Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

آموزش SQL

نویسنده : مجتبی اسکندری

SQL چیست؟

SQL بر پایه زبان پرس و جو ساخت یافته میباشد
SQL به شما اجازه دستیابی و کنترل داده ها را می دهد
SQL یک استاندارد ANSI(انجمن استاندارد ملی آمریکا) میباشد.

SQL چه کاری انجام میدهد؟

SQL میتواند درخواستهای پیوسته یک پایگاه داده را اجرا کند
SQL متواند دوباره اطلاعات را از پایگاه داده پس بگیرد
SQL میتواند یک رکورد شامل اطلاعات را در پایگاه داده ذخیره کند
SQL متواند اطلاعات پایگاه داده را به روز رسانی کند
SQL میتواند هر قسمت از اطلاعات را از پاگاه داده اصلاح یا حذف کند
SQL میتواند یک پایگاه داده تازه بسازد
SQL میتواند جداول حاوی اطلاعات جدید را به پایگاه داده اضافه کند
SQL اجازه تنظیم جداول و شیوه دستیابی به اطلاعات و نحوه نمایش اطلاعات را میدهد.

SQL یک استاندارد است اما ...

با وجود آنکه SQL یک استاندارد است اما ورژن های مختلفی از آن وجود دارد

به هرحال SQL یک استاندارد ANSI میباشد و با آن مطابقت دارد و از تمام دستورات اصلی از طریق مشابه پشتیبانی میکند مثل(SELECT,UPDATE ,DLELET,INSERT,WHERE)
نکته : بسیاری از برنامه های پایگاه داده ای SQL شاخه های اختصاصی خودشان را به استاندارد SQL اضافه میککنند.

استفاده از SQL در وب سایت خود

اگر میخواهید وب سایتی بنویسید که از پایگاه داده اطلاعاتی را نمایش دهد به ابزار زیر احتیاج پیدا خواهی کرد:
یک برنامه RDBMS (MY SQL,SQL SERVER, MS ACCESS)
یک زبان سمت کاربر مانند ASP, PHP,
SQL
HTML/CSS

RDBMS (سیستم مدیریت پایگاه داده ای)

RDBMSبر پایه RELATIONAL DATABASES MANAGEMENT SYSTEM میباشد.
RDBMSپایه ای برای SQL و تمام سیستم های پایگاه داده ای جدید SQL,) MYSQL,MS ACCESS) ORACLE, SYBASE,DB2 ,SERVERمیباشد.
در RDBMS داده ها و اطلاعات موجود در پایگاه داده در جداول ذخیره میگردد.
جدول یک مجموعه ای است از اطلاعات ثبت شده مرتبط و وابسته به هم که از ستون ها و ردیف ها تشکیل شده است .

جداول پایگاه داده

یک پایگاه داده اغلب شامل یک یا چند جدول میباشد .هر جدول با نامی شناخته می شود مثلا("customers"","order") و جداول شامل اطلاعاتی هستند که در آنها ثبت شده است ویا بعدا ثبت می شود.
جدول زیر مثالی است با نام "person"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Bayat	ALI	Nakoja	Mashhad
3	Davarnia	ALI	Nakoja	Mashhad

جدول بالا شامل ۳ رکورد اطلاعاتی (برای هر شخص یک ردیف) و۵ ستون (P_Id,LastName, FirstName , Address, City) میباشد.

گزاره های SQL

بیشتر اعمالی که شما در پایگاه داده به آنها احتیاج دارید تا انجامشان دهید با گزاره های SQL انجام میشود.
گزارهٔ SQL زیرتمام رکورد های اطلاعاتی موجود در جدول "persons" را انتخاب میکند.
SELECT * FROM Persons
ما سعی داریم دراین خود آموز تمام گزاره های مختلف SQL را به شما آموزش دهیم.
راستی بخاطر بسپارید که...
SQL تفاوتی بین حروف بزرگ یا کوچک قائل نیست!! یعنی select=SELECT .

سمی کلن (ویرگول نقطه!!) بعد از گزاره های SQL ؟...

بعضی از سیستم های مدیریت پایگاه داده به سمی کلن در انتهای گزاره های SQL احتیاج دارند.
سمی کلن راهی استاندارد است برای جدا کردن گزاره های SQL درسیستم های پایگاه داده ای که اجازه اجرای بیش از یک گزاره SQL در آنها برای سرویس گرفتن از سرویس دهنده وجود دارد
در کار با MS ACCESS, SQL SERVER 2000 هیچ احتیاجی به استفاده از سمی کلن نمیباشد اما در برخی از سیستم های پایگاه داده ای ما مجوریم از سمی کلن استفاده کنیم.

SQL,DML and DDL

SQL را میتوان به ۲ قسمت تقسیم کرد : (LANGUAGE DML,DATA MANIPULATION) زبان دستکاری داده هاو (DDL ,DATA DEFINITION LANGUAGE) زبان تعریف داده ها.

فرم برخی دستورات DML قسمتی از SQL عبارت است از:

SELECT : اطلاعات را از پایگاه داده بیرون میکشد
UPDATE: اطلاعات موجود را بروزرسانی میکند
DELETE: اطلاعات را از پایگاه داده حذف میکند
INSERT INTO: اطلاعات جدید را وارد پایگاه داده میکند
قسمست دیگر SQL یعنی DDL اجازهٔ ساخت و یاحذف جداول پایگاه داده را میدهد و همچنین زیروندهایی تعریف میکند(کلیدهایی) که موجب ارتباط بین جداول می شود و موجب محدودیت هایی در این رابطه می گردد.

مهمترین گزاره های DDL در SQL عبارتند از:

CREATE DATABASE: پایگاه داده جدید میسازد.
ALTER DATABASE: امکان تغییر در پایگاه داده را می دهد.
CREATE TABLE: امکان ساختن جدول جدید را می دهد.
ALTER TABLE: امکان تغییر در جداول را ایجاد میکند.
DROP TABLE: امکان حذف جدول را می دهد.
CREATE INDEX: ساختن یک شاخص(کلیدهای جستجو).
DROP INDEX: حذف شاخص .

عبارت SELECT برای انتخاب داده ها از پایگاه داده استفاده میشود.
نتایج در یک جدول نتایج ذخیره میشود که به آن مجموعه نتایج(RESULT-SET) میگویند (مجموعه نتایج در SQL قسمتی از جدول پایگاه داده ای است که اطلاعات در آن ذخیره میشود)

دستور عبارت SELECT در SQL

SELECT column_name(s)FROM table_name

و

SELECT * FROM table_name

یادآوری میکنم که حروف بزرگ و کوچک در SQL تفاوتی ندارند.

مثالی از عبارت SQL

نام جدول "PERSON":

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Bayat	ALI	Nakoja	Mashhad
3	Davarnia	ALI	Nakoja	Mashhad

حالا میخواهیم محتویات ستونی را که "LastName " و"FirstName" نامیده شده اند از جدول بالا انتخاب کنیم.
برای اینکار از عبارت SELECT به شکل زیر استفاده میکنیم.

SELECT LastName,FirstName FROM Persons

مجموعهٔ نتایج به شکل زیر میباشد.

