

فى هذه المرحلة من الشرح سوف نبدأ فى التعامل
بمرونه أكثر مع قاعده البيانات عن طريق إضافات لما تم
شرحه فى الجزء الأول من الدرس عن طريق جمل **SQL**

عبارة : **Order By**
تستخدم لترتيب الناتج من الاستعلام حسب حقل محدد
مثال على الترتيب :

Orders:

Company	OrderNumber
Sega	3412
ABC Shop	5678
W3Schools	2312
W3Schools	6798

سنقوم بتنفيذ جملة الاستعلام التالية :

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY Company
```

النتيجة: **Result:**

Company	OrderNumber
ABC Shop	5678
Sega	3412
W3Schools	6798
W3Schools	2312

نلاحظ ان البيانات جاءت فى ترتيب حسب حقل **Company**

إذا اردنا الترتيب بأكثر من حقل تكون كالتالى :

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY Company, OrderNumber
```

النتيجة: **Result:**

Company	OrderNumber
ABC Shop	5678
Sega	3412
W3Schools	2312
W3Schools	6798

ماذا لو اردنا عكس الترتيب كيف يتم ذلك

مثال :

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY Company DESC
```

Result: النتيجة

Company	OrderNumber
W3Schools	6798
W3Schools	2312
Sega	3412
ABC Shop	5678

فى حاله اننا نريد أستخدام الطريقتين كيف يتم ذلك

مثال :

يمكن عمل ذلك بطريقتين

الطريقه الأولى :

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY Company DESC, OrderNumber ASC
```

اما الطريقه الثانيه هى كتابه الحقول المطلوب ترتيبها
تصاعديا اولا ثم الحقول المطلوب ترتيبها تنازليا

مثال :

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY OrderNumber , Company DESC
```

وفى اى من الطريقتين تكون النتيجة واحده

Result: النتيجة

Company	OrderNumber
W3Schools	2312
W3Schools	6798
Sega	3412
ABC Shop	5678

الآن ننتقل لمعاملات جديده

المعاملين And – Or

يستخدم هذين المعاملين في ربط شرطين او أكثر

- المعامل **AND** يقوم بعرض النتائج في حالة تحقق جميع الشروط
- المعامل **OR** يقوم بعرض النتائج في حالة تحقق اى شرط من الشروط

مثال : على الجدول التالى

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Svendson	Tove	Borgvn 23	Sandnes
Svendson	Stephen	Kaivn 18	Sandnes

أستخدام المعامل **AND** :

```
SELECT * FROM Persons  
WHERE FirstName='Tove'  
AND LastName='Svendson'
```

النتيجه Result:

LastName	FirstName	Address	City
Svendson	Tove	Borgvn 23	Sandnes

أستخدام المعامل **OR** :

```
SELECT * FROM Persons  
WHERE firstname='Tove'  
OR lastname='Svendson'
```

النتيجه Result:

LastName	FirstName	Address	City
Svendson	Tove	Borgvn 23	Sandnes
Svendson	Stephen	Kaivn 18	Sandnes

أستخدام المعاملين معا :

```
SELECT * FROM Persons WHERE  
(FirstName='Tove' OR FirstName='Stephen')  
AND LastName='Svendson'
```

النتيجه Result:

LastName	FirstName	Address	City
Svendson	Tove	Borgvn 23	Sandnes
Svendson	Stephen	Kaivn 18	Sandnes

المعامل IN

له عدة استخدامات منها انه يمكنك تحديد القيمة المراد عرضها إذا كنت متأكد من وجودها في أحد الحقول

بناء الجملة :

```
SELECT column_name FROM table_name  
WHERE column_name IN (value1,value2,..)
```

مثال : على الجدول التالي

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Nordmann	Anna	Neset 18	Sandnes
Pettersen	Kari	Storgt 20	Stavanger
Svendson	Tove	Borgvn 23	Sandnes

```
SELECT * FROM Persons  
WHERE LastName IN ('Hansen','Pettersen')
```

النتيجة: Result:

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Pettersen	Kari	Storgt 20	Stavanger

ويمكن أيضا استخدامه في ربط أكثر من استعمال معا

