Displaying Data from Multiple Tables

Objectives

- After completing this lesson, you should be able to do the following:
  - Write SELECT statements to access data from more than one table using equality and nonequality joins
  - View data that generally does not meet a join condition by using outer joins
  - Join a table to itself

Obtaining Data from Multiple Tables

EMP

EMPNO	ENAME	...	DEPTNO
7839	KING	...	10
7698	BLAKE	...	30
...			
7934	MILLER	...	10

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

EMPNO	DEPTNO	LOC
7839	10	NEW YORK
7698	30	CHICAGO
7782	10	NEW YORK
7566	20	DALLAS
7654	30	CHICAGO
7499	30	CHICAGO

...

14 rows selected.

What Is a Join?

- Use a join to query data from more than one table.

```
SELECT    table1.column, table2.column
FROM      table1, table2
WHERE     table1.column1 = table2.column2;
```

  - Write the join condition in the WHERE clause.
  - Prefix the column name with the table name when the same column name appears in more than one table.

Cartesian Product

- A Cartesian product is formed when:
  - A join condition is omitted
  - A join condition is invalid
  - All rows in the first table are joined to all rows in the second table
- To avoid a Cartesian product, always include a valid join condition in a WHERE clause.

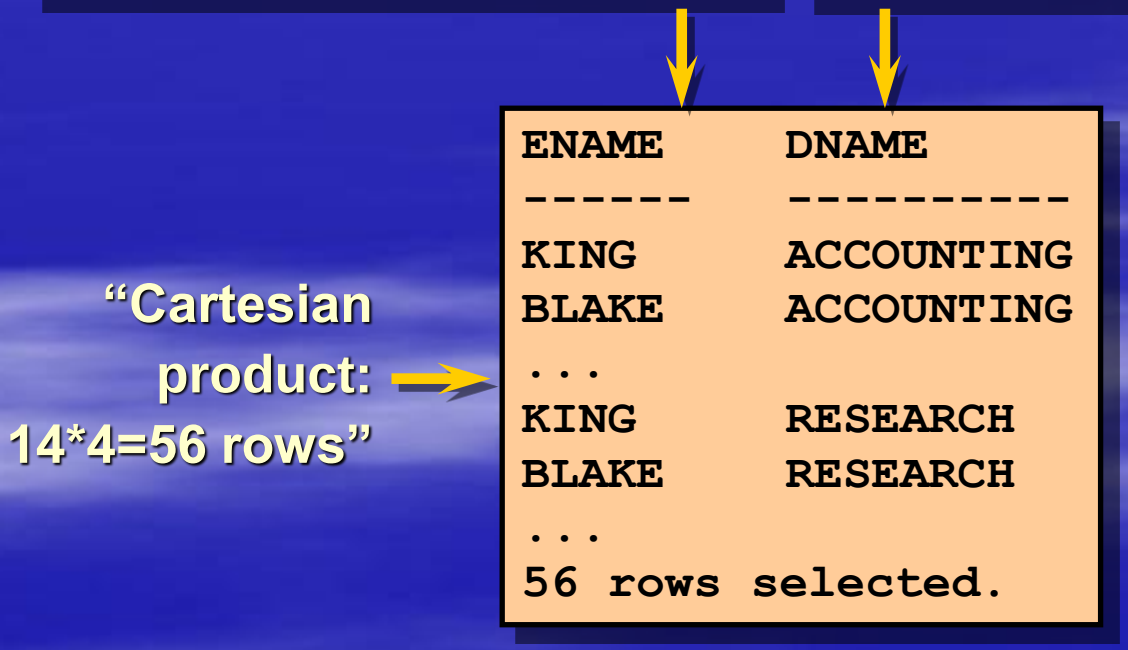
Generating a Cartesian Product

EMP (14 rows)

```
 EMPNO ENAME   ...  DEPTNO
------ -----   ...  ------
  7839 KING    ...      10
  7698 BLAKE   ...      30
   ...
  7934 MILLER  ...      10
```

DEPT (4 rows)

```
DEPTNO DNAME          LOC
------ ---------- --------
    10 ACCOUNTING NEW YORK
    20 RESEARCH   DALLAS
    30 SALES      CHICAGO
    40 OPERATIONS BOSTON
```



