

# The Impact of DBXten on IBM Informix Performance

## Table of Contents

Executive Summary .....	1
Introduction.....	2
Computing Environment.....	2
Server .....	2
Informix .....	3
The Data.....	4
Native Informix Table.....	5
Native Informix B-Tree Indexes.....	6
Native Informix R-Tree Index .....	7
DBXten Table .....	7
DBXten Index .....	8
Load Trials .....	8
Native Informix Load Trials .....	8
DBXten Load Trials.....	9
Loading Time and Space Comparison.....	10
Query Trials .....	10
Native Informix Queries .....	11
DBXten Queries.....	12
Basic Interface .....	12
VTI Interface.....	13
Query Time Comparison.....	13
Concurrent Query Testing.....	16
Acknowledgements.....	17
Appendix A – onconfig File .....	17
Appendix B – Abridged NetCDF File Header (ncdump).....	38
Appendix C – Source of fetchData Program .....	39

## Executive Summary

A 1.447 billion row scientific dataset was used to compare loading and retrieval performance of IBM Informix Dynamic Server, with and without the Barrodale Computing Services [DBXten DataBlade](#) — which is a database extension that can easily be plugged into other popular DBMSs too. “Native Informix” (i.e., Informix *without* DBXten) was used with its new compression features introduced by IBM in 2008. DBXten also uses compression, but it first loads the data into the database in a modular and highly efficient way using a form of “chunking”.

These comparison tests showed that (i) native Informix (with compression) required about 25 times more storage space than Informix with the DBXten DataBlade, and (ii) load times into an indexed table were typically more than 3 times faster for Informix with DBXten than for native Informix; for example, on a powerful IBM computer, native Informix took over 10 hours to load the full 1.447 billion row table, whereas Informix with DBXten loaded this same dataset in less than 3 hours.

The space savings ( $1/25$ ) and load time savings ( $1/3$ ) did not come at the cost of great complexity (e.g., DBXten compresses data automatically) or at the cost of degraded query performance. On the contrary, for multi-dimensional queries DBXten improved the performance of Informix, *sometimes by more than an order of magnitude*. Furthermore, in concurrent query testing on the 1.447 billion row table with N simulated concurrent users, the performance gains from using DBXten became more pronounced as N increased through eleven chosen values ranging from 1 to 100. In fact, for all  $N > 5$  these concurrency performance gains were *always by more than an order of magnitude*.

## Introduction

The purpose of this report is to compare the load and query performance of IBM Informix with and without the Barrodale Computing Services (BCS) [DBXten](#) [DataBlade](#)<sup>1</sup>. The computational trials detailed here were run on a large IBM System x configuration (described in the next section) located at the IBM Innovation Center in San Mateo, California.

Of particular interest was how performance would scale as the number of rows in the database rose from about 55 million to over 1.4 billion. For each of the six table sizes in this range, tables were loaded and a set of five queries were executed, firstly by a single user and then by (simulated) multiple concurrent users. The load times and query times, with and without DBXten, were then compared. In short, DBXten scaled very well. Loading with DBXten required much less space (by a factor of 25 in total — largely because of a factor of 600 in index space reduction) and load times were always faster (by a factor of more than 3). Query times with DBXten were faster by factors ranging from slightly more than 1 to about 100. Concurrency performance trials were run only on the largest table; these results clearly demonstrated that multiple concurrent users can obtain answers to their individual queries much faster (about 15 times faster on average) when Informix is enhanced with DBXten.

The body of this report contains technical specifications of the comparison testing, and the three Appendices provide further details on the Informix configuration parameters, sample data file header information, and the querying source code for when the DBXten C API was used. For readers of this report who wish to focus on the outcomes of our testing, we direct attention to the Tables on pages 8, 10, and 13–16.

## Computing Environment

### Server

#### [IBM eServer xSeries 3850-M2](#)

**CPUs:** 4 Intel XEON Quad-Core, 2.4 GHz / 1066MHz FSB, 2x3MB L2Cache, 64 bit

**RAM:** 16x 2GB PC2-5300 667MHz registered 240-pin ECC DDR2 SDRAM DIMMs

**OS:** Linux kernel 2.6.18

**Disk:** 1x73.4GB 10000rpm root (cooked) filesystem

**Storage Server:** IBM DS4700 Dual Controller Storage server, 2GB cache, 32 x 147GB 15000rpm fiber channel drives, configured as 16 Raid 1 146GB logical volumes. Each logical volume was accessed by Informix as a raw disk.

---

<sup>1</sup> Update: In December 2011 BCS was awarded US Patent 8077059 for DBXten.

## Informix

Informix Dynamic Server version 11.50FC6 was used throughout, and the `onconfig` file used is presented in [Appendix A](#). This file was defined by IBM at the start of the trials, as part of the installation of the Informix software on the testbed machine. The following subsequent changes were made to reflect the multiprocessor nature and general scale of the testbed machine:

```
MULTIPROCESSOR 1 # was 0
VPCCLASS cpu,num=10,noage # was num=1
CLEANERS 16 # was 8
PHYSFILE 10240000
```

Number of `BUFFERS` in `BUFFERPOOL` statements was set to 1000000.  
 Logical log size<sup>2</sup> was set to 700000 kb.

Informix `dbspaces` and `sbspaces` were defined as follows:

Name	Type	Size (kb)	Offset (kb)	Location
rootdbs	dbspace	2,000,000	4	raw partition 1
physlogdb	dbspace (physical log)	40,000,000	2,000,004	raw partition 1
loglogdb	dbspace (logical logs)	5,000,000	42,000,004	raw partition 1
temp1db	dbspace (temp)	10,000,000	4	raw partition 2
temp1sb	sbspace (temp)	10,000,000	4	raw partition 3
temp1sbifmx	sbspace (temp)	100,000	4	raw partition 3
miscdbtabsdg	dbspace	100,000,000	4	raw partition 4
miscsbtabsdg	sbspace	40,000,000	100,000,004	raw partition 4
miscdbtabsdh	dbspace	100,000,000	4	raw partition 5
miscsbtabsdh	sbspace	40,000,000	100,000,004	raw partition 5
miscdbtabsdi	dbspace	100,000,000	4	raw partition 6
miscsbtabsdi	sbspace	40,000,000	100,000,004	raw partition 6
miscdbtabsdj	dbspace	100,000,000	4	raw partition 7
miscsbtabsdj	sbspace	40,000,000	100,000,004	raw partition 7
miscdbtabsdk	dbspace	100,000,000	4	raw partition 8
miscsbtabsdk	sbspace	40,000,000	100,000,004	raw partition 8
miscdbtabsdl	dbspace	100,000,000	4	raw partition 9
miscsbtabsdl	sbspace	40,000,000	100,000,004	raw partition 9
miscdbtabsdm	dbspace	100,000,000	4	raw partition 10
miscsbtabsdm	sbspace	40,000,000	100,000,004	raw partition 10
miscdbtabsdn	dbspace	100,000,000	4	raw partition 11
miscsbtabsdn	sbspace	40,000,000	100,000,004	raw partition 11
miscdbtabsdo	dbspace	100,000,000	4	raw partition 12

<sup>2</sup> Logical logging was used only when issuing the compression commands described [here](#).

miscsbtabdsdo	sbspace	40,000,000	100,000,004	raw partition 12
miscsbindsdp	dbspace	140,000,000	4	raw partition 13
miscsbindsdq	dbspace	140,000,000	4	raw partition 14
miscsbindsds	dbspace	140,000,000	4	raw partition 15

**Table 1: Informix Dbspace and Sbspace Definitions**

## The Data

The data was generated using a high resolution model developed by the Ocean Circulation and Climate Advanced Modelling Project, using the OCCAM data selector at <http://www.noc.soton.ac.uk/JRD/OCCAM/EMODS/select.php>. The parameters shown in the following image were used to direct the generation of data:

**Figure 1: OCCAM Data Selector, with values filled in.**

This request<sup>3</sup> resulted in the generation of 105 netCDF files, one for each month between April 1996 and December 2004. The header information from one of these netCDF files is provided in [Appendix B](#). Each of the netCDF files was then converted to a comma-separated (CSV) text file, with each line containing values from the following variables:

- TIMESTEP

<sup>3</sup> The request was actually submitted as three individual requests, covering different periods of time.

- DEPTH
- LATITUDE\_T
- LONGITUDE\_T
- POTENTIAL\_TEMPERATURE\_\_MEAN\_\_
- SALINITY\_\_MEAN\_\_

### ***Native Informix Table***

The native Informix tables had the following schema:

<b>Column</b>	<b>Type</b>
timeval	integer
depth	float
latitude	float
longitude	float
temperature	float
salinity	float

**Table 2: Native Informix Table Schema**

The tables were defined using the following SQL<sup>4</sup>:

```
CREATE TABLE "barrodale".occam_conventional_hpl_ind_comp_5N
(
  timeval integer,
  depth float,
  latitude float,
  longitude float,
  temperature float,
  salinity float
)
```

<sup>4</sup> In early tests, the following fragmentation by expression scheme was used in an effort to reduce the amount of disk that needed to be searched in answering particular spatial queries:

```
FRAGMENT BY EXPRESSION
  ((latitude > -30 ) AND (longitude < 73 ) ) IN miscdbtabsdg,
  (((latitude > -30 ) AND (longitude < 85 ) ) AND (longitude
    >= 73 ) ) IN miscdbtabsdh,
  ((latitude > -30 ) AND (longitude >= 85 ) ) IN miscdbtabsdj,
  (((latitude > -43 ) AND (latitude <= -30 ) ) AND (longitude
    < 73 ) ) IN miscdbtabsdk,
  (((latitude > -43 ) AND (latitude <= -30 ) ) AND (longitude
    < 85 ) ) AND (longitude >= 73 ) ) IN miscdbtabSDL,
  (((latitude > -43 ) AND (latitude <= -30 ) ) AND (longitude
    >= 85 ) ) IN miscdbtabSDM,
  ((latitude <= -43 ) AND (longitude < 73 ) ) IN miscdbtabSDN,
  (((latitude <= -43 ) AND (longitude < 85 ) ) AND (longitude
    >= 73 ) ) IN miscdbtabSDO,
  ((latitude <= -43 ) AND (longitude >= 85 ) ) IN miscdbtabSDP
```

However, this scheme led to inferior load and query times for native Informix and hence was abandoned in favor of round robin fragmentation.

```

FRAGMENT BY ROUND ROBIN IN miscdbtabsdg, miscdbtabsdh, miscdbtabstdi,
miscdbtabstdj, miscdbtabstdk, miscdbtabstdl, miscdbtabstdm, miscdbtabstdn,
miscdbtabstdo , miscdbbindsdp, miscdbbindsdq, miscdbbindsds
EXTENT SIZE $SIZE NEXT SIZE $SIZE LOCK MODE PAGE;

```

The values \$N and \$SIZE are given in the following table:

Actual number of rows	\$N	\$SIZE
55,107,216	50,000,000	100,000
110,214,432	100,000,000	200,000
220,438,944	200,000,000	400,000
440,877,888	400,000,000	800,000
881,755,776	800,000,000	1,600,000
1,446,630,570	1,600,000,000	3,200,000

**Table 3: Values of \$N and \$SIZE**

The value of \$SIZE was chosen in order to reduce the number of extents.

