IBM Informix Database Software

New Features in Informix 11.70

The Technical White Paper

October 12, 2010

INTRODUCTION

Informix 11.70 - The Beginning of the Next Decade of IBM Informix...

Executive Summary

With the diverse set of Informix® customers across different industries and enterprise sizes, we wanted to ensure that 11.70 focuses on the core strengths of Informix while significantly enhancing specific areas of growth and demand.

Informix 11.70 raises the bar as a viable, low-cost RDBMS alternative for most custom small to enterprise level business applications. It provides the industry's most comprehensive set of high availability options in a platform independent environment while providing increasing value for embeddability, security, application development and data warehousing capabilities.

Besides enhancing availability, reliability, and scalability, we ensured continuous improvements in overall performance, ease of use and administration, all considered fundamental characteristics to our success over the years.

In this release, we introduce the use of **Flexible Grid**, further enhancing our ability to manage and distribute workload across disparate clusters. We are providing a true **Rolling Upgrade** capability where users of High Availability servers can plan their upgrades without any scheduled downtime. We are also allowing the important ability to **propagate DDL** as well as DML in this release. Informix Flexible Grid is easy to install and setup and easy to administer. In addition, adding or removing nodes or clusters to meet seasonal or peak business demands is a simple operation.

With the increasing demand for analytics of transactional and historical data, 11.70 significantly enhances its ability to handle warehousing workload through its support for **Star Join** queries and management of **Time-Cyclic** data. Informix customers currently running on XPS or Red Brick should seriously consider the warehousing capabilities now present in Informix 11.70.

Already considered a leader in its embedded capabilities, Informix 11.70 now provides **Storage Provisioning** to automate storage allocation for uninterrupted applications and recovery. For easy deployment across hundreds and possibly thousands of instances, 11.70 introduces **Deployment utilities** that allow not only Informix but also pre-built databases to be installed seamlessly.

No matter the capabilities of a DBMS, the key to its usage and long-term viability is its ability to work with applications and solutions built for it. With that in mind, 11.70 significantly increased its **Compatibility** capabilities in its SQL language and also its **Interoperability** with popular Open-Source products such as **Drupal, Hibernate** and others.

With industry and government compliance standards comes the need for proper auditing and authentication methods, 11.70 now provides **Selective Row-Level Auditing** to minimize audited data being collected and thus analyzed. **Trusted Context** allows for much easier and more fine-grained authentication for applications often controlled by middle-tier application servers.

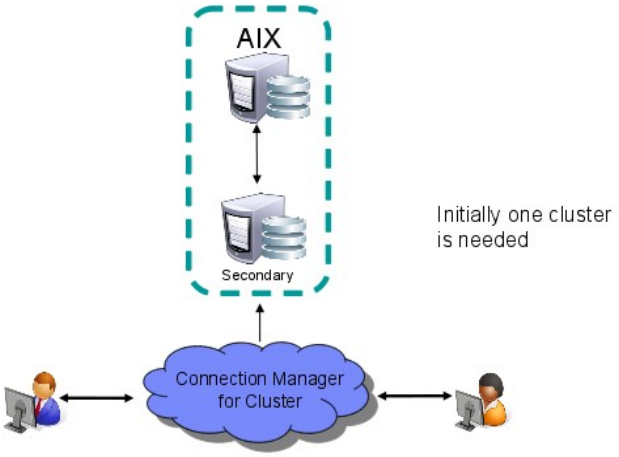
“Perfection is our goal, excellence will be tolerated.”
- J. Yahl

WHAT'S NEW?

FLEXIBLE GRID

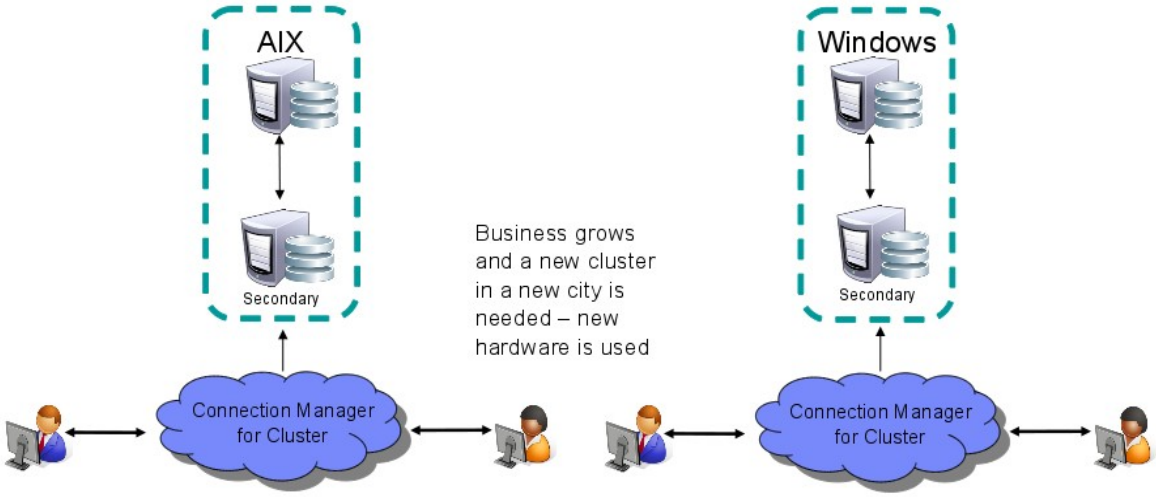
As your data processing requirements grow you may end up collecting a disparate group of data servers to address various needs. Often data and functionality will be duplicated among these servers and administration can become very difficult. Some servers may be running in the red while others remain mostly idle. The Informix Flexible Grid offers a solution to these problems. With the flexible grid you are able to administer your data servers as if they were a single entity. In addition the flexible grid lets you balance your workload across all your servers regardless of hardware, operating system, or version of Informix.

The foundation of the flexible grid is Enterprise Replication (ER). In 11.70 a significant amount of work has gone into ER to remove previous limitations, enhance the product, as well as to greatly simplify the usage. As an example of a Flexible Grid, consider a configuration using Enterprise Replication and MACH 11 Shared Disk Secondaries (below). Users use the Connection Manager to connect to the cluster.



Single cluster with ER and Mach 11 functionality

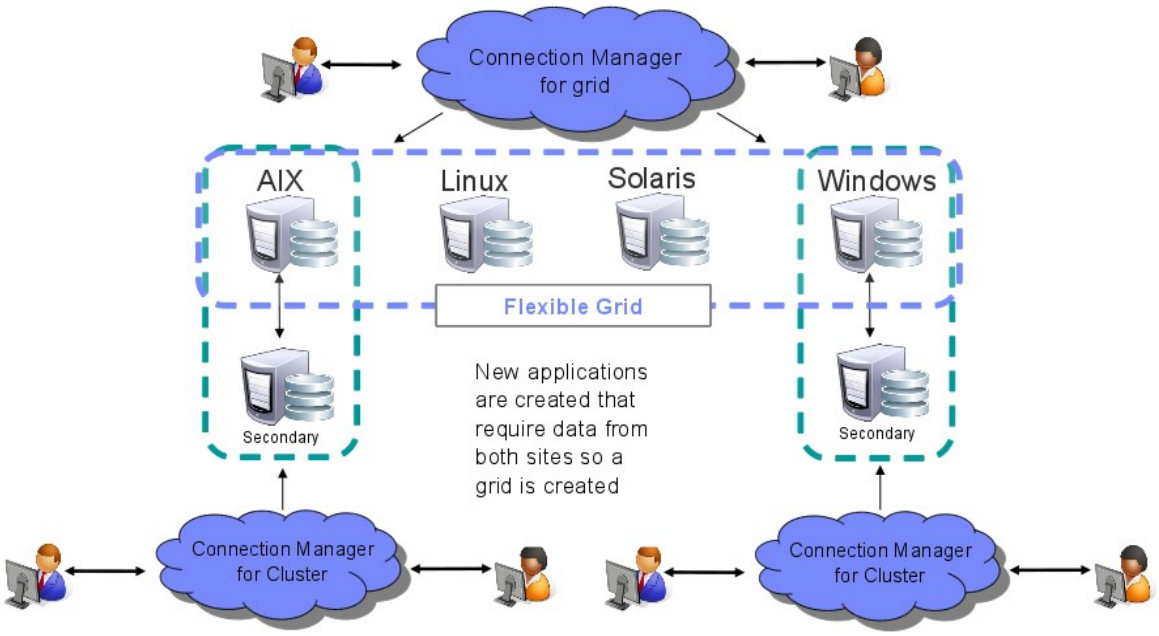
As business needs grow, the demand for a second cluster in a remote city is established. This cluster is established on a different hardware platform. Local requirements determine access to a different set of database tables. The second figure reflects this new configuration.



Second cluster, also using ER and MACH 11 functionality

As business needs continue to evolve, and access is needed to contents from both clusters, a Flexible Grid is created. Users access the grid connection manager to access the entire grid.

Users needing access only to information on individual clusters can still access the cluster's connection manager. The new solution offers the “global” application users quick and easy access to all information available on the Grid. The third figure shows the resulting configuration.



Informix Flexible Grid solution

Grid Functions

Grid Based Replication provides a means of propagating DDL and server administration commands across multiple nodes. It replicates the execution of a statement rather than just the results of the execution and provides a means of supporting the connection manager on top of Enterprise Replication. You don't have to worry about replicating data using ER without a primary key and Grid Based Replication provides the ability to turn on/off ER replication within the transaction, not just at the start of the transaction.

Defining The Grid

The Grid is based on using Enterprise Replication to control the movement of data and management commands though the system. You first need to use the *cdr* command to define the group of nodes in the grid. If you are exclusively using Informix 11.7 for all your server instances then you can use the *--all* parameter, otherwise in a mixed environment you will need to name the individual nodes.

```
cdr define grid <grid_name> --all

cdr define grid <grid_name> <node1 node2 ...>
```

Enabling The Grid

Now you must define the nodes within the grid which can be used to perform a grid level operation. You can also determine which users are allowed to perform the grid operations.

```
cdr enable grid –grid=<grid_name> --user=<user> --node=<node>
```

Disabling The Grid

If you need to it is simple to remove a node or user from being able to perform grid operations

```
cdr disable grid –grid=<grid_name> --node=<node_name>

cdr disable grid –grid=<gird_name> --user=<user_name>

cdr disable grid –g <grid_name> -n <node_name> -u <user_name>
```

Managing The Grid

In order to perform DDL operations at a grid level, you must first connect to the Grid on a Grid Enabled Node and as a Grid Enabled User. This is done by executing the built-in procedure

```
ifx_grid_connect(<gridName>, <autoRegister>, <tag>);
```

Where autoRegister is set to 1 if we want to register the DDL with ER and “tag” is an optional tag associated with any grid commands you perform. The tag can be used to make it easier to monitor the success/failure of grid operations. DDL operations will be performed on the target nodes within the Grid according to these rules.

- Within the same database as on the source
- By the same user as was on the source
- Using the same locale as on the source

The Grid connection is terminated by performing the built-in procedure *ifx_grid_disconnect().* In addition to DDL propagation, you can perform the execution of a procedure, function, or statement as a grid operation. A function will have a return value which is saved in the syscdr database and can be viewed using **cdr list grid** command**.**

Setup And Manage A Grid For A Replication Domain

You can create a grid of interconnected replication servers in a domain. You can use the grid commands to easily administer database servers defined within the grid. When you run the following types of commands from the grid, the commands are replicated to all servers in the grid:

- Database schema updates
- Administrative tasks
- Routines

Quickly Clone A Primary Server

A new utility, *ifxclone*, allows you to quickly clone an existing server as a standalone ER server or as an RSS (Remote Standalone Secondary) server. This utility automates the procedure of performing an archive, transferring the archive to a different system and then initializing the secondary system. If you want to produce an HDR clone then you must first make an RSS clone and then promote that to be an HDR server. The port and IP address need not be numeric but can also refer to a system port name or hostname respectively.

```
Usage:
ifxclone -SIPtip                    # Clone a server
-h      --help                    # Display this output
-S      --source=<name>             # Name of the source node
-I      --sourceIP=<IP>           # IP address of source node
-P      --sourcePort=<port>         # Port number of source
-t      --target=<name>           # Name of target server
```

```
-i     --targetIP=<IP>                # IP address of target
-p      --targetPort=<port>             # Port number of target
-d      --disposition=[RSS|ER]          # Clone disposition
                                #   (default:standard)
-s      --size=[tiny|small|medium|large] # Configuration size
-c      --configParm="PARAMETER=VALUE"   # Configuration override
-L      --useLocal                    # Use the local config an
                                #   sqlhost
-T      --trusted                     # User/password not required
-f      --file=<path_to_file>           # Path to Deployment Utility
                                #   configuration file
```

Note that the DIRECT_IO configuration parameter must be 0 for both source and target servers. This utility can be used to allow an HA cluster to be upgraded to a new version of the database server while keeping the database available a all times. See the Upgrades and Migration using ER section below.

Enterprise Replication Enhancements To Support Flexible Grid

Synchronize All Replicates Simultaneously

While repairing replicates with the *cdr check replicateset --repair* or *cdr sync replicateset* commands, you can now specify to repair all replicates instead of running the command once for every replicate set. Use the *--allrepl* option instead of the *--replset* option. If your replicates are do not belong to replicate sets, you can still use these commands with *--allrepl* to repair all replicates simultaneously.

Improved Enterprise Replication Error Code Descriptions

Enterprise Replication return code documentation now includes useful descriptions and user actions.

Control Enterprise Replication Capture From Within A Transaction

You can now control which statements within a replicated transaction are replicated by using the *ifx_set_erstate()* procedure.

Replicate Tables Without Primary Keys

If you do not want to have a primary key, or want to be able to update the primary key, on tables replicated by Enterprise Replication, you can use the ERKEY shadow columns in place of a primary key.

If you create a replicated table through a grid, the ERKEY shadow columns are created automatically.

Set Up Replication Through A Grid

When you create a table through a grid, you can specify to create a replicate and replicate set for the table and start replicating the data in the table.

Automating Application Connections To Enterprise Replication Servers

You can use the Connection Manager to direct application requests to the appropriate Enterprise Replication server. If you have created tables through a grid with replication enabled, you can route client connections to Enterprise Replication servers based on the quality of replicated data and transaction latency.

Upgrades And Migration Using ER

The CONVERSION_GUARD configuration parameter is now set to 2 by default. If an error occurs while you are upgrading to the new version of the database server, the upgrade continues. Previously, the default setting was 1 and the upgrade stopped when an error was encountered.

You can now upgrade the Informix database on a high-availability cluster without incurring any down time. First you convert the primary and the secondary server to standalone Enterprise Replication (ER) servers. You then upgrade the software on the secondary server, stop ER, and then clone the server using the *ifxclone* command. You can perform the upgrade while the servers are actively processing client requests because ER supports replication between dissimilar versions of the Informix database software.

