# Write script that fails or passes entirely – **myboys.sql**

CREATE TABLE bts\_boys (

id SERIAL PRIMARY KEY,

name VARCHAR(50) UNIQUE,

age INT

);

INSERT INTO bts\_boys (name, age) VALUES

('Yoongi', 30),

('JK', 27),

('Namjoon', 29);

SELECT \* FROM bts\_boys;

A computer screen shot of a computer code

Description automatically generated

# Write a script that exits on first error – **myboys\_exit.sql**

set ON\_ERROR\_STOP on

CREATE TABLE bts\_boys (

id SERIAL PRIMARY KEY,

name VARCHAR(50) UNIQUE,

age INT

);

INSERT INTO bts\_boys (name, age) VALUES

('Yoongi', 30),

('JK', 27),

('Namjoon', 29);

SELECT \* FROM bts\_boys;

A computer screen shot of white text

Description automatically generated

# write a script that uses psql variables – **myboys\_var.sql**

-- Define psql variables

\set username 'Yoongi'

\set userage 30

-- Create a table using variables

CREATE TABLE user\_info (

id SERIAL PRIMARY KEY,

name VARCHAR(50),

age INT

);

-- Insert data using variables

INSERT INTO user\_info (name, age) VALUES (:'username', :'userage');

-- Select data using variables

SELECT \* FROM user\_info;

A computer screen shot of white text

Description automatically generated

# placing query output into psql variables – **myboys\_q.sql**

-- Define psql variables

\set username 'Namjoon'

-- Create a table

CREATE TABLE user\_info (

id SERIAL PRIMARY KEY,

name VARCHAR(50),

age INT

);

-- Insert some data

INSERT INTO user\_info (name, age) VALUES ('Namjoon', 29), ('JK', 27), ('Yoongi', 30);

-- Query to retrieve age based on the username

\set userage :username

SELECT age INTO :userage FROM user\_info WHERE name = :'username';

-- Display the result

\echo User Age: :userage;

A computer screen shot of white text

Description automatically generated

# write a conditional psql script – **myboys\_con.sql**

-- Check if the table exists

\set table\_exists (SELECT EXISTS (SELECT 1 FROM information\_schema.tables WHERE table\_name = 'user\_info'));

-- Conditional block

\if :table\_exists

-- If the table exists, select data from it

SELECT \* FROM user\_info;

\else

-- If the table doesn't exist, create it and insert data

CREATE TABLE user\_info (

id SERIAL PRIMARY KEY,

name VARCHAR(50),

age INT

);

INSERT INTO user\_info (name, age) VALUES

('Ebony', 30),

('Yoongi', 31),

('Jin', 32);

-- Display a message

\echo Table user\_info created and data inserted.

\endif

A computer screen shot of a program

Description automatically generated

# investigating a psql error

# performing actions on many tables – **myscript.sql**

DO $$

DECLARE

loop\_table\_name text;

BEGIN

-- Loop through tables in the 'public' schema

FOR loop\_table\_name IN (SELECT table\_name FROM information\_schema.tables WHERE table\_schema = 'public' AND table\_type = 'BASE TABLE')

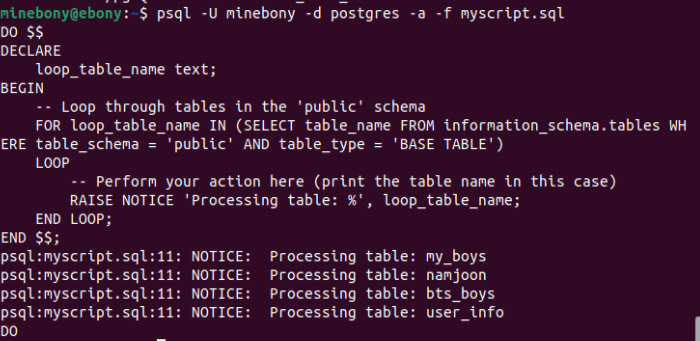
LOOP

-- Perform your action here (print the table name in this case)

RAISE NOTICE 'Processing table: %', loop\_table\_name;

END LOOP;

END $$;



# adding/removing columns of a table – **myscript\_add.sql**

-- Adding a new column to the table

ALTER TABLE public.my\_boys

ADD COLUMN songs VARCHAR(50);

-- Displaying the updated table structure

\dt+ public.my\_boys

-- Inserting data into the new column

UPDATE public.my\_boys

SET songs = 'New Data';

-- Displaying the updated data

SELECT \* FROM public.my\_boys;

-- Removing the added column

ALTER TABLE public.my\_boys

DROP COLUMN songs;

-- Displaying the final table structure

\dt+ public.my\_boys

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

# changing the data type of a column – **myscript\_data.sql**

-- Display the current structure of the table

\dt+ public.my\_boys

-- Changing the data type of the column

ALTER TABLE public.my\_boys

ALTER COLUMN songs TYPE INTEGER;

-- Display the updated structure of the table

\dt+ public.my\_boys

-- Displaying the data in the table

SELECT \* FROM public.my\_boys;

A computer screen shot of a computer program

Description automatically generated

# changing the definition of a data type

# adding/removing schemas – **myscript\_sch.sql**

-- Create a new data type with the desired definition

CREATE TYPE min\_new AS (

-- Define the new structure of the type

field1 INTEGER,

field2 VARCHAR(50)

);

-- Alter the table to use the new data type for the column

ALTER TABLE public.my\_boys

ALTER COLUMN songs TYPE min\_new

USING songs::min\_new;

-- Display the updated structure of the table

\dt+ public.my\_boys

-- Displaying the data in the table

SELECT \* FROM public.my\_boys;

A computer screen shot of white text

Description automatically generated

# adding/removing tablespaces – **myscript\_tbl.sql**

-- Adding a new tablespace named 'new\_tablespace' at '/home/minebony'

CREATE TABLESPACE new\_tablespace LOCATION '/home/minebony';

-- Displaying the list of tablespaces to verify the addition

\db

-- Removing an existing tablespace named 'old\_tablespace'

DROP TABLESPACE IF EXISTS old\_tablespace;

-- Displaying the list of tablespaces to verify the removal

\db

A computer screen shot of a computer program

Description automatically generated

# moving objects between schemas

# moving objects between tablespaces – **myscript\_mve.sql**

-- Moving an index named 'my\_index' to a different tablespace 'new\_tablespace'

ALTER INDEX my\_index SET TABLESPACE new\_tablespace;

A screenshot of a computer screen

Description automatically generated

# accessing objects in other PostgreSQL databases

# accessing objects in other foreign databases

# updating views – myboys\_vw.sql

-- Creating a sample table

CREATE TABLE my\_boys2 (

id SERIAL PRIMARY KEY,

name VARCHAR(50),

age INT

);

-- Inserting some sample data

INSERT INTO my\_boys2 (name, age) VALUES ('Yoongi', 25), ('Jin', 30), ('Hobi', 22);

-- Creating a view based on the table

CREATE VIEW my\_view AS

SELECT \* FROM my\_boys2;

-- Displaying the data in the view

SELECT \* FROM my\_view;

-- Updating the view (changing the age of 'Jin')

UPDATE my\_view

SET age = 26

WHERE name = 'Jin';

-- Displaying the updated data in the view

SELECT \* FROM my\_view;

A computer screen shot of a computer program

Description automatically generated

A computer screen shot of white text

Description automatically generated

# using materialized views – **myscript\_mat.sql**

-- Creating a materialized view named 'my\_materialized\_view'

CREATE MATERIALIZED VIEW my\_materialized\_view AS

SELECT id, name, age FROM my\_boys;

A screenshot of a computer code

Description automatically generated