Knowing your data cluster and startup processes

Amul Sul

Who am I?

- My name is Amul Sul.
- I’m a database developer at EnterpriseDB.
  - Working from Pune, India office.
- PostgreSQL contribution:
  - Hash Partitioning.
  - Extended hash functions.
  - Bug fixes and review work.

Agenda

Part 1 : Database File Layout

- Overview of storage format at the level of files and directories.
- Dig more into **`base`** and **`pg_tblspc`** directories.

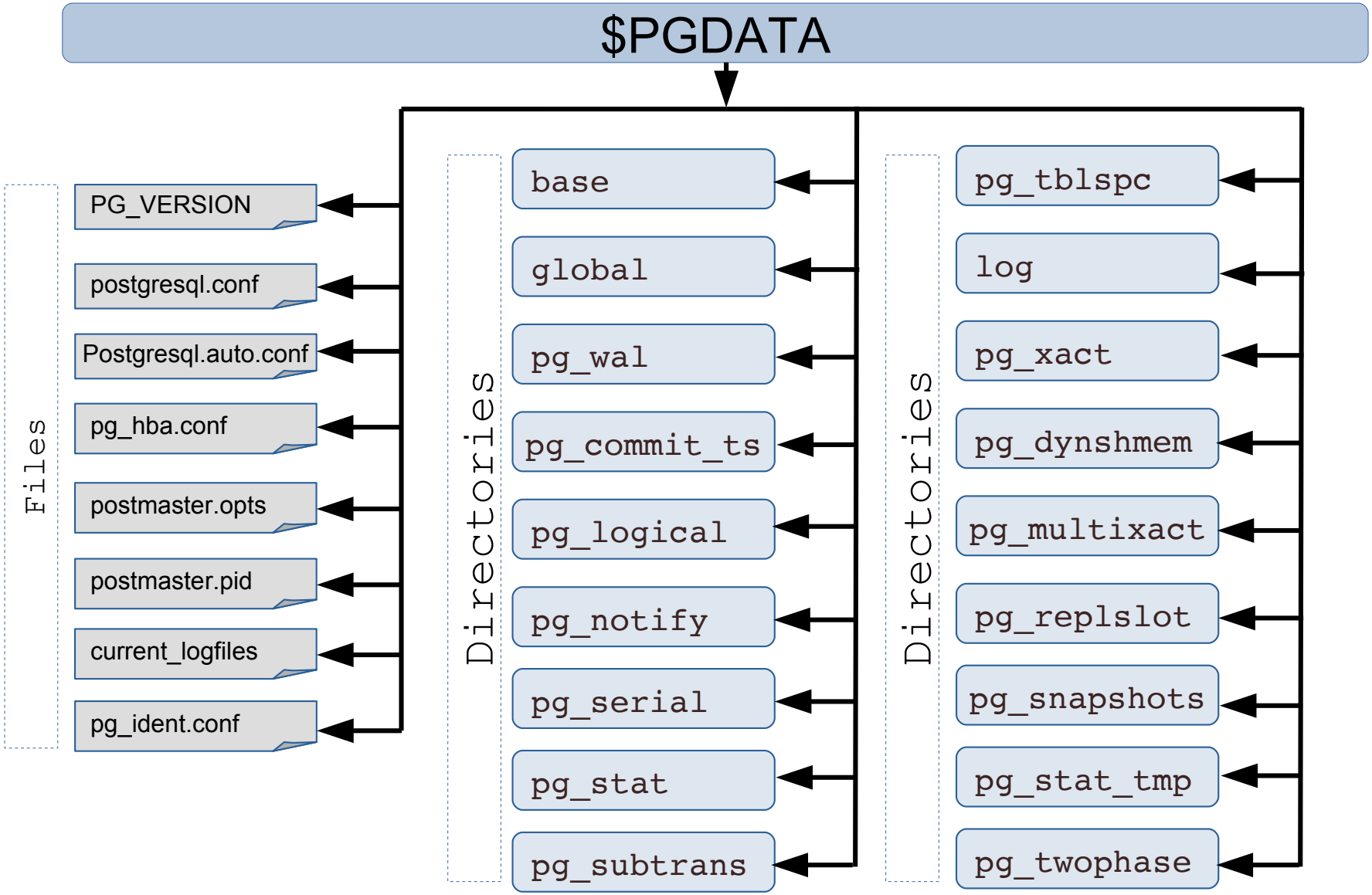
Part 2 : Process Architecture

- Overview of server processes.
- Quick walk through each of them.
- Client connection steps.

