

Rapid Upgrades With Pg_Upgrade

BRUCE MOMJIAN



January, 2012

Pg_Upgrade allows migration between major releases of Postgres without a data dump/reload. This presentation explains how pg_upgrade works.

Creative Commons Attribution License

<http://momjian.us/presentations>

Traditional Postgres Upgrade Options

- ▶ pg_dump (logical dump)/restore
- ▶ Slony

Why Upgrading Postgres Is Complex

- ▶ New features often require system table changes
- ▶ However, the internal data format rarely changes

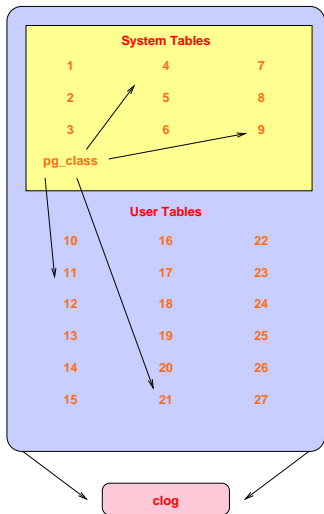
Why Pg_Upgrade

- ▶ Very fast upgrades
- ▶ Optionally no additional disk space

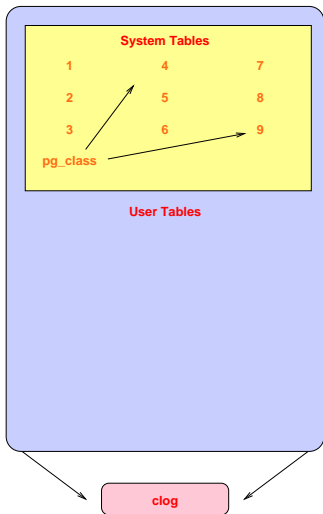
pg_upgrade installs new system tables while using data files from the previous Postgres version.