## ***Native Informix B-Tree Indexes***

For each Informix table four B-Tree indexes were built, one on each of the spatio-temporal columns timeval, latitude, longitude, and depth. The following SQL was used to define (attached) indexes on the tables<sup>5</sup>:

<sup>5</sup> In initial tests, the indexes were fragmented by expression:

```

CREATE INDEX "barrodale".occam_conventional_hpl_ind_comp_$SIZE_depth_idx
ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
(depth) FILLFACTOR 50
FRAGMENT BY EXPRESSION
(depth < 20000 ) IN miscdbbindsdp,
((depth < 2000000 ) AND (depth >= 20000 ) ) IN miscdbbindsdq,
(depth >= 2000000 ) IN miscdbbindsds;

CREATE INDEX "barrodale".occam_conventional_hpl_ind_comp_$SIZE_dt_idx
ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
(timeval) FILLFACTOR 50
FRAGMENT BY EXPRESSION
(timeval < 1000000 ) IN miscdbbindsdp,
((timeval < 5000000 ) AND (timeval >= 1000000 ) ) IN miscdbbindsdq,
(timeval >= 5000000 ) IN miscdbbindsds;

CREATE INDEX "barrodale".occam_conventional_hpl_ind_comp_$SIZE_latitude_idx
ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
(latitude) FILLFACTOR 50
FRAGMENT BY EXPRESSION
(latitude > -30 ) IN miscdbbindsdp,
((latitude > -43 ) AND (latitude <= -30 ) ) IN miscdbbindsdq,
(latitude <= -43 ) IN miscdbbindsds;

CREATE INDEX "barrodale".occam_conventional_hpl_ind_comp_$SIZE_longitude_idx
ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
(longitude) FILLFACTOR 50
FRAGMENT BY EXPRESSION
(longitude < 73 ) IN miscdbbindsdp,
((longitude < 85 ) AND (longitude >= 73 ) ) IN miscdbbindsdq,
(longitude >= 85 ) IN miscdbbindsds;

```

```

CREATE INDEX
"barrodale".occam_conventional_hpl_ind_comp_$SIZE_depth_idx
  ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
    (depth) FILLFACTOR 50;

CREATE INDEX "barrodale".occam_conventional_hpl_ind_comp_$SIZE_dt_idx
  ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
    (timeval) FILLFACTOR 50;

CREATE INDEX
"barrodale".occam_conventional_hpl_ind_comp_$SIZE_latitude_idx
  ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
    (latitude) FILLFACTOR 50;

CREATE INDEX
"barrodale".occam_conventional_hpl_ind_comp_$SIZE_longitude_idx
  ON "barrodale".occam_conventional_hpl_ind_comp_$SIZE
    (longitude) FILLFACTOR 50;

```

## ***Native Informix R-Tree Index***

In addition to using the B-Tree indexes described in the previous section, some experimentation on the use of a four-dimensional R-Tree index was done as well. The following SQL was used to define this index:

```

CREATE FUNCTION createbox(timeval integer,latitude float,longitude
float ,depth float)
RETURNING DSBox WITH (NOT VARIANT)
  RETURN DSRangeToBox('timeval '||timeval||' '||timeval||','||
    'latitude '||latitude||' '||latitude||','||
    'longitude '||longitude||' '||longitude||','||
    'depth '||depth||' '||depth)::DSBox;

END FUNCTION;

CREATE INDEX occam_conventional_hpl_ind_comp_$SIZE_rtree_idx ON
  occam_conventional_hpl_ind_comp_$SIZE
  (createbox(timeval,latitude,longitude,depth) dsbox_ops)
  USING RTREE IN miscdbindsdp;

```

Two trials involving R-Tree indexes were performed: one where the index was created on the table *before* it was loaded and one where the index was created *after* it was loaded.

## ***DBXten Table***

For the DBXten trials the table had a single column, a so-called “DSChip” type column with the following schema definition:

Column	Type	Precision
--------	------	-----------

---

This scheme was abandoned when it was decided to fragment tables by round robin.

timeval	integer	
depth	float	.01
latitude	float	.001
longitude	float	.001
temperature	float	.01
salinity	float	.000001

**Table 4: DBXten DSChip Schema**

One of the features of DBXten is that it is able to exploit any lack of precision in a particular dataset. The precision values shown above were chosen to correspond with the actual precision of the data in the input dataset (as converted from the original netCDF files).

The following SQL, using round robin fragmentation, was used to define the table:

```
CREATE TABLE occam_chips_${SIZE}_0_TM(occamchip dschip) IN miscdbtabsdg
PUT occamchip IN
(miscsbtabsdg,miscsbtabsdh,miscsbtabsdi,miscsbtabsdj,miscsbtabsdk,
miscsbtabsdl,miscsbtabsdm,miscsbtabsdn,miscsbtabsdo);
```

## ***DBXten Index***

The following SQL was used to define an R-Tree index on the DSChip's:

```
CREATE FUNCTION occam_cube(chipIn DSChip)
RETURNING dsbox WITH (NOT VARIANT)
RETURN
DSAsBoxString(chipIn, 'timeval,latitude,longitude,depth')::dsbox;
END FUNCTION;
CREATE INDEX occam_chips_${SIZE}_0_TM_idx ON
occam_chips_${SIZE}_0_TM(occam_cube(occamchip) dsbox_ops)
USING rtree IN miscdbindsds;
```

## **Load Trials**

### ***Native Informix Load Trials***

The following table defines the loading trials that were performed on native Informix tables:

<b>Case Name<sup>6</sup></b>	<b>Rows (N)</b>	<b>netCDF Files</b>	<b>Indexes</b>	<b>Pre-indexed?</b>
5M_R_E	4,745,800	< 1	1 R-Tree	yes
55M_R_E	55,107,216	4	1 R-Tree	no

<sup>6</sup> The first component of the case name indicates the number of rows. The second component indicates whether R-Tree (R) or B-Tree (B) indexes were built on the native Informix tables. The third component indicates whether fragmentation was by expression (E) or round robin (R).

55M_B_R	55,107,216	4	4 B-Tree	yes
110M_B_R	110,214,432	8	4 B-Tree	yes
220M_B_R	220,438,944	16	4 B-Tree	yes
440M_B_R	440,877,888	32	4 B-Tree	yes
880M_B_R	881,755,776	64	4 B-Tree	yes
1450M_B_R	1,446,630,570	105	4 B-Tree	yes

**Table 5: Load Trials – Native Informix**

Each of these loads was performed using the High-Performance Loader in “Deluxe Mode”. Although the High-Performance Loader offers a much faster “Express Mode”, which disables indexes during loads, this option was not used since it seemed inappropriate for an environment (such as a sea-floor observatory) where data is being ingested continuously and where there is a continuous need to be able to query the data.

[Informix row compression](#) was used throughout. Specifically, the following procedure was followed:

- 1) Load 3,000,000 rows into the previously empty, non-compressed table.
- 2) Turn on compression using:  

```
execute function sysadmin:task("table compress repack shrink",tablename)
```
- 3) Load the remaining rows.

This procedure ensured that a dictionary of adequate size was built to allow the remaining rows to be effectively compressed.

## ***DBXten Load Trials***

The following table defines the loading trials that were performed on the DBXten tables; the case names reflect the name of the comparable native Informix load trial:

<b>Case Name</b>	<b>Rows (N)</b>	<b>netCDF Files</b>	<b>Indexes</b>	<b>Pre-indexed?</b>
5M_R_E	4,745,800	<1	1 R-Tree	yes
55M_R_E	55,107,216	4	1 R-Tree	no
55M_B_R	55,107,216	4	1 R-Tree	yes
110M_B_R	110,214,432	8	1 R-Tree	yes
220M_B_R	220,438,944	16	1 R-Tree	yes
440M_B_R	440,877,888	32	1 R-Tree	yes
880M_B_R	881,755,776	64	1 R-Tree	yes
1450M_B_R	1,446,630,570	105	1 R-Tree	yes

**Table 6: Load Trials - DBXten**

The DBXten tables were loaded with a CSV file loading client program, which is distributed with the DBXten product.

Note that each of the DBXten trials in the table above has a corresponding native Informix load trial to which load times and space consumption can be compared. These comparisons were performed, and the results are presented in the next section.

### ***Loading Time and Space Comparison***

The following table presents the load times and table and index space consumption for each of the trials:

Case	Total Space (2K pages)			Index Space (2K pages)			Load Times (hh:mm:ss)		
	Native	DBXten	Ratio	Native	DBXten	Ratio	Native	DBXten	Ratio
5M_R	374,745	10,578	0.0282	305,094	224	0.000734	15:15:58	0:00:31	0.00057
55M_B_R	2,494,568	102,164	0.0410	1,532,313	2,752	0.001796	0:21:40	0:06:58	0.32180
55M_R	4,573,080	102,176	0.0223	3,588,012	2,764	0.000770	3:46:29	0:06:08	0.02709
110M_B_R	5,084,128	204,109	0.0401	3,102,923	5,492	0.001770	0:42:18	0:13:44	0.32448
220M_B_R	10,259,802	407,635	0.0397	6,241,563	10,988	0.001760	1:27:33	0:30:13	0.34511
440M_B_R	20,650,945	815,447	0.0395	12,538,938	21,036	0.001678	3:00:37	1:00:22	0.33419
880M_B_R	41,341,033	1,629,190	0.0394	25,086,858	42,123	0.001679	6:08:05	1:47:11	0.29119
1450M_B_R	68,032,078	2,673,376	0.0393	41,255,285	69,060	0.001674	10:08:22	2:56:28	0.29007

**Table 7: Load Time and Space Comparison**

For the cases where B-Trees were used with native Informix the results were quite consistent. For these six cases, total space ratios (DBXten/Native) ranged from .0393 to .0410, index space ratios ranged from .000167 to .000180 and load time ratios ranged from 0.290 to 0.325.

R-Tree indexes were expected to increase query performance, but the cost in load times is prohibitive. It took over 15 hours to load (using HPL) fewer than 5 million records into an R-Tree-indexed native Informix table. When building the index was deferred until after the load was finished, it still took over 3.75 hours to load 55 million records. These times compare with 31 seconds and 6.5 minutes, respectively, when using DBXten.

## **Query Trials**

Query performance, with and without DBXten, was assessed by running a variety of queries against each of the tables loaded as described in the previous sections.

For each of the cases listed in [Table 7](#), five queries were issued in succession against the native Informix tables and then against the DBXten tables. In between each query invocation the Informix engine was stopped and restarted with

```
onmode -ky
oninit
onmode -e flush
onmode -F
```

and file caches were cleared by issuing the following Unix commands:

```
sync
sync
echo 1 > /proc/sys/vm/drop_caches
```

Raw disk caches were cleared using the command:

```
dd of=/dev/null if=/dev/devicename count=1000 bs=1M
```

applied to each of the raw disk devices. Four of the queries – Q1 through Q4 – were basic spatial-temporal queries, requesting all the records that fell within a particular spatial-temporal cube. These cubes varied in both size and shape. The fifth query – Q5 – requested all the records that fell inside a particular cube and that also satisfied conditions on the two other columns. The dimensions of the cubes for each of the queries are shown in the following table:

Query	Time Bounds	Depth Bounds	Latitude Bounds	Longitude Bounds
Q1	1,297,200 to 1,297,300 (small)	266.8 to 266.9 (small)	-35 to -25 (medium)	65 to 75 (medium)
Q2	1,297,200 to 1,297,300 (small)	266.8 to 266.9 (small)	-56.25 to -56.15 (small)	64.25 to 64.3 (small)
Q3	1,000,000 to 10,000,000 (large)	100,000 to 400,000 (large)	-56.25 to -56.15 (small)	64.25 to 64.3 (small)
Q4	1,000,000 to 10,000,000 (large)	200 to 3,000 (medium)	-55 to -50 (medium)	65 to 70 (medium)
Q5	1,279,700 to 1,279,700 (small)	3,000 to 4,000,000 (large)	-35 to -15 (large)	70 to 80 (large)

**Table 8: Query Cube Sizes and Shapes**

For Q5, salinity was further restricted to values  $< -.00055$  and temperature was further restricted to values  $> 27.5$ .

