بسمه تعالی

فصل پنجم

آشنایی با زبان SQL

(Standard Query Language)

ارائه دهنده : محمد فرجی مهماندار

زبان SQL

یک زبان استاندارد کامپیوتر می‌باشد که برای ارتباط با سیستم مدیریت بانک اطلاعاتی رابطه‌ای مورد استفاده قرار می‌گیرد. استاندارد SQL توسط موسسه استاندارد ملی آمریکا (ANSI) و سازمان استانداردهای بین المللی (ISO) تعریف شده است. نام رسمی این زبان، زبان استاندارد بین‌المللی بانک اطلاعاتی SQL (۱۹۹۲) می‌باشد. معمولا از جدیدترین نسخه این استاندارد با عنوان SQL/92یاد می‌شود

یکی از ویژگی‌های اصلی زبان SQL این است که یک زبان محاوره‌ای (DECLARATIVE) یا ناروشمند (NONPROCEDURAL) می‌باشد. از نقطه نظر برنامه‌نویسان، این امر اشاره می‌کند که برنامه‌نویس، نیازی ندارد که تمامی عملیاتی را که کامپیوتر برای رسیدن به یک نتیجه مشخص باید انجام دهد، گام به گام مشخص نماید. در مقابل، برنامه نویس آنچه را که باید انجام بگیرد به سیستم مدیریت بانک اطلاعاتی نشان می‌دهد و سپس به سیستم اجازه می‌دهد تا خود در مورد نحوه رسیدن به نتایج مطلوب تصمیم گیری نماید.

جملات یا دستورات تشکیل دهنده زبان SQL عموما به ۴ دسته تقسیم می شود :

- زیر زبان تعریف داده ها (DDL)
- زیر زبان دستکاری داده ها (DML)
- زیر زبان جستجوی داده ها(DQL)
- زیر زبان کنترل داده ها (DCL)

زیر زبان تعریف داده ها (DDL)

زبان DDL برای ساختن جداول و تعریف بخشهای موجود در جداول استفاده می‌شود. جدول درهرزمانی می‌توانند ایجاد شوند حتی اگرکاربران درحال استفاده ازبانک‌اطلاعاتی باشند.

- **ایجاد یک جدول :** باید از دستور CREATE TABLE استفاده نمایید.

```
CREATE TABLE table_name
( column_name   type(size)   [ NULL/NOT NULL] ,
  column_name   type(size)   [NULL/NOT NULL]  ,
...) ;
```

- **نحوه اصلاح ساختاریک جدول :** استفاده ازدستورALTER TABLE می‌توان نحوه تعریف یک جدول را تغییرداد.

برای افزودن یک ستون جدید به جدول بایدازگزینه ADD استفاده نمود.

ALTER TABLE table_name ADD (column_name type(size) [NULL/NOT NULL])

مثال: برای افزودن ستونی برای نگهداری نام همسرهریک ازکارمندان درجدول EMP بایدازدستور زیراستفاده نمایید.

ALTER TABLE EMP ADD (SPOUSE_NAME CHAR(10) ;

برای تغییریک ستون بایدازگزینه MODIFY استفاده نمود:

ALTER TABLE table_name MODIFY(column_name type(size) [NULL/NOT NULL])

مثال :برای تغییرطول ستون ENAME درجدول EMP به مقدار 25 کاراکتر باید ازدستورزیراستفاده نمایید.

ALTER TABLE EMP MODIFY (ENAME CHAR(25)) ;

- **نحوه حذف یک جدول :**

با استفاده از دستورDROP TABLE میتوان یک جدول اوراکلی راحذف نمود.

DROP TABLE EMP ;

زیر زبان تعریف داده ها (DDL)

NAME	NULL?	DATA TYPE
DEPTNO	NOT NULL	NUMBER(۲)
DNAME		CHAR(۳۰)
LOC		CHAR(۳۰)

```
CREATE TABLE DEPT (
DEPTNO   NUMBER(2) NOT NULL ,
DNAME    CHAR(12) ,
LOC      CHAR(12)) ;
```

