ALFATEC Conference 2010

Informix Performance in a Heavy Duty Environment

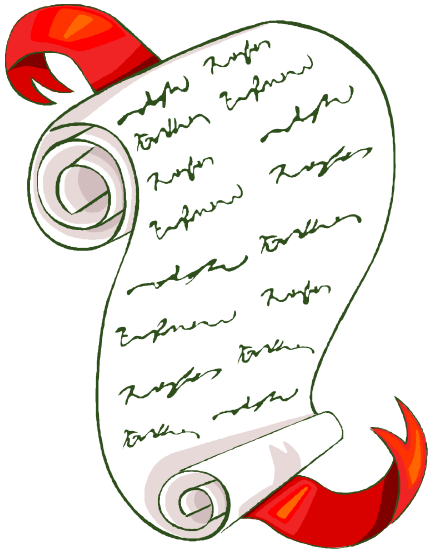
John F. Miller III

Support Architect

miller3@us.ibm.com

ALFATEC 2010 Conference

- **Review of IDS 11 Performance Features**
- **Storage Management Optimizations**
- **Finding poor performing SQL statements**
- **Sizing the VP Private Memory Cache**
- **Improved Optimizer Statistics**
  - **Temp Tables**
  - **Index**
- **Optimizing Database Network Traffic**



Some of Version 11 Performance Features

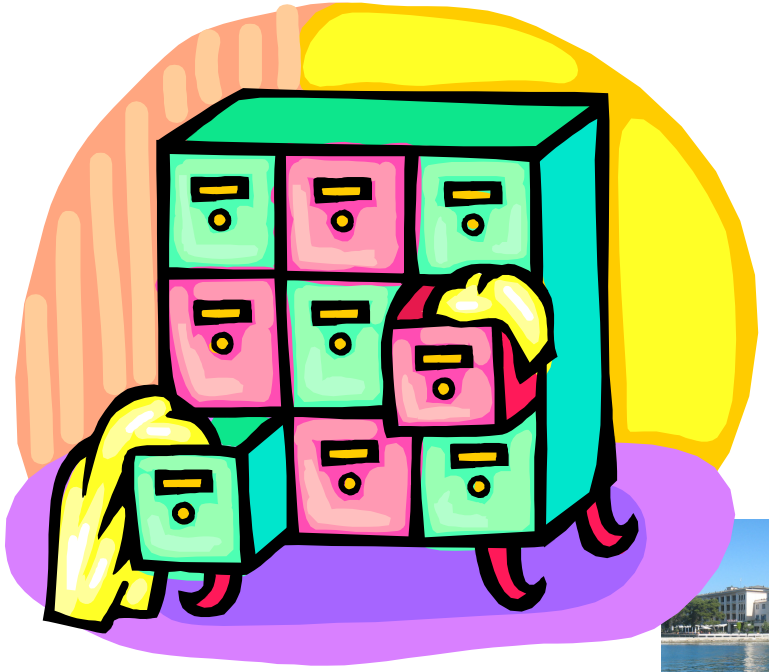
- Fast Polling
- Direct I/O for cooked files
- Determine data concurrency with a version column
- Checkpoint Improvements
- Enhanced Concurrency with Committed Read Isolation
- Improve Performance Monitoring
- Index Self-Join Query Plans
- Private Memory Caches for Virtual Processors
- Predicable Fast Recovery
- Prepare time with DBA cache
- Dynamic SQL Statements in SPL Routines
- Global control of temp table logging
- Multiple MISC VPs
- Enhanced Support for Large Tables
- Optimizer Improvements
- Engine Algorithm Changes
- Set Reads for Index Scans
- Native large Integers and Serial
- New automatic update stats
- Improved default onconfig
- Startup script customization

Some of Version 11 Performance Features

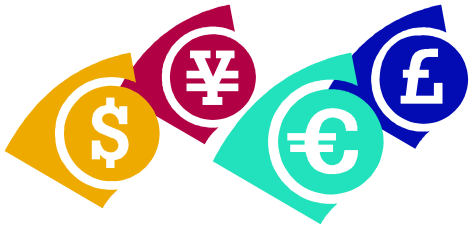
- 11.50.xC4
  - Table Compression
  - Table Repack
  - Table Shrink
  - Historic SQL Tracing
- 11.50.xC5
  - Merge
  - Connect By
- 11.50.xC6
  - Enhanced Performance of Light Scan
  - External Tables
  - BTS Process Multiple Simultaneous Queries

- 11.50.xC3
  - SQL Admin API updates onconfig
  - Runtime updating of parameters LTXEHWM, LTXHWM, and DYNAMIC_LOGS
  - Runtime updating of ER parameters in onconfig
  - Administrate ER with SQL Admin API
  - Alter a tables first extent size
  - Rolling Back SQL Transactions to a Savepoint
  - Capturing transactional data with the change data capture API
  - Basic Text Search supports smart blobs & MACH
  - Querying XML Attributes with the Basic Text DataBlade Module
  - Setting the Frequency of Error Checking for Smart Large Object Transmission
- 11.50.xC2
  - Configurable Runtime Index Compression
  - Subquery support in Delete and Update
  - Globally limiting the number of sessions that can connect to IDS
  - Limiting memory during ER sync
  - New sysmaster monitoring tables for ER
  - Longer return strings functions (i.e. TRIM)

Storage Optimization



Storage Management Overview



- Data Compression
  - Reduces the amount of storage taken by a single row

- Table Compaction
  - Reduce the number of pages utilized by a table

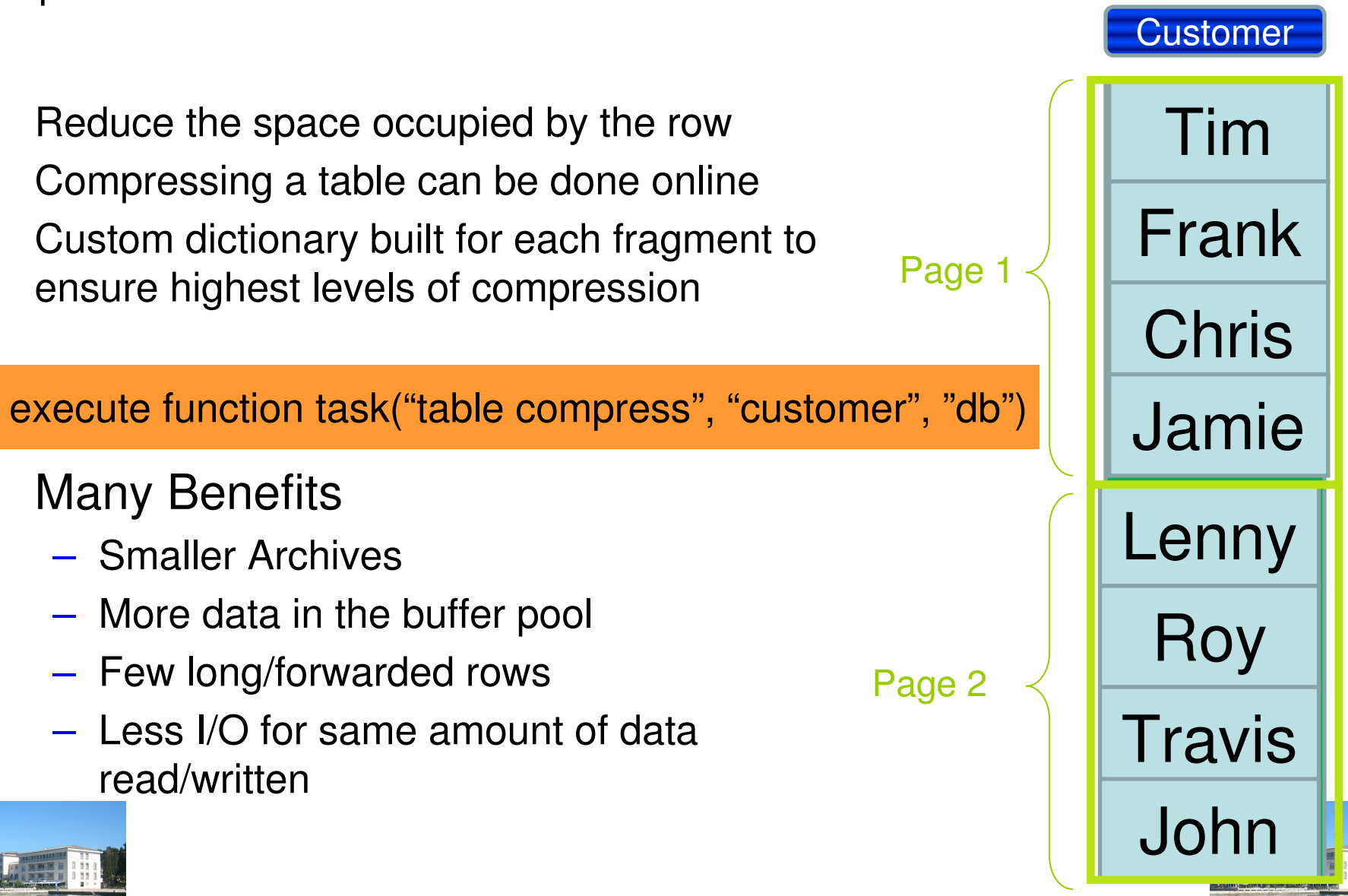
- Index Compaction
  - Ensure the index pages are kept full, reducing the number of pages it takes to store the index

