CODE CONVENTION SQL SERVER

User Manual

Naming

- **Penamaan Database**

  Penamaan database menggunakan huruf besar disetiap katanya. Hindari pengunaan underscore, spasi dan Sql Server Keyword untuk penamaan database.

```
SchoolSystem
```

- **Penamaan Table**

  Penamaan table akan menggunakan huruf kecil pada prefix, dan huruf besar disetiap kata. Hindari pengunaan underscore, spasi dan Sql Server Keyword untuk penamaan table.

  Kemudian gunakan prefix untuk memberi keterangan tipe table

  - **Table Master**

    Di awalin dengan menggunakan prefix ms

```
msCustomers
```

  - **Table Transaction**

    Di awalin dengan menggunakan prefix tr

```
trOrders
```

  - **Table History**

    Di awalin dengan menggunakan prefix hs[table name]

```
hmsCustomers
htrOrders
```

  Untuk special case jika nama table adalah akronim(singkatan) maka cara penulisannya adalah semua huruf akronim huruf besar, kemudian awal kata juga diawali dengan huruf besar.

```
msIOConfiguration
```

- **Penamaan Column/Field**

  Penamaan column/field akan menggunakan huruf besar disetiap kata. Hindari pengunaan underscore, spasi dan Sql Server Keyword untuk penamaan column/field.

```
CustomerID
```

- **Penamaan View**

  Penamaan view akan menggunakan huruf besar disetiap katanya. Hindari pengunaan spasi dan Sql Server Keyword, penggunaan. Tambahkan Prefix vw untuk penggunaan view. Gunakan purpose untuk menjelaskan tipe dari tujuan view tersebut. Penamaan View dapat dipolakan seperti berikut ini:

```
vw_[module]_[purpose]
```

```
vw_Customer_GetAllCustomer
```

- **Penamaan Stored Procedure**

  Penamaan Stored Procedure akan menggunakan huruf besar disetiap katanya. Hindari pengunaan spasi dan Sql Server Keyword. Tambahkan `[Get/Insert/Update/Delete]` di depan purpose untuk menjelaskan tipe dari tujuan view tersebut. Gunakan Prefix bn, penamaan stored procedure dapat dipolakan seperti berikut ini:

```
bn_[module]_[Get/Insert/Update/Delete][purpose]
```

```
bn_Customer_GetCustomerByID
```

- **Penamaan Function**

  Penamaan Function akan menggunakan huruf besar disetiap katanya. Hindari pengunaan spasi dan Sql Server Keyword. Gunakan Prefix fn, penamaan function dapat dipolakan seperti berikut ini:

```
fn_[module]_[purpose]
```

```
fn_Student_GetStudenNameByID
```

- **Penamaan Variable**

  Penamaan variable akan menggunakan huruf besar disetiap katanya dan diawali dengan menggunakan @

  Hindari penggunaan underscore, spasi, Sql Server Keyword, dan @@:

```
@BinusianID varchar(10)
```

- **Penamaan Alias Table**

  Penamaan alias menggunakan urutan abjad alphabet, dimana jika ada sub query maka table sub query tersebut ditambahkan hirarki

  Example:

```
SELECT a.Field1, b.Field2
FROM
   (SELECT * FROM Table1 a1)  a
   INNER JOIN Table2 b ON a.FieldPK=b.FieldPK
```

Comment, Description and Indenting Code

Pemberian comment ataupun deskripsi berdasarkan hal-hal dibawah ini

- **Describe Important Flow**

  Pada saat mendeskripsikan sebuah flow dari query yang ada.

  Example:

```
-- If student id exists in msSiblingGroup
IF EXISTS (SELECT StudentID FROM msSiblingGroup WHERE SiblingID = @GroupSibling AND AuditActivity
NOT IN('D'))
   BEGIN
      -- Backup to history table
      INSERT INTO hmsSiblingGroup(StudentID,SiblingID,AuditUserName, AuditTime, AuditActivity)
      SELECT StudentID,SiblingID,AuditUserName, AuditTime, AuditActivity FROM msSiblingGroup WHERE
SiblingID = @GroupSibling AND AuditActivity NOT IN('D')

      -- Update flag data to 'D'
      UPDATE msSiblingGroup
      SET AuditUserName = @AuditUserName,
         AuditTime = Getdate(),
         AuditActivity = 'D'
      WHERE SiblingID = @GroupSibling AND AuditActivity NOT IN('D')

   END
```