The number of rows returned by each of the queries, for each of the cases, is shown in the following table:

Case	Q1	Q2	Q3	Q4	Q5
55M_B_R	14,400	1	76	72,000	2
110M_B_R	14,400	1	152	144,000	2
220M_B_R	14,400	1	304	288,000	2
440M_B_R	14,400	1	608	576,000	2
880M_B_R	14,400	1	1,216	1,152,000	2
1450M_B_R	14,400	1	1,995	1,890,000	2

**Table 9: Number of Rows Returned by Each Query**

## ***Native Informix Queries***

For the native Informix cases the following SQL was run:

```
UNLOAD TO file SELECT
```

```
latitude, longitude, timeval, depth, temperature, salinity
FROM occam_conventional_hpl_ind_comp_${SIZE}
WHERE latitude BETWEEN $MINLAT AND $MAXLAT AND
      longitude BETWEEN $MINLONG AND $MAXLONG AND
      timeval BETWEEN $MINDATE AND $MAXDATE AND
      depth BETWEEN $MINDEPTH AND $MAXDEPTH AND
      temperature > $MINTEMP AND salinity < $MAXSAL ;
```

No explicit query optimizer directives were provided, but the following SQL commands were issued in between loading the table and starting the queries:

```
UPDATE STATISTICS FOR TABLE occam_conventional_hpl_ind_comp_${SIZE};
UPDATE STATISTICS HIGH FOR TABLE
      occam_conventional_hpl_ind_comp_${SIZE}(timeval);
UPDATE STATISTICS HIGH FOR TABLE
      occam_conventional_hpl_ind_comp_${SIZE}(latitude);
UPDATE STATISTICS HIGH FOR TABLE
      occam_conventional_hpl_ind_comp_${SIZE}(longitude);
UPDATE STATISTICS HIGH FOR TABLE
      occam_conventional_hpl_ind_comp_${SIZE}(depth);
```

Executing these commands should allow the SQL query optimizer to select the best access path for executing the queries. The following table lists, for each case and query, which B-Tree index (if any) was used to answer the query:

Case	Q1	Q2	Q3	Q4	Q5
55M_B_R	depth	latitude	latitude	depth	none
110M_B_R	depth	latitude	latitude	depth	none
220M_B_R	depth	latitude	latitude	none	timeval
440M_B_R	depth	latitude	latitude	none	timeval
880M_B_R	timeval	latitude	latitude	none	timeval
1450M_B_R	timeval	timeval	latitude	none	timeval

**Table 10: Indexes Used to Answer Queries**

## ***DBXten Queries***

DBXten has two interfaces for querying data. The basic interface exposes the DSChip data type and the programmer has to convert DSChip's to conventional columns. The other uses Informix's Virtual Table Interface; it provides the illusion that a table of DSChip's contains conventional columns — the benefit being that it allows the SELECT statements in programs to remain virtually unchanged when DBXten is installed.

### **Basic Interface**

The following command was used to perform the basic interface queries:

```
fetchData -table occam_chips_${SIZE}_0_tM -chip occamchip \
```

```

-boxcolumn 'occam_cube(occamchip)' \
-columns timeval,depth,latitude,longitude,temperature,salinity \
-key "timeval,$MINDATE,$MAXDATE" \
-key "Latitude,$MINLAT,$MAXLAT" \
-key "Longitude,$MINLONG,$MAXLONG" \
-key "Depth,$MINDEPTH,$MAXDEPTH" \
-key "Temperature,$MINTEMP,100.0" \
-key "Salinity,-1.0,$MAXSAL" >& file

```

The source code for the `fetchData` program, which is written using the DBXten C API, is provided in [Appendix C](#).

## VTI Interface

The following SQL was used to create the virtual table:

```

DROP TABLE occam_tempschemasource;
CREATE TABLE occam_tempschemasource(schematext lvarchar);
INSERT INTO occam_tempschemasource VALUES(' (maxtuples
1000,timeval integer,depth double .01,latitude double .001,longitude
double .001,temperature double .01,salinity double .0000001) ');

DROP TABLE occam_chips_${size}_0_tm_vti;
CREATE TABLE occam_chips_${size}_0_tm_vti(
    timeval integer,
    depth float,
    latitude float,
    longitude float,
    temperature float,
    salinity float)
USING DSChipAccess(
    basetable='occam_chips_${size}_0_tm',
    dschipcolumn='occamchip',keylist=
'occam_cube(occamchip):(timeval,latitude,longitude,depth)',
    schemasource='occam_tempschemasource',
    usescalarkeys='0');

```

The native Informix SQL [shown earlier](#) (modified to point to this virtual table) was then used to select data.

## Query Time Comparison

This section presents the timing results for the five queries and the various cases tested<sup>7</sup>.

---

<sup>7</sup> The queries are defined in [Table 8](#) on page 11.

<b>Q1</b>					
<b>Case</b>	<b>Native</b>	<b>DBXten Basic</b>	<b>DBXten VTI</b>	<b>Basic Ratio</b>	<b>VTI Ratio</b>
55M_B_R	00:05.3	00:01.7	00:02.1	0.32	0.40
110M_B_R	00:06.2	00:01.6	00:01.8	0.26	0.29
220M_B_R	00:10.3	00:01.7	00:02.0	0.16	0.20
440M_B_R	00:17.1	00:01.6	00:01.9	0.09	0.11
880M_B_R	00:11.1	00:01.3	00:02.2	0.12	0.20
1450M_B_R	00:11.2	00:01.6	00:02.0	0.14	0.18

Table 11: Query Timings for Query 1

<b>Q2</b>					
<b>Case</b>	<b>Native</b>	<b>DBXten Basic</b>	<b>DBXten VTI</b>	<b>Basic Ratio</b>	<b>VTI Ratio</b>
55M_B_R	00:08.3	00:01.0	00:01.1	0.12	0.13
110M_B_R	00:14.8	00:01.2	00:01.1	0.08	0.07
220M_B_R	00:19.7	00:01.4	00:01.0	0.07	0.05
440M_B_R	00:37.9	00:01.4	00:01.4	0.04	0.04
880M_B_R	01:24.0	00:01.4	00:01.4	0.02	0.02
1450M_B_R	02:10.3	00:01.2	00:00.9	0.01	0.01

Table 12: Query Timings for Query 2

<b>Q3</b>					
<b>Case</b>	<b>Native</b>	<b>DBXten Basic</b>	<b>DBXten VTI</b>	<b>Basic Ratio</b>	<b>VTI Ratio</b>
55M_B_R	00:09.3	00:02.8	00:03.0	0.30	0.32
110M_B_R	00:11.3	00:04.3	00:04.6	0.38	0.41
220M_B_R	00:22.3	00:07.9	00:07.9	0.35	0.35
440M_B_R	00:38.9	00:14.8	00:14.4	0.38	0.37
880M_B_R	01:17.2	00:28.6	00:29.3	0.37	0.38
1450M_B_R	02:11.7	00:45.6	00:44.4	0.35	0.34

Table 13: Query Timings for Query 3

Q4					
Case	Native	DBXten Basic	DBXten VTI	Basic Ratio	VTI Ratio
55M_B_R	00:11.6	00:03.3	00:06.1	0.29	0.52
110M_B_R	00:13.1	00:05.6	00:11.1	0.43	0.84
220M_B_R	00:47.8	00:10.1	00:20.6	0.21	0.43
440M_B_R	01:45.4	00:19.9	00:40.1	0.19	0.38
880M_B_R	03:16.8	00:36.9	01:30.9	0.19	0.46
1450M_B_R	05:12.8	01:08.7	02:23.1	0.22	0.46

Table 14: Query Timings for Query 4

Q5					
Case	Native	DBXten Basic	DBXten VTI	Basic Ratio	VTI Ratio
55M_B_R	00:12.3	00:07.4	00:14.5	0.60	1.18
110M_B_R	00:24.6	00:07.7	00:14.3	0.32	0.58
220M_B_R	00:14.9	00:08.0	00:14.7	0.54	0.98
440M_B_R	00:11.2	00:07.9	00:14.7	0.71	1.31
880M_B_R	00:11.4	00:07.8	00:15.3	0.69	1.35
1450M_B_R	00:12.3	00:07.7	00:14.3	0.63	1.16

Table 15: Query Timings for Query 5

The tables above indicate that, in general, DBXten, with either the Basic or VTI Interfaces, performs queries as fast, or faster, than when using just native Informix. The only exception is in Query 5, where using the VTI interface is somewhat slower using native Informix alone. The following table illustrates the cumulative query time, for all five queries, when using each of the three options on each of the six tested table sizes:

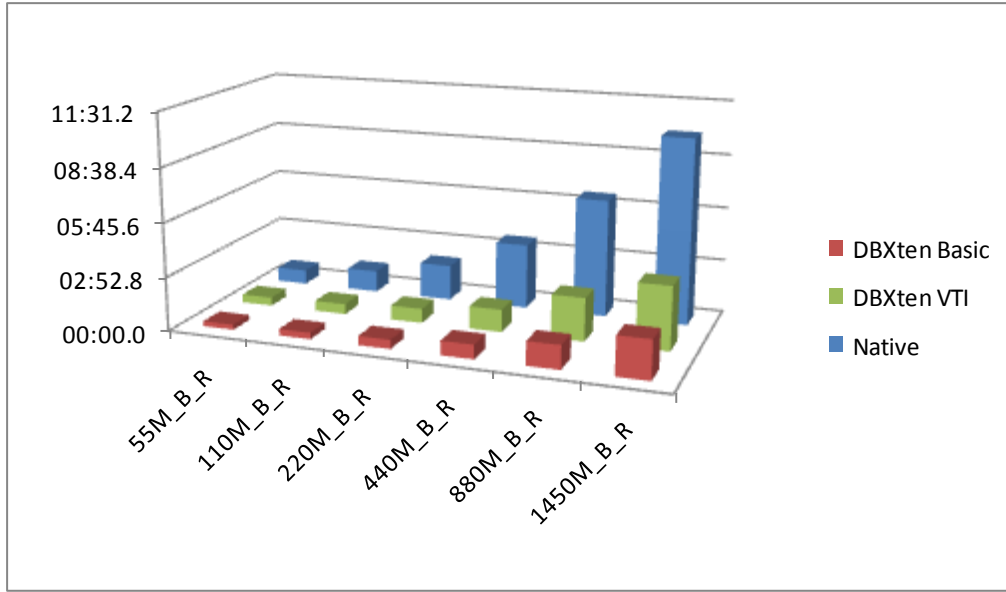


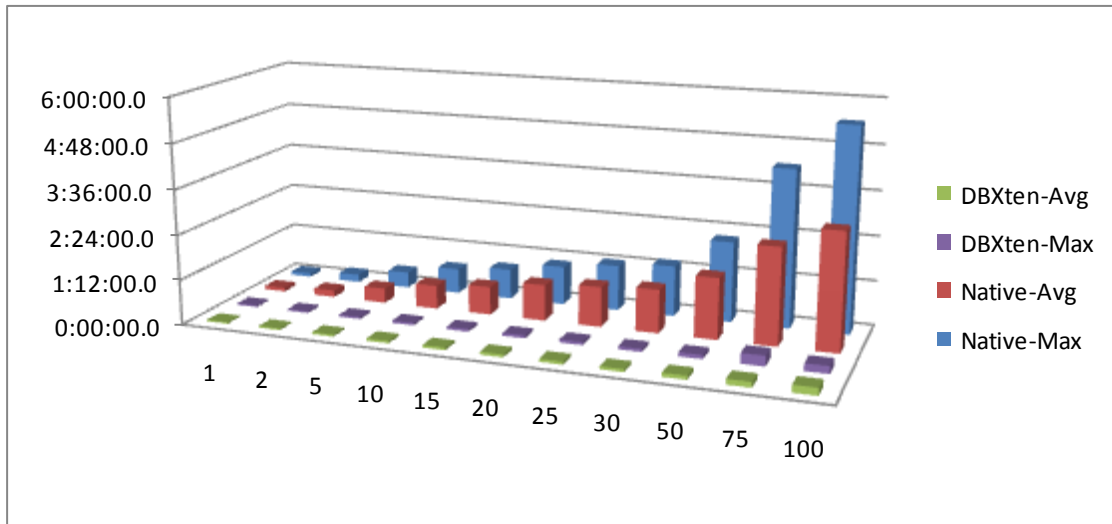
Figure 2: Cumulative Query Time as a Function of Table Size and Platform.