LastName	FirstName
Eskandari	Mojtaba
Bayat	ALI
Davarnia	ALI

مثالی از SELECT *

حال میخواهیم تمام ستون های جدول PERSON را انتخاب کنیم ، برای اینکار از عبارت SELECT * به شکل زیراستفاده میکنیم

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

SELECT * FROM Persons

جدول نتایج همان جدول PERSON است که با تمام ستون هایش انتخاب شده است.

پیمایش در جدول نتایج

بسیاری از سیستم های نرم افزاری پایگاه داده این اجازه را به کاربر میدهند تا در جدول نتایج(RESULT-SET) با توابع دستوری مانند Move-To-First-Record, Get-Record-Content, Move-To-Next-Record و غیره پیمایش نماید .
عبارات اینچنینی موضوع این خود آموز نیست اما برای یادگیری این دستورات به PHP TUTORIAL ویا ADO TUTORIAL مراجعه کنید .

ممکن است بعضی ستون ها در جدول پایگاه داده ما دارای اطلاعات تکراری باشد مشکلی ازاین بابت نیست .
شاید شما بخواهید لیستی با اطلاعات متفاوت وغیرتکراری بسازید !
در اینجا کلمه کلیدی DISTINCT است که به شما کمک میکند.

نحوه استفاده از دستور Distinct در SQL

SELECT DISTINCT column_name(s)FROM table_name

مثالی از SELECT DISTICNT
جدول با نام PERSON:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Bayat	ALI	Nakoja	Tehran
3	Davarnia	ALI	Nakoja	Mashhad

حالا ما میخواهیم فقط اطلاعات متفاوت را ازستون CITY جدول بیرون بکشیم.
از دستور زیر استفاده میکنیم.

SELECT DISTINCT City FROM Persons

نتیجه حاصل به شکل زیر خواهد بود

City
Mashhad
Tehran

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

عبارت WHERE فقط برای بیرون کشیدن رکورد های اطلاعاتی بر اساس معیاری مشخص استفاده میشود.

نحوه استفاده از عبارت WHERE در SQL

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

مثالی از عبارت WHERE

نام جدول person :

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Bayat	ALI	Nakoja	Tehran
3	Davarnia	ALI	Nakoja	Mashhad

حالا میخواهیم که از جدول بالا آن دسته از اسامی افراد را که درشهر Sandnes زندگی می کنند رابیرون بکشیم برای اینکار از دستور زیر استفاده میکنیم:

```
SELECT * FROM Persons
WHERE City=' Mashhad '
```

نتیجه به شکل زیر می باشد:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Davarnia	ALI	Nakoja	Mashhad

کوتیشن (' ') اطراف فیلدهای متنی در SQL:

SQL از کوتیشن یگانه (' ') اطراف پارامتر های متنی خود استفاده می کند (در برخی سیستم های پایگاه داده ای از کوتشن دوگانه (" ") نیزمیتوان استفاده میکنند)
و باید گفت که برعکس فیلدهای متنی ، فیلدهای عددی (پارامتر های عددی) باید بدون استفاده از کوتیشن به کار برده شوند.

برای فیلدهای متنی اینگونه صحیح میباشد:

```
SELECT * FROM Persons WHERE FirstName='MET'
```

واستفاده نکردن از کوتیشن اشتباه است :

SELECT * FROM Persons WHERE FirstName=MET

برای فیلدهای عددی اینگونه صحیح است:

SELECT * FROM Persons WHERE Year=1368

واستفاده از کوتیشن اشتباه است:

SELECT * FROM Persons WHERE Year='1368'

عملگرهای مجاز در عبارت WHERE:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	If you know the exact value you want to return for at least one of the columns

نکته : در برخی از ورژن های جدید SQL میتوان به جای عملگر <> از =! استفاده کرد.

عملگرهای AND و OR

عملگر AND رکورد اطلاعاتی را نمایش میدهد که هم شرط اول و هم شرط دوم برقرار باشد.
عملگر OR رکورد اطلاعاتی را نمایش میدهد که هریک از شروط اول یا دوم برقرار باشد.

مثالی برای AND:

نام جدول PERSON است:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad

2	Bayat	ALI	Nakoja	Tehran
3	Davarnia	ALI	Nakoja	Mashhad

حالا میخواهیم فقط شخصی را از جدول انتخاب کنیم که نامش Mojtaba و (AND) نام خانوادگیش Eskandari باشد.
از دستور زیر استفاده میکنیم:

```
SELECT * FROM Persons
WHERE FirstName='Mojtaba'
AND LastName='Eskandari'
```

نتیجه به شکل زیر میباشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad

مثالی برای OR:

حالا میخواهیم شخصی را از جدول انتخاب کنیم که نامش Mojtaba یا ALI باشد برای اینکار از دستور زیر استفاده میکنیم.

```
SELECT * FROM Persons
WHERE FirstName='Mojtaba'OR
FirstName='ALI'
```

نتیجه به شکل زیر می باشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nakoja	Mashhad
2	Bayat	ALI	Nakoja	Tehran

ترکیب AND و OR :

شما میتوانید دو عملگر AND و OR را با هم ترکیب کنید، فقط برای اینکار کافیست از پرانتز استفاده کنید.
حال میخواهیم شخصی را با نام خانوادگی Svendson و(AND) نام MOJTABA یا(TOVE OR) از جدول انتخاب کنیم.

```
SELECT * FROM Persons WHERE
LastName='Bayat'
AND (FirstName='ALI' OR FirstName='Mojtaba')
```

با استفاده از دستوربالا به نتیجه زیر میرسیم

P_Id	LastName	FirstName	Address	City
2	Bayat	ALI	Nakoja	Tehran

عبارت ORDER BY رکورد های اطلاعاتی را بر اساس ستونی مشخص مرتب میکند
این عبارت به طور پیش فرض اطلاعات را صعودی مرتب میکند، اگر شما تمایل دارید اطلاعات خود را به طور نزولی مرتب کنید باید از عبارتDESC استفاده کنید .
دستور ORDER BY:

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

ASC(ascending order) مرتب کردن به صورت صعودی و DESC(descending order) مرتب کردن به صورت نزولی میباشد.

