Progress®



Improve the Performance of the OpenEdge Database

OpenEdge®

Copyright

© 2021 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Icenium, Inspec, Ipswitch, iMacros, Kendo UI, Kinvey, MessageWay, MOVEit, NativeChat, NativeScript, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), SpeedScript, Stylus Studio, Stylized Design (Arrow/3D Box logo), Styleized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries.

Analytics360, AppServer, BusinessEdge, Chef Automate, Chef Compliance, Chef Desktop, Chef Habitat, Chef WorkStation, Corticon.js, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Everywhere, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, JustAssembly, JustDecompile, JustMock, KendoReact, NativeScript Sidekick, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Insight, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

July 2021

Product version: Progress OpenEdge 12.4

Updated: 2021/06/28

Table of Contents

Preface....7

Improve the Performance of the OpenEdge Database....9

Extend variable-length database extents....10
Database startup parameter default updates....15
Number and size of sequences....16

Preface

Purpose

The purpose of this book is to introduce performance features in the OpenEdge database.

Audience

This books is intended for anyone interested in OpenEdge database performance improvements.

Organization

Extend variable-length database extents on page 10

Explains how to extend the size of existing variable-length extents and create new variable-length extents.

Database startup parameter default updates on page 15

Introduces the feature capabilities and some of the tools created or enhanced for managing multi-tenant databases.

Number and size of sequences on page 16

Compares the maximum number of unique sequences supported by various versions of OpenEdge.

Documentation conventions

See Documentation Conventions for an explanation of the terminology, format, and typographical conventions used throughout the OpenEdge content library.

1

Improve the Performance of the OpenEdge Database

OpenEdge Database is designed for performance, responsiveness, and scalability.

Built-in functionality optimizes parameters and capacities and provides improvements for database administrators. Performance features include:

- **Ability to Extend Variable-length Database Extents--**Database administrators can extend and mark variable-length `ai`, `bi`, and `db` extents as fixed as well as create and pre-extend new variable-length extents. Learn more: Extend variable-length database extents.
- **Database Startup Parameter Defaults--**The default values of various startup parameters improve the overall performance of OpenEdge databases. Learn more: Database startup parameter default updates.
- **Database Maintenance Mode**--OpenEdge includes a restricted online mode, so that you can run maintenance tasks with all the buffer pool and helper processes still available, improving the performance for these tasks. Learn more: PROUTIL DBRESTRICT qualifier.
- **Maximum Limit on the Number of Sequences--**The maximum number of unique sequences in an OpenEdge Database is 32,000 regardless of block size. Learn more: Number and size of sequences.
- **Server-side Join Processing--**Certain queries that join multiple tables are resolved in the database server rather than the client. As a result fewer records may be returned to the client leading to faster performance. Learn more: Server-side join processing.

- **Manage Before-image File Growth** You can use the `-biscantime` parameter to tune the behavior for before-image space management. In some rare cases, BI files may keep growing even when the BI space management is running. This could happen under heavy database activities with very fast checkpoints. Reducing the value of `-biscantime` lets the before-image manager (BIM) process detect and reuse BI clusters more frequently at a cost of minor performance impact when compared to a bigger `-biscantime` value. Note that, in most cases, the default value of `-biscantime` (10) works well without any tuning. Setting the value of `-biscantime` to zero (0) turns off BI space management. (If you set `-biscantime` to 0 online, the before-image writer stays idle.) Whenever you set `-biscantime` to a non-zero value, you must start the before-image manager.

  For more information, see "Before-image scan time (-biscantime)" and "Start and stop a BIM" in *Manage the OpenEdge Database*.

For details, see the following topics:

- Extend variable-length database extents
- Database startup parameter default updates
- Number and size of sequences

Extend variable-length database extents

Variable-length database ai, bi, and data extents may be extended or pre-extended by the user in a database structure (`.st`) file using PROSTRCT utilities while the database is on- or offline.