Compress Command

- Reduce the space occupied by the row
- Compressing a table can be done online
- Custom dictionary built for each fragment to ensure highest levels of compression

```
execute function task("table compress", "customer", "db")
```

- Many Benefits
  - Smaller Archives
  - More data in the buffer pool
  - Few long/forwarded rows
  - Less I/O for same amount of data read/written



REPACK Command

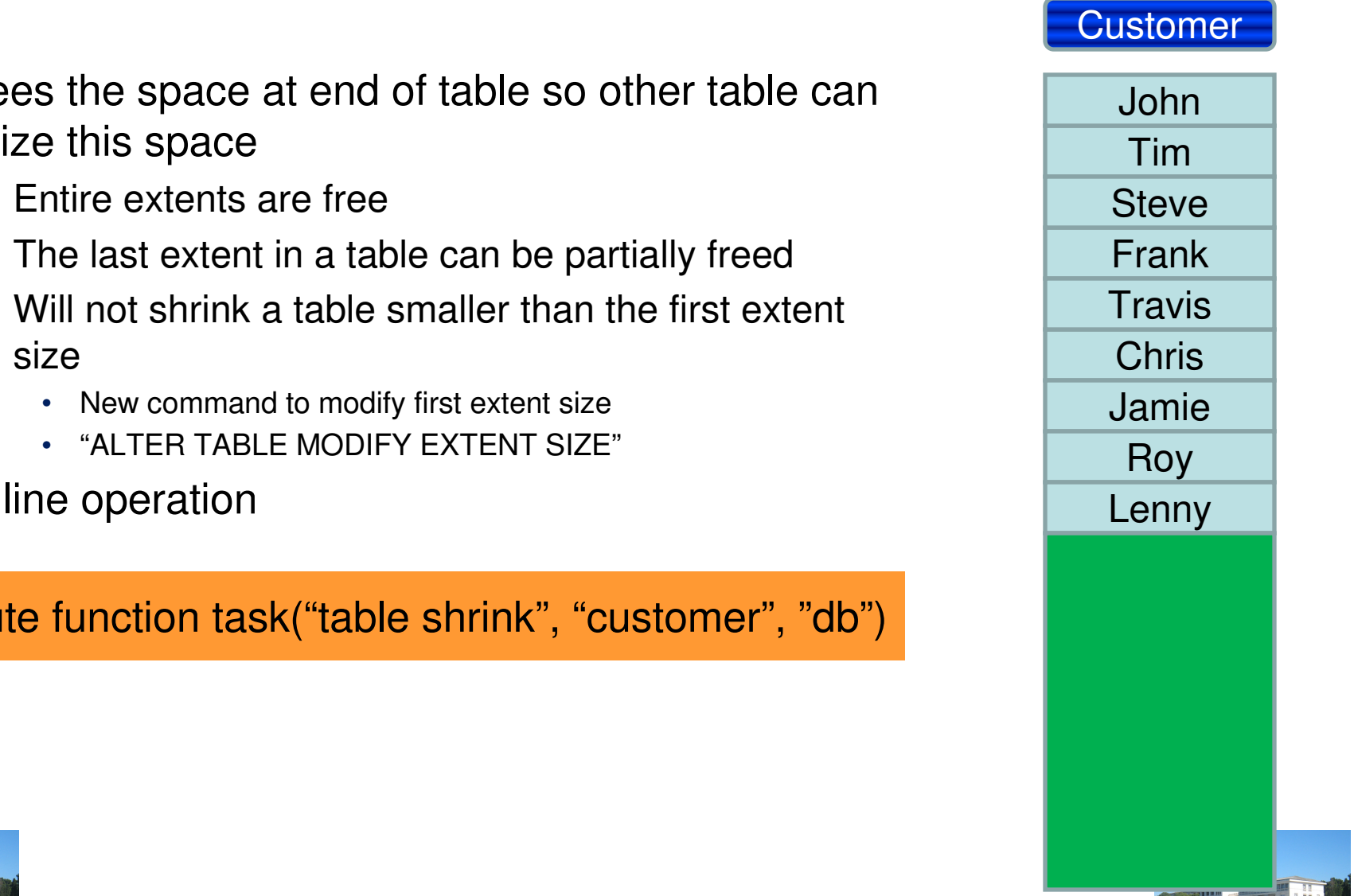
- Moves all rows in a table/fragment to the beginning, leaving all the free space at the end of the table
- Online operation, users can be modifying the table

execute function task(“table repack”, “customer”, ”db”)

SHRINK Command

- Frees the space at end of table so other table can utilize this space
  - Entire extents are free
  - The last extent in a table can be partially freed
  - Will not shrink a table smaller than the first extent size
    - New command to modify first extent size
    - “ALTER TABLE MODIFY EXTENT SIZE”
- Online operation

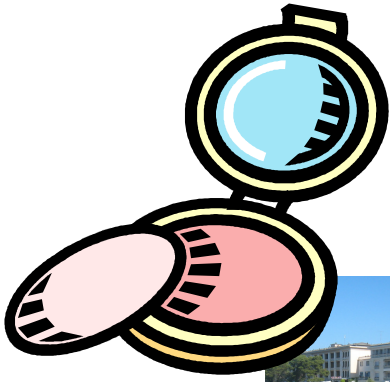
execute function task(“table shrink”, “customer”, ”db”)



Dynamic Runtime Index Compaction

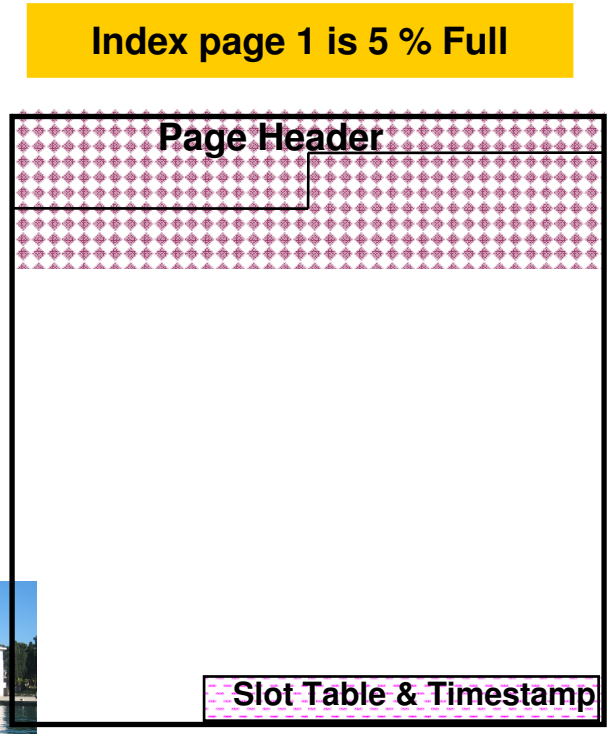
- Attempts to merge two partially used index pages and free one index page
- Currently an index page must be more than 90% free before Informix will attempt to look for merges
- Provide an adjustable system wide default

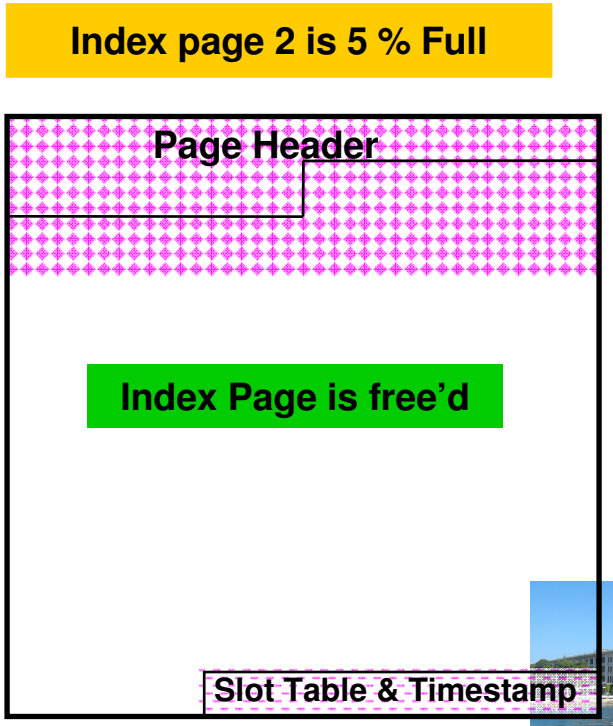
```
BTSCANNER num=1,threshold=5000,rangesize=-1,alice=6,compression=high
onmode –C compression high
```

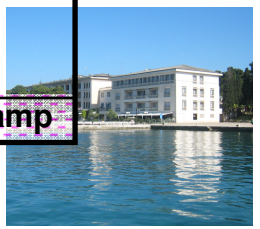


What is a Index Page Merge?

