```
select * from Employees
select LastName , FirstName from Employees
```

Alians

```
select LastName , FirstName as name from Employees
select LastName , FirstName " الاسم الاول" from Employees
select LastName , FirstName  الاسمالاصلي from Employees

--math operations
select Extension , Extension+2 as new_extension from Employees
select Extension , Extension+2 from Employees

select * from Customers
.
--DDL Data definition language
create database GazaUniversity
Drop database GazaUniversity
create database GazaUniversity
Drop Table employee

--To create your table inside your new database choose your
database name then add your table
```

```sql
create table student (st_NO int
, St_Name nvarchar(20) )
nchar


select lastName , employeeID *2+1 from
Employees
select lastName , 12 * (employeeID+2-1)
from Employees
select lastName , 12 * (employeeID+2-1)
as new_Number from Employees

// distinct لمنع تكرار السجلات

select distinct city from Employees
select  city from Employees


//DOING MATH.OPERATIONS WITH NULL
RECORDS
SELECT REPORTSTO FROM Employees
 SELECT REPORTSTO*2 FROM Employees
```

```
//COMPUTING CLOUMS
SELECT * FROM Products
 SELECT UnitPrice*UnitsInStock FROM
Products

//Filter Rows (condition)
 1. SELECT * FROM Products
     where ProductID = 2

 2. SELECT * FROM Products
     where UnitPrice>30

//AND & OR
     1.  SELECT * FROM Products
     where UnitPrice>30 and CategoryID=8

    2. SELECT * FROM Products
     where UnitPrice>30 or CategoryID=8
```

```
//SORT
SELECT Productid ,Productname,Unitprice,
UnitPrice*UnitsInStock as total
 FROM Products
 ORDER BY UnitPrice    تصاعدي


SELECT Productid ,Productname,Unitprice,
UnitPrice*UnitsInStock as total
 FROM Products
 ORDER BY UnitPrice DESC



//Case Expression in Select
1.
Select ProductName , UnitPrice,
Case
When UnitPrice>100 then 'High'
When UnitPrice>70 then 'Good'
```

```
 Else 'Low' End    as State
 From Products
```

2.

```
Select ProductName , UnitPrice,
Case
When UnitPrice>100 then 'High'
When UnitPrice>70 then 'Good'
 Else 'Low' End    as State
 From Products
 Order by State
```

3.

```
Select St_Name , Avg,
Case
When Avg>=50 and avg<=60 then 'F'
When Avg>=60 and avg<=70 then 'D'
When Avg>=70 and avg<=80 then 'C'
When Avg>=80 and avg<=90 then 'B'
When Avg>=90 and avg<=100 then 'A'
 Else 'راسب' End  as state
 From student
```

```
    Select * from students where st_ID
    Not between 5 and 10
    Select * from students where st_ID
```

```
between 5 and 10
Select * from students where st_Name
not between ‘Ali’ And ‘hasan’
```

Midterm

// Using Top

1.

```
select Top (5) * from Products
```

2.

```
select Top (5) * from Products
ORDER BY UnitPrice DESC
```

3.

```
select Top (8) ProductName,SupplierID,UnitPrice from
Products
ORDER BY UnitPrice DESC
```

//NULL

1.

```
Select * from Employees
Where Region is NULL
```

2.

```
Select * from Employees
Where Region is not NULL
```

// Like تشبه

1.

```
Select * from Products
Where ProductName like'Chai'
```

Try to make a spilling mistake

```
Select * from Products
Where ProductName like'Shai'
```

To solve this problem use like and %

%----- means any char or chars

يمكن وضعها في أي مكان في الكلمة

```
Select * from Products
Where ProductName like'%hai'
```

للبحث عن كل المنتجات التي تبدأ بحرف ال c

```
Select * from Products
Where ProductName ='C%'
```

للبحث عن كل المنتجات التي تنتهي بحرف الd

```
Select * from Products
Where ProductName ='%d'
```

للبحث عن كل المنتجات التي تحتوي على حرف الd

```
Select * from Products
Where ProductName ='%d%'
--
Select * from Customers
where City ='London' or City='Madrid'

In Statment بديل للأمر السابق يمكن استخدام ------

Select * from Customers
where City in ('London','Madrid')

Select * from Customers
where City in ('London','Madrid','Berlin')
```

--Using DML to Modify Data

--Insert

-- Insert whole columns

1. Insert into TableNAme values(لا بد من كتابة جميع القيم بالترتيب حسب الجدول)

```
Insert Into student values (156,'sami',3.4)
```

2.Insert some columns values

ادخال بعض بيانات الجدول (بشرط أن تكون البيانات المتروكة تقبل قيمة Nul

```
Insert Into student (studentId,Student_name)
values (157 ,'Rami')
Insert Into Products(ProductName,UnitPrice)
values ('Tea' ,10)
```

---Update

//update values

If you do not use condition you will update all your table

```
Update student set courseName=’cis125’
,score =88
Where stuId= 3
```

