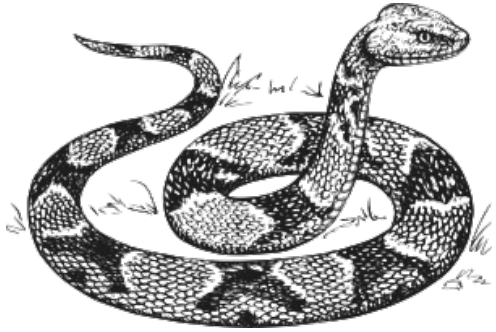
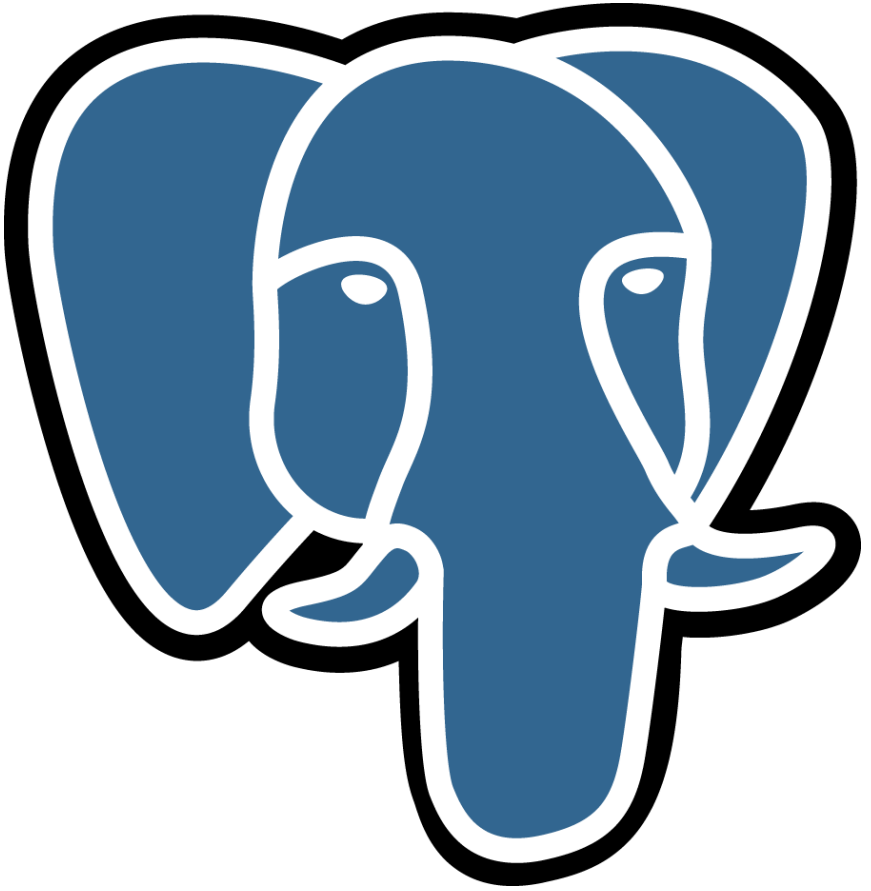
Python in the database
Pycon Canada – Nov 2012