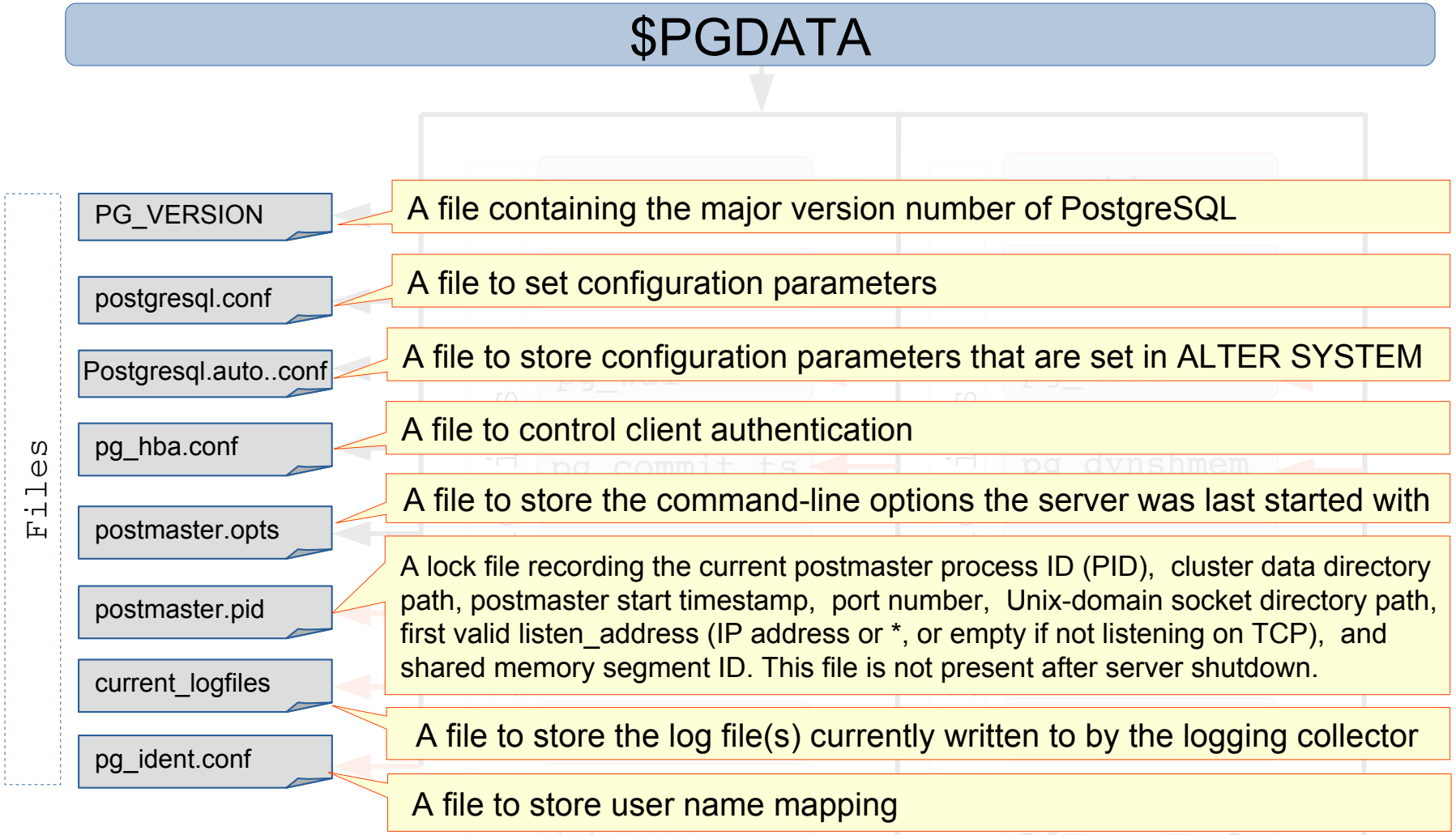
Part 1 : Database file layout

- Data directory, commonly referred to as **`PGDATA`** (name of the environment variable).
- A common location for **`PGDATA`** is **`/var/lib/pgsql/data`**.
- **`PGDATA`** directory contains several subdirectories and control files.
- Configuration files by default are located into PGDATA, can be placed elsewhere.

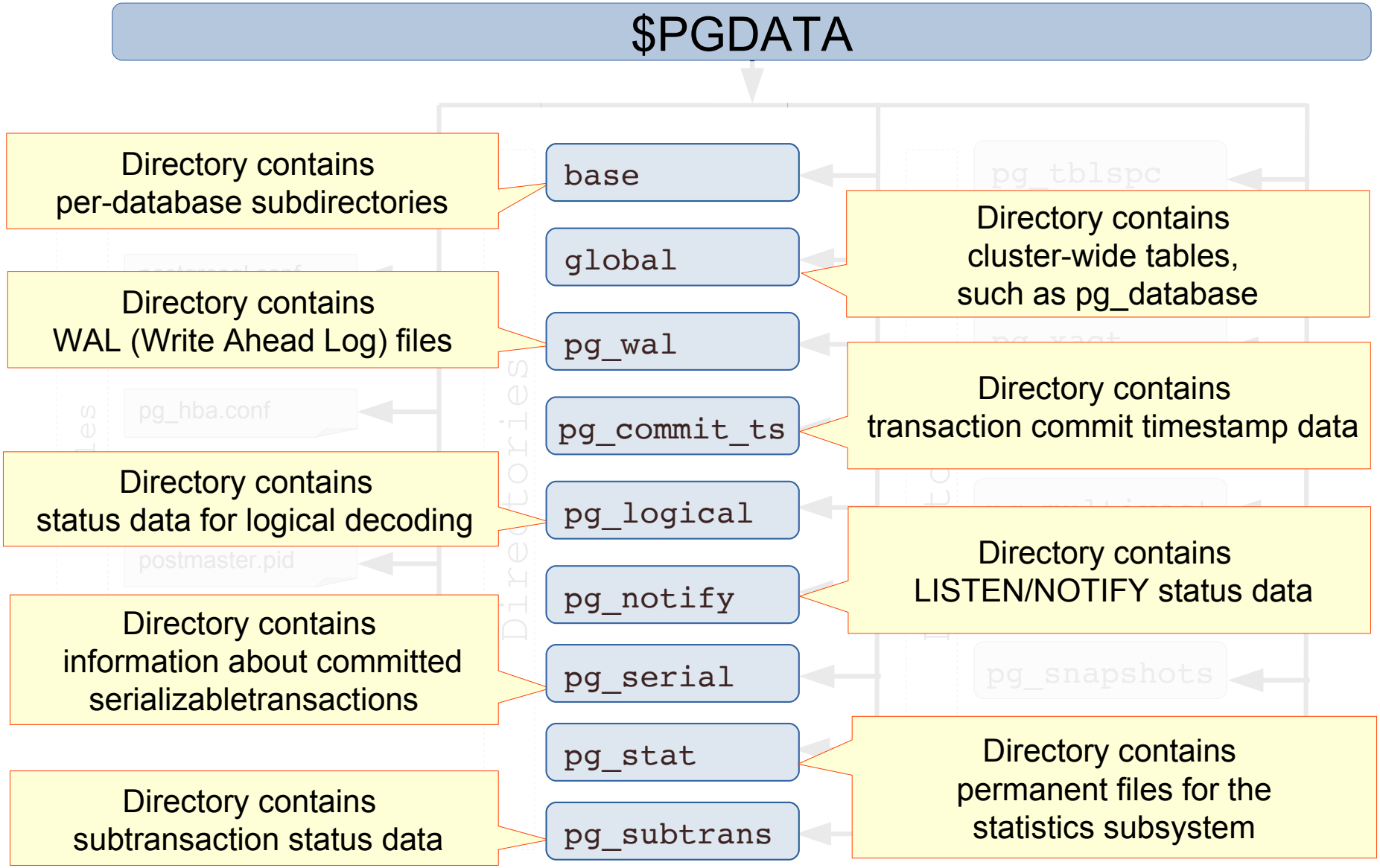
Data directory : 8 Files & 18 directories