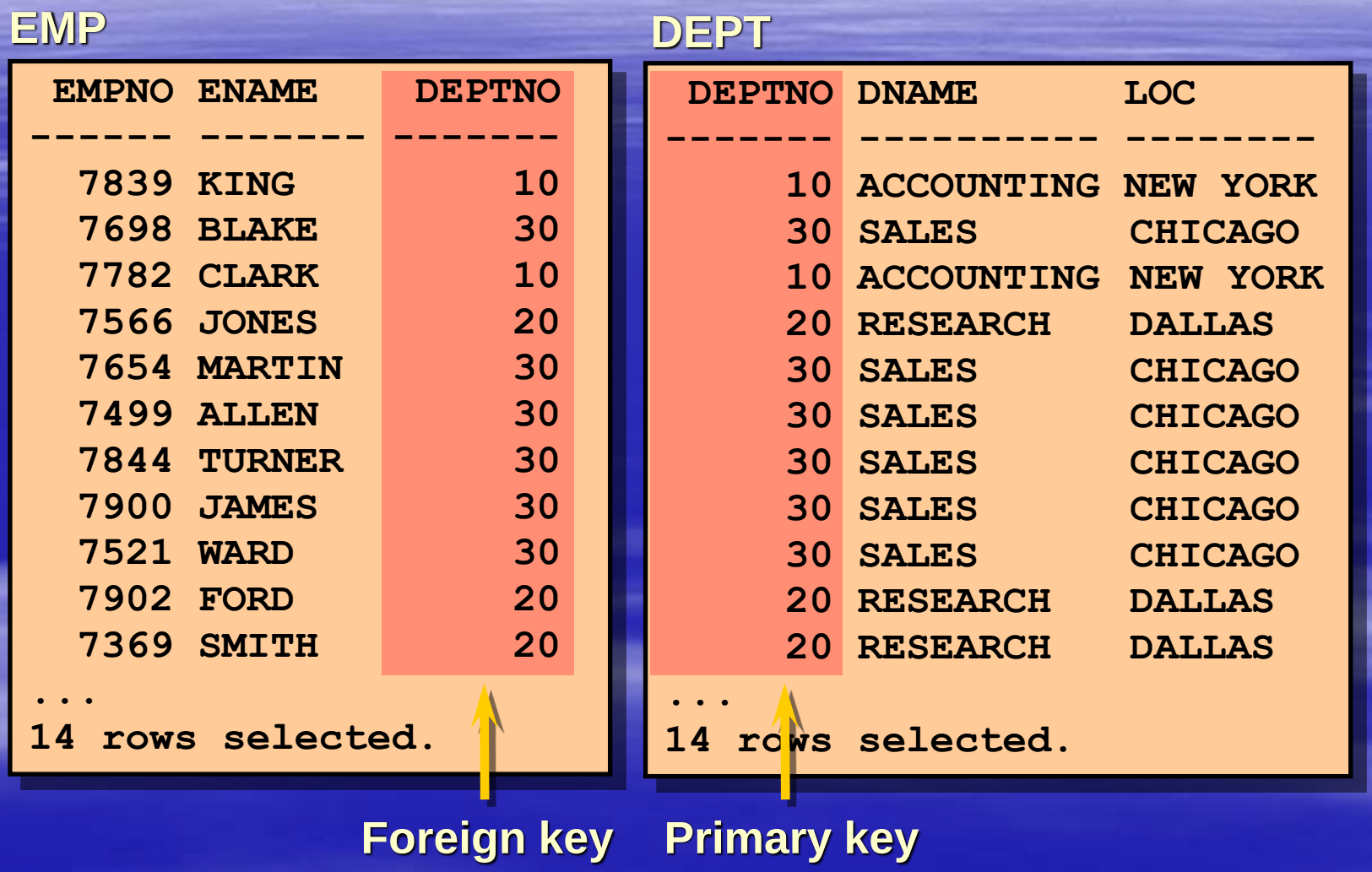
```
ENAME      DNAME
------     ----------
KING       ACCOUNTING
BLAKE      ACCOUNTING
...
KING       RESEARCH
BLAKE      RESEARCH
...
56 rows selected.
```

"Cartesian product: 14*4=56 rows"

Types of Joins

Equijoin **Non-equijoin** **Outer join** **Self join**

What Is an Equijoin?



Retrieving Records with Equijoins

```
SQL> SELECT   emp.empno, emp.ename, emp.deptno,
  2           dept.deptno, dept.loc
  3   FROM    emp, dept
  4   WHERE   emp.deptno=dept.deptno;
```

```
EMPNO ENAME      DEPTNO DEPTNO LOC
----- ------ ------ ------ ---------
 7839 KING         10     10 NEW YORK
 7698 BLAKE        30     30 CHICAGO
 7782 CLARK        10     10 NEW YORK
 7566 JONES        20     20 DALLAS
...
14 rows selected.
```

Qualifying Ambiguous Column Names

- Use table prefixes to qualify column names that are in multiple tables.
- Improve performance by using table prefixes.
- Distinguish columns that have identical names but reside in different tables by using column aliases.

Additional Search Conditions Using the AND Operator

EMP

EMPNO	ENAME	DEPTNO
7839	KING	10
7698	BLAKE	30
7782	CLARK	10
7566	JONES	20
7654	MARTIN	30
7499	ALLEN	30
7844	TURNER	30
7900	JAMES	30
7521	WARD	30
7902	FORD	20
7369	SMITH	20

...
14 rows selected.