- Moves index information to adjacent page, unlinks from index and removes pages from the index
- When are index pages merged?
  - Is index page 1 below the merge threshold? **YES**
  - Read Index page #2
  - Is index page 1 + index page 2 below the split threshold? **YES**







Automatic Index Merge and Compaction

- Will the current index page attempt an automatic index merging??
- Only the btree scanner thread(s) merge index pages

Operation	Before 11.50.xC2	11.50.xC2 and Later
Index Page which is 15% used	**No**	**Merged if below threshold**
Empty Index Page with deleted items	**Merged if below 10%**	**Merged if below threshold**
Empty Index Page w/o deleted items	**No**	**Merged if below threshold**

SQLTRACE

Detailed talk on this subject on IIUG website

“Help Yourself with SQLTRACE”

What is SQLTRACE?

- SQLTRACE is a new feature in IDS 11 that allows you to collect SQL statement history information, including statistics and resource usage at the statement level.

- Each trace information includes
  - Optimizer statistics (like SET Explain and EXPLAIN_STAT)
    - Iterator database and table names are available in MED mode
  - Buffer and read/write statistics (like in onstat –p)
  - SQL info like onstat –g sql (database name, session id, database isolation level, **statement host variables**)
    - Host variables are only available in HIGH mode
  - Procedure stacks are available in HIGH mode

Query Drill Down onstat –g his


```
Database:         sysmaster
Statement text:
 select count(*) from systables,syscolumns where systables.tabid > ? and
   systables.nrows < ?

 SELECT using tables [ systables syscolumns ]

Iterator/Explain
================
   ID    Left  Right    Est Cost    Est Rows    Num Rows     Partnum Type
    3       0      0          17          42         146     1048579 Index Scan
    4       0      0        5249        2366        2366     1048580 Seq Scan
    2       3      4        5266       99372      345436           0 Nested Join
    1       2      0           1           1           1           0 Group

Host Variables
==============
1 integer    100
2 float      1000.0000000000000000
```

Query Drill Down -- *Continued*

Statement information:

Sess_id	User_id	Stmt Type	Finish Time	Run Time
5	2053	SELECT	01:08:48	0.4247

Statement Statistics:

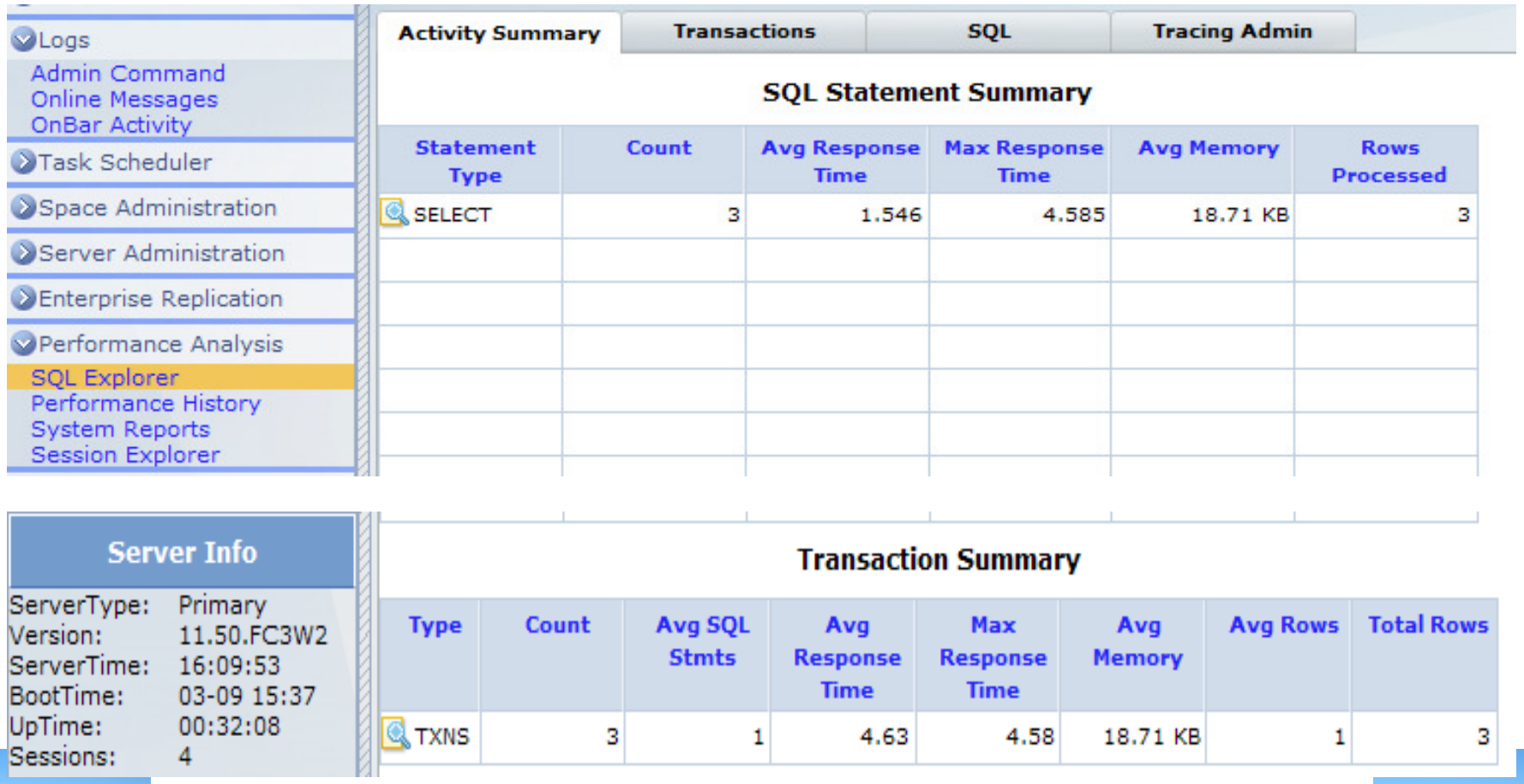
Page Read	Buffer Read	Read % Cache	Buffer IDX Read	Page Write	Buffer Write	Write % Cache
65	75638	99.91	0	0	0	0.00

Lock Requests	Lock Waits	LK Wait Time (S)	Log Space	Num Sorts	Disk Sorts	Memory Sorts
345729	0	0.0000	0.000 B	0	0	0

