

Database HW1

1. The process of creating the “lego” databases (can be screenshot and/or SQL/non-SQL statements with text explanation) (8pts)

Ans:

```
-- in postgresql
CREATE DATABASE lego;
```

```
# in terminal
createdb lego
```

```
lego=# \l
```

資料庫清單									
名稱	擁有者	字元編碼	Locale Provider	Collate	轉換型別	ICU Locale	ICU Rules	存取權限	
lego	postgres	UTF8	libc	Chinese (Traditional)_Taiwan.950	Chinese (Traditional)_Taiwan.950				
postgres	postgres	UTF8	libc	Chinese (Traditional)_Taiwan.950	Chinese (Traditional)_Taiwan.950				

2. The process of importing eight required .csv files into lego database (can be screenshot and/or SQL/non-SQL statements with text explanation) (32pts).

Please include/describe the data type and keys of the imported table in your screenshot, SQL statements, and explanations.

Ans:

```
CREATE TABLE themes(
  id INT PRIMARY KEY,
  name VARCHAR(255),
  parent_id INT
);

CREATE TABLE part_categories(
  id INT PRIMARY KEY,
  name VARCHAR(255)
```

```

);

CREATE TABLE sets(
    set_num VARCHAR(255) PRIMARY KEY,
    name VARCHAR(255),
    year INT,
    theme_id INT REFERENCES themes(id),
    num_parts INT
);

CREATE TABLE parts(
    part_num VARCHAR(255) PRIMARY KEY,
    name VARCHAR(400),
    part_cat_id INT REFERENCES part_categories(id)
);

CREATE TABLE inventories(
    id INT PRIMARY KEY,
    version INT,
    set_num VARCHAR(255) REFERENCES sets(set_num)
);

CREATE TABLE colors(
    id INT PRIMARY KEY,
    name VARCHAR(255),
    rgb VARCHAR(255),
    is_trans BOOLEAN
);

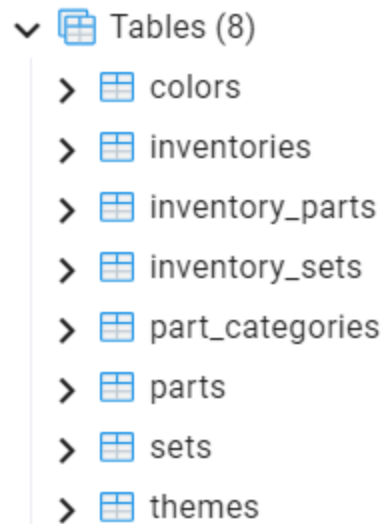
CREATE TABLE inventory_sets(
    inventory_id INT REFERENCES inventories(id),
    set_num VARCHAR(255) REFERENCES sets(set_num),
    quantity INT,
    PRIMARY KEY (inventory_id, set_num)
);

CREATE TABLE inventory_parts(
    inventory_id INT REFERENCES inventories(id),
    part_num VARCHAR(255), --ignore
    color_id INT REFERENCES colors(id),
    quantity INT,
    is_spare BOOLEAN,
    PRIMARY KEY (inventory_id, part_num, color_id, is_spare)
);

COPY themes FROM 'archive\\themes.csv' DELIMITER ',' CSV HEADER;
COPY part_categories FROM 'archive\\part_categories.csv' DELIMITER ',' CSV HEADER;
COPY sets FROM 'archive\\sets.csv' DELIMITER ',' CSV HEADER;
COPY parts FROM 'archive\\parts.csv' DELIMITER ',' CSV HEADER;
COPY inventories FROM 'archive\\inventories.csv' DELIMITER ',' CSV HEADER;
COPY colors FROM 'archive\\colors.csv' DELIMITER ',' CSV HEADER;

```

```
COPY inventory_parts FROM 'archive\\inventory_parts.csv' DELIMITER ',' CSV HEADER;  
COPY inventory_sets FROM 'archive\\inventory_sets.csv' DELIMITER ',' CSV HEADER;
```



3. The SQL statements and output results of 4a (10pts).

Extract the name of the set and name of the theme of all the LEGO sets published in 2017.

