

## The Progress Electronic Magazine

**In this issue:**

<b>Publisher's Statement:</b> .....	<b>2</b>
<b>Coding Article: Managing Dynamic Queries</b> .....	<b>3</b>
<b>Publishing Information:</b> .....	<b>7</b>
<b>Other Progress Publications Available:</b> .....	<b>7</b>
<b>Products Available From Amduus:</b> .....	<b>8</b>
<b>Article Submission Information:</b> .....	<b>8</b>
<b>Order Form for Progress Open Source CD-ROM</b> .....	<b>9</b>

*This document may be freely shared with others without modification. Subscribe for free here: <http://www.amduus.com/online/dev/ezine/EZineHome.html>*

*Though intended for users of the software tools provided by Progress Software Corporation, this document is NOT a product of Progress Software Corporation.*

© Amduus Information Works, Inc. 2003

**Publisher's Statement:**

Welcome to Issue 28! On review of previous issues, I see that we have nearly two years of publication. I remember when I was happy just to get 30 people reading the E-Zine – now I find it is read by over one thousand people in over five hundred companies across over twenty countries! Wow!

In this issue, Tim Kuehn continues his articles on dynamic query programming. This article steps up the power of the queries compared to the previous article by including some ways to manage the queries with some code.

This will be a quick read because I have to finish up some real work. The contract will be ending soon, so if you have any work available – let me know!

Beside this column is an ad about Amduus looking for consultants. Let me tell you more about the software. It is a version 2 of Service Express. Service Express allows companies to take in issues over the internet and manage those issues. Let's assume one's client is an equipment manufacturing company. They service their equipment and one of their customers has trouble with a piece of equipment. Their customer can inform the company of the problem by use of the web based software. More information is available in power point presentation. Contact me for more information.

**Reach over one thousand  
programmers and companies.**

**Your ad could be here!**

**Advertise in the E-Zine for  
\$10.00 per issue!**

**!!!WANTED!!!**

Amduus Information Works, Inc. is looking for consultants to resell access to up-coming ASP web based software. We will need you to find companies who would want use of this software, to configure the software to their needs, and to support them in the use of the software. The software is rented out – no licenses are sold. Each month, you would receive a portion of the revenue, as well be able to bill for training and support – modeled like an insurance agency. Contact [sauge@amduus.com](mailto:sauge@amduus.com) for more information.

## Coding Article: Managing Dynamic Queries

By Tim Kuehn - TDK Consulting Services

*(Editor's Note: You can find the underlying source at Tim's web site noted at the end of the article.)*

In my introduction to dynamic queries (<http://www.tdkcs.ca/progress/dynqyintro.html>) I illustrated the similarities and differences of dynamic and static queries as an aid to learning how use dynamic queries.

We also saw how dynamic queries can provide much greater flexibility in searching data. To recap, a simple dynamic query code looks like this:

```
DEFINE VARIABLE cqh AS HANDLE NO-UNDO.  
DEFINE VARIABLE cth AS HANDLE NO-UNDO.  
CREATE BUFFER cth FOR TABLE ("customer").  
CREATE QUERY cqh.  
cqh:ADD-BUFFER(cth).  
cqh:QUERY-PREPARE("FOR EACH customer").  
cqh:QUERY-OPEN().  
  
(Process records until cqh:QUERY-OFF-END = TRUE)  
  
DELETE OBJECT cqh.  
DELETE OBJECT cth.
```

A query that only involves a few tables is pretty easy to handle. But what if we want to create a query on an arbitrary number of different tables determined at run time? Or if the search condition changes depending on the tables used in the search? In such cases using static “handle” variables to hook into the various table buffers is no longer a viable option. Also, dynamic queries require the programmer to manage constructing and deleting the dynamic query and buffer objects associated with them. Finally, there's no way to delete the query and any dynamic buffers associated with it in a single call - a series of DELETE OBJECT statements are required. In and of itself that's not as much of a problem, until you have to program and maintain large numbers of these queries.

So what to do? Being a lazy “I-hate-to-do-the-same-thing-more-than-once” programmer my solution was to create a super-procedure to manage the dynamic query life-cycle so I didn't have to re-write the wheel over and over again.

