اولا وقبل البدء فى الشرح اود ان اذكر مصدر الشرح للأمانه هذه الدروس منقوله عن موقع
WWW.W3Schools.COM
ولقد قمت انا فقط بترجمتها وتوضيح وتعديل بعض النقاط الصغيره بها

سيكون هذا الشرح عباره عن مرجع شامل لجمل SQL

اولا مقدمه فى SQL :

هى لغه قياسيه من لغات الحاسب لدخول ومعالجه قواعد البيانات

ما هى SQL : Structured Query Language

لغه بناء الأستعلامات الهيكليه (يعنى هى الترجمه حرفيه شويه)

١- هى لغه قياسيه من لغات الحاسب الخاصه بمعهد ANSI
٢- تمكنك من الدخول لقواعد البيانات
٣- تمكنك من إستخراج البيانات من القاعده
٤- تمكنك من إضافه بيانات إلى قاعده البيانات
٥- تمكنك من من حذف بيانات من القاعده
٦- تمكنك من تعديل البيانات المسجله
٧- أخيرا هى لغه سهله التعلم والفهم

لغه SQL هى لغه قياسيه :
هى لغه من اللغات القياسيه الخاصه بمعهد
ANSI (American National Standards Institute)
تمكنك من دخول ومعالجه نظم قواعد البيانات Database System
جمل SQL تعمل مع برامج قواعد البيانات مثل :

Ms-Access, Ms-SQL Server, DB2, Oracle, etc.

تنقسم لغه SQL إلى قسمين :

1- SQL Data Manipulation Language (DML)

هو القسم المسئول عن : معالجه البيانات

١- Select : إستخراج البيانات من قاعده البيانات
٢- INSERT INTO : إضافه بيانات جديده
٣- Update : التعديل على البيانات المسجله
٤- Delete : حذف البيانات من القاعده

2- Data Definition Language (DDL)

هو القسم المسئول عن : تعريف البيانات

١- Create Database : لأنشاء قاعده بيانات جديده
٢- Create Table : لأنشاء جدول داخل قاعده بيانات
٣- ALTER TABLE : للتعديل فى الجدول
٤- DROP TABLE : لحذف الجدول من قاعده البيانات
٥- CREATE INDEX : لأنشاء مفتاح للبحث
٦- DROP INDEX : لحذف مفتاح البحث

أولا جمله Select

تستخدم فى إستخراج بيانات من داخل الجدول حسب المطلوب

Syntax :تكتب الجمله كالتالى

```
SELECT column_name(s)
FROM table_name
```

ملاحظه هامه : جمل SQL غير حساسه لحاله الحروف
لا يوجد فرق SELECT = select

مثال على الجدول التالى :

The database table "Persons":

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

نريد أختيار الأسم الأول و الأسم الأخير (FirstName , LastName)
تكتب الجمله كالتالى :

```
SELECT LastName,FirstName FROM Persons
```

ويكون الناتج كالتالى :

Result: **النتيجه**

LastName	FirstName
Hansen	Ola
Svendson	Tove
Pettersen	Kari

لأختيار جميع البيانات تكتب كالتالى :

```
SELECT * FROM Persons
```

وتكون النتيجه كالتالى :

Result: **النتيجه**

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

ملاحظه هامه : بعض نظم إداره قواعد البيانات تتطلب وجود (;) فى نهايه جمله SQL وهذا لان بعض النظم تدعم تنفيذ أكثر من جمله فى نفس الوقت.
من النظم التى لا تحتاج إلى (;) Ms-Sql server – Ms-Access .
ام من النظم التى تحتاج إلى (;) Oracle .

الأن نندرج لجمله أخرى او إضافه لجمله SELECT
وهى جمله SELECT DISTINCT

إضافه كلمه DISTINCT للجمله تمكنك من العرض القيم بدون تكرار (يعنى لو الجدول فيه بيان متكرر اكتر من مره بتعرضه مره واحده بس) .

