قواعد بيانات

المحاضرة الخامسة

- **الجداول Tables :**

الجداول عبارة عن وحدات لتخزين البيانات على شكل مصفوفة ثنائية الأبعاد تتكون من من صفوف واعمده .

انشاء جدول :

عند إنشاء جدول، يلزم عموما احترام مايلي:

- أن لا يكون اسم الجدول كلمة محجوزة في لغة SQL .

لإنشاء جدول فالصيغة كما يلي

```
CREATE TABLE MyTable (
ID INT,
FullName VARCHAR(50),
BirthDate DATETIME
)
```

الأمر أعلاه يقوم بإنشاء جدول اسمه MyTable ويتكون من حقول ثلاثة، الأول نوعه رقمي، الثاني نوعه نصي يتسع ل 50 حرف والأخير من نوع التاريخ DataTime.

حذف الجدول :

لحذف جدول نقوم بكتابة الأمر التالي:

```
DROP TABLE MyTable;
```

MyTable هو اسم الجدول المراد حذفه.

تعديل الجداول:

لإضافة بعض الحقول إلى جدول ما فالصيغة دائما هكذا:

```
ALTER TABLE MyTable ADD Age int;
```

هذا إذا أردنا إضافة حقل واحد للجدول عن طريق أوامر SQL فقط نكتب بعد الكلمة ADD اسم الحقل ونوعه لتتم إضافته إلى الجدول بعد تنفيذ الأمر.

أما إذا أردنا إضافة مجموعة من الحقول دفعة واحدة، نفصل بينها بفاصلة هكذا:

```
ALTER TABLE MyTable ADD Age int, Address
VARCHAR(250);
```

أنواع البيانات:

كل حقل من حقول أي جدول له بالضرورة نوع معين من البيانات، حسب القيمة المراد تخزينها فيه، وتنقسم أنواع البيانات إجمالا إلى:

1. **الأنواع الرقمية:**

وتستعمل لتخزين القيم الرقمية، مثلا لو عندنا حقل العمر Age في إحدى الجداول، فحتما علينا اختيار نوع رقمي لتخزين قيم الأعمار، والأنواع الرقمية بدورها تنقسم إلى:

- BIGINT: ويستعمل لتخزين القيم الرقمية . وهذا النوع من البيانات يستعمل 8 بايتات للتخزين البيانات.
- INT: يستعمل أربع بايتات لتخزين البيانات .
- SMALLINT: يستعمل 2 بايت يستعمل 2 بايت.
- TINYINT : يستعمل بايت واحد لتخزين البيانات.
- REAL: يستعمل لحفظ البيانات من نوع أرقام عشرية، وهو يحتاج إلى 4 بايت .
- FLOAT: يستعمل لحفظ البيانات من نوع أرقام عشرية، وهو يحتاج إلى 8 بايت .
- XML: لتخزين وثائق من نوع الإكس أم أل (XML Documents).

2. **الأنواع النصية:**

تستخدم لحفظ البيانات من نوع نصي، على سبيل المثال لو عندنا حقل لحفظ اسم أو عنوان أو أي قيمة نصية، فيلزم أن نختار نوع بيانات نصي، وتنقسم الأنواع النصية إلى:

- VARCHAR(n): وهو من أهم الأنواع النصية، ويسمح بتخزين 8000 بايت من البيانات، وبإمكانك تحديد عدد الأحرف الممكن تخزينها عن طريق تغيير n بالقيمة الرقمية المراد إعطاؤها.

- CHAR(n): يسمح بتخزين النصوص حسب القيمة الرقمية المكتوبة بين القوسين، القيمة الافتراضية لهذا النوع1 ، وقيمته القصوى 8000.
- NCHAR(n).:لتخزين النصوص، قيمته القصوى4000 .

1. التاريخ والوقت:

نحتاج هذا النوع من البيانات لحفظ بعض القيم التي تكون على شكل تاريخ ووقت مثل تاريخ البيع أو الشراء...، ومن أهم أقسامه:

- DateTime: يستخدم هذا النوع من البيانات 8 بايتات.

2. **بعض الأنواع الأخرى:**

- Binary: لتخزين البيانات الثنائية Binary Data .
- Image: ويستعمل لتخزين البيانات من نوع بايت، وأيضا لتخزين الصور.
- Bit: يسمح فقط بتخزين القيمتين 0 و1 ، وهو يستعمل غالبا حينما نتعامل مع بيانات منطقية .

