

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

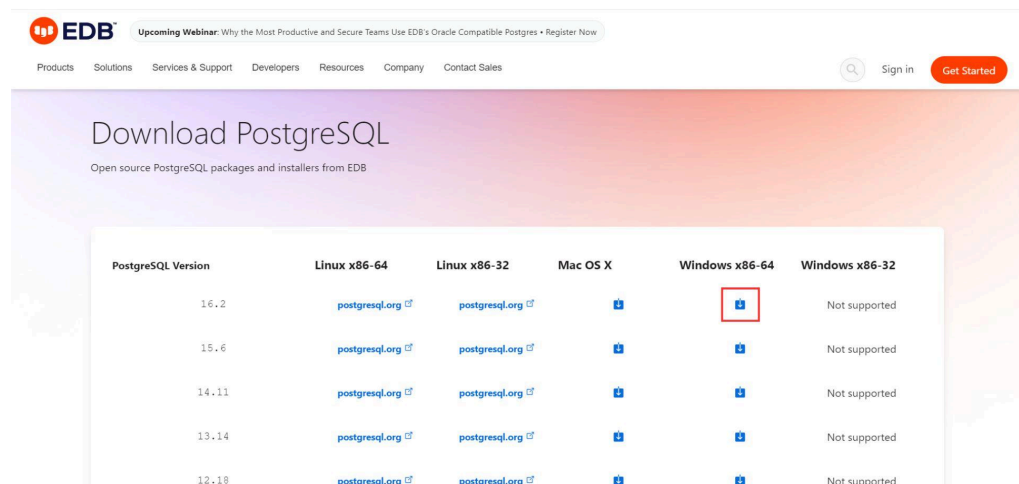
Deadline: Sabtu, 4 Mei 2024 23:55 WIB (Waktu SCoLE)

Untuk mengerjakan tutorial ini, Anda membutuhkan PostgreSQL pada perangkat Anda. Silakan mengikuti tutorial yang telah disediakan dan juga mengunduh data di [tautan berikut](#).

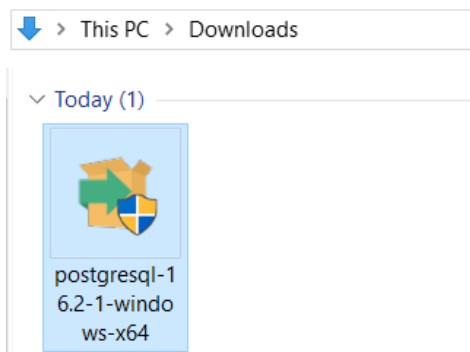
Restore Data ke PostgreSQL

- Windows

1. Download PostgreSQL melalui [tautan berikut](#).



2. Buka *file* yang telah di-*download* tersebut.

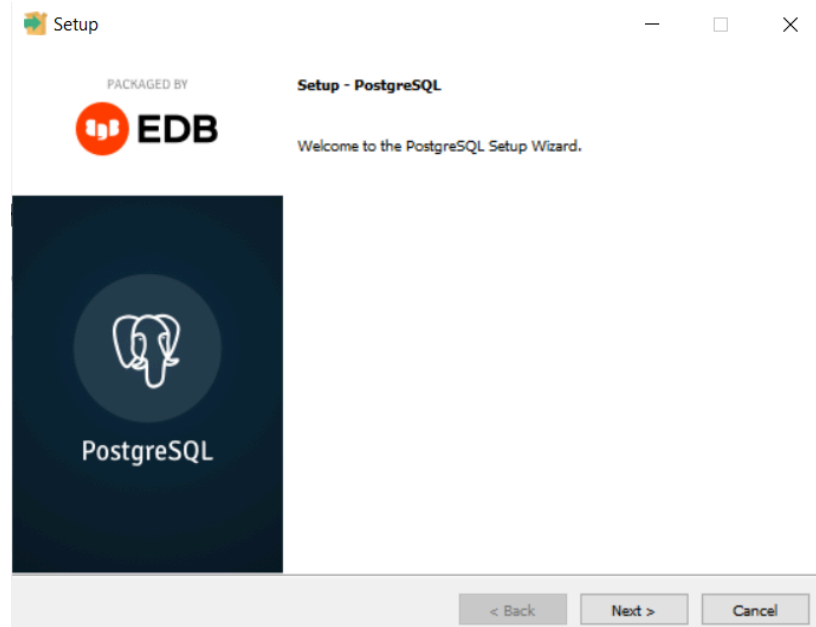


3. Klik "Next".

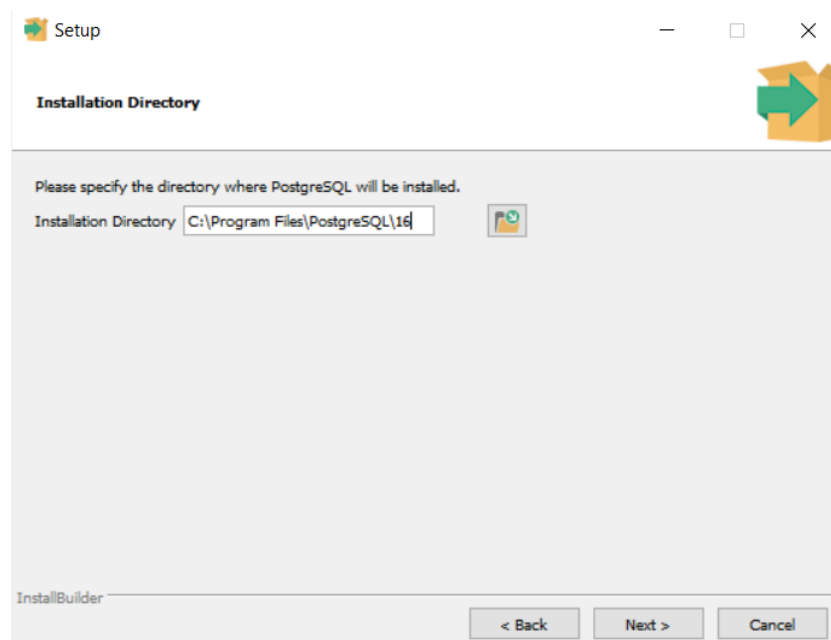
CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data



4. Sesuaikan lokasi instalasi. Anda dapat membiarkan lokasi tersebut *default*.. Pastikan Anda mengingat lokasi tersebut. Kemudian, klik “Next”.

