Progresswiz

27 Days to 27 Hours

Best Practices in Dump & Load Optimization

PAUL KOUFALIS
PRESIDENT
PROGRESSWIZ CONSULTING

27 Days to 27 Hours !?!

- What does that mean?
- At one client, the resident DBA/consultant told them it would take ***27 days*** to D&L their PROD environment
  - OFFLINE!!
  - Stop laughing! It's true!
- We did it in 27 hours.

Progresswiz

In 100 Slides or Less…

- You will learn:
  - Some basics
  - The time/cost trade-off: planning versus execution
  - Sneaky tricks à la *Why didn't I think of that !?!*
- Bonus:
  - Some cool benchmark results from the new 10.2B06 Index Rebuild parameters

What You Won't Learn

- Basic syntax – I will assume you can RTFM
- Scripting
- Windows vs. UNIX: it's all the same to me
- RAID 0, 1, 5, 6, 10, 50, 60…

- Feel free to ask me during Q&A at the end

Progresswiz

Paul Koufalis? Who is Paul Koufalis?

- Famous quote from my junior college days
  - Buy me a beer and I'll tell you the story
- Progress DBA and UNIX sysadmin for 18 years
  - I have clients with DBs of all sizes
  - Some have one prod DB, others 50
  - Each has its unique set of D&L challenges

Progresswiz

Let's Get Started

- Two kinds of dumps:
  - ASCII (via Data Admin or custom EXPORT)
  - Binary (via _proutil –C dump)
- Three kinds of loads:
  - ASCII (via Data Admin or custom EXPORT)
  - Bulk (via _proutil –C bulkload)
  - Binary (via _proutil –C load)
- Bonus
  - BUFFER-COPY() method

Progresswiz

ASCII Dump

- Dictionary or ABL EXPORT command
  - Really the same thing
- Dictionary dump uses the primary index
- Process
  - Load index block(s)
  - Locate ROWID of next row
  - Load data block
  - Copy record to program’s record buffer
  - Export data in dump-file format

Progresswiz

ASCII Dump – PROs

- Easy: Admin – Dump – Data Contents
- Overall faster for small databases
- Can write complex ABL dump code
  - Including data transformations

- Still always used for _user.d, _seqvals.d

ASCII Dump – CONs

- Slooooowwwwww for big-ish tables
- Uses primary index only (_admin.p)
- Custom ABL EXPORT could lead to errors
  - Skipped and/or mangled data
  - Codepage
  - Year-offset
  - …

Progresswiz

Binary Dump

- Export entire record in same format as stored in database
- No conversion
- NOT readable text

Binary Dump - PROs

- Fast!
- Multi-threaded in later version
- Can select index
- DumpSpecified allows limited WHERE clause

Binary Dump - CONs

- Requires scripting
  - One _proutil –C dump command per table
- Need to test and prep more than ASCII dump
  - Your scripts…
  - Load on your server
- Needs more comprehensive error checking
  - Grep through logs

ASCII Load

- Dictionary or ABL IMPORT command
- Read the file one line at a time:
  - REPEAT:
    - CREATE <table>.
    - IMPORT <record>.
  - END.
- Create index entries for every record creation

ASCII Load – PROs

- Easy: Admin – Load – Data Contents
- Can write complex ABL load code
- You see errors right away
- Still always used for _user.d, _seqvals.d

ASCII Load – CONs

- Slooooowwwwww
  - Create table…create indexes…create table…
- Same as dump: mostly just slow and painful

Binary Load

- Import record directly from binary dump file to db block
- Little manipulation needed

Binary Load - PROs

- Fast!
  - Imports entire record in one shot!
- CRC-check makes sure target schema is identical to source schema
  - Watch out for R-POS
- Your choice: build indexes inline with load or afterwards

Binary Load - CONs

- Same as dump
  - Requires scripting
    - Each bin-dump file must be loaded
  - Need to test and prep more
  - Needs more comprehensive error checking
- Can accidentally load the same file twice
  - Will find out during index rebuild!