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
30	SALES	CHICAGO
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
20	RESEARCH	DALLAS
20	RESEARCH	DALLAS

...
14 rows selected.

Using Table Aliases

- Simplify queries by using table aliases.

```
SQL> SELECT emp.empno, emp.ename, emp.deptno,
  2         dept.deptno, dept.loc
  3  FROM   emp, dept
  4  WHERE  emp.deptno=dept.deptno;
```

```
SQL> SELECT e.empno, e.ename, e.deptno,
  2         d.deptno, d.loc
  3  FROM   emp e, dept d
  4  WHERE  e.deptno=d.deptno;
```

Joining More Than Two Tables

CUSTOMER

NAME	CUSTID
JOCKSPORTS	100
TKB SPORT SHOP	101
VOLLYRITE	102
JUST TENNIS	103
K+T SPORTS	105
SHAPE UP	106
WOMENS SPORTS	107
...	...

9 rows selected.

ORD

CUSTID	ORDID
101	610
102	611
104	612
106	601
102	602
106	
106	
...	

21 rows s

ITEM

ORDID	ITEMID
610	3
611	1
612	1
601	1
602	1
...	

64 rows selected.

Non-Equijoins

EMP

EMPNO	ENAME	SAL
7839	KING	5000
7698	BLAKE	2850
7782	CLARK	2450
7566	JONES	2975
7654	MARTIN	1250
7499	ALLEN	1600
7844	TURNER	1500
7900	JAMES	950

...
14 rows selected.

SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

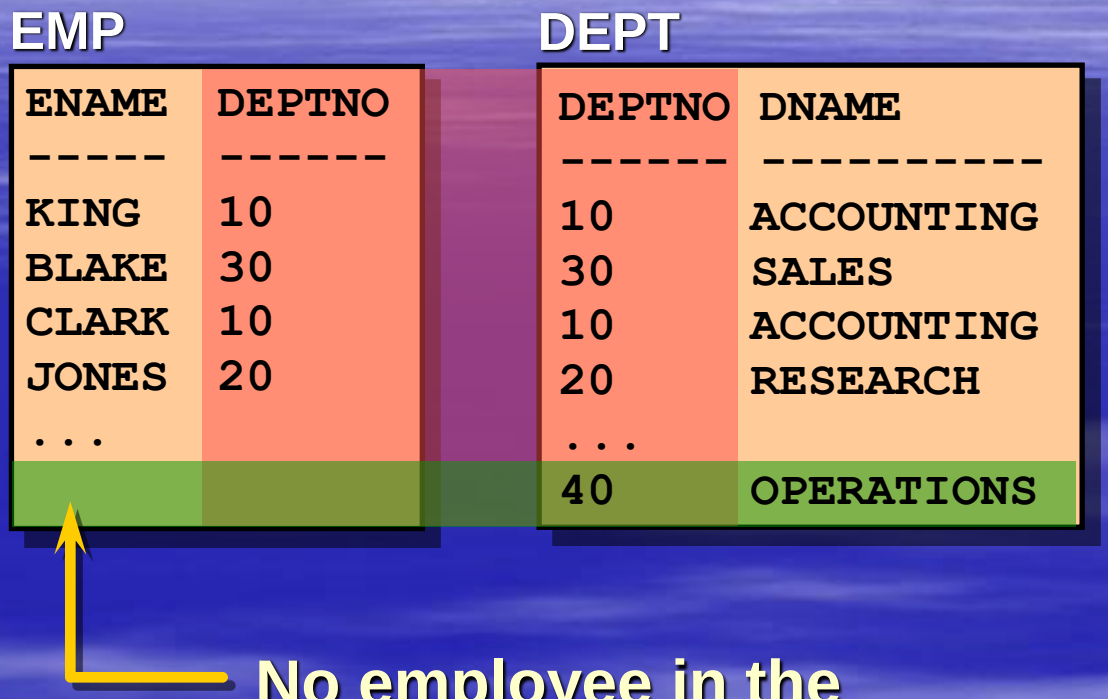
“salary in the EMP table is between low salary and high salary in the SALGRADE table”

Retrieving Records with Non-Equijoins

```
SQL>   SELECT    e.ename, e.sal, s.grade
  2    FROM      emp e, salgrade s
  3    WHERE     e.sal
  4    BETWEEN   s.losal AND s.hisal;
```

```
ENAME             SAL      GRADE
---------- ---------- ----------
JAMES             950          1
SMITH             800          1
ADAMS            1100          1
...
14 rows selected.
```