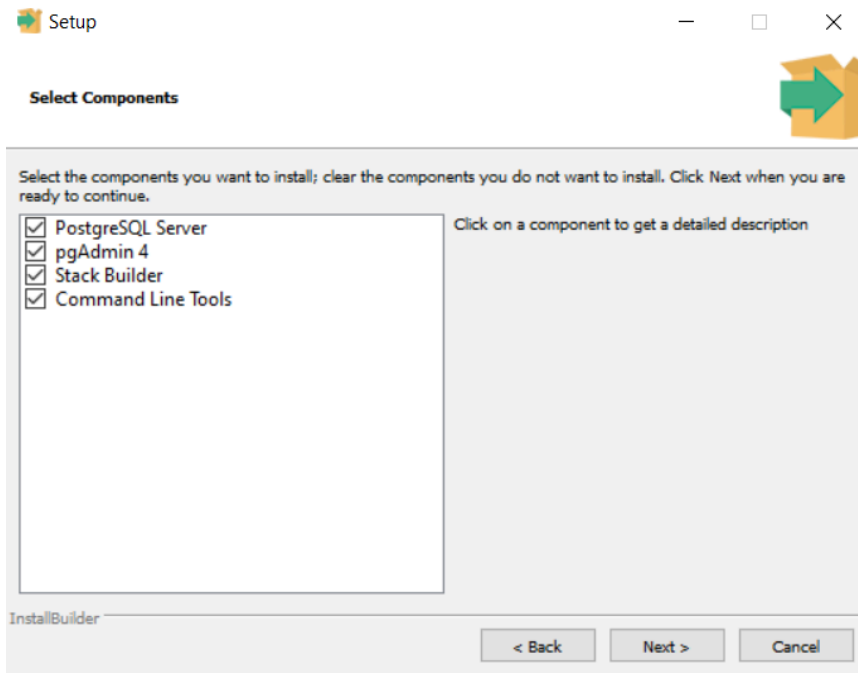


5. Klik “Next”.

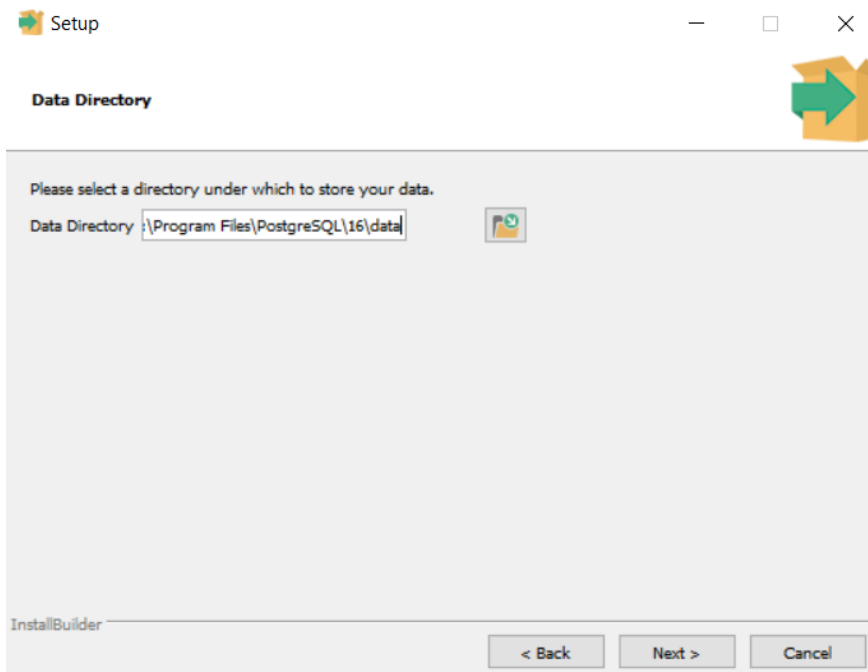
CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data



6. Sesuaikan lokasi penyimpanan data. Anda dapat membiarkan lokasi tersebut *default*. Kemudian, klik “Next”.

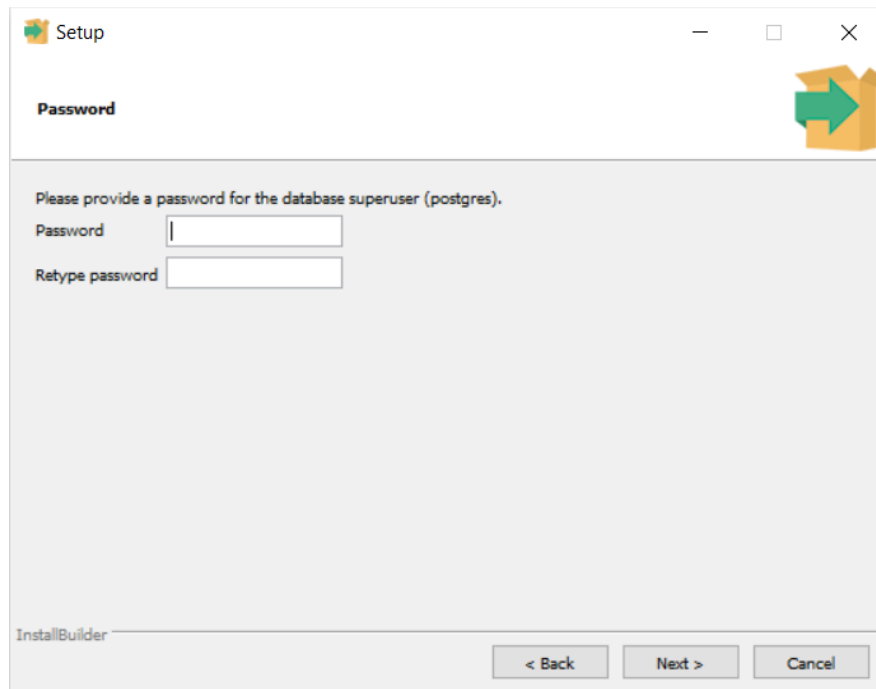


7. Masukkan *password* yang ingin digunakan untuk mengakses basis data. **Pastikan** bahwa Anda mengingat *password* Anda.

CSGE602070 Basis Data
Semester Genap 2023/2024

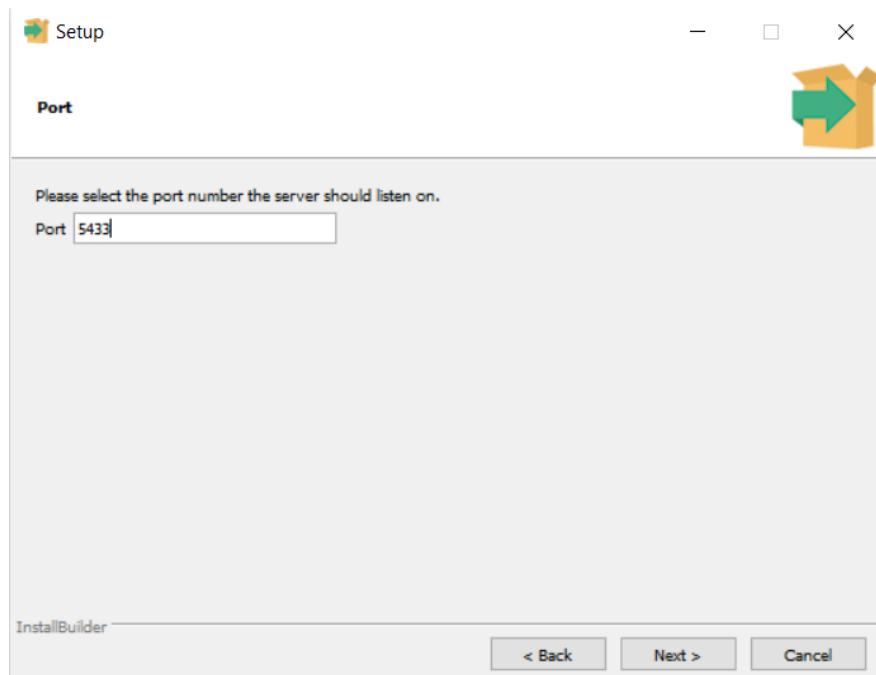
Tutorial 3

Triggered & Stored Procedure, Big Data



The screenshot shows the 'Setup' window for PostgreSQL, specifically the 'Password' step. The window has a title bar with 'Setup' and standard window controls. A green arrow icon is in the top right corner. The main area contains the text 'Please provide a password for the database superuser (postgres).' followed by two input fields: 'Password' and 'Retype password'. At the bottom, there is a progress bar labeled 'InstallBuilder' and three buttons: '< Back', 'Next >', and 'Cancel'.

