SQLTRACE—an in-depth view

Vijay Lolabattu
IBM

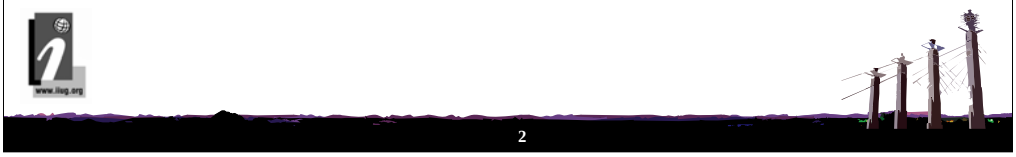
Session: D09
Tuesday, April 29, 13:00-14:00

2008 IIUG Informix Conference

www.iiug.org

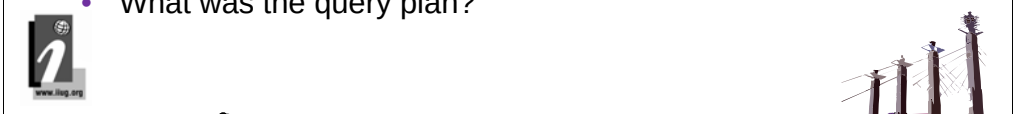
Databases and performance

- Well performing database server behind every successful application.
- Why databases underperform?
- Performance tuning a daunting task!
- Lack of having proper tools.
- Those expensive hardware changes may not be necessary!



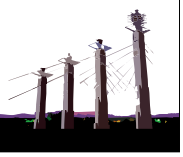
Common questions DBAs have

- How long do SQL statements take?
- How many resources are individual statements using?
- How long did statement execution take?
- How much time was involved in waiting for each resource?
- What was the query plan?



What is SQLTRACE?

- Prior to IDS 11:
  1. Set EXPLAIN
  2. I-SPY
- New feature introduced in IDS 11
  1. Simple
  2. Comprehensive
  3. Flexible
  4. In memory and realtime



Syntax

- Using $ONCONFIG

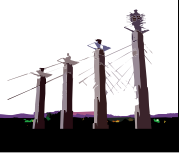
  SQLTRACE [Level=off|low|med|high],[Ntraces=number of traces],[Size=size of each trace buffer],[Mode=global|user]

- Using Administrative API

  admin(), task()

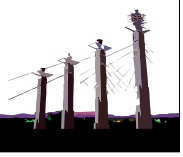
Syntax continued…

- **Level**: Determines the amount of information traced:
  - Off: No SQL tracing.
  - Low: statement statistics, statement text, and statement iterators.
  - Medium: all of Low, plus table names, the database name, and stored procedure stacks.
  - High: all of Medium plus host variables.



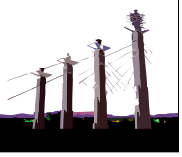
Syntax continued..

- Ntraces: Number of SQL statements to trace before reusing the resources. The range is from 500 to 2147483647.
- Size: Number of kilobytes for the size of the trace buffer. If data exceeded buffer size, text portion is truncated. The range is 1 KB to 100 KB.



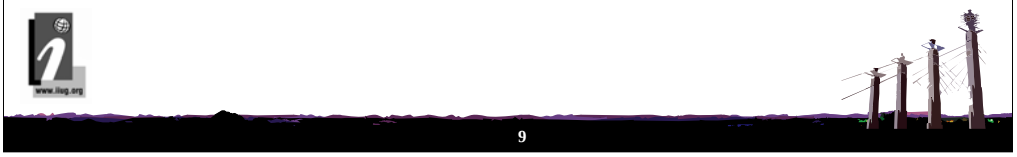
Syntax continued..

- Mode: Specifies the type of tracing performed:
  - Global: This is for all users on the system.
  - User: Use this to enable tracing at user or session level using API admin()/task()



Online.log

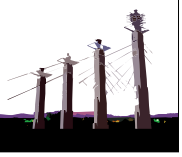
20:36:52 Event alarms enabled. ALARMPROG =
'/usr/informix/etc/alarmprogram.sh'
20:36:52 Booting Language <c> from module <>
20:36:52 Loading Module <CNULL>
20:36:52 Booting Language <builtin> from module <>
20:36:52 Loading Module <BUILTINNULL>
20:36:52 Dynamically allocated new virtual shared memory segment (size 8192KB)
20:36:52 Memory sizes:resident:12288 KB, virtual:16384 KB, no SHMTOTAL limit
20:36:52 SQLTRACE: SQL History Tracing set to 2000 SQL statements.
.....