Bulk Load

- This is an ASCII load without building indexes
- I rarely use this

- Could be useful if you did a custom dump
- Bulkload + idxbuild faster than dict load

BUFFER-COPY()

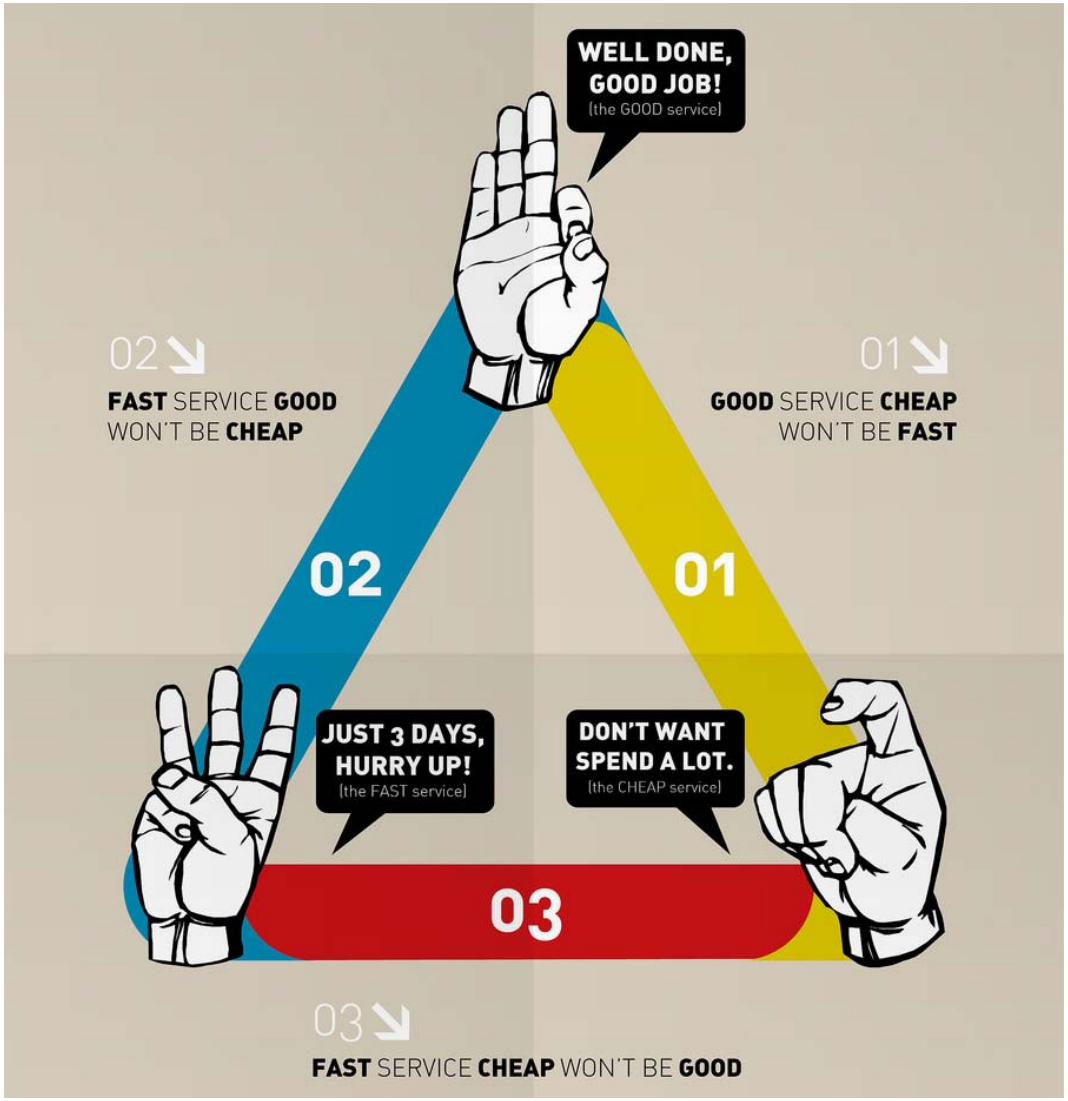
- ABL code method

- Connect the source and target DB
- FIND source.table.
- CREATE target.table
- BUFFER-COPY source.table TO target.table.

