بسم الله الرحمن الرحيم والصلاة والسلام على اشرف الانبياء والمرسلين ، سيتم التطرق في هذا الدرس بمشيئة الله الى ما يلي :

- عباراة الـ if .
- عباراة الـ else .
- عباراة الـ elseif .
- عباراة الـ switch .
- حلقة التكرار while .
- حلقة التكرار for .
- حلقة التكرار do while .

- عباراة الـ if :

استخدام الـ if في كتابة السكربات شي أساسي ، وكما في لغات البرمجة الأخرى فإن الـ PHP تتبع نفس الاسلوب في كتابة الـ if ، فيمكن تحديد شرط معين مقترن بالـ if وبالتالي اذا كان الشرط صحيحاً (true) فسيتم تنفيذ الاسطر المحددة ، وبتفصيل أكثر يجب وضع الشرط بين قوسين ( ) ، ووضع الاسطر المطلوب تنفيذها بين العلامات { } ، مع ملاحظة أنه يمكن التخلي عن العلامات { } في حال وجود سطر واحد فقط .

فلنفترض وجود نموذج بريدي (Mail Fourm) ، يحتوي على الإسم والبريد والرسالة ، ونرغب في معرفة ما اذا كان المرسل قد ملأ جميع الحقول وبالتالي إرسال الرسالة ، او انه لم بفعل ذلك وبالتالي عرض رسالة (فضلاً قم بتعبئة البيانات كاملة) ، لعمل ذلك نحتاج معرفة أسماء المتغيرات في النموذج ، ولذلك فلنفترض أن المتغيرات كالتالي :
(الإسم $name) ، (البريد $email) ، (الرسالة $later) ، ولعمل الشرط الأول (اذا كان الإسم لم يُدخل فلن يتم ارسال الرسالة) :

```
<?
 if ( $name == "" )
 "; فضلاً قم بتعبئة البيانات كاملة echo "
 ?>
```

والمعنى أنه إذا كان المتغير $name لا يحتوي على أي قيمة (أي فراغ) فسيتم تنفيذ السطر التالي وطباعة الجملة ، مع ملاحظة أن المطلوب تنفيذه هم سطر واحد فقط ولذلك لم نستخدم { } ، بل في حالة وجود أكثر من سطر يجب استخدامها كالتالي :

```
<?
if ( $name == "" )  {
<br> "; فضلاً قم بتعبئة البيانات كاملةecho "
 "; لم تقم بإدخال الإسمecho "
}
?>
```

- عباراة الـ else :

هذه العبارة تتيح امكانية وجود اجراء ثاني لعدم تحقق الشرط ، ففي مثالنا السابق كان الاجراء طباعة الجملة اذا تحقق الشرط ، ولكن في حالة عدم تحقق الشرط فلن يكون هناك اجراء لتنفيذه ، بل ان الاجراء سيتم تنفيذه اذا تحقق الشرط ومن ثم سيتم اكمال بقية الاسطر ، وفي حالة مثل هذه الحالة يتم استخدام الـ else لوضع اجراء آخر في حالة عدم تحقق الشرط ، وبالمثال يتضح المقال :

```
<?
if ( $name == "" )  {
"; فضلاً قم بتعبئة البيانات كاملةecho "
}
else
{
"; تم ارسال الرسالة ، شكرا لكecho "
}
?>
```

في هذا المثال سيتم طباعة الجملة (فضلاً قم بتعبئة البيانات كاملة) اذا تحقق الشرط أن المتغير $name لا يحتوي على أي قيمة ، وسيتم طباعة الجملة (تم ارسال الرسالة ، شكرا لك) في حالة عدم تحقق الشرط ، أي في حالة وجود قيمة في المتغير $name ، مع ملاحظة أن هذا المثال يحتوي على شرطين وليس شرط واحد ، فالظاهر هو شرط واحد ($name == "") ولكن العبارة else تعتبر شرطاً بحد ذاتها ولو لم يكن هذا الشرط مكتوباً ، وكما هو واضح فمعنى هذا الشرط هو (إذا كان غير ذلك) فقم بطباعة الجملة .

يمكن أن يكون الشرح غير واضح تماماً ، ولكن أهمية فهم الطريقة ستتضح في الأسطر القليلة القادمة .

- **عباراة الـ** elseif :

في العبارة السابقة ذكرنا أنه يوجد شرطين واجرائين ، أحد هذين الشرطين غير مكتوب بل هو مفهوم من ادراج العبارة else ، وفي حالات كثيرة لا يكفينا مجرد شرطين واجرائين لاتمام بعض السكربات المعقدة ، فلذلك يمكن نستخدم العبارة elseif مع الـ if لعمل مثل هذه السكربات ، فلو افترضنا أن لدينا عداد لزوار الموقع ونريد اظاهر العداد بحيث يتم قراءته بشكل جيد ، اي بمعني اخر اذا كان عدد الزوار (1) فسيتم طباعة الجملة (عدد الزوار : زائر واحد فقط) واذا كان (2) فسيتم طباعة الجملة (عدد الزوار : زائرين) ... وقس على ذلك ، فعندما يكون عدد الزوار (1) فسيتم عرض الجملة الأولى فقط وعندما يكون عدد الزوار (2) فسيتم عرض الجملة الثانية فقط ، وهكذا لبقية الشروط .

بافتراض أن المتغير ($counter) هو عداد الزوار ، فالمثال التالي يبين ما تم شرحه سابقاً :

```
<?
if ( $counter == 1 )  {
 "; عدد الزوار : زائر واحد فقطecho "
}
elseif ( $counter == 2 )  {
 "; عدد الزوار : زائرينecho "
}
elseif ( $counter >= 3 && $counter <= 10 )  {
"; زوار : $counter عدد الزوارecho "
}
else {
"; زائر : $counter عدد الزوارecho "
}
?>
```

كما هو واضح في المثال السابق سيتم ما يلي :

الشرط : العداد يساوي 1
الإجراء : طباعة (عدد الزوار : زائر واحد فقط)
الشرط : العداد يساوي 2
الإجراء : طباعة (عدد الزوار : زائرين)
الشرط : العداد أكبر أو يساوي 3 و اصغر أو يساوي 10
الإجراء : طباعة (عدد الزوار : (العداد) زوار)
الشرط : العداد لا يحقق أي من الشروط
الإجراء : طباعة (عدد الزوار : (العداد) زائر)

ملاحظة بسيطة فقط ، وهي على العلامة && التي تعني (و) ، وهي من علامات الجمع بين جملتين ، فيجب أن تكون الجملتين صحيحتين لتحقق الشرط .

- **عباراة الـ** switch :

هذه العبارة قريبة جداً من العبارة if ، ولكن يمكن استخدام اكثر من شرطرين باسلوب أخر ، غير انه يجب اسناد قيمة معينة للـ case وهي هنا بمثابة الشرط ، لكي يتم تنفيذ الاجراء المحدد لذلك الشرط أو الـ case ، وفي النهاية الأمر يعود الى المصمم وايهما يفضل ، وكما في المثال السابق يمكن كتابة مثال بالـ switch بنفس الطريقة ، والمشكلة الوحيدة هي كما قلنا أنه يجب اسناد قيمة معينة لكل case وبالتالي

فإن الشرط الثالث من المثال السابق يجب تفريقه لكل قيمة من (3 الى 10) ، وهذه العميلة مجهدة لانه يجب كتابة سطر لكل قيمة كما يلي :

القيمة : 3
الـ case : 3
الاجراء : طباعة (عدد الزوار : (العداد) زوار)
القيمة : 4
الـ case : 4
الاجراء : طباعة (عدد الزوار : (العداد) زوار)
القيمة : 5
الـ case : 5
الاجراء : طباعة (عدد الزوار : (العداد) زوار)
..... الخ ...

وفي المثال التالي ساتغاضى عن الشرط الثالث بكامله ، واذكر بقية الشروط والحالات لمجرد فهم طريقة عمل هذه العبارة :

```
<?
switch ($counter)
{
case "1";
 "; زائر واحد فقطعدد الزوار : echo "
break;
case "2";
 "; زائرينعدد الزوار : echo "
break;
default;
"; زائر : $counter عدد الزوارecho "
break;
}
?>
```

استخدمنا في هذه المثال بعض الجمل وتعني ما يلي :
Switch وتكتب في البداية مع ادراج اسم المتغير الذي سيتم عمل الشروط عليه .
Case أي في حالة (....) ويكتب بجانبها الشرط .
Break وتعني ايقاف العملية والخروج من الشرط بعد تنفيذ أحد الإجراءات .
Defaukt وهي تقابل العبارة else أي بمعنى أنها لأي حالة لم يتم ذكرها في الشروط .

- حلقة التكرار while :

وهي ابسط حلقات التكرار على الأطلاق ، بحيث تاخذ شرط واحد فقط وتبني على تنفيذ ما بين علامات الشروط { } ، والفرق الوحيد بينها وبين الـ if هو انها ستقوم بتنفيذ الاجراءات طالما كان الشرط صحيحاً ، وهذا يعني احتمال تنفيذ الإجراء أكثر من مرة ، وهذ الدالة مفيدة في ادراج الحقول من الجدوال وغيرها من الاستخدامات ، بحيث لو افترضنا وجود جدول معين في قاعد بيانات ونريد ادراجه في صفحة PHP ، فسيكون من اهم خيارات الاستخدام هذه الدالة ، وباذن الله سيتم التطرق لقواعد البيانات في الدروس القادمة ، وفي الوقت الحالي ساذكر مثال بسيط على هذه الدالة لفهم طريقة استخدامها :

```
<?
$total = 10;
while ( $total <= 50 )
{
<br>"; 50 العدد أقل منecho "
$total +=10;
}
?>
```

كير بسيط يمكن معرفة أن الجملة (العدد أقل من 50) سيتم طباعتها 5 مرات ، لان حلقة التكرار while قامت بتنفيذ الاجراء طالما أن الشرط صحيح ، وفي المرة الأولى كان المتغير ($total) يساوي (10) والشرط صحيح لان الـ ($total) فعلاً اصغر أو يساوي الـ (50) ، فتم تنفيذ ما بين علامات الشرط ، ومن ذلك زيادة متغير المجموع ($total) بقيمة (10) ومن ثم الرجوع والمقارنة من جديد ، وفي هذه الحالة صار المتغير ($total) يساوي (20) وأيضاً الشرط صحيح وبالتالي الدخول مرة أخرى وتنفيذ الأجراء .... وهكذا حتى يتم الوصول الى أن قيمة الـ ($total) يساوي (50) وبالتالي الشرط صحيح ، ومن ثم تصبح قيمة الـ ($total) تساوي (60) وفي هذه الحالة يتم ايقاف تنفيذ الاجراءات لأن الشرط غير صحيح .

- **حلقة التكرار** for :

يوجد طريقة أسهل للتعامل مع المثال السابق ، فاستخدام حلقة التكرار while كانت القيمة الابتدائية للمتغير ($total) في سطر ، والشرط في سطر والزيادة على المتغير في سطر أخر ، وبالتالي زيارة في عدد الأسطر عن ما يمكن استخدامه مع حلقة التكرار for ، فالمثال التالي يبين طريقة أخرى لاستخدام مثال الـ while بطريقة أسهل :

```
<?
for ( $total = 10;  $total <=50; $total +=10 )
{
<br>"; 50 من أقل العددecho "
}
?>
```

وللتوضيح فان تركيب الـ for هو على الشكل التالي :

```
for ( القيمة الأفتراضية; الشـرط; مقدار الزيـادة; )
{
الإجراء المطلوب تنفيذه
}
```

- **حلقة التكرار** do while :

وهي نسخة أخرى من الـ while والفرق الوحيد بينهما أن التأكد من الشرط وصحته من عدمها يتم بعد تنفيذ الاجراء وليس قبله كما في الـ while وكمثال عليها :

```
<?
$total = 10;
do
{
echo "50 من أقل العدد<br>";
$total +=10;
}
while ( $total <= 50 );
?>
```

وفي نهاية الدرس اتمنى الفائدة للجميع
بسم الله الرحمن الرحيم والصلاة والسلام على اشرف الانبياء والمرسلين اما بعد ، ففي هذا الدرس بمشيئة الله تعالى سنتطرق الى مفاهيم عامة عن قواعد البيانات عموما وعن الـ Mysql خصوصاً ، لتكون بداية فهم لقواعد البيانات الهامة لأي لغة برمجة .

في البداية سنتعرف على مصطلح الـ RDBMS ، ونعني بذلك قواعد البيانات العلائقية ، والتي من خصائصها سهولة الوصول الى البيانات المخزنة فيها ، وسرعة اتمام عمليات الاستعلام المختلفة ، وبالاضافة الى المميزات الأخرى فان هذه النوع يعتبر الأكثر استخداماً في جميع التطبيقات سواء المستخدمة في الانترنت أو ذات الطابع البرمجي الخاص ، وبطبيعة الحال فإن الـ Mysql من هذا النوع .

ومن المهم معرفة بعض الاساسيات في الـ RDBMS ، والتي من شأنها تسهيل عملية فهمك التام لطريقة عملها والتعامل معها ..

1- **الجداول** Tables :

تعتبر أكبر جزء في قاعد البيانات ، وهي عبارة عن أعمدة وصفوف تحتوي على قيم معينة .

2- **الأعمدة** Columns :
لكل عمود في الجدول أسم خاص يختلف عن أسماء الأعمدة الأخرى في نفس الجدول ، ويجب ان يكون لكل عمود نوع خاص به يصف نوع البيانات التي ستخزن فيه ، وكم يظهر في الصورة ، فان عمود الرقم من النوع الرقمي Integer ، اما الحقلين الآخرين فهي نصوص Text .

3- **الصفوف** Rows :
كل صف من صفوف الجدول يحتوي على قيم مختلفة ويمثل معلومات متكاملة عن قطاع معين ، وفي مثالنا يمثل معلومات متكاملة عن سخص معين .

4- **القيم** Values :
وهي ما تحتوي عليه تقاطعات الصفوف بالاعمدة .

5- **المفاتيح** Keys :
وتعتبر من اساليب تسهيل الوصول الى المعومات في قواعد البيانات ، وفي مثالنا السابق نرى أن العمود Id يحتوي على ارقام متسلسلة لا تتكرر نهائياً بل أنها تتكون بشكل تلقائي عند ادراج أي صف جديد للجدول ، وبالتالي فإنها تعتبر المفتاح المناسب لكل صف من صفوف الجدول لضمان عدم الالتباس في اختيار الصفوف .

فلو افترضنا أن لدينا جدولين في قاعدة بيانات ، يحتوي الجدول الأول على معلومات عن الدروس مفصلة على عدة حقول لتلك الدروس ، على سبيل المثال :
الرقم (id) ، الدرس (lesson) ، رقم الكاتب (Key_author) ..
ويحتوي الجدول الثاني على بيانات الأعضاء كما يلي :
الرقم (Key_author) ، الاسم (name) ..

