

به نام آنکه جان را فکرت آموخت

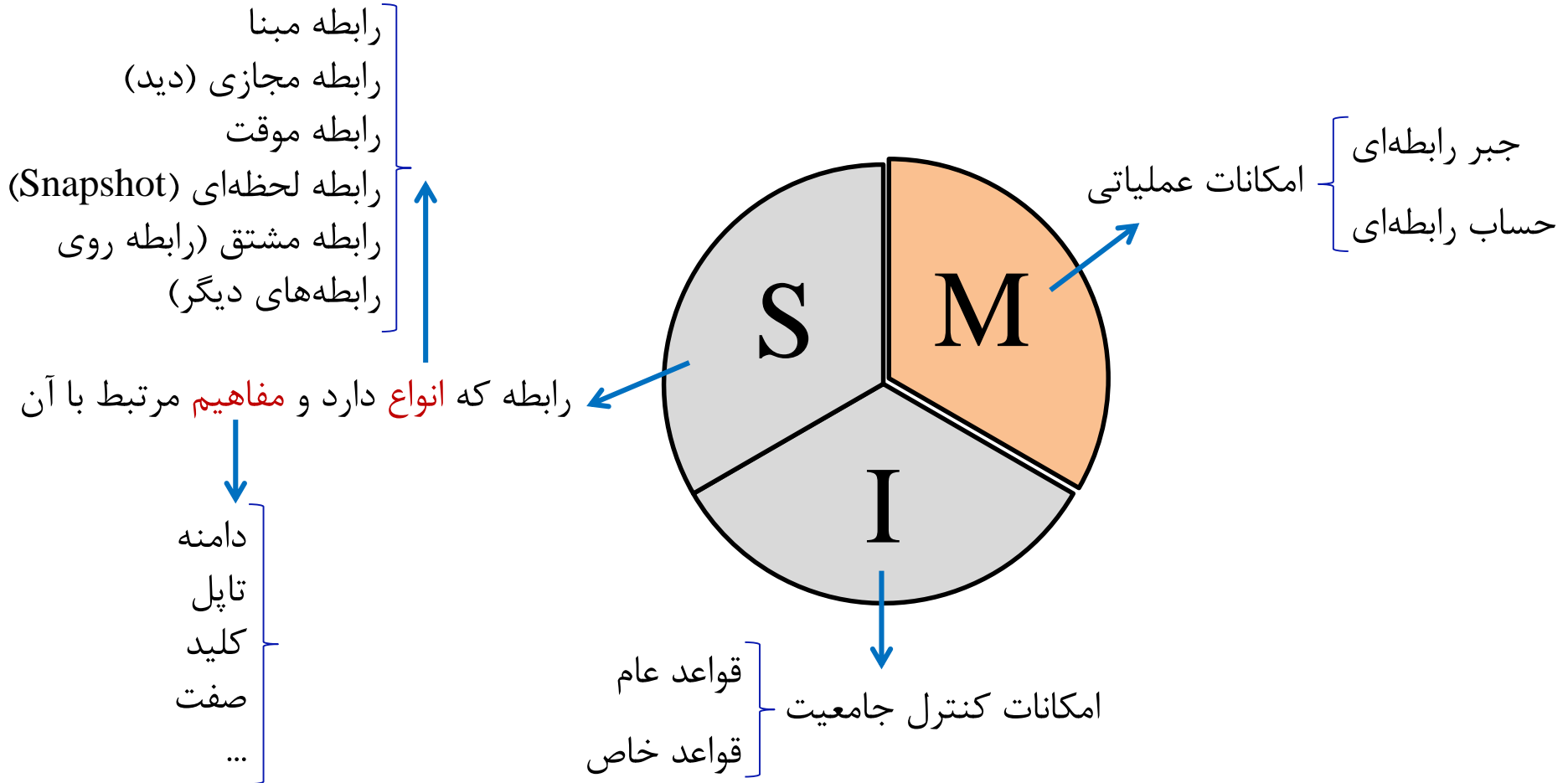


## بخش هفتم: عملیات در پایگاه داده رابطه‌ای

دکتر عیسی زارع پور

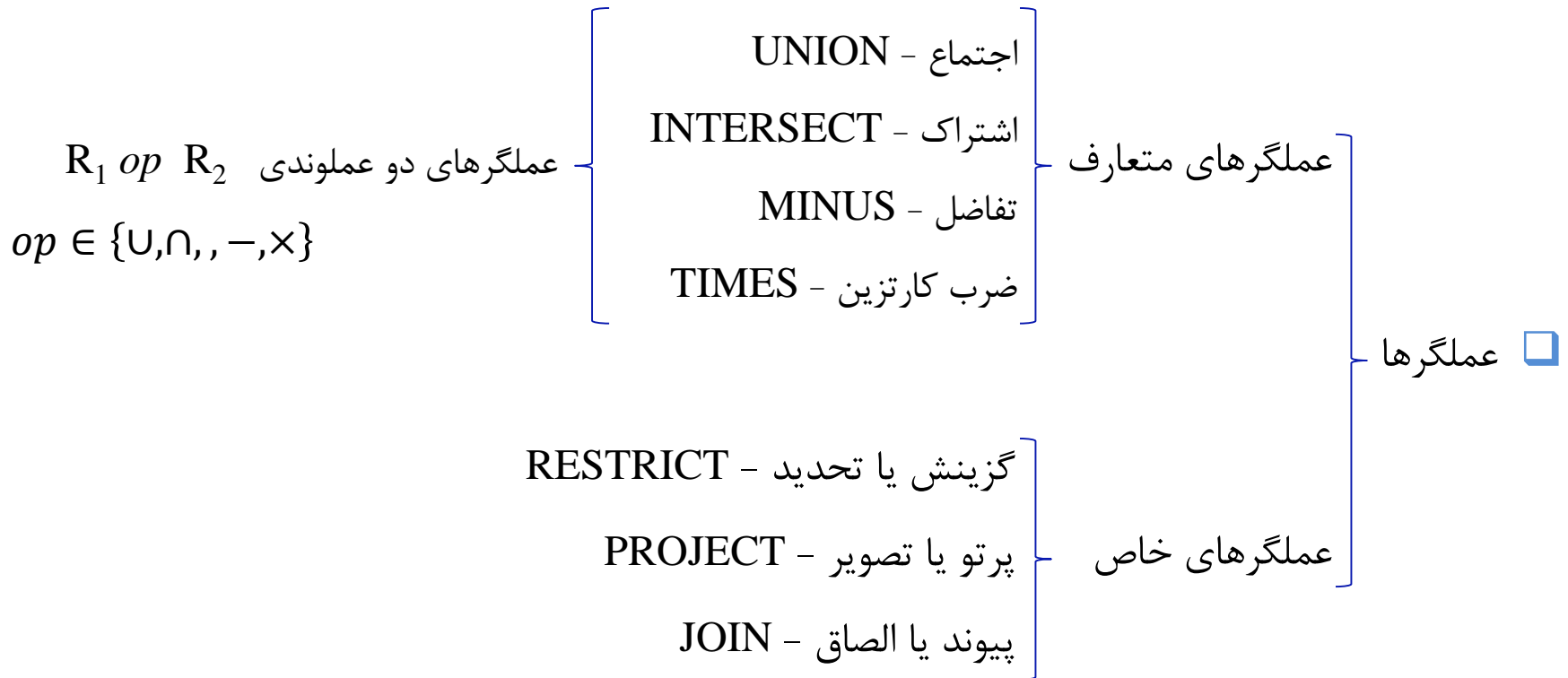
دانشکده مهندسی کامپیوتر، دانشگاه علم و صنعت

محتویات اسلایدها برگرفته از یادداشتهای کلاسی استاد محمدتقی روحانی رانکوهی است. اسلایدها توسط آقای دکتر مرتضی امینی (دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف) تهیه شده است.





## بخش هفتم: عملیات در پایگاه داده رابطه‌ای





□ **خاصیت بسته بودن:** حاصل ارزیابی هر عبارت جبر رابطه‌ای معتبر، باز هم یک رابطه است (که تاپل تکراری ندارد).

□ برای سه عملگر  $\cup$ ،  $\cap$  و  $-$ ، باید عملوندها نوع-سازگار (Type Compatible) باشند:

□ **پیش شرط:**  $H_{R_1} = H_{R_2}$

□  $R_3 = R_1 \text{ op } R_2 \longrightarrow H_{R_3} = H_{R_1} = H_{R_2} \quad \text{op} \in \{\cup, \cap, -, \dots\}$

□ بدنه نتیجه، حاصل انجام هر یک از اعمال اجتماع، اشتراک و یا تفاضل دو مجموعه بدنه است.

