Ways to improve performance in SQL

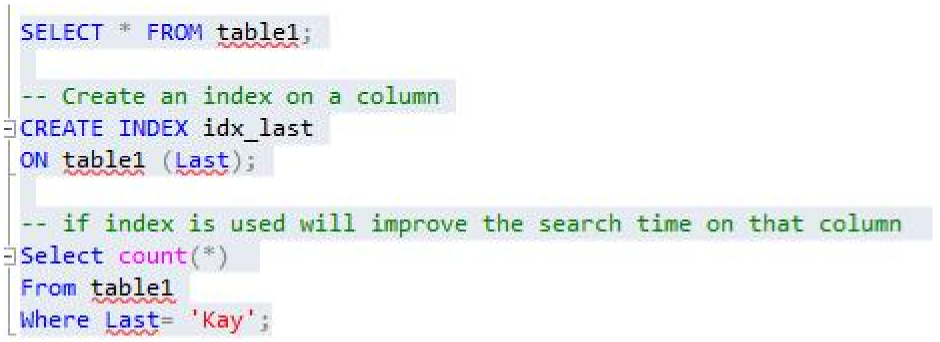
I. INDEX

Index is used to speed up searches/queries when retrieving data from tables with 1000.. of records

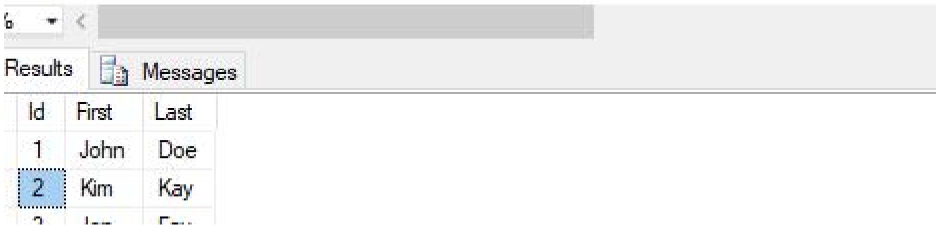
Syntax: CREATE INDEX

Code examples

<u>Creating a single column index:</u>



```
SELECT * FROM table1;

-- Create an index on a column
CREATE INDEX idx_last
ON table1 (Last);

-- if index is used will improve the search time on that column
Select count(*)
From table1
Where Last= 'Kay';
```



Id	First	Last
1	John	Doe
2	Kim	Kay

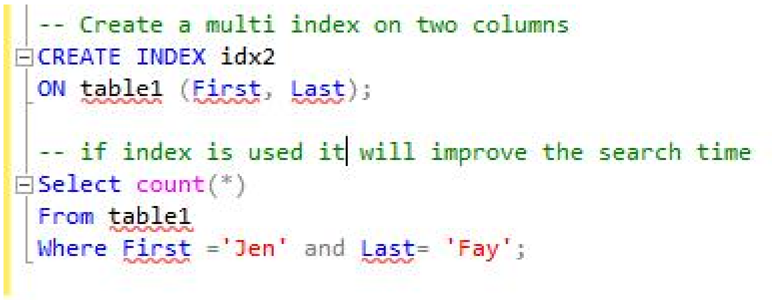
Note: When creating an index on a column, if combine that column with another column (eg see below), the speed time will also improve

SELECT count(*)

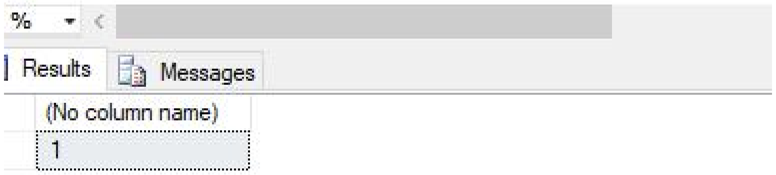
FROM mytable1

WHERE Last = ‘Kay’ and First = 'Kim';

Creating a multi column index

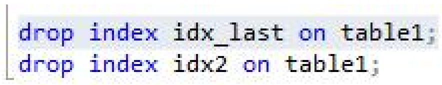


```
-- Create a multi index on two columns
CREATE INDEX idx2
ON table1 (First, Last);

-- if index is used it will improve the search time
Select count(*)
From table1
Where First ='Jen' and Last= 'Fay';
```

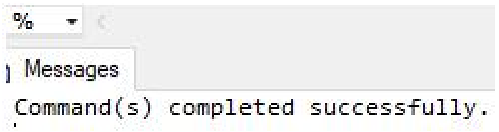


```
Results   Messages
(No column name)
1
```

Syntax: DROP INDEX



```
drop index idx_last on table1;
drop index idx2 on table1;
```



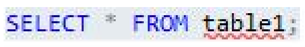
```
Messages
Command(s) completed successfully.
```

II. SQL vs PL/SQL, T-SQL

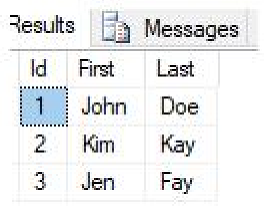
SQL (Structured Query Language) is used to perform DML, DDL operations. PL/SQL (Oracle), T-SQL (MS SQL Server and Azure SQL database) are Procedural Language Extensions to SQ that allow to work with block of statements one time. The ability to Define and Execute Procedures, Functions and packages leads to higher productivity.