NAME	NULL?	DATA TYPE
EMPNO	NOT NULL	NUMBER(۴)
ENAME		CHAR(۱۰)
JOB		CHAR(۱۰)
MGR		NUMBER(۴)
HIREDATE		DATE
SAL		NUMBER(۷,۲)
COMM		NUMBER(۷,۲)
DEPTNO	NOT NULL	NUMBER(۲)

```
CREATE TABLE EMP (
EMPNO        NUMBER(4)  NOT  NULL ,
ENAME        CHAR(10) ,
JOB          CHAR(10)  ,
MGR          NUMBER(4) ,
HIREDATE     DATE  ,
SAL          NUMBER  ,
COMM         NUMBER(7,2)  ,
DEPTNO       NUMBER(2)  NOT  NULL) ;
```

محدودیت‌های جامعیت (INTEGRITY CONSTRAINTS) :

محدودیت	تعریف
NOT NULL	از وارد شدن مقادیر خنثی در یک ستون جلوگیری می‌کند.
UNIQUE	از وارد شدن مقادیر تکراری در یک ستون جلوگیری می‌کند.
PRIMARY KEY	ملزم می‌داردکه‌تمامی‌مقادیروارد شده به یک ستون منحصربفرد بوده و NULL نباشند.
FOREIGN KEY	محدودیت کلید خارجی مکانیزم اصلی کلید تقویت جامعیت متقابل بین جداول یک بانک اطلاعاتی رابطه‌ای است
CHECK	با محدودیتهای بررسی درستی داده‌ها می توان شرط های لازم برای درستی داده ها را بررسی کرد

مثال زبان تعریف داده ها

```
CREATE  TABLE  EMPLOYEE _TBL
(EMP _ID            CHAR        (9)        NOT NULL      PRIMARY KEY,
EMP_NAME            VARCHAR2  (40)        NOT NULL,
EMP_ST_ADDR         VARCHAR2 (20)         NOT NULL,
EMP_ST_CITY         VARCHAR2  (15)        NOT NULL ,
EMP_ST              CHAR        (2)        NOT NULL,
EMP_ZIP             NUMBER      (5)        NOT NULL,
EMP_PHONE           NUMBER      (10)       NULL            UNIQUE,
CONSTRAINT CHK_EMP_ZIP CHECK (EMP_ZIP='46234') );

 CREATE  TABLE  EMPLOYE _PAY_TBL
(EMP _ID                    CHAR(9)                    NOT  NULL,
 PAY_ID                     CHAR(9)                    NOT NULL,
POSITION                    VARCHAR2 (15)            NOT  NULL,
DATE_HIRE                   DATE                       NULL,
PAY_RATE                    NUMBER (4,2)             NOT  NULL,
DATE_LAST_RAISE             DATE                       NULL,
CONSTRAINT  PK_EMP_PAY_TBL  PRIMARY KEY (EMP_ID,PAY_ID) ,
CONSTRAINT FK_EMP_ID  FOREIGN KEY (EMP_ID) REFERENCES
 EMPLOYEE_TBL (EMP_ID),
CONSTRAINT CHK_PAY CHECK (PAY_RATE >12. 50));
```

مثال زبان تعریف داده ها

دستورات معادل ایجاد بانک تهیه کنندگان – قطعات – پروژه‌ها، با مشخصات ذیل را بنویسید :

S (S# ,SNAME ,STATUS ,CITY)

P (P# ,PNAME ,COLOR , WEIGHT ,CITY)

J (J# ,JNAME , CITY)

SPJ (S# ,P# ,J# ,QTY)

NAME	NULL?	DATA TYPE
S#	NOT NULL	CHAR (۴)
SNAME	NOT NULL	VARCHAR(۱۰۰)
STATUS		CHAR(۳۰)
CITY		CHAR(۳۰)

NAME	NULL?	DATA TYPE
P#	NOT NULL	CHAR (۴)
PNAME	NOT NULL	VARCHAR(۱۰۰)
COLOR		CHAR(۳۰)
WEIGHT		NUMBER(8)
CITY		CHAR(۳۰)

NAME	NULL?	DATA TYPE
J#	NOT NULL	CHAR (۴)
JNAME	NOT NULL	VARCHAR(۱۰۰)
CITY		CHAR(۳۰)

NAME	NULL?	DATA TYPE
S#	NOT NULL	CHAR (۴)
P#	NOT NULL	CHAR (۴)
J#	NOT NULL	CHAR (۴)
QTY	NOT NULL	NUMBER(8)

