



دانشگاه صنعتی اصفهان
دانشکده برق و کامپیوتر

دستور کار آزمایشگاه پایگاه داده‌ها ترم اول ۹۹-۹۸

استاد درس
دکتر علیرضا بصیری

ویرایش ششم تابستان ۱۳۹۸

تهیه کنندگان:
دانیال اکبری
مهران صادقی
سید میثم غفاری

بازنگری و بازنویسی:
محمد سلیمان نژاد
وحید مروج
مهدی جودکی

تحت نظارت

دکتر علیرضا بصیری

فهرست مطالب

۴	۱. فصل اول : آشنایی با SQL مقدماتی
۴	۱.۱. مقدمه.....
۴	۱.۲. دستورات Data Definition Language.....
۴	۱.۲.۱. ساخت جدول.....
۵	۱.۲.۲. قیدها.....
۶	۱.۲.۳. ویرایش ساختار جداول.....
۷	۱.۳. دستورات Data Manipulation Language.....
۷	1.3.1. وارد کردن اطلاعات به یک جدول.....
۷	1.3.2. دستور SELECT.....
۸	1.3.3. دستور WHERE.....
۸	۱.۳.۴. دستور ORDER BY.....
۹	1.3.5. دستور UPDATE.....
۹	1.4. دستور JOIN و انواع آن.....
۱۳	۲. فصل دوم : آشنایی با Adventure Works 2012 و چند دستور مهم در SQL
۱۳	2.1. معرفی پایگاه داده AdventureWorks 2012.....
۱۳	۲.۱.۱. روش نصب.....
۱۳	۲.۱.۲. آشنایی با Adventure Works 2012.....
۱۶	۲.۲. عملگرها بر روی مجموعه نتایج (result set operators).....
۱۶	۲.۲.۱. اجتماع (UNION).....
۱۷	2.2.2. اشتراک (INTERSECT).....
۱۷	۲.۲.۳. تفاضل (EXCEPT).....
۱۸	2.3. دستور CASE.....
۱۸	۲.۳.۱. دستور CASE در قالب ساده.....
۱۹	2.3.2. دستور CASE در قالب جستجویی.....
۲۰	2.4. توابع تجمیعی در SQL.....
۲۱	2.5. Having و Group By.....
۲۳	۲.۶. تمرین.....

۱. فصل اول : آشنایی با SQL مقدماتی

۱,۱. مقدمه

در این جلسه مطالب مربوط به SQL مقدماتی که در کلاس تدریس شده، مرور می‌شوند. پایگاه داده‌ای به نام University ساخته می‌شود و با مرور دستورات SQL این پایگاه داده، تکمیل می‌شود.

۱,۲. دستورات Data Definition Language

این دستورات برای تعریف جداول، نوع داده‌های موجود در جداول و ویرایش و حذف آن‌ها، استفاده می‌شود.

۱,۲,۱. ساخت جدول

برای ساخت جداول از دستور Create Table استفاده می‌شود. ساختار کلی این دستور بصورت زیر است:

```
CREATE TABLE table_name
(
    column1 datatype [ NULL | NOT NULL ],
    column2 datatype [ NULL | NOT NULL ],
    ...
);
```

مثال:

```
CREATE TABLE Students
(
    FirstName varchar(20) NOT NULL,
    LastName varchar(30) NOT NULL ,
    StudentNumber char(7) PRIMARY KEY,
    BirthYear int,
);
```

چرا در ساخت جدول بجای فیلد Age از BirthYear استفاده شده؟

۱,۲,۳. قیدها

قیدهایی که در ساخت جداول استفاده می شوند عبارتند از:

NOT NULL	نشان می دهد که ستون مربوطه نمی تواند مقدار NULL داشته باشد.
UNIQUE	نشان می دهد که هر سطر از جدول باید مقداری یکتا برای این ستون داشته باشد.
PRIMARY KEY	به عنوان ترکیبی از دو قید قبلی کار می کند و با هر مقدار از آن می توان یک و دقیقاً یک سطر از جدول را مشخص کرد.
FOREIGN KEY	نشان می دهد که مقادیر این ستون باید از بین مقادیر موجود در ستونی مشابه اما در جدولی دیگر، باشند.
CHECK	نشان می دهد که مقادیر این ستون باید شرط خاصی داشته باشند.
DEFAULT	یک مقدار اولیه برای مقادیر این ستون مشخص می کند.

جدول ۱-۱

مثال:

```
CREATE TABLE Departments
(
    Name varchar(20) NOT NULL ,
    ID char(5) PRIMARY KEY,
    Budget numeric(12,2),
    Category varchar(15) Check (Category in
('Engineering','Science'))
);
```

```
CREATE TABLE Teachers
(
    FirstName varchar(20) NOT NULL,
    LastName varchar(30) NOT NULL ,
    ID char(7),
    BirthYear int,
    DepartmentID char(5),
    Salary numeric(7,2) Default 10000.00,
    PRIMARY KEY (ID),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(ID),
);
```

```
CREATE TABLE Students
(
    FirstName varchar(20) NOT NULL,
    LastName varchar(30) NOT NULL ,
    StudentNumber char(7) PRIMARY KEY,
    BirthYear int,
    DepartmentID char(5),
    AdvisorID char(7),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(ID),
    FOREIGN KEY (AdvisorID) REFERENCES Teachers(ID)
);
```

۱,۲,۳. ویرایش ساختار جداول

برای تغییر ساختار جداول از دستور ALTER TABLE با ساختار کلی زیر استفاده می شود:

برای اضافه کردن ستون:

```
ALTER TABLE table_name
ADD column_name column_type
```

برای حذف کردن ستون:

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

برای تغییر نوع داده یک ستون:

```
ALTER TABLE table_name
ALTER COLUMN column_name column_type
```

مثال:

```
ALTER TABLE Departments
ALTER COLUMN Name varchar(50)
```

۱,۳. دستورات Data Manipulation Language

۱,۳,۱. وارد کردن اطلاعات به یک جدول

برای وارد کردن اطلاعات به یک جدول از دستور INSERT با ساختار کلی زیر استفاده می شود:

```
INSERT INTO table (column1, column2, ... ) VALUES (expression1, expression2, ... );
```

مثال:

```
INSERT INTO Departments (Name, ID, Budget, Category) VALUES ('Electrical & Computer Engineering Department', 'ECE', 1200000.00 , 'Engineering')
```

```
INSERT INTO Departments (Name, ID, Budget) VALUES ('Mechanical Engineering Department', 'ME', 1000000.00)
```

```
INSERT INTO Departments (Name, ID, Category) VALUES ('Physics Department', 'P' , 'Science')
```

۱,۳,۲. دستور SELECT

برای خواندن اطلاعات از پایگاه داده، از دستور SELECT با ساختار کلی زیر استفاده می شود:

```
SELECT expressions  
FROM table
```

مثال:

```
SELECT Name, ID FROM Departments
```

نتیجه:

	Name	ID
1	Electrical & Computer Engineering Department	ECE
2	Mechanical Engineering Department	ME
3	Physics Department	P

مثال:

```
SELECT * from Departments
```

نتیجه:

	Name	ID	Budget	Category
1	Electrical & Computer Engineering Department	ECE	1200000.00	Engineering
2	Mechanical Engineering Department	ME	1000000.00	NULL
3	Physics Department	P	NULL	Science

۱,۳,۳. دستور WHERE

با استفاده از این دستور، نتایج پرس و جوی نوشته شده را فیلتر می کنیم.

