# TechTip: Converting Legacy Date Fields to DB2 Web Query Dates, Part I

Published Thursday, 29 May 2008 19:00 by MC Press On-line [Reprinted with permission from iTechnology Manager, published by MC Press, LP; http://www.mcpressonline.com.]

Written by Gene Cobb – cobbg@us.ibm.com

## Regardless of the data type and format that you use to represent dates in your DB2 for i5/OS files, there is a way to convert them.

From the time DB2 Web Query for IBM i was announced in July of 2008, I have given numerous presentations, workshops, demonstrations, and Webinars on the product. Among the wide variety of questions asked during these interactive sessions, one question seems to always come up: "Can DB2 Web Query handle my legacy date fields?"

The question obviously is a good one, because date/time is a vital business dimension that can be found in most business reports. After all, how often do you create a report against data in your database without filtering, sorting, or aggregating the data against a date- or time-related field? For most reports, the date/time dimension is an integral piece of information.

When it comes to storing date values in their database files, many System i shops use legacy date data types. Legacy dates are typically defined as numeric or alphanumeric fields that contain numbers or character strings that represent the date. An example of this is a field defined as zoned decimal (8,0), which contains the value 04102008 to represent the date April 10, 2008. However, to DB2 Web Query, this field is nothing more than a decimal field; the fact that it actually stores date data is not known to the tool.

But getting back to the original question, the answer is, "Yes, DB2 Web Query can handle your legacy date fields." In this TechTip, which is the first of two parts, I will show you some of the powerful built-in date functions available with DB2 Web Query. These functions allow the report developer to convert legacy date formats to fields that DB2 Web Query recognizes as true date fields (also referred to as "smart dates"). When the tool recognizes fields as dates, it can provide some additional reporting features:

- Advanced date/time manipulation, calculations, and analysis
- Report selection parameters that can be specified by invoking JavaScript calendar widgets for a more user-friendly experience
- Date decomposition to break the date into separate fields that represent the year, quarter, month, and day

All of these things enable the report developer to deliver a report that is easy to use and to provide the report formatting and information that is required.

## Virtual Columns and Built-In Functions (BIFs) Using Developer Workbench

Legacy-date-to-smart-date conversion is done by leveraging the metadata layer of DB2 Web Query. The metadata (also referred to as "synonyms") is an abstraction layer between the database and the product that describes the data source to DB2 Web Query. This gives you the ability to customize your DB2 Web Query data source structure without changing anything in the database itself. For this customization, the DB2 Web Query Developer Workbench client (Option 3 of Licensed Program Product 5733QU2) is an invaluable tool. DB2 Web Query Developer Workbench is a Windows-based client application used for advanced synonym editing and report development. While it is not required to either create synonyms or develop reports, it does provide additional development capabilities that database administrators (DBAs) and report developers will find immensely useful. One of these advanced features is the Synonym Editor, a component that provides an easy-to-use interface for synonym customization. Some of the key features of this editor include the following:

- Ability to create and edit virtual and computed fields, filters, and business views
- Editing of column names, usage, and descriptions
- Tools for data and impact analysis
- Date decomposition

An example of the Synonym Editor interface is shown in Figure 1.