```
SELECT * FROM table1_name  
WHERE FirstName in (SELECT * FROM table2_name)
```

في هذا المثال قمنا بعرض جميع القيم من الجدول الأول
Table1_name بشرط ان يكون الحقل **FirstName** موجود
في الجدول الثاني **table2_name**

المعامل BETWEEN ... AND

يستخدم لعرض مجموعه بيانات بين قيمتين نصوص أو أرقام أو تاريخ

بناء الجملة :

```
SELECT column_name FROM table_name  
WHERE column_name  
BETWEEN value1 AND value2
```

مثال : على الجدول التالي

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Nordmann	Anna	Neset 18	Sandnes
Pettersen	Kari	Storgt 20	Stavanger
Svendson	Tove	Borgvn 23	Sandnes

عرض البيانات الموجوده بين القيمتين

```
SELECT * FROM Persons WHERE LastName  
BETWEEN 'Hansen' AND 'Pettersen'
```

النتيجه Result:

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Nordmann	Anna	Neset 18	Sandnes

عرض البيانات الغير موجوده بين القيمتين

```
SELECT * FROM Persons WHERE LastName  
NOT BETWEEN 'Hansen' AND 'Pettersen'
```

النتيجه Result:

LastName	FirstName	Address	City
Pettersen	Kari	Storgt 20	Stavanger
Svendson	Tove	Borgvn 23	Sandnes

تستخدم نفس الطريقه مع الأرقام والتواريخ مع مراعاة
ان التواريخ لا تكون بين علامتى (') بل يستخدم (#) .

الأسم المستعار Alias

يستخدم في تغيير اسم الجدول او الحقل عند عرضه
نستخدمه عن طريق كلمه **AS**

البناء في حاله الجدول :

```
SELECT column AS column_alias FROM table
```

البناء في حاله الحقل :

```
SELECT column AS column_alias FROM table
```

مثال : على الجدول التالي

This table (Persons):

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Svendson	Tove	Borgvn 23	Sandnes
Pettersen	Kari	Storgt 20	Stavanger

```
SELECT LastName AS Family, FirstName AS Name  
FROM Persons
```

Result: النتيجة

نلاحظ أسماء الحقول (الأعمده)

Family	Name
Hansen	Ola
Svendson	Tove
Pettersen	Kari

```
SELECT LastName, FirstName  
FROM Persons AS Employees
```

Result: النتيجة

Table Employees: نلاحظ ان اسم الجدول

LastName	FirstName
Hansen	Ola
Svendson	Tove
Pettersen	Kari

الربط JOIN

موضوع من اهم مواضيع **SQL** هو الربط بين الجداول الذى يسهل كثير فى عرض البيانات من الجداول التى تحتوى على علاقه مثل الموظف و القسم .

فى بعض الأحيان نحتاج إلى عرض بيانات من جدولين أو أكثر لجعل النتيجة كامله يجب انشاء علاقه .

الربط بين الجداول يتم عن طريق المفتاح الأساسى (**Primary Key**) الحقل الذى يكون مفتاح اساسى لا يمكن تكرار البيانات بداخله .

فى الجدول التالى حقل (**Employee_ID**) هو حقل مفتاح اساسى الخاص بجدول الموظفين .

اما فى جدول الثانى الحقل (**Order_ID**) هو حقل المفتاح الاساسى للجدول وفيه ايضا مفتاح خارجى وهو حقل (**Employee_ID**).

الجدول الأول: **Employees**

Employee_ID	Name
01	Hansen, Ola
02	Svendson, Tove
03	Svendson, Stephen
04	Pettersen, Kari

الجدول الثانى: **Orders**

Prod_ID	Product	Employee_ID
234	Printer	01
657	Table	03
865	Chair	03

مثال : نريد ان نعرف من طلب منتج وما هو المنتج

