پیاده سازی پایگاه داده در وب

یک پایگاه داده تحت وب منبع بزرگی از اطلاعات، با قالبی سازماندهی شده است که به سادگی به وسیله زبان هایی مانند php قابل دسترسی است. در این فصل ابزارهای مورد نیاز ایجاد و توسعه پایگاه داده تحت وب در mysql را فراخواهید گرفت.

سیستم مدیریت پایگاه داده (database management system) یا dbms

نرم افزاری که مدیریت پایگاه داده را انجام می دهد. مانند:

- microsoft access :
- microsoft sql server :
- oracle :
- mysql : رایج ترین پایگاه داده متن باز (open source) محسوب می شود.

مقایسه ی سیستم های مدیریت پایگاه داده

	Microsoft Access	Microsoft SQL Server	MySQL	ORACLE
سریع‌ترین فناوری سمت سرور	Java Servlet	Java Servlet	PHP	ASP
محدودیت کاربران	۲۵۵	بدون محدودیت بسته به مقدار حافظه سیستم	بدون محدودیت	بدون محدودیت بسته به مقدار حافظه سیستم
مجوز	تجاری	تجاری	متن‌باز (رایگان)	تجاری
حداکثر ستون‌های جدول	۲۵۵	۱۰۲۴	۴۰۹۶	۱۰۰۰
واسط گرافیکی	دارد	دارد	ندارد	دارد
پیچیدگی در پیکربندی و نصب اولیه	ندارد	دارد - در سطح متوسط	ندارد- بسیار ساده	دارد- در سطح بالا
سخت‌افزار موردنیاز	با تجهیزات قدیمی نیز سازگار است	نیازمند تجهیزات سخت‌افزاری به‌روز	با تجهیزات قدیمی نیز سازگار است	نیازمند تجهیزات سخت‌افزاری به‌روز

دلایل استفاده از mysql :

۱. رایگان بودن
۲. هزینه حفظ و نگهداری بسیار پایین
۳. سهولت پیکربندی اولیه و داشتن محیط بسیار ساده و کاربرپسند

۴. صادر کردن(export) پایگاه های داده به صورت دستورهای sql

۵. در دسترس بودن کد اصلی (متن باز) برای توسعه ملی یا حتی خصوصی در سازمانها

۶. توانایی کار کردن همزمان با سایر سرویس دهنده های پایگاه داده تجاری مانند sql server و oracle

پیکربندی mysql

با استفاده از برنامه wamp می توان نسبت به پیکربندی پایگاه داده mysql اقدام کرد

تعیین سطوح دسترسی کاربر

برای اتصال به سرویس دهنده پایگاه داده mysql به صورت پیش فرض نام کاربری root و گذرواژه خالی در نظر گرفته شده است. کاربر root کاربر اصلی (مدیر ارشد) در پایگاه داده محسوب می شود و با استفاده از **سربرگ user account** در برنامه phpmyadmin می‌توان اقدام به رمزگذاری آن کرد و همچنین کاربران جدید با سطوح دسترسی مختلف ایجاد کرد.

زبان sql (structured query language)

sql زبان پرس وجوی ساخت یافته است و یکی از محبوب ترین زبان هایی است که برای برنامه هایی که به نوعی با بانک های اطلاعاتی رابطه ای سر و کار دارند، مورد استفاده قرار می گیرد. برای استفاده از دستورات این زبان از **سربرگ sql** در برنامه phpmyadmin استفاده می کنیم.

آشنایی با دستورات زبان SQL

قبل از اینکه به دستورات زبان SQL بپردازیم لازم است به نکته زیر دقت کنید.

نکته: در سرتاسر این جزوه کلاسی از پایگاه داده lib_school و جداول آن (جداول زیر) استفاده شده است.

