إحترف MySQL

أبو عبد الرحمن

وادي التقنية
http://itwadi.com

مجتمع اللينكس العربي
http://linuxac.org

فهرس المحتويات

مقدمة : تعريف قواعد البيانات ولغة SQL.....3
قاعدة البيانات أو Database 3
لغة ال SQL 3
لماذا MySQL على وجه الخصوص ؟.....4
تثبيت MySQL على Debian Gnu/Linux.....5
مقدمة للتعامل مع MySQL.....6
إنشاء الجداول أو ال Tables.....9
ولكن ماهي الجداول أو ال Tables ؟.....9
إضافة بيانات داخل ال Tables.....12
الاستعلام عن بيانات داخل ال Tables.....14
الاستعلام عن البيانات مع وضع شروط.....17
العلامتان = والتي تعنى " يساوى " و =! والتي تعنى " لا يساوى ".....17
العلامتان < والتي تعنى " أكبر من " و > والتي تعنى " أصغر من ".....18
العلامتان =< والتي تعنى " أكبر من أو يساوى " و => والتي تعنى " أصغر من أو يساوى ".....19
Pattern Matching Operators.....19
العلامات المنطقية أو Logical Operators.....24
العلامتان In و Between.....27
دوال المجموع أو Aggregate Functions 43
الدالة min() و max() 44
الدالة sum() و avg() 45
الدالة count() 49
جملة group by 50
جملة Having 54
استخدامات أخرى ل Select 55
الدوال الحسابية في MySQL أو Mathematical Functions 59
mod(x,y) Function 60
ABS(x) Function 61
SIGN(x) Function 62
POWER(x,y) Function 63
SQRT(x) Function 63
ROUND(x) and ROUND(x,y) Function 64
FLoor(x) Function 65
CEILING(x) Function 65
Trignometic Function (Tan(x) , Cos(x) , Sin(x)) 66
تحديث السجلات (Records) باستخدام جملة أمر Update 67

مقدمة : تعريف قواعد البيانات ولغة SQL

في الآونة الأخيرة انتشرت التطبيقات الكثيرة التي تقوم على ترتيب وتنسيق المعلومات وكيفية استغلال تلك المعلومات المرتبة والمنظمة بشكل عملي ، ويلمس ذلك القائمون على المشاريع الكبيرة التي تحتاج إلى برامج لإدارة تلك المعلومات المرتبة والمنظمة هذه البرامج اختلفت وتعددت نتيجة اختلاف عقول البشر في التفكير بشأن موضوع معين ، وذلك الاختلاف ناشيء نتيجة اختلاف إحتياجات كل فرد عن الآخر ، هذه البرامج التي نتحدث عنها برامج تسمى في العالم التقني بنظم إدارة قواعد البيانات أو Database Management System وبالتالي يفرض المصطلح قاعدة البيانات أو Database نفسه للتعريف كالتالي :

قاعدة البيانات أو Database

بشكل بسيط جداً يمكن تعريف قاعدة البيانات على أنها طريقة منظمة لتخزين مجموعة من المعلومات معاً ، وبالتالي فإن مصطلح قاعدة البيانات يشير إلى كيفية وضع المعلومة أو عدة معلومات في صورة مرتبة منسقة يسهل التعامل معها بعد ذلك .

وهنا تجدر الإشارة إلى قاعدة البيانات التي تخص الحاسوب ، ذلك أن البرنامج المسؤول عن إدارة قواعد بيانات الحاسوب تسمى نظم إدارة قواعد البيانات أو فيما يعرف ب Database Management System .

ولكن هناك مفهوم خاطي قد يتصوره البعض أن قواعد البيانات لا توجد إلا على الحاسوب فقط !!

بالفعل الكلام السابق خطأ ، فالحاجة إلى قواعد البيانات ليست مقصورة على الحاسب فقط ، ولكن توجد بين أيدينا قواعد بيانات قد لا نشعر بها ، فعلى سبيل المثال دليل الهاتف يعتبر قاعدة بيانات نظراً لاحتوائه على بيانات الاتصال الخاصة بعدد معين من الأفراد بطريقة هيكلية منظمة تستطيع من خلالها في أي وقت معرفة تفاصيل أي فرد بكل سهولة ويسر .

وما يهمنا في التعامل الآن هي قواعد البيانات التي تخص الحاسب أو الكمبيوتر وذلك أن قاعدة البيانات التي بداخل الحاسب ما هي إلا مجموعة من الجداول تستخدم لتخزين المعلومات المنظمة أو البيانات التي نريدها ثم بعد ذلك يتم إجراء مجموعة من العمليات على تلك الجداول بصورة معينة لحذف وإضافة أو تحديث تلك البيانات لا تقلق من مصطلح جداول الآن سنقوم بتناول تلك المصطلحات لاحقاً إن شاء الله .

لغة ال SQL

أحيانا يخطى البعض أيضا ولا يستطيع التفرقة بين SQL وبين MySQL كبرنامج وللتفرقة بين اللغة وهى SQL وبين نظم إدارة قواعد البيانات نذكر تلك النبذة عن SQL كما يلي :

يرمز المصطلح SQL إلى الكلمات Structured Query Language وهى عبارة عن لغة تستخدم في معالجة البيانات المخزنة في نظم إدارة قواعد البيانات العلائقية أو فيما يعرف ب RDBMS أو Relational Database Management System . ولذلك توفر SQL مجموعة من الأوامر قادرة على التعامل مع البيانات لكى يتم استخراج أو تخزين أو حذف أو إدخال تلك البيانات .

ولكى تكون تلك الأوامر متوافقة لشريحة أكبر من الأفراد تم إخضاع SQL كلغة للتعامل مع أنظمة إدارة قواعد البيانات تحت معايير ANSI أو American National Institute ، والتي تم وضع قواعد معينة لها لتنفيذ تلك الأوامر عند التعامل مع أنظمة إدارة قواعد البيانات المختلفة.

يمكن أن تستخدم SQL لكى تعمل مع أنظمة إدارة قواعد بيانات مختلفة مثل PostgresSQL ، mSQL ، MySQL ، Sybase ، Access ، Microsoft SQL Server ، Oracle ، وغيرها من تلك البرامج .

ونظراً لأن لغة SQL تندرج تحت معايير ال ANSI فإن معظم أوامر وجمل SQL تكون مدعومة من قبل تلك ال RDBMS ، وفى نفس الوقت توجد بعض الاختلافات بين تلك ال RDBMS المختلفة وذلك يتضح في عمل أوامر جديدة تخص كلاً منها على حده ، فأنا أطور نفس الشيء ألذى يقوم زميلي بتطويره وكلانا بفكره الخاص ولكن يوجد بيننا قاسم مشترك يشمل أغلب الأشياء ، كذلك الوضع مع كل تلك أنظمة إدارة قواعد البيانات المختلفة والتي سبق ذكرها وأشهرها على الإطلاق في بيئة مفتوحة المصدر هي MySQL والتي تم شراء الشركة المسؤولة عن تطوير البرنامج مؤخراً وهى شركة MySQL AB من قبل شركة SUN MicroSystems .

في الكلام السابق أشرت إلى مصطلح جديد وهو RDBMS أو Relational Database Management System أو فيما يعرف بنظم إدارة قواعد البيانات العلائقية ، وكما ذكرنا سابقا أن قاعدة بيانات الحاسب ماهي إلا مجموعة من الجداول أو Tables ، ولذلك مصطلح Relational يعنى أنك تستطيع تخزين البيانات في جداول مختلفة ، تكون تلك الجدول مرتبطة ببعضها البعض بصورة معينة ، أو نستطيع القول أن تلك الجداول تكون متعلقة ببعضها البعض بطرق معينة .

لماذا MySQL على وجه الخصوص ؟

اختيار نظام قواعد البيانات يعتمد على عاملين مهمين هما :

1- نوع بيئة نظام التشغيل التي تعمل أنت عليها .
2- الواجبات المطلوبة منك والمراد إنجازها .

ووقع الاختيار بالنسبة ل MySQL لأنها تعمل على أنظمة ال Unix وال Unix-Like بشكل أكثر من رائع كما أنها برنامج حر و مفتوح المصدر .

ملحوظة : توجد نسخة من MySQL تحت اسم ال Enterprise Edition والتي تندرج تحت فئة الدعم مقابل المال .

ولكن قد يطرح أحدنا سؤالاً آخرا يفرض نفسه **لماذا نحتاج إلى قواعد البيانات ؟**

قد أشرت في مقدمة حديثي أن قواعد البيانات عبارة عن مخزن للبيانات يتم معالجته بصورة ما لكى يسهل التعامل مع تلك البيانات ، ولنفترض جدلاً أنك تمتلك ملف نصي يحتوى على بعض الأسماء لعدد من أصدقائك وعناوين البريد الإلكتروني الخاص بهم هل تستطيع أن تسمى ذلك قاعدة بيانات ؟

علمياً وعملياً نعم !! لا تستغرب فعلاً هذا الملف يعتبر قاعدة بيانات في حد ذاته فأنت قمت بترتيب ووضع الأسماء في الملف وأمام كل اسم عنوان البريد الإلكتروني الخاص به وهكذا ، أي أنك قمت بترتيب المعلومات التي تريدها في ملف لكى يسهل لك بعد ذلك التعامل مع ذلك الملف ، فمثلاً تستطيع إضافة وحذف وتحديث تلك البيانات من خلال ذلك الملف ، كما أنك تستطيع كتابة بُريمج صغير وظيفته مثلا البحث داخل ذلك الملف عن اسم معين أو إضافة اسم جديد أو تحديث عنوان بريد إلكتروني لشخص ما قد قام بتغيير بريده الإلكتروني إلخ....

وبالتالي أحتياجات الفرد أو المنظمة إلى قواعد البيانات تكون مختلفة و MySQL بصورة أو بأخرى تمثل حلاً لا نقول مثالياً ولكن لنقول ممتازاً يعتمد عليها كاختيار لنظم إدارة قواعد البيانات وكما ترون أن شركة Sun ليست بالشركة الغبية لكى تقوم بشراء برنامج MySQL ولا ترى فيه أفق النجاح !

تثبيت MySQL على Debian Gnu/Linux

تستطيع تثبيت النسخة الأخيرة المستقرة من ال MySQL على Debian Gnu/Linux من خلال الأمر التالي :

```
debian:~# apt-get install mysql-server
```

كما سوف نحتاج إلى حزمة ال mysql-client أيضا لكى نستطيع التعامل مع حزمة ال mysql-server ونستطيع تثبيت تلك الحزمة من خلال الأمر التالي :

```
debian:~# apt-get install mysql-client
```

ملحوظة : من الممكن تثبيت الحزمتين السابقتين في سطر واحد دون مشاكل فقط افصل بين الاسمين ب مسافة من لوحة المفاتيح.

بعد أن يتم تثبيت كلاً من حزمة ال mysql-server وحزمة ال mysql-client بنجاح سنبدأ العمل بوضع كلمة سر

للمستخدم root لكى يكون له الحق للولوج إلى قاعدة البيانات باستخدام الأمر التالي :

```
debian:~# mysqladmin -u root password 'any_password'
```

طبعاً المكون الرئيسي لل MySQL هي حزمة ال mysql-server والتي هي عبارة عن برنامج يستخدم لتخزين وإدارة البيانات ، أما حزمة ال mysql-client فهي عبارة عن برنامج يستخدم للتعامل مع ال mysql-server وفيه يتم كتابة أوامر ال SQL والجمل الاستعلامية المختلفة الخاصة بال SQL .

ولذلك وبصورة عامة كلمة MySQL تستخدم للتعبير عن الخادم نفسه وهو ال mysql-server أما كلمة mysql بالحروف الصغيرة تستخدم للتعبير عن حزمة ال mysql-client وبالتالي نستخدم حزمة ال client لكى نتمكن من تنفيذ الأوامر داخل خادم ال MySQL وذلك باستخدام أوامر ال SQL كما ذكرنا سابقاً .

والآن لنبدأ في التعامل مع الخادم ونقوم بتنفيذ الأمر التالي في الطرفية على الشكل التالي :

```
debian:~# mysql -u root -p
Enter password:
```

بمجرد أن قمت بالضغط على زر Enter ظهر لك في الطرفية مباشرةً جملة طلب كلمة السر التي قمت بإنشائها سابقا ، والآن قم بإدخال كلمة السر ليظهر لك محث الأوامر على الصورة التالية :

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.0.32-Debian_7etch1-log Debian etch distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

مقدمة للتعامل مع MySQL

لاحظ الآن وقوفك على محث الأوامر الخاص ب mysql والذي منه تقوم بكتابة الجمل والأوامر التي تريد تمريرها إلى MySQL ونبدأ تلك الأوامر باستعراض قواعد البيانات الموجودة داخل MySQL عن طريق الأمر التالي :

```
mysql> show databases;

+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| ser                |
+--------------------+
```

```
3 rows in set (0.00 sec)
```

بعد أن استعرضنا قواعد البيانات التي توجد داخل MySQL سنقوم الآن بإنشاء قاعدة بيانات خاصة بنا ولتكن linux_ac ويكون ذلك باستخدام الأمر التالي من خلال محث أوامر mysql بالشكل التالي :

```
mysql> create database linux_ac;

Query OK, 1 row affected (0.00 sec)
```

نتأكد الآن من وجود قاعدة البيانات الجديدة لدينا من خلال الأمر التالي :

```
mysql> show databases;

+--------------------+
| Database           |
+--------------------+
| information_schema |
| linux_ac           |
| mysql              |
| ser                |
+--------------------+

4 rows in set (0.00 sec)
```

فعلاً تم إنشاء قاعدة البيانات الجديدة تحت اسم linux_ac ، ولاحظ أن الأوامر دائماً تنهى ب ; أو semi-colon ومن المهم ذكره هنا أن قاعدة البيانات linux_ac تم إنشائها من قبل المستخدم root وبالتالي فإن أي مستخدم آخر على التوزيعة لن يستطيع استخدام قاعدة البيانات تلك والعمل عليها إلا إذا تم منح التصريح للمستخدم الذي يريد العمل على تلك القاعدة من قبل ال root ، فعلى سبيل المثال نريد مثلا منح تصريح العمل على قاعدة البيانات linux_ac لمستخدم على التوزيعة تحت اسم muhammad فنستطيع عمل ذلك من خلال الأمر grant كما يلي :

```
mysql> grant all on linux_ac.* to muhammad@localhost identified by
'solaris';

Query OK, 0 rows affected (0.05 sec)
```

في المثال السابق قمنا بمنح المستخدم muhammad على ال localhost كل التصاريح الخاصة بقاعدة البيانات المنشأة حديثاً وهى linux_ac ، كذلك قمنا بوضع كلمة السر الخاصة به وفى مثالنا كانت solaris ومن البديهي تستطيع تغيير كلمة السر التي تريدها إلى أي كلمة سر أخرى كما تستطيع منح التصاريح لأي مستخدم آخر .

وللتأكد من ذلك نقوم بالخروج من ال mysql بكتابة التالي :

```
mysql> \q
Bye
```

أو كتابة كلمة quit أيضا :

```
mysql> quit
Bye
```

بعد ذلك نقوم بالاتصال بقاعدة البيانات عن طريق mysql ولكن هذه المرة مع المستخدم muhammad بالشكل التالي :

```
debian:~# mysql -u muhammad -p
Enter password:
```

قم بإدخال كلمة السر التي منحتها للمستخدم muhammad والتي كانت في مثالنا solaris ، بعد ذلك ستتمكن من الاتصال ب MySQL والعمل على قاعدة البيانات linux_ac بكل سهولة كما يلي :

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 5.0.32-Debian_7etch1-log Debian etch distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

ولاحظ أن المستخدم muhammad قد منح التصريح للعمل على قاعدة البيانات linux_ac فقط ، وبالتالي فلا يحق له العمل على أي قاعدة بيانات أخرى ما لم يتم منح التصريح الكافي له ، فمثلاً لو قمت بتنفيذ التالي أمام محث أوامر ال mysql :

```
mysql> use ser;

ERROR 1044 (42000): Access denied for user 'muhammad'@'localhost' to
database 'ser'
```

الأمر السابق use أخبرت محث أوامر mysql أنني أريد العمل على قاعدة البيانات والتي تسمى ser فظهرت رسالة خطأ تفيد بأن المستخدم muhammad ليست له الصلاحية للعمل على قاعدة البيانات ser ، أما لو قمت بتغيير الأمر

للعمل على قاعدة البيانات linux_ac من خلال نفس الأمر use بالشكل التالي :

```
mysql> use linux_ac;

Database changed
```

فنجد فعلا أنه تم التغيير إلى قاعدة البيانات linux_ac والتي تم منح التصريح اللازم للعمل عليها إلى المستخدم muhammad .

