Large Databases in Practice

Tales from the land of the northern lights

Matti Kiviranta

Qvantel Business Solutions Oy

Shortly about the speaker

- Coming from freezing cold country of Finland
  - About 3900 miles from here
  - Today's weather 4-22°C (39-72°F)
- Progress OpenEdge user since 2007 (10.1A)
- Started as OpenEdge developer
- Now a Senior Database Administrator
- Works also with other database engines such as MySQL & MariaDB.

Qvantel

- Based in Finland
- Provides systems and services for mobile operators sales, product management, customer care and billing.
- Customers
  - Yoigo

Disclaimer

- Shown values and settings are from various production system.
- YMMV, these might not work in your environment.

Qvantel RBS product

- RBS (Rating and Billing System) is Qvantel's product suite that is running on top of OpenEdge.
- Product uses multiple databases in which clients take simultaneous connections.
- Processes millions of new records daily.

Shortly about environments

- Linux and Solaris environments
- Largest database ever was over 2 terabytes.
- Largest currently running database is 1464 gigabytes.
- Daily database growth in single environment is up to 8.9 gigabytes.
- Largest growth recorded has been over 10 gigabytes.

Proutil database –C dbanalysis

Table	Records	Size
PUB.D	8 590 845 958	1.9T

Total	1.9T	81.3 %	446.8G	18.7 %	2.3T	100%

- **Single** database, **single** table.

Examples of parameters we use

- -L 100000
- -B 9000000
- -B2 1000000
- -aiarcdir /aiarchive/xx
- -DBService replserv
- -pica 8192
- -pf ../general.pf

General.pf

- -indexrangesize 711
- -tablerangesize 288
- -aiarcdircreate
- -aiarcinterval 3600
- -Mm 8192
- -spin 100000
- -semsets 20
- -napmax 250
- -lruskips 100

Various buffers

Layer	Time	# of Recs	# of Ops	Cost per Op	Relative
Progress to –B	0.96	100,000	203,473	0.000005	1
-B to FS Cache	10.24	100,000	26,711	0.000383	75
FS Cache to SAN	5.93	100,000	26,711	0.000222	45
-B to SAN Cache*	11.17	100,000	26,711	0.000605	120
SAN Cache to Disk	200.35	100,000	26,711	0.007500	1500
-B to Disk	211.52	100,000	26,711	0.007919	1585

Know your environment

- Know your operating systems limitations
- Know your environments limits
- Know required tune-ups for your system
- Collect history information about the system

History!?

01/28@05:43 BACKUP114: (1362) Full backup started.
01/28@07:54 BACKUP114: (13625) Wrote a total of 508954 backup blocks using 132.0 GBytes of media.
01/28@07:54 BACKUP114: (1364) Full backup successfully completed.

Total time 2h 11min

02/08@05:36 BACKUP 57: (1362) Full backup started.
02/08@07:30 BACKUP 57: (13625) Wrote a total of 520831 backup blocks using 135.1 GBytes of media.
02/08@07:30 BACKUP 57: (1364) Full backup successfully completed.

Total time 1h 54min

02/18@16:13 BACKUP 57: (1362) Full backup started.
02/18@23:14 BACKUP 57: (13625) Wrote a total of 520831 backup blocks using 135.1 GBytes of media.
02/18@23:14 BACKUP 57: (1364) Full backup successfully completed.

Total time 7h 1min !