مثال :

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Tom	Vingvn 23	Stavanger

فرض کنید میخواهیم از جدول بالا لیست تمام افراد را بیرون بکشیم در حالی که میخواهیم این اسامی بر اساس نامشان مرتب شده باشند.
از دستور زیر استفاده میکنیم:

```
SELECT * FROM Persons
ORDER BY LastName
```

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
4	Nilsen	Tom	Vingvn 23	Stavanger

3	Pettersen	Kari	Storgt 20	Stavanger
2	Svendson	Tove	Borgvn 23	Sandnes

ORDER BY به وسیله دستور DESC

حالا افراد را از جدول بالا خارج میکنیم به طوری که بر اساس نام خانوادگیشان و به صورت نزولی مرتب شده باشند.
برای این کار از دستور زیر استفاده میکنیم .

```
SELECT * FROM Persons
ORDER BY LastName DESC
```

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Tom	Vingvn 23	Stavanger
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes

این عبارت جهت ورود اطلاعات یا درواقع اضافه کردن یک ردیف به جدول مورد نظر میباشد
این عمل به دو صورت انجام میشود :
حالت اول اینکه اطلاعات را بدون ذکر نام ستون مورد نظر وارد کنیم به فرض مثال:

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

حالت دوم هم اطلاعات را در جدول در محل مورد نظر با ذکر نام ستون وارد کنیم:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

مثال:
نام جدول person میباشد:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

حال میخواهیم اطلاعات جدید را در جدول بالا وارد کنیم: با استفاده از دستور زیر داریم:

```
INSERT INTO Persons
VALUES (4,'Nilsen', 'Johan', 'Bakken 2', 'Stavanger')
```

نتیجه:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger

وارد کردن اطلاعات در یک ستون مشخص :

همانطور که بالا توضیح دادم شما میتوانید اطلاعات مورد نظر را در یک ستون خاص وارد کنید
مثال:
دستور sql زیر اطلاعات جدید را در ردیف جدید وارد جدول میکند اما اطلاعات فقط در ستونهای ""P_Id،"LastName"،"FirstName"" وارد میشود

```
INSERT INTO Persons (P_Id, LastName, FirstName)
VALUES (5, 'Tjessem', 'Jakob')
```

نتیجه به شکل زیر میباشد

P_Id	LastName	FirstName	Address	City

1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

این عبارت جهت به روز کردن اطلاعات موجود در جدول استفاده می شود.
نحوه استفاده از این دستور به صورت زیر است:

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

نکته : به عبارت where در دستور update دقت کنید . اگر شما از این عبارت استفاده نکنید تمام رکورد های جدول شما update میشود.

مثالی از دستور UPDATE:

نام جدول طبق معمول person است!

P_Id	**LastName**	**FirstName**	**Address**	**City**
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

حال ما اطلاعات شخص Tjessem را در جدول بالا update میکنیم.

با استفاده از دستور زیر داریم :

```
UPDATE Persons
SET Address='Nissestien 67', City='Sandnes'
WHERE LastName='Tjessem' AND FirstName='Jakob'
```

جدول ما به شکل زیر خواهد شد:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

هشدار مهم درباره دستور UPDATE!

همانطور که در ابتدای این بخش توضیح دادیم باید به لزوم استفاده از عبارت where در دستور update دقت کنیم. برای مثال فرض کنید در نمونه قبل از WHERE استفاده نمی کردیم:

```
UPDATE Persons
SET Address='Nissestien 67', City='Sandnes'
```

در عبارت بالا از where استفاده نکردیم ، به نتیجه آن توجه کنید:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Nissestien 67	Sandnes
2	Svendson	Tove	Nissestien 67	Sandnes
3	Pettersen	Kari	Nissestien 67	Sandnes
4	Nilsen	Johan	Nissestien 67	Sandnes
5	Tjessem	Jakob	Nissestien 67	Sandnes

از این عبارت برای پاک کردن اطلاعات از جدول استفاده میشود.

```
DELETE FROM table_name
WHERE some_column=some_value
```

دستور بالا نحوه استفاده از عبارت DELETE را نمایش میدهد.

نکته : به عبارت where در دستور DELETE دقت کنید . اگر شما از این عبارت استفاده نکنید تمام رکورد های جدول شما DELETE میشود!

مثالی از دستور DELETE در SQL

جدول PERSON را در نظر بگیرید:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

حال میخواهیم با استفاده از دستور زیر اطلاعات شخص TJESSEM را از جدول پاک کنیم:

```
DELETE FROM Persons
WHERE LastName='Tjessem' AND FirstName='Jakob'
```

نتیجه به شکل جدول زیر خواهد بود:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger

پاک کردن کل اطلاعات جدول

میتوان تمام اطاعات موجود در جدول را پاک نمود به نحوی که جدول ما پا برجا بماند این بدین معناست که شما جدول خود را در پایگاه داده ها حفظ میکنید اما اطلاعات آن را به طور کل حذف میکنید. البته توجه کنید که مراقب استفاده از دستور زیر باشید چرا که اطلاعات پاک شده قابل برگشت نخواهد بود.
توجه کنید که برای اینکار کافیست از عبارت WHERE استفاده نکنیم!

```
DELETE FROM table_name

or

DELETE * FROM table_name
```

عبارت TOP

این عبارت برای مشخص کردن تعداد رکورد هایی که لازم است اطلاعات آنها باز گردانده شود مورد استفاده قرار میگیرد.
عبارت TOP برای جداول بزرگ و با هزاران اطلاعات ذخیره شده در آنها مورد استفاده قرار میگیرد و شمار زیادی از رکوردهای اطلاعاتی را میتواند نمایش دهد .
نکته:عبارت TOP توسط همه سیستم های پایگاه داده ای پشتیبانی نمیشود.

SQL Server Syntax

```
SELECT TOP number|percent column_name(s)
FROM table_name
```

یا معادل عبارت SELECT TOP در SQL را در MYSQL و ORACLE

MySQL Syntax

```
SELECT column_name(s)
FROM table_name
LIMIT number
```

مثال:

```
SELECT *
FROM Persons
LIMIT 5
```

Oracle Syntax

```
SELECT column_name(s)
FROM table_name
WHERE ROWNUM <= number
```

مثال:

```
SELECT *
FROM Persons
WHERE ROWNUM <=5
```

مثال های از عبارت TOP در SQL:

نام جدول PERSON

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Tom	Vingvn 23	Stavanger

حالا ما فقط دو رکورد اطلاعاتی اول جدول بالا را لازم داریم برای جدا کردن آنها از روش زیر بهره میبریم:

```
SELECT TOP 2 * FROM Persons
```

و نتیجه به فرم زیر میباشد:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

مثال برای عبارت TOP PERECENT در SQL:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

4	Nilsen	Tom	Vingvn 23	Stavanger

حالا میخواهیم ۵۰% اطلاعات بالا را از جدول بیرون بکشیم، به همین دلیل از عبارت TOP به فرم زیر بهره میگیریم.

```
SELECT TOP 50 PERCENT * FROM Persons
```

نتیجه به شکل زیر ظاهر میشود:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

عملگر LIKE:

این عملگر برای جستجوی یک الگوی مشخص در ستون جدول پایگاه داده ما مورد استفاده قرار میگیرد

دستور LIKE:

```
SELECT column_name(s)
FROM table_name
WHERE column_name LIKE pattern
```

مثالی برای عملگر LIKE:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

حالا میخواهیم از جدول بالا مشخصات فردی را بیرون بکشیم که نام شهرمحل زندگی اش با حرف 's' آغاز میشود.