مثال زبان تعریف داده ها

ایجاد جدول sp در Oracle :

```
CREATE TABLE sp
(snum character(3) ,
 pnum character(3) ,
 qty integer ،
 CONSTRAINT pk_snumpnum PRIMARY KEY (snum,pnum),
 CONSTRAINT fk_snum
     FOREIGN  KEY (snum)
     REFERENCES  s(snum)
 ON DELETE CASCADE ،
 CONSTRAINT fk_pnum
     FOREIGN  KEY (pnum)
     REFERENCES  p(pnum)
     ON DELETE CASCADE)
```

- **درج سطرهای جدید دریک جدول :**

برای افزودن سطرهای جدید به یک جدول بایدازدستور INSERT استفاده نمایید شکل کلی این دستوربصورت زیراست:

INSERT INTO table-name [(column1, column2,…)]
VALUES (value1 , value2 ,…)

مثال :

INSERT INTO P

VALUES ('P7','Pn7','RED',10,'C2')

INSERT INTO P(COLOR,WEIGHT, P#,CITY,PNAME)

VALUES('RED',10,'P7', 'C2', 'PN7')

با استفاده از دستور INSERTمی‌توان سطرهای یک جدول را با استفاده از جدول دیگر کپی نمود

INSERT INTO table-name[(column, column,…)

SELECT select-list

FROM table(s);

مثال :

INSERT INTO DI0HISTORY (EMPNO,ENAME, SAL , JOB , HIREDATE)

SELECT EMPNO,ENAME, SAL , JOB , HIREDAT FROM EMP

WHERE DEPTNO=10;

- **اصلاح سطرهای یک جدول :**

برای تغییر مقادیرسطرهای یک جدول بایدازدستور UPDATE استفاده نمایید شکل کلی این دستور، بصورت زیراست :

```
UPDATE  table-name [alias]
SET column [, column...] =(expression, subquery)
[ WHERE condition] ;
```

مثال :برای تغییر سطرمربوط به SCOTT، باید دستورزیررا وارد نمایید.

```
UPDATE EMP
SET JOB = 'SALESMAN' , HIREDATE= SYSDATE, SAL = SAL*1. 1
WHERE ENAME = 'SCOTT' ;
```

به هنگام سازی بیش از یک جدول :

شماره تهیه کننده S1 را به S11 تغییر دهید.

```
UPDATE S
SET S#='S11'
WHERE S#='S1';

UPDATE SP
SET S#='S11'
WHERE S#='S1' ;
```

این UPDATE باید PROPAGATE شود توسط Cascade بخاطر رعایت قاعده C2.

این عمل ، توسط Cascade بخاطر رعایت قاعده C2 انجام شده است

مثال زبان دستکاری داده ها

مثال: ۱۰ واحد به QTY تهیه کنندگان ساکن C3 اضافه کنید.

```
UPDATE SP
SET QTY= QTY + 10
WHERE SNUM IN ( SELECT SNUM
                FROM S
                WHERE CITY = 'C3')
```

پاسخ به شکلهای دیگر :

```
UPDATE SP
SET QTY = QTY + 10
WHERE 'C3' = (SELECT CITY
              FROM S
               WHERE S.S# = SP.S#) ;
```

```
UPDATE SP
 SET QTY= QTY + 10
 WHERE EXISTS ( SELECT *
                FROM S
                WHERE SP.S# = S.S#  AND
                      S.CITY = 'C3')
```

تمرین : QTY تهیه کنندگان ساکن C2 را صفر (0) نمایید.

- **حذف سطرهای یک جدول**

برای حذف سطرهای یک جدول بایدازدستورDELETE استفاده نمایید

شکل کلی این دستور بصورت زیراست:

```
DELETE FROM  table [WHERE condition];
```

مثال : تهیه کننده S2 را حذف کنید .

```
DELETE FROM S      WHERE S#='S2';
```

بخاطر رعایت قاعده C2 باید این حکم نیز اجرا شود (بطور اتوماتیک سیستم اعمال می کند):

```
DELETE FROM SP    WHERE S#='S2';
```

