STORED PROCEDURE

Basis Data Lanjut

FUNGSI DAN MANFAAT *STORED PROCEDURE*

Stored Procedure adalah sekumpulan perintah SQL yang disusun dalam sebuah *procedure* (mirip dengan di pemrograman biasa) yang memiliki nama dan fungsi tertentu.

Stored procedure merupakan grup *transact SQL* (T-SQL). Jika anda memiliki kondisi dimana anda harus menuliskan *query* yang sama berulang kali, maka anda dapat menyimpan query tersebut ke dalam *stored procedure* dan apabila dibutuhkan tinggal memanggil procedure yang telah dibuat.

Pengertian lain menyebutkan, *store procedure* adalah program yang disimpan dalam database seperti halnya data. *Store procedur* memiliki manfaat antara lain :

- Dapat digunakan kapanpun Seperti halnya pembuatan *procedure* pada bahasa pemrograman. Apabila sebuah program cukup banyak memiliki proses yang akan dikerjakan, akan lebih baik program tersebut dipecah menjadi bagian-bagian kecil (*procedure*). Fungsi utama/program utama hanya tinggal memanggil bagian-bagian program tersebut. Seperti halnya pada basis data.
- Lebih cepat dan efisien karena bersifat *server side* Jika ingin membuat program yang cukup besar, pembuatan program *server sid*e akan terasa lebih mudah dibanding *client side*. *Server side* akan bersifat netral terhadap semua aplikasi, disisi lain database administrator tidak perlu mengetahui terlalu mendalam terhadap bahasa pemrograman
- Mudah dibuat dan dirawat karena kecil tetapi "*Power Full*"

Struktur penulisan store procedure pada SQL Server adalah sebagai berikut :

```
CREATE PROCEDURE <namaprocedure>
  <@Param1,sysname,@p1> <Datatype_For_Param1,,int> = <Default_Value_For_Param1,,0>
AS
BEGIN
  statement
END
```

Struktur penulisan untuk mengubah/memodifikasi STORE PROCEDURE yang telah dibuat :

```
ALTER PROCEDURE <namaprocedure>
  <@Param1,sysname,@p1> <Datatype_For_Param1,,int> = <Default_Value_For_Param1,,0>
AS
BEGIN
  statement
END
```

Untuk pemanggilan store procedure yang telah dibuat :

```
EXEC <namaprocedure> <@Param1, @Param2>
```

CONTOH STORE PROCEDURE

Misalkan ingin membuat store procedure untuk menampilkan data barang dengan kategori tertentu.

Maka querynya adalah sebagai berikut :

```
CREATE PROCEDURE spBarangPerKategori(@kategori VARCHAR(50)) AS
BEGIN TRANSACTION
	SELECT i.item_id, i.name, i.category_id, c.explanation
	FROM ITEMS i JOIN CATEGORIES c ON i.category_id = c.category_id
	WHERE c.explanation = @kategori
IF @@ERROR=0
	COMMIT TRANSACTION
ELSE
	ROLLBACK TRANSACTION
```

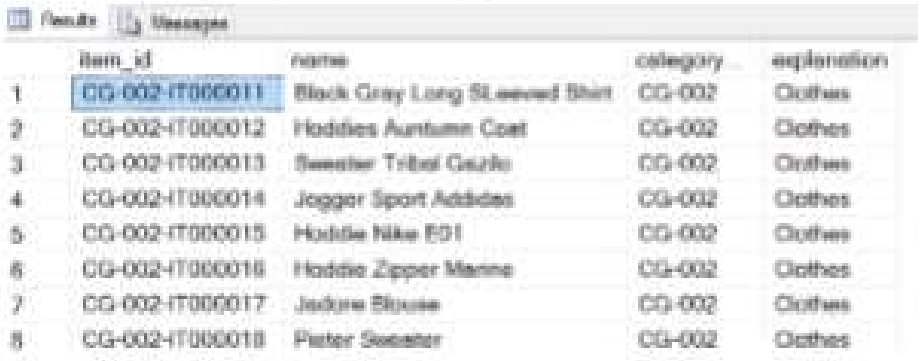
Keterangan :
Dibuat stored procedure dengan nama *spBarangPerKategori* dengan menambahkan 1 parameter yaitu *@kategori* bertipe *VARCHAR(50).*

Kemudian dibuat query untuk menyeleksi data barang beserta kategori barang dimana kategori barang sesuai dengan inputan pada saat eksekusi procedure.