<u>Syntax Differences:</u>

SQL standard syntax:



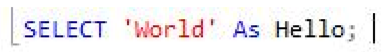
```
SELECT * FROM table1;
```



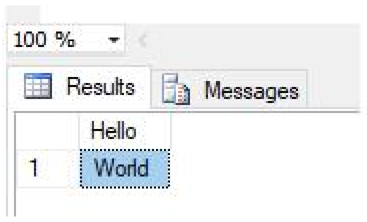
Id	First	Last
1	John	Doe
2	Kim	Kay
3	Jen	Fay

<u>SQL vs PL/SQL vs T-SQL :</u>

SQL print syntax:

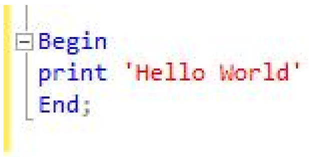


```
SELECT 'World' As Hello;
```

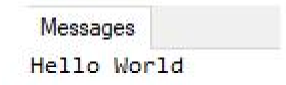


```
  Hello
1 World
```

T-SQL syntax:

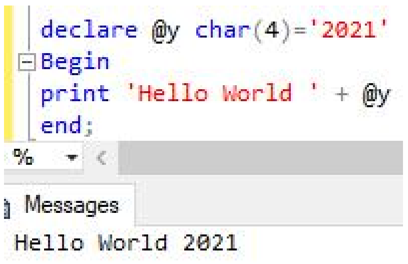


```
Begin
print 'Hello World'
End;
```



```
Hello World
```

To print variables in T-SQL: DECLARE, BEGIN, END:



```
declare @y char(4)='2021'
Begin
print 'Hello World ' + @y
end;
```

```
Hello World 2021
```

PL/SQL print syntax:

SET SERVEROUTPUT ON

```
BEGIN
dbms_output.put_line('Hello World');
END;
```

To print variables in PL/SQL:

- use '||'
- Variables: declared, initialized
- There are no brackets, instead there are terminators eg: IF...END IF

```
DECLARE
  d DATE:='28-Sept-2021';
  b BOOLEAN;
   n INT :=20;
BEGIN
IF n/2 = 2
    b:=TRUE;
ELSE
    b:=FALSE;
END IF;
dbms_output.put_line('Even or odd ' ||d)
END;
```

NOTE: n :=20 (assignment) vs n=2 (comparison)

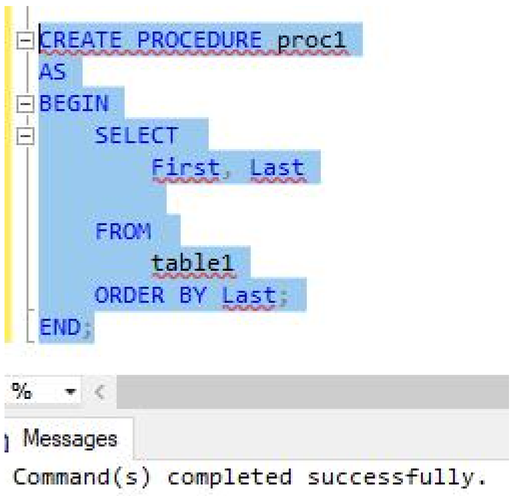
If need to get input from user := &

III. PROCEDURES

(Stored) Procedure : a block of code that can be reused.

T-SQL Syntax: CREATE PROCEDURE

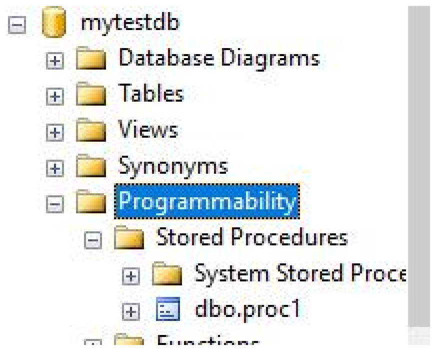
#AS separates the heading and the body of the procedure



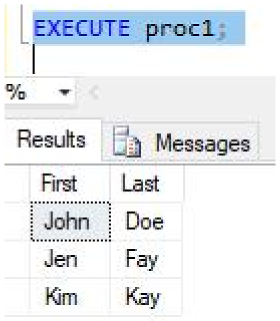
```
CREATE PROCEDURE proc1
AS
BEGIN
    SELECT
        First, Last

    FROM
        table1
    ORDER BY Last;
END;
```

```
Command(s) completed successfully.
```