  Pemberian comment pada flow hanya dilakukan, jika memang flow tersebut dirasa perlu untuk diberikan keterangan, sehingga akan mempermudah rekan programmer lain dalam memahami flow dalam query

- **Update Code**

  Pada saat melakukan perubahan code. Gunakan keterangan berikut :

  - Modified By : Yang melakukan perubahan
  - Date : Tanggal Terjadi Perubahan
  - Purpose : Alasan dilakukan perubahan code

  Example:

```
/*
        Modified by      : Hendry
        Date             : Mar 11,2014
        Purpose          : Change order condition from SiblingID to StudentID
*/
SELECT * FROM msSiblingGroup WHERE Type NOT IN(1) ORDER BY StudentID
```

  Jika perubahan dilakukan bukan untuk yang pertama kali, tambahkan dibawah keterangan modified yang lama

  Example:

```
/*
        Modified by      : Hendry
        Date             : Mar 10,2014
        Purpose          : Change order condition from SiblingID to StudentID

        Modified by      : Hendry
        Date             : Mar 11,2014
        Purpose          : Change back order condition to StudentID

*/
```

- **Keep Old Code**

  Pada saat anda ingin menyimpan code lama sebagai backup. Gunakan keterangan berikut :

  - Commented By : Yang melakukan comment code lama.
  - Date : Tanggal Terjadi Perubahan
  - Purpose : Alasan tetap menyimpan code lama

  Example:

```
/*
        Commented by     : Hendry
        Date             : Tuesday Feb 11, 2014
        Purpose          : backup code, change code to shorter query

        DECLARE @StudentStatusID char(1)
        SET @StudentStatusID=(SELECT StudentStatusID FROM msStudent WHERE StudentID=@StudentID)
        SELECT StatusDesc FROM ltStudentStatus WHERE StudentStatusID=@StudentStatusID

*/
SELECT StatusDesc FROM ltStudentStatus WHERE StudentStatusID IN (SELECT StudentStatusID FROM
msStudent1 WHERE StudentID=@StudentID)
```

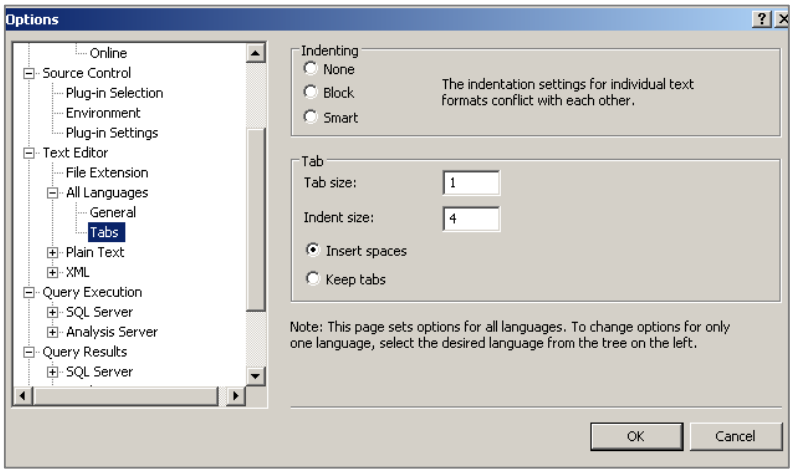
- **Indenting Code**

  Penggunaan indent pada code menggunakan tab, tidak boleh menggunakan spasi. Dimana jika sebuah statement merupakan bagian child stament, maka child statement harus menggunakan 1 indent lebih dalam dari statement parentnya.