والمطلوب هو طريقة لربط الجدولين ، بحيث أن رقم الكاتب في جدول الدروس (Key_author) يدل على اسم الكاتب في جدول الاعضاء (name) .

بالتدقيق في المثال يتضح أن الحقلين (أو العمودين) Key_author في كلا الجدولين هو مفتاح الربط بينهما ، ولذلك يمكن الوصول الى اسم الكاتب اعتماداً على رقمه من جدول الدروس ، وبالتالي الربط بين الجدولين .

لن اتحدث طويلاً عن مقدمات قواعد البيانات Mysql ، ولكن بهذه المقدمة البسيطة يمكن على الاقل تصور بعض الاساسيات حول قواعد البيانات عموماً والـ Mysql خصوصاً ، ومن وجهة نظري فالاهم هو كيفية التعامل مع قواعد البيانات بما يخدم احتاجاتنا مع الـ PHP ، ولذلك ساتطرق في هذ الدرس الى نقطة هامة جداً وهي ادارة قواعد البيانات ، وأعني بذلك عملية انشاء قواعد البيانات والجداول والتحكم في الحقول والبيانات وغيرها ، لتكون الاساس للتعامل مع قواعد البيانات لاحقاً عن طريق الـ PHP ، ولعمل ذلك يوجد عدة طرق من اهمها الطريقة التقليدية المباشرة بالاعتماد على نظام الدوس في ذلك وبدون استخدام أي برامج أخرى للادارة .

الاتصال بالـ Mysql **، والتعامل معها** :
كما قلنا أن الطريقة التقليدية هي الاتصال بقواعد البيانات عن طريق سيرفر الـ Mysql وبدون استخدام أي مكونات أخرى ، ولعمل ذلك نحتاج أن نعرف مسار سيرفر الـ Mysql على الجهاز المستخدم بعد عملية التثبيت ، كما قمنا بذلك في درس المقدمة ، وعادة يكون المسار كالتالي (C:\mysql\bin) ، وبذلك يمكن تشغيل البرنامج mysql.exe من داخل الـ Dos .

عموماً طريقة الاتصال بقاعدة البيانات هي كالتالي :

```
mysql –h HostName –u UserName –p
```

مع استبدال الـ HostName باسم السيرفر لديك ، سواء كان السيرفر على نفس الجهاز وفي هذه الحالية تكتب localhost ، أو أن السيرفر الذي تود الاتصال به ليس على نفس الجهاز وبذلك تكتب المسار الكامل لاسم السيرفر (HostName) ، ومع استبدال الـ UserName باسم المستخدم الخاص بالـ Mysql لديك ،

بعد ذلك سيتم طلب كلمة المرور الخاصة بقاعدة البيانات بعد الضغط على Enter ، قم بادخالها وسيتم فتح الاتصال بالـ Mysql ، كما يمكن كتابة mysql فقط ليتم فتح الاتصال بقاعدة البيانات فقط اذا كنت تعمل على نفس الجهاز وليس جهاز أخر .

سيظهر المؤشر الخاص باوامر الـ Mysql كالتالي :

```
mysql>
```

وبهذا نكون وصلنا الى المكان المطلوب لكتابة اوامر الـ Mysql والتحكم بها .

الأمر الأول الذي سنقوم بكتابته يقوم باستعراض قواعد البيانات الموجودة على السيرفر والامر هو :

```
show databases;
```

بعد كتابة هذا الأمر (بعد مؤشر الـ mysql>) ، سيتم استعراض قواعد البيانات في السيرفر الذي قمنا بالاتصال به ، وفي حالة عدم وجود أي قاعدة بيانات قمت باعدادها من قبل ، فان من الطبيعي أن تجد قاعدتي بيانات موجودة بشكل تلقائي عند تثبيت السيرفر Mysql ، وتلك القاعدتان هي mysql – test .

ولمحاولة فهم الموضوع بشكل أكبر ، سنقوم بالتطرق الى مثال يبين كيفية انشاء قاعدة بيانات ، وكيفية الدخول لها والتعامل معها وانشاء الجداول ، ومن ثم حذفها ..

بعد استعراض قواعد البيانات بالأمر السابق ، سنقوم بانشاء قاعدة بيانات باسم PHP ، ولعمل ذلك قم بكتابة الأمر التالي :

```
create database PHP;
```

لو قمنا بكتابة الأمر السابق (show database) سنرى أن قواعد البيانات أصبحت 3 باضافة القاعدة PHP الى القاعدتين mysql – test ، ولاستخدام اي منها نقوم بكتابة الأمر التالي في مثالنا مع القاعدة PHP :

```
use PHP;
```

وهذه يعني الدخول في قاعدة البيانات PHP واستخدام المؤشر (mysql>) لكتابة الأوامر المتعلقة بالتعامل مع قاعدة بيانات بعينها .

أول هذه الاوامر هو أمر انشاء جدول في قاعدة البيانات ، وهذه الأمر يحتاج الى تفصيل دقيق لبعض الخصائص مثل اسماء الحقول وانواع البيانات فيها ، وبعض الاشياء الأخرى ، عموماً قم بكتابة الأمر التالي وساقوم بشرح كافة التفاصيل بعد المثال :

```
create table users (
id Int not null auto_increment Primary Key,
name text not null,
counter int
);
```

شرح المثال :

- قمنا بكتابة (create table users) وهذا يعني انشاء جدول باسم users .
- القوس) يعني بداية تسمية حقول الجدول وخصائص تلك الحقول .
- السطر الأول من اسماء الحقول هو (id) والرمز (int) يعني وصف نوع البيانات التي ستخزن في الحقل (id) ، وهي في هذه الحالة تعني نوع البيانات الرقمية ، اما الرمز (not null) فيعني عدم امكانية أن يكون هذا الحقل فارغاً ، بل يجب أن يحتوي على قيمة ، والـ (auto_increment) يجعل الحقل يحتوي على قيم متسلسلة يستحيل تكرارها ، وسيبدأ من الرقم 1 ويبدأ بالزيادة بمقدار واحد في كل مرة يتم ادخال صف جديد الى هذا الجدول ، وفي النهاية الرمز (Primary Key) يعني أن الحقل هو المفتاح الرئيسي لهذا الجدول أو بمعني أنه سيتم التفريق بين صفوف الجدول اعتماداً على هذا الحقل ولهذا وضعنا (auto_increment) لضمان عدم اختلاط البيانات .
- السطر الثاني يحتوي على اسم الحقل (name) ونوع البيانات (text) أي نصي ، ونفس الرمز السابق الذي ذكرناه وهو (not null) .
- السطر الثالث يحتوي على اسم الحقل (counter) ونوع البيانات (int) ، ولاحظ أننا لم نذكر (not null) وبالتالي يمكن أن يكون هذا الحقل فارغاً لا يحتوي على أي قيمة ، ولن يكون هناك أي تعارض أو مشكلة بعكس الحقلين السابقتين .
- في السطر قبل الأخير ، أي قبل علامة الاغلاق (، سيكون بدون فاصلة .
- السطر الأخير يحتوي على اقفال عملية انشاء الجدول بالعلامة); .

عموماً هذا المثال يعطي نبذة بسيطة عن كيفية اجراء مثل هذه الاوامر ، وسنتطرق الى بقية الأوامر في الأسطر القليلة القادمة .

يمكنك استعراض الجداول الموجودة في قاعدة بيانات عن طريق الأمر :

```
show tables;
```

ولو قمت بتطبيق ذلك على المثال السابق فسترى أن الجدول users موجود في قاعدة البيانات PHP التي قمنا بانشاءها .

يمكن كذلك استعراض خصائص الجدول السابق users الذي قمنا بانشاءه في المثال السابق ، عن طريق الأمر التالي :

```
describe users;
```

سترى أن حقول الجدول وخصائص كل جدول ظهرت لك بشكل واضح .

- التعامل مع بيانات الجداول :

بقي أن نذكر الطرق التي يمكن من خلالها ادخال البيانات الى الجدول users ، بل وكيفية التعامل مع تلك البيانات بالتعديل والحذف وغير ذلك ، وكما قلنا سابقاً أن هذه الاساسيات مفيدة جداً في البرمجة بلغة الـ PHP ، بل إن فهم هذه الطرق هو المفتاح الاساسي للتعامل مع قواعد البيانات عن طريق البي اتش بي ،

عموماً أول تلك الأوامر هو اضافة صف جديد الى الجدول ، وهذا ما يبينه المثال التالي :

```
insert into users set
name = "Ahmad";
counter = 3
;
```

مع ملاحظة أن users هو اسم الجدول ، name اسم الحقل (العمود) الأول ، counter اسم الحقل (العمود) الثاني ، كما تلاحظ أن الحقل id لم نتطرق له ، لاننا في اعدادنا للجدول ذكرنا أن الحقل (id auto_increment) أي ستضاف اليه القيم بشكل تلقائي وبشكل منظم ، كما قلنا في كل مرة يزيد العداد بقيمة 1 ، و بطبيعة الحال يمكنك القياس على هذا المثال باستبدال ما يجب استبداله من اسم الجدول (users) واسماء الحقول (name – counter) وكذلك البيانات بما يناسب الذي تريد القيام به .

هذا بالنسبة لاضافة بيانات جديدة الى جدول معين ، اما بالنسبة لاستعراض البيانات في الجدول فكما يلي :

```
select * from users;
```

ومعني select (اختر) ، ولذلك ستجد أن جميع البيانات التي في الجدول users قد تم سردها ، واذا كنت ملتزماً بالمثال السابق حرفياً فستجد أن البيانات التي اضفناها في المثال السابق ظهرت على شكل صف من صفوف الجدول ، وبالتالي كلما اضفت صفاً جديداً الى الجدول وقمت باستعراض البيانات تجد أن بياناتك قد تم تخزينها ، وينطبق الكلام السابق حول الاستبدال هنا ايضا ، فيمكن استبدال اسم الجدول users باي اسم لجدول في قاعدة البيانات المستخدمة ، وللتاكد من اسماء الجداول قم باستخدام الطريقة السابق ذكرها وهي (show tables) .

النقطة الأخيرة التي ساتطرق لها هي ما يجب معرفته حول الأمر select وهو كثرت استخدامه في التعامل عن طريق البي اتش بي ، وبالتالي يجب عليك فهم طريقة كتابته بشكل كامل ، بالاضافة الى خيارات الاختيار إن صح التعبير ، وهي ما يتم كتابته بعد الجملة السابقة من خيارات تحدد طريقة اختيار البيانات من شروط وترتيب وحدود وهذا ما ساذكره في الاسطر القليلة القادمة .

فلفترض أن الجدول السابق يحتوي على أكثر من صف من البيانات بالشكل التالي :

اما البيانات التي نود جلبها فهي كما يلي لكل نقطة على حدة :

1- **بيانات الاعضاء الذين ليس لهم أي موضوع** .
2- **بيانات الاعضاء الذين لهم مواضيع أكثر من** 5 **مرتبين من الاكثر الى الاقل** .
3- **بيانات العضو** Ahmed .
4- **بيانات جميع الاعضاء مرتبين حسب الاسم** .
5- **بيانات العضو الأكثر مواضيعاً** .

سنأخذ كل حالة على حدة ، أما **الحالة الأولى** فيمكن التعامل معها كما يلي :

```
select * from users where counter=0;
```

الزيادة التي قمنا بوضعها هي (where counter=0) أي بحيث أن الحقل (counter) يساوي صفر ، وبالتالي سيتم اهمال أي صف من البيانات التي لا يحتوي الحقل (counter) فيها على القيمة صفر ، وسيتم جلب البيانات التي يحتوي هذا الحقل فيها على صفر .

الحالة الثانية :

```
select * from users where counter >= 5 order by counter;
```

في هذا المثال اضفنا الشرط (where counter =< 5) وهو واضح كما في المثال السابق ولكن تم تغيير الشرط لا اقل ولا اكثر ، اما الاضافة الأخرى فهي طريقة الترتيب وهي (order by counter) وتعني (قم بترتيب البيانات المختارة بحسب الحقل counter) ، وهناك طريقة أخرى للتحكم في الترتيب اما تصاعدي أو تنازلي وذلك باضافة كلمة asc ليكون الترتيب تنازلياً كما هو الحال في المثال السابق ، فسواء ذكرت ذلك أو سيتم اعتبارها تنازلياً بشكل تلقائي ، اما الاهم فهو طريقة الترتيب التصاعدي من الاقل الى الاكبر ويتم ذلك عن طريق كتابة الكلمة desc بعد الترتيب مباشرة لتصبح كما يلي :

```
select * from users where counter >= 5 order by counter desc;
```

الحالة الثالثة :

```
select * from users where name = "Ahmed";
```

لاحظ أن الفرق الوحيد هنا هو استخدام علامات التنصيص ، لان نوع البيانات نصية .

الحالة الرابعة :

```
select * from users order by name;
```

وقد أوردت هذا المثال لبيان أنه يمكن استخدام أحد الخيارات لجلب البيانات وترك باقي الخيارات ، فيمكن كما في المثال استخدام خيار الترتيب (order) وعدم استخدام الخيارات الباقية (where – limit) ، اما الخيار where فقد تطرقنا لنا سابقاً وتعرفنا على فائدته ، والخيار الأخر limit هي ما سيتم التطرق اليه في المثال التالي الخاص بالحالة الخامسة :

الحالة الخامسة :

```
select * from users order by counter limit 1;
```

والـ limit تعني عدد الصفوف المختارة ، أي لو قمنا بكتابة المثال السابق بدون الـ limit ستجد أن جميع البيانات سيتم اختيارها ، ولكن باستخدام الـ limit نقوم بتحديد عدد الصفوف التي سيتم اختيارها استناداً على طريقة ترتيبنا للبيانات ، فكما تلاحظ قمنا بترتيب البيانات بحسب الحقل counter ولم نذكر (desc) ولذلك فالبيانات يتم ترتيبها من الاكبر الى الاصغر ، وبالتالي فاختيارنا للحقل الأول يقضي باختيار بيانات الشخص الأكثر كتابة للمواضيع .

بقي أن نذكر طريقي التعديل والحذف ليكتمل الدرس ، وسنبدأ بطريقة التعديل على البيانات الموجودة في الجدول users من قاعدة البيانات PHP ، والمثال التالي يوضح الطريقة التي سيتم شرحها بعد المثال :

```
update users set
name = "Naser",
counter = 30
where name="Ahmad";
```

الجملة update تعني تحديث أو (قم بتحديث) ، والـ users هو اسم الجدول الذي نعمل عليه ، وفي السطر الثاني قمنا باسناد القيمة Naser الى الحقل name ، والسطر الذي يليه قمنا باسناد القيمة 30 الى الحقل counter ، ولكن لو توقفنا هنا بدون ذكر الصف الذي سيتم التعديل عليه ، سيتم تعديل كافة الصفوف في الجدول مهما كان عددها ، ولذلك كتبنا في النهاية "where name="Ahmad ، بمعنى أن التغيرات السابقة ستحدث فقط على الصف من البيانات التي يحتوي فيها الحقل name على القيمة Ahmad .

