

Design Tip #68 Simple Drill-Across in SQL

By Warren Thornthwaite

Drill-across refers to the process of querying multiple fact tables and combining the results into a single data set. A common example involves combining forecast data with actual data. The forecast data is typically kept in a separate table, captured at a different level of detail than the actual data. When a user wants a report that compares actual and forecast by customer, the query needs to go against two fact tables. (Note: Data from the two fact tables can only be combined if they are built using conformed dimensions. The Customer, Date and any other shared dimensions must be exactly the same in both star schemas.)

The most efficient way to combine data from the two fact tables is to issue separate queries against each fact table, then combine the two results sets by matching up their shared attributes. This tends to work best because most database optimizers recognize the single star query and quickly return the two results sets.

The following SQL is used to drill-across two star schemas, Actual Sales and Forecast, both with Customer and Date dimensions. The query uses SELECT statements in the FROM clause to create two sub-queries and join their results together, exactly as we'd like. Even if you don't have to write the SQL yourself, you'll get a sense for what your BI tool might be doing.

```

SELECT Act.Customer, Act.Year, Act.Month, Actual_Amount, Forecast_Amount
FROM
-- Subquery "Act" returns Actuals
(SELECT Customer_Name AS Customer, Year, Month_Name AS Month,
  SUM(Sale_Amount) Actual_Amount
  FROM Sales_Facts A
  INNER JOIN Customer C
  ON A.Customer_Key = C.Customer_Key
  INNER JOIN Date D
  ON A.Sales_Date_Key = D.Date_Key
  GROUP BY Customer_Name, Year, Month_Name) Act
INNER JOIN
-- Subquery "Fcst" returns Forecast
(SELECT Customer_Name AS Customer, Year, Month_Name AS Month,
  SUM(Forecast_Amount) Forecast_Amount
  FROM Forecast_Facts F
  INNER JOIN Customer C
  ON F.Customer_Key = C.Customer_Key
  INNER JOIN Date D
  ON F.Sales_Date_Key = D.Date_Key
  GROUP BY Customer_Name, Year, Month_Name) Fcst
-- Join condition for our small result sets
ON Act.Customer = Fcst.Customer
AND Act.Year = Fcst.Year
AND Act.Month = Fcst.Month

```

This should perform almost as fast as doing the two individual queries against the separate fact

tables because the join is on relatively small subset of data that's already in memory. The results should look like this:

Act sub-query:

Customer	Year	Month	Actual_Amount
Big Box	2005	May	472,394
Small Can	2005	May	1,312,034

Fcst sub-query:

Customer	Year	Month	Forecast_Amount
Big Box	2005	May	435,000
Small Can	2005	May	1,257,000

Final drill-across query results:

Customer	Year	Month	Actual_Amount	Forecast_Amount
Big Box	2005	May	472,394	435,000
Small Can	2005	May	1,312,034	1,257,000

For relational-based star schemas, many front-end BI tools can be configured to issue the separate SQL queries through their metadata, or at least in their user interface. Many OLAP engines do this through a "virtual cube" concept that ties the two underlying cubes together based on their shared dimensions.

Remember, if you do not have rigorously enforced conformed dimensions, you may not be comparing apples to apples when you drill-across!