How It Works: Initial Setup

Old Cluster

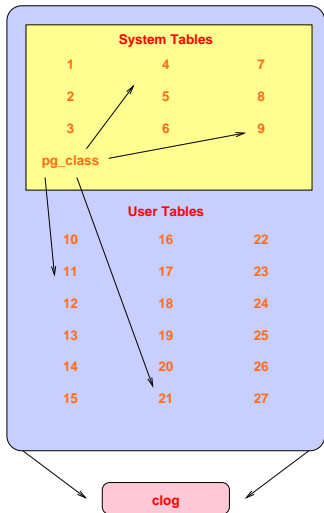


New Cluster

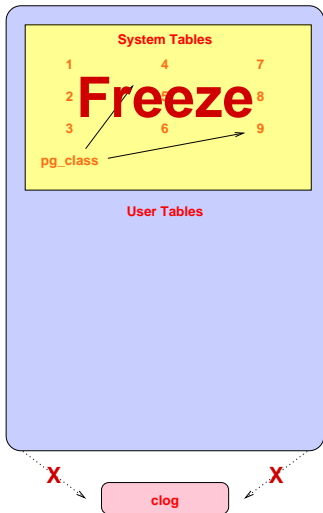


Decouple New Clog Via Freezing

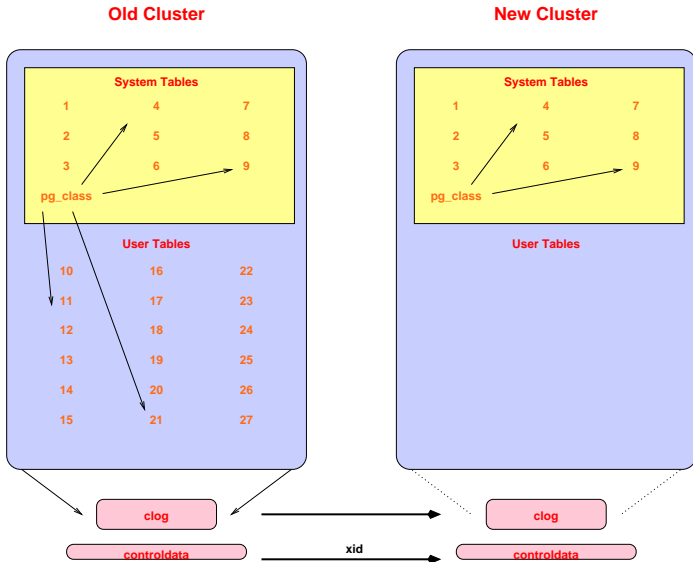
Old Cluster



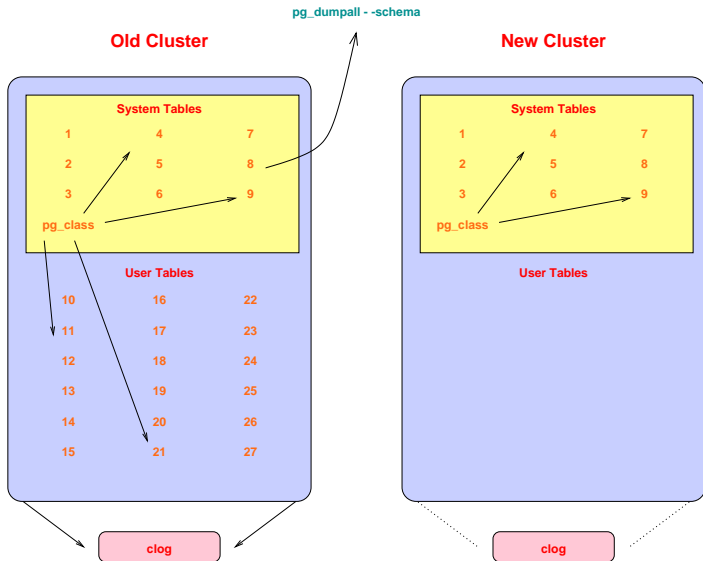
New Cluster



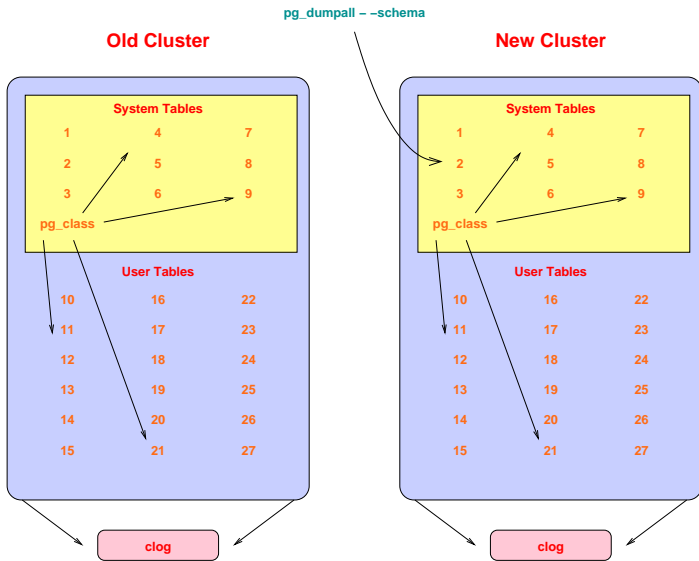
Transfer Clog and XID



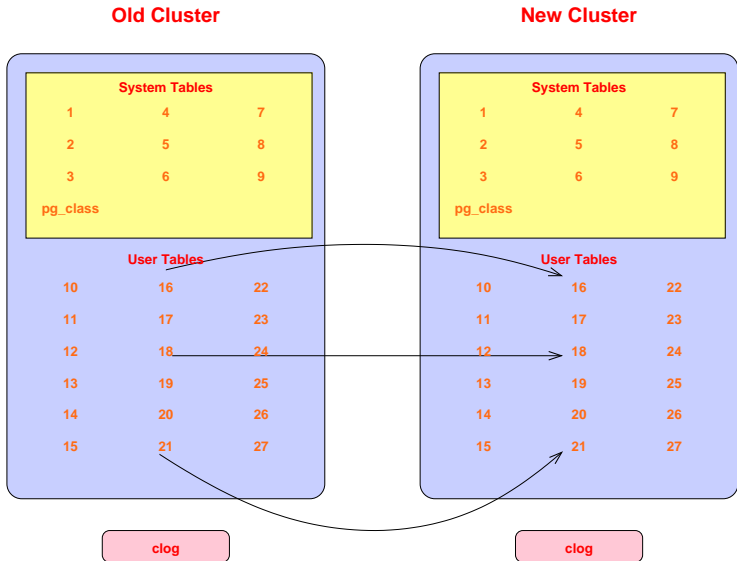
Get Schema Dump



Restore Schema In New Cluster

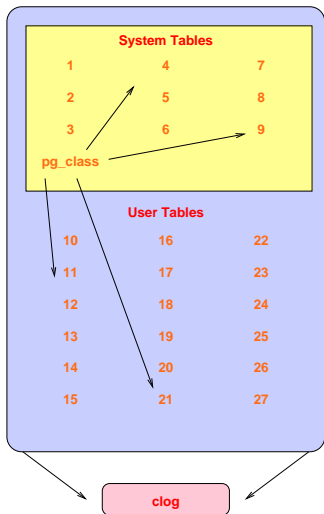


Copy User Heap/Index Files

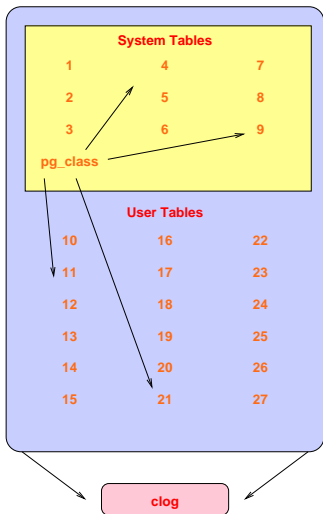


Complete

Old Cluster



New Cluster



How It Works: In Detail

- ▶ Check for cluster compatibility
 - ▶ locale
 - ▶ encoding
- ▶ Use `pg_dumpall` to dump old cluster schema (no data)
- ▶ Freeze all new cluster rows (remove reference to clog entries)
- ▶ New cluster uses old xid counter value (see freeze above)
 - ▶ Set system table frozen xids to match the current xid
- ▶ Create new users/databases
- ▶ Collect cluster information
- ▶ Install support functions that call internal backend functions
- ▶ Create schema in new cluster
- ▶ Copy or link files from old cluster to new cluster
- ▶ Warn about any remaining issues, like REINDEX requirements

Sample Run: Performing Consistency Checks

Performing Consistency Checks

```
Checking old data directory (/u/pgsql.old/data)      ok
Checking old bin directory (/u/pgsql.old/bin)        ok
Checking new data directory (/u/pgsql/data)         ok
Checking new bin directory (/u/pgsql/bin)           ok
Checking for /contrib/isn with bigint-passing mismatch ok
Checking for large objects                          ok
Creating catalog dump                               ok
Checking for presence of required libraries         ok
```

```
| If pg_upgrade fails after this point, you must
| re-initdb the new cluster before continuing.
| You will also need to remove the ".old" suffix
| from /u/pgsql.old/data/global/pg_control.old.
```

Sample Run: Performing Migration

Performing Upgrade

| | |
|---|----|
| Adding ".old" suffix to old global/pg_control | ok |
| Analyzing all rows in the new cluster | ok |
| Freezing all rows on the new cluster | ok |
