INFOH-417 Database System Architectures 2022-23
Lab session 4

Mariana Duarte
mariana.machado.garcez.duarte@ulb.be

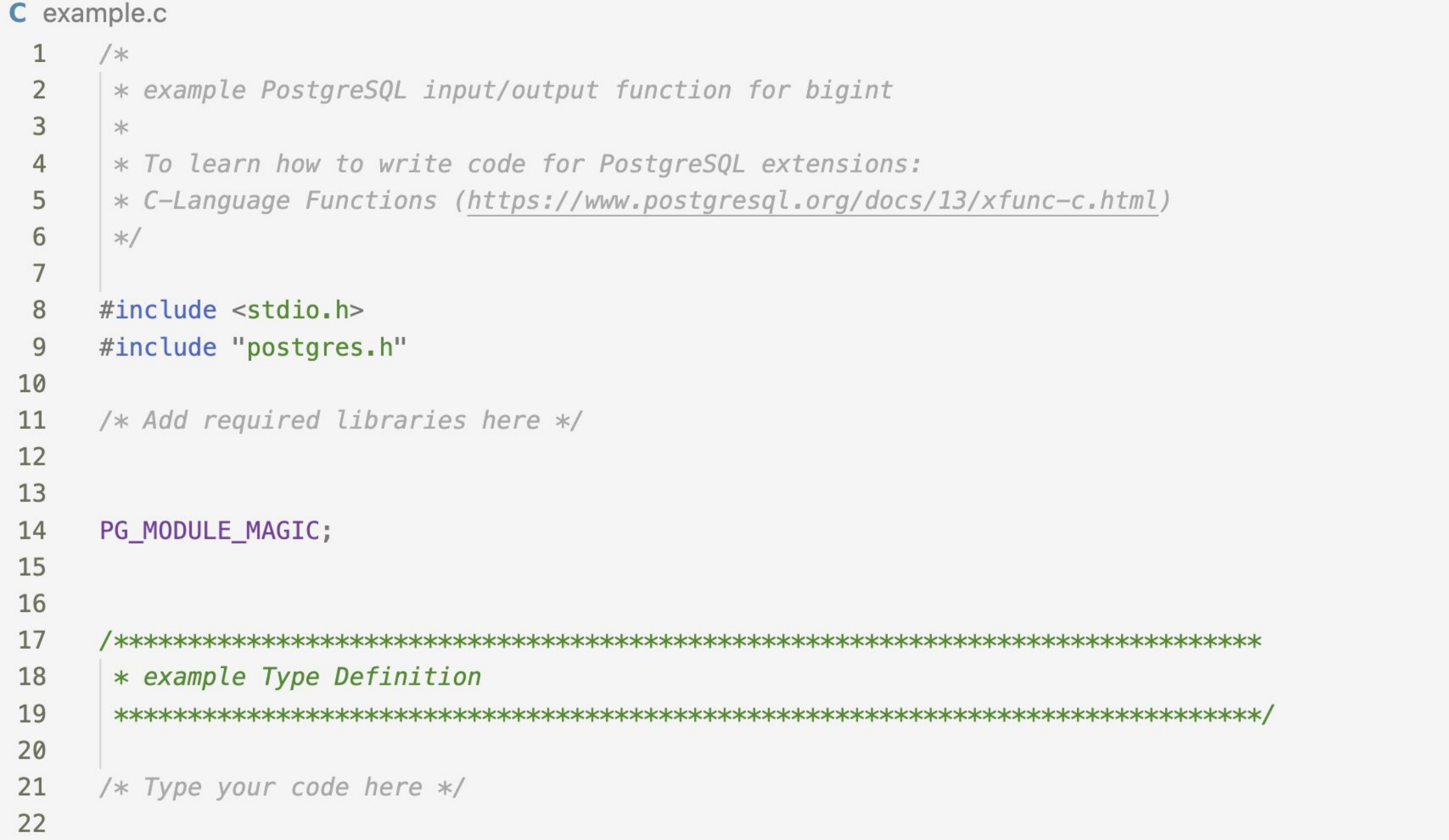
Create a new extension and data type

Create an empty extension

- C file
- Sql file
- Extension.control
- Makefile

Create an empty extension

- C file

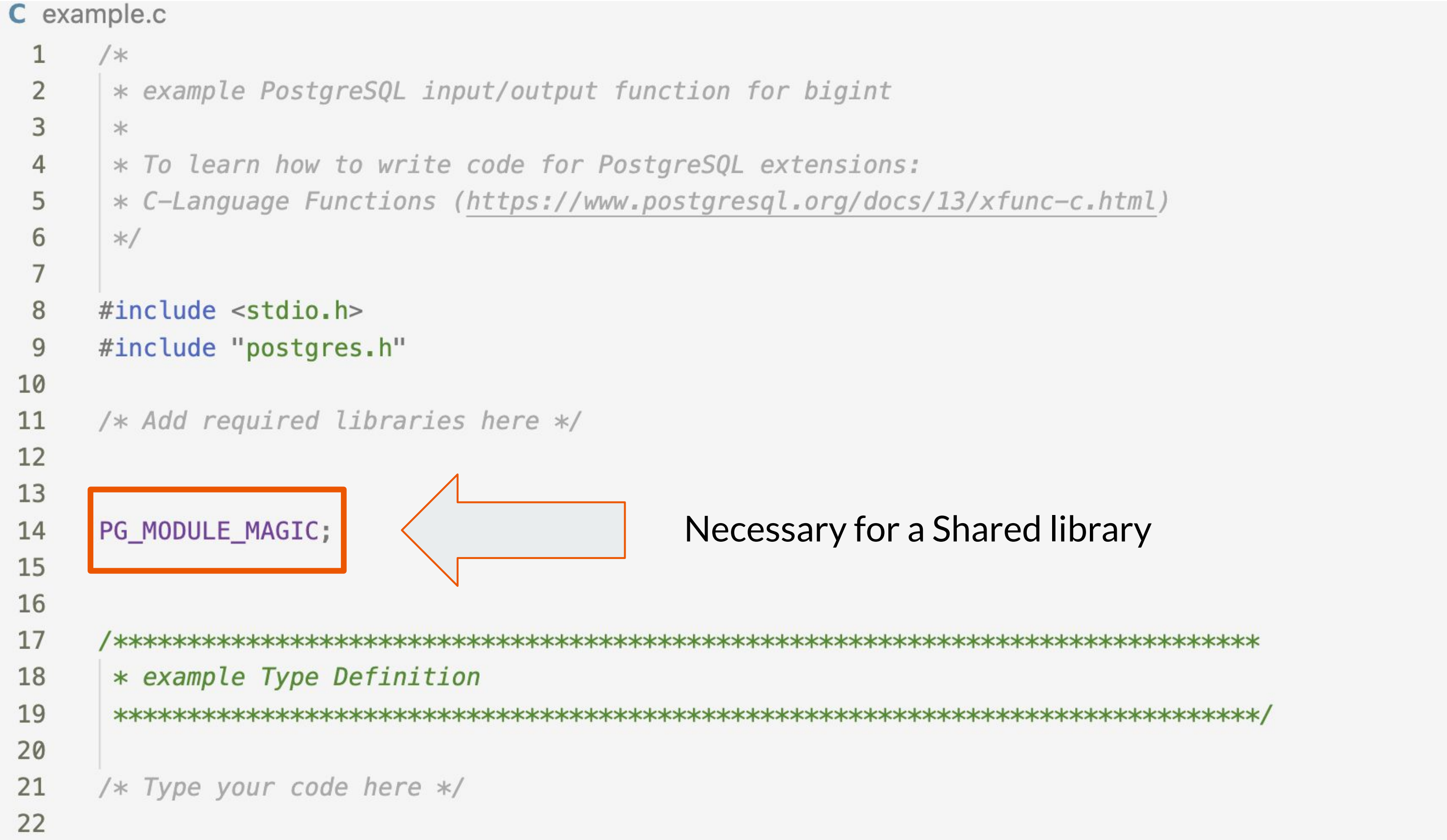


example.c

```
/*
 * example PostgreSQL input/output function for bigint
 *
 * To learn how to write code for PostgreSQL extensions:
 * C-Language Functions (https://www.postgresql.org/docs/13/xfunc-c.html)
 */

#include <stdio.h>
#include "postgres.h"

/* Add required libraries here */


PG_MODULE_MAGIC;


/*****************************************************************************
 * example Type Definition
 *****************************************************************************/

/* Type your code here */

```

Create an empty extension

- C file