Default

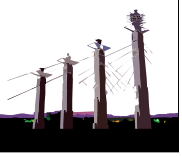
- Ntraces=1000
- Size=1kb
- Mode=low

```
informix@ramsay[144] dbaccess sysadmin -
Database selected.
> execute function task("set sql tracing on");
(expression) Global Tracing ON Number of Traces 1000 Trace Size 1000
    Mode Low
```



SQL trace using APIs task() and admin()

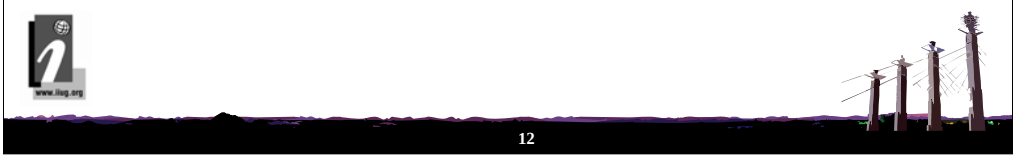
- Changes to SQLTRACE in $ONCONFIG require server restart
- Changes made by task() and admin() are dynamic
- User and Session level tracing



Enabling tracing using API task()

```
informix@ramsay[144] dbaccess sysadmin -
Database selected.
> execute function task("set sql tracing on");
(expression) Global Tracing ON Number of Traces
   1000 Trace Size 1000 Mode Low
```

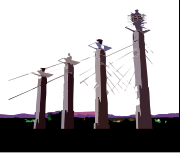
NOTE: Tracing is Global with default values



Disabling tracing using API task()

```
informix@ramsay[144] dbaccess sysadmin -
Database selected.
>execute function task('set sql tracing off');

(expression) SQL tracing off
```



Changing tracing parameters

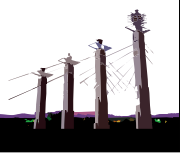
$ONCONFIG :

SQLTRACE level=HIGH,ntraces=1000,size=1,mode=user

execute function task("set sql tracing on", 3000, 2, "med", "global");
(expression) Global Tracing ON Number of Traces 3000 Trace Size 2024 Mode Med

- Original default values changed
  - Ntraces from 1000 to 3000
  - Size from 1k to 2k
  - Level from Low to Med
  - Mode kept as global

NOTE: Size parameter in KB but expressed as bytes



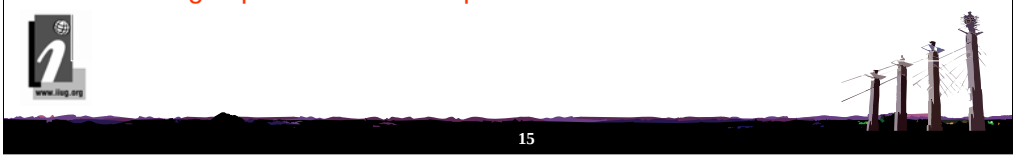
Changing mode: Be aware !

- $ONCONFIG:
  SQLTRACE level=HIGH,ntraces=1000,size=1,mode=user

execute function task("set sql tracing on")

(expression) User Tracing ON Number of Traces 1000 Trace Size 1000 Mode Low

NOTE: At this point tracing on at user level not Global level, no tracing captured until user specified

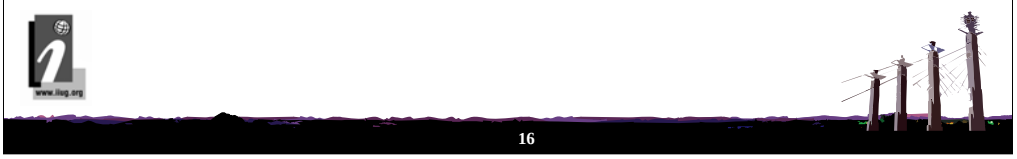


Tracing specific user(s)/session(s)

```
>execute function task("set sql user tracing on", 4);
    (expression)  SQL user tracing on for sid(4).

>execute function task("set sql tracing on", 1000, 1,"low","user");
>select task("set sql user tracing on", session_id)
    FROM sysmaster:syssessions
    WHERE username not in (“informix");
```

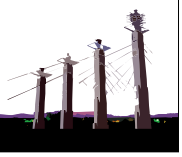
Note: Make note of 1st parameter in global vs user level tracing



Disabling user/session tracing

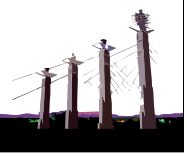
```
>execute function task("set sql user tracing off",17);
(expression) SQL user tracing off for sid(17).

>select task("set sql user tracing on", session_id)
    FROM sysmaster:syssessions
    WHERE username in ("vijayl");
```



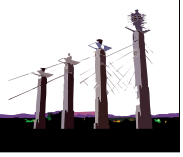
How to display traces?

