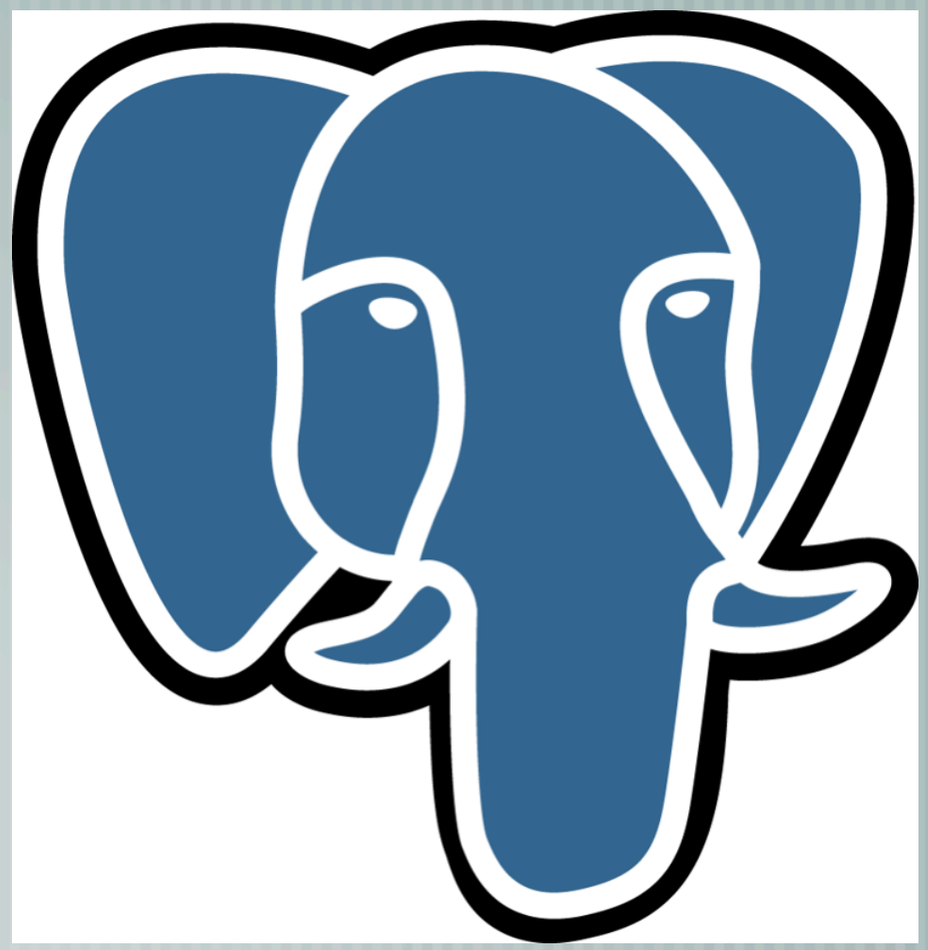
Postgres: A Babel of PLs

David Fetter http://fetter.org/
Copyright 2007
All Rights Reserved

Postgres



Procedural Languages

- PL/PgSQL
- PL/Perl(U)
- PL/Ruby(U)
- PL/LOLCODE(U?)
- PL/Parrot(U)?

The Trust Issue

- Untrusted
  - Can open pipes, filehandles
  - Superuser
- Trusted
  - Can't
  - Regular user

Perl

The Onion We Know and Love

Tools and Techniques

http://www.cpan.org

More Tools and Techniques

%__SHARED

Debugging Basics

```
warn
die
elog
```

More Advanced Debugging

```
use YAML;
...
warn Dump($complex_thing);
```

Debuggers

Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.

Kernighan

Ruby

A Nifty Gem

Tools and Techniques

http://rubygems.org/

A Custom Data Type

Bar Codes:

* UPC
* EAN
* GTIN
* Oy vey!

So you have to ask yourself a question...

How to do it

Luckily, they are all GTINs

And they have a checksum

How (not!) to do it in PL/Perl

```
CREATE OR REPLACE FUNCTION is_gtin(bigint) RETURNS
BOOLEAN
LANGUAGE plperl
AS $$
    my $i = 0;
    my $total = 0;
    for my $c (reverse split //, shift) {
        $i++;
        $total += $i % 2 ? $c : $c * 3;
    }
    return $total % 10 ? 'false' : 'true';
$$;
```

How to do it in PL/Ruby

```
CREATE OR REPLACE FUNCTION isa_gtin(bigint)
RETURNS BOOLEAN
LANGUAGE plruby
AS $$
    chars = args[0].to_s.split('').reverse
    total = 0
    chars.each_index {  | i |
      total += i % 2 == 0 ? chars[i].to_i : chars[i].to_i * 3
    }
    return !!(total % 10 == 0)
$$;
```

How to do it in SQL :)

```
CREATE OR REPLACE FUNCTION is_gtin(bigint)
RETURNS BOOLEAN
LANGUAGE sql
STRICT IMMUTABLE
AS $$
    SELECT ( sum(dgt) % 10 ) = 0
    FROM (
        SELECT substring($1 from idx for 1)::smallint AS dgt
        FROM   (SELECT generate_series(length($1), 1, -2) as idx) AS foo
        UNION ALL
        SELECT substring($1 from idx for 1)::smallint * 3 AS dgt
        FROM   (SELECT generate_series(length($1) -1, 1, -2) as idx) AS foo
    ) AS bar;
$$;
```

Creating The GTIN Data Type

```
CREATE DOMAIN gtin AS BIGINT
        CHECK ( is_gtin(VALUE) );
```

A More Complex Data Type

```
CREATE DOMAIN email AS TEXT
        CHECK ( is_email(VALUE) );
```