Outer Joins


EMP
ENAME DEPTNO
KING 10
BLAKE 30
CLARK 10
JONES 20
...
DEPT
DEPTNO DNAME
10 ACCOUNTING
30 SALES
10 ACCOUNTING
20 RESEARCH
...
40 OPERATIONS
No employee in the
OPERATIONS department

Outer Joins

- You use an outer join to also see rows that do not usually meet the join condition.
- Outer join operator is the plus sign (+).

```
SELECT table1.column, table2.column
FROM   table1, table2
WHERE  table1.column(+) = table2.column;
```

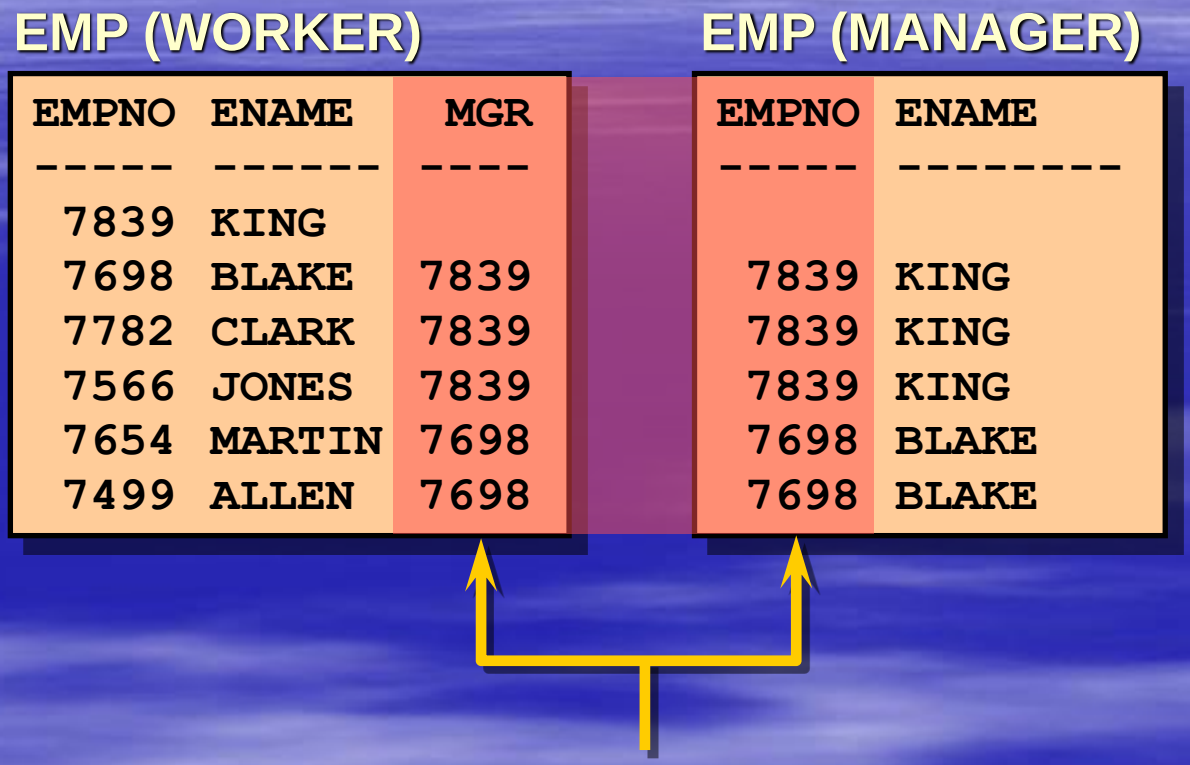
```
SELECT table1.column, table2.column
FROM   table1, table2
WHERE  table1.column = table2.column(+);
```

Using Outer Joins

```
SQL> SELECT   e.ename, d.deptno, d.dname
  2  FROM     emp e, dept d
  3  WHERE    e.deptno(+) = d.deptno
  4  ORDER BY e.deptno;
```

```
ENAME          DEPTNO DNAME
---------- --------- -------------
KING               10 ACCOUNTING
CLARK              10 ACCOUNTING
...
                   40 OPERATIONS
15 rows selected.
```

Self Joins



"MGR in the WORKER table is equal to EMPNO in the MANAGER table"

Joining a Table to Itself

```
SQL> SELECT worker.ename||' works for '||manager.ename
  2  FROM    emp worker, emp manager
  3  WHERE   worker.mgr = manager.empno;
```

```
WORKER.ENAME||'WORKSFOR'||MANAG
------------------------------
BLAKE works for KING
CLARK works for KING
JONES works for KING
MARTIN works for BLAKE
...
13 rows selected.
```

Summary

```
SELECT    table1.column, table2.column
FROM      table1, table2
WHERE     table1.column1 = table2.column2;
```

Equijoin Non-equijoin Outer join Self join

Practice Overview

- Joining tables using an equijoin
- Performing outer and self joins
- Adding conditions