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 سلیمان‌نژاد

وحید مروج

مهدی جودکی

تحت نظارت

دکتر علیرضا بصیری

فهرست مطالب

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؟

۱,۲,۲. قیدها

قیدهایی که در ساخت جداول استفاده می شوند عبارتند از:

NOT NULL	نشان می دهد که ستون مربوطه نمی‌تواند 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)
```

CRUD

1-CREAT
2-READ
3-UPDATE
4-DELETE

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:

۱-

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

۲-

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

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 پرس‌وجوها 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 منظوراز دستور JOIN استفاده می‌کنیم که برخی انواع پرکاربرد این دستور را در جدول ۱-۲ بیان می‌کنیم.

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 می‌شوند 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.