```
SELECT Employees.Name, Orders.Product
FROM Employees, Orders
WHERE Employees.Employee_ID=Orders.Employee_ID
```

Result: النتيجة

Name	Product
Hansen, Ola	Printer
Svendson, Stephen	Table
Svendson, Stephen	Chair

مثال : نريد ان نعرف من طلب المنتج (Printer)

```
SELECT Employees.Name
FROM Employees, Orders
WHERE Employees.Employee_ID=Orders.Employee_ID
AND Orders.Product='Printer'
```

Result: النتيجة

Name
Hansen, Ola

مثال على استخدام INNER JOIN :
بناء الجملة :

```
SELECT field1, field2, field3
FROM first_table
INNER JOIN second_table
ON first_table.keyfield = second_table.foreign_keyfield
```

تنفيذ الجملة على المثال :

```
SELECT Employees.Name, Orders.Product
FROM Employees
INNER JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

Result: النتيجة

Name	Product
Hansen, Ola	Printer
Svendson, Stephen	Table
Svendson, Stephen	Chair

تقوم الجملة INNER JOIN بعرض جميع البيانات
المشتركة بين الجدولين

مثال على استخدام LEFT JOIN :

بناء الجملة :

```
SELECT field1, field2, field3
FROM first_table
LEFT JOIN second_table
ON first_table.keyfield = second_table.foreign_keyfield
```

تنفيذ الجملة على المثال :

```
SELECT Employees.Name, Orders.Product
FROM Employees
LEFT JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

النتيجة: Result:

Name	Product
Hansen, Ola	Printer
Svendson, Tove	
Svendson, Stephen	Table
Svendson, Stephen	Chair
Pettersen, Kari	

تقوم الجملة LEFT JOIN بعرض جميع البيانات من الجدول الأول Employee حتى لو لم توجد في الجدول الثاني

مثال على استخدام RIGHT JOIN :

بناء الجملة :

```
SELECT field1, field2, field3
FROM first_table
RIGHT JOIN second_table
ON first_table.keyfield = second_table.foreign_keyfield
```

تنفيذ الجملة على المثال :

```
SELECT Employees.Name, Orders.Product
FROM Employees
RIGHT JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

النتيجة: Result:

Name	Product
Hansen, Ola	Printer
Svendson, Stephen	Table
Svendson, Stephen	Chair

تقوم الجملة RIGHT JOIN بعرض جميع البيانات من الجدول الثاني Orders حتى لو توجد في الجدول الأول

مثال على استخدام جملة **INNER JOIN** :

نريد عرض أسماء الموظفين الذين طلبوا المنتج (**Printer**)

```
SELECT Employees.Name  
FROM Employees  
INNER JOIN Orders  
ON Employees.Employee_ID=Orders.Employee_ID  
WHERE Orders.Product = 'Printer'
```

Result: النتيجة

Name
Hansen, Ola

وبهذا نكون انتهينا من شرح جملة **JOIN** .

الدمج UNION and UNION ALL

تستخدم لدمج حقلين من جدولين مختلفين ولكن يجب ان يكون نوع البيانات في الحقلين واحد اي يكون نصوص او ارقام.....إلخ

البناء للجمله يكون كالتالى :

```
SQL Statement 1
UNION
SQL Statement 2
```

مثال :

Employees_Norway:

E_ID	E_Name
01	Hansen, Ola
02	Svendson, Tove
03	Svendson, Stephen
04	Pettersen, Kari

Employees_USA:

E_ID	E_Name
01	Turner, Sally
02	Kent, Clark
03	Svendson, Stephen
04	Scott, Stephen

نريد دمج الحقل E_NAME فى كل من الجدولين :

```
SELECT E_Name FROM Employees_Norway
UNION
SELECT E_Name FROM Employees_USA
```

النتيجه Result:

E_Name
Hansen, Ola
Svendson, Tove
Svendson, Stephen
Pettersen, Kari
Turner, Sally
Kent, Clark
Scott, Stephen

نلاحظ ان النتيجه ظهرت بدون تكرار للبيانات .
أستخدم UNION ALL :

مثل استخدام UNION الفرق انه يقوم بعرض جميع البيانات حتى لو يوجد تكرار .