The following prerequisites apply when upgrading Informix database software on a cluster:

- Non-logged databases are not supported.
- Raw or unlogged tables are not supported.
- Typed tables are not supported unless the typed table contains a primary key.
- UDTs that do not support ER are not supported.
- For versions of Informix software earlier than 11.50xC7, converting a primary and secondary server pair to ER is not supported if a table does not have a primary key.

High Availability Enhancements To Support Flexible Grid

Transaction Completion During Cluster Failover

Previously in a high-availability cluster configuration, the failure of the primary would cause all the active transactions across the cluster to be rolled back. All SQL clients would then have to retry their transactions. Now, those transactions on surviving secondaries will run to completion after the failover to a new primary server has completed.

This new transaction survival feature is controlled by the FAILOVER_TX_TIMEOUT configuration parameter. This parameter controls the maximum number of seconds to wait after failover before rolling back open transactions. The default setting of 0 disables the feature and the setting should be the same across all servers in the cluster. The value can be changed dynamically using the *onmode -wf* command.

When transaction survival is enabled, the failover server must be able to contact the remaining secondary servers to synchronize and resume any open transactions. Similarly, the surviving secondary servers must be able to establish connections to the failover server to re-send any pending transactions. The FAILOVER_TX_TIMEOUT configuration parameter specifies how long the servers will wait for synchronization before taking action. On the failover server any open transactions that could not be synchronized will be rolled back, on the remaining secondary servers any open transactions will receive an error.

It is recommended to use a failover configuration of **SDS+HDR+RSS,0** when setting up the Connection Manager to make best use of this feature. This is because the failover server must be the one with the most recent log position.

Monitoring High-availability Servers

You can now monitor the status of the primary server and all secondary servers in a high-availability cluster by using a single command: *onstat -g cluster*. This command is an alternative to
the individual commands: *onstat -g dri*, *onstat -g sds*, and *onstat -g rss*.

Running DDL Statements On Secondary Servers

This new version of Informix has removed the previous limitation of only being able to run DML statements on secondary servers. Most DDL statements such as CREATE, ALTER, and DROP are supported on tables from any updatable server in a high-availability cluster.

Many of these new Flexible Grid features are documented in the *IBM Informix Enterprise Replication Guide.* Other features are documented in the *IBM Informix Administrator's Guide.*

EASY EMBEDDABILITY

Informix is the industry leader in DBMS embeddability functionality. Informix can be embedded deep within an application; it can be installed, initialized and running for extended periods of time, all without a DBA and the end user knowing that a DBMS is even there! Informix 11.70 adds additional functionality to further enhance the functionality in this area.

Informix 11.70's new embeddability toolkit includes a Deployment Assistant (DA) for creating database instance snapshots for deployment, a Deployment Utility (DU) for deploying these snapshots, sample DU configuration files, sample deployment scripts and a tutorial showing examples of the end to end process of creating a snapshot for deep embedding, and then installing and configuring the snapshot via a script. Additional Informix 11.70 features provide deeper customization of the base DA and DU functionality to support extensive options for creating server instance snapshots and utilities for deploying these within an application framework.

Informix 11.70 continues to extend the Informix capability for silent and scripted installation, a key feature for embedding and deploying Informix in environments where no administration is available (or needed). The new enhancements include a silent install response file which allows an administrator to create a file which specifies a detailed, non-trivial installation for later use or in a remote environment. The installation (via GUI, command line or silent mechanism) includes more options for specifying sizes of dbspaces, sbspaces, buffer pools, etc. Informix 11.70 also provides the infrastructure to initialize the configured server during the installation. This key new enhancement allows for immediate use of a custom-tuned server following installation. Naturally this can also be used with the silent install and template and thus having a custom-tuned server instance up and running following a hands-free install. These additions could be very useful for application developers and ISVs wanting to distribute their applications without awareness of the underlying database, or for corporate IT architects needing to distribute an infrastructure application without dedicated DBA assistance.

Let's take a deeper look at each of these enhancements for in the area of embeddability and product installation.

Enhanced Utility For Creating Informix Instance Snapshots

Deployment Assistant Simplifies Snapshot Capture And Configuration

The Deployment Assistant utility allows you to easily package snapshots of Informix instances and/or their data, in preparation for deployment. It also allows for reduction in the footprint of the packaged instances to the user's minimum desired configuration. This utility runs in both command line and GUI mode, and makes this series of operations to create a copy of your database instance much easier than the previously manual-only process. Options in the Deployment Assistant provide for capturing the instance, but also to configure the snapshot for eventual deployment, and compressing it for distribution.

The program *ifxdeployassist* ($INFORMIXDIR/bin directory) is the deployment assistant binary executable. It runs in GUI mode if started without options. If started with the **-c** option, it will start in command line mode, which can be used directly or as part of a scripting environment. The GUI mode should be used instead of the command line for extensive configuration on the target image, for example capturing a reduced-footprint snapshot that contains only specific features.

Create A Snapshot In A Command-line Interface

The command line interface can be used to perform all the Deployment Assistant function, except for the reduction of the instance's snapshot size (footprint). The GUI version of the utility must be used for customization of this option.

Here is an example of the command line version of *ifxdeployassist.*

```
$ ifxdeployassist -cvi ol_informix1170:localhost:1170 2>/tmp/deploy.log
INFORMIXSERVER:     ol_informix1170 (11.70.1)
INFORMIXDIR:        /products/11.70/

Packaging Snapshot...
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||/
The snapshot was packaged successfully.

SNAPSHOT LOCATION
Instance snapshot:
  /home/gcostanza/Panther/test/ol_informix1170.tar.gz
Data snapshot:
  (not included)
```

This command creates a snapshot of the *ol_informix1170* instance only (no data) from the local host, and saves it to the default location (the current directory) as a compressed tar file.

Clone A Server Using The Deployment Utility

The simplified method of packaging snapshots of Informix instances and/or data using the Deployment Assistant utility greatly reduces the work required by DBAs to perform this task manually. The configuration options allow DBAs to easily reduce the footprint of the snapshots by displaying file associations in a concise and easy to interpret manner, thus providing for a reduction of the packaged instance sizes to the user's minimum desired configuration. The packages are then ready for use by the Deployment Utility (*ifxdeploy)*.

Enhanced Utility For Deploying Informix Instances

In the 11.70 release, enhancements have been made to the deployment utility (*ifxdeploy* - initially released as part of Informix 11.50.xC6) to rapidly deploy a configured database server instance to multiple computers. The -**start** option deploys and starts the instance in a single operation so that you can silently deploy and start a database server in the target environment.

An example showing usage of the *ifxdeploy* utility:

.

```
ifxdeploy -verbose -config ifxdeploy.conf -f demo_on.tgz \
 -rootpath /data/demo_on/online_root -relocate /data/demo_on \
 -start
```

Shown in this example:

- Deployment of a saved database instance from the compressed gzip file - demo_on.tgz
- The image is relocated to the directory in the target environment - /data/demo_on
- The new rootpath is set to /data/demo_on/online_root
- The new instance is started after deployment
- Additional parameters are passed via the *ifxdeploy.conf* file

- Verbose massages are printed

The *ifxdeploy.conf* file contains new parameters so that you can run the deployment utility with fewer command-line options. In the example above, all of the command line arguments can be replaced with equivalent parameters in the *ifxdeploy.conf* file. Since options can be specified in the environment, in the configuration file, and on the command line, precedence has been established:

- If the same options are in the configuration file and on the command line, the command line takes precedence.
- If the same options are in the configuration file and in the environment, the configuration file takes precedence

Deployment Utility Connectivity Parameters

You can now specify the deployed instance's Informix SQL protocol and Distributed Relational Database Architecture (DRDA) network ports for the SQLHOSTS connectivity file. Set the SQLIPORT and DRDAPORT parameters in the deployment utility configuration file. Previously, you set the SQLI port in the *ifxdeploy.conf* file by using the PORT1 parameter, or by specifying both values on the deployment utility command line with the **-sqliport** and **-drdaport** options.

Enhanced Decompression Support In The Deployment Utility

Version 11.70 Deployment Utilities includes enhanced support to compress and to extract compressed snapshots using various standard compression programs. The deployment assistant now supports additional archive formats, including BZIP2, GZIP, TAR, and ZIP.

The deployment utility automatically extracts compressed snapshots. In the previous release, it was necessary to explicitly specify the -**extractcmd** option to extract from BZIP2 and GZIP formats.

Generating A Customized Configuration File Setting

The Deployment Utility (*ifxdeploy*) has been further enhanced in Informix 11.70 with the addition of the -**autorecommend** option which signals the generation of an alternate server configuration file, containing optimal settings for certain configuration parameters. This setting takes advantage of new Informix 11.70 technology which takes into account hardware performance for calculating parameter values (for example – disk read/write speeds).

To support this option, the Deployment Utility configuration file template, *ixdeploy.conf*, has been extended with a new section for the specification of various server settings that should be considered when generating the configuration for the new deployment. The new options are shown:

```
BEGIN AUTORECOMMEND
  MAXCPUS   (Max number of CPUs to use, Default = 1)
  MAXDISK   (Max amount of disk space to use, Default = 2048 MB)
  MAXMEM    (Max amount of memory to use, Default = 512 MB)
   MAXUSERS        (Max number of OLTP users, Default = 32)
  MAXDSUSERS      (Max number of DSS users, Default = 4)
  RTO_SERVER_RESTART (Real time objective, Default = 60 secs)
END AUTORECOMMEND
```

The alternate configuration file, generated using the options provided above, is saved as $INFORMIXDIR/etc/$ONCONFIG.autorec

New Archive Format Support And JRE Requirements

The Informix 11.70 Deployment Assistant (*ifxdeployassist*) supports the creation of archives in the following compression formats: BZIP2, GZIP, TAR, and ZIP. The Deployment utility (*ifxdeploy)* correspondingly, supports these same formats during deployment.

These utilities are developed using the Java™ language, and so have a dependency on the version of the Java Runtime Environment (JRE) that is available. These utilities require JRE version 1.6 (6.0) or later. If the operating systems are compatible, the Informix server's JRE can be used to satisfy this requirement. This is located at $INFORMIXDIR/extend/krakatoa/jre/bin.

Configuring An Informix Server Instance During Installation

Silent Installation Response File Template

Informix 11.70 extends the installation response file functionality, which is very useful when installing the database server in embedded or custom configurations where it may not be desirable to display an installation user interface. As in previous releases, you can generate a response file by recording an installation setup done in interactive mode. In addition, you now have the option to set configuration parameters by editing the response file in any text editor instead of passing command-line options. A response file facilitates installation of IBM Informix products in silent mode. The response file contains installation settings for a product and its features.

Informix 11.70 comes with an example response file, *bundle.properties*, for the server and related products. This can be copied and edited to customize options and locations. An example of installing in silent mode using the response file:

```
ids_install -i silent -f <response_file_path>
```

An example of installing in silent mode without the response file (the license terms are accepted via the command line and a non-default installation directory is specified):

```
ids_install -i silent -DUSER_INSTALL_DIR=<install_location> \
   -DLICENSE_ACCEPTED=TRUE
```

If you want to use the same installation settings in more than one directory or computer, first install a product in GUI or console-mode to capture the installation settings in a response file. To do this, run the product installation command with the `-r` option and specify a full path name. Do not name your response file bundle.properties or ids.properties. Use your .properties file to perform a silent installation elsewhere.

More Consistency Among Installation Scripts

The Informix 11.70 installer program is now common across all supported platforms. Previous releases contained two installer programs - a Windows®-only, Installshield-based installer, and a Java based installer for Linux® and UNIX® platforms. The new single Java-based installer provides the same look and feel across all platforms, making it easier for users installing on

several different platforms and also gives a single image for generating customized script-based installations.

Quicker Typical Installation Setup

The Informix 11.70 installation application provides many more default settings which should speed installation in typical scenarios and to also provide for a “smarter” configuration. The new installer, launched with the *ids_install* command, makes it easier to install and configure Informix products and features and to create custom installers for subsequent or remote installations. Some of the new functionality includes:

- The “typical” installation profile has improved default settings. This option can be used to quickly install all of the products and features in the software bundle, with pre-configured settings.
- The “custom” installation is smarter and has more options than in the previous releases. This can be used to customize which features and components are installed. Specific products can be selected/de-selected, and options such as role separation can be enabled.
- Regardless of which installation option is used, a server instance can be specified, configured and initialized and be ready to use after installation. Further customization on the instance configuration can be performed prior to initialization using a custom installation file.

Advanced Functionality In Custom Installation Setup

A very useful new feature of the Informix 11.70 installation is the ability to specify, configure, and then instantiate a server instance. In previous versions, the installer could create an o*nconfig* file, used to store server instance parameters, for further instantiation. Informix 11.70 takes this one step further by including the support to instantiate the new server instance during the installation using the specified configuration parameters, either the default settings, or a set of customized parameters. The instance can also be tuned to the host machine environment during the installation configuration.

There are two options for creating the server instance. A simple instance using default parameters is the basic option. A “tuned” instance can also be created, with customizations ranging from simple modifications to the default settings to fairly complex configurations. The configuration can be customized for single or multiple CPUs and a custom memory footprint.

Additionally there can be up to six different dbspaces created upon initialization (root dbspace, physical log and logical log, all based on tuning parameters; data, smart large object space and temporary dbspace, all user configured). The user is presented with the opportunity to adjust the sizes and locations of each of these dbspaces during the “Space Review” phase of configuration. Some of the environment factors that can be provided in order to optimize the instance for the environment include CPU processor speed, number of CPUs, total physical memory allocated to the instance, I/O speed on the selected volume(s), type of server instance (DSS/OLTP), concurrent user counts (OLTP and DSS), desired transaction logging status, and recovery time objective for OLTP applications. The installer program takes these values into account when generating the onconfig file for the new instance.

Another advanced feature of the new Informix 11.70 installer is the ability (on Unix and Linux environment) to install the server as a “normal” user, and also create a script for the super user to execute at some later time and location. This is useful for redistribution and for environments where super user access is strictly controlled. This option, called the legacy install option, is invoked:

```
ids_install -i [console | gui | silent ] -DLEGACY=TRUE
```

The generated installation script will be found in $INFORMIXDIR/RUNasroot.installserver. It will set the appropriate permissions on executables and shared libraries, as normal.

Enterprise Replication Control From The Embedded Application

Handle A Potential Log Wrap Situation In Enterprise Replication

You can configure what actions occur automatically if a potential log wrap situation is detected during replication. A potential log wrap situation occurs when the log processing by Enterprise Replication lags behind the entries in the current log so that the Enterprise Replication replay position might be overwritten. In previous releases you could add logical logs or Enterprise Replication would block user sessions until the potential for log wrap diminished.

