

## PEMROGRAMAN BASIS DATA

### Stored Procedure

**Kamarudin, M.Kom**

[kamarudin@amikom.ac.id](mailto:kamarudin@amikom.ac.id)

<http://coding4ever.net/>

<https://github.com/rudi-krsoftware/open-retail>

# Definisi

- ✓ Stored procedure merupakan sekumpulan perintah T-SQL untuk melakukan tugas tertentu yang diproses sebagai satu kesatuan.
- ✓ Secara sederhana dapat dikatakan sebagai sebuah sub-program yang berisi *logic* aplikasi dan tersimpan di database.
- ✓ Mengandung kode program (T-SQL) yang melakukan operasi di dalam database, termasuk memanggil stored procedure lainnya.
- ✓ Menerima parameter sebagai media input dan mengembalikan nilai-nilai dalam bentuk parameter output.
- ✓ Stored procedure sangat mirip dengan method pada beberapa bahasa pemrograman.

# Ilustrasi

## ■ Tanpa stored procedure

### Module #1

```
DECLARE @i INT

SET @i = 1
WHILE (@i <= 10)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```

### Module #2

```
DECLARE @i INT

SET @i = 5
WHILE (@i <= 15)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```

### Module #3

```
DECLARE @i INT

SET @i = 10
WHILE (@i <= 20)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```

## ■ Menggunakan stored procedure

### Module #1

```
CREATE PROCEDURE SP_PERULANGAN (
    @MULAI INT, @SELESAI INT
)
AS
BEGIN
    WHILE (@MULAI <= @SELESAI)
    BEGIN
        PRINT @MULAI
        SET @MULAI = @MULAI + 1
    END
END
```

### Module #2

```
EXEC SP_PERULANGAN 1, 10
```

### Module #3

```
EXEC SP_PERULANGAN 5, 15
```

# Keuntungan menggunakan SP

- ✓ Penggunaan ulang kode yang sama
- ✓ Mengurangi lalu lintas jaringan.
- ✓ Eksekusi program menjadi lebih cepat.
- ✓ Lebih mudah dimaintenance
- ✓ Dapat digunakan untuk mekanisme keamanan.

# Aturan dalam pembuatan SP

Yang perlu diperhatikan pada saat membuat store procedure :

- ✓ Stored procedure hanya dapat dibuat pada database yang aktif/ yang sedang dipakai.
- ✓ Biasanya, nama stored procedure dimulai dengan awalan `sp_` contohnya `sp_add_product`, `sp_edit_product`, dll.
- ✓ Nama stored procedure harus sesuai dengan tujuan atau maksudnya.

# Membuat Stored Procedure

```
CREATE PROCEDURE sp_nama_stored_procedure (  
    @parameter_1 TIPE_DATA_1, ...,  
    @parameter_n TIPE_DATA_n OUTPUT  
)  
AS  
    DECLARE @nama_variabel TIPE_DATA  
BEGIN  
    perintah_perintah_T-SQL  
END
```

## Keterangan :

- ✓ Parameter sifatnya opsional dan bisa mempunyai nilai default
- ✓ Parameter merupakan media input untuk melewatkan data yang bersumber dari luar stored procedure.
- ✓ Ada 2 jenis parameter yaitu :
  - INPUT
  - OUTPUT

# Cara Memanggil Store Procedure

Untuk memanggil stored procedure gunakan perintah ***EXECUTE*** atau disingkat ***EXEC***.