This functionality enables database administrators to:

- extend the size of an existing variable-length extent before converting it to fixed-length when adding a new variable-length extent to a storage area
- extend the size of an existing variable-length extent to a specific pre-grown size whether it is empty or already contains data
- extend the size of an existing variable-length extent to a specific pre-grown size and impose a maximum size cap
- create a new variable-length extent pre-extended to a specific size in a new or existing area
- create a new pre-extended variable-length extent with an imposed maximum size in a new or existing area

Extending or pre-extending the size of variable-length extents reduces overhead and improves runtime by minimizing resource requests to the file system.

Extent Length Format Tokens

Variable-length extent sizes may be pre-grown (extended or pre-extended) by editing the `extentType` token in the database structure (`.st`) file using the extent length format tokens listed in the following table.

Table 1: Extent Length Format Tokens

Extent Length Format Token	Function
e	extends an existing variable-length extent.
ev	extends an existing variable-length extent and imposes a maximum size.

Extent Length Format Token	Function
x	creates a new pre-extended variable-length extent.
xv | maximum size	creates a new pre-extended variable-length extent with an imposed maximum size.

The "e" and "x" extent length format tokens require an accompanying specified extend or pre-extend size value in the .st file as follows:

```
#
d "Data Area10":10,32;64   x_10.d1     x  128
#
```

The "ev" and "xv" extent length format tokens require two accompanying specified size values in the .st file as per the example below. The first designates the pre-extend value of the extent. The second designates the imposed maximum size of the extent.

```
#
d "Data Area10":10,32;64   x_10.d1     xv  256   512
#
```

Extent length format tokens operate for both Type I and Type II storage areas. Extended and pre-extended values are reported in PROSTRCT LIST, _areaExtent, the _Filelist VST, and PROMON.

Previously pre-extended variable-length extents may be pre-extended again. The new pre-extended value will be reported in PROSTRCT LIST, _areaExtent, the_Filelist VST, and PROMON.

The length specified by an extent length format token does not extend or pre-extend the current size of the target extent by the specified value but designates the final, resulting size of the extent length. The extent size specified by an extent length format token will be validated the same as if the "f" token was specified.

The following table cites operational rules for each specific extent token:

Table 2: Extent Token Operational Rules

Token	Operational Rules
e	• An "e" token extends an existing variable-length extent in existing ai and data storage areas during a PROSTRCT ADD or ADDONLINE operation. • An "e" token will extend the existing variable-length extent and then mark it as fixed for bi and data storage areas if followed by the creation of a new variable-length extent marked with an extent type token of "v", "x", "xv" or a blank value. • Designation of an "e" length format value without an existing variable-length extent in a storage area or in a new storage area will result in an error. • Designation of an "e" length format value following a "v" or "x" length format value for the same storage area in an `.st` file will result in an error. • If a filename is not specified in conjunction with an "e" extent token, the pre-extend will be applied to the existing extent in the storage area if it is a variable-length extent. If it is not, an error will result. • Applying an "e" value of "0" during a PROSTRCT ADD or ADDONLINE operation will remove any explicitly- or implicitly-stated pre-extend length from an existing variable-length extent. • The "e" length format does not apply to a PROSTRCT CREATE command. An error will result.
ev	• An "ev" token extends and imposes a maxium size cap on an existing variable-length extent in existing ai and data storage areas during a PROSTRCT ADD or ADDONLINE operation. • The "ev" token must be followed by two size values or an error will result. The first size value of "e" designates the pre-extend value of the existing extent. The second size value of "v" designates the maximum size of the extent (capped). • The "e" length format extent value must be less than the specified "v" length format value or an error will result. • Designation of an "ev" length format value without an existing variable-length extent in a storage area or in a new storage area will result in an error. • Designation of an "ev" length format value following a "v" or "x" length format value for the same storage area in an `.st` file will result in an error. • The "ev" extent token will operate regardless if a specific file name is explicitly requested or not. • If a filename is not specified in conjunction with an "ev" extent token, the pre-extend and maximum size will be applied to the existing extent in the storage area if it is a variable-length extent. If it is not, an error will result. • Applying an "ev" value of "0, 0" during a PROSTRCT ADD or ADDONLINE operation will remove any explicitly- or implicitly-stated pre-extend length and maximum size cap from an existing variable-length extent. • Applying an extend length value to "e" in conjunction with a maximum size value of "0" to "v" for an "ev" extent token during a PROSTRCT ADD or ADDONLINE operation will pre-extend an existing variable-length extent and remove any explicitly- or implicitly-stated maximum size cap. • The "ev" length format does not apply to a PROSTRCT CREATE command. An error will result.

