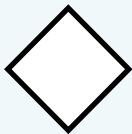


## یادآوری

### مروری بر حل مسئله

مرسوم ترین روش حل مسئله در هنگام نوشتن برنامه، استفاده از الگوریتم است و روندنا جریان کاری الگوریتم را نمایش می دهد. برای ترسیم روندنا از نمادهای زیر استفاده می کنیم.



شرط



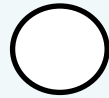
مسیر اجرا



ورودی و خروجی



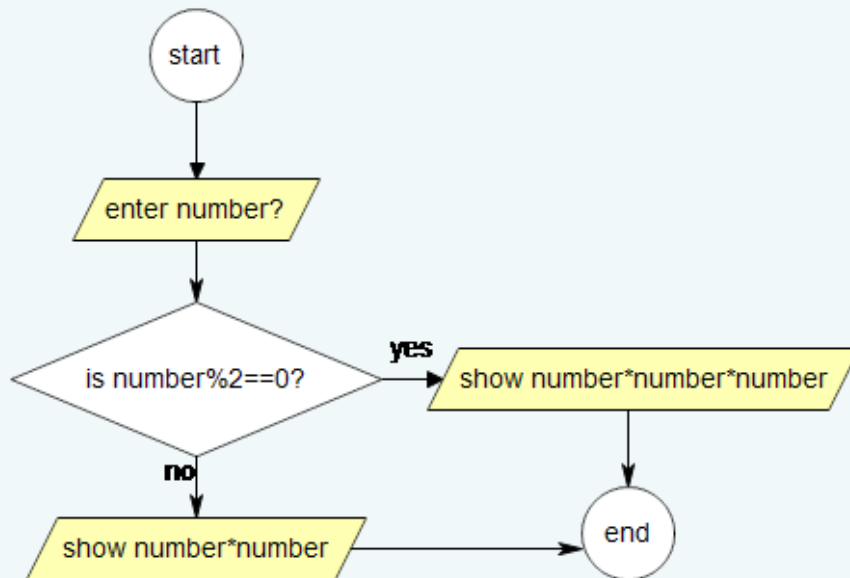
عملیات



شروع و پایان

**مسئله ۱:** الگوریتم و روندنمایی ترسیم کنید که یک عدد دریافت نماید، در صورتی که زوج بود، آن عدد را به توان سه برساند و نمایش دهد و در صورتی که عدد دریافتی فرد بود، مربع آن عدد را نمایش دهد.

**فلوچارت:**



**الگوریتم:**

۰- شروع

۱- عدد (number) را دریافت کن.

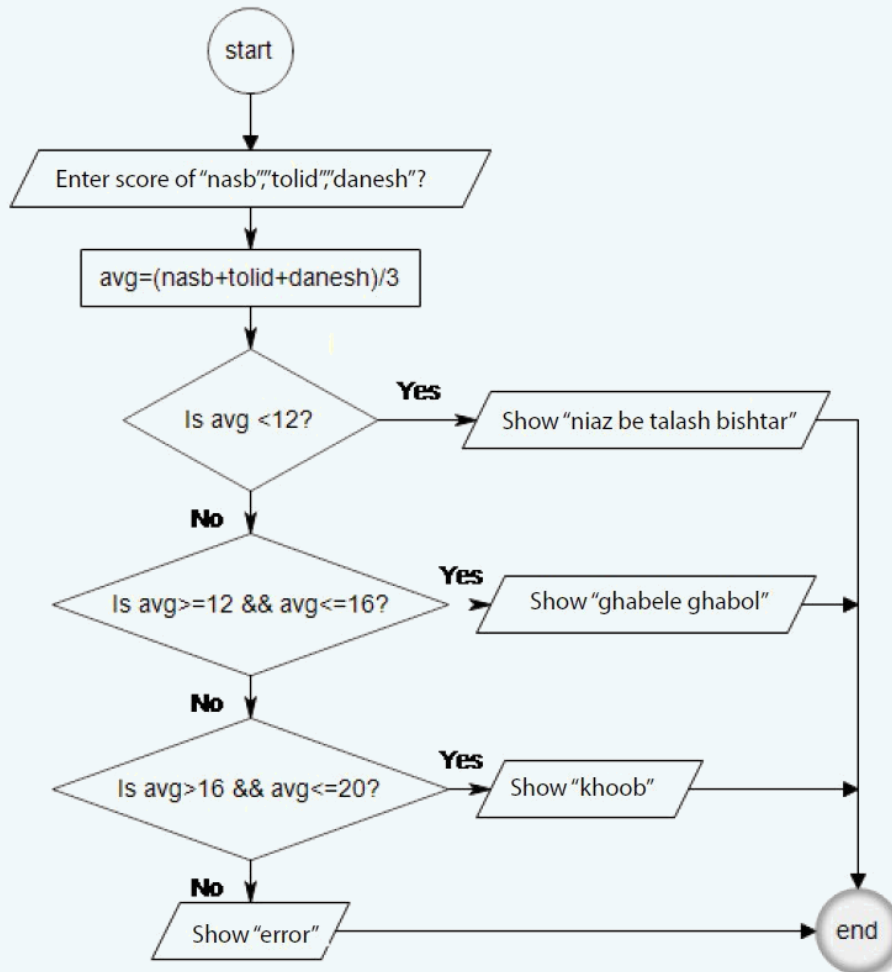
۲- اگر  $number \% 2 == 0$  بود،  $number * number * number$  را نمایش بده در غیر این صورت  $number * number$  را نمایش

بده

۳- پایان

**مسئله ۲:**

الگوریتم و روندنمایی ترسیم نمایید که نمره سه درس دانش فنی، نصب و راه‌اندازی و تولید محتوا یک دانش‌آموز را دریافت کند، اگر معدل این سه درس پائین‌تر از ۱۲ بود، پیغام نیاز به تلاش بیشتر، اگر بین ۱۲ تا ۱۶ بود، پیغام قابل قبول و اگر بالاتر از ۱۶ بود پیغام خوب را نمایش دهد.

**فلوچارت:****الگوریتم:**

۰- شروع

۱- دریافت نمره سه درس تولید محتوا (tolid)، دانش فنی (danesh) و نصب و راه‌اندازی (nasb).

$$avg = (tolid + nasb + danesh) / 3 \quad ۲-$$

۳- اگر  $avg < 12$ ، پیغام نیاز به تلاش بیشتر را نمایش بده در غیر این صورت به مرحله ۴ برو.۴- اگر  $avg \geq 12 \ \&\& \ avg \leq 16$ ، پیغام قابل قبول را نمایش بده در غیر این صورت به مرحله ۵ برو.۵- اگر  $avg > 16 \ \&\& \ avg \leq 20$ ، پیغام خوب را نمایش بده در غیر این صورت پیغام خطا را نشان بده.

۶- پایان

■ نکته: نرم‌افزار **raptor** و **RFFlow** از نرم‌افزارهای قدرتمند هستند که می‌توان با آن‌ها روندنما ترسیم نمود.

## مروری بر زبان C#

- این زبان از زبان‌های سطح بالاست و به زبان انسان نزدیک است.
- این زبان در سال ۲۰۰۰ انتشار یافت و این زبان بر پایه **.netframework** است.
- متد **WriteLine()** برای نمایش پیام در صفحه خروجی استفاده می‌شود. این دستور در کلاس **Console** قرار دارد.
- برای توقف و مشاهده پنجره خروجی، از متد **ReadKey()** که در کلاس کنسول قرار دارد استفاده می‌شود.
- متد **Write()** همانند متد **WriteLine()** است با این تفاوت که پس از نمایش، خط جدیدی درج نمی‌نماید.
- برای ایجاد صدا با فرکانس‌های مختلف، از متد (مدت زمان، فرکانس) **Beep** استفاده می‌شود.
- برای تغییر رنگ قلم نوشته از دستور **ForegroundColor** و برای تغییر رنگ پس زمینه از دستور **BackgroundColor** از کلاس کنسول استفاده می‌شود.

- **ConsoleColor** جعبه رنگ ۱۶ تایی در زبان **C#** است که می‌توان از آن برای رنگ پس‌زمینه و قلم نوشته، استفاده کرد.

☞ مثال: **Console.BackgroundColor=ConsoleColor.black**

- برای تعیین مختصات مکان نما در کنسول می‌بایست از دستور **SetCursorPosition(left,right)** استفاده کرد.
- برای نگهداری نام افراد، نشانی و ... از نوع داده رشته (**string**) می‌توان استفاده کرد.
- برای دریافت از متد **ReadLine()** که در کلاس کنسول قرار دارد استفاده می‌شود.
- برای اعداد صحیح از داده‌های عددی صحیح **byte**، **short**، **int** و **long** استفاده می‌نمائیم.
- نوع داده‌های اعشاری شامل **float** و **double** است.
- در هنگام تعریف ثابت‌ها، عبارت **const** قبل از نام متغیر نوشته می‌شود.
- متد **Parse()** برای تبدیل نوع متغیر مورد استفاده قرار می‌گیرد.
- نوع داده منطقی (**bool**)، می‌تواند مقدار درست (**true**) یا نادرست (**false**) داشته باشد.



## مروری بر دستورات شرطی در زبان C#

- اولویت عملگرهای محاسباتی به ترتیب شامل قرینه (-)، \* / % و + - هستند.

- عملگرهای انتساب از سمت راست عبارت به سمت چپ عبارت مقداردهی می‌شوند.  $x=y=z=1$

- عملگرهای افزایشی و کاهشی شامل ++ جهت افزایش و -- جهت کاهش هستند.