مثال:

```
SELECT * from Departments WHERE Category <> 'NULL'
```

نتیجه:

	Name	ID	Budget	Category
1	Electrical & Computer Engineering Department	ECE	1200000.00	Engineering
2	Physics Department	P	NULL	Science

۱,۳,۴. دستور ORDER BY

با استفاده از این دستور، نتایج پرس و جوی نوشته شده را مرتب می کنیم.

مثال:

- 1-

```
SELECT * from Departments WHERE Budget>550000 ORDER BY Name asc
```
- 2-

```
SELECT * from Departments WHERE Budget>550000 ORDER BY Name desc
```

نتیجه:

	Name	ID	Budget	Category
1	Electrical & Computer Engineering Department	ECE	900000.00	Engineering
2	Mechanical Engineering Department	ME	1000000.00	NULL

-۱

	Name	ID	Budget	Category
1	Mechanical Engineering Department	ME	1000000.00	NULL
2	Electrical & Computer Engineering Department	ECE	900000.00	Engineering

-۲

۱,۳,۵. دستور UPDATE

برای ویرایش اطلاعات پایگاه داده، از دستور UPDATE با ساختار کلی زیر استفاده می شود:

```
UPDATE table
SET column1 = expression1,
    column2 = expression2,
    ...
WHERE conditions;
```

مثال:

```
UPDATE Departments
SET Category = 'Engineering'
WHERE ID = 'ME';
```

۱,۴. دستور JOIN و انواع آن

هنگام نوشتن برخی پرس و جوها تمام اطلاعات مورد نیاز در یک جدول وجود ندارد و باید اطلاعات چند جدول را با هم در اختیار داشته باشیم. همانطور که به یاد دارید، برای این منظور از دستور JOIN استفاده می کنیم که برخی انواع پرکاربرد این دستور را در جدول ۱-۲ بیان می کنیم.

نوع JOIN	توضیح
INNER JOIN	در این روش سطرهایی نمایش داده می شوند که در هر دو جدولی که با هم Join شده اند وجود دارند. در واقع رکوردهایی در نتیجه ظاهر می شوند که متناظرشان (بر اساس فیلدهایی که JOIN روی آنها انجام شده است) در جدول دیگر هم رکورد وجود داشته باشد.
RIGHT JOIN	در نتیجهی این JOIN کلیهی رکوردهای جدول سمت راست عبارت JOIN وجود دارند و برای رکوردهایی که متناظرشان در جدول سمت چپ رکوردی وجود ندارد، مقدار NULL وجود خواهد داشت.
LEFT JOIN	در نتیجهی این JOIN کلیهی رکوردهای جدول سمت چپ عبارت JOIN وجود دارند و برای رکوردهایی که متناظرشان در جدول سمت راست رکوردی وجود ندارد، مقدار NULL وجود خواهد داشت.
FULL JOIN	در نتیجهی این JOIN کلیهی رکوردهای جدولهای هر دو سمت عبارت JOIN وجود خواهند داشت.

جدول ۱-۲

نکته: در صورتی که هنگام JOIN برای یک رکورد از جدول اول n رکورد در جدول دوم وجود داشته باشد، در خروجی n رکورد مشاهده می شود.

در مثال زیر ساختار این دستورات و نتایج حاصل از آن‌ها را مرور خواهیم کرد:

فرض کنید محتویات دو جدول PASSENGER (حاوی اطلاعات مسافران) و PASSENGER_PHONE (حاوی شماره تلفن‌های مسافران) به صورت زیر است:

PASSENGER:

SSN	First_Name	Last_Name	Gender
1111111111	Mike	Anderson	Male
2222222222	Julie	Brown	Female
3333333333	Sue	Jones	Female
4444444444	Andrew	James	Male
5555555555	Ben	Mayer	Male

PASSENGER_PHONE:

Passenger_SSN	Passenger_Phone
1111111111	1230000123
2222222222	4560000456
2222222222	7890000789
5555555555	2580000258
5555555555	3690000369

INNER JOIN:

```
SELECT SSN, First_Name, Last_Name, Gender, Passenger_Phone
FROM Passenger INNER JOIN Passenger_Phone ON
(Passenger.SSN=Passenger_Phone.Passenger_SSN)
```

نتیجه:

SSN	First_Name	Last_Name	Gender	Passenger_Phone
1111111111	Mike	Anderson	Male	1230000123
2222222222	Julie	Brown	Female	4560000456
2222222222	Julie	Brown	Female	7890000789
5555555555	Ben	Mayer	Male	2580000258
5555555555	Ben	Mayer	Male	3690000369

RIGHT JOIN

```
SELECT SSN, First_Name, Last_Name, Gender, Passenger_Phone
```

```
FROM Passenger RIGHT JOIN Passenger_Phone ON
(Passenger.SSN=Passenger_Phone.Passenger_SSN)
```

نتیجه:

SSN	First_Name	Last_Name	Gender	Passenger_Phone
1111111111	Mike	Anderson	Male	1230000123
2222222222	Julie	Brown	Female	4560000456
2222222222	Julie	Brown	Female	7890000789
5555555555	Ben	Mayer	Male	2580000258
5555555555	Ben	Mayer	Male	3690000369

LEFT JOIN

```
SELECT SSN, First_Name, Last_Name, Gender, Passenger_Phone
FROM Passenger LEFT JOIN Passenger_Phone ON
(Passenger.SSN=Passenger_Phone.Passenger_SSN)
```

نتیجه:

SSN	First_Name	Last_Name	Gender	Passenger_Phone
1111111111	Mike	Anderson	Male	1230000123
2222222222	Julie	Brown	Female	4560000456
2222222222	Julie	Brown	Female	7890000789
3333333333	Sue	Jones	Female	NULL
4444444444	Andrew	James	Male	NULL
5555555555	Ben	Mayer	Male	2580000258
5555555555	Ben	Mayer	Male	3690000369

FULL JOIN

```
SELECT SSN, First_Name, Last_Name, Gender, Passenger_Phone
FROM Passenger FULL JOIN Passenger_Phone ON
(Passenger.SSN=Passenger_Phone.Passenger_SSN)
```

نتیجه:

SSN	First_Name	Last_Name	Gender	Passenger_Phone
1111111111	Mike	Anderson	Male	1230000123
2222222222	Julie	Brown	Female	4560000456
2222222222	Julie	Brown	Female	7890000789
3333333333	Sue	Jones	Female	NULL
4444444444	Andrew	James	Male	NULL

5555555555 Ben	Mayer	Male	2580000258
5555555555 Ben	Mayer	Male	3690000369

- در این مثال استثنائاً نتیجه‌ی FULL JOIN و LEFT JOIN یکسان شد.