Specify one or more of the following actions, in prioritized order, with the CDR_LOG_LAG_ACTION configuration parameter:

- Compress logical logs in a staging directory
- Add logical logs dynamically
- Prevent blocking user sessions, but potentially overwrite the replay position
- Block user sessions (default)
- *Shut down Enterprise Replication*

Repair Replication Inconsistencies By Time Stamp

The *cdr check –R* or *sync* command is used to repair data inconsistencies found in source and target database tables for a given replicate. Running just the *cdr check* command prints the statistics while *cdr check* with repair and sync options repair inconsistencies found during the scan. In previous releases, you had to choose a master server to act as the correct version of the data and the repair made all the other participants' data match the master server's data. However there are a number of limitations with this approach:

- It cannot be used in a multi-master environment
- It does not take into account row timestamp
- You need to take care of *deletewins* conflicts
- The row with latest timestamp needs to be replicated.
- You must check the delete table whether the row is already deleted.

Now, If you have a replication domain with multiple master servers and your conflict resolution rule is time stamp or delete wins, you can repair inconsistencies based on the latest time stamps. This works well with another new ER feature, the ability to temporarily disable an ER server.

To repair by time stamp, use the *cdr check replicate* or *cdr check replicateset* commands with the *--repair* and *--timestamp* options and omit the *--master* option.

Temporarily Disable An Enterprise Replication Server

You can temporarily stop replicating data to and from a replication server by using the *cdr disable server* command. The replication server stops queuing and receiving replicated data. Other replication servers in the replication domain also stop queuing data to the disabled replication server. However, because deleted row information on the disabled replication server is saved, you can quickly and accurately synchronize the data with a time stamp repair when you run the *cdr enable server* command.

Enhanced Server Control From The Embedded Application

In addition to the following features, the new Automatic storage provisioning function detailed below on Page 26 will also be of interested in creating a more autonomic environment for embedded applications.

Notification Of Corrupt Indexes

If an index becomes corrupt an alert is logged indicating which index needs to be rebuilt. Previously it was only possible to know of a corrupted index by manually running an *oncheck* command. The alert can be easily viewed through OAT or by examining the *sysadmin:ph_alert* table.

Easier Event Alarm Handling

Event alarms now have a unique identification number for each specific message. You can write scripts to handle event alarms based on the unique identification number that corresponds to each specific message in an alarm class. Previously, event alarm handling scripts had to combine the class ID and the specific message.

Alerts For Tables With In-place Alter Operations

Tables that have outstanding in-place alters are noted in the *sysadmin:ph_alert* table which can be most usefully monitored using the Open Admin Tool (OAT) It is useful to keep note of such tables as there is the potential for some small performance drop and also the need to know which tables need to be brought up to date in case of a version upgrade.

Controlling Disk Unexpected Initialization

The use of oninit -i to completely initialise the system was open to accidental usage that could destroy an existing instance. You can use the new FULL_DISK_INIT configuration parameter to prevent the major problems that can occur if you or someone else accidentally initializes your instance or another instance when the first page of the first chunk (page zero) exists at the root path location. Page zero, which is created when Informix is initialized, is the system page that contains general information about the server.

The FULL_DISK_INIT configuration parameter specifies whether or not the disk initialization command (oninit -i) can run on your Informix instance when a page zero exists at the root path location. When this configuration parameter is set to 0, the oninit -i command runs only if there is not a page zero at the root path location. Even if you rspond 'y' to the "Do you wish to continue (y/n)?" prompt it will not initialise and you will see the following in the log:

```
DISK INITIALIZATION ABORTED: potential instance overwrite detected.
To disable this check, set FULL_DISK_INIT to 1 in your config file and retry.
```

If you change the setting of the FULL_DISK_INIT configuration parameter to 1, the oninit -i command runs under all circumstances, but also resets the FULL_DISK_INIT configuration parameter to 0 after the disk initialization.

The *Ifxcollect* Command Line Tool

This tool collects diagnostic data that is useful for troubleshooting problems such as assertion failures.

```
ifxcollect: <options>
        General Options
```

```
-r <Num Times to repeat Collection>
-d <Seconds for dealy between Collection>
-y - Answer yes to all prompts
-V Version Information
-version Extended Version Information

FTP Options
-f - FTP the data collection
-e <Email Address>
-p <PMR Number>
-m <Machine to ftp to>
-l <Directory Location for ftp>
-u <Username for ftp>
-w <Password for ftp>

Example FTP
-f -e user@company.org -p 9999.999.999
-f -e user@company.org
-f -m machine -l /tmp -u username -w password

Collection Options
-c ids -s general
        General Collector For All Informix Family Products
-c af -s general
        General Collector For Assertion Failures
-c er -s general
        Collect general information For ER
-c er -s init
        Collect information For ER Initialize Issues
-c performance -s general
        Collect general information for performance Issues
-c performance -s cpu
        Collect information for cpu utilization issues
-c onbar -s archive_failure
        Collect information for onbar archive failures.
-c onbar -s restore_failure
        Collect information for onbar restore failures.
-c ontape -s archive_failure
        Collect information for ontape archive failures.
-c ontape -s restore_failure
        Collect information for ontape restore failures.
-c connection -s failure
        Collect information for connection failures.
-c connection -s hang
        Collect information for connection hangs.
-c cust -s prof
        Customer profile
```

Improved Return Codes For The Oninit Utility

Version 11.70 adds enhancements to the *oninit* utility to help improve programmatic usage. Reasons for this are varied, and include the requirement to perform database maintenance from within an embedded database application, creation of a script based utility that starts the server in various conditions, as well as a program performing packaged deployment of the database instance as part of application installation. These changes are characterized as improving consistency in the return codes for server initialization, which can be very helpful for applications which administer Informix in deep embedded environments. Developers can program their applications to react to certain return codes, and then take appropriate action to bring the instance successfully online.

Some of the new return conditions that are now recognized by the *oninit* program include:

- Could not find *libelf* library
- Could not find *libpam* library
- Incorrect command line syntax
- Error reading *onconfig* file
- Error updating *onconfig* file
- Error calculating defaults in *onconfig* file
- Incorrect serial number
- Requesting/calling user does not have proper DBSA role
- Shared memory creation/initialization failed
- Incorrect SQLHOSTS entries

The actual return code values for these conditions are described in the reference material.

Installing Informix Products On Mac OS X

Informix 11.70 is supported on the Macintosh OS X platform. The Macintosh installation image is distributed as a Mac OS X disk image file (.dmg). The recommended installation method is to select the disk image file, click on it to open the application, and then run *ids_install.app* to begin the installation. In addition to the GUI mode, silent and console modes are also supported. Un-installation is performed by dragging the application to the trash can, which is similar to uninstalling other Mac applications.

Removing Installed Informix Client Products

With Informix 11.70, it is now possible to use the uninstall command to remove any client products that have been installed along with the server. In previous releases, each would have had to be uninstalled separately. Use the new *uninstallids* command to remove the server, any bundled software, or both. You can remove specific products by using the following commands, which are in new subdirectories relative to the $INFORMIXDIR/uninstall directory:

- uninstall_server/uninstallserver
- uninstall_clientsdk/uninstallclientsdk
- uninstall_connect/uninstallconnect (formerly uninstallconn)
- uninstall_jdbc/uninstalljdbc.exe or java -jar uninstall/uninstall_jdbc/uninstaller.jar (depending on how you install the JDBC driver)

Backup And Restore Is Now Cloud Aware

With Informix 11.70, the *ontape* utility has been enhanced to back up and restore Informix database data to or from cloud storage. Storing data on the cloud provides scalable storage that can be accessed from the web.

Ontape can now be configured to back up and restore data to or from Amazon Simple Storage Service (S3). Since this is hosted by a 3rd party, an account on the Amazon cloud is required to perform cloud storage backups. See the Amazon web site for instructions about setting up an account. In addition, Java version 1.5 or later is required. Amazon storage can be used to backup database objects that are 5 GB in size, or smaller.

For security reasons, it is recommended that data be encrypted before transferring to cloud storage. Additionally, https secure data transmission should be used when transferring data to and from cloud storage.

Tutorial To Deploy And Embed Informix

The Information Center for Informix 11.70 contains a new "Informix Embeddability Tutorial" which provides a complete step by step example on creation and silently deploying a pre-configured Informix instance with desired footprint on multiple computers. The steps describe how to use the Deployment Assistant utility to configure and create a snapshot of the instance, and how to use the Deployment utility to deploy the instance in an embedded environment. A sample script (*ifx_silent_deploy*) is provided (and discussed) for automating the process. This tutorial contains instructions for both Linux and Windows environments.

The 'Informix Embeddability Tutorial" is contained within the "Embedding IBM Informix" topic in the *Informix 11.70 Information Center*.

New Embedding Informix Dynamic Server Manual

In Informix 11.70, information about deploying embedded Informix instances can be found in the *Embedding Informix* section of the online information center, or in the new *IBM Informix Embeddability Guide*. This guide is new for Informix 11.70, and consolidates all the information on creating and deploying embedded Informix instances from other guides and manuals.

The Deployment Assistant utility, *ifxdeployassist*, and the Deployment utility, *ifxdeploy,* are documented in the *IBM Informix Embeddability Guide.* The new Install features are documented in the *IBM Informix Installation Guide for UNIX, Linux, and Mac OS X* and the *IBM Informix Installation Guide for Windows.* Embedded Replication functionality is documented in the *IBM Informix Enterprise Replication Guide.*

EXPAND WAREHOUSE INFRASTRUCTURE

Recognizing the increasing demand of warehousing capabilities within the Informix community, the Informix 11.70 release strives to provide significant improvements in both query performance and manageability. Together with the Cognos suite of BI products, Informix now provides a complete solution to serve the needs of typical warehouse architectures and Business Intelligence applications including Enterprise Data Warehouse, Data Marts and Operational Data Stores.

For warehouse queries that involve low-selectivity predicates, e.g. WHERE gender = 'M' AND income_level ='HIGH', the support for Multi-Index Scans exploits multiple indexes to qualify the rows and deploys bit-vector technology to combine the results of single index scans to greatly reduce the number of records fetched from the table. This reduces the number of indexes needed to efficiently evaluate queries with multiple predicates on same table. The Star-Join plan reduces the number of rows qualified from the Fact table and the size of intermediate result set for Star and Snowflake schema queries. It takes advantage of multi-index scans in combination with an enhanced Hash-join algorithm. Users can expect significant performance improvement in commonly used warehousing queries.

Besides query performance, data warehouse environments require efficient time-cyclic data management. Many enterprises collect warehouse data by defined time intervals such as month or week. For example, Sales data is kept on a rotating 24-month cycle. When the 25th month of data arrives, the oldest partition is dropped to maintain the fixed number of partitions. Fragmentation by Interval was implemented for the purpose of defining such time intervals and the ONLINE Alter Fragment feature for Attach, Detach and Modify provides for automated or scripted management of these fragments. Fragment Level Statistics efficiently re-calculates table level statistics when fragments are attached or detached.

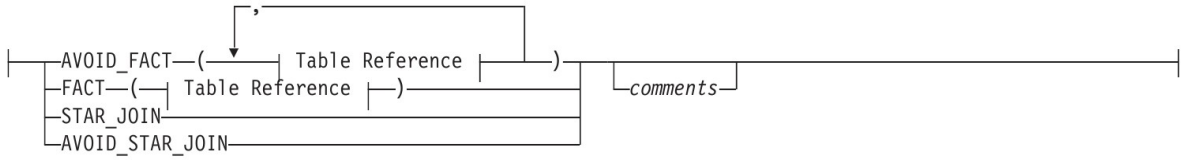
Finally, Informix 11.70 supports Light Scans for tables with VARCHAR columns**.** This means that for sequential scans, the buffer pool will be bypassed for significant performance improvement. This also facilitates a mixed-workload environment where the buffer pool servicing OLTP queries usually with random data will not be invalidated by the large sequential scans commonly used with data warehousing queries.

Star Joins

This release provides enhanced query optimizer support for data warehousing operations on tables for which star-schema dependencies exist between a fact table and a set of dimension tables. A primary key column in each dimension table corresponds to a foreign key in the fact table. A number of components are brought together for this feature:

Optimizer Directives

You can use the new star-join optimizer directives to enhance query performance in warehousing applications.



The star-join directives require that the parallel database query feature (PDQ) be enabled. Star join query optimization is disabled when PDQ is off and requires that all tables in the query have at least low level statistics. If table statistics are not available for any table in the query, star-join

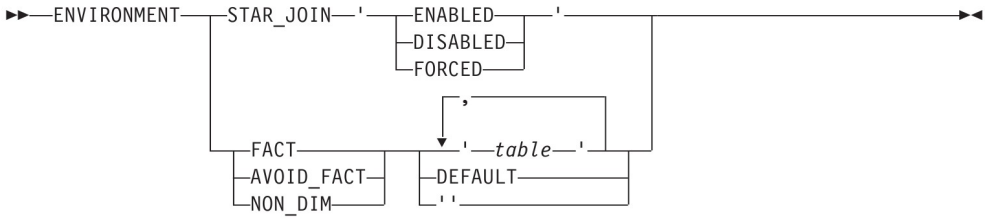
query optimization is disabled. Additionally only one fact table is allowed and the transaction isolation level cannot be COMMITTED READ LAST COMMITTED or CURSOR STABILITY, all other transaction isolation levels are supported.

In High Availability cluster environments, the star-join optimizer directives are valid on these types of secondary servers:

- Shared disk secondary servers (SDS)
- Remote standalone secondary servers (RSS)
- High-availability data replication secondary servers (HDR)

SET OPTIMIZATION ENVIRONMENT

In addition, the SET OPTIMIZATION statement supports new syntax to define a general optimization environment for all SQL statements in the session.



The options remain in effect for the session or until another SET OPTIMIZATION ENVIRONMENT statement is issued. To set these options for clients that are not aware of the new feature you can use the *sysdbopen()* function.

Note the placement of quotes around the second keyword or table identifier; also when speciying a list of multiple tables, only commas and no spaces should appear.

The effect of the ENVIRONMENT options is summarised in the following table.

Keyword	Effect	optimizer Action
STAR_JOIN	The 'ENABLED' setting turns on (and 'DISABLED' turns off) star-join support for the current session. The 'FORCED' setting favors a star-join execution path, when possible, for all queries	For 'ENABLED', the optimizer considers the possibility of a star-join execution plan. For 'FORCED', a star join plan will be chosen, if available. For 'DISABLED', star-join is not considered.
FACT	Identifies tables that correspond to fact tables in a star schema. If an AVOID_FACT table is also listed as FACT, then FACT takes precedence. DEFAULT (or an empty string) turns off this environment setting for the session.	Only tables in the FACT list are considered as fact tables in star-join optimization. Multiple tables can be listed as FACT.
AVOID_FACT	Do not use the table (or any table in the list of tables) as a fact table in star-join optimization. DEFAULT (or an empty string) turns off this environment setting for the session.	Tables in the AVOID_FACT list are not considered as fact tables in star-join optimization. Multiple tables can be listed as AVOID_FACT.
NON_DIM	Identifies tables that do not correspond to dimension tables in a star schema. DEFAULT (or an empty string) turns off this environment setting for the session.	Tables in the NON_DIM list are not considered as dimension tables in star-join optimization. Multiple tables can be listed as NON_DIM.