خاصيات الإدخـــــال: وهي مجموعة من الأوامر التي نطبقها على الحقول، من أجل التحقق من القيمة المراد حفظها، وأول هذه الكلمات هي:

- **NOT NULL**: وتستعمل هذه الخاصية لمنع ترك قيمة حقل معين فارغة، وصيغتها هكذا:

```
CREATE TABLE MyTable (
ID INT NOT NULL,
FullName VARCHAR(60))
```

- **IDENTITY**: هذا النوع من الكلمات يطبق فقط على الحقول التي يكون نوعها رقميا من أجل جعل قيمها تزداد تلقائيا عند إضافة أي سطر جديد، مثلا لو أنشأت الجدول التالي وطبقت هذه الكلمة على الحقل ID فسوف تزداد قيمته تصاعديا:

```
CREATE TABLE MyTable (
ID INT IDENTITY(1,1),
```

```
FullName VARCHAR(60))
```

عند إضافة أي سطر جديد، ستلاحظ بأن خانة الحقل ID تمنعك من الكتابة فيها، ولكن بمجرد ما تتجاوزها تقوم برفع قيمة الحقل بواحد تلقائيا، كما تبين الصورة التالية:

	ID	FullName
	1	Khalid
	2	Ahmed
	3	Youssef
▶*	NULL	NULL

تستطيع تغيير القيمة التي يبدأ منها الحقل، وأيضا معامل زيادة القيمة هكذا:

```
CREATE TABLE MyTable (
ID INT IDENTITY(2,3),
FullName VARCHAR(60))
```

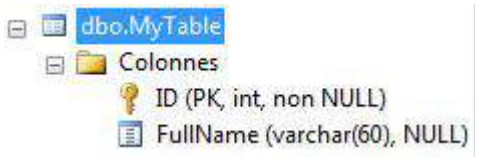
في هذا المثال، ستبدأ قيمة الحقل من الرقم2 ، ويكون معامل الزيادة هو3 ، أي أن القيمة الأولى ستكون2 ، والقيمة الثانية ستكون5 ، والثالثة ستكون... 8 إلخ.

- **PRIMARY KEY**: تطبق هذه الخاصية على الحقل لجعل قيمته متفردة، غير قابلة للتكرار ولا ينبغي أن تترك قيمة الحقل الذي تطبق عليه هذه الخاصية فارغة أي ينبغي أن تكون Not Null ، فمثلا لو عندنا جدول يضم بيانات المواطنين، فمن غير شك هنالك حقل يميز كل مواطن عن غيره، في هذه الحالة الحقل الذي سيكون مفتاحا أساسيا PRIMARY KEY هو الحقل الذي يضم أرقام بطاقات التعريف الوطنية، لأنه من الممكن أن تجد أكثر من مواطن يحملون نفس الاسم، ولكن يستحيل أن يحملوا نفس رقم بطاقة التعريف، عند التعامل مع جداول متصلة فيما بينها عن طريق العلاقات يصبح إلزاميا استخدام المفاتيح الأساسية.

لجعل أحد الحقول مفتاحا أساسيا، نكتب الصيغة التالية:

```
CREATE TABLE MyTable (
ID INT PRIMARY KEY,
FullName VARCHAR(60))
```

لو ذهبت إلى الجدول في القائمة، ودخلت إلى الأعمدة Columns ستجده بالشكل التالي:



رمز المفتاح للدلالة على أن الحقل ID حقل أساسي PRIMARY KEY.

- **UNIQUE**: تستعمل هذه الخاصية لمنع قيمة حقل معين من التكرار، وبالتالي لا يمكن أن يضم نفس الحقل قيمة أكثر من مرة، الفرق بين الخاصية UNIQUE وبين الخاصية PRIMARY KEY أن هذه الأخيرة لا يمكن تطبيقها على الحقول التي تسمح بالقيم الفارغة NULL بينما تسمح الخاصية UNIQUE بالقيمة NULL، ويمكننا تطبيق هذه الخاصية على الحقل هكذا:

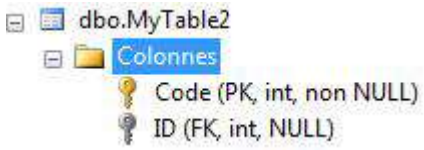
```
CREATE TABLE MyTable (
ID INT UNIQUE,
FullName VARCHAR(60))
```

- **REFERENCE**: هذه الخاصية تطبق على الحقول الأجنبية القادمة من جدول آخر، وتستعمل لتعريف الحقل الأجنبي وتحديد الجدول القادم منه. في هذا المثال سننشئ جدولين، ونحاول تطبيق الخاصية REFERENCE على الجدول الثاني:

```
CREATE TABLE MyTable1 (
ID INT NOT NULL PRIMARY KEY ,
FullName VARCHAR(60))

CREATE TABLE MyTable2 (
Code INT NOT NULL PRIMARY KEY ,
ID INT CONSTRAINT PK_MyTable2 REFERENCES
MyTable1(ID)
)
```

الجدول الثاني يتكون من حقلين، الأول اسمه Code وهو الحقل الأساسي، والثاني اسمه وهو أجنبي قادم من الجدول الأول. لو ذهبت إلى الجدول في القائمة، ودخلت إلى الأعمدة Columns ستجده بالشكل التالي:



المفتاح الذهبي للدلالة على أن الحقل أساسي، PRIMARY KEY والمفتاح الرمادي للدلالة على أن هذا الحقل أجنبي FOREIGN KEY.