SQL SELECT

Database Systems Lecture 7
Natasha Alechina

In this Lecture

• SQL SELECT
  • WHERE clauses
  • SELECT from multiple tables
  • JOINs
• For more information
  • Connolly and Begg Chapter 5
  • Ullman and Widom Chapter 6.1-6.3

SQL SELECT Overview

**SELECT**

[**DISTINCT** | **ALL**] <column-list>

FROM <table-names>

[**WHERE** <condition>]

[**ORDER BY** <column-list>]

[**GROUP BY** <column-list>]

[**HAVING** <condition>]

• ([ ] - optional, | - or)

Example Tables

<table>
<thead>
<tr>
<th>Student</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>S103</td>
<td>John</td>
</tr>
<tr>
<td>S104</td>
<td>Mary</td>
</tr>
<tr>
<td>S105</td>
<td>Jane</td>
</tr>
<tr>
<td>S106</td>
<td>Mark</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DISTINCT and ALL

- Sometimes you end up with duplicate entries
- Using **DISTINCT** removes duplicates
- Using **ALL** retains them - this is the default

WHERE Clauses

- Usually you don’t want all the rows
  • A **WHERE** clause restricts the rows that are returned
  • It takes the form of a condition - only those rows that satisfy the condition are returned
- Example conditions:
  • Mark < 40
  • First = 'John'
  • First <> 'John'
  • First = Last
  • (First = 'John') AND (Last = 'Smith')
  • (Mark < 40) OR (Mark > 70)
WHERE Examples

SELECT * FROM Grade
WHERE Mark >= 60

SELECT DISTINCT ID
FROM Grade
WHERE Mark >= 60

ID Code Mark
S103 DBS 72
S104 PR1 68
S104 IAI 65
S107 PR1 76
S107 PR2 60

WHERE Example

- Given the table
- Write an SQL query to find a list of the ID numbers and marks in IAI of students who have passed (scored 40 or higher) IAI

ID Code Mark
S103 DBS 72
S104 PR1 68
S104 IAI 65
S106 PR2 43
S107 PR1 76
S107 PR2 60
S107 IAI 35

One Solution

We only want the ID and Mark, not the Code
Single quotes around the string

SELECT ID, Mark FROM Grade
WHERE (Code = 'IAI') AND (Mark >= 40)

We're only interested in IAI
We're looking for entries with pass marks

SELECT from Multiple Tables

- Often you need to combine information from two or more tables
- You can get the effect of a product by using

SELECT * FROM Table1, Table2...

- If the tables have columns with the same name ambiguity results
- You resolve this by referencing columns with the table name

SELECT from Multiple Tables

SELECT First, Last, Mark
FROM Student, Grade
WHERE (Student.ID = Grade.ID) AND (Mark >= 40)
SELECT from Multiple Tables

### Example 1

```sql
SELECT ID, First, Last, ID, Code, Mark
FROM Student, Grade
WHERE (Student.ID = Grade.ID) AND ...
```

<table>
<thead>
<tr>
<th>ID</th>
<th>First</th>
<th>Last</th>
<th>ID</th>
<th>Code</th>
<th>Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>S103</td>
<td>John</td>
<td>Smith</td>
<td>S103</td>
<td>DBS</td>
<td>72</td>
</tr>
<tr>
<td>S103</td>
<td>John</td>
<td>Smith</td>
<td>S103</td>
<td>IAI</td>
<td>58</td>
</tr>
<tr>
<td>S104</td>
<td>Mary</td>
<td>Jones</td>
<td>S104</td>
<td>AI</td>
<td>65</td>
</tr>
<tr>
<td>S106</td>
<td>Mark</td>
<td>Jones</td>
<td>S106</td>
<td>PR2</td>
<td>43</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
<td>Brown</td>
<td>S107</td>
<td>PR1</td>
<td>76</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
<td>Brown</td>
<td>S107</td>
<td>IAI</td>
<td>35</td>
</tr>
</tbody>
</table>

### Example 2

```sql
SELECT First, Last, Mark
FROM Student, Grade
WHERE (Student.ID = Grade.ID) AND (Mark >= 40)
```

<table>
<thead>
<tr>
<th>First</th>
<th>Last</th>
<th>Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Smith</td>
<td>72</td>
</tr>
<tr>
<td>John</td>
<td>Smith</td>
<td>58</td>
</tr>
<tr>
<td>Mary</td>
<td>Jones</td>
<td>68</td>
</tr>
<tr>
<td>Mary</td>
<td>Jones</td>
<td>65</td>
</tr>
<tr>
<td>Mark</td>
<td>Jones</td>
<td>43</td>
</tr>
<tr>
<td>John</td>
<td>Brown</td>
<td>76</td>
</tr>
<tr>
<td>John</td>
<td>Brown</td>
<td>60</td>
</tr>
</tbody>
</table>

### Example 3

```sql
SELECT * FROM Student, Grade, Course
WHERE Student.ID = Grade.ID AND Course.Code = Grade.Code
```

### Example 4

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 5

```sql
SELECT * FROM Student, Grade, Course
WHERE Student.ID = Grade.ID AND Course.Code = Grade.Code
```