۲. فصل دوم: آشنایی با Adventure Works 2012 و چند دستور مهم در SQL

در این جلسه ضمن معرفی پایگاه داده AdventureWorks 2012، دو ساختار دستور Case و دستوراتی برای کار با مجموعه نتایج معرفی می شود.

۲,۱. معرفی پایگاه داده AdventureWorks 2012

در این آزمایشگاه سعی خواهد شد اکثر مثال‌ها و تمرین‌ها روی این پایگاه داده معروف انجام پذیرند و بنابراین در این بخش با این پایگاه داده و روش نصب آن آشنا خواهیم شد.

۲,۱,۱. روش نصب

برای نصب Adventure Works 2012 چندین روش وجود دارد که یک روش را در اینجا بررسی می کنیم:
ابتدا AdventureWorks2012_Database.zip را از سامانه الکترونیکی دروس، دانلود کنید.

محتویات فایل زیپ را در آدرس زیر، کپی کنید:

C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DATA
سپس در SQL SERVER 2012 در قسمت OBJECT EXPLORER روی DATABASES کلیک راست کنید و گزینه‌ی ATTACH را انتخاب نمایید و در پنجره‌ای که باز می شود با کلیک روی ADD فایل mdf کپی شده را انتخاب کنید تا به SQL SERVER اضافه شود.

برای مشاهده‌ی سایر روش‌های نصب Adventure Works 2012 به آدرس زیر بروید:

http://social.technet.microsoft.com/wiki/contents/articles/3735.sql-server-samples-readme.aspx#Readme_for_Adventure_Works_Sample_Databases

۲,۱,۲. آشنایی با Adventure Works 2012

Adventure Works نام یک شرکت بین‌المللی کاملاً فرضی است که محصولات آن انواع دوچرخه و محصولات مرتبط با آن است، و پایگاه داده Adventure Works 2012 هم بر اساس محصولات، مشتریان، سفارش‌ها، کارمندان و ... این شرکت فرضی بنا نهاده شده.

از آنجایی که در این آزمایشگاه عمده مثال‌ها و تمرین‌ها بر روی این پایگاه داده انجام می شود، آشنایی کلی با شمای این پایگاه داده ضروری است، از طرفی هم انتظار نمی رود با همه‌ی جداول در این پایگاه داده بطور کامل آشنا باشید اما باید در ابتدای این آزمایشگاه با جداول مهم این پایگاه داده آشنا شوید.

• جدول SalesOrderHeader

این جدول حاوی اطلاعات کلی یک سفارش است. (چیزی شبیه به سربرگ یک فاکتور)
برخی از مهمترین فیلدهای این جدول، در جدول زیر مشاهده می شود:

یک عدد صحیح به عنوان شناسه سفارش	SalesOrderID
تاریخ ثبت سفارش	OrderDate
یک عدد صحیح، نشانگر وضعیت سفارش (۱=در حال پردازش، ۲= تأیید شده و ...)	Status
شناسه صاحب سفارش (کلید خارجی از جدول Customer)	CustomerID
شناسه محلی که سفارش در آن ثبت شده (کلید خارجی از جدول SalesTerritory)	TerritoryID
مجموع قیمت کالاهای موجود در سفارش	SubTotal
قیمت نهایی سفارش برای مشتری (مجموع قیمت کالاهای موجود در سفارش + هزینه ارسال سفارش + مالیات)	TotalDue

جدول ۱-۲

• جدول SalesOrderDetail

این جدول حاوی اطلاعات کالاهای موجود در یک سفارش است. (ردیف های فاکتور)
برخی از مهمترین فیلدهای این جدول، در جدول زیر مشاهده می شود:

شناسه سفارش (کلید خارجی از جدول SalesOrderHeader)	SalesOrderID
کلید اصلی جدول	SalesOrderDetailID
تعداد کالای سفارش داده شده	OrderQty
شناسه کالای سفارش داده شده	ProductID
قیمت واحد کالای سفارش داده شده	UnitPrice
قیمت کل این سطر (قیمت واحد کالا X تعداد کالا)	LineTotal

جدول ۲-۲

- جدول Product

این جدول حاوی اطلاعات کالاهای AdventureWorks است.
برخی از مهمترین فیلدهای این جدول، در جدول زیر مشاهده می شود:

شناسه کالا (کلید اصلی جدول)	ProductID
نام کالا	Name
رنگ کالا	Color
قیمت تمام شده کالا برای AdventureWorks	StandardCost
قیمت کالا برای فروش	ListPrice
اندازه کالا	Size
وزن کالا	Weight
تعداد روزهای مورد نیاز برای تولید کالا	DaysToManufacture
نوع کالا (جاده، کوهستان و ...)	ProductLine
نوع کالا (مخصوص آقایان، مخصوص بانوان، قابل استفاده برای همه)	Style

جدول ۳-۲

- جدول SalesTerritory

این جدول حاوی اطلاعات مکان‌هایی است که AdventureWorks در آن مکان‌ها فعالیت دارد.
اطلاعات بیشتر در مورد این جداول و سایر جداول مهم این پایگاه داده را حتماً در آدرس زیر مطالعه کنید. همچنین در آینده نیز برای نوشتن پرس‌وجوهای مختلف با مراجعه به آدرس زیر مطمئن شوید که اطلاعات را از جداول صحیح استخراج می کنید:

[http://technet.microsoft.com/en-us/library/ms124438\(v=sql.100\).aspx](http://technet.microsoft.com/en-us/library/ms124438(v=sql.100).aspx)

۲,۲. عملگرها بر روی مجموعه نتایج (result set operators)

همانطور که می دانید حاصل هر پرس و جو بصورت `select ... from ...`، یک مجموعه جواب است. در این بخش با عملگرهایی آشنا می شویم که روی این مجموعه نتایج کار می کنند.