ربما يكون المثال غير واضح بشكل كافي ، ولكن مع التمرس والمحاولة ستجد أن المسألة منطقية وواضحة

بشكل كبير ، عموماً لم يبقى لدينا الا طريقة الحذف ، سواء كان لكل البيانات في الجدول ، أو لصف معين من البيانات وسنرى ذلك في المثالين التاليين ، وهما ما سنختم به هذا الدرس :

```
delete from users;
```

الأمر السابق كفيل بالغاء جميع الصفوف في الجدول users كما هو واضح ، ولذلك كن متأكداً من أن التجارب التي تقوم بها هي على بيانات غير هامة .

```
delete from users
where id = 1 ;
```

وهذا الحذف سيتم على الصف الذي يتحقق عليه الشرط ، وفي هذه الحالة على الصف من البيانات التي يحتوي فيها الحقل id على القيمة 1 .

الدوال (Function) :

يوجد في PHP العديد من الدوال التي تقوم بوظيفة معينة (محددة) كذلك توجد إمكانية إنشاء دوال تؤدي وظيفة خاصة وحديثنا هنا عن هذا النوع من الدوال (كيفية إنشاء دوال)

الدالة تقوم بتنفيذ شئ معين حيث تأخذ (متغيرات – معطيات) ثم تقوم بمعالجة هذه المتغيرات وتخرج قيمة أخرى .

- الشكل العام – التركيب :

```
Function اسم الدالة ( المعطيات – المتغيرات – البارامتر )
{
 هنا يتم كتابة الكود
Return ( المعطيات – المتغيرات – البارامتر )  ;
}
```

- تعريف الدالة :

لكي نقوم بتعريف دالة نكتب كلمة function بعدها اسم الدالة وبعد الاسم نكتب المعطيات – المتغيرات بين قوسين .

مثال :

```
<?
Function aa($s)
?>
```

حيث aa هو اسم الدالة ، وبالتأكيد يمكن أن يكون أي اسم .
($s) هو (المتغر – المعطى – البارامتر) ، أي اسم من هذه كما تحب أن تسميه .
مع ملاحظة عدم وضع فاصلة منقوطة بعد هذا السطر .

بعد ذلك نقوم بكتابة كود الدالة (عمل الدالة) بين العلامتين { } ، كما يجب أن ننهي الدالة بكلمة return لإعلام الدالة بأن وظيفتها قد انتهت بالاضافة الى ذكر اسم المتغير المذكور في تعريف الدالة سابقا ..

مثال :

```
<?
Return($s) ;
?>
```

- استخدامات الدالة :

يمكن وضع الدالة في أي مكان في شفرة php في أولها أو آخرها بمعنى انه يمكن استدعاء دالة تم تعريفها في أخر الشفرة أو العكس .

- إظهار نتيجة الدالة (طباعة الدالة) :

نستخدم الأمر الخاصة بالطباعة echo أو print وبعده طبعا اسم الدالة ..

مثال :

```
<?
echo aa(5);
```

```
print aa(5);
?>
```

مثال كامل :

```
<?
 //تعريف الدالة
function aa($a)
{
$a=$a*$a*$a*$a;
return($a);
}
 //طباعة ناتج الدالة عند ادخال الرقم 5 فيها
echo aa(5);
?>
```

هذه الدالة تقوم بحساب عدد مرفوع لأس أربعة بمعنى أن العدد مضروب في نفسه أربع مرات اسم الدالة aa وعند طباعة مخرجات الدالة لرقم ، كتبنا أمر الطباعة قبل اسم الدالة والرقم المراد حساب الأس الرابع له بين قوسين (5) وهكذا إذا وضعنا أي رقم آخر سوف تقوم الدالة بحساب الأس الرابع للرقم مباشر وفي مثالنا هذا يتم طبع الرقم 625 .

نقطة أخرى هي أننا قمنا بتمرير قيمة ثابتة الى الدالة ، ولذلك يمكننا أن نمرر للدالة متغير كما في المثال التالي :

```
<?
function as($a)
{
$a=$a*$a*$a*3  ;
return($a) ;
}
$z=10 ;
echo as ($z) ;
?>
```

في هذا المثال تقوم الدالة بضرب العدد في نفسه ثلاث مرات ثم في الرقم 3 ، ونلاحظ أننا مررنا المتغير $z الى الدالة as وكتبناها جميعها في سطر طباعة نتيجة الدالة بالأمر echo . ولذلك تقوم الدالة في هذا المثال بضرب الرقم 10 في نفسه ثلاث مرات ثم في 3 يكون الناتج 3000 ومن ثم يتم طباعة الناتج ، وبطبيعة الحال كلما غيرنا قيمة المتغير أختلفت نتيجة الدالة .

- **العمليات الرياضية** :

هي نفسها العمليات التي درستها في المرحلة الابتدائية من (جمع + ، طرح - ، ضرب * ، قسمة /) والزائد عليهم التي لم تدرسه تقريبا هو باقي القسمة (%) ..

مثال شامل على كل العمليات في الـ PHP :

```
<?
$a = 6;
$b=2;
$c= $a +$b;
// 8 سوف نحصل على ناتج الجمع

$c= $a -$b;
// 4 سوف نحصل على ناتج الطرح

$c= $a * $b;
// 12 سوف نحصل على ناتج الضرب
```

```
$c= $a /$b;
// 3 سوف نحصل على ناتج القسمة

$a = 7;
$b=2;
$c= $a % $b;
// 1 سوف نحصل على باقي القسمة
?>
```

- **عمليات** Assignment :

=

احفظ القيمة في المتغير ، بمعنى خزن القيمة 3 في المتغير $a :

```
<?
$a = 3;
print $a;
// 3 يطبع
?>
```

+ =

إضافة قيمة إلى قيمة في نفس المتغير :

```
<?
$a = 3;
$a += 3;
print $a;
// 6 يطبع
?>
```

- =

اطرح المقدار واحد من المقدار ثلاثة في المتغير $a

```
<?
$a = 3;
$a -= 1;
print $a;
// 2 يطبع
?>
```

* =

يضرب القيمة 3 بالقيمة 2 ويكون الناتج مخزن في نفس المتغير :

```
<?
$a = 3;
$a *= 2;
print $a;
// 6 يطبع الناتج
?>
```

/ =

يقسم قيمة على قيمه أخرى :

```
<?
$a = 6;
$a /= 2;
print $a;
// 3 يطبع ناتج القسمة
?>
```

.=
دمج سلسلة حرفية :

```
<?
$a = "This is ";
$a .= "a test.";
print $a;
// : يطبع الجملة التالية
// This is a test.
?>
```

- عوامل الإضافة و الطرح :

لو افترضنا أننا لدينا المتغير $a=3 و أردنا إضافة واحد إليه بحيث يصبح 4 أو طرح واحد منه بحيث يصبح 2 ، لدينا العوامل التالية :

$++a ارجع قيمة a ثم اضف واحد إليها
$a++ اضف واحد إليها ثم ارجع القيمة
$a-- ارجع القيمة ثم اطرح واحد منها
$--a اطرح واحد ثم ارجع القيمة

++value
يتم إضافة واحد إلى الرقم خمسة :

```
<?
$a = 5;
print ++$a;
// يطبع القيمة 6
?>
```

++value
يرجع القيمة نفسها وفي استخدام ثاني تزيد القيمة واحد :

```
<?
$a = 5;
print $a++;
// طباعة الرقم 6
print "<br>";
print $a;
// طباعة الرقم 5
?>
```

value--
يطرح من القيمة واحد :

```
<?
$a = 5;
print --$a;
// يطبع الرقم 4
?>
```

--value
يرجع القيمة نفسها وفي استخدام ثاني يطرح منها واحد :

```
<?
$a = 5;
print $a--;
// يطبع الرقم 4
print "<br>";
print $a;
```

```
// 5 يطبع الرقم
?>
```

- عمليات المقارنة Comparasion Operators :

$a == $b المتغيران متساويان ..
$a ===$b المتغيران متساويان و من نفس النوع ..
$a !=$b المتغير الاول لا يساوي الثاني ..
$a !==$b المتغير الاول لا يساوي الثاني وليس من نفس النوع ..
$a < $b أكبر من ..
$a > $b أصغر من ..
$a =<$b أكبر من او يساوي ..
$a =>$b أصغر من او يساوي ..

== (تساوي)

تساوي القيمة المخزنة في المتغير الأول بالقيمة المخزنة في المتغير الثاني :

```
<?
$x = 7;
$y = "7";
 " . $y; تساويif ($x == $y) print $x . "
 7 تساوي 7 يطبع//
?>
```

=== (تساوي ومن نفس النوع)

تساوي القيمة المخزنة في المتغير الأول بالقيمة المخزنة في المتغير الثاني وتكون القيم من نفس النوع (حرفية – عددية) :

```
<?
$x = 7;
$y = 7;
if ($x === $y) print $x . " is identical to " . $y;
// 7 يطبع is identical to 7
?>
```

!= (لا تساوي)

إذا كانت القيم المخزنة في المتغيرين غير متساويين :

```
<?
$x = 8;
$y = 4;
if ($x != $y) print $x . " تساوي لا " . $y;
// 4 يطبع 8 لا تساوي
?>
```

!== (لا تساوي ولا من نفس النوع)

إذا كانت القيم المخزنة في المتغيرين غير متساويين وليست من نفس النوع :

```
<?
$x = 8;
$y = 9;
if ($x !== $y) print $x . " iتست نوع نفس من " . $y;
// 9 يطبع 8 ليست من نفس نوع
?>
```

< (أقل من)

مقارنة بين قيمتين واحدة أقل من الاخرى :

```
<?
$x = 5;
$y = 9;
```

```
if ($x < $y) print $x . " من أقل  " . $y;
// يطبع 5 أقل من 9
?>
```

(أكبر من) >

مقارنة بين قيمتين واحدة أكبر من الاخرى :

```
<?
$x = 9 ;
$y = 5;
if ($x > $y) print $x . " من اكبر  " . $y;
// يطبع  9 أكبر من 5
?>
```

(أقل من ويساوي) =>

مقارنة بين قيمتين واحدة أقل من الاخرى أو مساوية لها :

```
<?
$x = 5;
$y = 5;
if ($x <= $y) print $x;
// يطبع القيمة 5
?>
```

(أكبر من ويساوي) =<

مقارنة بين قيمتين واحدة أكبر من الاخرى و مساوية لها :

```
<?
$x = 7;
$y = 5;
if ($x >= $y) print $x;
// يطبع القيمة 7
?>
```

العمليات المنطقية Logical Operations :

لكي تكون قيمة الشرط صحيحة فيجب أن تنطبق القواعد التالية الخاصة بكل عامل منطقي على حدة ، والعوامل هي :

(and) يجب تحقق الاثنين a and b

(or) يجب تحقق كلاهما او احدهما a or b

(Xor) يجب تحقق احدهما و ليس كلاهما a xor b

(!) نفي تحقق الشرط نفي لقيمة a !$a

ملاحظة : يمكن كتابة الـ (and) بالشكل التالي (&) والـ (or) بالشكل التالي (|) والـ (Xor) بالشكل التالي (^) ..

(و) And

إذا تحقق الشرطان ، بمعنى المتغير الأول يساوي 7 والمتغير الثاني يساوي 5 نفذ أمر الطباعة واطبع صحيح :

```
<?
$x = 7;
$y = 5;
if (($x == 7) and ($y == 5)) print "صحيح";
// يتم طباعة صحيح
?>
```

(أو) Or

إذا كان أحد الشرطين صحيح أو الاثنين صحيحين نفذ أمر الطباعة :

```
<?
$x = 7;
```

```
$y = 5;
if (($x == 7) or ($y == 8)) print "True";
// True  يطبع
?>
```

Xor

إذا تحقق أحد الشرطين وليس الاثنين معا ينفذ أمر الطباعة :

```
<?
$x = 7;
$y = 5;
if (($x == 7) xor ($y == 8)) print "True";
// True تحقق شرط واحد فقط فيتم طباعة كلمة
?>
```

! (**النفي**)

إذا كانت جملة الشرط غير صحيحة نفذ أمر الطباعة :

```
<?
$y = 5;
if (! ($y == 10)) print "True";
لأن المتغير القيمة المخزنة فيه غير صحيحة True يطبع
?>
```

&&

المعامل && له نفس وظيفة (and) لكن الاختلاف في ترتيب تنفيذ أولويات العمليات :

```
<?
$x = 7;
$y = 5;
if (($x == 7) && ($y == 5)) print "True";
// True يطبع
?>
```

||

المعامل || له نفس وظيفة (or) لكن الاختلاف في ترتيب تنفيذ أولويات العمليات :

```
<?
$x = 7;
$y = 5;
if (($x == 7) || ($y == 5)) print "True";
// True يطبع
?>
```

لاهمية موضوع قواعد البيانات ، سوف نقوم في هذه الدورة بتغطية دوال قواعد البيانات وهي اثنتان وثلاثون دالة فإلى الدرس الأول :

1- **الدالة** mysql_connect :

```
integer mysql_connect(string host, string username,
string password);
```

تقوم هذه الدالة بالاتصال مع قاعدة البيانات وتعيد لك رقم يفيدك اذا كان لديك أكثر من اتصال بقواعد البيانات ، احتفظ به لاستخدامه في دوال أخرى تالية اذا كان هناك حاجة لذلك كما قلنا ، اما الوضع الطبيعي فلا يحتاج الا الى الاتصال بالطريقة السابقة فقط وبدون الاحتفاظ بأي رقم ، فقط مرر للدالة اسم الخادم واسم المستخدم وكلمة المرور ، ولكن يتوجب عليك بعد الانتهاء أن تغلق الاتصال باستخدام الدالة mysql_close

مثال :

```
<?
$link = mysql_connect("db.azzozhsn.f2s.com","mag","Pass");
?>
```

2- **الدالة** mysql_pconnect :

```
integer mysql_pconnect(string host, string username,
strin password);
```

هذه الدالة تقوم بما تقوم به الدلة السابقة إلا أنه لا يتوجب عليك إغلاق الاتصال ، مثال:

```
<?
$link = mysql_pconnect("db.azzozhsn.f2s.com","mag","Pass");
?>
```

3- **الدالة** mysql_select_db :

```
boolean mysql_select_db(string database, integer link);
```

تقوم هذه الدالة باختيار قاعد البيانات المحدد لها. مثال:

```
<?
mysql_select_db(string database, integer link);
?>
```

4- **الدالة** mysql_db_query :

```
boolean mysql_db_query(string database, string query,
integer link);
```

تقوم هذه الدالة بتنفيذ سطر SQL على قاعدة البيانات المفتوحة بالمعطى database مثال:

```
<?
$link = mysql_connect("db.azzozhsn.f2s.com","mag","Pass");
$Query = "DELETE FROM magazine";
$result = mysql_db_query("mag", $Query, $link);
?>
```

5- **الدالة** mysql_close :

```
boolean mysql_close(integer link);
```

تقوم هذه الدالة بقطع (إغلاق) قاعدة البيانات ، مرر لها رقم الاتصال المعاد من الدالة mysql_connect
مثال:

```
<?
// البيانات الاتصال بقاعدة ..
$link = mysql_connect("localhost","mag","Pass");
// اغلاق الاتصال بقاعدة البيانات ..
mysql_close($link);
?>
```

6- **الدالة** mysql_query:

```
integer = mysql_query(string query, integer link);
```

تقوم هذه الدالة بما تقوم به الدالة mysql_db_query تقريباً إلا أن الدالة mysql_query يقتصر عملها على قاعدة البيانات المحددة بالدالة mysql_select_db .
في حالة عدم تمرير رقم الاتصال فستعمل الدالة على الاتصال الأخير.
مثال:

```
<?
$link = mysql_connect("localhost","mag","Pass");
$query = "DELETE FROM magazine";
$result = mysql_query($query, $link);
?>
```

7- **الدالة** mysql_errno :

```
integer mysql_errno(integer link);
```

تقوم هذه الدالة بإعادة رقم آخر خطأ حدث في التعامل مع قاعدة البيانات.

