SQL: Multi-Relation Queries (Joins)

Abdu Alawini

University of Illinois at Urbana-Champaign

CS411: Database Systems

Some slides are adopted from D. Maier, S. Davidson and A. Parameswaran

Learning Objectives

After this lecture, you should be able to:

- Write **INNER, NATURAL and OUTER JOIN SQL** queries
- Express the effect of Null on SQL queries

Multi-relation Queries

- Interesting queries often combine data from more than one relation.
- We can address several relations in one query by listing them all in the FROM clause.
- Distinguish attributes of the same name by "<relation>.<attribute>"

Example of Multi-Relation Query

```
SELECT      A.Owner, A.Balance
FROM        Account A, Deposit D
WHERE       D.AcctNo = A.Number and A.Balance > 1000;
```

"A" is a correlation name for Account

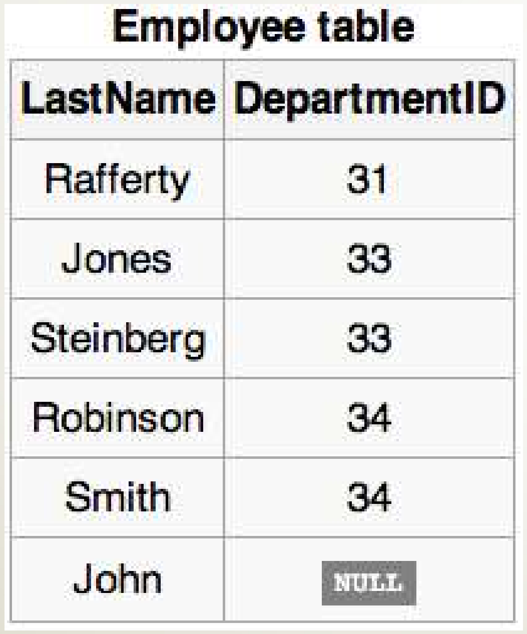
and

"D" is a correlation name for Deposit.

Correlation names are like local variables – they hold one tuple or row from the corresponding table.

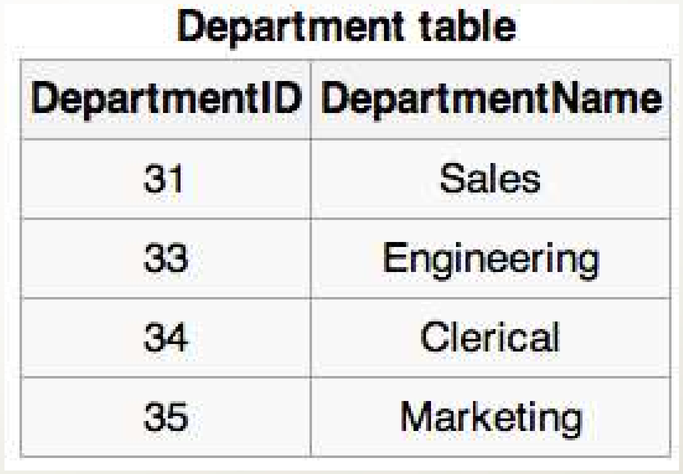
You choose correlation names when you write the query.

