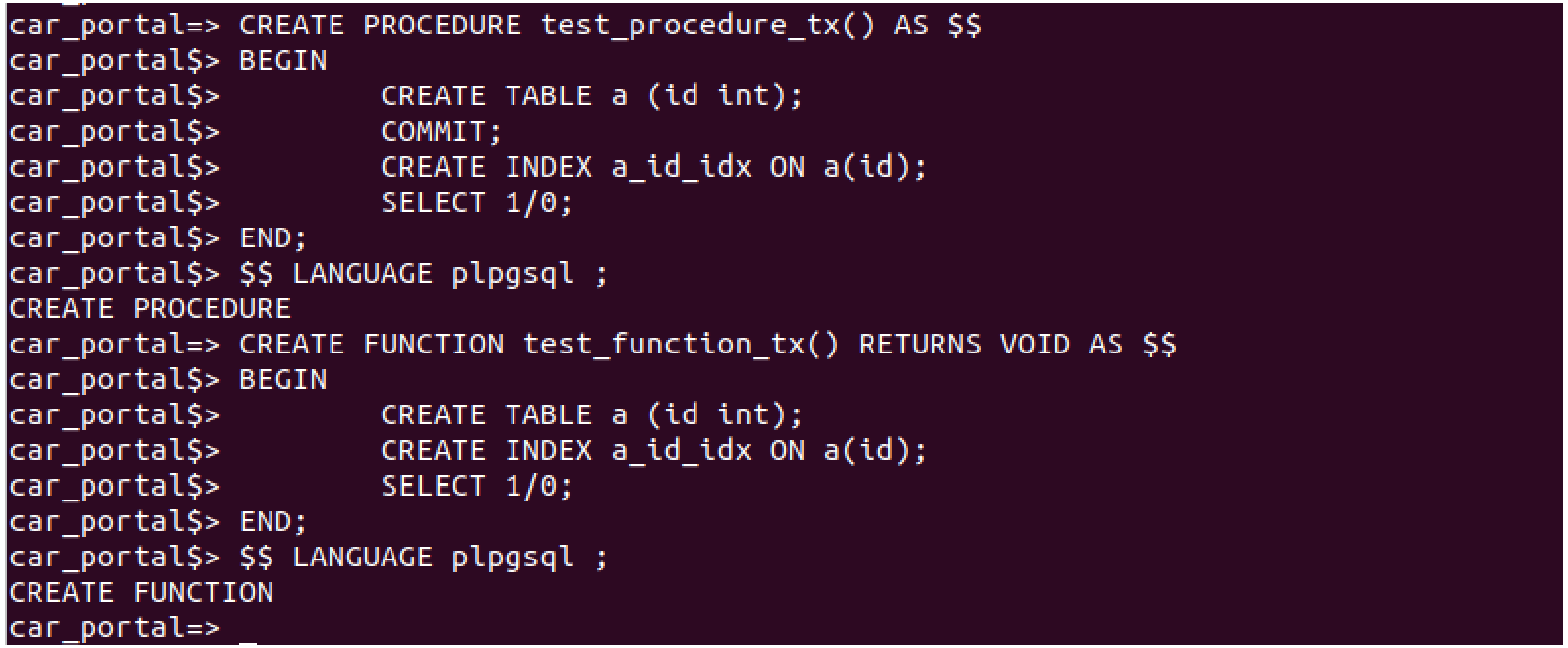
Functions and procedures – a comparison

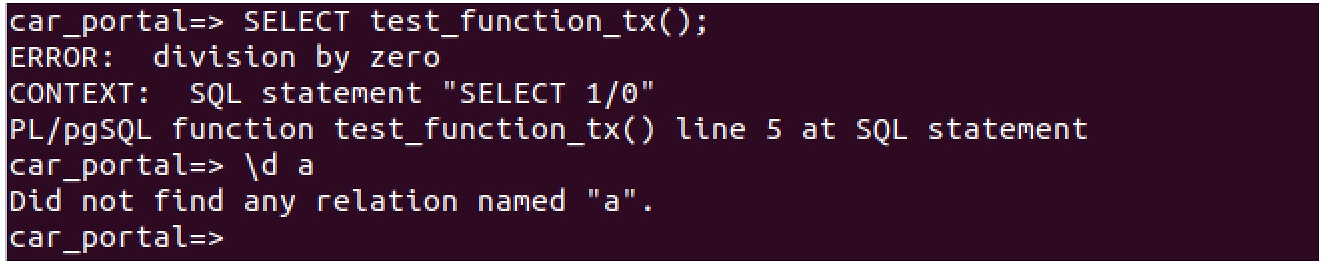
- In PostgreSQL 11, procedures were introduced.
- The main difference between functions and procedures is transaction control.
- Functions are explicitly run in a transaction, while a procedure can control transactions.
- In procedures, you can explicitly `COMMIT` and `ROLLBACK` a transaction.
- Functions can be called within SQL code, but procedures are executed via `CALL` statements.
- Finally, functions should have a return data type; if the function does not return anything, the `VOID` pseudo data type can be used.

- Procedures have great benefits.
- Let's suppose that we would like to index all foreign keys.
- In this case, using a function is not very practical, because the indexes are created when the function completely finishes the execution.