۲,۲,۱. اجتماع (UNION)

همانطور که از اسمش پیداست این عملگر اجتماع دو مجموعه جواب را به عنوان خروجی بر می گرداند. برای استفاده از آن، عبارت "UNION" را در میان دو `select statement` قرار می دهیم و اجتماع آن ها را (بدون در نظر گرفتن تکرارها) در خروجی خواهیم داشت.
نحوه استفاده از عملگر اجتماع :

`select_statement UNION select_statement`

توجه: بدیهی است که دو مجموعه جواب حاصل از دو `select statement` که در دو طرف عبارت "UNION" قرار می گیرند دارای ساختار کاملاً یکسان باشند. (تعداد و ترتیب قرار گرفتن ستونها یکسان و نوع داده ی ستون های متناظر هم یکسان) و البته این مطلب برای همه ی عملگرهایی که در ادامه به آن ها خواهیم پرداخت صادق است.
به عنوان مثال دو جدول زیر را در نظر بگیرید:

Coloumn A char(3)	Coloumn B int	Coloumn C char(3)	Column D int
ABC	1	GHI	3
DEF	2	JKL	4
GHI	3	MNO	5

Table2

Table۱

در مثال زیر اجتماع تمام رکوردهای دو جدول فوق مشاهده می شود:

```
SELECT * FROM Table1
UNION
SELECT * FROM Table2
```

و نتیجه برابر خواهد بود با:

```
ColumnA  ColumnB
-----  -
abc      1
def      2
ghi      3
jkl      4
mno      5
```


همانطور که مشاهده کردید، union تکرار را در نظر نمی گیرد. اگر بخواهیم اجتماع دو مجموع جواب را با احتساب تکرارها داشته باشیم از عبارت union all استفاده می کنیم. مثال قبل را با union all امتحان می کنیم:

```
SELECT * FROM Table1
UNION ALL
SELECT * FROM Table2
```

و نتیجه برابر خواهد بود با:

ColumnA	ColumnB
abc	1
def	2
ghi	3
ghi	3
jkl	4
mno	5

۲.۲.۲. اشتراک (INTERSECT)

این عملگر هم دقیقاً مثل عملگر union بر روی دو مجموعه جواب کار می کند با این تفاوت که اشتراک دو مجموعه جواب را بر می گرداند. به مثال زیر توجه کنید:

```
SELECT * FROM Table1
INTERSECT
SELECT * FROM Table2
```

و نتیجه برابر خواهد بود با:

ColumnA	ColumnB
ghi	3

۲.۲.۳. تفاضل (EXCEPT)

این عملگر همانطور که در مثال زیر مشاهده می شود، حاصل تفاضل دو مجموعه نتیجه را به عنوان خروجی بر می گرداند.

```
SELECT * FROM Table1
EXCEPT
SELECT * FROM Table2
```

و نتیجه برابر خواهد بود با:

ColumnA	ColumnB
abc	1
def	2

۲,۳. دستور CASE

به کمک این دستور می توان تعدادی شرط را بررسی نمود و یکی از نتایج ممکن را انتخاب کرد. این دستور در دو قالب استفاده می شود:

۲,۳,۱. دستور CASE در قالب ساده

دستور CASE در این قالب، یک عبارت را با تعدادی عبارت دیگر مقایسه می کند و سپس در مورد نتیجه تصمیم گیری می کند. قالب این دستور بصورت زیر است:

```
CASE input_expression
  WHEN when_expression1 THEN result_expression1
  WHEN when_expression2 THEN result_expression2
  WHEN when_expression3 THEN result_expression3
  ...
  WHEN when_expressionN THEN result_expressionN

[
  ELSE else_result_expression
]
END
```

مثال: در این مثال با استفاده از قالب ساده ی دستور CASE در میان کتاب های یک انتشارات، دسته بندی به نحوی تغییر داده شده که برای مشتریان قابل فهم تر باشد:

```
SELECT
  CASE type
    WHEN 'popular_comp' THEN 'Popular Computing'
    WHEN 'mod_cook' THEN 'Modern Cooking'
    WHEN 'business' THEN 'Business'
    WHEN 'psychology' THEN 'Psychology'
    WHEN 'trad_cook' THEN 'Traditional Cooking'
    ELSE 'Not yet categorized'
  END AS Category,
  Title, Price
FROM titles
WHERE price IS NOT NULL
ORDER BY Category
```

و نتیجه برابر است با:

Category	Title	Price
Business	Cooking with Computers: Surrep	11.95
Business	Straight Talk About Computers	19.99
Business	The Busy Executive's Database	19.99
Business	You Can Combat Computer Stress	2.99
Modern Cooking	Silicon Valley Gastronomic Tre	19.99
Modern Cooking	The Gourmet Microwave	2.99
Popular Computing	But Is It User Friendly?	22.95
Popular Computing	Secrets of Silicon Valley	20.00
Psychology	Computer Phobic AND Non-Phobic	21.59
Psychology	Emotional Security: A New Algo	7.99
Psychology	Is Anger the Enemy?	10.95
Psychology	Life Without Fear	7.00
Psychology	Prolonged Data Deprivation: Fo	19.99
Traditional Cooking	Fifty Years in Buckingham Pala	11.95
Traditional Cooking	Onions, Leeks, and Garlic: Co	20.95
Traditional Cooking	Sushi, Anyone?	14.99

۲,۳,۲. دستور CASE در قالب جستجویی

چندین عبارت BOOLEAN را بررسی می‌کند و بر این اساس در مورد نتیجه تصمیم‌گیری می‌کند. قالب این دستور بصورت زیر است:

```
CASE
  WHEN Boolean_expression1 THEN result_expression1
  WHEN Boolean_expression2 THEN result_expression2
  WHEN Boolean_expression3 THEN result_expression3
  ...
  WHEN Boolean_expressionN THEN result_expressionN
  [
    ELSE else_result_expression
  ]
END
```

مثال: در این مثال با استفاده از قالب جستجویی دستور CASE برای کتاب‌ها دسته‌بندی جدیدی بر اساس قیمت آن‌ها صورت پذیرفته:

```
SELECT
  CASE
    WHEN price IS NULL THEN 'Not yet priced'
    WHEN price < 10 THEN 'Very Reasonable Title'
    WHEN price >= 10 and price < 20 THEN 'Coffee Table Title'
    ELSE 'Expensive book!'
  END AS "Price Category",
  Title
FROM titles
ORDER BY price
```

و نتیجه برابر خواهد بود با:

Price Category	Title
Not yet priced	The Psychology of Co
Not yet priced	Net Etiquette
Very Reasonable Title	You Can Combat Compu
Very Reasonable Title	The Gourmet Microwav
Very Reasonable Title	Life Without Fear
Very Reasonable Title	Emotional Security:
Coffee Table Title	Is Anger the Enemy?
Coffee Table Title	Cooking with Compute
Coffee Table Title	Fifty Years in Bucki
Coffee Table Title	Sushi, Anyone?
Coffee Table Title	The Busy Executive's
Coffee Table Title	Straight Talk About
Coffee Table Title	Silicon Valley Gastr
Coffee Table Title	Prolonged Data Depri
Expensive book!	Secrets of Silicon V
Expensive book!	Onions, Leeks, and G
Expensive book!	Computer Phobic AND
Expensive book!	But Is It User Frien

۲.۴. توابع تجمیعی در SQL

این نوع از توابع بر روی مجموعه ای از داده ها عمل می کنند (مجموعه مقادیر یک ستون) و تنها یک مقدار را بر می گردانند. برخی از توابع پر کاربرد این گروه عبارت اند از :

نام تابع	توضیحات
AVG()	میانگین مقادیر را بر می گرداند
COUNT()	تعداد سطرها را بر می گرداند
SUM()	مجموع مقادیر یک لیست را بر می گرداند
MAX()	بزرگترین مقدار را بر می گرداند
MIN()	کوچکترین مقدار را بر می گرداند
DISTINCT()	مقادیر تکراری را از لیست انتخاب حذف می کند
Group By()	به منظور دسته بندی اطلاعات به کار می رود

جدول ۲-۲

مثلاً تابع COUNT() تعداد مقادیر موجود در یک لیست را بر می گرداند، مقادیر NULL و مقادیر تکراری را نیز شامل می شود. به منظور حذف مقادیر تکراری می توان از واژه DISTINCT استفاده کرد.