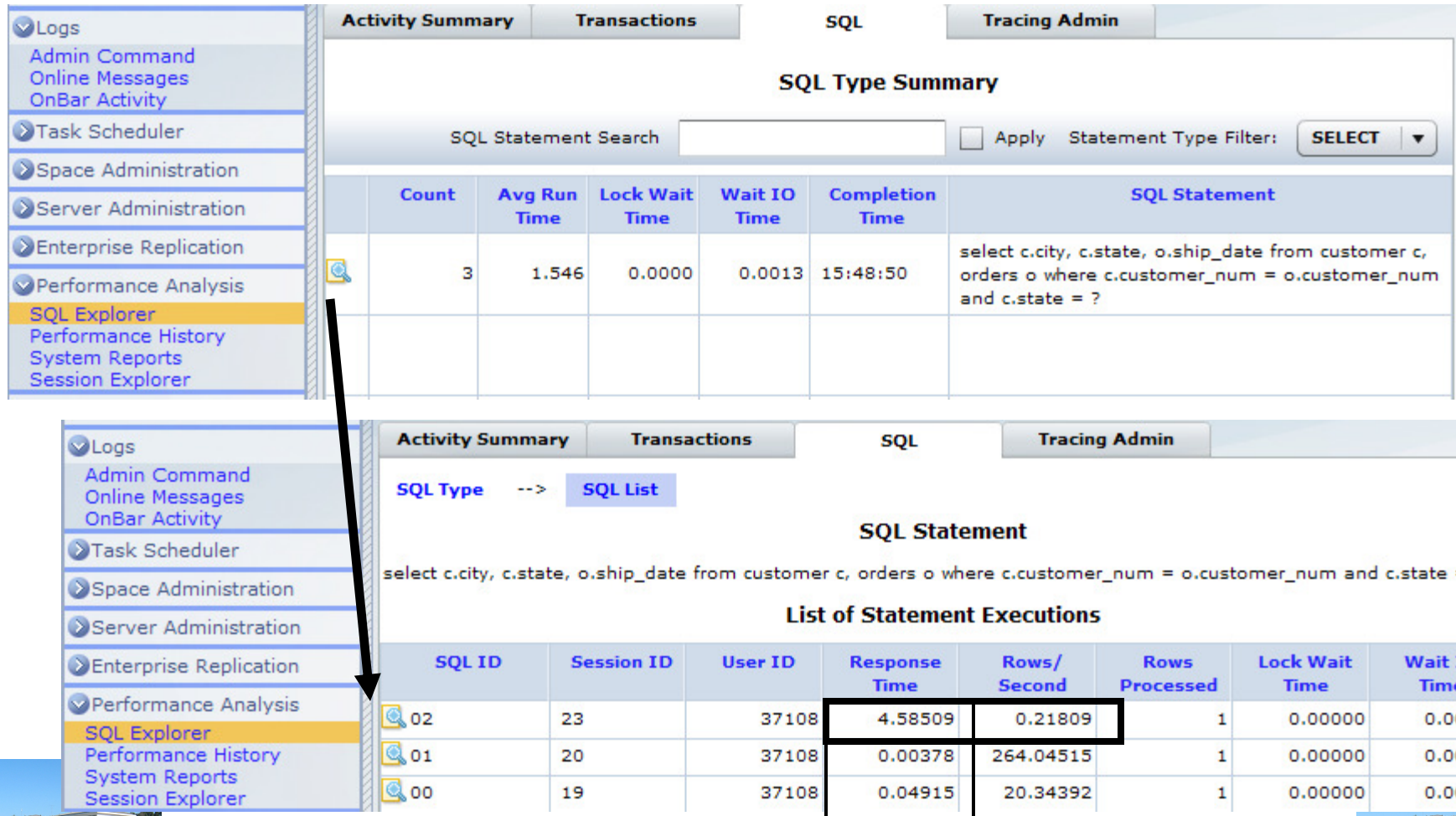
Total Executions	Total Time (S)	Avg Time (S)	Max Time (S)	Avg IO Wait	I/O Wait Time (S)	Avg Rows Per Sec
1	0.8493	0.8493	0.4247	0.000203	0.0026	2.3548

Estimated Cost	Estimated Rows	Actual Rows	SQL Error	ISAM Error	Isolation Level	SQL Memory
5266	1	1	0	0	CR	25504

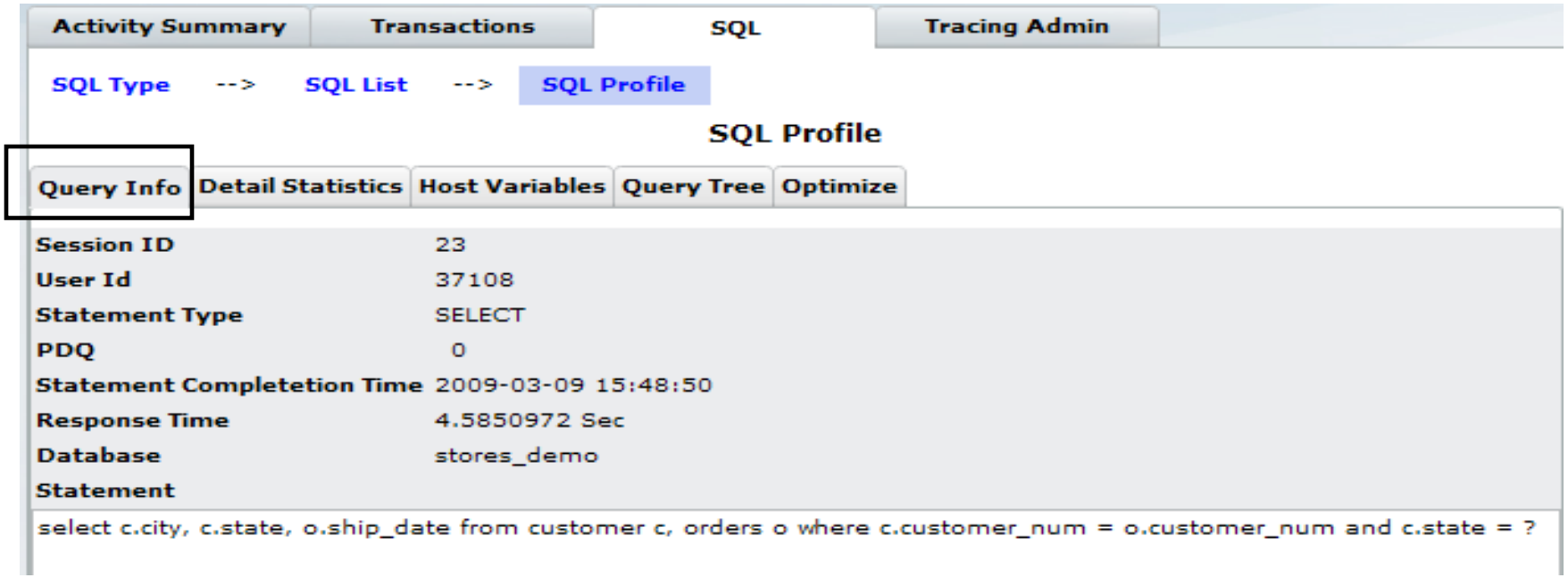
Query Drill-down (cont'd)



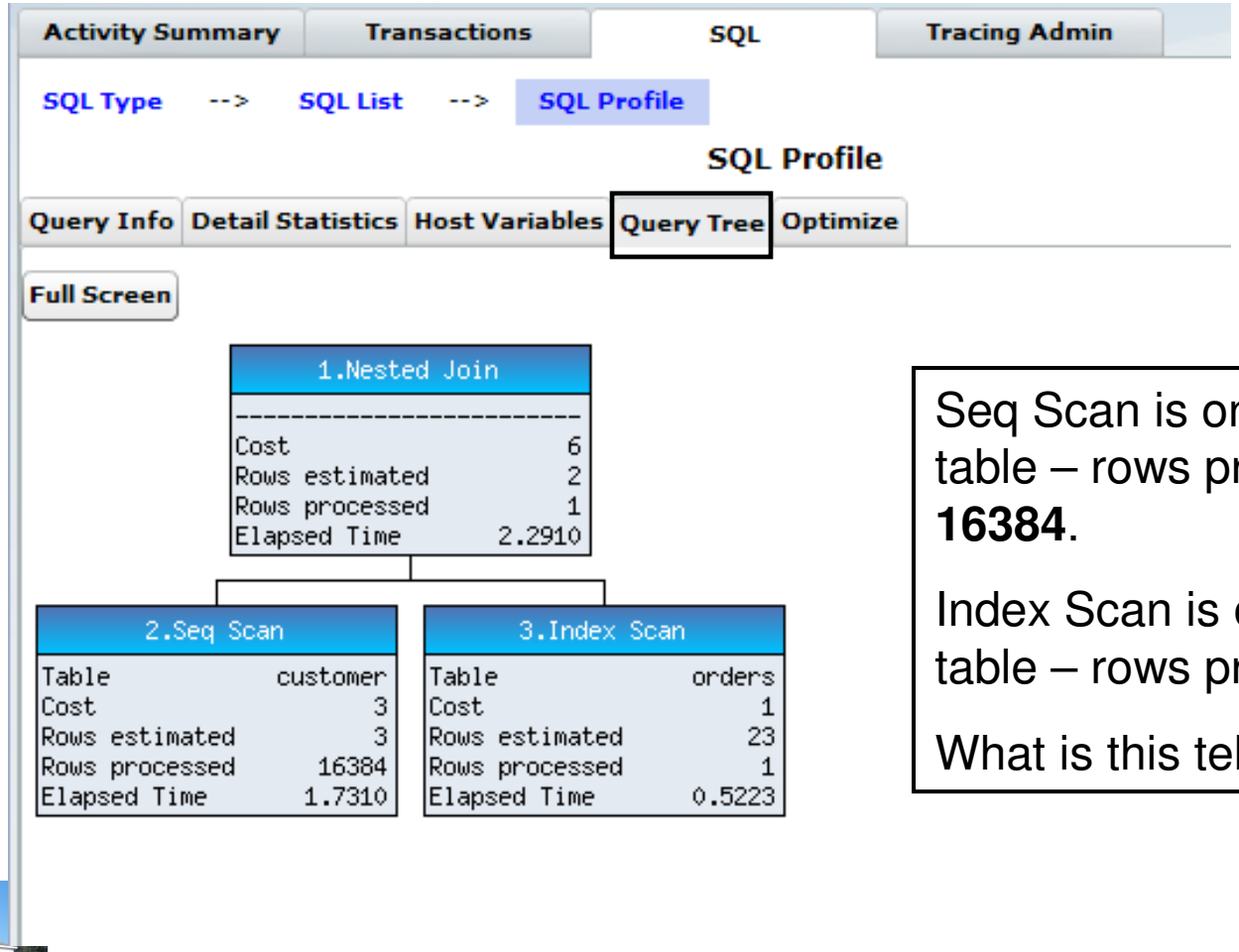
Query Drill-down (cont'd)