```
-- memanggil SP tanpa parameter  
EXEC sp_cetak
```

```
-- memanggil SP menggunakan parameter  
EXEC sp_add_product 'Modem Bolt', 400000, 500000
```

```
-- alternatif memanggil SP menggunakan parameter  
EXEC sp_add_product @nama_product = 'Modem Bolt',  
    @hrg_beli = 400000, @hrg_jual = 500000
```

```
-- alternatif memanggil SP menggunakan parameter  
-- urutan parameter bebas  
EXEC sp_add_product @hrg_jual = 500000, @hrg_beli = 400000,  
    @nama_product = 'Modem Bolt'
```

# Contoh #1

Contoh stored procedure *tanpa parameter*

```
DECLARE @i INT

SET @i = 1
WHILE (@i <= 10)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```



```
CREATE PROCEDURE sp_perulangan
AS
    DECLARE @i INT
BEGIN
    SET @i = 1
    WHILE (@i <= 10)
    BEGIN
        PRINT @i
        SET @i = @i + 1
    END
END
```

Contoh pemanggilan :

```
EXEC sp_perulangan
```



# Contoh #2 (Lanjutan)

Contoh stored procedure *menggunakan parameter*

```
DECLARE @i INT

SET @i = 1
WHILE (@i <= 10)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```



```
CREATE PROCEDURE sp_perulangan2 (
    @mulai INT, @selesai INT
)
AS
BEGIN
    WHILE (@mulai <= @selesai)
    BEGIN
        PRINT @mulai
        SET @mulai = @mulai + 1
    END
END
```

Contoh pemanggilan :

```
EXEC sp_perulangan2 4, 10
EXEC sp_perulangan2 @mulai = 4, @selesai = 10
```

# Contoh #3 (Lanjutan)

Contoh stored procedure *menggunakan parameter default*

```
DECLARE @i INT

SET @i = 1
WHILE (@i <= 10)
BEGIN
    PRINT @i
    SET @i = @i + 1
END
```



```
CREATE PROCEDURE sp_perulangan3 (
    @mulai INT = 1, @selesai INT = 5
)
AS
BEGIN
    WHILE (@mulai <= @selesai)
    BEGIN
        PRINT @mulai
        SET @mulai = @mulai + 1
    END
END
```

Contoh pemanggilan :

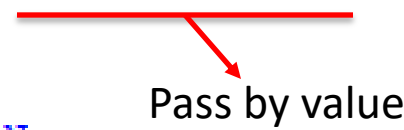
```
EXEC sp_perulangan3
EXEC sp_perulangan3 5, 10
EXEC sp_perulangan3 @mulai = 7, @selesai = 15
```

# Contoh #4 (Lanjutan)

Contoh stored procedure *menggunakan parameter* **OUTPUT**

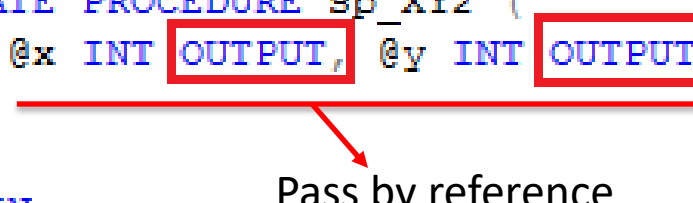
```
CREATE PROCEDURE sp_XY1 (  
    @x INT, @y INT  
)  
AS  
BEGIN  
    SET @x = @x + 2  
    SET @y = @y * 2  
END
```

Pass by value



```
CREATE PROCEDURE sp_XY2 (  
    @x INT OUTPUT, @y INT OUTPUT  
)  
AS  
BEGIN  
    SET @x = @x + 2  
    SET @y = @y * 2  
END
```

Pass by reference



Contoh pemanggilan :

```
DECLARE @nilaiX INT  
DECLARE @nilaiY INT  
  
SET @nilaiX = 2  
SET @nilaiY = 3  
PRINT CAST(@nilaiX AS VARCHAR) + ', ' + CAST(@nilaiY AS VARCHAR)  
  
EXEC sp_XY1 @nilaiX, @nilaiY  
PRINT CAST(@nilaiX AS VARCHAR) + ', ' + CAST(@nilaiY AS VARCHAR)  
  
EXEC sp_XY2 @nilaiX OUTPUT, @nilaiY OUTPUT  
PRINT CAST(@nilaiX AS VARCHAR) + ', ' + CAST(@nilaiY AS VARCHAR)
```

# Contoh #5 (Lanjutan)


Contoh stored procedure *menggunakan parameter* **OUTPUT**