- عملگرهای مقایسه‌ای شامل برابری (=)، نامساوی (!=)، کوچک‌تر (<)، بزرگ‌تر (>)، کوچک‌تر مساوی (<=)، بزرگ‌تر مساوی (>=) اند.

- از عملگرهای پر کاربرد منطقی می‌توان به "و" (&&)، "یا" (| |)، "نقیض" (!) اشاره کرد.

برای تصمیم‌گیری و شرط از متدهای if() else if() و switch() استفاده می‌شود.

### مثال:

۱- برنامه‌ای بنویسید که نام کاربر را سؤال نماید و یک پیام خوشامدگویی به وی اعلام کند.

```
using System;

class Review
{
    static void Main ( )
    {
        Console.Write ("Your Name:" );

        string userName = Console.ReadLine ();

        Console.WriteLine ("Hi " + userName + " ,Welcom Back to C#! ");
    }
}
```

۲- برنامه‌ای بنویسید که از لیست زبان‌های نمایش داده شده، زبان سطح پائین را تشخیص دهد و انتخاب نماید .

```
using System;
class Program
{
    static void Main ()
    {
        Console.WriteLine (" which of the following languages are in the low level language ");
        Console.WriteLine (" a. Assembly");
        Console.WriteLine (" b. C");
        Console.WriteLine (" c. C#");
        Console.WriteLine (" d. JAVA");
        Console.Write ("Select Answer (a Or b Or c Or d)? ");
        string choice = Console.ReadLine();
        if (choice == "a")
            Console.WriteLine ("Bravo! Your Answer is correct");
        else if ((choice == "b") || (choice == "c") || (choice == "d"))
            Console.WriteLine("Your Answer is incorrect");
        else
            Console.WriteLine ("ERROR");
    }
}
```

۳- برنامه‌ای بنویسید که سن یک فرد را براساس سال از ورودی دریافت کند و تشخیص دهد آیا سن وارد شده معتبر است یا خیر، سپس اگر سن آن فرد کم‌تر از ۱۲ بود، پیغام کودک، بین ۱۲ تا ۱۸ پیغام نوجوان، بین ۱۸ تا ۴۰ پیغام جوان، بین ۴۰ تا ۶۵ پیغام میانسال و اگر بالاتر از ۶۵ بود، پیغام کهنسال را چاپ نماید.

```
using System;
class Review
{
    static void Main ()
    {
        float age ;
        string input ;
        Console.Write ("Enter a age:");
        input = Console.ReadLine ();
        age = float.Parse(input);
        if ( (age>=125) || (age<=0) )
            Console.WriteLine ("Invalid Age!");
        else
        {
            Console.WriteLine ("Age is in valid range");
            if (age>=0 && age<=12) Console.WriteLine("you are a child.");
            else if (age>12 && age<=18) Console.WriteLine("you are a teenager.");
            else if (age>18 && age<=40) Console.WriteLine("you are a young.");
            else if (age>40 && age<=65) Console.WriteLine("you are a adult.");
            else if (age>65) Console.WriteLine("you are a old.");
        }
    }
}
```



۴- برنامه‌ای بنویسید که نام رنگ‌های قرمز، آبی، زرد را به لاتین دریافت نماید و سپس بر اساس آن نام فارسی آن رنگ را نمایش و رنگ قلم نوشته را هم رنگ با رنگ انتخاب شده در آورد.

```
using System;
class Review
{
static void Main ( )
{
string color;
Console.WriteLine("enter color.");
color=Console.ReadLine();
switch(color)
{
case "Red":
Console.ForegroundColor=ConsoleColor.Red;
Console.WriteLine("Ghermez.");
break;
case "Blue":
Console.ForegroundColor=ConsoleColor.Blue;
Console.WriteLine("Abii.");
break;
case "Yellow":
Console.ForegroundColor=ConsoleColor.Yellow;
Console.WriteLine("Zard.");
break;
default:
Console.WriteLine("Color Error.");
break;
}
}
}
```

در این برنامه با دریافت یک رشته با نام `color` و استفاده از دستور شرطی `switch()` باعث می‌شود تا براساس مقدار این متغیر، حالت‌های مختلف در نظر گرفته شود.

**مقدمه:** در گذشته‌های دور، اطلاعات به صورت اسناد دست نوشته در بایگانی‌ها نگهداری می‌شدند. این شیوه ذخیره‌سازی اطلاعات کند، حجیم و دشوار بوده است از طرفی با گذر زمان حجم داده نگهداری اطلاعات هر روز افزایش می‌یابد. برای رفع این معضل در سیستم‌های جدید از مفاهیم جدید ذخیره و بازیابی اطلاعات و پایگاه داده استفاده می‌گردد.

برخی از مشکلاتی که در صورت عدم استفاده از پایگاه داده به وجود می‌آید موارد زیر است:

۱- وجود داده‌های تکراری (Redundancy) و احتمال بروز افزونگی و اتلاف حافظه وجود دارد.

۲- وجود تداخل (Conflict) یا مغایرت در داده‌ها (احتمال بروز ناسازگاری داده‌ها)

۳- دشواری در به‌روزرسانی (Update)

تعریف سیستم ذخیره و بازیابی: سیستمی است که به کاربر امکان می‌دهد تا داده‌ها و اطلاعات خود را ذخیره، بازیابی و پردازش نماید. برای راه‌اندازی یک سیستم ذخیره و بازیابی از مدل‌های داده‌ای استفاده می‌شود.

مدل داده‌ای: شیوه‌ای که برای کنترل و دستیابی به داده‌ها و ارتباط بین آن‌ها برقرار می‌کنیم که به سه روش زیر وجود دارد:

۱- رابطه‌ای Relational Data Structure

۲- سلسله مراتبی Hierarchical Data Structure

۳- شبکه‌ای Network Data Structure

■ **نکته:** مدل داده‌ای رابطه‌ای رایج‌ترین شیوه است.

## مفاهیم پایه پایگاه داده رابطه‌ای

۱- محیط عملیاتی (Operational Environment): محیطی است که می‌خواهیم یک سیستم ذخیره و بازیابی (پایگاه داده) براساس موجودیت‌های آن به وجود آوریم همانند محیط عملیاتی مدرسه، کارخانه، اداره و ...

۲- موجودیت (Entity): هر مفهوم (شیء، شخص، محل و...) قرار گرفته در یک محیط عملیاتی که می‌خواهیم درباره آن اطلاعاتی را در پایگاه داده ذخیره نماییم را موجودیت گویند. به عنوان مثال موجودیت‌های یک مدرسه را می‌توان مدیر، معلم، دانش‌آموز، درس، میز، کلاس و ... نام برد.

■ **نکته:** در یک محیط عملیاتی ممکن است موجودیت‌های مختلفی وجود داشته باشد ولی همگی آن‌ها برای آن محیط عملیاتی اهمیت نداشته باشد. فقط موجودیت‌هایی اهمیت دارند که می‌خواهیم در مورد آن‌ها اطلاع داشته باشیم.

۳- ویژگی‌های (صفت‌های) موجودیت: صفاتی که هر موجودیت داراست در حقیقت ویژگی‌هایی هستند که آن موجودیت داراست نظیر نام، نام خانوادگی، نام پدر و ...

■ **نکته:** همه ویژگی‌های یک موجودیت اهمیت ندارند فقط صفاتی مهم‌اند که مرتبط با عنوان محیط عملیاتی و پایگاه داده باشند.

☞ **مثال:** محیط عملیاتی مدرسه را در نظر بگیرید برای ثبت نام و انتخاب واحد می‌خواهیم موجودیت‌ها و ویژگی‌هایشان را در نظر بگیریم:

موجودیت‌ها: ۱- دانش‌آموز ۲- درس ۳- معلم



ویژگی‌های موجودیت دانش آموز:

|                                    |                 |                  |               |
|------------------------------------|-----------------|------------------|---------------|
| (۱) شماره دانش آموزی               | (۲) نام         | (۳) نام خانوادگی | (۴) تلفن      |
| (۵) آدرس                           | (۶) رشته تحصیلی | (۷) معدل         |               |
| ویژگی موجودیت درس: (۱) شماره درس   | (۲) نام درس     | (۳) تعداد واحد   | (۴) ساعت کلاس |
| ویژگی موجودیت معلم: (۱) شماره دبیر | (۲) نام         | (۳) نام خانوادگی | (۴) تلفن      |
| (۵) نام کلاس                       |                 |                  | (۵) آدرس      |

۴- صفت کلیدی (Primary Key): به صفت یا مجموعه‌ای از چند صفت، که سبب یکتایی و منحصر به فردی هر یک از موجودیت‌ها شوند، صفت کلید گوئیم که از طریق آن می‌توان به یک موجودیت مشخص دسترسی داشت.

مثال:

|                  |     |              |         |                 |
|------------------|-----|--------------|---------|-----------------|
| شماره دانش آموزی | نام | نام خانوادگی | نام پدر | تاریخ تولد معدل |
|------------------|-----|--------------|---------|-----------------|

صفت کلید ترکیبی

صفت کلید ساده

۵- ارتباط (Relationship): در هر محیط عملیاتی معمولاً بین موجودیت‌ها ارتباط یا ارتباط‌هایی وجود دارد که قابل شناسایی و معرفی می‌باشد به عنوان نمونه ارتباط موجودیت دانش آموز با موجودیت درس‌های ثبت نام شده یک رابطه چند به چند است.

