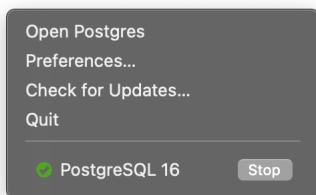


112-1 Database HW1

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.

```
lego=# CREATE TABLE colors(
    id int,
    name varchar(35),
    rgb char(6),
    is_trans char(1),
    primary key (id)
);
CREATE TABLE
lego=# CREATE TABLE part_categories(
    id int,
    name varchar(50),
    primary key(id)
);
CREATE TABLE
lego=# CREATE TABLE themes(
    id int,
    name varchar(40),
    parent_id int,
    primary key(id)
);
CREATE TABLE
lego=# CREATE TABLE sets(
    set_num varchar(20),
    name varchar(80),
    year int,
    theme_id int,
    num_parts int,
    primary key(set_num),
    foreign key(theme_id) references themes(id)
);
CREATE TABLE
lego=# CREATE TABLE parts(
    part_num varchar(20),
    name varchar(100),
    part_cat_id int,
    primary key(part_num),
    foreign key(part_cat_id) references part_categories(id)
);
CREATE TABLE
lego=# CREATE TABLE inventories(
    id int,
    version int,
```

- 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.

```

logo=# CREATE TABLE inventories(
    id int,
    version int,
    set_num varchar(20),
    primary key(id),
    foreign key(set_num) references sets(set_num)
);
CREATE TABLE
logo=# CREATE TABLE inventory_sets(
    inventory_id int,
    set_num varchar(20),
    quantity int,
    primary key(inventory_id, set_num),
    foreign key(inventory_id) references inventories(id),
    foreign key(set_num) references sets(set_num)
);
CREATE TABLE
logo=# CREATE TABLE inventory_parts(
    inventory_id int,
    part_num varchar(20),
    color_id int,
    quantity int,
    is_spare char(1),
    primary key(inventory_id, part_num, color_id, is_spare),
    foreign key(inventory_id) references inventories(id),
    foreign key(color_id) references colors(id)
);
CREATE TABLE
logo=# \d
      List of relations
 Schema | Name   | Type  | Owner
-----+-----+-----+-----+
 public | colors | table | tinghsinchuang
 public | inventories | table | tinghsinchuang
 public | inventory_parts | table | tinghsinchuang
 public | inventory_sets | table | tinghsinchuang
 public | part_categories | table | tinghsinchuang
 public | parts | table | tinghsinchuang
 public | sets | table | tinghsinchuang
 public | themes | table | tinghsinchuang
(8 rows)

```

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;

```

```

logo=# COPY colors(id,name,rgb,is_trans)
FROM '/Users/tinghsinchuang/Downloads/archive/colors.csv'
DELIMITER ','
CSV HEADER;
COPY 135
logo# select * from colors
logo# ;
   id |      name      |   rgb   | is_trans
-----+-----+-----+-----+
-1 | Unknown       | 003B2 | f
  0 | Black          | 05131D | f
  1 | Blue           | 0055BF | f
  2 | Green          | 237841 | f
  3 | Dark Turquoise | 008F9B | f
  4 | Red            | C91A09 | f
  5 | Dark Pink      | C870A0 | f
  6 | Brown          | 583927 | f
  7 | Light Gray     | 9BA19D | f
  8 | Dark Gray      | 606E5C | f
  9 | Light Blue     | B4D2E3 | f
 10 | Bright Green   | 489F4A | f
 11 | Light Turquoise | 55A5AF | f
 12 | Salmon          | F2705E | f
 13 | Pink            | FC97AC | f
 14 | Yellow          | F2CD37 | f
 15 | White           | FFFFFF | f
 17 | Light Green     | C2DAB8 | f
 18 | Light Yellow    | FBE696 | f
 19 | Tan              | E4CD9E | f
 20 | Light Violet    | C9CAE1 | f
 21 | Glow In Dark Opaque | D4D5C9 | f
 22 | Purple          | 81007B | f
 23 | Dark Blue-Violet | 2032B0 | f
 25 | Orange          | FE8A18 | f
 26 | Magenta          | 923978 | f
 27 | Lime             | BBE90B | f
 28 | Dark Tan         | 958A73 | f
 29 | Bright Pink      | E4ADC8 | f
 30 | Medium Lavender | AC78BA | f
 31 | Lavender         | E1D5ED | f
 32 | Trans-Black IR Lens | 635F52 | t
 33 | Trans-Dark Blue  | 0020A0 | t

```