--- Delete

//delete

*To delete all data of the table

1. Delete From TableName
2. Truncate Table TableName

Ex: Delete From Students

Or Truncate Table Students

To delete a row

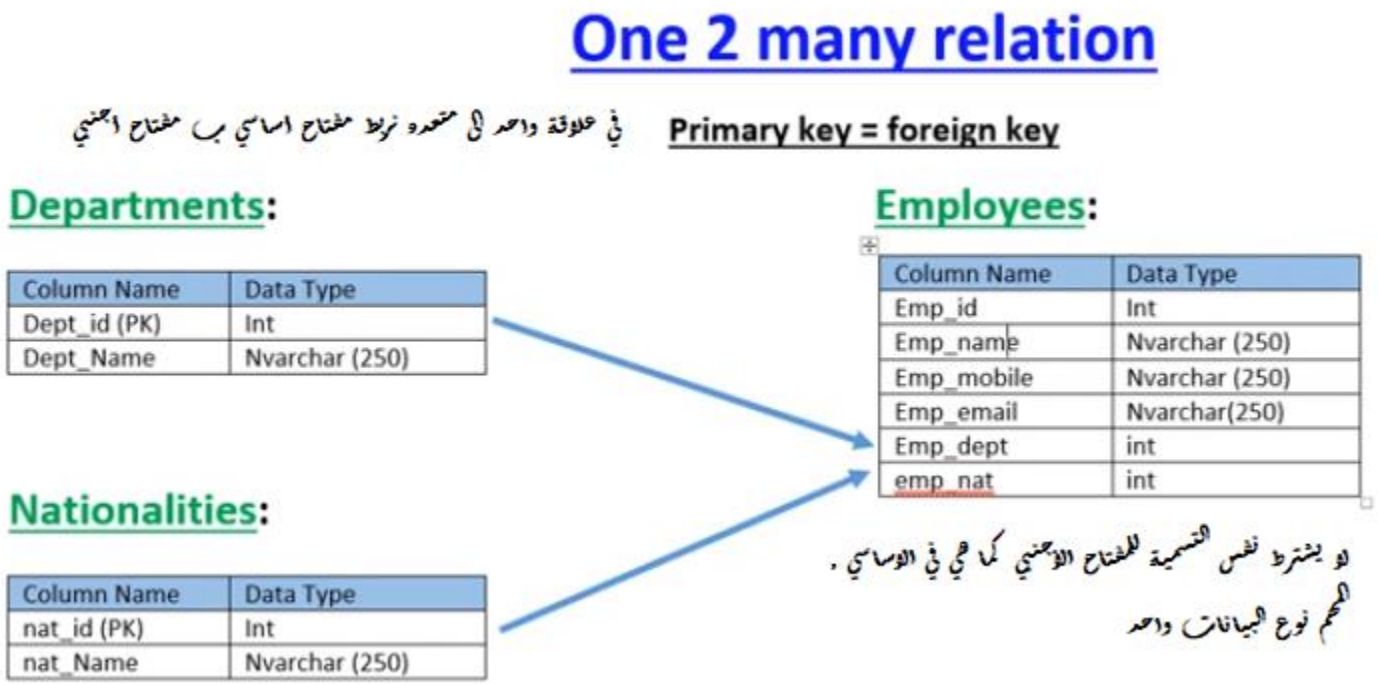
```
Delete From Students where studentId=3
*To Drop All the table
Drop Table employee


//Server2014 DataType
-SmallInt -32,000 to +32.000
-bigInt   ...أكبر بكثير(مليارات
-Floate , real,double(اكبر ها الfloat
-Binary(50) To add (object
picture,video…the max limt
is(ex) 50Byte.
-Bit   0,1
-Char(10) Fixed character  بالحرف…
Varchar…
Nvarchar(20) maximum limit is (4000).
Nvarchar(Max)..up to 2GB for all rows
```

We use it to write notes

Date , datetime,

Geography خريطة




```
select*from department , nationalities ,
employees
where department.dept_id=employees.dept_id
And nationalities.nat_id=employees.nat_id
```

```sql
select LEN (ProductName) as long  from
Products

select LEN (dept_Name)  from department
```

- فحص طول الخلية في عمود

```sql
select LEN (ProductName) as long  from
Products
 where ProductID=10

select LEN (ProductName)  long  from
Products
 where ProductID In (1,10,15)


select LEN (dept_Name) long  from
department
where dept_id = 456
```

```
- - تحويل الاحرف لأحرف صغيرة
- select LOWER ('chai') from Products
    where productid=1

select * from employees
select LOWER ('Amani') from employees
where emp_id=154
--تحويل الاحرف للصيغة الكبيرة

-select UPPER ('chai') from Products
  where productid=1

-select UPPER (ProductName) from Products
……….
select UPPER ('Amani') from employees
where emp_id=154
select UPPER (emp_Name) from employees
……..
```

- فحص هل قيمة الخلية رقمية

```
select ISNUMERIC (ProductID) from Products
select ISNUMERIC (ProductName) from
Products
```

```sql
select ISNUMERIC (emp_id) from employees

select ISNUMERIC (emp_Name) from employees
```

……..

//*inner join

```sql
select CourseId,score,Name from course
inner join student on
course.studentId=student.studentID
 --(FK=PK)
Use northwind
Go;
select top(2) * from Products
select top(2) * from Categories

select ProductName,UnitPrice,categoryName
     from Products   --الجدول الاول
   inner  join Categories    --الجدول الثاني
     on
Products.CategoryID=Categories.CategoryID
```

المفتاح الأجنبي = المفتاح الأساسي لثاني

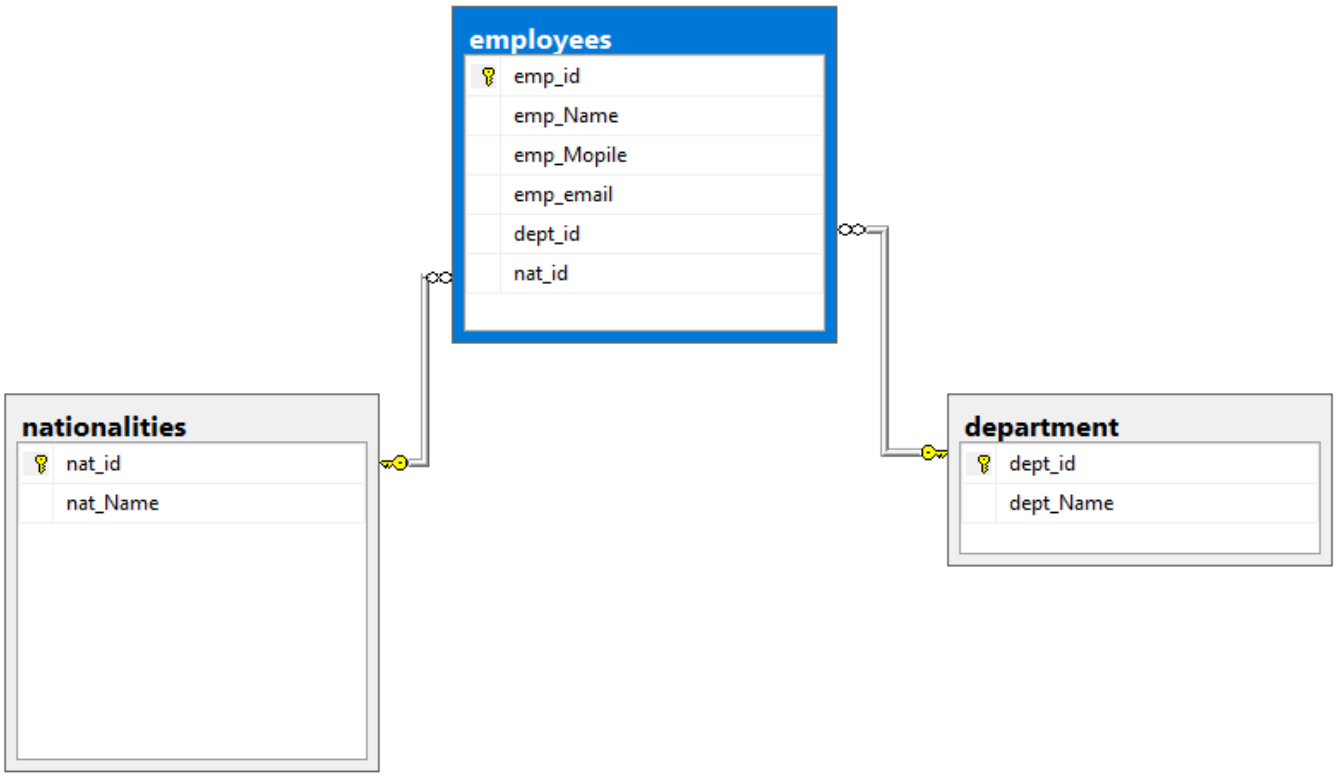
```
SELECT  employees.emp_Name,
employees.emp_email, department.dept_Name
,department.dept_id
 FROM department INNER JOIN employees
 ON department.dept_id = employees.dept_id


-- Inner JOIN WITH MORE THAN ONE TABLE


select top(1) * from Products
select top(1) * from Orders
select top(1) * from [Order Details]


select ProductName,orderDate,Quantity
     from [Order Details]
    inner  join Orders
    on [Order Details].OrderID=Orders.OrderID
    inner join Products
    on [Order Details] .ProductID=Products.ProductID
```

```sql
SELECT
employees.emp_Name,department.dept_Name,
nationalities.nat_Name
FROM department
      INNER JOIN employees
      ON department.dept_id = employees.dept_id
      INNER JOIN nationalities
      ON employees.nat_id = nationalities.nat_id
```