نکته ۱: در هر ارتباط موجودیت‌هایی شرکت می‌کنند.

نکته ۲: ارتباط نظیر موجودیت می‌تواند ویژگی‌هایی (صفات) برای خود داشته باشد.

نکته ۳: در مدل داده‌ای رابطه‌ای ارتباط بین موجودیت‌ها از طریق صفت کلید صورت می‌گیرد.

۶- ماهیت ارتباط (Cardinality): تناظر بین موجودیت‌ها، را ماهیت ارتباط گویند که به سه نوع زیر می‌باشند:

۱- یک به یک (1:1 one-to-one): یک نمونه از موجودیت اول فقط با یک نمونه از موجودیت دوم ارتباط دارد. نظیر ارتباط سرمربی تیم با تیم مورد نظر، مدیر مدرسه با مدرسه و راننده با ماشین.

۲- یک به چند (1:N one-to-many): یک نمونه از موجودیت اول با چندین نمونه از موجودیت دوم مرتبط است ولی بر عکس آن صادق نمی‌باشد، نظیر ارتباط سرمربی با بازیکنان تیم، راننده با مسافران، رئیس جمهور با مردم کشور و مدیر مدرسه با دانش آموزان.

۳- چند به چند (M:N many-to-many): یک نمونه از موجودیت اول با چندین نمونه از موجودیت دوم مرتبط است و برعکس. نظیر ارتباط دانش آموز با درس (هر دانش آموز چند درس اخذ می‌کند و یک درس به وسیله چند دانش آموز ثبت نام می‌گردد)، یا ارتباط بین هواداران و بازیکنان تیم و رابطه مترجم با کتاب.

۷- نمودار ERD (ارتباط بین موجودیت Entity Relationship Diagram): برای نمایش دادن ارتباط بین موجودیت‌ها و بیان عملکرد ارتباط، از نمودار ERD استفاده می‌شود. به عبارت بهتر نمودار ERD یک مدل سازی معنایی از داده‌هاست.

اجزاء نمودار ER:

(۱) موجودیت‌ها (۲) عملکرد ارتباط (۳) ماهیت ارتباط (۴) ویژگی‌های موجودیت

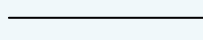
اشکال مورد استفاده در نمودار ER



موجودیت



ویژگی موجودیت



ارتباط بین موجودیت‌ها



عملکرد

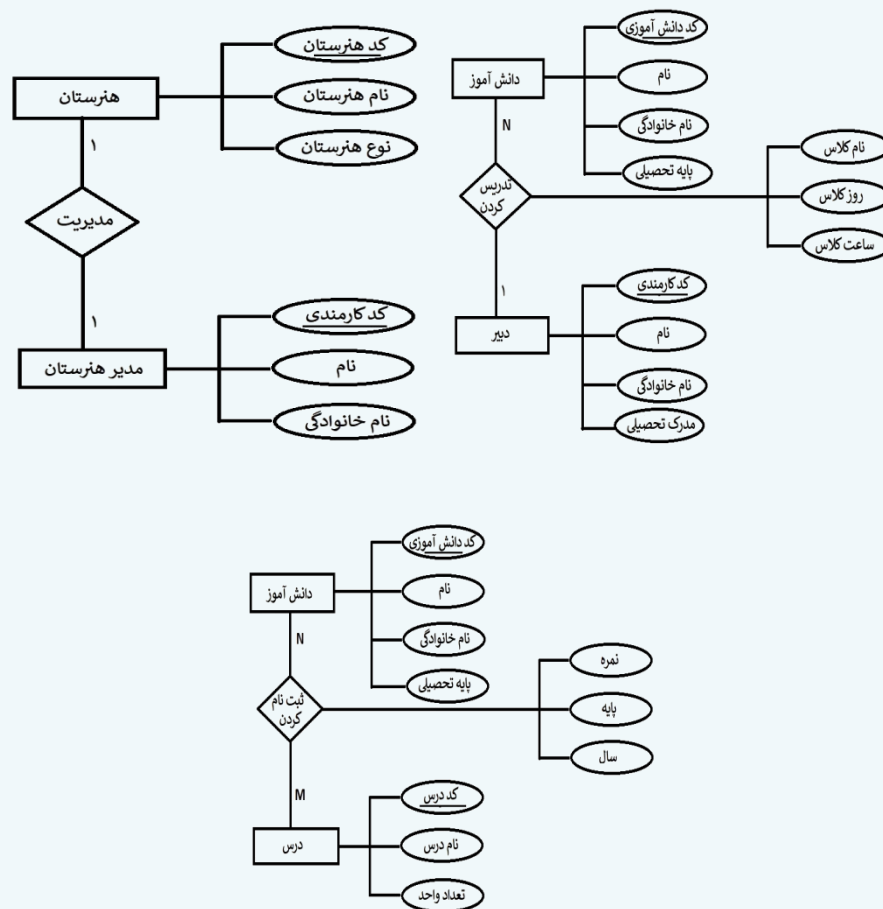


■ **نکته ۱:** موجودیت‌ها از جنس اسم هستند نظیر دانش آموز و درس، ولی ویژگی‌های موجودیت همان صفات قابل شناسایی موجودیت است نظیر نام، معدل و ... اما عملکرد ارتباط از جنس فعل یا مصدر است که ارتباط بین دو موجودیت را بیان می‌کند به عنوان نمونه بین موجودیت درس و دبیر عملکرد تدریس کردن وجود دارد. از عملکرد می‌توان به اخذ کردن، عضو شدن و ارائه کردن اشاره کرد.

■ **نکته ۲:** ماهیت ارتباط روی خطوط متصل بین موجودیت‌ها در دو طرف شکل لوزی ذکر می‌گردد.

■ **نکته ۳:** در نمودار ERD برای مشخص کردن صفت کلید از زیر خط استفاده نمایید.

☞ مثال:



پایگاه داده‌ها (DataBase): یکی از انواع دستگاه‌های ذخیره و بازیابی اطلاعات است که برای سرعت بخشیدن در سازماندهی ذخیره‌سازی و بازیابی اطلاعات توسط کاربر مورد استفاده قرار می‌گیرد به عبارت دیگر به مجموعه‌ای یکپارچه از داده‌ها با حداقل ناسازگاری و افزونگی که تحت کنترل یک سیستم متمرکز و یک مدل داده‌ای استاندارد مدیریت می‌شوند، پایگاه داده‌ای یا بانک اطلاعاتی می‌گویند. این سیستم دارای ویژگی زیر می‌باشد:

۱- افزونگی (تکرار در ذخیره‌سازی) و ناسازگاری در آن حداقل شده است.

۲- طراحی به صورت یکپارچه و با مدیریت متمرکز صورت می‌گیرد.

۳- استفاده بهتر از حافظه و کاهش حجم داده‌ها



■ **نکته ۱:** با ترکیب دست آورده‌های شبکه‌ای، پایگاه داده می‌تواند به وسیله چند کاربر و همزمان استفاده شود.

■ **نکته ۲:** سیستم مدیریت پایگاه داده، یکی از سیستم‌های توسعه یافته ذخیره و بازیابی اطلاعاتی است.

سیستم مدیریت پایگاه داده (DBMS): نرم‌افزاری جامع است که بانک اطلاعاتی و تمامی فایل‌های آن در اختیار این نرم‌افزار قدرتمند قرار می‌گیرد که وظیفه نظارت و کنترل همه عملیات (ذخیره‌سازی، بازیابی، امنیت) را بر عهده دارد. به عبارت بهتر کاربران درخواست خود را به این نرم‌افزار ارسال می‌کنند و در صورت تأیید، کارخواسته شده انجام می‌گیرد. نرم‌افزارهایی نظیر Access, Sql, Oracle و ... از این گروه‌اند.

### مفاهیم مقدماتی در ذخیره و بازیابی

۱- **فیلد (Field):** قطعه داده‌ای است که ویژگی یا صفت خاصی را در مورد یک شیء بیان می‌کند به عبارت دیگر فیلد همان صفت‌های یک موجودیت در محیط عملیاتی می‌باشد نظیر نام، نام خانوادگی، کد دانش‌آموزی و ... هر فیلد از سه جزء به‌وجود آمده است:

| نام فیلد (۱) | نوع فیلد (۲) | مقدار فیلد (۳)            |
|--------------|--------------|---------------------------|
| نام          | نام خانوادگی | معدل → نام فیلد           |
| امیر         | احمدی        | مقدار فیلد → { ۱۲/۵<br>۱۹ |
| رضا          | حسینی        |                           |
| رشته‌ای      | رشته‌ای      | نوع فیلد →                |
|              | عددی         |                           |

۲- **رکورد (Record):** به مجموعه‌ای از چند فیلد مرتبط، رکورد گویند.

مثال: رکورد دانش‌آموز یک کلاس

| نام | نام خانوادگی | نام پدر | سال تولد | معدل |
|-----|--------------|---------|----------|------|
| علی | حسینی        | امین    | ۱۳۷۳     | ۱۶   |
| رضا | امینی        | اکبر    | ۱۳۷۴     | ۱۵/۵ |

↑  
فیلدها

۳- **جدول (Table):** مجموعه‌ای از رکوردهای مرتبط به هم، جدول را به‌وجود می‌آورند. در پایگاه داده هر جدول نماد یک موجودیت می‌باشد

مثال: جدول دانش‌آموزان کلاس کامپیوتر که برای نگهداری اطلاعات موجودیت دانش‌آموز به کار رفته است.