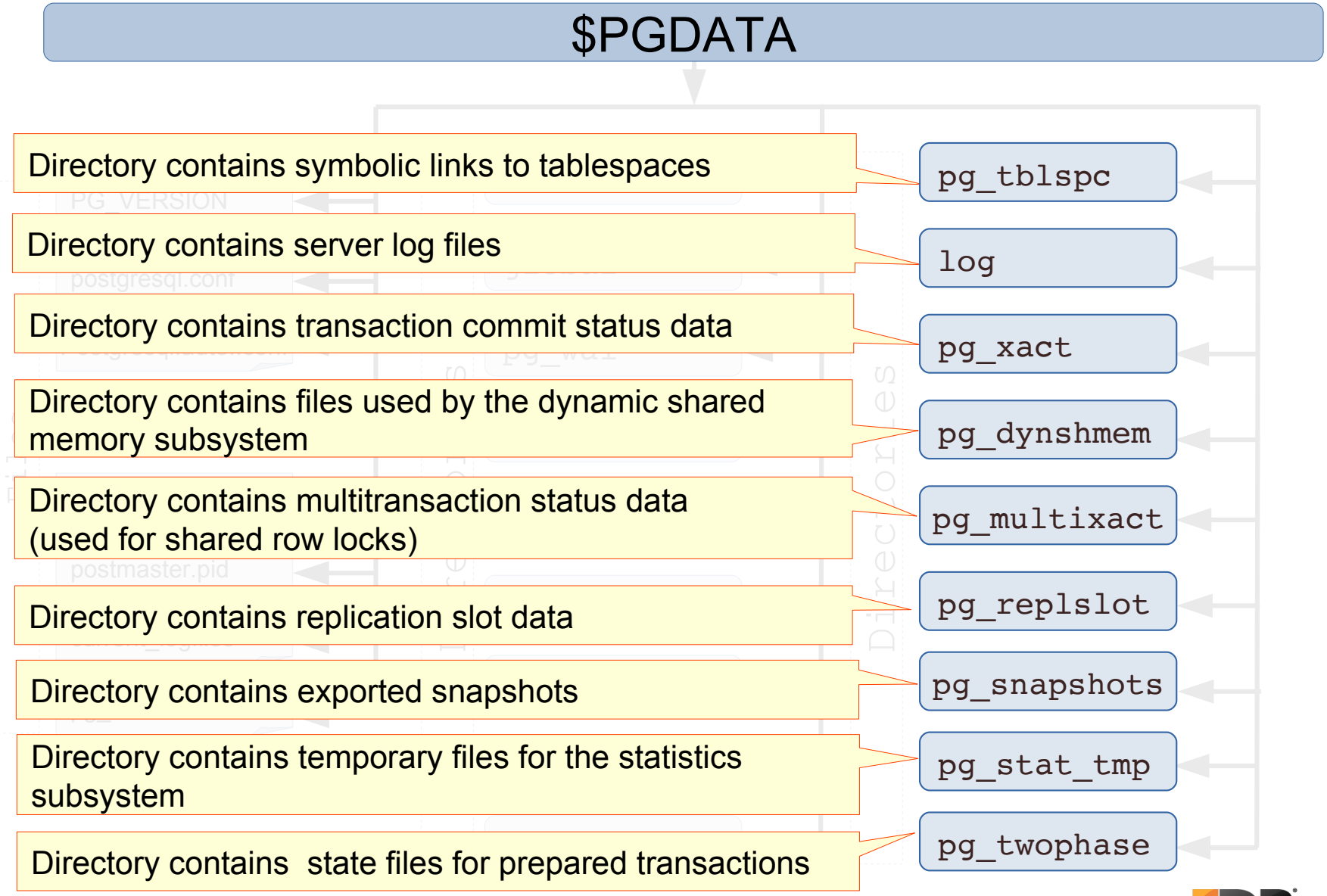
Data directory : 8 Files



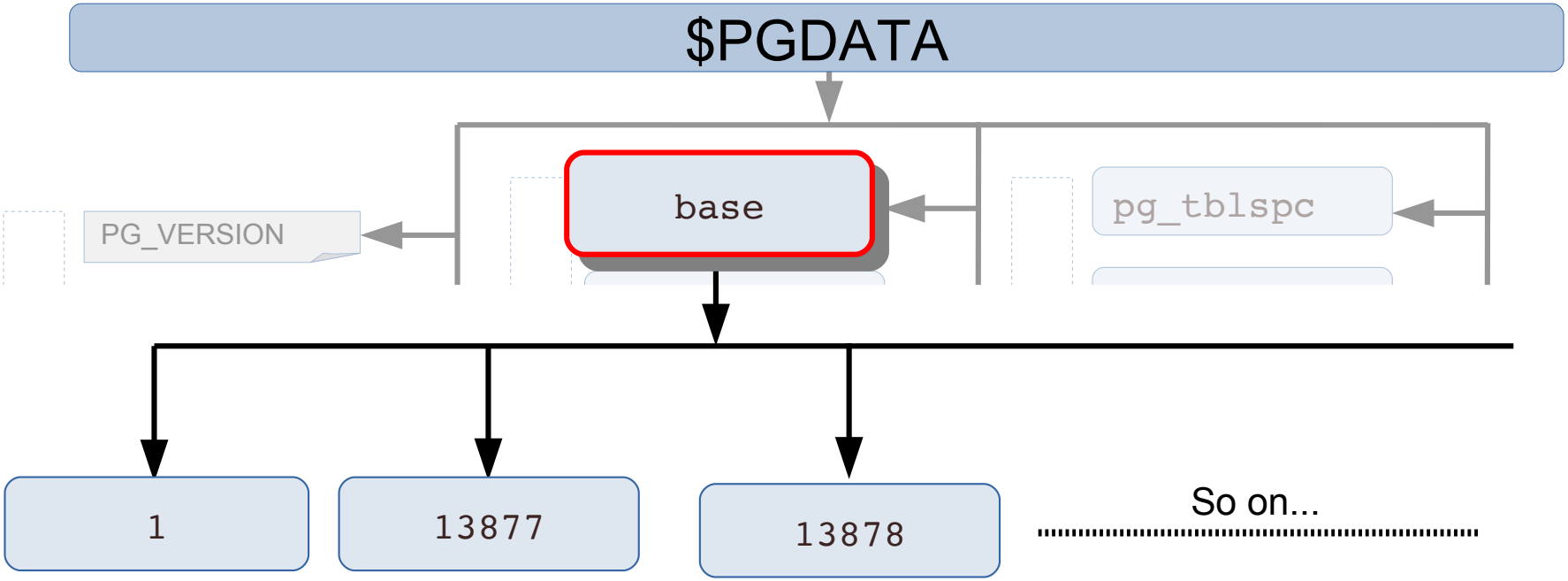
Data directory : 18 directories



Data directory : 18 directories (cont..)



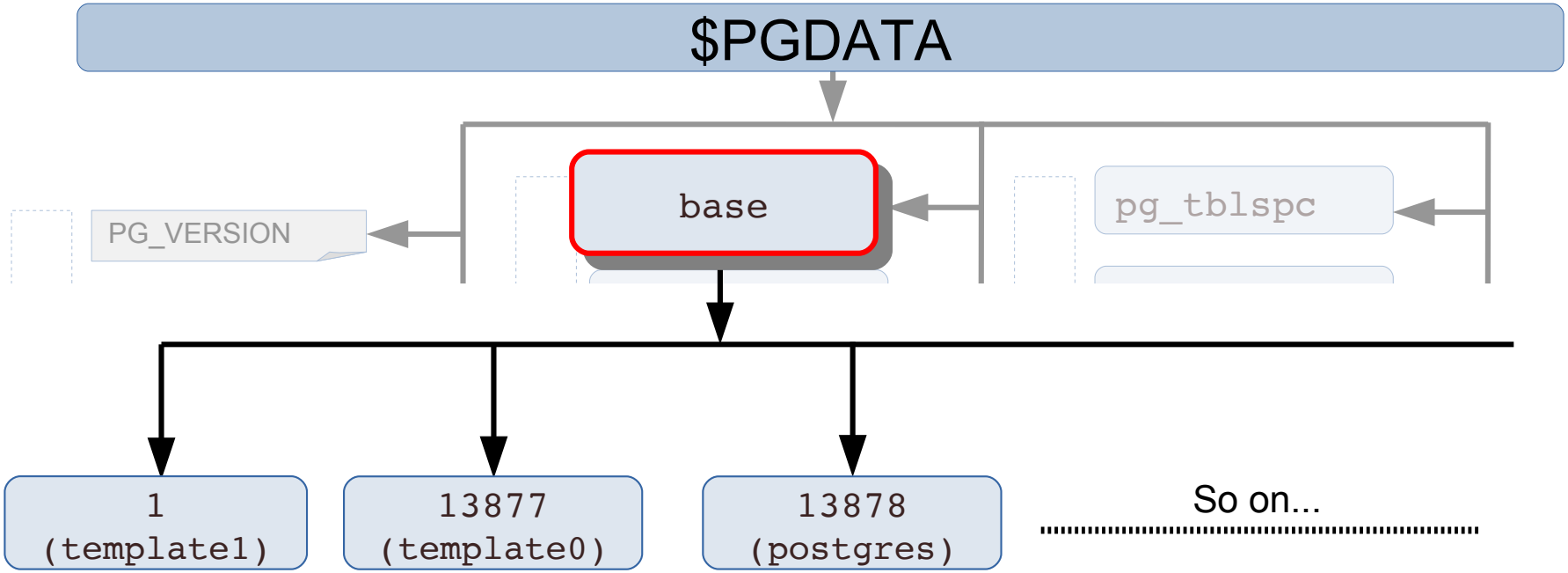
Data directory : base directory



These are database oids, let's find the names:

```
postgres=# SELECT oid, datname FROM pg_database
postgres-# WHERE oid in (1,13877,13878) ORDER BY oid;
  oid  |  datname
-------+-----------
     1 | template1
 13877 | template0
 13878 | postgres
(3 rows)
```