- Onstat –g his
- Syssqltrace
- Syssqltrace_info
- Syssqltrace_iter



Onstat –g his

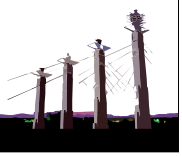
- New onstat option in IDS 11
- Prints entire trace together
- No single trace display option at this time
- Display depends on the trace mode in the setting (high, medium, low)



Onstat –g his output

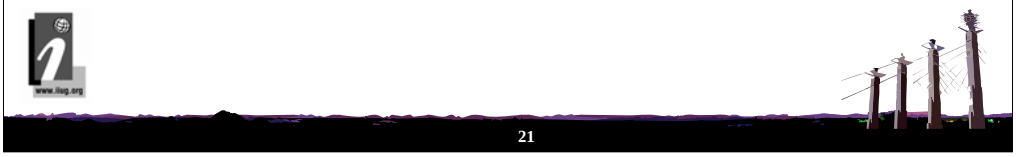
Divided into 3 logical categories for clarity

- Trace settings
- Statement text and iterators
- Statement statistics



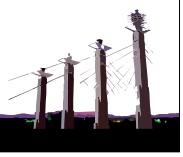
Trace settings

Trace Level	High
Trace Mode	Global
Number of traces	3000
Current Stmt ID	0
Trace Buffer size	2024
Duration of buffer	280 Seconds
Trace Flags	0x00007F11
Control Block	b272018
Statement # 977:	@ b2982d0



Statement text and iterators

Low:
Database: **0x100161**
Statement text:
insert into customer values(?,?,?,?,?,?,?,?,?,?)
Med:
Database: **stores_demo**
Statement text:
insert into customer values(?,?,?,?,?,?,?,?,?,?)
INSERT using tables [customer]
HIGH:
Database: stores_demo
Statement text:
insert into customer values(?,?,?,?,?,?,?,?,?,?)
INSERT using tables [customer]
Iterator/Explain
==========
ID Left Right Est Cost Est Rows Num Rows Type
1 0 0 1 1 1 Insert
Host Variables
==========
1 char
2 char



Statement statistics

Statement information:

Sess_id	User_id	Stmt Type	Finish Time	Run Time
19	200	INSERT	21:27:29	0.0019

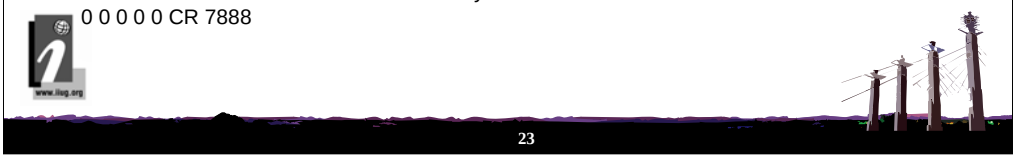
Statement Statistics:

Page Read	Buffer Read %	Read Cache	Buffer IDX Read	Page Write	Buffer Write %	Write Cache
0	90	100.00	0	0	90	100.00

Lock Requests	Lock Waits	LK Wait Time (S)	Log Space	Num Sorts	Disk Sorts	Memory Sorts
197	0	0.0000	9.79 KB	0	0	0

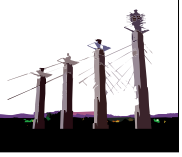
Total Executions	Total Time (S)	Avg Time (S)	Max Time (S)	Avg I/O Wait Time (S)	Avg Rows Per Sec
1	0.0019	0.0019	0.0019	0.000000	0.000000 539.1708

Estimated Cost	Actual Rows	SQL Rows Error	ISAM Error	Isolation Level	SQL Memory
0	0	0	0	CR	7888



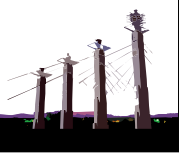
Syssqltrace

```
select * from syssqltrace where sql_sid = 25;
sql_id 3
sql_address 194531856
sql_sid 25
sql_uid 200
sql_stmttype 2
sql_stmtname SELECT
sql_finishtime 1180420443
sql_begintxtime 1180420263
sql_runtime 1.997457400000
sql_pgreads 0
sql_bfreads 1284
sql_rdcache 100.0000000000
sql_bfidxreads 0
sql_pgwrites 0
sql_bfwrites 0
sql_wrcache 0.00
sql_lockreq 636
sql_lockwaits 0
.....
```



This is a in memory pseudo table. Each row is exactly one SQL trace. Provides the flexibility to filter the information to display based on a single or multiple columns from the schema. Also you can display only the information that is pertaining to you or interests you, unlike onstat –his which dumps all the information together.

sql_sortdisk 0
sql_sortmem 0
sql_executions 1
sql_totaltime 3.544297100000
sql_avgtime 3.544297100000
sql_maxtime 1.997457400000
sql_numiowaits 0
sql_avgiowaits 0.00
sql_totaliowaits 0.00
sql_rowspersec 106.1349293357
sql_estcost 105
sql_estrows 211
sql_actualrows 212
sql_sqlerror 0
sql_isamerror 0
sql_isollevel 2
sql_sqlmemory 19248
sql_numiterators 3
sql_database <None>
sql_numtables 0
sql_tablelist None
sql_statement select l.tabname, r.created from sysmaster:systables l,
sysmaster@vj_acme_tcp:systables r where l.tabname = r.tabname