The first step in creating this query manager is to detail the life-cycle of a dynamic query. Specifically, the process is:

- 1) Create the dynamic query object
- 2) Create any required dynamic buffer(s)
- 3) Associate the (dynamic) buffer(s) with the dynamic query

- 4) “query-prepare” the query
- 5) Open the query
- 6) Step through the query results.
- 7) Close the query
- 8) Delete any dynamic buffers and the query object to release any associated memory.

In addition, a mechanism for the developer to obtain specific buffer or field object handles is required.

Given these characteristics, it’s time to design the query manager. A flexible programming interface is the first step in this process.

Steps 1 - 5 will be handled by the following call:

```
RUN qrymgr-query-create("table1,table2,...,tableN", "query-string", OUTPUT cur-
query-handle).
```

This will create the dynamic query object, create a dynamic buffer for each table in the table list, and associate that buffer with the query in the order the tables are specified. The query object then has the “query-string” search criteria “query-prepare”d and the query is then opened.

Handles for all query and buffer objects are stored in a pair of temp tables. Query handles go in the *ttquery* table, while buffer handles go in *tt-query-buffer-handles* table, with a link by the query-number and query-handle fields.

```
DEFINE TEMP-TABLE tt-query NO-UNDO
FIELD query-number AS INTEGER
FIELD query-handle AS HANDLE
INDEX i-qy-hdl query-handle
INDEX i-qy-num query-number
.
DEFINE TEMP-TABLE tt-query-buffer-handles NO-UNDO
FIELD query-number AS INTEGER
FIELD query-handle AS HANDLE
FIELD db-table-name AS CHARACTER
FIELD buffer-handle AS HANDLE
INDEX i-qy-hdl query-handle
INDEX i-qy-num query-number
.
```

By using temp-tables to hold these handles, an arbitrary number of queries and associated dynamic buffers can be managed. These buffer and query handles are retained for later reference between calls to the super procedure.

To complete step 6 (process the query results) we’ll need access to the buffer and field objects the query manager created. The functions to get these handle(s) will look like so:

```
cbh = qrymgr-query-buffer-handle-get(cur-query-handle, "db.table-name").
```

```
cfh = qrymgr-query-buffer-field-handle-get(cur-query-handle, "db.table-name",
"field-name").
```

To perform step 7 (query and dynamic buffer cleanup) - the following procedure call will be used:

```
RUN qrymgr-query-delete(cur-query-handle).
```

The query manager will lookup the original *tt-query* record, find all the associated *tt-query-buffer-handles* records, delete the associated buffers and then the *tt-query-buffer-handles* record, then the query object referenced by the *tt-query* record, and then the *tt-query* record itself.

The following example shows some sample code using this structure. This program reports all table and field names which are part of an index.

```
/* code2.p */
{qrymgr.spi} /* Super procedure header */
DEFINE VARIABLE cur-sp-handle AS HANDLE NO-UNDO.
DEFINE VARIABLE cur-query-handle AS HANDLE NO-UNDO.
DEFINE VARIABLE cur-table-name-handle AS HANDLE NO-UNDO.
DEFINE VARIABLE cur-field-name-handle AS HANDLE NO-UNDO.
RUN qrymgr.sp PERSISTENT SET cur-sp-handle. /* Run procedure persistently */
SESSION:ADD-SUPER-PROCEDURE(cur-sp-handle). /* Add it as a super-procedure */
/* Setup and start the dynamic query */
RUN qrymgr-query-create("sports._file," +
"sports._index," +
"sports._index-field," +
"sports._field",
"FOR EACH &1, " +
"EACH &2 " +
"WHERE &2._file-rcid = RECID(&1)," +
"EACH &3 " +
"WHERE &3._index-rcid = RECID(&2)," +
"EACH &4 " +
"WHERE RECID(&4) = &3._field-rcid",
OUTPUT cur-query-handle).
/* Get the buffer-field handle for
_file._file-name */
cur-table-name-handle =
qrymgr-query-buffer-field-handle-
get(cur-query-handle,
"sports._file",
"_file-name").
/* Get the buffer-field handle for _field._field-name */
cur-field-name-handle =
qrymgr-query-buffer-field-handle-get(cur-query-handle,
"sports._field",
"_field-name").
/* Process the query results */
cur-query-handle:GET-FIRST(NO-LOCK).
DO WHILE cur-query-handle:QUERY-OFF-END = FALSE:
DISPLAY
cur-table-name-handle:BUFFER-VALUE FORMAT "X(20)"
cur-field-name-handle:BUFFER-VALUE FORMAT "X(20)"
WITH DOWN.
DOWN.
cur-query-handle:GET-NEXT(NO-LOCK).
END.
/* Cleanup the query */
RUN qrymgr-query-delete(cur-query-handle).
```