جدول کتاب ها (books)

b_isbn شابک	b_name نام کتاب	b_auther نویسنده	b_publisher انتشارات	b_year سال چاپ	b_qty تعداد

جدول اطلاعات دانش آموز (student_info)

stu_id کد دانش آموز	stu _name نام	stu _family نام خانوادگی	stu _age سن	stu _nationalcode کد ملی	stu _address نشانی

جدول امانت (loan)

l_id کد امانت	Stu_id کد دانش آموز	B_ISBN کد کتاب	l_date تاریخ امانت

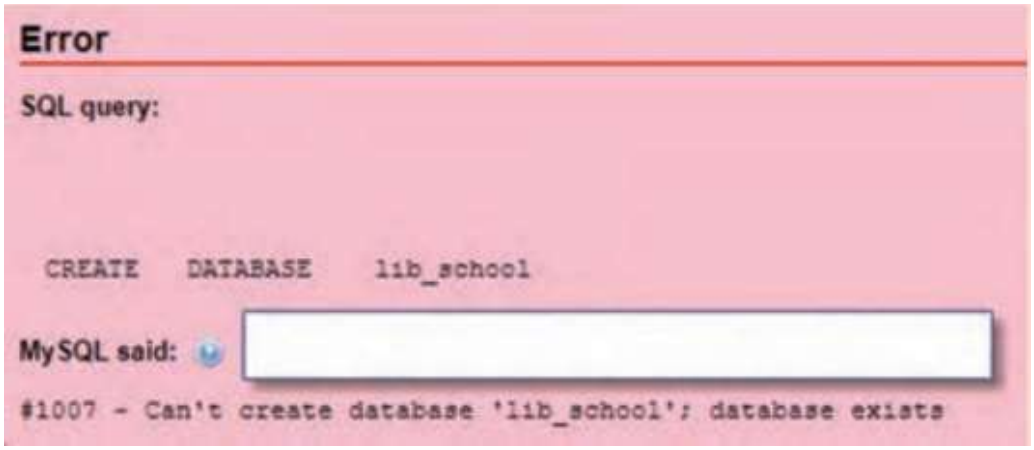
ایجاد پایگاه داده (database)

create database نام پایگاه داده;

مثال: ایجاد پایگاه داده جدید با نام lib_school

```
create database lib_school;
```

نکته: هنگام ایجاد پایگاه داده جدید، اگر پایگاه داده از قبل موجود باشد پیغام خطای زیر ظاهر می شود:



حذف پایگاه داده

drop database نام پایگاه داده;

مثال: پایگاه داده lib_school را حذف کنید

```
drop database lib_school;
```

انواع داده (datatype)

انواع داده ای که سرویس دهندهٔ پایگاه داده mysql از آن پشتیبانی می کند در جدول زیر آمده است:

نوع داده	ماهیت	اندازه	توضیحات
VARCHAR(n)	رشته‌ای	وابسته به تعداد نویسه‌های تعریف‌شده	n تعداد نویسه‌ها را مشخص می‌کند که حداکثر ۲۵۵ نویسه است. توجه: اگر رشته‌ای با طول بیش از ۲۵۵ نویسه را در آن قرار دهید، به نوع TEXT تبدیل خواهد شد.
TEXT	رشته‌ای	وابسته به تعداد نویسه‌های تعریف‌شده	رشته‌ای با طول حداکثر ۶۵۵۳۵ نویسه
INT	صحیح	۴ بایت	محدوده اعداد مجاز به‌صورت علامت‌دار: ۲۱۴۷۴۸۳۶۴۸- تا ۲۱۴۷۴۸۳۶۴۷ محدوده اعداد مجاز به‌صورت بدون علامت: ۰ تا ۴۲۹۴۹۶۷۲۹۵
FLOAT(size,d)	اعشاری	۴ یا ۸ بایت	از این نوع داده برای ذخیره اعداد اعشاری با ممیز شناور استفاده می‌شود. size حداکثر تعداد ارقام و d حداکثر تعداد ارقام سمت راست ممیز اعشار را مشخص می‌کند.
DATE	تاریخ	۳ بایت	این نوع داده برای نگهداری تاریخ میلادی، بدون ساعت است. محدوده تاریخ از '1000-01-01' تا '9999-12-31' است.
TIME	ساعت	۳ بایت	این نوع داده برای نگهداری زمان استفاده می‌شود.