C example.c

```c
/*
 * example PostgreSQL input/output function for bigint
 *
 * To learn how to write code for PostgreSQL extensions:
 * C-Language Functions (https://www.postgresql.org/docs/13/xfunc-c.html)
 */

#include <stdio.h>
#include "postgres.h"

/* Add required libraries here */


PG_MODULE_MAGIC;


/*****************************************************************************
 * example Type Definition
 *****************************************************************************/

/* Type your code here */

```

Necessary for a Shared library

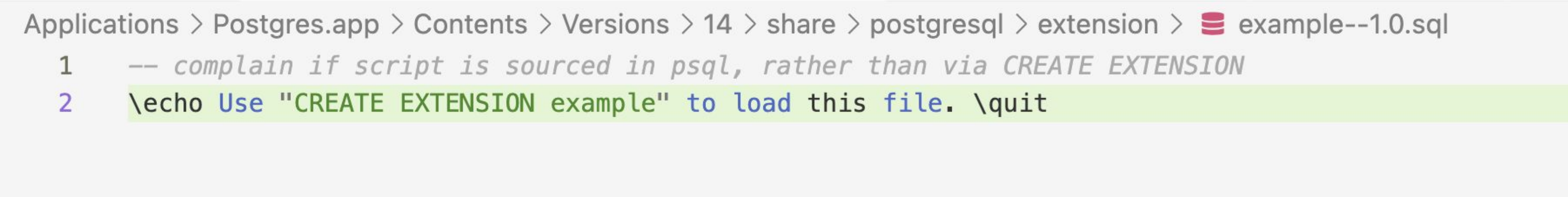
Create an empty extension

If not used, you will find this error

```
marianaduarte=# create extension example;
ERROR:  incompatible library "/Applications/Postgres.app/Contents/Versions/14/lib/postgresql/example.so":
missing magic block
HINT:  Extension libraries are required to use the PG_MODULE_MAGIC macro.
```

