

به نام آنکه جان را فکرت آموخت



## بخش چهارم: مقدمات پیاده سازی و SQL

مرتضی امینی

نیمسال دوم ۹۴-۹۵

(محتویات اسلایدها برگرفته از یادداشت های کلاسی استاد محمدتقی روحانی رانکوهی است.)



- برای پیاده‌سازی طراحی منطقی انجام شده در یک سیستم مدیریت پایگاه داده‌ها نیاز به یک زبان پایگاهی داریم.
- زبان SQL زبان استاندارد انجام عملیات پایگاهی در پایگاه داده‌های رابطه‌ای (از دیدگاه کاربردی: جدولی) است.

□ دستورهای (Structured Query Language (SQL

Data Definition Language (DDL)	}	
Data Manipulation Language (DML)		
Data Control Language (DCL)		

□ چند دستور از DDL

CREATE TABLE ایجاد جدول	}	
DROP TABLE حذف جدول		
ALTER TABLE تغییر جدول		

- **نکته:** در دستورات SQL در دو طرف مقادیر متنی یا رشته‌ای از single quote استفاده می‌شود (بسیاری از سیستم‌های پایگاه داده double quote را هم می‌پذیرند) ولی در اطراف مقادیر عددی چیزی قرار نمی‌گیرد.



# تعریف و حذف پایگاه داده و شِما

بخش چهارم: مقدمات پیاده‌سازی و SQL

۳

☐ دستور تعریف پایگاه داده

**CREATE DATABASE** *DatabaseName*

☐ دستور حذف پایگاه داده

**DROP DATABASE** *DatabaseName*

☐ در اغلب سمپادها می‌توان در یک پایگاه داده چند شما تعریف کرد.

☐ دستور تعریف و حذف شِما

**CREATE SCHEMA** *SchemaName*

**DROP SCHEMA** *SchemaName*

☐ شماي پایگاه داده‌ها عبارت است از تعریف (توصیف) ساختهای منطقی طراحی شده و نوعی برنامه است

شامل تعدادی دستور برای تعریف و کنترل داده‌ها.

☐ در واقع شما شامل همه جداول، نوعها، دامنه‌ها، دیدها و محدودیتهای مرتبط با یک برنامه کاربردی است.



## دستور تعریف جدول CREATE TABLE ☐

**CREATE TABLE** *TableName*

```
{ ( columnName dataType [NOT NULL | UNIQUE]
[DEFAULTL defaultOption][CHECK (searchCondition)] [, ...] ) }
[PRIMARY KEY (listOfColumns), ]
{[UNIQUE (listOfColumns),][, ...]}
{[FOREIGN KEY (listOfForeignKeyColumns)
REFERENCES ParentTableName [(listOfCandidateKeyColumns)],
[ON UPDATE referentialAction]
[ON DELETE referentialAction]][, ...]}
{[CHECK (searchCondition)][, ...]}
```

تعریف جدول‌ها: شمای پایگاه جدولی

می‌توان جدول را به صورت موقت نیز (با استفاده از CREATE TEMPORARY TABLE) ایجاد کرد. جدول ☐

موقت حاوی داده‌های ناپایا است و پس از اینکه برنامه کاربر (SQL Session) اجرایش تمام بشود، این جدول توسط سیستم حذف می‌شود.



□ انواع داده‌های قابل استفاده در تعریف ستون‌ها عبارتند از:

□ کاراکتری: CHAR(n), VARCHAR(n)

□ بیتی: BIT [VARYING] (n)

□ عددی: NUMERIC(p, q), REAL, INTEGER, SMALLINT, FLOAT(p),

DOUBLE PRECISION

□ زمانی: DATE, TIME, TIMESTAMP, INTERVAL

□ ....

□ در برخی DBMS ها، نوع داده‌های خاصی پشتیبانی می‌شود که امکان ذخیره، بازیابی و پردازش داده‌های

از آن نوع را برای کاربر تسهیل می‌نماید. به طور مثال نوع داده جغرافیایی در PostgreSQL.