Many to many

```
SELECT student.st_Name ,nat.nat_NAme ,
Teacher.T_Name
 from nat   inner join student
 on nat.nat_ID = student.nat_ID

 inner join TsJoin
 on student.st_ID =TsJoin.st_ID
 inner join Teacher
 on Teacher.T_ID = TsJoin.Teacher
```

Outer join ====➔ Left outer join , Right outer join , Full Outer join

```
--outer join
-- عند تنفيذ الانر جوين فان بيانات الموظف هيثم لن تظهر
بسبب وجود بياناته في جدول الموظفين فقط
select
Employee.EmpName,Department.Dept_Name
from Department Right outer join Employee
```

```
On Employee.Dept_Id=Department.Dept_Id
```

استخدمنا في المثال السابق right لان هيثم موجود في جدول employee و الذي موقعه على يمين كلمة join و ليس له بيانات تربطه بجدول الdepartment أي ليس له رقم قسم يعمل به بعد بينما مضاف لجدول الموظفين

```
select Name ,CourseID , score from student
left outer join
course ON
student.studentID=course.studentId

-- استخدمنا لليسار لأن البيانات المطلوبة موجودة في جدول الطالب
--وهو موجود على يسار الجوين
```

```
select Name ,CourseID , score from student
Right outer join
course ON
student.studentID=course.studentId
-- يستخدم لليمين اذا كانت البيانات المطلوب عرضها موجودة عفي الجدول على يمين الجوين
```

```
-- full join
select Name ,CourseID , score from student
FULL outer join
course ON
student.studentID=course.studentId
```

-- يستخدم لعرض جميع البيانات الموجودةفي الجداول على يمين و يسار الجوين

Self join

```
select  e.FirstName as
employees,m.FirstName as manager
  From Employees as m
  right outer join Employees as e
  on m.EmployeeID =e.ReportsTo
```

.....

```
Select emp.FirstName,emp.LastName,mang.ReportsTo
from Employees as mang
full outer join Employees as emp
 On mang.EmployeeID= emp.ReportsTo

 ---------
 Select
emp.FirstName,emp.LastName,mang.ReportsTo from
Employees as mang
left join Employees as emp
 On mang.EmployeeID= emp.ReportsTo
---------------
select emp.FirstName,emp.LastName,mang.ReportsTo
from Employees as mang
inner join Employees as emp
 On mang.EmployeeID= emp.ReportsTo
```

Cross join : مث الاحتمالات يوزع الجميع (مثل توزيع الطلاب على جميع الأقسام)

```
select * from student
  cross  join department
```

- String Functions
  1--- String Concatenation

- SQL Server uses the + (plus) sign to concatenate characters:
- Concatenating a value with a NULL returns a Null
  EX :

```
select city,region,Country, City + ', '+Region+ ','
+Country as location   from Customers
```

- SQL server introduced the **CONCAT** function ,it Converts NULL to empty string before concatenation.

EX :

```
select city,region,Country,
concat (City , ', '+Region, ',' +Country )
as     location   from Customers
```

2--- SUBSTRING FUNCTION

- SUBSTRING(expression,start,length)
- Returns part of an expression .

```
Ex :  select SUBSTRING ('Ahmed',1,2)
      select SUBSTRING ('Ahmed',3,5)
```
يبدأ العرض من الحرف الثالث حتى الخامس

```
EX : select SUBSTRING (city,1,2) from
     Customers  -- مع اسم عمود
```

3--- DATALENGTH function

- Returns the number of bytes used.
- Remember that LEN returns number of characters.

```
EX : select DATALENGTH (city) from
     Customers
4--- LEFT function

EX :
select 'abcdef'
select left ('abcdef',3) – يرجع أول ٣ حروف من جهة اليسار
select left (city,2) from Customers

5--- Right function
Ex : select right ('abcdef',3)

6--- CHARINDEX function
```