Create an empty extension

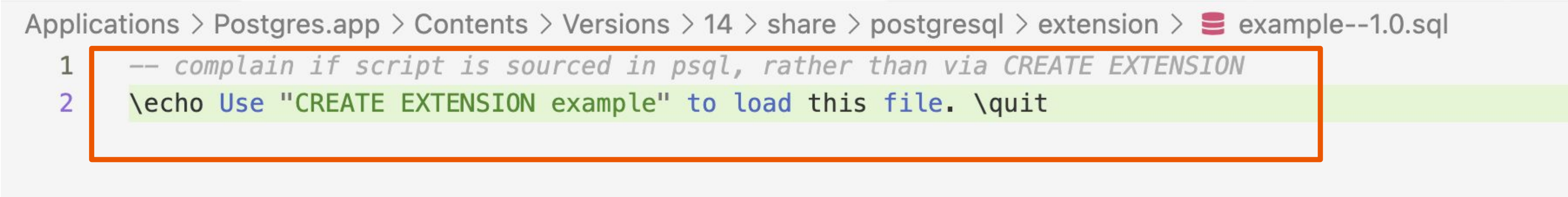
- SQL file


Applications > Postgres.app > Contents > Versions > 14 > share > postgresql > extension > example--1.0.sql

```
-- complain if script is sourced in psql, rather than via CREATE EXTENSION
\echo Use "CREATE EXTENSION example" to load this file. \quit
```

Create an empty extension

- SQL file



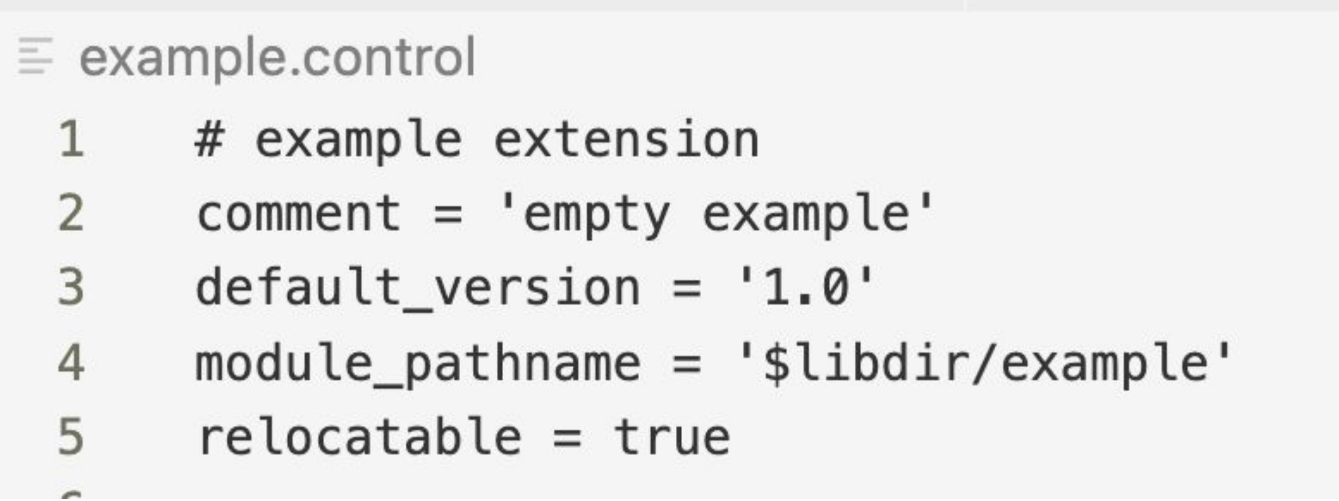
Applications > Postgres.app > Contents > Versions > 14 > share > postgresql > extension > example--1.0.sql

```sql
-- complain if script is sourced in psql, rather than via CREATE EXTENSION
\echo Use "CREATE EXTENSION example" to load this file. \quit
```

This avoids beeing used as a sql file instead of a library file
Cannot run this directly

Create an empty extension

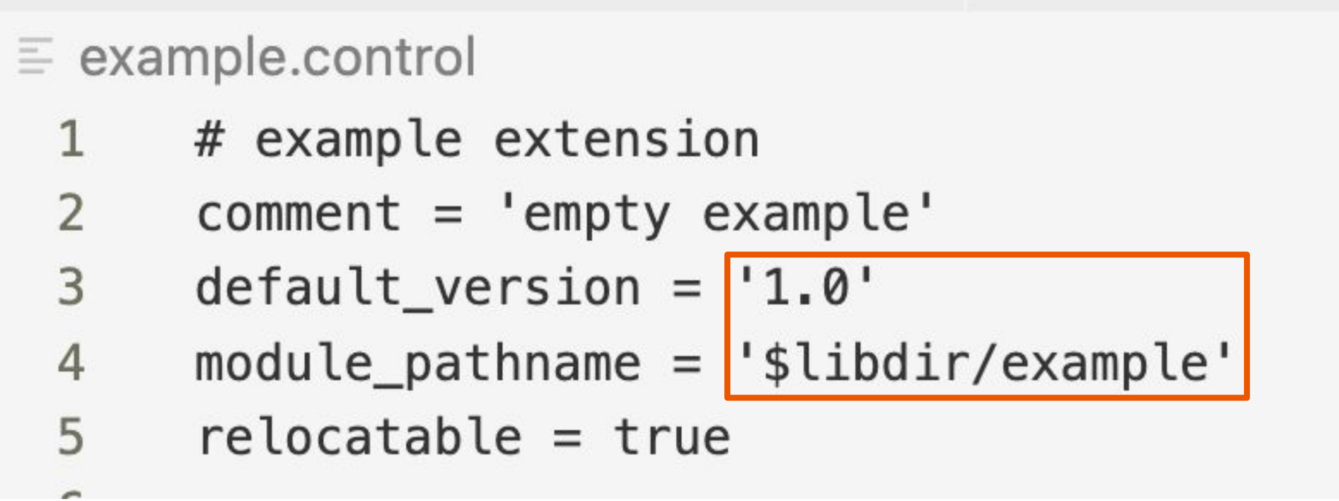
- Control File



```
# example extension
comment = 'empty example'
default_version = '1.0'
module_pathname = '$libdir/example'
relocatable = true
```