8. Sesuaikan *port* yang ingin digunakan. Kemudian, klik “Next”.



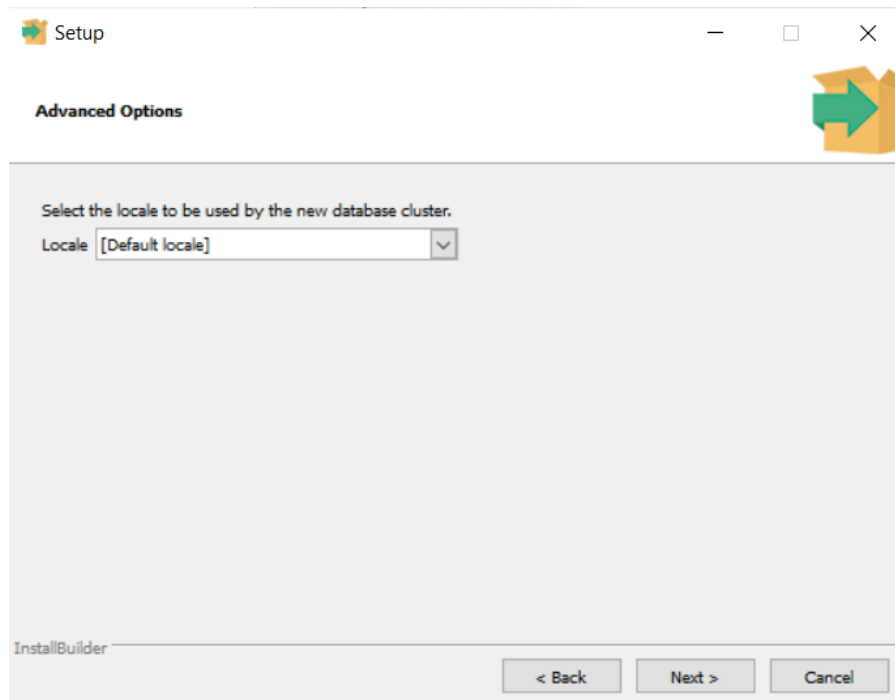
The screenshot shows the 'Setup' window for PostgreSQL, specifically the 'Port' step. The window has a title bar with 'Setup' and standard window controls. A green arrow icon is in the top right corner. The main area contains the text 'Please select the port number the server should listen on.' followed by a 'Port' label and an input field containing the value '5433'. At the bottom, there is a progress bar labeled 'InstallBuilder' and three buttons: '< Back', 'Next >', and 'Cancel'.

9. Klik “Next”.

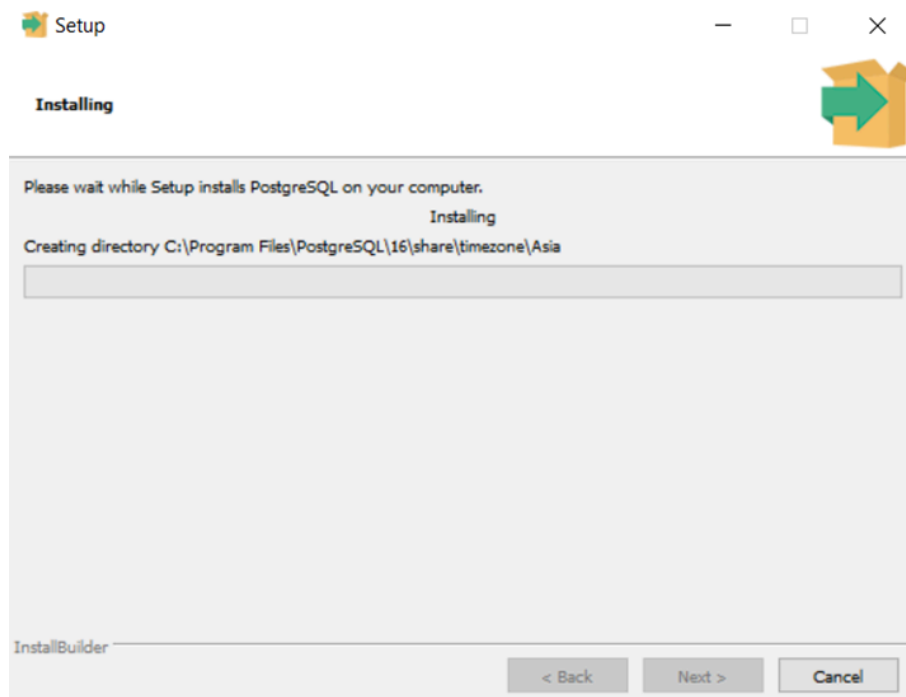
CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data



10. Lanjutkan dengan klik “Next” hingga proses instalasi berlangsung.



11. Klik “Finish”.

12. Untuk memastikan PostgreSQL telah terkonfigurasi dengan baik, buka

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

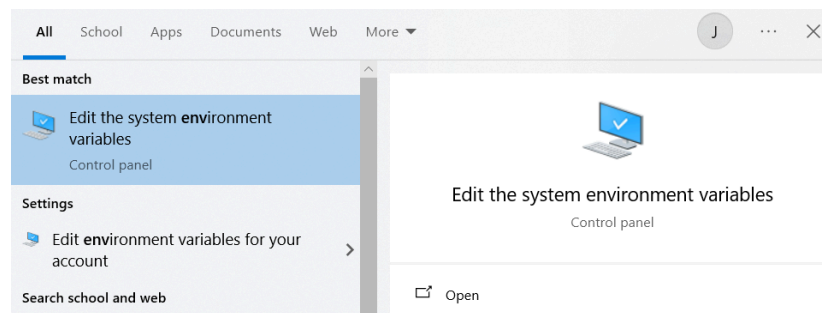
Triggered & Stored Procedure, Big Data

terminal dan jalankan perintah `psql --version`. Jika tidak muncul *output* seperti gambar di bawah ini, Anda dapat melanjutkan ke langkah 18.

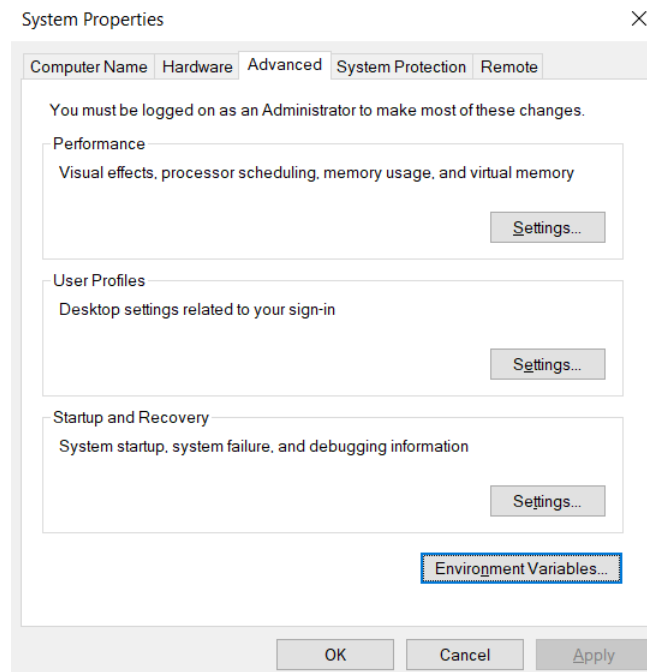
Command Prompt

```
C:\Users\jonat>psql --version
'psql' is not recognized as an internal or external command,
operable program or batch file.
```

13. Ketik “env” di *search bar* Windows. Kemudian, klik pilihan berikut.



14. Klik *environment variables*.



15. Kemudian, pada bagian *system variables*, pilih *variable* bernama “Path” kemudian tekan “Edit”. Tambahkan *path* menuju lokasi instalasi PostgreSQL \ bin. Lokasi instalasi PostgreSQL merupakan lokasi yang ditentukan pada langkah 4. Dalam hal ini, `C:\Program Files\PostgreSQL\16\bin`.