Query Drill-down (cont'd)



Query Drill-down (cont'd)

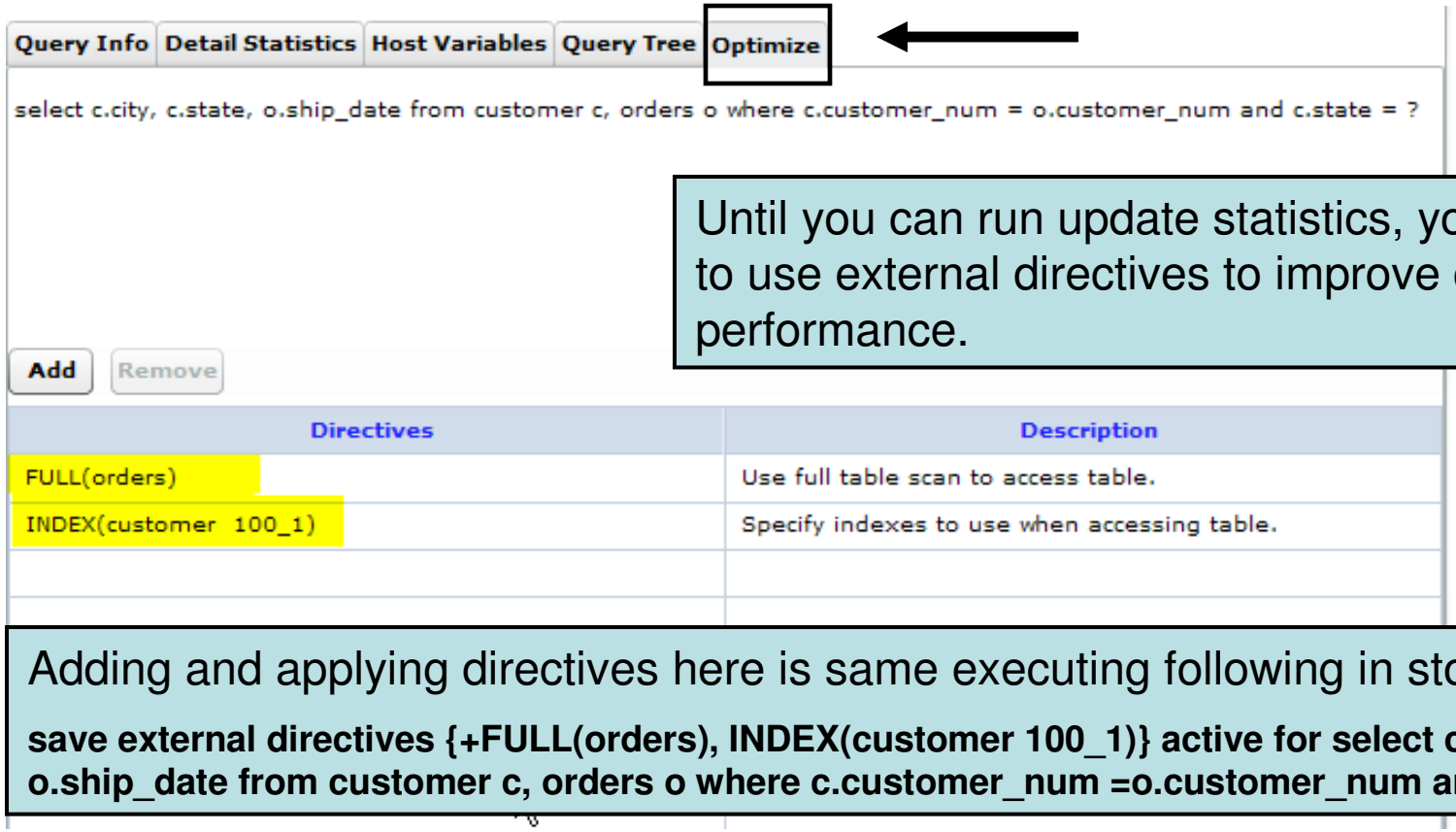


Seq Scan is on the **customer** table – rows processed is **16384**.

Index Scan is on the **orders** table – rows processed is **23**.

What is this telling you ?

Query Drill-down: Add External Directive



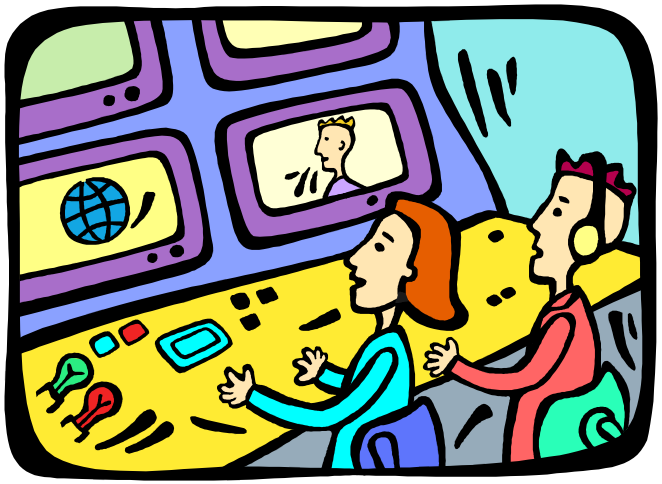
Until you can run update statistics, you decide to use external directives to improve query performance.

Adding and applying directives here is same executing following in stores_demo:

save external directives {+FULL(orders), INDEX(customer 100_1)} active for select c.city, c.state, o.ship_date from customer c, orders o where c.customer_num =o.customer_num and c.state = ?;

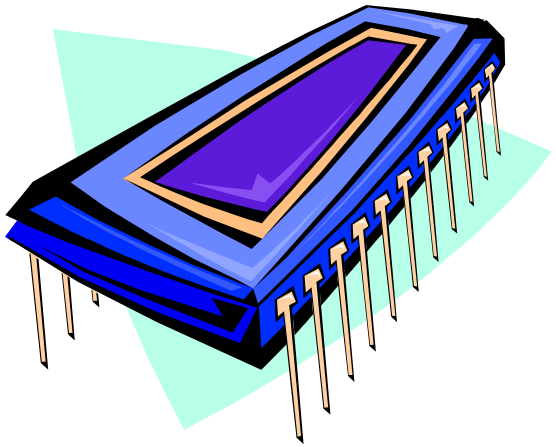
** Check that the directive has been added to stores_demo: sysdirectives .