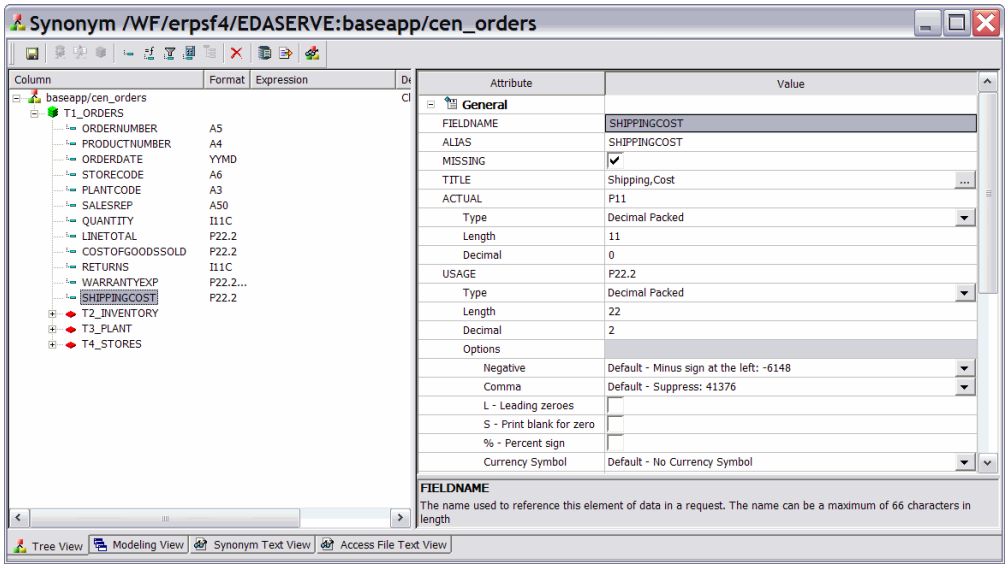


Figure 1: This is what the Developer Workbench synonym editor looks like. (Click images to enlarge.)

From a date-conversion perspective, the most useful of these synonym editor features is the ability to create virtual columns in your synonyms. Virtual columns can be defined as true DB2 Web Query date fields and can invoke the built-in conversion functions to transform the legacy data type (whichever one is used) to a date. When created at the synonym level, virtual columns are available for use in all reports and graphs that use that synonym.

Alternatively, you can use the same built-in functions (BIFs) to create defined fields at the report or graph definition level. These defined fields are created using the report development tools (Report Assistant, Graph Assistant, and Power Painter) and provide the same ability for developers to create converted, defined columns without Developer Workbench. The disadvantage of this method is that the defined fields are available for use only within that particular report. In this TechTip, I am only going to focus on creating virtual columns in synonyms, but the same techniques could be applied to defined fields in the report development tools.

## Creating Virtual Date Columns, Example 1: Packed Decimal Fields

To help you better understand the process of creating virtual columns for this purpose, consider the example order header file named ORDHDR. The DDS for this file looks like this:

```
A            R ORDHDRR
A              ORDER          7P 0B      TEXT('Order number')
A              CUST           5A  B      TEXT('Customer identifier')
A              ORDDAT         8P 0B      TEXT('Date order was entered')
A              SHPDAT         8P 0B      TEXT('Scheduled ship date')
A              SHPVIA        15A  B      TEXT('Ship via')
A              ORDSTS         1A  B      TEXT('Order status: 1=Open +
A                                        2=Closed 3=Canceled')
A              ORDAMT        11P 2B      TEXT('Order amount')
A              TOTLIN         3P 0B      TEXT('Total items in order')
A              INVNUM         7A  B      TEXT('Invoice number')
```

Because DB2 Web Query cannot query the file without a synonym, a synonym was created for ORDHDR (and was given a prefix value of "cobbg_"). The master file for this synonym is displayed in Figure 2.

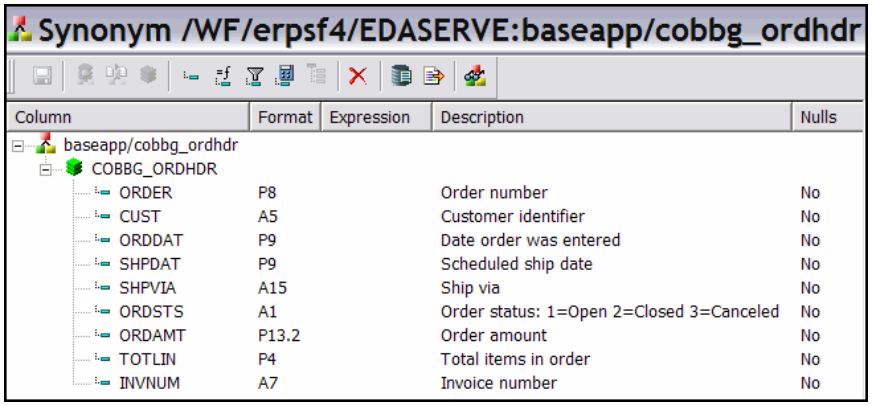


Figure 2: Here's the ORDHDR master file synonym.