- تبحث عن موقع حرف محدد بالنسبة للجملة المدخلة

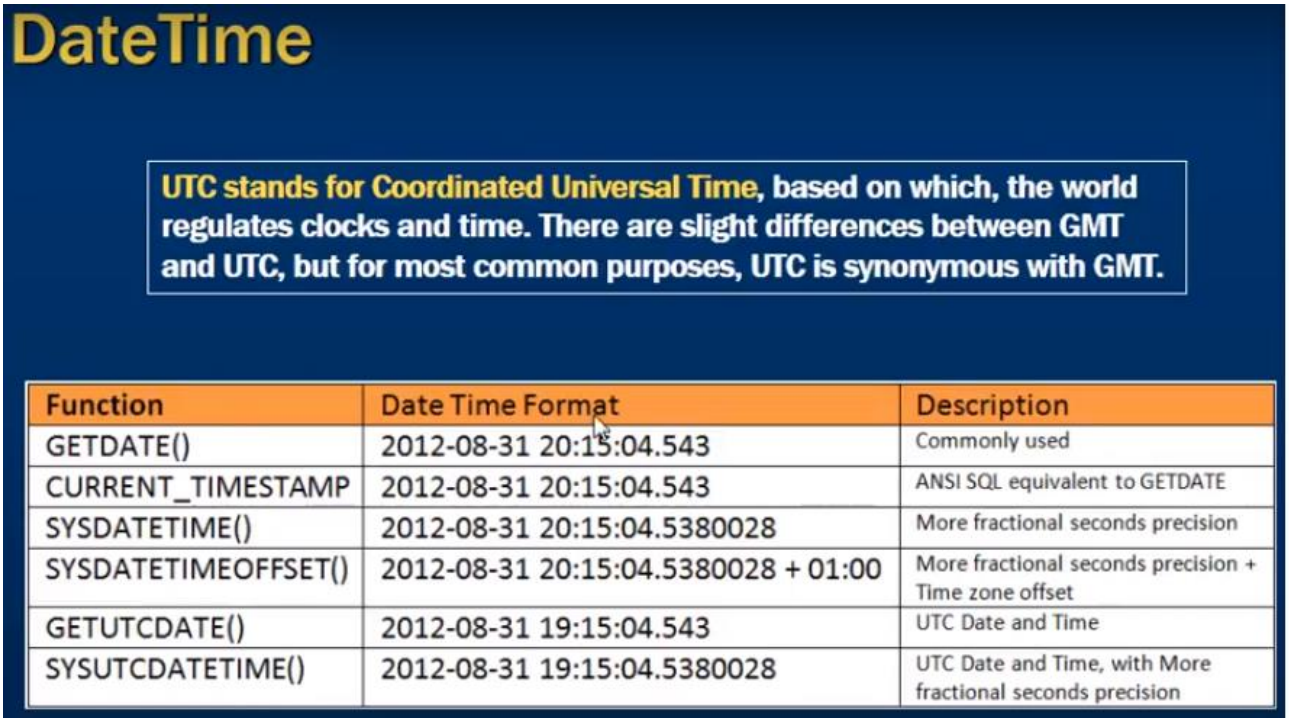
- مثلا يحدد أين موقع الرمز @ في ايميل معين

```
EX : select CHARINDEX('@','Ali@hotmail')
                                    4
```

- `select upper ('Husam')`
- `select lower ('Husam')`

■ Date Time function

There are several built-in DateTime functions available in SQL Server. All the following functions can be used to get the current system date and time, where you have sql server installed.



Function	Date Time Format	Description
GETDATE()	2012-08-31 20:15:04.543	Commonly used
CURRENT_TIMESTAMP	2012-08-31 20:15:04.543	ANSI SQL equivalent to GETDATE
SYSDATETIME()	2012-08-31 20:15:04.5380028	More fractional seconds precision
SYSDATETIMEOFFSET()	2012-08-31 20:15:04.5380028 + 01:00	More fractional seconds precision + Time zone offset
GETUTCDATE()	2012-08-31 19:15:04.543	UTC Date and Time
SYSUTCDATETIME()	2012-08-31 19:15:04.5380028	UTC Date and Time, with More fractional seconds precision

Ex :

```
--GETDATE()--
Select GETDATE()

--CURRENT_TIMESTAMP--
Select CURRENT_TIMESTAMP

--SYSDATETIME()--
Select SYSDATETIME()

--SYSDATETIMEOFFSET()--
Select SYSDATETIMEOFFSET()

--GETUTCDATE()--
Select GETUTCDATE()

--SYSUTCDATETIME()--
Select SYSUTCDATETIME()
```

Data modification

Using INSERT to Add Data

- The INSERT...VALUES statement inserts a single row by default

```
INSERT INTO Sales.OrderDetails(
orderid, productid, unitprice, qty, discount)
VALUES(12000,39,18,2,0.05);
```

- Table and row constructors add multi-row capability to INSERT...VALUES

```
INSERT INTO Sales.OrderDetails(
orderid, productid, unitprice, qty, discount)
VALUES
        (12001,39,18,2,0.05),
        (12002,39,18,5,0.10);
```

اضافة أكثر من صف من خلال أمر واحد: EX

```
insert into Employees(LastName,FirstName,Title)
 values ('husain','Ahmed','Manager') ,
        ('Murad','mohammed','sales Manager'),
       ('Lina','al ali','employee');
```

**** USING INSERT WITH SELECT**

IS USED TO INSERT THE RESULT SET OF A QUERY INTO AN EXISTING TABLE

تستخدم لنقل نتيجة طلب و ادخالها في جدول آخر

EX1 :

```
create table Student (st_name nvarchar(50) , stLastNAme nvarchar (50) )    ---- empty table
```

then>>>

```
insert into Student (st_name ,stLastNAme)
      select firstname,lastName  from Employees
```

ملاحظة / تم انشاء جدول STUDENT في NORTHWIND يحتوي على العمودين المراد نقل البيانات لهما .....

EX2:

المثال التالي ينقل بيانات من جدول ال teacher في قاعدة بيانات اسمها test لجدول ال student الموجود في قاعدة بيانات ال northwind

```
INSERT INTO Student (st_name , stLastname)
          select tname,tlastname from
```

test.[dbo].[teacher]

أمر أخر صيغته كالتالي :

