

Import/Export of Databases 12/30/2010

To dump (export) a psql db:

```
pg_dump dbname > dbname.out
```

To recreate (import) the db:

```
createdb -D PGDATA_LOCAL -U postgres -E SQL_ASCII dbname
```

```
psql -f dbname.out -U postgres dbname
```

The dump file (called dbname.out above) contains SQL statements for recreating the tables, triggers and functions and also contains the ascii dump of all records in the tables. This will be a VERY large file for a fully stocked IHFS db. The dump file generated by the hd_ob5rhax db at OHD was approx 125 Mbytes. According to a user group posting in July 2008 and FAQ 4.5, the size of the database on disk can be between 1.5 and 5 times the size of the dump file.

pg_dump can also be used to dump individual tables.

The “-D” option creates the database in the PGDATA_LOCAL partition. Note the absence of a \$ in front of the PGDATA_LOCAL partition name. This partition is available at all RFCs. The PGDATA_LOCAL partition is sized at 32 GBytes. Before databases can be created in this partition, the “initlocation” command must be run. See Section 18.5 entitled “Alternative Locations” for more information.

If a database is created without the “-D” option, it will be created in the PGDATA partition which is only .5 GBytes in size. If this partition fills up, the postgres engine will crash!

The entire export and import process using pg_dump took less than 10 minutes at OHD for the hd_ob5rhax db.

Note that if the "-U postgres" does not work, add the following line to the pg_hba.conf file:

```
local all all trust
```

and either bounce postgres or execute "pg_ctl reload". See Section 19.2 of the PostgreSQL Documentation for details on "trust authentication".

Dumping the Schema Only

To dump the schema of a database, use

```
pg_dump -s dbname > dbname.out
```

The following command generates the schema for the location table from the hd_ob7oun db and writes it to the file location.sql:

```
pg_dump -s -t location -f location.sql -d hd_ob7oun
```

Serial Column Values

When converting Informix database tables with serial columns, the user should reset the serial value internally using the setval function described in Section 9.11 of the PostgreSQL documentation. Failing to do this will result in the internal counter being set incorrectly for future inserts.

Upgrading to New Versions of postgres

When upgrading between major releases such as 8.2 to 8.3, a dump and restore of the database is required.

Generating a Table Schema

The following command generates the schema for the location table from the hd_ob7oun db and writes it to the file location .sql:

```
pg_dump -s -t location -f location.sql -d hd_ob7oun
```

Dumping Large Databases

For large databases, output from pg_dump can be

- redirected to a single file
- piped to gzip (to reduce the size of the output file)
- piped to split to split up the output into multiple files

For example,

```
pg_dump hd_ob92tar | gzip > hd_ob92tar.dump.gz
```

To restore,

```
gunzip hd_ob92tar.dump.gz | psql -f hd_ob92.dump -U postgres <db_name>
```

Dumping All Databases

The pg_dumpall command can be used to dump all databases in a postgres cluster. To dump all databases

```
pg_dumpall > dumpall.out
```

to restore databases dumped by the above command

```
psql -U postgres -f dumpall.out postgres
```

At OHD, a `pg_dumpall` is run weekly from the postgres cron on genessee.

Using `pg_dump` with different postgres Versions

`pg_dump` is designed specifically to be able to dump from all supported older versions so you can convert the data "forwards".

The key word in that sentence is "forwards". Dumping from an 8.3 database with 9.0 `pg_dump` will likely produce SQL that doesn't reload into an 8.3 server, only into 9.0.