- So, if an exception occurs in the middle of the execution, then all of the work is lost.
- In a stored procedure, we can create one index at a time, and we can keep iterating through all indexes until the procedure finishes the execution.



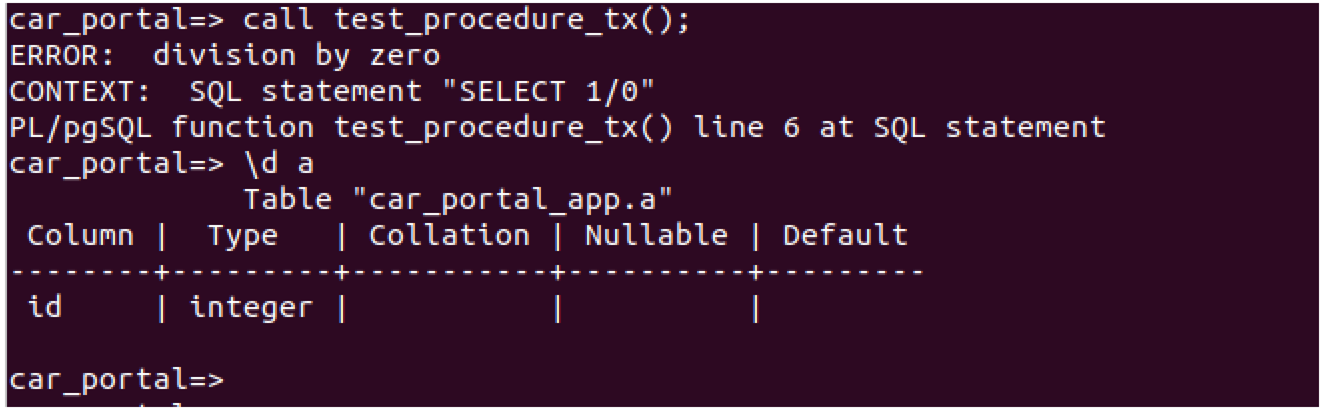
```
car_portal=> CREATE PROCEDURE test_procedure_tx() AS $$
car_portal$> BEGIN
car_portal$>         CREATE TABLE a (id int);
car_portal$>         COMMIT;
car_portal$>         CREATE INDEX a_id_idx ON a(id);
car_portal$>         SELECT 1/0;
car_portal$> END;
car_portal$> $$ LANGUAGE plpgsql ;
CREATE PROCEDURE
car_portal=> CREATE FUNCTION test_function_tx() RETURNS VOID AS $$
car_portal$> BEGIN
car_portal$>         CREATE TABLE a (id int);
car_portal$>         CREATE INDEX a_id_idx ON a(id);
car_portal$>         SELECT 1/0;
car_portal$> END;
car_portal$> $$ LANGUAGE plpgsql ;
CREATE FUNCTION
car_portal=> _
```