به شکل زیر عمل میکنیم:

ITMYT

```
SELECT * FROM Persons
WHERE City LIKE 's%'
```

علامت % همانند یک WILDCARD(یعنی کاراکتر گمشده در الگوی مورد نظر) مورد استفاده قرار میگیرد،این کاراکتر میتواند قبل یا بعد از الگوی مورد نظر باشد.
راهنمایی : کسانی که با جستجو در WINDOWS آشنا باشند میدانند که برای جستجوی یک فایل گمشده با پسوند TEXT عبارت زیر را برای جستجو مینویسند:
TEXT.*
خب توجه کنید به نتجه مثال بالا:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

و حالا میخواهیم این بار مشخصات فردی را از جدول "PERSON"پیدا کنیم که نام شهرمحل زندگی اش با حرف "S"پایان یابد.
برای اینکار به نحو زیر عمل میکنیم :

```
SELECT * FROM Persons
WHERE City LIKE '%s'
```

و این بار نتیجه مانند جدول زیر می شود :

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

بعد از آن میخواهیم از جدول "PERSON" مشخصات فردی را بیابیم که نام شهر محل زندگی او دارای عبارت "TAV" باشد.

به دستور زیر دقت کنید:

```
SELECT * FROM Persons
WHERE City LIKE '%tav%'
```

و نتیجه نیز به فرم زیر است:

P_Id	LastName	FirstName	Address	City
3	Pettersen	Kari	Storgt 20	Stavanger

همچنین این قابلیت نیز وجود دارد که ما مشخصات فردی را بیابیم که شهر محل زندگی اش دارای عبارت "TAV" نباشد. این عمل با به کار گیری کلمه کلیدی NOT در ساختار عملگر LIKE ممکن میشود.

```
SELECT * FROM Persons
WHERE City NOT LIKE '%tav%'
```

که نتیجه آن، چنین میباشد:

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

Wildcards (کاراکترهايي که نشان دهنده مجموعه اي از کاراکترها مي باشند مثلا *.txt) : زمانی که شما در پایگاه داده خود به دنبال داده ای هستید این عبارت جانشین یک یا چند کاراکترگمشده میشود.

عبارت Wildcards باید حتما با عملگر LIKE در SQL به کار رود

با SQL میتوان از Wildcard های زیر استفاده کرد:

Wildcard	Description
%	جانشین هیچ یا چند کاراکتر میشود.
_	دقیقا جانشین یک کاراکتر میشود .
[charlist]	جانشین یک تک کاراکتر در آرایه.
[^charlist] or [[charlist]	هر تک کاراکتر که در آرایه نباشد.

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

مثال :

نام جدولمان "PERSON" میباشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

استفاده از "%" برای جانشینی :

میخواهیم مشخصات فردی را از جدول پیدا کنیم که نام شهر محل زندگی اش با عبارت " SA " آغا ز شود.
از عبارت SELECT به صورت زیر استفاده میکنیم .

```
SELECT * FROM Persons
WHERE City LIKE 'sa%'
```

نتیجه به فرم زیر است :

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

حال میخواهیم از جدول "person" مشخصات فردی را بیابیم که نام شهر محل زندگی اش از دارای عبارت " nes" باشد.
به شکل زیر عمل میکنیم :

```
SELECT * FROM Persons
WHERE City LIKE '%nes%'
```

و نتیجه آن به فرم زیر است :

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

استفاده از "_" برای جانشینی :

میخواهیم مشخصات فردی را از جدول "person" انتخاب کنیم که نام آن شخص با هرحرفی (نامشخص) آغاز شده باشد و در ادامه آن عبارت "la" وجود داشته باشد.
از عبارت select به شکل زیر بهر میگیرم:

```
SELECT * FROM Persons
WHERE FirstName LIKE '_la'
```

نتیجه نیز به شکل زیر می باشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes

و بعد به دنبال شخصی در جدول "person " میباشیم که نام خانوادگی او با "s" آغاز شده و بعد از چند کاراکتر نامشخص به عبارت "end" رسیده و بازهم بعد از چند کاراکتر نا معلوم با عبارت "on" پایان یابد.

به دستور زیر دقت کنید:

```
SELECT * FROM Persons
WHERE LastName LIKE 'S_end_on'
```

نتیجه استفاده از این دستور به شکل زیر می باشد.

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes

استفاده از [charlist] برای جانشینی :

اینک به دنبال مشخصات فردی هستیم که نام او در جدول "person" با یکی از حروف "s" یا "b" یا "p" آغاز شود
برای رسیدن به نتیجه مطلوب از دستور زیر استفاده میکنیم :

```
SELECT * FROM Persons
WHERE LastName LIKE '[sbp]%'
```

نتیجه :

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

و حالا برخلاف بالا میخواهیم مشخصات فردی را که نامش با هیچ یک از حروف "s" یا "b" یا "p" آغاز نشود را از جدول بیابیم.
برای اینکار از دستور زیر استفاده میکنیم .

```
SELECT * FROM Persons
WHERE LastName LIKE '[ !sbp ]%'
```

که نتیجه مطلوب زیر را میدهد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes

این عملگر به شما اجازه میدهد که برای استفاده از عبارت WHERE از چند "VALUE" یا ارزش موجود در جدول بهره بگیرید

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1,value2,...)
```

مثلا:

نام جدول "PERSON" میباشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

حالا از جدول بالا مشخصات فردی را می یابیم که نام خانوادگی اش "Eskandari" یا "Pettersen" باشد

و برای این کار ازدستور زیر بهره میگیریم.

```
SELECT * FROM Persons
WHERE LastName IN ('Eskandari','Pettersen')
```

و نتیجه نیز به شکل زیر می باشد.

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes

ITMYT

3	Pettersen	Kari	Storgt 20	Stavanger

این عملگر برای انتخاب مجموعه مقادیری بین دو مقدار معلوم به کار می رود دو مقدار که قابل شمارش باشد یا نوشته شود ویا قابل تاریخ گذاردن باشد(دو مقدار مورد آزمایش). ضمنا این عملگر در عبارت WHERE که قبلا راجع با آن توضیح دادیم به کار می رود.

دستور BETWEEN

```
SELECT column_name(s)
FROM table_name
WHERE column_name
BETWEEN value1 AND value2
```

مثالی از عملگر BETWEEN
جدول "PERSON"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

حالا ما میخواهیم از جدول بالا افرادی را با نام خانوادگی بین "Eskandari" و "Pettersen" انتخاب کنیم .
از عبارت دستوری زیر استفاده میکنیم.

```
SELECT * FROM Persons
WHERE LastName
BETWEEN 'Eskandari' AND 'Pettersen'
```

نتیجه همانند جدول زیر خواهد بود

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes

نکته :

عملگر BETWEEN رفتار مختلفی بین پایگاه داده های مختلف دارد .
در برخی پایگاه داده ها افراد با نام خانوادگی 'Eskandari'و'Pettersen' وارد جدول مورد نظرنخواهند شد زیرا عملگر BETWEEN فقط مقادیر بین دو نام بالا را در نظر میگیرد و دو مقدار اولیه و ابتدایی یعنی دو اسم مورد آزمایش را محاسبه نمی کند.

در برخی دیگر افراد با نام خانوادگی 'Eskandari'و'Pettersen' وارد جدول مورد نظرخواهند شد زیرا عملگر BETWEENعلاوه بر مقادیر میانی دو نام مورد آزمایش را در نظر گرفته و وارد جدول نهایی می کند.
و در دیگر پایگاه داده ها افراد با نام خانوادگی 'Eskandari' یعنی اسم اول مورد آزمایش وارد جدول شده و اسم دوم مورد آزمایش یعنی 'Pettersen' مانند مثال بالا وارد جدول نخواهد شد. زیرا عملگر BETWEEN فقط مقادیر میانی به علاوه مقدار اولیه مورد آزمایش را وارد جدول می کند.
بنا بر این :
هنگام استفاده از عملگر BETWEEN آن را آزمایش کنید و نحوه عملکرد آنرا بررسی کنید
مثال دوم :
برای نمایش افراد خارج از مقادیر مثال قبل یعنی دقیقا عکس مثال قبل از NOT BETWEEN استفاده میشود.