مثال : على أستخدام DISTINCT وعدم أستخدامها

```
SELECT Company FROM Orders
```

"Orders" table

Company	OrderNumber
Sega	3412
W3Schools	2312
Trio	4678
W3Schools	6798

Result: **النتيجه**

Company
Sega
W3Schools
Trio
W3Schools

هنا واضح الفرق عند أستخدام DISTINCT

```
SELECT DISTINCT Company FROM Orders
```

Result: **النتيجه**

Company
Sega
W3Schools
Trio

الان ننتقل لجزء اخر الا وهو استخدام الشرط
استخدام عباره WHERE

تأتى بعد جمله Select وتحتوى على الشرط المطلوب
ويكون بنائها بهذا الشكل :

Syntax

```
SELECT column FROM table
WHERE column operator value
```

مع جمله Where يمكننا أستخدام المعاملات الأتيه :

المعامل	الوصف
=	يساوى
<>	لا يساوى
>	أكبر من
<	أصغر من
>=	أكبر من او يساوى
<=	اصغر من او يساوى
Between	يكون الشرط بين قيمتين
LIKE	للبحث عن كلمات متشابهه

ملاحظه هامه : فى بعض الأصدارات من SQL المعامل<> يكتب هكذا =!

مثال على أستخدام عباره Where

```
SELECT * FROM Persons
WHERE City='Sandnes'
```

"Persons" table

LastName	FirstName	Address	City	Year
Hansen	Ola	Timoteivn 10	Sandnes	1951
Svendson	Tove	Borgvn 23	Sandnes	1978
Svendson	Stale	Kaivn 18	Sandnes	1980
Pettersen	Kari	Storgt 20	Stavanger	1960

Result النتيجه

LastName	FirstName	Address	City	Year
Hansen	Ola	Timoteivn 10	Sandnes	1951
Svendson	Tove	Borgvn 23	Sandnes	1978
Svendson	Stale	Kaivn 18	Sandnes	1980

ملاحظه هامه : على أستخدام علامه التنصيص مع الجمله (')
يجب أستخدام علامه تنصيص مفرده مع القيم النصيه مثل الأسماء اما بالنسبه للبيانات الرقميه تكتبه من دون علامات .
مثال على ذلك :

For text values: للبيانات النصيه

```
This is correct: طريقه صحيحه
SELECT * FROM Persons WHERE FirstName='Tove'
This is wrong: طريقه خاطئه
SELECT * FROM Persons WHERE FirstName=Tove
```

For numeric values: للبيانات الرقميه

```
This is correct: طريقه صحيحه
SELECT * FROM Persons WHERE Year>1965
This is wrong: طريقه خاطئه
SELECT * FROM Persons WHERE Year>'1965'
```

استخدام المعامل Like فى الشرط

تستخدم فى تحديد البحث بكلمه معينه مثال البحث عن كل الأسماء الموجود بها حرف او كلمه معينه .

بنائها بهذا الشكل Syntax

```
SELECT column FROM table
WHERE column LIKE pattern
```

تستخدم علامه (%) لتحديد عدد الحروف قبل و بعد الكلمه او الحرف الذى نبحث به مثال :

فى هذه الحاله سوف تظهر اسماء الأشخاص التى تبدا بحرف A ثم يأتى بعدها اى عدد من الحروف

```
SELECT * FROM Persons
WHERE FirstName LIKE 'a%'
```

فى هذه الحاله سوف تظهر اسماء الأشخاص التى تنتهى بحرف A ويكون قبلها اى عدد من الحروف

```
SELECT * FROM Persons
WHERE FirstName LIKE '%a'
```

فى هذه الحاله سوف تظهر اسماء الأشخاص التى تحتوى على حرف A فى أى موضوع فى الأسم

```
SELECT * FROM Persons
WHERE FirstName LIKE '%la%'
```

بقى لنا الجزء الخاص بـ Between وسوف نندرج إليه فيما بعد.