Create an empty extension

- Control File



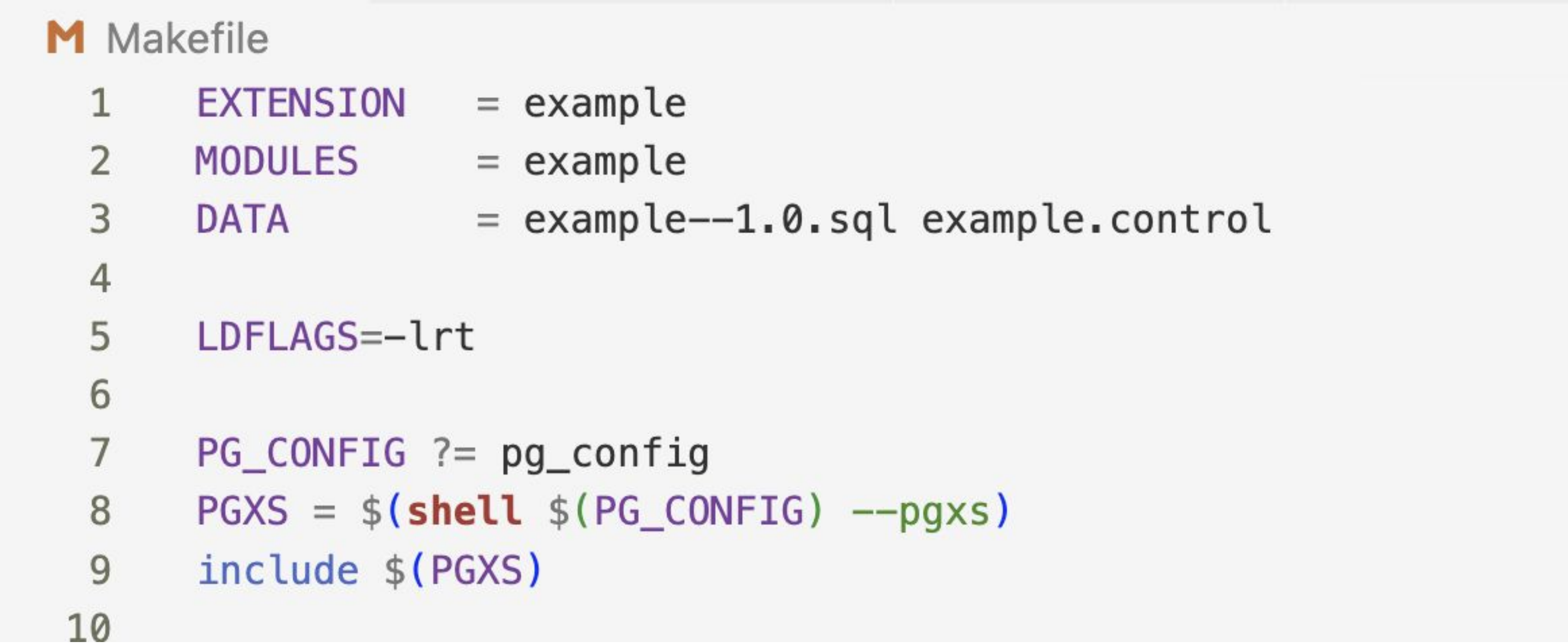
example.control

```
# example extension
comment = 'empty example'
default_version = '1.0'
module_pathname = '$libdir/example'
relocatable = true
```

Shows version and library path

Create an empty extension

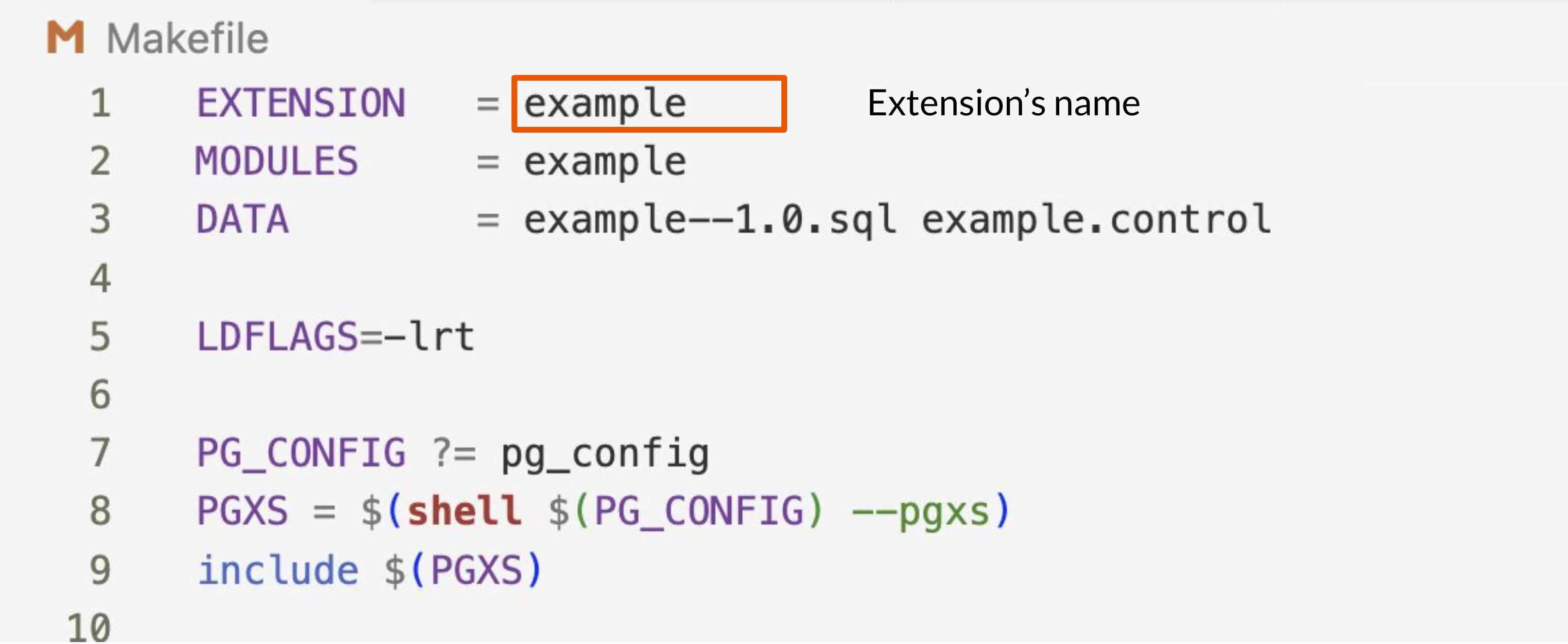
- Makefile



```
EXTENSION   = example
MODULES     = example
DATA        = example--1.0.sql example.control

LDFLAGS=-lrt

PG_CONFIG ?= pg_config
PGXS = $(shell $(PG_CONFIG) --pgxs)
include $(PGXS)

