8. PEMBUATAN STRUKTUR KONTROL

Objektif

Setelah mengikuti materi ini, mahasiswa dapat :

1. Pemahaman statement perulangan.
2. Pemahaman statement percabangan.

8.1. Statement IF

Statement IF digunakan untuk mengeksekusi sebuah perintah jika kondisi yang telah ditetapkan terpenuhi.

Sintaks:

IF <condition>
    statement;

Contoh :

```
DECLARE
@VDEGGRE CHAR(1)
SELECT @VDEGGRE ='U'
IF @VDEGGRE = 'U'
    PRINT 'Undergraduate';
```

8.2. Statement IF – ELSE

Sama seperti statement IF, statement IF ELSE digunakan untuk mengeksekusi sebuah perintah jika kondisi yang telah ditetapkan terpenuhi. Ketika kondisi yang telah ditetapkan tidak terpenuhi maka perintah yang akan dieksekusi adalah perintah yang berada pada statemen ELSE.

Sintaks:

IF < condition1>

statement1; IF <

condition 2>

statement2; ELSE

statement3;

Contoh :

```
DECLARE
@VDEGGRE CHAR(1)
SELECT @VDEGGRE ='J'
IF @VDEGGRE = 'U'
    PRINT 'Undergraduate';
IF @VDEGGRE = 'M'
    PRINT 'Master';
IF @VDEGGRE = 'P'
    PRINT 'PhD';
ELSE
    PRINT 'Unknown';
```

8.3. Statement CASE

Statement CASE digunakan untuk melakukan evaluasi terhadap beberapa kondisi dan mengembalikan satu hasil dari beberapa kemungkinan hasil yang ada. Statement CASE dapat digunakan dalam pernyataan atau klausaa apapun yang memperbolehkan ekspresi yang valid. Statement CASE dapat digunakan dalam pernyataan seperti SELECT, UPDATE, DELETE, dan SET, dan dalam klausa seperti SELECT_LIST, IN, WHERE, ORDER by, dan HAVING.

Sintaks:

CASE WHEN <condition1> THEN <statement1>
WHEN <condition2> THEN <statement2>
WHEN <condition3> THEN <statement3>
ELSE <statement4>
END

Contoh:

```
DECLARE
@VDEGREE CHAR(1),
@VDEGREE_NAME VARCHAR(20)
SELECT @VDEGREE = 'M'
SELECT @VDEGREE_NAME=
CASE @VDEGREE
WHEN 'U' THEN 'Undergraduate'
WHEN 'M' THEN 'Master'
WHEN 'P' THEN 'PhD'
ELSE 'Unknown'
END
PRINT @VDEGREE_NAME
```

8.4. Statement WHILE

Statement WHILE merupakan klausa perulangan atau looping yang biasa digunakan dalam mengeksekusi satu blok program berulang-ulang hingga kondisi pada WHILE menjadi FALSE.

Sintaks:

WHILE <condition>
BEGIN
    <statement>
END

Contoh:

```
DECLARE @i int
SELECT @i = 5
WHILE @i > 0
BEGIN
    PRINT 'Nilai i = ' + str(@i)
    SELECT @i = @i  - 1
END
```

Perhatikan bahwa fungsi str() diperlukan untuk mengkonversi bilangan integer menjadi string (teks).

8.5. Perintah GOTO

Perintah GOTO digunakan untuk mengubah alur program secara tiba tiba untuk mengeksekusi sebuah blok kode yang diberi label. Contoh penulisan sebuah label adalah: nama_label:;

Contoh :

```
DECLARE @i int
SELECT @i = 0
WHILE @i <= 10
BEGIN
    PRINT 'Nilai i = ' + str(@i)
    SELECT @i = @i  + 1
    IF @i = 6
    GOTO perintah_goto;
END

perintah_goto:;
PRINT ' Mencoba perintah GOTO';
```

8.6. Perintah CONTINUE dan BREAK

Perintah CONTINUE dan BREAK berkaitan dengan statement WHILE. Perintah CONTINUE melanjutkan alur program pada kondisi WHILE, sedangkan perintah BREAK mengakibatkan alur program keluar dari WHILE.

Contoh perintah CONTINUE :

```
SELECT @i = 0
WHILE @i < 6
BEGIN
    SELECT @i = @i + 1
    IF (@i = 4)
        BEGIN
            PRINT 'Continue'
            CONTINUE
        END
    PRINT 'Nilai i =' + str(@i)
END
PRINT 'Penggunaan perintah CONTINUE'
```

Contoh perintah CONTINUE :

```
SELECT @i = 0
WHILE @i < 6
BEGIN
    SELECT @i = @i + 1
    IF (@i = 4)
        BEGIN
            PRINT 'Break'
            BREAK
        END
    PRINT 'Nilai i =' + str(@i)
END
PRINT 'Penggunaan perintah BREAK'
```

8.7. Perintah RETURN

Perintah RETURN akan menghentikan program dari eksekusi. Perintah RETURN memperbolehkan program keluar tanpa harus memenuhi suatu kondisi atau tanpa harus mengeksekusi sebuah statement terlebih dahulu.

Contoh :

```
SELECT @i = 0
WHILE @i < 6
BEGIN
    SELECT @i = @i + 1
    IF (@i = 4)
        BEGIN
            PRINT 'Return'
            RETURN
        END
    PRINT 'Nilai i =' + str(@i)
END
PRINT 'Penggunaan perintah RETURN'
```

8.8. Penanganan ERROR

1. Blok TRY CATCH

Cara kerja blok TRY CATCH adalah, pada bagian blok TRY akan berfungsi sebagai penjaga kode-kode SQL di bawahnya. Apabila baris kode tersebut menimbulkan error maka alur program kemudian akan dilemparkan pada blok CATCH sebagai penanganan error.

Sintaks:

BEGIN TRY;
    <statement1>;
    <statement2>;
END TRY
BEGIN CATCH
    <statement1>;
    <statement2>;
END CATCH;

Contoh;

```
BEGIN TRY;
    PRINT 'Statement1 blok TRY';
    RAISERROR 55550055 'Error yang disimulasikan.';
    PRINT 'Statement2 blok TRY';
END TRY
BEGIN CATCH
    PRINT 'Maaf telah terjadi kesalahan.';
END CATCH;
```

2. Perintah RAISERROR

Pada penanganan kesalahan menggunakan blok TRY CATCH biasanya digunakan juga perintah RAISERROR. Perintah ini digunakan untuk membuat sebuah pesan kesalahan.

Sintaks:

RAISERROR <nomor_erro> <pesan_error>;

Contoh:

```
RAISERROR 55550055 'Error yang disimulasikan.';
```

Nomor error yang digunakan tidak boleh kurang dari 50000 karena semua nomor dibawah itu sudah dimiliki oleh sistem database.