



Kimball Design Tip #23: A Rolling Prediction Of The Future, Now And In The Past

By Ralph Kimball

>>>> Original message from Richard to Ralph describing his problem <<<<

Hi Ralph & Co,

Help! I'm really stuck but it is an interesting design question for anyone who feels up to the challenge:

How do I predict the future? I need a nice simple (ha ha) fact table design...

I have an 'Account' dimension table in a data mart that has a 'Status' field, let's say this status is "OVERDUE". I also know the 'Status_Effective_Date', lets say this is "May-2nd-2001".

The business user wants to know the following : FOR THE NEXT MONTH, HOW MANY DAYS WILL THE ACCOUNT HAVE BEEN AT THE STATUS OF 'OVERDUE'?

If TODAY is "May-5th-2001", the user wants to see the following:

5th May: 4 days
6th May: 5 days
7th May: 6 days
etc... up to 4th-Jun-2001 - This is a rolling month in the future starting TODAY.

They then want to count all the accounts in the account table and group them by day bandings at status 'OVERDUE' to see the following:

5th May: 100 accounts - 4 to 6 days at 'OVERDUE'
6th May: 78 accounts - 4 to 6 days at 'OVERDUE'
etc

5th May: 200 account - 7 to 9 days at 'OVERDUE'
6th May: 245 account - 7 to 9 days at 'OVERDUE'
etc

I also need to travel back in time and do the same for any point in time + 1 month, however many accounts will have the 'Status_Ineffective_Date' then set and will no longer be at Status 'OVERDUE', but they will have a historical record in the account table to show when they were at this status via an SCD Type 2.

If anyone has similar experience with such a problem or know a design that could supports this, please let me know.... my brain hurts.

Regards,
Richard.

>>>> Ralph's Response to Richard <<<<

Hi Richard,

well I took a quick look at this. Maybe the following will work.

Assume your account dimension table has the following fields (as you described):

```
ACCOUNT_KEY
STATUS
STATUS_EFFECTIVE_DT
```

Now build a another table with the following fields

```
STATUS_EFFECTIVE_DT
STATUS_REPORTING_DT
DELTA
```

where DELTA is just the number of days between the effective date and the reporting date. You need a record in this table for every combination of EFFECTIVE date and REPORTING date your users could possibly be interested in. If you have EFFECTIVE dates going back one year (365 dates) and you want to report forward one year then you would need about 365*365 rows in this table, or about 133,000 rows. You may have other assumptions. Rather large but maybe workable if you have lots of RAM.

Anyway, join this second table to the first on the STATUS_EFFECTIVE_DT.

Then your first query should be satisfied with

```
SELECT STATUS_REPORTING_DT, DELTA
FROM .
WHERE STATUS = 'OVERDUE'
AND STATUS_REPORTING_DT BETWEEN 'May 5, 2001' and 'June 4, 2001'
ORDER BY STATUS_REPORTING_DT
```

To get your banding report, maybe you can build a third table (a banding table) with fields

```
BAND_NAME
UPPER_DELTA
LOWER_DELTA
```

You join this table to the second where

```
DELTA <= UPPER_DELTA
DELTA > LOWER_DELTA
```

Your SQL is something like

```
SELECT BAND_NAME, COUNT(*)
FROM (all three tables joined as described)
WHERE STATUS_REPORTING_DT BETWEEN 'May 5, 2001' and 'June 4, 2001'
AND DELTA <= UPPER_DELTA
AND DELTA > LOWER_DELTA
ORDER BY UPPER_DELTA
GROUP BY BAND_NAME
```

I describe this value banding approach in the Lifecycle Toolkit on pages 251-252. Try this on a small example in Access. Let me know how it turns out.

Good luck,

Ralph Kimball

>>>> Richard's Follow-up Response After He Tried The Solution <<<<

Hi Ralph,

I'm very pleased to say that your recommendations worked! - Thank you. I tried it out on a few sample rows in SQL Server 7 and the results look good. I've now taken it to the next level to include a Status_Ineffective_Date so we can exclude accounts that stopped the 'OVERDUE' status somewhere within the reporting period. Maybe you would like to use this is one of your Design Tip mails?

Firstly, I slightly modified your SQL to the following:

```
SELECT
  COUNT(account.Account_Id),
  days_overdue.reporting_date
FROM
  account,
  days_overdue
WHERE
  ( account.Status_Effective_Date=days_overdue.effective_date )
  AND (
    days_overdue.reporting_date BETWEEN '05/05/2001' AND '06/04/2001'
    AND days_overdue.delta >= 10 {an arbitrary number or banding that the user
    decides at run time}
  )
GROUP BY
  days_overdue.reporting_date
```

The key to the above SQL is to constrain on 'delta' (which we call 'days_overdue'), or else the query pulls back every account for every day that the reporting period is for, and the COUNT is always the same number. The constraint can be =, <, > or BETWEEN, but must be present.

Secondly, to exclude accounts that are no longer overdue (we assume we know the date they will pay up or have paid up!) we simply include a status_ineffective_date in the account table and can do the following:

```
SELECT
  COUNT(account.Account_Id),
  days_overdue.reporting_date
FROM
  account,
  days_overdue
WHERE
  ( account.Status_Effective_Date=days_overdue.effective_date )
  AND (
    days_overdue.reporting_date BETWEEN '05/05/2001' AND '06/04/2001'
    AND days_overdue.delta >= 10
```

```
AND account.Status_Ineffective_Date > days_overdue.reporting_date
)
GROUP BY
days_overdue.reporting_date
```

You would of course have to set the ineffective_date for all currently overdue accounts to something in the future like 01/01/3000. This also fits perfectly with the SCD type 2 where we are creating a new row in the account dimension each time the status changes, to give us a complete historical comparison!!

Thank you very much for your help and I wish you all the best,
Richard Tomlinson.

>>>> Final Postscript <<<<

This was a fun little puzzle. It's great to receive these but if you send me a request, PLEASE be understanding! I certainly can't deal with all of them, and in some cases the results won't be so serendipitous.