8- **الدالة** mysql_error :

```
string mysql_error(integer link);
```

تعيد هذه الدالة رسالة الخطأ الحاصل في قاعدة البيانات .

9 - **الدالة** mysql_create_db :

```
boolean mysql_create_db(string databasename, integer link);
```

تقوم هذه الدالة بإنشاء قاعدة بيانات جديدة مرر لها اسم قاعدة البيانات ورقم الاتصال العائد من الدالة mysql_connect أومن الدالة mysql_pconnect ..
مثال:

```
<?
// الإتصال بقاعدة بيانات اسمها az الباسورد حيث أن الفراغ هو
$link = mysql_pconnect("localhost", "az", "");
// انشاء قاعدة بيانات جديدة
if (! mysql_create_db($link, "mag"))
{
   print("الجديدة فشل إنشاء قاعدة البيانات")
   exit();
}
?>
```

10- **الدالة** mysql_drop_db :

```
boolean mysql_drop_db(string databasename, integer link);
```

تقوم هذه الدالة بحذف قاعدة البيانات المحددة بالمعطى databasename ..

11- **الدالة** mysql_list_dbs :

```
integer mysql_list_dbs(integer link);
```

تقوم هذه الدالة بإعادة موشر لكل قواعد البيانات الموجودة في الخادم لغرض استعمالها مع الدالة mysql_fetch_row وأمثالها .

12- **الدالة** mysql_field_seek :

```
boolean mysql_field_seek(integer result, integer field);
```

تقوم هذه الدالة بتحديد الحقل المرر إليها رقمه . مثال :

```
<?
// الإتصال بقاعدة بيانات اسمها az الباسورد حيث أن الفراغ هو
$dbLink = mysql_pconnect("localhost","az","");
//  Authors أختيار قاعدة البيانات
mysql_select_db("Authers", $dbLink);
// Adress اختيار جميع الحقول من الجدول
$Query = "SELECT * FROM adress";
$result = mysql_query($Query, $dbLink);
// الانتقال الى الحقل الثاني اعتماداً على عملية الاختيار
mysql_field_seek($reslut, 1);
?>
```

13- **الدالة** mysql_field_name :

```
string mysql_field_name(integer result, integer feild);
```

تعيد هذه الدالة اسم الحقل المحدد بالرقم المرر إليها والذي يبدأ بالرقم صفر للحقل (العمود) الأول .
مثالها سيأتي بعد قليل .

14- **الدالة** mysql_field_type :

```
string mysql_field_type(integer result, integer feild);
```

تعيد هذه الدالة نوع الحقل المحدد بالرقم المرر إليها والذي يبدأ بالرقم صفر للحقل (العمود) الأول . المثال سيأتي بعد قليل أيضاً ..

15- **الدالة** mysql_field_len :

```
string mysql_field_len(integer result, integer feild);
```

تعيد هذه الدالة طول الحقل بالبايت المحدد بالرقم المرر إليها والذي يبدأ بالرقم صفر للحقل (العمود) الأول . المثال بعد قليل ..

16- **الدالة** mysql_field_flags :

```
string mysql_field_flags(integer result, integer feild);
```

تعيد هذه الدالة وصف الحقل المحدد بالرقم المرر إليها والذي يبدأ بالرقم صفر للحقل (العمود) الأول .

17- **الدالة** mysql_list:

```
mysql_list(string database, string table, integer link);
```

المثال الشامل :

```
<?
// الإتصال بقاعدة بيانات اسمها az الباسورد حيث أن الفراغ هو
$link = mysql_pconnect("localhost", "az", "");
// ترتيب الحقول وجلبها
$result = mysql_list_field("mag", "table", integer link);
// حلقة تكرار للمرور على كل حقل
for ($a = 0; $a < mysql_field_num($result); $a++)
{
   print(mysql_field_name($result, $i);
   print(mysql_field_type($result, $i));
   print(mysql_field_len($result, $i));
   print(mysql_field_flags($result, i));
}
?>
```

18- **الدالة** mysql_fetch_field :

```
<?
object mysql_fetch_field(integer result, integer field);
?>
```

استخدم هذه الدالة لتحصل على معلومات حول حقول الجدول المراد، الحقول ترقم بدايةً من صفر وصف الحقل مشروح في الجدول التالي:

الخاصة	الوصف
blob	إذا كانت TRUE فالحقل عبارة عن عن حقل بيانات كبير
maxlength	الطول الأقصى للحقل
multiple_key	تكون TRUE إذا كان الحقل مفتاحاً
name	أسم الحقل
not_null	تكون TRUE إذا كان الحقل لا يمكن أن يكون فارغاً
numric	تكون TRUE إذا كان الحقل يرقم تلقائياً
primary_key	تكون TRUE إذا كان الحقل يمثل مفتاحاً رئيساً

unque_key	تكون TRUE إذا كان الحقل يمثل مفتاحاً ثانوياً
zerofill	تكون TRUE إذاكان الحقل يملأ بالقيمة 0

19 - **الدالة** mysql_fetch_lengths :

```
<?
array mysql_fetch_lengths(integer result);
?>
```

استخدم هذه الدالة لتعيد مصفوفة تحتوي على الطول الأقصى لكل حقل محدد في المعطيresult.

```
<?
//Connect to server as azzozhsn no password
$link = mysql_pconnect("localhost","azzozhsn","");
//Select th magazine database
mysql_select_db("magazine",$link);
//Get name and id from magazine
$Query = 'SELECT name, id FROM magazine';
$result = mysql_query($Query, $link);
$length = mysql_fetch($result);
//Print length of the third column
print($lengths[2]);
?>
```

20 - **الدالة** mysql_fetch_array :

```
<?
array mysql_fetch_array(integer result);
?>
```

هذه الدالة تعيد مصفوفة تحتوي على قيم سجل وتنقل المؤشر إلى السجل التالي. مثال:

```
<?
//Connect to server as azzozhsn no password
$link = mysql_pconnect("localhost","azzozhsn","");
//Select th magazine database
mysql_select_db("magazine",$link);
//Get name and id from magazine
$Query = 'SELECT name, id FROM magazine';
$result = mysql_query($Query, $link);
//Get every row
while($row=mysql_fetch_array($result, MYSQL_ASSOC)){
 //Print mane and id
    print({$row["id"]}={$row["name"]});
}
?>
```

21- **الدالة** mysql_fetch_object :

```
<?
object mysql_fetch_object(integer result)
?>
```

هذه الدالة تشبه الدالة mysql_fetch_array إلا أنها تعيد كائن. عند استدعاء الدالة ينتقل المؤشر إلى السجل التالي في الجدول، وإذا ةصل إلى نهاية الجدول ثم استدعيت الدالة مرة أخرى فإنها تعيد القيمة FALSE مثال:

```
<?
while($row=mysql_fetch_object(result)){
 //print id and name
```

```
   print ("$row->id, $row->name")
}
?>
```

22- **الدالة** mysql_fetch_row :

هذه الدالة تعيد مصفوفة تحتوي على قيم حقول سجل من الجدول وكل استدعاء يعيد قيمة الحقول في السجل التالي في الواقع هذه الدالة تشبه الدالتين السابقتين. مثال:

```
<?
while($row=mysql_fetch_row(result)){
 //print id and name
   print ("$row[0], $row[1]")
}
?>
```

23- **الدالة** mysql_change_user :

```
<?
musql_change_user(string user, string password, string db, integer link);
?>
```

استخدم هذه الدالة لتغيير مستخدم قاعدة بيانات المتصل بها.
المعطيان db, link اختيارية وفي حالة فقدهما يستعاض عنهما بالاتصال الحالي.
هذه الدالة تتطلب إصدار MySQL 3.23.3 أو ما بعدها.

بسم الله الرحمن الرحيم والصلاة والسلام على اشرف الانبياء والمرسلين ، في هذا الدرس احببت أن اجيب على تسآول كثيراً ما يطرح في المنتديات وهو عن التاريخ باللغة العربية ، واتمنى من الله التوفيق في طرح الموضوع بشكل مبسط وفي متناول الجميع ..

في البداية وكمقدمة للموضوع أود الإشارة الا أن الطريقة المشروحة في هذا الدرس تعتمد على التاريخ الميلادي ، وبالتاكيد يمكن استخدامها مع التاريخ الهجري ولكن تحتاج الى تعديل وحسابات خاصة ، وباذن الله سيتم شرحها في المستقبل .

دالة التاريخ في البي اتش بي هي Date ، ولها معاملين (أي قيمتين لاعداد مخرجات الدالة) ، احد المعاملين اجباري والثاني اختياري ، اما الاول وهو الاهم تعتمد عليه مخرجات التاريخ بشكل اساسي مثل ضبط السنة بخانتين أو ضبط الشهر باسم الشهر .. وغيرها ، اما المعامل الثاني فهو ما يسمى بـ (UNIX time stamp) وهو خاص بنظام اليونكس وكيفية تخزين التاريخ فيه ، عموماً ما يهمنا هنا هو المعامل الأول وهو ما يسمى بـ (Format String) ، وكمثال على ما ذكرنا :

```
<?
$today = date(Y-m-d);
echo $today;
?>
```

هذا المثال سيقوم بطباعة تاريخ اليوم على الشكل التالي 2002-03-13 ، ولاهمية الرموز التي يمكن استخدامها مع الـ Date ساذكر اهمها :
d رقم اليوم في الشهر على شكل خانتين من 01 الى 31 .
D اسم اليوم في الاسبوع على شكل 3 خانات مثل Mon أي الاثنين .
g رقم الساعة في اليوم من 1 الى 12 .
j رقم اليوم في الشهر من 1 الى 31 بدون وضع الصفر .
m رقم الشهر في السنة على شكل خانتين من 01 الى 12 .
y رقم السنة على شكل خانتين ، مثلا 02 .
Y رقم السنة على شكل اربع خانات ، ومثالها 2002 .

هذه من اهم الرموز لكي تتضح الصورة فقط ، ولعلنا نتطرق لها بشكل اوسع قريباً .

لتحويل التاريخ الى اللغة العربية نحتاج أن ننشئ جدولاً في قاعدة البيانات ، فلذلك قم بنسخ الكود التالي والصقه في خانة Run SQL query في الـ PHPMyadmin أو بأي طريقة اخرى تراها ، الاهم انشاء الجدول .

```
CREATE TABLE month_name (
id tinyint(4) NOT NULL default '0',
month text NOT NULL
) TYPE=MyISAM;

INSERT INTO month_name VALUES (1, 'يناير');
INSERT INTO month_name VALUES (2, 'فبراير');
INSERT INTO month_name VALUES (3, 'مارس');
INSERT INTO month_name VALUES (4, 'ابريل');
INSERT INTO month_name VALUES (5, 'مايو');
INSERT INTO month_name VALUES (6, 'يونيو');
INSERT INTO month_name VALUES (7, 'يوليو');
INSERT INTO month_name VALUES (8, 'اغسطس');
INSERT INTO month_name VALUES (9, 'سبتمبر');
INSERT INTO month_name VALUES (10, 'اكتوبر');
INSERT INTO month_name VALUES (11, 'نوفمبر');
INSERT INTO month_name VALUES (12, 'ديسمبر');
```

بعد انشاء هذا الجدول يجب أن يكون لديك جدول اخر يحتوي على التاريخ المراد تحويله ، ولنفترض أن لديك الجدول (news) يحتوي على الحقول (title , date) ويحتوي على البيانات التالية :

date	title
20-04-2002	الخبر الأول
25-04-2002	الخبر الثاني
01-05-2002	الخبر الثالث

قم بانشاء الجدول :

```
CREATE TABLE news (
  title text NOT NULL,
  date date NOT NULL default '0000-00-00'
) TYPE=MyISAM;


INSERT INTO news VALUES ('20-04-2002','الخبر الأول');
INSERT INTO news VALUES ('25-04-2002','الخبر الثاني');
INSERT INTO news VALUES ('01-05-2002','الخبر الثالث');
```

بقي أن نقوم بتحويل التاريخ الى العربية ، وادراجه في صفحة PHP ، ولعمل ذلك سنقوم باستخدام دالة تسمى Date_Format من خلال طلب لقاعدة البيانات ، نحدد من خلاله طريقة جلب البيانات ووضعها بالصورة المطلوبة .

بقي أن نذكر أننا سوف نضطر الى كتابة طلبين لقاعدة البيانات احدهما لجلب حقول العنوان (title) والاخر لجلب حقول التاريخ (date) كما يلي :

```
<?
$result = mysql_query("select * from news");
$sql = "SELECT CONCAT(DATE_FORMAT(date,'%d'),\"
\",month_name.month,\" \",DATE_FORMAT(date,'%Y'))
AS date FROM news ,month_name
WHERE month_name.id = month(date)";
$result2 = mysql_query("$sql");
while ($row=mysql_fetch_array($result)
and  $row2=mysql_fetch_array($result2))
```

```
{
$title = $row["title"];
$date = $row2["date"];
echo "$title , $date<br>";
}
?>
```

عند تنفيذ السكربت ، سترى ما يلي :

الخبر الأول , 20 ابريل 2002
الخبر الثاني , 25 ابريل 2002
الخبر الثالث , 01 مايو 2002

في حالات كثيرة تكون كتابة السكربت السابق بهذا الشكل مسببه للكثير من المشاكل ، و خاصة عند طلب ترتيب للجدول على حسب حقل معين ، وهذه المشاكل هي في توافق البيانات مع بعضها البعض ، فلو افترضنا في مثالنا السابق أن الخبر الأول الذي يحمل التاريخ 2002-04-20 كان باسم آخر ، مثلا (العنوان الأول) ، وبعد اضافة حقول ترتيب لجلب البيانات كالتالي :

```
<?
$result = mysql_query("select * from news
order by title");
$sql = "SELECT CONCAT(DATE_FORMAT(date,'%d'),\"
\",month_name.month,\" \",DATE_FORMAT(date,'%Y'))
AS date FROM news ,month_name
WHERE month_name.id = month(date)";
$result2 = mysql_query("$sql");
while ($row=mysql_fetch_array($result)
and  $row2=mysql_fetch_array($result2))
{
$title = $row["title"];
$date = $row2["date"];
echo "$title , $date<br>";
}
?>
```

ستجد أن النتائج هي :

الخبر الثالث , 20 ابريل 2002
الخبر الثاني , 25 ابريل 2002
العنوان الأول , 01 مايو 2002 وهذا بطبيعة الحال مشكلة في توافق البيانات .