نکته: در mysql سه نوع داده اصلی وجود دارد :**رشته ای، عددی و تاریخ و زمان**

ایجاد جدول (table)

شکل کلی

```
CREATE TABLE        نام جدول. نام پایگاه داده (
نام فیلد ۱      نوع داده ,
نام فیلد ۲      نوع داده ,
نام فیلد ۳      نوع داده ,
   ....
);
```

کلید اصلی (primary key)

برای اطمینان از تکراری نبودن داده‌ها در ستون‌های تعریف‌شده، در هر جدول می‌توان یک فیلد یا ترکیبی از چند فیلد را به‌عنوان کلید اصلی معرفی کرد. برای تعریف کلید اصلی از دستور PRIMARY KEY هنگام ساخت جدول استفاده می‌کنیم. فیلد کلیدی نمی‌تواند بدون مقدار باشد.

نکته: فیلد کلید نباید بدون مقدار باشد و با عبارت not null تعریف می شود.

شکل کلی استفاده از دستور PRIMARY KEY

```
CREATE TABLE        نام جدول. نام پایگاه داده (
نام فیلد ۱      نوع داده ,
نام فیلد ۲      نوع داده ,
نام فیلد ۳      نوع داده ,
   ....
PRIMARY KEY (نام فیلد)
);
```

مثال: جدول books را در پایگاه داده lib_school ایجاد کنید

نام فیلد	نوع داده	اندازه	شرح
b_id	int	۱۰	کد کتاب (کلید اصلی)
b_name	varchar	۱۲۰	نام کتاب
b_auther	varchar	۵۰	نویسنده کتاب
b_publisher	varchar	۵۰	انتشارات
b_year	int	۱۰	سال چاپ
b_qty	int	۱۰	تعداد

```
CREATE  TABLE  lib_school.books (
b_id   INT(10)  NOT NULL  ,
b_name      VARCHAR(120)  ,
b_auther     VARCHAR(50)   ,
b_publisher  VARCHAR(50)  ,
b_year  INT(10)  ,
b_qty    INT(10) ,
PRIMARY  KEY  (b_id)
);
```

حذف جدول

drop table نام پایگاه داده . نام جدول ;

مثال: جدول books را از پایگاه داده lib_school حذف کنید

```
drop  table  lib_school.books;
```

ویرایش ساختار جدول

تغییر در هر یک از موارد زیر باعث تغییر ساختار جدول شود:

- اضافه و حذف فیلد (ستون)
- تغییر نام فیلدها
- تغییر نوع داده فیلد یا طول آن
- حذف کلید اصلی یا تغییر آن
- ...

برای تغییر ساختار جدول از دستور alter استفاده می شود

نکته: اضافه یا حذف رکوردهای جدول(داده ها) منجر به تغییر ساختار جدول نمی شود.

افزودن فیلد جدید

alter table نام جدول

add نام فیلد نوع د اده (طول) ;

مثال: فیلد آدرس (stu_address) را به جدول student_info اضافه کنید

```
alter table student_info
add  stu_address  varchar(200);
```

حذف یک فیلد

alter table نام جدول

drop column نام فیلد;

مثال: فیلد stu_age را از جدول student_info حذف کنید

```
Alter  table  student_info
drop  column  stu_age ;
```

تغییر اندازه فیلد

alter table نام جدول

change نام فیلد نام فیلد (طول)نوع داده;

مثال: اندازه فیلد stu_name که ۵۰ تعریف شده است به ۶۰ تغییر دهید

```
alter table student_info
change stu_name stu_name varchar(60);
```

