Stored Procedures

A **stored procedure** is a set of Structured Query Language (SQL) statements with an assigned name, which are **stored** in a relational database management system as a group, so it can be reused and shared by multiple programs.

Create procedure basic syntax

Declaration

```
CREATE PROCEDURE
<SchemaName>.<ProcedureName>
AS
BEGIN

  --Your code ..
  SELECT

END
```

Usage

```
Execute <schemaName>.<ProcedureName>
```

Example

```
CREATE PROCEDURE HomePro.GetAllCustomers
AS
BEGIN
        Select
                CustomerId, FirstName, LastName, ..
        From HomePro.Customers
END
-----
ALTER PROCEDURE HomePro.GetAllCustomers
AS
BEGIN
                .......
END
```

```
Execute HomePro.GetAllCustomers
```

Or

```
Exec HomePro.GetAllCustomers
```

Naming conventions

- Schema name
- Procedures action name: GET, SET, UPDATE and so on
- Actions detail: AllClients, Clients without schedule
- Alias or owner name for distinguish

Examples:

- HomePro.GetAllCustomers_Andrey
- Bank.GetClientsNoSchedules_Andrey

How to see the Stored Procedure code.

```
exec sp_helptext [HomePro.GetEstimationsWithPercentage_Andrey]
```

Parameters

Declaration

```
CREATE PROCEDURE Bank.GetClientsByAge_Andrey
        @Age int
AS
BEGIN
    select ClientId, FirstName, LastName
    from Bank.Clients
    where age > @Age
END
```

Usage

```
EXEC Bank.GetClientsByAge_Andrey
        @Age = 10
```

Verify the passed value of parameter

```
CREATE PROCEDURE Bank.GetClientsByAge_Andrey
      @Age int
AS
BEGIN
   if (@Age < 10 or @Age > 100)
   begin
      Raiserror ('The parameter Age is not valid ', 16,10);
      Return
   end

   select ClientId, FirstName, LastName, Age
   from Bank.Clients
   where age > @Age
END
```

Using variables

```
declare @TotalEstimation numeric(10,2)

select @TotalEstimation = sum (Estimation)
from HomePro.Quotes


select
   Q.Estimation,
   Q.Estimation/@TotalEstimation * 100 as PercentOfTotal,
   @TotalEstimation as TotalEstimation

from HomePro.Customers C
   join HomePro.Quotes Q
   on c.CustomerId = Q.CustomerId
```

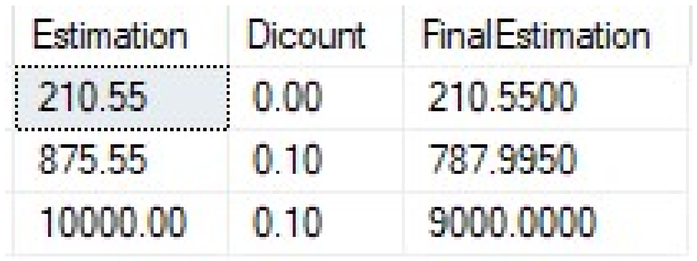
Calculate result base on conditions: CASE expression

```
CASE
    WHEN Boolean_expression THEN result_expression
    [WHEN Boolean_expression THEN result_expression]
    [ ...n ]
    [ ELSE else_result_expression ]
END
```

Example: apply the given discount to the eligible purchases

```
declare @Discount numeric(10,2)  = 0.10
declare @EligibleAmount  numeric(10,2) = 500

select
        Q.Estimation,
        case when Q.Estimation > @EligibleAmount then @Discount
                else 0 end as Discount,
        case when Q.Estimation > @EligibleAmount then Q.Estimation *(1- @Discount)
                else Q.Estimation end as FinalEstimation
from HomePro.Customers C
        join HomePro.Quotes Q
        on c.CustomerId = Q.CustomerId
```



Estimation	Dicount	FinalEstimation
210.55	0.00	210.5500
875.55	0.10	787.9950
10000.00	0.10	9000.0000