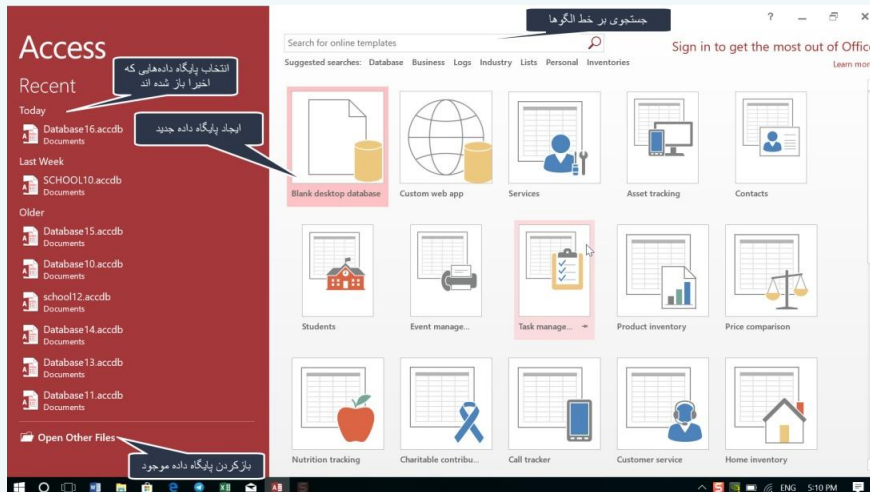
| نام     | نام خانوادگی | نام پدر | سال تولد | معدل  |
|---------|--------------|---------|----------|-------|
| علی رضا | امیدی        | احمد    | ۱۳۷۰     | ۱۷    |
| امیر    | محسنی        | رضا     | ۱۳۷۲     | ۱۴/۲۳ |
| ⋮       | ⋮            | ⋮       | ⋮        | ⋮     |
| امیرعلی | احمدی        | اکبر    | ۱۳۷۱     | ۱۵/۲۵ |

رکورد ۱ ←  
رکورد ۲ ←  
رکورد n ←

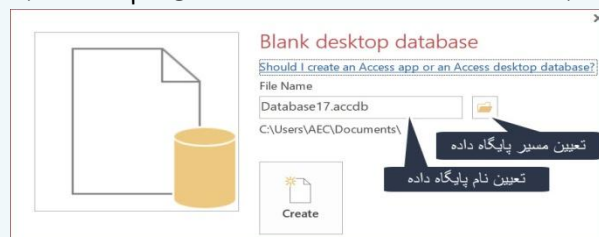
جدول {

## ایجاد پایگاه داده

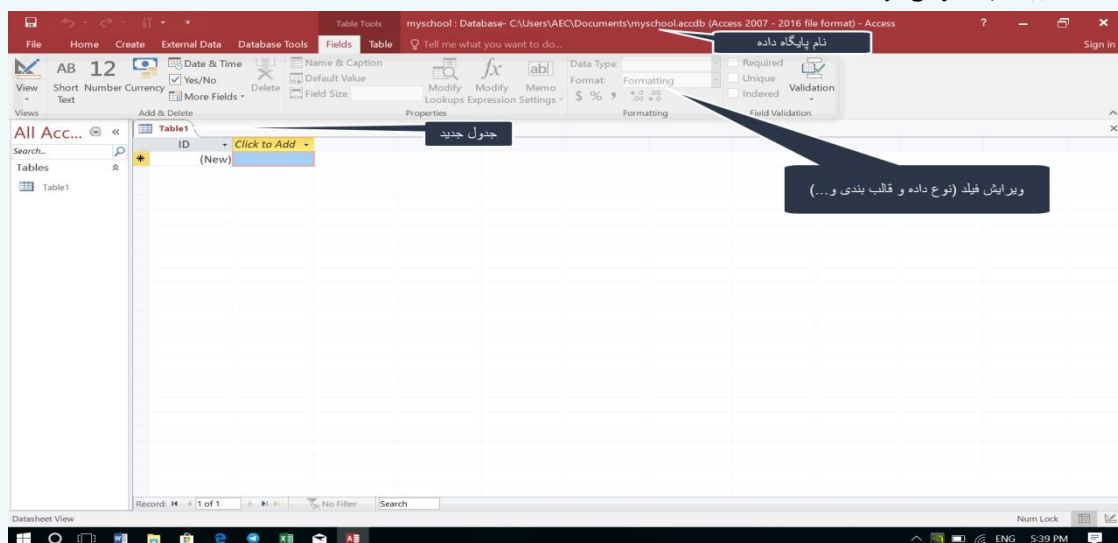
برای ایجاد پایگاه داده از نرم‌افزار Access مجموعه Office 2016 استفاده می‌کنیم. با اجرای برنامه صفحه خوش‌آمدگویی به شکل زیر ظاهر می‌شود.



برای ایجاد پایگاه داده جدید بر روی گزینه **Blank desktop database** کلیک می‌کنیم تا کادر زیر ظاهر شود.

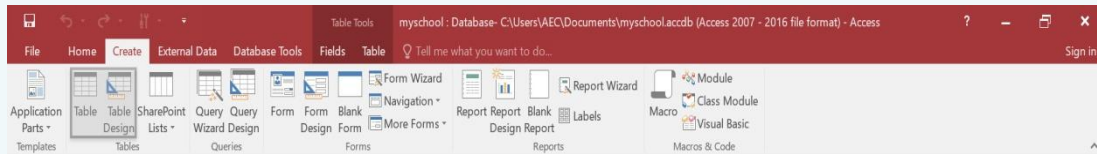


با کلیک بر روی دکمه **Create** پایگاه داده جدید ایجاد می‌شود و به‌طور پیش‌فرض جدول **table1** در آن به‌صورت زیر در نمای **Datasheet** ایجاد و ظاهر می‌شود.

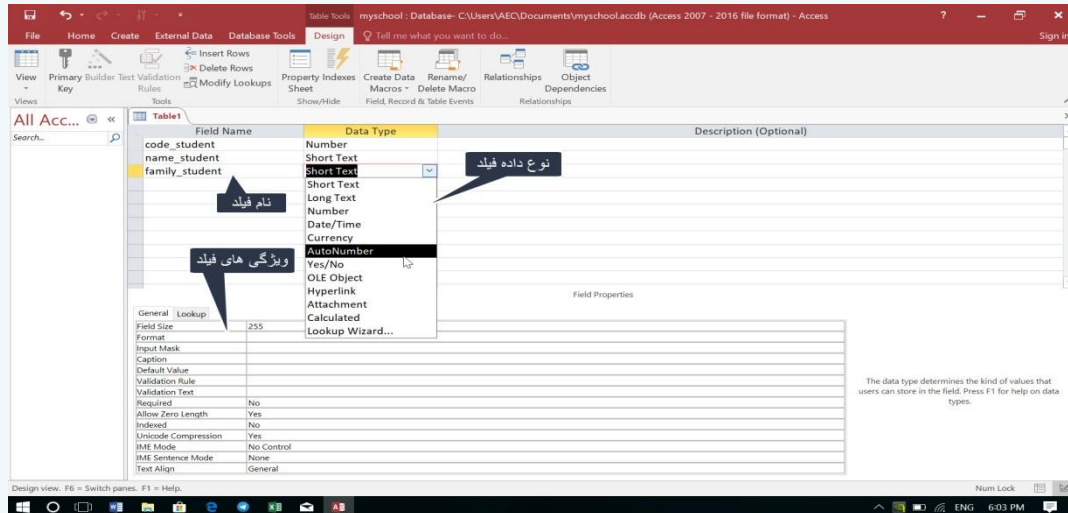




برای ایجاد جدول می توان از زبانه **Create** از بخش **Tables** گزینه **Table Design** و **Table** را مورد استفاده قرار داد.



با انتخاب **Table Design** کاربر می تواند یک جدول جدید برای خود طراحی و به وجود آورد.



نام فیلد نامی است از حروف لاتین و **space** که برای دسترسی به فیلد استفاده می شود. همچنین نوع فیلد نوع داده ای است که در فیلد می تواند قرار بگیرد. در محیط **Access** نوع فیلد به شکل های زیر است:

۱- نوع **ShortText**: نوع داده پیش فرض فیلدها در **Access** می باشد این نوع ترکیبی از حروف و ارقام است که حداکثر تا ۲۵۵ نویسه را شامل می شود. نظیر نام، نام خانوادگی و ...

۲- نوع **LongText**: این نوع ترکیبی از حروف و ارقام است که حداکثر تا یک **GB** نویسه را می تواند در خود ذخیره نماید. نظیر فیلد آدرس و ...

۳- نوع **Number**: انواع اعداد را می پذیرد که شامل اعداد صحیح و اعشاری ممیزدار و سایر اعداد بخش **General** می باشد.

■ **نکته:** طول عدد، صحیح و اعشاری بودن عدد از طریق مشخصه **Field size** معین می گردد.

۴- نوع **AutoNumber**: در هنگام ایجاد رکورد جدید، **Access** به طور خودکار عدد صحیح منحصر به فرد را برای آن جدول، تولید و به این فیلد انتساب می دهد. این فیلد به دو شیوه می تواند اعداد منحصر به فرد را دریافت نماید.

الف) **Increment**: عدد منحصر به فرد به طور سریال تولید می شود و برای هر رکورد نسبت به رکورد قبل بیش تر می گردد.

ب) **Random**: عدد منحصر به فرد به طور تصادفی تولید می شود.