تغییر نوع داده یک فیلد

alter table نام جدول

modify نام فیلد نوع داده جدید ;

مثال: نوع داده فیلد stu_age را از عددی به رشته ای تغییر دهید.

```
Alter table student_info
modify stu_age varchar(2) ;
```

تغییر کلید اصلی

شکل کلی تغییر کلید اصلی یک جدول

```
ALTER TABLE نام جدول
DROP PRIMARY KEY ,
ADD PRIMARY KEY (نام فیلد یا فیلدها) ;
```

مثال: کلید اصلی جدول student_info را از کد دانش آموزی(stu-id) به کد ملی(**stu_nationalcode**) تغییر دهید

```
alter table student_info
drop primary key ,
add primary key (stu_nationalcode);
```

نمایه (index)

- نمایه عبارت است از یک شماره که به هر یک از فیلدهای جدول اختصاص داده می‌شود.
- استفاده از نمایه باعث می‌شود تا برنامه بتواند مقادیر رکوردهای مختلف را برحسب مقدار یک فیلد و برحسب شماره نمایه ی آنها از کم به زیاد یا برعکس مرتب کرده، در عملیات جستجو باعث بالا رفتن سرعت شود.
- نمایه از دید کاربر کاملاً مخفی است.
- در هنگام ساخته شدن نمایه برای فیلد یا فیلدها، جدولی به صورت مجازی حاوی ستون های نمایه شدهبه صورت مرتب ایجاد میشود. در هنگام درج رکورد یا ویرایش، جدول مجازی همیشه به روز میشودر جستجو بر اساس ستون نمایه شده در این جدول مجازی انجام می‌شود به همین دلیل سرعت دسترسی به داده ها افزایش می یابد.

شکل کلی ایجاد نمایه

```
CREATE INDEX نام نمایه
ON نام جدول (نام فیلد) ;
```

مثال: ایجاد نمایه بر اساس فیلد نام خانوادگی (stu_family) در جدول اطلاعات دانش آموزان با نام student_info

```
create  index  family
on  student_info (stu_family)
```

درج رکورد در جدول

از دستور insert برای درج یک رکورد جدید در جدول استفاده می‌شود.

```
insert into نام پایگاه داده . نام جدول (فیلد۱, فیلد۲, ...)
values (مقدار فیلدها)
```

نکته:

۱. در صورت مقدار دهی همه ی فیلد، نیازی به ذکر نام فیلدها در جلوی نام جدول نیست.
۲. هنگام مقداردهی به فیلدها به ترتیب فیلدها دقت کنید که با مقداری که وارد می کنید، مطابقت داشته باشد
۳. هنگام مقدار دهی به فیلدها به نوع داده آنها دقت کنید که با مقداری که وارد می کنید، مطابقت داشته باشد

مثال:اضافه کردن اطلاعات یک کتاب جدید بعنوان یک رکورد در جدول books

```
INSERT  INTO  lib_school.books (b_ISBN, b_name, b_auther, b_publisher, b_year,
b_qty)
VALUES (
'978_964_05_2714_6' ,
'تولید محتوای الکترونیک و برنامه‌سازی' ,
'سازمان پژوهش و برنامه‌ریزی آموزشی' ,
'شرکت چاپ و نشر کتاب‌های درسی' ,
'1396' ,
'60'
) ;
```

اسامی فیلدها

مقادیر فیلدها

حذف رکوردهای جدول (داده های جدول)

شکل کلی حذف رکورد

```
DELETE FROM نام پایگاه داده . نام جدول
WHERE شرط ;
```

مثال: حذف کتاب با شماره شابک '978-964-05-2588-3' از جدول books

```
delete  from  lib_school.books
where  b_isbn = '978-964-05-2588-3' ;
```

مثال: حذف کتاب هایی که سال انتشار آنها ۱۳۸۰ است.

```
delete from lib_school.books
where b_year = 1380 ;
```

حذف همه ی رکوردهای جدول

اگر در فرمان delete از دستور where استفاده نکنیم همه ی رکوردهای جدول حذف می شوند.