### Concurrent Query Testing

We conducted a final test to assess how well DBXten performs queries relative to native Informix under various load conditions. Specifically, each of Q1 through Q5 was run on the 1.447 billion row table, in sequence, by each of N (simulated) concurrent users. The maximum and average cumulative query times, taken across all N users, for all queries were noted. The values of N were 1, 2, 5, 10, 15, 20, 25, 30, 50, 75, and 100. This was done first for native Informix, and then for DBXten with the basic interface. Before each creation of the N simulated users, Informix was restarted and caches were flushed as described [earlier](#). The following table and graph present the results of this experiment:

N	Native-Avg	DBXten-Avg	Native-Max	DBXten-Max
1	0:05:48.3	0:02:19.2	0:05:48.3	0:02:19.2
2	0:10:49.2	0:02:33.1	0:12:30.7	0:02:35.6
5	0:23:54.7	0:03:08.1	0:25:47.3	0:03:22.9
10	0:38:03.8	0:03:42.2	0:40:28.8	0:03:44.8
15	0:45:26.7	0:04:05.6	0:48:47.1	0:04:06.7
20	0:58:11.2	0:04:24.7	1:02:32.1	0:04:33.0
25	1:04:24.3	0:04:41.8	1:13:17.3	0:04:55.3
30	1:09:46.9	0:04:30.9	1:21:54.3	0:04:47.0
50	1:38:09.6	0:06:11.7	2:09:51.3	0:06:26.5
75	2:35:12.9	0:08:39.2	4:13:06.0	0:16:33.1
100	3:07:59.4	0:11:47.7	5:26:50.0	0:12:03.1

Table 16: Cumulative Query Average and Maximum Times (h:mm:ss.f) as a Function of Concurrency Level (N) and Platform



**Figure 3: Cumulative Query Average and Maximum Times (h:mm:ss.f) as a Function of Concurrency Level (N) and Platform**

## Acknowledgements

IBM kindly provided us with access to the System x configuration at San Mateo and the services of Syed Zaidi, Joseph Baric, Larry Garibay, and Kent Stong. Andrew Coward helped us to acquire the OCCAM dataset, John Cartwright of NOAA provided feedback on a first draft of this report, and Mark Ashworth, an IBM Informix Datablade Architect, carefully reviewed the procedures used in conducting these tests. Barrodale Computing Services is grateful for this assistance, and for the interest demonstrated in this project.

## Appendix A – onconfig File

```
#####
# Licensed Material - Property Of IBM
#
# "Restricted Materials of IBM"
#
# IBM Informix Dynamic Server
# Copyright IBM Corporation 1996, 2009. All rights reserved.
#
# Title: onconfig.std
# Description: IBM Informix Dynamic Server Configuration Parameters
#
# Important: $INFORMIXDIR now resolves to the environment
# variable INFORMIXDIR. Replace the value of the INFORMIXDIR
# environment variable only if the path you want is not under
# $INFORMIXDIR.
#
# For additional information on the parameters:
# http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp
#####
```

```
#####
# Root Dbspace Configuration Parameters
#####
# ROOTNAME      - The root dbspace name to contain reserved pages and
#                internal tracking tables.
# ROOTPATH      - The path for the device containing the root dbspace
# ROOTOFFSET    - The offset, in KB, of the root dbspace into the
#                device. The offset is required for some raw devices.
# ROOTSIZE      - The size of the root dbspace, in KB. The value of
#                200000 allows for a default user space of about
#                100 MB and the default system space requirements.
# MIRROR        - Enable (1) or disable (0) mirroring
# MIRRORPATH    - The path for the device containing the mirrored
#                root dbspace
# MIRROROFFSET  - The offset, in KB, into the mirrored device
#
# Warning: Always verify ROOTPATH before performing
#          disk initialization (oninit -i or -iy) to
#          avoid disk corruption of another instance
#####
```

```
ROOTNAME rootdbs
ROOTPATH /home/informix/dev/sdd
ROOTOFFSET 4
ROOTSIZE 2000000
MIRROR 0
MIRRORPATH $INFORMIXDIR/tmp/demo_on.root_mirror
MIRROROFFSET 0
```

```
#####
# Physical Log Configuration Parameters
#####
# PHYSFILE      - The size, in KB, of the physical log on disk.
#                If RTO_SERVER_RESTART is enabled, the
#                suggested formula for the size of PHSYFILE
#                (up to about 1 GB) is:
#                PHYSFILE = Size of BUFFERS * 1.1
# PLOG_OVERFLOW_PATH - The directory for extra physical log files
#                if the physical log overflows during recovery
#                or long transaction rollback
# PHYSBUFF      - The size of the physical log buffer, in KB
#####
```

```
PHYSFILE 10240000
PLOG_OVERFLOW_PATH $INFORMIXDIR/tmp
PHYSBUFF 128
```

```
#####
# Logical Log Configuration Parameters
#####
# LOGFILES      - The number of logical log files
# LOGSIZE       - The size of each logical log, in KB
# DYNAMIC_LOGS  - The type of dynamic log allocation.
#                Acceptable values are:
#                2 Automatic. IDS adds a new logical log to the
#                root dbspace when necessary.
#                1 Manual. IDS notifies the DBA to add new logical
```

```

#             logs when necessary.
#             0 Disabled
# LOGBUFF     - The size of the logical log buffer, in KB
#####

LOGFILES      7
LOGSIZE 10000
DYNAMIC_LOGS 2
LOGBUFF 64

#####
# Long Transaction Configuration Parameters
#####
# If IDS cannot roll back a long transaction, the server hangs
# until more disk space is available.
#
# LTXHWM      - The percentage of the logical logs that can be
#             filled before a transaction is determined to be a
#             long transaction and is rolled back
# LTXEHWM     - The percentage of the logical logs that have been
#             filled before the server suspends all other
#             transactions so that the long transaction being
#             rolled back has exclusive use of the logs
#
# When dynamic logging is on, you can set higher values for
# LTXHWM and LTXEHWM because the server can add new logical logs
# during long transaction rollback. Set lower values to limit the
# number of new logical logs added.
#
# If dynamic logging is off, set LTXHWM and LTXEHWM to
# lower values, such as 50 and 60 or lower, to prevent long
# transaction rollback from hanging the server due to lack of
# logical log space.
#
# When using Enterprise Replication, set LTXEHWM to at least 30%
# higher than LTXHWM to minimize log overruns.
#####

LTXHWM 70
LTXEHWM 80

#####
# Server Message File Configuration Parameters
#####
# MSGPATH     - The path of the IDS message log file
# CONSOLE     - The path of the IDS console message file
#####

MSGPATH /home/informix/online.log
CONSOLE /home/informix/console

#####
# Tblspace Configuration Parameters
#####
# TBLTBLFIRST - The first extent size, in KB, for the tblspace
#             tblspace. Must be in multiples of the page size.
# TBLTBLNEXT  - The next extent size, in KB, for the tblspace

```

```

#          tblspace. Must be in multiples of the page size.
# The default setting for both is 0, which allows IDS to manage
# extent sizes automatically.
#
# TBLSPACE_STATS - Enables (1) or disables (0) IDS to maintain
#          tblspace statistics
#####

TBLTBLFIRST 0
TBLTBLNEXT 0
TBLSPACE_STATS 1

#####
# Temporary dbspace and sbspace Configuration Parameters
#####
# DBSPACETEMP - The list of dbspaces used to store temporary
#          tables and other objects. Specify a colon
#          separated list of dbspaces that exist when the
#          server is started. If no dbspaces are specified,
#          or if all specified dbspaces are not valid,
#          temporary files are created in the /tmp directory
#          instead.
# SBSPACETEMP - The list of sbspaces used to store temporary
#          tables for smart large objects. If no sbspace
#          is specified, temporary files are created in
#          a standard sbspace.
#####

DBSPACETEMP templdb
SBSPACETEMP templsb

#####
# Dbspace and sbspace Configuration Parameters
#####
# SBSPACENAME - The default sbspace name where smart large objects
#          are stored if no sbspace is specified during
#          smart large object creation. Some DataBlade
#          modules store smart large objects in this
#          location.
# SYSSBSPACENAME - The default sbspace for system statistics
#          collection. Otherwise, IDS stores statistics
#          in the sysdistrib system catalog table.
# ONDBSPACEDOWN - Specifies how IDS behaves when it encounters a
#          dbspace that is offline. Acceptable values
#          are:
#          0 Continue
#          1 Stop
#          2 Wait for DBA action
#####

SBSPACENAME templsb
SYSSBSPACENAME templsbifmx

ONDBSPACEDOWN 2

#####

```

```

# System Configuration Parameters
#####
# SERVERNUM      - The unique ID for the IDS instance. Acceptable
#                  values are 0 through 255, inclusive.
# DBSERVERNAME   - The name of the default database server
# DBSERVERALIASES - The list of up to 32 alternative dbservernames,
#                  separated by commas
#####

SERVERNUM 0
DBSERVERNAME barrodale
DBSERVERALIASES

#####
# Network Configuration Parameters
#####
# NETTYPE        - The configuration of poll threads
#                  for a specific protocol. The
#                  format is:
#                  NETTYPE <protocol>,<# poll threads>
#                  ,<number of connections/thread>
#                  ,(NET|CPU)
#                  You can include multiple NETTYPE
#                  entries for multiple protocols.
# LISTEN_TIMEOUT - The number of seconds that IDS
#                  waits for a connection
# MAX_INCOMPLETE_CONNECTIONS - The maximum number of incomplete
#                  connections before IDS logs a Denial
#                  of Service (DoS) error
# FASTPOLL       - Enables (1) or disables (0) fast
#                  polling of your network, if your
#                  operating system supports it.
#####

#NETTYPE ipcshm,1,50,CPU
LISTEN_TIMEOUT 60
MAX_INCOMPLETE_CONNECTIONS 1024
FASTPOLL 1

#####
# CPU-Related Configuration Parameters
#####
# MULTIPROCESSOR - Specifies whether the computer has multiple
#                  CPUs. Acceptable values are: 0 (single
#                  processor), 1 (multiple processors or
#                  multi-core chips)
# VPCLASS cpu     - Configures the CPU VPs. The format is:
#                  VPCLASS cpu,num=<#>[,max=<#>][,aff=<#>]
#                  [,noage]
# VP_MEMORY_CACHE_KB - Specifies the amount of private memory
#                  blocks of your CPU VP, in KB, that the
#                  database server can access.
#                  Acceptable values are:
#                  0 (disable)
#                  800 through 40% of the value of SHMTOTAL
# SINGLE_CPU_VP  - Optimizes performance if IDS runs with
#                  only one CPU VP. Acceptable values are:

```

```

#                               0 multiple CPU VPs
#                               Any nonzero value (optimize for one CPU VP)
#####

MULTIPROCESSOR 1 # was 0
VPCLASS cpu,num=10,noage # was num=1
VP_MEMORY_CACHE_KB 0
SINGLE_CPU_VP 0

#####
# AIO and Cleaner-Related Configuration Parameters
#####
# VPCLASS aio - Configures the AIO VPs. The format is:
#               VPCLASS aio,num=<#>[,max=<#>][,aff=<#>][,noage]
# CLEANERS    - The number of page cleaner threads
# AUTO_AIOVPS - Enables (1) or disables (0) automatic management
#               of AIO VPs
# DIRECT_IO   - Specifies whether direct I/O is used for cooked
#               files used for dbspace chunks.
#               Acceptable values are:
#               0 Disable
#               1 Enable direct I/O
#               2 Enable concurrent I/O
#####

#VPCLASS aio,num=1
CLEANERS 16 # was 8
AUTO_AIOVPS 1
DIRECT_IO 0

#####
# Lock-Related Configuration Parameters
#####
# LOCKS          - The initial number of locks when IDS starts.
#                 Dynamic locking can add extra locks if needed.
# DEF_TABLE_LOCKMODE - The default table lock mode for new tables.
#                 Acceptable values are ROW and PAGE (default).
#####

LOCKS 20000
DEF_TABLE_LOCKMODE page

#####
# Shared Memory Configuration Parameters
#####
# RESIDENT      - Controls whether shared memory is resident.
#                 Acceptable values are:
#                 0 off (default)
#                 1 lock the resident segment only
#                 n lock the resident segment and the next n-1
#                   virtual segments, where n < 100
#                 -1 lock all resident and virtual segments
# SHMBASE      - The shared memory base address; do not change
# SHMVRTSIZE   - The initial size, in KB, of the virtual
#                 segment of shared memory
# SHMADD       - The size, in KB, of additional virtual shared
#                 memory segments
#