  Example:

```
DECLARE @ValStudentID CHAR(10)
IF EXISTS (SELECT StudentID FROM msSiblingGroup WHERE SiblingID = @GroupSibling AND AuditActivity
NOT IN('D'))
   BEGIN
        @ValStudentID=(SELECT StudentID FROM msStudent WHERE StudentID=@StudentID)
   END
ELSE
   BEGIN
        @ValStudentID=(SELECT StudentID FROM msSiblingGroup WHERE SiblingID = @GroupSibling)
   END
```

  Penggaturan identing code pada SQL Management Studio dapat dilakukan melalui option



Structure, Formating, and Code Concept

- **Structure Concept**
  1. Batasi pemberian nama Database, Table, Field, Parameter, dan Variable untuk tidak melebihi 50 karakter.
  2. Pada setiap Table, diwajibkan memiliki salah satu skema tracking audit record, tujuannya adalah untuk mempermudah tracking audit record suatu table.

     a. Skema 1

Field Name	Data Type	Description
Stsrc	CHAR(1)	Status Record terakhir : Active(A), Delete(D)
UserIn	VARCHAR(50)	User yang pertama kali insert
UserUp	VARCHAR(50)	User yang terakhir kali update
DateIn	DATETIME	Tanggal insert data
DateUp	DATETIME	Tanggal update data terakhir kali

     b. Skema 2

Field Name	Data Type	Description
AuditUsername	VARCHAR(50)	User yang memasukan(insert) /terakhir kali mengubah(update)/ menghapus(delete) data.
AuditTime	DATETIME	Tanggal record dimasukan(insert)/terakhir kali diubah(update)/dihapus(dalete).
AuditActivity	CHAR(1)	Status audit terakhir : Insert(I), Update(U), Delete(D)

     Penentuan Skema yang akan digunakan ditentukan berdasarkan kondisi-kondisi berikut:

     - Untuk database yang sudah ada, penentuan skema mengikuti skema tracking audit record yang pada database tersebut.
     - Untuk database baru, maka pemilihan skema diwajibkan menggunakan skema 1.

  3. Jika skema 2, maka setiap table Master dan Transaction wajib memiliki Table History. Penamaan table History, hanya tinggal menambahkan h pada setiap nama tables yang akan di buatkan table historynya.
     Ex: msStudent table historynya menjadi hmsStudents
  4. Pada saat ada perubahan data pada table, baik delete maupun update data. Maka semua data lama harus terlebih dahulu di backup ke table history.
     Example :

```
      -- Backup to history table
      INSERT INTO hmsSiblingGroup(StudentID,SiblingID,AuditUserName, AuditTime,
AuditActivity)  SELECT StudentID,SiblingID,AuditUserName, AuditTime, AuditActivity FROM
msSiblingGroup WHERE SiblingID = @GroupSibling AND AuditActivity NOT IN('D')

      -- Update flag data to 'D'
      UPDATE msSiblingGroup
```

```
SET AuditUserName = @AuditUserName,
   AuditTime = Getdate(),
   AuditActivity = 'D'
WHERE SiblingID = @GroupSibling AND AuditActivity NOT IN('D')
```

5. Gunakan tipe data UNICODE(NCHAR, NVARCHAR, NTEXT), hanya jika table akan menyimpan data yang tidak hanya alphabet karakter, tetapi untuk menyimpan berbagai karakter dari dunia. Misalkan Bahasa Mandarin, Hebrew, etc.
6. Gunakan tipe data CHAR/NCHAR jika memang data sudah memiliki format length yang pasti, contoh BinusianID. Jika format length bisa berbeda-beda maka gunakan tipe data VARCHAR/NVARCHAR.

- **Formatting Concept**
  1. Gunakan huruf besar pada semua SQL Keywords

     Example:

     ```
     SELECT StudentID FROM msSiblingGroup WHERE SiblingID = @GroupSibling AND AuditActivity NOT
     IN('D')
     ```
  2. Gunakan indenting code agar mudah membaca code
  3. Gunakan kurung parentheses agar code mudah di baca

     Example:

     ```
     WHERE (StatusPayment='P' AND (Size = 1 OR Size = 2))
     ```

- **Code Concept**
  1. Pada penggunaan statement SELECT, maka nama semua field yang inggin di panggil wajib di sebutkan, tidak diperkenankan menggunakan SELECT * untuk memanggil keseluruhan field pada statement tersebut .

     Example:

     ```
     -- False
     SELECT * FROM msStudent WHERE AuditActivity NOT IN('D')

     -- Correct
     SELECT StudentID,StudentName,Address FROM msStudent WHERE AuditActivity NOT IN('D')
     ```
  2. Gunakan SET NOCOUNT ON pada awal stored procedure. Hal ini dilakukan untuk meningkatkan performance dengan mengurangi trafik network, karena dengan penggunakan SET NOCOUNT ON akan menghilangkan pesan seperti 1 row(s) affected setelah melaksanakan perintah INSERT, UPDATE, DELETE,dan SELECT.