Defining A Function

```
CREATE OR REPLACE FUNCTION
is_email(text)
RETURNS BOOLEAN
LANGUAGE plperlu
AS $$

...
$$;
```

Perl Regex for Email

```
(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:(?:[^\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*")(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*"))*(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*@(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\]))*|(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*")(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*@(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*)?(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*")(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*"))*<(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:@(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*")(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*)*<(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:@(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*(?:,(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*)*:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*)?(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*")(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | "(?:[^\\\x80-\xff\n\015"] | \\[^\x80-\xff])*"))*(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*@(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])(?:(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\.(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*(?:[^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]+(?![^(\040)<>@,;:".\\\[\]\000-\037\x80-\xff]) | \[(?:[^\\\x80-\xff\n\015\[\]] | \\[^\x80-\xff])*\])*)*>(?:[\040\t] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff] | \((?:[^\\\x80-\xff\n\015()] | \\[^\x80-\xff])*\))*\)*
```

Better Perl (1/3)

use strict;

use warnings;

Better Perl (2/3)

```
use Email::Valid;
my $address = shift;
my $checks = {
    -address  => $address,
    -mxcheck  => 1,
    -tldcheck => 1,
    -rfc822   => 1,
};
```

Better Perl (3/3)

```
if (defined Email::Valid->address( %$checks )) {
    return 'true'
}
warn "address failed $Email::Valid::Details check.";
return 'false';
```

The Email Domain in Action

```
pg_pdx_fall_2007=> SELECT 'david@fetter.org'::email;
      email
________________
 david@fetter.org
(1 row)

pg_pdx_fall_2007=> SELECT 'david@fetter.con'::email;
ERROR:  value for domain email violates check constraint "email_check"
```

Perl on Both Sides

```
#!/usr/bin/env perl

use strict;
use warnings;

use DBI;
my $dbh = DBI->connect('dbi:Pg:dbname=wow','user','secret',
    {RaiseError => 1, AutoCommit => 0});

# And go on as usual
```

A Usual Program (1/3)

```
use Proc::ProcessTable;
my $t = new Proc::ProcessTable;
my $p = shift ( @{ $t->table });
my @fields = ();
```

Still Usual (2/3)

```
foreach my $f ($t->fields) {
    if ($p->{$f} =~ m/-?\d*\.?\d+/) {
        push @fields, "$f FLOAT8";
    }
    elsif ($p->{$f} =~ m/^-?\d+$/) {
        push @fields, "$f INT8";
    }
    else {
        push @fields, "$f TEXT";
    }
}
```

Still Usual (3/3)

```
print <<CREATE_TYPE;
CREATE TYPE process_table_type AS (
    @{[ join(",\n    ", @fields) ]}
);
CREATE_TYPE
```

Dynamic Custom Type?!

```
$ psql pg_pdx_fall_2007
Welcome to psql 8.2.5, the PostgreSQL interactive terminal.
...
pg_pdx_fall_2007=> \set mptt `./make_process_table_type.pl`
pg_pdx_fall_2007=> :mptt
CREATE TYPE
```

The PL/PerlU Function

```
CREATE OR REPLACE FUNCTION get_ps ()
RETURNS SETOF process_table_type
LANGUAGE plperlu
AS $$
use Proc::ProcessTable;
my $proctab = new Proc::ProcessTable;
return $proctab->table;
$$;
```

A Query

```
SELECT *
FROM
    pg_class "c1",
    pg_class "c2",
    pg_class "c3";
```

What is happening?!?

```
SELECT
      a.current_query,
      MAX(p.size) AS “size”
FROM
      pg_stat_activity AS "a"
JOIN
      get_ps() AS "p"
      ON (a.procpid = p.pid)
GROUP BY a.current_query;
```

Aha!

```
     current_query       |    size
-------------------------+------------
SELECT *                 |
FROM                     |
     pg_class "c1",      |
     pg_class "c2",      |
     pg_class "c3";      | 195290000

(1 row)
```

LOLCODE!

IM IN UR DATABUKKIT, READIN' UR ROWS

http://pgfoundry.org/projects/pllolcode/

Compiling PL/LOLCODE

Download the source.

Make sure you have flex 2.5.33.

2.5.4 does NOT work!

make && make install

Install PL/LOLCODE

```
CREATE FUNCTION pl_lolcode_call_handler()
    RETURNS language_handler
    AS 'pllolcode' LANGUAGE C;

CREATE LANGUAGE pllolcode handler pl_lolcode_call_handler;
```

Use PL/LOLCODE (1/2)

```
CREATE OR REPLACE FUNCTION lol_main_test()
RETURNS BOOLEAN
LANGUAGE pllolcode
AS $$
HAI
    VISIBLE NOTICE "1 This is a test"
    VISIBLE INFO "2 This is a test"
    FOUND YR WIN
KTHXBYE
$$;
```

Use PL/LOLCODE (2/2)

```
SELECT lol_main_test();
--------------------------
 t
(1 row)
```

PREVIEWZ UV CUMMIN' ATRTAKSHUNZ

```
HAI
    CAN HAS DATABUKKIT?
    I HAS A RESULT
    I HAS A RECORD
    GIMMEH RESULT OUTTA DATABUKKIT "SELECT field FROM mytable"
    IZ RESULT NOOB?
        YARLY
            BYES "SUMWUNZ IN YR PGSQL STEELIN YR DATA"
    KTHX
    IM IN YR LOOP
        GIMMEH RECORD OUTTA RESULT
        VISIBLE RECORD!!FIELD
        IZ RESULT NOOB? KTHXBYE
    IM OUTTA YR LOOP
KTHXBYE
```

Thanks!

Copyright David Fetter 2007
All Rights Reserved
http://fetter.org/