```

Create an empty extension

- Makefile

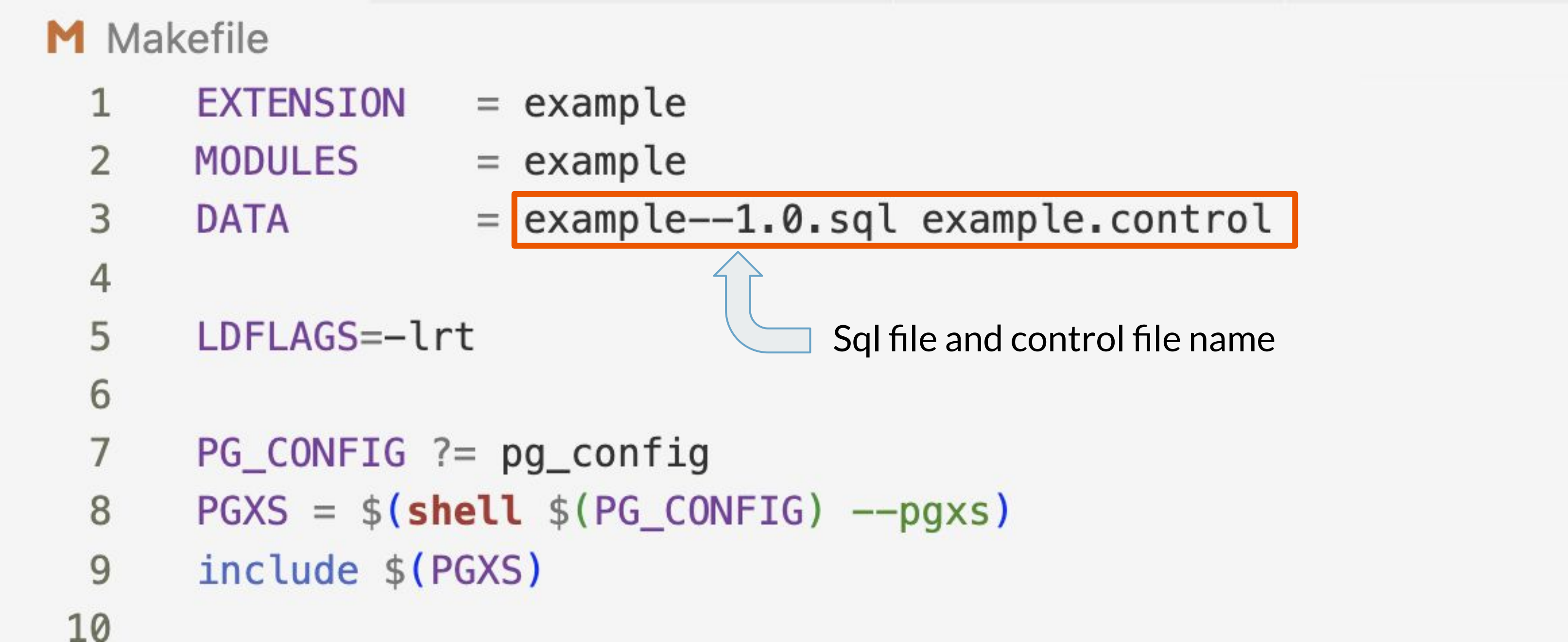


```
EXTENSION    = example
MODULES      = example
DATA         = example--1.0.sql example.control

LDFLAGS=-lrt

PG_CONFIG ?= pg_config
PGXS = $(shell $(PG_CONFIG) --pgxs)
include $(PGXS)

```

Extension’s name

Create an empty extension

- Makefile



Makefile

```
EXTENSION    = example
MODULES      = example
DATA         = example--1.0.sql example.control

LDFLAGS=-lrt

PG_CONFIG ?= pg_config
PGXS = $(shell $(PG_CONFIG) --pgxs)
include $(PGXS)

```

Sql file and control file name

Create an empty extension

Compile C code

```
(base) marianaduarte@MacBook-Pro Template % make

(base) marianaduarte@MacBook-Pro Template % ls
Makefile  README.md  example--1.0.sql  example.c  example.control  example.o  example.so  test.sql
```