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

16. Ketika sudah selesai, klik “OK” hingga *settings* telah tertutup.
17. Tutup terminal sebelumnya. Kemudian, buka kembali terminal dan jalankan perintah `psql --version`. Jika konfigurasi telah berhasil, Anda akan melihat *output* berikut.

```
C:\> Command Prompt
C:\Users\jonat>psql --version
psql (PostgreSQL) 16.2
```

18. Kini, buatlah *database* dengan nama *username* SSO Anda. Salah satu caranya adalah dengan menjalankan perintah `createdb -U postgres "username SSO Anda"`. Sebagai contoh, untuk *username* SSO john.doe, perintahnya akan terlihat seperti berikut.

```
C:\> Command Prompt
C:\Users\jonat>createdb -U postgres "john.doe"
Password:
```

Masukkan *password* yang telah Anda buat pada langkah 7.

19. Kemudian, jalankan perintah `pg_restore -U postgres -d "username SSO Anda" < path\to\indiemart_dump.custom`. Sebagai contoh, untuk lokasi *indiemart_dump.custom* pada “Downloads”, perintahnya akan terlihat seperti berikut.

```
C:\> Command Prompt
C:\Users\jonat>pg_restore -U postgres -d "john.doe" < Downloads\indiemart_dump.custom
Password:
```

Masukkan *password* yang telah Anda buat pada langkah 7. Kemudian, tunggu beberapa menit.

20. Untuk mengakses *database* tersebut, Anda dapat menjalankan perintah `psql -U postgres -d "username SSO Anda"`. Sebagai contoh, untuk *username* SSO john.doe, perintahnya akan terlihat seperti berikut.

```
C:\> Command Prompt - psql -U postgres -d "john.doe"
C:\Users\jonat>psql -U postgres -d "john.doe"
Password for user postgres:
psql (16.2, server 15.2)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

john.doe=#
```

CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data

21. Kini, Anda dapat melihat isi dari *database* tersebut.

```
john.doe=# \d
List of relations
Schema | Name | Type | Owner
public | belanja | table | postgres
public | belanja_link | table | postgres
public | discounts | table | postgres
public | item_item | table | postgres
public | items | table | postgres
public | prices | table | postgres
(6 rows)

john.doe=# SELECT * FROM BELANJA;
 id | belanja_link_id | items_id | custom_price | created_at | updated_at | deleted_at
-----+-----+-----+-----+-----+-----+-----
(0 rows)

john.doe=# SELECT * FROM DISCOUNTS;
 id | items_id | discount_price | original_price | percentage | description | created_at
-----+-----+-----+-----+-----+-----+-----
5LuJfE2qhVntggvLkvd3DL | 783543 | 36000 | 38000 | 5.26315789473684 |  | 2024-02-21 09:10:48
btM4EEhvTLhdsRQANETr2p | 794964 | 8900 | 9500 | 6.31578947368421 |  | 2024-02-21 09:10:48
JxetTXfQhKzRgcDgz7QayU | 796560 | 16500 | 21000 | 21.4285714285714 |  | 2024-02-21 09:10:48
H33S6hbvPCYQ53wXA9go3o | 178861 | 43400 | 43400 | 0.0 |  | 2024-02-21 09:10:48
3RwPyIfuJzyxVrSDcqtYDh | 800347 | 45000 | 55900 | 19.4991055456172 |  | 2024-02-21 09:10:48
VyXZaI2nRoZwJpXdsO7 | 766897 | 90000 | 155000 | 41.9354838709677 |  | 2024-02-21 09:10:48
7JwPxfafGUGH7pH3KkyRr | 766898 | 90000 | 155000 | 41.9354838709677 |  | 2024-02-21 09:10:48
Kqvbuyt4eVzyXmLF5NVR8T | 788638 | 62000 | 165000 | 62.4242424242424 |  | 2024-02-21 09:10:48
eGvzPKemku3Wsh9Z7hozhp | 14514 | 15000 | 19000 | 21.0526315789474 |  | 2024-02-21 09:10:48
JfHgAwomtk6CVTPct7ec | 783162 | 15000 | 18200 | 17.5824175824176 |  | 2024-02-21 09:10:48
SxwU5oAy5dks73xnE5G7b | 16387 | 79900 | 91500 | 12.6775956284153 |  | 2024-02-21 09:10:48
Kyxxq9PacwPy9P5zbtFi8 | 16406 | 79900 | 91500 | 12.6775956284153 |  | 2024-02-21 09:10:48
biOmwe8oSVwBo7KHAFJNA | 16621 | 79900 | 91500 | 12.6775956284153 |  | 2024-02-21 09:10:48
cDMiGAXdpq9hmkJdgnW5gK | 780836 | 4500 | 7700 | 41.5584415584416 |  | 2024-02-21 09:10:48
GBcyac9TgPhtZlaa183QPsU | 780835 | 4500 | 7700 | 41.5584415584416 |  | 2024-02-21 09:10:48
mMdlpxwepRzdE7fyxWiyEg | 790302 | 331500 | 390000 | 15.0 |  | 2024-02-21 09:10:48
9zGqJwHqQH7Lr1cpF9R5g | 776728 | 53000 | 89600 | 48.8482142857143 |  | 2024-02-21 09:10:48
```