Star-Join Query In Action

Here is an example of a star join type of query. The *item_sales* table has many, many rows and is joined to two much smaller, dimension tables, one describing the store that the sale took place and another the stock item. We need to join to these two tables to find the total cost of on sale items from stores located in postal code KT14.

The key parts of the Star-Join, highlighted in red, is the ability to use a push down hash join on the dimension tables and the stream retrieval of the dimension data in the filter on the fact table.

```
SELECT {+ FACT ( item_sales ), STAR_JOIN }
sum (cost*volume)
FROM item_sales, stock, stores
WHERE item_sales.store_id = stores.store_id
AND item_sales.stock_id = stock.stock_id
AND stock.stock_status = 'ON SALE' AND stores.store_pcode = 'KT14'

DIRECTIVES FOLLOWED:
FACT ( item_sales )
STAR_JOIN
DIRECTIVES NOT FOLLOWED:

Estimated Cost: 37867
```

```
Estimated # of Rows Returned: 1
Maximum Threads: 7

  1) gcostanza.item_sales: SEQUENTIAL SCAN

     Bit Vector Filter: gcostanza.item_sales.store_id = stream from
gcostanza.stores.store_id

  2) gcostanza.stock: INDEX PATH

     (1) Index Name: gcostanza.stock_type_idx
         Index Keys: stock_type   (Parallel, fragments: ALL)
         Lower Index Filter: gcostanza.stock.stock_type = 'SALE'


DYNAMIC HASH JOIN
     Dynamic Hash Filters: gcostanza.item_sales.stock_id =
gcostanza.stock.stock_id

  3) gcostanza.stores: INDEX PATH

     (1) Index Name: gcostanza.store_pcode_idx
         Index Keys: store_pcode   (Parallel, fragments: ALL)
         Lower Index Filter: gcostanza.stores.store_pcode = 'KT14'


DYNAMIC HASH JOIN (Bit Vector Push Down Key: gcostanza.stores.store_id to
gcostanza.item_sales)
     Dynamic Hash Filters: gcostanza.item_sales.store_id =
gcostanza.stores.store_id
```

The *item_sales* table in this example was created using the new INDEX DISABLED feature for foreign key constraints. This means that instead of using an Index Push Down Key in the join and passing key values to the fact filter, the optimizer uses a Bit Vector instead. When index keys are used then the scan of the fact table can take advanatage of the Multi-Index Scan feature described on Page 45.

The basics of the star-join as shown here can be extended to cover more complex, multi-way join types such as starflakes.

ALTER FRAGMENT ONLINE

The ALTER FRAGMENT SQL statement now supports the ONLINE keyword. This allows processing of the statement without holding exclusive locks on the tables. An intent exclusive lock is taken instead while the database server restructures the storage design and updates the system catalog in the background. This will minimize the performance impact of ALTER FRAGMENT operations on queries and on other data-manipulation activities by other applications in concurrent sessions that attempt to access the same tables. This is especially useful for the administration of tables where historical data fragments are detached and new, future fragments are attached.

The following restrictions apply to the ALTER FRAGMENT ONLINE statement:

- Only the ATTACH, DETACH, and MODIFY options are valid.
- The table specified must be fragmented by a range interval scheme.
- The table that is being altered cannot be locked explicitly by the LOCK TABLE statement.
- The ALTER FRAGMENT ONLINE operation must be the first statement in the transaction that modifies any database object or table.
- No data movement is required between fragments.
- No other operation that modifies an object in the database can follow the ALTER FRAGMENT ONLINE statement in the same transaction.

The database server issues an error if the specified ALTER FRAGMENT ONLINE operation cannot be performed.

Compression Functionality Automatically Enabled

Previously it was necessary to run an administration API command to enable compression on a table before compression could take place. This is no longer required and all tables are available as compression candidates.

Additionally a new task template named auto_crsd() in the sysadmin database has been introduced. This task identifies all fragments meeting the requirements for automatic compression, repack or shrink or de-fragmentation. The requirements can be separately defined for each operation and need to be specified in the ph_threshold table in the sysadmin database. Each operation can be enabled or disabled separately. The required operation is automatically applied to the identified fragment and an alert based on the return code of the executed operation is inserted into the ph_alert table.

Easier Event Alarm Handling

Event alarms now have a unique identification number for each specific message. You can write scripts to handle event alarms based on the unique identification number that corresponds to each specific message in an alarm class. Previously, event alarm handling scripts had to combine the class ID and the specific message.

Space Management

This version of Informix extends the syntax for specifying fragmentation using two new schemes LIST and INTERVAL. There are also new features for the easy administration of partition extents and the ability to automatically add new data storage to the instance before data space is exhausted.

Fragmentation By LIST

Fragmentation by list is a convenient way of specifying a fragmentation scheme based on a column that has a relatively small set of discrete values. Instead of having to generate a complex set of expressions, a much simpler syntax can be used.

```
CREATE TABLE cust_calls (
   customer_num integer,
   call_dtime datetime year to minute,
   user_id char(32) default user,
   call_code char(1),
   call_descr char(240),
   res_dtime datetime year to minute,
   res_descr char(240),
```

```
   primary key (customer_num,call_dtime)
)
FRAGMENT BY LIST (call_code)
PARTITION P_B VALUES ('B') IN datadbs1,
PARTITION P_D VALUES ('D') IN datadbs2,
PARTITION P_I VALUES ('I') IN datadbs3,
PARTITION P_L VALUES ('L') IN datadbs4,
PARTITION P_O_X VALUES ('O', 'X') IN datadbs5,
PARTITION P_NULL VALUES (NULL) IN datadbs0,
PARTITION P_REM REMAINDER IN datadbs0;
```

Note that as well as the usual remainder fragment for values that aren't directly specified there is an additional partition specified for call_code values that are NULL.

Fragment By INTERVAL

For fragmentation schemes using a numeric or date column the new FRAGMENT BY INTERVAL syntax allows for automatic extension of the fragmentation definition as the table grows. This is very useful for data that is typically always increasing in value, such as a time stamp, and the DBA does not want to pre-allocate fragments for data that has not yet been inserted.

To make use of this feature, as well as the starting list of value expressions and dbspaces, you need to specify the interval size to determine how many of the values are stored in each fragment and one or more dbspaces to store new fragments in when the values go outside of the currently defined range. There is no need to define a remainder fragment as new values outside of the defined expression will always create a new fragment. New fragments are created in dbspaces chosen in a round robin fashion from the list of dbspaces specified by the STORE IN keyword.

```
> CREATE TABLE customer (
   customer_num serial not null ,
   fname char(15),
   lname char(15),
   company char(30),
   address1 char(20),
   address2 char(20),
   city char(15),
   state char(2),
   zipcode char(5),
   phone char(18),
   primary key (customer_num)
) FRAGMENT BY RANGE (customer_num) INTERVAL (10000)
STORE IN (datadbs2, datadbs3, datadbs4, datadbs5)
PARTITION p_null VALUES IS NULL IN datadbs0,
PARTITION p_10000 VALUES < 10000 IN datadbs1;

Table created.

> INSERT INTO customer (customer_num, fname, lname, company)
> VALUES (30400, 'Art', 'Vandelay', 'Vandelay Industries');

1 row(s) inserted.
```

```
> SELECT partition, exprtext FROM sysfragments f, systables t
> WHERE f.tabid = t.tabid AND t.tabname = 'customer'
> AND f.fragtype = 'T' AND f.evalpos >= 0;

partition  p_null
exprtext
VALUES IS NULL

partition  p_10000
exprtext
VALUES < 10000

partition  sys_p4
exprtext
VALUES >= 30000. AND VALUES < 40000.

3 row(s) retrieved.
```

In the above example you can see the customer table is built with a fragmentation scheme that is assumed to grow over time as more customers are added. Inserting a new customer with a customer_num outside of the current fragmentation scheme automatically causes a new fragment, with the appropriate values range to be created. The new partition will have a system generated name. The STORE IN clause is optional, without it new fragments will be created in the same dbspaces that were specified for the initial fragments.

Interval fragmentation works in conjunction with the new ONLINE keyword for ALTER FRAGMENT - allowing the modification of interval fragmentation specifications without interrupting other table users.

New Extent Sizing Features

If a table grows at a rate greater than originally envisioned then it is likely that the extent sizing will be too small. Without intervention, many small extents will be created and over time this extent fragmentation will degrade the performance of queries accessing the table. Similarly, creating a lot of empty tables is wasteful as the initial extent allocation, even when empty, requires space.

Partition Table Overflows

If you have a table that needs more extents and the database server runs out of space on the partition header page, instead of returning an error the database server now automatically allocates extended secondary partition header pages to accommodate new extent entries. The database server can now allocate an unlimited number of extents for any partition, unless the size of a table dictates a limit to the number of extents.

Defragmentation

Although it is now possible to have very many extents for a table it is undesirable for performance reasons. It takes longer to process the long list of extents and sequential data scans must skip over non-contiguous areas of disk. The new defragmentation server feature is able to rewrite the data to defragment it. This brings the rows physically closer together and avoids partition header page overflow problems.

The function can be initiated directly using the SQL administration API *task()* or *admin()* function with the *defragment* argument, specifying the table name or partition number that you want to defragment. The function is also available in the latest OAT release as part of the Admin → Storage functionality. In addition the feature can be fully automated by setting the optimization policies for the table and enabling the scheduler task *auto_crsd*.

To monitor the process a new *onstat -g defragment* option has been introduced. It should be noted that the physical re-writing of the pages will be resource intensive and consume logical log space. This feature also works in a high-availability cluster environment.

Deferred Extent Sizing

In previous versions, when a table is created the first extent for the data partition is initialized. This is not always desirable when many empty tables are created but only a small subset of those tables will ever actually contain data. This behavior can easily be seen using *oncheck -pt.*

```
$ echo 'create table empty1 (id integer, data char (400))' | \
        dbaccess stores_demo -

Database selected.

Table created.

$ oncheck -pt stores_demo:empty1

TBLspace Report for stores_demo:gcostanza.empty1

   Physical Address            4:541
   Creation date               09/27/2010 15:51:30
   TBLspace Flags              801      Page Locking
                               TBLspace use 4 bit bit-maps
   Maximum row size            404
   Number of special columns   0
   Number of keys              0
   Number of extents           0
   Current serial value        1
   Current SERIAL8 value       1
   Current BIGSERIAL value     1
   Current REFID value         1
   Pagesize (k)                2
   First extent size           8
   Next extent size            8
   Number of pages allocated   0
   Number of pages used        0
   Number of data pages        0
   Number of rows              0
   Partition partnum           4194386
   Partition lockid            4194386

   Extents
```

```
      Logical Page     Physical Page       Size Physical Pages
```

You can see that there are no physical data pages allocated to the table. But, once the first data row is inserted then the space is allocated.

```
$ echo "insert into empty1 values (1, 'First data row')" | \
        dbaccess stores_demo -

Database selected.

1 row(s) inserted.

$ oncheck -pt stores_demo:empty1
TBLspace Report for stores_demo:gcostanza.empty1

   Physical Address             4:541
   Creation date                09/27/2010 15:51:30
   TBLspace Flags               801      Page Locking
                                TBLspace use 4 bit bit-maps
   Maximum row size             404
   Number of special columns    0
   Number of keys               0
   Number of extents            1
   Current serial value         1
   Current SERIAL8 value        1
   Current BIGSERIAL value      1
   Current REFID value          1
   Pagesize (k)                 2
   First extent size            8
   Next extent size             8
   Number of pages allocated    8
   Number of pages used         2
   Number of data pages         1
   Number of rows               1
   Partition partnum            4194386
   Partition lockid             4194386

   Extents
      Logical Page     Physical Page       Size Physical Pages
                0          4:1415          8          8
```

Note that this feature is only applicable to tables that are created without specifying an initial extent size. If the CREATE TABLE uses an EXTENT SIZE n clause then it is assumed that the table will be populated and the first extent is allocated as normal. Otherwise it is possible to modify the first extent size after the creating the table, using ALTER TABLE, and still not allocate any space. When the first data row is inserted it will allocate an initial extent of the new size.

The same storage allocation deferral applies to tables defined by the CREATE TEMP TABLE statement that do not include any of the storage specifications listed above.

Specifying The Extent Size On User-defined Indexes

In earlier releases, when a user define an index using the CREATE INDEX statement, the database server calculates the extent sizes based on the data extents allocated.

In this release, the CREATE INDEX statement supports new syntax to manually specify the first extent size and the next extent size when the index is defined. The existing CREATE INDEX statement has been extended to support a new EXTENT SIZE clause, similar to the SQL syntax for defining table extent sizes in the CREATE TABLE and ALTER TABLE statements.

Automatic Storage Provisioning

To minimize the possibility of the database instance running out of space, Automatic Storage Provisioning allows the DBA to configure the system so that it can automatically add new space. This removes the requirement to continually monitor space usage and manually add data space. Adding space could be expanding an existing chunk or creating a new chunk.

You can configure Informix to automatically expand an existing storage space if the space is full. You can also configure Informix to expand the space before it is full, when its free pages fall below a specified threshold. Even if you prefer to add space manually, automatic storage provisioning simplifies the process of adding space, because once configured you do not need to determine where to get the space.

The first thing to do is configure a Storage Pool that defines a set of raw devices, files or directories that can be used to allocate space. A new table, storagepool, is created in the sysadmin database to maintain the pool details which is usable by the whole instance. The administration of the pool can be done by OAT or the Admin API using the following comands:

storagepool add
storagepool modify
storagepool delete
storagepool purge
modify space sp_sizes
modify space expand
modify chunk extendable [off]
modify chunk extend

To add an entry to the storage pool use:

```
EXECUTE FUNCTION task("storagepool add",
      "<path>",
      "<offset>",
      "<total_size>",
      "<chunk_size>",
      "<priority>");
```

The offset and sizes are in kilobytes by default, though other units can be specified. The priority is 1 (High), 2 (Medium), or 3 (Low). When adding a directory to the storage pool, the offset and total size must be specified as 0. Internally, the pool management includes checks such as chunk overlap and will mark a pool entry as unavailable if a problem is found when accessing it.

To expand a dbspace manually use the following API command.

```
EXECUTE FUNCTION task("modify space expand",
      "<space_name>",
      "<min_size>");
```

The minimum size may be rounded up, depending on the space's page size and any chosen pool entry's minimum chunk size. The space will be expanded by either extending a chunk or adding a chunk using the storage pool. In order to extend an existing chunk it needs to be first marked as extendable.

```
EXECUTE FUNCTION task("modify chunk extendable",
      "<chunk_num>");
```

If you want to manually extend a chunk, as distinct from expanding a space, you would use:

```
EXECUTE FUNCTION task("modify chunk extend",
      "<chunk_num>",
      "<min_size>");
```

As well as expanding/extending existing space it is also possible to use the storage pool to create new spaces and chunks, freeing the DBA from having to continually think about the specific OS disk layout.

New configuration parameters have been introduced to control Storage Provisioning.