إنشاء الجداول أو ال Tables

بعد أن استعرضنا كيفية إنشاء قاعدة بيانات جديدة باستخدام الأمر create database سوف ننتقل إلى جزئية أخرى وهى إنشاء جداول أو Tables داخل تلك قاعدة البيانات ، وبالتالي نستطيع القول أن قواعد البيانات تقوم بتخزين البيانات اللازمة داخل الجداول .

ولكن ماهي الجداول أو ال Tables ؟

الجدول أو ال Table عبارة عن هيكل مكون من صفوف أو rows وأعمدة أو columns وكل عمود يقوم بتعريف نوع معين من البيانات أو data type سواء كانت بيانات رقمية أو عددية ، أما الصفوف فتحتوى على مجموع البيانات التي يتم تخزينها وتسمى سجلات أو records ومثال على ذلك الجدول التالي :

```
+---------+-----------+------+
| f_name  | l_name    | age  |
+---------+-----------+------+
| ahmad   | hassan    |   28 |
| amr     | muhsen    |   32 |
| sayed   | muhammad  |   32 |
| Mustafa | Karim     |   27 |
| sherif  | shahin    |   26 |
| Youssef | ahmad     |   27 |
| Shahida | Ali       |   32 |
| Marriam | Mahmoud   |   36 |
```

الجدول السابق يحتوى على ثلاثة أعمدة تشمل الاسم الأول والذي رمزنا إليه ب f_name كذلك الاسم الأخير والذي رمزنا إليه ب l_name وأخيراً العمر والذي رمزنا إليه ب age ، وكما أشرنا سابقاً أن الأعمدة تعرف نوع البيانات المدخلة ، أما الصفوف فتمثل البيانات المدخلة نفسها أو فيما تسمى بالسجلات أو records ، وأود أن أشير أيضاً أن قاعدة البيانات قد تحتوى جدول واحد أو أكثر من جدول يتم الاستعلام عن البيانات المتعلقة ببعضها في تلك الجداول باستخدام جمل ال SQL الاستعلامية .

والآن نبدأ بالتعامل الحقيقي مع قاعدة البيانات linux_ac والتي سيتم إنشاء جدول فيها يحتوى على مجموعة من الأعمدة والتي بدورها سوف تحتوى على البيانات الحقيقة التي سوف يتم إدخالها لذلك الجدول ونبدأ مباشرةً بتنفيذ الأمر التالي من أمام محث أوامر mysql :

```
mysql> use linux_ac;

Database changed

mysql> create table members_data
    -> (
    -> mem_id int unsigned not null auto_increment primary key,
    -> f_name varchar(20),
    -> l_name varchar(20),
    -> age int,
    -> email varchar(60)
    -> );

Query OK, 0 rows affected (0.08 sec)
```

ملحوظة : الأوامر وأسماء الأعمدة في ال MySQL ليست حساسة تجاه الحروف الكبيرة والصغيرة أي تكون not case-sensitive بمعنى الأمر create هو نفسه الأمر CREATE هو نفسه الأمر Create إلخ ، أما أسماء قواعد البيانات والجداول فمن الممكن أن تكون حساسة تجاه الحروف الكبيرة والصغيرة وذلك اعتماد على بيئة نظام التشغيل التي تعمل عليها ولذلك في حالتنا هذه أي بالعمل على Debian Gnu/Linux فإننا سوف نتنبه لهذه النقطة .

شرح الجملة السابقة

الأمر create table :يستخدم الأمر create table لإنشاء جدول داخل قاعدة بيانات نعمل عليها ويتم إلحاق اسم الجدول المراد إنشاؤه بعد الأمر create table وفى حالتنا هذه قمنا بتسمية الجدول باسم members_data ويحق لك تسمية الجدول بأي اسم تريده .

بعد ذلك قمنا بضغط زر Enter كنوع من ترتيب المدخلات بشكل جيد ثم ضغطنا Enter مرة أخرى ، بعد ذلك قمنا بإنشاء أول عمود داخل الجدول تحت اسم mem_id والذي سوف يحتوى على ال id الخاص بكل عضو وهنا قمنا بتعريف ذلك العمود بالقيم integer أو int أي أن المتغير mem_id سوف يأخذ قيم أعداد صحيحة ليس فيها كسور ، بعد ذلك منحنا العمود mem_id ال attribute الخاصة به وهى unsigned أي أن رقم العضو سيكون عدد صحيح وبإشارة موجبة (Positive Integer) ، ثم أضفنا خاصية أخرى وهى not null أي أن العمود mem_id سوف يأخذ قيمة على الدوام ولا يمكن أن يكون فارغاً ، بعد ذلك أضفنا الخاصية auto_increment والتي تعنى زيادة رقم العضوية بشكل تلقائي دون تدخل منا أو وضع قيم لهذا العمود بشكل يدوى ، وبالتالي لن نحتاج أبداً لإضافة قيم لهذا العمود طالما أن MySQL سوف تقوم بهذه المهمة ، كما أن ذلك سوف يؤدى بدوره لأن تكون قيم العمود mem_id قيم

وحيدة لن تتكرر أي لن تجد رقميّ عضوية متشابهان ، ثم أخيراً الصفة primary key والتي تساعد في فهرسة العمود لتسهيل أكثر عند عملية البحث عن قيمة معينة وكل قيمة أيضا لابد أن تكون قيمة وحيدة أي لا تتكرر .

ولنوضح قليلاً مفهوم صفة الفهرسة باستخدام ال primary key مثلا حينما تشترى كتاب في صورة ورقية فيوجد للكتاب فهرس منظم يسهل على القاري البحث في موضوعات الكتاب إما عن طريق اسم الموضوع أو عن طريق القسم الذي يخضع إليه عنوان البحث وهكذا ، ولذلك في مثالنا عرفنا العمود mem_id أنه ذو صفة primary key أي أننا قررنا تنظيم بيانات الأشخاص تبعا للعمود mem_id ولذلك يجب أن نخبر MySQL أن هذا العمود هو مفتاح التعامل مع محتويات الجدول و ذلك بأن نعطيه الصفة PRIMARY KEY ، وال primary key في حد ذاته كما ذكرنا يتضمن فهرسا للقيم وفى نفس الوقت لا تتغير والفائدة من ذلك تتضح جلياً من المثال التالي :

نفترض أنه يوجد لدينا عضوان يحملان نفس الاسم في العمود f_name والعمود l_name فالفيصل مثلاً حينها هو رقم ال mem_id الخاص بكل واحد منها ، لأنه كما ذكرنا أننا وضعنا صفة ال primary key للعمود بحيث لا يمكن أن يتكرر رقمان في نفس العمود وهو mem_id .

بعد ذلك قمنا بإضافة العمود الثاني وهو خاص بالاسم الأول للعضو تحت اسم f_name وعرفنا نوع البيانات التي سيتضمنها ذلك الحقل أو العمود من نوع varchar أو اختصاراً ل variable character وقمنا بوضع حد لأقصى عدد من الحروف للاسم الأول وهو 20 حرف بين قوسين ، ثم أضفنا عموداً آخر خاص بالاسم الأخير للعضو تحت اسم l_name وعرفنا نوع البيانات التي سيتضمنها ذلك الحقل أو العمود من نوع varchar أيضاً وقمنا بوضع حد لأقصى عدد من الحروف للاسم الأخير وهو 20 حرف بين قوسين ، ثم أضفنا عموداً يشمل العمر أو age وتم تعريف نوع البيانات الخاصة به من نوع int أو integer ، ثم أخيراً أضفنا حقلاً آخر للبريد الإلكتروني تحت اسم email وتم تعريف نوع البيانات الخاصة به من نوع varchar وتم تحديد أقصى عدد للحروف تخص هذا الحقل وكانت 60 حرفاً ثم أخيراً أنهينا الأمر بوضع قوس ثم مباشرةً بوضع ; أو semi-colon .

ملحوظة : لابد أن تفصل بين كل اسم عمود وعمود ب , أو colon ثم تنهى الأمر ب ; أو semi-colon لكى يتم تنفيذ الأمر بشكل صحيح داخل ال MySQL وذلك بالضغط على زر Enter لكى تتم عملية التنفيذ .

أيضاً سوف تلاحظ كلما ضغطت على زر Enter مع نهاية كل سطر إذا لم يحتوى على ; أو semi-colon أن محث أوامر mysql قد فهم أن الأمر لم يكتمل بعد وما زال هناك بعض الأمور لم يتم إدخالها ولذلك يتحول محث الأوامر مع كل ضغطة Enter إلى العلامة -> وبمجرد وضعك لل semi-colon بعد القوس الأخير فحينها سوف يتم تنفيذ الأمر كما أشرنا سابقا .

والآن بعد أن قمنا بإنشاء أول جدول داخل قاعدة البيانات linux_ac سنقوم باستعراض الجداول داخل القاعدة linux_ac لنتأكد فعلياً من إنشاء الجدول وذلك كما يلي :

```
mysql> show tables;
```

```
+--------------------+
| Tables_in_linux_ac |
+--------------------+
| members_data       |
+--------------------+

1 row in set (0.00 sec)
```

كما تلاحظ فعلاً تم إنشاء الجدول ووضعه داخل قاعدة البيانات linux_ac ، والآن لنتعرض لأمر آخر وهو استعراض أعمدة الجدول أو الحقول التي بداخله وهو الأمر describe كما يلي :

```
mysql> describe members_data;

+--------+------------------+------+-----+---------+----------------+
| Field  | Type             | Null | Key | Default | Extra          |
+--------+------------------+------+-----+---------+----------------+
| mem_id | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| f_name | varchar(20)      | YES  |     | NULL    |                |
| l_name | varchar(20)      | YES  |     | NULL    |                |
| age    | int(11)          | YES  |     | NULL    |                |
| email  | varchar(60)      | YES  |     | NULL    |                |
+--------+------------------+------+-----+---------+----------------+

5 rows in set (0.02 sec)
```

كما رأينا قام الأمر describe باستعراض أعمدة الجدول members_data كما قام أيضا بتوضيح نوع البيانات التي تخص كل عمود أو حقل والصفات التي تخص كل حقل على حده .

إضافة بيانات داخل ال Tables

الصيغة العامة لإضافة بيانات داخل جداول ال MySQL تكون على الشكل التالي :

```
INSERT into table_name (column1, column2....)
values (value1, value2...);
```

حيث table_name اسم الجدول الذي سيتم إضافة البيانات فيه ويكون إضافة البيانات داخل الأعمدة التي نريدها على وجه التحديد والتي تتمثل في اسم العمود الأول والعمود ألتأني وهكذا .... ثم بعد ذلك تأتى القيم التي ينبغي إضافتها وذلك من خلال value1 و value2 ...

ونأخذ مثالاً عملياً على ذلك :

```
mysql> insert into members_data (f_name, l_name, age, email) values
```

```
("muhammad","ahmad",23,"wxyz@gmail.com");

Query OK, 1 row affected (0.06 sec)
```

شرح الأمر :

أولاً: قمنا باستخدام الأمر insert مع into لإضافة مدخلات جديدة داخل الجدول members_data .

ثانيا: القيم التي تخص كل من العمود f_name والعمود l_name والعمود email تكون داخل " " أو double quotes وذلك للدلالة على أنها عبارة عن قيم نصية سوف تدخل إلى تلك الحقول كما هي .

ثالثا : القيمة بالنسبة للعمود age لم نضعها داخل " " لأنها عبارة عن أرقام صحيحة (INTEGER).

رابعا: لو لاحظت في الأمر السابق أننا لم نضع العمود mem_id داخل جملة أمر ال insert لأننا كما قلنا سابقا أن ال MySQL سوف تتولى هذه المهمة وتقوم بمنح كل صف جديد داخل الجدول رقم id غير مكرر مع التنبيه أن الأرقام التي يتم إضافتها إلى قاعدة البيانات تكون أكبر من سابقتها بمقدار واحد في العدد .

ومن ذلك يتضح أنه عند الرغبة في إضافة بيانات جديدة لابد من تكرار جملة أمر ال insert بحسب عدد البيانات المراد إدخالها ، ولكن قد يكون الأمر شاق عندما تريد إدخال 100 اسم داخل الجدول أليس كذلك ؟

لجعل الأمور أكثر سهولة نقوم بإنشاء ملف نصي على سطح المكتب مثلا تحت اسم members.dat ونقوم بإضافة جميع البيانات المراد إدخالها من خلال جملة أمر ال insert ثم بعد ذلك نستدعى الملف إلى قاعدة البيانات المراد إدخال جمل أمر ال insert من خلال الطرفية بالشكل التالي :

```
debian:~# mysql linux_ac <members.dat -u root -p
Enter password:
```

طبعاً في البداية لابد أن تكون داخل المجلد الذي يحتوى الملف members.dat ، بعد الضغط على زر Enter ستتطلب منك الطرفية إدخال كلمة المرور الخاص بالمستخدم الذي تريده أن يعمل على قاعدة البيانات linux_ac وفى حالتنا هذه قمنا بالدخول عن طريق المستخدم الجذر أو root ولنتأكد فعلا من إدخال البيانات داخل قاعدة البيانات linux_ac بشكل سليم ننفذ الأمر التالي :

```
mysql> select * from members_data;

+--------+----------+---------+------+----------------------+
| mem_id | f_name   | l_name  | age  | email                |
+--------+----------+---------+------+----------------------+
|      1 | muhammad | ahmad   |   23 | wxyz@gmail.com       |
|      2 | ahmad    | ameen   |   48 | ahmad@gmail.com      |
```

```
|      3 | ahmad     | youssef |   32 | a_youssef@bignet.com |
|      4 | muhammad  | ismael  |   32 | m_ismael@bignet.com  |
|      5 | sherif    | shahin  |   32 | s_shahin@bignet.com  |
|      6 | sherif    | faroo2  |   32 | s_faroo2@bignet.com  |
|      7 | muhammad  | mahfouz |   32 | m_mahfouz@bignet.com |
|      8 | sarah     | mahmoud |   32 | s_mahmoud@bignet.com |
|      9 | marwa     | hassan  |   32 | m_hassan@bignet.com  |
|     10 | muhammad  | wadood  |   32 | m_wadood@bignet.com  |
|     11 | muhammad  | antary  |   32 | m_antary@bignet.com  |
|     12 | safwat    | hegazy  |   32 | s_hegazy@bignet.com  |
|     13 | ahmad     | antar   |   32 | a_antar@bignet.com   |
|     14 | kamel     | ahmad   |   32 | k_ahmad@bignet.com   |
|     15 | muhammad  | kamal   |   32 | m_kamal@bignet.com   |
|     16 | muhammad  | taha    |   32 | m_taha@bignet.com    |
+--------+-----------+---------+------+----------------------+

16 rows in set (0.00 sec)
```

فعلا تم وضع البيانات التي قمنا بتحريرها داخل الملف النصي members.dat بشكل سليم كما ترى .

الاستعلام عن بيانات داخل ال Tables

الآن أصبح الجدول members_data مليء بالبيانات التي تُمكننا من العمل عليه بشكل أوسع وتطبيق جمل استعلامية أخرى مثل جملة الأمر select والتي نحن بصدد الحديث عنها ، فالآن سوف نرى كيفية استخراج بيانات أو الاستعلام عنها كما قلنا باستخدام الأمر select وتكون صيغته العامة بالشكل التالي :

```
SELECT column_names from table_name [WHERE ...conditions];
```

كما تلاحظ في البداية جملة أمر select بدأت بالأمر نفسه ثم بعد ذلك اسم العمود أو الحقل المراد استخراج البيانات منه أو الاستعلام عنها وذلك من خلال اسم الجدول الخاص بنا ، أما جملة [WHERE ...conditions] فيمكن أن نضعها أو لا نضعها وسوف نتطرق إلى الخيارات المتاحة مع تلك الجملة لاحقاً فالهيكل الأساسي لعملية الاستعلام أو استخراج البيانات هو اسم الحقل أو العمود كذلك اسم الجدول الذي يحتوى ذلك العمود .

