Fun with C and PostgreSQL

Sailesh Krishnamurthy
sailesh+cs186@cs.berkeley.edu

January 31, 2003

Overview

① "C" Review
② Installing Postgres
③ Running Postgres
④ Debugging Postgres

Overview

① **"C" Review**
② Installing Postgres
③ Running Postgres
④ Debugging Postgres

C Trivia

Q. What does the following code do?

```
MASK_FREE = 2;
flags |= MASK_FREE;
```

A. Sets second bit of `flags`.

C Trivia

Q. How do you clear the second bit?

A.

```
flags &= ~MASK_FREE;
```


C Trivia

Q. What is wrong with the following code?

```
typedef struct node_t {
   int *data;
   struct node_t *next;
} node_t;
node_t *p;
...
/* p != NULL */
do {
   p = p->next;
   if (p->data == NULL)
      continue;
} while (*p->data < 5);
```

C Trivia

Q. What is wrong with the following code?

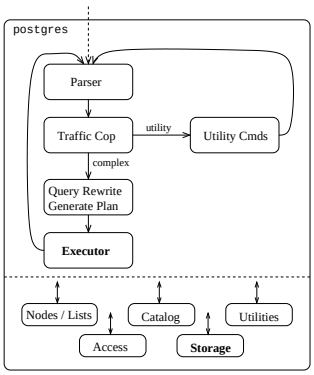
```
typedef struct node_t {
   int *data;
   struct node_t *next;
} node_t;
node_t *p;
...
/* p != NULL */
do {
   p = p->next;
   if (p == NULL)
      break;
   if (p->data == NULL)
      continue;
} while (*p->data < 5);
```


C Trivia

A. Careful about `continue` semantics: termination condition is evaluated. Use `goto` if you *absolutely* have to.

```
typedef struct node_t {
   int *data;
   struct node_t *next;
} node_t;
node_t *p;
...
/* p != NULL */
do {
   p = p->next;
   if (p == NULL)
      break;
} while (p->data == NULL || *p->data < 5);
```


Preview



Assignments:

1. Buffer manager (storage)
2. Query executor

Overview

① "C" Review
② **Installing Postgres**
③ Running Postgres
④ Debugging Postgres

Source Distribution

- Unpack source tarball:

```
$ cp ~cs186/sp03/Hw1/hw1_pkg_tar.gz .
$ gtar -zxvf hw1_pkg.tar.gz
```

- You should see three subdirectories:
  - `postgresql-7.2.2/` Original source code
  - `src/` Our implementation of MRU
  - `exec/` Some scripts to ease your pain
- Compile, install with `postgresql-7.2.2/compile.sh`

Overview

① "C" Review
② Installing Postgres
③ **Running Postgres**
④ Debugging Postgres

Basic Initialization

- Environment variables

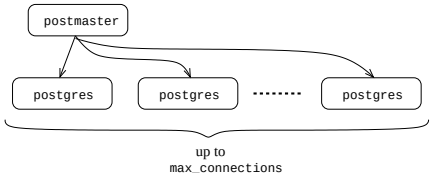
```
$ export PGPORT=nnnnn          # Pre-assigned
$ export PGDATA=$HOME/pgdata
```

- Database cluster, etc. . .

```
$ initdb
$ pg_ctl start -l $PGDATA/serverlog -o "-i"
$ createdb test
$ pg_ctl stop
```


Postmaster

- Manages multiple client connections
  - Starts one backend process (`postgres`) per client.
- Backends communicate via shared memory.
  - Initialized by `postmaster`.



Bare Backend (Postgres)

```
pentagon:~$ postgres test
DEBUG:  database system was shut down at 2003-01-23 14:37:33 ES
DEBUG:  checkpoint record is at 0/CF280C
DEBUG:  redo record is at 0/CF280C;
undo record is at 0/0; shutdown TRUE
DEBUG:  next transaction id: 656; next oid: 186847
DEBUG:  database system is ready

POSTGRES backend interactive interface
$Revision: 1.1.1.1.4.2 $ $Date: 2003/01/18 13:18:41 $

backend> select * from foo;
[...]
```


Basic Options

- Either as command line arguments to `postgres`. . .
- . . . or in `$PGDATA/postgresql.conf`.
- Buffer pool size
  - `pg_ctl start ... -o "-i -B 16"`
  - `shared_buffers = 16`
  - Must be at least 2×`max_connections`

Basic Options

- Optimizer—disable merge joins:
  - `pg_ctl start ... -o "-i -fm"`
  - `enable_mergejoin = false`
- Optimizer—disable hash joins:
  `-fh` or `enable_hashjoin = false`

Overview

Log Messages

- `elog(DEBUG, ...)`
- Similar to `printf`, but flushes output to the logfile
- Examining log output:

```
$ tail -f $PGDATA/serverlog
ERROR:  zero-length delimited identifier
DEBUG:  pq_recvbuf: unexpected EOF on client connection
DEBUG:  pq_recvbuf: unexpected EOF on client connection
DEBUG:  smart shutdown request
DEBUG:  shutting down
DEBUG:  database system is shut down
```


Log Messages

- You may want to define a preprocessor conditional

```
#define MY_DEBUG_MSGS  /* at top of file */

#ifdef MY_DEBUG_MSGS
elog(DEBUG, ...);
#endif
```

or possibly

```
#define MY_DEBUG_LEVEL 1   /* at top of file */

#if MY_DEBUG_LEVEL > 0
elog(DEBUG, ...);
#endif
```


Assert

- Catch bugs often, catch bugs early!
- `Assert(`*`condition`*`)`
- If condition is false, raises an (uncaught) `FailedAssertion` exception.

GDB / DDD

- Start backend in interactive mode from debugger
- DDD: graphical frontend for GDB
  - Graphical variable displays
  - Or use `print()` and `pprint()` (just for `Nodes`)

```
(gdb) set print elements 0  # no truncation
(gdb) call pprint(nodeptr)
```

BRIEF DEMO