الان ننتقل لجزء جديد الا وهو التعامل مع البيانات
يوجد لدينا ثلاثه تعاملات مع البيانات
١- إضافه : INSERT INTO ٢-تعديل : UPDATE
٣- حذف : DELETE

اولا إصافه صف للجدول : INSERT INTO

يكون بنائها كالتالى :

Syntax

```
INSERT INTO table_name
VALUES (value1, value2,....)
```

ويمكنك ايضا تحديد الحقول المطلوب إضافتها فقط وتكون كالتالى :

```
INSERT INTO table_name (column1, column2,...)
VALUES (value1, value2,....)
```

مثال إضافه صف جديد للجدول التالى :

This "Persons" table:

LastName	FirstName	Address	City
Pettersen	Kari	Storgt 20	Stavanger

```
INSERT INTO Persons
VALUES ('Hetland', 'Camilla', 'Hagabakka 24', 'Sandnes')
```

النتيجه تكون كالتالى :

LastName	FirstName	Address	City
Pettersen	Kari	Storgt 20	Stavanger
Hetland	Camilla	Hagabakka 24	Sandnes

لإضافه بيانات فى أعمده (حقول) محدده فقط تكتب كالتالى :

```
INSERT INTO Persons (LastName, Address)
VALUES ('Rasmussen', 'Storgt 67')
```

تكون النتيجه كالتالى :

LastName	FirstName	Address	City
Pettersen	Kari	Storgt 20	Stavanger
Hetland	Camilla	Hagabakka 24	Sandnes
Rasmussen		Storgt 67	

ثانيا التعديل فى الجدول : UPDATE

يكون بناء الجمله كالتالى :

Syntax

```
UPDATE table_name
SET column_name = new_value
WHERE column_name = some_value
```

مثال على التعديل على الجدول الحالى :

Person:

LastName	FirstName	Address	City
Nilsen	Fred	Kirkegt 56	Stavanger
Rasmussen		Storgt 67	

لإضافه الأسم الأول فى الصف الثانى :

```
UPDATE Person SET FirstName = 'Nina'
WHERE LastName = 'Rasmussen'
```

Result: النتيجه

LastName	FirstName	Address	City
Nilsen	Fred	Kirkegt 56	Stavanger
Rasmussen	Nina	Storgt 67	

لإضافه أسم المدينه وتغير العنوان :

تعديل أكثر من عمود فى نفس الوقت

```
UPDATE Person
SET Address = 'Stien 12', City = 'Stavanger'
WHERE LastName = 'Rasmussen'
```

Result: النتيجه

LastName	FirstName	Address	City
Nilsen	Fred	Kirkegt 56	Stavanger
Rasmussen	Nina	Stien 12	Stavanger

ثالثا الحذف من الجدول : DELETE

تستخدم فى حذف الصفوف من الجدول وبنائها كالتالى :

Syntax

```
DELETE FROM table_name
WHERE column_name = some_value
```

مثال على حذف صف من الجدول :

Person:

LastName	FirstName	Address	City
Nilsen	Fred	Kirkegt 56	Stavanger
Rasmussen	Nina	Stien 12	Stavanger

لحذف الصف

```
DELETE FROM Person WHERE LastName = 'Rasmussen'
```

Result :النتيجه

LastName	FirstName	Address	City
Nilsen	Fred	Kirkegt 56	Stavanger

لحذف جميع البيانات (الصفوف) من الجدول

تستخدم الجمله التاليه :

```
DELETE FROM table_name
or
DELETE * FROM table_name
```

تم بحمد الله تعالى الجزء الأول من الدرس والخاص

بـ (DML) *SQL Data Manipulation Language*

فى هذه المرحله 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

عباره : Order By

تستخدم لترتيب الناتج من الأستعلام حسب حقل محدد
مثال على الترتيب :

Orders:

Company	OrderNumber
Sega	3412
ABC Shop	5678
W3Schools	2312
W3Schools	6798

سنقوم بتنفيذ جمله الأستعلام التاليه :

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

Result: النتيجه

Company	OrderNumber
ABC Shop	5678
Sega	3412
W3Schools	6798
W3Schools	2312

نلاحظ ان البيانات ظهرت فى ترتيب حسب حقلCompany

إذا اردنا الترتيب بأكثر من حقل تكون كالتالى :

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

Result: النتيجه

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: النتيجه

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 النتيجه واحده

Result: النتيجه

Company	OrderNumber
W3Schools	2312
W3Schools	6798
Sega	3412
ABC Shop	5678

الأن ننتقل لمعاملات جديده

المعاملين And – Or

يستخدم هذين المعاملين فى ربط شرطين او أكثر

- المعامل AND يقوم بعرض النتائج فى حاله تحقق جميع الشروط
- المعامل OR يقوم بعرض النتائج فى حاله 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'
```

Result: النتيجه

LastName	FirstName	Address	City
Svendson	Tove	Borgvn 23	Sandnes

أستخدام المعامل OR :

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

Result: النتيجه

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'
```