```
SELECT * FROM Persons
WHERE LastName
NOT BETWEEN 'Eskandari' AND 'Pettersen'
```

که نتیجه به شکل زیر خواهد بود

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

با استفاده از این قابلیت در SQL میتوانیم به جداول و یا برخی ستون ها در جداول نام دیگری اختصاص دهیم و این زمانی مفید می باشد که نام این جداول یا ستون ها طولانی و یا پیچیده باشد ،که موجب میشود ما راحتتر از این نامها استفاده کنیم.(ALIAS: در لغت به معنی نام مستعار یا هم اثر است)
از هر نامی به عنوان ALIAS میتوان استفاده کرد اما معمولا این نام ها کوتاه می باشد.
دستور ALIAS برای جداول

```
SELECT column_name(s)
FROM table_name
AS alias_name
```

دستور ALIAS برای ستون ها

```
SELECT column_name AS alias_name
FROM table_name
```

مثال:

فرض کنید ما جدولی داریم به نام "PERSON" و جدول دیگری با نام "Product_Orders" داریم
ما با استفاده از دستور ALIAS در جداول به ترتیب نام "p" و "po" را به این جداول میدهیم .
حالا ما میخواهیم لیستی از تمام سفا رشاتی که "Mojtaba Eskandari" مسئول آنهاست را مشخص کنیم
ما از دستور زیر استفاده می کنیم.

```
SELECT po.OrderID, p.LastName, p.FirstName
FROM Persons AS p,
Product_Orders AS po
WHERE p.LastName='Eskandari'
WHERE p.FirstName='Mojtaba'
```

استفاده از SELECT بدون ALIAS :

```
SELECT Product_Orders.OrderID, Persons.LastName, Persons.FirstName
FROM Persons,
Product_Orders
WHERE Persons.LastName='Eskandari'
WHERE Persons.FirstName='Mojtaba'
```

همانطور که مشاهده می کنید استفاده از ALIAS موجب کوتاهتر شدن دستورSELECT شده است و این خواندن و نوشتن دستور را ساده تر نموده است .

این کلمه کلیدی در یک عبارات SQL برای جستجو اطلاعات از ۲ یا تعداد بیشتری جدول بر اساس ارتباط بین دو ستون مشخص در آن جدول ها میباشد.
جدول ها در پایگاه داده معمولا با هم توسط کلیدهایی در ارتباط هستند
کلیدهای اولیه ستون یا ترکیبی از ستون ها هستند با یک مشخصه منحصر به فرد برای هر ردیف ، هر کلید اولیه باید در جدول منحصر به فرد باشد. هدف ترکیب داده ها یا اطلاعات از میان جداول بدون اینکه کل اطلاعات در هر جدول مورد بررسی قرار گیرد ، میباشد .
به جدول "person" نگاه کنید

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

توجه داشته باشید که" P_Id " یک کلید اولیه برای جدول بالا محسوب میشود . به این معنا که هیچ یک از دو ستون جدول "PERSON " نمیتوانند "P_Id" یکسان داشته باشند .
کلید اولیه این جدول "P_Id" حتی افراد با نام های یکسان را هم از هم جدا میکند.
حالا جدول بعدی "ORDER " نام دارد

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

توجه داشته باشید که" O_Id " یک کلید اولیه برای جدول بالا محسوب میشود. "P_Id" در این جدول اشاره به افراد در جدول "PERSON" بدون ذکر نام آنها .
نکته اینکه ارتباط بین این دو جدول بالا ستون "P_Id" میباشد.

Different SQL JOINs

قبل ازینکه کارمون را با مثال ادامه بدیم انواع JOIN را که شما میتوانید استفاده کنید ذکر کرده و تفاوت آنها را بیان میکنیم

JOIN : ردیف هایی را نشان میدهد که حداقل یک همخوانی در دو جدول داشته باشند

LEFT JOIN : تمام ردیف های جدول سمت چپ را نمایش میدهد حتی اگر هیچ همخوانی با جدول سمت راست نداشته باشد.

RIGHT JOIN : تمام ردیف های جدول سمت راست را نمایش میدهد حتی اگر هیچ همخوانی با جدول سمت چپ نداشته باشد.

FULL JOIN : ردیف هایی را که یک همخوانی در دو جدول را دشته باشد نمایش میدهد .

کلمه کلیدی INNER JOIN ردیف هایی را بر می گرداند که حداقل یک همخوانی بین دو جدول وجود داشته باشد.

```
SELECT column_name(s)
FROM table_name1
INNER JOIN table_name2
ON table_name1.column_name=table_name2.column_name
```

INNER JOIN همانند JOIN عمل میکند.

مثال :

جدول"PERSONS"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

جدول "ORDER" :

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

حالا ما میخواهیم لیستی درست کنیم که این دو جدول را با هم مرتبط کند با توجه بهP_Id آنها
به صورت زیر عمل میکنیم.

```
SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
INNER JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName
```

و نتیجه به شکل مقابل میباشد

LastName	FirstName	OrderNo
Eskandari	Mojtaba	22456
Eskandari	Mojtaba	24562
Pettersen	Kari	77895
Pettersen	Kari	44678

این دستور ردیف های خوانده شده را از جدول شماره ۱ یعنی جدول سمت چپ برمیگرداند حتی اگر هیچ همخوانی با جدول شماره ۲ یعنی جدول سمت راست نداشته باشد.
به دستور زیر دقت کنید

```
SELECT column_name(s)
FROM table_name1
LEFT JOIN table_name2
ON table_name1.column_name=table_name2.column_name
```

این دستور در برخی از پایگاه داده ها به صورت LEFT OUTER JOIN استفاده میشود.
مثال : "PERSON"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

جدول "ORDER"

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

حالا ما دو جدول بالا را به هم مرتبط میکنیم و با توجه به ترتیبشان در جدول دیگر مینویسیم.
از عبارت زیر بهره میگیریم:

```
SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
LEFT JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName
```

و نتیجه:

LastName	FirstName	OrderNo
Eskandari	Mojtaba	22456
Eskandari	Mojtaba	24562
Pettersen	Kari	77895
Pettersen	Kari	44678
Svendson	Tove	

این دستور ردیف های خوانده شده را از جدول شماره ۲ یعنی جدول سمت راست برمیگرداند حتی اگر هیچ همخوانی با جدول شماره ۱ یعنی جدول سمت چپ نداشته باشد.
به دستور زیر دقت کنید

