

Kirk's Korner



Quick & Simple Tips

Kirk Paul Lafler, Software Intelligence Corporation

Cartesian Product Joins

By Kirk Paul Lafler, Software Intelligence Corporation

When two or more tables are specified in the FROM clause of a SELECT query without a corresponding WHERE clause expression, a special type of join is created. The Cartesian product (or Cross join) represents all possible combinations of rows and columns from the joined tables. To be exact, Cartesian product joins represent the sum of the number of columns of the input tables plus the product of the number of rows of the input tables. Essentially the Cartesian product contains $m * n$ rows, where m is the number of rows in the first table and n is the number of rows in the second table. Put another way, it represents each row from the first table matched with each possible row from the second table, and so on and so forth. All other types of joins are classified as subsets of Cartesian products essentially being created by deriving the Cartesian product and then excluding rows that fail the specified WHERE clause expression.

Although the Cartesian product serves a very useful purpose in the relational model, it is essentially meaningless for a user to intentionally produce it as a final table. Besides being on the large size – the results contain way too much information making it difficult, if not impossible, for the practitioner to select what is interesting.

To illustrate this point, the example displayed below illustrates a SELECT query in a two-table join without the specification of a WHERE clause. The result of such a complex query is a Cartesian product. The results consist of the PRODNAME, PRODCOST, and MANUNAME columns from the first row in the PRODUCTS table combined with the MANUNUM column from the first row in the MANUFACTURERS table. Next the PRODNAME, PRODCOST, and MANUNAME columns from the second row in the PRODUCTS table are combined with the MANUNUM column from the first row in the MANUFACTURERS table, and so on, until each row in the PRODUCTS table are combined with the first MANUFACTURERS row. Then the results contain each row from the PRODUCTS table matched with the MANUNUM column from the second row in the MANUFACTURERS table, and so on, until all possible combinations of selected columns from all rows in the two tables are joined. It quickly becomes obvious that Cartesian product joins can become huge and are something SAS users should avoid.

SQL Code

```
PROC SQL;  
  SELECT prodname, prodcost,  
         manufacturers.manunum, manuname  
  FROM PRODUCTS, MANUFACTURERS;  
QUIT;
```

The PRODUCTS table is the first table specified in the FROM clause.

The MANUFACTURERS table is the second table specified in the FROM clause.

The specification of two tables in a FROM clause without the specification of a WHERE clause is referred to as a two-way Cartesian product join.

Contact Information

If you would like more information or have any questions about this tip, please contact: Kirk Paul Lafler, Software Intelligence Corporation at KirkLafler@cs.com. Kirk has been working with the SAS System since 1979 and is a SAS Certified Professional®. His company provides custom SAS programming, application design and development, consulting services, and hands-on SAS training to clients around the world. Kirk is the author of four books including PROC SQL: Beyond the Basics Using SAS by SAS Institute, and more than two hundred peer-reviewed articles and papers that have appeared in professional journals and SAS User Group proceedings. Kirk can be reached at:

Kirk Paul Lafler
Software Intelligence Corporation
World Headquarters
P.O. Box 1390
Spring Valley, California 91979-1390
E-mail: KirkLafler@cs.com



SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.