```

42 | Trans-Neon Green          | F8F184 | t
logo# 
logo# COPY part_categories(id,name)
FROM '/Users/tinghsinchuang/Downloads/archive/part_categories.csv'
DELIMITER ','
CSV HEADER;
COPY 57
logo# COPY themes(id,name,parent_id)
FROM '/Users/tinghsinchuang/Downloads/archive/themes.csv'
DELIMITER ','
CSV HEADER;
COPY 614
logo# COPY sets(set_num,name,year,theme_id,num_parts)
FROM '/Users/tinghsinchuang/Downloads/archive/sets.csv'
DELIMITER ','
CSV HEADER;
ERROR: value too long for type character varying(80)
CONTEXT: COPY sets, line 1866, column name: "Magnet Set, Minifig Retro Ninja Printcess - with 2 x 4 Brick Base (Bricktober Week 1)"
logo# alter table sets drop name
logo# ;
ALTER TABLE
logo# alter table sets add name varchar(120);
ALTER TABLE
logo# COPY sets(set_num,name,year,theme_id,num_parts)
FROM '/Users/tinghsinchuang/Downloads/archive/sets.csv'
DELIMITER ','
CSV HEADER;
COPY 11673
logo# COPY parts(part_num,name,part_cat_id)
FROM '/Users/tinghsinchuang/Downloads/archive/parts.csv'
DELIMITER ','
CSV HEADER;
ERROR: value too long for type character varying(100)
CONTEXT: COPY parts, line 151, column name: "Windscreen 10 x 6 x 3 Bubble Canopy Double Tapered with Square Front Cutout and Light Bluish Gray Je..."
logo# alter table parts drop name
logo# ;
ALTER TABLE
logo# alter table parts add name varchar(200);
ALTER TABLE
logo# 

```

```

CONTEXT: COPY parts, line 151, column name: "Windscreen 10 x 6 x 3 Bubble Canopy
Double Tapered with Square Front Cutout and Light Bluish Gray Je...""
lego=# alter table parts drop name
lego=# ;
ALTER TABLE
lego=# alter table parts add name varchar(200);
ALTER TABLE
lego=# COPY parts(part_num,name,part_cat_id)
FROM '/Users/tinghsinchuang/Downloads/archive/part.csv'
DELIMITER ','
CSV HEADER;
ERROR: value too long for type character varying(200)
CONTEXT: COPY parts, line 10109, column name: "Minifig Head with Hollow Stud and
Dual Side Print - Side A - Light Flesh Face with White and Gray Ey...""
lego=# alter table parts drop name
lego=# ;
ALTER TABLE
lego=# alter table parts add name varchar(300);
ALTER TABLE
lego=# COPY parts(part_num,name,part_cat_id)
FROM '/Users/tinghsinchuang/Downloads/archive/part.csv'
DELIMITER ','
CSV HEADER;
COPY 25993
lego=# COPY inventories(id,version,set_num)
FROM '/Users/tinghsinchuang/Downloads/archive/inventories.csv'
DELIMITER ','
CSV HEADER;
COPY 11681
lego=# COPY inventory_sets(inventory_id,set_num,quantity)
FROM '/Users/tinghsinchuang/Downloads/archive/inventory_sets.csv'
DELIMITER ','
CSV HEADER;
COPY 2846
lego=# COPY inventory_parts(inventory_id,part_num,color_id,quantity,is_spare)
FROM '/Users/tinghsinchuang/Downloads/archive/inventory_parts.csv'
DELIMITER ','
CSV HEADER;
COPY 580251
lego=#

```

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.