Improving Memory Allocation Performance

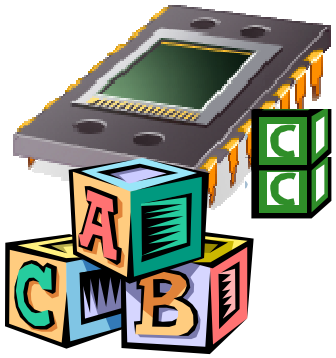


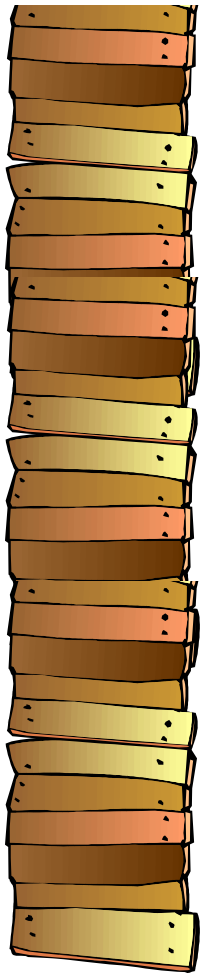
Understanding VP Private Memory Cache

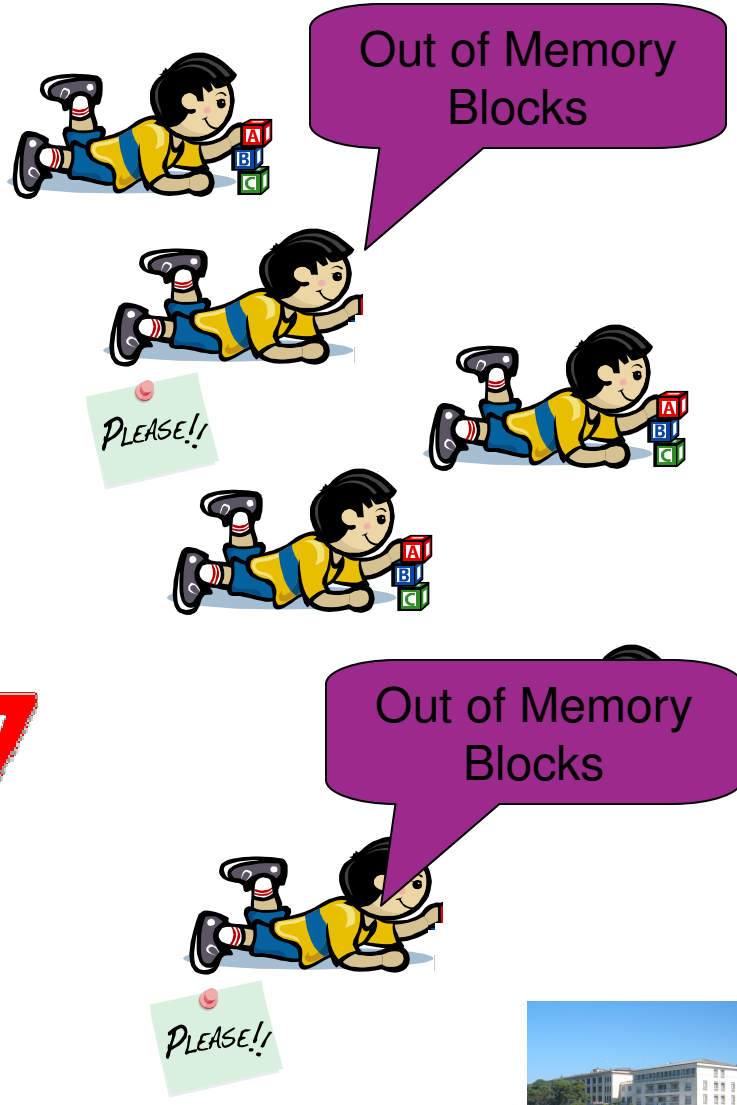
- Allows CPU VPs to allocate & free memory without acquiring a mutex
- Improves the speed when multiple CPU are frequently allocating and freeing memory
- Self tuning to the current work load
- Dynamically enable or disabled
- Significant benefit seen on 4 CPU VP systems