ساختار:

```
SELECT COUNT(column_name)
FROM table_name;
```

```
SELECT COUNT(DISTINCT column_name)
FROM table_name;
```

مثال:

```
SELECT COUNT(*)
FROM [Sales].[Customer];
```

۲,۵. Group By و Having

گاهی اوقات به منظور گزارش گیری و دستیابی به اطلاعات هوشمند تجاری نیاز به خلاصه کردن اطلاعات وجود دارد. با استفاده از دستورهای می توانید اطلاعات را دسته بندی کرده و خلاصه ای از اطلاعات یک فیلد خاص، در هر دسته را در یک مقدار نشان دهید.

مثال:

```
SELECT [CustomerID], avg([SubTotal]) as Customer_AVG_Pay
FROM [Sales].[SalesOrderHeader]
GROUP BY [CustomerID]
```

نتیجه:

	CustomerID	Customer_AVG_Pay
1	14324	1707.1427
2	22814	4.99
3	11407	53.99
4	28387	583.97
5	19897	596.96
6	15675	2402.1266
7	24165	1523.42
8	27036	7.28
9	18546	29.48
10	11453	2725.66
11	17195	1665.3337

در این پرس و جوها، هر فیلدی که در مقابل Select آمده اما مؤلفه ی تابع تجمیعی نیست، باید به عنوان مؤلفه Group By ظاهر شود.

مثال :

```
SELECT [TerritoryID]
      ,max([SalesQuota]) max_SalesQuota
      ,avg([Bonus]) avg_Bonus
FROM [AdventureWorks2012].[Sales].[SalesPerson]
group by [TerritoryID]
```

نتیجه:

	TerritoryID	max_SalesQuota	avg_Bonus
1	NULL	NULL	0.00
2	1	300000.00	4133.3333
3	2	300000.00	4100.00
4	3	250000.00	2500.00
5	4	250000.00	2775.00
6	5	300000.00	6700.00
7	6	250000.00	2750.00
8	7	250000.00	985.00
9	8	250000.00	75.00
10	9	250000.00	5650.00

وقتی می خواهید برای مقدار یک تابع تجمیعی در پرس و جو شرطی تعیین کنیم، باید بجای where از having استفاده کنید.

مثال:

```
SELECT [TerritoryID]
      ,max([SalesQuota]) max_SalesQuota
      ,avg([Bonus]) avg_Bonus
FROM [AdventureWorks2012].[Sales].[SalesPerson]
group by [TerritoryID]
having max([SalesQuota]) > 30000
```

نتیجه:

	TerritoryID	max_SalesQuota	avg_Bonus
1	1	300000.00	4133.3333
2	2	300000.00	4100.00
3	3	250000.00	2500.00
4	4	250000.00	2775.00
5	5	300000.00	6700.00
6	6	250000.00	2750.00
7	7	250000.00	985.00
8	8	250000.00	75.00
9	9	250000.00	5650.00
10	10	250000.00	5150.00

۲.۶. تمرین

۱- برای adventure worker 2012 پرس و جویی بنویسید اطلاعات کلی سفارشات در حال پردازشی را برگرداند که مبلغ آن ها بین ۱۰۰۰۰۰ و ۵۰۰۰۰۰ است و محل سفارش از کشور فرانسه یا یکی از کشورهای منطقه آمریکای شمالی است.

۲- برای Adventure Works 2012 پرس و جویی بنویسید که برای هر سفارش، شماره شناسایی سفارش، شماره شناسایی مشتری، مبلغ سفارش، تاریخ سفارش و نام محلی که سفارش در آن ثبت شده را در خروجی نشان دهد.

۳- برای adventure worker 2012 پرس و جویی بنویسید که نشان دهد هر کالا در کدام منطقه بیشترین تعداد سفارش را داشته است.

۴- در adventure worker 2012 جدولی با نام NAmerica_Sales ایجاد کنید که شمای آن مطابق با شمای پرس جوی تمرین شماره ۱ باشد .

سپس از بین رکوردهای موجود در خروجی تمرین ۱ ، فقط رکوردهای مربوط به کشورهای منطقه آمریکای شمالی را وارد این جدول نمایید .

یک ستون به جدول فوق اضافه کنید که نوع داده ای آن (4)char باشد و نتوان چیزی جز LOW یا High یا Mid در آن نوشت .

سپس کاری کنید که مقدار این ستون برای سفارش هایی که مبلغ آن از مبلغ میانگین سفارشات منطقه آمریکای شمالی بیشتر است High ، برای سفارش هایی که مبلغ آن ها با مبلغ میانگین سفارشات منطقه آمریکای شمالی برابر است Mid و برای سایر سفارش ها Low باشد.

۵- با اجرای پرس و جوی زیر میزان حقوق کارمندان را به ازای هر ساعت کار همراه با

BusinessEntityID هر کارمند خواهید داشت:

```
USE AdventureWorks2012
GO
SELECT BusinessEntityID ,max(Rate)FROM HumanResources.EmployeePayHistory
GROUPBY BusinessEntityID

ORDERBY BusinessEntityID
```

حال پرس و جویی بنویسید که نتیجه‌ی آن دارای سه ستون است. ستون اول BusinessEntityID هر کارمند، ستون دوم شامل مقادیر جدید حقوق کارکنان است که این گونه محاسبه می‌شود:

طیف حقوق پرداختی را بصورت صعودی مرتب کنید و آن را به چهار قسمت تقسیم کنید:

✓ کارمندانی که میزان حقوقشان به ازای هر ساعت کار جز یک چهارم اول طیف است، (کسانی که کمترین حقوق را دارند) حقوقشان ۲۰ درصد افزایش یابد.

✓ کارمندانی که میزان حقوقشان به ازای هر ساعت کار جز یک چهارم دوم طیف است، حقوقشان ۱۵ درصد افزایش یابد.

✓ کارمندانی که میزان حقوقشان به ازای هر ساعت کار جز یک چهارم سوم طیف است، حقوقشان ۱۰ درصد افزایش یابد و بقیه ، حقوقشان ۵ درصد افزایش یابد.