Ans:

```
SELECT sets.name AS set_name, themes.name AS theme_name  
FROM sets  
JOIN themes ON sets.theme_id = themes.id  
WHERE sets.year = 2017;
```

Display only a portion of the answer

	set_name character varying (255) 🔒	theme_name character varying (255) 🔒
1	Assembly Square	Modular Buildings
2	Carousel	Creator
3	Creative Builder Box	Classic
4	Creative Box	Classic
5	Blue Creative Box	Classic
6	Red Creative Box	Classic

4. The SQL statements and output results of 4b (10pts)

Extract the total number of LEGO sets in each year from 1950 to 2017, in descending order of total number of LEGO sets.

Ans:

```
SELECT year, COUNT(*) AS total_sets
FROM sets
WHERE year BETWEEN 1950 AND 2017
GROUP BY year
ORDER BY total_sets DESC;
```

Display only a portion of the answer


	year integer 🔒	total_sets bigint 🔒
1	2014	713
2	2015	665
3	2012	615
4	2016	596
5	2013	593
6	2011	503

5. The SQL statements and output results of 4c (10pts)

Extract the name of the most popular theme, defined by the number of sets in the themes.

Ans:

```
SELECT themes.name
FROM sets
JOIN themes ON sets.theme_id = themes.id
GROUP BY themes.id
ORDER BY COUNT(sets.set_num) DESC
LIMIT 1;
```

	name character varying (255) 
1	Gear

6. The SQL statements and output results of 4d (10pts)

Extract the average number of parts in a set for each unique theme, showing the results with the name of the theme and the average number of parts per set. In ascending order of average number of parts in a set

Ans:

```
SELECT themes.id AS theme_ID, themes.name AS theme_name, AVG(sets.num_parts) AS avg_num_parts
FROM sets
JOIN themes ON sets.theme_id = themes.id
GROUP BY themes.id
ORDER BY avg_num_parts ASC;
```

Display only a portion of the answer

	theme_id integer	theme_name character varying (255)	avg_num_parts numeric
1	383	Wooden Box Set	-1.00000000000000000000
2	499	Train	0.00000000000000000000
3	523	Samsonite	0.00000000000000000000
4	258	Mindstorms	0.00000000000000000000
5	503	Key Chain	0.18181818181818181818
6	498	Technic	1.00000000000000000000
7	150	Imperial Guards	1.00000000000000000000
8	289	Supplemental	1.80000000000000000000

7. The SQL statements and output results of 4e (10pts)

Find out the name of the colors that are most used in the unique LEGO parts, and list the top 10.

Ans:

```
SELECT colors.name AS color_name, COUNT(DISTINCT inventory_parts.part_num) AS unique_parts_count
FROM inventory_parts
JOIN colors ON inventory_parts.color_id = colors.id
GROUP BY colors.name
ORDER BY unique_parts_count DESC
LIMIT 10;
```

	color_name character varying (255) 🔒	unique_parts_count bigint 🔒
1	White	4714
2	Black	4376
3	Yellow	2938
4	Red	2882
5	[No Color]	2000
6	Blue	1833
7	Light Bluish Gray	1596
8	Dark Bluish Gray	1519
9	Light Gray	1351
10	Tan	1048

8. The SQL statements and output results of 4f (10pts)

Find out the name of the colors that are most used in the LEGO parts, for each theme, and list the top 1 for each theme (please provide the name of the theme, too).

Rank and dense rank functions

Ans:

```
WITH ColorUsage AS (
    SELECT
        themes.id AS ThemeID,
        themes.name AS ThemeName,
        colors.name AS ColorName,
        SUM(inventory_parts.quantity) AS TotalQuantity,
        RANK() OVER (PARTITION BY themes.id ORDER BY SUM(inventory_parts.quantity) DESC) AS Rank
    FROM inventory_parts
    JOIN inventories ON inventory_parts.inventory_id = inventories.id
    JOIN sets ON inventories.set_num = sets.set_num
    JOIN themes ON sets.theme_id = themes.id
    JOIN colors ON inventory_parts.color_id = colors.id
    GROUP BY themes.id, colors.name
)
```

```

SELECT
    ThemeID,
    ThemeName,
    ColorName,
    TotalQuantity
FROM
    ColorUsage
WHERE
    Rank = 1
ORDER BY
    ThemeName;

```

Display only a portion of the answer

	themeid integer	themename character varying (255)	colorname character varying (255)	totalquantity bigint
1	234	12V	Black	2468
2	242	12V	Light Gray	143
3	279	4 Juniors	White	48
4	235	4.5V	Black	3231
5	243	4.5V	Blue	124
6	244	9V	Dark Bluish Gray	39
7	236	9V	Black	4105
8	207	Advent	Red	132