM Students S, Enrolled E
WHERE S.SNb = E.SNb AND E.Cid = 'SY306'

Would the result be different with DISTINCT?

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 ‘Spring2015’ semester

Find the names of all students enrolled in ‘ComSci’ courses
Limit (MySQL specific)

- Limit the number of rows in the result
- SELECT [DISTINCT] column_name(s) | aggregate_expr
  FROM table_name(s)
  WHERE row_conditions
  ORDER BY column_name(s) [ASC/DESC]
  LIMIT [offset,] row_count
- Returns at most row_count rows, starting with offset
  (offset of first row is 0)

UNION

- Students(Alpha, Lname, Fname, Email)
- GraduatedStudents(Alpha2, Lname2, Fname2, Email2)
- List the Alpha, last name and first name of all students
  (current or graduated)

SELECT Alpha, Lname, Fname
FROM Students
UNION
SELECT Alpha2, Lname2, Fname2
FROM GraduatedStudents

- Same number of columns and data types in the 2 select
- Duplicates in results are eliminated by default (use UNION ALL if want to keep duplicates)
Summary (partial)

- SELECT [DISTINCT] column_name(s) 
  FROM table_name(s) 
  WHERE conditions 
  ORDER BY column_name(s) [ASC/DESC] 
  LIMIT [offset,] row_count