مثال: حذف کلیه رکوردهای جدول books

```
delete from lib_school.books;
```

نکته: برای حذف همه ی رکوردها از دستور truncate نیز می توان استفاده کرد. (بصورت زیر)

```
truncate table lib_school.books;
```

تفاوت فرمان های delete و truncate

۱. فرمان truncate سریعتر از delete عمل می کند.

۲. هنگام حذف رکوردها با فرمان truncate فیلد auto increment به مقدار اولیه خود باز می گردد در صورتی که در delete اینچنین نیست.

ویرایش رکوردها

امکان اشتباه، هنگام ورود داده ها اجتناب ناپذیر است. بنابراین باید روی رکوردها ویرایش انجام داد. برای این منظور از فرمان update استفاده می شود.

شکل کلی ویرایش رکوردها

```
UPDATE نام جدول. نام پایگاه داده
SET
نام فیلد ۱ = مقدار ۱ ,
نام فیلد ۲ = مقدار ۲ ,
...
WHERE شرط ;
```

توجه داشته باشید شرط جلوی WHERE تعیین می‌کند کدام رکوردها به‌روزرسانی شوند. بنابراین درصورتی‌که از شرط استفاده نشود، تمام رکوردها به‌روز می‌شوند.

مثال: موجودی کتاب با شماره شابک '978-964-05-2588-3' را به ۵۰ تغییر دهید

```
update lib_school.books
set b_qty = 50
where b_isbn = '978-964-05-2588-3' ;
```

مثال: با استفاده از دستورات sql موجودی کتابهایی که سال چاپ آنها 1395است را 30 عدد اضافه کنید

```
update lib_school.books
```

```
set   b_qty = b_qty +50
where  b _year =1395 ;
```

ویژگی افزایش خودکار (AUTO INCREMENT)

درصورتی‌که ویژگی AUTO_INCREMENT برای یک فیلد عددی اختصاص یابد، فیلد مربوطه در هنگام درج اطلاعات نیاز به مقداردهی ندارد و سرویس‌دهنده پایگاه داده MySQL به‌صورت خودکار در درج هر رکورد یک واحد به محتوای فیلد موردنظر اضافه می‌کند. به‌صورت پیش‌فرض واحد افزایش ۱ و شروع آن نیز از عدد ۱ است. البته می‌توان شروع را نیز تغییر داد.

برای اینکه یک فیلد دارای ویژگی AUTO_INCREMENT شود باید حداقل یکی از ویژگی‌های کلید اصلی (Primery Key) یا نمایه (Index) را داشته باشد. در غیر این صورت سرویس‌دهندهٔ پایگاه داده اجازه افزوده‌شدن این ویژگی را به فیلد مربوطه نمی‌دهد.

ایجاد پرس وجو(query)

مهم‌ترین بخش SQL، پرس‌وجو است. برای کسب اطلاعات و تهیه گزارش‌های مختلف، ذخیره داده‌ها به تنهایی کفایت نمی‌کند. با استفاده از پرس‌وجو از این داده‌ها می‌توان در تهیه گزارش استفاده کرد.
از دستور SELECT برای ایجاد پرس‌وجو و نمایش رکورد یا رکوردهای موجود در یک جدول استفاده می‌شود. درصورتی‌که از شرط استفاده نکنید همه رکوردها نمایش داده می‌شوند.

شکل کلی ایجاد پرس‌وجو

```
SELECT ۱ فیلد ،۲فیلد ،...
FROM نام جدول
WHERE شرط ;
```

فیلد ۱ و فیلد ۲ و... نام فیلدهایی از جدول هستند که داده‌های آن نمایش داده می‌شود. درصورتی‌که بخواهید همهٔ فیلدها را نمایش دهید به‌جای واردکردن نام آنها از نویسهٔ * استفاده کنید. برای اینکه شرط‌های متعددی قرار دهیم، باید تنوع عملگرها را بررسی کنیم.