■ **نکته:** هر جدول می بایست دارای یک فیلد کلید اصلی باشد در صورتی که از فیلدهای جدول یعنی صفات موجودیت مرتبط نتوانیم

کلید را مشخص کنیم ایجاد فیلد جدیدی از نوع **AutoNumber** به عنوان فیلد کلید اصلی می تواند مفید باشد.

۵- **Date/Time**: برای نگه‌داری ساعت و تاریخ میلادی و یا ترکیبی از آن‌ها استفاده می‌گردد.

■ **نکته:** نوع **Date/Time** برای تاریخ‌های لاتین می‌باشد برای آنکه تاریخ شمسی را اعلام کنید باید از نوع **ShortText** نوع فیلد را تعیین نمایید.

۶- **Currency**: برای نگه‌داری مقادیر پولی به کار می‌رود و می‌تواند به‌طور خودکار علامت \$ داشته و محل‌های علامت جداساز هزارتایی «کاما» را نگه‌داری کند.

۷- **Yes/No**: برای نگه‌داری مقادیر دو ارزشی **On/Off**, **Yes/No**, **True/False** و یا دیگر مقادیر به کار می‌رود.

۸- **OLE Object**: این نوع از داده‌ها می‌تواند صدا، تصویر، ویدیو، نمودار و یا دیگر اشیاء را در برگرد.

۹- **Hyperlink**: نوعی است که به آدرس وب سایت (منبع اینترنتی) متصل می‌شود.

۱۰- **Attachment**: این نوع داده به شما امکان می‌دهد تا پرونده‌های خارجی را به پایگاه داده **Access** ضمیمه نمایید.

۱۱- **Lookup Wizard**: داده‌ها را از یک فهرست تایپ شده و یا سایر جداول نمایش می‌دهد.

### ویژگی‌های (Properties) فیلد

۱- **Field size**: اندازه نوع فیلد **ShortText** را به تعداد مشخصی نویسه (یک تا ۲۵۵ نویسه) محدود می‌کند و یا نوع فیلد **Number** را به یک دامنه اعداد محدود می‌کند.

■ **نکته ۱:** فیلدهای نوع **ShortText** به طور پیش‌فرض دارای طول ۲۵۵ نویسه می‌باشد.

■ **نکته ۲:** این بخش فقط برای داده‌هایی با نوع **ShortText**, **Number** و **AutoNumber** می‌باشد.

■ **نکته ۳:** **Field size** می‌تواند برای مقادیر نوع **Number** مقادیر زیر را دریافت نماید.

| نوع داده | مقادیر ویژگی   | محدوده اعداد                             | تعداد ارقام اعشار |
|----------|----------------|------------------------------------------|-------------------|
| صحیح     | Byte           | ۰ تا ۲۵۵                                 | -                 |
|          | Integer        | ۳۲۷۶۸- تا ۳۲۷۶۷                          | -                 |
|          | Long Integer   | ۲۱۴۷۴۸۳۶۴۸- تا ۲۱۴۷۴۸۳۶۴۷                | -                 |
| اعشاری   | Single         | $۳/۴ * ۱۰^{۳۸}$ تا $۳/۴ * ۱۰^{۳۸}$       | ۷                 |
|          | Double         | $۱/۷۹۷ * ۱۰^{۳۰۸}$ تا $۱/۷۹۷ * ۱۰^{۳۰۸}$ | ۱۵                |
|          | Replication ID | --                                       | -                 |
|          | Decimal        | $۹/۹۹۹ * ۱۰^{۲۷}$ تا $۹/۹۹۹ * ۱۰^{۲۷}$   | ۱۵                |



۲- Format: قالب نمایش داده‌ها را در نمای Data Sheet مشخص می‌کند.

■ **نکته ۱:** این مشخصه برای تمام نوع داده‌ها به کار می‌رود.

■ **نکته ۲:** این مشخصه نحوه ذخیره شدن داده را در جدول و یا بر نحوه شرکت مقدار عددی ذخیره شده آن در محاسبات تأثیری ندارد.

قالب‌های عددی (نوع Number):

| عدد ورودی              | عدد قالب بندی شده               | توضیح                                                                                                                                     | Format     |
|------------------------|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|------------|
| 7345.937               | 7345.937                        | عدد ورودی را به همان صورتی که وارد شده است نمایش می‌دهد (قالب پیش فرض)                                                                    | General    |
| 137932.75<br>-453269.7 | \$137,932.75<br>\$(-453,269.70) | از کاما به عنوان جداساز هزارتایی استفاده می‌کند و علامت ارز را نمایش می‌دهد. در هنگام نمایش اعداد منفی آن‌ها را داخل پرانتز نمایش می‌دهد. | Currency   |
| 7493456.3<br>49        | 7,493,456.349                   | حداقل یک رقم را نمایش می‌دهد و مقدار پیش فرض Decimal Place=2 است.                                                                         | Fixed      |
| 739.653                | 73965.30%                       | عدد وارد شده را در ۱۰۰ ضرب می‌کند و علامت درصد را به انتهای آن اضافه می‌کند. مقدار پیش فرض Decimal Place=2 است.                           | Percent    |
| 734.95                 | 7.3495E+2                       | اعداد را به صورت نماد علمی توان ۱۰ و با نماد E نمایش می‌دهد.                                                                              | Scientific |
| 324793.19<br>7         | 324,793.197                     | ارقام هزارگان را جدا می‌کند (با دو رقم اعشار)                                                                                             | Standard   |

قالب‌های تاریخ / زمان (Date/Time)

| مقدار قالب بندی شده   | توضیح                                                          | Format       |
|-----------------------|----------------------------------------------------------------|--------------|
| 7/4/99 4:33:10 AM     | برای نمایش تاریخ بدون زمان و یا زمان بدون تاریخ به کار می‌رود  | General Date |
| Tuesday, July 4, 2016 | روز، هفته و ماه به صورت متنی در قالب تاریخ کامل نمایش می‌یابد. | Long Date    |
| 04-Jul-99             | ماه به صورت مخفف رشته‌ای بدون روز هفته نمایش می‌یابد.          | Medium Date  |
| 7/4/99                | تاریخ به صورت عددی با علامت جداکننده "/" نمایش می‌یابد.        | Short Date   |
| 8:35:17 PM            | AM یا PM ثانیه:دقیقه:ساعت                                      | Long Time    |
| 8:35 PM               | AM یا PM دقیقه:ساعت                                            | Medium Time  |
| 20:35                 | دقیقه:ساعت (در قالب ۲۴ ساعتی)                                  | Short Time   |

۳- **Decimal places**: تعداد ارقام سمت راست نقطه اعشار را تعریف می‌کند. این مشخصه در نوع داده‌های **Number** و **Currency** اعشاری نظیر **Decimal**، **Double**، **Single** استفاده می‌شود. توجه کنید در صورتی که ویژگی **Format** مقدار خالی و یا مقدار **General** باشد تنظیمات این ویژگی تاثیر ندارد ولی در سایر موارد می‌توان **Format** تعیین شده را تغییر داد.

۴- **Input Mask**: قالبی است که تعیین می‌کند داده در زمان ورود اطلاعات به چه شکلی دریافت شود. مثلاً برای تلفن موبایل تعریف **0####-#####** نمونه داده **0912-1234567** را دریافت می‌کند. یعنی ابتدای آن صفر و پس از چهارمین رقم، علامت — قرار گیرد و کلاً ۱۱ رقم باشد.

قالب‌ها و مقادیر مشخصه **Input Mask**:

کاربر می‌تواند در هنگام ورود اطلاعات شکل ورودی دریافت هر فیلد را از طریق عبارت‌های مجاز در مشخصه **Input Mask** معین کند.

| کاراکتر  | توضیحات                                                                                                                                                                   |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0        | رقم (صفر تا ۹ و علامت [+ ] یا [- ] مجاز نیست).                                                                                                                            |
| 9        | رقم یا خالی (ورود اطلاعات ضروری نیست و علامت مثبت یا منفی مجاز نیست).                                                                                                     |
| #        | رقم یا فضای خالی (ورود اطلاع ضروری نیست. در زمان ویرایش علامت Space به صورت فضای خالی نمایش داده می‌شود ولی در زمان ذخیره‌سازی حذف می‌گردد. علامت مثبت و منفی مجاز است).  |
| L        | حروف (فارسی از الف تا ی و انگلیسی از A تا Z). ورود اطلاع ضروری است.                                                                                                       |
| ?        | حروف (فارسی از الف تا ی و انگلیسی از A تا Z). ورود اطلاع ضروری نیست.                                                                                                      |
| A        | حروف یا رقم (ورود اطلاع ضروری است).                                                                                                                                       |
| a        | حروف یا رقم (ورود اطلاع ضروری نیست).                                                                                                                                      |
| &        | هر کاراکتری و یا فضای خالی (ورود اطلاع ضروری است).                                                                                                                        |
| C        | هر کاراکتری و یا فضای خالی (ورود اطلاع ضروری نیست).                                                                                                                       |
| / , - :: | جداساز (برای محل رقم هزارگان، تاریخ، زمان و استفاده از کاراکترهای جداساز استاندارد در محیط ویندوز).                                                                       |
| <        | باعث می‌شود تمام کاراکترها به حروف کوچک تبدیل شوند.                                                                                                                       |
| >        | باعث می‌شود تمام کاراکترها به حروف بزرگ تبدیل شوند.                                                                                                                       |
| !        | باعث می‌شود، عبارت از راست به چپ نمایش یابد. کاراکترهایی که در این فیلد تایپ می‌شوند، فیلد را از چپ به راست پر می‌کنند. این علامت در هر جایی از عبارت می‌تواند قرار گیرد. |
| \        | باعث می‌شود کاراکترها به صورت کاراکتر ثابت نمایش داده شوند (مثلاً A\ به صورت A نمایش داده می‌شود).                                                                        |