SP_AUTOEXPAND This parameter controls (0 = off, 1 = on) the automatic expansion of spaces - both automatic chunk creations and automatic chunk extensions. It does not affect manual space creations or expansions

SP_THRESHOLD The lowest amount of free space, expressed as a percentage or an absolute number of kilobytes, tolerated in a dbspace before the server will attempt to expand the space when the low-space-monitoring task runs. This setting applies to all spaces in the instance, you cannot configure this for each dbspace.

SP_WAITTIME The number of seconds that a thread will wait for space expansion before returning an out-of-space error.

It is possible to set some minimums on the extension or creation size for a space.

```
EXECUTE FUNCTION task("modify space sp_sizes",
      "<space name>", "<create_size>", "<extend_size>");
```

Where the create size is the minimum size of a chunk automatically created in this space and the extend size is the minimum amount by which a chunk in this space can be automatically extended. Neither setting affects manual operations. The values can be express as a percentage or a number of kilobytes with the defaults being10% and 10000k, respectively.

For example:

```
> execute function task ('storagepool add',
>  '/spare/chunks/11.70', 0, 0, '50M', 1);

(expression)  Succeeded: Space added to storage pool

> execute function task ('modify space expand', 'datadbs', '25M');

(expression)  Succeeded: Space 'datadbs' has been expanded.
```

There are new flags printed by *onstat -d* to show the status of dbspaces and chunks subject to storage provisioning (flag position 5, value 'A' indicates the dbspace is auto-expandable). You can see in red the convention for naming of a new chunk when using space expansion.

```
Dbspaces
address          number   flags     fchunk   nchunks  pgsize   flags    owner
name
4fedc028         1        0x70001   1        1        2048     N  BA    informix rootdbs
5003b860         2        0x70001   2        1        2048     N  BA    informix physdbs
5003ba08         3        0x60001   3        1        2048     N  BA    informix logdbs
5003bbb0         4        0x60001   4        2        2048     N  BA    informix datadbs
5003bd58         5        0x68001   5        1        2048     N SBA    informix sbspace
50021c50         6        0x42001   6        1        2048     N TBA    informix
tempdbs
 6 active, 2047 maximum

Chunks
address          chunk/dbs     offset     size       free       bpages     flags pathname
4fedc1d0         1     1       0          109568     97275                 PO-B-D
/products/11.70/ol_informix1170/dbspaces/rootdbs
50021df8         2     2       0          34816      21963                 PO-B-D
/products/11.70/ol_informix1170/dbspaces/physdbs
4fedc438         3     3       0          13312      971                   PO-B-D
/products/11.70/ol_informix1170/dbspaces/logdbs
4fedc638         4     4       0          25600      25547                 PO-B-D
/products/11.70/ol_informix1170/dbspaces/datadbs
4fedc838         5     5       0          16384      15204      15205      POSB-D
/products/11.70/ol_informix1170/dbspaces/sbspace
                       Metadata 1126       837        1126     4fedca38         6     6
0          8192       8139                 PO-B--
/products/11.70/ol_informix1170/dbspaces/tempdbs
525f9dd8         7     4       0          25600      25597                 PO-BED
/spare/chunks/11.70/ol_informix1170_datadbs_p_1
 7 active, 32766 maximum
```

Automatic storage provisioning is supported in a high-availability cluster. In this environment, any storage pool entry (directory, cooked file, or raw device) on the primary server must also be available through the same path on all secondary servers. Storage provisioning does not support dbspaces that have mirror paths defined.

Fragment-level Statistics

In previous releases data distributions were calculated at the table level to optimize query plans. This release supports a finer granularity of statistics for fragmented tables. The statistics are calculated and stored at the individual fragment level. Two new table properties for fragmented tables have been introduced

1. STATCHANGE specifies a percentage threshold on the number of rows changed in each table or fragment since the data distributions were last calculated. When UPDATE STATISTICS is run it will selectively update only the data distributions for those tables or fragments that have exceeded this threshold.

2. STATLEVEL, specifies whether TABLE or FRAGMENT is the granularity for data distributions. There is also an AUTO setting to allow the database server to automatically choose the granularity of the distribution statistics for each fragmented table.

Two new configuration parameters AUTO_STAT_MODE and STATCHANGE have been introduced to control the default values on a system and session basis. Set AUTO_STAT_MODE as an onconfig value, or as a per-session environment parameter, to enable or disable the ability of the database server to selectively update only stale or missing data distributions when performing an UPDATE STATISTICS operation.

Note that fragment distribution statistics for a table or index will be dropped if an ALTER FRAGMENT statement redistributes data rows. Any of the ATTACH, DETACH, DROP, and INIT fragment operations may cause this to happen. The next UPDATE STATISTICS operation rebuilds all fragment level distributions, so that the query optimizer will not choose a query execution plan that is based on statistics that the ALTER FRAGMENT operation has made stale.

For each table or index fragment, the *encdist* column of the SYSFRAGDIST system catalog table stores these distribution statistics as an encrypted BLOB object. The *sbnum* column of SYSFRAGDIST now identifies the sbspace in which that smart large object is stored. By default, this is the sbspace specified in the setting of the SBSPACENAME configuration parameter.

Improving Performance By Reducing Buffer Reads

A new onconfig parameter BATCHEDREAD_INDEX allows a query to read a group of keys from a page as a single operation. This reduces the number of times that a buffer has to be accessed. There is also an equivalent BATCHEDREAD_KEYONLY for when Key-only reads have been chosen by the optimizer.

Reduced Overhead For Foreign Key Constraints

Foreign key constraints are associated with an index on the child table that the constraint references. For child tables with a very large number of rows, but only a few distinct foreign key values, DML operations using the index can impose substantial overhead on the server, compared to sequentially scanning the child table.

For these cases, the ALTER TABLE ADD CONSTRAINT FOREIGN KEY statement of SQL can now include the optional INDEX DISABLED keywords. These disable the index when the foreign key constraint is created, and can improve the efficiency of insert, delete, and update operations on very large child tables. (In CREATE TABLE statements that define foreign key constraints, the legacy syntax is unchanged.)

The new features supporting Data Warehouse type queries are documented in the *IBM Informix Performance Guide* and the *IBM Informix Guide to SQL: Syntax.* The new Informix 11.70 general administration features are documented in the *IBM Informix Administrator's Guide*, the *IBM Informix Administrator's Reference*, the *IBM Informix Guide to SQL: Syntax* and the *IBM Informix Migration Guide*.

EMPOWER APPLICATION DEVELOPMENT

Informix 11.70 continues to add functionality to help improve application developer's productivity and to help minimize the work to keep an Informix database application in sync with a version of the same application that supports another DBMS. Many of our ISVs have asked for SQL enhancements so that the SQL generated in the application can be used with Informix without modification. In addition, options for debugging Stored Procedure Language (SPL) functions and procedures have been limited. Informix 11.70 enhancements in both of these areas will make things easier for application developers.

Changes to the some of the SQL arguments to recognize options that are common in other DBMS greatly reduces the work required to keep an application in sync with Informix and some other DBMS. This is a key consideration for ISVs when deciding to support multiple DBMS with their applications. Informix 11.70 enhancements include recognizing various SQL expressions as part of the COUNT construct, more flexibility when defining default and NULL behavior in CREATE TABLE and ALTER TABLE commands, and changes to all of the CREATE <database object> and DROP <database object> to reduce logic required to determine if the object already exists prior to issuing the CREATE or DROP statement.

Informix 11.70 addresses another area of support where application developers have frequently asked for support – Open Source applications. While strictly not a Informix server feature, Informix 11.70 is now certified with Hibernate, Drupal, and other open source applications. Informix DataBlades have long been a powerful mechanism to extend the database capabilities for application specific data types and access methods. Working with the DataBlade modules did require a few configuration tasks before usage however. Informix 11.70 eliminates most of this configuration, making this support available right “out of the box”. Another Informix 11.70 enhancement to make things easier is in the area of stored procedure maintenance. SPL has long been the standard Informix language for defining portable stored procedures, functions and triggers. Unfortunately the options for debugging these constructs in-place has been limited. Informix 11.70 now includes support to work with IBM Optim™ Development Studio and the free IBM Data Studio products to provide this support. The Optim family of graphical tools is also making continuous enhancements to support Informix within more of its product offerings. Continue to look for enhancements in these tools to correspond with enhancements within the Informix database engine. Lets take a more detailed look at some of the changes in Informix 11.70 in the application development area.

Automatic Registration Of Database Extensions

Built-in database extensions (formerly known as built-in DataBlade® modules) can now be used without performing some of the previously required prerequisite tasks, such as registering the extensions or creating specialized Virtual Processors. The following built-in database extensions are automatically registered when they are first used: basic text search (BTS), node data type, binary data types, large object locator, WebSphere® MQ messaging, Informix web feature service (WFS), Spatial and TimeSeries.

As an example, in order to use the BTS functionality, create the BTS index (after creating the database and table containing LVARCHAR data) using the bts access method. Note that this example also demonstrates the new feature of BTS which allows for multi-column indexes to be created. This speeds up processing of queries that need to search more than one column.

```
> create database mydb with log;

Database created.

> create table tab1 (defect bigserial, abstract varchar(100), desc lvarchar);
```

```
Table created.

> create index tab1_desc_bts on tab1 (abstract bts_varchar_ops,
   desc bts_lvarchar_ops) using bts;

Index created.
```

The BTS virtual processor class is dynamically created, if it does not exist. A smart blob space (sbspace) is also created automatically for the BTS searches if a default sbspace (onconfig SBSPACE or SYSSBSPACE) does not exist. If required, the sbspace creation will be subject to Storage Provisioning. If this is not configured then it is created in the same directory as rootdbs; unless it is a device when a new space is created in $INFORMIXDIR/tmp. You can see the auto registration information recorded in the online.log.

```
INFO (autoregexe 1) (EXECUTE FUNCTION sysbldprepare ('bts.*', 'create');)
Loading Module <$INFORMIXDIR/extend/bts.2.00/bts.bld>
The C Language Module </products/11.70/extend/bts.2.00/bts.bld> loaded
INFO (autoregvp 2) (Execution of [ autoregvp ] dymamically creating VPCLASS bts)
```

In another example, a table is created containing a column of the Node data type:

```
> create table tab2 (n_id node, abstract varchar(100), desc lvarchar);

Table created.
```

Using the Node data type in the CREATE TABLE statement or by using any Node data type function triggers the registration of the database extension. With the WFS and MQ database extensions, the virtual processors are created automatically in a similar manner. Additionally, if not already created, running an XML function will automatically add a *idsxmlvp* VP.

Database extensions are discussed in the *IBM Informix Database Extensions User's Guide*.

Debugging Informix SPL Routines

Informix Stored Procedure Language (SPL) routines can now be debugged using IBM Data Studio, IBM Optim Development Studio (ODS, version 2.2.1.0 or later), or with Microsoft Visual Studio by installing the IBM Database Add-Ins for Visual Studio (IDAIVS) debugger for Informix SPL Procedures. The SPL Routine Debugger connects to the database server via a DRDA protocol connection, and so for use with Data Studio or Optim Development Studio, the IBM® Data Server Driver for JDBC and SQLJ should be used. If using IDAIVS, then the IBM Informix .Net provider should be used for database connectivity.

The session manager component is a daemon process that manages the debugging context between the database and the client debugger. It can be hosted on the database machine, the machine on which the development environment is running, or any other machine. The session manager can be configured to start with the client debugger, or can be started manually. If manually started, the IP address and port number of the session manager should be provided to the debugger client during configuration.

There are some caveats for the current implementation:

1. Not all of the Informix 11.70 data types can be fully manipulated from within the debugger. Refer to the data type support document for information on which Informix data types are read-only and which are updateable.

2. Databases must have LOGGING enabled (created using the WITH LOG keywords) to support SPL debugging.

3. Secondary servers within a cluster environment do not support 'Step into' trigger procedure operation for Insert, Delete, or Update triggers. This is supported with Primary servers.

4. Behavior of delimited identifiers and the DELIMIDENT environment variable may impact how the Informix database server interprets quoted strings. Please read the documentation carefully and follow the recommended steps.

5. When debugging using Data Studio or ODS, which use Java based JDBC connectivity, it is important to note that by default, the IBM Data Server Driver for JDBC and SQLJ sets 'AUTOCOMMIT' transaction mode to 'TRUE'. This is enabled in the Data Studio or ODS debugging session for all logging databases, including databases that were created WITH LOG MODE ANSI, and also databases that are not ANSI-compliant. Currently, the Informix server performs an implicit commit operation after every SQL statement completes in an Optim Development Studio debugging session. If there is an explicit transaction started inside the Informix SPL procedure, the database server ignores SQL ERROR -535 and continues the debugging session.

6. Currently, SPL *function* debugging is not supported by IBM Database Add-Ins for Visual Studio. You can only use the IDAIVS debugging environment with Informix SPL *procedures* that do not return any values to the calling context.

SQL Compatibility Enhancements

SQL Expressions As Arguments To The COUNT Function

The **COUNT** function has been enhanced to accept as its argument all of the same expressions that are allowed in the argument list of other built-in aggregate functions, as well as the asterisk (*) notation that only **COUNT** supports. The following examples (syntactically correct, but perhaps not semantically useful) depict several categories of built-in expressions are now supported in version 11.70 as the argument to **COUNT**:

Arithmetic Expressions

```
> select count(quantity + 1) from items;

      (count)

          67

1 row(s) retrieved.

> select count(times(quantity,2)) from items;

      (count)

          67

1 row(s) retrieved.
```

Bitwise Logical Functions

```
> select count(BITAND(quantity,1)) from items;

      (count)
          67

1 row(s) retrieved.
```

Cast Expressions

```
> select count(NULL::int) from items;

      (count)

           0

1 row(s) retrieved.
```

Conditional Expressions

```
> select count (case when stock.description = "baseball gloves" then 1 else
NULL end) from stock;

      (count)

           3

1 row(s) retrieved.
```

Constant Expressions

```
> select count(TODAY) from items;

      (count)

          67

1 row(s) retrieved.
```

Function Expressions

```
> select count (length('abc') + length('def')) from items;

      (count)

          67

1 row(s) retrieved.
```

Column Expressions

```
> select count(customer.customer_num) from customer;

      (count)

          28

1 row(s) retrieved.
```

Simplified SQL Syntax For Defining Database Tables

Syntax for the CREATE TABLE and ALTER TABLE statements has been enhanced to allow greater flexibility in the order of keyword specification in three areas:

- Default keyword placement has been relaxed so that it can precede or follow any of the constraint definitions for the column. The NOT NULL constraint is no longer required to be listed first if additional constraints are defined. The constraints (on a single column or on a set of multiple columns) can be defined in any order within the constraint specifications, and that list of constraint definitions can be followed (or preceded) by the default value, if a default is defined on the column.
- The list of constraints can now include the NULL keyword to indicate that the column can accept NULL values. The NULL constraint cannot also be specified with NOT NULL or PRIMARY KEY in the constraint list.
- Placement of the ON DELETE CASCADE keywords has been relaxed so that it can appear before or after the CONSTRAINT name.