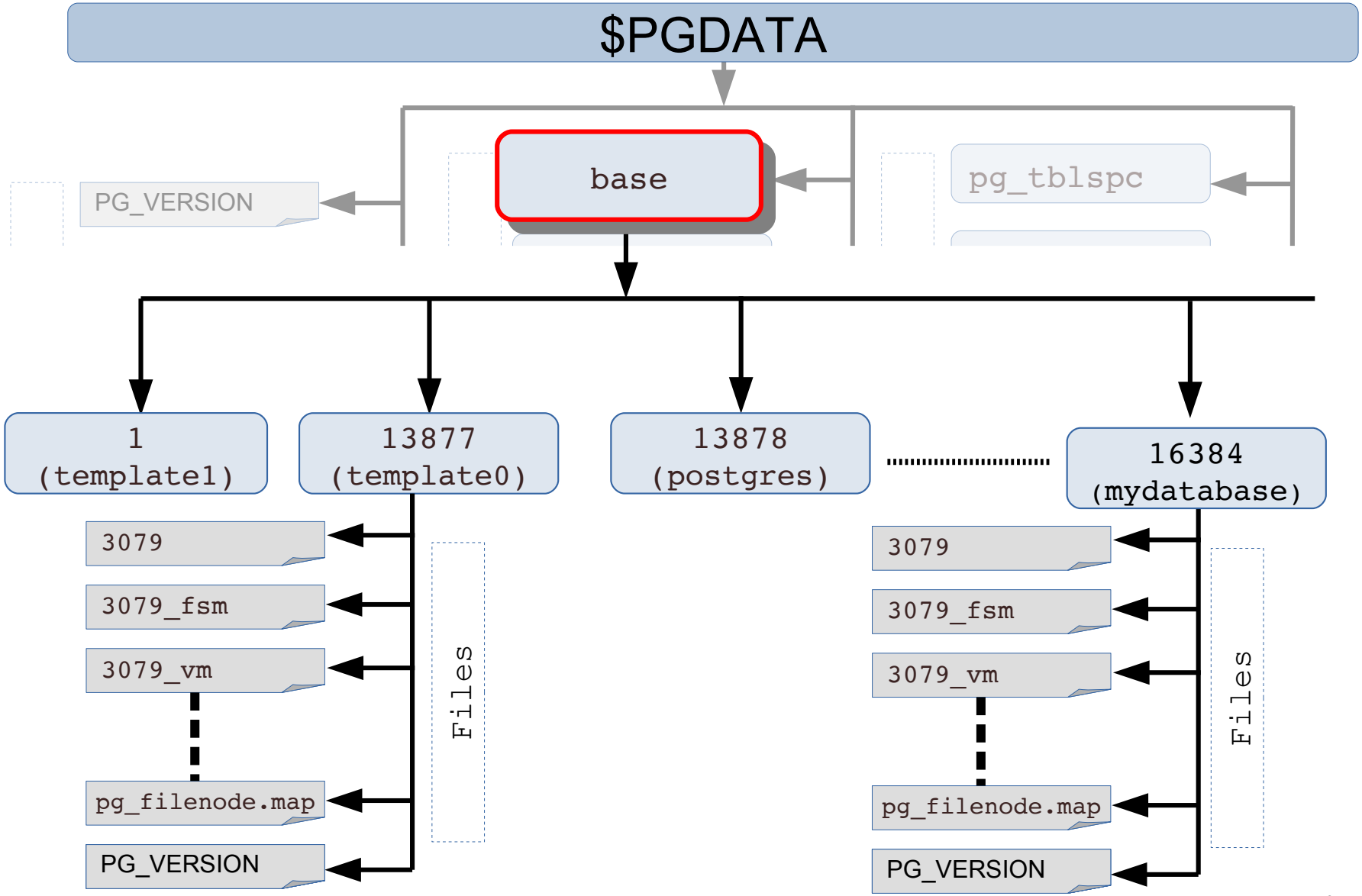
Data directory : base directory



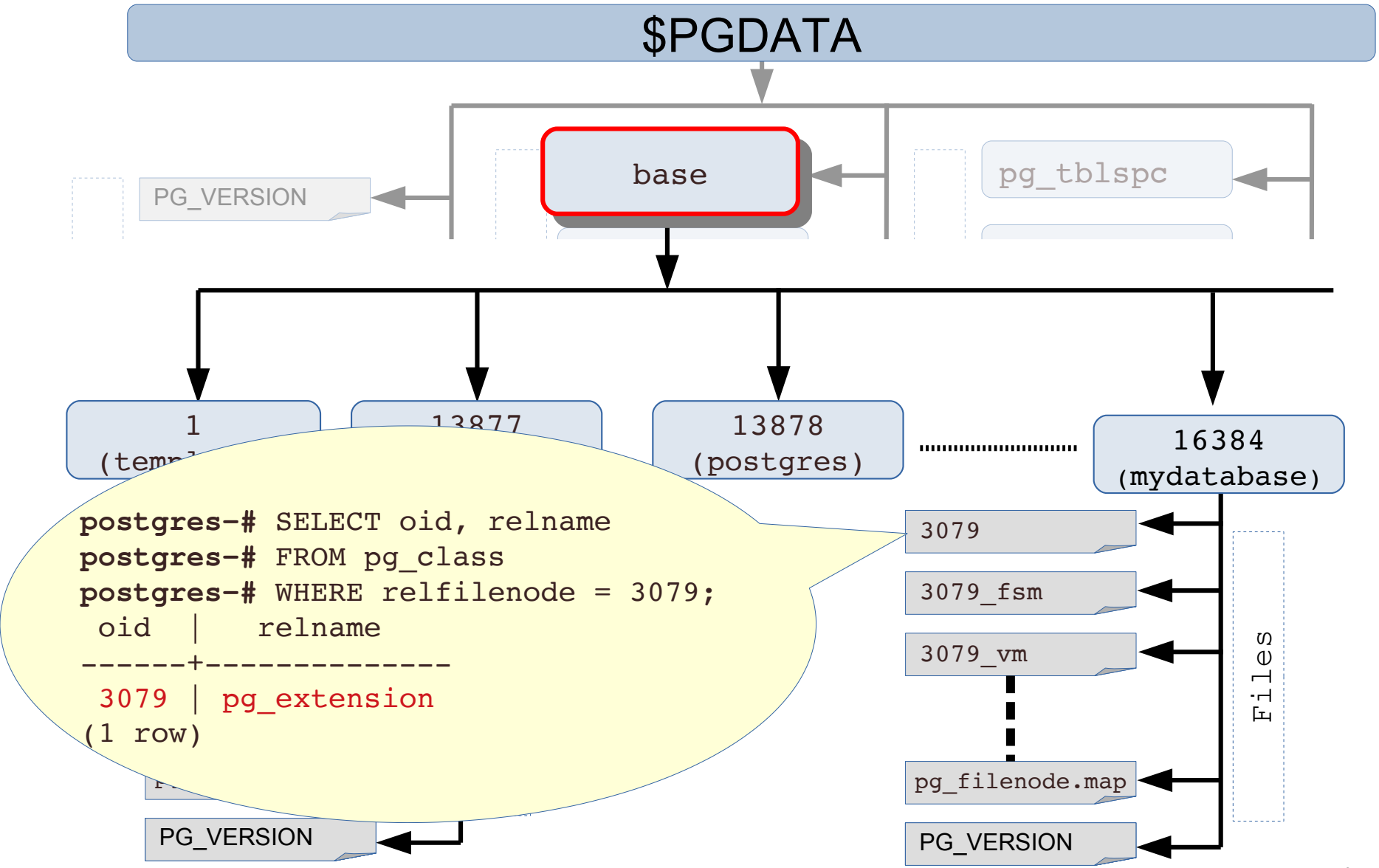
If you create another database you'll have another directory then:

```
postgres=# CREATE DATABASE mydatabase;
CREATE DATABASE

postgres=# SELECT oid FROM pg_database WHERE datname = 'mydatabase';
  oid
-------
 16384
(1 row)
```