NOTE: Refresh the Object Explorer, then go to Programmability -> Stored Procedures to see the procedure saved in the DB:



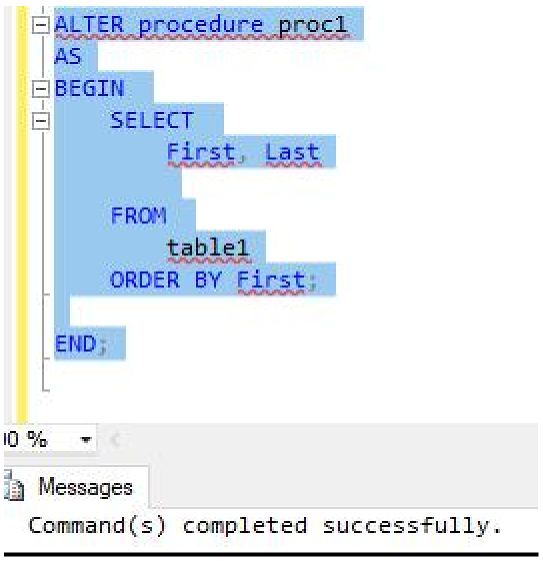
Syntax to execute a procedure: EXECUTE or EXEC



```
EXECUTE proc1;
```

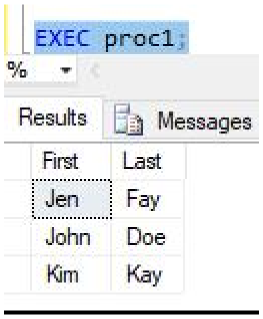
	First	Last
1	John	Doe
2	Jen	Fay
3	Kim	Kay

Syntax to modify an existing procedure: ALTER PROCEDURE



```
ALTER procedure proc1
AS
BEGIN
    SELECT
        First, Last

    FROM
        table1
    ORDER BY First;

END;
```

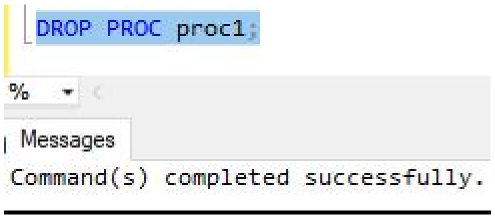
```
0 %
Messages
Command(s) completed successfully.
```



```
EXEC proc1;
%
Results  Messages
```

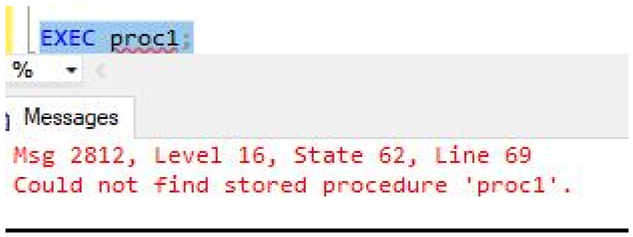
First	Last
Jen	Fay
John	Doe
Kim	Kay

<u>**Syntax to delete a procedure: DROP PROCEDURE or DROP PROC**</u>



```
DROP PROC proc1;
%
Messages
Command(s) completed successfully.
```

Note: check if procedure was deleted:



```
EXEC proc1;
```

Messages

```
Msg 2812, Level 16, State 62, Line 69
Could not find stored procedure 'proc1'.
```

IV. FUNCTIONS

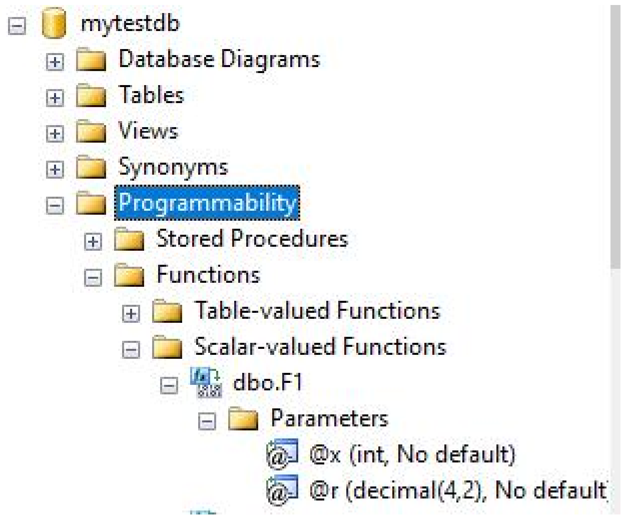
Functions are similar to procedures in that they store a block of code that can be reused. The difference:

A procedure performs an action.