A CREATE TABLE example showing the three SQL enhancements mentioned above:

```
> create table foo (
> col1 int NOT NULL default 0,
> col2 int NULL,
> col3 int REFERENCES customer (customer_num) CONSTRAINT foo_c1
> ON DELETE CASCADE);

Table created.
```

A simple ALTER TABLE example showing adding a new column with the NULL keyword:

```
> ALTER TABLE foo ADD (col4 int NULL);

Table altered.
```

These table definition and modification enhancements in the Informix SQL parser enhance SQL language compatibility and allow SQL constructs written for other DBMS to be used directly with Informix.

New IF [NOT] EXISTS SQL Statement

The IF NOT EXISTS keyword has been added to all of the CREATE <database object> (and the CREATE database) SQL statements. This functionality, often found in other DBMS, enhances the flexibility for Informix users wanting to migrate SQL developed for these other DBMS to Informix. This clause causes the existence of a database object of the same name to be checked during the execution of the CREATE <database object> statement. Previously, if a SQL request is made to create a database object with the same name as an existing database object,

an error would be generated. With this new functionality, if the object already exists, then the CREATE <database object> SQL statement is ignored, and no error condition is raised.

Similarly, the IF EXISTS keyword has been added to all of the DROP <database object> statements. While processing this statement, the existence of the database object is determined. If the object exists, then it is dropped. If the object does not exist, then the DROP <database object> SQL statement is ignored, and no error condition is raised.

In the following example, the CREATE TABLE IF NOT EXISTS statement determined that the table does exist, and so no operation is performed. A check on the columns confirms that it is the table created and then altered above. The subsequent operation DROP TABLE IF EXISTS determines that the table does exist and successfully drops it. The same CREATE TABLE IF NOT EXISTS statement is then re-executed, and since the table does not exist, a new one is created. Finally the same statement is again executed, and since the table does exist, the statement returns without error.

```
> create table IF NOT EXISTS foo (col1 int);
> info columns for foo;


Column name        Type                          Nulls

col1               integer                       no
col2               integer                       yes
col3               integer                       yes
col4               integer                       yes
>
> drop table IF EXISTS foo;

Table dropped.

> create table IF NOT EXISTS foo (col1 int);

Table created.

> create table IF NOT EXISTS foo (col1 int);
```

These new clauses in the CREATE <database object> and DROP <database object> SQL statement make programming easier because the logic for checking for the existence of the object before creating or dropping is no longer needed.

Note that when this programming construct is used in a client API, such as ESQL/C or 4GL the failure to execute the statement because of the EXISTS clause is signaled by the *sqlca.sqlwarn.sqlwarn0* being set to 'W' and the *sqlca.sqlerrm* being set to 'IGNORE'. It will also trigger a WHENEVER SQLWARNING statement.

New Informix Open Source Support

During the Informix 11.70 time frame, Informix-specific customizations and certification have been performed on several popular open source projects. Patches have been created for Hibernate and Drupal and are available via the International Informix User's Group (IIUG) on their web site (iiug.org/opensource). They are working on contributing the source code changes back to the respective open source projects. Certification has been completed for Geronimo and

Tomcat. MediaWiki and XWiki are both currently in beta through the IIUG. Please see the IIUG web site for additional information and for updates on these and other open source projects with Informix.

Controlling Character Conversion And Defining The Escape Character

The IFX_UNLOAD_EILSEQ_MODE environment variable enables DB-Access, *dbexport*, and High Performance Loader (HPL) to retrieve character data that is invalid for the locale specified in the environment.

The DEFAULTESCCHAR configuration parameter specifies the default escape character that is used in a MATCHES or LIKE clause. Acceptable values are NONE for no escape, or a single character to specify the escape character. Prior to version 11.70 the default was '\'. This setting can be overridden at the session level using SET ENVIRONMENT DEFAULTESCCHAR or at the statement level using the ESCAPE keyword.

Setting The File Seek Position For Large Files

In the development API for Database Extensions (LIBDMI) the file manipulation functions were not fully 64-bit capable. Two new functions have been added to allow access to file offsets larger than 2GB. Use the mi_file_seek8() function to set the file seek position for the next read or write operation on an open file of length greater that 2GB. You can return the current file seek position, relative to the beginning of the file, for an operating-system file of length greater than 2 GB by using the mi_file_tell8() function.

Automatically Terminate Idle Sessions

There is a new server scheduler task, *idle_user_timeout*, which when enabled will automatically terminate a user session that has been idle for more that a set number of minutes. The default setting is for an idle timeout of 60 minutes. This feature can easily be configured using the OpenAdmin Tool (OAT) where the task can be enabled and the parameter value, IDLE TIMEOUT, can be set.

When a user tries to make use of a connection after being idle for longer than the specified timeout, then they will receive the error: **25582: Network connection is broken**. A log of the termination event is recorded in the Alert list with severity GREEN, alert type INFO and alert state of ADDRESSED. Note that this feature will not attempt to disconnect users *informix* or *root*.

Session-level Memory Allocation

For data warehouse type queries it can be advantageous to allow more memory to be allocated for processing than for a regular query. Ordinarily the PDQ settings will allow for more memory to be allocated but this may be inappropriate in mixed workload environments. There are now new IMPLICIT_PDQ and BOUND_IMPL_PDQ session environment options that can automatically increase the available memory for a query without having to set an explicit PDQ level.

- When IMPLICIT_PDQ is set to ON, unless BOUND_IMPL_PDQ is also set, the database server ignores the current explicit setting of PDQPRIORITY, and automatically determines an appropriate PDQPRIORITY value for each query.
- When IMPLICIT_PDQ is set to OFF (or to zero), the server does not override the current PDQPRIORITY setting.
- When the BOUND_IMPL_PDQ session environment option is set to ON (or to one), you require the database server to use the explicit PDQPRIORITY setting as the upper bound for memory that can be allocated to a query.

You can use the SET EXPLAIN statement to see the effect of these variables on the calculated memory limit, and the IMPLICIT_PDQ value that was granted for the query. Note that these values are modified on a per-session basis using the SET ENVIRONMENT statement. Use the *sysdbopen()* function to control these settings on a more global level.

MQ Messaging Enhancements

The WebSphere® MQ Database Extension allows for the simple association of an IBM MQ messaging framework with a set of database tables. At the simplest level, inserting into a table will send an MQ message, selecting from a table will receive an MQ message. The queue managers can reside any where in the network and participate in a transaction. There is no limit to the number of queue managers that can participate in a transaction.

The MQ Database Extension enhancements include some new functions:

- *MQHasMessage()*, which tells you if there is a message in the queue
- *MQInquire()*, which queries for attributes of the queue
- *MQCreateVtiWrite()*, which creates a table and maps it to a queue managed by WebSphere MQ (WMQ)

There are also some new onconfig parameters, they are required when using MQ messaging over a network.

- MQSERVER - Specifies the location of the WebSphere MQ server and the communication method to be used
- MQCHLLIB - Specifies the path to the directory containing the client channel definition table.
- MQCHLTAB - Specifies the name of the client channel definition table

These enhancements reduce the number of needed WMQ licenses and streamline administrative tasks. A full description of the MQ functionality, along with the new configuration parameters, can be found in the *IBM Informix Database Extensions User's Guide.*

Enhancements To *Dbschema*, *Dbexport* & *Dbimport*

Both *dbschema* and *dbexport* now support the -nw option which omits the object owner name when printing out DDL statements.

The *dbschema* utility can now generate commands that can be used to rebuild the physical characteristics of the instance such as storage spaces, chunks, and logs. Either SQL Admin API commands can be generated (*dbschema -c*) or as a command line script (*dbschema -c -ns*).

It is now possible to use the *dbschema* and *dbimport* utilities on all types of secondary servers in a high-availability cluster. The *dbexport* command can be run in a cluster environment but only on the primary server, or on an RSS server where STOP_APPLY and USELASTCOMMITTED have been set.

IBM Optim Development Studio And IBM Data Server Driver Installation

Informix 11.70 (manufactured) bundles include the images to install the IBM Optim Development Studio (ODS) and the IBM Data Server Drivers. ODS is an Eclipse based development environment, used for creating database applications. Informix stored procedure language (SPL) debugging is now supported within ODS (and Data Studio), and so this installation is intended to provide a copy of the development environment to perform SPL debugging or use for other development activities. The IBM Data Server Drivers are DRDA based connectivity (Java JDBC and .NET) for Informix. Use of these drivers necessitates DRDA connectivity between the application and database.

ODS can be installed using the separate installation media. The files to install the IBM Data Server Drivers are placed into the Informix target location *IBM_Data_Server_Driver_Package/installDSDriver* after installation of the 11.70 server. On Windows operating systems, unzip the ibm_data_studio_standalone_win.zip file and run the install.exe program.

When you use the Informix installation application on Windows, you can select to install the IBM Data Server Driver Package with either Informix Client Software Development Kit or Informix Connect. You do not need to download installation media for that product from the web.

Discover the new application development features in the *IBM Informix Guide to SQL: Syntax* manual. Installation-related features are documented in the *IBM Informix Installation Guide for UNIX, Linux, and Mac OS X* and the *IBM Informix Installation Guide for Windows.*

ENHANCED SECURITY MANAGEMENT

Informix 11.70 continues to focus on new functionality to help keep your database contents secure and to help you control access to your database. It is critical to know who is connecting to the database and what information is being accessed. Informix 11.70 adds several new features to help in both of these areas.

Selective row level auditing provides a finer granularity of control over which tables generate audit information. Trusted Contexts and Trusted Connections provide a solution to the age old problem of determining the actual user that is connecting to the database via a multi-tier web application. Most web application servers connect to the database using a default userid, making it virtually impossible for accurate auditing of the modifications made to the database from the web application. Trusted Contexts and Trusted Connections allow the DBA to set up the environment where real userids, and associated privileges, can be used. Another valuable enhancement in this area is the ability to define users to the database without having to define them on the host operating system on which the database resides. Not only does this reduce the work of the DBA, but it eliminates the possible security issues of extraneous userids where local operating system level access is not needed. Let's take a deeper look at these Informix 11.70 enhancements in the area of Security.

Selective Row-level Auditing

With Informix 11.70, the database system security officer (DBSSO) can configure auditing so that row-level events are recorded for designated tables, rather than for all tables used by the database server. In previous releases, auditing was an “all or nothing” type activity. By selecting only the tables that are required to be audited on the row level, selective row level auditing (SLRA) can improve database server performance, simplify audit trail records, and mine audit data more effectively. Many tables in the database, for example reference tables, do not require row-level audit information to be created. Informix 11.70’s new support to individually specify tables for which row level auditing information will be captured will reduce the overall scope of the captured audit information, and thus reducing the disk storage required for storage and improving the performance of the audit activity which accesses this information.

Informix 11.70 adds a new table level property, “AUDIT” which can be specified with the CREATE TABLE or ALTER TABLE COMMAND. This property controls the auditing on the specific table. Examples of this command:

```
CREATE TABLE {existing syntax} | with AUDIT;

ALTER TABLE {existing syntax} | add AUDIT;

ALTER TABLE {existing syntax} | drop AUDIT;
```

In addition, a new parameter, ADTROWS was added to the *adtcfg* file to indicate whether this behavior is enabled. Settings for the ADTROWS parameter:

0 - No changes in existing row level auditing behavior (default)
1 - SRLA is enabled and only "audit" enabled tables will generate row-level audit records

Database Users Without Operating System Accounts

Informix 11.70 provides a mechanism to define users that are known to the database, but are not necessarily users on the host computer on which the database resides. In previous releases, each user who required access to the database server also needed an operating system account on the host computer. New Informix 11.70 functionality allows configuration so that users who are

authenticated by an external authentication service (such as Kerberos or Microsoft Active Directory) can connect to Informix without local host machine accounts. This feature will greatly help ease some of the administration required for environments with a large number of users.

The new USERMAPPING configuration parameter in the *onconfig* file specifies whether or not this capability is enabled, and if so which users can access the database server, and whether any of these users can have administrative privileges. When Informix is configured to allow user mapping, the DBA can still control which externally authenticated users are allowed to connect to Informix and their privileges.

The process of granting access to these users follows similar authorization granting activities, by using the GRANT statement. Use the GRANT statement with the ACCESS TO clause to map users to user properties required for access to Informix server resources. As part of this command, the properties of another known (operating system based) user can be transferred to the new user. This can be done by mentioning a known user with a desired level of authorization, or the operating system user and group ID numbers can be provided. The user mapping tables in the SYSUSER database are system catalog tables that map users to OS-level properties that enable Informix database access and control level of privileges. System tables that have been updated include *sysusermap*, *syssurrogates*, and *syssurrogategroups*.

Some examples of GRANT ACCESS and REVOKE ACCESS commands:

```
GRANT ACCESS TO ckramer PROPERTIES USER sbross;
```

This command grants access to the non-operating system user 'ckramer'. When user 'ckramer' connects to Informix, it will use the UID, GID(s) and home directory for user 'sbross', which must be a user name known to the operating system.

```
GRANT ACCESS TO ckramer PROPERTIES UID 101, GROUP 10011;
```

This command grants access to the non-operating system user 'ckramer'. When user 'ckramer' connects to Informix, it will use the anonymous UID 101 and the anonymous group 10011 when an o/s identity is required.

```
GRANT ACCESS TO PUBLIC PROPERTIES USER dbuser;
```

This command grants access to public, and so any user that can authenticate but does not have an explicit entry designating the mapped (surrogate) user will use the identity of 'dbuser'.

```
REVOKE ACCESS FROM ckramer;
```

This command revokes access from user 'ckramer', which means that 'ckramer' no longer has access to the machine via user mapping unless PUBLIC is given mapped access, in which case 'ckramer' now uses the same privileges that PUBLIC uses. Alternatively, user 'ckramer' may have been created as an operating system user, in which case those privileges take precedence over entries in *sysusermap* and *syssurrogates*.

Trusted Connections

Informix 11.70 enhancements help improve security for multiple-tier application environments. It provides support for Trusted Contexts, which can then be used to establish trusted connections between an application server and the Informix database server on a network. Trusted connections allow the DBA to set the identity for each specific user accessing a database through the middle-tier server, which facilitates discretionary access control and auditing based on user identity. Without a trusted connection in such an environment, each action on a database is performed with the single, generally anonymous, user ID of the middle-tier server, potentially lessening granular control and oversight of database security.

Trusted connections are very useful in that they allow connection reuse under a different userid with authentication to avoid the overhead of establishing a new connection. They allow connection reuse under a different userid without authentication to accommodate application servers that need to connect on behalf of an end-user but do not have access to that end-user's password to establish a new connection on their behalf. They allow users to gain additional privileges when their connection satisfies certain conditions defined at the database server. Trusted context helps remove some of the user authorization issues that are common in a 3 (or more) tier application server environment. Some of these issues are loss of individual user identity, diminished user accountability, over granting of privileges to the middle tier's userid, and weakened security.