Data directory : base directory



Data directory : base directory

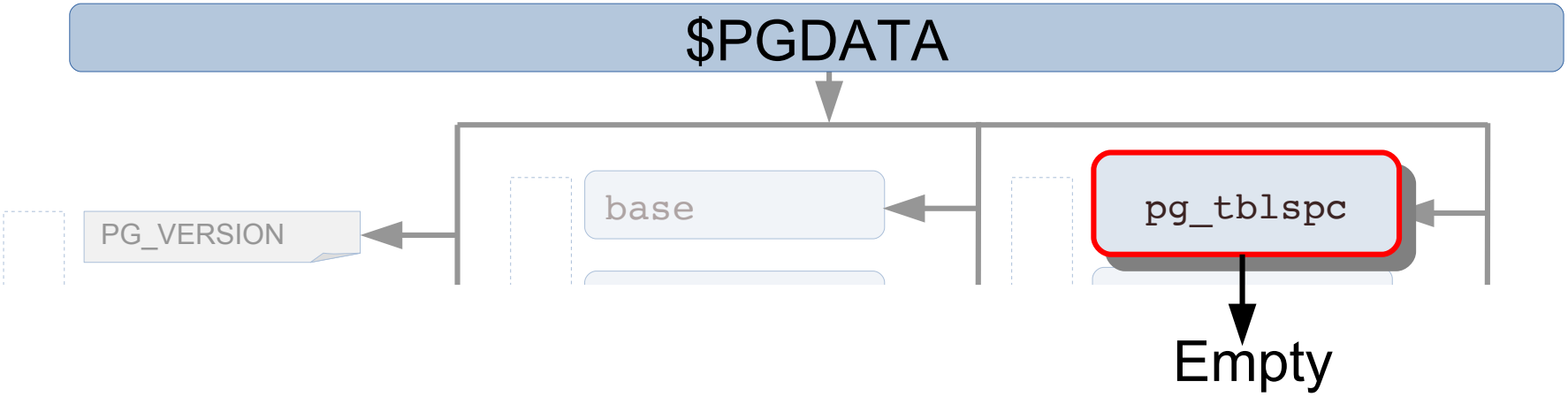


Data directory : base directory

- Relation of table oid,and relfilenode

```
mydatabase=# CREATE TABLE foo(i int);
CREATE TABLE

mydatabase=# SELECT oid, relfilenode,
mydatabase=# pg_relation_filepath(relname::text)
mydatabase-# FROM pg_class WHERE relname = 'foo';
  oid  | relfilenode | pg_relation_filepath
-------+-------------+----------------------
 16427 |       16427 | base/16384/16427
(1 row)

mydatabase=# truncate foo;
TRUNCATE TABLE

mydatabase=# SELECT oid, relfilenode,
mydatabase=# pg_relation_filepath(relname::text)
mydatabase=# FROM pg_class WHERE relname = 'foo';
  oid  | relfilenode | pg_relation_filepath
-------+-------------+----------------------
 16427 |       16430 | base/16384/16430
(1 row)
```

Data directory : pg_tblspc directory




```
mydatabase=# CREATE TABLESPACE my_tablespace LOCATION '/tmp/tblspc';
CREATE TABLESPACE
(1 row)
```

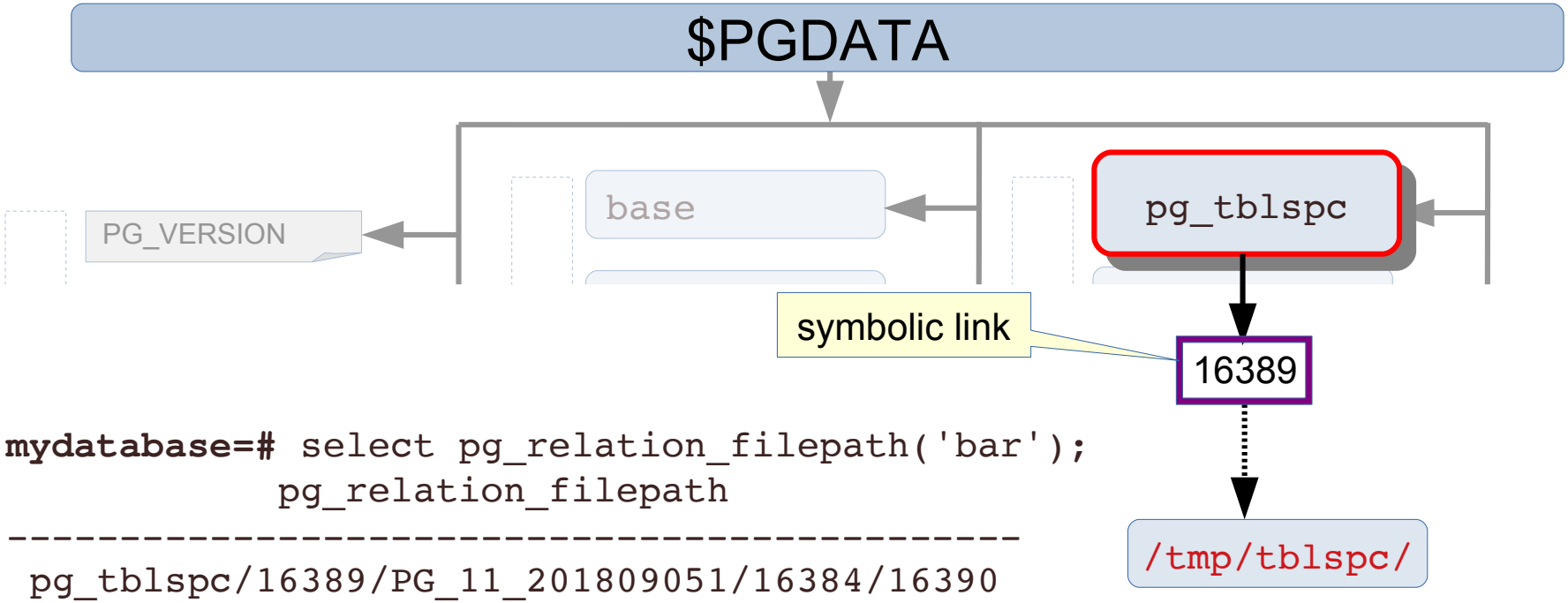
– On linux terminal

```
$ tree -C -a /var/lib/pgsql/11/data/pg_tblspc/

/var/lib/pgsql/11/data/pg_tblspc/
└── 16389 -> /tmp/tblspc

1 directory, 0 files
```

The tablespace directory is addressed by a symbolic link from the pg_tblspc subdirectory.