Note: You may have noticed that the field data lengths between the DDS and the synonym do not always match. Don't be concerned; this is normal. DB2 Web Query will sometimes increase the synonym's usage field length to accommodate such things as the decimal symbol when displaying the data.

The example ODRHDR file has two legacy date fields (ORDAT and SHPDAT) defined as packed decimal with scale of 8 and precision of 0. By sampling the data in this file, I determined that the date format of these fields is MDYY. However, because they are packed decimal fields, DB2 Web Query does not recognize them as dates.

To remedy this, a new virtual column must be defined (in the synonym) for both fields. The new two columns will be defined as date types and will be based on an expression that contains a BIF to convert the original legacy fields from packed decimal to dates. To do this take, the following steps:

1. Open DB2 Web Query Developer Workbench.
2. Under the system name, select Data Servers > EDASERV > Applications > baseapp.
3. Find the synonym master file (.mas extension) of the ORDHDR file and from the right-mouse click menu, select Edit in Synonym Editor as shown in Figure 3.

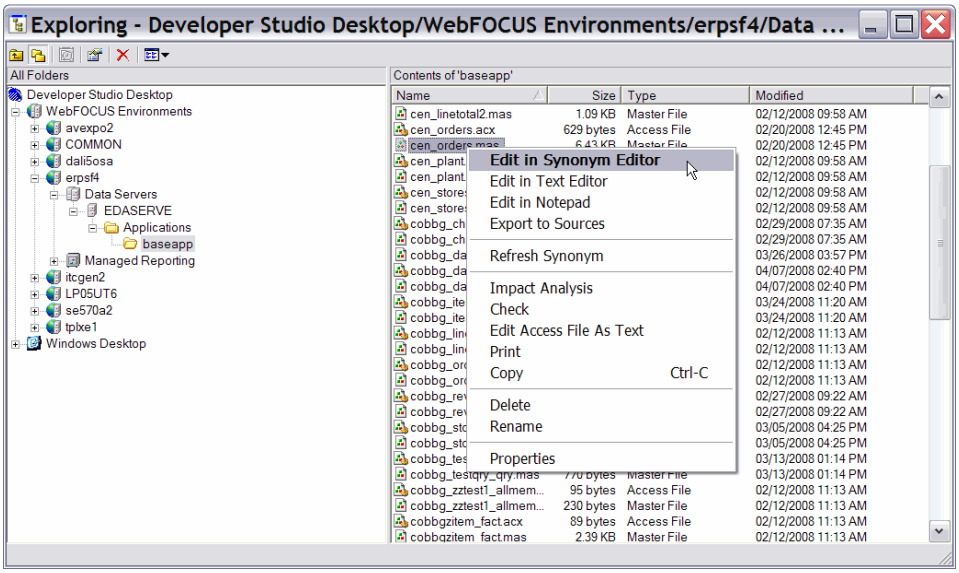


Figure 3: Select Edit in the Synonym Editor.

The master file is displayed in the Synonym Editor window.

4. Create a new virtual column by clicking on the icon shown in Figure 4.

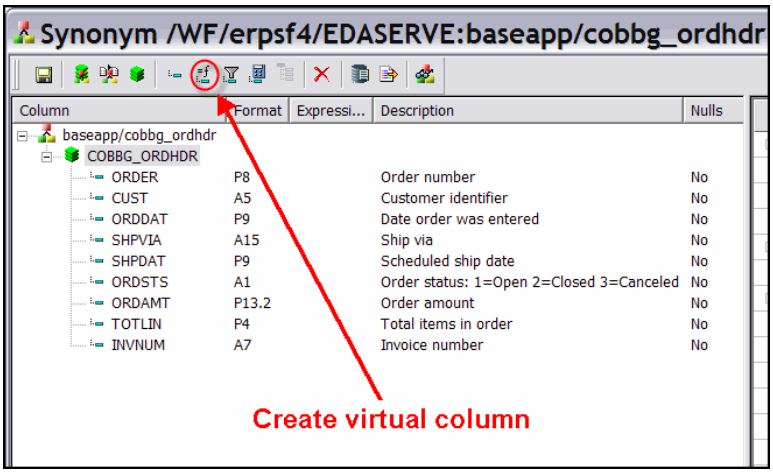


Figure 4: Create a new virtual column.