Example Database



Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

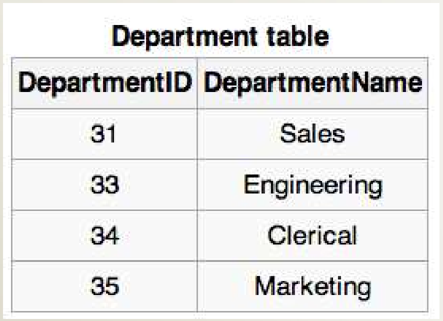


Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

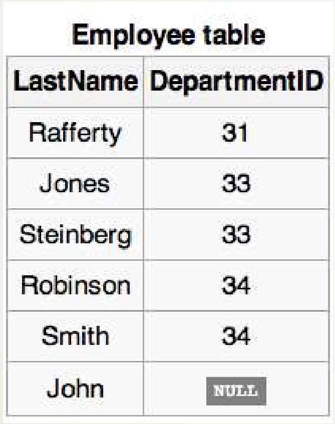
Cross Product

Department × Employee



Department table

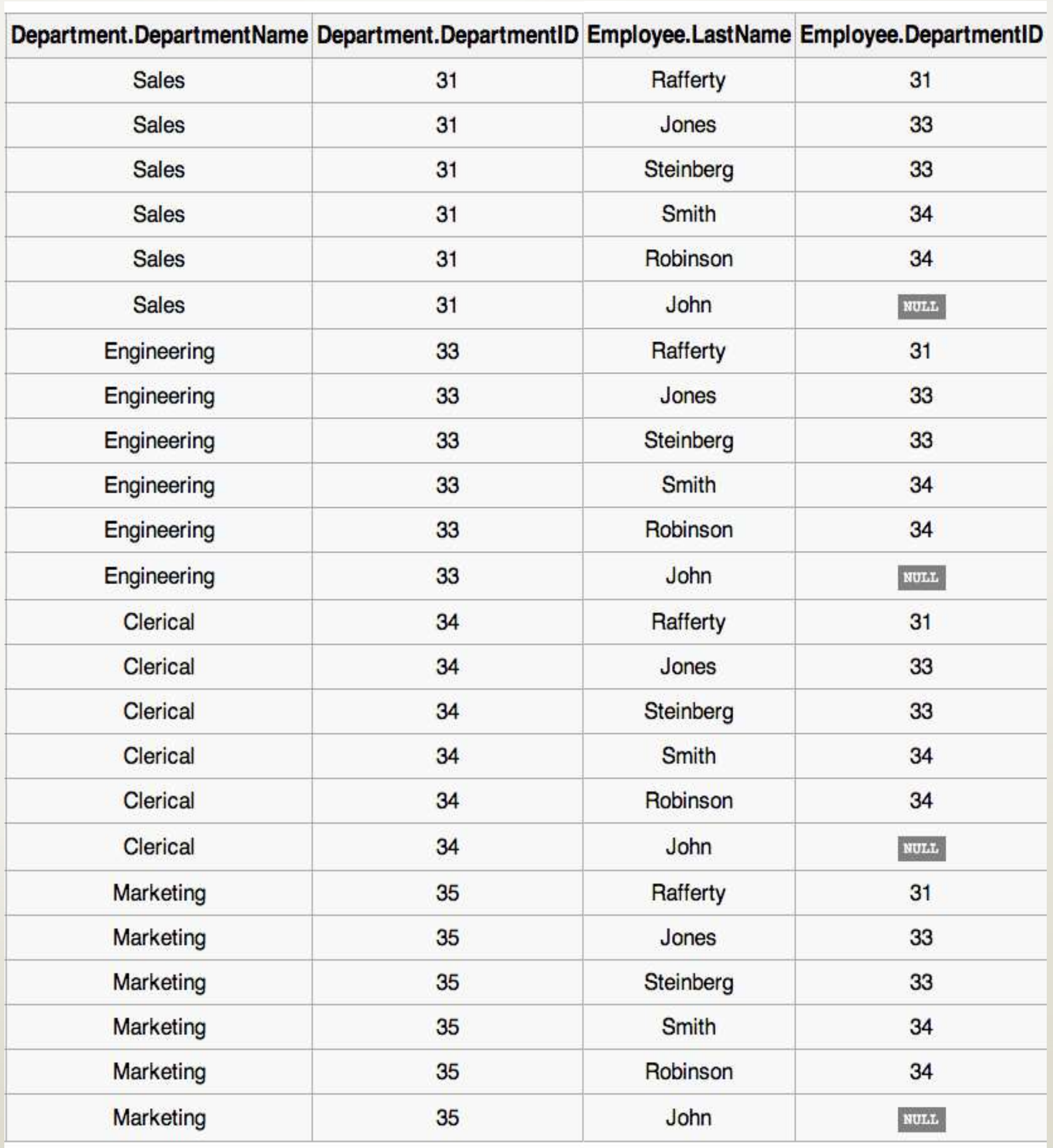
DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing



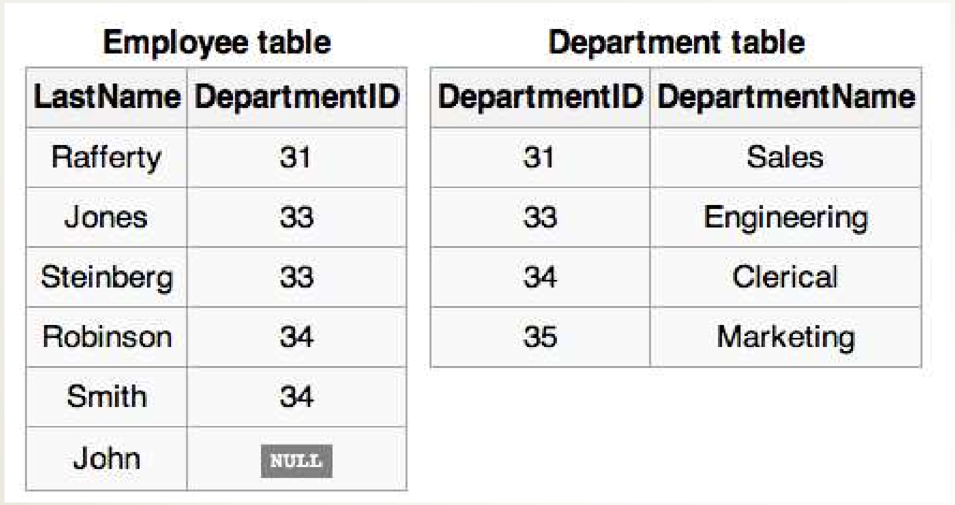
Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

```
SELECT *
FROM Department, Employee
```



Department.DepartmentName	Department.DepartmentID	Employee.LastName	Employee.DepartmentID
Sales	31	Rafferty	31
Sales	31	Jones	33
Sales	31	Steinberg	33
Sales	31	Smith	34
Sales	31	Robinson	34
Sales	31	John	NULL
Engineering	33	Rafferty	31
Engineering	33	Jones	33
Engineering	33	Steinberg	33
Engineering	33	Smith	34
Engineering	33	Robinson	34
Engineering	33	John	NULL
Clerical	34	Rafferty	31
Clerical	34	Jones	33
Clerical	34	Steinberg	33
Clerical	34	Smith	34
Clerical	34	Robinson	34
Clerical	34	John	NULL
Marketing	35	Rafferty	31
Marketing	35	Jones	33
Marketing	35	Steinberg	33
Marketing	35	Smith	34
Marketing	35	Robinson	34
Marketing	35	John	NULL



Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

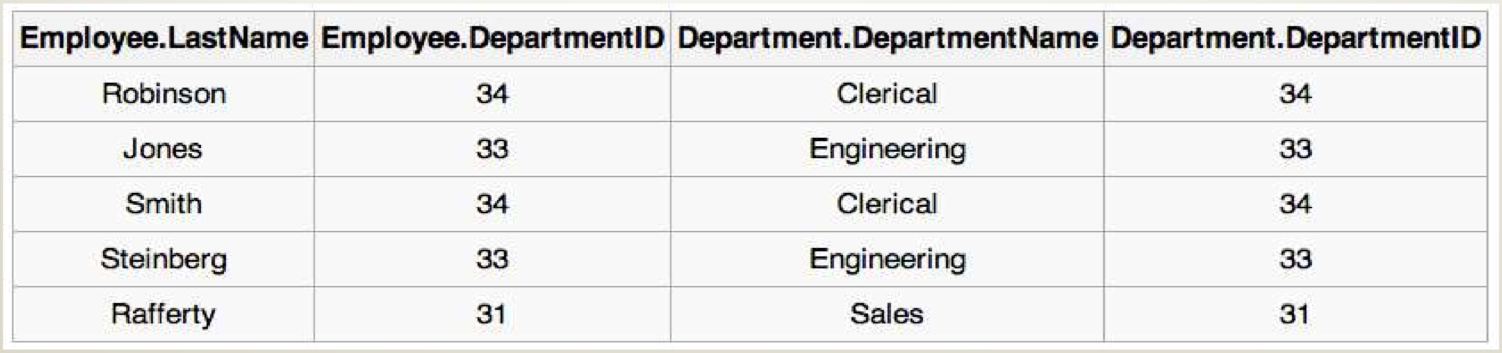
Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