ولحلها يجب أن نوافق بين الطلبين لقاعدة البيانات ، بمعنى أنه اذا رتبنا الطلب الاول حسب (title) يجب أن نفعل ذلك مع الطلب الثاني بتعديله ليصبح :

```
<?
$sql = "SELECT CONCAT(DATE_FORMAT(date,'%d'),\"
\",month_name.month,\" \",DATE_FORMAT(date,'%Y'))
AS date FROM news ,month_name
WHERE month_name.id = month(date)
order by title";
 ?>
```

وبالتالي تصبح البيانات المخرجه كالتالي :

الخبر الثالث , 01 مايو 2002
الخبر الثاني , 25 ابريل 2002
العنوان الأول , 20 ابريل 2002

وهي بالتاكيد صحيحة .

بسم الله الرحمن الرحيم والصلاة والسلام على اشرف الانبياء والمرسلين ، بعد المقدمات السابقة والهامة في معرفة اساسيات اللغة يمكننا بداية كتابة البرامج بلغة البي اتش بي ، وبطبيعة الحال سنبدأ من اصغر الاساسيات واهمها في كتابة البرامج عموماً وهي المتغيرات .

المتغيرات في لغة الـ PHP تبدأ بعلامة الدولار ($) ، ولاسناد قيمة لذلك المتغير نستخدم علامة المساواة (=) ، فرضا لدينا المتغير (Name) والقيمة (Khaled) فنكتب ما يلي :

```
<?
$Name = "Khaled";
?>
```

هذا في حالة المتغيرات النصية (Text) ، وفي حالة المتغيرات الرقمية (Numbers) يمكن تعريف متغير (Counter) الذي يحمل القيمة (17) كالتالي :

```
<?
$Counter = 17;
?>
```

الفرق الواضح في طريقة تعريف المتغيرين النصي والرقمي هو عدم وجود علامات التنصيص في تعريف المتغيرات الرقمية بينما يجب وضع علامات التنصيص في تعريف المتغيرات النصية .

نقاط هامة في تسمية المتغيرات :

- اسماء المتغيرات في كثير من لغات البرمجة لا تتعدى 255 حرف (المقصود بها الخانات سواء كانت حروف أو ارقام أو علامات اخرى) ، و في لغة الـ PHP لا يوجد حدود على عدد الخانات في تسمية المتغيرات ، ولكن في الغالب لن تحتاج الى اكثر من 15 خانة لتسمية أي متغير ، لان المبالغة في تسمية المتغيرات تسبب مشاكل في تذكر المتغيرات وما تحتوية من قيم .

- بداية كل متغير يجب أن تبدأ بحرف (يعني حرف هجائي) أو علامة (_) Underscore ، مع تجاهل علامة الـ $ لانها لا تحسب من اسم المتغير .

- يمكن أن يحتوي اسم المتغير على الحروف أو الارقام أو علامة (_) فقط ، اما العلامات الأخرى مثل (+ , - , * , /) أو الـ & لا يمكن كتابتها في اسم المتغير .

- المتغير ($Name) يختلف عن المتغير ($name) لاختلاف حالة حرف الـ N ، ولذلك يجب التاكد من اسم المتغيرات بدقة لتجنب حدوث مشاكل في الوصول الى متغير معين ، وبالتاكيد لو كان لديك اسلوب خاص في تسمية المتغيرات لسهولة الوصول اليها وتذكرها ستكون كتابة السكربات اسهل بكثير .

- يستحسن أن تكون اسماء المتغيرات دالة على معانيها ، بمعنى أنه لمتغير مثل عداد الزوار يستحسن أن يكون ($counter) ، ولمتغير مثل اسم المستخدم ($user) .. الخ .

التعامل مع المتغيرات :

فائدة المتغيرات تكمن في طريقة استخدامها في كتابة السكربت ، وكما ذكرنا سابقاً أنه لطباعة متغير معين نستخدم امر الطباعة (echo) أو (print) كما يلي :

```
<?
$name = "Naser";
echo $name;
?>
```

في البداية سيتم اسناد القيمة (Naser) الى المتغير ($name) ، وفي السطر الثاني يتم طباعة المتغير ، أو بالاحرى القيمة المسندة الى المتغير .

انواع البيانات (Data Types) :

في الامثلة السابقة قمنا باسناد قيمتين عددية ونصية الى متغيرين ، وبينًا الفرق بينهما ، وفي لغة الـ PHP بشكل عام يوجد أكثر من هذين النوعين من البيانات ، ساشرح بعضاً منها الآن ، والبقية في الدروس القادمة :

- البيانات النصية (String) .
- البيانات العددية الصحيحة (Integer) .

- البيانات العددية الكسرية (Double) .
- المصفوفات (Array) .
- الكائنات (Object) .
- البيانات الغير معروفة ! .

البيانات النصية (String) :
هي البيانات التي تكون بين علامات التنصيص " " بغض النظر عن محتوها ، فيمكن أن تكون حروف أو اعداد أو رموز أو غيرها ، ومثال ذلك كما ذكرنا سابقاً :

```
<?
$user = "Khaled";
$age = "13.5";
?>
```

التعامل مع البيانات النصية (String) :
لاضافة المتغيرات التي تحتوي على بيانات نصية مع متغيرات من نفس النوع نحتاج الى عملية دمج بين المتغيرات ، ولعمل ذلك نكتب :

```
<?
$total = $user . $age;
?>
```

في هذه الحالة سيتم اسناد القيمة Khaled13.5 الى المتغير ($total) .
اذا اردنا وضع مسافة بين المتغيرين نضيف متغير جديد يحتوي على المسافة وهو ($space) ثم نقوم بعملية الدمج كالتالي :

```
<?
$space = " ";
$total = $user . $space . $age;
?>
```

وفي هذه الحالة سيتم وضع القيمة Khaled 13.5 في المتغير ($total) ، وبطبيعة الحال يمكن استخدم المتغيرات النصية داخل متغيرات نصية أخرى ، حيث سيتم تعويض المتغير بقيمتة الأصلية .

البيانات العددية (Numeric) :
وكما ذكرنا في التقسيم السابق أنها نوعين (الاعداد الصحيحة Integer) و (الاعداد الكسرية Double) ، وكمثال على النوعين :

```
<?
$integer1 = 233;
$integer2 = -29
$double1 = 5.27
$double2 = -4.6
?>
```

التعامل مع البيانات العددية (Numeric) :
العمليات الحسابية المشهورة (+ , - , * , /) بالاضافة الى باقي القسمة (%) عمليات شائعة جداً في التعامل مع المتغيرات العددية ، وبطبيعة الحال لن نحتاج الى ذكر أي مثال عن هذه العمليات ، وسنكتفي بذكر بعض النقاط الاساسية التي قل ما يخلو سكربت منها .

اول النقاط هي اضافة المتغير الى نفسه ، بمعنى تعريف عملية حسابية على متغير معين بحيث تخزن القيمة في نفس المتغير ، مثلا لو كان لديك عدد الزوار وتريد في كل مرة أن يزود عدد الزوار بـ 1 ، يمكنك كتابة ما يلي :

```
<?
$counter = $counter + 1;
?>
```

بالتالي سيتم زيادة المتغير ($counter) بـ 1 في كل مرة يتم فيها تنفيذ السكربت ، وبطريقة أخرى يمكن كتابة السطر السابق كالتالي :

```php
<?
$ccounter = $counter++;
?>
```

والـ ++ تعني زيادة قدررها (1) على قيمة المتغير الأصلية ، وكذلك الـ -- تعني طرح 1 من القيمة الأصلية .

وفي حالة الرغبة بزيادة أي عدد أخر (غير الواحد) على أي متغير باسلوب الطريقة الثانية يمكن كتابة ما يلي :

```php
<?
$counter +=4;
?>
```

وهذا يعني زيادة مقدارها 4 على قيمة المتغير الأصلية ، وبالسالب كذلك بنفس الأسلوب .

ترتيب انجاز العمليات الحسابية :

يوجد بعض الرموز والعمليات التي تسبق غيرها عند البدء في انجاز عملية حسابية معينة ، والترتيب المستخدم في البي اتش بي كالتالي :

] -
(int) (double) (string) (array) (object) -- ++ ~ ! -
% / * -
. - + -
<< >> -
> => < =< -
== =! === ==! -
& -
| -
&& -
|| -
? : -
= =+ =* =/ =. =% =& =^ =~ =>> =<< -
print -
AND -
XOR -
OR -
, -

بالتأكيد القائمة طويلة وفيها تفاصيل كثيرة ، ولكن من المهم معرفة طريقة انجاز العمليات الحسابية المختلفة لسهولة اكتشاف الأخطاء ومعرفة الطريقة الصحيحة لكتابة بعض العمليات المعقدة للحصول على ناتج صحيح .

بعض الدوال الهامة في التعامل مع المتغيرات :

- isset : وهي دالة للتأكد من وجود متغير معين ، فمثلا :

```php
<?
echo isset($age);
?>
```

سيتم طباعة الرقم 1 اذا كان المتغير ($age) موجوداً (تم انشاءه مسباقاً) ، والعكس اذا كان غير موجود سيتم طباعة الرقم 0 ، وهذه الدالة يتم استخدامها كثيراً في الشروط وهذا ما سنتطرق اليه لاحقاً .

- unset : هذه الدالة تعمل على مسح المتغير من الذاكرة كلياً ، فقط قم بعمل التالي :

```php
<?
unset($age);
?>
```

وفي هذه الحالة سيتم مسح المتغير ($age) بشكل كامل .

- empty : وهذه الدالة معاكسة للدالة isset بحيث لو كتبنا ما يلي :

```
<?
echo empty($age);
?>
```

سيتم طباعة الرقم 1 في حالة عدم وجود المتغير ($age) أو أن قيمة المتغير تساوي 0 أو (فراغ) ، وفي حالة وجود المتغير ($age) لن يتم طباعة أي شي .
بسم الله الرحمن الرحيم والصلاة والسلام على اشرف الانبياء والمرسلين اما بعد ، في هذا الدرس بمشيئة الله سنتحدث عن مقدمة للتحكم في المواقع عن طريق الـ Session أو الجلسات كما اصطلح على تسميتها ، ففي البداية سنتعرف على الـ Session وعن التحكم فيها ، ومن ثم استخداماتها بالاضافة الى بعض الامثلة ، وفي النهاية سنتطرق الى بعض الأخطاء في كتابة الـ Session وحلول تلك الأخطاء ، وفي الدرس القادم بإذن الله تعالى سنتطرق الى مثال كامل للوحة تحكم مبسطة تتعامل بالـ Session ، والأمل أن يكون في هذا الشرح المبسط فائدة للجميع ..

- مقدمة عن الـ Session :

عند الانتقال من صفحة الى أخرى في موقع معين فإن بروتوكول الـ HTTP لا يمكنه معرفة أن تلك الصفحات قد تم تصفحها من قبل نفس الشخص ، ولكن مع الـ cookies وما نحن بصدده هنا الـ Session تقدم تلك الطريقة ، ولذلك وببساطة فإن الـ Session هي مكان على جهاز المتصفح يمكن من خلاله تخزين قيمة معينة للرجوع اليها في حال قام نفس الشخص بالانتقال من صفحة الى أخرى ، ولعل هذا التعريف يصف ببساطة معناها العام ولا يعني ذلك أنه تعريف شامل لكل المعاني ..

إذاً التعرف على الشخص الذي يقوم بتصفح الموقع هو الهدف الرئيسي للـ Session أو الجلسات ، ولكن كيف يتم ذلك ، وما هي النقاط الرئيسية التي يجب معرفتها لفهم طريقة التعامل مع الـ Session ؟

أول تلك النقاط أن عملية تسجيل المتغير على جهاز المستخدم له مدة معينة تنتهي بانتهاء الجلسة ، ومن هنا جاءت التسمية ، أما ما تعنيه الجلسة فهي مصطلح لقيامك بالتصفح من الموقع ومن ثم اغلاق الموقع ، ببساطة كل مرة تقوم بزيارة الموقع تبدأ جلسة أو Session جديدة ، مع ملاحظة أن هناك طرق للتحكم بوقت الانتهاء كما في الـ cookies ، بالاضافة الى طرق اخرى عن طريق قواعد البيانات وهو حديث سابق لاوانه .

بالنسبة للنقطة الأخرى التي يجب وضعها في الحسبان هي ما يسمى بالـ ID Session أو اختصاراً SID ويعني ذلك (رقم الجلسة) ، وهو رقم عشوائي فريد يصعب تكراره أو فلنقل أنه مستحيل لاحتوائه على ارقام واحرف كبيرة وصغيرة في متغير طويل نسبياً ، وهذه القيمة هي الأهم في ما ذكرت ، لإنها القيمة الوحيدة التي تربط ما يسمى بالـ Session Variables أو (متغيرات الجلسة) مع جهاز المستخدم ، فالـ SID هي القيمة الوحيدة التي يتم تخزينها في جهاز المستخدم (Client) ، أما الـ متغيرات الجلسة Session Variables يتم تخزينها في السيرفر (Server) ، فعند التحقق منه وجود هذه القيمة على جهاز المستخدم يمكن الدخول الى المتغير الآخر المتربط به والمسمى بالـ Session Variable .

النقطة الثالثة هي طريقة التخزين للـ SID و الـ Variables Session ، أما الـ SID وكما قلنا أنها تخزن على جهاز العميل (Client) إما عن طريق الـ cookies والتي لها سلبياتها المتعددة أو عن طريق تمريرها عبر الـ HTTP ، أما بالنسبة للـ Session Variables فيتم تخزينها في ملفات فارغة على جهاز الـ Server وكذلك في مستويات متقدمة يمكن التحكم بها وتخزينها في قواعد بيانات .