وعلى سبيل المثال لكى نقوم بالاستعلام عن الاسم الأول والاسم الأخير لكل الأعضاء داخل الجدول members_data نقوم بتنفيذ التالي :

```
mysql> select f_name, l_name from members_data;

+----------+---------+
| f_name   | l_name  |
```

```
+----------+---------+
| muhammad | ahmad   |
| ahmad    | ameen   |
| ahmad    | youssef |
| muhammad | ismael  |
| sherif   | shahin  |
| sherif   | faroo2  |
| muhammad | mahfouz |
| sarah    | mahmoud |
| marwa    | hassan  |
| muhammad | wadood  |
| muhammad | antary  |
| safwat   | hegazy  |
| ahmad    | antar   |
| kamel    | ahmad   |
| muhammad | kamal   |
| muhammad | taha    |
+----------+---------+

16 rows in set (0.00 sec)
```

طبعا تنبه للفاصلة أو ال colon بين كل اسم عمود وعمود ثم في آخر جملة الاستعلام تنبه أيضا لوضع ال ; أو ال semi-colon لكى يبدأ mysql prompt في تنفيذ الحملة .

وبالتالي تستطيع الآن اللعب مع جملة أمر select لتنفيذ استعلامات أخرى مختلفة كأن تستعلم مثلاً عن عمر كل الأعضاء من خلال الجملة التالية :

```
mysql> select age from members_data;

+------+
| age  |
+------+
|   32 |
|   30 |
|   38 |
|   27 |
|   29 |
|   39 |
|   45 |
|   42 |
|   24 |
|   36 |
|   22 |
|   17 |
|   53 |
|   18 |
+------+
```

```
14 rows in set (0.00 sec)
```

جملة ترفيهية أليس كذلك حسناً العب قليلاً مع الجملة !!

وتستطيع أيضا استخدام ال * والتي تعنى كل الأعمدة لجلب كل البيانات الخاصة بالأعضاء دفعة واحدة دون كتابة جميع الحقول وذلك كما فعلنا سابقا من خلال التالي :

```
mysql> select * from members_data;

+--------+----------+---------+------+----------------------+
| mem_id | f_name   | l_name  | age  | email                |
+--------+----------+---------+------+----------------------+
|      3 | ahmad    | youssef |   32 | a_youssef@bignet.com |
|      4 | muhammad | ismael  |   30 | m_ismael@bignet.com  |
|      5 | sherif   | shahin  |   38 | s_shahin@bignet.com  |
|      6 | sherif   | faroo2  |   27 | s_faroo2@bignet.com  |
|      7 | muhammad | mahfouz |   29 | m_mahfouz@bignet.com |
|      8 | sarah    | mahmoud |   39 | s_mahmoud@bignet.com |
|      9 | marwa    | hassan  |   45 | m_hassan@bignet.com  |
|     10 | muhammad | wadood  |   42 | m_wadood@bignet.com  |
|     11 | muhammad | antary  |   24 | m_antary@bignet.com  |
|     12 | safwat   | hegazy  |   36 | s_hegazy@bignet.com  |
|     13 | ahmad    | antar   |   22 | a_antar@bignet.com   |
|     14 | kamel    | ahmad   |   17 | k_ahmad@bignet.com   |
|     15 | muhammad | kamal   |   53 | m_kamal@bignet.com   |
|     16 | muhammad | taha    |   18 | m_taha@bignet.com    |
+--------+----------+---------+------+----------------------+

16 rows in set (0.00 sec)
```

لاحظ أيضاً أن العمود mem_id فعلاً قد تم إدخال البيانات فيه بشكل تلقائي دون تدخل منا أثناء تنفيذ جملة أمر insert .

الاستعلام عن البيانات مع وضع شروط

في هذه النقطة سنتناول إن شاء الله كيفية الاستعلام عن بيانات ولكن هذه المرة مع وضع قيد أو شرط في جملة أمر select وسيكون القيد أو الشرط داخل جملة أمر select مرتبط بجملة where والتي يأتي بعدها الشرط المراد وضعه على جملة الاستعلام ، وقد ذكرنا سابقاً الصيغة العامة لجملة أمر select وأكررها حتى تثبت في ذهن القارىء وهى بالشكل التالي :

```
SELECT column_names from table_name [WHERE ...conditions];
```

وكما ذكرنا سابقاً أن جملة [WHERE ...conditions] هي جملة اختيارية يمكن وضعها أو قد لا نحتاج إليها كما ، وسنتعرض في كلامنا التالي إلى استخدام الجمل الاستعلامية مع الشروط وهنا تظهر قوة نظم قواعد البيانات العلائقية أو

RDBMS بصورة جلية حيث يمكن تطبيق شروط معينة لجلب أو استخراج بيانات بُناءاً على تلك الشروط ، وسنبدأ أولاً مع الشروط التي تخص عمليات المقارنة وأولها :

العلامتان = والتي تعنى " يساوى " و =! والتي تعنى " لا يساوى "

مثال يوضح استخدامات تلك العلامات :

```
mysql> select f_name, l_name from members_data
    -> where f_name='muhammad';

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| muhammad | ismael  |
| muhammad | mahfouz |
| muhammad | wadood  |
| muhammad | antary  |
| muhammad | kamal   |
| muhammad | taha    |
+----------+---------+

6 rows in set (0.03 sec)
```

هنا استخدمنا جملة أمر select لجلب الاسم الأول والاسم الأخير من الجدول members_data ولكن قمنا بوضع شرط وهو جلب كل المستخدمين والذي يبدأ اسم كل منهم ب muhammad وظهرت النتيجة بالشكل السابق .

لاحظ : الشرط الذي قمنا ببناء جملة أمر ال select عليه هو أن يكون الاسم الأول للمستخدمين muhammad وتم وضعه داخل single quotes أي الموضوع سيان لا يوجد فرق بين ال double quotes أو single quotes المهم أن القيم التي تكون من نوع varchar لابد من وضعها داخل quotes سواء كانت single أو double .

مثال آخر :

```
mysql> select f_name, l_name from members_data
    -> where age=32;

+--------+---------+
| f_name | l_name  |
+--------+---------+
| ahmad  | youssef |
+--------+---------+

1 row in set (0.00 sec)
```

أترك لكم شرح الأمر

طبعا تستطيع استخدام العلامة =! والتي تعنى " لا يساوى " فضع ما تشاء من أمثلة تريدها .

العلامتان < والتي تعنى " أكبر من " و > والتي تعنى " أصغر من "

تعتبر كلاً من العلامة " أكبر من " أو > والعلامة " أصغر من " أو < من الأهمية بمكان داخل جملة الشروط ويتضح ذلك جلياً مثلاً عندما تريد الاستعلام عن عدد من الأعضاء يكون سنهم أكبر من قيمة معينة وهكذا.

ونأخذ مثالاً عملياً على ذلك من خلال جدولنا members_data داخل قاعدة البيانات linux_ac وفى المثال نريد الاستعلام عن الاسم الأول والاسم الأخير للأعضاء أصحاب عمر أكبر من 32 عام كالتالي :

```
mysql> select f_name, l_name, age from members_data
    -> where age > 32;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| sherif   | shahin  |   38 |
| sarah    | mahmoud |   39 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| safwat   | hegazy  |   36 |
| muhammad | kamal   |   53 |
+----------+---------+------+

6 rows in set (0.00 sec)
```

كذلك الأمر مع العلامة "أصغر من" أو < تستطيع عمل أمثلة عليها بنفسك للتدريب .

العلامتان =< والتي تعنى " أكبر من أو يساوى " و =< والتي تعنى " أصغر من أو يساوى "

رأينا في الأمثلة السابقة كيفية استخدام العلامات الأربع وهى " يساوى " أو = والعلامة الثانية " لا يساوى " أو =! والعلامة الثالثة " أكبر " من أو > والعلامة الرابعة " أصغر من " أو < ورأينا الفائدة التي تعود علينا من استخدام تلك العلامات ولكن لو تلاحظ في المثال الأخير عن كيفية تطبيق الشرط باستخدام العلامة " أكبر من " كانت النتائج كلها أكبر من الشرط الذي قمنا بوضعه وهو عمر 32 أي أن أصحاب ذلك العمر لم يظهروا من خلال نتيجة جملة أمر ال select ولذا ماذا لو أردنا الاستعلام عن الأعضاء أصحاب عمر أكبر من أو يساوى 32 مثلاً ؟

يأتي هنا دور العلامة =< والتي تعنى " أكبر من أويساوى " ، والآن لنقوم بتطبيق الجملة السابقة مرة أخرى ولكن

باستخدام العلامة =< وتكون بالشكل التالي :

```
mysql> select f_name, l_name, age from members_data
    -> where age >= 32;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| ahmad    | youssef |   32 |
| sherif   | shahin  |   38 |
| sarah    | mahmoud |   39 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| safwat   | hegazy  |   36 |
| muhammad | kamal   |   53 |
+----------+---------+------+

7 rows in set (0.01 sec)
```

كما ترى ظهر العضو ahmad صاحب العمر 32 عام في نتيجة الجملة التي قمنا بتنفيذها سابقاً ونفس الشيء مع العلامة => والتي تعنى " أصغر من أويساوى " وقم بتطبيق عدة أمثلة عليها كي تستقر الأمور لديك .

Pattern Matching Operators

بعد أن تطرقنا فيما سبق إلى كيفية استخدام علامات المقارنة أو Comparison Operators مثل = و =! و < و > ، سننتقل الآن إلى جزئية أخرى وهى ال Pattern Matching وصراحة لا أريد ترجمة المصطلح حتى لا يفقد معناه المطلوب وسأكتفي بتوضيح معنى مصطلح ال Pattern Matching من خلال الأمثلة التي سوف نطبقها على تلك الجزئية وسيكون تطبيق ال Pattern Matching باستخدام جملة where أيضاً ، وبذلك تتفق ال Pattern Matching مع ال Comparison Operators في استخدامهما لجملة Where ، كما أننا سوف نستعمل أداة تسمى like أو like operator بدلا من العلامة = أو يساوى .

تعلمنا سابقا أن ال Operator " يساوى " أو = كانت تستخدم للاستعلام المشروط وجلب بيانات متماثلة للشرط الذي وضعناه ، فمثلاً حينما أردنا جلب كل المستخدمين أصحاب الاسم الأول "muhammad" كل ما فعلناه وضعنا الشرط وهو أن يكون الاسم الأول يساوى "muhammad"وكانت جملة الشرط بالشكل التالي :

```
where f_name='muhammad'
```

أي أن ال Operator "يساوى" أو = استخدمناه لعمل موائمة لجملة الاستعلام المشروطة فعلى سبيل المثال نريد الاستعلام عن أسماء الأعضاء (الاسم الأول ، الاسم الأخير) بشرط أن يكون الاسم الأول للعضو sherif فكيف ننفذ ذلك ؟

للإجابة على ذلك السؤال تعالوا أولاً نحلل السؤال خطوة خطوة ، جملة الاستعلام المطلوبة تريد الاستعلام عن الاسم الأول وهو f_name والذي يمثل عموداً أو حقلاً داخل قاعدة البيانات لدينا ،كما أننا نريد الاستعلام عن الاسم الأخير وهو l_name والذي يمثل لدينا عموداً أو حقلاً داخل قاعدة البيانات ولكن في السؤال ظهر قيد أو شرط جديد ألا وهو أن يكون الاسم الأول للعضو sherif وتكون الجملة بالشكل التالي :

```
mysql> select f_name, l_name from members_data
    -> where f_name = 'sherif';

+--------+--------+
| f_name | l_name |
+--------+--------+
| sherif | shahin |
| sherif | faroo2 |
+--------+--------+

2 rows in set (0.03 sec)
```

كما ترى نتيجة جملة الاستعلام التي قمنا بتنفيذها ، ظهرت النتيجة كما أردنا وبالتالي تستطيع تغيير القيد أو الشرط كيفما تشاء بأن تقول مثلا أريد الاستعلام عن الأعضاء أصحاب عمر أكبر من 20 عاما إلخ من تلك الأمور .

أرى أن الملل بدأ يتسرب إلى أذنيك P: ، وأسمع همس القراء ولسان حالهم يقول " قد شرحت ذلك سابقا "

في الحقيقة نعم شرحت ذلك الجزء سابقاً ولكن أردت التركيز على تلك الجزئية لكى تلمس الفارق بين ال Operator الذي سوف نستخدمه وهو like وبين ال Operator الذي استخدمناه وهى علامة يساوى أو = .

ولكى نوضح الفارق في الاستخدام بين like وبين = سأقوم بتطبيق مثال ، وليكن التالي نفترض أنك تريد الاستعلام عن أسماء الأعضاء الاسم الأول والاسم الأخير لكل عضو بشرط أن يبدأ الاسم الأول للعضو بحرف "m" هل تستطيع فعل ذلك بال Operator يساوى أو = ؟

من البديهي والمنطقي لا ، لأن = تقوم بعملية مماثلة لمتغير نصي بصورة كاملة أي أنك مثلا لو كتبت الجملة على الصورة التالية :

```
mysql> select f_name, l_name from members_data where f_name = 'm';
Empty set (0.00 sec)
```

فكانت النتيجة هي أن الجدول لا يوجد به الشرط الذي قمت أنت بوضعه مع أنك تعلم تمام العلم أنه توجد أسماء لديك داخل الجدول تبدأ بحرف " m " هل اتضح الأمر قليلاً لديك واستطعت الآن أن تميز الفارق ؟؟ ما زال في الأمر غموضا ! لا تقلق سأوضح أكثر وبتفصيل .

الجملة التي تستخدم في السماح لنا بعمل Pattern Matching تكون بالشكل التالي :

```
mysql> select f_name, l_name from members_data
    -> where f_name like 'm%';

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| muhammad | ismael  |
| muhammad | mahfouz |
| marwa    | hassan  |
| muhammad | wadood  |
| muhammad | antary  |
| muhammad | kamal   |
| muhammad | taha    |
+----------+---------+

7 rows in set (0.00 sec)
```

لاحظ في المثال السابق أن ال Pattern Matching في جملة الاستعلام كانت مع شرط أن يبدأ الاسم الأول للعضو بحرف " m " وتم استخدام ال Operator في الجملة وهو like مع ال percentage sign وهى " % " والتي تعنى إهمال تأثير ما قبلها أو إهمال تأثير ما بعدها كما هو حال ال * أو asterisk على أنظمة ال unix و unix-like .

ملحوظة : أود أن ألفت انتباهك إلى جزئية مهمة هنا أن " % " تساوى في الوظيفة " * " ولكن ذلك يتم على أنظمة unix و unix-like أما عند استخدام ال * مع ال MySQL فهي تعنى " All Columns " فأرجو أن تنتبه لتلك النقطة ولا تخلط بين معنى ال" * " على Gnu/linux وبين معنى ال" * " على MySQL .