☐ **Default:** تعیین مقدار پیش فرض یک ستون

☐ **Not Null:** ستون ناهیچ مقدار

☐ **Unique:** یکتایی مقادیر ستون(ها)

☐ **Primary Key:** کلید اصلی (می توان تعدادی از ستونها را با یکدیگر به عنوان کلید اصلی تعریف کرد)

☐ **Foreign Key .... References ....:** کلید خارجی (می توان تعدادی از ستونها را با یکدیگر به عنوان

کلید خارجی تعریف کرد)

☐ **Check:** تعیین محدودیت مقداری برای مقادیر ستون



## مثالی از تعریف جدول

بخش چهارم: مقدمات پیاده‌سازی و SQL

۷

شِمای پایگاه داده جدولی:



```
CREATE TABLE STT
( STID          CHAR(8) NOT NULL,
  STNAME        CHAR(25) ,
  STLEV         CHAR(12) ,
  STMJR         CHAR(20) ,
  STDEID        CHAR(4)
)
PRIMARY KEY (STID) ;
CHECK STMJR IN { 'bs', 'ms', 'doc', '???' }
```

```
CREATE TABLE COT
( COID          CHAR(6) NOT NULL,
  COTITLE       CHAR(16) ,
  CREDIT        SMALLINT ,
  COTYPE        CHAR(1) ,
  CODEID        CHAR(4) ,
)
PRIMARY KEY (COID) ;
```

محدودیت صفتی (ستونی) [کلاز کنترلی]



## مثالی از تعریف جدول (ادامه)

بخش چهارم: مقدمات پیاده‌سازی و SQL



**CREATE TABLE SCT**

( STID                      CHAR(8) NOT NULL ,  
COID                      CHAR(6) NOT NULL ,  
TR                        CHAR(1) ,  
YR                        CHAR(5) ,  
GRADE                    DECIMAL(2 , 2)  
)

**PRIMARY KEY ( STID, COID )**

**CHECK  $0 \leq \text{GRADE} \leq 20$**

محدودیت صفتی (ستونی) [کلاس کنترلی]

**FOREIGN KEY (STID) REFERENCES STT (STID)**

**ON DELETE CASCADE**

**ON UPDATE CASCADE**

**FOREIGN KEY (COID) REFERENCES COT (COID)**

**ON DELETE CASCADE**


**ON UPDATE CASCADE**






## دستور حذف جدول DROP TABLE

**DROP TABLE *tablename* [CASCADE| RESTRICT]**

**CASCADE**  باعث می‌شود که همه اشیاء وابسته به جدول (مانند دیدهای تعریف شده بر روی آن یا

محدودیت‌هایی مانند کلید خارجی وابسته به آن) نیز به صورت خودکار حذف شود.

**RESTRICT**  در صورت وجود دیگر اشیاء وابسته به جدول، از حذف آن جلوگیری می‌کند. پیش‌فرض

این دستور، RESTRICT است.



**DROP TABLE SCT**



## □ دستور تغییر جدول ALTER TABLE

ALTER TABLE *tableName* اضافه کردن ستون، تغییر تعریف ستون، حذف ستون و ...

[ADD [COLUMN] *columnName* *dataType* [NOT NULL] [UNIQUE]

[DEFAULT *defaultOption*] [CHECK (*searchCondition*)] ]

[DROP [COLUMN] *columnName* [RESTRICT | CASCADE] ]

[ADD [CONSTRAINT [*constraintName*]] *tableConstraintDefinition*]

[DROP [CONSTRAINT *constraintName* [RESTRICT | CASCADE] ]

[ALTER [COLUMN] SET DEFAULT *defaultOption*]

[ALTER [COLUMN] DROP DEFAULT]

...

اضافه کردن ستون «وضعیت» به جدول اطلاعات دانشجو



ALTER TABLE STT

ADD COLUMN STATE CHAR(10)



بخش چهارم: مقدمات پیاده‌سازی و SQL

و نه دستورات

**Data Manipulation (DM)**

**Data Definition (DD)**

**Data Controller (DC)**

در شمای پایگاهی ← دستورات

این جدایی چه مزایایی دارد؟



سیستم با شمای پایگاهی چه می‌کند؟



اطلاعات موجود در آن را در جایی به نحوی ذخیره می‌کند. ← **در تعدادی جدول**

**کاتالوگ سیستم**

دیکشنری سیستم

مِتا داده‌ها

حاوی فراداده‌ها و داده‌های کنترلی در مورد داده‌های کاربران



مثالی از جدول‌های کاتالوگ:



SysTables

...	تعداد ستون	تاریخ	ایجاد کننده	نام جدول
	5	D1	C1	STT
	5	D2	C1	COT
	5	D2	C2	SCT
	⋮	⋮	⋮	⋮

جدولی که جدول‌ها را مدیریت می‌کند.