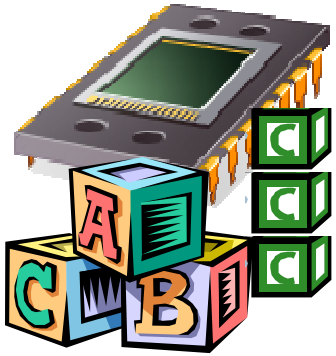
Allocating Memory Blocks

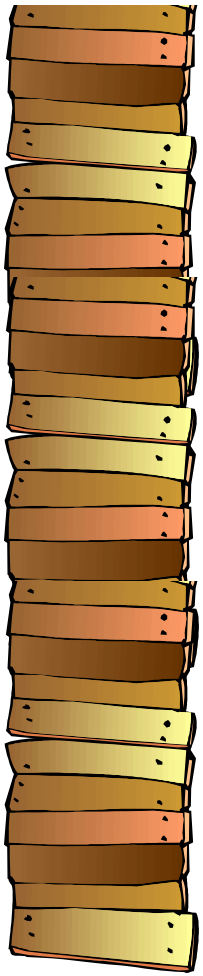


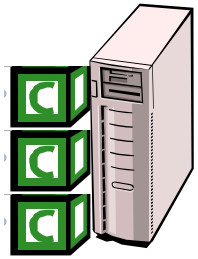


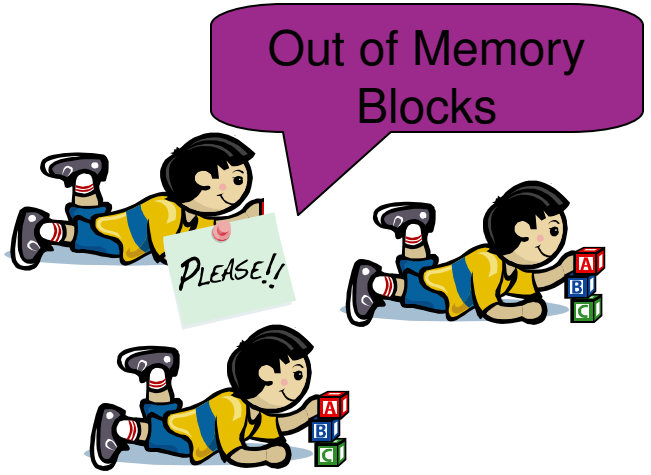


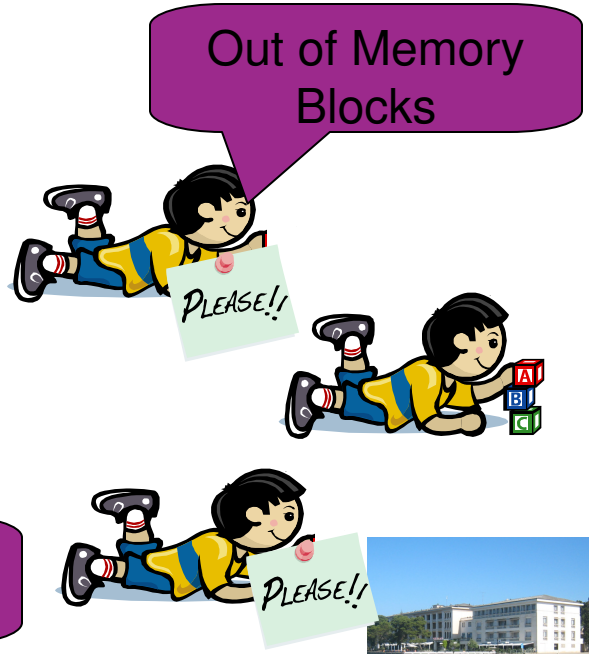
Allocating Memory Block with VP Cache

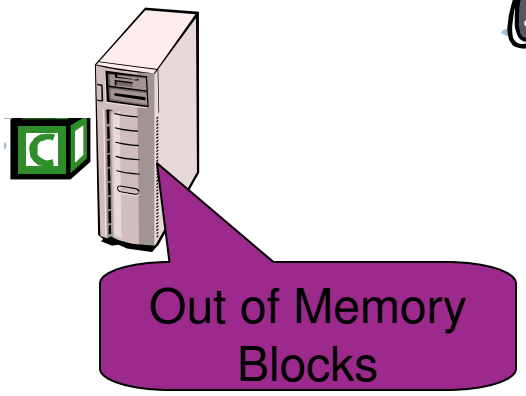




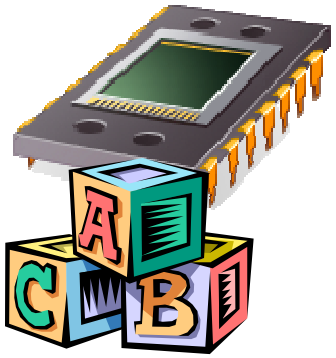


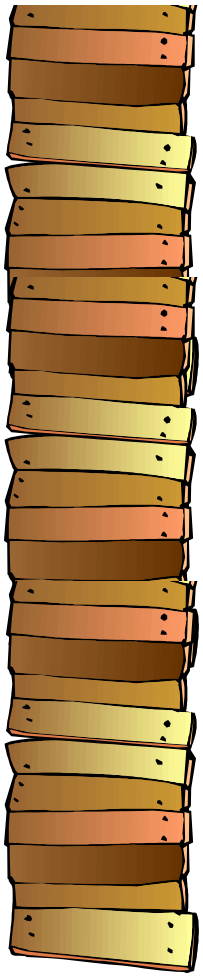


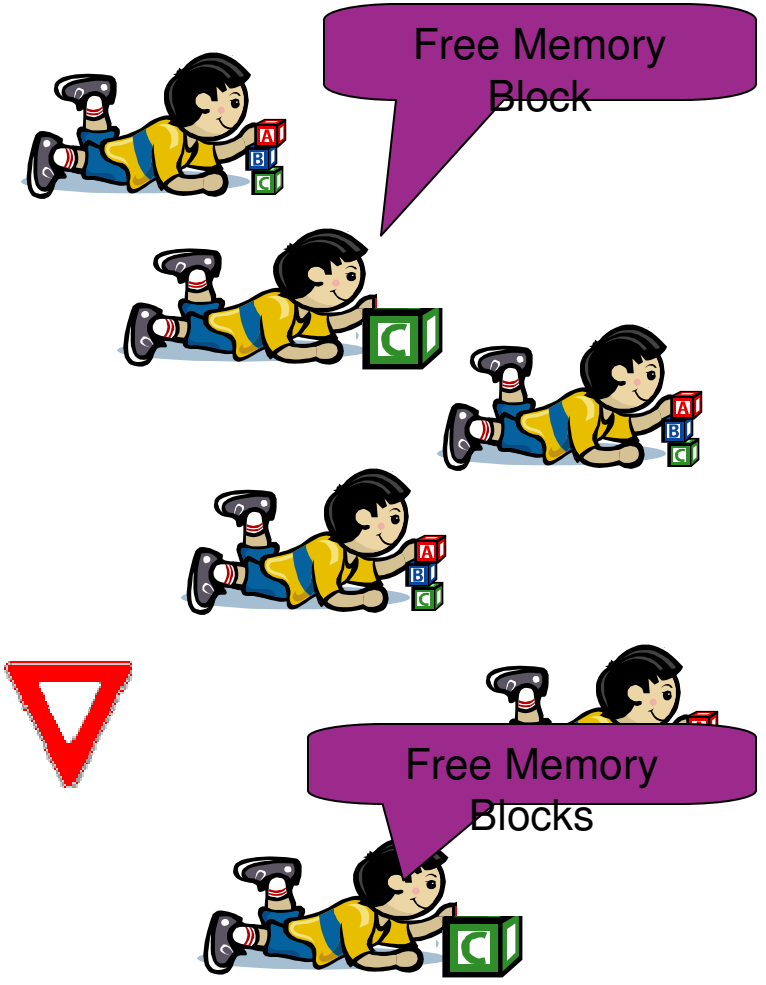




Freeing Memory Blocks

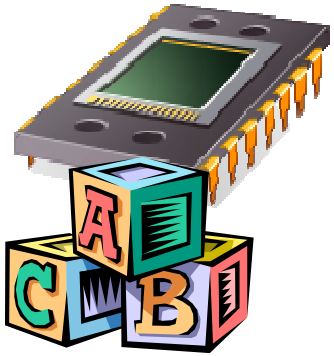


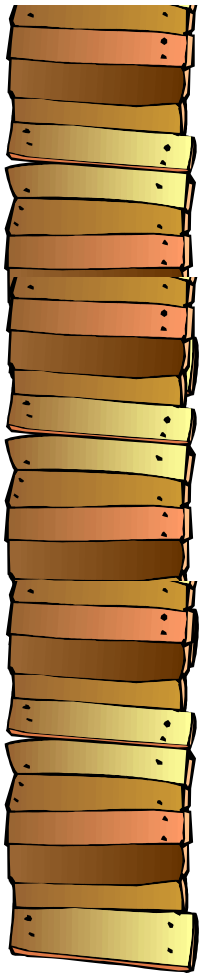


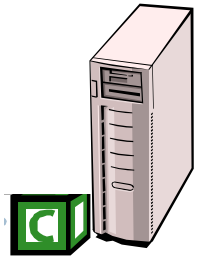


Freeing Memory Block with VP Cache

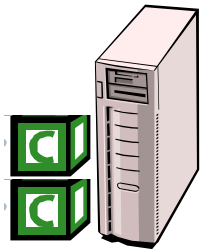
VP_MEMORY_CACHE_KB is set to 12KB (or 3 Blocks)

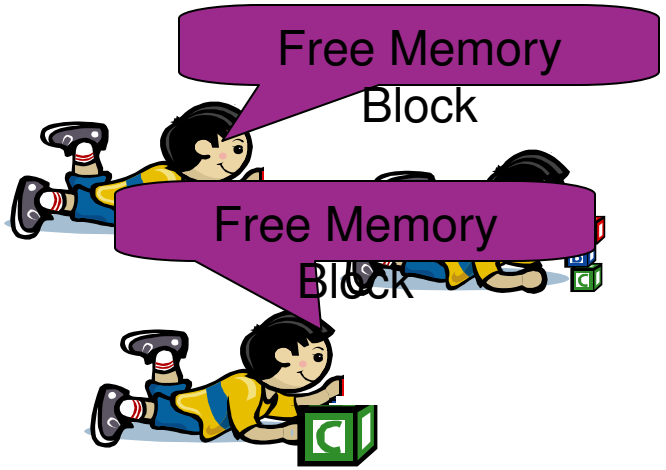


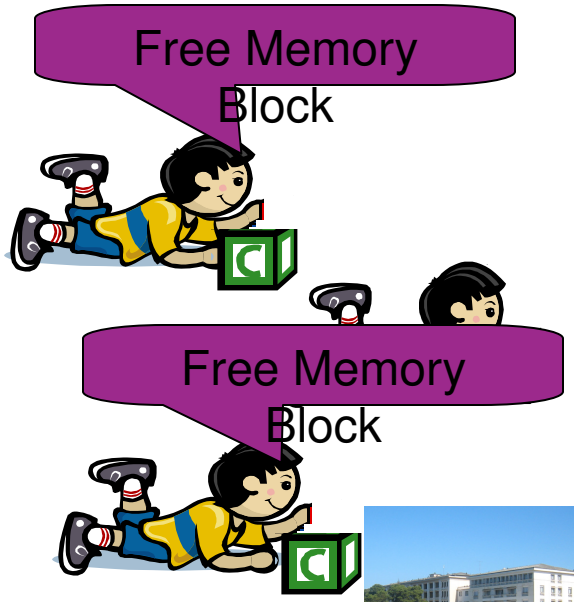




VP Cache is full





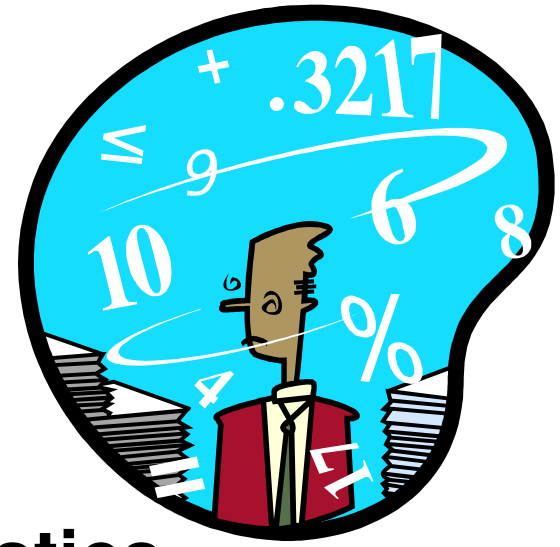


VP Private Memory Cache onstat –g vpcache

Number of times we have allocated memory and avoid IDS's main memory

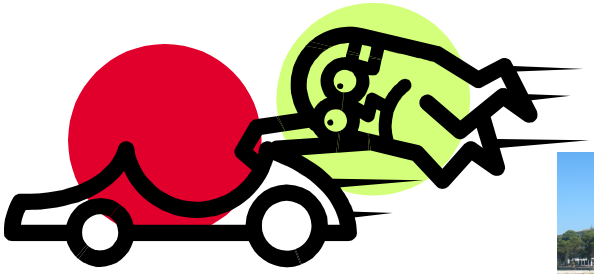
Number of times we free memory and avoid avoid IDS's main memory

```
CPU VP memory block cache statistics - 4096 byte blocks

Number of 4096 byte memory blocks requested for each CPU VP:256

vpid    pid         Blocks held  Hit percentage  Free cache
1       29655       273           73.9 %          94.6 %

Current VP total allocations from cache:     8233, Total frees:     8818

    size  cur blks   alloc      miss       free      drain
    1     31         5045       2660       5128      52
    2     44         1839       115        2010      149
    3     21         358        47         429       64
    4     4          0          0          62        61
    5     10         247        3          265       16
```



Improved Optimizer Statistics

- Auto Statistics on index creation
- Improved Tracking of Update Statistics
- Improved Temp Table Statistics
- Auto Update Statistics



Update Statistics Improvements

- Update statistics high and low are created automatically when an index is created either implicit or explicit.
  - Leverages the sorted data produced by create index
  - Little to no additional time is added to the create index
- Users are no longer required to run update statistics low on temp tables.
  - Number of rows and pages is updated every time we access the temp tables data dictionary entry.
  - Adding indexes to temp tables will automatically create distributions and statistics for the temp table.