Selanjutnya eksekusi query tersebut. Jika view menampilkan data/informasi barang yang dibutuhkan tanpa ada filter/kriteria, dengan menggunakan *store procedure* dapat menampilkan databarang .dengan menambahkan kriteria tertentu. Untuk menjalankan *store procedure* tersebut maka querynya adalah :

```
EXEC spBarangPerKategori 'Clothes'
```

Maka hasil outputnya adalah sebagai berikut :



	item_id	name	category	explanation
1	CG-002-IT000011	Black Gray Long SLeeved Shirt	CG-002	Clothes
2	CG-002-IT000012	Hoddies Auntumn Coat	CG-002	Clothes
3	CG-002-IT000013	Sweater Tribal Gazilo	CG-002	Clothes
4	CG-002-IT000014	Jogger Sport Addidas	CG-002	Clothes
5	CG-002-IT000015	Hoddie Nike E01	CG-002	Clothes
6	CG-002-IT000016	Hoddie Zipper Mienne	CG-002	Clothes
7	CG-002-IT000017	Jadyne Blouse	CG-002	Clothes
8	CG-002-IT000018	Pieter Sweater	CG-002	Clothes

Contoh lain :
STORED PROCEDURE MENGGUNAKAN PARAMETER OUTPUT
Stored procedure untuk menghitung berapa jumlah barang berdasarkan kategori yang diinputkan.

```
CREATE PROCEDURE spJumlahItemPerKategori (@kategori VARCHAR(50), @jumlah INT OUTPUT)
AS
BEGIN TRANSACTION
	SELECT @jumlah = COUNT(*)
	FROM Items i JOIN Categories c ON i.category_id = c.category_id
	WHERE C.explanation = @kategori
IF @@ERROR=0
	COMMIT TRANSACTION
ELSE
	ROLLBACK TRANSACTION
```

Keterangan :
Variable @*kategori* merupakan parameter yang digunakan untuk memberikan kondisi kategori
tertentu (*explanation*). Variabel @*jumlah* merupakan variable yang digunakan untuk menampung hasil output.

Untuk menjalankan stored procedure tersebut

```
DECLARE @jumlahBarang INT
EXEC spJumlahItemPerKategori 'Clothes', @jumlahBarang OUTPUT
PRINT @jumlahBarang
```

Atau

```
DECLARE @jumlahBarang INT
EXEC spJumlahItemPerKategori @jumlah = @jumlahBarang OUT, @kategori = 'Clothes'
PRINT @jumlahBarang
```

Maka hasilnya adalah :

Namun apabila dimodifikasi menjadi : (keyword OUT/OUTPUT pada jumlahBarang dihilangkan)

```
DECLARE @jumlahBarang INT
EXEC spJumlahItemPerKategori 'Clothes', @jumlahBarang
PRINT @jumlahBarang
```

Maka hasilnya tidak akan muncul.

Ini menunjukkan bahwa keyword **OUTPUT** harus ditambahkan, jika tidak @jumlahBarang akan NULL.

Contoh 3 STORED PROCEDURE DENGAN RETURN VALUE

```
CREATE PROCEDURE spJumlahSupplier
AS
BEGIN
	RETURN (SELECT COUNT(*) FROM Suppliers)
END
IF @@ERROR = 0
	COMMIT TRANSACTION
ELSE
	ROLLBACK TRANSACTION
```

Stored Procedure vs. View

A Stored Procedure:

- accepts parameters
- can NOT be used as building block in a larger query
- can contain several statements, loops, IF ELSE, etc.
- can perform modifications to one or several tables
- can NOT be used as the target of an INSERT, UPDATE or DELETE statement.

A View:

- does NOT accept parameters
- CAN be used as building block in a larger query
- can contain only one single SELECT query
- can NOT perform modifications to any table
- but can (sometimes) be used as the target of an INSERT, UPDATE or DELETE statement.

TERIMAKASIH