

# گفتار سیزدهم

دیدهای رابطه‌ای



## دیدهای رابطه‌ای

هدف اصلی دید تامین سطح خارجی پایگاه داده است

دید در مدل رابطه‌ای نوعی رابطه است، بنابراین می‌توان آن را به کمک یک عبارت جبر رابطه‌ای یا حساب رابطه‌ای به شکل زیر تعریف کرد:

**View name = Relational expression**

(یک عبارت معتبر در جبر رابطه‌ای یا حساب رابطه‌ای)

## دستور ایجاد جدول مجازی (دید خارجی):

می دانیم دید، عبارتی نامدار از جبر رابطه ای است. در SQL برای تعریف دید از دستور زیر استفاده می شود.

```
CREATE VIEW VIEW_NAME [ ( COLUMN[,COLUMN, ....]....).]
AS SUB QUERY
```

تمام قدرت دستور Select در خدمت view است و در واقع مکانیزم اشتقاق view یک پرس و جو است که از طریق Select داده می شود. پرس و جوی مربوطه در زمان تعریف view اجرا نمی شود بلکه فقط به عنوان تعریف با شمای خارجی در کاتالوگ نگهداری می شود تا هرگاه لازم باشد سیستم به آن مراجعه کند.

مثال:

```
CREATE VIEW PARTS (P#,PNAME,WT,CITY)
AS SELECT P#,PNAME, WEIGHT,CITY
FROM P
WHERE COLOR = 'RED'
```

با حکم بالا یک دید بنام Parts تعریف می شود.

## دستور حذف دید:

```
DROP VIEW view-name option [CASCADE | RESTRICT]
```



# دید در SQL

مثال :

```
CREATE VIEW PARTS (P#,PNAME,WT,CITY)
AS SELECT P#,PNAME, WEIGHT,CITY
FROM P
WHERE COLOR = 'RED'
```

با حکم بالا یک دید بنام Parts تعریف می شود .

```
CREATE VIEW MAPHSTUD( STNUM, STLEV, STAREA)
AS SELECT STID, STDEG, STMJR
FROM STT
WHERE STMJR='Math' OR STMJR='Phys'
WITH LOCAL CHECK OPTION;
```



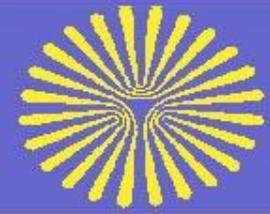
# دید در SQL

دستور حذف دید

**DROP VIEW** Viewname {restrict | cascade}

مثال:

**DROP VIEW MAPHSTUD CASCADE;**



## مزایای دید

۱. تامین کننده محیط انتزاعی برای کاربران سطح خارجی
۲. تامین کننده پویایی بالا در تعریف پایگاه توسط کاربر
۳. تسهیل کننده واسط کاربر برنامه‌ساز با پایگاه
۴. امکانی است برای کوتاه‌نویسی یا ماکرونویسی پرسشها
۵. تامین کننده اشتراک داده‌ای
۶. تامین کننده نوعی مکانیسم خودکار ایمنی داده‌ها
۷. تامین کننده استقلال داده‌ای فیزیکی و منطقی
۸. امکان تعریف شیء با اندازه‌های مختلف



# معایب دید

۱. ایجاد فزونکاری در سیستم برای انجام تبدیل خارجی/ادراکی و احیانا خارجی/خارجی
۲. عدم امکان انجام عملیات ذخیره‌سازی در بسیاری از گونه‌های دید و در نتیجه ایجاد محدودیت ساختاری برای کاربر

## موارد عدم استفاده:

- سیستم پایگاهی تک کاربره
- برای افزایش سرعت اجرای برنامه‌ها یک روش نوشتن برنامه روی رابطه است و نه دید

## ۶ - ۴ - ۱ : عملیات در دید

### الف) بازیابی :

عمل بازیابی از نظر تئوری مشکلی ندارد هر چند در بعضی سیستم ها محدودیت هایی در این زمینه وجود دارد چون view ماهیتا جدول است . لذا همان حکم Select نیز برای آن عمل می کند .

