kokchun giang

sometimes you need to **generate data** to enhance the dataset or for testing

using **VALUES clause** to create set of rows

```
VALUES (1), (2), (3);
```

creates a column with three rows

```
CREATE TABLE IF NOT EXISTS people_records AS (
SELECT
    *
FROM
    (
VALUES (1,
'Johan'),
(2,
'Johanna'),
(3,
'Mustafa')) AS people(id,
    name));
```

create table to store the synthetic data in the database

creates two columns and three rows

alias to give column names to the people table

simulating data using **random() function**

RANDOM() gives a floating number between 0 and 1, this gives integers between 1 and 6

```
SELECT
    FLOOR(RANDOM()* 6 + 1) AS dice,
FROM
    generate_series(100);
```

generates 100 rows

```
SELECT
    UNNEST (generate_series(1,
    100)) AS id,
    'student_' || id AS student,
    -- concatenation
    ROUND(RANDOM()* 100) AS score;
```

concatenates 'student_' with a number between 1 and 100

generate random score between 0 and 100

generate **temporal data**

```sql
SELECT
    *
FROM
    generate_series(DATE '2024-11-01',
    DATE '2024-11-30',
    INTERVAL '1 day') AS t(november);
```

generates date series between first argument and second with 1 day interval

```sql
CREATE TABLE IF NOT EXISTS dim_date AS (
SELECT
    strftime(date_series,
    '%Y-%m-%d') AS date,
    month(date_series) AS month,
    week(date_series) AS week_number,
    weekday(date_series) AS day_of_week,
FROM
    generate_series(DATE '2024-1-01',
    DATE '2024-12-31',
    INTERVAL 1 DAY) AS t(date_series));
```

format date with strftime

date parts functions to extract parts of date to different columns