BUFFER-COPY()

- Conceptually very cool
  - One or more _progres session per table
  - Only one small index active on target
    - Rebuild the rest later
- I have not had the opportunity to try it
- Certain respected DBAs like this method

Planning Your Dump & Load



Planning Your Dump & Load

- I like to emphasize FAST and GOOD
  - But this means it WON'T be CHEAP
- More time (money) preparing = less surprises
- In your world, you need to decide

Define Your Limits

- How much time to you have?
  - Don’t forget UAT time
- Example:
  - Down Friday 6pm
  - Finished by Sunday noon
  - UAT until Sunday 4pm
  - Go/No-Go Sunday 4:30pm
  - Up Sunday 8pm

Progresswiz

Define Your Limits

- How much disk space do you have?
  - Dump files
  - Temp files (index rebuild)
  - New databases
  - Backups
    - Old DB
    - New DB at multiple stages in process
- Fast disks? SSD? FC-SAN? Or NFS?
- Can you *borrow* fast disk space?

Progresswiz

Define Your Limits

- How many servers can you use?
  - Can you dump on one and load on another?
  - Is it worth it?
- How much RAM do they have? CPU?
  - Can you use it ALL? (be greedy)
  - Can you *borrow* more? (VM/LPAR)

Preparing the DUMP

- Use Table Analysis to tell you approximately how big each dump will be
- Note the big tables
  - Often 5-10 tables = 90% of data

Baseline DUMP

- Baseline
  - Start the DB with the biggest –B you can
  - binary dump everything in parallel
    - _proutil *db* –C dump *table* &
- By default uses the primary index
- Typically ALL the tables except a handful will finish dumping in a very short period

Baseline DUMP

- Server is freaking out for the first 10-20 min
  - 3-4-500 parallel dumps running!
  - Don’t stress
- If it’s taking forever – abort
  - Extrapolate dump time for remaining tables
- If it’s fast enough for you – you’re done
  - Good, fast AND cheap! Lucky you!

Progresswiz

DUMP #2

- You have identified the x slooooow dumps
  - Surprise! Some of the slow dumps may not be your biggest tables!
- Make a few changes:
  - Dump the slow tables in parallel
  - Dump the fast/small tables in series
  - Will take some of the stress off the server
  - Won't affect overall dump time

Progresswiz

DUMP #2

- What to do with the slow tables?
- Option 1: Use “-index 0”
  - No index. Just dump in physical order in DB
    - MOST EFFICIENT
  - Requires Type II S.A. and 10.2B0x
    - Watch out for bugs in earlier versions
  - Won’t fix logical scatter

Progresswiz

Dump #2

- Option 2: Pick a small index
  - Use Index Analysis
  - Good when no Type II S.A. or –index 0 not supported
- Option 3: Pick most-used index
  - Use IndexStat VST
  - May not be fastest dump but will be fastest access in target

Progresswiz

Dump #2

- So? Better?
- What did we forget?
  - Oh yeah…multi-threaded binary dump!

Dump #3

- Small/fast tables dump
  - Single-threaded
  - In series
  - Default (primary) index
- Large/slow tables
  - Multi-threaded
  - In parallel
  - -index 0 or other specific index

Progresswiz

Sneaky Dump Tricks

- Still not fast enough !?! Here's some ideas:

- Multi-threaded dump
  - May not be very multi-threaded
    - Index limitations
  - Use multiple DUMPSPECIFIED

Sneaky Dump Tricks

- No fast index?
  - Add an index
  - Add + idxbuild + dump may still be faster than dump with slow index
- Do you have a DEV / TEST / DR box?
  - Dump from multiple servers
    - Use last backup + AI files to prepare

Sneaky Dump Tricks

- Convert Your V9 DB to OE 10.2B06
  - proutil db –C conv910
- Now you can use all the cool toys in OE 10
- Don’t forget to backup your source DB!