همه رکوردهای جدول را حذف می کند.

```
DELETE FROM SP;
```

مثال : محمولات مربوط به تهیه کنندگان شهر c3 را حذف کنید

```
DELETE FROM SP
WHERE SNUM IN (SELECT SNUM
               FROM S
               WHERE CITY= 'C3')
```

```
DELETE FROM SP
WHERE 'C3' =( SELECT CITY
              FROM S
              WHERE SP.SNUM= S.SNUM )
```

زیر زبان جستجوی داده ها(DQL)

شکل کلی دستور SELECT به صورت زیر است :

```
SELECT [DISTINCT] items
FROM table(s)
[WHERE condition(s)]
[ORDER BY]
[GROUP BY]
[HAVING ]
```

[ORDER BY] → برای خروجیهای Sorted است.
Option ایجاد نظم در خروجیها می باشد

مثال : مشخصات تهیه کنندگان ساکن شهر C2 را بدهید ؟

```
SELECT S# , Sname , Status , City
FROM S
WHERE City ='c2' ;
```

S#	SNAME	STATUS	CITY
S1	Sn1	20	C2
S4	Sn4	20	C2

وقتی تمام ستونهای جدول را بخواهید نیازی به ذکر نام صفات خاصه نیست ، بلکه یک * بجای آنها کافی است.

* SELECT تمامی مشخصات را می آورد و مخصوصا در مواردی که تعداد صفات خاصه زیاد است مفید است.

زیر زبان جستجوی داده ها(DQL)

مثال زیر دو ستون از جدول را می دهد.

S#	CITY
S1	C2
S2	C3
S3	C3
S4	C2
S5	C1

```
SELECT S#, City
FROM S ;
```

مثال :

Qualifier ستون و یا Qualifier صفت خاصه ، قید ستون

```
SELECT S.S# , S.Status
FROM S
WHERE City='C2'  or City='C3' ;
```

S#	Status	City
S1	20	ساکن شهر C2
S3	30	ساکن شهر C3
S4	20	ساکن شهرC2

در این مثال نیازی به تصریح قید نیست ، مواردی وجود دارد که در آنها استفاده از Qualifier توصیه می شود و گاه الزامی است.

(توصیه در مواردی است که بخواهید به Query وضوح ببخشید. الزام وقتی است که ستونهای همنام در جداول مختلف داشته باشیم

.مثل : City در S,SP)

تمرین : همین Query را با جبر رابطه ای بنویسید.

زیر زبان جستجوی داده ها(DQL)

شماره تهیه کنندگان و وضعیت آنها را بدهید. جدول جواب به نظم صعودی مقادیر status مرتب شده باشد.

SELECT S# ,Status

FROM S

ORDER BY Status

ORDER BY Status DESC

ORDER BY 2 شماره ستون درجدول جواب

ORDER BY 2 DESC

ASCENDING صعودی default است و نیاز به تصریح ندارد.

شماره هر قطعه و وزن آنرا به گرم بدهید. فرض کنید DBA وزن را به واحد کیلو گرم ذخیره نموده.

SELECT P# , WEIGHT*1000 AS 'WEIGHT IN GRAMS'

FROM P

جدول جواب صورتی چنین دارد:

P#	WEIGHT IN GRAMS
P1	12000
P2	17000
P3	17000
P4	14000
p5	12000

این دو ستون بی نامند.

برای اینکه جدولی با ستونهای نامدار داشت این جدول جواب را می توان با حکم INSERT در جدولی مناسب با ستونهای نامدار وارد کرد.

زیر زبان جستجوی داده ها(DQL)

روش دوم :

SELECT P# , 'WEIGHT IN GRAMS' , WEIGHT*1000

FROM P

P#		
P1	WEIGHT IN GRAMS	12000
P2	WEIGHT IN GRAMS	17000
P3	WEIGHT IN GRAMS	17000
P4	WEIGHT IN GRAMS	14000
p5	WEIGHT IN GRAMS	12000
p6	WEIGHT IN GRAMS	19000

عملگرهای SQL	
وجود بین دومقدارمشخص	BETWEEN
معادل هر یک ازمقادیرلیست مشخص شده	IN
منطبق بایک الگوی کاراکتری	LIKE
آیا معادل مقدارنول است؟	IS NULL

تمامی مدیران دپارتمانها، بهمراه همه افراد منشی موجود در دپارتمان شماره 10 باید از دستور زیراستفاده نماییم.

```
SELECT  *     FROM  EMP
WHERE JOB = 'MANAGER'  OR (JOB='CLERK' AND DEPTNO =10)
```

پیوند بین جدول

بازیابی از چند جدول:

Query: مشخصات قطعات و تهیه کنندگان از یک شهررا بدهید.