SysCols

...	طول	نوع	نام جدول	نام ستون
	8	CHAR	STT	STID
	25	CHAR	STT	STNAME
	⋮	⋮	⋮	⋮
	2,2	DEC	SCT	GR

جدولی که ستون‌ها را مدیریت می‌کند.



آیا برنامه ساز می تواند محتوای کاتالوگ را مستقیماً تغییر دهد؟ (با دستورات INSERT, DELETE, UPDATE)



تمرین: حداقل سه جدول دیگر برای کاتالوگ طراحی کنید.

تمرین: چه اطلاعاتی در کاتالوگ ذخیره می شود؟



❑ عملیات در TDB : دستورهای DML

SELECT

بازیابی

INSERT

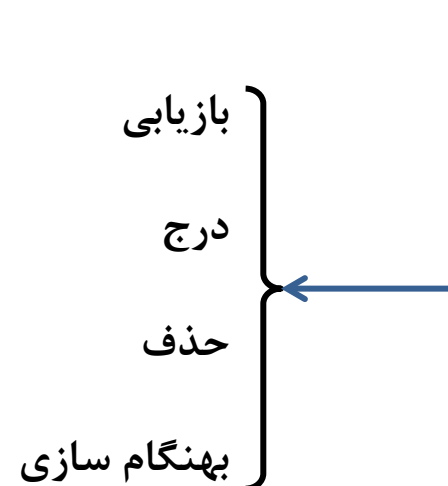
درج

DELETE

حذف

UPDATE

بهنگام سازی





## دستور بازیابی SELECT ☐

```
SELECT [ALL | DISTINCT ] item(s) list  
FROM table(s) expression  
[WHERE condition(s)]  
[ORDER BY Col(s)]  
[GROUP BY Col(s)]  
[HAVING condition(s)]
```

☐ خروجی دستور SELECT یک جدول است.

☐ از DISTINCT برای حذف سطرهای تکراری در جدول نتیجه استفاده می‌شود.

☐ در شرط WHERE می‌توان از =، <، >، <=، >=، <>، <، >، LIKE، BETWEEN و IN استفاده کرد.



بخش چهارم: مقدمات پیاده‌سازی و SQL

```
SELECT STT.STID AS SN,  
       STT.STNAME AS SName  
FROM   STT  
WHERE  STT.STMJR='phys'  
       AND  
       STT.STLEV='bs'
```



```
SELECT STT1.STID AS SN,  
       STT1.STNAME AS SName  
FROM   STT AS STT1  
WHERE  STT1.STMJR='phys'  
       AND  
       STT1.STLEV='bs'
```







یک کپی از جدول با نام جدید، نام‌گذاری جدول جواب:

(SELECT S.\*

FROM S) AS MyS

■ مرتب شده:

ORDER BY SNAME یا 2

■ پیش فرض صعودی: (Ascending)



شماره ستون

■ نزولی (Descending): باید قید شود.



قابلیت‌های پیشرفته (Advanced features):

SELECT S#, CITY

FROM S

WHERE SNAME

$\left\{ \begin{array}{l} \text{LIKE} \\ \text{NOT LIKE} \end{array} \right\}$

'%N' → با N تمام شود

'M%' → با M شروع شود

'\_ \_ A \_ \_' → دقیقاً ۵ کاراکتر، کاراکتر سوم A



**SELECT P#**

**FROM P**

**WHERE WEIGHT BETWEEN (5,15)**

یا

**WHERE WEIGHT >=5 AND WEIGHT <=15**

BETWEEN



□ شماره قطعاتی را بدهید که وزن آنها بین ۵ و ۱۵ است.



NULL



**SELECT** S#, CITY

**FROM** S

**WHERE** STATUS

$\left\{ \begin{array}{l} \text{IS NULL} \\ \text{IS NOT NULL} \end{array} \right\}$

بررسی برخورد یک package با NULL؟





*tablename1* **op** *tablename2* [CORRESPONDING [BY {*column*, [, *column* ...}]]]

$$op \in \left\{ \begin{array}{l} \text{UNION [ALL]} \\ \text{INTERSECT [ALL]} \\ \text{EXCEPT [ALL]} \end{array} \right\}$$

☐ اگر از گزینه CORRESPONDING BY استفاده شود، عمل درخواست شده روی ستون‌های تصریح شده انجام می‌شود.

