White Paper

Published: March 1999 (with revisions)

Ingres Variable Size Data Pages and Indexes

Contents

Preface

Size matters! If you are thinking about implementing something other than the default 2KB data pages size in your Ingres installation, it will be helpful to read through the considerations discussed in this white paper. Please note that Advantage Ingres 2.6 implemented changes ("v3" and "v4" page types) that reduce the overhead discussed in this paper. While some of the issues still exist (e.g., rows in base tables are TID limited) the problem has been greatly diminished.

Overview

As many of you are well aware, Ingres supports multiple data page sizes. This article will explore some of the issues in implementing this feature, as well as list some of the pros and cons in using it. This is not intended to be an all-inclusive list of every pro and con, nor is it intended to be a step-by-step implementation guide. Rather, it is intended to provide some important basic information to assist with the decision making process on whether or not this new feature will benefit your site.

The Basics

The traditional 2 KB data page is still available and remains unchanged (it is still the default page size), but now it is also possible to create data pages that are 4 KB, 8 KB, 16 KB, 32 KB and 64 KB in size. These pages are used to store data from tables and/or indexes, both on disk and in memory (i.e., the DMF cache). Since the I/O subsystem is the slowest subsystem in a computer, the goal for performance tuning is to minimize I/O. Larger pages could potentially help accomplish this (just as making indexing changes, changing storage structures, rewriting queries, reconfiguring the DMF cache, etc. could reduce the overall amount of I/O within an installation). Larger page sizes could also potentially provide large savings in disk space. Remember, a row cannot span a data page, so a row that is 1,005 bytes in a 2KB page is really consuming the entire data page.

Having a basic understanding of how data pages are implemented will provide the foundation for making informed decisions about how best to implement this feature.

Getting Started

First the specific cache size will need to be enabled. Configuring the installation to use the larger data pages is fairly easy to do using Configuration by Forms (CBF). Which page size(s) should you use? Unfortunately there is no one right answer for this. It is important to remember that one page size may not be ideal for all tables and therefore multiple page sizes may be implemented. Most likely you will find that the ideal configuration consists of two or three page sizes. We have created an Excel spreadsheet to help determine which page size is best, and it is available for free download from our website. This spreadsheet takes into account the various limitations that are listed later in this article.

Next, that cache should be configured. Selecting the ideal configuration requires benchmarking to select the optimal size, so initially just pick values that appear reasonable. At this state of the process it is more important to perform an "apples to apples" comparison than it is to fine-tune the DMF cache. Some issues to consider are maximum I/O size available (which is dependent on many things, such as: computer; operating system; physical disk devices; file systems, etc.), available RAM, and single page vs. group page utilization (the trace POINT DM420 can help determine this). Baseline data should consist of table/index size (in MB), response time for specific queries, evaluation of Query Execution Plans (QEPs), and evaluation of the impact on the I/O subsystem (using tools such as "sar –d", "iostat" and "monitor disk", or DBAnalyzer from Database Management Technology). This data can then be compared against both the 2 KB pages and the other page sizes to determine which size is best for your specific environment.

Configuration Considerations

It is very important to think about the group buffer configuration. Normally with the 2 KB buffers we recommend not changing the group size (i.e., the number of pages that each group buffer will consist of) from the default of 8 pages. While larger I/Os may be possible, the impact on the optimizer generally outweighs that benefit (the optimizer will often skew towards table scanning execution plans). This *rule of thumb* is not true with larger page sizes, though. Our *rule of thumb* for >2 KB pages is to configure the group buffers to be between 16 KB and 32 KB in total size. That means that for 32 KB and 64 KB cache sizes we will typically configure the "dmf_group_size" parameter to have a value of zero since the single page buffers are large enough already. The key is to try various configurations and see what works best in your environment.

Another configuration issue is whether or not to use separate "chunks" of physical memory using the "dmf separate" parameter. We recommend setting this to a value of "ON", to use separate physical memory for each page size. In UNIX the DBMS Server will not start if the amount of memory that it tries to allocate is greater than the maximum shared memory segment size (the OS kernel parameter SHMMAX on most SVR4 machines). That can happen very quickly when several DMF Page caches are "lumped together" in a single shared memory segment, especially when configuring very generous DMF caches.