□ در عملگر ضرب کارتزین (TIMES):

□ **شرط:** در عنوان دو رابطه نباید صفت هم‌نام وجود داشته باشد.  $H_{R_2} \cap H_{R_1} = \emptyset$

□ عنوان رابطه نتیجه برابر است با  $H_{R_2} \cup H_{R_1}$  و بدنه نتیجه برابر ضرب کارتزین دو مجموعه بدنه است.

□ در TIMES در SQL چگونه شبیه‌سازی می‌شود؟



## بخش هفتم: عملیات در پایگاه داده رابطه‌ای

یک عبارت بولی تشکیل شده از شرطهای ساده به صورت  $(A_i \text{ theta } A_j)$  یا  $(A_i \text{ theta literal})$  که در آن  $\text{theta}$  یکی از عملگرهای  $=, \neq, <, >, \leq$  و  $\geq$  است و  $\text{literal}$  یک مقدار ثابت است.

### عملگر گزینش یا تحدید - RESTRICT

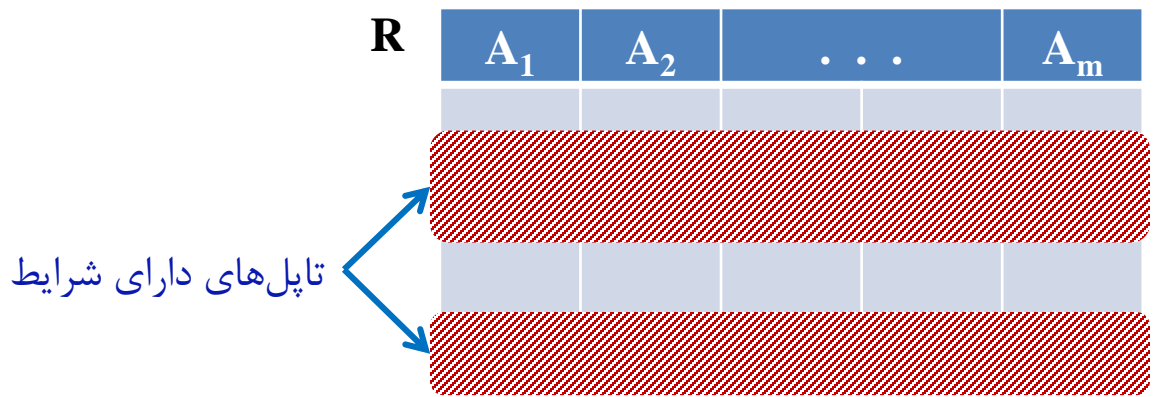
نماد ریاضی:  $\sigma_c$

← شرط یا شرایط گزینش ←

شکل کلی:  $\sigma_c(R)$  یا **RESTRICT R WHERE c** یا **R WHERE c**

تک عملوندی: Monadic

عملکرد (در نمایش جدولی رابطه): زیرمجموعه‌ای افقی می‌دهد. ← عملگر تاپل (ها) یاب





مشخصات کامل دانشجویان رشته فیزیک دوره کارشناسی را بدهید.



$\sigma_{STJ='phys' \wedge STL='bs'}(STT)$

```
SELECT STT.*  
FROM STT  
WHERE STJ='phys' AND STL='bs'
```

وقتی در شرط C (یا کلاز WHERE) بخشی از کلید را با شرط تساوی داده باشیم.

اگر  $R' = \sigma_C(R)$  باشد آنگاه  $CK_{R'} \subseteq CK_R$ .





عملگر گزینش جابجایی پذیر است، یعنی:

$$\sigma_{c_1}(\sigma_{c_2}(R)) = \sigma_{c_2}(\sigma_{c_1}(R)) = \sigma_{c_1 \wedge c_2}(R)$$

عبارتهای جبری معادل:

**R WHERE (C<sub>1</sub> AND C<sub>2</sub>) ≡ (R WHERE C<sub>1</sub>) INTERSECT (R WHERE C<sub>2</sub>)**

**R WHERE (C<sub>1</sub> OR C<sub>2</sub>) ≡ (R WHERE C<sub>1</sub>) UNION (R WHERE C<sub>2</sub>)**

**R WHERE NOT C ≡ R MINUS (R WHERE C)**



## PROJECT پرتو - PROJECT

□ نماد ریاضی:  $\Pi$

□ شکل کلی:  $\Pi_{\langle L \rangle}(R)$  یا  $(R)[L]$  یا **PROJECT R OVER (L)**

لیست صفات پرتو ←

□ تک عملوندی: Monodic

□ عملکرد (در نمایش جدولی رابطه): زیرمجموعه عمودی می‌دهد. ← عملگر ستون(ها) یاب

<b>R</b>	$A_1$	...	$A_i$	...	$A_j$	...	$A_m$





□ عملگر پرتو **تکراری‌ها** را حذف می‌کند. ← چون جواب رابطه است، پس یک مجموعه است و عضو تکراری ندارد.