☐ اگر CORRESPONDING بدون BY استفاده شود، عمل درخواست شده روی ستون‌های مشترک انجام می‌شود.

☐ اگر از این گزینه استفاده نشود، عمل روی تمام ستون‌های دو جدول انجام می‌شود.

☐ شرط استفاده: برابری Heading: هم‌نامی و هم نوعی ستون (های) دو جدول

☐ **توجه:** تکراری‌ها در نتیجه اجرای عملگرهای جبر مجموعه‌ها حذف می‌شوند مگر آنکه از ALL استفاده شود.



```
SELECT S.S#,  
FROM S  
INTERSECT
```

شماره تهیه کنندگانی را بدهید که حداقل یک قطعه تولید می‌کنند.



```
SELECT SP.S#,  
FROM SP
```

```
SELECT SP.S#,  
FROM SP  
EXCEPT
```

تست سازگاری پایگاه داده‌ها: هر فردی که قطعه ای تولید کرده

باید یکی از افراد ثبت شده در جدول تولیدکنندگان باشد.



```
SELECT S.S#,  
FROM S
```

مدل دیگر



SP *EXCEPT* S Using S# یا Corresponding by S#





شماره تهیه کنندگانی را بدهید که هیچ قطعه‌ای تولید نمی‌کنند.



```
SELECT S.S#,  
FROM S  
EXCEPT  
SELECT SP.S#,  
FROM SP
```

تمرین: این مثال‌ها به طرز دیگر هم نوشته شود. □



## Aggregation Functions ☐

AVG ☐ ← میانگین

MIN ☐ ← مینیمم

MAX ☐ ← ماکزیمم

SUM ☐ ← جمع

COUNT(\*) / COUNT ☐ ← تعداد عبارات ناهیچمقدار / تعداد کل سطرها

بیشینه وضعیت تهیه کنندگان در شهرهای c1 یا c2



```
SELECT MAX ( STATUS ) AS SMAX
FROM S
WHERE CITY='c1'
OR
CITY='c2'
```



تعداد انواع قطعات تولیدی توسط تولیدکنندگان



```
SELECT COUNT (DISTINCT P#) AS N1  
FROM SP
```

تعداد انواع قطعات قابل تولید



```
SELECT COUNT (*) AS N2  
FROM P
```

تعداد کل قطعات تولیدی توسط s2



```
SELECT SUM (QTY) AS N3  
FROM SP  
WHERE S# = 's2'
```

NULL و توابع جمعی؟ (در سه پکیج بررسی شود)







### GROUP BY

□ سطرهای جدول داده شده در کلاز FROM را گروه بندی می کند، به نحوی که مقدار ستون(های) گروه بندی در گروه یکسان است.

تعداد کل قطعات تولیدی توسط هر تولیدکننده



```
SELECT S# AS SN ,SUM (QTY) AS SQ  
FROM SP  
GROUP BY S#
```

SP  
گروه بندی  
شده

S#	P#	QTY
s1	p1	...
s1	p2	...
s1	p4	...
s2	p2	...
s2	p3	...
s3	p5	...
...	...	...




جدول جواب

SN	SQ
s1	280
s2	100
s3	203
...	...



در کلاز SELECT نمی توان نام ستونی را آورد که در کلاز GROUP BY نباشد، غیر از ستون‌هایی که با توابع جمعی به دست آمده‌اند.

### HAVING

 امکانی است برای دادن شرط یا شرایط ناظر به گروه سطرها



شماره تهیه‌کنندگانی را بدهید که بیش از ۱۰۰ قطعه تولید کرده‌اند.

**SELECT S#**

**FROM SP**

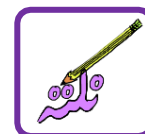
**GROUP BY S#**

**HAVING SUM(QTY) > 100**



تمرین : شماره دانشجویانی را بدهید که در ترم دوم سال ۸۷-۸۸ بیش از ۲۰ واحد گرفته باشند.

تمرین : شماره دانشجویانی را بدهید که در ترم دوم سال ۸۷-۸۸ بیش از ۷ درس گرفته باشند.



GROUP BY و HAVING در SQL افزونه‌اند، اما نوشتن QUERY بدون آنها پیچیده است.

