More SQL Select

Database Systems Lecture 8
Natasha Alechina

In This Lecture

• More SQL Select
• Aliases
• ‘Self-joins’
• Subqueries
• IN, EXISTS, ANY, ALL
• For more information
  • Connoly and Begg Chapter 5

More SQL SELECT

SQL SELECT Overview

SELECT
[DISTINCT | ALL] <column-list>
FROM <table-names>
[WHERE <condition>]
[GROUP BY <column-list>]
[HAVING <condition>]
([ ]= optional, | or )

More SQL SELECT

Aliases

• Aliases rename columns or tables to
• Make names more meaningful
• Make names shorter and easier to type
• Resolve ambiguous names

• Two forms:
  • Column alias
    SELECT column AS newName...
  • Table alias
    SELECT ...
    FROM table AS newName

This 'AS' is optional, but Oracle doesn't accept it at all

More SQL SELECT

Example

Employee

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>John</td>
</tr>
<tr>
<td>124</td>
<td>Mary</td>
</tr>
</tbody>
</table>

WorksIn

<table>
<thead>
<tr>
<th>ID</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Marketing</td>
</tr>
<tr>
<td>124</td>
<td>Sales</td>
</tr>
<tr>
<td>124</td>
<td>Marketing</td>
</tr>
</tbody>
</table>

SELECT
E.ID AS empID,
E.Name, W.Dept
FROM
Employee E
WorksIn W
WHERE
E.ID = W.ID

More SQL SELECT

Example

SELECT
E.ID AS empID,
E.Name, W.Dept
FROM
Employee E
WorksIn W
WHERE
E.ID = W.ID

More SQL SELECT
Aliases and 'Self-Joins'

Aliases can be used to copy a table, so that it can be combined with itself:

```
SELECT A.Name FROM Employee A, Employee B
WHERE A.Dept = B.Dept
AND B.Name = 'Andy'
```

The result is the names of all employees who work in the same department as Andy.
Subqueries

- A SELECT statement can be nested inside another query to form a subquery.
- The results of the subquery are passed back to the containing query.
- E.g. get the names of people who are in Andy’s department:

\[
\text{SELECT Name FROM Employee WHERE Dept = (SELECT Dept FROM Employee WHERE Name='Andy')}\]

- First the subquery is evaluated, returning the value ‘Marketing’.
- This result is passed to the main query:

\[
\text{SELECT Name FROM Employee WHERE Dept = 'Marketing'}\]

Subqueries

- Often a subquery will return a set of values rather than a single value.
- You can’t directly compare a single value to a set.
- Options:
  - IN - checks to see if a value is in the set.
  - EXISTS - checks to see if the set is empty or not.
  - ALL/ANY - checks to see if a relationship holds for every/one member of the set.

- Using IN we can see if a given value is in a set of values.
- NOT IN checks to see if a given value is not in the set.
- The set can be given explicitly or from a subquery.

\[
\text{SELECT * FROM Employee WHERE Department IN ('Marketing', 'Sales')}\]

\[
\text{SELECT * FROM Employee WHERE Name NOT IN (SELECT Manager FROM Employee)}\]

( NOT ) IN

- Using IN we can see if a given value is in a set of values.
- NOT IN checks to see if a given value is not in the set.
- The set can be given explicitly or from a subquery.

\[
\text{SELECT <columns> FROM <tables> WHERE <value> IN <set>}\]

<table>
<thead>
<tr>
<th>Name</th>
<th>Department</th>
<th>Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Marketing</td>
<td>Chris</td>
</tr>
<tr>
<td>Mary</td>
<td>Marketing</td>
<td>Chris</td>
</tr>
<tr>
<td>Chris</td>
<td>Marketing</td>
<td>Jane</td>
</tr>
<tr>
<td>Peter</td>
<td>Sales</td>
<td>Jane</td>
</tr>
<tr>
<td>Jane</td>
<td>Management</td>
<td>Jane</td>
</tr>
</tbody>
</table>
(NOT) IN
- First the subquery
  ```sql
  SELECT Manager
  FROM Employee
  ```
- is evaluated giving
  - This gives
  ```sql
  SELECT *
  FROM Employee
  WHERE Name NOT IN ('Chris', 'Jane')
  ```

(NOT) EXISTS
- Using EXISTS we see if there is at least one element in a set
- NOT EXISTS is true if the set is empty
- The set is always given by a subquery
  ```sql
  SELECT <columns>
  FROM <tables>
  WHERE EXISTS <set>
  ```

More SQL SELECT

(NOT) EXISTS

<table>
<thead>
<tr>
<th>Employee</th>
<th>Name</th>
<th>Department</th>
<th>Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Marketing</td>
<td>Chris</td>
<td></td>
</tr>
<tr>
<td>Mary</td>
<td>Marketing</td>
<td>Chris</td>
<td></td>
</tr>
<tr>
<td>Chris</td>
<td>Marketing</td>
<td>Jane</td>
<td></td>
</tr>
<tr>
<td>Peter</td>
<td>Sales</td>
<td>Jane</td>
<td></td>
</tr>
</tbody>
</table>

More SQL SELECT

ANY and ALL
- ANY and ALL compare a single value to a set of values
- They are used with comparison operators like =, >, <, <=, <>
  ```sql
  val = ANY (set) is true if there is at least one member of the set equal to the value
  val = ALL (set) is true if all members of the set are equal to the value
  ```

More SQL SELECT

ALL
- Find the names of the employee(s) who earn the highest salary
  ```sql
  SELECT Name
  FROM Employee
  WHERE Salary >= ALL (SELECT Salary FROM Employee)
  ```

More SQL SELECT

ANY
- Find the names of employee(s) who earn more than someone else
  ```sql
  SELECT Name
  FROM Employee
  WHERE Salary > ANY (SELECT Salary FROM Employee)
  ```

More SQL SELECT
Word Searches

• Word Searches
  • Commonly used for searching product catalogues etc.
  • Want to be able to search by keyword
  • Want to be able to use word stemming for flexible searching

• EG: Given a database of my books
  • Searching for "crypt" would return
    • "Cryptonomicon" by Neil Stephenson
    • "Applied Cryptography" by Bruce Schneier

To do a word search we can keep
• A table of items to be searched
• A table of keywords
• A linking table saying which keywords belong to which items

To search we can use queries like

```sql
SELECT * FROM Items
WHERE itmID IN (
    SELECT itmID FROM ItemKey
    WHERE keyID IN (
        SELECT keyID FROM Keywords
        WHERE keyWord LIKE 'crypt%'))
```

Sometimes you need to search for a set of words
• To find entries with all words you can link conditions with AND
• To find entries with any of the words use OR

```
SELECT * FROM Items
WHERE itmID IN (
    SELECT itmID FROM ItemKey
    WHERE keyID IN (
        SELECT keyID FROM Keywords
        WHERE keyWord LIKE 'word1%'))
AND
    itmID IN (
        SELECT itmID FROM ItemKey
        WHERE keyID IN (
            SELECT keyID FROM Keywords
            WHERE keyWord LIKE 'word2%'))
```

This Lecture in Exams

Find a list of the names of those artists who have a track on the CD with the title "Compilation".

Note that this is one of the questions from the previous lecture, but there are alternative solutions using subqueries. Try solving this with a query where you never list more than one table for a single SELECT statement.
Next Lecture

- Yet more SQL
  - ORDER BY
  - Aggregate functions
  - GROUP BY and HAVING
  - UNION etc.
- For more information
  - Connoly and Begg Chapter 5