```

```

# EXTSHMADD      - The size, in KB, of each extension shared
#                memory segment
# SHMTOTAL      - The maximum amount of shared memory for IDS,
#                in KB. A 0 indicates no specific limit.
# SHMVIRT_ALLOCSEG - Controls when IDS adds a memory segment and
#                the alarm level if the memory segment cannot
#                be added.
#                For the first field, acceptable values are:
#                - 0 Disabled
#                - A decimal number indicating the percentage
#                of memory used before a segment is added
#                - The number of KB remaining when a segment
#                is added
#                For the second field, specify an alarm level
#                from 1 (non-event) to 5 (fatal error).
# SHMNOACCESS   - A list of up to 10 memory address ranges
#                that IDS cannot use to attach shared memory.
#                Each address range is the start and end memory
#                address in hex format, separated by a hyphen.
#                Use a comma to separate each range in the list.
#####

```

```

RESIDENT 0
SHMBASE 0x44000000L
SHMVIRT_SIZE 32656
SHMADD 8192
EXTSHMADD 8192
SHMTOTAL 0
SHMVIRT_ALLOCSEG 0,3
SHMNOACCESS

```

```

#####
# Checkpoint and System Block Configuration Parameters
#####
# CKPINTVL      - Specifies how often, in seconds, IDS checks
#                if a checkpoint is needed. 0 indicates that
#                IDS does not check for checkpoints. Ignored
#                if RTO_SERVER_RESTART is set.
# AUTO_CKPTS    - Enables (1) or disables (0) monitoring of
#                critical resource to trigger checkpoints
#                more frequently if there is a chance that
#                transaction blocking might occur.
# RTO_SERVER_RESTART - Specifies, in seconds, the Recovery Time
#                Objective for IDS restart after a server
#                failure. Acceptable values are 0 (off) and
#                any number from 60-1800, inclusive.
# BLOCKTIMEOUT  - Specifies the amount of time, in seconds,
#                for a system block.
#####

```

```

CKPTINTVL 300
AUTO_CKPTS 1
RTO_SERVER_RESTART 0
BLOCKTIMEOUT 3600

```

```

#####
# Conversion Guard Related Configuration Parameters

```

```
#####
# CONVERSION_GUARD - To turn on conversion guard feature.
#                   - 0 - Off,
#                   - 1 - On, Abort conversion on Conversion Guard
error,
#                   - 2 - On, Continue conversion; ignore Conversion
#                   Guard error
#
# RESTORE_POINT_DIR - The directory, which stores the Conversion Guard
#                   - feature generated files.
#####
```

```
CONVERSION_GUARD 1
RESTORE_POINT_DIR $INFORMIXDIR/tmp
```

```
#####
# Transaction-Related Configuration Parameters
#####
# TXTIMEOUT        - The distributed transaction timeout, in seconds
# DEADLOCK_TIMEOUT - The maximum time, in seconds, to wait for a
#                   lock in a distributed transaction.
# HETERO_COMMIT    - Enables (1) or disables (0) heterogeneous
#                   commits for a distributed transaction
#                   involving an EGM gateway.
#####
```

```
TXTIMEOUT 300
DEADLOCK_TIMEOUT 60
HETERO_COMMIT 0
```

```
#####
# ontape Tape Device Configuration Parameters
#####
# TAPEDEV          - The tape device path for backups. To use standard
#                   I/O instead of a device, set to STDIO.
# TAPEBLK          - The tape block size, in KB, for backups
# TAPESIZE         - The maximum amount of data to put on one backup
#                   tape. Acceptable values are 0 (unlimited) or any
#                   positive integral multiple of TAPEBLK.
#####
```

```
TAPEDEV /dev/null
TAPEBLK 32
TAPESIZE 0
```

```
#####
# ontape Logical Log Tape Device Configuration Parameters
#####
# LTAPEDEV         - The tape device path for logical logs
# LTAPEBLK         - The tape block size, in KB, for backing up logical
#                   logs
# LTAPESIZE        - The maximum amount of data to put on one logical
#                   log tape. Acceptable values are 0 (unlimited) or any
#                   positive integral multiple of LTAPEBLK.
#####
```

```
LTAPEDEV /dev/null
LTAPEBLK 32
LTAPESIZE 0
```

```
#####
# Backup and Restore Configuration Parameters
#####
# BAR_ACT_LOG - The ON-Bar activity log file location.
# Do not use the /tmp directory. Use a
# directory with restricted permissions.
# BAR_DEBUG_LOG - The ON-Bar debug log file location.
# Do not use the /tmp directory. Use a
# directory with restricted permissions.
# BAR_DEBUG - The debug level for ON-Bar. Acceptable
# values are 0 (off) through 9 (high).
# BAR_MAX_BACKUP - The number of backup threads used in a
# backup. Acceptable values are 0 (unlimited)
# or any positive integer.
# BAR_RETRY - Specifies the number of time to retry a
# backup or restore operation before reporting
# a failure
# BAR_NB_XPORT_COUNT - Specifies the number of data buffers that
# each onbar_d process uses to communicate
# with the database server
# BAR_XFER_BUF_SIZE - The size, in pages, of each data buffer.
# Acceptable values are 1 through 15 for
# 4 KB pages and 1 through 31 for 2 KB pages.
# RESTARTABLE_RESTORE - Enables ON-Bar to continue a backup after a
# failure. Acceptable values are OFF or ON.
# BAR_PROGRESS_FREQ - Specifies, in minutes, how often progress
# messages are placed in the ON-Bar activity
# log. Acceptable values are: 0 (record only
# completion messages) or 5 and above.
# BAR_BSAIB_PATH - The shared library for ON-Bar and the
# storage manager. The default value is
# $INFORMIXDIR/lib/ibsad001 (with a
# platform-specific file extension).
# BACKUP_FILTER - Specifies the pathname of a filter program
# to transform data during a backup, plus any
# program options
# RESTORE_FILTER - Specifies the pathname of a filter program
# to transform data during a restore, plus any
# program options
# BAR_PERFORMANCE - Specifies the type of performance statistics
# to report to the ON-Bar activity log for backup
# and restore operations.
# Acceptable values are:
# 0 = Turn off performance monitoring (Default)
# 1 = Display the time spent transferring data
# between the IDS instance and the storage
# manager
# 2 = Display timestamps in microseconds
# 3 = Display both timestamps and transfer
# statistics
#####
BAR_ACT_LOG $INFORMIXDIR/tmp/bar_act.log
```

```
BAR_DEBUG_LOG $INFORMIXDIR/tmp/bar_dbug.log
BAR_DEBUG 0
BAR_MAX_BACKUP 0
BAR_RETRY 1
BAR_NB_XPORT_COUNT 20
BAR_XFER_BUF_SIZE 31
RESTARTABLE_RESTORE ON
BAR_PROGRESS_FREQ 0
BAR_BSALIB_PATH
BACKUP_FILTER
RESTORE_FILTER
BAR_PERFORMANCE 0
```

```
#####
# Informix Storage Manager (ISM) Configuration Parameters
#####
# ISM_DATA_POOL - Specifies the name for the ISM data pool
# ISM_LOG_POOL - Specifies the name for the ISM log pool
#####
```

```
ISM_DATA_POOL ISMData
ISM_LOG_POOL ISMLogs
```

```
#####
# Data Dictionary Cache Configuration Parameters
#####
# DD_HASHSIZE - The number of data dictionary pools. Set to any
#               positive integer; a prime number is recommended.
# DD_HASHMAX - The number of entries per pool.
#               Set to any positive integer.
#####
```

```
DD_HASHSIZE 31
DD_HASHMAX 10
```

```
#####
# Data Distribution Configuration Parameters
#####
# DS_HASHSIZE - The number of data Distribution pools.
#               Set to any positive integer; a prime number is
#               recommended.
# DS_POOLSIZ - The maximum number of entries in the data
#               distribution cache. Set to any positive integer.
#####
```

```
DS_HASHSIZE 31
DS_POOLSIZ 127
```

```
#####
# User Defined Routine (UDR) Cache Configuration Parameters
#####
# PC_HASHSIZE - The number of UDR pools. Set to any
#               positive integer; a prime number is recommended.
# PC_POOLSIZ - The maximum number of entries in the
#               UDR cache. Set to any positive integer.
#####
```

```
PC_HASHSIZE 31
PC_POOLSIZE 127
```

```
#####
# SQL Statement Cache Configuration Parameters
#####
# STMT_CACHE          - Controls SQL statement caching. Acceptable
#                      values are:
#                      0 Disabled
#                      1 Enabled at the session level
#                      2 All statements are cached
# STMT_CACHE_HITS     - The number of times an SQL statement must be
#                      executed before becoming fully cached.
#                      0 indicates that all statements are
#                      fully cached the first time.
# STMT_CACHE_SIZE     - The size, in KB, of the SQL statement cache
# STMT_CACHE_NOLIMIT  - Controls additional memory consumption.
#                      Acceptable values are:
#                      0 Limit memory to STMT_CACHE_SIZE
#                      1 Obtain as much memory, temporarily, as needed
# STMT_CACHE_NUMPOOL - The number of pools for the SQL statement
#                      cache. Acceptable value is a positive
#                      integer between 1 and 256, inclusive.
#####
```

```
STMT_CACHE 0
STMT_CACHE_HITS 0
STMT_CACHE_SIZE 512
STMT_CACHE_NOLIMIT 0
STMT_CACHE_NUMPOOL 1
```

```
#####
# Operating System Session-Related Configuration Parameters
#####
# USEOSTIME           - The precision of SQL statement timing.
#                      Accepted values are 0 (precision to seconds)
#                      and 1 (precision to subseconds). Subsecond
#                      precision can degrade performance.
# STACKSIZE          - The size, in KB, for a session stack
# ALLOW_NEWLINE       - Controls whether embedded new line characters
#                      in string literals are allowed in SQL
#                      statements. Acceptable values are 1 (allowed)
#                      and any number other than 1 (not allowed).
# USELASTCOMMITTED   - Controls the committed read isolation level.
#                      Acceptable values are:
#                      - NONE Waits on a lock
#                      - DIRTY READ Uses the last committed value in
#                      place of a dirty read
#                      - COMMITTED READ Uses the last committed value
#                      in place of a committed read
#                      - ALL Uses the last committed value in place
#                      of all isolation levels that support the last
#                      committed option
#####
```

```
USEOSTIME 0
STACKSIZE 64
```

```
ALLOW_NEWLINE 0
USELASTCOMMITTED NONE
```

```
#####
# Index Related Configuration Parameters
#####
# FILLFACTOR          - The percentage of index page fullness
# MAX_FILL_DATA_PAGES - Enables (1) or disables (0) filling data
#                    - pages that have variable length rows as
#                    - full as possible
# BTSCANNER           - Specifies the configuration settings for all
#                    - btscanner threads. The format is:
#                    - BTSCANNER num=<#>,threshold=<#>,rangesize=<#>,
#                    - alice=(0-12),compression=[low|med|high|default]
# ONLIDX_MAXMEM       - The amount of memory, in KB, allocated for
#                    - the pre-image pool and updator log pool for
#                    - each partition.
#####
```

```
FILLFACTOR 90
MAX_FILL_DATA_PAGES 0
BTSCANNER num=1,threshold=5000,rangesize=-1,alice=6,compression=default
ONLIDX_MAXMEM 5120
```

```
#####
# Parallel Database Query (PDQ) Configuration Parameters
#####
# MAX_PDQPRIORITY     - The maximum amount of resources, as a
#                    - percentage, that PDQ can allocate to any
#                    - one decision support query
# DS_MAX_QUERIES      - The maximum number of concurrent decision
#                    - support queries
# DS_TOTAL_MEMORY     - The maximum amount, in KB, of decision
#                    - support query memory
# DS_MAX_SCANS        - The maximum number of concurrent decision
#                    - support scans
# DS_NONPDQ_QUERY_MEM - The amount of non-PDQ query memory, in KB.
#                    - Acceptable values are 128 to 25% of
#                    - DS_TOTAL_MEMORY.
# DATASKIP            - Specifies whether to skip dbspaces when
#                    - processing a query. Acceptable values are:
#                    - ALL Skip all unavailable fragments
#                    - ON <dbspace1> <dbspace2>... Skip listed
#                    - dbspaces
#                    - OFF Do not skip dbspaces (default)
#####
```

```
MAX_PDQPRIORITY 100
DS_MAX_QUERIES
DS_TOTAL_MEMORY
DS_MAX_SCANS 1048576
DS_NONPDQ_QUERY_MEM 128
DATASKIP
```

```
#####
# Optimizer Configuration Parameters
#####
```

```

# OPTCOMPIND      - Controls how the optimizer determines the best
#                  query path. Acceptable values are:
#                  0 Nested loop joins are preferred
#                  1 If isolation level is repeatable read,
#                  works the same as 0, otherwise works same as 2
#                  2 Optimizer decisions are based on cost only
# DIRECTIVES      - Specifies whether optimizer directives are
#                  enabled (1) or disabled (0). Default is 1.
# EXT_DIRECTIVES  - Controls the use of external SQL directives.
#                  Acceptable values are:
#                  0 Disabled
#                  1 Enabled if the IFX_EXTDIRECTIVES environment
#                  variable is enabled
#                  2 Enabled even if the IFX_EXTDIRECTIVES
#                  environment is not set
# OPT_GOAL        - Controls how the optimizer should optimize for
#                  fastest retrieval. Acceptable values are:
#                  -1 All rows in a query
#                  0 The first rows in a query
# IFX_FOLDVIEW    - Enables (1) or disables (0) folding views that
#                  have multiple tables or a UNION ALL clause.
#                  Disabled by default.
# AUTO_REPREPARE - Enables (1) or disables (0) automatically
#                  re-optimizing stored procedures and re-preparing
#                  prepared statements when tables that are referenced
#                  by them change. Minimizes the occurrence of the
#                  -710 error.
#####