     Example:

     ```
     BEGIN
             SET NOCOUNT ON;
             SELECT StudentID,StudentName,Address FROM msStudent WHERE AuditActivity NOT IN('D')
     END
     ```

3. Gunakan nama field lengkap pada saat melakukan INSERT. Hal ini digunakan untuk menghindari error jika terjadi perubahan pada struktur table.
   Example:

```
-- False
INSERT INTO hmsSiblingGroup
VALUES(@StudentID,@SiblingID,@AuditUserName,@AuditTime,@AuditActivity)

-- Correct
INSERT INTO hmsSiblingGroup(StudentID,SiblingID,AuditUserName, AuditTime, AuditActivity)
VALUES(@StudentID,@SiblingID,@AuditUserName,@AuditTime,@AuditActivity)
```

4. Gunakan error handling.
   Example:

```
BEGIN TRY
  -- SQL Statement
END TRY
BEGIN CATCH
  DECLARE @ErrorMessage NVARCHAR(4000);
  DECLARE @ErrorSeverity INT;
  DECLARE @ErrorState INT;
  SELECT ERROR_NUMBER() AS ErrorNumber,ERROR_MESSAGE() AS ErrorMessage;
  SELECT @ErrorMessage = ERROR_MESSAGE(), @ErrorSeverity = ERROR_SEVERITY(), @ErrorState =
ERROR_STATE();
  RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState)
 END CATCH
```

5. Gunakan BEGIN TRAN Statement, jika terdiri dari beberapa proses yang harus di eksekusi secara bersamaan, tujuannya agar proses dapat di rollback jika terjadi kesalahan pada salah satu proses. Jika anda menggunakan juga TRANSACTION pada sisi aplikasi, maka BEGIN TRAN di SP tidak wajib dituliskan. TRANSACTION di maksudkan untuk menjaga integarasi data.
   Example:

```
BEGIN TRY
  BEGIN TRAN
          -- Backup to history table
      INSERT INTO hmsSiblingGroup(StudentID,SiblingID,AuditUserName, AuditTime,
AuditActivity)
      SELECT StudentID,SiblingID,AuditUserName, AuditTime, AuditActivity FROM
msSiblingGroup WHERE SiblingID = @GroupSibling AND AuditActivity NOT IN('D')

      -- Update flag data to 'D'
      UPDATE msSiblingGroup
      SET AuditUserName = @AuditUserName,
         AuditTime = Getdate(),
         AuditActivity = 'D'
      WHERE SiblingID = @GroupSibling AND AuditActivity NOT IN('D')
  COMMIT TRAN
END TRY
BEGIN CATCH
  ROLLBACK TRAN;
  DECLARE @ErrorMessage NVARCHAR(4000);
  DECLARE @ErrorSeverity INT;
  DECLARE @ErrorState INT;
  SELECT ERROR_NUMBER() AS ErrorNumber,ERROR_MESSAGE() AS ErrorMessage;
  SELECT @ErrorMessage = ERROR_MESSAGE(), @ErrorSeverity = ERROR_SEVERITY(), @ErrorState =
ERROR_STATE();
  RAISERROR(@ErrorMessage, @ErrorSeverity, @ErrorState)
 END CATCH
```

6. Gunakan local variable tables(@table) daripada menggunakan temporary table(#table), untuk mengurangi unnecessary lock pada system tables, juga untuk meningkatkan performance karena variable tables dibuat di RAM yang lebih cepat diakses di bandingkan physical disk.

Example:

```
-- False
CREATE TABLE #TempTable
(
 SchoolLevelID VARCHAR(3),
 YearLevelID TINYINT,
 ClassPackageSetID INT
)

-- True
DECLARE @TempTable TABLE
(
 SchoolLevelID VARCHAR(3),
 YearLevelID TINYINT,
 ClassPackageSetID INT
)
```

7. Penamaan Link Server hindari penggunaan IP, untuk Link Server harap menggunakan nama alias.
8. Untuk tipe Field dengan tanggal atau waktu, gunakan tipe data DateTime dan jangan menggunakan tipe data VarChar,char,etc. Hal ini untuk memudahkan formating pada query dan filtering field.