The Virtual Column Calculator window is displayed as shown in Figure 5.

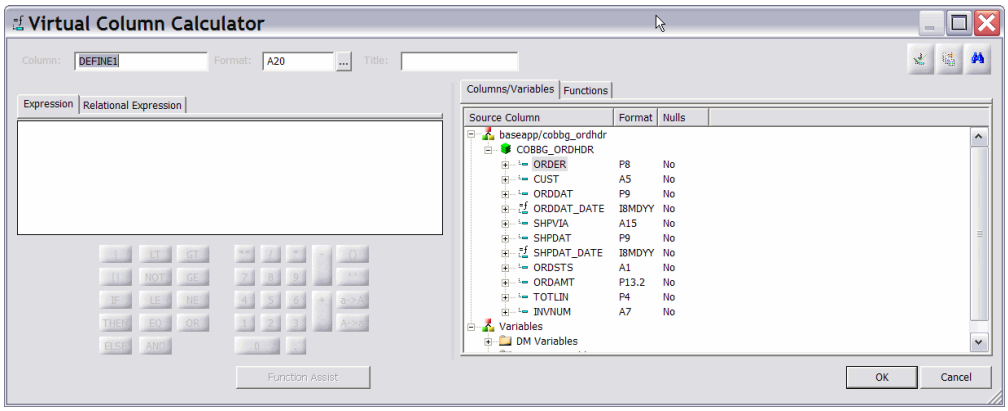


Figure 5: Now you'll see the Virtual Column Calculator.

Notice the three helpful icons on the top right corner of this window, as shown in Figure 6.

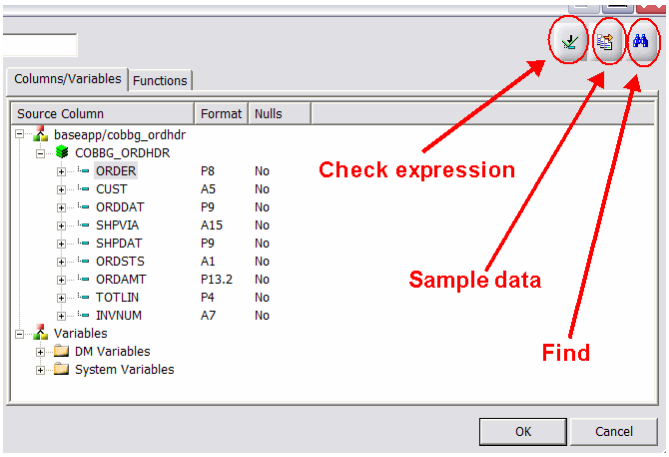


Figure 6: The virtual column calculator provides three helpful icons.

- **Check expression** checks the validity of the expression's syntax.
- **Sample data**, once the expression is entered, further validates the expression by displaying a sample of the column's values.
- **Find** can be used to find functions and field names.

Keep these useful tools in your back pocket; they can be used to help you find and specify the correct function and syntax.

To convert the packed decimal fields to dates, use the DATECVT function. This function converts the field value of any standard date format or legacy date format into a new date, in either the desired standard date format or legacy date format. These are the parameters for this function:

- **date** is the input legacy field to be converted.
- **in_format** is the format of the input legacy date. Examples: I8MDYY, I6YMD, A8MDYY.
- **output_format** is the output date format. Examples: YYMD, YQ, M, DMY, JUL.

5. From the Virtual Column Calculator window, specify the following:

- Column: SHPDAT_DATE (this is the new virtual column name)
- Format: MDYY (this is the output date format of the new virtual column)
- Expression: DATECVT( SHPDAT, 'I8MDYY', 'MDYY' )

When you're finished, the virtual column should look like the example in Figure 7.

Tip: Experiment with the Find icon and the Function Assist button to help guide you.