ولنكمل حديثنا مع مثال آخر وهو أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) والذي يبدأ اسم العضو بحرف " s " فيكف نفعل ذلك راقب معى التالي :

```
mysql> select f_name, l_name from members_data
    -> where f_name like 's%';

+--------+---------+
| f_name | l_name  |
+--------+---------+
| sherif | shahin  |
| sherif | faroo2  |
| sarah  | mahmoud |
| safwat | hegazy  |
+--------+---------+

4 rows in set (0.00 sec)
```

كما ترى ستلاحظ الفرق بدون شك لو أنك قمت بوضع جملة الشرط على الصورة التالية

```
where f_name = 'sherif';
```

أو

```
where f_name = 'sarah';
```

هنا ظهرت قوة like بشكل واضح أليس كذلك ؟ كذلك تستطيع عمل العكس بمعنى أنك تريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) ولكن هذه المرة بشرط أن يكون الاسم الأول منتهى بحرف d مثلاً فيكون ذلك كالتالي :

```
mysql> select f_name, l_name from members_data
    -> where f_name like '%d';

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| ahmad    | youssef |
| muhammad | ismael  |
| muhammad | mahfouz |
| muhammad | wadood  |
| muhammad | antary  |
| ahmad    | antar   |
| muhammad | kamal   |
| muhammad | taha    |
+----------+---------+

8 rows in set (0.00 sec)
```

كذلك تستطيع استخدام % لجلب الأسماء والتي تحتوى على حرف معين تريده كالتالي :

```
mysql> select f_name, l_name from members_data
    -> where f_name like '%h%';

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| ahmad    | youssef |
| muhammad | ismael  |
| sherif   | shahin  |
| sherif   | faroo2  |
| muhammad | mahfouz |
| sarah    | mahmoud |
| muhammad | wadood  |
| muhammad | antary  |
| ahmad    | antar   |
```

```
| muhammad | kamal    |
| muhammad | taha     |
+----------+---------+

11 rows in set (0.00 sec)
```

طبعاً تستطيع اللعب مع like مثلاً بأن تقول أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والأخير) وأعمارهم بشرط أن يكون عمر العضو محتويا على رقم 3 فكيف تفعل ذلك ؟

```
mysql> select f_name, l_name, age from members_data
    -> where age like '%3%';

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| ahmad    | youssef |   32 |
| muhammad | ismael  |   30 |
| sherif   | shahin  |   38 |
| sarah    | mahmoud |   39 |
| safwat   | hegazy  |   36 |
| muhammad | kamal   |   53 |
+----------+---------+------+

6 rows in set (0.00 sec)
```

العلامات المنطقية أو Logical Operators

في هذا الجزء إن شاء الله سنلقى الضوء على كيفية استخدام العلامات المنطقية أو Logical Operators داخل جملة أمر select ، والعلامات المنطقية تُمكننا من تطبيق أكثر من شرط داخل الجملة الواحدة وهى ثلاث علامات :

```
1- and
2- or
3- not
```

كأن تقول مثلاً أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم بشرط أن يكون الاسم الأول يبدأ بحرف " m " وفى نفس الوقت عمره يكون أكبر من 32 عاما .

تريد تطبيق ذلك تخيل معى شكل الجملة ..... :

```
mysql> select f_name, l_name, age from members_data
    -> where f_name like 'm%' and age > 20;

+----------+---------+------+
```

```
| f_name   | l_name   | age  |
+----------+---------+------+
| muhammad | ismael   |   30 |
| muhammad | mahfouz  |   29 |
| marwa    | hassan   |   45 |
| muhammad | wadood   |   42 |
| muhammad | antary   |   24 |
| muhammad | kamal    |   53 |
+----------+---------+------+

6 rows in set (0.00 sec)
```

أو كأن تقول مثلاً أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم بشرط أن يكون العمر أكبر من 20 وأقل من 30 :

```
mysql> select f_name, l_name, age from members_data
    -> where age > 20 and age < 30;

+----------+---------+------+
| f_name   | l_name   | age  |
+----------+---------+------+
| sherif   | faroo2   |   27 |
| muhammad | mahfouz  |   29 |
| muhammad | antary   |   24 |
| ahmad    | antar    |   22 |
+----------+---------+------+

4 rows in set (0.00 sec)
```

من هنا يتضح أن العلامة المنطقية and تعمل فقط في حال توافر الشرطين معاً أي لا بد من وجود كلا الشرطين لكى تكون هناك نتيجة لجملة الاستعلام ، أما العلامة المنطقية or فتستخدم في حال توافر أحد الشرطين ، فمثلا تريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم ولكن بشرط أن ينتهى الاسم الأخير للعضو بحرف " a " أو أن يكون عمره أكبر من 20 عاماً :

```
mysql> select f_name, l_name, age from members_data
    -> where l_name like '%a' or age > 20;

+----------+---------+------+
| f_name   | l_name   | age  |
+----------+---------+------+
| ahmad    | youssef  |   32 |
| muhammad | ismael   |   30 |
| sherif   | shahin   |   38 |
| sherif   | faroo2   |   27 |
| muhammad | mahfouz  |   29 |
| sarah    | mahmoud  |   39 |
| marwa    | hassan   |   45 |
```

```
| muhammad | wadood   |   42 |
| muhammad | antary   |   24 |
| safwat   | hegazy   |   36 |
| ahmad    | antar    |   22 |
| muhammad | kamal    |   53 |
| muhammad | taha     |   18 |
+----------+---------+------+

13 rows in set (0.00 sec)
```

كما تلاحظ في حال توافر أحد الشرطين أو كلاهما ظهرت نتيجة لجملة الاستعلام بالشكل السابق .

والآن لنأخذ مثالاً أكثر صعوبة : أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) كذلك أعمارهم بشرط أن يبدأ الاسم الأخير بحرف " m " أو حرف " a " و تكون أعمارهم أكبر من 25 عاماً ؟

الأمر أصبح معقد أليس كذلك ؟ لا تقلق قم بتحليل السؤال في ذهنك بتريث ثم اكتب جملة الاستعلام بترتيب مطالب السؤال ، في البداية نريد الاستعلام عن الاسم الأول والأخير والعمر وذلك داخل جملة أمر select
والأمر سهل أظن ليس فيه مشكلة ثم بعد ذلك ندخل في الشروط :

أولا : أن يبدأ الاسم الأخير بحرف m أو بحرف a وتسطيع فعل ذلك باستخدام ال Operator وهو like مع ال percentage sign وهى % إلى جانب استخدام العلامة المنطقية or .

ثانيا : في نفس الوقت أن يكون العمر أكبر من 25 عاماً أي أننا سنستخدم هنا Comparison Operator وهى العلامة >.
الآن لنرى كيفية كتابة الجملة من التحليل السابق للسؤال :

```
mysql> select f_name, l_name, age from members_data
    -> where (l_name like 'm%' or 'a%') and age > 25;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| muhammad | mahfouz |   29 |
| sarah    | mahmoud |   39 |
+----------+---------+------+

2 rows in set, 1 warning (0.00 sec)
```

لاحظ أنني قمت بالفصل بين كل من العلامة المنطقية or والعلامة المنطقية and في جملة where وذلك فقط للتوضيح أن هناك علامتان منطقيتان وكسبيل للترتيب ،فمن الممكن استخدام جملة where بدون تلك الأقواس وستعمل معك الجملة بدون أي مشاكل .

أما العلامة الأخيرة not فتستخدم لنفى شيء والإتيان بنقيضه كأن نقول مثلا أريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم بشرط ألا يبدأ الاسم الأخير بحرف m :

```
mysql> select f_name, l_name, age from members_data
    -> where l_name not like 'm%';

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| ahmad    | youssef |   32 |
| muhammad | ismael  |   30 |
| sherif   | shahin  |   38 |
| sherif   | faroo2  |   27 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| muhammad | antary  |   24 |
| safwat   | hegazy  |   36 |
| ahmad    | antar   |   22 |
| kamel    | ahmad   |   17 |
| muhammad | kamal   |   53 |
| muhammad | taha    |   18 |
+----------+---------+------+

12 rows in set (0.00 sec)
```

العلامتان In و Between

ذكرنا في الجزء السابق الخاص بالعلامات المنطقية أو Logical Operators أنها تقوم للربط بين أكثر من شرط سواء كانوا شرطين أو ثلاثة أو أكثر ، فعلى سبيل المثال لو أنك تريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم بشرط أن يكون الاسم الأخير ahmad أو taha فتكون جملة الاستعلام بالشكل التالي :

```
mysql> select f_name, l_name, age from members_data
    -> where l_name ='ahmad' or
    -> l_name ='taha';

+----------+--------+------+
| f_name   | l_name | age  |
+----------+--------+------+
| kamel    | ahmad  |   17 |
| muhammad | taha   |   18 |
+----------+--------+------+

2 rows in set (0.00 sec)
```

فتستطيع عمل ذلك باستخدام IN Operator وحصر الشروط داخل الأقواس بالشكل التالي :

```
mysql> select f_name, l_name, age from members_data
    -> where l_name in ('ahmad' , 'taha');

+----------+--------+------+
| f_name   | l_name | age  |
+----------+--------+------+
| kamel    | ahmad  |   17 |
| muhammad | taha   |   18 |
+----------+--------+------+

2 rows in set (0.00 sec)
```

أظن أن استخدام In في تحديد الشروط أفضل من استخدام or عدة مرات أليس كذلك ؟ فمن الممكن أن تكون جملة الشرط محتوية على أكثر من عنصر للشرط فمثلاً جملة الشرط قد تكون الاستعلام عن أسماء الأعضاء وأعمارهم بشرط أن يكون الاسم الأخير يكون ahmad أو youssef أو mahmoud أو kamal إلخ فتستطيع توفير على نفسك بعض ال Ors ! عموماً الاختيار متروك لك .

كما يمكننا استخدام not مع In للاستعلام عن نقيض ما بداخل الأقواس أو قد نقول مكمل ما بداخل الأقواس ولاحظ معى المثال التالي :

```
mysql> select f_name, l_name, age from members_data
    -> where l_name not in ('ahmad', 'taha');

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| ahmad    | youssef |   32 |
| muhammad | ismael  |   30 |
| sherif   | shahin  |   38 |
| sherif   | faroo2  |   27 |
| muhammad | mahfouz |   29 |
| sarah    | mahmoud |   39 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| muhammad | antary  |   24 |
| safwat   | hegazy  |   36 |
| ahmad    | antar   |   22 |
| muhammad | kamal   |   53 |
+----------+---------+------+

12 rows in set (0.00 sec)
```

أي أننا في المثال السابق قمنا بالاستعلام عن كل الأسماء مع عدا كل من الاسم ahmad والاسم taha .

أما العلامة الثانية وهى between فتستخدم لتحديد فترة داخل الأرقام الصحيحة فمثلاً تريد الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم بشرط أن يكون العمر محصورا من 20 إلى 25 عاماً :

```
mysql> select f_name, l_name, age from members_data
    -> where age between 20 and 25;

+----------+--------+------+
| f_name   | l_name | age  |
+----------+--------+------+
| muhammad | antary |   24 |
| ahmad    | antar  |   22 |
+----------+--------+------+

2 rows in set (0.00 sec)
```

كما أنك تستطيع أيضا استخدام not مع between لعمل وظيفة أخرى وهى الاستعلام عن أسماء الأعضاء وأعمارهم بشرط ألا يكون العمر محصوراً ما بين 20 و 25 عاماً :

```
mysql> select f_name, l_name, age from members_data
    -> where age not between 20 and 25;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| ahmad    | youssef |   32 |
| muhammad | ismael  |   30 |
| sherif   | shahin  |   38 |
| sherif   | faroo2  |   27 |
| muhammad | mahfouz |   29 |
| sarah    | mahmoud |   39 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| safwat   | hegazy  |   36 |
| kamel    | ahmad   |   17 |
| muhammad | kamal   |   53 |
| muhammad | taha    |   18 |
+----------+---------+------+

12 rows in set (0.00 sec)
```

كما رأيت فعلا لا توجد الأعمار داخل الفترة التي قمنا بتحديدها .

جملة order by

في معرض كلامنا السابق تناولنا عدة نقاط مهمة خلال تعاملنا اليومي مع نظام إدارة قواعد البيانات MySQL وذكرنا من ضمن تلك النقاط كيفية إنشاء قاعدة بيانات جديدة ، كذلك كيفية إنشاء جداول داخل تلك القاعدة ، وأيضاً كيفية إضافة بيانات جديدة داخل صفوف تلك الجداول ، وبالتالي كان التعامل فقط مع البيانات المدخلة بأي شكل ثم كيفية الاستعلام عن تلك البيانات دون الاهتمام بكيفية ظهورها أو ترتيبها خلال نتيجة عملية الاستعلام .

قد يخمن البعض الآن ما سوف أقوم بشرحه ، أو ما سنقوم بالتركيز عليه خلال تلك الجزئية وهى كيفية ظهور البيانات نفسها وكيفية ترتيب نتيجة جمل الاستعلام المختلفة .

في السابق كنا نستخدم جملة [Where .....Conditions] لوضع مجموعة من الشروط أو القيود على جملة الاستعلام بغرض تطويع الجملة لكى نستعلم عما نريد ، ولكن لم نكن لنهتم كثيراً هل ظهرت البيانات بشكل مرتب أم لم تظهر ، كذلك كيفية ظهور تلك البيانات بوضعية معينة مثلاً تريد أن تستعرض البيانات عن طريق حقل ال age وتريد ترتيب أعمار الأعضاء بشكل تصاعدي أو تنازلي طبقا لعمر العضو داخل الجدول وهكذا .

لذا من السابق نستنتج أن محور الحديث خلال السطور القادمة سيكون عن كيفية ظهور البيانات من خلال جمل الاستعلام المختلفة ولكن سنستخدم جملة أخرى بدلاً من جملة where وستكون هذه المرة الجملة المسماة order by والتي هي مسؤولة عن تلك النقطة أي ترتيب البيانات بشكل معين .

ظهور البيانات في السابق كان معتمداً على مفهوم مهم وهو ما يدخل أولاً يعرض أولاً ، نعم ولنفترض مثلاً أنك قمت بإدخال أسماء عدة أعضاء جدد داخل الجدول الخاص بنا وهو members_data داخل قاعدة البيانات linux_ac فعند عمل جملة استعلام عن الاسم الأول والاسم الأخير داخل الجدول سيظهر لديك ترتيب أسماء الأعضاء ترتيباً زمنياً أي كما ذكرنا سابقاً أن من تمت إضافته أولاً سيظهر في البداية ثم يليه الذي تم إضافته مؤخراً وهكذا ، وبالتالي تلك الجزئية أنت مجبر عليها من قبل نتيجة جملة أمر الاستعلام Select ، ولكن الجديد هنا باستخدام جملة order by سنتمكن من إظهار البيانات بوضعيات وأشكال مختلفة نحن نريدها لسنا مجبرين عليها ، البعض قد سئم الكلام وكثرة الحديث ولذلك سنقوم بتسخين أيدينا ونبدأ بمثال عملي يوضح تلك الجزئية بالشكل التالي :

```
mysql> select f_name, l_name from members_data
    -> order by f_name;

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| ahmad    | youssef |
| ahmad    | antar   |
| kamel    | ahmad   |
```

```
| marwa    | hassan   |
| muhammad | taha     |
| muhammad | kamal    |
| muhammad | antary   |
| muhammad | wadood   |
| muhammad | mahfouz  |
| muhammad | ismael   |
| muhammad | nagib    |
| safwat   | hegazy   |
| sarah    | mahmoud  |
| sherif   | faroo2   |
| sherif   | shahin   |
+----------+----------+

15 rows in set (0.00 sec)
```

ماهي ملاحظاتك لنتيجة جملة أمر select ؟ هلا تلاحظ شيئاً جديداً لم تكن معتاداً على رؤيته ؟ بالطبع نعم !

أولا نبدأ مع شرح جملة أمر select :

في البداية قمنا بالاستعلام عن الاسم الأول والاسم الأخير للأعضاء ولكن هذه المرة قمنا باستخدام جملة جديدة هي order by بدلاً من جملة where ، ولكن وضعنا شرط لعملية الترتيب أن تكون عن طريق العمود الخاص بالاسم الأول للأعضاء وفى نفس الوقت بترتيب تصاعدي للحروف الإنجليزية أي أن يكون ترتيب الأسماء بحسب ترتيب الحروف باللغة الإنجليزية وبشكل أبسط من ذلك أن يكون العمود بادئاً بالأسماء التي تبدأ بحرف a ثم الأسماء التي تبدأ بحرف b وهكذا حتى نهاية العمود وذلك بالأسماء التي تبدأ بحرف z .