```
SELECT column_name(s)
FROM table_name1
RIGHT JOIN table_name2
ON table_name1.column_name=table_name2.column_name
```

این دستور در برخی از پایگاه داده ها به صورت RIGHT OUTER JOIN استفاده میشود.
مثال : جدول" PERSON"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

جدول"ORDER"

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

حالا ما دو جدول بالا را به هم مرتبط میکنیم و با توجه به ترتیبشان در جدول دیگر مینویسیم.
از عبارت زیر بهره میگیریم:

```
SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
RIGHT JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName
```

و نتیجه:

LastName	FirstName	OrderNo
Eskandari	Mojtaba	22456
Eskandari	Mojtaba	24562
Pettersen	Kari	77895
Pettersen	Kari	44678
		34764

عبارت کلیدی FULL JOIN تمام ردیف های موجود در جداول را با وجود حتی یک همخوانی میان جداول بر میگرداند .

دستور FULL JOIN در SQL

```
SELECT column_name(s)
FROM table_name1
FULL JOIN table_name2
ON table_name1.column_name=table_name2.column_name
```

مثال از FULL JOIN:

جدول "PERSON":

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

جدول "ORDER":

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

حالا میخواهیم که از جداول بالا افراد را با ترتیبشان و تمام ترتیب ها از جدول ORDER را با افراد متناسب خودبه صورت یک فهرست در آوریم.

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

از عبارت زیر استفاده میکنیم

```
SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
FULL JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName
```

نتیجه به شکل زیر میباشد

LastName	FirstName	OrderNo
Eskandari	Mojtaba	22456
Eskandari	Mojtaba	24562
Pettersen	Kari	77895
Pettersen	Kari	44678
Svendson	Tove	
		34764

FUULL JOIN تمام ردیف ها از جدول PERSON و همچنین تمام ردیف ها از جدول ORDER را برمیگرداند حتی اگر هیچ همخوانی بین آنها موجود نباشد در هر صورت اطلاعات بین دو جدول به صورت فهرست در خواهد آمد

عملگر UNION در SQL دو یا چند عبارت SLELECT را با هم ترکیب میکند.
نکته اینکه هر عبارت SELECT داخل عملگر UNION باید حتما دارای تعداد یکسان ستون وستون ها نیز دارای اطلاعات یکسان و تعداد داده ها در هر ستون باید یکسان باشد
دستور UNION :

```
SELECT column_name(s) FROM table_name1
UNION
SELECT column_name(s) FROM table_name2
```

مثال از UNION :
جدول "Employees_Norway":

E_ID	E_Name
01	Eskandari, Mojtaba
02	Svendson, Tove
03	Svendson, Stephen
04	Pettersen, Kari

جدول "Employees_USA":

E_ID	E_Name
01	Turner, Sally
02	Kent, Clark

03	Svendson, Stephen
04	Scott, Stephen

حالا ما میخواهیم اسامی تمام کارمندان در نوروژ و آمریکا را به صورت لیست در آوریم.
از عبارت SELECT به صورت زیر استفاده میکنیم.

```
SELECT E_Name FROM Employees_Norway
UNION
SELECT E_Name FROM Employees_USA
```

و نتیجه مانند جدول زیر میشود.

E_Name
Eskandari, Mojtaba
Svendson, Tove
Svendson, Stephen
Pettersen, Kari
Turner, Sally
Kent, Clark
Scott, Stephen

نکته :توجه کنید که این دستور نمیتواند تمام کارمندان را درنوروژ و آمریکا به صورت لیست در آورد همانطور که مشاهده میکنید ما در دو جدول بالا دو کارمند با نام های یکسان داریم اما هنگام لیست شدن فقط نام یکی از آنها نوشته شد این بدان معناست که عملگر UNION به صورت(DISTINCT) عمل میکند .(برای درک بهتر به قسمت DISTINCTدر بخش SQL مراجعه کنید)

حال برای رفع این مشکل باید از دستور زیر استفاده کنیم

```
SELECT E_Name FROM Employees_Norway
UNION ALL
SELECT E_Name FROM Employees_USA
```

در این صورت ما لیست افراد حتی با نام های یکسان را دارا میباشیم

E_Name
Eskandari, Mojtaba
Svendson, Tove
Svendson, Stephen
Pettersen, Kari
Turner, Sally
Kent, Clark
Svendson, Stephen
Scott, Stephen

عبارت SELECT INTO در SQL برای انتخاب اطلاعات یک جدول و وارد کردن این اطلاعات در جدول دیگر میباشد.
این عبارت برای داشتن یک کپی از اطلاعات به عنوان پشتیبان میباشد
دستور SELECT INTO :

```
SELECT *
INTO new_table_name [IN externaldatabase]
FROM old_tablename
```

این عبارت تمامی اطلاعات را وارد جدول جدید میکند.

شاید لازم باشد فقط ستونی از اطلاعات را که میخواهیم وارد جدول جدید کنیم:

```
SELECT column_name(s)
INTO new_table_name [IN externaldatabase]
FROM old_tablename
```

مثال از SELECT INTO :
حالا ما میخواهیم یک کپی کامل از جدول PERSON را داشته باشیم .
از دستور زیر استفاده میکنیم :

```
SELECT *
INTO Persons_Backup
FROM Persons
```

همچنین با استفاده از عبارت IN میتوانیم اطلاعات را در پایگاه داده دیگری کپی میکنیم.

```
SELECT *
INTO Persons_Backup IN 'Backup.mdb'
FROM Persons
```

همچنین ما میتوانیم مقدار کمی از اطلاعات که میخواهیم را کپی کنیم.

```
SELECT LastName,FirstName
INTO Persons_Backup
FROM Persons
```

SELECT INTO با استفاده از عبارت WHERE:

با دستور زیر ما یک کپی از جدول PERSON آن هم فقط از افرادی که در شهر "Sandnes" زندگی میکنند را بدست می آوریم

```
SELECT LastName,Firstname
INTO Persons_Backup
FROM Persons
WHERE City='Sandnes'
```

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

SELECT INTO همراه با JOINED:

ما میتوانیم از دو جدول یا بیشتر هم کپی برداری کنیم
عبارت زیر یک جدول پشتیبان با نام "Persons_Order_Backup" میسازد و سپس اطلاعات را از جدول "Persons" و "Orders" داخل آن وارد میکند

```
SELECT Persons.LastName,Orders.OrderNo
INTO Persons_Order_Backup
FROM Persons
INNER JOIN Orders
ON Persons.P_Id=Orders.P_Id
```

ساختن پایگاه داده CREATE DATABASE

برای ساختن یک پایگاه داده از این عبارت استفاده میکنیم
دستور آن به صورت زیر است:

```
CREATE DATABASE database_name
```

مثال از CREATE DATABASE:

حالا م میخواهیم که پایگاه داده ای با نام "my_db" بسازیم.
از دستور بالا به این صورت استفاده میکنیم

```
CREATE DATABASE my_db
```

عبارت CREATE TABLE برای ساختن یک جدول در پایگاه داده مان استفاده میشود.
دستور CREATE TABLE:

```
CREATE TABLE table_name
(
column_name1 data_type,
column_name2 data_type,
column_name3 data_type,
....
)
```

در جدول ما انواع داده ها میتواند ذخیره شود.
مثال برای CREATE TABLE:
حالا ما میخواهیم جدولی به نام "Persons" بسازیم که دارای ۵ ستون با نامهای P_Id, LastName, FirstName, Address, and City میباشد.
از دستور زیر استفاده میکنیم :

```
CREATE TABLE Persons
(
```

```
P_Id int,
LastName varchar(255),
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

ستونP_Id حاوی داده با نوع INT میباشد و ستون های دیگر حاوی داده ها با نوع varchar میباشند که ماکزیمم طول آنها ۲۵۵ کاراکتر میباشد.