و ستون سوم سطح دریافتی هر کارمند را نشان می‌دهد و بدین صورت محاسبه می‌شود:

✓ کارمندانی که میزان حقوقشان (قبل از اضافه شدن درصدی به آن) به ازای هر ساعت کار کمتر از ۲۹ بوده سطح ۳

✓ کارمندانی که میزان حقوقشان (قبل از اضافه شدن درصدی به آن) به ازای هر ساعت کار بین ۲۹ و ۵۰ بوده سطح

✓ و بقیه کارمندان در سطح ۱ هستند.

✓ نمونه خروجی مورد نظر در شکل ۲-۱ مشاهده می‌شود.

	BID	NewSalary	LEVEL
1	1	131.775000	1
2	2	69.807650	1
3	3	49.759580	2
4	4	34.323130	2
5	5	37.596145	2
6	6	37.596145	2
7	7	58.052920	1
8	8	46.995210	2
9	9	46.995210	2
10	10	48.852920	2
11	11	34.615440	3
12	12	30.000000	3
13	13	30.000000	3
14	14	41.466355	2

شکل ۲-۱

۳. فصل سوم: مجوزها، login ها و امنیت

۳,۱. مقدمه

در این فصل به بررسی برخی از مهمترین موارد امنیتی در SQL Server می پردازیم. بحث پیرامون مسائل امنیتی بسیار گسترده و در مواردی پیچیده است و از حوصله این درس خارج است. بنابراین سعی شده در این فصل به طور خلاصه و مفید مفاهیم مهم و اولیه در بحث امنیت، بیان شود.

در ابتدا کار را با آشنایی با مجوزها و مدیریت آنها شروع می کنیم و در ادامه به بحث login ها در SQL Server می پردازیم و در آخر هم به معرفی انواع نقش ها و نحوه مدیریت آنها اشاره خواهیم کرد.

۳,۲. مجوزها^۱

مجوزها مجموعه کارهایی که یک کاربر می تواند انجام دهد را مشخص می کنند. این مجوزها شامل مجوز login کردن، مجوز دسترسی به پایگاه داده و جداول و محتویات آن می شود.

در اینجا به بررسی مجوزهای مهم برای کار در پایگاه داده ها می پردازیم که به این مجوزها اصطلاحاً Object Permissions گفته می شود.

۳,۲,۱. Object Permissions

در نگاه اول به نظر می رسد بعد از دادن مجوز login و مجوز دسترسی به پایگاه داده، مجوز دیگری مورد نیاز نباشد اما در sql server برای استفاده از سطوح دسترسی متفاوت برای کاربران و نقش های متفاوت، از Object Permissions استفاده می شود که قابل ارائه برای جداول، view ها و رویه های ذخیره شده هستند. این مجوزها شش نوع مختلف دارند:

۱- SELECT

کاربرانی که این مجوز را داشته باشند، می توانند دستور select را برای view یا جدولی که این مجوز را برای آن دارند، اجرا کنند.

۲- INSERT

کاربرانی که این مجوز را داشته باشند، می توانند دستور insert را اجرا کنند. (دقت کنید که این مجوز کاملاً مستقل از مجوز select است)

^۱ Permissions

۳- UPDATE

کاربرانی که این مجوز را داشته باشند، می توانند دستور update را اجرا کنند. (دقت کنید در اینجا هم مثل insert، کاربری که این مجوز را دارد لزوماً مجوز select را ندارد.)

۴- DELETE

کاربرانی که این مجوز را داشته باشند، می توانند دستور delete را اجرا کنند.

۵- REFERENCES

کاربرانی که این مجوز را داشته باشند، می توانند از قیودی مانند کلید خارجی استفاده کنند.

۶- EXECUTE

کاربرانی که این مجوز را داشته باشند، می توانند رویه ها یا توابع را اجرا کنند.

۳,۲,۲. مدیریت مجوزهای کاربران

برای مدیریت مجوزهای کاربران دو روش وجود دارد. هم می توان بصورت گرافیکی و با استفاده از SQL Server Management Studio این کار را انجام داد و هم می توان با اجرای کدهای مربوطه، این کار را انجام داد. در ادامه هر دو روش را بررسی خواهیم کرد.

• مدیریت مجوزهای کاربران با استفاده از SQL Server Management Studio

برای این منظور در Object Explorer وارد قسمت Security شوید و سپس به قسمت Logins بروید و در آنجا با انتخاب کاربر موردنظر، روی آن کلیک راست کنید و Properties را انتخاب کنید. در هر یک از صفحات پنجره ای که باز می شود، مجوزهایی وجود دارد که بصورت گرافیکی می توانید آن ها را لغو یا واگذار کنید

• مدیریت مجوزهای کاربران با اجرای کد

برای این منظور ۳ دستور متفاوت وجود دارد که آن ها را در ادامه بررسی می کنیم.

• GRANT

به کمک دستور GRANT مجوزها را به کاربران واگذار می کنیم. ساختار کلی این دستور بصورت زیر است:

```
GRANT ALL [ PRIVILEGES ] | permission_name [ ,...n ]  
ON [ schema_name ]. object_name [ ( column [ ,...n ] ) ]  
TO <database_principal> [ ,...n ]  
[ WITH GRANT OPTION ]
```

در ادامه ساختار فوق را بررسی می کنیم.

پس از عبارت GRANT می‌توان عبارت ALL یا ALL PRIVILEGES را قرارداد که به معنی واگذاری کلیه مجوزهای ممکن برای آن OBJECT خواهد بود. با توجه به اینکه مجوز برای چه چیزی واگذار می‌شود، معنای متفاوتی خواهد داشت. (مثلاً اگر مجوز برای جدول یا VIEW باشد، ALL به معنی INSERT, DELETE, SELECT, UPDATE, REFERENCES خواهد بود اما اگر مجوز برای یک رَوِیه باشد، ALL به معنی EXECUTE خواهد بود.)

البته می‌توان بجای واگذاری همه‌ی مجوزهای ممکن، فقط نام مجوزهایی را که می‌خواهیم واگذار کنیم را بعد از عبارت GRANT بنویسیم.

پس از مشخص شدن مجوزها، عبارت ON و پس از آن نیز نام OBJECT موردنظر را می‌آوریم. و بعد از آن عبارت TO را نوشته و پس از آن کاربر یا نقشی را که باید مجوز به آن واگذار شود را مشخص می‌کنیم در نهایت هم می‌توان با نوشتن عبارت WITH GRANT OPTION این اجازه را داد تا کاربرانی که این مجوز را می‌گیرند، بتوانند این مجوز را به دیگران هم بدهند.