وبالتالي يتضح من ذلك أن الوضع الافتراضي عند استخدام جملة order by اعتماداً على اسم عمود يحتوى على نصوص سيكون الترتيب الظاهر لدينا هو ترتيب حرفي بحسب وضعية الحروف وترتيبها داخل نطاق اللغة التي تشمل تلك الحروف ، ففي مثالنا السابق قمنا بوضع شرط لجملة order by أن يكون الترتيب الظاهر لدينا في الجدول بحسب الاسم الأول فظهرت الأسماء التي تبدأ ب a أولاً ثم التي تليها ثم التي تليها ..... وهكذا

أي نستطيع استنتاج قاعدة مهمة :

ترتيب الحقول أو الأعمدة عند وضعها كشرط بعد جملة order by سيكون ترتيب تصاعدي سواء كانت تلك الحقول تشمل متغيرات نصية أو (Strings) أو متغيرات عددية صحيحة أو (Integers) ، ففي حالة الأعداد والأرقام سيكون ظهور الرقم من أقل إلى أكبر ، وفى حالة ظهور الحروف سيكون بموقعها الترتيبي داخل حروف اللغة نفسها ، فعلى سبيل المثال حروف اللغة الإنجليزية ستبدأ من a ثم b ثم c ..... انتهاءً ب z .

مثال : نريد الاستعلام عن الاسم الأول والاسم الأخير لأعضاء قاعدة البيانات linux_ac داخل الجدول members_data بحسب ترتيب الاسم الأخير ؟

```
mysql> select f_name, l_name from members_data
    -> order by l_name;

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| kamel    | ahmad   |
| ahmad    | antar   |
| muhammad | antary  |
| sherif   | faroo2  |
| marwa    | hassan  |
| safwat   | hegazy  |
| muhammad | ismael  |
| muhammad | kamal   |
| muhammad | mahfouz |
| sarah    | mahmoud |
| muhammad | nagib   |
| sherif   | shahin  |
| muhammad | taha    |
| muhammad | wadood  |
| ahmad    | youssef |
+----------+---------+

15 rows in set (0.00 sec)
```

لاحظ في المثال السابق أن العمود المسمى l_name هو الذي ظهر بالترتيب الحرفي وذلك لأننا أردنا الاستعلام عن أسماء الأعضاء ولكن بحسب ترتيب الاسم الأخير ومن ثم لمزيد من السهولة في كيفية اكتشاف ذلك قم بالتبديل بين اسم العمود l_name و f_name داخل جملة أمر select لكى يسهل عليك فهم ذلك كأن يكون بالصورة التالية :

```
mysql> select l_name, f_name from members_data
    -> order by l_name;

+---------+----------+
| l_name  | f_name   |
+---------+----------+
| ahmad   | kamel    |
| antar   | ahmad    |
| antary  | muhammad |
| faroo2  | sherif   |
| hassan  | marwa    |
| hegazy  | safwat   |
| ismael  | muhammad |
| kamal   | muhammad |
| mahfouz | muhammad |
| mahmoud | sarah    |
| nagib   | muhammad |
| shahin  | sherif   |
```

```
| taha     | muhammad |
| wadood   | muhammad |
| youssef | ahmad    |
+---------+----------+

15 rows in set (0.00 sec)
```

كل ما فعلناه أننا بدلنا عمود مكان الآخر لكى يسهل علينا فهم ما فعلناه سابقاً ليس إلا .

وبالتالي تستطيع الاستعلام عن البيانات التي تُريدها وكيفية ظهورها تِبعاً لعمود أو حقل معين ، كأن نقول أريد الاستعلام عن الاسم الأول والاسم الأخير والعمر للأعضاء بحسب ترتيب أعمارهم :

```
mysql> select f_name, l_name, age
    -> from members_data order by age;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| muhammad | nagib   |   16 |
| kamel    | ahmad   |   17 |
| muhammad | taha    |   18 |
| ahmad    | antar   |   22 |
| muhammad | antary  |   24 |
| sherif   | faroo2  |   27 |
| muhammad | mahfouz |   29 |
| muhammad | ismael  |   30 |
| ahmad    | youssef |   32 |
| safwat   | hegazy  |   36 |
| sherif   | shahin  |   38 |
| sarah    | mahmoud |   39 |
| muhammad | wadood  |   42 |
| marwa    | hassan  |   45 |
| muhammad | kamal   |   53 |
+----------+---------+------+

15 rows in set (0.00 sec)
```

هنا تم ظهور العمر بشكل تصاعدي كما ذكرنا من قبل أي العمر الأقل ، ثم الأكبر فالأكبر وهكذا دواليك .

لكن قد أرى في عيون البعض سؤالاً يفرض نفسه علينا ماذا لو أردنا قلب الموضوع رأساً على عقب ؟ بمعنى أريد مثلاً الاستعلام عن أسماء الأعضاء (الاسم الأول والاسم الأخير) وأعمارهم أيضاً ولكن بحسب ترتيب العمر ترتيباً تنازلياً أي الأكبر ثم الأقل فالأقل فكيف لنا فعل ذلك ؟

سؤال منطقي وقد يخطر لدى البعض ، وهنا يأتي دور الخيار descending والتي يدخل ضمن نطاق جملة order by

ولايوجد أسرع من الأمثلة لكى تفهم ذلك سريعاً فقد يغنى مثالاً واحداً تقوم بتطبيقه عن عدة صفحات نقوم بسردها ، ونفذ معى التالي من أمام محث أوامر mysql :

```
mysql> select f_name, l_name, age
    -> from members_data order by age DESC;

+----------+---------+------+
| f_name   | l_name  | age  |
+----------+---------+------+
| muhammad | kamal   |   53 |
| marwa    | hassan  |   45 |
| muhammad | wadood  |   42 |
| sarah    | mahmoud |   39 |
| sherif   | shahin  |   38 |
| safwat   | hegazy  |   36 |
| ahmad    | youssef |   32 |
| muhammad | ismael  |   30 |
| muhammad | mahfouz |   29 |
| sherif   | faroo2  |   27 |
| muhammad | antary  |   24 |
| ahmad    | antar   |   22 |
| muhammad | taha    |   18 |
| kamel    | ahmad   |   17 |
| muhammad | nagib   |   16 |
+----------+---------+------+

15 rows in set (0.00 sec)
```

لاحظ في نهاية الجملة قمت بوضع الخيار DESC بالحروف الكبيرة لمجرد أن تلاحظ ما قمنا بإضافته بعد اسم العمود المراد ظهور البيانات اعتماداً عليه ، وبالتالي يُمكنك استخدام خاصية DESC مع أي عمود تريد ظهور البيانات فيه بشكل تنازلي سواء كان ذلك للأرقام العددية أو للحروف وطبيعي في حالة الحروف سيكون ظهور أسماء الأعضاء بداية من الحرف z وانتهاءً بالحرف a ولنلمس ذلك عن قرب كما يلي :

```
mysql> select f_name, l_name from members_data
    -> order by f_name DESC;

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| sherif   | shahin  |
| sherif   | faroo2  |
| sarah    | mahmoud |
| safwat   | hegazy  |
| muhammad | nagib   |
| muhammad | taha    |
| muhammad | kamal   |
```

```
| muhammad | antary  |
| muhammad | wadood  |
| muhammad | mahfouz |
| muhammad | ismael  |
| marwa    | hassan  |
| kamel    | ahmad   |
| ahmad    | antar   |
| ahmad    | youssef |
+----------+---------+

15 rows in set (0.00 sec)
```

ملحوظة : كما قلنا سابقاً أن الترتيب التصاعدي هو الوضع الافتراضي بالنسبة لجملة order by ويمكنك أن تستخدمه أيضا داخل نطاق الجملة وذلك بوضع asc بالشكل التالي :

```
order by f_name ASC;
```

بعد أن استطعنا من خلال جملة order by عرض وترتيب شكل البيانات بالطريقة التي تعجبنا ننتقل إلى جزئية أخرى وهى كيفية تحديد عدد البيانات التي تظهر نتيجة لجملة أمر select كأن تقول مثلاً أريد عرض أقل أربعة أعمار باستخدام حقل العمر داخل نطاق جملة order by أو أن تقول أريد عرض أول خمس نتائج داخل الجدول وهكذا ، والجملة التي تمكننا من فعل ذلك هي جملة limit ويتبعها مباشرة الرقم الذي تريد تحديد فيه عدد النتائج .

مثالاً على ذلك :

تريد الاستعلام عن أول خمسة أسماء للأعضاء (الاسم الأول والاسم الأخير)من خلال الجدول members_data:

```
mysql> select f_name, l_name
    -> from members_data limit 5;

+----------+---------+
| f_name   | l_name  |
+----------+---------+
| ahmad    | youssef |
| muhammad | ismael  |
| sherif   | shahin  |
| sherif   | faroo2  |
| muhammad | mahfouz |
+----------+---------+

5 rows in set (0.00 sec)
```

ستلاحظ فعلاً أن أول خمسة أسماء من أسماء الأعضاء هي التي ظهرت ، وتستطيع تغيير الرقم 5 إلى أي رقم تريده ،

لكن تظهر قوة limit بصورة أكبر من المثال السابق عند دمج limit مع جمل order by ، فالمثال التالي نريد الاستعلام فيه عن أصغر أربعة أعضاء في العمر من خلال الجدول وتكون صيغة جملة أمر select كما يلي :

```
mysql> select f_name, l_name, age
    -> from members_data order by age limit 4;

+----------+--------+------+
| f_name   | l_name | age  |
+----------+--------+------+
| muhammad | nagib  |   16 |
| kamel    | ahmad  |   17 |
| muhammad | taha   |   18 |
| ahmad    | antar  |   22 |
+----------+--------+------+

4 rows in set (0.00 sec)
```

كما ترى أن لجملة limit خاصية مفيدة في حال دمجها مع جملة order by لتكوين جمل تؤدى مهام عديدة وتستطيع اللعب مع معها !

لجملة limit خاصية مهمة جداً وهى تحديد السطر الذي تريد استخراج منه مجموعة فرعية من البيانات ، فمثلا تريد الاستعلام عن بعض بيانات الأعضاء ولكن تريد تحديد بداية الاستعلام كأن تقول لجملة الاستعلام باستخدام جملة limit ابدأ من السطر السادس واستعلم عن أربع نتائج ، فكيف يمكننا فعل ذلك ؟ :
لاحظ معى في البداية سوف أقوم باستعراض كافة البيانات لتوضيح تلك الخاصية المهمة بشكل واضح كالتالي :

```
mysql> select * from members_data;

+--------+----------+---------+------+----------------------+
| mem_id | f_name   | l_name  | age  | email                |
+--------+----------+---------+------+----------------------+
|      1 | ahmad    | youssef |   32 | a_youssef@bignet.com |
|      2 | muhammad | ismael  |   30 | m_ismael@bignet.com  |
|      3 | sherif   | shahin  |   38 | s_shahin@bignet.com  |
|      4 | sherif   | faroo2  |   27 | s_faroo2@bignet.com  |
|      5 | muhammad | mahfouz |   29 | m_mahfouz@bignet.com |
|      6 | sarah    | mahmoud |   39 | s_mahmoud@bignet.com |
|      7 | marwa    | hassan  |   45 | m_hassan@bignet.com  |
|      8 | muhammad | wadood  |   42 | m_wadood@bignet.com  |
|      9 | muhammad | antary  |   24 | m_antary@bignet.com  |
|     10 | safwat   | hegazy  |   36 | s_hegazy@bignet.com  |
|     11 | ahmad    | antar   |   22 | a_antar@bignet.com   |
|     12 | kamel    | ahmad   |   17 | k_ahmad@bignet.com   |
|     13 | muhammad | kamal   |   53 | m_kamal@bignet.com   |
|     14 | muhammad | taha    |   18 | m_taha@bignet.com    |
|     15 | muhammad | nagib   |   16 | m_nagib              |
+--------+----------+---------+------+----------------------+
```

```
15 rows in set (0.00 sec)
```

بعد ذلك سأقوم الآن بتحديد لجملة الاستعلام كي تبدأ من السطر السادس بعدد أربع نتائج لجملة limit ويكون شكل الجملة كما يلي :

```
mysql> select mem_id, f_name, l_name
    -> from members_data limit 6,4;

+--------+----------+--------+
| mem_id | f_name   | l_name |
+--------+----------+--------+
|      7 | marwa    | hassan |
|      8 | muhammad | wadood |
|      9 | muhammad | antary |
|     10 | safwat   | hegazy |
+--------+----------+--------+

4 rows in set (0.00 sec)
```

حيث الرقم 6 هو الرقم الذي سوف يبدأ بعده مباشرة للاستعلام عن النتائج ثم بعد ذلك الرقم 4 وهو الرقم الذي يحدد عدد النتائج المستعلم عنها .

ملحوظة مهمة : الرقم 6 هو عدد الصفوف التي سيتم إهمالها لتبدأ جملة الاستعلام في العمل ، ففي المثال السابق تم إهمال أول ستة صفوف وبدأت جملة الاستعلام في جلب النتائج بداية من الصف رقم 7 وهكذا إذا أردت الاستعلام عن بيانات بداية من الرقم 4 كمثال سيتم إهمال أول أربعة نتائج وتبدأ جملة الاستعلام في جلب النتائج بداية من الصف رقم 5 .

النتائج السابقة التي كانت تظهر لنتيجة جمل أمر ال select المختلفة كان تشمل احتمالية حدوث عدة بيانات مختلفة في نفس الحقل كأن يحدث مثلاً أن تظهر عدة أسماء في العمود f_name بنفس الاسم وليكن على سبيل المثال ، ولكن في بعض الأحيان لا نريد أن تظهر تلك النتائج المتشابهة ونريد ظهور البيانات دون تكرار بيان في حقل معين فكيف لنا ذلك ؟
هنا يأتي دور الأداة distinct والتي تعنى منع ظهور بيانات متشابهة داخل العمود الواحد أو الحقل الواحد ، وتأتى الأداة distinct قبل اسم العمود المراد عدم تكرار بيانات متشابهة فيه ونأخذ مثالاً على ذلك :
نريد الاستعلام عن الأسماء داخل العمود الأول f_name بشرط ألا تظهر أسماء مكررة في نتائج الجملة :

```
mysql> select  f_name from members_data;

+----------+
```

```
| f_name   |
+----------+
| ahmad    |
| muhammad |
| sherif   |
| sherif   |
| muhammad |
| sarah    |
| marwa    |
| muhammad |
| muhammad |
| safwat   |
| ahmad    |
| kamel    |
| muhammad |
| muhammad |
| muhammad |
+----------+

15 rows in set (0.00 sec)
```

طبعا قمنا أولاً بعرض الأسماء داخل العمود لكى نتأكد من وجود أسماء مكررة لكى تلاحظ الفارق، الآن لاحظ التالي :

```
mysql> select distinct f_name from members_data;

+----------+
| f_name   |
+----------+
| ahmad    |
| muhammad |
| sherif   |
| sarah    |
| marwa    |
| safwat   |
| kamel    |
+----------+

7 rows in set (0.00 sec)
```

ومثال آخر إذا أردت الاستعلام عن العمود الخاص بأعمار الأعضاء وفى نفس الوقت عدم تكرار أعمار متشابهة خلال نتائج جملة أمر select سيكون ذلك بالشكل التالي :

```
mysql> select distinct age from
    -> members_data order by age;

+------+
```

```
| age  |
+------+
|   16 |
|   17 |
|   18 |
|   24 |
|   29 |
|   30 |
|   32 |
|   36 |
|   38 |
|   39 |
|   42 |
|   45 |
|   53 |
+------+

15 rows in set (0.00 sec)
```

هنا أيضا استخدمنا جملة order by لكى تظهر البيانات المستعلم عنها بشكل بترتيب تصاعدي .

استخدامات متقدمة لنظام إدارة قواعد البيانات MySQL

في السابق كنا نتحدث عن بعض الأمور الأساسية والتي تشكل قاعدة أساسية للتعامل مع خادم MySQL بشكل قد يكون يومي لدى البعض ، وتطرقنا إلى عدة نقاط مهمة وقمنا بتطبيق عدة أمثلة على تلك النقاط وأعتقد أننا قد وفينا بشكل أو بآخر تلك النقاط أو على الأقل قمنا بذكر أمثلة مهمة توضح وظائف تلك النقاط .