Preparing the Load

- Hopefully will take advantage of Type II S.A.
- Prepare your empty DB (DF only) in advance
  - Take a backup for easy restart
- DON’T throw away that test DB
  - It’s already pre-grown to the right size!
  - Just prorest DF_ONLY.probkp into it
- Create a /loaded dir
  - Move loaded .bd files after successful load

Load #1

- Single user, single threaded, one after the other
- So many people disagree, but this is sooo often the fastest way

- It is also usually fast enough
  - See “Sneaky Load Tricks”

Load #2 ???

- What load #2 ??? #1 is the BOMB!
- OK…try massively parallel loads
  - BIG –B
  - _mprosrv with “-i”
  - Binload’s with “-i”
  - -bibufs 200-300-400 (low cost)
  - HUGE BI cluster size + 16 Kb BI block size
  - Two APW + BIW

Progresswiz

Sneaky Load Tricks

- My favourite: ulimit during dump
  - Set it to x Gb (some fraction of the total size)
  - As soon as bindump switches to .bd(n+1), start loading .bd(n)
  - BROKEN IN 10.2B0x ??? (Breaking news)

Progresswiz

Sneaky Load Tricks

- Use your DR or some other server to load
  - You can start loading within minutes of starting the dump
- Use shared drive for .bd* files
  - Disk native to dump server
  - NFS mount to load servers
- Especially beneficial when many DBs

Progresswiz

Sneaky Load Tricks

- Use -i (no-integrity)
  - ATM load tests:
    - 1h36 (no -i)
    - 0h07 (with -i)
- CAREFUL: If it crashes you need to restart from empty

Sneaky Load Tricks

- Backup your target(s) at key points
- So many people ignore this
  - OK to ignore if time is not an issue
  - Or if you can try again next week

Question Marks

- LOBs
  - I still have to benchmark single-user versus multi-user
  - THIS may be faster in multi-user

Index Rebuild

- ONE important point:

MEMORY = SPEED

- The more you do in RAM, the faster it will be

Index Rebuild – PRE 10.2B06

- Use RAM disks
- Beg/borrow/steal as much RAM as you can
- Use srt file and –t to monitor

Index Rebuild – PRE 10.2B06

- SRT File:
  - Progress will stripe tmp files across all the dirs in the SRT

- This is NOT optimal:

```
10000000 /ramdisk/
0 /scratch_on_disk
```

- Will use half-half

Index Rebuild – PRE 10.2B06

- This is better:

```
1000000 /ramdisk/D1
1000000 /ramdisk/D2
...
1000000 /ramdisk/D10
0 /scratch_on_disk
```

Index Rebuild – 10.2B06

- Lots of new toys (Thanks Rich!)
- Mostly moves bottlenecks from OpenEdge to hardware