البناء للجمله يكون كالتالى :

```
SQL Statement 1
UNION ALL
SQL Statement 2
```

مثال : نريد دمج الحقل E_NAME فى الجدولين السابقين

```
SELECT E_Name FROM Employees_Norway
UNION ALL
SELECT E_Name FROM Employees_USA
```

Result: النتيجة

E_Name
Hansen, Ola
Svendson, Tove
Svendson, Stephen
Pettersen, Kari
Turner, Sally
Kent, Clark
Svendson, Stephen
Scott, Stephen

نلاحظ ظهور جميع البيانات مع تكرار احد الأسماء المشترك فى الجدولين .

(الآن ننتقل إلى درس مهم جدا فى لغه SQL الا وهو الدوال)

الدوال SQL Functions

لغة SQL بها الكثير من الدوال العددية والحسابية

البناء الأساسي لأي دالة :

```
SELECT function(column) FROM table
```

الدوال :

الوصف	الدالة
إيجاد الوسط الحسابي للحقل المحدد	AVG(column)
معرفة عدد الصفوف (السجلات) في الحقل بدون السجلات الفارغة	COUNT(column)
معرفة عدد الصفوف في الجدول	COUNT(*)
معرفة قيمة أول سجل في الحقل	First(column)
معرفة قيمة آخر سجل في الحقل	last(column)
معرفة أكبر قيمة سجل في الحقل	Max(column)
معرفة أصغر قيمة سجل في الحقل	Min(column)
معرفة إجمالي القيم في الحقل	SUM(column)
عدد السجلات في الحقل بدون تكرار تعمل فقط على SQL SERVER	COUNT(DISTINCT column)

مجموعه من الأمثلة :

SELECT AVG(Column) From Table
SELECT COUNT(column) From Table
SELECT COUNT(*) From Table
SELECT First(column) From Table
SELECT last(column) From Table
SELECT Max(column) From Table
SELECT Min(column) From Table
SELECT SUM(column) From Table
SELECT COUNT(DISTINCT column) From Table

التجميع والفرز SQL GROUP BY and HAVING

الدوال العددية مثل الدالة SUM كثيرا ما تحتاج إلى التجميع GROUP BY

بناء الجملة :

```
SELECT column, SUM(column) FROM table GROUP BY column
```

مثال للتوضيح : على الجدول التالي

Company	Amount
W3Schools	5500
IBM	4500
W3Schools	7100

نفذ الجملة التالية:

```
SELECT Company, SUM(Amount) FROM Sales
```

النتيجة: Returns this result:

Company	SUM(Amount)
W3Schools	17100
IBM	17100
W3Schools	17100

نلاحظ ان جميع السجلات اخذت المجموع كله ولم نعرف مجموع كل سجل .

الآن نجرب جملة الاستعلام بعد إضافة GROUP BY :

```
SELECT Company, SUM(Amount) FROM Sales  
GROUP BY Company
```

النتيجة: Returns this result:

Company	SUM(Amount)
W3Schools	12600
IBM	4500

نلاحظ انه تم جمع كل سجل وحده وأصبحت النتيجة اوضح

الداله **HAVING** تستخدم لفرز البيانات حسب شرط معين

بناء الجملة :

```
SELECT column,SUM(column) FROM table  
GROUP BY column  
HAVING SUM(column) condition value
```

تطبيق على نفس المثال السابق :

Company	Amount
W3Schools	5500
IBM	4500
W3Schools	7100

نفذ جملة الاستعلام التاليه :

```
SELECT Company,SUM(Amount) FROM Sales  
GROUP BY Company  
HAVING SUM(Amount)>10000
```

النتيجه : Returns this result :

Company	SUM(Amount)
W3Schools	12600

نلاحظ انه تم تحقق الشرط وعرض البيانات اكبر من ١٠٠٠٠ فقط.

تم بحمد الله تعالى الجزء الثانى من الدرس والخاص

SQL FUNCTION

وسوف نكمل معا باذن الله الجزء الثالث الخاص

Data Definition Language (DDL)

تحياتى للجميع

GENIUS-IT@HotMail.Com