ومن الآن فصاعداً سنبدأ في ذكر بعض النقاط المهمة الأخرى والتي تميل إلى الاستخدامات المتقدمة نوعاً ما ، وبالتالي ستشعر بقوة ومتانة MySQL في أداء وظائف قد لا يتخيلها أو يُدركها البعض ومن ضمن هذه النقاط :

* دوال المجموع أو Aggregate Functions
* استخدام جملة HAVING
* استخدامات أخرى لجملة أمر Select
* الدوال الحسابية في MySQL أو Mathematical Functions
* تحديث السجلات (Records) باستخدام جملة أمر Update

النقاط الخمس السابقة تحتوى تفاصيل ونقاط فرعية سنقوم بشرحها لاحقاً ، ولكن في البداية سنغير من خطة العمل التي كنا نعمل عليها بمعنى توجد لدينا الآن شركة ما (نفترض أن اسمها linuxsoft) هذه الشركة متخصصة في عمل تطبيقات مختلفة تخص نظام التشغيل جنو/لينوكس ، وتريد الشركة عمل قاعدة بيانات تضم تفاصيل الموظفين لديها من أسماء الموظفين ، ورواتبهم ، وعدد سنين العمل في الشركة لكل موظف ، كذلك الألقاب الوظيفية لكل موظف

بالإضافة إلى أعمار هؤلاء الموظفين ، علاوة على ذلك عناوين البريد الإلكتروني الخاصة بالموظفين .

طبعا بدأنا الآن في عمل حقيقي أي أننا سنتمكن من عمل تطبيق حقيقي يستخدم فعلياً داخل الشركات التي تحوى عدد من الموظفين ، ونبدأ مباشرة في تحليل التطبيق السابق لنعرف ماهو المطلوب عمله لكى نبنى قاعدة البيانات تلك :

أولاً : ـفي البداية سنقوم بإنشاء قاعدة بيانات تخص الموظفين ولذلك سنقوم بتسمية قاعدة البيانات تلك باسم employees للدلالة على أن هذه القاعدة تخص موظفين .

ثانيا : سنقوم بإنشاء جدول داخل قاعدة البيانات employees ونقوم بتسميته employee_data أو employee_detail اختر ما يناسبك .

ملحوظة : اختيار اسم للجدول من الاسمين السابقين ليس فرض اختر أي اسم يحلو لك .

ثالثا : نبدأ في تحليل الأعمدة والحقول المطلوب عملها وإنجازها ، سنحتاج إلى التالي :

* عمود أو حقل لبيان الترتيب الرقمي للموظفين داخل الشركة ونسميه مثلا emp_id أو اختصاراً ل employee identifier .
* عمود أو حقل لتعريف الاسم الأول للموظفين ونقوم بتسميته على سبيل المثال f_name أي first name .
* عمود أو حقل لتعريف الاسم الأخير للموظفين ونقوم بتسميته ب l_name أي last name .
* عمود أو حقل لتعريف اللقب الوظيفي للموظفين وليكن مثلاً title .
* عمود أو حقل لتعريف العمر الخاص بكل موظف داخل الشركة ويكون اسم العمود age .
* عمود أو حقل لتعريف عدد سنين العمل لكل موظف داخل الشركة وليكن اسم العمود yos أو اختصاراً ل years of service .
* عمود أو حقل لتعريف رواتب الموظفين داخل الشركة وليكن اسم العمود salary .
* عمود أو حقل لتعريف حوافز الموظفين ونقوم بتسمية العمود باسم perks .
•عمود أو حقل لتعريف عناوين البريد الإلكتروني للموظفين تحت اسم email .

من التحليل السابق لقاعدة البيانات التي سوف نقوم بإنشائها أننا سوف نحتاج إلى جدول تحت اسم employee_data بالإضافة إلى تسعة أعمدة هي على الترتيب التالي :

```
1- emp_id   ( data type = integer )
2- f_name   ( data type = varchar(25) )
3- l_name   ( data type = varchar(25) )
4- title    ( data type = varchar(50) )
5- age      ( data type = integer )
6- yos      ( data type  = integer )
7- salary   ( data type = integer )
```

```
8- perks    ( data type = integer )
9- email    ( data type = varchar(60) )
```

كما تلاحظ كتبنا الأعمدة المطلوبة والتي ينبغي أن تكون مدرجة داخل الجدول employee_data كما أننا أيضا قمنا بتعريف نوع البيانات التي سيتم إدخالها في كل عمود سواء كانت تلك البيانات متغيرات نصية (strings) أو متغيرات عددية صحيحة (integers) ، سنقوم الآن بالشروع في إنشاء قاعدة البيانات الجديدة والتي قلنا أنها سوف تحمل الاسم employees كما يلي :

```
mysql> create database employees;
Query OK, 1 row affected (0.02 sec)
```

ثم نتأكد من إنشاء القاعدة باستخدام الأمر show databases كما يلي :

```
mysql> show databases;

+--------------------+
| Database           |
+--------------------+
| information_schema |
| employees          |
| linux_ac           |
| mysql              |
| ser                |
+--------------------+

5 rows in set (0.00 sec)
```

فعلا تم إنشاء القاعدة ، الخطوة التالية نقوم بإنشاء الجدول من خلال الأمر create table والذي سيكون باسم employee_data ، ثم نقوم بإضافة الأعمدة السابقة مع أنواع البيانات التي سوف تخص كل عمود على حده بالشكل التالي :

```
mysql> CREATE TABLE employee_data
    -> (
    -> emp_id int unsigned not null auto_increment primary key,
    -> f_name varchar(20),
    -> l_name varchar(20),
    -> title varchar(30),
    -> age int,
    -> yos int,
    -> salary int,
    -> perks int,
```

```
    -> email varchar(60)
    -> );
Query OK, 0 rows affected (0.13 sec)
```

بعد أن قمنا بتنفيذ جملة أمر create table سنتأكد الآن من إضافة الأعمدة بشكل صحيح ، كذلك نوع البيانات التي تخص كل عمود هل هي فعلاً ما نريده أم لا باستخدام الأمر describe :

```
mysql> describe employee_data;

+--------+------------------+------+-----+---------+----------------+
| Field  | Type             | Null | Key | Default | Extra          |
+--------+------------------+------+-----+---------+----------------+
| emp_id | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| f_name | varchar(20)      | YES  |     | NULL    |                |
| l_name | varchar(20)      | YES  |     | NULL    |                |
| title  | varchar(30)      | YES  |     | NULL    |                |
| age    | int(11)          | YES  |     | NULL    |                |
| yos    | int(11)          | YES  |     | NULL    |                |
| salary | int(11)          | YES  |     | NULL    |                |
| perks  | int(11)          | YES  |     | NULL    |                |
| email  | varchar(60)      | YES  |     | NULL    |                |
+--------+------------------+------+-----+---------+----------------+

9 rows in set (0.00 sec)
```

الآن تأكدنا من أن كل شيء يسير على ما يرام ، سنقوم الآن بإدخال بيانات الموظفين إلى الجدول empolyee_data باستخدام جملة أمر insert into ولكن عملية الإدخال ستتم من خلال استيراد ملف نصي جاهز يحتوى على جمل أمر insert into بكافة البيانات المطلوبة وتسطيع تحميل الملف من على الرابط التالي :

```
http://muhammad.akl.googlepages.com/employee.dat
```

بعد الانتهاء من تحميل الملف قم بالولوج إلى المجلد الذي يحتوى ذلك الملف ألا وهو ملف employee.dat ثم قم بتنفيذ الأمر التالي في الطرفية :

```
debian:~# mysql employees < employee.dat -u root -p
Enter password:
```

بعد ذلك سيظهر مؤشر إدخال كلمة المرور التي تخص المستخدم root لخادم MySQL قم بإدخالها ، وإذا لم تظهر لديك أية رسائل تفيد بأن هناك خطأ ما فكل شيء أصبح الآن جاهزاً ولنبدأ في العمل .

وكنا قد تحدثنا عن بعض النقاط التي سنتناولها في معرض حديثنا ونبدأ مباشرة مع أول نقطة :

دوال المجموع أو Aggregate Functions

توفر MySQL مجموعة من الدوال الداخلية والتي تُمكننا من تلخيص بيانات الجدول بدون الاستعلام عن كل حقل على حده ، بمعنى تستطيع مثلاً الاستعلام عن أكبر قيمة و أقل قيمة داخل عمود الرواتب لكى تعرف من هو صاحب أكبر وأقل راتب ، كذلك تستطيع الاستعلام عن متوسط قيمة معينة داخل عمود إلخ .
هذه الدوال هي :

```
MySQL provides 5 aggregate functions. They are:
1). MIN($column_name): Minimum value
2). MAX($column_name): Maximum value
3). SUM($column_name): The sum of values
4). AVG($column_name): The average values
5). COUNT(): Counts the number of entries.
```

توفر الدالة min() إيجاد أقل قيمة داخل عمود معين ، والدالة max() تقوم بإيجاد أكبر قيمة ، أما الدالة sum() فتقوم بجمع عدة قيم والدالة avg() تقوم بإيجاد متوسط عدة قيم وأخيرا الدالة count() تقوم بحساب عدد مجموعة قيم من البيانات بناءاً على اسم العمود التي يتم العد منه ، قد يكون الأمر مُبهما في البداية ولكن عندما نتطرق إلى الأمثلة سيصلك المعنى الذي قمنا بشرحه لوظيفة كل دالة على حده .

الدالة min() و max()

مثال : نريد الاستعلام عن أقل قيمة لراتب موظف داخل الشركة ؟ كيف يكون شكل جملة أمر select باستخدام أين من الدوال السابقة ؟

الإجابة : المثال يريد الاستعلام عن أقل قيمة في الجدول وبالتالي سوف نقوم باستخدام الدالة min() والحقل المراد الاستعلام عنه هو حقل الرواتب أي salary وبالتالي نستطيع كتابة الجملة بالشكل التالي :

```
mysql> select min(salary)
    -> from employee_data;

+-------------+
| min(salary) |
+-------------+
|       70000 |
+-------------+
```

```
1 row in set (0.00 sec)
```

أي أن ما بداخل أقواس الدالة ()min هو اسم الحقل الذي تريد الاستعلام عن أقل قيمة فيه ، كذلك الأمر مع الدالة () max مع نفس المثال السابق ولكن سيكون الاستعلام عن قيمة أكبر راتب :

```
mysql> select max(salary)
    -> from employee_data;

+-------------+
| max(salary) |
+-------------+
|      120000 |
+-------------+

1 row in set (0.01 sec)
```

إذا نستطيع القول :

أنّ كلاً من الدالتين ()min و ()max يتم استخدامهما مع الأعمدة التي تحوى بيانات رقمية سواء كانت تلك البيانات رقمية صحيحة أو كسور عشرية .

ملحوظة : من الممكن أن تقوم بعمل أمثلة على العمود الخاص بالعمر وهو age كذلك العمود الخاص بالحوافز وهو perks .

الدالة sum() و avg()

تختلف كلاً من وظيفة الدالة ()sum والدالة ()avg عن بعضهما البعض ، فالدالة ()sum تقوم بحساب مجموعة من القيم داخل عمود معين ، بينما تقوم الدالة ()avg بحساب متوسط مجموعة قيم داخل عمود معين . ونبدأ مع الأمثلة على كلا الدالتين كما يلي :

مثال : نريد الاستعلام عن مجموع قيم الرواتب كلها ، كذلك متوسط قيمة تلك الرواتب ؟

```
mysql> select sum(salary)
    -> from employee_data;

+-------------+
| sum(salary) |
+-------------+
|     1797000 |
```

```
+-------------+

1 row in set (0.00 sec)
```

كما أن من الممكن استخدام الدالة sum() لإجراء بعض العمليات الحسابية ، كأن تقول مثلا أريد حساب مجموع الرواتب والحوافز الخاصة بالموظفين داخل الشركة ؟

```
mysql> select sum(salary) + sum(perks)
    -> from employee_data;

+--------------------------+
| sum(salary) + sum(perks) |
+--------------------------+
|                  2137000 |
+--------------------------+

1 row in set (0.03 sec)
```

دالة لذيذة بالفعل أليس كذلك ؟ ! تستطيع الآن اللعب مع الدالة باستخدام العمليات الحسابية المختلفة كالضرب والقسمة والطرح باستخدام هذه العلامات :

- العلامة / تشير إلى عملية القسمة .
- العلامة * تشير إلى عملية الضرب .
- العلامة - تشير إلى عملية الطرح .

ملحوظة : تستطيع تعقيد الأمور أكثر من ذلك كأن تقول مثلا أريد الاستعلام عن النسبة المئوية لناتج قسمة كلاً من مجموع قيم العمود perks إلى مجموع قيم العمود salary فكيف نقوم بذلك ؟

لاحظ معي الجملة التالية :

```
mysql> select (sum(perks) / sum(salary) * 100 )
    -> from employee_data;

+----------------------------------+
| (sum(perks) / sum(salary) * 100 ) |
+----------------------------------+
|                          18.9204 |
+----------------------------------+

1 row in set (0.00 sec)
```

أما لحساب متوسط قيمة الراتب للموظفين فتكون الجملة بالشكل كالتالي :

```
mysql> select avg(salary)
    -> from employee_data;

+-------------+
| avg(salary) |
+-------------+
|  89850.0000 |
+-------------+

1 row in set (0.00 sec)
```

كما ترى قامت الدالة ()avg بحساب متوسط قيمة الرواتب الخاصة بالموظفين بالفعل .

وقبل أن ننتقل إلى الدالة الأخيرة وهى ()count سنتعرض إلى جزئية مفيدة وهى تسمية الأعمدة أو الحقول أثناء إجراء عملية الاستعلام ! البعض قد يندهش ماذا تقصد بتسمية الأعمدة إثناء إجراء عملية الاستعلام ؟؟
الأعمدة بالطبع لها أسماء تدل على ما تحتويه من بيانات ، فمثلاً العمود الذي يحتوى على الاسم الأول للموظفين قمنا بتسميته f_name أو اختصاراً ل first name وكذلك الاسم الأخير والعمر إلى آخر أسماء الأعمدة ، أي أن اسم العمود مرتبط بشكل ما مع محتوياته لذا ماهو مفهوم تسمية العمود أثناء إجراء عملية الاستعلام ؟

لو لاحظت معى مثلاً أثناء إجراء الاستعلام عن الاسم الأول للموظفين ستجد اسم العمود الفعلي موجود في أعلى خانة من خانات القيم المستعلم عنها فعلى سبيل المثال قم بتنفيذ التالي :

```
mysql> select f_name
    -> from employee_data;

+----------+
| f_name   |
+----------+
| Sherif   |
| Muhammad |
| Ahmad    |
    .
    .
    .
    .
    .
| Shahida  |
| Abdullah |
+----------+
```

```
20 rows in set (0.03 sec)
```

كما تلاحظ أول خانة في البيانات المستعلم عنها هي اسم العمود والتي كانت في مثالنا الاسم الأول تحت اسم f_name ، الآن نريد تسميه العمود أثناء إجراء عملية الاستعلام بشكل مؤقت فكيف يتسنى لنا فعل ذلك ؟ كأن نقول مثلاً اجعل اسم العمود first name ، كي يكون واضحاً أن العمود يحتوى على الاسم الأول فربما لا يعرف البعض ما المقصود ب f_name أو إلى ماذا تشير .

هنا يأتي دور الأداة as والتي تعنى " ك " ثم تقوم بوضع الاسم الذي تريد وصف العمود به والمثال التالي يوضح الاستعلام عن بيانات العمود f_name وفى نفس الوقت قمنا بوصف العمود ب " first name " :

```
mysql> select f_name as 'first name'
    -> from employee_data;

+------------+
| first name |
+------------+
| Sherif     |
| Muhammad   |
| Ahmad      |
| Muhammad   |

    .
    .
    .
    .
    .
| Youssef    |
| Shahida    |
| Abdullah   |
+------------+

20 rows in set (0.05 sec)
```

كما تلاحظ فعلاً تم تغيير اسم العمود إلى الوصف الذي اخترناه ، ولو قمت بعمل جملة الاستعلام مرة أخرى على العمود f_name ستجد أنه يحتفظ بنفس الاسم ولم يتغير ، أي أن تغيير اسم العمود أو وصفه لعمل شيء معين باستخدام as يكون تغييراً مؤقتاً ليس إلا .