• DENY

به کمک دستور DENY مجوزها را از کاربران سلب می‌کنیم. در واقع این دستور بیان می‌کند که کاربران چه کارهایی را نمی‌توانند انجام دهند. ساختار کلی این دستور بصورت زیر است:

```
DENY ALL [ PRIVILEGES ] | permission [ ,...n ]  
ON [ schema_name ]. object_name [ ( column [ ,...n ] ) ]  
TO <database_principal> [ ,...n ]  
[ CASCADE ]
```

تنها عبارت جدیدی که در این ساختار به چشم می‌خورد، عبارت CASCADE است که بیانگر آن است که مجوزهای ذکر شده در عبارت DENY، نه تنها از کاربران ذکر شده سلب می‌شود بلکه از سایر کاربرانی که این مجوزها توسط کاربران ذکر شده به آن‌ها داده شده نیز، سلب می‌شود.

توجه! همانطور که می‌دانید یک کاربر ممکن است چند نقش متفاوت داشته باشد. در صورتی که یک مجوز را به یکی از نقش‌های یک کاربر واگذار کنیم و همان مجوز را از نقش دیگری که آن کاربر دارد بگیریم، همیشه قدرت دستور DENY بیشتر است و بنابراین آن مجوز از کاربر بطور کل سلب می‌شود.

• REVOKE

این دستور اثر دستور GRANT یا DENY که قبلاً اجرا شده را لغو می‌کند. ساختار کلی این دستور بصورت زیر است:

```
REVOKE ALL [ PRIVILEGES ] | permission [ ,...n ]
ON [ schema_name ]. object_name [ ( column [ ,...n ] ) ]
{ FROM | TO } <database_principal> [ ,...n ]
[ CASCADE ]
```

۳,۳. LOGIN

به طور کلی می توان گفت هر کاربر با دو روش می تواند در SQL Server شناخته شود:

۱- استفاده از Windows Authentication

در این روش کاربر وارد Windows می شود و SQL Server توسط ارتباط های مطمئن، اطلاعات کاربر را از Windows دریافت می کند و بدین ترتیب کاربر شناخته می شود.

۲- استفاده از SQL Server Authentication

در این روش برای ورود به SQL Server از یک LOGIN که پیشتر ساخته ایم استفاده می کنیم. به این منظور ابتدا با در محیط SQL Server یک LOGIN ایجاد می کنیم و سپس از SQL Server خارج می شویم و از اینجا به بعد می توانیم با هر دو روش وارد SQL Server شویم. در ادامه این روش را بیشتر توضیح خواهیم داد.

۳,۳,۱. روش ساخت LOGIN

در اینجا دو روش را برای ساخت LOGIN در SQL Server توضیح می دهیم:

۱- ساخت LOGIN با اجرای دستور CREATE LOGIN

در این روش با اجرای دستور CREATE LOGIN یک LOGIN را می سازیم. این دستور دارای ساختار کلی زیر است:

```
CREATE LOGIN login_name { WITH PASSWORD = { 'password' } [ MUST_CHANGE ]
[ , SID = sid | DEFAULT_DATABASE = database | DEFAULT_LANGUAGE = language [ ,...
] ]
| FROM WINDOWS [ WITH DEFAULT_DATABASE = database | DEFAULT_LANGUAGE = language [
,... ] ] }
```

در این ساختار برخی از عبارات مهم را بررسی خواهیم کرد و اطلاعات بیشتر و ساختار مفصل تر را می توانید از آدرس زیر بدست آورید:

<http://msdn.microsoft.com/en-us/library/ms189751.aspx>

- [MUST_CHANGE]
آوردن این عبارت بعد از تعیین رمز ورود برای LOGIN اختیاری است و بیانگر این است که کاربر باید پس از اولین ورود با این LOGIN رمز ورود را عوض کند.
- همچنین می توان برخی تنظیمات اولیه را برای LOGIN در تعریف در نظر گرفت مثل SID = sid (یک شناسه برای LOGIN)، DEFAULT_DATABASE (تعیین پایگاه داده پیش فرض برای این LOGIN)، DEFAULT_LANGUAGE تعیین زبان پیش فرض LOGIN و ...
- FROM
آوردن این عبارت به معنای آن است که این لاگین ویژه ی SQL Server نیست بلکه مربوط به منبع دیگری است که اجازه ی دسترسی به SQL Server به آن داده شده.

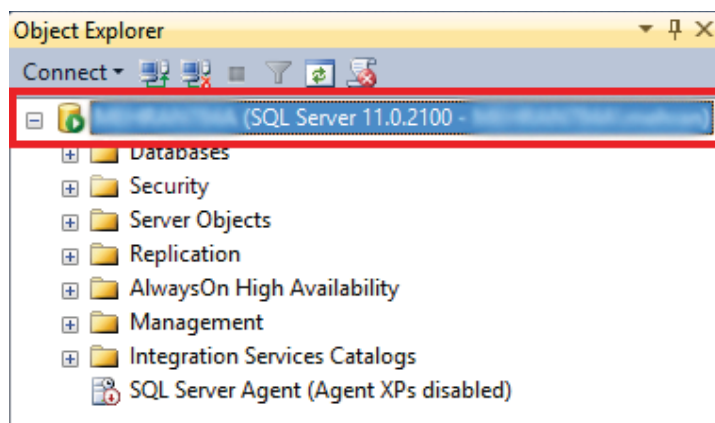
۲- ساخت LOGIN با استفاده از SQL Server Management Studio

بدین منظور در قسمت Object Explorer شاخه ی Security را انتخاب کرده و روی زیرشاخه ی Logins کلیک راست کرده و New Login را انتخاب کنید.

در پنجره ای که باز می شود ابتدا گزینه ی SQL Server Authentication را انتخاب نمایید و سپس مشخصات login موردنظرتان را تنظیم نمایید و در نهایت Ok را بزنید تا login ساخته شود.

توجه!

- پس از ساخت Login لازم است مراحل زیر را طی کنید:
- روی SQL Server Instance کلیک راست کنید و گزینه Properties و سپس Security را انتخاب کنید. (در شکل ۵-۱، مشخص شده است)



شکل ۱-۳

- از پنجره باز شده گزینه ی SQL Server and Windows authentication mode را انتخاب کنید.
- حال سرویس SQL Server را (از قسمت سرویس های ویندوز) restart کنید.

۳,۴. نقش‌ها^۲

هر نقش در واقع مجموعه‌ای از مجوزهای داده شده یا گرفته شده، است که به کاربران اختصاص داده می‌شود. در هر لحظه به یک کاربر می‌توان چندین نقش را اختصاص داد. نقش‌ها را می‌توان به دو دسته تقسیم کرد:

۱- نقش‌های مربوط به Server^۳

۲- نقش‌های مربوط به پایگاه داده^۴

۳,۴,۱. نقش‌های مربوط به Server

این نقش‌ها که درون SQL Server موجود هستند بیشتر به منظور نگهداری سیستم و انجام وظایف مدیریتی مربوط به server و همچنین اعطای مجوزهای مربوط به کارهای مستقل از پایگاه داده‌ها (مثل ساخت login) استفاده می‌شوند. ۹ نقش مربوط به Server فیکس شده در SQL Server را در زیر بررسی می‌کنیم.