```

```

OPTCOMPIND 2
DIRECTIVES 1
EXT_DIRECTIVES 0
OPT_GOAL -1
IFX_FOLDVIEW 0
AUTO_REPREPARE 1

```

```

#####
# Scan Configuration Parameters
#####
#RA_PAGES      - The number of pages, as a positive integer, to
#               attempt to read ahead
#RA_THRESHOLD  - The number of pages, as a positive integer, left
#               before the next read-ahead group
#BATCHEDREAD_TABLE - Turn on/off xps api for table scans
#####

```

```

RA_PAGES      64
RA_THRESHOLD  16
BATCHEDREAD_TABLE 0

```

```

#####
# SQL Tracing and EXPLAIN Plan Configuration Parameters
#####
# EXPLAIN_STAT - Enables (1) or disables (0) including the Query
#               Statistics section in the EXPLAIN output file
# SQLTRACE     - Configures SQL tracing. The format is:
#               SQLTRACE level=(low|med|high),ntraces=<#>,size=<#>,

```

```

#                               mode=(global|user)
#####

EXPLAIN_STAT 1
#SQLTRACE level=low,ntraces=1000,size=2,mode=global

#####
# Security Configuration Parameters
#####
# DBCREATE_PERMISSION          - Specifies the users who can create
#                               databases (by default, any user can).
#                               Add a DBCREATE_PERMISSION entry
#                               for each user who needs database
#                               creation privileges. Ensure user
#                               informix is authorized when you
#                               first initialize IDS.
# DB_LIBRARY_PATH              - Specifies the locations, separated
#                               by commas, from which IDS can use
#                               UDR or UDT shared libraries. If set,
#                               make sure that all directories
containing
#                               the blade modules are listed, to
#                               ensure all DataBlade modules will
#                               work.
# IFX_EXTEND_ROLE              - Controls whether administrators
#                               can use the EXTEND role to specify
#                               which users can register external
#                               routines. Acceptable values are:
#                               0 Any user can register external
#                               routines
#                               1 Only users granted the ability
#                               to register external routines
#                               can do so (Default)
# SECURITY_LOCALCONNECTION      - Specifies whether IDS performs
#                               security checking for local
#                               connections. Acceptable values are:
#                               0 Off
#                               1 Validate ID
#                               2 Validate ID and port
# UNSECURE_ONSTAT              - Controls whether non-DBSA users are
#                               allowed to run all onstat commands.
#                               Acceptable values are:
#                               1 Enabled
#                               0 Disabled (Default)
# ADMIN_USER_MODE_WITH_DBSA     - Controls who can connect to IDS
#                               in administration mode. Acceptable
#                               values are:
#                               1 DBSAs, users specified by
#                               ADMIN_MODE_USERS, and the user
#                               informix
#                               0 Only the user informix (Default)
# ADMIN_MODE_USERS              - Specifies the user names, separated by
#                               commas, who can connect to IDS in
#                               administration mode, in addition to
#                               the user informix
# SSL_KEYSTORE_LABEL            - The label, up to 512 characters, of
#                               the IDS certificate used in Secure

```

```

#                               Sockets Layer (SSL) protocol
#                               communications.
#####

#DBCREATE_PERMISSION informix
#DB_LIBRARY_PATH
IFX_EXTEND_ROLE 1
SECURITY_LOCALCONNECTION
UNSECURE_ONSTAT
ADMIN_USER_MODE_WITH_DBSA
ADMIN_MODE_USERS
SSL_KEYSTORE_LABEL
#####
# LBAC Configuration Parameters
#####
# PLCY_POOLSIZE - The maximum number of entries in each hash
#                 bucket of the LBAC security information cache
# PLCY_HASHSIZE - The number of hash buckets in the LBAC security
#                 information cache
# USRC_POOLSIZE - The maximum number of entries in each hash
#                 bucket of the LBAC credential memory cache
# USRC_HASHSIZE - The number of hash buckets in the LBAC credential
#                 memory cache
#####

PLCY_POOLSIZE 127
PLCY_HASHSIZE 31
USRC_POOLSIZE 127
USRC_HASHSIZE 31

#####
# Optical Configuration Parameters
#####
# STAGEBLOB      - The name of the optical blobspace. Must be set to
#                 use the optical-storage subsystem.
# OPCACHEMAX    - Maximum optical cache size, in KB
#####

STAGEBLOB
OPCACHEMAX 0

#####
# High Availability and Enterprise Replication Security
# Configuration Parameters
#####
# ENCRYPT_HDR    - Enables (1) or disables (0) encryption for HDR.
# ENCRYPT_SMX    - Controls the level of encryption for RSS and
#                 SDS servers. Acceptable values are:
#                 0 Do not encrypt (Default)
#                 1 Encrypt if possible
#                 2 Always encrypt
# ENCRYPT_CDR    - Controls the level of encryption for ER.
#                 Acceptable values are:
#                 0 Do not encrypt (Default)
#                 1 Encrypt if possible
#                 2 Always encrypt

```

```
# ENCRYPT_CIPHERS - A list of encryption ciphers and modes,
#                 separated by commas. Default is all.
# ENCRYPT_MAC      - Controls the level of message authentication
#                 code (MAC). Acceptable values are off, high,
#                 medium, and low. List multiple values separated
#                 by commas; the highest common level between
#                 servers is used.
# ENCRYPT_MACFILE  - The paths of the MAC key files, separated
#                 by commas. Use the builtin keyword to specify
#                 the built-in key. Default is builtin.
# ENCRYPT_SWITCH   - Defines the frequencies, in minutes, at which
#                 ciphers and keys are renegotiated. Format is:
#                 <cipher_switch_time>,<key_switch_time>
#                 Default is 60,60.
#####
```

```
ENCRYPT_HDR
ENCRYPT_SMX
ENCRYPT_CDR 0
ENCRYPT_CIPHERS
ENCRYPT_MAC
ENCRYPT_MACFILE
ENCRYPT_SWITCH
```

```
#####
# Enterprise Replication (ER) Configuration Parameters
#####
# CDR_EVALTHREADS - The number of evaluator threads per
#                 CPU VP and the number of additional
#                 threads, separated by a comma.
#                 Acceptable values are: a non-zero value
#                 followed by a non-negative value
# CDR_DSLOCKWAIT  - The number of seconds the Datasync
#                 waits for database locks.
# CDR_QUEUEMEM    - The maximum amount of memory, in KB,
#                 for the send and receive queues.
# CDR_NIFCOMPRESS - Controls the network interface
#                 compression level.
#                 Acceptable values are:
#                 -1 Never
#                 0 None
#                 1-9 Compression level
# CDR_SERIAL      - Specifies the incremental size and
#                 the starting value of replicated
#                 serial columns. The format is:
#                 <delta>,<offset>
# CDR_DBSPACE     - The dbspace name for the syscdr
#                 database.
# CDR_QHDR_DBSPACE - The name of the transaction record
#                 dbspace. Default is the root dbspace.
# CDR_QDATA_SBSpace - The names of sbspaces for spooled
#                 transaction data, separated by commas.
# CDR_MAX_DYNAMIC_LOGS - The maximum number of dynamic log
#                 requests that ER can make within one
#                 server session. Acceptable values are:
#                 -1 (unlimited), 0 (disabled),
```

```
#
#           1 through n (limit to n requests)
# CDR_SUPPRESS_ATSRISWARN - The Datasync error and warning code
#
#           numbers to be suppressed in ATS and RIS
#
#           files. Acceptable values are: numbers
#
#           or ranges of numbers separated by commas.
#
#           Separate numbers in a range by a hyphen.
#####
```

```
CDR_EVALTHREADS 1,2
CDR_DSLOCKWAIT 5
CDR_QUEUEMEM 4096
CDR_NIFCOMPRESS 0
CDR_SERIAL 0
CDR_DBSPACE
CDR_QHDR_DBSPACE
CDR_QDATA_SBSPACE
CDR_MAX_DYNAMIC_LOGS 0
CDR_SUPPRESS_ATSRISWARN
```

```
#####
# High Availability Cluster (HDR, SDS, and RSS)
# Configuration Parameters
#####
# DRAUTO           - Controls automatic failover of primary
#                   servers. Valid for HDR, SDS, and RSS.
#                   Acceptable values are:
#                   0 Manual
#                   1 Retain server type
#                   2 Reverse server type
#                   3 Connection Manager Arbitrator controls
#                   server type
# DRINTERVAL      - The maximum interval, in seconds, between HDR
#                   buffer flushes. Valid for HDR only.
# DRTIMEOUT       - The time, in seconds, before a network
#                   timeout occurs. Valid for HDR only.
# DRLOSTFOUND     - The path of the HDR lost-and-found file.
#                   Valid of HDR only.
# DRIDXAUTO       - Enables (1) or disables (0) automatic index
#                   repair for an HDR pair. Default is 0.
# HA_ALIAS        - The server alias for a high-availability
#                   cluster. Must be the same as a value of
#                   DBSERVERNAME or DBSERVERALIASES that uses a
#                   network-based connection type. Valid for HDR,
#                   SDS, and RSS.
# LOG_INDEX_BUILDS - Enable (1) or disable (0) index page logging.
#                   Required for RSS. Optional for HDR and SDS.
# SDS_ENABLE      - Enables (1) or disables (0) an SDS server.
#                   Set this value on an SDS server after setting
#                   up the primary. Valid for SDS only.
# SDS_TIMEOUT     - The time, in seconds, that the primary waits
#                   for an acknowledgement from an SDS server
#                   while performing page flushing before marking
#                   the SDS server as down. Valid for SDS only.
# SDS_TEMPDBS    - The temporary dbspace used by an SDS server.
#                   The format is:
```

```

#           <dbspace_name>,<path>,<pagesize in KB>,<offset in
KB>,
#           <size in KB>
#           You can include up to 16 entries of SDS_TEMPDBS
to
#           specify additional dbspaces. Valid for SDS.
# SDS_PAGING           - The paths of two buffer paging files,
#                       Separated by a comma. Valid for SDS only.
# UPDATABLE_SECONDARY - Controls whether secondary servers can accept
#                       update, insert, and delete operations from
clients.
#                       If enabled, specifies the number of connection
#                       threads between the secondary and primary servers
#                       for transmitting updates from the secondary.
#                       Acceptable values are:
#                       0 Secondary server is read-only (default)
#                       1 through twice the number of CPU VPs,
threads
#                       for performing updates from the secondary.
#                       Valid for HDR, SDS, and RSS.
# FAILOVER_CALLBACK - Specifies the path and program name called when a
#                       secondary server transitions to a standard or
#                       primary server. Valid for HDR, SDS, and RSS.
# TEMPTAB_NOLOG       - Controls the default logging mode for temporary
#                       tables that are explicitly created with the
#                       CREATE TEMP TABLE or SELECT INTO TEMP statements.
#                       Secondary servers must not have logged temporary
#                       tables. Acceptable values are:
#                       0 Create temporary tables with logging enabled by
#                       default.
#                       1 Create temporary tables without logging.
#                       Required to be set to 1 on HDR, RSS, and SDS
#                       secondary servers.
# DELAY_APPLY         - Specifies a delay factor for RSS
#                       secondary nodes. The format is ###[DHMS] where
#                       D stands for days
#                       H stands for hours
#                       M stands for minutes
#                       S stands for seconds (default)
# STOP_APPLY          - Halts the apply on an RSS node
#                       1 halts the apply
#                       0 resumes the apply (default)
#                       YYYY:MM:DD:hh:mm:ss - time at which to stop
# LOG_STAGING_DIR     - Specifies a directory in which to stage log files
#####

DRAUTO           0
DRINTERVAL       30
DRTIMEOUT        30
HA_ALIAS
DRLOSTFOUND      $INFORMIXDIR/etc/dr.lostfound
DRIDXAUTO        0
LOG_INDEX_BUILDS
SDS_ENABLE
SDS_TIMEOUT      20
SDS_TEMPDBS
SDS_PAGING