- Mac

1. Unduh PostgreSQL pada [tautan berikut](#).

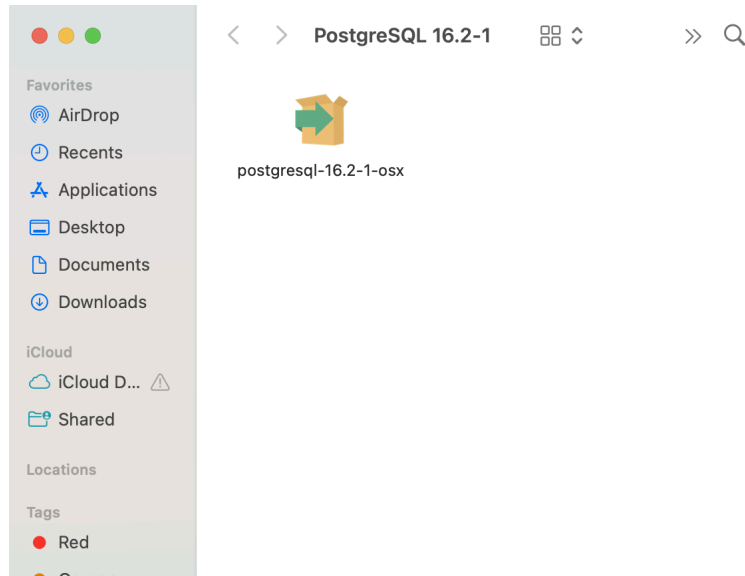


2. Buka file .dmg yang telah terunduh. Apabila diminta, masukkan *password* mac ANDA untuk melanjutkan proses instalasi.

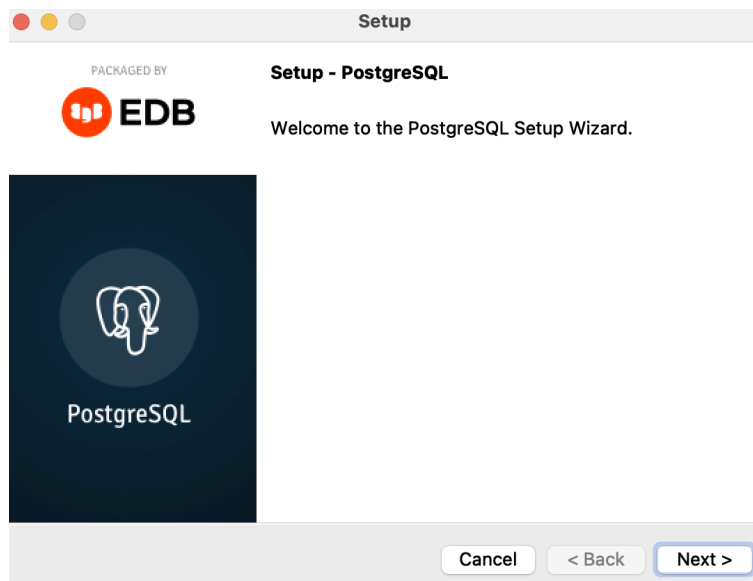
CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data



3. Installer akan terbuka. Klik “Next” untuk melanjutkan.

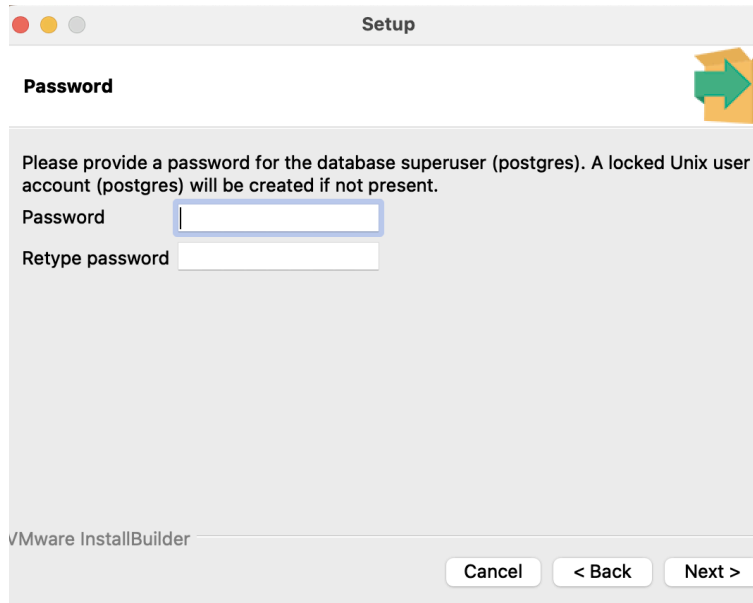


4. [PENTING] Ketika sudah sampai pada tahap pembuatan *password*, pastikan Anda mengingat *password* Anda. *Password* ini akan selalu digunakan untuk mengakses basis data.

CSGE602070 Basis Data
Semester Genap 2023/2024

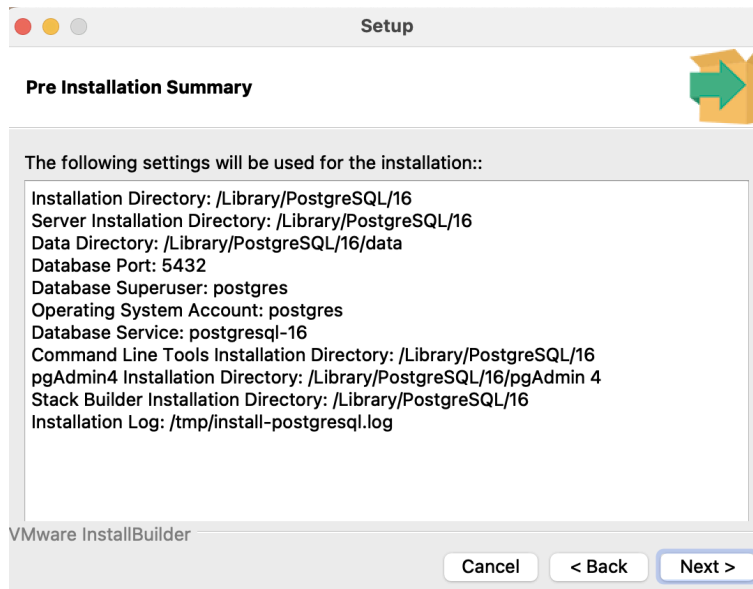
Tutorial 3

Triggered & Stored Procedure, Big Data



The screenshot shows the 'Setup' window for PostgreSQL. The title bar says 'Setup'. Below the title bar, there's a 'Password' section with a green arrow icon pointing right. The text says: 'Please provide a password for the database superuser (postgres). A locked Unix user account (postgres) will be created if not present.' There are two input fields: 'Password' and 'Retype password'. At the bottom, there are three buttons: 'Cancel', '< Back', and 'Next >'. The text 'VMware InstallBuilder' is visible in the bottom left corner.

5. Simpan juga Installation Directory dan Database Port. Klik “Next” untuk melanjutkan proses instalasi.



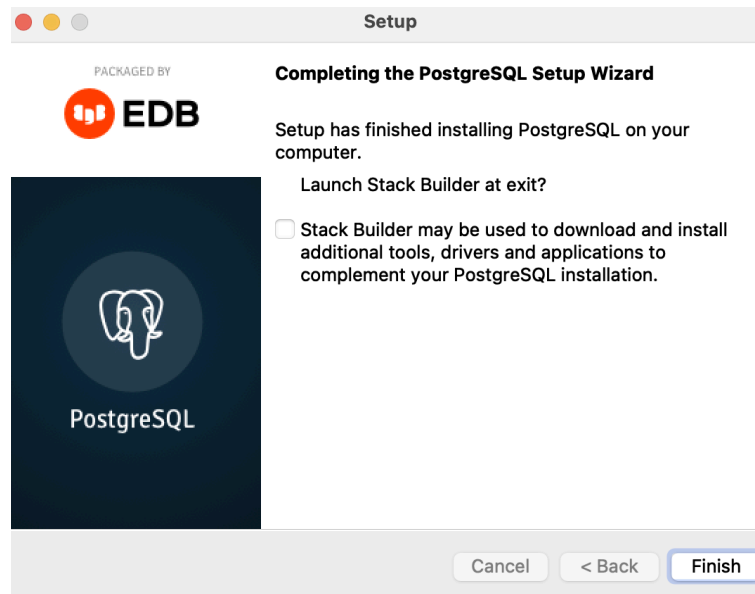
The screenshot shows the 'Setup' window for PostgreSQL. The title bar says 'Setup'. Below the title bar, there's a 'Pre Installation Summary' section with a green arrow icon pointing right. The text says: 'The following settings will be used for the installation::'. Below this, there's a list of settings: 'Installation Directory: /Library/PostgreSQL/16', 'Server Installation Directory: /Library/PostgreSQL/16', 'Data Directory: /Library/PostgreSQL/16/data', 'Database Port: 5432', 'Database Superuser: postgres', 'Operating System Account: postgres', 'Database Service: postgresql-16', 'Command Line Tools Installation Directory: /Library/PostgreSQL/16', 'pgAdmin4 Installation Directory: /Library/PostgreSQL/16/pgAdmin 4', 'Stack Builder Installation Directory: /Library/PostgreSQL/16', and 'Installation Log: /tmp/install-postgresql.log'. At the bottom, there are three buttons: 'Cancel', '< Back', and 'Next >'. The text 'VMware InstallBuilder' is visible in the bottom left corner.