Equijoin

$$\textbf{Employee} \bowtie_{\textbf{Employee.DeptID} = \textbf{Department.DeptID}} \textbf{Department}$$

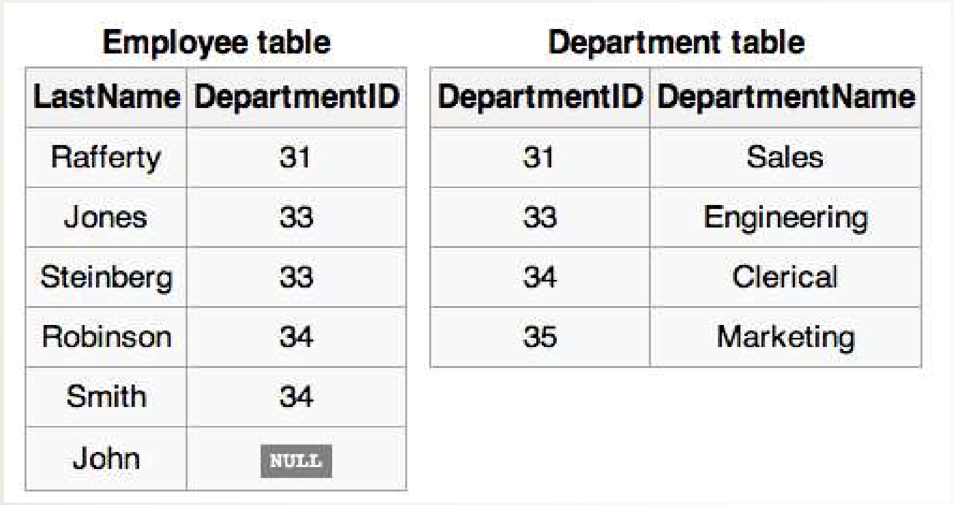
```
SELECT *
FROM Employee emp JOIN Department dept
                ON emp.DepartmentID = dept.DepartmentID
```



Employee.LastName	Employee.DepartmentID	Department.DepartmentName	Department.DepartmentID
Robinson	34	Clerical	34
Jones	33	Engineering	33
Smith	34	Clerical	34
Steinberg	33	Engineering	33
Rafferty	31	Sales	31

Natural Join

Employee ⋈ Department



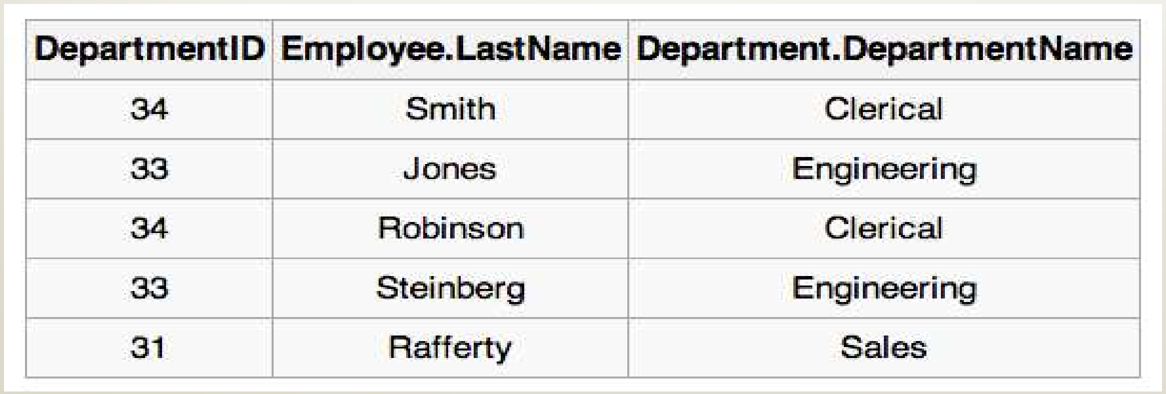
Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

```
SELECT  *
FROM Employee emp NATURAL JOIN Department dept
```