Steve Singer
steve@ssinger.info
http://scanningpages.wordpress.com

PostgreSQL

Stored Functions

aka Stored Procedures

Why Stored Functions: Performance

http://www.flickr.com/photos/toastforbrekkie/2228668790

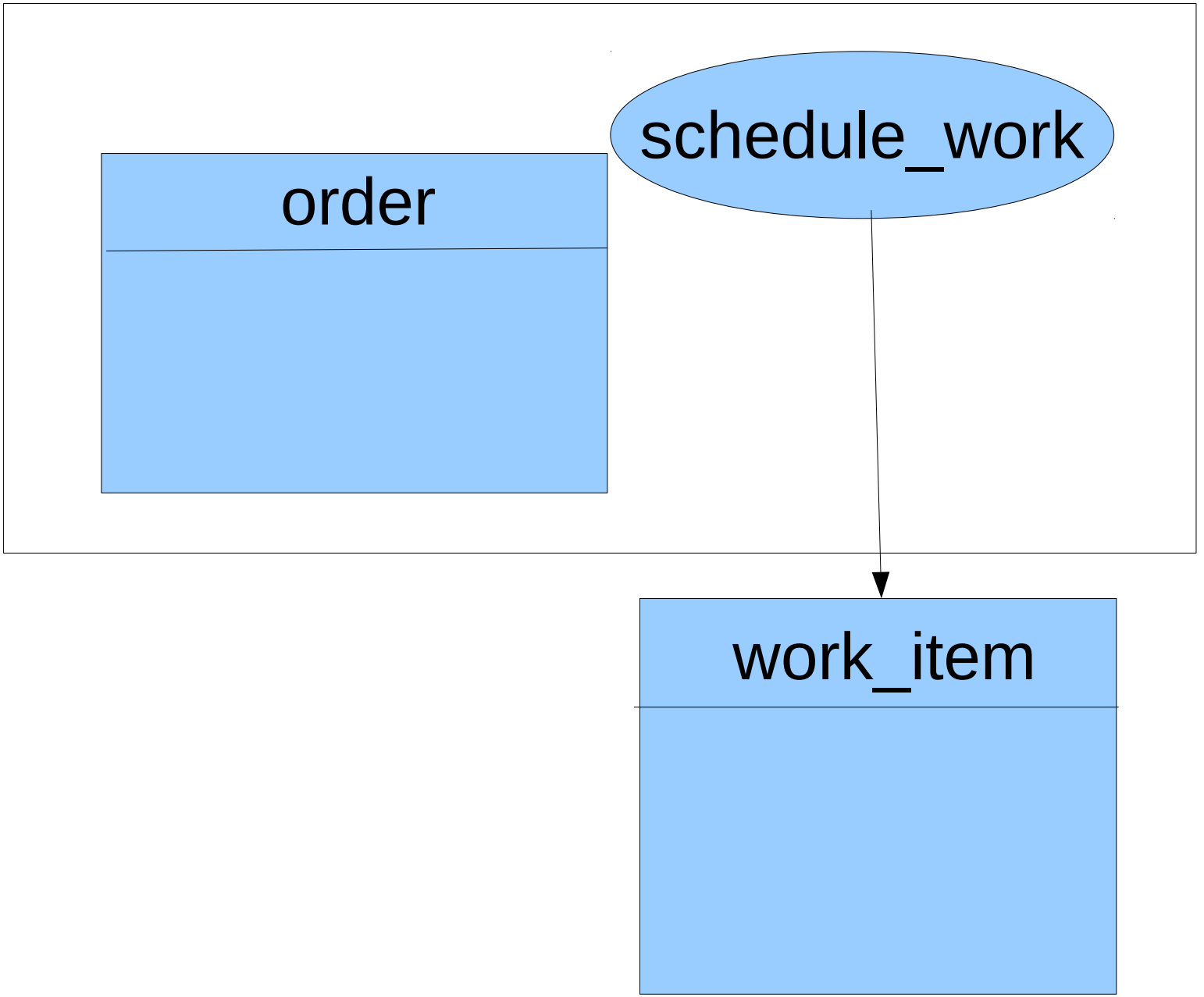
Why Stored Functions: Code Reuse

https://picasaweb.google.com/lh/photo/QmIw9uv6xYsrZRFyGaAT5dMTjNZETYmyPJy0liipFm0