6. Apabila proses telah selesai, klik “Finish”.

CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data



7. Untuk memastikan PostgreSQL telah terinstal dengan baik, buka terminal dan ketikkan `psql --version`. Apabila muncul pesan seperti gambar di bawah ini, ketikkan `export PATH={Installation Directory}/bin:$PATH` (sesuaikan Installation Directory)

```
alifiyahariandri ~ -zsh — 80x24
Last login: Sun Apr 14 22:21:43 on ttys002
alifiyahariandri@Alifiyachs-Laptop ~ % psql
zsh: command not found: psql
alifiyahariandri@Alifiyachs-Laptop ~ %
```

8. Berikut merupakan contoh tampilan saat berhasil menampilkan versi.

```
alifiyahariandri ~ -zsh — 85x24
alifiyahariandri@Alifiyachs-Laptop ~ % export PATH=/Library/PostgreSQL/16/bin:$PATH
alifiyahariandri@Alifiyachs-Laptop ~ % psql --version
psql (PostgreSQL) 16.2
alifiyahariandri@Alifiyachs-Laptop ~ %
```

9. Buat *database* dengan nama *username* SSO Anda. Salah satu caranya dapat dengan menjalankan perintah `createdb -U postgres {username SSO}`. Sebagai contoh, untuk *username* SSO *alifiyah.nur* seperti berikut.

```
alifiyahariandri ~ -zsh — 80x24
alifiyahariandri@Alifiyachs-Laptop ~ % export PATH=/Library/PostgreSQL/16/bin:$PATH
alifiyahariandri@Alifiyachs-Laptop ~ % createdb -U postgres "alifiyah.nur"
Password:
alifiyahariandri@Alifiyachs-Laptop ~ %
```

10. Arahkan ke folder tempat berkas *indiemart_dump.custom* disimpan, lalu ketikkan `pg_restore -U postgres -d {username SSO} < indiemart_dump.custom`

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

Terminal akan meminta *password*. Masukkan *password* yang telah dibuat saat instalasi PostgreSQL.

```
Downloads — -zsh — 80x24
[alifiyahariandri@Alifiyahs-Laptop ~ % export PATH=/Library/PostgreSQL/16/bin:$PATH]
[alifiyahariandri@Alifiyahs-Laptop ~ % createdb -U postgres "alifiyah.nur"]
[Password:]
[alifiyahariandri@Alifiyahs-Laptop ~ % cd Downloads]
[alifiyahariandri@Alifiyahs-Laptop Downloads % pg_restore -U postgres -d alifiyah]
.nur < indiemart_dump.custom
[Password:]
[alifiyahariandri@Alifiyahs-Laptop Downloads %]
```

11. Untuk mengakses *database* tersebut, ketikkan `psql -U postgres -d {username SSO}`. Berikut merupakan *database* yang telah berhasil di-restore.

```
Downloads — psql -U postgres alifiyah.nur — 80x24
[Password:]
[alifiyahariandri@Alifiyahs-Laptop ~ % cd Downloads]
[alifiyahariandri@Alifiyahs-Laptop Downloads % pg_restore -U postgres -d alifiyah]
.nur < indiemart_dump.custom
[Password:]
[alifiyahariandri@Alifiyahs-Laptop Downloads % psql -U postgres alifiyah.nur]
[Password for user postgres:]
psql (16.2)
Type "help" for help.

alifiyah.nur=#
alifiyah.nur=# \dt
          List of relations
Schema | Name      | Type  | Owner
-----|-----|-----|-----
public | belanja   | table | postgres
public | belanja_link | table | postgres
public | discounts | table | postgres
public | item_item | table | postgres
public | items     | table | postgres
public | prices    | table | postgres
(6 rows)

alifiyah.nur=#
```

Big Data

Big data mengacu pada kumpulan data yang sangat besar dan kompleks. Kumpulan data ini biasanya memiliki *volume*, *velocity*, dan *variety*, yang biasa disebut "3V" dari big data. Volume mengacu pada jumlah data yang besar, velocity mengacu pada kecepatan data dihasilkan dan diproses, dan variasi mengacu pada berbagai jenis sumber data dan format.

Pada tutorial kali ini, kita akan menggunakan tabel-tabel berikut:

ITEMS	Berisi data item-item yang dijual di IndieMart, terdiri dari 18.064 baris	<ul style="list-style-type: none"> - Id (PK) - SKU - Name - Category
-------	---	--

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

		<ul style="list-style-type: none"> - Image - Link - Source - Created_at
PRICES	Berisi harga dari items (6.029.350 rows)	<ul style="list-style-type: none"> - Id (PK) - Items_id - Price - Description - Created_at
DISCOUNTS	Menyimpan diskon dari items (6.029.346 rows)	<ul style="list-style-type: none"> - Id (PK) - Items_id - Discount_price - Original_price - Percentage - Description - Created_at

Kita akan mencoba melihat perbandingan *query* pada big data. Misalnya, kita ingin mencari harga termurah dari produk Frisian Flag Kental Manis Coklat 370G dengan dua *query* yang berbeda.

```
SELECT MIN(PRICES.Price)
FROM PRICES
JOIN ITEMS ON PRICES.Items_id = ITEMS.Id
WHERE ITEMS.Name LIKE '%Frisian Flag Kental Manis Coklat 370G%';
```

```
SELECT MIN(subquery.Price) AS min
FROM (
  SELECT PRICES.Price, ITEMS.Name
  FROM PRICES
  JOIN ITEMS ON PRICES.Items_id = ITEMS.Id
  WHERE ITEMS.Name LIKE '%Frisian Flag Kental Manis Coklat 370G%'
  ORDER BY PRICES.Price
) AS subquery;
```

CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data

Latihan Soal 1

1. **[SQL]** Jalankan kedua *query* di atas.
2. **[Trivia]** Tanpa menggunakan *index*, lakukan analisis terhadap kedua *query* tersebut.

Hint: Anda dapat menggunakan perintah `EXPLAIN ANALYZE`

3. **[SQL]** Jalankan kedua *query* di bawah ini dengan dan tanpa *index*
Query A (jalankan versi tanpa *index* dan ketika sudah menerapkan *index*)

```
EXPLAIN ANALYZE
SELECT I.id, P.price, AVG(D.discount_price) AS average_discount_price
FROM DISCOUNTS D
JOIN PRICES P ON P.items_id = D.items_id
JOIN ITEMS I ON P.items_id = I.id
WHERE I.category = 'Personal Care'
GROUP BY I.id, P.price
ORDER BY I.id;
```

Query B (jalankan versi tanpa *index* dan ketika sudah menerapkan *index*)

```
EXPLAIN ANALYZE
SELECT I.id, P.price, D.average_discount_price
FROM ITEMS I
JOIN PRICES P ON P.items_id = I.id
JOIN (
    SELECT items_id, AVG(discount_price) AS average_discount_price
    FROM DISCOUNTS
    GROUP BY items_id
) AS D ON D.items_id = I.id
WHERE I.category = 'Personal Care'
ORDER BY I.id;
```

Index

```
CREATE INDEX index_prices ON PRICES (items_id, price);
CREATE INDEX index_discounts ON DISCOUNTS (items_id, discount_price);
```

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

4. **[Trivia]** Analisis perbedaan *execution time* untuk setiap kondisi pada nomor 3, baik dengan maupun tanpa *index*. Apa saja faktor yang menyebabkan perbedaan signifikan dalam *execution time* antar setiap kondisi? Bagaimana pengaruh jumlah *row data* dan jumlah operasi JOIN pada performa *query*?

