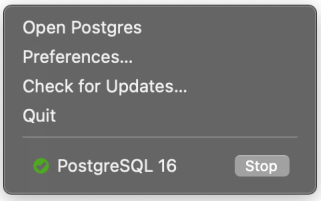
112-1 Database HW1

111550057 莊婷馨

A. The process of creating the “lego” database.

1. Start PostgreSQL server running.

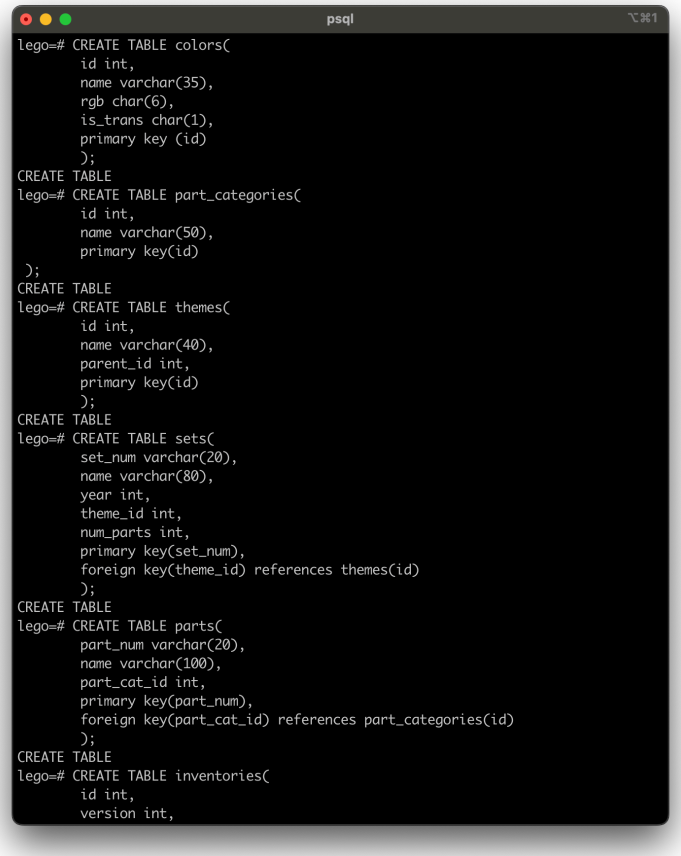


2. Open terminal. Use command “ `psql` ” to connect to PostgreSQL.
3. Use command “ `CREATE DATABASE lego;` ” to create a database named lego.
4. Use command “ `\c lego` ” to switch to the database named lego.

B. The process of importing eight required .csv files into lego database.

1. Create relations. Use command ” `CREATE TABLE` “, followed by the name of the relation, multiple attributes and its datatype.

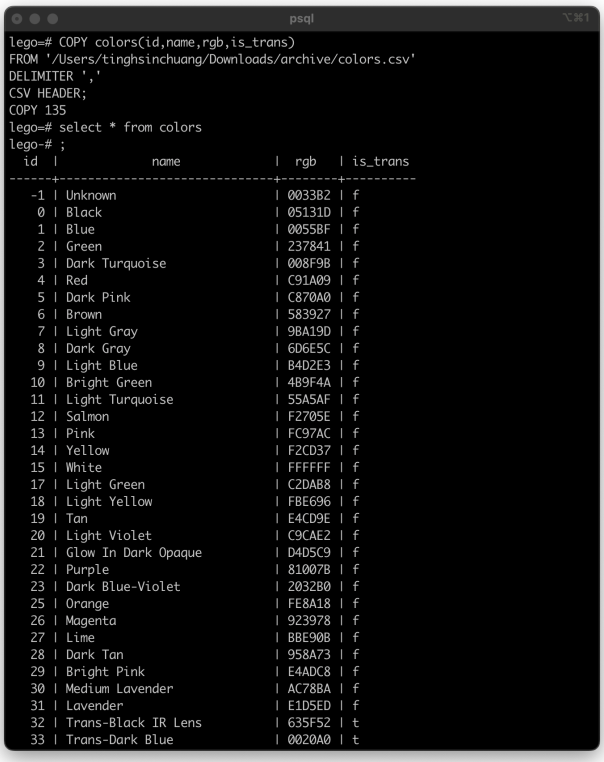
- `colors`
  - Use “ id ” as primary key because every color has an unique ID.
- `part_categories`
  - Use “ id ” as primary key because every category has an unique ID.
- `themes`
  - Use “ id ” as primary key because every theme has an unique ID.
- `sets`
  - Use “ set_num ” as primary key because every set has an unique set_num.
  - Use “ theme_id ” as foreign key because every theme of a set has to exist in the theme relation.
- `parts`
  - Use “ part_num ” as primary key because every part has an unique part_num.
  - Use “ part_cat_id ” as foreign key because every category of a part has to exist in the part_categories relation.