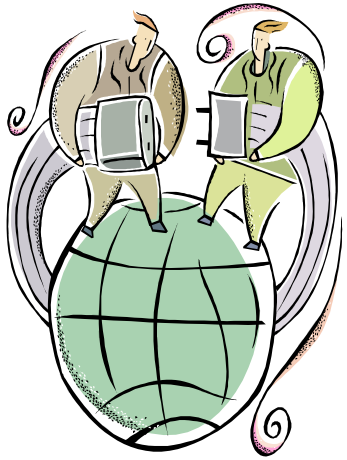
Tracking of Update Statistics

- Low / Statistics
  - Columns added to systables to track last time low was run

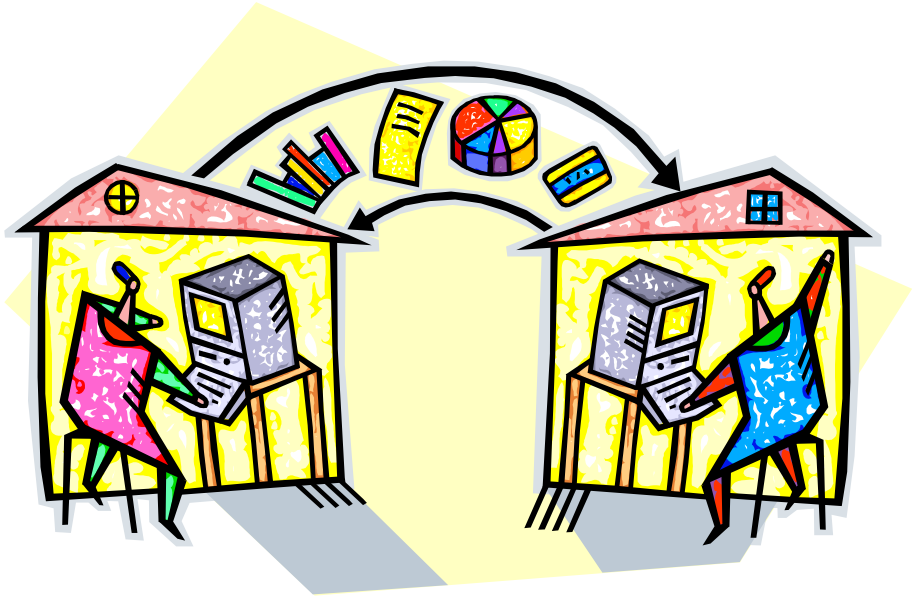
```
SELECT ustlowts FROM SYSTABLES WHERE tabname = “foo”;

        ustlowts   2006-09-26 11:42:50.00000
```

- Medium & High / Distributions
  - Four new columns added to sysdistrib to track distribution creation

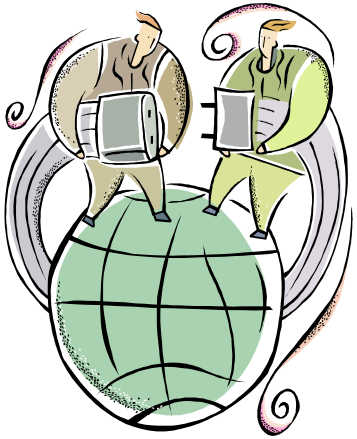
```
SELECT constr_time, smplsize, rowssmpld, ustnrows FROM sysdistrib WHERE colno=1
   AND tabid = (SELECT tabid FROM systables WHERE tabname='foo');

    constr_time                  smplsize     rowssmpld      ustnrows

    2007-09-07 13:43:59.00000        0.00     66.00000         66.00000
```

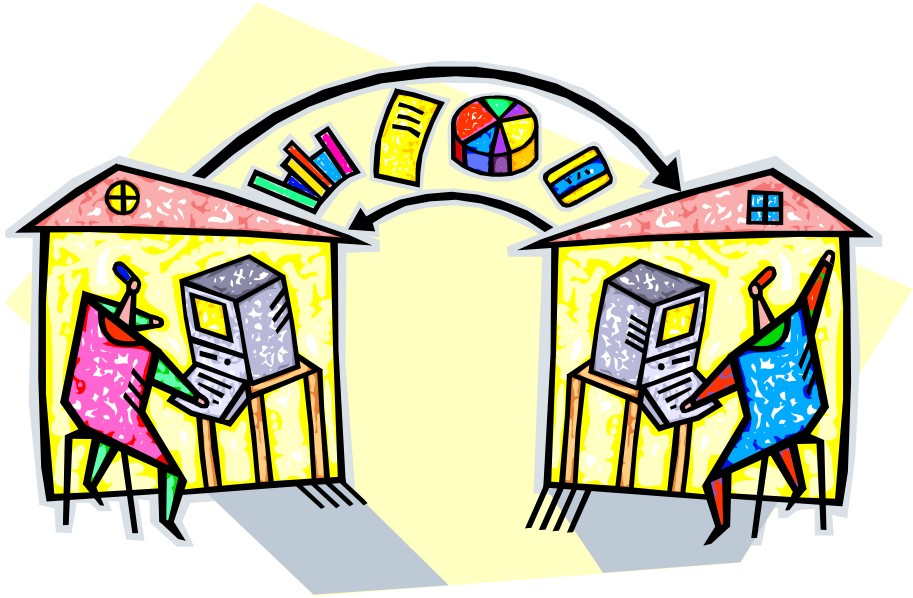


Network Messaging



- **Data Buffer Management**
- **Optimize Open-Fetch-Close (OPTOFC)**
- **Deferred Prepare**
- **Auto Free**





Examining SQL Network Messages

- Network messages sent between the application and the database engine to accomplish SQL operations

```
PREPARE stmt FROM
        "select * from t where a > ?";
DECLARE cursor_1 from :stmnt;
OPEN cursor_1 USING var_1;
FETCH cursor_1 into results_1;
CLOSE cursor_1;
FREE cursor_1;
FREE stmt;
```

Normal Message Traffic

- 128 KB of data returned by the select
- No blob data and rows smaller than 4KB

PREP/DEC ⟷ Small Sent - Medium Returned
OPEN ⟷ Small Sent - Medium Returned
FETCH ⟷ (32x) Small Sent - Medium Returned
CLOSE ⟷ Small Sent - Small Returned
FREE ⟷ Small Sent - Small Returned

Total number of messages sent = 72

Improving SQL Network Messages

- FET_BUF_SIZE
  - Set the buffer size for the fetch/insert buffers
  - Size is specified in bytes up to 32KB
  - Requires no application changes
  - Yields similar performance as fetch array
- OPTOFC – Optimize Open Fetch Close
  - Requires only error checking changes
  - Delays sending the *open* until the first fetch is requested by the application
  - The engine will close the cursor when the last row is processed
- Deferred Prepare
  - Control whether a client process postpones sending a PREPARE statement to the database server until the OPEN or EXECUTE statement is sent
- Auto Free
  - Frees the cursor when the cursor is closed

OPTOFC Message Traffic

- 128 KB of data returned by the select
- No blob data and rows smaller than 4KB

- OPTOFC enabled ⟶ OPTOFC=1
- Fetch buffer set at 32KB ⟶ FET_BUF_SIZE=32000

PREP/DEC	⟷	Small Sent - Medium Returned
OPEN/FETCH	⟷	Medium Sent - 32KB Returned
FETCH	2x ⟷	Small Sent - 32KB Returned
FETCH/CLOSE	⟷	Medium Sent - 32KB Returned
FREE	⟷	Small Sent - Small Returned

Total number of messages sent = 12

Small Select Statements Traffic Improvement

- Less than 32 KB of data returned by the select
- No blob data and rows smaller than 4KB

PREP/DEC	⟷	Small Sent - Medium Returned
OPEN	⟷	Small Sent - Medium Returned
FETCH	⟷	Small Sent - Medium Returned
CLOSE	⟷	Small Sent - Small Returned

Total number of messages sent = 8

- OPTOFC enabled — OPTOFC=1
- Fetch buffer set at 32KB — FET_BUF_SIZE=32000
- Defer Prepare enabled — IFX_DEFERRED_PREPARE=1

PREP/DEC/OPEN/FETCH/CLOSE Medium Sent - 32KB Returned

Total number of messages sent = 2

Are My Application Using OPTOFC?

- onstat –g ses 145

```
session           effective                                  #RSAM    total      used       dynamic
id       user     user      tty      pid      hostname       threads  memory     memory     explain
145      informix -         20       30096    talo           1        274432     270896     off

tid      name     rstcb             flags    curstk   status
232      sqlexec  44d28d78          Y--P---  12272    cond wait  cmd_cond  -

Memory pools    count 2
name         class addr              totalsize  freesize   #allocfrag #freefrag
145          V     46853040          270336     2728       259        6

name            free       used           name            free       used
overhead        0          6576           resident        0          72
scb             0          144            opentable       0          8656
filetable       0          1344           ru              0          600
misc            0          1088           log             0          16536

sqscb info
scb              sqscb            optofc   pdqpriority sqlstats optcompind  directives
45ce0028         4620c028         0        0           0        2            1

Sess       SQL             Current            Iso Lock       SQL  ISAM F.E.
Id         Stmt type       Database           Lvl Mode       ERR  ERR  Vers  Explain
145        EXEC PROCEDURE  sysadmin           CR  Not Wait    0    0    9.24  Off
```

Questions?