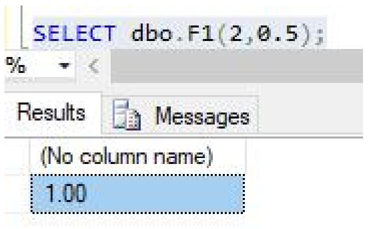
A functions returns a value

Syntax for user-defined functions in T-SQL: Create Function

To check if the function was created: refresh Object Explorer, then go to Programmability/Functions/Scalar-valued Functions:



Now the function can be used like any other build-in function to make calculations:



```
SELECT dbo.F1(2,0.5);
```

```
(No column name)
1.00
```

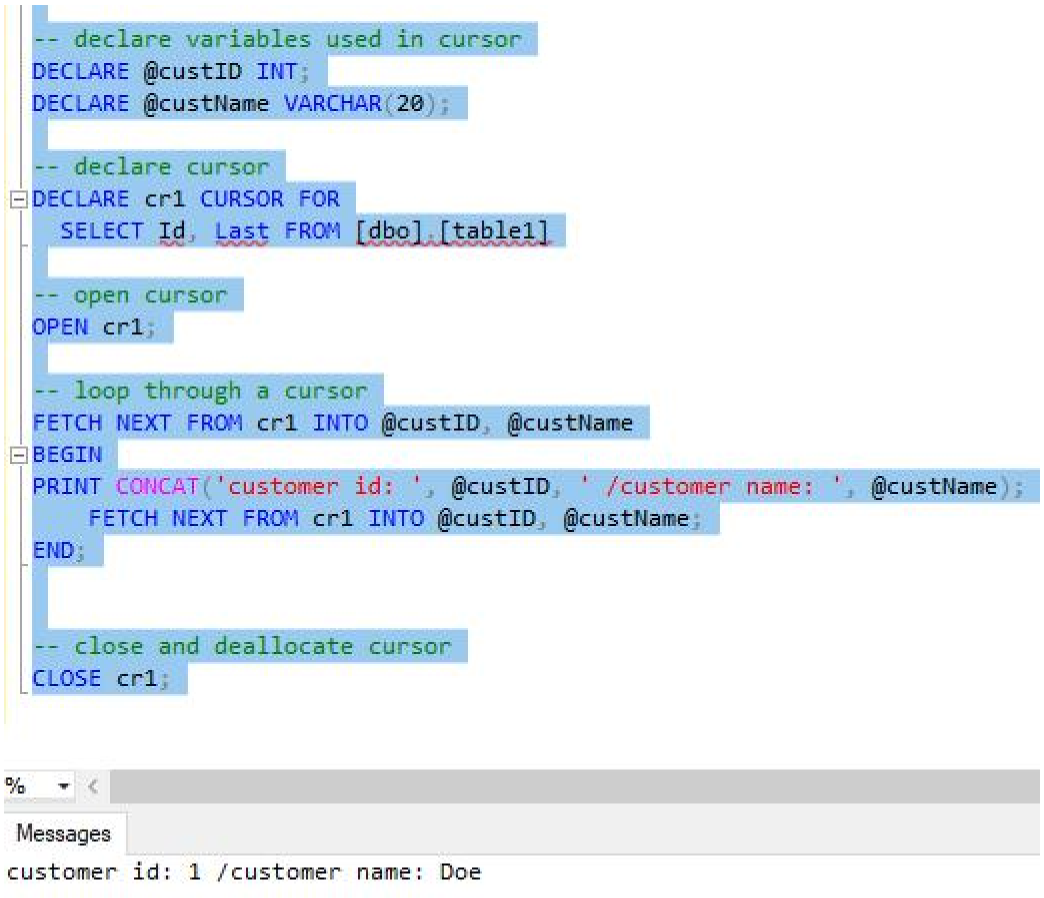
V. CURSORS

Cursor: can hold multiple rows that can be returned (loop through) by SQL statement.

Cursors: Implicit and explicit(user-defined)

- cursors are used for database administration tasks e.g., backups, integrity checks, rebuilding indexes. etc.

Structure of a cursor: Declare , Open, Fetch, Close.



```
-- declare variables used in cursor
DECLARE @custID INT;
DECLARE @custName VARCHAR(20);

-- declare cursor
DECLARE cr1 CURSOR FOR
  SELECT Id, Last FROM [dbo].[table1]

-- open cursor
OPEN cr1;

-- loop through a cursor
FETCH NEXT FROM cr1 INTO @custID, @custName
BEGIN
PRINT CONCAT('customer id: ', @custID, ' /customer name: ', @custName);
    FETCH NEXT FROM cr1 INTO @custID, @custName;
END;


-- close and deallocate cursor
CLOSE cr1;
```

```
customer id: 1 /customer name: Doe
```