جدول۶ـ انواع عملگرها در SQL

عملگر	عملکرد	مثال
=	مساوی بودن مقداری را با مقادیر فیلد انتخاب‌شده بررسی می‌کند.	"تولید محتوا"= مقادیری که مساوی عبارت «تولید محتوا» باشد را جست‌وجو می‌کند.
=!	نامساوی بودن مقداری را با مقادیر فیلد انتخاب‌شده بررسی می‌کند.	50 =! مقادیری که با عدد 50 مساوی نباشند را جست‌وجو می‌کند.
IN	مساوی بودن با چند مقدار را بررسی می‌کند.	IN ("الزامات"، "برنامه‌سازی 2") مساوی بودن مقادیر فیلد را با یکی از مقادیر داخل پرانتز بررسی می‌کند.
BETWEEN	برای انتخاب اطلاعات در یک محدوده خاص، در بین دو مقدار تعیین‌شده استفاده می‌شود.	BETWEEN 10 AND 100 بین دو مقدار 10 و 100 را بررسی می‌کند.
IS NULL	NULL بودن مقدار فیلد را مشخص می‌کند.	WHERE b_auther IS NULL رکوردهایی که مقدار نام مؤلف آنها تهی است

مثال: لیست کلیه ی کتاب های موجود در جدول books را نمایش دهید

```
select   *
from   books;
```

مثال: همه کتاب هایی که بین سال های 1380 تا 1390چاپ شده اند را نمایش دهید

```
select   *
from   books
where b_year between 1380 and 1390;
```

مثال: همه کتاب هایی که از ناشر ««شرکت چاپ و نشر کتابهای درسی»» در کتابخانه موجود است را نمایش دهید

```
select   *
from   books
where b_publisher=' شرکت چاپ و نشر کتابهای درسی';
```

مثال: فقط شماره شابک، نام، موجودی و سال چاپ کتابهایی که چاپ ۱۳۹۵ هستند را نمایش دهید.

```
select b_isbn , b_name ,b_qty , b_year
from books
where b_year = 1395 ;
```

مرتب‌سازی رکوردهای جدول

گاهی لازم است رکوردها براساس یک فیلد مرتب شوند.
با استفاده از ORDER BY می‌توان خروجی پرس‌وجو را بر اساس فیلد یا فیلدهای موردنظر به‌صورت صعودی (ASC) و یا نزولی (DESC) مرتب کرد.

شکل کلی مرتب‌سازی رکوردها

```
SELECT ۱ فیلد , ۲ فیلد , ...
FROM نام جدول
ORDER BY ۱ فیلد , ۲ فیلد , ... ASC|DESC ;
```

نکته: اگر نوع مرتب سازی ذکر نشود به صورت پیش فرض asc در نظر گرفته می شود. (مرتب سازی صعودی)

مثال: لیست کتاب های موجود در کتابخانه را بصورت مرتب بر اساس نام کتاب نمایش دهید

```
select   *
from books
order by   b_name;
```

مثال: لیست کتاب های موجود در کتابخانه را بصورت مرتب از بیشترین موجودی به کمترین موجودی نمایش دهید (مرتب سازی نزولی)

```
select   *
from books
order by   b_qty    desc;
```

ویژگی AS (Aliases)

این ویژگی باعث ایجاد یک نام مستعار به‌صورت موقت برای یک فیلد یا نتیجه یک پرس‌وجوی محاسباتی می‌شود.