مثال: در زیر برای فیلدی خاص قالب‌های مختلف را اعمال نموده ایم .

| قالب ورودی            | شرح قالب                                                                      | مثال‌های مجاز                        | مثال‌های غیرمجاز           |
|-----------------------|-------------------------------------------------------------------------------|--------------------------------------|----------------------------|
| LLAAA                 | ۳ حرف یا رقم اجباری → ۲ حرف اجباری                                            | AHMAD<br>AMIR3<br>ALI53              | ALI<br>12ALI<br>ALI3       |
| 000aa                 | ۲ حرف یا رقم اختیاری → ۳ رقم اجباری                                           | 748MK<br>57623<br>379                | 75<br>ALI37<br>7abmn       |
| #990LL                | ۲ حرف اجباری → ۱ رقم اجباری → دو رقم یا خالی → رقم، فضا یا علامت              | +762ab<br>-57xm<br>7440mp            | +53ab<br>+763<br>-435d     |
| >LL<LL0a              | ۱ کارکتر کوچک اختیاری → ۱ رقم اجباری → دو حرف کوچک اجباری → ۲ حرف بزرگ اجباری | ABmn7b<br>MRef53<br>EHdr4            | mnAB7b<br>ABrp7M<br>Emdr4n |
| \0 9 3 5 -<br>0000000 | ۷ رقم اجباری → خط تیره → شروع قالب 0935                                       | 0935-<br>7452123<br>0935-<br>5691473 | 0935-6321                  |
| \N\u m:00             | ۲ رقم اجباری → علامت: → شروع قالب Num                                         | Num:75<br>Num:36                     | Num:5                      |

۵- **Caption**: برجسبی است که به جای نام فیلد، در جدول‌ها، فرم‌ها و گزارش‌ها نمایش داده می‌شود.

۶- **Default Value**: مقدار پیش فرضی است که در زمان ایجاد رکورد جدید اگر مقداری به فیلد مربوطه اختصاص داده نشده باشد در آن ذخیره می‌شود. این مشخصه برای داده‌های نوع Memo و AutoNumber کاربرد ندارد.

۷- **Validation Rule**: شرطی است که داده‌های وارد شده را محدود می‌کند. مثلاً اینکه داده وارد شده نمره‌اش بین صفر تا ۲۰ باشد.

۸- **Validation Text**: متنی است که در زمان عدم احراز شرایط **Validation Rule** به عنوان پیغام خطا یا اخطار نمایش داده می‌شود.

۹- **Required**: مشخص می‌کند در زمان ورود داده، آن فیلد نمی‌تواند خالی رها شود بلکه حتماً باید مقداری به آن نسبت داده شود.

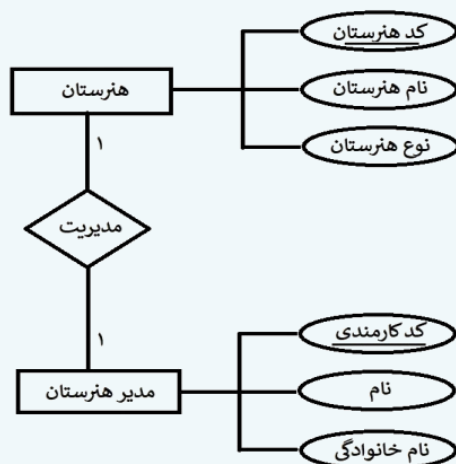
۱۰- **Allow Zero Length**: مشخص می‌کند آیا می‌توان مقدار خالی یعنی "" را به فیلد نسبت داد یا خیر.

۱۱- **Indexed**: بازیافت داده از فیلد را تسریع می‌کند. در صورتی که مقدار این ویژگی برابر با yes باشد معین می‌کند مقدار تکراری در آن فیلد مجاز است یا خیر.

■ **نکته**: فیلدهای کلید اصلی به‌طور پیش فرض **Indexed** شان دارای مقدار yes هست.

**گام‌های تبدیل نمودار ER به جدول**: هر موجودیت مستقل در نمودار ER به یک جدول مجزا تبدیل می‌شود به طوری که صفات موجودیت به فیلدهای جدول ایجاد شده تبدیل خواهد شد. می‌خواهیم نمودار ER زیر را به جدول تبدیل نماییم.





در بانک اطلاعاتی myschool.acddb قرار می‌گیریم جدول جدیدی از طریق شیوه Design View برای مدیر هنرستان به وجود می‌آوریم.

#### فیلم مربوط به این جدول جدید (فایل access\_1)

تبدیل موجودیت‌هایی که ارتباط 1:1 دارند در پایگاه داده، موجودیت‌هایی که ارتباط 1:1 دارند به یک جدول تبدیل می‌شوند. بنابراین می‌توان دو موجودیت هنرستان و مدیر هنرستان را در یک جدول ساماندهی کرد

#### فیلم تغییر ساختار جدول و properties (فایل‌های access\_2, access\_3)

کلید خارجی (Foreign Key): کلید خارجی به فیلدی از جدول گفته می‌شود که به صورت منحصر به فرد و یکتا به رکوردی در جدول دیگر اشاره می‌کند و آنرا از سایر رکوردها متمایز می‌کند به عبارت دیگر کلید خارجی کلید اصلی جدول اول است که به جدول دیگر اضافه شده است که ارتباط بین دو جدول از طریق آن صورت می‌گیرد.

■ نکته ۱: مقدار فیلد کلید خارجی می‌تواند خالی باشد.

■ نکته ۲: مقدار فیلد کلید خارجی می‌تواند تکراری باشد.

تبدیل موجودیت‌هایی که ارتباط 1:N دارند در پایگاه داده و ارتباط بین آن‌ها: موجودیت‌هایی که ارتباط 1:N دارند، کلید اصلی جدول طرف یک به جدول طرف N اضافه می‌شود این کلید در جدول طرف N به عنوان کلید خارجی محسوب می‌گردد. آن‌گاه برای ایجاد ارتباط بین جدول‌ها لازم است بین آن‌ها یکپارچگی صورت گیرد.

ایجاد ارتباط بین جدول‌ها و کنترل یکپارچگی ارتباطها (Referential Integrity): هدف از یکپارچگی این است که وقتی عملیاتی در یک قسمت از بانک صورت گیرد به طور خودکار و موازی در تمام جاهایی که لازم است همان عملیات صورت گیرد تا ناسازگاری و تناقض به وجود نیاید. به عنوان نمونه اگر مقدار فیلد کلید اصلی در جدول اصلی تغییر یابد مقدار فیلد کلید خارجی در جدول جزئیات به طور خودکار بروزرسانی شود و یا به عنوان نمونه هنگام حذف رکورد از جدولی که ارتباط یک به چند با جدولی دیگر دارد باید رکوردهای جدول جزئیات نیز حذف گردد چرا که در غیر این صورت چون اطلاعات اصلی (Msater) حذف شده است رکوردهای جدول دوم (detail) بلا تکلیف می‌ماند.



## شرایط ایجاد یکپارچگی در بانک

- ۱- جدول‌های مرتبط با هم در یک بانک اطلاعاتی قرارداشته باشند.
  - ۲- فیلدهای برقرار کننده در هر جدول، نوع داده‌ای (Data Type) یکسانی داشته باشد.
  - ۳- فیلدی که بین دو جدول ارتباط برقرار کرده در جدول اصلی (Master) کلید اصلی (Primary key) باشد یا اندیس آن منحصر به فرد (Unique) باشد.
- چند پیشگیری مهم به منظور حفظ یکپارچگی در بانک‌ها:

- ۱- نمی‌توانیم در فیلد کلید خارجی (کلید اصلی جدول پدر که به جدول فرزند اضافه شده) مقداری وارد کنیم که در جدول پدر موجود نباشد.
  - ۲- نمی‌توانیم از جدول پدر رکوردی را حذف کنیم که رکوردهای مشابه آن در جدول فرزند وجود داشته باشند مگر اینکه گزینه Cascade Delete Related Records را هنگام ایجاد ارتباط انتخاب کرده باشیم که در این صورت با حذف رکورد در جدول پدر تمام رکوردهای مرتبط به آن در جدول فرزند به طور خودکار حذف می‌شوند.
  - ۳- در صورتی که رکوردهای متناظری در جدول پدر و فرزند وجود داشته باشد نمی‌توانیم فیلد کلید اصلی در جدول پدر را تغییر دهیم مگر اینکه گزینه Cascade Update Related Fields را هنگام ایجاد ارتباط انتخاب کرده باشیم که در این صورت با تغییرات در مقدار فیلد کلید اصلی در جدول پدر تمام فیلدهای مرتبط در جدول فرزند به طور خودکار بروزرسانی می‌شوند.
- نحوه ایجاد رابطه:



- ۱- از نوار ابزار Database Tools آیکون (Relationships) را انتخاب کنید تا پنجره Show Table ظاهر گردد. از طریق پنجره فوق جدول‌هایی که می‌خواهیم با هم ارتباط داشته باشند را با دو بار کلیک کردن انتخاب می‌کنیم. آن‌گاه از فیلدهای مورد نظر جدول اول (کلید اصلی) به فیلد متناظر (کلید خارجی) در جدول دوم، درگ می‌کنیم تا ارتباط برقرار شود. در این صورت پنجره Edit relationship ظاهر شده تا نوع رابطه را معین می‌کنیم.
- در پنجره Edit Relation گزینه‌های زیر قرار دارد:

- ۱- انتخاب گزینه Enforce Referential integrity سبب به وجود آمدن رابطه 1:N می‌شود.
- ۲- انتخاب گزینه Cascade Update Related سبب می‌شود که با تغییر کلید اصلی تمام مقادیر کلیدهای خارجی جداول دیگر تغییر کنند.
- ۳- انتخاب گزینه Cascade Delete Related Records سبب می‌شود که در صورت حذف رکوردی شامل کلید اصلی، تمام رکوردهایی که شامل فیلدهای متناظر با آن هستند حذف می‌شوند.