```

```

UPDATABLE_SECONDARY      0
FAILOVER_CALLBACK
TEMPTAB_NOLOG           0
DELAY_APPLY             0
STOP_APPLY              0
LOG_STAGING_DIR

```

```

#####
# Logical Recovery Parameters
#####
# ON_RECVRY_THREADS - The number of logical recovery threads that
#                   run in parallel during a warm restore.
# OFF_RECVRY_THREADS - The number of logical recovery threads used
#                   in a cold restore. Also, the number of
#                   threads used during fast recovery.
#####

```

```

ON_RECVRY_THREADS 1
OFF_RECVRY_THREADS 10

```

```

#####
# Diagnostic Dump Configuration Parameters
#####
# DUMPDIR - The location Assertion Failure (AF) diagnostic
#         files
# DUMPSHMEM - Controls shared memory dumps. Acceptable values
#         are:
#         0 Disabled
#         1 Dump all shared memory
#         2 Exclude the buffer pool from the dump
# DUMPGCORE - Enables (1) or disables (0) whether IDS dumps a
#         core using gcore
# DUMPCORE - Enables (1) or disables (0) whether IDS dumps a
#         core after an AF
# DUMPCNT - The maximum number of shared memory dumps or
#         core files for a single session
#####

```

```

DUMPDIR $INFORMIXDIR/tmp
DUMPSHMEM 1
DUMPGCORE 0
DUMPCORE 0
DUMPCNT 1

```

```

#####
# Alarm Program Configuration Parameters
#####
# ALARMPROGRAM - Specifies the alarm program to display event
#               alarms. To enable automatic logical log backup,
#               edit alarmprogram.sh and set BACKUPLLOGS=Y.
# ALRM_ALL_EVENTS - Controls whether the alarm program runs for
#                 every event. Acceptable values are:
#                 0 Logs only noteworthy events
#                 1 Logs all events
# STORAGE_FULL_ALARM - <time interval in seconds>,<alarm severity>
#                   specifies in what interval:
#

```

```
#           - a message will be printed to the online.log
file
#           - an alarm will be raised
#           when
#           - a dbspace becomes full
#             (ISAM error -131)
#           - a partition runs out of pages or extents
#             (ISAM error -136)
#           time interval = 0 : OFF
#           severity = 0 : no alarm, only message
# SYSALARMPROGRAM - Specifies the system alarm program triggered
#                 when an AF occurs
#####
```

```
ALARMPROGRAM $INFORMIXDIR/etc/alarmprogram.sh
ALRM_ALL_EVENTS 0
STORAGE_FULL_ALARM 600,3
SYSALARMPROGRAM $INFORMIXDIR/etc/evidence.sh
```

```
#####
# RAS Configuration Parameters
#####
# RAS_PLOG_SPEED - Technical Support diagnostic parameter.
#                 Do not change; automatically updated.
# RAS_LLOG_SPEED - Technical Support diagnostic parameter.
#                 Do not change; automatically updated.
#####
```

```
RAS_PLOG_SPEED 25000
RAS_LLOG_SPEED 0
```

```
#####
# Character Processing Configuration Parameter
#####
# EILSEQ_COMPAT_MODE - Controls whether when processing characters,
#                     IDS checks if the characters are valid for
#                     the locale and returns error -202 if they are
#                     not. Acceptable values are:
#                     0 Return an error for characters that are not
#                     valid (Default)
#                     1 Allow characters that are not valid
#####
```

```
EILSEQ_COMPAT_MODE 0
```

```
#####
# Statistic Configuration Parameters
#####
# QSTATS - Enables (1) or disables (0) the collection of queue
#          statistics that can be viewed with onstat -g qst
# WSTATS - Enables (1) or disables (0) the collection of wait
#          statistics that can be viewed with onstat -g wst
#####
```

```
QSTATS 0
WSTATS 0
```

```
#####
```

```

# Java Configuration Parameters
#####
# VPCLASS jvp - Configures the Java VP. The format is:
#               VPCLASS jvp,num=<#>[,max=<#>][,aff=<#>][,noage]
# JVPJAVAHOME - The JRE root directory
# JVPHOME     - The Krakatoa installation directory
# JVPPROFILE  - The Java VP property file
# JVPLOGFILE  - The Java VP log file
#             This parameter is deprecated and is no longer required
# JDKVERSION - The version of JDK supported by this server
# JVPJAVALIB  - The location of the JRE libraries, relative
#             to JVPJAVAHOME
# JVPJAVAVM   - The JRE libraries to use for the Java VM
# JVPARGS     - Configures the Java VM. To display JNI calls,
#             use JVPARGS -verbose:jni. Separate options with
#             semicolons.
# JVPCLASSPATH - The Java classpath to use. Use krakatoa_g.jar
#             for debugging. Comment out the JVPCLASSPATH
#             entry you do not want to use.
#####

#VPCLASS          jvp,num=1
JVPJAVAHOME       $INFORMIXDIR/extend/krakatoa/jre
JVPHOME           $INFORMIXDIR/extend/krakatoa
JVPPROFILE        $INFORMIXDIR/extend/krakatoa/.jvpprops
JVPLOGFILE        $INFORMIXDIR/jvp.log
#JDKVERSION       1.5
JVPJAVALIB        /bin/j9vm
JVPJAVAVM         jvm
#JVPARGS          -verbose:jni
#JVPCLASSPATH
$INFORMIXDIR/extend/krakatoa/krakatoa_g.jar:$INFORMIXDIR/extend/krakatoa/
jdbc_g.jar
JVPCLASSPATH
$INFORMIXDIR/extend/krakatoa/krakatoa.jar:$INFORMIXDIR/extend/krakatoa/
jdbc.jar

#####
# Buffer pool and LRU Configuration Parameters
#####
# BUFFERPOOL     - Specifies the default values for buffers and LRU
#               queues in each buffer pool. Each page size used
#               by a dbspace has a buffer pool and needs a
#               BUFFERPOOL entry. The onconfig.std file contains
#               two initial entries: a default entry from which
#               to base new page size entries on, and an entry
#               for the operating system default page size.
#               When you add a dbspace with a different page size,
#               IDS adds a BUFFERPOOL entry to the onconfig file
#               with values that are the same as the default
#               BUFFERPOOL entry, except that the default
#               keyword is replaced by size=Nk, where N is the
#               new page size. With interval checkpoints, these
#               values can now be set higher than in previous
#               versions of IDS in an OLTP environment.
# AUTO_LRU_TUNING - Enables (1) or disables (0) automatic tuning of

```

```
#           LRU queues. When this parameter is enabled, IDS
#           increases the LRU flushing if it cannot find low
#           priority buffers for number page faults.
#####

BUFFERPOOL
    default,buffers=1000000,lrus=8,lru_min_dirty=50.000000,lru_max_di
rty=60.500000
BUFFERPOOL
    size=2K,buffers=1000000,lrus=8,lru_min_dirty=50.000000,lru_max_di
rty=60.000000
AUTO_LRU_TUNING 1
VPCLASS dbxten,num=1,noyield8
```

## Appendix B – Abridged NetCDF File Header (ncdump)

```
dimensions:
    TIMESTEP = 1 ;
    LONGITUDE_T = 447 ;
    LATITUDE_T = 467 ;
    LONGITUDE_U = 447 ;
    LATITUDE_U = 467 ;
    DEPTH = 66 ;
    DEPTH_EDGES = 67 ;

variables:
    int TIMESTEP(TIMESTEP) ;
        TIMESTEP:long_name = "Timestep" ;
    float LONGITUDE_T(LONGITUDE_T) ;
        LONGITUDE_T:long_name = "Longitude" ;
        LONGITUDE_T:units = "degrees" ;
        LONGITUDE_T:Format = "F10.4" ;
    float LATITUDE_T(LATITUDE_T) ;
        LATITUDE_T:long_name = "Latitude" ;
        LATITUDE_T:units = "degrees" ;
        LATITUDE_T:Format = "F10.4" ;
    float DEPTH(DEPTH) ;
        DEPTH:long_name = "Depth" ;
        DEPTH:units = "cm" ;
        DEPTH:Format = "F5.2" ;
        DEPTH:positive = "down" ;
        DEPTH:edges = "DEPTH_EDGES" ;
    float POTENTIAL_TEMPERATURE__MEAN_(DEPTH, LATITUDE_T,
LONGITUDE_T) ;
        POTENTIAL_TEMPERATURE__MEAN_:long_name = "potential
temperature (mean)" ;
        POTENTIAL_TEMPERATURE__MEAN_:units = "C" ;
        POTENTIAL_TEMPERATURE__MEAN_:FillValue = 0.f ;
        POTENTIAL_TEMPERATURE__MEAN_:LEVEL = 0 ;
        POTENTIAL_TEMPERATURE__MEAN_:T_GRID = -1 ;
    float SALINITY__MEAN_(DEPTH, LATITUDE_T, LONGITUDE_T) ;
        SALINITY__MEAN_:long_name = "salinity (mean)" ;
        SALINITY__MEAN_:units = "(PSU-35)/1000" ;
        SALINITY__MEAN_:FillValue = 0.f ;
```

<sup>8</sup> The dbxten VPCLASS was defined as shown, but was not actually used in any of the trials.

```

        SALINITY__MEAN__:LEVEL = 0 ;
        SALINITY__MEAN__:T_GRID = -1 ;
...

// global attributes:
        :parentfile =
"/scratch/arther3/occamp083/run401/monthly/nov2003.h5m1" ;
        :creation_command =
"/noc/users/acc/SUBVOL_UTILS/h52ncsubvol.novel -f
/scratch/arther3/occamp083/run401/monthly/nov2003.h5m1 -o /noc/om
f/scratch/lsm/javad/WORKING/4739/nov2003.h5m1subvol -include
\"POTENTIAL TEMPERATURE, SALINITY, U VELOCITY, V VELOCITY,\" -sw 726
264 -ne 1172 730 -k 1
66 -stride 1 1 1 -domain 66 4320 1735" ;
        :FMODE = 2 ;
        :FTYPE = 2 ;
        :ROTATION = 1 ;

data:

        TIMESTEP = 2077464 ;

...