| Deleting new commit clogs | ok |
| Copying old commit clogs to new server | ok |
| Setting next transaction id for new cluster | ok |
| Resetting WAL archives | ok |
| Setting frozenxid counters in new cluster | ok |
| Creating databases in the new cluster | ok |
| Adding support functions to new cluster | ok |
| Restoring database schema to new cluster | ok |
| Removing support functions from new cluster | ok |
| Restoring user relation files | ok |
| | |
| Setting next oid for new cluster | ok |
| Creating script to delete old cluster | ok |
| Checking for large objects | ok |

Sample Run: Completion

Upgrade complete

```
-----  
| Optimizer statistics is not transferred by pg_upgrade  
| so consider running:  
|     vacuumdb --all --analyze-only  
| on the newly-upgraded cluster.  
  
| Running this script will delete the old cluster's data files:  
|     /u/postgres/pg_upgrade_output/delete_old_cluster.sh
```

Possible Data Format Changes

| Change | Conversion Method |
|-------------------------------------|-------------------------------------|
| clog | none |
| heap page header, including bitmask | convert to new page format on read |
| tuple header, including bitmask | convert to new page format on read |
| data value format | create old data type in new cluster |
| index page format | reindex, or recreate index methods |
| TOAST page format | convert to new page format on read |

Migration Timings

| Migration Method | Minutes |
|----------------------------|---------|
| dump/restore | 300.0 |
| dump with parallel restore | 180.0 |
| pg_upgrade in copy mode | 44.0 |
| pg_upgrade in link mode | 0.7 |

Database size: 150GB, 850 tables

The last duration is 44 *seconds*.

*Timings courtesy of
Stefan Kaltenbrunner
(mastermind on IRC)*

PostgreSQL Version Compatibility

| Old | New | | |
|------------|-------------|----------------|------------|
| | 8.3 | 8.4 | 9.0+ |
| 8.3 | pg_migrator | pg_migrator(1) | pg_upgrade |
| 8.4 | x | pg_migrator | pg_upgrade |
| 9.0+ | x | x | pg_upgrade |

1. Unable to migrate composites, arrays, and enums.

Conclusion



<http://momjian.us/presentations>