نقش	توضیح
Sysadmin	صاحب این نقش می‌تواند هر عملی را در SQL Server انجام دهد.
Serveradmin	صاحب این نقش توانایی انجام تنظیمات مربوط به پیکربندی server و shutdown کردن server را دارد.
Setupadmin	صاحب این نقش توانایی مدیریت linked Server ها را دارد.
Securityadmin	این نقش توانایی مدیریت loginها، و ساخت مجوزهای پایگاه داده را به کاربران می‌دهد.
Processadmin	صاحبان این نقش توانایی مدیریت processهایی که روی موتور پایگاه داده اجرا می‌شوند را دارند.
Dbcreator	توانایی صاحبان این نقش به ایجاد، تغییر و حذف پایگاه‌های داده محدود می‌شود.
Diskadmin	صاحب این نقش توانایی مدیریت فایل‌های مرتبط با server را دارد.
Bulkadmin	صاحب این نقش اجازه‌ی ورود وسیع اطلاعات (اجرای دستور Bulk Insert) را دارد.
Public	همه‌ی loginها در SQL Server بصورت پیش‌فرض این نقش را دارند.

جدول ۳-۱

از SQL Server 2012 امکانی اضافه شد که در نتیجه‌ی آن کاربران می‌توانند نقش‌های مربوط به سرور را خود چنان تعریف کنند که متناسب با نیازشان باشد. برای این منظور سه گام زیر باید پیموده شود:

^۲ Roles

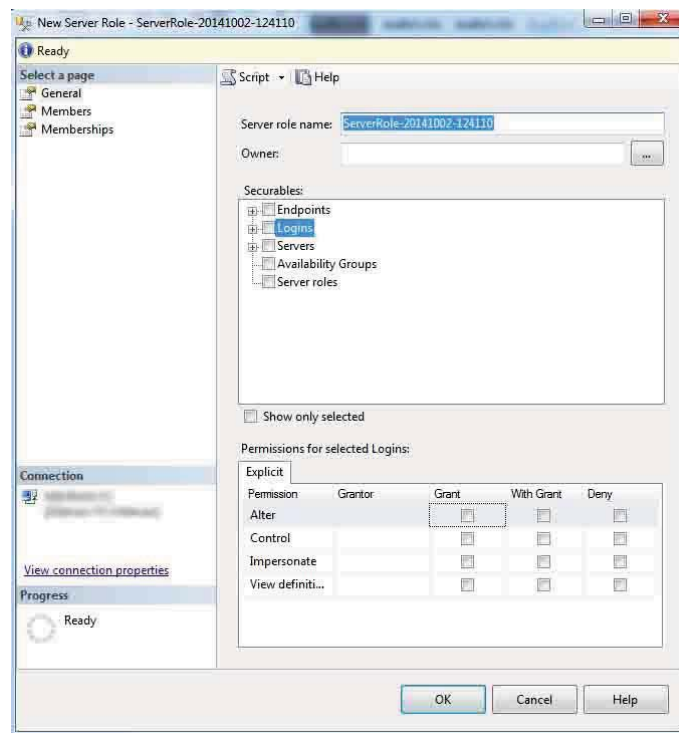
^۳ Server Roles

^۴ Database Roles

- ۱- ساخت یک نقش مربوط به Server
- ۲- تخصیص مجوزهای مربوط به Server به نقش ساخته شده
- ۳- تخصیص نقش به loginهای موردنظر

در محیط SQL Server Management Studio هم با پیمودن گامهای زیر می توان نقش مربوط به Server موردنظر را ساخت:

- ۱- در قسمت Object Explorer بخش Security را باز کنید، سپس روی Server Roles کلیک راست کنید و New Server Role را انتخاب کنید. تا پنجره ی زیر باز شود:



شکل ۲-۳

- ۲- در قسمت Server Role Name نامی برای نقش موردنظر وارد کنید.
- ۳- در قسمت Securables مشخص کنید که این نقش در کدام موارد باید مجوز داشته باشد و در قسمت Permissions مربوطه، مجوزهای موردنظر برای نقش را انتخاب کنید.

۴- سپس در صفحه‌ی Members همین پنجره، Add را بزنید و سپس با کلیک روی Browse، می‌توانید login‌های موجود و قابل تخصیص به نقش موردنظر تان را ببینید. پس از این می‌توانید نقش موردنظر را بسازید.

۳،۴،۲. نقش‌های مربوط به پایگاه داده

این نقش‌ها از نظر دامنه^۵ به یک پایگاه داده محدود می‌شوند. یعنی داشتن یک نقش پایگاه داده‌ای در یک پایگاه داده، به معنای داشتن آن نقش در سایر پایگاه داده‌ها نیست. ۹ نقش مربوط به پایگاه داده فیکس شده در SQL Server را در ادامه بررسی می‌کنیم.

نقش	توضیح
db_owner	این نقش را باید به کسانی داد که می‌خواهیم کنترل کامل روی پایگاه داده داشته باشند.
db_securityadmin	صاحب این نقش می‌تواند مدیریت کلیدهای نقش‌های مربوط به پایگاه داده و تخصیص مجوزها در آن پایگاه داده را انجام دهد.
db_accessadmin	صاحب این نقش مدیریت تخصیص کاربران و loginها به پایگاه داده مربوطه را انجام می‌دهد.
db_backupoperator	صاحب این نقش می‌تواند از پایگاه داده نسخه پشتیبانی ^۶ تهیه کند.
db_ddladmin	صاحب این نقش توانایی اجرای دستورات DDL را در پایگاه داده مربوطه خواهد داشت.
db_datawriter	صاحب این نقش می‌تواند عملیات Insert, Delete, Update را روی جداول پایگاه داده مربوطه، انجام دهد.
db_datareader	صاحب این نقش می‌تواند کلیه داده‌های جداول پایگاه داده مربوطه را بخواند.
db_denydatawriter	صاحب این نقش نمی‌تواند عملیات Insert, Delete, Update را روی جداول پایگاه داده مربوطه، انجام دهد.
db_denydatareader	صاحب این نقش نمی‌تواند هیچ یک از داده‌های جداول پایگاه داده مربوطه را بخواند.

جدول ۲-۳

اما معمولاً این نقش‌های فیکس شده برای شروع کار هستند و بخش عمده‌ای از کار امنیتی یک پایگاه داده مربوط به تشخیص و ساخت نقش‌های مختلف برای پایگاه داده است. برای این نقش‌ها هم دقیقاً مثل آنچه که برای مجوزهای کاربران ذکر شد، می‌توان مجوزها را با دستورات GRANT, REVOKE, DENY مدیریت کرد.

^۵ Scope

^۶ Backup