**** select (اسم العمود/أعمدة) into (اسم جدول غير موجود) from (جدول موجود)**

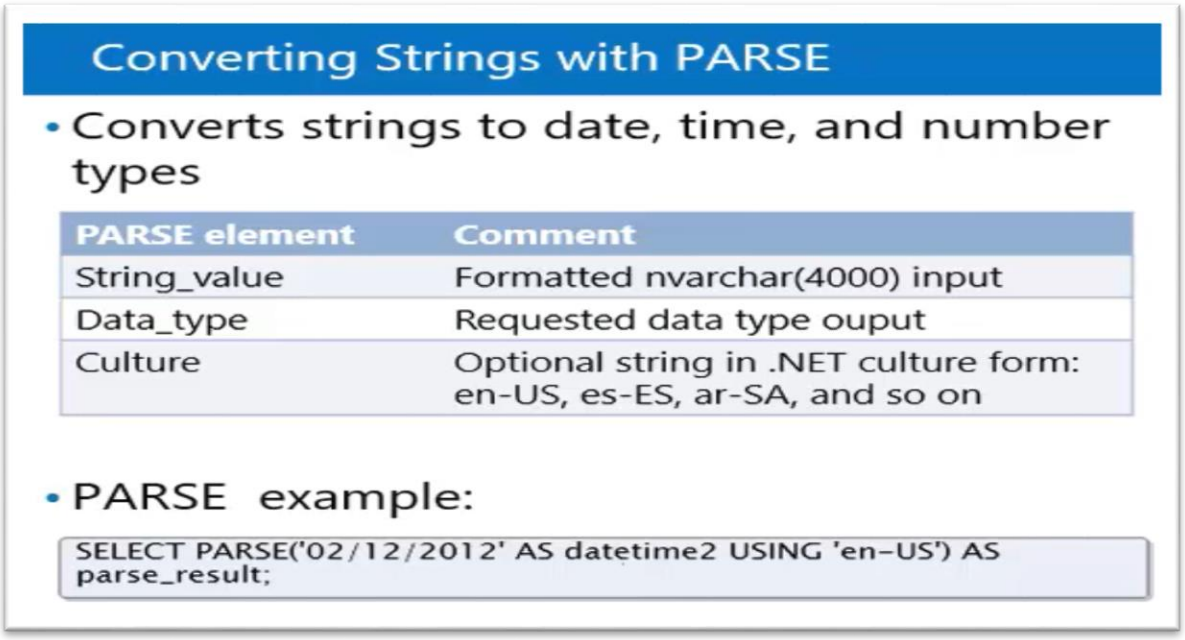
بامكاننا نقل بيانات عمود أو أكثر من جدول موجود في قاعدة البيانات لجدول (غير موجود) يتم انشاءه خلال الامر

```
select LastName,FirstName into newtable
      from Employees
```

لاحظ الجدول newtable غير موجود أصلا ---- الان جرب الأمر التالي

```
select * from newtable
```

** **Built In Function**



Converting Strings with PARSE

- Converts strings to date, time, and number types

PARSE element	Comment
String_value	Formatted nvarchar(4000) input
Data_type	Requested data type ouput
Culture	Optional string in .NET culture form: en-US, es-ES, ar-SA, and so on

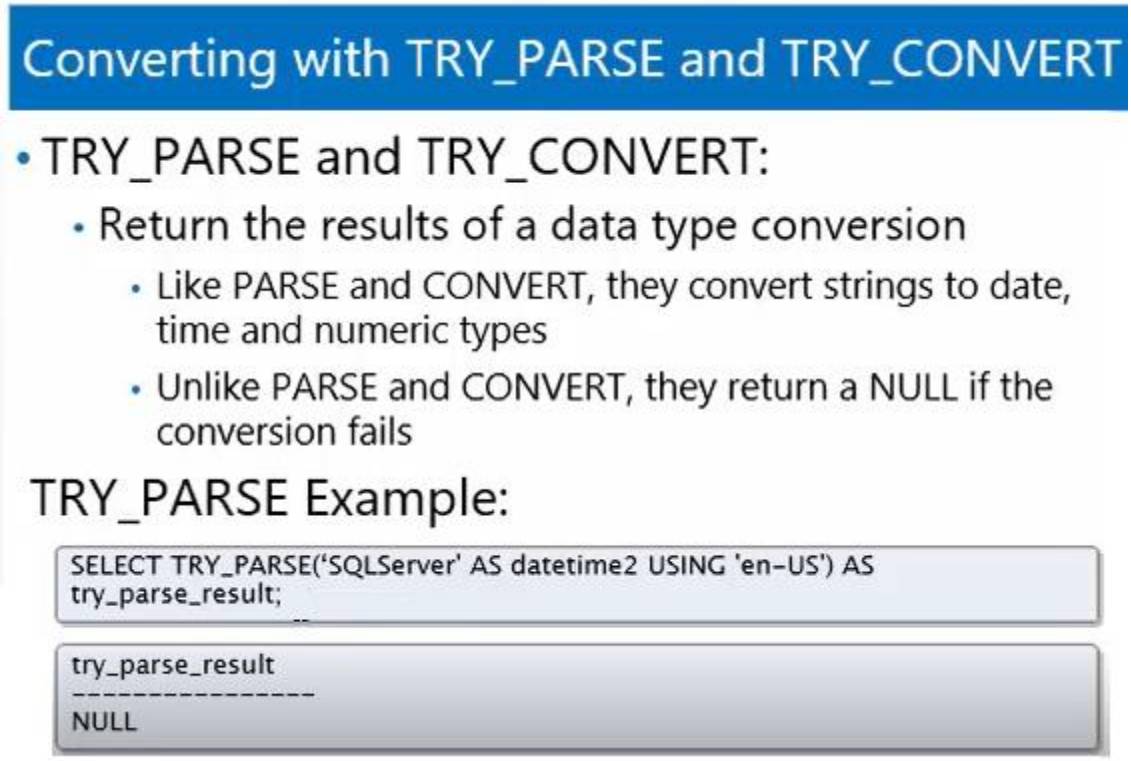
- PARSE example:

```
SELECT PARSE('02/12/2012' AS datetime2 USING 'en-US') AS
parse_result;
```

EX :

```
select PARSE ('15/02/2020 ' as datetime2
using 'AR-eg')  As parse_Result
```

تحول نص الى رقم/وقت وتاريخ



تحول صيغة التاريخ النصية الى صيغة تاريخ/وقت الا أنها تعطي نتيجة Null في حال كان التاريخ المدرج قيمة نصية