Why Stored Functions: Security

http://www.flickr.com/photos/37815348@N00/5398546351

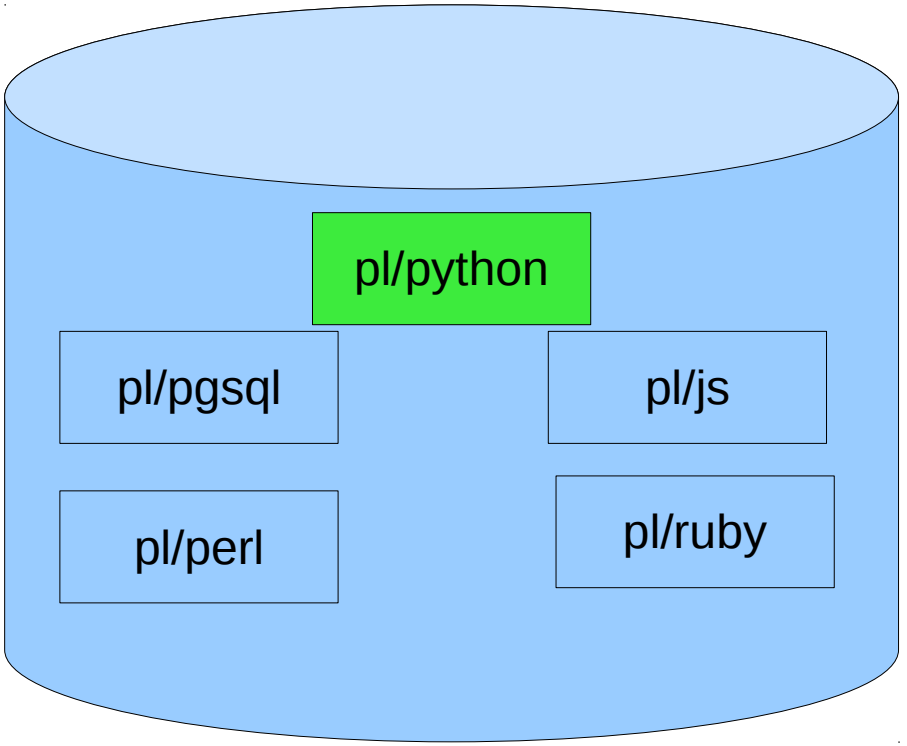
Why Stored Functions: Triggers



Tradtional pl/SQL functions

```
CREATE OR REPLACE FUNCTION myfunc(count integer)
returns SETOF integer AS
$$
declare
ind integer;
begin
ind:=1;
loop
  exit when ind>=count;
  return next ind;
  ind:=ind+1;
end loop;
end
$$ LANGUAGE plpgsql;
```

Pl/Python



Why Python

Basic Steps: Create Extension

```
CREATE EXTENSION plpythonu;
```

Basic function

```
CREATE OR REPLACE FUNCTION myfunc2(counter integer)
 returns SETOF integer AS
$$
  #return range(1,counter)
  ret=[]
  for x in range(1,counter):
   ret=ret+[x]
  return ret
$$ language plpythonu;
```

Calling the function

```
test=# select myfunc2(10);
 myfunc2
---------
       1
       2
       3
       4
       5
       6
       7
       8
       9
(9 rows)

test=#
```

Using Modules

```
CREATE OR REPLACE FUNCTION myfunc4(a integer,b
integer) returns  float AS
$$
  import fractions
  afrac=fractions.Fraction(1,a)
  bfrac=fractions.Fraction(1,b)
  sum=afrac+bfrac
  return float(sum)
$$ language plpythonu;
```

```
 select myfunc4(3,3);
    myfunc4
---------------
 0.666666666667
(1 row)
```

Being a polite guest

Your function is part of the of PostgreSQL process!

So are your modules

Don't fork

Don't mess with signal handlers

Calling the database

```
CREATE OR REPLACE FUNCTION myfunc5(sale_date date)
returns float AS $$
import plpy
import fractions
plan=plpy.prepare('select id, sold,allocation from sales '\
 'where date=$1',['date'])
rs = plpy.execute(plan,[sale_date])
total=fractions.Fraction()
for row in rs:
   frac=fractions.Fraction(row['sold'],row['allocation'])
   total=total+frac
return float(total)
$$ LANGUAGE plpythonu;
```

Transaction Management

```
CREATE OR REPLACE FUNCTION myfunc6(id integer)
returns integer AS $$
import plpy
from plpy import spiexceptions
res=id
while True:
  try:
    with plpy.subtransaction():
      plan=plpy.prepare('insert into sales (id,sold,allocation,date) '\
'values ($1,$2,$3,now()::date)',['integer','integer','integer'])
      plpy.execute(plan,[res,1,1])
      break
  except spiexceptions.UniqueViolation,e:
     res=res+1
return res
$$ LANGUAGE plpythonu;
```

plpydbabi
Make plpy look like DB API

https://github.com/petere/plpydbapi

```
import plpydbapi

dbconn = plpydbapi.connect()
cursor = dbconn.cursor()
cursor.execute("SELECT ... FROM ...")
for row in cursor.fetchall():
    plpy.notice("got row %s" % row)
dbconn.close()
```

Organization

myfunc.sql

```
CREATE OR REPLACE FUNCTION myfunc.....
$$
import mymodule
mymodule.myfunc()
$$ LANGUAGE plpythonu;
```

mymodule.py

```
import plpy
def myfunc():
  res=plpy.execute('select ....')
```

Questions ?

http://www.postgresql.org/docs/9.2/interactive/plpython.html

Steve Singer
steve@sssinger.info
http://scanningpages.wordpress.com