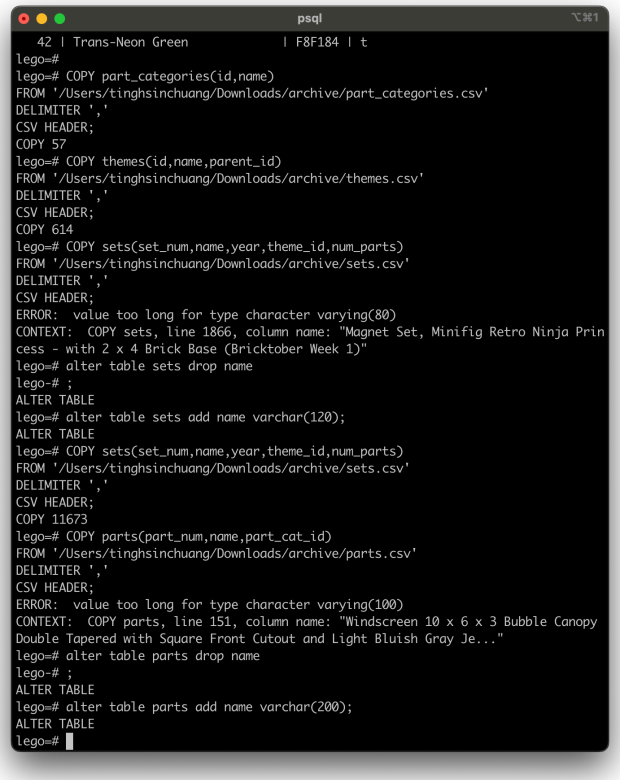


- `inventories`
  - Use “ id ” as primary key because every inventory has an unique ID.
- `inventory_sets`
  - Use “ inventory_id ” and “ set_num ” as primary key because its the minimal superkey to identify an unique tuple.
  - Use “ inventory_id ” as foreign key because every inventory has to exist in the inventories relation.
  - Use “ set_num ” as foreign key because every set has to exist in the sets relation.
- `inventory_parts`
  - Use “ inventory_id ”, “ part_num ”, “ color_id ”, “ is_spare ” as primary key because its the minimal superkey to identify an unique tuple.
  - Use “ inventory_id ” as foreign key because every inventory has to exist in the inventories relation.
  - Use “ color_id ” as foreign key because every color has to exist in the colors relation.