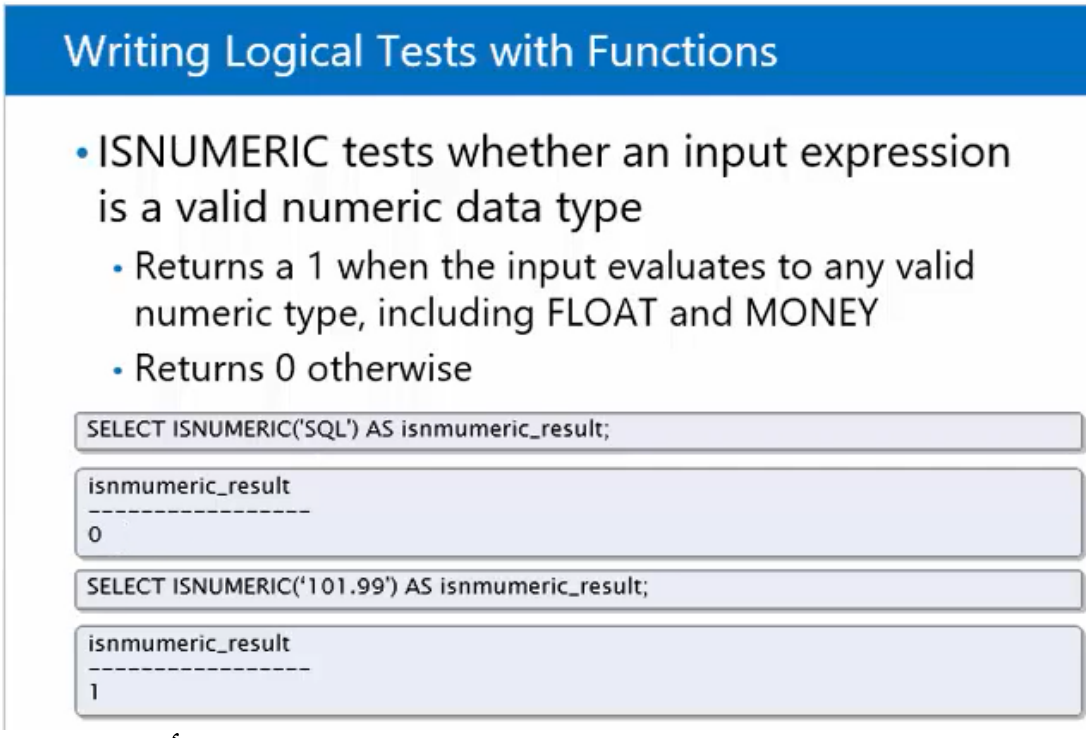
EX :

```
select TRY_PARSE('HelloSql' As datetime
using 'en-us') As TRyParse_Result

--Try_PARSE RETURNS NULL IF FAIL TO CONVERT

select TRY_PARSE('02/15/2020' As datetime
using 'en-us') As TRyParse_Result

try this
select TRY_PARSE('15/02/2020' As datetime using
'en-us') As TRyParse_Result
```



فحص القيمة المدخلة هل هي رقمية أم لا

EX :

```
SELECT ISNUMERIC ('SQL') AS RESULT
SELECT ISNUMERIC ('123') AS RESULT

SELECT ISNUMERIC (CITY) AS RESULT FROM Employees
SELECT ISNUMERIC (EmployeeID) AS RESULT FROM Employees
```

Performing Conditional Tests with IIF

- IIF returns one of two values, depending on a logical test
- Shorthand for a two-outcome CASE expression

IIF Element	Comments
Boolean_expression	Logical test evaluating to TRUE, FALSE, or UNKNOWN
True_value	Value returned if expression evaluates to TRUE
False_value	Value returned if expression evaluates to FALSE or UNKNOWN

```
SELECT    productid, unitprice,
                IIF(unitprice > 50, 'high','low') AS pricepoint
FROM Production.Products;
```

قاعدة `iif` كما الموجود في برنامج الاكسل

EX :

```
Select ProductName,UnitPrice,
iif(unitprice>100 ,'High','Low')as state
from Products
```

CHOOSE : choose returns an item from a list as specified by an index value

تعمل على اختيار قيمة من المدخلات حسب الرقم المعطى

EX :

```
select choose (3 ,'Ahmed' ,'hasan','mohammed') as
result
```

Using NULLIF to Return NULL If Values Match

- NULLIF compares two expressions
  - Returns NULL if both arguments are equal
  - Returns the first argument if the two arguments are not equal

emp_id	goal	actual
1	100	110
2	90	90
3	100	90
4	100	80

```
SELECT emp_id, NULLIF(actual,goal) AS actual_if_different
FROM dbo.employee_goals;
```

```
emp_id      actual_if_different
----------- -------------------
1           110
2           NULL
3           90
4           80
```

هذه الدالة تقارن قيم عمودين اذا كانوا متشابهين في القيم تعيد null ، اذا كانو مختلفين تعيد القيمة الأولى

EX :

```
select LastName,NULLIF(math,arabic) from
[dbo].[mazn]
```

COALESCE Function : returns the not null VALUE , لكن اذا كانت القيم كلها null ستعيد null

```
select coalesce (Region,' ') from
Customers
```

تستخدم الدالة بقيمتين

لاحظ في المثال سوف تعيد القيمة غير النال non Null ، أما النال nul ستضع مكانها فراغ

```
select CustomerID,Country,Region,city,
Country + ', ' +COALESCE(region,' ')+ '
, '+city as location from Customers
```

```
--AGGREGATION FUNCTIONS :
 EX :
select sum(UnitPrice) as sumprice from Products
select count(UnitPrice) as countprice from
Products
```

```
select sum(UnitPrice) as sumprice ,
COUNT(UnitPrice) as cont, Max(UnitPrice) as
MAximum, MIN(UnitPrice)as minimum
,AVG(UnitPrice)as average from Products
```

```
using of Group by–
select * from [dbo].[Order Details]
   order by ProductID
```

لاحظ وجود عدد كبير لبيع الاصناف من كل رقم حسب رقم المنتج
جرب الان الأمر التالي

```
Select ProductID,Sum (Unitprice) From
[dbo].[Order Details]
   order by ProductId
```

لاحظ وجود مشكلة لتعدد الproductID في حين الدالة تعطي قيمة واحدة الحل هنا استخدام ال Group By

لو أردنا جمع مبيعات كل منتج على حدى نستخدم group by

```
 Select ProductID,Sum (Unitprice) From
[dbo].[Order Details]
   Group by productId
   order by ProductId
```

للتأكيد و مقارنة النتيجة .....

```
Select Sum (Unitprice) From [dbo].[Order Details]
      where productId=1;
```

عند استخدام ال aggregate function و معها عمود آخر ليس Aggregate لابد من وضعه في group by

```
Select ProductID,OrderID,Sum (Unitprice) From
[dbo].[Order Details]
Group by productId,OrderID
```

الأن حاول عرض مجاميع الأسعار التي تزيد عن

١٥٠

```
Select ProductID,Sum (Unitprice) as total
From [dbo].[Order Details]
   where Sum (Unitprice)>50
   Group by productId
   order by total
```