(بحث کلید خارجی بحثی داشتیم که آیا ارتباط تنها توسط کلید خارجی است؟)

سیستم ابتدا ضرب کارترین دو رابطه را ایجاد می کند. سپس tuple های حائز شرایط را استخراج می کند. می توان در عمل join شرایطی را نیز مطرح کرد

```
SELECT  S.*   ,  P.*
   FROM     S , P
    WHERE  S.CITY = P.CITY
```

عمل join (Natural Join)

S#	SNAME	STATUS	CITY	P#	PNAME	COLOR	WEIGHT	CITY
S1	SN1	20	C2	P1	Nut	Red	12	C2
S1	SN1	20	C2	P4	Screw	Red	14	C2
S1	SN1	20	C2	P6	Cog	Red	19	C2
S2	SN2	10	C3	P2	Bolt	Green	17	C3
S2	SN2	10	C3	P5	Cam	Blue	12	C3
S3	SN3	30	C3	P2	Bolt	Green	17	C3
S3	SN3	30	C3	P5	Cam	Blue	12	C3
S4	SN4	20	C2	P1	Nut	Red	12	C2
S4	SN4	20	C2	P4	Screw	Red	14	C2

پیوند بین جدول

مثال :اسم جفت شهرهایی را بدهید که تهیه کننده شهر اول قطعه ای انبار شده در شهر دوم را تولید کند.

(الزام به استفاده از Qualifier داریم)

```
SELECT DISTINCT S.CITY , P.CITY
FROM S , SP , P
WHERE S.S#=SP.S#   AND   P.P#=SP.P#
```

مثال: شماره جفت تهیه کنندگان همشهری را بدهید

S

S#	Sname	Status	City
S1	.	.	C۲
S2	.	.	C۳
S3	.	.	C۳
S4	.	.	C۲
S5	.	.	C1

جدول جواب

S#	S#
S1	S4
S2	S3

برای پاسخگویی به این قبیل Query ها باید S را با خود در SQL ، Join کرد

```
SELECT FIRST.S# , SECOND.S#
FROM S  FIRST , S  SECOND
WHERE FIRST.CITY = SECOND.CITY
```

پیوند بین جدول

از یک تکنیک دگر نامی استفاده شده است S را به نام First و به نام Second نامیده ایم .

بعد از FROM هرگاه نام یک رابطه بیابید و بعد از حداقل یک Blank ، یک نام بیاید ، سیستم آن نام را ، نام دیگر برای آن رابطه در نظر می گیرد.

حاصل SELECT:

S#	S#
S1	S1
S1	S4
S2	S2
S2	S3
S3	S2
S3	S3
S4	S1
S4	S4

تکراری

بدیهات

سیستم چه می کند ؟

۲ تا CURSOR می گیرد و دومی را MOVE می دهد

برای حذف بدیهات و تکراریها شرط زیر را اضافه کنید :

AND FIRST.S# <> SECOND.S#

در نتیجه خواهیم داشت:

```
SELECT DIDTINCT FIRST.S# , SECOND.S#
FROM S  FIRST , S  SECOND
WHERE FIRST.CITY = SECOND.CITY
AND FIRST.S# < SECOND.S#
```

پیوند بین جدول

شماره جفت تهیه کنندگانی را بدهید که از یک شهر نباشد.

```
SELECT DISTINCT FIRST.S# , SECOND.S#
FROM S  FIRST , S  SECOND
WHERE FIRST.CITY != SECOND.CITY
AND FIRST.S# < SECOND.S#
```

توابع جمعی یا گروهی (Aggregate Functions)

برای پاسخگویی به سوالاتی از قبیل چه تعداد قطعه داریم ؟ ماکزیمم Quantity چقدر است ؟

خود Select قادر به پاسخگویی نیست. توابعی پیش بینی شده است که در متن Select بکار می روند.