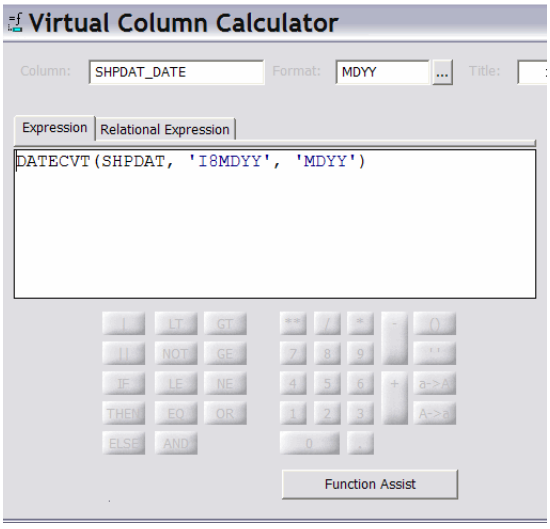


Figure 7: You now have an expression for your new virtual date column.

6. To verify the correct syntax of your expression, click on the Check Expression icon.
7. To further verify the conversion and view sample data, click on the Sample Data icon. If the conversion was successful, you will see valid sample date data as shown in Figure 8.

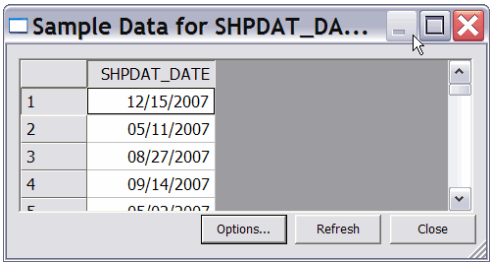


Figure 8: You should now see sample data for your new virtual date column.

8. Repeat steps 4 through 7 to create another virtual date column for the legacy ORDDAT field.

DB2 Web Query provides many different formatting codes to display the values of date columns in various ways. You simply specify one of these formatting codes in the virtual column's Format specification. When that virtual column is used in your report, the date will be displayed in the format that follows the specified formatting code. A list of these valid codes is shown in the table below.

## Date Formatting Codes

| | | |
|---|---|---|
| D | Day | Value from 1 to 31 for the day |
| M | Month | Value from 1 to 12 for the month |
| Y | Year | Two-digit year |
| YY | Four-Digit Year | Four-Digit Year |
| T | Translate Month or Day | Three-letter abbreviation for months in uppercase, if M is included in the USAGE specification |
| t | Translate Month or Day | Functions the same as uppercase T (described above), except that the first letter of the month or day is uppercase and the following letters are lowercase |
| TR | Translate Month or Day | Functions the same as uppercase T (described above), except that the entire month or day name is printed rather than an abbreviation |
| tr | Translate Month or Day | Functions the same as lowercase t (described above), except that the entire month or day name is printed rather than an abbreviation |
| Q | Quarter | Quarter (1-4 if Q is specified by itself, or Q1-Q4 if it is specified together with other date format items such as Y) |
| W | Day of Week | If it is included in a USAGE specification with other date component options, it prints a three-letter abbreviation of the day of the week in uppercase. If it is the only date component option in the USAGE specification, it prints the number of the day of the week (1-7, Mon=1). |
| w | Day of Week | Functions the same as uppercase W (described above), except that the first letter is uppercase and the following letters are lowercase. |
| WR | Day of Week | Functions the same as uppercase W (described above), except that the entire day name is displayed instead of an abbreviation. |
| wr | Day of Week | Functions the same as lowercase w (described above), except that the entire day |

| | | |
|---|---|---|
| | | name is displayed instead of an abbreviation. |
| JUL | Julian format | Displays date in Julian format. |
| YYJUL | Julian format | Displays a Julian format date in the format YYYYDDD. The 7-digit format displays the four-digit year and the number of days counting from January 1. For example, January 3, 2001 in Julian format is 2001003. |

Note: Dates can be formatted by any single code or any combination of codes, such as Q or YYQ.

The final requirement in this particular example is to create a virtual column that contains the entire name of the ORDDAT month (for example, September). From the information provided in the date-formatting codes table, you can determine that the formatting code for this requirement is Mtr ("M" for month and "tr" for entire month name).

9. Create a new virtual column named ORDDAT_MONTH. Specify Mtr for the format and ORDDAT_DATE for the expression. It will look like the example shown in Figure 9.

Note: Make sure you specify the smart date column ORDDAT_DATE, not the legacy field ORDDATE.