A Trusted Context is a database object created by the database security administrator (DBSECADM) that defines a set of properties for a connection that when met; allow that connection to be a "trusted connection" with special properties. Some of the properties of a trusted connection include: the connection must be established by a specific user; the connection must come from a trusted client machine; and the port over which the connection is made must have the required encryption. If these criteria are met, the connection will allow changes in userid and privileges as defined in the trusted context.

An example of creating a trusted context:

```
CREATE TRUSTED CONTEXT ctx1
     BASED UPON CONNECTION USING SYSTEM AUTHID gcostanza
   DEFAULT ROLE MANAGER
   ENABLE
     ATTRIBUTES (ADDRESS '14.26.223.106')
     WITH USE FOR jpeterman, sbross WITHOUT AUTHENTICATION
```

This example creates the trusted context object named 'ctx1' which will allow connections from the IP address 14.26.223.106. The connection can switch to user 'jpeterman' or 'sbross' once the trusted connection has been established.

An example of creating a trusted connection from an ESQL/C program:

```
EXEC SQL CONNECT TO 'dbname@ol_informix1170' TRUSTED;
```

Client API support for TRUSTED connections is provided in Informix ESQL/C, JDBC and ODBC.

In order to switch from one user to another from within the trusted context scope, use the SET SESSION AUTHORIZATION command.

```
SET SESSION AUTHORIZATION TO 'jpeterman';
```

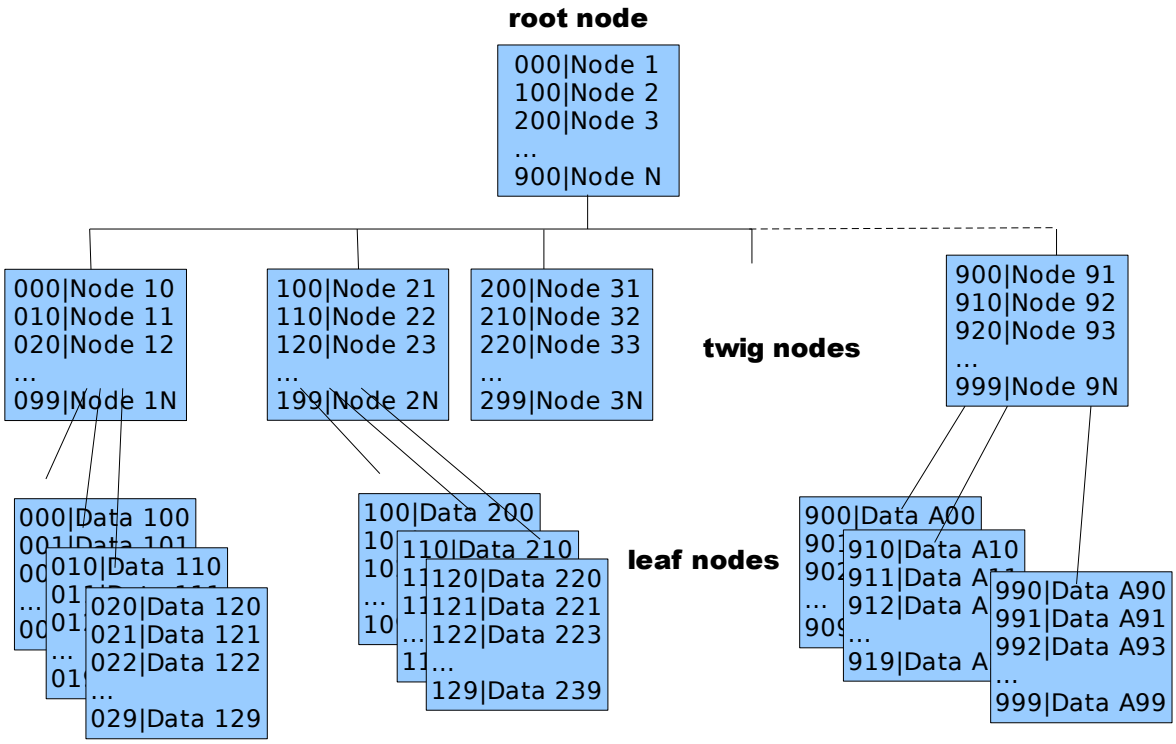
In this example, using the above trusted context example, the user is switched to 'jpeterman'. This operation can be performed to switch to any user defined in the trusted context object scope. The new user can perform all valid database operations as defined in the scope. Audit records will show the switched user as the originator of the operations. If using database transactions, a COMMIT or ROLLBACK operation should be performed before switching to a new user.

All the new Informix 11.70 security features are documented in the *IBM Informix Security Guide*.

IMPROVED PERFORMANCE

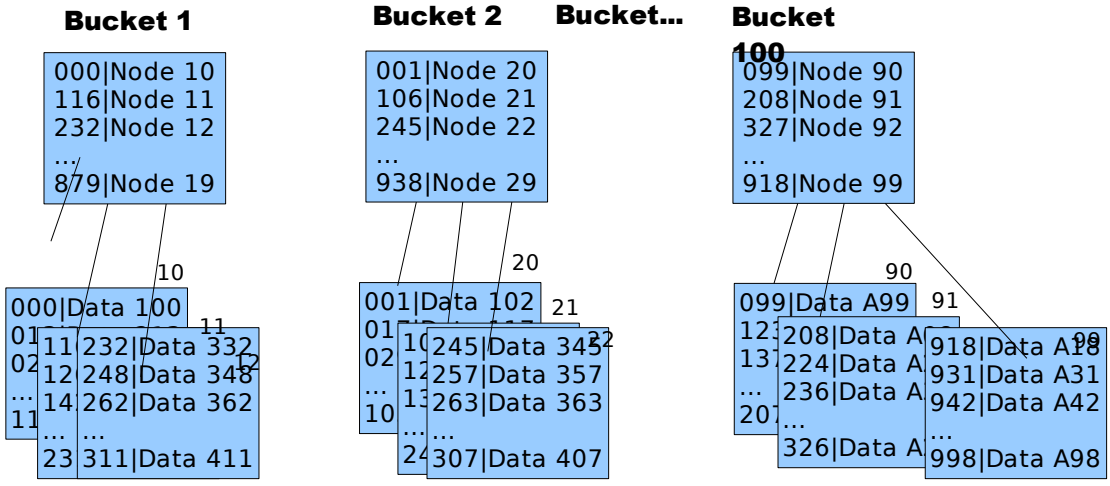
Forest Of Trees Indexes

To increase the performance of some types of query, a new way of organizing btree indexes is now available. Instead of a single btree for each table or fragment. A Forest of Trees index uses multiple small btrees, with the first level of storage being a hash lookup rather than a conventional root node. The benefits include less contention on the single btree root node and a reduction in depth compared a large, conventional btree, which minimizes the number of buffer reads required to reach the lowest level index leaf page.



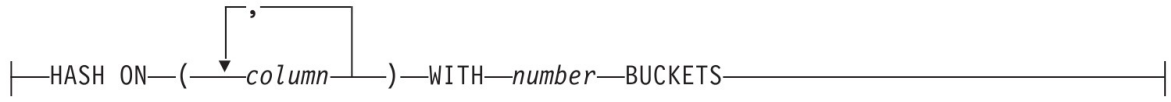
All access to the low level leaf nodes has to go through the single root node and takes 3 buffer reads to find.

The forest of trees implementation uses a hash lookup to split the range of index values into a number of buckets. Each bucket is the then the root node of a smaller subtree.



Now there is no single point of contention at the upper level and the number of buffer reads required to reach the lowest level leaf page is reduced to 2.

The use of a forest of trees is specified at time of index creation using the HASH ON clause.



The column list is used in the top level hash lookup and must be a prefix of the index columns in the main CREATE INDEX statement. Because of the hash nature of the lookup, the column list should be chosen so that it generates key values that are unique.

The number of buckets determines the number of subtrees to create and that choice will depend on the reason chosen for using a forest of trees index. If the goal is to reduce contention then use at least 2 buckets per CPU VP, more if there are many rows and/or some duplicate values.

If the intention is to reduce the number of levels in the btree then take an *onstat -pT* of the original index to assess the current number of levels and approximate number of keys per node Example: A regular btree built on 1,000,000 rows using a CHAR(20) key – all unique values. The *oncheck -pT* shows the following details for the index pages.

```
Index Usage Report for index tab1_idx on stores_demo:gcostanza.tab1

             Average    Average
Level    Total No. Keys Free Bytes
----- -------- -------- ----------
    1        1        5       1900
    2        5       50        607
    3      253       63        234
    4    16130       61        222
----- -------- -------- ----------
Total    16389       62        222
```

There are up to 63 keys per page so to reduce the number of levels by one then there needs to be at least this many buckets. Rebuilding the index as a forest of trees with 75 buckets reduces the index levels by one.

```
Index Usage Report for index tab1_fot on stores_demo:gcostanza.tab1

             Average    Average
Level    Total No. Keys Free Bytes
----- -------- -------- ----------
    1       75        9       1761
    2      747       36       1005
    3    27132       36        951
----- -------- -------- ----------
Total    27954       36        954
```

Note that the actual number of index nodes has increased because the forest of trees index is sparser than the equivalent standard btree. This is one of the trade offs when changing the index type to forest of trees.

There are also some limitations when using forest of trees indexes.

- Only standard built-in base types are supported – complex datatypes, UDTs and functional indexes are not supported.
- There is no concept of clustering when using a forest of trees index.
- You cannot use the ALTER INDEX statement.
- Because the index nodes are built from top to bottom you cannot specify a FILLFACTOR.
- A forest of trees index cannot be used to process aggregates.
- It is not possible to perform range scans directly using the HASH ON columns.

Although, as noted above, it is not possible to use a range scan on the HASH ON columns it is possible to combine equality and range scans on selected columns in the forest of trees index. If the index was created as follows:

```
CREATE INDEX tab4_fot (col1, col2) HASH ON (col1);
```

Valid index usage:

WHERE col1 = 1000
WHERE col1 = 2000 AND col2 = 26
WHERE col1 = 3000 AND col2 BETWEEN 24 AND 102

Invalid index usage:

WHERE col1 > 6000
WHERE col1 < 5000 AND col2 = 780

If you need to do this type of range scan then you will need to create an additional, standard btree index to cover those queries. Note that it cannot contain exactly the same column specification, (col1, col2) in this example. You will need to create an index with more columns – such as (col1, col2, col3) - or even less columns if that will suffice for the proposed queries.

Faster C User Defined Routines

When using C UDRs in previous Informix versions it could not be guaranteed that the Operating System would be able to dynamically load the shared object containing the C code at the same place in the process address space for all VPs in the instance. This restricted the ability of a thread running a C UDR to migrate between VPs and potentially limited performance in multiprocessor architectures.

A new configuration parameter, PRELOAD_DLL_FILE, allows you to load selected shared objects when the server is first started. You must specify a separate occurrence of this parameter for each shared object that you want to preload. The *onstat -g dll* output has been enhanced to indicate, with the P flag, which libraries have been preloaded. For example, the default behavior loads shared objects at different addresses which can be seen in the *onstat -g dll* output:

```
addr     slot vp baseaddr     flags filename
4916f310 15   1  7fcd6670f000 -- /product/11.70/extend/bts.2.00/bts.bld
4b18f310      2  7fcd66911000
4b196310      3  7fcd6670f000
4b1be310      4  0
4b1df310      5  0
4b200310      6  0
4b232310      7  0
4b254310      8  0
4b275310      9  0
```

```
4b2e5310      10 0
4bc96310      11 7fcd66911000
```

After setting PRELOAD_DLL_FILE to $INFORMIX/extend/bts.2.00/bts.bld you can see that the load address is consistent across all VPs and that the P flag has been set.

```
addr     slot vp baseaddr     flags filename
4916f310 15   1  7ff3d31a5000 PM /product/11.70/extend/bts.2.00/bts.bld
4b196310      2  7ff3d31a5000
4b19d310      3  7ff3d31a5000
4b1be310      4  7ff3d31a5000
4b1df310      5  7ff3d31a5000
4b200310      6  7ff3d31a5000
4b232310      7  7ff3d31a5000
4b254310      8  7ff3d31a5000
4b275310      9  7ff3d31a5000
4b2e5310      10 7ff3d31a5000
4bc96310      11 7ff3d31a5000
```

Note also the M flag which indicates that threads using this shared library are free to migrate between VPs. If the OS does manage to dynamically load a shared object at consistent addresses then it is also a candidate for automatic thread migration. This new behavior may cause instability in UDRs that have not been written to be aware of thread migration. To prevent thread migration in untested UDRs the *mi_udr_lock()* routine should be used to pin a thread to the current VP.

This feature is not available on Windows based platforms as it lacks the necessary supporting functions in the operating system.

Query Optimizer Support For Multi-Index Scans

Queries in earlier releases typically use no more than one index to scan each table for qualifying rows. In this release, you can specify new access-method optimizer directives so that the query optimizer can combine one or multiple B-tree indexes and the Boolean operations in the WHERE clause to fetch qualifying data rows. Using these directives can provide better performance than full-table scans, both for OLTP queries and for data warehousing applications that query large tables.

To turn on this feature on use the MULTI_INDEX directive (INDEX_ALL can be used as a synonym). To tell the optimizer to not consider this feature it can be disabled with the AVOID_MULTI_INDEX directive. There are three ways that the directive can be used.

- If you specify a table as the only argument to the directive, the optimizer considers all of the available indexes on that table, and uses all of them (or a subset) when it searches the table for qualifying rows.
- If you specify a table and only a single index, the optimizer considers using only that index to scan the table.
- If you specify a table and more than one index, the optimizer considers a search path that uses all of the specified indexes.

For example, the query plan of a single table lookup using two different indexes. One is on the id column and another is on the group_id column.

```
SELECT {+ MULTI_INDEX (tab3) } data
FROM tab3 WHERE id BETWEEN 100 AND 200 AND group_id = 34

DIRECTIVES FOLLOWED:
MULTI_INDEX ( tab3 )
DIRECTIVES NOT FOLLOWED:

Estimated Cost: 7
Estimated # of Rows Returned: 1

 1) gcostanza.tab3: MULTI INDEX PATH (SKIP SCAN)

    (1) Index Name: gcostanza.tab3_idx1
        Index Keys: id   (Serial, fragments: ALL)
        Lower Index Filter: gcostanza.tab3.id >= 100
        Upper Index Filter: gcostanza.tab3.id <= 200
  AND
    (2) Index Name: gcostanza.tab3_idx2
        Index Keys: group   (Serial, fragments: ALL)
        Lower Index Filter: gcostanza.tab3.group_id = 34
```

There are some restrictions on when a multi-index scan can be used. The transaction isolation level affects whether the MULTI_INDEX or INDEX_ALL directive can force a multi-index scan execution path, which is not available while the isolation level is Cursor Stability, or is Committed Read with the LAST COMMITTED option. (This directive is supported, however, in the Dirty Read and Repeatable Read isolation levels, and in Committed Read without the LAST COMMITTED option.) The following additional restrictions apply to multi-index scan access paths:

- The indexes must be B-tree indexes and can be attached or detached indexes.
- These directives are ignored for R-tree indexes, functional indexes, and indexes based on the Virtual Index Interface (VII).
- The table cannot be a remote table, a pseudo-table, a system catalog table, an external table, or a hierarchical table.
- A multi-index scan cannot support join predicates as index filters in the underlying index scans.
- A multi-index scan ignores all columns of a composite index except the leading column.
- DML statements that perform cascade deletes or declare statement local variables (SLVs) cannot use a multi-index scan.
- Update queries that activate a FOR EACH ROW triggered action cannot use a multi-index scan.
- In ANSI-compliant databases, the MULTI_INDEX or INDEX_ALL directive is not followed for a SELECT statement that has no ORDER BY clause, no GROUP BY clause, and no FOR READ ONLY clause, if the FROM clause specifies only a single table. (In this special case, the query has implicit cursor behavior that conflicts with a multi-index scan access path.)