- إعدادت الـ Session :

عن طريق ملف الـ php.ini والذي يحتوي على إعدادت الـ PHP يمكن التحكم باعدادات الـ Session ، وكاستعراض لتلك النقاط المتحكمة بالـ Session سنتطرق أهم النقاط ومعانيها ، وللوصول الى ما نحن بصدده تذكر أن ملف الـ php.ini يوجد في دليل الـ Windows ، وللوصول الى خصائص الـ Session إبحث عنه كلمة (Session) وستجد السطر التالي :

[Session]

من هنا تستطيع التحكم بخيارات الـ Sessions ، وكما يظهر في الجدول التالي وصف لأهم الخيارات ..

1- الخيار Session.auto_start :
بداية تلقائية للـ Session (دون الحاجة لعمل ذلك يدوياً عن طريق Session_start) .

2- الخيـار Session.cache_expire :
وقت انتهاء الجلسة بالدقائق .

3- الخيـار Session.cookie_lifetime :
وقت انتهاء الـ cookie المرتبطة بالجلسة ، وهي افتراضياً ستكون 0 أي أن الـ cookie ستنتهي فترتها مع اقفـال الشخص المتصفح للموقع .

4- الخيـار Session_name :
إسم الـ Session التي ستستخدم كـ cookie وافتراضياً ستكون PHPSESSID .

5- الخيـار session.save_path :
هذا السطر يعني مكان تخزين ملفات الـ Session في جهازك باعتباره سيرفر ، وهنا تستطيع أن تضع أي عنوان في جهازك ، أما تركه فارغاً فيعني عدم تفعيل الـ Session لديك ، بالنسبة لي أقترح أن يكون المجلد Temp داخل الـ Windows هو الدليل الأمثل لاحتوائه على ملفات مؤقتة يمكن حذفها ، اذا العنوان سيكون c:\windows\Temp

هذه في نظري أهم الخيـارات التي يجب فهمها ،

- بداية الـ Session :
قبل أن تستخدم أياً من دوال الـ Session يجب اخبار السكربت أن يبدأ جلسة Session ، والطريقة هي أن تضع في بداية السكربت وفي أول سطر فيه بعد علامة الفتح ما يلي :

```
<?
session_start();
?>
```

في هذه الحالة فقط يمكن أن تقوم باستخدام دوال الـ Session الأخرى ، أما اذا لم يتم كتابة هذا السطر فلن يتم ذلك .

ملاحظة مهمة حول عملية بداية الـ Session وهي أن تتأكد من أن هذا السطر لا يسبقه عملية اخراج مخرجات ، بمعنى أخرى أي استخدام لدوال مثل echo أو print ، وكذلك لا يسبق هذا السطر أي فراغ وتأكد من هذه النقطة جيداً لانها كثيرة الحدوث وتعطى الخطأ التالي :

وأسلم طريقة من وجهة نظري أن تضع هذا السطر في بداية ملف الـ header لانك سنقوم بادراج هذه الصفحة في كل الصفحات الأخرى وبالتالي يكون السطر هو الأول في كل الحالات ..

- تخزين متغيرات الجلسات :
وهي ما نسميها بالـ Session Variables ، ولعمل ذلك يوجد لدينا الدالة الواردة في المثال التالي :

```
<?
$user = "AbdulAziz";
session_register("user");
?>
```

ما قمنا بعمله هو التالي :
1- عرفنا متغيراً هو user يحتوي على قيمة حرفية .
2- قمنا بتسجيل هذا المتغير في متغير جلسة (Session Variable) وبنفس الاسم user ولكن بدون علامة $.

- التعامل مع متغيرات الجلسة :
بعد تسجيل المتغير ، يمكن الرجوع اليه بعدة طرق تعتمد على الخيار register_globals في ملف الـ php.ini ، أما اذا كان on وهذا هو الاختيار الإفتراضي فإن المتغير الذي تم تسجيله في الـ Session يمكن

الرجوع اليه كأي متغير آخر ، عن طريق اسم المتغير فقط ، وفي مثالنا الحالي سيكون $user ، أما اذا كان الخيار غير مفعل وليس بالصورة التي ذكرتها فيمكن الرجوع الى المتغير عن طريق الأمر $HTTP_SESSION_VARs["user"] .

أيضا كنقطة مهمة يجب معرفتها وهي طريقة التحقق من أن متغيراً معيناً قد تم تسجيله أم لا ، وهذه الطريقة مفيدة في الصفحات التي يجب أن يكون فيها المستخدم قد سجل الدخول وبالفعل تمت عملية تسجيل الـ Session له ، في المثال التالي تلك الطريقة :

```
<?
if (session_is_registered("user")) {
echo "أهلا وسهلا بكم في قرية بي اتش بي";
}
else {
echo "لا يسمح لك بالدخول ..";
}
?>
```

في هذا المثال سيتم عرض الجملة (أهلا وسهلا بكم في قرية بي اتش بي) إذا كان عملية تسجيل الـ Session تمت للمتغير user ، وسيتم عرض الجملة (لا يسمح لك بالدخول ..) في حالة عدم تسجيل الـ Session .

نقطة أخيرة في التعامل مع متغيرات الجلسة ، وهي عملية الغاء تسجيل الـ Session لمتغير معين ، وهذه الطريقة تتم عن طريق الدوال session_unregister و session_unset و session_destroy ، أما الفرق بينهم فهو أن الدالة الأولى تقوم بعملية الغاء التسجيل لـ Session معينة ، أي بتمرير إسم المتغير لها كما في المثال التالي :

```
<?
session_unregister("user");
?>
```

اذا سيتم الغاء تسجيل الـ Session المتعلقة بالمتغير user فقط ، أما الدالة الثانية فستقوم بالغاء تسجيل جميع الـ Session التي تم تسجيلها من قبل ، وفي النهاية يجب أستخدام الدالة الثالثة session_destroy لالغاء الـ SID والانتهاء من التعامل مع الـ Session .

- مثال بسيط عن الـ Session :

سأطرق الى مثال بسيط جداً لتوضيح كيفية عمل الـ Session ، في البداية قم بوضع الكود التالي في ملف وقم بتسميته phpex1.php :

```
<?
$age = 12;
session_register("age");
echo "$age تحتوي على القيمة age الـجلسة <br>";
echo "<a href=phpex2.php>التـالي ..</a>";
?>
```

الصفحة الثانية احفظها بإسم phpex2.php ، وضع الكود التالي فيها :

```
<?
echo "أنت في الصفحة الثانية<br>";
echo "$age تحتوي على القيمة age الـجلسة <br>";
session_unregister("age");
echo "<a href=phpex3.php>التـالي ..</a>";
?>
```

الصفحة الثالثة تحتوي على الكود التالي ، واسمها phpex3.php :

```
<?
echo "أنت في الصفحة الثالثة<br>";
echo "$age تحتوي على القيمة age الـجلسة <br>";
?>
```

ابدأ من الصفحة الأولى ومن ثم انتقل من صفحة الى أخرى ، حتى تصل الى الثالثة ، بافتراض أنك قمت بتجربة المثال ، ستلاحظ أن الصفحة الأولى سيتم طباعة الـ Session التي تم تسجيلها وهي age

وستظهر القيمة 12 في الجملة الطويلة التي تبين أن المتغير age يحتوي على قيمة معينة ، وفي الصفحة الثانية ستلاحظ نفس الجملة ونفس القيمة تمت طباعتهما ، أما في الصفحة الثالثة والأخيرة فتمت طباعة الجملة ، لكن الاختلاف أن القيمة 12 في متغير الـ age Session لم تتم طباعتها ، لماذا ؟

لسبب بسيط وهو أننا في الصفحة السابقة قمنا بالغاء تسجيل الـ Session للمتغير age وبالتالي فإن الصفحة الثالثة لم تتعرف على متغير مباشر له الاسم age ولا على متغير الـ age Session ، وبالتالي تم طباعة الجملة بدون القيمة .

التعامل مع الملفات والمجلدات

كل مبرمج يجب أن يتعامل مع الملفات والمجلدات في بعض النقاط ، برنامجك سوف يستخدم الملفات لكي يقوم بتخزين معلومات الإعداد للسكربت ، أو يقوم بتخزين البيانات لقراءتها وكتابتها ، أو لكي يقوم بحفظ البيانات المؤقتة ، وكمثال فإن أتفه برنامج عداد يحتاج إلي ملف يقوم بتخزين آخر رقم تم الوصول إليه ..

الملف : ليس عبارة عن أكثر من بايتات متسلسلة يتم تخزينها على القرص الصلب أو أي ماده تخزينية أخرى .
والمجلد : هو عبارة عن نوع محدد من الملفات يحتفظ بأسماء ملفات أخري ومجلدات أخري (تسمي بالمجلدات الفرعية) ، كل ما تحتاجه لتتعامل مع الملفات والمجلدات هو كيف يمكنك ربط سكربتك بهم ..

هذا الدرس سيأخذك إلي جولة لتعلم التعامل مع الملفات والمجلدات وفي نفس الوقت يوفر لك مرجعية لبعض الدوال التي تساعدك في ذلك مما يجعل مهمتك أسهل ...

سيقوم هذا الدرس بتغطيه المواضيع التالية :

1. فتح وإغلاق الملف .
2. القراءة من الملف والكتابة إليه .
3. مسح وإعادة تسمية الملفات
4. استعراض وتجول في الملف .
5. فتح وإغلاق المجلدات .
6. نسح وإعادة تسميه المجلدات .

ملاحظة :
قبل أن نبدأ دعنا ننبهك أن التعامل مع الملفات يختلف من نظام تشغيل إلي آخر ففي أنظمة اليونكس تستخدم المسارات العلامة المائلة للأمام
مثال
/home/usr/bin/data.txt
بينما في الويندوز فإن المسار يكون كالتالى
C:\usr\bin\perl
وإذا استخدمنا العلامة الأمامية في PHP للويندوز فإنه يقوم بتحويلها بشكل تلقائي إلي علامة خلفية بينما إذا أردنا استخدام العلامة الأمامية فإننا يجب أن نقوم بتكرار العلامة لكي يتم التعرف عليها
C:\\windows\\PHP

التعامل مع الملفات

يوفر الـPHP نوعين من الدوال المتعلقة بالملفات فهناك نوع من الدوال يستخدم مقبض للملف (file handle) أو ما يسمونه بالمؤشر (pointer) في العادة ، بينما بعض الدوال يستخدم قيمه حرفيه تشير إلي موضع الملف مباشره ...

مقبض الملف ليس أكثر من عدد صحيح (integer) يقوم بتعريف الملف المراد فتحه حتي يتم إغلاقه ، إذا كان هناك أكثر من ملف مفتوح فان لكل ملف مقبضه التعريفي الخاص به ، وبالطبع فإنه لا يتوجب عليك معرفه هذا الرقم ....

على سبيل المثال فإن الدالة fwrite() تقوم بفتح الملف لكتابة بيانات إليه وهي تستخدم مقبض لكي تقوم بالتعرف إلي الملف وفتحه ..

```
Fwrite ($fp,'Hello World');
```

بينما الدالة file() التي تستخدم للقراءة من الملف تقوم باستخدام قيمة نصية تقوم بالإشارة إلى مكان الملف بشكل مباشر لكي يتم التعامل معه ..

لا تصب بالرعب والخوف من هذا الكلام فأنا أعلم أنه قد يكون غامضاً عليك .. تنفس الصعداء وجهز لنفسك كأساً من الشاي لأننا سنبدأ في الجد الآن ....

ملاحظة : ستجد أن اغلب الدوال أو معظمها أو كلها تقريبا تقوم بإرجاع القيمة True إذا تمت بنجاح والقيمة False إذا فشلت في الحصول على هدفها ..
لنبدأ الآن مع سكربتات مبسطة للعمل مع الملفات ..

فتح واغلاق الملفات

Fopen
تستخدم هذه الدالة ثلاث عوامل هي مسار الملف (path) والوضع له (للقراءة ، للكتابة .....) بالاضافة إلى مسار الـInclude فيه وتقوم هذه الدالة بإرجاع مقبض للملف ...

قد تواجهنا مشاكل أحيانا فقد يكون الملف غير منشأ أو أننا لا نملك صلاحيات عليه ولذلك فإنه يمكننا اختبار القيمة التي ترجعها هذه الدالة فإذا كانت القيمة صفر فهذا معناه أن الدالة فشلت في إرجاع مقبض الملف أو نوعه ، أما إذا كانت القيمة هي واحد فهذا معناه أن الدالة قد نجحت في فتح الملف

مثال

```
$fp=fopen ("./data.txt", "r");
if (!$fp) die ("فشل في قراءه الملف تأكد من التراخيص ومن مسار الملف");
```

يمكننا كتابة المثال أيضا بالشكل التالي :

```
If  (!($fp=fopen ("./data.txt", "r"))) die ("لا يمكن القراءة من الملف");
```

لاحظ أننا قلنا سابقاً أن هناك دوال تستخدم للتعامل مع الملفات تستخدم مقبض وهذا المقبض هو عبارة عن رقم ، في مثالنا هذا يتحدد رقم المقبض الذي هو المتغير $fp الذي يخزن فيه مكان الملف وما إذا كان قابلاً للفتح أو لا أو يعمل أو لا يعمل ، والنتيجة التي تتخزن في المتغير $fp هي رقم مثلما قلنا سابقاً وهو صفر إذا كان الملف لا يعمل أو واحد إذا تم فتح الملف بنجاح ..

الآن دعنا نناقش معاملات الدالة fopen الذي تقوم بإعطائنا رقم المقبض ..

أول معامل هو مسار الملف على القرص الصلب

لنفرض أن لديك مجلدا قمت بإنشائه في مجلد السكربتات الرئيسي لديك الذي يسمى htdocs وأسميته data

ولنفرض أن سكربتك يستخدم ملفين :
1- ملف للقراءة والكتابة يسمي data.txt .
2- وملف يقوم بعرض المدخلات والإضافة إليها اسمه script.txt .

حسنا لدينا الآن ثلاث حالات للسكربت

الحالة الأولى :
أن يكون الملفين في نفس المجلد (data) وعند ذلك يمكنك فتح الملف الذي تريد فتحه بذكر اسمه فقط من غير إضافات

```
$fp=fopen ("data.txt", "r");
```

الحالة الثانية:

أن يكون هناك مجلد في نفس مجلد الـdata باسم آخر ولنقل أن هذا الاسم هو gb وفيه ملف data.txt على ذلك فإننا نكتب المسار المطلق لهذا المجلد كالتالي :

```
$fp=fopen ("./gb/data.txt", "r");
```

الحالة الثالثة :

أن يكون الملف الذي تريد قراءته موجود في المجلد htdocs بينما السكربت موجود في المجلد data الموجود داخل htdocs على ذلك نكتب المسار النسبي كالتالي

```
$fp=fopen ("../data.txt", "r");
```

لاحظ النقطه التي تسبق العلامة الأمامية جيدا..