Result: النتيجه

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

المعامل IN

له عده أستخدامات منها انه يمكنك تحديد القيمه المراد عرضها إذا كنت متأكد من وجودها فى أحد الحقول

بناء الجمله :

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

مثال : على الجدول التالى

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')
```

Result: النتيجه

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

ويمكن أيضا أستخدامه فى ربط أكثر من أستعلام معا

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

فى هذا المثال قمنا بعرض جميع القيم من الجدول الأول Table1_name بشرط ان يكون الحقل FirstNameموجود فى الجدول الثانى table2_name

المعامل BETWEEN ... AND

يستخدم لعرض مجموعه بيانات بين قيمتين نصوص أو أرقام أو تاريخ

بناء الجمله :

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

مثال : على الجدول التالى

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'
```

Result: النتيجه

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'
```

Result: النتيجه

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

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

الأسم المستعار Alias

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

البناء فى حاله الجدول :

```
SELECT column FROM table as Table Alias
```

البناء فى حاله الحقل :

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

مثال : على الجدول التالى

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: النتيجه

نلاحظ أسماء الحقول (الأعمده)

Family	Name
Hansen	Ola
Svendson	Tove
Pettersen	Kari

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

Result: النتيجه

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 او أكثر لجعل النتيجه كامله يجب انشاء علاقه .

الربط بين الجداو 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).

Employees:الجدول الأول

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

Orders: الجدول الثانى

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: النتيجه

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: النتيجه

Name
Hansen, Ola

مثال على استخدام INNER JOIN :

بناء الجمله :

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

تنفيذ الجمله على المثال :

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

Result: النتيجه

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

تقوم الجمله INNER JOIN بعرض جميع البيانات المشتركه بين الجدولين

مثال على أستخدام LEFT JOIN :

بناء الجمله :

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

تنفيذ الجمله على المثال :

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

Result: النتيجه

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

تقوم الجمله LEFT JOIN بعرض جميع البيانات من الجدول الأول Employee حتى لو لم توجد فى الجدول الثانى

مثال على أستخدام RIGHT JOIN :

بناء الجمله :

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

تنفيذ الجمله على المثال :

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

Result: النتيجه

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

تقوم الجمله RIGHT JOIN بعرض جميع البيانات من الجدول الثانى Orders حتى لو لم توجد فى الجدول الأول

مثال على أستخدام جمله INNER JOIN :

نريد عرض اسماء الموظفين الذن طلبوا المنتج (Printer)

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

Result: النتيجه

Name
Hansen, Ola

وبهذا نكون انتهينا من شرح جمله JOIN .

الدمج UNION and UNION ALL

تستخدم لدمج حقلين من جدولين مختلفين ولكن يجب ان يكون نوع البيانات فى الحقلين واحد اى 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
```

Result: النتيجه

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

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

مثل أستخدام 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: النتيجه

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 .

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

الدوال SQL Functions

لغه SQL بها الكثير من الدوال العدديه والحسابيه

البناء الأساسى لأى داله :

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

الدوال :

الداله	الوصف
AVG(column)	لإيجاد الوسط الحسابى للحقل المحدد
COUNT(column)	معرفه عدد الصفوف(السجلات) فى الحقل بدون السجلات الفارغه
COUNT(*)	معرفه عدد الصفوف فى الجدول
First(column)	معرفه قيمه اول سجل فى الحقل
last(column)	معرفه قيمه أخر سجل فى الحقل
Max(column)	معرفه أكبر قيمه سجل فى الحقل
Min(column)	معرفه أضغر قيمه سجل فى الحقل
SUM(column)	معرفه إجمالى القيم فى الحقل
COUNT(DISTINCT column)	عدد السجلات فى الحقل بدون تكرار تعمل فقط على SQL SERVER

مجموعه من الأمثله :

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

SQL GROUP BY and HAVING التجميع والفرز

الدوال العدديه مثل الداله SUM كثيرا ما تحتاج إلى التجميع GROUP BY

بناء الجمله :