DepartmentID	Employee.LastName	Department.DepartmentName
34	Smith	Clerical
33	Jones	Engineering
34	Robinson	Clerical
33	Steinberg	Engineering
31	Rafferty	Sales

Null Values

- Tuples in SQL relations can have NULL as a value for one or more components.
- Meaning depends on context. Two common cases:
  - *Missing value* : e.g., we know Royal cafe has some address, but we don’t know what it is.
  - *Inapplicable* : e.g., the value of attribute *spouse* for an unmarried person.

Comparing NULL's to Values

- The logic of conditions in SQL is really 3-valued logic: TRUE, FALSE, UNKNOWN.
- *Comparison:* When any value is compared with NULL, the truth value is UNKNOWN.
- *Outcome:* But a query only produces a tuple in the answer if its truth value for the WHERE clause is TRUE (not FALSE or UNKNOWN).

Three-Valued Logic

- To understand how AND, OR, and NOT work in 3-valued logic, think of TRUE = 1, FALSE = 0, and UNKNOWN = ½.
- AND = MIN; OR = MAX, NOT(x) = 1-x.
- Example:

TRUE AND (FALSE OR NOT(UNKNOWN))

= MIN(1, MAX(0, (1 - ½))) =

MIN(1, MAX(0, ½)) = MIN(1, ½) = ½

= UNKNOWN.

Another Example

- C1 AND C2 = min(C1, C2)
- C1 OR C2 = max(C1, C2)
- NOT C1 = 1 – C1

```
SELECT *
FROM Person
WHERE  (age < 25) AND
       (height > 6 OR weight > 190)
```

E.g.
age=20
height=NULL
weight=200

Rule in SQL: include only tuples that yield TRUE

Nulls and Joins

- Sometimes need special variations of joins:
  - I want to see all employees and their departments
  - … But what if there’s a department with no employees?
  - Or what if an employee has not been assigned to a department?
- Outer join:
  - Most common is *left outer join*

Outer Joins

- Left outer join:
  - Include the left tuple even if there’s no match
- Right outer join:
  - Include the right tuple even if there’s no match
- Full outer join:
  - Include both the left and right tuples even if there’s no match

Left Outer Join

Employee ⟕ Department
Employee.DepartmentID = Department.DepartmentID

Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

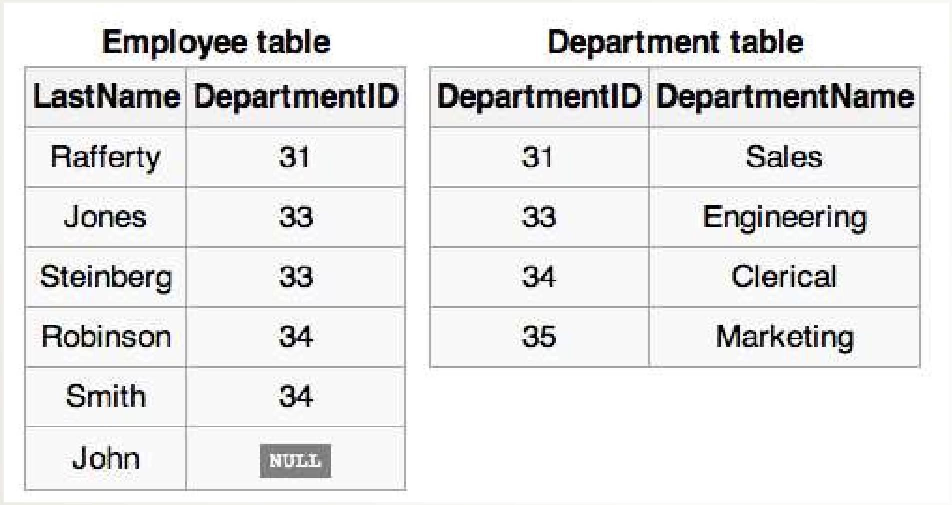
```
SELECT  *
FROM Employee emp LEFT OUTER JOIN Department dept
                        ON emp.DepartmentID = dept.DepartmentID
```