2. Use command “ \d “ to check all relations
3. Use command as below to copy datas from the .csv files.

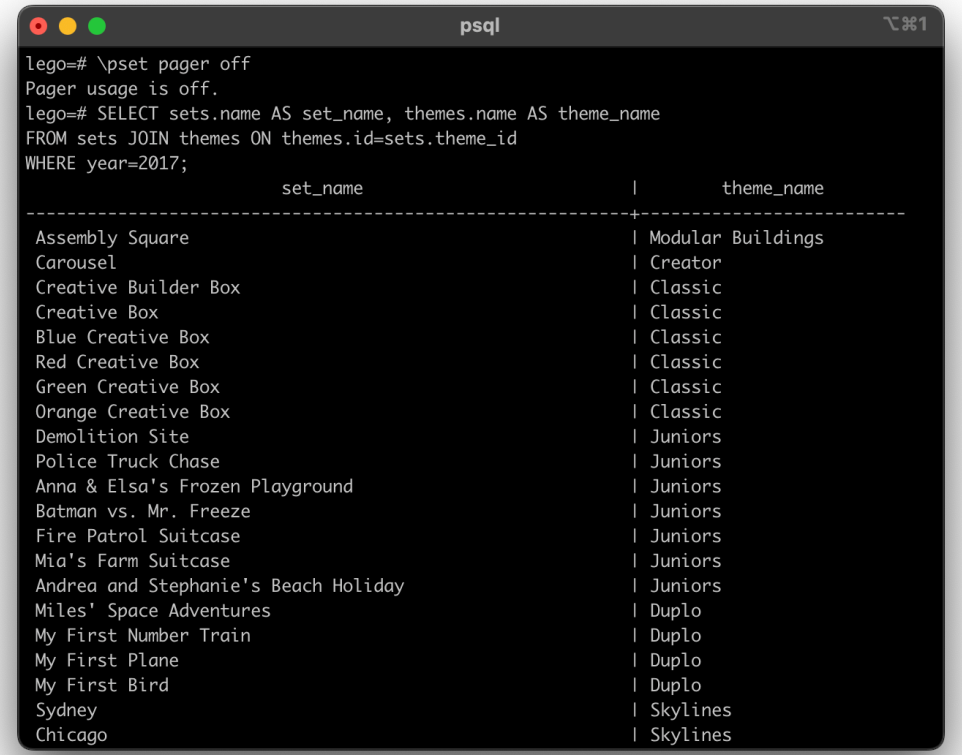
```
COPY <tablename>(<columnname1>,<columnname2>,...)
FROM '<path>'
DELIMITER ','
CSV HEADER;
```





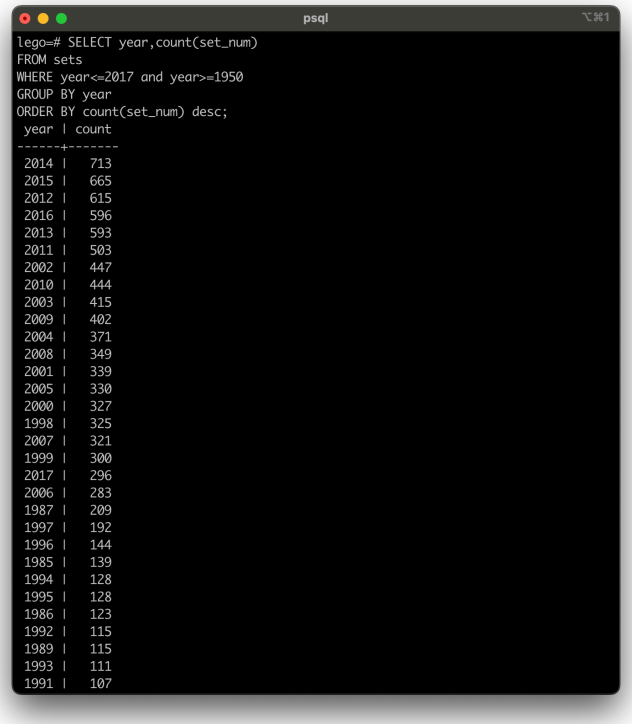
4. If the datatype doesn't match the actual datas, use command " `alter table <relation_name> drop <attribute_name>;` " and " `alter table <relation_name> add <attribute_name>;` " to change attribute datatype.

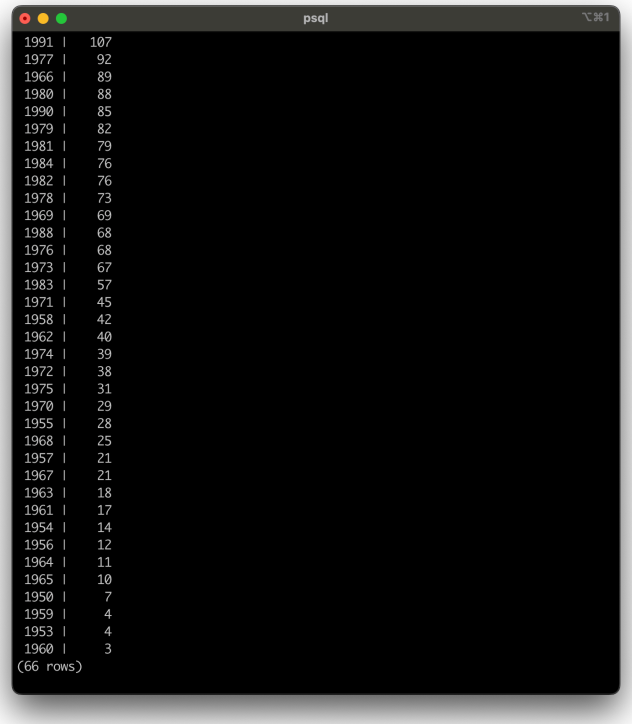
C. The SQL statements and output results of 4a.



