Creating a Simple "Information Database" with DB2® Universal Database™

By Jackie Jansen

Every iSeries™ server customer that I know has users running some sort of queries and reports with varying degrees of success. Whether users are still using green screen Query/400™ (cringe) or one of the many ODBC query products on the market, most companies require users to be able to access their data.

A few years ago Forrester Research did a study of IT departments and found that 70% of the requests that come into IT are requests for data that is already in the database. In other words, these requests aren't for new leading-edge applications, they are for things like new columns on a report, a new sorting sequence, or new reports against existing data. Forrester continued to say that only about 20% of production data was accessible by the end users that need it.

In this article, I will not delve into traditional data warehouse design, star schema models, snowflake models, or any schemas that require specific database training and education. Instead, I will help you quickly design a database that will allow your business users to get simple, fast access to the data that they need to do their jobs. I am purposely not calling this database a data warehouse or even a formal data mart. It is a quick and easy information database for end-user access. If you have long-term requirements for better reporting systems, then you should look at an architected database design. You should also investigate different types of reporting products on the market. Multi-dimensional database software such as the IBM® DB2 OLAP Server™ for iSeries is ideally suited for summary-level analysis.

Tables

Creating tables for an "information database" is something that a lot of programmers find inherently difficult to do. I define an "information database" as a set of tables that are designed specifically for end-user querying and reporting purposes. Typically these tables are stored in one or more libraries created just for these purposes. Many of you have taken database training that covered normalizing files, removing redundant information, and making sure that any value is stored once and only once.

The design requirements for an information database may seem like a move in the wrong direction for many programmers; however, everything that you have been taught about designing tables for a transaction environment still holds true. The following discussion applies to users executing IT developed queries as well as users writing their own ad hoc queries.

There are a few things that you should keep in mind when you are designing tables for business users to query. Typical users are not, and never want to be, programmers. That isn't their job or their expertise. It is your job in IT to make things fast and easy for your customers – the end users. An information database has two main purposes: to simplify tasks for the end user, and to make queries perform faster.

The main differences between the tables in an information database and your production tables are as follows. For those of you who haven't started using long field names in your production files, you should do so in your information database. Codes or special values that are stored in the production database need to be translated for the information database. For example, if "1" means married and "2" means single, then put "Married" and "Single" in the information database.

IBM Global Services

SecureWorld: The IBM End-to-End Security Conference
September 23-27, 2002
Hilton Hotel, Walt Disney Resort, Orlando, Florida
http://www-3.ibm.com/services/learning/conf/us/sw/index.html

WebSphere Technical Exchange
November 11-15, 2002
Ceasars Palace, Las Vegas, Nevada
http://www-3.ibm.com/services/learning/conf/us/websphere/index.html

Files should be denormalized and multiple tables should be joined together to make it easier for end users to navigate and find the data that they need. Additionally, many of the measures or values that users calculate directly in their queries should be precalculated and stored in these new tables. Some tables will contain detail information while others will contain summarized or "roll up" values, for example, store sales data at the day level. You may need to create a summary table at the month level for many of your users to query. Similarly, data may be stored at a store or sales rep level and you may need to create a summary table with this same data rolled up to both a region and country level. In an information database, data will be replicated in multiple tables. If "item_number" appears in more than one table then you probably want the item description in each of these tables also. Tables in your information database will normally be read-only and not update capable. This eliminates the concern of keeping duplicated data in sync.

Indexes

In addition to creating these new tables you will also need to heavily index them based on your best estimate of end-user query requirements. These indexes can be monitored and evaluated using a tool like Operations Navigator. Aside from the benefits to the end user, creating an information database simplifies security. It separates the users from the production data and allows you to control exactly what is put in the information database for querying purposes.

Let's look at these differences in more detail. There are multiple ways to create a table on the iSeries. If you are using data description specifications (DDS) then you can use the "ALIAS" keyword to give your field a meaningful name that will be used by ODBC query tools. If you are using SQL, then the column name itself can be your long name. If you are using Operations Navigator, then the column name is your long name and the system name is the shorter name for RPG coding purposes. Make these names meaningful for the business users.

As an aside, I have found it better to use underscore characters in long names instead of blanks. There are a few query products on the market that have problems with blanks in a field name. Therefore, for example, I use "Customer_Number" instead of "Customer Number" as a field name.

Translation

Translating codes can save users a lot of time and effort when making queries or reports. Standardizing the values in fields can also help. If your table has a field that contains the province name, for example, you might want to standardize things and have all values for Ontario show up as "Ontario" in the information database. Depending on how the data was entered this may or may not be a problem. If the data was entered simply as an address for mailing you might find values like "ON", "Ont", "Ont." Think about what you might find for Saskatchewan!