```

lego=# \pset pager off
Pager usage is off.
lego=# SELECT sets.name AS set_name, themes.name AS theme_name
FROM sets JOIN themes ON themes.id=sets.theme_id
WHERE year=2017;

```

set_name	theme_name
Assembly Square	Modular Buildings
Carousel	Creator
Creative Builder Box	Classic
Creative Box	Classic
Blue Creative Box	Classic
Red Creative Box	Classic
Green Creative Box	Classic
Orange Creative Box	Classic
Demolition Site	Juniors
Police Truck Chase	Juniors
Anna & Elsa's Frozen Playground	Juniors
Batman vs. Mr. Freeze	Juniors
Fire Patrol Suitcase	Juniors
Mia's Farm Suitcase	Juniors
Andrea and Stephanie's Beach Holiday	Juniors
Miles' Space Adventures	Duplo
My First Number Train	Duplo
My First Plane	Duplo
My First Bird	Duplo
Sydney	Skylines
Chicago	Skylines

Mighty Micros: Spider-Man vs. Scorpion	Spider-Man
Mighty Micros: Iron Man vs. Thanos	Super Heroes
Mighty Micros: Wolverine vs. Magneto	Marvel
Wonder Woman" Warrior Battle	Super Heroes
Captain America Jet Pursuit	Marvel
Iron Man: Detroit Steel Strikes	Marvel
Hulk vs. Red Hulk	Marvel
Ravager Attack	Guardians of the Galaxy
Ayesha's Revenge	Guardians of the Galaxy
The Milano vs. The Abilisk	Guardians of the Galaxy
ATM Heist Battle	Marvel
Beware the Vulture	Marvel
Elves Roblin Bag Charm	Elves
NINJAGO Accessory Set	Ninjago
Flash Speeder	Star Wars
Policeman and Crook Foil Pack	Police
Bear in Ice Cave foil pack	Friends
The LEGO Batman Movie: Chaos in Gotham City	Books
Clay and Training Stand	Nexo Knights
Robin	Nexo Knights
Kanan Jarrus foilpack	Star Wars Rebels
The Ghost	Star Wars Rebels
TIE Advanced foil pack	Star Wars
Vulture Droid foil pack	Star Wars Episode 3
A-Wing	Star Wars Rebels
Wishing Well	Friends
(296 rows)	

- total number of rows: 296 rows
- Full result: <https://drive.google.com/file/d/1AVkKkYWPIGIAzQEtluuNa9sCnv9jUw/view>

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

```
lego=# SELECT year, count(set_num)
  FROM sets
 WHERE year<=2017 and year>=1950
 GROUP BY year
 ORDER BY count(set_num) desc;
   year | count
-----+
  2014 | 713
  2015 | 665
  2012 | 615
  2016 | 596
  2013 | 593
  2011 | 503
  2002 | 447
  2010 | 444
  2003 | 415
  2009 | 402
  2004 | 371
  2008 | 349
  2001 | 339
  2005 | 330
  2000 | 327
  1998 | 325
  2007 | 321
  1999 | 300
  2017 | 296
  2006 | 283
  1987 | 209
  1997 | 192
  1996 | 144
  1985 | 139
  1994 | 128
  1995 | 128
  1986 | 123
  1992 | 115
  1989 | 115
  1993 | 111
  1991 | 107
(66 rows)
```

year	count
1991	107
1977	92
1966	89
1980	88
1990	85
1979	82
1981	79
1984	76
1982	76
1978	73
1969	69
1988	68
1976	68
1973	67
1983	57
1971	45
1958	42
1962	40
1974	39
1972	38
1975	31
1970	29
1955	28
1968	25
1957	21
1967	21
1963	18
1961	17
1954	14
1956	12
1964	11
1965	10
1950	7
1959	4
1953	4
1960	3
(66 rows)	

- 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
lego=# WITH scchemes(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 scchemes)
SELECT theme_name
FROM mptheme,scchemes
WHERE scchemes.setcount=mptheme.setcount;
theme_name
-----
Gear
(1 row)
```