Token	Operational Rules
x	• An "x" format extent token creates a new variable-length extent and pre-extends its length in new and existing ai, bi, and data storage areas. • Designation of an "x" token for a new variable-length extent will mark an existing variable-length bi or data extent in a storage area as fixed. • The "x" extent token will operate regardless if a specific file name is explicitly requested or not. • Specifying the filename of an existing extent for an "x" extent token will result in an error. • Once an extent has been extended by an "x" extent token, subsequent PROSTRCT LIST operations will list the "x" value until the extent becomes fixed. Formatting an extent as fixed will clear the pre-extend "x" value.
xv	• An "xv" token creates a new variable-length extent with both a pre-extended size and maximum size cap in new and existing ai, bi, and data storage areas. • Designation of an "xv" token for a new variable-length extent will mark an existing variable-length bi or data extent in a storage area as fixed. • The "xv" token must be followed by two size values or an error will result. The first size value of "x" designates the pre-extend value of the newly created extent. The second size value of "v" designates the maximum size of the extent (capped). • The "x" length format extent value must be less than the specified "v" length format value or an error will result. • The "xv" extent token will operate regardless if a specific file name is explicitly requested or not. • Specifying the filename of an existing extent for an "xv" extent token will result in an error.

Database Extent Utilities

Extend and pre-extend functionality is supported by the following database extent utilities:

- PROREST
- PROSTRCT ADD
- PROSTRCT ADDONLINE
- PROSTRCT BUILDDB
- PROSTRCT CREATE
- PROSTRCT LIST
- PROSTRCT PROCOPY
- PROSTRCT `-validate`
- PROUTIL TRUNCATE BI
- RFUTIL AIMAGE EXTENT EMPTY

PROREST and PROSTRCT PROCOPY will honor the setting in the target `.st` file or create an `.st` file from the source. The target of a PROREST or PROSTRCT PROCOPY will pre-extend according to the specified target `.st` file.

Truncate operations for ai and bi extents will honor the specified value for "x" format tokens. RFUTIL AIMAGE EXTENT EMPTY will honor the pre-extend setting for ai files by pre-extending the ai files after truncation. PROUTIL TRUNCATE BI will honor the pre-extend setting for bi files by pre-extending the bi files after truncation.

The current bithold (if set) will be honored when performing a PROSTRCT ADDONLINE to pre-extend bi extents.