Large Pages Support On Linux

Large pages for non-message shared memory segments that reside in physical memory are now supported on Linux platforms. Previously, large pages were supported only on AIX and Solaris systems. The use of large pages can provide performance benefits in large memory

configurations. To enable or disable support for large pages, use the IFX_LARGE_PAGES environment variable. Large Pages support is enabled by default on Solaris and Linux platforms.

Automatically Add CPU Virtual Processors

A new start up task, *auto_tune_cpu_vps*, instructs the server to automatically add CPU VPs to make best use of the available CPU resources. The new task simply detects the difference between the currently defined CPU VPs in the ONCONFIG file and a suggested value based on a logic calculating an optimum using the amount of available CPUs in the operating system. This scheduler task is disabled by default.

Improved Name Service Connection Time

In order to improve response times by saving a potentially expensive system call, internal caches are now used to store some OS information regarding hosts and users. The services involved are host name & service name lookups to /etc/hosts and /etc/services and user id and group id lookups. This information is always required when a connection is established, and using an internal cache improves the session connection time.

The individual internal caches used for each service are controlled by the onconfig parameter NS_CACHE which specifies in seconds the expiry time for each cache. For example, the default setting is: **NS_CACHE host=900,service=900,user=900,group=900** which will cause each cache to be checked every 15 minutes. The values can be changed dynamically using *onmode -wf* and can be disabled by setting the expiration time to 0.

The new Informix 11.70 performance features are documented in the *IBM Informix Performance Guide.*

APPENDIX A - OPENADMIN TOOL (OAT)

A large number of the new features in the Informix 11.70 server can be accessed and controlled using the OpenAdmin Tool. In addition to specific references in features mentioned above, the following is a summary of the enhancements in OpenAdmin Tool 2.70.

Enhancements To The OpenAdmin Tool

- You can monitor, administer, and optimize storage space from the *Space Administration → Storage* pages.
    - Monitor space usage for the database server, tables and indexes, spaces, chunks, and storage pool.
    - Compress, shrink, and repack tables and fragments on the Tables and Indexes page.
    - Defragment tables, fragments, and indexes on the *Tables* and *Indexes* page. Defragmenting merges extents to reduce the number of extents in a table or table fragment.
    - Configure the database server to automatically compress, shrink, repack, and defragment tables and fragments by setting the optimization policies on the *Tables and Indexes → Storage Optimization Policies* page. Set the thresholds, enable the policies, and schedule the task for the enabled policies. Monitor the status of the tasks that optimize storage space on the *Tables and Indexes → Task Status* page.
    - Configure the database server to automatically expand an existing storage space when more space is needed by creating a storage pool of directories, cooked files, and raw devices on the *Storage Pool* page. The database server uses the storage pool entries to expand a space if its free space falls below a specified threshold by extending a chunk in the space or by adding a chunk. When you create a space or add a chunk, you can use the space in the storage pool. When you drop a space or a chunk, you can return the available space to the storage pool. You can also expand a space or extend a chunk when needed, without waiting for the database server to expand the space.
    - Create or drop a space, modify the amount by which a particular space can be expanded, or expand a space on the *Spaces* page.
    - Add or drop a chunk, mark a chunk as extendable or not extendable, or extend a chunk on the *Chunks* page.

- You can use a multi-index scan when you create an external directive to apply to an SQL statement on the *SQL Explorer → SQL Profile → Optimize* page. With a multi-index scan, all the indexes for the table are used to access the table.
- You can use the star-join optimizer directives on the *SQL Explorer → SQL Profile → Optimize* page to enhance query performance in warehousing applications. The directives improve performance for data warehousing operations on tables for which star-schema dependencies exist between a fact table and a set of dimension tables.
- You can improve the efficiency of Automatic Update Statistics (AUS) by running AUS tasks in parallel through OAT. On the *Server Administration → Automatic Update Statistics → Configuration* page, you can increase the number of threads the Scheduler uses for the AUS tasks.
- You can manage replication from a central location. The *Replication* menu includes the *Clusters* page for administering high availability clusters and the *Replication* plug-in pages: *Grid, ER Domain, Node Details*, and *Replicates*.
- You can give authenticated users without operating system accounts access to an Informix database server on the *Server Administration → User Privileges* page. The user name is mapped either to an operating system account or to a default set of properties.

- The database security administrator (SECADM) can create a trusted context on the *Space Administration → Trusted Context* page. A trusted context is a database security object that defines a trusted connection between an Informix database server and an external entity, such as a middle-ware server. A trusted connection to a server permits changes in the user ID and privileges and provides a way for the tier between the client and the server to assert the identity of the client user.

Enhancements To The Schema Manager Plug-in

- You can monitor information about the statistics for tables and fragments on the *Schema Manager -> Statistics* page. This page displays the percentage of change, the date of the last statistics build and the build duration, a count of the rows that were updated, deleted, and inserted, and the column distribution statistics. The *Indexes* page displays the date of the last statistics build and the build duration. The *Fragments* page displays the changed row count.
- You can specify whether data distribution statistics are calculated only on fragments that have changed in a fragmented table or on an entire table, or you can specify that the database server determine whether to create fragment-level statistics. You can also set the threshold for recalculating the statistics.
    - Specify the granularity of data distribution statistics and the threshold for a table when you create a table with the *Create Table* wizard on the *SQL ToolBox → Schema Manager* page. You can change the specifications for a table on the *Schema Manager → Statistics* page.
    - Set the configuration parameters that control the change threshold and that enable automatically updating statistics with the STATCHANGE and AUTO_STAT_MODE configuration parameters on the *Server Administration → Configuration* page.
- You can use three additional distribution schemes when you create fragmented tables and indexes with the *Create Table* and *Create Index* wizards. With the range distribution scheme, you can fragment data based on an interval value, for example, every million customer records. With the date-range distribution scheme, you can fragment date based on a time period, for example, every three months or every year. With the list distribution strategy, you can fragment data based on a list of values. For example, you can fragment data based on the states in a country.
- You can specify that row-level events are recorded for a table when selective row-level auditing is enabled for the database server when you create a table with the *Create Table* wizard. When selective row-level auditing is enabled for the database server, the row-level events of only the selected tables are recorded in the audit trail. Selective row-level auditing can compact audit records so that they are more manageable and potentially improve database server performance.
- You can create shadow columns that Enterprise Replication (ER) uses for a primary key when you create a table with the *Create Table* wizard. If you do not want to have a primary key or want to be able to update the primary key on tables replicated by ER, you can use the ERKEY shadow columns in place of a primary key. A unique index and a unique constraint are created on the table using these columns. ER uses that index instead of requiring a primary key.
- You can specify the extent size information when you create an index with the *Create Index* wizard.
- When you create a unique index with the *Create Index* wizard, you can also make the index a constraint. You can specify that it is a primary key constraint or a unique constraint.

Enhancements To The Enterprise Replication Plug-in

- You can create and administer grids of interconnected replication servers in a domain on the *Replication → Grid* pages.

    - Create a grid, add member servers, enable source servers, and authorize users to run grid commands from the source servers.
    - View information about the grids in a domain including the members of the grid, their host and status, and whether they are a source server in the grid.
    - Modify a grid to add or remove member servers, enable and disable source servers in the grid, or authorize additional users to run grid commands.
    - Review the status of grid tasks and select commands to rerun on the *Replication → Grid → Task Status* page.
    - Rerun a grid command that failed on one or more servers in the grid. For example, if a server in the grid is offline or is not connected to the network, a grid command will fail on that server.
    - Route client connections to servers in the grid based on the quality of replicated data and transaction latency by configuring service-level agreements for a Connection Manager.
- You can temporarily stop replicating data to and from a replication server by using the *Disable Server* action on the *Replication → ER Domain* page. The replication server stops queuing and receiving replicated data. Other replication servers in the replication domain also stop queuing data to the disabled replication server. Because deleted row information on the disabled replication server is saved, you can enable any disabled replication participant servers and immediately synchronize and repair inconsistent data by using the *Enable Server* action.
- You can define a replicate for a table that does not have a primary key if the table contains the ERKEY shadow columns. On the *Replication → Replicates* page, in the *Define New Replicate* wizard, you can select the table to include as a participant in the replicate. The ERKEY shadow columns are used in place of a primary key.
- You can repair replication inconsistencies based on time stamps on the *Replication → Replicates → Replicate Sets* and *Replicates* pages. If you have a replication domain with multiple master servers and your conflict resolution rule is time stamp or delete wins, you can repair inconsistencies based on the latest time stamps. In previous releases, you chose a master server to act as the correct version of the data and the repair action made the data of the participants match the data of the master server.

APPENDIX B – ONCONFIG CHANGES

The following new parameters have been introduced with Informix 11.70.

Parameter	Description	Default Value
FULL_DISK_INIT	Specifies if oninit -i can run: 0 allows full disk initialization only if no instance is detected at the rootchunk location. 1 required if an existing instance is detected at the rootchunk location.	0
NS_CACHE	The number of seconds for IDS name service cache (host, service, user, group) expiration time. 0 to disable cache.	host=900,service=900, user=900,group=900
BAR_CKPTSEC_TIMEOUT	Time in seconds to wait for an archive checkpoint to complete in the secondary server.	15
PRELOAD_DLL_FILE	Specifies a C UDR shared library path name to load when the server starts. Each shared library file needs a separate PRELOAD_DLL_FILE entry.	
AUTO_STAT_MODE	Enables (1) or disables (0) update statistics automatic mode. In automatic mode, statistics of table, fragment or index are rebuilt only if existing statistics are considered stale. A table, fragment or index can change by STATCHANGE percentage before its statistics are regarded as stale.	1
STATCHANGE	In automatic mode, rebuild statistics only for table, fragment or index changed by STATCHANGE percentage since last statistics run.	10
BATCHEDREAD_INDEX	Turn on/off xps api for index scans.	1
BATCHEDREAD_KEYONLY	Turn on/off xps api for key-only index scans. Only takes effect if BATCHEDREAD_INDEX is off.	0
CDR_DELAY_PURGE_DTC	Specifies the time at which delete table purge can be delayed.	0
CDR_LOG_LAG_ACTION	Specifies the action when ER log processing lags behind the current log Separate multiple actions with a plus sign (+). Actions are prioritized	ddrblock

Parameter	Description	Default Value
	from left to right.	
CDR_LOG_STAGING_MAXSIZE	Maximum size, in KB (default), MB, GB, or TB, that ER can use to stage log files in the directory specified by the LOG_STAGING_DIR configuration parameter. ER temporarily stops staging log files, at a log file boundary, when the staging directory reaches this value.	0
FAILOVER_TX_TIMEOUT	Specifies the timeout for a failover to take before transaction survival is abandoned.	0
ENABLE_SNAPSHOT_COPY	Specifies whether we can clone this instance Directly to another machine using the Snapshot Clone facility. 1 - Enable snapshot copies 0 - Disable snapshot copies	0
SMX_COMPRESS	Controls the network interface compression level. Acceptable values are: -1 - Never 0 - None 1-9 - Compression level	0
USERMAPPING	Control access to IDS for users without operating system accounts. OFF - users without operating system accounts cannot use IDS BASIC - users without operating system accounts can use IDS but not as privileged users ADMIN - users without operating system accounts can use IDS as privileged users	OFF
SP_AUTOEXPAND	When set to 1, IDS will automatically expand spaces that are low on or out of free pages. Set this param to 0 to disable automatic chunk extensions and chunk additions.	1
SP_THRESHOLD	Minimum amount of free space in a DBspace, BLOBspace, or Smart BLOBspace before the space will automatically be expanded. Value is a decimal, and can be an absolute number of kilobytes or a percentage of the total size in the DBspace.	0 - Disabled

Parameter	Description	Default Value
SP_WAITTIME	Access to the storage pool is serialized. When one thread is accessing the storage pool, SP_WAITIME is the number of seconds another thread will wait before giving up on its own access.	30
DEFAULTESCCHAR	The default escape character. If not defined, '\' (the fixed setting in previous versions) is used as escape character. Acceptable values: 'NONE' - no default escape character c - any one-character value	\
MQSERVER	Specifies the location of the WebSphere MQ server and the communication method to be used	
MQCHLLIB	Specifies the path to the directory containing the client channel definition table	
MQCHLTAB	Specifies the name of the client channel definition table	

In addition to these new parameters there are changes to some existing parameters. The default value for CONVERSION_GUARD is now 2. The VPCLASS parameter now as an extended syntax to specify processor affinity.

aff=<CPU num> | <startcpu>-<endcpu> | (<startcpu>-<endcpu>/<skip>)

So “num=4,aff=(1-10/3)” means assign 4 CPU VPs to processors 1,4,7,10.

REFERENCES

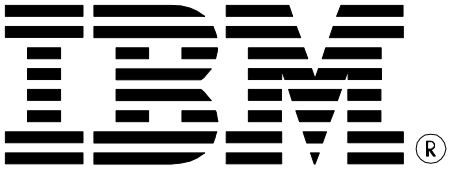
The Informix 11.70 Information Center integrates the entire Informix 11.70 and CSDK 3.70 documentation sets into an HTML frameset, with full text search, logical categories, easy navigation, and links to troubleshooting and support files.

The following IBM Manuals and Guides were used as references for this paper.

IBM Informix Installation Guide for UNIX, Linux, and Mac OS X
IBM Informix Installation Guide for Windows
IBM Informix Administrator's Guide
IBM Informix Administrator's Reference
IBM Informix Backup and Restore Guide
IBM Informix Embeddability Guide
IBM Informix Enterprise Replication Guide
IBM Informix Guide to SQL: Reference
IBM Informix Guide to SQL: Syntax
IBM Informix Migration Guide
IBM Informix Performance Guide
IBM Informix Security Guide
IBM Informix Spatial DataBlade Module User's Guide
IBM Informix Database Extensions User's Guide
IBM Informix User-Defined Routines and Data Types Developer's Guide

The OpenAdmin Tool and associated documentation is available from www.openadmintool.com.

The IBM website contains more information on the IBM Data Studio and Optim Development Studio products which enhance the development and administration of Informix products.



Informix 11.70
The Technical White Paper

October 12, 2010
Simon David – cosmo@uk.ibm.com
Shawn Moe – smoe@uk.ibm.com

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.