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

مثال للتوضيح : على الجدول التالى

Company	Amount
W3Schools	5500
IBM	4500
W3Schools	7100

ننفذ الجمله التاليه:

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

Returns this result: النتيجه

Company	SUM(Amount)
W3Schools	17100
IBM	17100
W3Schools	17100

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

الأن نجرب جمله الأستعلام بعد إضافه GROUP BY :

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

Returns this result: النتيجه

Company	SUM(Amount)
W3Schools	12600
IBM	4500

نلاحظ انه تم جمع كل سجل وحده وأصبحت النتيجه اوضح

الداله HAVING تستخدم لفرز البيانات حسب شرط معين

بناء الجمله :

```
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

ننفذ جمله الأستعلام التاليه :

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

Returns this result : النتيجه

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

بـ *SQL FUNCTION*

فى هذا الجزء من الدرس سوف نقوم بشرح

Data Definition Language (DDL)

هو القسم المسئول عن : تعريف البيانات

١- Create Database : لأنشاء قاعده بيانات جديده
٢- Create Table : لأنشاء جدول داخل قاعده بيانات
٣- ALTER TABLE : للتعديل فى الجدول
٤- DROP TABLE : لحذف الجدول من قاعده البيانات
٥-CREATE INDEX : لأنشاء مفتاح للبحث (الفهارس)
٦- DROP INDEX : لحذف مفتاح البحث

اولا انشاء قاعده بيانات Create Database :

بناء الجمله كالتالى

```
CREATE DATABASE database_name
```

طبعا نحدد اسم قاعده البيانات

قاعده البيانات ديه هتكون فاضيه يعنى مفيش جواها اى جداول .

لأنشاء جدول داخل قاعده البيانات Create Table:

يكون بناء الجمله كالتالى :

```
CREATE TABLE table_name
(
column_name1 data_type,
column_name2 data_type,
.......
)
```

مثال على أنشاء جدول :

```
CREATE TABLE Person
(
LastName text(30),
FirstName text(30),
Address text(150),
Age (Number)
)
```

الان يمكننا انشاء جدوا وتحديد الحقول وانواع وحجم البيانات فيها .

الأن انشاء الفهارس CREATE INDEX :

الفهرس يصمم فى الجدول حتى يجعل عمليه الأستعلام أسرع كما يمكن انشاء اكثر من فهرس نفس الجدول المستخدم لا يرى هذه الفهارس انما هى لتسرع عمليه الأستعلام فقط .

هناك نوعين من الفهارس النوع الأول لا يمكن ان يتكرر به البيانات اما النوع الثانى يمكن تكرار البيانات به .

البناء للفهرس من النوع الفريد (الذى لا يتكرر) Unique Index

```
CREATE UNIQUE INDEX index_name
ON table_name (column_name)
```

البناء للفهرس من النوع العادى Simple Index

```
CREATE INDEX index_name
ON table_name (column_name)
```

مثال :

```
CREATE INDEX PersonIndex
ON Person (LastName)
```

لانشاء فهرس بترتيب عكسى :

```
CREATE INDEX PersonIndex
ON Person (LastName DESC)
```

لانشاء فهرس فى حقلين فى نفس الجدول :

```
CREATE INDEX PersonIndex
ON Person (LastName, FirstName)
```

الأمر Drop

Dropترجمتها الحرفيه إلقاء ولكننا هنا نستخدمها كأمر حذف ولكن حذف ايه (قاعده بيانات – فهرس – جدول)

اولا حذف قاعد بيانات : يكون بناء الجمله كالتالى

```
DROP DATABASE database_name
```

ثانيا حذف جدول من قاعده البيانات : يكون بناء الجمله كالتالى :

```
DROP TABLE table_name
```

ثالثا حذف فهرس : ويختلف البناء حسب نوع قاعده البيانات
مثال :

Syntax for Microsoft SQLJet (and Microsoft Access): نوع قاعده البيانات

```
DROP INDEX index_name ON table_name
```

Syntax for MS SQL Server: نوع قاعده البيانات

```
DROP INDEX table_name.index_name
```

Syntax for IBM DB2 and Oracle: نوع قاعده البيانات

```
DROP INDEX index_name
```

Syntax for MySQL: نوع قاعده البيانات

```
ALTER TABLE table_name DROP INDEX index_name
```

أخير امر حذف البيانات من داخل الجدول دون حذف الجدول :