The link name is the same as the OID value of tablespace.

Data directory : pg_tblspc directory




```
mydatabase=# select pg_relation_filepath('bar');
            pg_relation_filepath
-----------------------------------------------
 pg_tblspc/16389/PG_11_201809051/16384/16390
(1 row)
```

```
– On linux terminal
$ tree -C -a /tmp/tblspc
/tmp/tblspc
└── PG_11_201809051
    └── 16384
        └── 16390

2 directories, 1 file
```

'my_tablespace' - naming conventios is as follow:

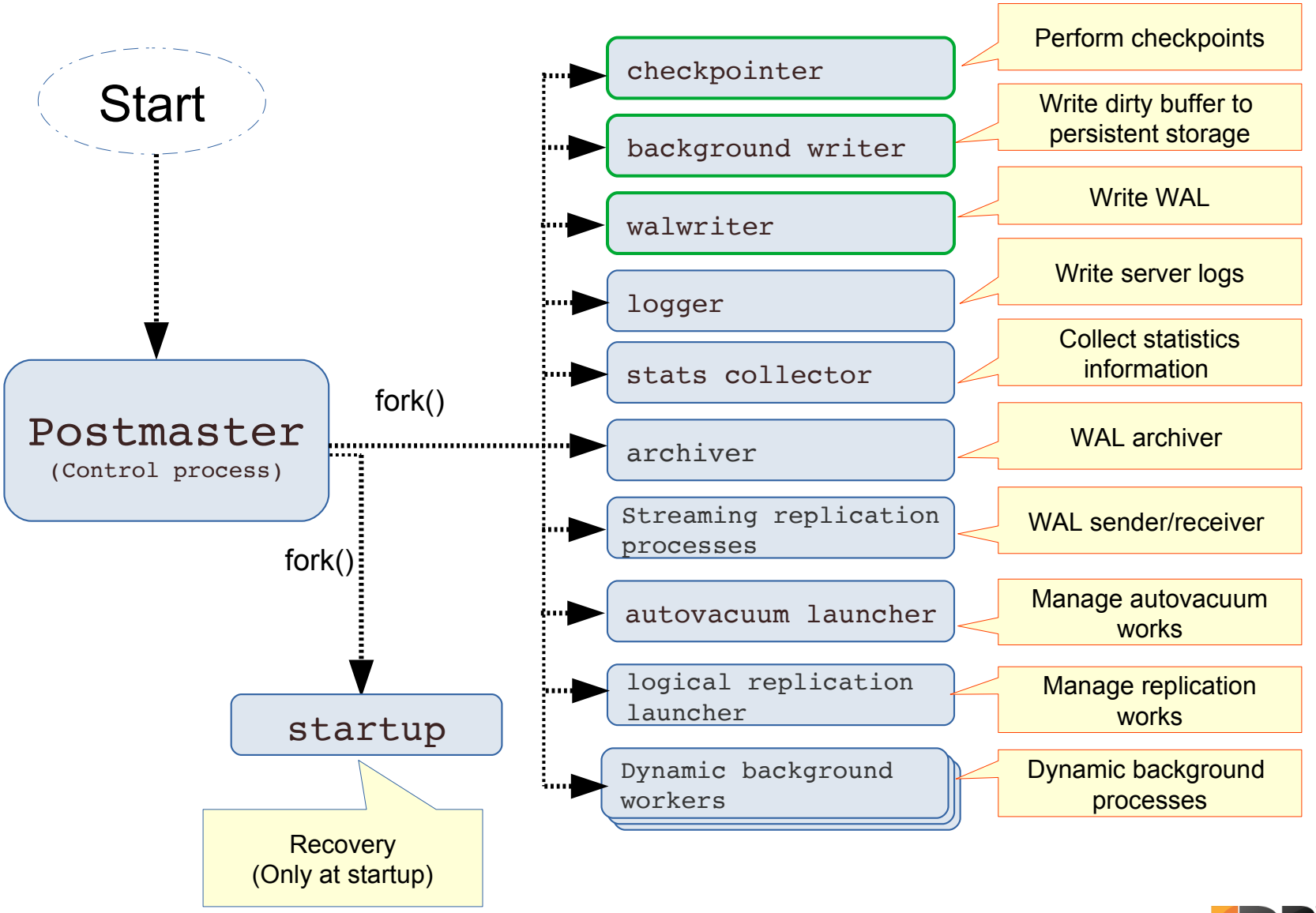
PG _ <Major version> _ <Catalogue version number>

Directory for 'mydatabase'