این توابع عبارتند از :

AVG([DISTINCT |ALL] n)	میانگین n رابا صرفنظر کردن مقادیرنول بدست می‌آورد.
MAX([DISTINCT|ALL] expr)	حداکثر مقدارعبارت را درسطرهای مورد پرس‌وجو نشان می‌دهد.
MIN([DISTINCT |ALL]expr)	حداکثر مقدارعبارت را درسطرهای مورد پرس‌وجو نشان می‌دهد.
SUM([DISTINCT | ALL] n)	مجموع n رادرسطرهای مورد پرس‌وجو با صرف‌نظرکردن مقادیرنول بدست می‌آورد.
COUNT([DISTINCT |ALL] n)	تعداد n را در سطرهای مورد پرس وجو، با صرف‌نظر کردن مقادیر نول، بدست می‌آورد.

حالتی از COUNT موسوم به COUNT(*) تمام سطرها را – حتی تکراری – می شمارد.

توابع جمعی یا گروهی (Aggregate Functions)

تعداد محمولات را بدهید.(هر سطر SP یک محموله است)

```
SELECT COUNT(*)
FROM SP ;
```

چند نوع قطعه تهیه شده است.

```
SELECT COUNT (DISTINCT P#)
FROM SP  ;
```

کلا چند نوع قطعه وجود دارد.

```
SELECT COUNT(*)
FROM P ;
```

برای محاسبه میانگین حقوق همه کارمندان بایدعبارت زیررادرج نمایید.

```
SELECT  AVG(SAL) FROM  EMP ;
```

برای پیداکردن حداقل حقوق افراد منشی باید عبارت زیر را وارد نمایید.

```
SELECT  MIN(SAL)
FROM  EMP
WHERE JOB='CLERK';
```

توابع جمعی یا گروهی (Aggregate Functions)

استفاده از GROUP BY :

شماره هر قطعه تهیه شده و کل تعداد تهیه شده از هر قطعه را بدهید .

```
SELECT P# ,SUM(QTY)
FROM SP
GROUP BY P# ;
```

- GROUP BY CLAUSE جدول داده شده بعد از FROM را منطقا گروه بندی می کند
- به نحوی که در هر گروه مقدار ستون یا ستونهای داده شده پس از GROUP BY یکسان است.
- گروه بندی بر اساس آنچه بعد از GROUP BY داده می شود
- آنگاه تابع SUM عمل می شود.
- ستونهایی که حاصل اعمال توابع هستند بی نام می باشند

S#	P#	QTY
S1	P1	300
S2	P1	300
S1	P2	200
S2	P2	400
S3	P2	200
S4	P2	200
S1	P3	400
S1	P4	200
S1	P5	100

P#	
P1	600
P2	1000
P3	400
P4	200
P5	100

توابع جمعی یا گروهی (Aggregate Functions)

کاربرد HAVING: شماره قطعاتی را بدهید که توسط بیش از یک تهیه کننده تهیه شده باشد.

```
SELECT      P#
FROM   SP
GROUP  BY  P#
HAVING  COUNT ( * ) > 1 ;
```

- نقش having درگروه همان است که نقش where در tuple.
- Where برای بیان شرط یا شرایط ناظر به سطر است. having برای بیان شرط یا شرایط ناظر به گروه.
- نکته ۱ : HAVING همیشه با GROUP BY می آید و مستقل معنی ندارد.
- نکته ۲: استفاده از GROUP BY ناروشمندی SQL را کمی تضعیف می کند. (زیرا می گویید گروهبندی کن) و آنرا تا حدی روشمند میکند.

شماره هر تهیه کننده و تعداد کل قطعاتی را که تهیه کرده بدهید

```
select s#, sum(QTY)
from sp
group by s#
```

شماره تهیه کنندگانی را بدهید که دو نوع قطعه تهیه کرده اند

```
select s#    from sp
group by s#
having count(*)=2
```

توابع جمعی یا گروهی (Aggregate Functions)

شماره هر تهیه کننده و تعداد انواع قطعاتی که تهیه کرده را در مورد تهیه کنندگانی که بیش از یک قطعه تهیه کرده اند ، بدهید

```
select s#, count(*)
from sp
group by s#
having count(*)>1
```