HAVING بدون GROUP BY؟



به چند روش می‌توان یک کپی از جدول ساخت؟





### روش اول

```
SELECT SNAME
FROM S, SP
WHERE SP.S# = S.S# AND SP.P# = 'p2'
```

شبیه سازی عملگر پیوند

نام تهیه کنندگان قطعه 'p2' را بدهید:

در جدول SP

در جدول S



```
SELECT T1.*, T2.*
FROM T1, T2
```

ضرب دکارتی در SQL



مکانیزم اجرا از دید برنامه‌ساز: □

- به ازای هر سطر جدول S، بررسی می‌کند که آیا S# آن در SP وجود دارد یا نه و P# آن سطر در SP، p2 است یا نه. اگر درست بود SNAME آن سطر جزو جواب است.



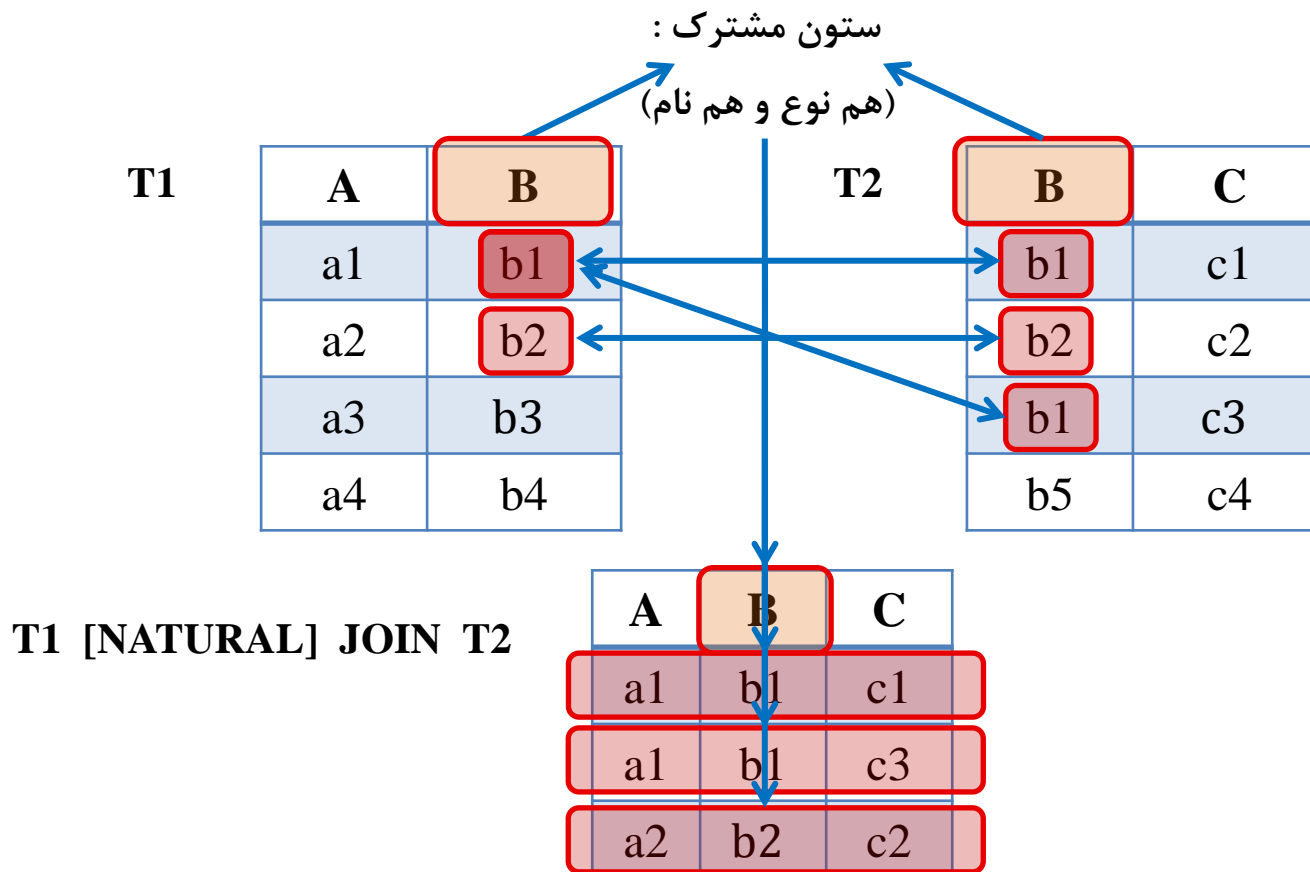
# بازیابی از بیش از یک جدول – عملگر پیوند یا JOIN