ولكن تظهر فوائد as بشكل أكثر وضوحا عند استخدامها مع ال aggregate functions ، بمعنى يمكن أن تستخدم as في وصف حقل أو عمود غير موجود أصلا للدلالة على نتيجة الاستعلام عن شيء معين ، فمثلاً تريد الاستعلام عن النسبة المئوية لناتج قسمة كلاً من مجموع قيم العمود perks إلى مجموع قيم العمود salary وتقوم بتسمية خارج

القسمة باسم يدل على ذلك كما يلي :

```
mysql> select (sum(perks) / sum(salary) * 100 )
    -> as " perks's percentage " from employee_data;

+---------------------+
| perks's percentage  |
+---------------------+
|             18.9204 |
+---------------------+

1 row in set, 1 warning (0.00 sec)
```

مثال آخر : نريد الاستعلام عن متوسط قيم رواتب الموظفين ونقوم بتسمية النتيجة الخارجة لجملة الاستعلام ب Average Salary ؟

```
mysql> select avg(salary)
    -> as 'Average Salary'
    -> from employee_data;

+----------------+
| Average Salary |
+----------------+
|     89850.0000 |
+----------------+

1 row in set (0.00 sec)
```

الدالة count()

تقوم الدالة count() بعمل بعض الوظائف الحيوية ، من ضمن تلك المهام هي حساب عدد المدخلات ثم عرض تلك المدخلات التي تخص قيم معينة في جدول ، فعلى سبيل المثال تريد حساب عدد المدخلات داخل الجدول employee_data ؟

```
mysql> select count(*)
    -> from employee_data;

+----------+
| count(*) |
+----------+
|       20 |
+----------+

1 row in set (0.02 sec)
```

ملحوظة : تعلمنا سابقاً أن العلامة * تعني عند استخدامها مع MySQL العبارة " All Data " .

نستطيع استخدام الدالة count() لعمل بعض المهام الإضافية ودمجها مع جملة where لتنفيذ مهام أكثر ، فعلى سبيل المثال نريد الاستعلام ومعرفة عدد الموظفين بالشركة ممن يعملون في مجال في وظيفة "programmer" فكيف نقوم بفعل ذلك ؟

```
mysql> select count(*)
    -> as "number of programmer"
    -> from employee_data
    -> where title = 'programmer';

+----------------------+
| number of programmer |
+----------------------+
|                    4 |
+----------------------+

1 row in set (0.00 sec)
```

لاحظ هنا قمنا بدمج as مع الدالة count() بالإضافة إلى جملة where لأداء وظيفة جيدة .

جملة group by

خلال السطور السابقة من حديثنا عن الدالة count() لم نكن لنحصل على الكثير من الذي يمكن أن تقدمه الدالة ، ولكن قد استنتجنا أن الدالة لديها الكثير في حال دمجها مع أدوات وجمل أخرى كما رأينا سابقاً مع جملة where ، الجديد هنا هو جملة group by والتي تقوم بعمل جروب أو group لقيم متشابهة داخل الجدول ، بمعنى حينما نقول مثلاً أريد الاستعلام عن اللقب الوظيفي لكل الموظفين داخل الجدول وبدون تكرار لألقاب متشابهة كيف يمكن تنفيذ ذلك ؟

```
mysql> select title
    -> from employee_data
    -> group by title;

+---------------------------+
| title                     |
+---------------------------+
| Customer Service Manager  |
| Finance Manager           |
| Marketing Executive       |
| Multimedia Programmer     |
| Programmer                |
```

```
| Senior Marketing Executive |
| Senior Programmer          |
| Senior Web Designer        |
| System Administrator       |
| Web Designer               |
+----------------------------+

10 rows in set (0.00 sec)
```

أي قامت جملة group بتخصيص داخلي لا تراه أنت ووضعت كل مجموعة من القيم المتشابهة تحت مجموعة تحمل اسم اللقب المتشابه ، فمثلا إذا كان في الجدول خمسة موظفين يعملون في مهنة Web Designer فستقوم جملة group by بعمل مجموعة تحمل نفس اسم الوظيفة ويندرج تحت تلك المجموعة الخمسة موظفين ، وبالتالي نستطيع الاستنتاج أن عند دمج group by مع الدالة ()count فسوف نستطيع الاستعلام عن عدد الموظفين الذي يعملون في نفس الوظيفة ، أو عدد الموظفين الذين يمتلكون نفس العمر أو نفس الراتب إلخ .

قد يصيح أحد القراء ! مهلاً إذا ما الفارق بين group by والأداة distinct !! في المثال السابق ؟

سؤال جيد ، جملة group by قد تعمل بالشكل البسيط الذي فعلناه في المثال السابق وتؤدى نفس الوظيفة التي تقوم بها الأداة distinct أي جلب البيانات الغير متكررة ، لكن وظائف الأداة distinct عند إضافتها إلى الدالة ()count ستكون محدودة ولا تفي باحتياجاتنا لذلك جملة group by توفر العديد من الوظائف المفيدة فعلاً والتي سوف تلمسها من خلال الأمثلة القادمة .

مثال : نريد الاستعلام عن اللقب الوظيفي مع بيان عدد الألقاب المتشابهة لكل وظيفة داخل الجدول :

```
mysql> select title, count(*)
    -> from employee_data
    -> group by title;

+----------------------------+----------+
| title                      | count(*) |
+----------------------------+----------+
| Customer Service Manager   |        1 |
| Finance Manager            |        1 |
| Marketing Executive        |        3 |
| Multimedia Programmer      |        3 |
| Programmer                 |        4 |
| Senior Marketing Executive |        1 |
| Senior Programmer          |        2 |
| Senior Web Designer        |        1 |
| System Administrator       |        2 |
| Web Designer               |        2 |
+----------------------------+----------+

10 rows in set (0.00 sec)
```

كما تلاحظ من خلال تنفذينا لجملة الاستعلام السابقة تم عرض الألقاب الوظيفية داخل الجدول ، بالإضافة إلى بيان عدد الموظفين الذين يشغلون تلك الوظائف داخل الجدول .

طيب ماذا لو أردنا الاستعلام عن اللقب الوظيفي وعدد الموظفين الذين يعملون في وظيفة "programmer" فقط ؟

```
mysql> select title ,count(*)
    -> from employee_data
    -> where title = 'programmer'
    -> group by title;

+------------+----------+
| title      | count(*) |
+------------+----------+
| Programmer |        4 |
+------------+----------+

1 row in set (0.00 sec)
```

طبعا تستطيع تعقيد الأمور أكثر من ذلك كأن تقول أريد الاستعلام عن اللقب الوظيفي وعدد الموظفين الذين يشغلون مثلا وظيفة Web Designer مع تسمية الحقل الذي يحوى عدد الموظفين الناتج باسم Number of Web Designers ؟

```
mysql> select title, count(*)
    -> as " Number of Web Designers "
    -> from employee_data
    -> where title = 'Web Designer'
    -> group by title;

+--------------+--------------------------+
| title        | Number of Web Designers  |
+--------------+--------------------------+
| Web Designer |                        2 |
+--------------+--------------------------+

1 row in set, 1 warning (0.00 sec)
```

ونأتي إلى استخدام آخر لجمة where مع جملة group by وهذه المرة مع جملة order by كي نقوم بترتيب مثلاً أعداد الموظفين الذين يشغلون الوظائف السابقة بترتيب تصاعدي :

```
mysql> select title, count(*) as Number
```

```
    -> from employee_data
    -> group by title
    -> order by Number;

+----------------------------+--------+
| title                      | Number |
+----------------------------+--------+
| Senior Marketing Executive |      1 |
| Customer Service Manager   |      1 |
| Finance Manager            |      1 |
| Senior Web Designer        |      1 |
| Senior Programmer          |      2 |
| Web Designer               |      2 |
| System Administrator       |      2 |
| Marketing Executive        |      3 |
| Multimedia Programmer      |      3 |
| Programmer                 |      4 |
+----------------------------+--------+

10 rows in set (0.00 sec)
```

لاحظ : قمنا بعمل اسم مؤقت للحقل الذي سوف يحتوى على أعداد الموظفين الخاصة بكل وظيفة وقمنا بتسمية ذلك الحقل Number ، فقد كان من الممكن تنفيذ الجملة السابقة على النحو التالي :

```
mysql> select title, count(*)
    -> from employee_data
    -> group by title
    -> order by count(*);

+----------------------------+----------+
| title                      | count(*) |
+----------------------------+----------+
| Senior Marketing Executive |        1 |
| Customer Service Manager   |        1 |
| Senior Web Designer        |        1 |
| Finance Manager            |        1 |
| Senior Programmer          |        2 |
| System Administrator       |        2 |
| Web Designer               |        2 |
| Marketing Executive        |        3 |
| Multimedia Programmer      |        3 |
| Programmer                 |        4 |
+----------------------------+----------+

10 rows in set (0.00 sec)
```

كما تلاحظ نفس النتيجة السابقة دون اختلاف .

جملة Having

جملة Having التي سوف تكون محور حديثنا خلال السطور القادمة ستلعب دوراً محورياً في إضافة العديد من المهام المفيدة عند دمجها مع كلٍ من الدالة avg() وجملة group by ويتضح ذلك جلياً عندما تريد الاستعلام مثلاً عن اللقب الوظيفي و متوسط قيمة راتب مجموعة من الموظفين يعملون في نفس الوظيفة فيكون شكل جملة الاستعلام كما يلي :

```
mysql> select title, avg(salary)
    -> from employee_data
    -> group by title;

+---------------------------+-------------+
| title                     | avg(salary) |
+---------------------------+-------------+
| Customer Service Manager  |  70000.0000 |
| Finance Manager           | 120000.0000 |
| Marketing Executive       |  77333.3333 |
| Multimedia Programmer     |  83333.3333 |
| Programmer                |  75000.0000 |
| Senior Marketing Executive| 120000.0000 |
| Senior Programmer         | 115000.0000 |
| Senior Web Designer       | 110000.0000 |
| System Administrator      |  95000.0000 |
| Web Designer              |  87500.0000 |
+---------------------------+-------------+

10 rows in set (0.00 sec)
```

أما إذا كنت تريد الاستعلام مثلاً عن اللقب الوظيفي و متوسط قيمة راتب مجموعة من الموظفين يعملون في نفس الوظيفة مع ترتيب القيم الناتجة ترتيباً تصاعديا ًفيكون شكل جملة الاستعلام كما يلي :

```
mysql> select title, avg(salary) as Average
    -> from employee_data
    -> group by title
    -> order by Average;

+---------------------------+-------------+
| title                     | Average     |
+---------------------------+-------------+
| Customer Service Manager  |  70000.0000 |
| Programmer                |  75000.0000 |
| Marketing Executive       |  77333.3333 |
| Multimedia Programmer     |  83333.3333 |
| Web Designer              |  87500.0000 |
| System Administrator      |  95000.0000 |
| Senior Web Designer       | 110000.0000 |
```

```
| Senior Programmer           | 115000.0000 |
| Finance Manager             | 120000.0000 |
| Senior Marketing Executive | 120000.0000 |
+----------------------------+-------------+

10 rows in set (0.00 sec)
```

إلى الآن لم نستخدم جملة Having ولم نتعرض لها ونبدأ الآن في استعراض ما يمكن أن تقدمه لنا تلك الجملة ونأخذ المثال التالي :

مثال : نريد الاستعلام عن القسم الوظيفي أو اللقب الوظيفي والذي يكون متوسط قيمة راتب العاملين داخل تلك الوظيفة أو القسم الوظيفي يزيد على $ 100,000 فكيف نقوم بذلك ؟ راقب معى شكل الجملة :

```
mysql> select title, avg(salary)
    -> from employee_data
    -> group by title
    -> having avg(salary) > 100000;

+----------------------------+-------------+
| title                      | avg(salary) |
+----------------------------+-------------+
| Finance Manager            | 120000.0000 |
| Senior Marketing Executive | 120000.0000 |
| Senior Programmer          | 115000.0000 |
| Senior Web Designer        | 110000.0000 |
+----------------------------+-------------+

4 rows in set (0.00 sec)
```

طبعا من الملاحظ هنا أننا قمنا باستخدام علامات المقارنة أو Comparison Operators لكى نتمكن من تنفيذ الاستعلام بشكل سليم .

إستخدامات أخرى ل Select

قد يتصور البعض أن جملة أمر select يقتصر دورها على الاستعلامات فقط ولا تستطيع أداء مهام أخرى فهل هذا صحيح ؟

لا بالعكس لم ينته دور Select على ما قمنا بتطبيقه من أمثلة خلال السطور السابقة ، بل لجملة أمر select مهام أخرى تجعلها بحق في طليعة أوامر لغة SQL فجملة أمر select لها مهام أخرى قد لا تتخيلها أو تخطر لك على بال .

وهنا تجدر الإشارة لتوضيح معنى نستدركه الآن ، أنك لو لاحظت ودققت في طبيعة المهام التي تقوم بها Select

ستجدها تشبه دوال الطباعة مثل print أو echo في لغات برمجية أخرى ، فعن طريق select تستطيع عرض البيانات والمتغيرات النصية ، كذلك عرض البيانات الرقمية وإجراء عمليات حسابية مختلفة على تلك البيانات الرقمية .

ولكن ماهي الاستخدامات الأخرى والتي لم تذكرها لنا تقوم بها select ؟

الاستخدامات التي لم نذكرها خلال السطور السابقة والتي حان دور ذكرها الآن سوف نلخصها في صورة أمثلة على Select كما يلي :

مثال 1 : عرض إصدار خادم MySQL الذي نعمل عليه الآن :

```
mysql> select version();

+--------------------------+
| version()                |
+--------------------------+
| 5.0.32-Debian_7etch1-log |
+--------------------------+

1 row in set (0.00 sec)
```

مثال 2 : عرض الوقت والتاريخ الحالي :

```
mysql> select now();

+---------------------+
| now()               |
+---------------------+
| 2008-02-16 18:43:45 |
+---------------------+

1 row in set (0.00 sec)
```

مثال 3 : عرض اليوم الحالي فقط :

```
mysql> select dayofmonth(current_date);

+--------------------------+
| dayofmonth(current_date) |
+--------------------------+
|                       16 |
+--------------------------+

1 row in set (0.00 sec)
```

مثال 4 : عرض الشهر الحالي فقط :

```
mysql> select month(current_date);

+---------------------+
| month(current_date) |
+---------------------+
|                   2 |
+---------------------+

1 row in set (0.00 sec)
```

مثال 5 : عرض السنة الحالية فقط:

```
mysql> select year(current_date);

+--------------------+
| year(current_date) |
+--------------------+
|               2008 |
+--------------------+

1 row in set (0.01 sec)
```

مثال 6 : طباعة نصوص كتابية :

```
mysql> select 'i love MySQL too much ! ';

+--------------------------+
| i love MySQL too much !  |
+--------------------------+
| i love MySQL too much !  |
+--------------------------+

1 row in set (0.00 sec)

mysql> select 'i love mysql too much ! '
    -> as "MySQL's Lover";

+--------------------------+
| MySQL's Lover            |
```

```
+--------------------------+
| i love mysql too much !  |
+--------------------------+

1 row in set (0.00 sec)
```

المهام السابقة أعتقد أنها بسيطة وبعضها طريف ، ولكن يمكن أن تستخدم select بشكل مفيد أكثر . قد يحملق البعض في وجهى يا ترى ماهو وجه الاستفادة الأكبر من تلك المهام السابقة والبسيطة D: مهلا لا داعى للعصبية ! P:

جملة أمر select يمكن أن تستخدمها كآلة حاسبة !! أرجوك لا تنفعل أكثر قد يؤثر ذلك بشكل أو بآخر على الأولاد عندما يطلبون مصروف الجيب الخاص بهم P:

الآن نترك التهريج قليلا ولنقم ببعض العمليات الحسابية بالشكل التالي :

مثال 1 : نريد حاصل ضرب كلاً من الرقمين 5.2 و 7.8 ؟

```
mysql> select 5.2*7.8;

+---------+
| 5.2*7.8 |
+---------+
|   40.56 |
+---------+

1 row in set (0.00 sec)
```

وتسطيع تحسين وصف نتيجة الضرب السابقة باستخدام الأداة as بالشكل التالي :

```
mysql> select 5.2*7.8
    -> as "Result of Multiplication";

+--------------------------+
| Result of Multiplication |
+--------------------------+
|                    40.56 |
+--------------------------+

1 row in set (0.00 sec)
```

مثال 2 : نريد خارج قسمة كلاً من الرقمين 120000000 والرقم 1860 ؟

```
mysql> select 120000000 / 1860
    -> as 'Result of Dividing';
```

```
+--------------------+
| Result of Dividing |
+--------------------+
|         64516.1290 |
+--------------------+

1 row in set (0.24 sec)
```

مثال 3 : تريد ضرب كلاً من الرقمين 140 و 169 ثم تقوم بجمع ناتج عملية الضرب على الرقم 4589 ؟

```
mysql> select ((140*169) + 4589)
    -> as 'Result';

+--------+
| Result |
+--------+
|  28249 |
+--------+

1 row in set (0.00 sec)
```

وهكذا تستطيع تنويع العمليات الحسابية المختلفة التي تريدها ، وبالتالي لا تترد في استخدام MySQL ك Second Calculator ! .