- Benchmarks showing MASSIVE reductions in time

Index Rebuild – Sneaky Tricks

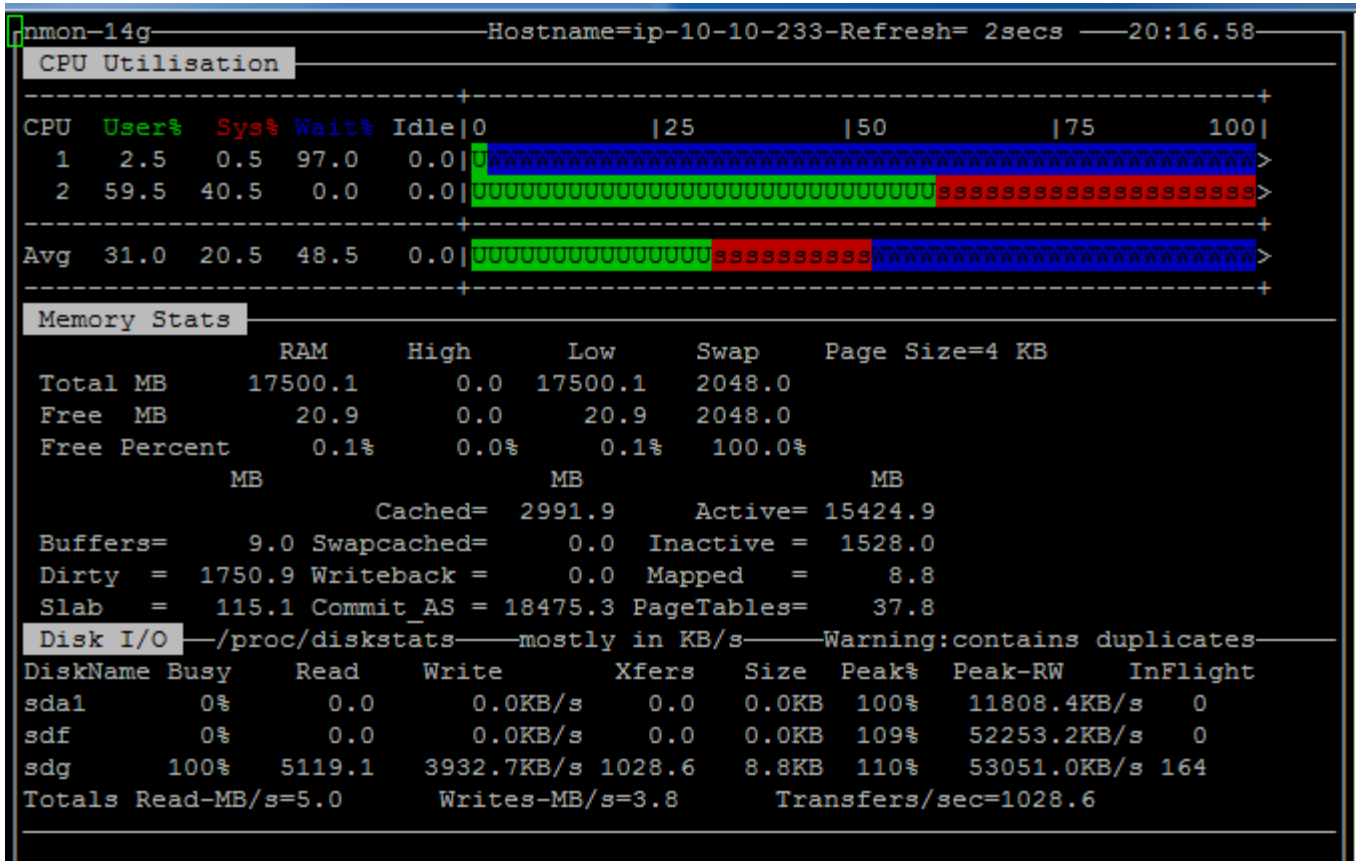
- Build indexes by AREA
  - If idxbuild crashes can restart with confidence
  - Can also control resource use better
- CAREFUL after idxbuild crash:
  - Delete the .xb
  - Do a dummy index rebuild
    - Ex.: a schema table
    - To clear errors

Progresswiz

Index Rebuild – Sneaky Tricks

- Use all the tools available to you
- See Idxbuild Results

Nmon is your friend



Progresswiz

Other Tips

- DB Analysis before and after
  - Compare rows in tables
  - Check that all the indexes have data
- Script anything you have to do more than a handful of times
  - Ex.: “grep Binary Load Complete” or “Binary Load Failed”

Other Tips

- Document your procedure in enough detail that your boss could run it
  - Include estimated times
- Create and print an XLS checklist
  - D&L steps vs. DBs
  - Check off boxes when completed successfully
  - Critical when running multiple tasks concurrently

Progresswiz

Other Tips

- Keep a close eye on virtual memory usage
  - You never want paging space activity
  - Do not confuse with FS cache paging
    - This is normal
- Careful how you interpret “I/O Wait”
  - Does not always imply problem

Credits

Progresswiz

Progresswiz Consulting

- Based in Montréal, Québec, Canada
- Providing technical consulting in Progress®, UNIX, Windows, MFG/PRO and more
- Specialized in performance tuning, system availability and business continuity planning
- …and security of Progress-based systems

Progresswiz

Questions?