Employee.LastName	Employee.DepartmentID	Department.DepartmentName	Department.DepartmentID
Jones	33	Engineering	33
Rafferty	31	Sales	31
Robinson	34	Clerical	34
Smith	34	Clerical	34
John	NULL	NULL	NULL
Steinberg	33	Engineering	33

Right Outer Join

Employee ⟖ Department

Employee.DepartmentID = Department.DepartmentID



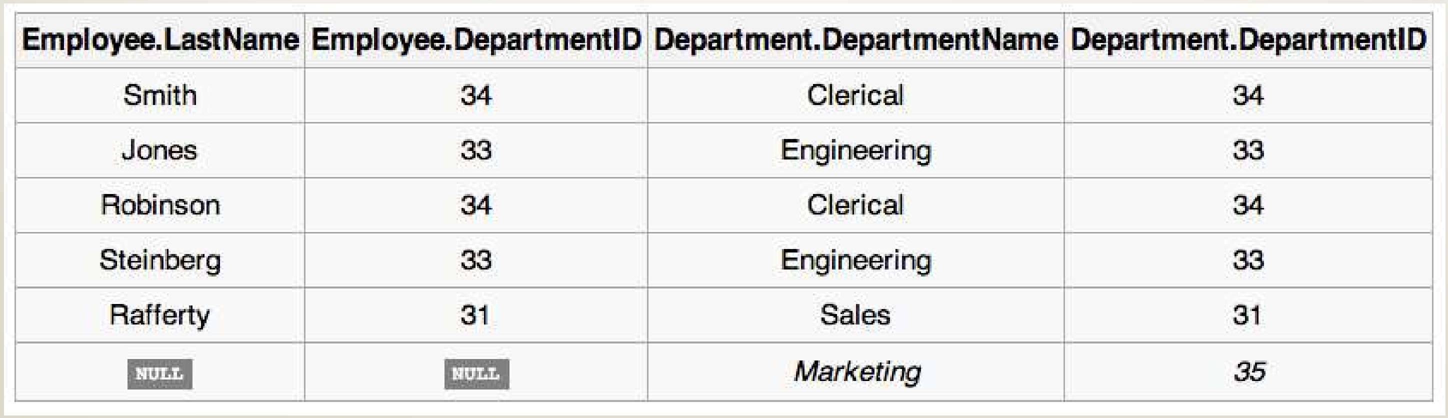
Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

```
SELECT  *
FROM Employee emp RIGHT OUTER JOIN Department dept
        ON emp.DepartmentID = dept.DepartmentID
```



Employee.LastName	Employee.DepartmentID	Department.DepartmentName	Department.DepartmentID
Smith	34	Clerical	34
Jones	33	Engineering	33
Robinson	34	Clerical	34
Steinberg	33	Engineering	33
Rafferty	31	Sales	31
NULL	NULL	*Marketing*	*35*

Full Outer Join

Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Steinberg	33
Robinson	34
Smith	34
John	NULL

Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

Employee ⟗ Department
Employee.DepartmentID = Department.DepartmentID

```
SELECT  *
FROM Employee FULL OUTER JOIN Department dept
                ON emp.DepartmentID = dept.DepartmentID
```

Employee.LastName	Employee.DepartmentID	Department.DepartmentName	Department.DepartmentID
Smith	34	Clerical	34
Jones	33	Engineering	33
Robinson	34	Clerical	34
John	NULL	NULL	NULL
Steinberg	33	Engineering	33
Rafferty	31	Sales	31
NULL	NULL	*Marketing*	*35*