Hint: Perhatikan struktur *query* dan penggunaan *index*

Stored Procedure

Stored procedure dan *function* (atau *user-defined function*) merupakan **bagian dari bahasa prosedural** di dalam SQL. Di dalam PostgreSQL, bahasa ini disebut **PL/pgSQL**. Dengan keduanya, kita bisa menambahkan berbagai **elemen prosedural**, seperti *loop*, perhitungan kompleks, dan masih banyak lagi. Kita juga bisa mengembangkan fungsi yang kompleks yang **tidak bisa dicapai** dengan statement SQL biasa.

Pada awalnya PostgreSQL hanya memiliki objek ***function*** kemudian pada PostgreSQL versi 11 ke atas baru diperkenalkan yang disebut ***stored procedure***. Oleh karenanya, penting untuk memperhatikan versi PostgreSQL yang Anda gunakan. Terdapat banyak kemiripan antara *function* dan *stored procedure*. Perbedaannya adalah ***function* dapat me-return value** sementara ***stored procedure* tidak mengembalikan value**.

Untuk membuat *function*, sintaksnya adalah sebagai berikut.

```
CREATE [OR REPLACE] FUNCTION function_name (arguments)
RETURNS return_datatype AS
$$
DECLARE
    declaration;
    [...]
BEGIN
    < function_body >
    [...]
    RETURN { variable_name | value }
END;
$$
LANGUAGE plpgsql;
```

CSGE602070 Basis Data
Semester Genap 2023/2024

Tutorial 3

Triggered & Stored Procedure, Big Data

Contoh 1:

Sekarang, kita akan mencoba untuk membuat suatu *function* yang dapat mengubah nilai `discount_price` dari suatu data `discounts` berdasarkan nilai *percentage* yang dimasukkan. Contoh implementasi dari *function* tersebut adalah sebagai berikut.

```
CREATE OR REPLACE FUNCTION update_discount_price(discount_id VARCHAR(100),
new_percentage DECIMAL) RETURNS INTEGER AS
$$
DECLARE
    item_price INTEGER;
    updated_discount_price INTEGER;
BEGIN
    -- Mengambil harga original item tersebut
    SELECT original_price INTO item_price FROM discounts
    WHERE id = discount_id;
    -- Mengubah percentage item tersebut
    UPDATE discounts
    SET percentage = new_percentage
    WHERE id = discount_id;
    -- Menghitung harga diskon berdasarkan nilai percentage baru
    updated_discount_price := item_price - (item_price * new_percentage / 100);
    -- Mengubah harga diskon dengan nilai baru
    UPDATE discounts
    SET discount_price = updated_discount_price
    WHERE id = discount_id;
    RETURN updated_discount_price;
END;
$$
LANGUAGE plpgsql;
```

Untuk melihat *function* yang sudah dibuat, dapat menggunakan perintah berikut.

```
\df
```


CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

Untuk melihat kodingan fungsi yang sudah dibuat, dapat menggunakan perintah berikut.

```
\df+
```

Kita dapat menjalankan fungsi yang sudah dibuat dengan sintaks berikut.

```
SELECT SCHEMA_NAME.function_name([<arg1>, <arg2>, ...]);
```

Note: Tidak perlu menggunakan `SCHEMA_NAME` jika sudah set `search_path`.

Contoh 3:

Pertama, coba kita melihat data di tabel `discounts` dengan id `'223oLpuqqg8pZvMkT9sam4'`.

```
SELECT * FROM discounts WHERE id='223oLpuqqg8pZvMkT9sam4';
```

```
airel.camilo=# SELECT * FROM discounts WHERE id='223oLpuqqg8pZvMkT9sam4';
 id          | items_id | discount_price | original_price | percentage | description | created_at
-----+-----+-----+-----+-----+-----+-----
 223oLpuqqg8pZvMkT9sam4 | 13647   |          37500 |          38900 | 3.59897172236504 |              | 2024-02-23 09:10:46
(1 row)
```

Kemudian, kita ubah nilai *percentage* data dengan id tersebut dengan menjalankan fungsi `update_discount_price`.

```
SELECT update_discount_price('223oLpuqqg8pZvMkT9sam4', 15.0);
```

```
airel.camilo=# SELECT update_discount_price('223oLpuqqg8pZvMkT9sam4', 15.0);
update_discount_price
-----
                33065
(1 row)
```

Seharusnya, nilai `discount_price` data dengan id tersebut berubah. Untuk memastikannya, kita dapat menjalankan *query* sebelumnya.

```
airel.camilo=# SELECT * FROM discounts WHERE id='223oLpuqqg8pZvMkT9sam4';
 id          | items_id | discount_price | original_price | percentage | description | created_at
-----+-----+-----+-----+-----+-----+-----
 223oLpuqqg8pZvMkT9sam4 | 13647   |          33065 |          38900 | 15.0       |              | 2024-02-23 09:10:46
(1 row)
```

Untuk menerapkan *function* `update_discount_price()` ke semua baris data di tabel `discounts`, kita bisa menjalankan *query* berikut.

```
SELECT update_discount_price(id, 15.0);
```

Note: Jika menjalankan *query* tersebut, perlu setelahnya mengulang pengisian tabel.

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

Selanjutnya, untuk menghapus suatu *function* yang sudah dibuat, kita dapat menjalankan *query* berikut.

```
DROP FUNCTION function_name;  
-- jika nama function unik, atau  
DROP FUNCTION function_name([<arg1>, <arg2> , ...]);
```

Contoh 2:

Untuk menghapus *function* `update_discount_price()`, kita dapat menjalankan *query* berikut.

```
DROP FUNCTION update_discount_price;  
-- atau  
DROP FUNCTION update_discount_price(VARCHAR, DECIMAL);
```

Trigger

Trigger merupakan operasi pada sebuah tabel yang **otomatis dijalankan** ketika ada kejadian tertentu. Kejadian ini bisa berupa ketika melakukan INSERT, UPDATE, atau DELETE. Agar *trigger* bisa bekerja, kita perlu **membuat *stored procedure* terlebih dahulu**, baru kemudian **menyambungkan suatu *trigger* dengan *stored procedure* tersebut ke suatu tabel**. Format sintaks PL/SQL yang digunakan untuk membuat *stored procedure* yang akan dipanggil *trigger* ini sama seperti *function* biasa. Akan tetapi, **harus bernilai *trigger* dan tidak boleh mempunyai argumen**.