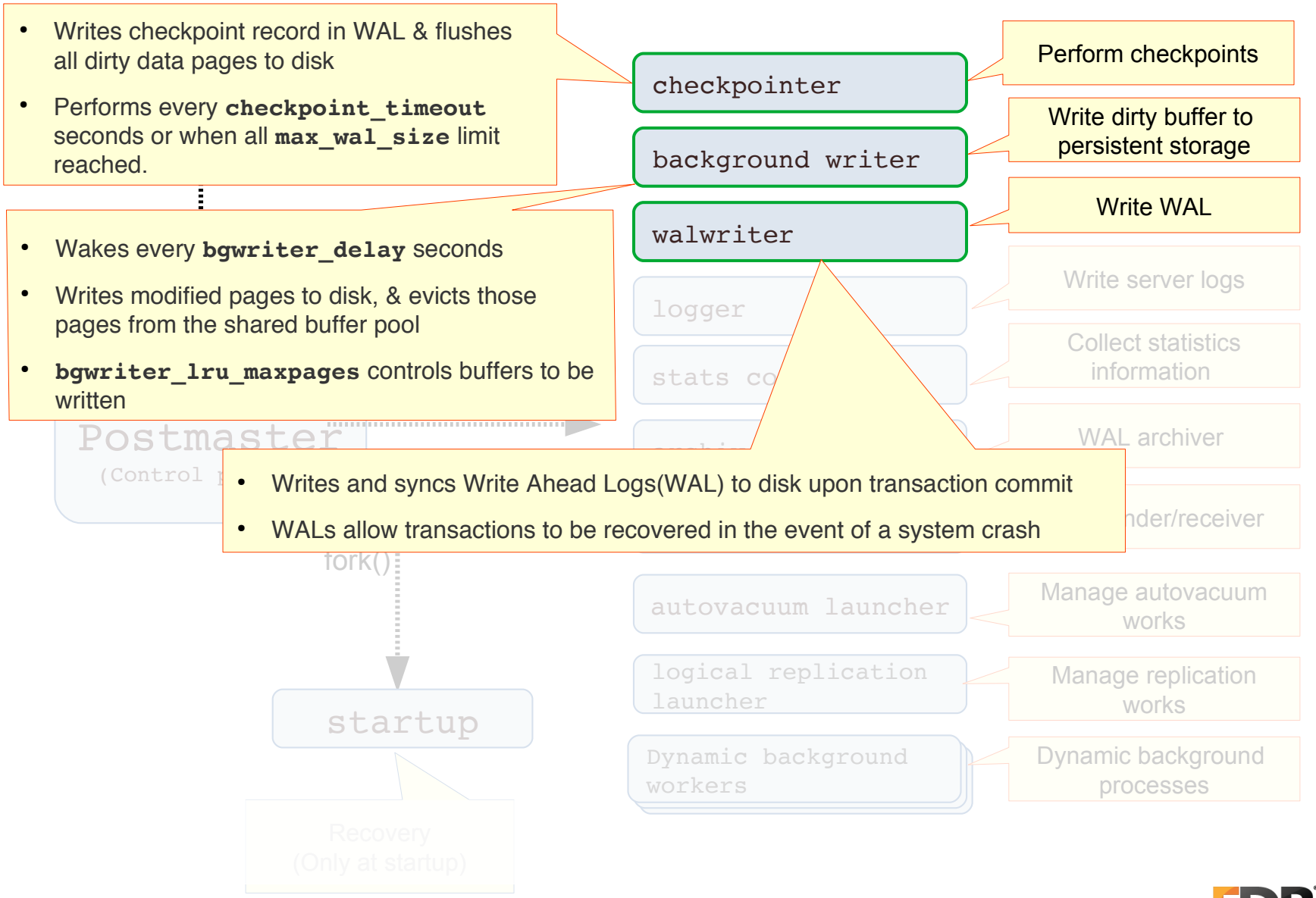
Relfilenode of 'bar' relation

- PostgreSQL is a client/server type RDMS with the multi-process architecture and runs on a single host.
- A collection of multiple processes cooperatively managing one database cluster is usually referred to as a 'PostgreSQL server', and it contains the following type of processes:
  - A `postmaster` process is the parent of all the processes related to a database cluster management.
  - Various `background processes` perform processes of each feature (e.g., `VACUUM, CHECKPOINT, etc`) for database management.
  - Each `backend process` handles all queries and statements issued by a connected client.
  - In the `background worker process` supported from version 9.3, it can perform any processing implemented by users.