Our post office can handle any of these different values but a simple query select statement cannot. Typically you do not want to go back and touch your production programs. This can cause far too many problems and delays. A simple solution is to create a table that contains the values from your production system and a mapping to the new values for your information database (see Table 1). In this example the translate table would have three records for Ontario. Next month this might be four records if we find that a data-entry clerk has mistakenly entered a new customer with "Ontar" as the province name.

From	To
Ont	Ontario
ON	Ontario
Ont.	Ontario
Ontar	Ontario

Table 1: Translation table

DOMINATE

ProData's Tools are the Dominant Leader in iSeries Utilities.

DBU, the "original" iSeries AS/400 database utility, allows users to view and update any file instantly without time consuming queries, DFU or programming. Now available in multiple power packed interfaces!

• Graphical User Interface • Green-Screen • Operations Navigator

Ask about DBU pricing AT PAR

(for a limited time only)

CvtRPGIV Move to ILE! Program templates and RPGIV book included. All for $995!

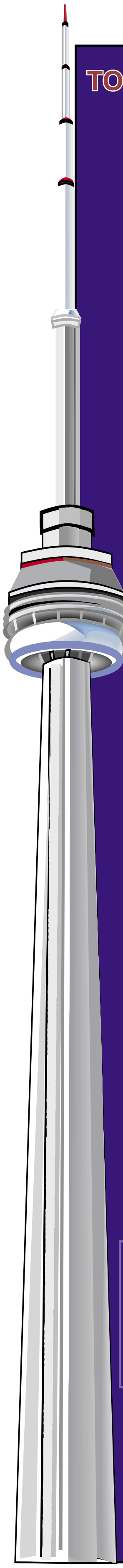
ProTools Zip/unzip files, email direct from your iSeries, and more...with the source code included. Priced at only $495

SQL/Pro to query and format reports.

RDR to retrieve deleted records.

ProData

FREE TRIALS 800.228.6318 WWW.DODBU.COM



Joins

Most users have no idea what a simple join is, never mind an inner or outer join. If users require data in one query from multiple production files you should seriously evaluate creating a new table that contains the joined information. Users get disheartened very quickly when they have to try and navigate their way through multiple cryptic tables to find the information that they want. The new tables that you create may easily have redundant data in them. As described earlier, if a table contains an item number it should probably also contain the item description for the user. You may have different tables with very similar data for different functions. Finance and Sales may each want customer and product information but they may want the data summarized differently or captured for different time periods. In this case you should have a sales table for the finance department and a separate one for the sales department.

Calculated Values

If there are calculated values that users require in their reports, then where possible, these values should be precalculated and stored in the information database. As a simple example, you could include calculated net sales or profit dollars by customer or by product.

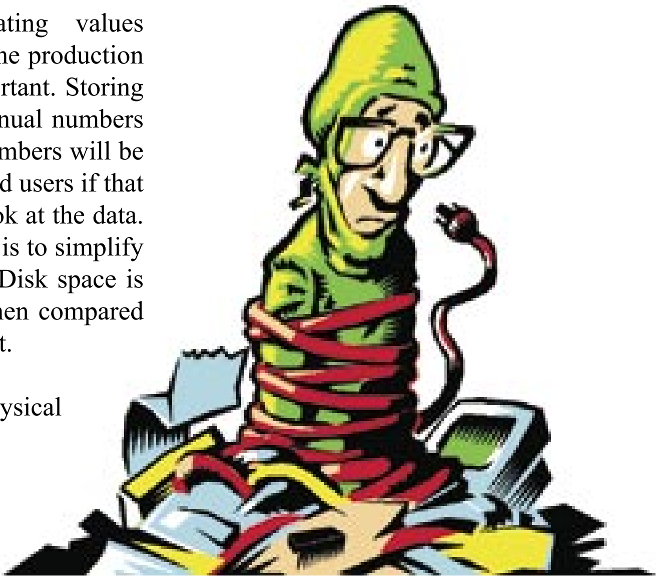
Summaries

Creating summary tables will be a mainstay of this information database. Rolling up or aggregating values stored at a detail level in the production database is also very important. Storing monthly, quarterly, and annual numbers instead of just the daily numbers will be extremely helpful to the end users if that is the way they want to look at the data. Remember, the intent here is to simplify life for the business user. Disk space is inexpensive these days when compared to the user’s time and effort.

Once you have built the physical tables for your information database, you will need to consider your indexes. Dynamic index maintenance is not a concern since these files are read-only and the index will typically not be updated during the day. If you haven’t already started using Encoded Vector Indexes (EVI’s), now is your opportunity to do so. While EVIs won’t speed up all queries, the queries that they are appropriate for will get a tremendous performance boost. You must explicitly tell the iSeries if you want an index created as an EVI. Learn how to use the new database functionality in Operations Navigator. It can help tremendously when you are setting up your indexes and testing new query performance.