```

## Appendix C – Source of fetchData Program

```

#include <math.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <DSError.h>
#include <genparseopt.h>
#include <dschip_const.h>
#include <dschip_exports.h>
#include <dschipInformixClient.h>

EXEC SQL include sqltypes;
EXEC SQL include exp_chk.ec;

static FILE *outfile;
static int verbose = 0;

static char *outFileName = NULL;
static char *tableName = NULL;
static char *chipColumnName = NULL;
static char *boxColumn = NULL;

#define maxCOLUMNS (40)
static int numColumns = 0;
static char *columns[maxCOLUMNS];

#define maxRANGES (40)
int numRanges = 0;
char *rangeColumn[maxRANGES];
char *rangeLow[maxRANGES];
char *rangeHigh[maxRANGES];
double rangeHighVal[maxRANGES];

```

```
double rangeLowVal[maxRANGES];
int rangeIsKey[maxRANGES];
int numKeys = 0;

static void SetBoxColumn(char *i)
{
    boxColumn = i;
}

static void SetChipName(char *i)
{
    chipColumnName = i;
}

static void SetFileName(char *i)
{
    outFileNames = i;
}

static void SetTableName(char *i)
{
    tableName = i;
}

static void SetColumn(char *i)
{
    char *item;
    for(;;) {
        item = strtok(i, ",");
        if( !item ) break;

        columns[numColumns++] = item;
        i = NULL;
    }
}

static double ParseScalar(char *strValue)
{
    if( strchr(strValue, '-') > strValue ) {
        return DSGMTStringToDouble(strValue);
    }
    else {
        double val;
        if( sscanf(strValue, "%lf", &val) != 1 ) {
            DSUserError("Bad numeric value '%s'", strValue);
        }
        return val;
    }
}

static void SetRangeCommon(char *i) {
    char buffer[255];
    char *name;
```

```

char *lowStr;
char *highStr;
double lowVal, highVal;

strncpy(buffer, i, sizeof(buffer));
rangeColumn[numRanges] = strtok(i, ",");
rangeLow[numRanges] = strtok(NULL, ",");
rangeHigh[numRanges]= strtok(NULL, ",");
if( !rangeHigh[numRanges]) {
    DSUserError("bad range '%s'", buffer);
}
rangeLowVal[numRanges] = ParseScalar(rangeLow[numRanges]);
rangeHighVal[numRanges] = ParseScalar(rangeHigh[numRanges]);

numRanges++;
}

static void SetRange(char *i){
    rangeIsKey[numRanges] = 0;
    SetRangeCommon(i);
}

static void SetKey(char *i) {
    rangeIsKey[numRanges] = 1;
    numKeys++;
    SetRangeCommon(i);
}

static void ProcessSingleChip(DSChip *chip)
{
    int chipRows, numVars;
    int i, j;
    DSChipVar *vars[maxCOLUMNS];
    DSChipVar *rangeVars[maxCOLUMNS];
    char format[maxCOLUMNS][10];
    int types[maxCOLUMNS];

    chipRows = DSChipGetNumRows(chip);
    for(i = 0; i < numColumns; i++ ) {
        vars[i] = DSChipGetVarByName(chip, columns[i]);
        if( vars[i] == NULL ) {
            if( verbose ) {
                fprintf(stderr, "skipping chip without column %s",
columns[i]);
            }
            return;
        }
        types[i] = DSChipVarGetType(vars[i]);
        if( types[i] == DSVarTypeDOUBLE ) {
            int fractionLength;
            double tolerance;
            tolerance = DSChipVarGetTolerance(vars[i]);
            if( tolerance == 0 ) {
                fractionLength = 14;
            }
            else if( tolerance >= 1 ) {
                fractionLength = 0;
            }
        }
    }
}

```

```

    }
    else {
        fractionLength = (int)ceil(-log10(tolerance));
    }
    sprintf(format[i], "%%.%df", fractionLength);
}
}

for(i = 0; i < numRanges; i++ ) {
    rangeVars[i] = DSChipGetVarByName(chip, rangeColumn[i]);
}

for( j = 0; j < chipRows; j++ ) {
    int isInside = 1;

    for( i = 0; i < numRanges; i++ ) {
        double val = DSChipVarGetDouble(rangeVars[i], j);
        if( val < rangeLowVal[i] || val > rangeHighVal[i] ) {
            isInside = 0;
            break;
        }
    }
    if( !isInside ) continue;

    for( i = 0; i < numColumns; i++ ) {
        if( i > 0 ) {
            fprintf(outfile, ",");
        }
        switch( types[i] ) {
            case DSVarTypeDATE:
                {
                    char timeBuffer[80];
                    DSDoubleToGMTString(timeBuffer,
                                        DSChipVarGetDouble(vars[i], j), 4);
                    fprintf(outfile, "%s", timeBuffer);
                }
                break;
            case DSVarTypeDOUBLE:
                fprintf(outfile, format[i], DSChipVarGetDouble(vars[i],
j));
                break;
            case DSVarTypeSTRING:
                fprintf(outfile, "%s",
                    DSChipVarGetString(vars[i], j));
                break;
            case DSVarTypeINT:
                fprintf(outfile, "%d",
                    DSChipVarGetInt(vars[i], j));
                break;
        }
    }
    fprintf(outfile, "\n");
}
}

static char *BuildWhereClause()
{

```

```

int i;
int keyCount;
char *buffer;
int bufferLen;

if( numKeys == 0 ) {
    return "";
}
if( boxColumn == NULL ) {
    DSUserError("keys specified but no boxcolumn argument");
}

bufferLen = numRanges*256+1 + 256;
buffer = DSMalloc(bufferLen);
buffer[0] = '\0';

strncat(buffer," where overlap(", bufferLen);
strncat(buffer, boxColumn, bufferLen);
strncat(buffer," ,DSRangeToBox('", bufferLen);
keyCount = 0;
for( i = 0; i < numRanges; i++ ) {
    if( rangeIsKey[i] ) {
        if( keyCount > 0 ) {
            strncat(buffer, ",", bufferLen);
        }
        strncat(buffer, rangeColumn[i], bufferLen);
        strncat(buffer, " ", bufferLen);
        strncat(buffer, rangeLow[i], bufferLen);
        strncat(buffer, " ", bufferLen);
        strncat(buffer, rangeHigh[i], bufferLen);
        keyCount++;
    }
}
strncat(buffer,"')", bufferLen);
return buffer;
}

static char * BuildQueryText() {
    int bufferLen;
    char *buffer;
    char *whereClause;

    whereClause = BuildWhereClause();
    bufferLen = strlen(whereClause) + numColumns*80 + 256;
    buffer = DSMalloc(bufferLen);
    buffer[0] = '\0';
    strncat(buffer, "select ", bufferLen);
    strncat(buffer, chipColumnName, bufferLen);
    strncat(buffer, " from ", bufferLen);
    strncat(buffer, tableName, bufferLen);
    strncat(buffer, whereClause, bufferLen);
    return buffer;
}

static void FetchChips()
{

```

```
EXEC SQL BEGIN DECLARE SECTION;
    var binary "dschip" chipVar=NULL;
    char *queryText;
EXEC SQL END DECLARE SECTION;
DSChip *chip;
int chipCnt;

queryText = BuildQueryText();
if( verbose ) {
    fprintf(stderr, "query was:\n%s\n", queryText);
}

/*
 * set the size of the fetch buffer to reduce the number of
roundtrips
 * to the server. The default is max(4096,one row).
 */
FetBufSize = 32000;

EXEC SQL whenever sqlerror CALL processSQLException;

EXEC SQL connect to "noaa";

EXEC SQL prepare coll_stmt from
    :queryText ;

EXEC SQL declare noaacursor cursor for coll_stmt;

EXEC SQL open noaacursor;

for(chipCnt=0;;chipCnt++) {
    EXEC SQL fetch noaacursor into :chipVar ;

    if( strcmp( SQLSTATE, "00", 2) != 0 ) {
        break;
    }
    if( verbose ) {
        fprintf(stderr,"fetched chip\n");
    }
    chip = DSChipUnPack(chipVar);
    ProcessSingleChip(chip);
    DSChipFree(chip);
    ifx_var_dealloc(&chipVar);
}

if( verbose ) {
    fprintf(stderr,"about to close cursor\n");
}
EXEC SQL close noaacursor;
EXEC SQL disconnect current;

if( verbose ) {
    fprintf(stderr, "chip count was %d\n", chipCnt);
}
}
```

```

static void SetVerbose() {
    verbose = 1;
}

static OptLstType MyOpts[] = {
    { "o-", SetFileName },
    { "table-", SetTableName },
    { "boxcolumn-", SetBoxColumn},
    { "columns-", SetColumn },
    { "chip-", SetChipName },
    { "key-", SetKey },
    { "range-", SetRange },
    { "verbose", SetVerbose },
    { "\0", NULL }
};

static void printUsage(char *name) {
    fprintf(stderr, "Usage: %s arguments\n", name);
    fprintf(stderr, "where arguments include:\n");
    fprintf(stderr, "\t-o output filename # default is stdout\n");
    fprintf(stderr, "\t-table tablename\n");
    fprintf(stderr, "\t-boxcolumn boxcolumn # for rtree indexing\n");
    fprintf(stderr,
        "\t-columns outputColumns # comma separated, may be
repeated\n");
    fprintf(stderr, "\t-chip name_of_dschip_column\n");
    fprintf(stderr, "\t-range 'columnname,lowValue,highValue'\n");
    fprintf(stderr, "\t-key 'columnname,lowValue,highValue'\n");
    fprintf(stderr, "Note: keys are special cases of range.\n");
}

int main(int argc, char *argv[]) {
    DSChip *chip;

    DSErrSource(argv[0]);

    if( argc == 1 ) {
        printUsage(argv[0]);
    }
    argc = GenParseOpt( argc, argv, MyOpts);
    if( argc != 1 ) {
        fprintf(stderr, "unrecognized arg: %s\n", argv[1]);
        printUsage(argv[0]);
    }
    if( tableName == NULL ) {
        DSUserError("missing -table argument");
    }
    if( numColumns == 0 ) {
        DSUserError("No -columns argument");
    }
    if( chipColumnName == NULL ) {
        DSUserError("No -chip argument");
    }
    if( outFile != NULL ) {
        outfile = fopen(outFile, "w");
        if( !outfile ) {
            DSUserError("Unable to open file '%s' for output");
        }
    }
}

```

```
        }
    }
    else {
        outfile = stdout;
    }
    FetchChips();
    if( outfile != stdout ) {
        fclose(outfile);
    }
    return 0;
}
```