Sample Database Structure File

The sample database structure (`.st`) file below demonstrates the correct and incorrect usage the `extentType` format tokens for extending and pre-extending variable-length extents using PROSTRCT ADD. The comments explain possible errors for new and existing extents when applying the "e", "ev", "x", and "xv" tokens.

```
#
b /usr2/dbadmin/11/x.b1          f 32768
b /usr2/dbadmin/11/x.b2          x 64000     ### OK - new variable-length extent
#                                                  created and pre-extended to 64000KB
#
d "Schema Area":6,64;1   x.d1
#
d "Index Area":7,32;64   x_7.d2  e  128      ### OK - existing d2 var. length extent
#                                                   extended and marked as fixed if
#                                                   followed by "xv"
d "Index Area":7,32;64   x_7.d3  xv 256 512  ### OK - new var. length extent added and
#                                                  pre-extended 256 with a max. size
#                                                   of 512
#
d "Index Area":8,32;64           ev 128 256  ### OK - if var. length extent already
#                                                   exists, existing extent
#                                                   extended 128 with a max. size
#                                                   of 256
d "Index Area":8,32;64           xv 256 512  ### OK - var. length extent added and
#                                                  pre-extended 256 with a max. size
#                                                   of 512
#
d "Data Area":9,32;8    x_9.d1  x  128      ### OK - new pre-extended var. length
#                                                   extent created
d "Data Area":9,32;8    x_9.d2  v  256      ### Error - attempt to create another
#                                                    variable-length extent
#
d "Data Area10":10,32;64 x_10.d1 x  128
d "Data Area10":10,32;64 x_10.d2             ### Error - "x" and " " indicate 2
#                                                   variable-length extents
#
d "Data Area11":11,32;64 x_11.d1 v  256
d "Data Area11":11,32;64 x_11.d1 x  128      ### Error - attempt to create another
#                                                   variable-length extent
#
d "Data Area12":12,32;64 x_12.d1 f  128      ### OK - in this case, .d1 behaves just
                                                   as "f 128"
d "Data Area12":12,32;64 x_12.d2 x  256      ### OK - new var. length extent
#                                                   pre-extended to 256
#
d "Data Area13":13,32;64 x_13.d1 f  128
d "Data Area13":13,32;64 x_13.d2 e  256      ### Error - new var. length extent; no
#                                                  existing extent for "e" to act
on
#
d "Data Area14":14,32;64         x  128
d "Data Area14":14,32;64                     ### Error - attempt to add two variable-
#                                                   length extents
#
d "Data Area15":15,32;64         f  256
d "Data Area15":15,32;64         xv 128 128  ### OK - new var. extent pre-extended
#                                                   to 128 with a max. size of 128
#
```

```
d "Data Area16":16,32;64 x_16.d1 xv 128      ### Error - only one size value specified
#
#
### The following describes the addition of 5 ai areas/extents
a .          f  128       ### Fixed-length extent at 128K
a .                       ### Variable-length extent
a .          x  128       ### New pre-extended variable-length extent
a .          e  128       ### Error - no existing area/extent name to enact "e" on
a   x.a5     e  256       ### This will extend existing var.-length extent to 256KB
a .          xv 128 256   ### New pre-extended var.-length extent created and capped
a   x.a7     v  256       ### New maximum-sized variable-length extent created
a   x.a7     x  128       ### Error - attempt to use same filename as existing extent
a .          e  819200    ### Error: Extend existing extent option 'e' requires area
#                                    name or file name
a   x.a5                       e  819200         ### is OK
a "After Image Area 3" .       ev 8192   819200  ### is OK
a "After Image Area 2" ./x.a2  ev 81920  819200  ### OK
```

Database startup parameter default updates

The default values of various database startup parameters have been increased to better align with the demands of today's modern applications. The parameters and their updated default values are listed in the following table.

Table 3: Database startup parameter default value changes (license restricted)

Parameter name	Old value	New value
-aibufs	20	64
-bibufs	20	64
-lruskips	0	100
-lru2skips	0	100
-pica	64K	128K
-prefetchDelay	0	1
-prefetchFactor	0	100
-prefetchNumRecs	16	64
-prefetchPriority	0	1
-Mm	1024	8192
-Mxs	16K	32K

Number and size of sequences

The maximum number of unique shared sequences supported in an OpenEdge database is 32,000 regardless of block size.

All newly-created and converted databases have 64-bit sequences and are bounded by the maximum of a signed 64-bit integer.

For more information on 64-bit sequences, see the online Help for the Data Dictionary or Data Admin.

Note: The maximum number of Multi-tenant sequences is limited to 2000.