مثال :

```
SELECT * FROM PARTS  
WHERE P# = 'P2'
```

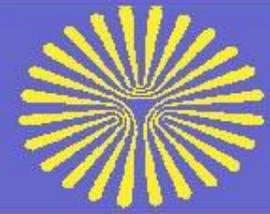
برای اجرای حکم بالا بایستی سیستم آن را به حکمی در سطح ادراکی تبدیل کند و برای این منظور شرط یا شرایط داده شده در تعریف دید را با شرط در حکم بازیابی ترکیب می کند .

مثال :

```
SELECT *  
FROM PARTS
```

دید کاربر را به عینیت درمی آورد .

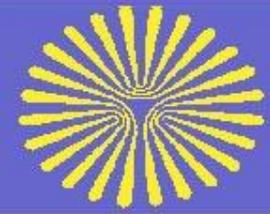
چون VIEW خود یک جدول است و لذا می توان روی آن VIEW تعریف کرد و بدین ترتیب سطوح دیگری از انتزاع را ایجاد کرد .



## عملیات در دیدهای رابطه‌ای

### بازیابی از دید رابطه‌ای

چون دید خود نوعی رابطه است، پس برای بازیابی از دید هم یک عبارت جبری یا حسابی می‌نویسیم



# عملیات در دیدهای رابطه‌ای

مثال بازیابی از یک دید با استفاده از دستورات SQL

```
CREATE VIEW V1  
AS SELECT STID, STDEG  
FROM STT  
WHERE STPROG='Math'
```

```
SELECT STID  
FROM V1  
WHERE STDEG='bs';
```

□ در برخی سیستم ها در عمل بازیابی از VIEW محدودیت هایی وجود دارد که از جمله می توان مشکل تابع جمعی را مطرح کرد.

```
CREATE VIEW PQ
```

```
AS SELECT SP.P#, SUM(SP.QTY) AS TOTALQTY  
FROM SP  
GROUP BY SP.P#
```

دستور زیر غیرمجاز است .

```
SELECT AVG(PQ.TOTALQTY) AS PT  
FROM PQ
```



# عملیات در دیدهای رابطه‌ای

## عملیات ذخیره‌سازی در دیدهای رابطه‌ای

برای انجام عملیات ذخیره‌سازی در دید از همان دستورات سه‌گانه **INSERT**، **UPDATE** و **DELETE** استفاده می‌شود.

## ب) عملیات ذخیره سازی در VIEW : ( بروز در آوری دیدها )

تمام دیدهای قابل تعریف در SQL قابل به هنگام سازی نیستند . به بیان دیگر دیدهایی وجود دارند که نمی توان از طریق آنها عمل درج ، تغییر و حذف را انجام داد . دیدها را معمولا به دو دسته تقسیم می کنند .

۱. دیدهای فاقد مشکل در عملیات به هنگام سازی (پذیرا)

۲. دیدهای دارای مشکل (ناپذیرا)

آنچه که در سیستم های موجود به عنوان به هنگام سازی دیدها انجام می شود در بعضی موارد از نظر منطقی قابل دفاع نیست و یا برعکس از نظر منطقی شدنی است ولی سیستم های موجود انجام نمی دهند . یکی از ایرادهایی که به سیستم های رابطه ای می گیرند نیز بحث به هنگام سازی دید است .

دیدنی قابل به هنگام سازی است اگر روی یک جدول مبنا تعریف شده باشد و هر سطر دید متناظر با سطر شخصی از جدول مبنا باشد و هر ستون دید نیز متناظر با ستون مشخص و نامداری از جدول مبنا باشد .  
مثال (۱) :

```
CREATE VIEW SUPC2
AS SELECT S#,SNAME
FROM S
WHERE CITY = 'C2'
```

فرض کنیم دستور را داشته باشیم :

```
UPDATE SUPC2
Set Sname = '****'
Where S# = 'S2'
```