Creates .so (shared library) file

Create an empty extension

Add into lib/postgres

```
(base) marianaduarte@MacBook-Pro Template % sudo make install
/opt/homebrew/opt/coreutils/libexec/gnubin/mkdir -p '/opt/homebrew/pgsql/14.4/share/extension'
/opt/homebrew/opt/coreutils/libexec/gnubin/mkdir -p '/opt/homebrew/pgsql/14.4/share/extension'
/opt/homebrew/opt/coreutils/libexec/gnubin/mkdir -p '/opt/homebrew/pgsql/14.4/lib'
/opt/homebrew/opt/coreutils/libexec/gnubin/install -c -m 644 .//example.control
'/opt/homebrew/pgsql/14.4/share/extension/'
/opt/homebrew/opt/coreutils/libexec/gnubin/install -c -m 644 .//example--1.0.sql .//example.control
'/opt/homebrew/pgsql/14.4/share/extension/'
/opt/homebrew/opt/coreutils/libexec/gnubin/install -c -m 755  example.so '/opt/homebrew/pgsql/14.4/lib/'
```

Create an empty extension

If it has not been added to the library folder correctly, you will find this error:

```
marianaduarte=# create extension example;
ERROR:  could not open extension control file "/14/share/postgresql/extension/example.control": No such file or directory
```

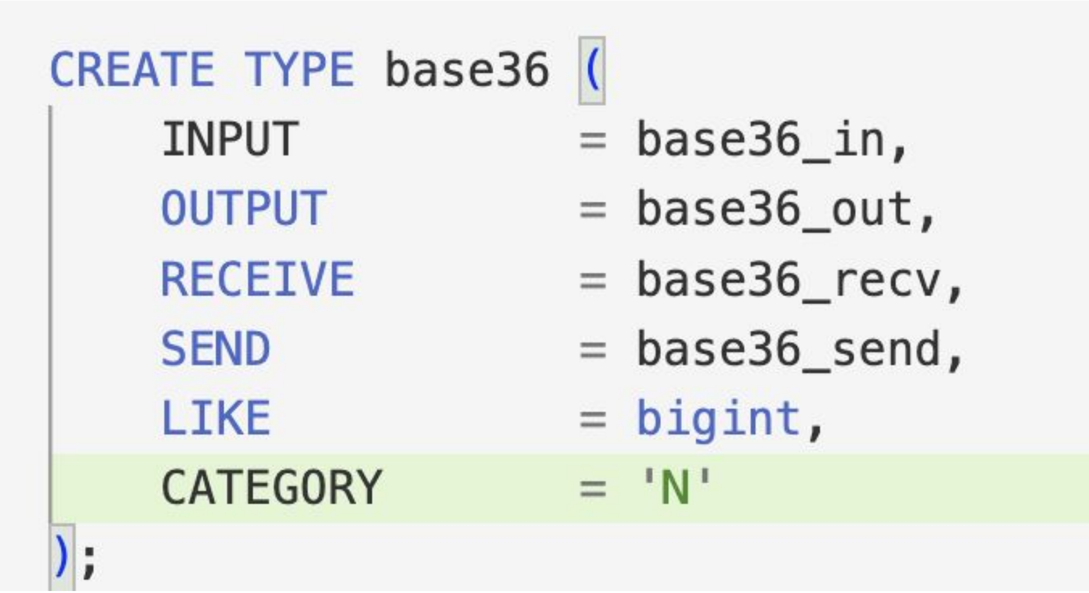
Show extensions \dx

```
marianaduarte=# \dx
                     List of installed extensions
   Name     | Version |   Schema   |  Description
------------+---------+------------+---------------------------------------------------
 example    | 1.0     | public     | empty example
 mobilitydb | 1.1.0   | public     | MobilityDB Extension
 plpgsql    | 1.0     | pg_catalog | PL/pgSQL procedural language
 postgis    | 3.2.2   | public     | PostGIS geometry and geography spatial types and functions
(4 rows)
```