■ **نکته ۱:** اگر در پنجره هیچ گزینه‌ای انتخاب نشود رابطه به شکل 1:1 (یک به یک) خواهد بود.

■ **نکته ۲:** در رابطه 1:1 یک رکورد از جدول اول به یک رکورد از جدول دوم مرتبط است. برای ارتباط می‌بایست فیلدهای ارتباطی در دو جدول، فیلد کلید اصلی باشند.

■ **نکته ۳:** در رابطه 1:N یک رکورد از جدول اول با چند رکورد از جدول دوم مرتبط است. به جدول طرف ۱ جدول والد (Parent) و به جدول طرف N فرزند (Child) گویند. برای ارتباط صحیح می‌بایست فیلد ربط‌دهنده در جدول طرف ۱ (والد) کلید اصلی باشد ولی فیلد طرف n (فرزند) فیلد کلید خارجی است. فیلد کلید خارجی می‌تواند مقادیر تکراری داشته باشد ولی کلید اصلی خیر.

■ **نکته ۴:** رابطه M:N در حقیقت وجود ندارد و این رابطه از طریق دو رابطه 1:N شبیه‌سازی می‌گردد برای این هدف جدول واسط به وجود می‌آید که در این جدول حداقل دو فیلد کلید اصلی جدول‌های طرف M و N قرار دارد.

■ **نکته ۵:** در هنگام ارتباط و در نوع ارتباط یک به چند، فیلد کلید اصلی جدول «یک» به جدول طرف «چند» اضافه می‌شود اگر در جدول دوم فیلد کلید خارجی همانم با فیلد کلید اصلی در جدول اول نباشد خطایی رخ نمی‌دهد ولی اگر هم‌نوع نباشد پیام خطای زیر ظاهر می‌گردد.

Relationship must be on the same number of fields with the same data types.



فیلم مربوط به این بخش (ساخت ارتباط یک به چند و چند به چند) (فایل‌های access\_4, access\_5)



### ایجاد و کار با پرس‌وجوها

پرس‌وجوها ابزاری هستند که برای جستجو، مشاهده، ویرایش و به‌هنگام‌سازی داده‌های موجود در جدول به کار می‌رود. در حقیقت پرس‌وجو درخواستی برای دریافت داده‌ها و اطلاعات از جدول یا جدول‌های یک پایگاه داده است. خروجی این درخواست می‌تواند به شکل نمودار، نتایج و یا دستورات SQL باشد. این ابزار در پایگاه داده کارهای زیر را انجام می‌دهد:

- به عنوان یک فیلتر عمل نماید و داده درخواستی شما را نمایش دهد.

- اطلاعات موردنظر شما را از چندین جدول استخراج می‌کند و در جدول جدید قرار می‌دهد.

- عملیات محاسباتی ریاضی را بر روی فیلدها انجام می‌دهد.

- عملیات مرتب‌سازی و جستجو را در جدول‌ها انجام می‌دهد.

- رکوردهایی را به جدول اضافه و یا حذف نماید.

- به عنوان یک منبع رکورد برای فرم‌ها و گزارش‌ها استفاده می‌شود.

ایجاد پرس و جو با استفاده از Wizard:

گزینه Query Design => گروه Query => زبانه Create

در کادر ظاهر شده گزینه Simple Query Wizard را انتخاب نمایید و سپس Ok نمایید. از فهرست کشویی Tables/Queries جدول موردنظر را انتخاب کرده سپس از ناحیه Available Fields فیلدهای موردنظر را که می‌خواهید در پرس‌وجو لحاظ شود را از طریق دکمه‌های > و >> به ناحیه Selected Fields منتقل می‌کنیم سپس روی دکمه Next کلیک می‌کنیم و در کادر بعدی پس از وارد کردن نام پرس‌وجو بر روی دکمه Finish کلیک می‌کنیم.



فیلم مربوط به این بخش (ایجاد پرس‌وجو با استفاده از Wizard) (فایل access\_6)



ایجاد پرس‌وجو با استفاده از Design View: ابزاری است که کاربران را قادر می‌سازد پرس‌وجوهای قبلی را ویرایش کنند یا پرس‌وجوی جدیدی را با انتخاب جدول‌ها و فیلدها به وجود آورند.

گزینه Query Design => گروه Query => زبانه Create

از طریق کادر show Table کاربر می‌تواند برای پرس‌وجوی جدید جدول‌ها و یا پرس‌وجوهای مورد نیاز خود را انتخاب نموده و با فشردن دکمه Add آن‌ها را به بخش بالایی کادر select query اضافه نماید. کادر select query دارای دو بخش است: الف) نیمه بالایی: بخش قرارگیری جدول‌ها یا پرس‌وجوهای انتخاب شده که برای پرس‌وجوی جدید استفاده می‌گردد.



ب) نیمه پایینی: بخش تعیین فیلدهایی که می‌خواهیم در خروجی پرس‌وجو مشاهده شوند می‌باشد. این قسمت دارای زیر بخش‌های زیر می‌باشد:

- ۱) **Fields**: از جدول نیمه بالایی فیلدهایی که مورد نیاز در خروجی پرس‌وجو می‌باشد در این قسمت قرار می‌گیرد.
- ۲) **Table**: بخشی است که کاربر نام جدولی که قرار است فیلد مورد نظر از آن جدول انتخاب شود را معین می‌کند.
- ۳) **Sort**: ترتیب نمایش رکوردها را بر اساس فیلد مورد نظر به شکل صعودی (**Ascending**) و نزولی (**Descending**) و یا (**No sorted**) معین می‌کند.
- ۴) **Show**: نمایش یا عدم نمایش فیلد در اجرای پرس‌وجو را معین می‌کند.
- ۵) **Criteria**: معیار و شرطی را برای فیلد مورد نظر در پرس‌وجو در نظر می‌گیرد تا فقط رکوردهایی که دارای شرط مورد نظر باشد را نشان دهد.
- ۶) **Or**: هرگاه چند شرط و عبارت که می‌خواهند با هم اجرا شوند در این قسمت مابقی معیار شرط ذکر می‌گردد.

#### جدول عملگرهای شرطی مورد استفاده در Criteria

| عملگر   | تعریف                                      | مثال               | شرح مثال                                                                                   |
|---------|--------------------------------------------|--------------------|--------------------------------------------------------------------------------------------|
| >       | بزرگ‌تر از                                 | >10                | هرگاه مقدار فیلد بزرگ‌تر از ۱۰ باشد رکورد مربوطه بازیابی می‌گردد.                          |
| <       | کوچک‌تر از                                 | <20                | هرگاه مقدار فیلد کوچک‌تر از ۲۰ باشد رکورد مربوطه بازیابی می‌گردد.                          |
| >=      | بزرگ‌تر یا مساوی                           | >=15               | هرگاه مقدار فیلد بزرگ‌تر یا مساوی ۱۵ باشد رکورد مربوطه بازیابی می‌گردد.                    |
| <=      | کوچک‌تر یا مساوی                           | <=12               | هرگاه مقدار فیلد کوچک‌تر یا مساوی ۱۲ باشد رکورد مربوطه بازیابی می‌گردد.                    |
| =       | مساوی                                      | = 'امیر'           | هرگاه مقدار فیلد برابر با رشته امیر باشد رکورد مربوطه بازیابی می‌گردد.                     |
| <>      | نامساوی                                    | <>"REZA"           | هرگاه مقدار فیلد برابر با رشته REZA نباشد رکورد مربوطه بازیابی می‌گردد.                    |
| AND     | و (شرطها باید درست باشند)                  | >=10 AND <=15      | هرگاه فیلد مورد نظر بزرگ‌تر مساوی ۱۰ و کوچک‌تر مساوی ۱۵ باشد رکورد مربوطه بازیابی می‌گردد. |
| Or      | یا (حداقل یکی از شرطها باید درست باشد)     | "ALI" or "REZA"    | هرگاه فیلد مورد نظر مقدارش برابر ALI یا REZA باشد رکورد مربوطه بازیابی می‌گردد.            |
| Like    | Like شبیه یا مانند "قالب"                  | Like "ALI*"        | تمام رکوردهایی را بازیابی می‌کند که نامشان با ALI شروع می‌شود.                             |
| Between | between A و B بین دو مقدار And B           | Between 10 And 16  | هرگاه فیلد مورد نظر مقدارش بین ۱۰ تا ۱۵ باشد رکورد بازیابی می‌گردد.                        |
| In      | در (...مقدار ۲، مقدار ۱)                   | In ("ALI", "REZA") | هرگاه فیلد مورد نظر مقدارش برابر ALI یا REZA باشد رکورد مربوطه بازیابی می‌گردد.            |
| Is Null | پوچ بودن مقدار فیلد رشته‌ای را مشخص می‌کند | Name Is Null       | تمام رکوردهایی را که مقدار نام آنها تهی باشد بازیابی می‌شود.                               |
| Not     | شرط را عکس می‌کند                          | Not name="REZA"    | هرگاه فیلد مورد نظر مقدارش برابر REZA نباشد رکورد مربوطه بازیابی می‌گردد.                  |

■ **نکته ۱:** عملگر \* در قالب برای هر رشته دلخواه که نمی‌دانیم به کار می‌رود (به‌جای چند حرف) و عملگر ؟ برای هر کاراکتر دلخواه که نمی‌دانیم به کار می‌رود.