شکل کلی ایجاد نام مستعار

```
SELECT نام فیلد AS نام مستعار
FROM نام جدول نام پایگاه داده ;
```

مثال: نام و شابک کلیه ی کتاب های موجود در جدول books را نمایش دهید بطوریکه در خروجی فیلد b_name با نام مستعار «نام کتاب» و فیلد b_isbn با نام مستعار «شابک» نمایش داده شود

```
select b_name as 'نام کتاب' , b_isbn as 'شابک'
from books;
```

توابع آماری

هنگام ایجاد پرس و جو با دستور select، اگر نیاز به انجام محاسبات باشد از توابع زیر می توان استفاده کرد:

- تابع avg() : محاسبه میانگین
- تابع sum() : محاسبه مجموع
- تابع count() : محاسبه تعداد
- تابع min() : محاسبه کمترین
- تابع max() : محاسبه بیشترین

مثال: تعداد کل عناوین کتابهای موجود در کتابخانه را به دست آورید

```
select count (*) as 'تعداد عناوین'
from books
```

مثال: تعداد عناوین کتابهای موجود در کتابخانه با چاپ پس از سال ۱۳۹۰ را گزارش دهید

```
select count (*) as 'تعداد عناوین'
from books
where b_ year >1390 ;
```

مثال: نام کتاب با کمترین موجودی

```
select b_name, min(b_qty)
from books;
```

مثال: نام و کمترین موجودی کتاب چاپ شده در سال 1395را به دست آورید

```
select b_name, min(b_qty)
from books
where b_year=1395;
```

مثال: قدیمی ترین کتاب موجود در جدول books را نمایش دهید

```
select b_name, min(b_year)
from books
```

نکته: دستور select بصورت تودرتو نیز قابل استفاده است.

مثال: بین کتاب هایی که بعد از سال ۱۳۹۲ منتشر شده اند نام کتابی که بیشترین موجودی را دارد به دست آورید

```
select b_name, b_qty
from books
where b_qty=( select max(b_qty) from books where b_year>1392);
```

گروه بندی (group by)

در تولید برخی گزارشات نیاز به گروه بندی داده ها است. مثلا برای محاسبه تعداد عناوین کتاب برای هر ناشر نیاز به گروه بندی داده ها بر اساس ناشر است. یا برای محاسبه تعداد موجودی کتاب های هر نویسنده نیاز به گروه بندی داده ها بر اساس نویسنده است. برای تولید این گزارشات از group by استفاده می شود.

شکل کلی گروه‌بندی در SELECT

```
SELECT ۱ فیلد , ۲ فیلد , ...
FROM نام جدول
WHERE شرط ;
GROUP BY ۱ فیلد , ۲ فیلد , ... ;
```

مثال: تعداد عنوان کتاب و مجموع کتاب های **هر مؤلف** را نمایش دهید.

```
SELECT
`b_auther` AS 'نویسنده',
COUNT (`b_auther`) AS 'تعداد عنوان کتاب'
SUM(b_qty) AS 'مجموع'
FROM
`books`
GROUP BY `b_auther`
```

دستور having

برای شرط گذاشتن روی تابع از این دستور استفاده می شود.

شکل کلی HAVING در SELECT

```
SELECT ۱ فیلد , ۲ فیلد , ...
FROM نام جدول
WHERE شرط ;
GROUP BY ۱ فیلد , ۲ فیلد , ... ;
HAVING شرط ;
```

مثال: تعداد عنوان کتاب و مجموع کتابهای **هر مؤلف** به شرط اینکه مجموع کتابهای نویسنده بیشتر از 100باشد را نمایش دهید