Figure 9: You now have a new virtual column named ORDDAT_MONTH.

10. To verify the results, click the Sample Data icon. If successful, you will see a window similar to Figure 10.

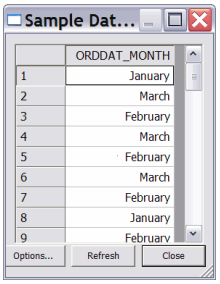


Figure 10: You should now see your sample of ORDDAT_MONTH.

When you are done, the synonym will look like the example shown in Figure 11.

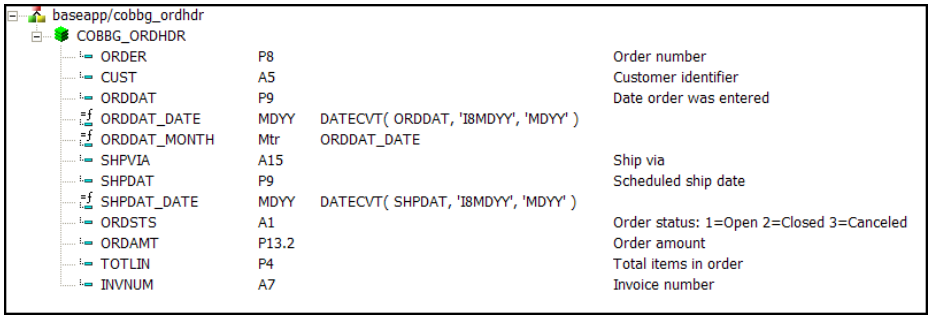


Figure 11: Finally! Your synonym after the virtual columns have been added.

Creating Virtual Date Columns Example 2: Separate Date Fields

Many legacy database designs represent the date by creating four separate decimal (zoned or packed) fields: one each for century, year, month, and day. This is often referred to as a "decomposed date" and is shown in the following table.

## Four Zoned Decimal Fields Representing a Date

| Example Legacy Date Field Name | Data Type | Date Format | Example (April 10, 2008) |
|---|---|---|---|
| ZN_CENTURY | ZONED (2,0) | C | 20 |
| ZN_YEAR | ZONED (2,0) | Y | 08 |
| ZN_MONTH | ZONED (2,0) | M | 04 |
| ZN_DAY | ZONED (2,0) | D | 10 |

To convert these fields into one consolidated DB2 Web Query smart date field, follow the same steps described above to create a new virtual column. However, for this particular virtual column, specify the following:

- Format: YYMD
- Expression value: `DATECVT( ((ZN_CENTURY * 1000000) + (ZN_YEAR * 10000) + (ZN_MONTH * 100) + ZN_DAY), 'I8YYMD', 'YYMD' )`

In this technique, arithmetic expressions are used to combine the legacy date fields into a single value, which is then passed as the first parameter into the DATCVT function. This example is displayed in Figure 12.

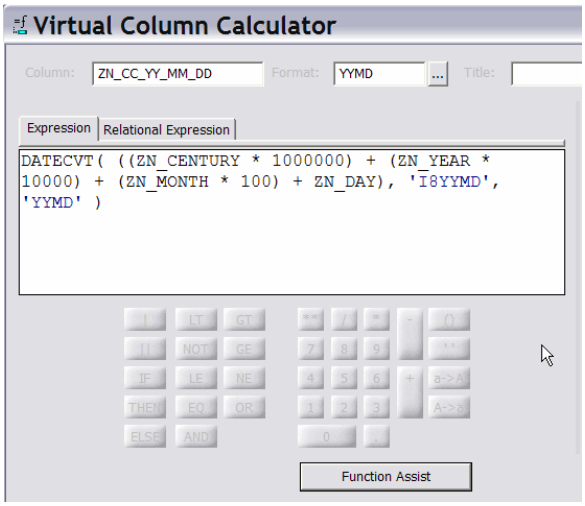

Figure 12: Use the date conversion expression to combine four legacy date fields.

## Creating Virtual Date Columns Example 3: Julian Dates

Another common method of representing dates in legacy applications is to store them as decimal fields in the Julian date format. A date in the Julian format is either a five- or seven-digit number. The first two or four digits are the year; the last three digits are the number of the day of the year (starting from January 1). For example, the Julian value for January 1, 2008, is either 08001 or 2008001; and April 17, 2008 ,is represented as either 08107 or 2008107.