Syssqltrace_info

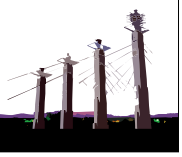
Column	Type	Description
flags	integer	SQL trace flags
ntraces	integer	Number of items to trace
tracesize	integer	Size of the text to store for each SQL trace item
duration	integer	Trace buffer (in seconds)
sqlseen	int8	Number of SQL items traced since start or resizing
starttime	integer	Time tracing was enabled
memoryused	int8	Number of bytes of memory used by SQL tracing

This is another pseudo table which stores the SQL trace profile information. The profile information

consists of the trace settings, such as ntraces, size, memory used, and trace starting time.

Syssqltrace_iter

Column	Type	Description
sql_id	int8	SQL execution ID
sql_address	int8	Address of the SQL statement block
sql_itr_address	int	Address of the iterator
sql_itr_id	int	Iterator ID
sql_itr_left	int	Iterator ID to the left
sql_itr_right	int	Iterator ID to the right
sql_itr_cost	int	Iterator cost
sql_itr_estrows	int	Iterator estimated rows
sql_itr_numrows	int	Iterator actual rows processed
sql_itr_type	int	Iterator type
sql_itr_misc	int	Iterator miscellaneous flags
sql_it_info	char(256)	Iterator miscellaneous flags displayed as text

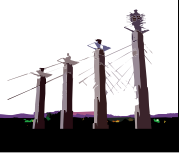


The *syssqltrace_iter* table is yet another pseudo table that sqltracing creates and uses for the tracing mechanism. The syssqltrace_iter tables stores the SQL query plan and iterators information that is also used by **onstat -g his** to display the iterators and explain portion of the onstat output. This table also allows you to run SQL statements to query the table data. This is extremely useful if you just want to know the iterator plan and explain information for a specific SQL, which you need to obtain by running a query using the sql_id on this table instead of using **onstat**, which prints all of the tracing information together.

Buffer overflow

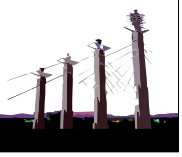
Statement # 847: @ b293190
WARNING: Data has been truncated, trace buffer not large enough.
Database:
Statement text:
create dba procedure informix.systdist (table_id int, column_no int)
returning int, datetime year to fraction (5), char(1),
smallfloat, smallfloat, float , stat, char(1);

- Adjust the size parameter accordingly



Sql tracing on distributed queries

- Tracing turned on only on Coordinator
  - Coordinator collects iterators, explain and statement statistics
  - Participant collects only iterators and explain information
- Tracing turned on both Coordinator and Participant
  - Both servers collect iterators, explain and statistics information.



Coordinator in distributed query

- Onstat –g his

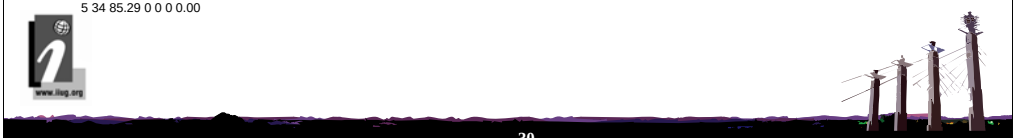
Statement # 226: @ bac1d70
Database: stores_demo

Statement text:
select l.customer_num, l.lname, l.company,
l.phone, r.call_dtime, r.call_descr
from customer l, stores_demo@vj_acme_tcp:cust_calls r
where l.customer_num = r.customer_num
SELECT using tables [customer cust_calls]

Iterator/Explain
==========
ID Left Right Est Cost Est Rows Num Rows Type
2 0 0 4 7 7 RemScan
3 0 0 1 28 1 Index Scan
1 2 3 10 7 7 Nested Join

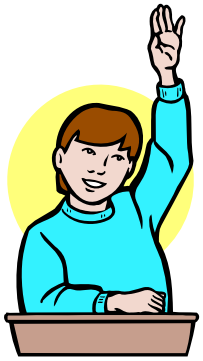
Statement information:
Sess_id User_id Stmt Type Finish Time Run Time
18 200 SELECT 21:27:05 0.1350

Statement Statistics:
Page Buffer Read Buffer Page Buffer Write
Read Read % Cache IDX Read Write Write % Cache
5 34 85.29 0 0 0 0.00



SQL Query Drill-Down from the IDSAdmin console

Questions



Session D09
SQLTRACE—an in-depth view

Vijay Lolabattu
IBM
lvijay@us.ibm.com