

Database Programming with SQL

* 1. : Oracle Equijoin and Cartesian Product Practice Activities

# Objectives

* + - Name the Oracle proprietary joins and their ANSI/ISO SQL: 1999 counterparts
    - Describe the purpose of join conditions
    - Construct and execute a SELECT statement that results in a Cartesian product
    - Construct and execute SELECT statements to access data from more than one table using an equijoin
    - Construct and execute SELECT statements that add search conditions using the AND operator
    - Apply the rule for using column aliases in a join statement

# Vocabulary

Identify the vocabulary word for each definition below.

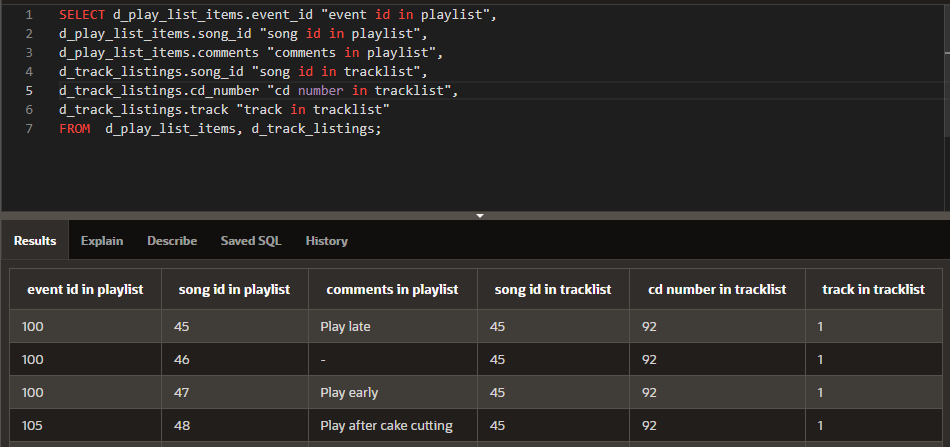
|  |  |
| --- | --- |
| **Cartesian Product** | Results from an invalid or omitted join condition; all combinations of rows are displayed |
| **equijoin** | Values in a column in one table are equal to a value in another table; also called an inner join or simple join |
| **proprietary join** | Connection command exclusive to a specific company |
| **alias** | Gives a table another name to simplify queries and improve performance |
| **join conditions** | Display data from two or more related tables |

# Try It / Solve It

1. Create a Cartesian product that displays the columns in the d\_play\_list\_items and the d\_track\_listings in the DJs on Demand database.

**SELECT d\_play\_list\_items.event\_id "event id in playlist", d\_play\_list\_items.song\_id "song id in playlist", d\_play\_list\_items.comments "comments in playlist", d\_track\_listings.song\_id "song id in tracklist", d\_track\_listings.cd\_number "cd number in tracklist",  d\_track\_listings.track "track in tracklist"**

**FROM  d\_play\_list\_items, d\_track\_listings;**



1. Correct the Cartesian product produced in question 1 by creating an equijoin using a common column.

SELECT d\_play\_list\_items.event\_id "event id in playlist",

d\_play\_list\_items.song\_id "song id in playlist",

d\_play\_list\_items.comments "comments in playlist",

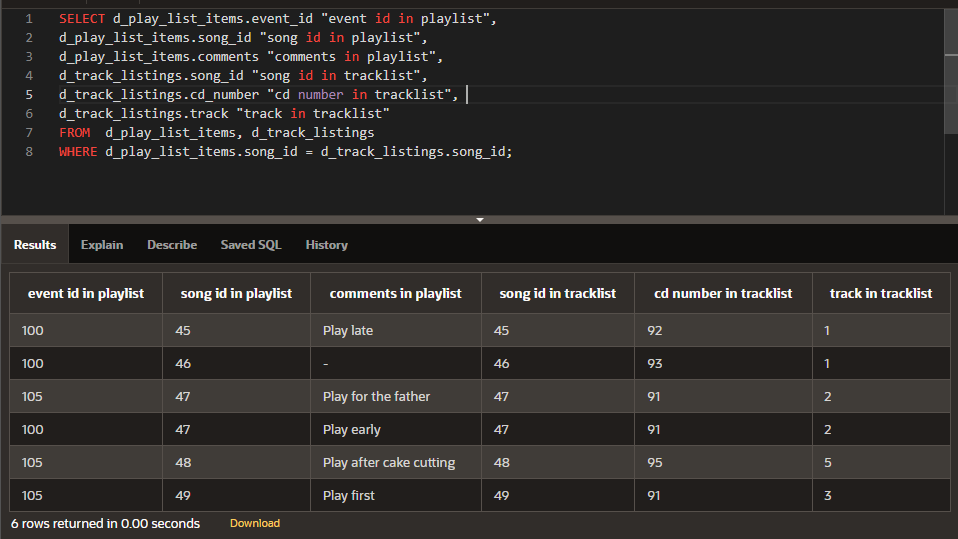
d\_track\_listings.song\_id "song id in tracklist",

d\_track\_listings.cd\_number "cd number in tracklist",

d\_track\_listings.track "track in tracklist"