أتمني أن تكون فهمت من هذا الكلام ما هو المقصود بالمسار المطلق والمسار النسبي ..

يمكننا أيضا وضع رابط صفحة في موقع آخر ولكننا لن نستطيع الكتابة عليه بل قراءته فقط

مثال :

```
If  (!($fp=fopen ("http://www.swalif.net/softs/index.php", "r"))) die ("لا يمكن القراءة من الملف");
```

ينقصنا نقطه يجب أن نتكلم عنها وهي عند تحديد العامل use_include_path

العامل الثاني الذي نستخدمه للملفات هو حاله الملف

(للقراءة ، للكتابة ، للإضافة إليه) يحدد وضعية الملف حال فتحه إذا كان للقراءة فقط أو للكتابة فقط أو للاثنين معاً أو للإضافة ، وأرتبها هنا في جدول بسيط ..

القيمة	الوصف
r	تفتح الملف للقراءة فقط ويكون المؤشر في بداية الملف
r+	يفتح الملف للقراءة والكتابة ويضع المؤشر في بداية الملف
w	يفتح الملف للقراءة فقط ، أي بيانات موجودة سيتم مسحها ، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه
w+	يفتح الملف للقراءة والكتابة ، أي بيانات موجودة سيتم مسحها ، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه
a	يفتح الملف للإضافة فقط ، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه ، سيكون المؤشر في نهاية الملف
a+	يفتح الملف للقراءة و للإضافة ، إذا لم يكن الملف موجودا سيحاول PHP إنشاؤه ، سيكون المؤشر في نهاية الملف
b	يستخدم لفتح وقراءه ملفات الصور على نظام أو سيرفرات الويندوز فقط .. أما الينوكس فالعوامل السابقه تتعامل مع ملفات الصور بشكل عادي ..

هناك مؤشر للملفات يحدد إذا ما كنت ستكتب من نهاية أو بداية الملف أو حتى من وسطه أو من أي مكان بالملف ، ستعرف كيفية التحكم بهذا المؤشر بعد قليل .

العامل الثالث هو تحديد use_include_path

فإذا قمت بتحديد قيمته إلي (1) وقمت بكتابة اسم الملف مباشرة فسيبحث الـPHP عن الملف في نفس المجلد الموجود به السكربت ثم سيقوم بالبحث عن الملف في المجلدات التي تم تحديدها في المتغير use_include_path في ملف php.ini

```
;;;;;;;;;;;;;;;;;;;;;;;;
; Paths and Directories ;
;;;;;;;;;;;;;;;;;;;;;;;;

; UNIX: "/path1:/path2"
;include_path=".:/php/includes"
;
; Windows: "\path1;\path2"
;include_path=".;c:\php\includes"
```

مثال :

```
$fp=fopen (“./data.txt”, “r”,1);
```

fclose

عندما تنتهي من التعامل مع الملف ، تحتاج إلى إغلاقه لكي يتم حفظ التعديلات عليه ، إذا تم إحباط سكربتك لأي سبب أو أن السكربت انتهي عمله فإن الـPHP يقوم بإغلاق جميع الملفات تلقائيا
تقوم الدالة ()fclose بإغلاق الملف عندما تريد إغلاقه وهي تحتاج إلي معامل واحد فقط وهو مقبض الملف الذي تريد إغلاقه
مثال :

```
Fclose ($fp) ;
```

قراءه وكتابه الملفات

لقد تعرفنا الآن كيفية فتح وإغلاق الملف ، لنقم الآن بالتعرف علي كيفية قراءة و كتابة البيانات من الملف ،

Fread

تقوم هذه الدالة بقراءة واستخراج البيانات الموجودة في الملفات ووضعها بمتغير وهي تأخذ معاملين المعامل الأول هو مقبض الملف والعدد الثاني هو عدد الحروف المراد قراءتها ..
مثال :

```
$fp=fopen("data.txt","r");
$data=fread($fp,10);
```

وخذ باعتبارك نقطتين وهما :

1- إذا مثلاً قرأت عشر حروف من الملف وكان في الملف عشرين حرف وقمت بطلب الدالة fread مره أخري فسيتم قراءة العشر أحرف الثانية ..
2- إذا كان في الملف أقل من عشر أحرف فسيتم قراءة الموجود .

Fwrite

تقوم هذه الدالة بالكتابة إلى الملف وتحتاج إلى عاملين وهي مقبض الملف والقيمة المراد كتابتها إلى الملف ، فعلى افتراض أنك قد فتحت الملف والمقبض هو $fp فإننا نكتب الكلمة PHP إلى الملف بالطريقة التالية :

```
Fwrite ($fp, "PHP");
```

وهناك معامل ثالث لهذه الدالة يحدد كم حرفا سنقوم بكتابته من القيمة الحرفية الموجودة في المعامل الثاني فلو مثلا كتبنا

```
Fwrite ($fp, "PHP",1);
```

فسوف يتم كتابه أول حرف فقط ...

قراءة وكتابة الحروف في الملفات

Fgetc

تستخدم هذه الدالة لقراءة حرف واحد من الملف في كل مرة ، وهي تستخدم معاملاً واحداً وهو مقبض الملف
وتقوم بإرجاع حرف واحد من الملف أو (False) عند الوصول إلى نهاية الملف ..

Feof

تقوم هذه الدالة بخدمتنا في هدف بسيط وشي ممتاز وهي معرفة إذا ما كنا قد وصلنا إلى نهاية الملف عند قراءته وتقوم بإرجاع (true) عند الوصول إلى نهاية الملف أو حصول خطأ ما ، وهي تأخذ معاملاً واحد وهو مقبض الملف .
فقد تكون مثلاً تريد أن تتأكد أن المؤشر لم يصل إلي نهاية الملف بعد استخدامك لأحد الدوال التي تقوم بنقل المؤشر من مكان إلي آخر ، عند ذلك ستكون هذه الدالة مفيدة لتخبرك إذا ما وصلت إلى نهاية الملف أو لا ...

تطبيق عملي :

قم بإنشاء ملف سمه file.txt واكتب فيه أكثر من سطر ثم قم بإنشاء ملف PHP وسمه بأي اسم وضع فيه الشفرة التالية ثم اختبره ، لكي ترى عمل الدالتين

```
 <?
$fp= fopen("file.txt","r");
While (!feof($fp))
{
 $char=fgetc($fp);
    echo $char;
} ?>
```

Fgets

إذا استخدمنا الدالة fgetc لقراءة الملفات الطويلة فإنها ستأخذ وقتاً وعمراً حتى يتم قراءتها ، يقوم الـPHP بتوفير دالة fgets لتساعدنا في قراءة عدد محدد من البايتات وهي تأخذ معاملين ، المعامل الأول هو مقبض الملف والمعامل الثاني هو عدد الحروف المراد قراءتها +1 ، فإذا مثلا أردت قراءة ملف يتكون من خمس حروف فسيكون المعامل الثاني للدالة هو الرقم 6 وتتوقف الدالة عند حدوث أحد من ثلاث حالات
الأول : هو إذا تم قراءة عدد البايتات المحدد ة
الثاني : إذا تم الوصول إلى نهاية سطر في الملف .
الثالث : إذا وصلت إلى نهاية الملف .
مثال :

```
$fd = fopen ("/tmp/inputfile.txt", "r");
while (!feof ($fd)) {
   $buffer = fgets($fd, 4096);
   echo $buffer;
}
fclose ($fd);
```

Fputs

تقوم بنفس وظيفة الداله fwrite وتأخذ نفس معاملاتها ونفس طريقتها ..

القراءة داخل الملفات

File

تحتاج هذه الدالة إلى معامل واحد هو مسار الملف ولا تحتاج إلي مقبض ، وعملها هو قراءة ما بداخل الملف وتخزينه سطراً سطراً في مصفوفة حيث أن هذه المصفوفة تقوم بأخذ كل سطر في الملف كأنه عنصر لوحده وتظل السطور سطورا (أي أن المصفوفة تحتفظ بالمعامل للسطر الجديد (\n) بداخلها) ، هذه الدالة لا تحتاج إلي مقبض للملف بل تحتاج إلى مسار الملف فقط ، وهي تقوم بفتح وقراءة وإغلاق الملف تلقائيا ...
وكغيرها من الدوال فإنها تستطيع قراءة صفحات الإنترنت الخارجية ..

مع ذلك يستحسن أن لا تقوم باستعمال هذه الدالة لقراءة الملفات الطويلة لأنها تقوم باستخدام قدر كبير من الذاكرة المحجوزة للـ PHP وقد تستخدمها كلها ...
مثال :

```
<?
$fcontents = file ('file.txt');
while (list ($line_num, $line) = each ($fcontents)) {
   echo "<b>Line $line_num:</b>  $line  <br>\n";
}
?>
```

Fpassthru

تقوم هذه الدالة بقراءة محتويات الملف بداية من النقطة التي توقف منها المؤشر الوهمي عند أي عملية قراءه أخرى ، وتقوم بالتوقف عند نهاية الملف وتقوم بإغلاق الملف من تلقاء نفسها لذلك لا داعي لإغلاق الملف بواسطة الدالة fclose بعد استخدامك لهذه الدالة ، وتقوم الدالة بقراءة المحتويات وطباعتها بشكل قياسي ، وهي تحتاج إلى معامل واحد فقط وهو مقبض الملف ...
مثال :

```
<?
$fp=fopen("file.txt","r");
fpassthru($fp)
?>
```

Readfile

تقوم هذه الدالة بقراءة جميع محتويات الملف ولا تحتاج إلي مقبض بل إلى مسار الملف فقط وتقوم بقراءة كامل محتويات الملف ثم طباعتها بشكل قياسي وتقوم بإرجاع عدد البايتات التي تم قراءتها أو (false) عند حدوث خطأ ما

```
<?
Readfile ("file.txt");
?>
```

الوصول العشوائي إلى الملفات

أخبرناكم سابقا بأن هناك طريقة تجعلك تتحكم في التحكم بالمؤشر الوهمي للملف والوصول إلي أي مكان في الملف أو عند أي حرف تريده ، بالدوال السابقة كنا عندما نصل إلي حرف معين مثلاً بدالة من الدوال فإننا نقوم بإغلاق الملف ثم نعاود فتحه كي نكمل القراءة من عند الحرف الذي تم الوصول إليه ولكن هذه الطريقة غير عمليه نهائيا .....
يوصل لنا الـPHP بعض الدوال التي تمكننا من الوصول إلي الملف بالمكان الذي نريده ومن هذه الدوال :

Fseek

تحتاج هذه الدالة إلى عاملين ، العامل الأول هو مقبض الملف $fp والعامل الثاني هو عبارة عن رقم صحيح يسمونه كمصطلح بالـ(offset) أي المكان الذي سيتوقف فيه المؤشر ، سيقوم الـPHP بالتحرك في الملف إلي أن يصل إلى المكان الذي تم تحديده .. أي أنه إذا كان في الملف سطر واحد مكون من عشره حروف وقمنا بجعل الـoffset خمسة ، سيقوم الـPHP بالتحرك حتى يصل إلى نهاية الحرف الخامس ...
وهناك معامل ثالث اختياري لهذه الدالة ويسمونه كمصطلح بالـ (whence) وله إحدى ثلاث خيارات :

Seek_set ويقوم بقراءة الملف من بدايته حتى يصل إلى المكان المطلوب بالـoffset
Seek_cur يقوم بالقراءة من المكان الحالي حتى يصل إلي المكان المطلوب بالـoffset
Seek_End يقوم بالقراءة من نهاية الملف حتى يصل إلي المكان المحدد بالـoffset

تعتبر هذه الدالة نادرة في عملها (أو كما يسميها المبرمجون شاذة) بسبب أنها تقوم بإرجاع القيمة (0) عند نجاحها والقيمة (-1) عند حصول خطأ ما ..

مثال :

قم بفتح ملف واكتب فيه ثمان حروف متتالية ثم قم بحفظه باسم file.txt ثم قم بوضعه مع ملف PHP فيه الشفرة التالية ، ثم بعد ذلك شغل ملف الـPHP وانتظر النتيجة :

```
 <?
$fp = fopen("file.txt");
fseek($fp,4,SEEK_SET);
fpassthru($fp);
?>
```

Ftell

هذه الدالة من الدوال المفيدة فهي تقوم بإرجاع مكان الـoffset (أو المؤشر الوهمي) في الملف وتحتاج إلي معامل واحد وهو مقبض الملف ...

```
<?
$fp = fopen ("file.txt");
$p = ftell($fp);
echo $p;
?>
```

Rewind

تقوم بإرجاع المؤشر إلي بداية الملف ...

```
<?
$fp = fopen ("file.txt");
rewind($fp)
?>
```

جلب معلومات الملف

يوفر الـPHP دوال تساعدنا في معرفه حجم الملف وما إذا كان الملف موجوداً أم لا من هذه الدوال :

File_exists

تقوم هذه الدالة بالقيام بالتأكد ما إذا كان الملف موجوداً أم لا وهي تحتاج على معامل واحد وهو مسار الملف ، وتقوم بإرجاع true (1) إذا كان الملف موجوداً و false إذا كان الملف غير موجود

```
<?
$Th=File_exists("file.txt");
echo $Th ;
?>
```

Filesize

تقوم هذه الدالة بإرجاع حجم الملف بالبايتات أو false عند حصول خطأ ...

دوال الملفات المتعلقة بالوقت :

هذه الدوال تقوم بإرجاع معلومات مفيدة عن وقت التغيير الذي طرا على الملف أو آخر مره تم قراءته وهي على حسب نظام التشغيل فإذا كان نظام السيرفير هو يونكس أو لينوكس ستقوم الدوال بإرجاع الوقت بنظام (timestamp) وهو الوقت مترجم إلي عدد الثواني منذ صدور يونكس ومولده على العالم ، بينما تقوم بإرجاع وقت التعديل على نظام الويندوز مباشره ...

يقوم الـPHP بتزويدنا بدالتين لمعرفه الوقت :

Filectime وتقوم بإرجاع آخر وقت تم فيه التغيير على الملف على شكل timestamp ويشمل هذا آخر وقت تم فيه إنشاء الملف أو الكتابة إليه أو تغيير تراخيصه ...

Filemtime تقوم بإرجاع آخر وقت تم فيه التعديل على الملف على شكل timestamp ويشمل هذا إنشاء الملف أو تغيير محتوياته ...

تقوم الدالة getdate بعمل مفيد وهو تحويل الوقت من timestamp إلي الوقت العادي

الملكية والتراخيص

على أنظمة تشغيل اليونكس مثل يونكس ترتبط الملفات مع مستخدم خاص أو مجموعة من المستخدمين (group) وتحتوي على علامات وتراخيص تقوم بتوضيح من له صلاحية على استخدامها .. يمكننا أن نخلص التراخيص كالتالي :

1 / ممتلك الملف (owner) ، بشكل افتراضي ، وهو المستخدم الذي تم استخدام حسابه في استخدام الملف .