```
TRUNCATE TABLE table_name
```

الأمر ALTER TABLE

يستخدم فى التعديل على الجدول من إضافه وحذف أعمده (حقول) .

لإضافه حقل : يكون بناء الجمله كالتالى

```
ALTER TABLE table_name
ADD column_name datatype
```

لحذف حقل : يكون بناء الجمله كالتالى :

```
ALTER TABLE table_name
DROP COLUMN column_name
```

مثال : على الجدول التالى

Person:

LastName	FirstName	Address
Pettersen	Kari	Storgt 20

لإضافه حقل جديد :

To add a column named "City" in the "Person" table:

```
ALTER TABLE Person ADD City varchar(30)
```

Result: النتيجه

LastName	FirstName	Address	City
Pettersen	Kari	Storgt 20	

لحذف حقل من الجدول :

To drop the "Address" column in the "Person" table:

```
ALTER TABLE Person DROP COLUMN Address
```

Result: النتيجه

LastName	FirstName	City
Pettersen	Kari	

جمله SELECT INTO

تستخدم عاده فى حفظ نسخه أحتياطيه من الجدول

بناء الجمل كالتالى :

```
SELECT column_name(s) INTO newtable [IN externaldatabase]
FROM source
```

لعمل نسخه إحتياطيه من الجدول فى نفس القاعده :

```
SELECT * INTO Persons_backup
FROM Persons
```

لنسخ الجدول لقاعده بيانات أخرى : نستخدم IN

```
SELECT Persons.* INTO Persons IN 'Backup.mdb'
FROM Persons
```

يمكن أيضا نسخ حقول محدده فقط :

```
SELECT LastName,FirstName INTO Persons_backup
FROM Persons
```

نفس المثال السابق ولكن مع إدخال شرط : Where

```
SELECT LastName,Firstname INTO Persons_backup
FROM Persons
WHERE City='Sandnes'
```

إذا اردنا نسخ الجدول لكن فى وجود علاق مع جدول أخر :

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

جمله CREATE VIEW

تستخدم لانشاء جمله تعرض بيانات تحت شرط معين يمكن أستخدام اى من جمل SQL بها مثل عمل علاقه او إضافه داله من الدوال .

يكون بنائها العام كالتالى :

```
CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition
```

ملاحظات هامه :

١- قاعده البيانات لا تتأثر بما يتم تنفيذه من شروط او دوال

٢- قاعده البيانات لا تقوم بحفظ الناتج فى جدول وانما يقوم محرك قاعده البيانات بتنفيذ الجمله كل مره تقوم بتشغيله فيها

بعض الأمثله المبنيه على قاعده البيانات Northwind :

مثال ١ :

```
CREATE VIEW [Current Product List] AS
SELECT ProductID,ProductName
FROM Products
WHERE Discontinued=No
```

كيف يتم تشغيله : عن طريق الجمله الأتيه

```
SELECT * FROM [Current Product List]
```

مثال ٢ :

```
CREATE VIEW [Products Above Average Price] AS
SELECT ProductName,UnitPrice
FROM Products
WHERE UnitPrice>(SELECT AVG(UnitPrice) FROM Products)
```

يتم عرضه بالجمله:

```
SELECT * FROM [Products Above Average Price]
```

مثال ٣ :

```
CREATE VIEW [Category Sales For 1997] AS
SELECT DISTINCT CategoryName,Sum(ProductSales) AS CategorySales
FROM [Product Sales for 1997]
GROUP BY CategoryName
```

يتم عرضه بالجمله :

```
SELECT * FROM [Category Sales For 1997]
```

ويمكن أستخدام الشرط اثناء عرض البيانات :

```

SELECT * FROM [Category Sales For 1997]
WHERE CategoryName='Beverages'
```

تم بحمد الله الأنتهاء من الدروس
وندعوا الله ان يوفك الجميع لما فيه الخير
اسئلكم الدعاء بظهر الغائب
تحياتى للجميع أحمد حامد
GENIUS-IT@HOTMAIL.COM
عضو فى
الفريق العربى للبرمجه
WWW.ARABTEAM2000.COM
بأسم GENIUS-IT