FROM d\_play\_list\_items, d\_track\_listings

WHERE d\_play\_list\_items.song\_id = d\_track\_listings.song\_id;

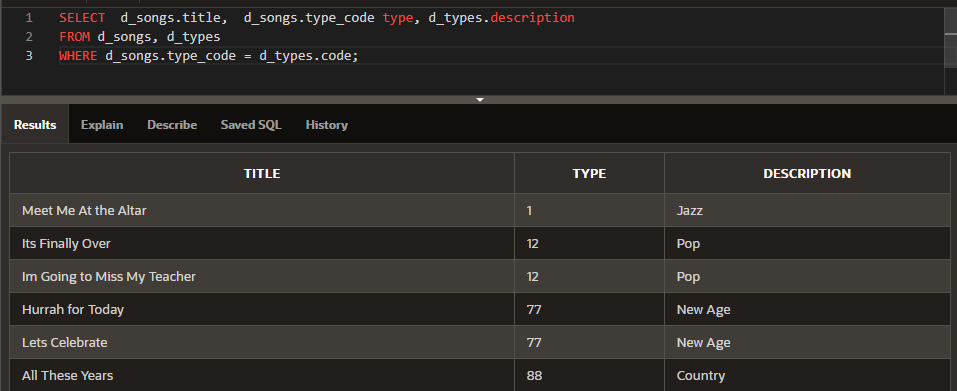


1. Write a query to display the title, type, description, and artist from the DJs on Demand database.

**SELECT  d\_songs.title,  d\_songs.type\_code type, d\_types.description**

**FROM d\_songs, d\_types**

**WHERE d\_songs.type\_code = d\_types.code;**

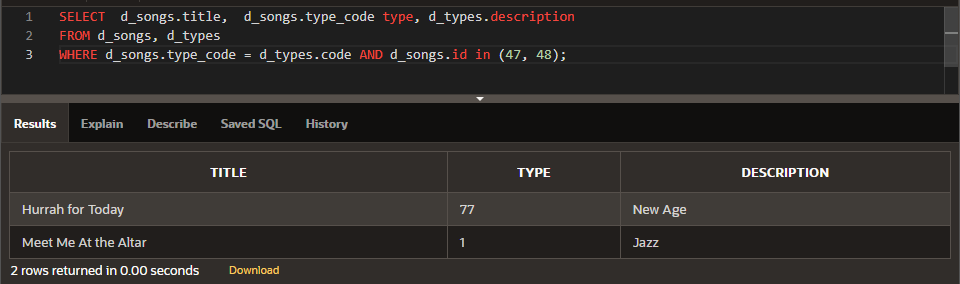


1. Rewrite the query in question 3 to select only those titles with an ID of 47 or 48.

**SELECT  d\_songs.title,  d\_songs.type\_code type, d\_types.description**

**FROM d\_songs, d\_types**

**WHERE d\_songs.type\_code = d\_types.code AND d\_songs.id in (47, 48);**



1. Write a query that extracts information from three tables in the DJs on Demand database, the d\_clients table, the d\_events table, and the d\_job\_assignments table.

SELECT d\_clients.email AS "d\_clients - email" ,

d\_clients.phone AS "d\_clients - phone" ,

d\_clients.last\_name AS "d\_clients - last\_name",

d\_clients.first\_name AS "d\_clients - first\_name",

d\_clients.client\_number AS "d\_clients - client\_number",

d\_events.id AS "d\_events - id", d\_events.name AS "d\_events - name",

d\_events.event\_date AS "d\_events - event\_date", d\_events.description AS "d\_events - description",

d\_events.cost AS "d\_events - cost", d\_events.venue\_id AS "d\_events - venue\_id",

d\_events.package\_code AS "d\_events - package\_code",

d\_events.theme\_code AS "d\_events - theme\_code", d\_events.client\_number AS "d\_events - client\_number",