<table>
<thead>
<tr>
<th>ID</th>
<th>First</th>
<th>Last</th>
<th>ID</th>
<th>Code</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>S103</td>
<td>John</td>
<td>Smith</td>
<td>S103</td>
<td>DBS</td>
<td>Database Systems</td>
</tr>
<tr>
<td>S103</td>
<td>John</td>
<td>Smith</td>
<td>S103</td>
<td>IAI</td>
<td>Intro to AI</td>
</tr>
<tr>
<td>S104</td>
<td>Mary</td>
<td>Jones</td>
<td>S104</td>
<td>PR1</td>
<td>Programming 1</td>
</tr>
<tr>
<td>S104</td>
<td>Mary</td>
<td>Jones</td>
<td>S104</td>
<td>IAI</td>
<td>Intro to AI</td>
</tr>
<tr>
<td>S106</td>
<td>Mark</td>
<td>Jones</td>
<td>S106</td>
<td>PR2</td>
<td>Programming 2</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
<td>Brown</td>
<td>S107</td>
<td>PR1</td>
<td>Programming 1</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
<td>Brown</td>
<td>S107</td>
<td>PR2</td>
<td>Programming 2</td>
</tr>
<tr>
<td>S107</td>
<td>John</td>
<td>Brown</td>
<td>S107</td>
<td>IAI</td>
<td>Intro to AI</td>
</tr>
</tbody>
</table>

### Example 6

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 7

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 8

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 9

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 10

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 11

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 12

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 13

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 14

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 15

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 16

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 17

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 18

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 19

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 20

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 21

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 22

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 23

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 24

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later

### Example 25

JOINs can be used to combine tables:
- There are many types of JOIN:
  - CROSS JOIN
  - INNER JOIN
  - NATURAL JOIN
  - OUTER JOIN
  - OUTER JOINs are linked with NULLs - more later
**CROSS JOIN**

SELECT * FROM Student CROSS JOIN Enrolment

**NATURAL JOIN**

SELECT * FROM Student NATURAL JOIN Enrolment

**CROSS and NATURAL JOIN**

SELECT * FROM A CROSS JOIN B

• is the same as

SELECT * FROM A NATURAL JOIN B

• is the same as

SELECT A.col1, A.col2, [and all other columns apart from B.col1, B.col2] FROM A, B

WHERE A.col1 = B.col1

AND A.col2 = B.col2

...AND A.coln = B.coln

(this assumes that coln in A and B have common names)

**INNER JOIN**

• INNER JOINS specify a condition which the pairs of rows satisfy

SELECT * FROM A INNER JOIN B ON <condition>

• Can also use

SELECT * FROM A INNER JOIN B USING (col1, col2...)

• Chooses rows where the given columns are equal

**INNER JOIN**

SELECT * FROM Student INNER JOIN Enrolment USING (ID)

SELECT * FROM Buyer INNER JOIN Property ON Price <= Budget

<table>
<thead>
<tr>
<th>Name</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>100,000</td>
</tr>
<tr>
<td>Jones</td>
<td>150,000</td>
</tr>
<tr>
<td>Green</td>
<td>10,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
<tr>
<td>12 Queen St</td>
<td>125,000</td>
</tr>
<tr>
<td>87 Oak Row</td>
<td>175,000</td>
</tr>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
<tr>
<td>12 Queen St</td>
<td>125,000</td>
</tr>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>100,000</td>
</tr>
<tr>
<td>Jones</td>
<td>150,000</td>
</tr>
<tr>
<td>Green</td>
<td>10,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
<tr>
<td>12 Queen St</td>
<td>125,000</td>
</tr>
<tr>
<td>87 Oak Row</td>
<td>175,000</td>
</tr>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
<tr>
<td>12 Queen St</td>
<td>125,000</td>
</tr>
<tr>
<td>15 High St</td>
<td>85,000</td>
</tr>
</tbody>
</table>
INNER JOIN

SELECT * FROM A INNER JOIN B
ON <condition>
• is the same as
SELECT * FROM A, B
WHERE <condition>

JOINs vs WHERE Clauses

• JOINs (so far) are not needed
• You can have the same effect by selecting from multiple tables with an appropriate WHERE clause
• So should you use JOINs or not?

• Yes, because
• They often lead to concise queries
• NATURAL JOINs are very common
• No, because
• Support for JOINs varies a fair bit among SQL dialects

Writing Queries

• When writing queries
  • There are often many ways to write the query
  • You should worry about being correct, clear, and concise in that order
  • Don’t worry about being clever or efficient

• Most DBMSs have query optimisers
  • These take a user’s query and figure out how to efficiently execute it
  • A simple query is easier to optimise
  • We’ll look at some ways to improve efficiency later

This Lecture in Exams

Find a list of all the CD titles. (1 mark)
Find a list of the titles of tracks that are more than 300 seconds long. (2 marks)
Find a list of the names of those artists who have a track on the CD with the title “Compilation”. (4 marks)

This Lecture in Exams

<table>
<thead>
<tr>
<th>CD</th>
<th>Num</th>
<th>Title</th>
<th>Time</th>
<th>aID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Violent</td>
<td>239</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Every Girl</td>
<td>410</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Breather</td>
<td>217</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>Part of Me</td>
<td>279</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Star</td>
<td>362</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Teaboy</td>
<td>417</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CD</th>
<th>Title</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mix</td>
<td>9.99</td>
</tr>
<tr>
<td>2</td>
<td>Compilation</td>
<td>12.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Artist</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Stellar</td>
</tr>
<tr>
<td>2 Cloudboy</td>
</tr>
</tbody>
</table>

Next Lecture

• More SQL SELECT
• Aliases
• ‘Self-joins’
• Subqueries
• IN, EXISTS, ANY, ALL
• For more information
  • Connolly and Begg Chapter 5
  • Ullman and Widom Chapter 6