Process structure



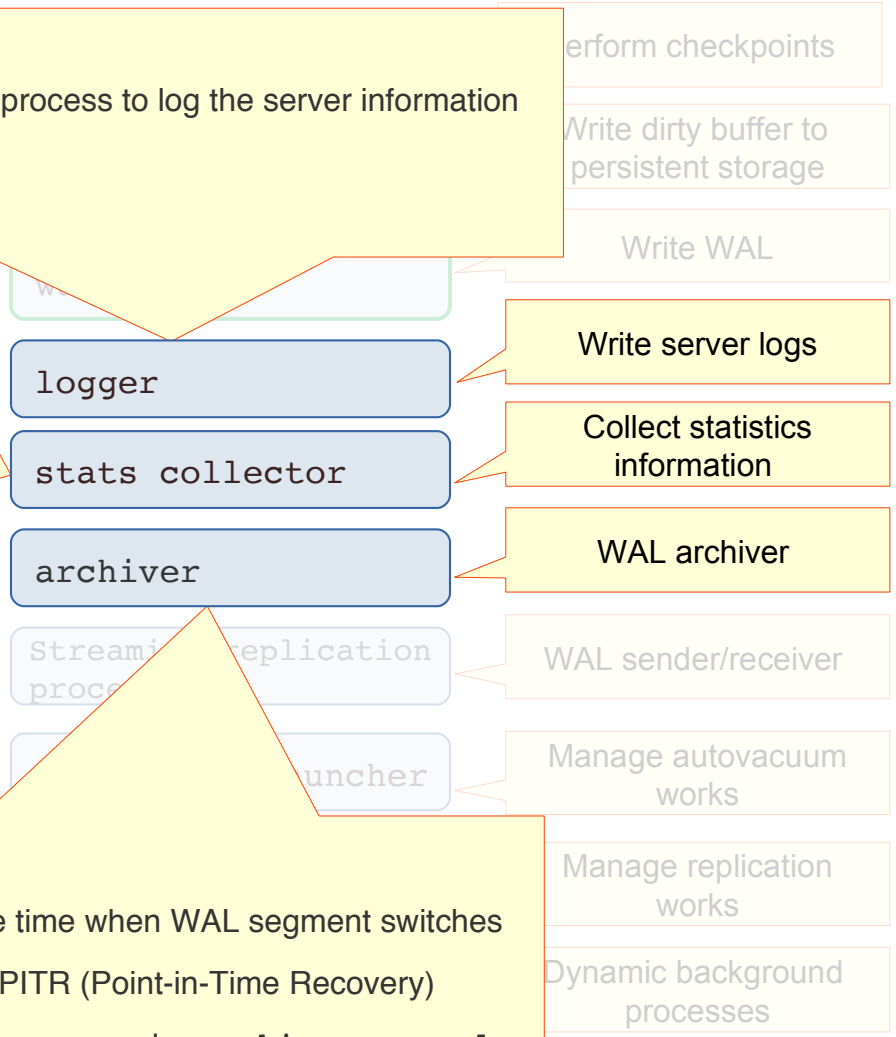
Process structure



Process structure

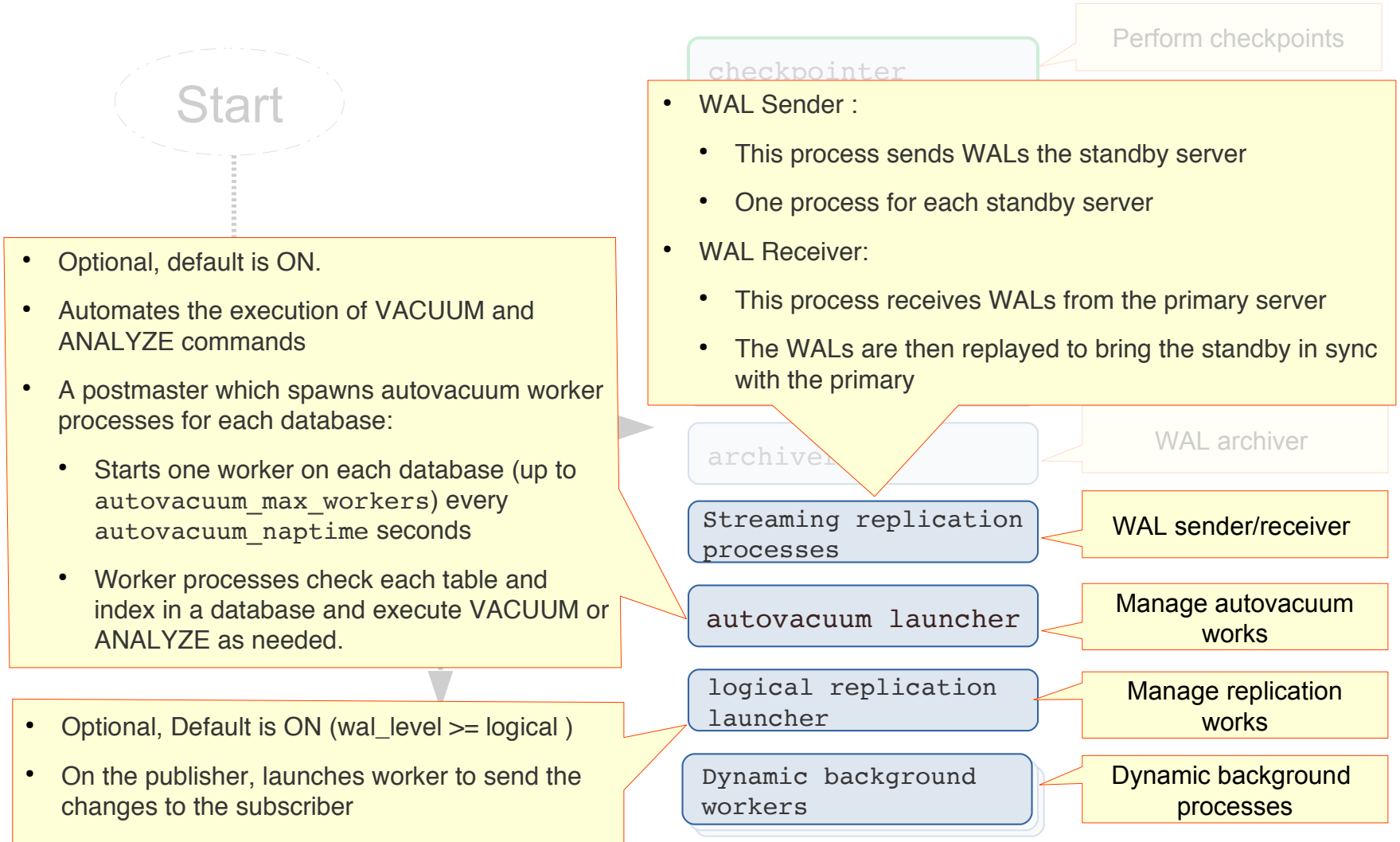
- Optional, default OFF
- All processes including postmaster do attache to this process to log the server information
- All information logged into **`$PGDATA/log`** directory
- Do supports different file formats supported

- Optional, default is ON
- Collects information about cluster activity:
  - Number of access to the tables and indexes
  - Total number of rows in each table
  - Information about VACUUM and ANALYZE actions for each table
- Collection of statistics adds some overhead to query execution, but allows the query planner to make better choices

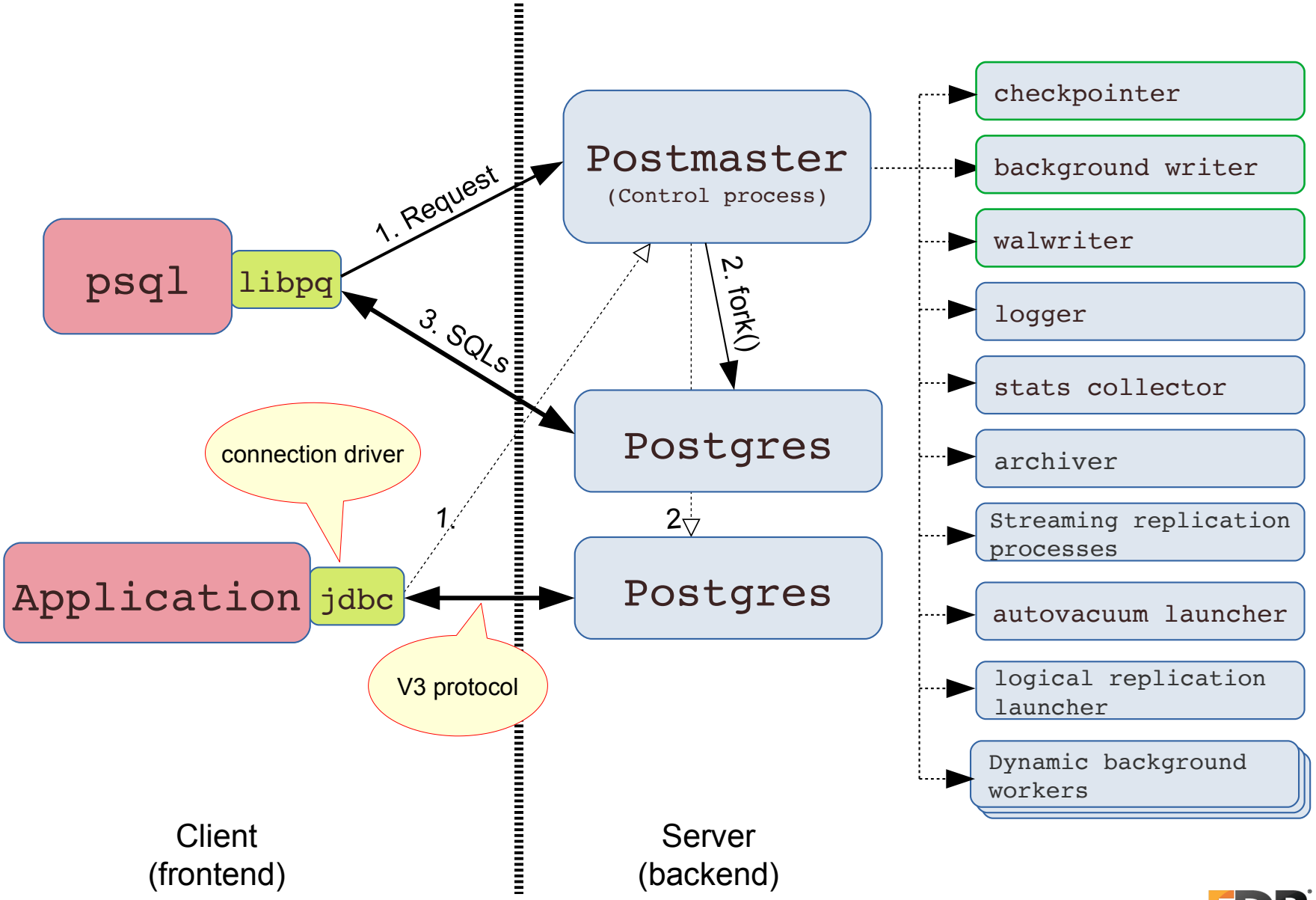


- Optional, default OFF
- Copies WAL segment files to the archival area at the time when WAL segment switches
- Feature is usually used for hot physical backup and PITR (Point-in-Time Recovery)
- The path of the archival area is set to the configuration parameter **`archive_command`**
  - **`wal_level`** configuration parameter to replica or higher, **`archive_mode`** to ON.
  - **archive_command** = 'cp %p /home/postgres/archives/%f'

Process structure



Client connection



21

References:

1. PostgreSQL 11 official documentation
2. Overview of Postgres Utility Processes by EDB Dave Thomas
3. The Internals of PostgreSQL by Hironobu SUZUKI.

Thank You !!

Questions?

!! We Are Hiring !!

- Java Full Stack Developers
  Python Full Stack Developers
  Java Technical Architect
- UI Developers
  Technical Support
  Technical Sales Engineer
- Sales

Contact us on akshay.chavan@enterprisedb.com or visit us at the booth.