To convert these legacy decimal fields to dates, the GREGDT function is used. This function converts a date in Julian format (year-number_of_the_day) to Gregorian format (year-month-day).

These are the parameters for this function:

- **indate** is the input Julian date field to be converted. It must be a numeric field of five or seven digits.
- **output_format** is I6, I8, I6YMD, or I8YYMD.

| Example Legacy Date Field Name | Legacy Data Type | Legacy Date Format | Legacy Example (April 10, 2008) |
|---|---|---|---|
| ZN_JUL | ZONED (7,0) | YYDDD Julian Date | 2008100 |

To convert Julian date fields into a DB2 Web Query smart date, follow the same steps described above to create a new virtual column, and specify the following:

- Format: YYMD
- Expression value: `GREGDT( ZN_JUL, 'I8YYMD' )`

Using this conversion method, the new virtual column will display a legacy Julian date value of 2008107 as a date field with the value 2008/04/17. GREGDT always returns the date in YYMD format. If you would rather display this virtual column in MDYY format in your reports, specify the following:

- Format: MDYY
- Expression value: `DATECVT(GREGDT(ZN_JUL, 'I8YYMD' ), 'I8YYMD', 'MDYY' )`

## Creating Virtual Date Columns: Other Common Examples

I have seen System i shops represent their date fields in many ways. A list of some of the more commonly used legacy date formats and the corresponding DB2 Web Query conversion formats and expressions are show in the following table.

## Quick Date Conversion Reference

| Example Legacy Date Field Name | Legacy Data Type | Legacy Date Format | Legacy Example (April 10, 2008) | Output Date Format | Conversion Expression |
|---|---|---|---|---|---|
| CH_MDYY | A(8) | MDYY | '04102008' | MDYY | `DATECVT(CH_MDYY, 'A8MDYY', 'MDYY')` |
| CH_YYMD | A(8) | YYMD | '20080410' | YYMD | `DATECVT( CH_YYMD, 'A8YYMD', 'YYMD' )` |
| ZN_MDYY | ZONED (8,0) | MDYY | 04102008 | MDYY | `DATECVT( ZN_MDYY, 'I8MDYY', 'MDYY' )` |
| ZN_YYMD | ZONED (8,0) | YYMD | 20080410 | YYMD | `DATECVT( ZN_YYMD, 'I8YYMD', 'YYMD' )` |
| PK_MDYY | PACKED (8,0) | MDYY | 04102008 | MDYY | `DATECVT( PK_MDYY, 'I8MDYY', 'MDYY' )` |
| PK_YYMD | PACKED (8,0) | YYMD | 20080410 | YYMD | `DATECVT( PK_YYMD, 'I8YYMD', 'YYMD' )` |
| ZN_JUL | ZONED (7,0) | YYDDD<br>Julian Date | 2008100 | YYMD | `GREGDT( ZN_JUL, 'I8YYMD' )` |
| PK_JUL | PACKED (7,0) | YYDDD<br>Julian Date | 2008100 | YYMD | `GREGDT( PK_JUL, 'I8YYMD' )` |
| ZN_CENTURY<br>ZN_YEAR<br>ZN_MONTH<br>ZN_DAY | ZONED (2,0)<br>ZONED (2,0)<br>ZONED (2,0)<br>ZONED (2,0) | CC<br>YY<br>MM<br>DD | 20<br>08<br>04<br>10 | YYMD | `DATECVT( ((ZN_CENTURY * 1000000) + (ZN_YEAR * 10000) + (ZN_MONTH * 100) + ZN_DAY), 'I8YYMD', 'YYMD' )` |

## Summary

DB2 Web Query provides many BIFs to help convert your legacy date fields to smart dates. Regardless of the data type and format that you use to represent dates in your DB2 for i5/OS files, there is a way to convert them. But these BIFs are not the only option that you have when it comes to date conversion. In an upcoming TechTip, I will describe a technique using a date conversion table that is easy to implement and, when utilized, can provide significant report performance enhancements.