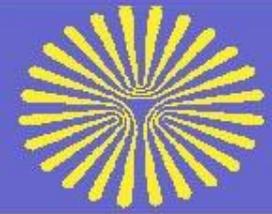
این دستور بایستی به دستوری در سطح ادراکی نگاشته شود .

```
UPDATE S
SET SNAME = '****'
WHERE S# = 'S2'
AND CITY = 'C2'
```

در صورتی که بخواهیم سطری به دید درج کنیم :

```
INSERT INTO SUPC2
VALUES(S12,SN12)
```

کمترین مشکل آن است که در دو ستون city، status پدیده هیچمقدار بروز می کند . صرفنظر از اینکه وجود این پدیده مطلوب نیست ، بروز آن می تواند یکی از قواعد جامعیت پایگاه را خدشه دار کند .



# دیدها از نظر پذیرش عملیات ذخیره‌سازی

۱. دیدهای پذیرا

۲. دیدهای ناپذیرا



۱. دیدهای گزینشی

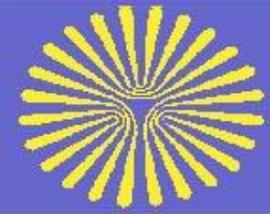
۲. دید گزینش - پرتوی دارای کلید رابطه مبنا

۳. دید پیوندی CK-CK

۴. دید پیوندی CK-FK

۵. دید حاصل اجتماع، اشتراک و تفاضل دو رابطه

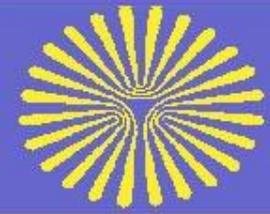
دیدهای پذیرا



دید **گزینشی** حاصل عملگر گزینش در  
یک رابطه است.



در دید “گزینش-پرتوی دارای کلید رابطه مبنا”  
علاوه بر گزینش تاپلهایی از رابطه مبنا، عملگر  
پرتو نیز اعمال شده است. این نوع دید را  
اصطلاحاً دید دارای کلید با تاپلهای ناقص  
می گوئیم.



دید پیوندی **CK-CK**، حاصل پیوند دو رابطه  
روی کلید کاندید مشترک آنها است و در  
عملیات ذخیره‌سازی مشکلی ندارد

داشته باشیم :

SX(S#,Sname,city)  
SY(S#,Status)

و S# در هر دو رابطه SY,SX کلید است .

$SX \infty SY = S$

اگر دیدی داشته باشیم بفرم مقابل :

```
CREATE VIEW S
AS SELECT SX.S#,SNAME,STATUS,CITY
FROM SX,SY
WHERE SX.S# = SY.S#
```

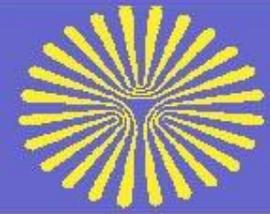
در دید بالا یک پیوند داریم و پیوند از نوع PK -PK است . ( صفت خاصه دخیل در پیوند در هر دو رابطه کلید اصلی است ) دیده‌های حاصل چنین پیوندهایی مشکلی در عملیات به هنگام سازی ندارند . وجود پیوندهای PK -P ک سبب شده است که تناظر یک یک بین سطرها و ستونهای جداول مبنای زیرین وجود داشته باشد .

```
Insert Into S (S#,Sname,Status,city)
Values (S9,Sn9,20,C9)
```



```
Insert Into SX (S#,Sname,city)
Values (S9,Sn9,C9)
```

```
Insert Into SY (S#, Status)
Values (S9,20)
```



**دید پیوندی CK-FK**، حاصل پیوند روی کلید کاندید یک رابطه و کلید خارجی رابطه دیگر است. این دید در حذف مشکل دارد، زیرا با حذف یک تاپل از این دید، در هریک از دو رابطه مبنا، یک تاپل حذف می‌شود و چنانچه کاربر بخواهد محتوای دید خود را نمایش دهد، تاپلهای دیگر هم از دید او حذف می‌شوند که درخواست نکرده است.



دید حاصل اجتماع، اشتراک و تفاضل دو رابطه  
در عملیات ذخیره‌سازی مشکلی ندارد، به شرط  
آنکه سیستم بتواند تشخیص دهد که عمل  
درخواست‌شده، در کدامیک از دو رابطه مبنا  
انجام شود.