شماره و رشته تمام دانشجویان را بدهید.



$$\Pi_{\langle \text{STID}, \text{STJ} \rangle}(\text{STT})$$

```
SELECT STID, STJ FROM STT
```

شماره دانشجویانی که درسی انتخاب نکرده‌اند.



$$R := \Pi_{\langle \text{STID} \rangle}(\text{STT}) - \Pi_{\langle \text{STID} \rangle}(\text{STCOT})$$

شماره و مقطع تحصیلی دانشجویان رشته IT را بدهید.




$$\Pi_{\langle \text{STID}, \text{STL} \rangle}(\sigma_{\text{STJ}='IT'}(\text{STT}))$$



اگر  $R' = \Pi_{\langle L \rangle}(R)$  باشد آنگاه:

اگر  $CK_R \subseteq L$  آنگاه  $CK_{R'} = CK_R$ .

اگر نه، در بدترین حالت  $CK_{R'} = L$ .

اگر  $R' = R_1 \text{ op } R_2$  و  $op \in \{ \cup, \cap, -, \times \}$ ، آنگاه  $CK_{R'} = ?$  

SELECT در SQL استاندارد، در حالت کلی ترکیبی از دو عملگر RESTRICT و PROJECT است.

تمرین سر کلاس: نام و رنگ قطعاتی که در شهر تهران تولید می‌شوند را به دست آورید.

P(P#, PName, PColor, PWeight, PCity)



## EXTENDED PROJECT – عملگر پرتو گسترش یافته

□ نماد ریاضی:  $\Pi$

□ شکل کلی:  $\Pi_{\langle F_1, F_2, \dots, F_n \rangle}(\mathbf{R})$

لیست صفات و یا توابع حسابی پرتو ←

□ این عملگر امکان می‌دهد تا در لیست صفات پرتو، از توابع حسابی استفاده شود و صفت (صفاتی) با

مقادیر حاصل از اجرای تابع (توابع) در رابطه جواب داشت.

رابطه‌ای با صفات شماره دانشجو، شماره درس و نمره دانشجو در درس، تغییر یافته با فرمول



$G := 1.2 * \text{GRADE}$  بدهید.

$\Pi_{\langle \text{STID}, \text{COID}, (1.2 * \text{GRADE}) \text{ RENAME AS } G \rangle}(\text{STCOT})$



## عملگر تغییر نام - RENAME

نماد ریاضی:  $\rho$

شکل کلی:  $\rho_R(E)$

← نام رابطه حاصل از عبارت جبر رابطه‌ای E

مثال : شماره و مقطع دانشجویان رشته IT را در یک رابطه جدید با نام STT2 ذخیره کن

$$\rho_{STT2}(\Pi_{\langle STID, STL \rangle}(\sigma_{STJ='IT'}(STT)))$$

این عملگر برای نامیدن رابطه حاصل از یک عبارت جبر رابطه‌ای به کار می‌رود.

عملکرد:  $\rho_R(E)$  رابطه حاصل از عبارت جبر رابطه‌ای E را با نام R برمی‌گرداند.

از عملگر RENAME برای دگرنامی صفت هم می‌توان استفاده کرد (مشابه آنچه در مثال اسلاید قبل آمد). مثلاً با

دستور **RENAME A<sub>i</sub> AS B<sub>j</sub>** R، به صفت A<sub>i</sub> از R، نام دیگر B<sub>j</sub> داده می‌شود.



□ عملگر انتساب یا ذخیره سازی موقت ← یا :=

□ با استفاده از عملگر ← می توان خروجی یک مجموعه دستورات را در یک رابطه جدید ذخیره کرد.

□ مثلاً شماره دانشجویی و مقطع دانشجویان رشته فناوری اطلاعات را استخراج کنید.

$$\text{Temp} \leftarrow \Pi_{\langle \text{STID}, \text{STL} \rangle} (\sigma_{\text{STJ} = 'IT'} (\text{STT}))$$

□ در ادامه دستورات می توانیم ، به جای نوشتن دستورات قبلی ، از نام جدول ذخیره شده استفاده نماییم.