- total number of rows: 296 rows
- Full result: https://drive.google.com/file/d/1AVkKkYWPlGlAzQEtTDluuNa9sCnv9jUw/view

D. The SQL statements and output results of 4b.





- total number of rows: 66 rows
- Full result: https://drive.google.com/file/d/1UodS3exbAqKYjByCAUtYhM_x_PPqsWSj/view

E. The SQL statements and output results of 4c.

```
psql                                                    ⌥⌘1
lego=# WITH scthemes(theme_name,setcount) AS
                        (SELECT themes.name, count(set_num) AS setcount
                         FROM sets
                         JOIN themes ON sets.theme_id=themes.id
                         GROUP BY themes.id),
                   mptheme(setcount) AS
                        (SELECT max(setcount)
                         FROM scthemes)
SELECT theme_name
FROM mptheme,scthemes
WHERE scthemes.setcount=mptheme.setcount;
 theme_name
------------
 Gear
(1 row)
```

- total number of rows: 1 row

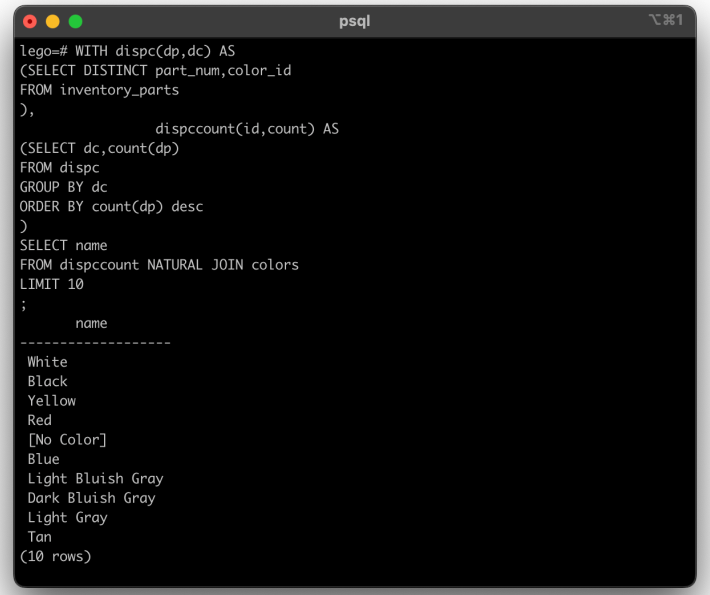
F. The SQL statements and output results of 4d.

```
psql                                                    ⌥⌘1
lego=# WITH avgpart(theme_id,avg_part_num) AS
(SELECT theme_id,avg(num_parts)
 FROM sets
 GROUP BY theme_id
)
SELECT themes.name,avg_part_num
FROM avgpart
JOIN themes ON theme_id=themes.id
ORDER BY avg_part_num;
             name              |      avg_part_num
-------------------------------+------------------------
 Wooden Box Set                | -1.0000000000000000000
 Mindstorms                    |  0.0000000000000000000
 Train                         |  0.0000000000000000000
 Samsonite                     |  0.0000000000000000000
 Key Chain                     |  0.1818181818181818181
 Technic                       |  1.0000000000000000000
 Imperial Guards               |  1.0000000000000000000
 Supplemental                  |      1.8000000000000000
 Power Functions               |      1.8823529411764706
 Control Lab                   |      2.0000000000000000
 Classic Town                  |      2.4000000000000000
 Star Wars                     |      2.5000000000000000
 Planet Series 1               |      3.0000000000000000
 Western                       |      3.0000000000000000
 Adventurers                   |      3.0000000000000000
 Value Packs                   |      3.1666666666666667
 Minifig Pack                  |      3.5000000000000000
 Train                         |      3.5753424657534247
 4 Juniors                     |      4.0000000000000000
 Indiana Jones                 |      4.0000000000000000