بخش چهارم: مقدمات پیاده‌سازی و SQL

۲۹

پیوند: ارائه مقدماتی (غیر ریاضی) □

T1 [NATURAL] JOIN T2 □





# بازیابی از بیش از یک جدول – عملگر پیوند یا JOIN (ادامه)

بخش چهارم: مقدمات پیاده‌سازی و SQL

۳۰

□ توضیح مقدماتی عملگر پیوند:

□ صرف نظر از جزئیات تئوریک، سطرهای دو جدول را که مقدار ستون(های) مشترکشان یکسان است،

به هم پیوند می‌زند.

روش دوم

```
SELECT SNAME
FROM S [NATURAL] JOIN SP
WHERE P# = 'p2'
```

نام تهیه کنندگان قطعه 'p2' را بدهید:



S

S#	SNAME	...
s1	sn1	...
s2	sn2	...
s3	sn3	...
s3	sn4	...
...	...	...

SP

S#	P#	QTY
s1	p1	100
s1	p2	120
s1	p3	500
s2	p1	50
...	...	...

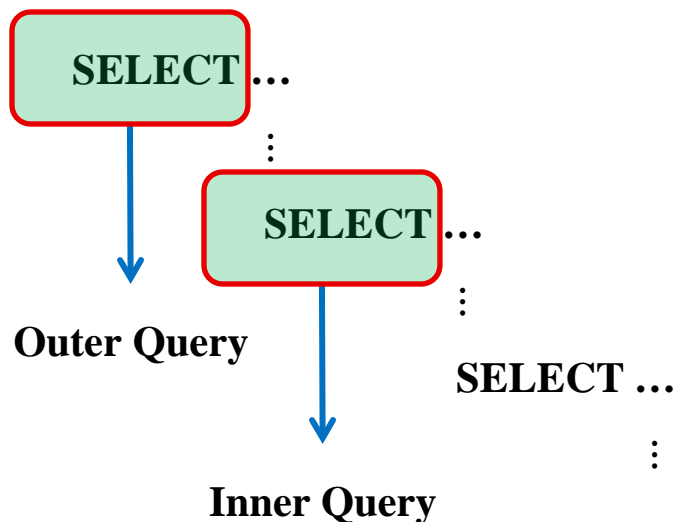
S [NATURAL] JOIN SP

S#	SNAME	...	P#	QTY
s1	sn1	...	p1	100
s1	sn1	...	p2	120
s1	sn1	...	p3	500
s2	sn2	...	p1	50
...	...	...		



## زیر پرسش یا SubQuery

یک SELECT است در درون SELECT دیگر. ← پرسش تو در تو





□ IN و NOT IN: عملگر تعلق



روش سوم

SELECT SNAME

FROM S

WHERE S# IN (SELECT S# FROM SP

WHERE P# = 'p2')

روش چهارم

یا  
= ANY

روش پنجم

یا  
= SOME

عملگر تعلق

□ مکانیزم اجرا:

- سیستم ابتدا SELECT درونی را اجرا می‌کند، آنگاه به ازای هر سطر S بررسی می‌کند که S# در مجموعه جواب SELECT درونی هست یا نه.







# بازیابی از بیش از یک جدول – پرسش های بهم بسته

بخش چهارم: مقدمات پیاده سازی و SQL

۳۳

 دو پرسش درونی و بیرونی (در یک پرسش تو در تو) را **بهم بسته (Correlated)** گوئیم هرگاه در کلاز WHERE پرسش درونی به ستونی از جدول موجود در کلاز FROM پرسش بیرونی، ارجاع داشته باشیم.

 **توجه:** نحوه اجرای پرسش های بهم بسته با طرز اجرای پرسش های نابهم بسته متفاوت است: در حالت بهم بسته، سیستم پرسش درونی را به ازای هر سطر از جدول پرسش بیرونی یک بار اجرا می کند.

روش ششم

SELECT SNAME

FROM S

WHERE 'p2' IN ( SELECT P# FROM SP  
WHERE SP.S# = S.S# )

روش هفتم

یا

= ANY

روش هشتم

یا

= SOME

CORRELATED یا زیرپرسش بهم بسته





$$\text{theta} \in \left\{ \begin{array}{c} = \\ \neq \\ < \\ \leq \\ \geq \\ > \end{array} \right\} \quad \left\{ \begin{array}{ll} \text{theta} & \text{ANY} \\ \text{theta} & \text{SOME} \\ \text{theta} & \text{ALL} \end{array} \right\} \quad \square \text{ امکان}$$

شماره تهیه کنندگانی را بدهید که مقدار وضعیت آنها بیشینه نباشد.