Checkpoints

```
Ckpt                                  ------ Database Writes ------
No. Time        Len   Freq   Dirty   CPT Q    Scan   APW Q Flushes   Duration  Sync Time

3104 08:51:33    283      0  195347   84056      40     272       0       0.69       0.06

3103 08:38:51    726    762  195288  203066    3508     647       0       0.67       0.04
3102 08:25:28    765    803  193021  200814    3381     557       0       0.66       0.04
```

Promon after debug command

```
Ckpt                                  ------ Database Writes ------
No. Time        Len   Freq   Dirty   CPT Q    Scan   APW Q Flushes   Duration  Sync Time  ClFil%  FuzzTm NumChkpt

3104 08:51:33    291      0  195347   86369      40     282       0       0.69       0.06    37     291    86616

3103 08:38:51    726    762  195288  203066    3508     647       0       0.67       0.04   100     726   203454
3102 08:25:28    765    803  193021  200814    3381     557       0       0.66       0.04   100     765   201165
```

VST tables

- *_actsummary*
  - Single day values (all databases):
  - Db read: 15 424 789 709
  - Records read: 7 061 389 560

- *_tablestat* & *_indexstat*
- *_usertablestat* & *_userindexstat*

Have you updated?

- Hardware
- Operating system
- OpenEdge

REMEMBER TO TEST BEFORE YOU CHANGE PRODUCTION!

Something completely different

TZ, Why does it matter?

- Environment variable TZ, Time zone
- If left unset, will cause extra system calls to see what is the local time zone.
- Important especially on Linux due to glibc futex and signal handling.
- How to set it?

```
vi /etc/profile.d/tz.sh
# Set TZ for Helsinki, Finland (EET/EEST)
export TZ=“Europe/Helsinki”
```

```
PAUSE.
TODAY.
PAUSE.
```

NO TZ

- poll([{fd=0, events=POLLIN|POLLPRI}], 1, -1) = 1 ([{fd=0, revents=POLLIN}])
- **stat("/etc/localtime", {st_mode=S_IFREG|0644, st_size=1892, ...})**
- ioctl(0, TCFLSH, 0)
- write(1, "\33[24;1HPress space bar to continue.", 35)

TZ set

- poll([{fd=0, events=POLLIN|POLLPRI}], 1, -1) = 1 ([{fd=0, revents=POLLIN}])
- ioctl(0, TCFLSH, 0)
- write(1, "\33[24;1HPress space bar to continue.", 35)

Case X

- MariaDB backend
- Web application
- Database connections go through a proxy for monitoring purposes.

Issue on hand:

- Queues starting to grow to application servers regardless of application in question.
- Page/response generation times start to increase.

Case X continued

- Number of database connections increase as more applications are active.
- Database response times are still reasonable when checked manually.
- Queries per minute does stays about same during this time. Higher number of queries have been processed just fine.

So where is the problem?

Looking for the issue

write(23, "{\0\0\0\3SELECT .. FROM `tablex` .. ", 127) = 127 <0.000068>

read(23, "\1..\0", 16384) = 56 <**0.005376**>

write(23, "\1\0\0\0\16", 5) = 5 <0.000065>

read(23, "\7\0\0\1\0\0\0\1\0\0\0", 16384) = 11 <**0.004470**>

write(24, "{\0\0\0\3SELECT .. FROM `tablex` .. ", 127) = 127 <0.000025>

read(24, "\1..\0", 16384) = 56 <**0.023255**>

write(24, "\1\0\0\0\16", 5) = 5 <0.000045>

read(24, "\7\0\0\1\0\0\0\1\0\0\0", 16384) = 11 <**0.020023**>

Fixing the issue

- Application creates connections to the database per every page load.
- As the number of connections was high enough (netstat –nt|wc –l gave over 6000) proxy was using most of its time in TCP handling.
- Luckily this could be corrected by adding a extra proxy to another port for load balancing.

How did it look after

write(23, "{\0\0\0\3SELECT .. FROM `tablex` .. ", 127) = 127 <0.000027>

read(23, "\1..\0", 16384) = 56 <**0.000450**>

write(23, "\1\0\0\0\16", 5) = 5 <0.000029>

read(23, "\7\0\0\1\0\0\0\1\0\0\0", 16384) = 11 <**0.000169**>

- Improvement from **0.020023** to **0.000169** seconds**.**

Kysymyksiä?

QUESTIONS?

Kiitos
Thank you!

Matti Kiviranta
matti.kiviranta@qvantel.com