در مورد قطعاتی که توسط سه تهیه کننده تهیه شده اند شماره قطعه و تعداد کل تهیه شده از قطعه را بدهید

```
select p#, sum(qty)
from sp
group by p#
having count(*)=3
```

شماره هر تهیه کننده ، شماره قطعه تهیه شده و تعداد کل تهیه شده از آن قطعه را بدهید

```
select s#, p#, sum(qty)
from sp
group by s#, p#
```

با فرض داشتن جدول COMPUTERS که ستونهای آن عبارتند از: نام دفتر، شماره اتاق، شماره کامپیوتر، نوع CPU، مقدارRAM،حجم HARD DISK و ... ، گزارشی تهیه کنید که ستونهای آن نام دفتر، شماره اتاق و تعداد کامپیوترهای موجود در اتاق باشد.

```
select officename, roomno, count(*)
from computers
group by officename, roomno
order by officename, roomno
```

پرس وجوهای فرعی (SUB QUERIES) یاتودرتو

یک پرس‌وجوی فرعی یک عبارتSELECT است که در درون یک عبارت SELECT دیگر قرارمی‌گیرد.

شکل کلی آن بصورت زیراست:

```
SELECT  column1, column2,...
FROM  EMP
WHERE column=
                (SELECT  column
                 FROM  table
                 WHERE condition)
```

بطور معمول، نخست، دستور SELECT داخلی‌تر اجرا شده وسپس حاصل آن بعنوان شرط دستور SELECTاصلی مورداستفاده قرارمی‌گیرد. استفاده ازاین نوع پرس‌وجو زمانی مفید است که بخواهیم سطرهای یک جدول را براساس شرایط داده‌های موجود در همان جدول، بدست آوریم. بعنوان مثال، برای نمایش اسامی کارمندانی که کمترین حقوق رادریافت می‌کنند باید روال زیر را اجرا نماییم.

- حداقل حقوق را بیابیم.

```
SELECT  MIN(SAL) FROM  EMP ;
```

- کارمندانی را پیدا کنیم که آن مقدار حقوق را دریافت می‌کنند.

```
SELECT  ENAME, JOB , SAL
FROM  EMP
WHERE SAL=(کمترین حقوق')
```

می‌توان دو عبارت فوق را بصورت یک پرس‌وجوی تودرتو با یکدیگر تلفیق نمود.

```
SELECT  ENAME, JOB , SAL     FROM  EMP
WHERE SAL = (SELECT  MIN(SAL) FROM  EMP) ;
```

پرس وجوهای فرعی (SUB QUERIES) یاتودرتو

مثال : اسامی تهیه کنندگانی که حداقل یک قطعه آبی رنگ تهیه می کنند.

```
SELECT  Sname
 FROM     S
 WHERE  S# IN ( SELECT  S#
                FROM   SP
                WHERE  P#  IN  ( SELECT  P#
                                 FROM     P
                                 WHERE   P.COLOR = 'BLUE '  )) ;
```

سیستم از درونی ترین SELECT شروع میکند.

با استفاده از JOIN

```
SELECT  S.SNAME
FROM S , SP , P
WHERE S.S# = SP.S# AND
      SP.P# = P.P# AND
      P.COLOR = 'BLUE'
```

مثال : شماره تهیه کنندگان هم شهر با 'S1' را بنویسید.

تمرین: همین Q را با JOIN بنویسید .

```
SELECT    S#
FROM     S
WHERE  CITY  = ( SELECT  CITY
                 FROM      S
                 WHERE   S#  =  ' S1' )
```

پرس وجوهای فرعی (SUB QUERIES) یاتودرتو

نمایش دپارتمان‌هایی که دارای میانگین حقوقی کم تراز دپارتمان ۳۰ هستند، باید عبارت زیر را وارد نمایید.

```
SELECT  DEPTNO , AVG (SAL)
FROM  EMP
HAVING  AVG (SAL)> (SELECT  AVG (SAL)
                    FROM  EMP
                    WHERE DEPTNO =30)
GROUP BY DEPTNO ;
```

نمایش نام، شغل، وتاریخ استخدام کارمندانی که حقوقشان بیشترازحداکثر میزان حقوقشان در دپارتمان SALES است

```
SELECT  ENAME , JOB , HIREDATE , SAL
FROM  EMP
WHERE SAL > (SELECT  MAX(SAL)
             FROM  EMP
             WHERE DEPTNO =(SELECT  DEPTNO
                            FROM  DEPT
                            WHERE DNAME = 'SALES'));
```