Simplification

There is one additional point that I would like to make and this is one that is often overlooked. Users have been heard to complain about how hard some of the PC query tools are to use. From an IT viewpoint, these tools are so much simpler than what we had in the past, yet we often overlook the business users capabilities and perspective. A major benefit of creating a single table for a specific query is that this will simplify the use and perception of the query tool. Let me explain what I mean. With most query tools it is quite simple, once you have chosen a table, to choose the fields you want to see, the sequence you want the fields to be on the report, and the sorted sequence that you want the records in. What gets much more complex for business users is when they have to start joining tables and specifying



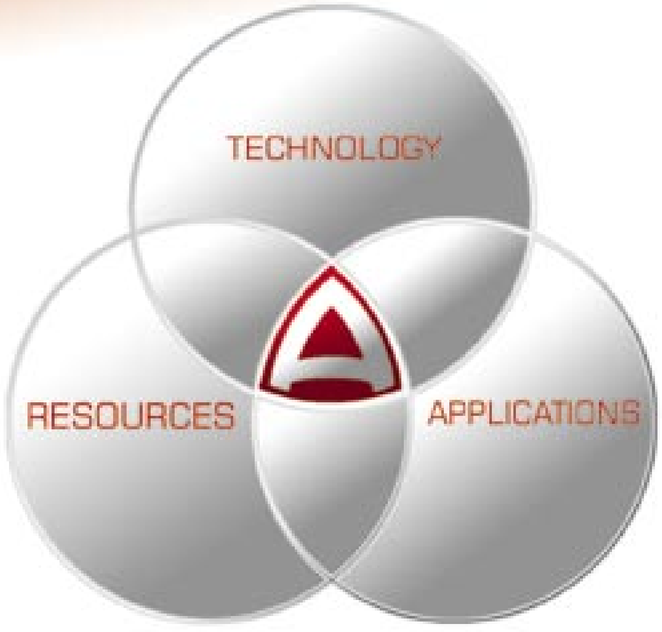
DRIVING YOUR INFORMATION TECHNOLOGY SOLUTIONS

ABLE-ONE *delivers successful . . .*

Technology
IBM @server, Storage, Printing Systems, Barcode & Wireless Networking, & more.

Resources
Project Management, Business Continuity Planning, Solution Design, Systems Consolidation, & more.

Applications
Security, CRM, B.I., ERP, e-Mail, Collaboration, Workflow, Financial & HR, Manufacturing, & more.



"Dedicated to excellence in delivering service and solutions"

IBM "Champion Award" four consecutive years.

Call ABLE-ONE To Drive Your Solutions From "Need" to "Succeed"!

ABLE-ONE
SYSTEMS INC.

TEL: 800-461-2253
SWO: 519-570-9100
www.ableone.com

IBM Business Partner

Lotus. Business Partner ™

e-business

join columns, using "group by" logic and sub-totals to summarize detail data and creating new calculated fields from scratch. The more of this that IT can do ahead of time, the simpler the query tool appears to be to the end users. They become protected from the complexities of the advanced functionality of the tool. This functionality remains for the power users that need it. Another benefit is that the queries will run much faster from the users point of view. If the summarization and joining has been done during the night then the user is basically running a very fast, formatted, sorted listing of the information that is needed. I have seen queries go from hours to minutes or less using summary tables.

This is where users who run prewritten queries and don't actually run any ad hoc reports can still benefit from an information database. This technique also moves the heavy CPU query requirements outside of prime shift hours.

Table Creation

Once you have decided to create summary tables you need to decide how you are going to do this. You can write a program to create these tables, run a query yourself and output to a table or use a software tool specifically designed for this purpose. Two tools that can assist you in building your information database are IBM DB2 OLAP Builder and IBM Warehouse Manager.

Both of these tools use SQL based logic. A more sophisticated solution would be Rodin, from Coglin Mills. This product is marketed in Canada by Global Software Solutions. These are some of the most common products that I see being used for this purpose with iSeries server customers.

Most high availability software packages can also build an information database but they may be an expensive choice if you don't already own or require the additional high availability capabilities.

Conclusion

The point of this entire article is to change your way of thinking when you are designing tables for query purposes. Many of the issues that you have when designing a traditional transaction oriented database are different from the concerns that you have when business users want to access data on an ad hoc basis. Design your new tables around the subject matter that the users are querying and not around an Order Entry type program. On a final note, I want to reiterate that the techniques described here are intended to quickly give your business users better access to their data. If you are planning on implementing a fully architected data warehouse or data mart you should be taking the appropriate education to design these systems. Good luck, and don't hesitate to contact me if I can be of further assistance. TUG

"This part of the test tells us whether you're personally suited to the job of network administrator."

Jacqueline Jansen
Business Intelligence Consultant
IBM Business Intelligence / CRM Solutions Centre
jjansen@ca.ibm.com