```
DECLARE @nilaiAngka INT
DECLARE @nilaiHuruf CHAR(1)

SET @nilaiAngka = 3

IF (@nilaiAngka = 4)
    SET @nilaiHuruf = 'A'
ELSE IF (@nilaiAngka = 3)
    SET @nilaiHuruf = 'B'
ELSE IF (@nilaiAngka = 2)
    SET @nilaiHuruf = 'C'
ELSE
    SET @nilaiHuruf = 'D'

PRINT 'Nilai = ' + @nilaiHuruf
```




```
CREATE PROCEDURE sp_nilaiAngkaKeHuruf (
    @nilaiAngka INT, @nilaiHuruf CHAR(1) OUTPUT
)
AS
BEGIN
    IF (@nilaiAngka = 4)
        SET @nilaiHuruf = 'A'
    ELSE IF (@nilaiAngka = 3)
        SET @nilaiHuruf = 'B'
    ELSE IF (@nilaiAngka = 2)
        SET @nilaiHuruf = 'C'
    ELSE
        SET @nilaiHuruf = 'D'
END
```

Contoh pemanggilan :

```
DECLARE @nilHuruf CHAR(1)

EXEC sp_nilaiAngkaKeHuruf 3, @nilHuruf OUTPUT

PRINT 'Nilai = ' + @nilHuruf
```

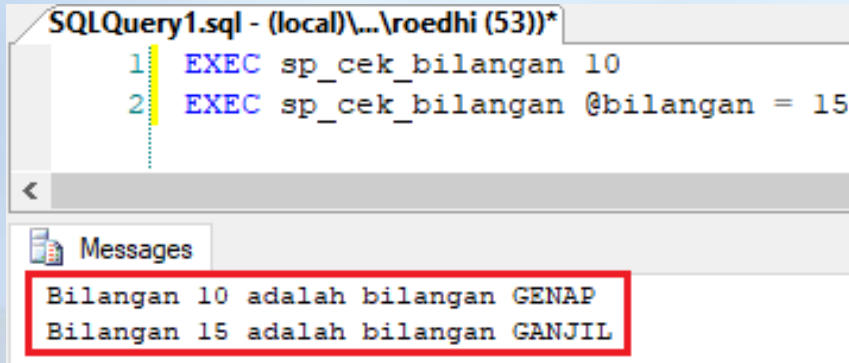


# Latihan

Latihan stored procedure *menggunakan parameter*.

1. Buatlah stored procedure untuk mencetak bilangan ganjil/genap dengan nama ***sp\_cek\_bilangan***.

Jika store procedure tersebut dipanggil akan menghasilkan output seperti berikut :



The screenshot shows a SQL Server Enterprise Manager interface. The top window is titled 'SQLQuery1.sql - (local)\...\roedhi (53))\*' and contains two lines of SQL code: '1 EXEC sp\_cek\_bilangan 10' and '2 EXEC sp\_cek\_bilangan @bilangan = 15'. Below this is a 'Messages' window with a red border containing the output: 'Bilangan 10 adalah bilangan GENAP' and 'Bilangan 15 adalah bilangan GANJIL'.

```
SQLQuery1.sql - (local)\...\roedhi (53))*
1 EXEC sp_cek_bilangan 10
2 EXEC sp_cek_bilangan @bilangan = 15
```

Messages

Bilangan 10 adalah bilangan GENAP  
Bilangan 15 adalah bilangan GANJIL

# Latihan (Lanjutan)

Jawaban :

```
CREATE PROCEDURE sp_cek_bilangan (  
    @bilangan INT  
)  
AS  
    DECLARE @keterangan VARCHAR(6)  
BEGIN  
    IF @bilangan % 2 = 0  
        SET @keterangan = 'GENAP'  
    ELSE  
        SET @keterangan = 'GANJIL'  
  
    PRINT 'Bilangan ' + CAST(@bilangan AS VARCHAR) +  
        ' adalah bilangan ' + @keterangan  
END
```