مثال (۳) : دید آماری :

```
Create view V3(Pnum,SumQ)
AS Select P#,sum(QTY)
From SP
Group by P#;
```

این دید حاوی یک فیلد مجازی است ( به عینیت درآوردن غیر مستقیم ) : S umQ یک فیلد مجازی است و به عینه روی جدول فیلد متناظر ندارد و لذا هیچ گونه عملی روی این ستون نمی توان انجام داد .

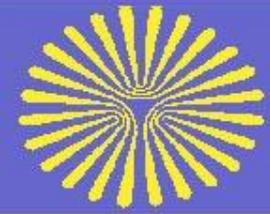
```
INSERT INTO V3 (Pnum,SumQ)
Values < P11 , 111>
```

مقداری است برای sum(Qty) و نه برای Qty . لذا عمل درج امکان پذیر نیست .



## مشکلات مهمتر دیدهای پذیرا

- بروز عارضه جانبی در خود دید
- بروز عارضه جانبی در دیدهای دیگر
- نقض قاعده جامعیت
- بروز فزونکاری در سیستم
- تغییر ماهیت عمل درخواست شده
- تعدد تبدیلات و مشکل تصمیم گیری



۱. دید پرتوی یا گزینشی فاقد کلید رابطه مبنا

۲. دید پیوندی NK-NK

۳. دید پیوندی FK-FK

۴. دید حاوی صفت مجازی

۵. دید حاصل تقسیم

دیدهای  
ناپذیرا



# ویژگیهای دیدهای قابل بهنگام سازی

۱. عبارت تعریف کننده محدوده دید، یک عبارت معتبر SELECT باشد.
۲. در کلاز FROM، عبارت SELECT، فقط یک جدول وجود داشته باشد.
۳. جدول قیدشده در کلاز FROM، یک جدول مبنا یا یک دید قابل بهنگام سازی باشد.
۴. در item-list عبارت SELECT، ستونهای مورد نظر باید در جدول مبنا متناظر باشد.
۵. در عبارت SELECT نباید کلاز GROUP BY و HAVING و گزینه DISTINCT وجود داشته باشد.
۶. در کلاز WHERE عبارت SELECT نباید عبارتی حاوی کلاز FROM باشد، به گونه ای که در آن به همان جدولی ارجاع داده شده باشد که در کلاز FROM قبلی به آن ارجاع شده است.

## نکات مهم :

۱. قابلیت به هنگام سازی در view به گونه ای است که یا هر سه عمل INSERT و UPDATE و DELETE می توانند بر روی یک دید اعمال شوند و یا هیچ کدام را نمی توان اعمال کرد .
۲. در view این امکان وجود ندارد که بعضی ستونها را به هنگام سازی کرد و برخی ستونها را در داخل همان دید به هنگام سازی نکرد .

مثال (۳) : فرض کنید دید V5 بصورت زیر تعریف شده باشد :

```
CREATE VIEW V5
AS Select S#,Status,city
From S
Where Status > 15
With check option;
```

S	V5
<u>S#.....Status</u>	<u>S#.....Status</u>
S1.....16	S1.....16
S2.....10	S3.....17
S3.....17	
S4.....15	

تهیه کنند S2 یا S4 در این دید وجود ندارد . آیا کاربر حق دارد عمل زیر را انجام دهد ؟

```
Insert Into V5
Values (S2,18)
```

در این صورت بایستی جلوی عمل درج را بگیرد زیرا باعث تکرار در کلید می شود. و نیز آیا کاربر حق دارد دستور مقابل را وارد کند؟

```
update      V5  
  Set      Status = 10  
  Where    S# = 'S3'
```

در چنین مواردی در بعضی سیستم ها گزینه `check option` را قرار می دهند. به این معنی که اگر عملیات درج و به هنگام سازی جامعیت اعمال شده توسط عبارت تعریف کننده دید را نقض کنند آنگاه این عملیات روی دید رد می شوند.