□ شماره دانشجویان مقطع کارشناسی ارشد فناوری اطلاعات

$$\Pi_{\langle \text{STID} \rangle} (\sigma_{\text{STL} = 'MS'} (\text{TEMP}))$$



## عملگر پیوند JOIN (مدل ریاضی عمومی) □

نام عمومی: Theta Join □

نماد ریاضی:  $\bowtie_{Cond(s)}$  □

← شرط پیوند

$$\left\{ \begin{array}{l} R_1 (A_1, A_2, \dots, A_n) \\ R_2 (B_1, B_2, \dots, B_m) \end{array} \right.$$

فرض: دو رابطه  $R_1$  و  $R_2$  نام صفت مشترک ندارند. □

شکل کلی:  $R_1 \bowtie_C R_2$  یا  $R_1 \theta\text{-JOIN}_C R_2$  یا فقط  $R_1 \text{JOIN}_C R_2$  □

$R_1.A_i$  theta  $R_2.B_j$  شرط پیوند (c):

- |                   |   |           |
|-------------------|---|-----------|
| EQUI-JOIN         | = | } Theta □ |
| NOT EQUI-JOIN     | ≠ |           |
| LESS THAN-JOIN    | < |           |
| LESS EQUI-JOIN    | ≤ |           |
| GREATER THAN-JOIN | > |           |
| GREATER EQUI-JOIN | ≥ |           |



$R_1.A_i$  **theta**  $R_2.B_j$

□ شرط پیوند (c):

صفات پیوند

که باید **هم‌دامنه** و **ناهم‌نام** باشند.



چون نتیجه JOIN رابطه است و در headingش صفت تکراری نباید وجود داشته باشد.

□ **نکته:** اگر صفات پیوند هم‌نام باشند، حداقل یکی را باید دگرنامی کرد (به دلیل وجود این راه حل،

حساسیتی در عدم وجود صفت مشترک نداریم).

□ در حالت کلی شرط پیوند می‌تواند به صورت زیر باشد که در آن  $c_1, \dots, c_n$  قالب بالا (قالب شرط

$\langle c_1 \rangle$  AND  $\langle c_2 \rangle$  AND ... AND  $\langle c_n \rangle$

پیوند) را دارند.

$\langle R1.A1 = R2.B1 \rangle$  AND  $\langle R1.A2 = R2.B2 \rangle$





مشخصات کامل جفت تهیه‌کننده-قطعه از یک شهر را بدهید.



$$R_1 := S \bowtie_{S.CITY=P.PCITY} (P \text{ RENAME CITY AS PCITY})$$

**S (S#, SNAME, STATUS, CITY)**

S1	C1
S2	C2
S3	C3
S4	C4
S5	C5
S6	C6

**P (P#, ... , W, CITY)**

P1	5	C1
P2	6	C2
P3	4	C1
P4	7	C4
P5	10	C5

**R<sub>1</sub> (S#, ..., CITY, P#, ... , W, PCITY)**

S1	C1	P1	5	C1
S1	C1	P3	4	C1
S2	C2	P2	6	C2
<del>S3</del>	تا پل پیوندشده ندارد.			
S4	C4	P4	7	C4
S5	C5	P5	10	C5
<del>S6</del>	تا پل پیوندشده ندارد.			





بخش هفتم: عملیات در پایگاه داده رابطه‌ای

$$R_3 = R_1 \bowtie_C R_2 \quad \square \text{ عملکرد:}$$

$$H_{R_3} = H_{R_1} \cup H_{R_2}$$

▪ در بدنه  $R_3$  تاپل‌های پیوندشده از دو رابطه قرار دارند.

□ خصوصیات:

▪  $R_1 \bowtie_C R_2 = R_2 \bowtie_C R_1$  چون صفات در heading رابطه نظم مکانی ندارند.

▪  $R_1 \bowtie_C R_2 = \sigma_C(R_1 \times R_2)$  حاصل Theta-Join در حالت عمومی، زیرمجموعه‌ای افقی از

ضرب کارتیزین است که در آن تاپل‌هایی از حاصلضرب که حائز شرط پیوند هستند حضور دارند.