- total number of rows: 1 row

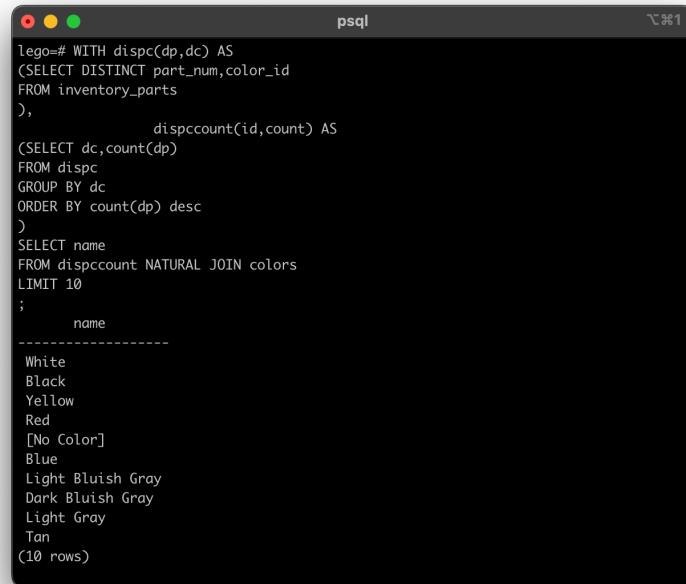
F. The SQL statements and output results of 4d.

name	avg_part_num
Wooden Box Set	-1.0000000000000000
Mindstorms	0.0000000000000000
Train	0.0000000000000000
Samsonite	0.0000000000000000
Key Chain	0.1818181818181818
Technic	1.0000000000000000
Imperial Guards	1.0000000000000000
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

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	693.63636363636364
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

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

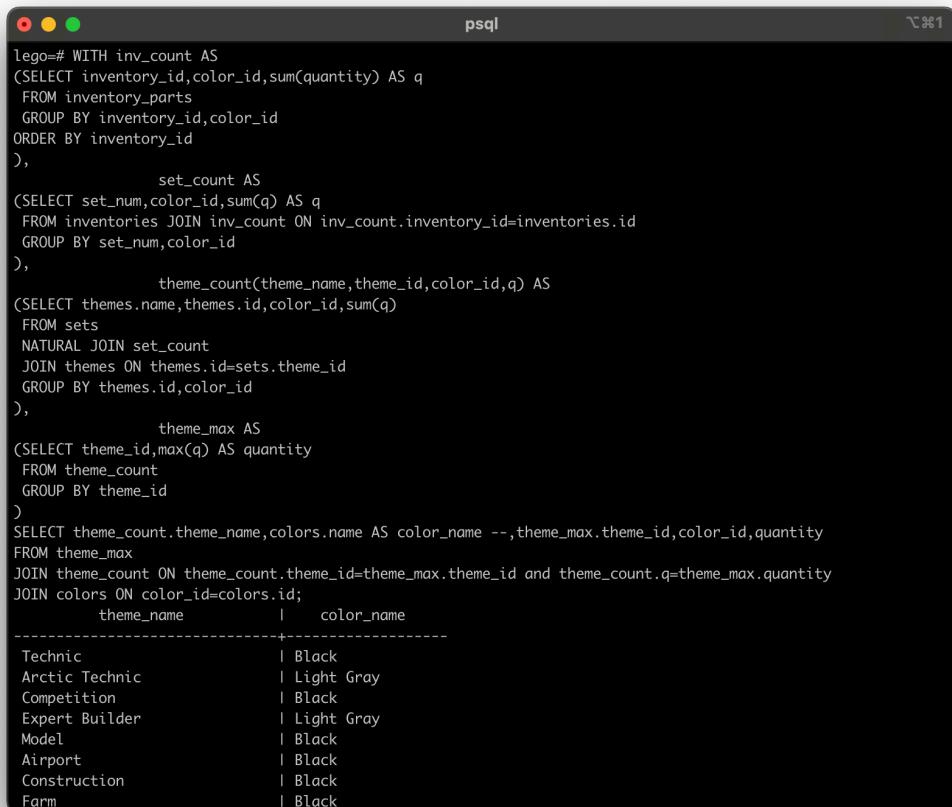
G. The SQL statements and output results of 4e.



```
lego=# WITH dispcc(dp,dc) AS
  (SELECT DISTINCT part_num,color_id
   FROM inventory_parts
  ),
    dispccount(id,count) AS
  (SELECT dc,count(dp)
   FROM dispcc
   GROUP BY dc
   ORDER BY count(dp) desc
  )
  SELECT name
  FROM dispccount NATURAL JOIN colors
  LIMIT 10
;
      name
-----
White
Black
Yellow
Red
[No Color]
Blue
Light Bluish Gray
Dark Bluish Gray
Light Gray
Tan
(10 rows)
```