الدوال الحسابية في MySQL أو Mathematical Functions

توفر MySQL مجموعة من الدوال الحسابية والتي تقوم كلاً منها بمهمة مختلفة ، كحساب رقم مرفوع لأُس رقم معين ، أو كحساب الجذر التربيعي لعدد معين إلخ من تلك العمليات ، وتجدر الإشارة هنا أيضاً أن هذه الدوال تستخدم مباشرةً مع جملة أمر Select فكما ذكرنا سابقاً أن Select تشبه في طريقة عملها دوال الطباعة في اللغات البرمجية المختلفة ، عموماً نحاول الآن أن نتعرف على تلك الدوال من خلال الأمثلة التي سوف نقوم بتطبيقها على تلك الدوال.

mod(x,y) Function

سنبدأ حديثنا عن الدوال الحسابية في MySQL مع الدالة mod() أو mod(x,y) Function ولنتعرف أكثر على طبيعة عمل الدالة ، لو لاحظت في الأمثلة السابقة والتي استخدمنا فيها جملة أمر select لأداء العمليات الحسابية المختلفة مثل الضرب والقسمة والطرح والجمع أننا حينما أردنا مثلا قسمة عددين لا يقبل أحدهما القسمة على الآخر كنت ستجد في نتيجة العملية الحسابية كسور عشرية خرجت نتيجة لذلك الأمر وهو عدم قابلية قسمة أحد العددين على الآخر ، لذلك ما سوف تقدمه لنا الدالة mod() هو توضيح باقي ناتج القسمة من العملية ونأخذ مثالاً على ذلك حتى

تتضح الأمور .

مثال : تريد خارج قسمة كلاً من العددين 17 والعدد 4 ؟ ستقوم بفعل ذلك بالشكل التالي :

```
mysql> select 17 / 4;

+--------+
| 17 / 4 |
+--------+
| 4.2500 |
+--------+

1 row in set (0.00 sec)
```

كما ترى خارج عملية القسمة يوجد به كسور عشرية ، والآن نستخدم الدالة mod() للرقمان المراد إجراء عملية القسمة لهما بالشكل التالي :

```
mysql> select mod(17,4);

+-----------+
| mod(17,4) |
+-----------+
|         1 |
+-----------+

1 row in set (0.00 sec)
```

لاحظ هنا أن خارج عملية القسمة من المفترض أن يكون 4 وبالتالي باقي خارج القسمة من العملية هو 1 ، إذاً فائدة الدالة mod() تقوم بإخراج باقي نتيجة عملية القسمة التي تمت .

ملحوظة : الدالة mod() لها متغيران داخل الأقواس الأول المتغير x وهو الرقم الأكبر لعملية القسمة والرقم الثاني y وهو الرقم الأصغر لعملية القسمة وتقوم بالفصل بين المتغيرين باستخدام , أي colon .

ABS(x) Function

تقوم الدالة ABS() أو ABS(x) Function بحساب القيمة المطلقة لعدد ما ، بمعنى لو قمت بتمرير قيمة للمتغير x وكانت قيمة سالبة ستقوم الدالة بحساب القيمة المطلقة لهذا المتغير عن طريق تحول تلك القيمة السالبة إلى قيمة موجبة ، كذلك القيمة الموجبة تظل كما هي قيمة موجبة لأنها في حد ذاتها لم يطرأ عليها تغيير قيمة الإشارة بالسالب .

مثال على ذلك :

```
mysql> select abs(-4);

+---------+
| abs(-4) |
+---------+
|       4 |
+---------+

1 row in set (0.00 sec)
```

هنا قامت الدالة ABS() بحساب القيمة المطلقة للرقم 4- وذلك بتحويل تلك القيمة إلى قيمة موجبة وكانت النتيجة لذلك هي فعلاً 4 بإشارة موجبة .

أما لو قمت بتمرير قيمة موجبة للدالة ABS() فستكون النتيجة هي نفسها لم تتغير ولاحظ معى التالي :

```
mysql> select abs(4);

+--------+
| abs(4) |
+--------+
|      4 |
+--------+

1 row in set (0.00 sec)
```

كما تلاحظ نفس النتيجة كما ذكرنا لم تتغير .

SIGN(x) Function

تقوم الدالة Sign() بعرض ثلاث قيم مناظرة لأي رقم يتم تمريره لها عن طريق المتغير x ، هذه القيم الثلاث :

```
Sign(x) display -1 when x is negative
sign(x) display  1 when x is positive
sign(x) display  0 when x is zero value
```

أي أن الدالة sign() تقوم بعرض 1 عندما تكون قيمة x موجبة ، بينما تقوم بعرض 1- عندما تكون قيمة x سالبة وفى الأخير تقوم بعرض 0 عندما تكون قيمة x تساوى صفر .
أمثلة على ذلك :

```
mysql> select sign(12);

+----------+
| sign(12) |
+----------+
|        1 |
+----------+

1 row in set (0.03 sec)
```

```
mysql> select sign(-100);

+------------+
| sign(-100) |
+------------+
|         -1 |
+------------+

1 row in set (0.00 sec)
```

```
mysql> select sign(0);

+---------+
| sign(0) |
+---------+
|       0 |
+---------+

1 row in set (0.00 sec)
```

POWER(x,y) Function

تقوم الدالة power() بحساب رقم مرفوع لأُس رقم آخر أي مثلاً تريد حساب x أس y ، وبالطبع مكان كلٍ من المتغير x و y أرقام ، ونوضح ذلك بالأمثلة التالية :

مثال : تريد حساب 2 أس 3 والتي من المعلوم أن نتيجة العملية هي 8 :

```
mysql> select power(2,3);

+------------+
| power(2,3) |
+------------+
|          8 |
+------------+
```

```
1 row in set (0.00 sec)
```

فعلا نتيجة العملية كانت 8 وتستطيع تغيير الأرقام السابقة بكل سهولة شريطة أن تعرف أي الرقمين ستبدأ بهما هل تريد مثلاً 4 أس 5 أو العكس 5 أس 4 فطبيعي نتيجة كلا العملتين ستكون مختلفة كل ما هنالك أنك ستقوم بتحديد الأساس والذي كان في مثالنا السابق هو الرقم 2 ثم بعد ذلك تفصل بين الأساس والأس والذي كان 3 ب colon .

SQRT(x) Function

تقوم الدالة SQRT() أو SQRT(x) Function بحساب الجذر التربيعي للمتغير x ، فمثلاً تريد الجذر التربيعي للرقم 9 :

```
mysql> select sqrt(9);

+---------+
| sqrt(9) |
+---------+
|       3 |
+---------+

1 row in set (0.00 sec)
```

فعلا تم حساب الجذر التربيعي للرقم 9 وهو 3 وتستطيع بالتأكيد تغيير الرقم 9 إلى أي رقم آخر .

ROUND(x) and ROUND(x,y) Function

تقوم الدالة Round() أو Round(x) Function بتقريب المتغير x والذي يحتوى على كسور عشرية إلى أقرب رقم صحيح .

مثال : تريد تقريب الرقم 20.6 إلى أقرب رقم صحيح :

```
mysql> select round(20.6);

+-------------+
| round(20.6) |
+-------------+
|          21 |
+-------------+

1 row in set (0.00 sec)
```

كما أن الدالة round() تستطيع أن تأخذ خيار إضافي يمرر لها لتقريب المتغير x والذي يحتوى على كسور عشرية إلى عدد أقرب أرقام عشرية ممكنة والتي نحددها للدالة من خلال المتغير y .

مثال : نريد تقريب الرقم 96.874698563 إلى أقرب ثلاثة أرقام عشرية :

```
mysql> select round(96.874698563,3);

+-----------------------+
| round(96.874698563,3) |
+-----------------------+
|                96.875 |
+-----------------------+

1 row in set (0.00 sec)
```

```
mysql> select round(-12.7);

+--------------+
| round(-12.7) |
+--------------+
|          -13 |
+--------------+

1 row in set (0.11 sec)
```

FLoor(x) Function

تقوم الدالة ()Floor أو Floor(x) Function باسترجاع أكبر قيمة صحيحة تكون أقل من أو تساوى قيمة المتغير x ، بمعنى أن المتغير x قيمته مثلاً 20.2 عند ذلك ستكون أكبر قيمة صحيحة أقل من أو تساوى قيمة المتغير x هي الرقم 20 ولاحظ ذلك كما يلي :

```
mysql> select floor(20.2);

+-------------+
| floor(20.2) |
+-------------+
|          20 |
+-------------+

1 row in set (0.00 sec)
```

```
mysql> select floor(-3.2);

+-------------+
| floor(-3.2) |
+-------------+
|          -4 |
+-------------+
```

```
1 row in set (0.00 sec)
```

ملحوظة : طبعا من المسلم به أنه كلما كان الرقم ذو الإشارة السالبة كبيرا كلما قلت قيمته العددية بمعنى الرقم 3- أكبر من الرقم 4- وهكذا الرقم 99- أكبر من الرقم 100- فتنبه لذلك .

CEILING(x) Function

تقوم الدالة ceiling() أو Ceiling (x) Function باسترجاع أقل قيمة صحيحة تكون أكبر من أو تساوى قيمة المتغير x ، بمعنى أن المتغير x قيمته مثلاً 30.1 عند ذلك ستكون أقل قيمة صحيحة أكبر من أو تساوى قيمة المتغير x هي الرقم 31 ولاحظ ذلك كما يلي :

```
mysql> select ceiling(30.1);

+---------------+
| ceiling(30.1) |
+---------------+
|            31 |
+---------------+

1 row in set (0.00 sec)
```

```
mysql> select ceiling(-99.5);

+----------------+
| ceiling(-99.5) |
+----------------+
|            -99 |
+----------------+

1 row in set (0.00 sec)
```

Trignometic Function (Tan(x) , Cos(x) , Sin(x))

أولا الدالة : tan()

مثال نريد حساب قيمة الزاوية 30 باستخدام الدالة tan() :

```
mysql> select tan(30);

+------------------+
| tan(30)          |
+------------------+
```

```
| -6.4053311966463 |
+------------------+

1 row in set (0.00 sec)
```

```
mysql> select cos(30);

+------------------+
| cos(30)          |
+------------------+
| 0.15425144988758 |
+------------------+

1 row in set (0.00 sec)
```

```
mysql> select sin(30);

+-------------------+
| sin(30)           |
+-------------------+
| -0.98803162409286 |
+-------------------+

1 row in set (0.00 sec)
```

لاحظ أن جميع الحسابات السابقة كانت ب radian وليست بال degrees .

تحديث السجلات (Records) باستخدام جملة أمر Update

تعاملنا السابق مع MySQL كان يتناول محوران رئيسيان :

الأول : هو كيفية إضافة بيانات جديدة داخل أعمدة الجدول
الثاني : هو كيفية الاستعلام عن تلك البيانات

لذلك لم نكن لنهتم كثيراً إذا حدث تغير في أحد بيانات الموظفين سواء كان زيادة الراتب مثلاً ، أو تغير عدد سنين العمل بالشركة وهكذا ، لأنه حينها لم نكن نعلم ماهي الجملة المسؤولة عن تحديث تلك البيانات القديمة بأخرى جديدة مستحدثة ، لذا يأتي هنا دور جملة أمر Update والتي تقوم بتلك المهمة على أكمل وجه وتأخذ جملة أمر Update الصيغة العامة على النحو التالي :

```
UPDATE table_name SET
column_name1 = value1,
column_name2 = value2,
column_name3 = value3 ...
[WHERE conditions];
```

حيث column_name1 و column_name2 و column_name3 هي أسماء الأعمدة أو الحقول المراد تحديث البيانات بها ، أما value1 و value2 و value3 هي القيم الجديدة التي تحل مكان القيم القديمة ، وفى الأخير جملة where لوضع الشروط المراد تمريرها إلى حقل معين ، قد يكون الأمر في البداية غامضاً ولكن إن شاء الله من خلال تطبيقنا للأمثلة القادمة سنكون قادرين على استيعاب عمل الجملة بشكل جيد .

الآن نتطرق إلى بعض الأمثلة العملية كي تستوعب ما قمناه بسرده حول جملة أمر Update سابقاً وستكون الأمثلة منوعة لكى نقوم بتغطية ذلك المفهوم بشكل جيد إن شاء الله .

مثال : نريد تحديث كلاً من العمود salary والعمود perks لإضافة رواتب وحوافز جديدة تخص مثلاً الموظف Sherif Shahin وجعل المرتب $ 150000 والحوافز $ 50000؟

في البداية سنقوم بالاستعلام عن راتب الموظف Sherif Shahin لكى نلاحظ الفارق بعد عملية التحديث :

```
mysql> select f_name, l_name, salary, perks
    -> from employee_data
    -> where f_name='Sherif' and l_name='Shahin';

+--------+--------+--------+-------+
| f_name | l_name | salary | perks |
+--------+--------+--------+-------+
| Sherif | shahin | 120000 | 25000 |
+--------+--------+--------+-------+

1 row in set (0.00 sec)
```

الآن سوف نقوم بعملية تحديث لكل من العمود salary والعمود perks بالقيم السابقة والتي ذكرناها في المثال وكانت $ 150000 بالنسبة للراتب و $ 50000 بالنسبة للحوافز :

```
mysql> update employee_data set
    -> salary=15000,
    -> perks=50000
    -> where f_name='sherif' and l_name='shahin';

Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

ثم نقوم بالاستعلام مرة أخرى عن الموظف Sherif Shahin للتأكد من تحديث كلاً من بيانات الراتب والحوافز الخاصة به :

```
```

```
mysql> select f_name, l_name, salary, perks
    -> from employee_data
    -> where f_name='sherif' and l_name='Shahin';

+--------+--------+--------+-------+
| f_name | l_name | salary | perks |
+--------+--------+--------+-------+
| Sherif | shahin |  15000 | 50000 |
+--------+--------+--------+-------+

1 row in set (0.02 sec)
```

كما تلاحظ أن كلا العمودين salary و perks تم تحديث البيانات الخاصة بهما والتي تخص الموظف Sherif Shahin وبالتالي من خلال الصيغة العامة لجملة أمر Update تستطيع تحديد الحقول المراد إجراء عملية التحديث لها ، ثم بعد ذلك نقوم بوضع الشروط المناسبة لتنفيذ عملية التحديث .

ملحوظة مهمة جداً :

ينبغي عليك عند استخدامك لجملة أمر Update أن تتحقق من الشروط التي تقوم بوضعها حتى لا تقوم بتحديث أعمدة بالخطأ وتُحدِث بلبلة في البيانات داخل الجدول ولذلك فلتعط ذلك الأمر أهميته حتى لا تتسبب في ما لا يمكن أن تدرك نتائجه ، كما يجب عليك عند تنفيذ جملة أمر Update التحقق من وجود جملة الشروط where حيث أنه في حالة عدم وجود الجملة ستقوم جملة أمر update بتحديث جميع السجلات داخل الصفوف كلها مرة واحدة فتنبه لذلك !!!