Show extensions \dx

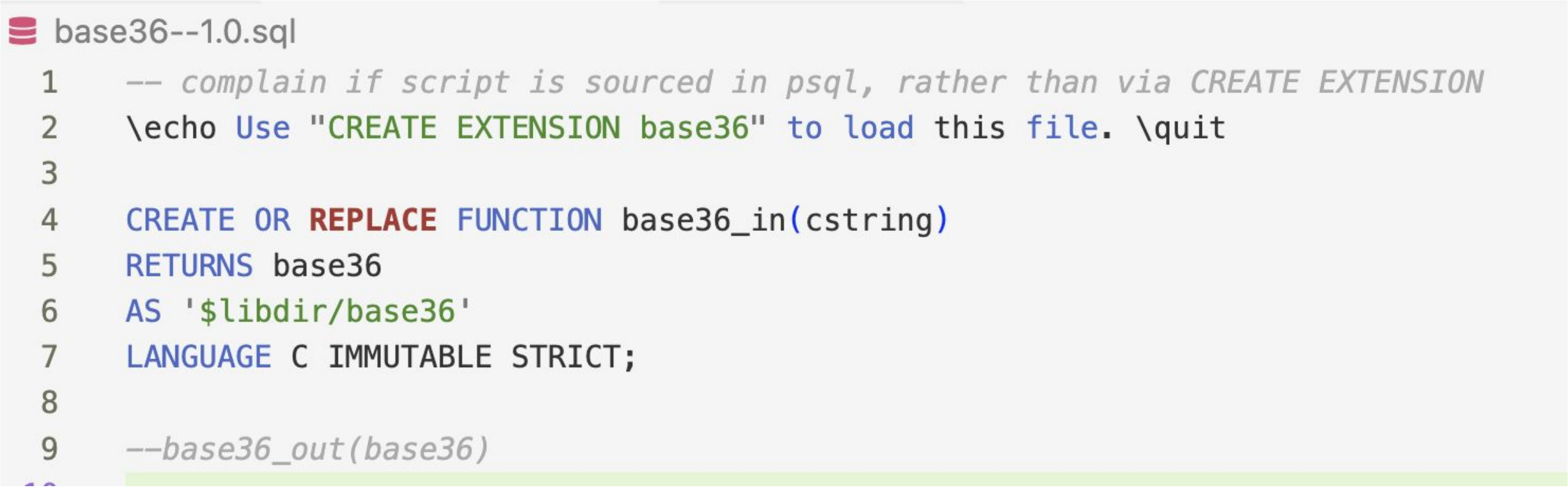
Create Base36 type

For this, we need to create functions for each paramenter



```
CREATE TYPE base36 (
    INPUT          = base36_in,
    OUTPUT         = base36_out,
    RECEIVE        = base36_recv,
    SEND           = base36_send,
    LIKE           = bigint,
    CATEGORY       = 'N'
);
```

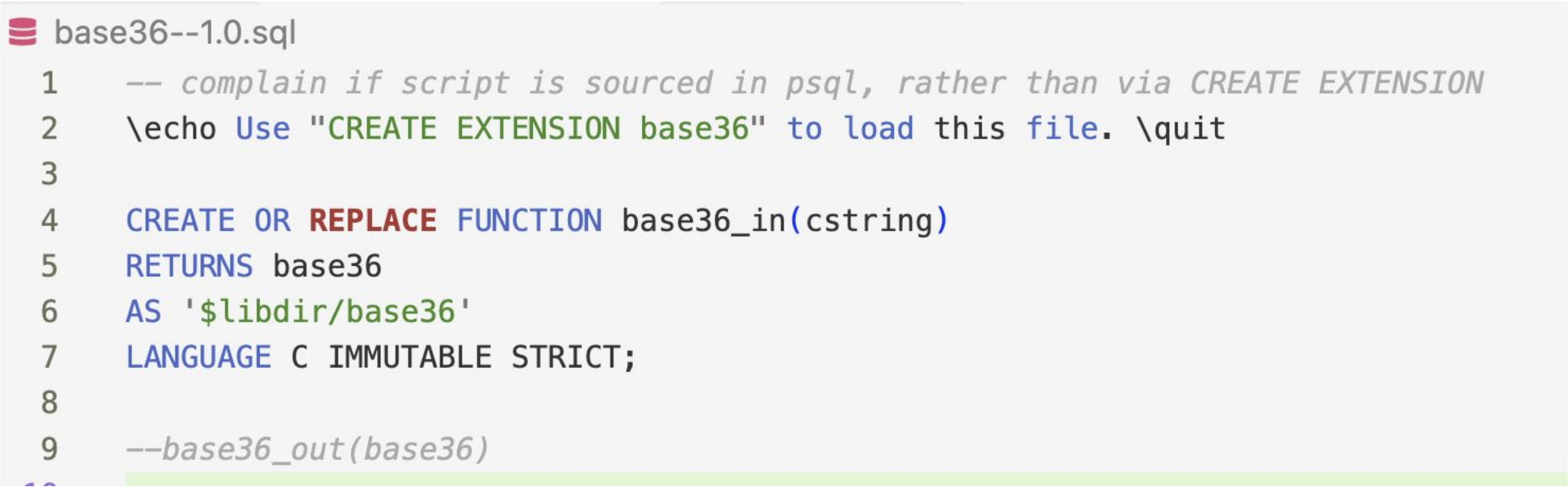
For example, creating a input function



base36--1.0.sql

```sql
-- complain if script is sourced in psql, rather than via CREATE EXTENSION
\echo Use "CREATE EXTENSION base36" to load this file. \quit

CREATE OR REPLACE FUNCTION base36_in(cstring)
RETURNS base36
AS '$libdir/base36'
LANGUAGE C IMMUTABLE STRICT;

--base36_out(base36)
```