- The following code executes the `test_function_tx()` function:



```
car_portal=> SELECT test_function_tx();
ERROR:  division by zero
CONTEXT:  SQL statement "SELECT 1/0"
PL/pgSQL function test_function_tx() line 5 at SQL statement
car_portal=> \d a
Did not find any relation named "a".
car_portal=>
```

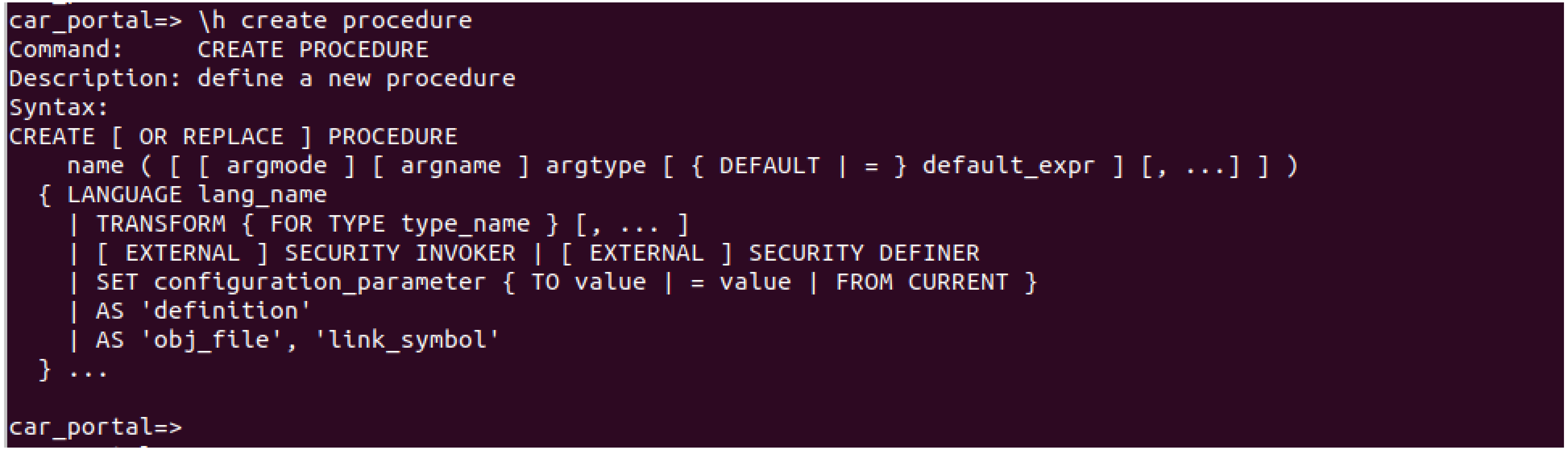
- The following code executes the `test_procedure_tx()` procedure :



```
car_portal=> call test_procedure_tx();
ERROR:  division by zero
CONTEXT:  SQL statement "SELECT 1/0"
PL/pgSQL function test_procedure_tx() line 6 at SQL statement
car_portal=> \d a
             Table "car_portal_app.a"
 Column |  Type   | Collation | Nullable | Default
--------+---------+-----------+----------+---------
 id     | integer |           |          |

car_portal=>
```

- The syntax for creating functions and procedures is quite similar, and they share some common features, such as configuration parameters, authorization parameters, control statements, and iteration statements.
- The synopsis for creating a procedure is as follows:



```
car_portal=> \h create procedure
Command:     CREATE PROCEDURE
Description: define a new procedure
Syntax:
CREATE [ OR REPLACE ] PROCEDURE
    name ( [ [ argmode ] [ argname ] argtype [ { DEFAULT | = } default_expr ] [, ...] ] )
  { LANGUAGE lang_name
    | TRANSFORM { FOR TYPE type_name } [, ... ]
    | [ EXTERNAL ] SECURITY INVOKER | [ EXTERNAL ] SECURITY DEFINER
    | SET configuration_parameter { TO value | = value | FROM CURRENT }
    | AS 'definition'
    | AS 'obj_file', 'link_symbol'
  } ...

car_portal=>
```