■ **نکته ۲:** عملکرد عملگر In مشابه عملگر Or می‌باشد.



### فیلیم مربوط به این بخشی (ایجاد پرس و جو با استفاده از Query Design) (فایل access\_7)



زبان پرس‌وجوی ساخت یافته SQL: SQL، زبان ایجاد پرس‌وجو در پایگاه داده‌هاست که می‌توان از آن برای ایجاد، تغییر ساختار، به‌روزرسانی داده‌ها و بازیابی داده‌های یک پایگاه داده استفاده کرد به عبارت دیگر این زبان به شما امکان ارتباط با پایگاه داده را می‌دهد، غالب پایگاه‌های داده رابطه‌ای نظیر Access از این زبان پشتیبانی می‌کنند. تمام کارهایی که از طریق رابط کاربری پایگاه داده توسط شما انجام می‌گرفت، از طریق دستورات SQL نیز ممکن است. برای آنکه بتوانیم در قسمت کدنویسی SQL قرار گیریم می‌توانیم در بخش Query Design کادر Show Table را با زدن Cancel ببندیم سپس از نوار ابزار گزینه SQL View را انتخاب کنیم یا در قسمت بالا Query (در هنگام طراحی پنجره Design) راست کلیک کنیم آن‌گاه از منوی ظاهر شده گزینه SQL view را انتخاب کنیم. در این صورت پنجره دستورات SQL مشاهده می‌شود.

**دستور Select:** برای انتخاب و استخراج رکوردها از یک یا چند جدول استفاده می‌گردد. همچنین می‌توان این فرمان را در فرمان‌های دیگر نیز استفاده کرد شکل کلی این فرمان به صورت زیر می‌باشد:

نام جدول FROM ... نام فیلد ۲. نام جدول , نام فیلد ۱. نام جدول SELECT

**مثال:** SELECT Name, family FROM student

نام و نام خانوادگی تمامی دانش‌آموزان جدول Student را بازیابی می‌کند.

\* هرگاه بخواهیم تمامی فیلدهای جدول در فرمان SELECT لحاظ شوند به‌جای نام فیلدها از \* استفاده می‌کنیم.

**مثال:** SELECT \* FROM student

تمامی فیلدهای رکوردهای دانش‌آموزان را نمایش و بازیابی می‌کند.

پارامتر Distinct: از بین مقادیر تکراری فقط یک نمونه را باز می‌گرداند.

**SELECT Distinct Name FROM student**

نام رکوردهایی از جدول student را نشان می‌دهد به طوری که اگر نام برای چندین رکورد یکسان باشد فقط یک نمونه از آن را بازیابی می‌کند.

**SELECT Distinct Name, family FROM student**

نام و نام خانوادگی رکوردهایی از جدول student را نشان می‌دهد به طوری که اگر نام و نام خانوادگی برای چندین رکورد یکسان باشد فقط یک نمونه از آن را بازیابی می‌کند.

■ **نکته:** هرگاه در فرمان SELECT فیلدهای انتخابی بخواهند از جدول‌های مختلف در نظر گرفته شوند به شکل زیر قبل از نام فیلد

نام فیلد ۱. نام جدول

می‌بایست نام جدول ذکر شود.

شرط WHERE

پارامتر WHERE: تمام رکوردهایی که دارای شرایط ذکر شده می‌باشد بازیابی می‌کند.

**SELECT \* FROM student WHERE Average >=12**

**مثال:**

تمام رکوردهایی را بازیابی می‌کند که معدل شان بزرگ‌تر مساوی ۱۲ باشد.



**SELECT name , family, Average FROM student WHERE Name="رضا"**

نام، نام خانوادگی و معدل تمام رکوردهایی را بازیابی می کند که نام آن ها "رضا" باشد.

■ **نکته ۱:** در هنگام به کار بردن شرط می توان از عملگرهای AND و OR نیز استفاده کرد.

مثال:

**SELECT name, family, Average FROM student WHERE name="علی" Or name="امین"**

تمام رکوردهایی را بازیابی می کند که نامشان "علی" یا "امین" باشد.

توجه: مثال فوق را می توان از طریق عملگر In نیز نوشت.

**SELECT name, family, Average FROM student WHERE name In ("علی", "امین")**

مثال:

**SELECT name, family, Average FROM student WHERE name Is Null**

تمام رکوردهایی را بازیابی می کند که نامشان تهی باشند و مقداری در آن قرار نداشته باشد.

مثال:

**SELECT name, family, Average FROM student WHERE name Is Not Null**

تمام رکوردهایی را بازیابی می کند که نامشان دارای مقدار باشند (تهی نباشند).

مثال:

**SELECT name, family, Average FROM student WHERE Average >=12 And Average <=15**

تمام رکوردهایی را بازیابی می کند که معدل شان بزرگتر مساوی ۱۲ و کوچکتر مساوی ۱۵ باشد.

توجه: مثال فوق را می توان از طریق عملگر between نیز نوشت.

**SELECT name,family, Average FROM student WHERE Average between 12 and 15**

■ **نکته ۲:** در هنگام به کار بردن شرط می توان از عملگر Like و کاراکترهای جایگزین \* و ? استفاده کرد.

مثال:

**SELECT name, family, Average FROM student WHERE name Like "علی\*"**

تمام رکوردهایی را بازیابی می کند که نامشان با "علی" آغاز گردد.

مثال:

**SELECT name, family, Average FROM student WHERE name Like "علی\*"**

تمام رکوردهایی را بازیابی می کند که نامشان به "علی" ختم گردد.

مثال:

**SELECT name, family, Average FROM student WHERE name Like "علی\*"**

تمام رکوردهایی را بازیابی می کند که در نامشان "علی" قرار داشته باشد (هرجای نامشان، چه ابتدا چه انتها چه وسط نامشان).

مثال:

**SELECT name, family, Average FROM student WHERE not name Like "علی\*"**

تمام رکوردهایی را بازیابی می کند که در نامشان "علی" قرار نداشته باشد (هرجای نامشان، چه ابتدا چه انتها چه وسط نامشان).

پارامتر **ORDER BY**: این پارامتر برای مرتب‌سازی رکوردهای جدول به کار می‌رود.

, ... ASC یا Desc نام فیلد ۲. نام جدول, ASC یا Desc نام فیلد ۱. نام جدول **ORDER BY**

**ASC**: مرتب‌سازی به‌صورت صعودی است یعنی از مقدار کم‌تر به مقدار بیش‌تر مرتب می‌کند.

**DESC**: مرتب‌سازی به‌صورت نزولی است یعنی از مقدار بیش‌تر به مقدار کم‌تر مرتب می‌کند.

■ **نکته ۱:** اگر **ORDER BY** ذکر نشود رکوردها بدون هیچ ترتیب خاصی و براساس همان ترتیب ورود اطلاعات نمایش داده می‌شوند.

■ **نکته ۲:** هرگاه حالت مرتب‌سازی ذکر نگردد مرتب‌سازی به‌صورت صعودی است.

■ **نکته ۳:** هرگاه در پارامتر **ORDER BY** دو یا چند فیلد برای مرتب‌سازی ذکر گردد ترتیب مرتب‌سازی براساس قرارگیری فیلدها

از چپ به راست است. یعنی ابتدا فیلدهای براساس اولین فیلد ذکر شده مرتب می‌شوند در صورتی که دو رکورد با هم یک مقدار داشته باشند مرتب‌سازی براساس فیلدهای بعدی صورت می‌گیرد.

☞ **مثال:**

**SELECT name, family Average FROM student ORDER BY name, Average DESC**

تمام رکوردهای جدول دانش‌آموز را براساس نامشان به‌صورت صعودی مرتب می‌کند در صورتی که دو رکورد همنام باشد براساس معدل‌شان به‌صورت نزولی مرتب می‌گردد.

ویژگی **AS (Aliases)**: هرگاه بخواهیم برای یک ستون به‌جای نام فیلد یا نتیجه یک عبارت، نام مستعار جدید را به کار ببریم از عبارت **AS** به شکل زیر بهره می‌گیریم.

نام جدید **AS** نام فیلد یا عبارت

☞ **مثال:**

**SELECT name AS نام , family AS فامیلی , Average AS معدل FROM student**

■ **نکته ۵:** در هنگام استفاده از فرمان **SELECT** می‌توانید عبارت‌های محاسباتی را به کار ببرید.

**SELECT name, family, vb, access, network, (vb+access+network) /3 FROM class**

☞ **مثال:**

توجه کنید ستون میانگین نمرات با عنوان **expr1005** نمایش داده می‌شود.

**SELECT name, family, vb, access, network, (vb+access+network) /3 AS میانگین FROM class;**

پارامتر **GROUP BY**: رکوردهایی بازبایی می‌شوند که در چند فیلد مشترک هستند نام این فیلدها در جلوی **GROUP BY** قرار می‌گیرند.

, ... نام فیلدها **GROUP BY**

**SELECT name FROM class GROUP BY name**

☞ **مثال:**