P_Id	LastName	FirstName	Address	City

این جدول با دستور SELECT INTO پر خواهد شد .

Constraints برای محدود ساختن آن داده هایی که متوانند در جدول ذخیره شوند استفاده میشود

میتواند زمانی که یک جدول ساخته میشود و یا بعد از ساخته شدن آن (همراه دستور ALTER TABLE که بعدا توضیح خواهیم داد) مشخص شود.
ما بر روی CONSTRAINTS های زیر تمرکز میکنیم .

* NOT NULL
* UNIQUE
* PRIMARY KEY
* FOREIGN KEY
* CHECK
* DEFAULT

عبارت not null موجب میشود تا ستون مورد نظر نتواند مقدار null را بپذیرد

not null باعث میشود تا فیلد مورد نظر همیشه دارای مقدار بوده به این معنا که شما نمیتوانید یک record جدید بدون value یا یک record موجود را بدون مقدار(value) , به روز کنید(update).

عبارت SQL زیر موجب میشود تا ستون های "P_Id" و "LastName" مقدار null را نپذیرند(یعنی نمیتوان این دو فیلد را خالی بگذاریم).

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
```

```
City varchar(255)
)
```

unique در دستورات SQL باعث میشود که فیلد مورد نظر در database یکتا شناخته شود.

primery key و unique در واقع تضمین یکتا بودن فیلد مورد نظر در جدول ما میشود.

در یک primery key به طور خودکار دستور unique وجود دارد.

نکته:

در یک جدول پایگاه داده ای شما میتوانید چندین unique را داشته باشید اما فقط یک primery key میتوانید داشته باشید.

دستور unique در جدول زیر باعث یکتا شدن مقدار "P_Id" میشود.

:MySQL

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
UNIQUE (P_Id)
)
```

:SQL Server / Oracle / MS Access

```
CREATE TABLE Persons
(
P_Id int NOT NULL UNIQUE,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

برای نام گذاری یک مجدودیت UNIQUE و یا برای معین کردن UNIQUE های موجود در یک جودل می توان از دستور زیر استفاده نمود.

:MySQL / SQL Server / Oracle / MS Access

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
```

```
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT uc_PersonID UNIQUE (P_Id,LastName)
)
```

ترکیب دستور های ALTER TABLE و UNIQUE یعنی UNIQE کردن یک مقدار برای جدول ساخته شده.

:MySQL / SQL Server / Oracle / MS Access

```
ALTER TABLE Persons
ADD UNIQUE (P_Id)
```

UNIQE کردن فیلد DROP :

MySQL:

```
ALTER TABLE Persons
DROP INDEX uc_PersonID
```

SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
DROP CONSTRAINT uc_PersonID
```

این محدودیت یک مشخصه یکتایی است که برای هر داده در جدول پایگاه داده تعریف میشود

primary key حتما باید مقدارش یکتا باشد

مقدار null را نمیتواند بپذیرد

هر جدول باید یک primary key داشته باشد و البته فقط یک primary key هم میتواند داشته باشد.

ایجاد محدودیت primary key در هنگام تولید جدول

MySQL:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
PRIMARY KEY (P_Id)
)
```

SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons
(
P_Id int NOT NULL PRIMARY KEY,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

و اما برای اینکه یک محدودیت primary key را نامگذاری کنیم و اینکه برای یک جدول با چندین ستون یک primary key تعریف کنیم از دستور زیر استفاده می کنیم خوب به این دستور دقت کنید

MySQL / SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT pk_PersonID PRIMARY KEY (P_Id,LastName)
)
```

ایجاد primary key بعد از ساخت یک جدول

برای انجام عمل فوق از دستور زیر استفاده میکنیم و حتما دستور alter را بخاطر دارید ، دقت کنید

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ADD PRIMARY KEY (P_Id)
```

و همچنین این عمل برای یک جدول با داده های بسیار از دستور زیر بهره میگیریم

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ADD CONSTRAINT pk_PersonID PRIMARY KEY (P_Id,LastName)
```

نکته:

دقت کنید زمانی که شما بعد از ساخت یک جدول اقدام به تولید primary key میکنید باید حتما انرا not null تعریف کنید

حذف یک primary key

برای این کار از دستور زیر استفاده میکنیم

Www.ITMYT.Blogfa.Com
دانلود جزوه و سوالات درسی

MySQL

```
ALTER TABLE Persons
DROP PRIMARY KEY
```

SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
DROP CONSTRAINT pk_PersonID
```

محدودیت foreign key در یک جدول به کلید اصلی یک جدول دیگر یا همان primary key در یک جدول دیگر اشاره دارد .

بیاید برای درک بهتر به مثال زیر دقت کنیم:

جدول "person"

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

جدول "order"

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	2
4	24562	1

دقت کنید که عنصر P_Id درجدول order به عنصر همنام خود یعنی P_Id در جدول person اشره دارد و حاوی همان اطلاعات میباشد P_Id در order یک foreign key و در person یک key primary| میباشد.

foreign key در واقع از وارد شدن اطلاعات نا معتبر در جدول و همچنین هرچیز که ارتباط بین دو جدول را تخریب کند جلوگیری میکند زیرا که به مقدار یکتایی در جدول اصلی اشاره دارد.

ایجاد foreign key هنگام ساخت جدول

برای انجام عمل فوق از دستور زیر استفاده میکنیم

MySQL:

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
PRIMARY KEY (O_Id),
FOREIGN KEY (P_Id) REFERENCES Persons(P_Id)
)
```

SQL Server / Oracle / MS Access:

```
CREATE TABLE Orders
(
O_Id int NOT NULL PRIMARY KEY,
OrderNo int NOT NULL,
P_Id int FOREIGN KEY REFERENCES Persons(P_Id)
)
```

و برای محدود کردن چند ستون در جدول به عنوان foreign key از دستور زیر استفاده کنید

MySQL / SQL Server / Oracle / MS Access:

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
PRIMARY KEY (O_Id),
CONSTRAINT fk_PerOrders FOREIGN KEY (P_Id)
REFERENCES Persons(P_Id)
)
```

برای ساخت یک foreign key در جدولی که قبلا ساخته شده است باید از دستور زیر استفاده کرد.