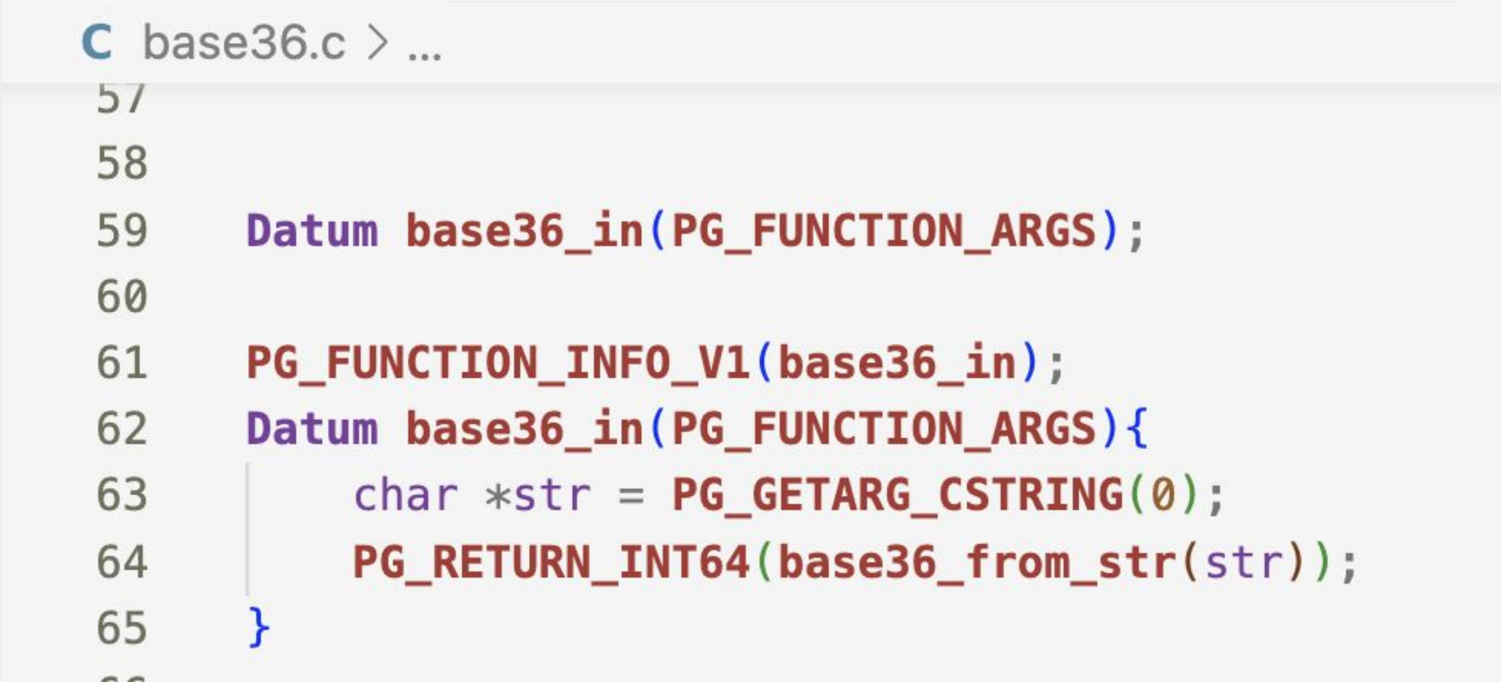
For example, creating a input function



base36--1.0.sql

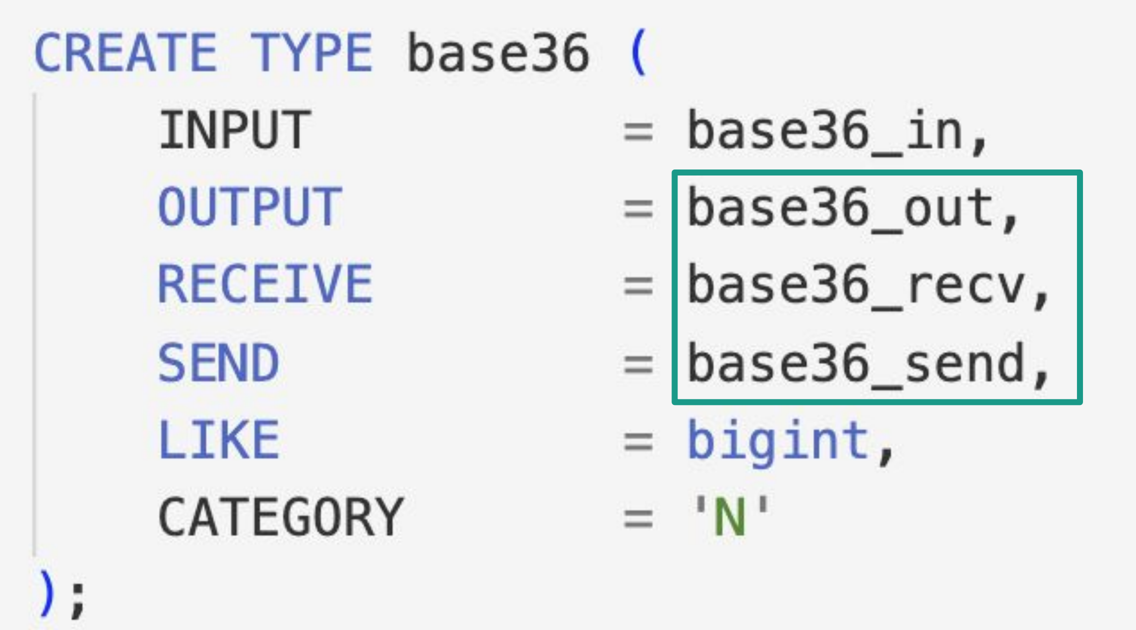
```sql
-- complain if script is sourced in psql, rather than via CREATE EXTENSION
\echo Use "CREATE EXTENSION base36" to load this file. \quit

CREATE OR REPLACE FUNCTION base36_in(cstring)
RETURNS base36
AS '$libdir/base36'
LANGUAGE C IMMUTABLE STRICT;

--base36_out(base36)
```

For example, creating a input function



```c


Datum base36_in(PG_FUNCTION_ARGS);

PG_FUNCTION_INFO_V1(base36_in);
Datum base36_in(PG_FUNCTION_ARGS){
    char *str = PG_GETARG_CSTRING(0);
    PG_RETURN_INT64(base36_from_str(str));
}
```

For now, create the missing functions:



```
CREATE TYPE base36 (
     INPUT          = base36_in,
     OUTPUT         = base36_out,
     RECEIVE        = base36_recv,
     SEND           = base36_send,
     LIKE           = bigint,
     CATEGORY       = 'N'
);
```