d\_job\_assignments.partner\_id AS "d\_job\_assignments - partner\_id",

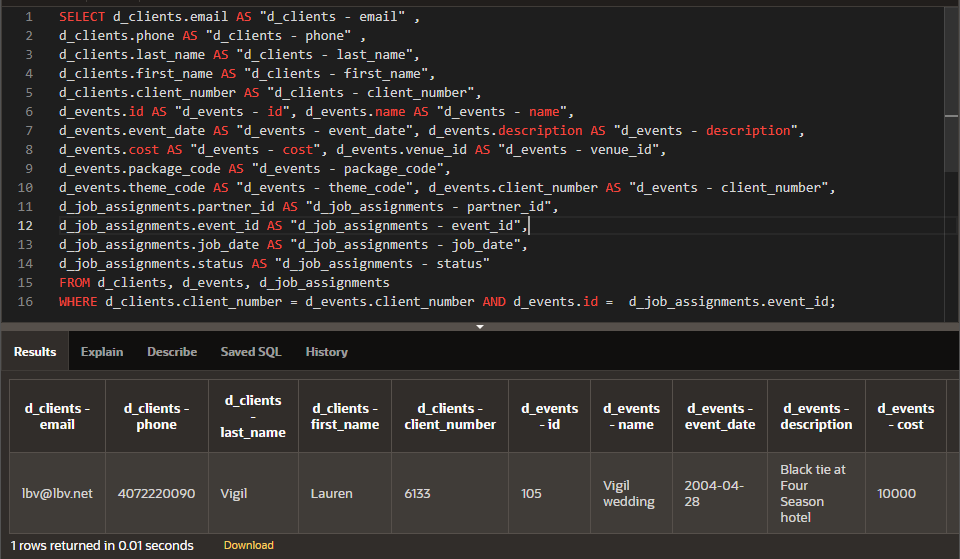
d\_job\_assignments.event\_id AS "d\_job\_assignments - event\_id",

d\_job\_assignments.job\_date AS "d\_job\_assignments - job\_date",

d\_job\_assignments.status AS "d\_job\_assignments - status"

FROM d\_clients, d\_events, d\_job\_assignments

WHERE d\_clients.client\_number = d\_events.client\_number AND d\_events.id = d\_job\_assignments.event\_id;

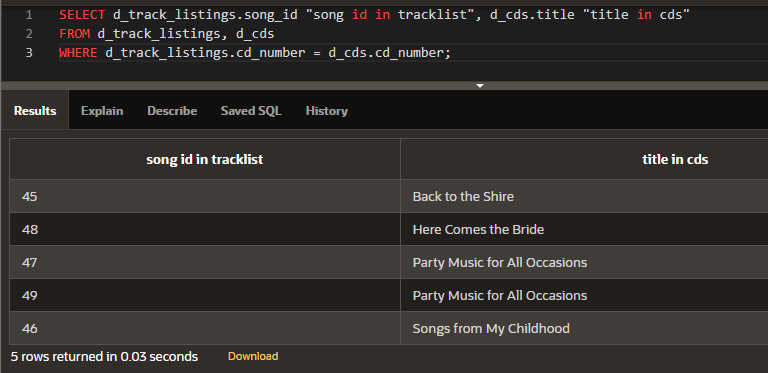


1. Create and execute an equijoin between DJs on Demand tables d\_track\_listings and d\_cds. Return the song\_id and the title only.

**SELECT d\_track\_listings.song\_id "song id in tracklist", d\_cds.title "title in cds"**

**FROM d\_track\_listings, d\_cds**

**WHERE d\_track\_listings.cd\_number = d\_cds.cd\_number;**



1. Mark T for the statements that are true and F for the statements that are false.

a. A join is a type of query that gets data from more than one table based on columns with the same name. FALSE

b. To join tables using an equijoin, there must be a common column in both tables and that column is usually a primary key in one of the tables. TRUE

c. A Cartesian product occurs because the query does not specify a WHERE clause. TRUE

d. Table aliases are required to create a join condition. FALSE

e. If a table alias is used for a table name in the FROM clause, it must be substituted for the table name throughout the SELECT statement. TRUE

f. Table alias must be only one character in length. FALSE

g. A simple join or inner join is the same as an equijoin. TRUE

1. What advantage does being able to combine data from multiple tables have for a business?

Объединение данных позволяет более наглядно отображать данные из разных таблиц связанные по какому либо правилу (полю).

Copyright © 2020, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.