During configuration it is possible to set the "default_page_size" parameter to a value other than 2KB. While we have tested this and it works, we recommend controlling the use of larger page sizes manually and will leave this parameter alone. While it is possible to turn larger page size caches on and off, it is not possible to turn the 2 KB page cache off (due to the system catalog's use of 2 KB data pages).

Another new (and nice) configuration feature is the ability to set the maximum tuple (row) length. The default size is 2008 bytes. The following is a list of the maximum tuple length for each page size (page size / max row size): 2 KB / 2008, 4 KB / 3988, 8 KB / 8084, 16 KB / 16276, 32 KB / 32660, and 64 KB / 32767. This can be especially valuable when migrating data from another product (e.g., DB2), designing a data warehouse, denormalizing data for archival purposes, etc. Since this parameter can affect previous work (through recovery and journaling) it is important that this value be increased only and not decreased.

Once the cache has been configured and the DBMS server(s) restarted it is possible to use the new page sizes. A PAGE_SIZE clause has been added to the DDL statements CREATE TABLE, CREATE INDEX, MODIFY and DECLARE GLOBAL. Valid page sizes are 2048, 4096, 8192, 16384, 32768, and 65536. Please note that it is not necessary to unload and reload your database to take advantage of these features! It should also be noted that tables and indexes are not required to have the same page size. These new pages have a different physical format than the 2 KB page (described in the "Tables" section of the OpenIngres 2.0 Database Administrator's Guide (Chapter 17)). These pages contain more page overhead (80 bytes) as well as more tuple (row) overhead (26 bytes). The reasons for this additional overhead are to provide 64-bit TID support in the future as well as to provide row-level locking now.

Page Layout and Limitations

Currently these pages use what appears to be the same 32 bit TID scheme that is used with 2 KB pages. While this scheme allows for much larger tables (2^23 pages* page_size), it does not provide for more than 512 rows per page. The following is a matrix of the maximum number of rows possible per page. This test was conducted by creating a table with a single column defined as "char(1) not null not default".

Page Size	Max # Rows Per Page
2 KB	511
4 KB	138
8 KB	279
16 KB	512
32 KB	512
64 KB	512

This may not be much of an issue when using large tuple sizes, but is definitely an issue that needs to be evaluated prior to implementation. Given the large tuple overhead and limited row capacity per page it is generally not advisable to use this feature on very small rows.

Locking Considerations

Row-level locking is a newer feature that can be beneficial to some environments (usually those with concurrency problems due to schema and/or transaction design, where it is not feasible to redesign the database and/or those programs). This feature is only available for the larger page sizes and is set with the "set lockmode" statement. When row-level locking is used it is very important to increase both the locks per transaction and the maxlocks values (or concurrency will likely decrease instead of improve).

Advantage Ingres / Ingres II provide significantly more locks (up to 2 billion within an installation) to accommodate this feature.

Pros & Cons

Some of the PROS of using > 2 KB page sizes include: the ability to have a larger row size; the ability to have row-level locking; the potential to improve performance due to decreased I/O; and the potential to save disk space.

The CONS include: limited row capacity; increased row overhead; the potential to have a negative impact on performance if improperly configured; and the potential to consume more disk space. Also, if tools like "auditdb" are important to your site make sure to test them thoroughly on the larger sized pages.

Summary

This is a great feature for many environments. As with any type of change it is important to define the goal of the change, collect baseline metrics, benchmark the change, and only implement if it makes good business and technical sense.

Click here to view a slide presentation on this topic from a conference presentation.

Please refer to the Advantage Ingres / Ingres II Database Administrator's Guide for more information on the topics discussed in this article.

About the Author

Chip Nickolett, MBA, PMP is the President and Founder of Comprehensive Solutions. He was a Senior Consultant with the Ingres Products Group of ASK, as well as the top-ranked Ingres Consultant in North America and Consulting Manager with Computer Associates.

This article was originally published in the Spring, 1999 North American Ingres Users Association (NAIUA) newsletter.

Let Us Help You Succeed!

Call today to discuss ways that Comprehensive Solutions can help your organization save money and achieve better results with your IT projects.   We provide the *confidence* that you want and deliver the *results* that you need.

View our "Ingres Support" Brochure

View our "Confidence" Brochure

Back to White Papers

Back to Services

Comprehensive Solutions
4040 N. Calhoun Road
Suite 105
Brookfield, WI 53005
U.S.A.

Phone: (262) 544-9954
Fax: (262) 544-1236