2 / مجموعه من المستخدمين (group) ، بشكل افتراضي ، المجموعة التي يكون ضمنها مالك الملف

3 / جميع المستخدمين (all) كل شخص له حساب على النظام .

المستخدمين والمجموعات في أنظمة اليونكس يتم تعرفهم عن طريق رقم (ID) مثلما يتم تعريفهم عبر أسمائهم ، إذا كنت تريد معرفه معلومات شخص عن طريق رقمه ، يمكنك استخدام هذه الدالة :

Posix_getpwind

التي ستقوم بإعطائنا مصفوفة تحتوي على المعلومات التالية

Name	اسم المستخدم الذي يدخل به في حسابه
passwd	كلمة السر المشفرة للمستخدم
uid	رقم الحساب للمستخدم
gid	رقم حساب المجموعة التي فيها المستخدم
gecos	اسم المستخدم الكامل ، رقم تلفونه ومعلومات إضافية
dir	المجلد الرئيسي للمستخدم
shell	المسار الرئيسي لحساب المستخدم

Posix_getgrgid

تقوم هذه الدالة بإرجاع مصفوفة عن معلومات المجموعة ، وهي تحتاج إلى معامل واحد فقط وهو رقم الـID للمجموعة ...
وسوف تحتوي على العناصر التالية :

Name	اسم المجموعة
Gid	رقم المجموعة
members	عدد أعضاء المجموعة

وهناك ايضا خمس دوال تساعدنا في معرفة معلومات أكثر عن الملفات وتحتاج فقط إلي مسار الملفات

Fileowner

تقوم بإرجاع رقم المعرف (ID) لمالك الملف ...

Filegroup

تقوم بإرجاع رقم المعرف (ID) لرقم المجموعة التي يعتبر مالك الملف ضمنهم ..

Filetype

تقوم بإرجاع رقم نوع الملف وقد تعود بإحدى هذه القيم (file ، dir ، char ، fifo، link ، block) والذي يهمنا منهم هو file وdir ...

Is_dir

وتقوم بإرجاع True إذا كانت قيمه المسار هو مجلد ..

Is_file
وتقوم بإرجاع True إذا كانت قيمه المسار هو ملف ..

الحصول على اسم الملف من وسط مسار الملف ..
Basename()
مثال :

```
<?
$path = "/home/httpd/html/index.php3";
$file = basename ($path);
echo '$file <br>';
$file = basename ($path,".php3");
echo '$file <br>';
?>
```

هذه الدالة مفيدة جداً للحصول على الملف من وسط مسار مجلد ..

نسخ ، اعادة تسمية وحذف الملفات

تسمح لك الـPHP بنسخ ، وإعادة تسمية ، وحذف والدوال التي تستخدم لتنفيذ هذه العمليات هي

Copy ()
تقوم بأخذ قيمتين حرفتين وتشير إلي مصدر الملف الرئيسي الذي يوجد فيه الملف والمصدر الهدف الذي سيتم نسخ الـPHP إليه ...

```
<?
if (!copy($file, $file.'.bak')) {
    print ("failed to copy $file...<br>\n");
}
?>
```

Rename
نستطيع الآن استخدام هذه الدالة لإعادة تسمية الملف وهي تحتاج إلي قيمتين حرفتين وهي المصدر الملف أو مكانه واسمه الرئيسي ثم الاسم الجديد الذي تريد إعادة التسمية به ..
مثال :

```
<?
Rename ('file.txt','newfile.txt');
?>
```

Unlink()
تحتاج إلي قيمه حرفيه واحده وهي مسار الملف الذي تريد حذفه

```
<?
unlink ('file.txt');
?>
```

العمل مع المجلدات

مثلما تعاملنا مع الملفات في الـPHP فإننا نتعامل مع المجلدات ، فهناك دوال للمجلدات تتطلب مقبض المجلد ، وهناك دوال تحتاج فقط إلي القيمة الحرفية فقط وبدلاً من الإطالة دعنا نقوم بالدخول في الموضوع مباشرة

Opendir
تقوم بفتح المجلد وإعطائنا مقبض المجلد

Closedir()

تقوم بإغلاق المجلد المفتوح وتحتاج فقط إلى مقبض المجلد ...

Readdir
تقوم بقراءة المدخل الحالي للمجلد ...

Rewindir
تقوم بإرجاع المدخل من الصفر ..

Chdir
للانتقال إلي مجلد آخر ، وتتطلب المسار للمجلد الذي تريد الانتقال إليه ..

Rmdir
تقوم بمسح مجلد ، ولكن يجب أن يكون المجلد خاليا من أي ملفات أو مجلدات ، وتتطلب مسار المجلد الذي تريد مسحه ..

Mkdir
تقوم بإنشاء مجلد جديد وتتطلب أن يكون هذا المجلد غير موجود مسبقاً وتحتاج إلي قيمتين وهما اسم المجلد الجديد مع مساره ، والترخيص المطلوب له ..

Dirname
تقوم بإعطائنا اسم المجلد الحالي الذي فيه الملف ، وتحتاج إلي مسار الملف ..

تطبيق عملي :
أنشئ مجلد اسمه tmp في مجلد الـ htdocs وضع فيه ملفات ، ثم أنشئ ملف اسمه test.php في مجلد الـhtdocs واكتب الشفرة التالية ثم شغله :

```
<?php
if ($dir = @opendir("/tmp")) {
  while($file = readdir($dir)) {
    echo "$file\n";
  }
  closedir($dir);
}

?>
```

Dir()
عبارة عن كائن يحتوي على ثلاث وظائف .. ونقوم بإعطائه مسار المجلد الذي نريده أن يتعامل معه ثم بعد ذلك نقوم بوضع قيمته في متغير يقوم بوراثة صفاته

خصائص الكائن :

handle
تقوم بإعطائك مقبض المجلد ..

Path
تقوم بإعطائك المسار للمجلد ..

Read
تقوم بإعطائنا المجلدات اعتماداً على المؤشر الحالي للمجلد ..

Rewind
تقوم بإرجاع مؤشر المجلد من الصفر ..تقريباً نفس عمليه rewinddir ..

Closedir
تقوم بإغلاق المجلد ..

بهذا يكون انتهي الدرس
قد تكون بعض النقاط غير واضحة ، في الواقع لن تحتاج إلي كل هذه الأمور في تعاملك مع الملفات

دعنا نأخذ فكرة عن طرف التخزين في البداية وكيف كانت على الإنترنت في السابق ....
كان من أكثر طرق التخزين انتشارا في السابق على الإنترنت وربما هو لا يزال يستخدم في بعض المواقع والمنتديات يعتمد على الملفات ....فكان صاحب الموقع الذي لديه هذه الطريقة في تخزين البيانات خوف وتعب من فقدانها مثلاً وكان الشبح الذي يكدر عليه صفوة نجاح موقعه هو عمل نسخ احتياطية لهذه الملفات لكي يتمكن من استرجعها في حال فقدانها فكانت هذه العملية تأخذ وقت وجهد ومال ....كما كان من عيوب تخزين البيانات في الجداول ضغط الخادم أو (server) في حال الاستعلام عن معلومة معينة والبحث عنها كما أنه يستهلك الكثير من ذاكرة هذا الخادم في عملية بحث معينة فهو يحجز مساحة ليست بالهينة في عملية بحث عن أسم مثالاً أو ما شابهها .

ربما يتردد عند البعض ذالك السؤال وهو ....ما هي قواعد البيانات بالضبط ؟

قواعد البيانات ببساطة جمع المعطيات أو المدخلات .
كل قاعدة بيانات ربما تتكون من جدول (Table) واحد أو عدة جداول هذه الجداول تحتوي علي أعمدة وصفوف تهيكل البيانات وترتبها ,,,,,وسوف أجعل لك مهمة اكتشاف فوائد قواعد البيانات في آخر الدرس .
لترى الجدول الذي بالأسفل كمثال :

#Table "Coustomers"

Lname	Fname	Id
صالح	عبدالواهب	025
خالد	محمد	044

كما تلاحظ , الجدول قسم البيانات إلى صفوف ...مع كل إضافة عميل جديد سوف يكون هناك صف (سجل) جديد ... ربما لو تطلق لخيالك العنان سوف تلاحظ أن هذا الجدول مشابه للدولاب والصفوف رفوف فإذا أردت أن تضيف كتب أو ملابس أو أي كان سوف تضيفها في رف جديد ..كما يحصل في إضافة عميل جديد سوف تضيفه في صف (سجل) جديد .

البيانات في كل صف قسمت إلى مدى أبعد في الخلايا (أو الحقول) , كل من هذه البيانات تحتوي على قيمة محددة وصفة محددة , على سبيل المثال محمد خالد سوف ترى أن هذا العميل انقسمت بياناته في الحقل إلى id والاسم الأول والاسم الأخير .

الصفوف في الجدول ليس لها ترتيب معين .. يمكن أن يكون الترتيب أبجديا ويمكن أن يكون باسم العضو أو باسمه الأخير أو بآي معيار أخر يمكن أن تحدده مسبقاً لترتيب الصفوف ولهذا يكون من الضروري تحديد طريقة ليسهل عليك تحديد صف(سجل) معين ....في المثال السابق نستطيع إخراج السجل من بين باقي السجلات بـ id وهو رقم العميل الذي هو عدد فريد لا يتكرر في أي صف(سجل) أخر وسبب استنادي في استخراج السجل علي id لانه ربما يكون هناك عميلان لها نفس الاسم .... وهذا ليس شرط أن يكون للجدول مفتاح فريد لكن هنا حددنه لكي يتم استخراج السجلات المطلوبة بسهوله وبسرعة اكبر .

العلاقات

الكثير من قواعد البيانات اليوم هي نظم إدارة قواعد بيانات علائقية (relational database management systems) تختصر في RDBMS , قواعد البيانات العلائقية هذه عبارة عن مجموعة من الجداول أو نموذج من الجداول النموذجية المتعددة التي تحتوي على معلومات مترابطة .
ربما تسمع ايضاً الكثير عن SQL وهي اختصار ل(Structured Query Language) وهي تسمح لك أن توحد هذه المعلومات من الجداول المترابطة وبذالك تسمح لك بإنشاء وتحليل العلاقات الجديدة .

المثال السابق للعملاء كان عبارة عن جدول واحد فقط , ولذالك لن تحتاج إلى ربط بينه وبين جدول آخر لأنه لا يجد .

لكن إذا كان هناك أكثر من جدول وكانت هذه الجداول مترابطة مع بعضها البعض في البيانات سوف تلاحظ أنك بحاجة إلى نظم إدارة البيانات العلائقية (RDBMS) ....فلنرى هذه المثال لكي تتضح الصورة أكثر :

#Table "Coustomers "

Lname	Fname	Id
صالح	عبدالواهب	025
خالد	محمد	044
طارق	حمد	022

#Table "Address"

Country	City	Street	Tel	Id
مصر	القاهرة	شارع الاهرام	018522	044
السعودية	الرياض	طريق الملك فهد	01225505	022
الكويت	الكويت	طريق الاربعين	0122505	025

#Table "Account"

accountb	Id
10.0000	044
20.0000	025
20.000	022

كل من هذه الجداول الثلاثة كيان مستقل .... لكن تلاحظ أنهم مرتبطين مع بعضهم البعض ب(id) , على سبيل المثال بأماكننا أن نعرف رصيد العميل عبد الواهب صالح من id , كما يمكننا معرفة اين يسكن حمد طارق وكم رقم التلفون وايضاً بمكننا أن نعرف من هو صاحب الرصيد 20.000 أيضاً كم واحد من مدينة القاهرة والكثير الكثير ..... ربما اتضح لك اهمية العلاقات .

إذاً عرفنا أن العلافات هي الأساس الجوهري لنظم قاعدة البيانات العلائقية , بجعلها مرنه وسهلة بحث تتمكن من ربط السجلات المختلفة مع بعضها البعض في الجداول .

المفتاح الأجنبي

سوف تلاحظ أن حقل (id) الذي يظهر في الجداول في الثلاثة والذي جعل من الممكن ربط الجداول المختلفة معاً أنة مفتاح أجنبي لأنة بالأصل مفتاح فريد (primary key) في جدول (COUSTOMERS) ...

ليس ضرورياً أن يكون هناك مفتاح أجنبي في كل جدول ولكن يتم إضافة على حسب حاجتك فإذا كنت تريد ربط بيانات الجداول مع بعضها فسوف تحتاج إليها.

في كل جدول يجود به المفتاح الأجنبي سوف يكون له مرجعية للجدول الأصل فمثلاً هنا المرجعية ستكون جدول (customers) ....بمعنى أن المفتاح الأجنبي سوف يقوم بربط البيانات ما بين الجدول الأصل وبين الجدول الذي يتواجد به كمفتاح أجنبي ....من هنا يتضح لنا مفهوم الاستقامة المرجعية وهذا مفهوم أساسي ومهم عندما تصمم قاعدة بيانات بأكثر من جدول . سوف يكون للمفتاح الأجنبي قيمة ثابتة في جميع الجداول بمعنى لو كان قيمة المفتاح الأجنبي في جدوله الأصل عدد صحيح فسوف يكون بنفس القيمة في جميع الجداول ,,,,ونقطة أخرى إذا حدث تحديث أو تغيير أو حذف لا حد القيم في المفتاح الأجنبي فسوف تتم في جميع الجداول ...هذا هو مفهوم الاستقامة المرجعية .

كثير من قواعد البيانات اليوم يتم تعديل القيم بها تلقائياً كمكر سوفت أكسس وبعض قواعد البيانات الأخرى , لكن هناك بعض قواعد البيانات التي تحتاج إلى تعديل يدوي على كل قيمة يتم التعديل عليه ...وهذا لاشك انه متعب !!

الفهرسة

لو كان لديك جدول به الكثير من السجلات , يمكنك أن تستعلم بسرعة كبيرة عن أيمن هذه السجلات بواسطة "فهرسة" كل السجلات . هذا المفهوم تقريباً شبيه جداً بالفهرس الذي يوجد نهاية كل كتاب ...كما يسهل عليك هذا الفهرس الموجود في الكتاب في سرعة البحث عن المواضيع التي يتضمنها الكتب , نفس الكلام ينطبق على فهرسة السجلات في الجدول ....دعنا نرى مثال لتتضح الصورة :

```
SELECT * FROM names WHERE ID = 220;
```

سوف يقوم هذا الاستعلام في البحث في جميع المعلومات وإرجاع قيمتها بشرط أن يكون رقم السجل (الفهرس) 022

هنا سهلت علينا المهمة كثيراً وذالك لان السجلات مرتبه بأرقام فلي كل سجل رقم فريد يميزه عن الأخر وبهذه الحالة سوف يقوم هذا الاستعلام السابق بجلب جميع بيانات العميل "حمد طارق " ....