- total number of rows: 10 rows

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



```
lego=# WITH inv_count AS
  (SELECT inventory_id,color_id,sum(quantity) AS q
   FROM inventory_parts
   GROUP BY inventory_id,color_id
   ORDER BY inventory_id
  ),
    set_count AS
  (SELECT set_num,color_id,sum(q) AS q
   FROM inventories JOIN inv_count ON inv_count.inventory_id=inventories.id
   GROUP BY set_num,color_id
  ),
    theme_count(theme_name,theme_id,color_id,q) AS
  (SELECT themes.name,themes.id,color_id,sum(q)
   FROM sets
   NATURAL JOIN set_count
   JOIN themes ON themes.id=sets.theme_id
   GROUP BY themes.id,color_id
  ),
    theme_max AS
  (SELECT theme_id,max(q) AS quantity
   FROM theme_count
   GROUP BY theme_id
  )
  SELECT theme_count.theme_name,colors.name AS color_name --,theme_max.theme_id,color_id,quantity
  FROM theme_max
  JOIN theme_count ON theme_count.theme_id=theme_max.theme_id and theme_count.q=theme_max.quantity
  JOIN colors ON color_id=colors.id;
      theme_name      |      color_name
-----
Technic           | Black
Arctic Technic   | Light Gray
Competition       | Black
Expert Builder    | Light Gray
Model             | Black
Airport           | Black
Construction      | Black
Farm              | Black
```

psql

Legends of Chima	Black
Speedorz	Black
Construction	Black
Legend Beasts	White
The Lone Ranger	Black
LEGO Ideas and CUUSOO	Light Bluish Gray
Minecraft	Light Bluish Gray
The LEGO Movie	Black
Disney Princess	White
Series 1	Black
Series 2	Black
Series 3	Black
Series 4	Black
Series 5	Black
Series 6	Black
Series 7	Light Bluish Gray
Series 8	White
Series 9	Black
Fusion	Tan
Juniors	White
Promotional	White
LEGO Exclusive	Black
Elves	Black
Speed Champions	Black
Jurassic World	Dark Bluish Gray
Scooby-Doo	Light Bluish Gray
Dimensions	Black
Nexo Knights	Black
Angry Birds	Light Bluish Gray
Ghostbusters	Black
Disney	White
Brickheadz	Black
Series 17 Minifigures	Black
Star Wars Episode 8	Light Bluish Gray
Jungle	Black
(568 rows)	

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