Amduus Information Works, Inc. also provides documentation services! Scott Auge notes, "One of the things I have noticed throughout my contracting career is that companies with developed software always seem to be missing or weak on user documentation, administration documentation, and programmer documentation." Amduus can help you with this!

The actual code to implement the query manager this is a bit beyond the scope of this article, but a complete copy of this article and sample code can be found at

<http://www.tdkcs.ca/progress/qrymgr.zip>.

Or for a bit of a challenge and extra learning you can write the code yourself!

For extra credit,

- 1) add the capability to pass external buffer-handles to the query manager,
- 2) add the ability to substitute buffer names in the query string – ie

```
RUN qrymgr-query-create("customer", "FOR EACH &1", OUTPUT cur-query-handle).
```

**Publishing Information:**

Scott Auge publishes this document. I can be reached at [sauge@amduus.com](mailto:sauge@amduus.com).

Amduus Information Works, Inc. assists in the publication of this document by providing an internet connection and web site for redistribution:

Amduus Information Works, Inc.  
1818 Briarwood  
Flint, MI 48507  
<http://www.amduus.com>

**Other Progress Publications Available:**

This document focuses on the programming of Progress applications. If you wish to read more business oriented articles about Progress, be sure to see the Profile's magazine put out by Progress software:



```
def var x as int init 1.  
    x eq 12.  
display x.
```

**If you were using Prolint, you wouldn't have spent the last two hours tracking down this bug.**

**Proparse Lite - Only US\$65**

**Lint your code before check-in.**

**Find bugs before your customer does.**

**www . joanju . com**

<http://www.progress.com/profiles/>

There are other documents/links available at <http://www.peg.com> .

**Products Available From Amduus:**

Amduus Information Works, Inc. is a Progress reseller and ASPen partner. We primarily develop UNIX/Linux based applications with web interfaces for manufacturing, service, and law enforcement communities.

We also perform integration of Progress applications to non-Progress applications through such languages and tools as MQ Series, C, and C++.

Amduus provides support for the following applications: Blue Diamond, Denkh, Denkh HTML Reporter, Red Arrow Portal (CMS), Survey Express and other software.

Amduus is looking for consultants who might want to promote the use of our tools at user groups and companies they might work in. Send some information to [sauge@amduus.com](mailto:sauge@amduus.com) to let me know you are out there!

**Article Submission Information:**

Please submit your article in Microsoft Word format or as text. Please include a little bit about yourself for the About the Author paragraph.

Looking for technical articles, *marketing Progress* articles, articles about books relevant to programming/software industry, white papers, etc.

**Order Form for Progress Open Source CD-ROM**

COUPON 001A

This is an offer for the CD-ROM at lower than list savings!  
This is a great way to support the E-Zine too!

Mail this form to:  
**Amduus Information Works, Inc.**  
**1818 Briarwood**  
**Flint, MI 48507**



Please send \_\_\_\_\_ copies of the Open Source CD-ROM at \$25.00 per disk to:

Name \_\_\_\_\_

Company \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_

State \_\_\_\_\_ Country \_\_\_\_\_

Zip \_\_\_\_\_

Please make your checks/money orders out to: [Amduus Information Works, Inc.](#) Cash works too!  
This offer only valid in the United States of America.

The CD-ROM includes (all source code included):

- Blue Diamond/IRIS – Webspeed alternatives
- Survey Express – easily create text templates of surveys and then have the program generate the web pages automatically
- Service Express – Web based Help Desk.
- The Progress E-Zines, books on learning to program in Webspeed (PDF/Word/HTML)
- Denkh HTML Reporter – web based report writer
- CMS – a web content management system
- DB Email – Use pop3 to download emails into a Progress database
- Neural Networks – experiments in spam recognition and text message classification
- Denkh – create PDF file reports for Webspeed/UNIX CHUI!
- More!

