

Obtaining Data from Multiple Tables

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
...		
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
30	Shipping	1900
40	IT	1600
50	Sales	2500
60	Executive	1700
70	Accounting	1700
80	Contracting	1700



EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
...		
102	90	Executive
205	110	Accounting
206	110	Accounting

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.



Cartesian Products

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.

ORACLE

Copyright © Oracle Corporation, 2001. All rights reserved.



Generating a Cartesian Product

EMPLOYEES (20 rows)

DEPARTMENTS (8 rows)

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
...		
202	Fay	20
205	Higgins	110
206	Gietz	110

20 rows selected

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
30	Shipping	1900
40	IT	1400
50	Sales	2500
60	Executive	1700
70	Accounting	1700
80	Contracting	1700

8 rows selected

Cartesian
product: →
20x8=160 rows

EMPLOYEE_ID	DEPARTMENT_ID	LOCATION_ID
100	90	1700
101	90	1700
102	90	1700
103	90	1700
104	90	1700
105	90	1700
106	90	1700
...		

160 rows selected

$A \cup B = A \cup B$

Types of Joins

Oracle Proprietary Joins (8i and prior):

Equijoin

Non-equijoin

Outer join

Self join

SQL: 1999 Compliant Joins:

Cross joins

Natural joins

Using clause

Full or two sided outer
joins

Arbitrary join conditions
for outer joins

ORACLE

Joining Tables Using Oracle Syntax

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.

ORACLE



What is an Equijoin?

EMPLOYEES

EMPLOYEE_ID	DEPARTMENT_ID
100	10
101	10
102	10
103	10
104	10
105	10
106	10
107	10
108	10
109	10
110	10
111	10
112	10
113	10
114	10
115	10
116	10
117	10
118	10
119	10
120	10
121	10
122	10
123	10
124	10
125	10
126	10
127	10
128	10
129	10
130	10
131	10
132	10
133	10
134	10
135	10
136	10
137	10
138	10
139	10
140	10
141	10
142	10
143	10
144	10
145	10
146	10
147	10
148	10
149	10
150	10
151	10
152	10
153	10
154	10
155	10
156	10
157	10
158	10
159	10
160	10
161	10
162	10
163	10
164	10
165	10
166	10
167	10
168	10
169	10
170	10
171	10
172	10
173	10
174	10
175	10
176	10
177	10
178	10
179	10
180	10
181	10
182	10
183	10
184	10
185	10
186	10
187	10
188	10
189	10
190	10
191	10
192	10
193	10
194	10
195	10
196	10
197	10
198	10
199	10

PK

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
30	Sales and Operations
40	Human Resources
50	Finance
60	IT
70	Legal
80	Compliance
90	Security
100	Facilities
110	Procurement
120	Quality Assurance
130	Research and Development
140	Product Development
150	Customer Support
160	Operations
170	Manufacturing
180	Logistics
190	Supply Chain
200	Inventory Management
210	Warehouse Management
220	Transportation
230	Shipping
240	Receiving
250	Inventory Control
260	Inventory Tracking
270	Inventory Auditing
280	Inventory Reconciliation
290	Inventory Reporting
300	Inventory Analysis
310	Inventory Forecasting
320	Inventory Optimization
330	Inventory Improvement
340	Inventory Innovation
350	Inventory Integration
360	Inventory Interoperability
370	Inventory Interconnectivity
380	Inventory Interoperability
390	Inventory Interoperability
400	Inventory Interoperability
410	Inventory Interoperability
420	Inventory Interoperability
430	Inventory Interoperability
440	Inventory Interoperability
450	Inventory Interoperability
460	Inventory Interoperability
470	Inventory Interoperability
480	Inventory Interoperability
490	Inventory Interoperability
500	Inventory Interoperability
510	Inventory Interoperability
520	Inventory Interoperability
530	Inventory Interoperability
540	Inventory Interoperability
550	Inventory Interoperability
560	Inventory Interoperability
570	Inventory Interoperability
580	Inventory Interoperability
590	Inventory Interoperability
600	Inventory Interoperability
610	Inventory Interoperability
620	Inventory Interoperability
630	Inventory Interoperability
640	Inventory Interoperability
650	Inventory Interoperability
660	Inventory Interoperability
670	Inventory Interoperability
680	Inventory Interoperability
690	Inventory Interoperability
700	Inventory Interoperability
710	Inventory Interoperability
720	Inventory Interoperability
730	Inventory Interoperability
740	Inventory Interoperability
750	Inventory Interoperability
760	Inventory Interoperability
770	Inventory Interoperability
780	Inventory Interoperability
790	Inventory Interoperability
800	Inventory Interoperability
810	Inventory Interoperability
820	Inventory Interoperability
830	Inventory Interoperability
840	Inventory Interoperability
850	Inventory Interoperability
860	Inventory Interoperability
870	Inventory Interoperability
880	Inventory Interoperability
890	Inventory Interoperability
900	Inventory Interoperability
910	Inventory Interoperability
920	Inventory Interoperability
930	Inventory Interoperability
940	Inventory Interoperability
950	Inventory Interoperability
960	Inventory Interoperability
970	Inventory Interoperability
980	Inventory Interoperability
990	Inventory Interoperability

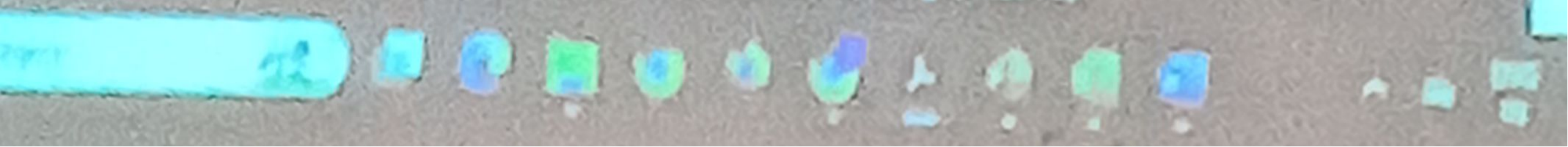
PK

Foreign key

Primary key

ORACLE

Copyright © Oracle Corporation, 2011. All rights reserved.



Retrieving Records with Equijoins

```
SELECT employees.employee_id, employees.last_name,  
       employees.department_id, departments.department_id,  
       departments.location_id  
FROM   employees, departments  
WHERE  employees.department_id = departments.department_id
```

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID	LOCATION_ID
10	King	90	90	1100
20	DeMott	90	90	1100
30	Fay	90	90	1100
40	Murphy	90	90	1100
50	Patel	90	90	1100
60	Chen	90	90	1100
70	Martin	90	90	1100
80	Tracy	90	90	1100

19 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.

ORACLE

Using Table Aliases

Simplify queries by using table aliases.

Improve performance by using table prefixes.

```
SELECT [e].employee_id, [e].last_name, [e].department_id,  
       [d].department_id, [d].location_id  
FROM   employees[e], departments[d]  
WHERE  [e].department_id = [d].department_id;
```