■ لاحظ WHERE توضع قبل Group by و لتنفيذ الجملة بشكل صحيح يتطلب استخدام Group by قبل الشرط لترتيب نتيجة الجمع حسب رقم المنتج

لحل المشكلة السابقة نستخدم HAVING :

```
Select ProductID,Sum (Unitprice) as total
   From [dbo].[Order Details]
   Group by productId
   Having Sum (Unitprice)>200
   order by total
```

نستخدم دائما جملة having مع group by حال وجود شرط

لاحظ ترتيب أوامر الجملة

Logical Order	Phase	Comments
5	SELECT	
1	FROM	
2	WHERE	
3	GROUP BY	Creates groups
4	HAVING	Operates on groups
6	ORDER BY	

كذلك يمكن وضع join مع الجملة

```
SELECT c.custid, COUNT(*) AS cnt
FROM Sales.Customers AS c
    JOIN Sales.Orders AS o ON c.custid = o.custid
GROUP BY c.custid
HAVING COUNT(*) > 1;
```

- Most aggregate functions ignore NULL أغلب هذه الدوال تتجاهل ال
- COUNT (<column>)ignores NULL
- COUNT (*) counts all rows حتى لو فيها نل

Now try the following :

1. select count (OrderID) from [Order Details]

2. select count (distinct OrderID) from [Order Details]

……----------

❖ SUB QUERY

THE IDEA IS TO USE A QUERY INSIDE ANOTHR QUERY

وضع أمر استعلام داخل أمر استعلام أخر

لعمل مقارنة بين السعر الحقيقي و معدل الأسعار

```
SELECT ProductName,UnitPrice AS REAL_PRICE ,
UNITPRICE-(SELECT  AVG (UnitPrice)FROM PRODUCTS)
 FROM Products
```

يتكون الSUB QUERY من INNER QOUERY وهو الاستعلام الداخلي و OUTER QUERY و هو الاستعلام الخارجي .

❖ استخرج اسم/أسماء الموظفين الذين قاموا ببيع طلبات بتاريخ ١٩٩٦/٠٧/٠٥ ؟

افحص جدول الطلبات

```
SELECT * FROM Orders
```

لمعاينة الطلبات التي بيعت بهذا التاريخ جرب الأمر التالي

```
SELECT * FROM Orders WHERE
ORDERDATE='1996-07-05'
```

من الطرق المضمونة لكتابة التاريخ مع SQL الصيغة التالية

```
'19960705'
```

```
-- TO FINDE THE EMPLOYEE NAME, WE NEED TO
USE SUB QUERY OR JOIN
```

لمعرفة اسم الوظف الذي قام بعملية البيع بهذا التاريخ نستخدم الامر التالي

```
SELECT FirstName FROM Employees
WHERE EmployeeID=(SELECT EmployeeID FROM Orders
                  WHERE ORDERDATE='19960705')
```

تذكر هنا اشارة = تعطي في النتيجة موظف واحد لو أردنا أسماء عدة موظفين قاموا بعمليات بيع في يوم واحد نستخدم IN بدل = ، كما موضح في التالي

```
SELECT FirstName FROM Employees
WHERE EmployeeID IN(SELECT EmployeeID FROM Orders
                    WHERE ORDERDATE='19960708')
```

لحل السؤال السابق باستخدام ال JOIN كالتالي ...، قمنا باضافة بيانات أخرى لاستخراجها مثل CUSTOMERID

```
-- TO FIND IT BY JOIN
SELECT Employees.FirstName , Orders.CustomerID
 FROM Employees  JOIN Orders ON
EmployeeS.EmployeeID=Orders.EmployeeID
```

```
WHERE Orders.OrderDate='19960708'
```

ميزة في استخدام ال JOIN يمكن اضافة بيانات لأكثر من عمود في كلا الجدولين

- استخرج أسماء ال CUSTOMERS الذين قاموا بعملية شراء طلبات فعلية (أي موجودين في جدول ال ORDERS)

يمكن استخدام الدالة EXISTS

```
SELECT CUSTOMERID,COMPANYNAME
FROM Customers
WHERE EXISTS ( SELECT * FROM ORDERS WHERE
          Customers.CUSTOMERID=ORDERS.CustomerID);
```

الامر السابق سوف يعرض بيانات ال CUSTOMERS الموجودة في جدول ال CUSTOMERS وقاموا بعمليات شراء تم تسجيلها في جدول ال ORDERS

عكسه يمكن استخدام الدالة NOT EXISTS

```
SELECT CUSTOMERID,COMPANYNAME
FROM Customers
WHERE NOT EXISTS ( SELECT * FROM ORDERS WHERE
          Customers.CUSTOMERID=ORDERS.CustomerID);
```

```
--SQL SPACE() Function
```

دالة تستخدم لترك مسافة -فراغ بين الكلمات

```
SELECT 'HELLO' + SPACE(5) + 'WORLD!'
select * from Employees
SELECT city + SPACE(5) + 'WORLD!' from Employees
SELECT FirstName + SPACE(8) + LastName  as
FullNAme from Employees
```

VIEWs

View is a logical table based on a table or another view. It does not contain any physically.

Constraints defined on base table are enforced by view .

نستخدم ال view في حال أردنا اعطاء مستخدم قواعد البيانات مساحة ضيقة (لا يستطيع الاطلاع على جميع بيانات الجدول) اما زيادة في الأمانsecurity أو لأن الجدول معقد فنعطيه مساحة لاستخدام البيانات التي تلزمه فقط.

انشاءHow to create View

EX :

```
create view highProduct
as
select * from Products where UnitPrice>150
```

المثال السابق أنشئ view اسمها highProduct تحتوي على الأصناف التي يزيد سعرها عن ١٥٠ فقط

How to call View

EX :

```
Select * from highProduct
```

تماما مثل الجدول

To modify the body of view we use **"Alter"**

Ex :

```
Alter view highProduct
as
select * from Products where UnitPrice>200
```

call the view again

```
Select * from highProduct
```

ملاحظة /يمكن التعديل على بيانات ال view باستخدام insert ,delete, update بشرط أن تكون تشمل على بيانات لجدول حقيقي واحد

Note1: you can create a view by wizard
Just click new view , add your tables then chose the columns then save it

يمكن انشاء view باستخدام المعالج

Note2 : you can not use 'order by' when creating the view .