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Orders
ADD FOREIGN KEY (P_Id)
REFERENCES Persons(P_Id)
```

و همچنین برای ساخت چند ستون به عنوان یک foreign key از دستور زیر استفاده کنید

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Orders
ADD CONSTRAINT fk_PerOrders
FOREIGN KEY (P_Id)
REFERENCES Persons(P_Id)
```

و برای حذف یک foreign key از دستورات زیر بهره بگیرید

MySQL:

```
ALTER TABLE Orders
DROP FOREIGN KEY fk_PerOrders
```

SQL Server / Oracle / MS Access:

```
ALTER TABLE Orders
DROP CONSTRAINT fk_PerOrders
```

محدودیت check تعیین میکند که چه بازه ای از مقادیر و یا داده با چه مشخصاتی میتواند در ستون پایگاه داده قرار بگیرد.

همچنین میتوان برای جدول با توجه به ستونهایش محدودیت check ایجاد کنیم

میخواهیم محدودیت check را هنگام ساخت جدول اعمال کنیم ، به دستورات زیر دقت کنید ، پس از ساخت جدول شما فقط میتوانید مقدار integer را برای P_Id وارد کنید

My SQL:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CHECK (P_Id>0)
)
```

SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons
(
P_Id int NOT NULL CHECK (P_Id>0),
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

و برای محدود کردن چند ستون به طور همزمان هم از دستور زیر استفاده میکنیم

MySQL / SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT chk_Person CHECK (P_Id>0 AND City='Sandnes')
)
```

برای ایجاد دستور check در جدولی که قبلا ساخته شده است از دستور زیر بهره میگیریم

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ADD CHECK (P_Id>0)
```

و همچنین این عمل برای محدودیت چندین ستون به صورت زیر اعمال میشود

MySQL / SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ADD CONSTRAINT chk_Person CHECK (P_Id>0 AND City='Sandnes')
```

برای حذف یک محدودیت check نیز به این صورت عمل میکنیم

SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
DROP CONSTRAINT chk_Person
```

این محدودیت موجب میشود که یک مقدار به صورت پیشفرض در ستون مورد نظر قرار بگیرد و اگر برای آن ستون مقدار دیگری وارد نشود همان مقدار به صورت پیشفرض وارد جدول گردد

دستور ساخت default هنگام ساخت یک جدول به صورت زیر میباشد.

My SQL / SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255) DEFAULT 'Sandnes'
)
```

همچنین این مقدار پیشفرض میتواند توسط توابع sql در نظر گرفته شود.

مثال

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
OrderDate date DEFAULT GETDATE()
)
```

برای ایجاد default برای یک عنصر در جدولی که قبلا ساخته شده است با کمک alter از دستور زیر بهره میگیریم

MySQL:

```
ALTER TABLE Persons
ALTER City SET DEFAULT 'SANDNES'
```

SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ALTER COLUMN City SET DEFAULT 'SANDNES'
```

و در اخر برای حذف یک default از جدول به صورت زیر عمل میکنیم

MySQL:

```
ALTER TABLE Persons
ALTER City DROP DEFAULT
```

SQL Server / Oracle / MS Access:

```
ALTER TABLE Persons
ALTER COLUMN City DROP DEFAULT
```

دستور index این امکلن را میدهد تا پایگاه داده راحت و سریعتر به داده ها دسترسی پیدا کند و نیازی به خواندن تمام اطلاعات برای پیدا کردن داده ای خاص نمیباشد

کاربر index را نمیبیند و فقط سرعت عملکرد را متوجه میشود

نکته:

update شدن یک جدول با index زمان بیشتری نسبت به update شدن یک جدول بدون index میباشد زیرا index ها نیز باید update شوند

دستور ساخت index

```
CREATE INDEX index_name
ON table_name (column_name)
```

اما در دستور بالا تکرار برای عنصری که index است مجاز است و برای یکتا ساختن index از دستور زیر استفاده میکنیم.

```
CREATE UNIQUE INDEX index_name
ON table_name (column_name)
```

دستور index در پایگاه داده های متفاوت ، مختلف است پس برای ساخت آن دقت کنید.

به مثال های زیر دقت کنید

```
CREATE INDEX PIndex
ON Persons (LastName)
```

این دستور PIndex را به عنوان index برای جدول person در ستون LastName ایجاد میکند.

برای hndex ساختن چندین ستون هم از فرم زیر کمک بگیرید

```
CREATE INDEX PIndex
ON Persons (LastName, FirstName)
```

index ها ، جداول(table ها) و پایگاه های داده (databases) به راحتی با دستور drop قابل حذف هستند.

drop index در MS Access

```
DROP INDEX index_name ON table_name
```

drop index در MS sql server

```
DROP INDEX table_name.index_name
```

drop index برای DB2/oracle

```
DROP INDEX index_name
```

drop index برای MySQL

```
ALTER TABLE table_name DROP INDEX index_name
```

دستور drop table که برای حذف جدول از آن استفاده میشود

```
DROP TABLE table_name
```

drop database نیز دستوری است برای حذف یک پایگاه داده که از آن به شکل زیر استفاده میشود

```
DROP DATABASE database_name
```

و اما اگر بخواهیم که فقط اطلاعات جدول را پاک کنیم و خود جدول باقی بماند از دستور truncate استفاده میکنیم.

```
TRUNCATE TABLE table_name
```

عبارت ALTER TABLE در sql برای تغییر و حذف کردن یا اضافه کردن ستون ها در جداول استفاده میشود.

دستور alter table در sql

```
ALTER TABLE table_name
ADD column_name datatype
```

این دستور برای اضافه کردن ستونی جدید به جدول استفاده میشود.

```
ALTER TABLE table_name
DROP COLUMN column_name
```

و این دستور برای حذف کردند یک ستون استفاده میشود که البته بخاطر داشته باشید برخی از پایگاه های داده این اجازه را به شما نمیدهند.

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype
```

و در آخر دستور بالا برای تغییر در داده های یک ستون از جدول استفاده میشود.

مثال:

جدول person را در نظر بگیرید

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

حالا میخواهیم به جدول بالا یک ستون با نام DateOfBirth اضافه کنیم

از دستور زیر بهره میگیریم

```
ALTER TABLE Persons
ADD DateOfBirth date
```

توجه داشته باشید که این ستون جدید مقدار تاریخ را نگه میدارد یعنی در واقع میخواهم به این نکته اشاره کنم که dataType نوع داده ای را مشخص میکند که در آن ستون نگهداری میشود.

جدول بالا به شکل زیر میشود.

P_Id	LastName	FirstName	Address	City	DateOfBirth
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes	
2	Svendson	Tove	Borgvn 23	Sandnes	
3	Pettersen	Kari	Storgt 20	Stavanger	

تغییر در dataType را با استفاده از دستور زیر انجام میدهیم

```
ALTER TABLE Persons
ALTER COLUMN DateOfBirth year
```

حال این ستون از جدول یعنی DateOfBirth مقدار سال را در دو رقم یا رقم در خود نگهداری میکند.

و در آخر میخواهیم که این ستون از جدول را حذف کنیم .

```
ALTER TABLE Persons
DROP COLUMN DateOfBirth
```

با اجرای دستور بالا ستون DateOfBirth از جدول حذف خواهد شد.

جدول به شکل اول خود بر میگردد

P_Id	LastName	FirstName	Address	City
1	Eskandari	Mojtaba	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

ITMYT