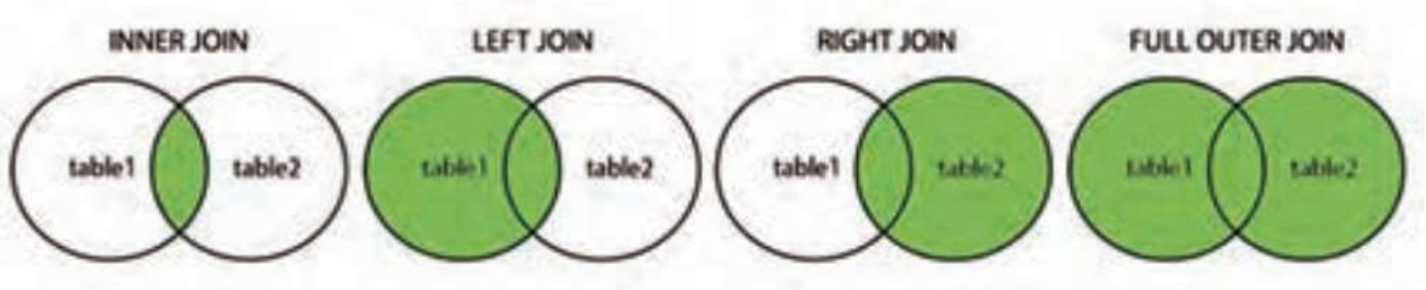
```
SELECT
`b_auther`  AS  'نویسنده',
COUNT (`b_auther`)  AS  'تعداد عنوان کتاب'
SUM(b_qty)  AS  'مجموع'
FROM
`books`
GROUP BY `b_auther`
HAVING SUM(b_qty) > ۱۰۰;
```

پیوند جدول (JOIN TABLE)

این دستور برای ترکیب رکوردهای دو یا چند جدول بر اساس یک ستون مرتبط بین جدول‌ها استفاده می‌شود. عمل ترکیب جداول بر اساس مقدار یک ستون انجام می‌شود. عمل ترکیب جداول بر اساس خواسته‌ای که داریم به روش‌های مختلف انجام می‌شود.

انواع JOIN در SQL

INNER JOIN: مقادیری را برمی‌گرداند که بین دو جدول، مشترک هستند.
LEFT JOIN: همه رکوردهای جدول ۱ و رکوردهایی از جدول ۲ که با هم مطابقت دارند را برمی‌گرداند.
RIGHT JOIN: همه رکوردهای جدول ۲ و رکوردهایی از جدول ۱ که با هم مطابقت دارند را برمی‌گرداند.
FULL JOIN: همه رکوردهایی که در جدول ۱ و ۲ وجود دارند را برمی‌گرداند.



مثال: لیست کتابهایی که به امانت داده شده اند را نمایش دهید (رکوردهای مشترک بین جداول books و loan)

```
SELECT loan.l_ id , loan.b_ ISBN, loan.stu_ id , loan.l _ date , books.b_ name
FROM loan
INNER JOIN books ON loan.b_ ISBN = books.b_ ISBN ;
```

مثال: لیست کامل کتاب ها را نمایش دهید (چه کتاب هایی که به امانت برده شده اند چه کتاب هایی که به امانت برده نشده اند)

```
SELECT
books.b_ name , loan.l_ id , loan.stu_ id , loan.l_ date , loan.b_ ISBN
FROM books
LEFT JOIN loan
ON loan.b_ ISBN = books.b_ ISBN ;
```

پشتیبان گیری از پایگاه داده

پشتیبان گیری راهکاری مناسب برای نگهداری اطلاعات است و در زمان از بین رفتن داده ها، پشتیبان های ایجاد شده، نجات دهنده کسب وکار برای ادامه فعالیت است

- **سربرگ export** در برنامه phpmyadmin برای پشتیبان گیری از داده ها استفاده می شود.
- **سربرگ import** برای بازیابی نسخه پشتیبان استفاده می شود.
- امکان پشتیبان گیری از ساختار جداول(structure) یا داده های جدول (data) امکان پذیر است
- در حین پشتیان گیری امکان فشرده سازی داده ها وجود دارد.
- پشتیبان گیری با فرمت های مختلفی مانند sql, pdf, csv,... قابل انجام است.

در پرتو ایمان پایدار باشید و جاودان

سالارنیا