1- SELECT S#

FROM S

WHERE STATUS < ANY ( SELECT DISTINCT STATUS FROM S )

2- SELECT S#

FROM S

WHERE STATUS < ( SELECT MAX (STATUS) FROM S )

چون جواب SELECT تک مقداری است نیازی به ANY نیست.



روش نهم



```
SELECT SNAME
FROM S
WHERE 0 < ( SELECT COUNT(*)
              FROM SP
              WHERE SP.S# = S.S#
              AND
              SP.P# = 'p2' )
```



## NOT EXISTS و EXISTS ☐

☐ امکان بررسی وجود یا عدم وجود سطر در جدول بازگشتی

روش دهم

```
SELECT SNAME
FROM S
WHERE EXISTS ( SELECT *
                FROM SP
                WHERE SP.S# = S.S#
                AND
                SP.P# = 'p2' )
```



روش‌های دیگر؟





### دستورهای INSERT, UPDATE, DELETE ☐

#### درج INSERT: ☐

**INSERT INTO** *table-name* [(*col1*,*col2*, ...)]  
**VALUES** ( *one row* ) | *subquery*

#### بهنگام سازی UPDATE: ☐

**UPDATE** *table-name*  
**SET** *col = value / expression* [, *col = value / expression* ]...  
⋮  
**WHERE** *condition(s) / subquery*

#### حذف DELETE: ☐

**DELETE FROM** *table-name*  
**WHERE** *condition(s) / subquery*



درج سطری (سطر کامل – سطر ناقص):



```
INSERT INTO STT  
VALUES ( '222' , 'st2' , 'IT' , 'bs' , 'D17' )
```

```
INSERT INTO STT  
VALUES ( '333' , 'st3' , Null , 'ms' , Null )
```

درج گروهی:



```
CREATE TEMPORARY TABLE T1  
( STN, .... )
```

اطلاعات دانشجویان مقطع کارشناسی ارشد

```
INSERT INTO T1  
( SELECT STT.*
```

رشته کامپیوتر در جدول موقت T1 درج شود.

```
FROM STT
```

```
WHERE STJ = 'comp'
```

```
AND
```

```
STL = 'ms' )
```



بهنگام سازی چند سطر:



تعداد واحد تمام درس های عملی گروه آموزشی D11 را برابر یک کن.

```
UPDATE COT
SET CREDIT = '1'
WHERE COTYPE = 'p' AND CODEID = 'D11'
```

بهنگام سازی در بیش از یک جدول:



```
UPDATE STT
SET STID = 88104444
WHERE STID = 88107777

UPDATE STCOT
SET STID = 88104444
WHERE STID = 88107777
```

اگر دستور دوم اجرا نشود؟





نمره دانشجویان گروه آموزشی D111 در درس 'com222' در ترم دوم سال ۸۵-۸۶ را ناتمام

اعلان کن.



```
UPDATE STCOT
```

```
SET STCOT.GRADE = 'U'
```

```
WHERE STCOT.TR = '2' AND STCOT.YRYR = '85-86'
```

```
AND STCOT.COID = 'COM222'
```

```
AND STID IN (SELECT STID
```

```
FROM STT
```

```
WHERE STT.STDEID = 'D111');
```





حذف تکدرس: درس com111 را برای دانشجوی 88104444 حذف کنید.



```
DELETE FROM STOCOT
```

```
WHERE STID = 88104444
```

```
AND
```

```
COID = 'COM111'
```

آیا این حذف باید انتشار یابد؟



حذف از بیش از یک جدول:



```
DELETE FROM DEPT
```

```
WHERE DEID = 'D333'
```

```
UPDATE STT
```

```
SET DEID = 'Null'
```

```
WHERE DEID = 'D333'
```



☐ مطالعه شود :

☐ پرسش بازگشتی (Recursive)

☐ SQL ادغام شده

☐ SQL پویا

☐ نوشتن رویه

☐ نوشتن تابع

☐ امکانات شیء- رابطه‌ای

☐ مدیریت تراکنش



**پرسش و پاسخ ...**

**amini@sharif.edu**