 Dinosaurs                     |      4.0000000000000000
 Clikits                       |      5.0000000000000000
 Series 14 Minifigures         |      5.3684210526315789
 The Hobbit                    |      5.5000000000000000
 DFB Minifigures               |      5.6470588235294118
 Food & Drink                  |      6.0000000000000000
```

```
psql                                                    ⌥⌘1
 RC Train                      |    595.0000000000000000
 Space                         |    598.0000000000000000
 Construction                  |    600.4426229508196721
 Star Wars                     |    618.0000000000000000
 My Own Creation               |    629.0000000000000000
 Universal Building Set        |    649.7692307692307692
 Fire                          |    664.2500000000000000
 Farm                          |    665.0000000000000000
 Building                      |    667.0000000000000000
 Star Wars Episode 2           |    675.0000000000000000
 Model Team                    |    683.1875000000000000
 Model                         |    693.6363636363636364
 eLAB                          |    713.0000000000000000
 Lamborghini                   |    770.0000000000000000
 Fire                          |    771.0000000000000000
 Williams F1                   |    787.5000000000000000
 Mindstorms                    |    850.4000000000000000
 The Two Towers                |    869.2000000000000000
 Order of the Phoenix          |    904.0000000000000000
 Building                      |    919.9090909090909091
 Traffic                       |   1001.5384615384615385
 Recreation                    |   1066.0000000000000000
 Star Wars Episode 3           |   1084.0000000000000000
 Harbor                        |   1279.0000000000000000
 Mini                          |   1358.0000000000000000
 FIRST LEGO League             |   1387.0000000000000000
 Creator                       |   1437.0000000000000000
 Ultimate Collector Series     |   1455.5000000000000000
 Sculptures                    |   1716.6956521739130435
 Town Plan                     |   2017.0000000000000000
 Mosaic                        |   2099.8571428571428571
 Ultimate Collector Series     |   2130.0000000000000000
 Star Wars Episode 4/5/6       |   2199.7058823529411765
 Modular Buildings             |   2350.5833333333333333
 Disney                        |   4060.0000000000000000
(575 rows)
```

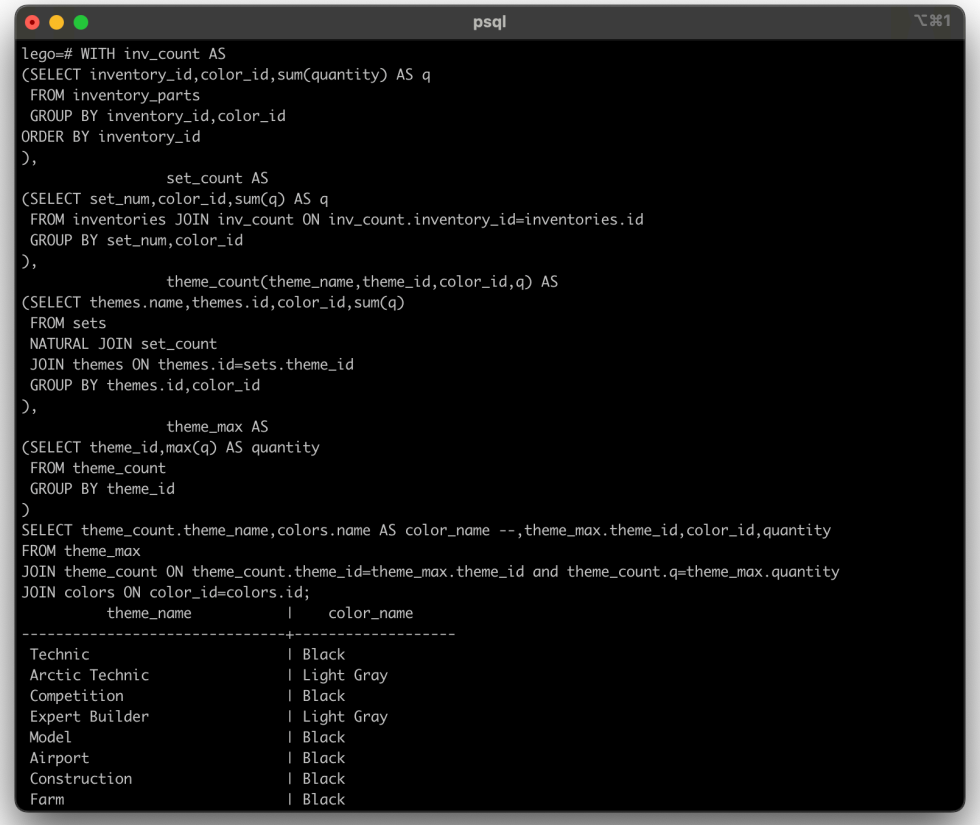
- total number of rows: 575 rows
- Full result: https://drive.google.com/file/d/1_hJwX7mDb8D7r91AiptDnWSF0sQE07R-/view