پیداکردن کارمندانی که میزان حقوقشان بیشتر از میانگین حقوق دپارتمان آنهاست

```
SELECT  EMPNO , ENAME , SAL , DEPTNO
FROM  EMP E
WHERE SAL > (SELECT  AVG(SAL)
             FROM  EMP
             WHERE DEPTNO = E. DEPTNO)
ORDER BY DEPTNO ;
```

بازیابی به کمک EXISTS

Q : اسامی تهیه کنندگان قطعه P2 را بدهید . (تاکنون به سه روش این Q را زدیم) از این جا نتیجه می گیریم که یک Query در SQL به گونه های مختلف (چندین گونه) تنظیم (formulate) می شود که منطقا نوعی افزونگی در روش می باشد . Date این را یک ایراد مطرح کرده است (اما از دید برنامه نویسان یک نکته مثبت است .)

```
SELECT SNAME FROM S
WHERE EXISTS (SELECT * FROM SP
        (1)    WHERE SP.S#=S.S# AND SP.P#='P2')
```

نکته : وقتی در یک Query درونی به صفت خاصه ای بیرون از آن پرس و جو ارجاع می دهیم ، Qualifier باید قید شود . اما اگر صفت خاصه مربوط به همان Q درونی باشد ، الزامی نیست .

نکته : Clause شماره (۱) ظاهرا شبیه به Clause ای است که در join می نویسیم . سیستم در EXISTS عمل پیوند انجام نمی دهد . شاید بتوان گفت که از این لحاظ EXISTS از join کاراتر است زیرا ضرب کارتزین انجام نمی دهد .

نحوه اجرا : به ازای هر سطر از رابطه S سیستم بررسی می کند آیا وجود دارد سطری در SP که S# آن همان سطر از S و P# آن برابر P2 ؟ پس وجود را چک می کند (EXISTANCIAL) (اگر وجود داشته باشد یعنی عبارت محاسباتی True باشد) SNAME آن S# جواب است .

بازیابی به کمک EXISTS

مثال : اسامی تهیه کنندگانی را بدهید که قطعه P2 را تهیه نکرده اند .

پاسخ با استفاده ا ز NOT EXISTS :

```
SELECT SNAME FROM S
    WHERE NOT EXISTS (SELECT * FROM SP
                      WHERE SP.S#=S.S# AND SP.P#='P2')
```

پاسخ با استفاده از SUBQUERY:

```
SELECT SNAME
FROM S
WHERE SNUM NOT IN ( SELECT SNUM
                    FROM SP
                    WHERE PNUM = 'P2')
```

توجه کنید که query ذیل پاسخ صحیح نمی باشد.

```
SELECT SNAME
FROM S,SP
WHERE S.SNUM = SP.SNUM
      AND SP.PNUM != 'P2'
```

در واقع اسامی تهیه کنندگانی را می دهد که حد اقل یک قطعه بجز P2 تهیه کرده اند. لذا ممکن است تهیه کنندگان P2 را نیز شامل باشد و پاسخ سوال مورد نظر نمی باشد.

بازیابی به کمک EXISTS

نام تهیه کنندگانی را بدهید که همه قطعات را تهیه کرده اند

```
SELECT SNAME
FROM S
WHERE   NOT EXISTS
        (SELECT *
           FROM P
           WHERE NOT EXISTS
           (SELECT *
            FROM SP
            WHERE SP.P#=P.P# AND S.S#=SP.S#))
```

روش دوم : با استفاده از SUBQUERY :

```
select sname
from s
where s# in  ( select s#
               from sp
               group by s#
               having count(*) = (select count(*)
                                  from p))
```

بازیابی به کمک EXISTS

نام تهیه کنندگانی را بدهید که هیچ قطعه ای را تهیه نکرده اند.

```
SELECT SNAME
FROM S
WHERE  NOT EXISTS
        ( SELECT *
            FROM SP
            WHERE S.S#=SP.S#)
```

روش دوم : با استفاده از SUBQUERY :

```
SELECT SNAME
FROM S
WHERE S# NOT IN (SELECT S#
                 FROM SP)
```