Contoh 4:

Misalkan terdapat suatu kasus di mana kita ingin mencegah penambahan atau pengubahan data dalam tabel `items` dengan nilai `category` yang belum tersimpan di tabel tersebut. Untuk mengimplementasi batasan tersebut, kita perlu membuat *trigger*-nya terlebih dahulu.

```
CREATE OR REPLACE FUNCTION check_category_exists() RETURNS TRIGGER AS  
$$  
  DECLARE  
    category_exists BOOLEAN;  
  BEGIN  
    -- Cek apabila category ada dalam tabel  
    SELECT EXISTS(  

```

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

```
SELECT 1
FROM items
WHERE LOWER(category) = LOWER(NEW.category)
) INTO category_exists;

IF NOT category_exists THEN
    -- Raise exception jika category tidak ada
    RAISE EXCEPTION 'Category % does not exist', NEW.category;
END IF;

RETURN NEW;
END;
$$
LANGUAGE plpgsql;
```

Setelah itu, barulah membuat *trigger*-nya. Berikut adalah sintaks untuk membuat *trigger*.

```
CREATE [OR REPLACE] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF} {event [OR ...]}
ON table_name
[FOR [EACH] {ROW | STATEMENT}]
EXECUTE PROCEDURE trigger_function
```

NOTE:

Dalam konteks ini, “events” merujuk pada peristiwa yang akan memicu suatu *trigger*, seperti INSERT, UPDATE [OF column_name [, ...]], DELETE, atau TRUNCATE. Jika *trigger* diatur dengan “FOR EACH ROW”, maka *trigger function* akan dipanggil untuk setiap baris data yang terpengaruh oleh *query* yang memicu *trigger* tersebut. Sebagai contoh, jika *trigger* tersebut dipicu oleh *query* yang menambahkan N baris, maka *trigger function* tersebut akan dipanggil N kali. Di sisi lain, jika *trigger* diatur dengan “FOR EACH STATEMENT”, *trigger function* tersebut hanya dipanggil satu kali untuk setiap *query* yang memicu *trigger* tersebut.

Contoh 5:

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

Berdasarkan sintaks tersebut, kita dapat membuat *trigger* yang akan menjalankan *function* `check_category_exists()` setiap sebelum INSERT atau UPDATE tabel items.

```
CREATE TRIGGER prevent_invalid_category
BEFORE INSERT OR UPDATE OF category ON items
FOR EACH ROW
EXECUTE FUNCTION check_category_exists();
```

Untuk memeriksa apakah *trigger* tersebut berjalan atau tidak, kita bisa menjalankan salah satu *event* yang memicu *trigger* tersebut.

Berikut adalah contoh *query* INSERT data ke tabel items yang nilai *category*-nya tidak tersimpan dalam tabel.

```
INSERT INTO items VALUES('820818', 'A8208180002275', 'Indomie Mi Instan Goreng 120 g', 'Snack',
'https://c.alfagift.id/product/1/1_A8208180002275_20240304103450733_base.jpg',
'https://alfagift.id/p/820818', 'alfagift', '2024-15-04 14:11:17');
```

```
airel.camilo=# INSERT INTO items VALUES('820818', 'A8208180002275', 'Indomie Mi Instan Goreng 120 g', 'Snack',
', 'https://c.alfagift.id/product/1/1_A8208180002275_20240304103450733_base.jpg', 'https://alfagift.id/p/820818', 'alfagift', '2024-15-04 14:11:17');
ERROR:  Category Snack does not exist
CONTEXT:  PL/pgSQL function check_category_exists() line 14 at RAISE
```

Sedangkan berikut adalah contoh *query* INSERT data ke tabel items yang nilai *category*-nya tersimpan dalam tabel.

```
INSERT INTO items VALUES('820818', 'A8208180002275', 'Indomie Mi Instan Goreng 120 g', 'Makanan',
'https://c.alfagift.id/product/1/1_A8208180002275_20240304103450733_base.jpg',
'https://alfagift.id/p/820818', 'alfagift', '2024-15-04 14:11:17');
```

```
airel.camilo=# INSERT INTO items VALUES('820818', 'A8208180002275', 'Indomie Mi Instan Goreng 120 g', 'Makanan',
', 'https://c.alfagift.id/product/1/1_A8208180002275_20240304103450733_base.jpg', 'https://alfagift.id/p/820818', 'alfagift', '2024-15-04 14:11:17');
INSERT 0 1
```

Latihan Soal 2

1. **[SQL]** Jalankan seluruh contoh diatas.
2. **[SQL]** Buat suatu *trigger* yang dapat menambahkan nilai `created_at` secara otomatis

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

ketika menambahkan item di tabel items.

Note: SQL *trigger* tersebut perlu disertakan *screenshot*-nya.

Kemudian, jalankan *query* untuk menambahkan suatu item dan menampilkan item tersebut. Misalkan menjalankan *query* berikut ini.

```
INSERT INTO items VALUES('820850', 'A8208500002275', 'Frisian Flag Susu Cair Uht  
Matcha 225ML', 'susu',  
'https://c.alfagift.id/product/1/1_A8208500001465_20241504103467263_base.jpg',  
'https://alfagift.id/p/820850', 'alfagift', NULL);  
  
SELECT * FROM items WHERE id = '820850';
```

Note: Kedua *query* di atas (INSERT dan SELECT) beserta *output*-nya perlu disertakan *screenshot*-nya. Pastikan kedua *query* beserta *output*-nya tersebut berada pada 1 *screenshot*.

3. **[SQL]** Buat suatu *trigger* yang memicu suatu *function* yang dapat memvalidasi bahwa nilai *percentage* tidak boleh di luar rentang 0-100. Jika di luar rentang tersebut, maka *raise exception*. *Trigger* tersebut akan dijalankan ketika menambahkan item di tabel discounts.

Note: SQL *function* dan *trigger* tersebut perlu disertakan *screenshot*-nya.

Kemudian, jalankan *query* untuk menambahkan suatu item berikut ini.

```
INSERT INTO discounts VALUES('224fvhJL743b8cG6Lh3Gji', '770517', 0, 21500, 101.0,  
null, '2024-04-15 22:10:35');  
  
INSERT INTO discounts VALUES('224fvhJL743b8cG6Lh3Gji', '770517', 19350, 21500,  
10.0, null, '2024-04-15 22:10:35');
```

Note: Sertakan *screenshot* masing-masing *query* di atas beserta *output*-nya.

4. **[SQL]** Buat suatu *trigger* yang memicu suatu *function* yang memvalidasi bahwa nilai SKU tidak boleh duplikat sehingga akan menampilkan pesan exception jika SKU item yang akan **ditambahkan** sudah ada pada tabel items.

CSGE602070 Basis Data
Semester Genap 2023/2024
Tutorial 3

Triggered & Stored Procedure, Big Data

Note: SQL *function* dan *trigger* tersebut perlu disertakan *screenshot*-nya.

Kemudian, jalankan *query* untuk menambahkan suatu item berikut ini.

```
INSERT INTO items VALUES( 'Jy2hq44sf2PKt8ivAx6Gha',  
'16bcc729-09aa-4d93-a0ac-be137ddc866f', 'Cimory Susu Cair Uht Chocomint 250ml',  
'susu', 'https://assets.klikindomaret.com/products/20133605/20133605_1.jpg',  
'https://www.klikindomaret.com/product/fresh-milk-uht-10', 'klikindomaret', NULL);
```

```
INSERT INTO items VALUES( 'Jy2hq44sf2PKt8ivAx6Gha',  
'16bcc729-09aa-4d93-a0ac-be137ddc866x', 'Cimory Susu Cair Uht Chocomint 250ml',  
'susu', 'https://assets.klikindomaret.com/products/20133605/20133605_1.jpg',  
'https://www.klikindomaret.com/product/fresh-milk-uht-10', 'klikindomaret', NULL);
```

Note: Sertakan *screenshot* masing-masing *query* di atas beserta *output*-nya.