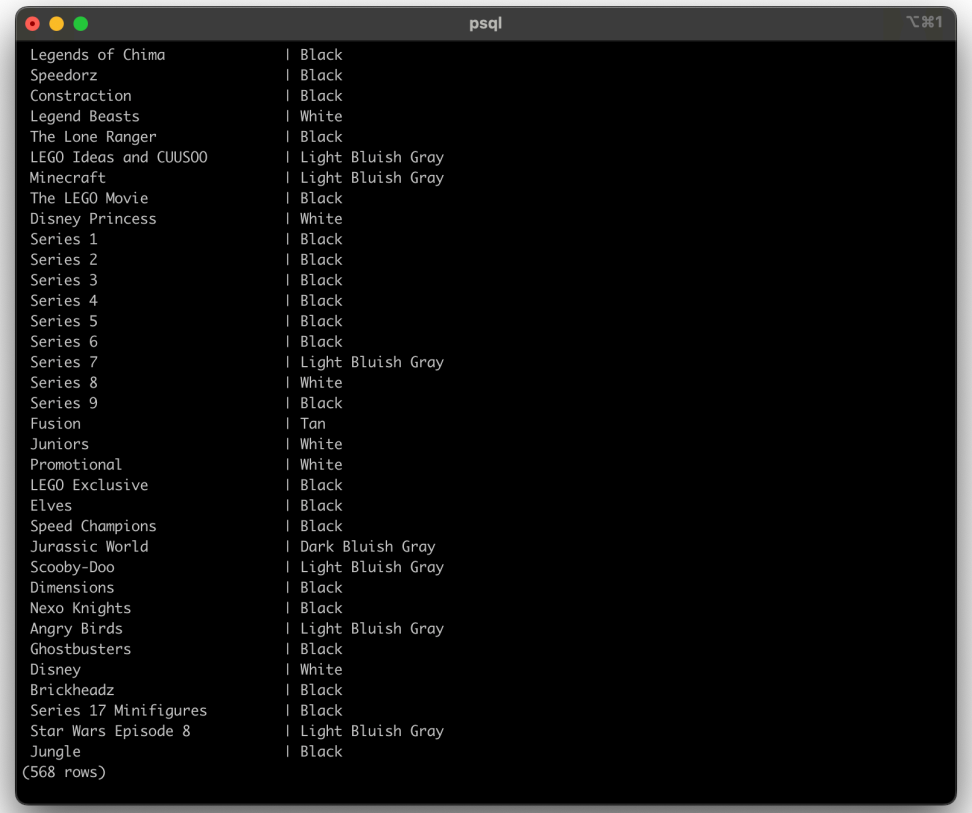
G. The SQL statements and output results of 4e.



- total number of rows: 10 rows

H. The SQL statements and output results of 4f.





- total number of rows: 568 rows
- Full result: https://drive.google.com/file/d/1Ti0HC-ITKKSqFsQ20_LKxCIJTn1xBypU/view