Store UNIQUE values in a column

عندما تحاول إنشاء حساب جديد في أي موقع إلكتروني، تلاحظ في بعض الأحيان أنه يطلب منك إدخال معلومات لم يدخلها أحد غيرك من قبل.
فمثلاً، عندما يطلب منك إدخال إسم المستخدم (**Username**) قد تجده يخبرك بأن الإسم الذي أدخلته غير متاح.
أيضاً، إذا كنت تملك حساب في موقع ما و حاولت إنشاء حساب آخر في نفس الموقع و باستخدام نفس البريد الإلكتروني ستجد الموقع لا يسمح لك بذلك.

فوائدة القيم الموحدة في التطبيقات و المواقع

في الأمثلة التي ذكرناها سابقاً، الفائدة من جعل إسم المستخدم الموحد قد يكون لها فوائد عديدة و نذكر منها:

- يمكنك تسجيل الدخول به.
- يمكنك البحث عن أي مستخدم من خلال إسم المستخدم الخاص به.
- يمكنك الإبلاغ عن أي مستخدم من خلال إسم المستخدم الخاص به.
- في مواقع التواصل الإجتماعي مثل تويتر و فيسبوك، يمكنك أن تشير لأي حساب (أي تفعل له **Mention**) من خلال وضع الرمز @ و من ثم ذكر إسم المستخدم.

الكلمة UNIQUE

في حال أردت جعل العامود لا يقبل أن يتم تخزين نفس القيمة فيه أكثر من مرة يمكنك إضافة الخاصية UNIQUE إلى نوع العامود.

الآن، عليك معرفة أن طريقة استخدام هذه الكلمة تختلف من قاعدة بيانات لأخرى و لكنها تستخدم لنفس الغرض.

Ex :

- CREATE TABLE table_name (column_name datatype, UNIQUE (column_name));

```sql
CREATE TABLE users (
id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
first_name VARCHAR(50),
last_name VARCHAR(50),
username VARCHAR(30),
UNIQUE(username));
-- هنا قمنا بتحديد username لا يقبل تخزين قيم مكررة أن العامود
```

Columns Merge

دمج الأعمدة

```sql
SELECT 5 + 2 + 7;
SELECT 5 + 2 + 7 AS 'Total';
SELECT 'Ahmed ' + 'Hassan';
SELECT CONCAT('Ahmed', 'Hassan') AS 'Full Name';

SELECT CONCAT('Ahmed',  '  ' ,'Hassan') AS 'Full
Name';



SELECT   CONCAT(firstName, ' ', lastname) AS
'Employee' FROM[dbo].[Employees]
```

Add new column Salary , thn apply :

```
SELECT     CONCAT(firstName, ' ', lastname) AS
'Employee',CONCAT(salary, '$') AS 'Salary'
            FROM[dbo].[Employees]
```

STORED PROCEDURE

- Small function saved at the database server دالة مصغرة محفوظة في سيرفر قواعد البيانات
- Most the application(java ,c#,vb.net,…) can call it.،،، يمكن استدعائها بسهولة من خلال التطبيقات مثل جافا
- Fast Performance and wide security أداء سريع و أمن معلومات كبير

سريعة جدا مقارنة بال views لأنها بعد استدعائها أول مرة توضع في الكاش لذلك يطلق عليها Precached function

How To Create a Procedure

EX :

```
create proc GetProdutInfo
as
select * from Products
```

To call the Procedure we use "exec"

EX :

```
exec GetProdutInfo
```

to modify it right click , chose modify then change the code

Stored Procedure with arguments "Parameters":

لاعطاء Parameter استخدم @ أتبعها باسم و نوع ال DataType .

مثال : انشاء دالة بحث ديناميكية من خلال حرف ما

Dynamic search proc with parameters

EX :

```
create Proc SearchByName
 @ProductName nvarchar(50)
 as
 select ProductID,ProductName,UnitPrice
 from Products
 where ProductName like @ProductName + '%'
```

أنشأنا ال Proc الان لاستدعاءه كالتالي :

```
 exec SearchByName 'c'
 exec SearchByName 'A'
```

لاحظ لابد من وضع بارامتر عند الاستدعاء لان العملية ديناميكية

س : اجعل الProc السابق يبحث فقط عن منتجات بقيمة افتراضية و لتكن ch ؟؟

الان سوف نقوم بتعديل الProc ليقوم بعملية بحث باستخدام قيمة افتراضية ثابتة ، استخدم ALTER لتعديل الدالة لتصبح كالتالي :

```sql
-- search by Defoult value
Alter Proc SearchByName
 @ProductName nvarchar(50)='ch'
 as
 select ProductID,ProductName,UnitPrice
 from Products
 where ProductName like @ProductName + '%'

 -- call it now
 exec SearchByName
```

أصبحت الدالة تبحث فقط عن المنتجات التي تبدأ ب ch .

ملاحظة : لا يمكن استخدام ال Stored Proc في Query لكن يمكن استخدام (Insert,Update ,Delete) .

سنقوم بانشاء Proc يستخدم لادخال بيانات على جدول ال products و ليكن في ثلاث أعمدة

EX :

```sql
create Proc NewProduct
 @ProductName nvarchar(50),
 @price decimal,
 @stock smallint
 as
 Insert
Products(ProductName,UnitPrice,UnitsInStock)
 values (@ProductName,@price,@stock)
```

الأن جرب ال Proc : Test your Proc

```
exec NewProduct 'Cola',20,30
exec NewProduct 'Arial',50,100

 select * from Products
```

- User Define Function

  Very similar to store procedure شبيه جدا ب

  stored procedure دائما يفضل استخدام ال

  Just in one case is better to use 'Define function ' than store procedure

  Witch is **Scalar Function**.

  "scalar function" فقط في حالة واحدة يفضل استخدامها و هي

  We have 3 types of 'Define Function' ثلاث أنواع منها

  1. Scalar Function
  2. Table valued Function (inline/multiline)
  3. Aggregate Function

*Scalar Function

How to create Function

```
User Define Function
-- HOw to create scalar function

Create Function dbo.MySum
(@n1 int, @n2 int) returns int
 as
  begin
  return @n1+@n2 ;
  end

  -- how to call function

  select dbo.Mysum(5,20)


how to use with procedure
ex :
select ProductName,ProductID,UnitPrice
,dbo.Mysum(UnitPrice,UnitsInStock)as
total_sum

from Products
```