وقتی در شرط پیوند، تساوی بخشی از کلید هر دو رابطه را داده باشیم.

$$CK_{R'} \subseteq CK_{R_1} \cup CK_{R_2} \quad \text{اگر } R' = R_1 \bowtie_C R_2 \text{ باشد، آنگاه}$$





## پیوند طبیعی (Natural Join)

 گونه‌ای از پیوند است که دو ویژگی دارد:

۱- Theta عملگر مساوی است (=)

۲- صفات پیوند یک بار در جواب می‌آیند. (صفت یا صفات پیوند باید هم‌نام هم باشند).

$$R_2 := S \bowtie_{S.CITY=P.CITY} P$$



$R_2 (S\#, \dots, CITY, P\#, \dots, W)$

S1	C1	P1	5
S1	C1	P3	4
S2	C2	P2	6
S4	C4	P4	7
S5	C5	P5	10



## گونه‌های خاص عملگر پیوند – پیوند طبیعی (ادامه)

بخش هفتم: عملیات در پایگاه داده رابطه‌ای

۱۹

□ اگر صفت مشترک [هم‌نام و هم‌دامنه] یک صفت باشد، نیازی به قید کردن نیست.

اما اگر بیش از یک صفت باشد، باید صفت یا صفات پیوند را قید کنیم.

اگر قید نکنیم، پیوند روی تساوی مقادیر تمام صفات مشترک انجام می‌شود.

$$R_1: (A, B, C)$$

$$R_2: (A, F, C)$$

$$R' = R_1 \bowtie R_2$$

$$R': (A, B, C, F)$$

$$R_1 \bowtie R_2 = R_1 \times R_2 \quad \text{اگر } H_{R_1} \cap H_{R_2} = \emptyset \text{ آنگاه}$$



$$R_1 \bowtie R_2 = R_1 \cap R_2 \quad \text{اگر } H_{R_1} = H_{R_2} \text{ آنگاه}$$



## نیم‌پیوند (Semijoin)

در شکل عمومی با هر  $\Theta$  نوشته می‌شود.

نماد:  $\bowtie_C$  (در چپ تعریف شده)

مدل ریاضی:  $R_3 := R_1 \bowtie_C R_2 = \Pi_{\langle H_{R_1} \rangle}(R_1 \bowtie_C R_2)$

عملکرد:

$$H_{R_3} = H_{R_1} \quad \blacksquare$$

در بدنه  $R_3$ : تاپل‌های پیوند شدنی از رابطه چپ

عملگر نیم پیوند خاصیت جابجایی ندارد:  $R_1 \bowtie_C R_2 \neq R_2 \bowtie_C R_1$



# گونه‌های خاص عملگر پیوند – نیم‌پیوند (ادامه)

بخش هفتم: عملیات در پایگاه داده رابطه‌ای

۲۱

$R_3 := S \bowtie_{S.CITY=P.PCITY} (P \text{ RENAME CITY AS PCITY})$



$R_3 (S\#, \dots, CITY)$

S1	C1
S2	C2
S4	C4
S5	C5

کاربرد این عملگر چیست؟



**تمرین:** عملگر نیم‌پیوند در SQL شبیه‌سازی شود.



## برون پیوند (Outer Join)

Theta هر چیزی می‌تواند باشد.

سه گونه دارد:

$\bowtie_C$  Left O. J. -۱

$\ltimes_C$  Right O. J. -۲

$\bowtie\ltimes_C$  Full O. J. -۳

عملکرد  $R_4 := R_1 \bowtie_C R_2$ :

$$H_{R_4} = H_{R_1} \cup H_{R_2} \quad \blacksquare$$

در بدنه  $R_4$ : تاپل‌های پیوند شدنی از دو رابطه و

تاپل‌های پیوندناشدنی از رابطه چپ گسترش یافته با هیچ مقدار (Null Value)



# گونه‌های خاص عملگر پیوند - برون‌پیوند (ادامه)

بخش هفتم: عملیات در پایگاه داده رابطه‌ای

$$R_4 := S \bowtie P$$



$R_4(S\#, \dots, CITY, P\#, \dots, W)$

S1	C1	P1	5
S1	C1	P3	4
S2	C2	P2	6
S4	C4	P4	7
S5	C5	P5	10
S3	C3	?	?
S6	C6	?	?

کلید  $R_4$  ( $CK_{R_4}$ ) چیست؟ بی تردید کلید اصلی ندارد.



مشکل Outer Join:

۱- از نظر ریاضی رابطه نیست، چون کلید اصلی ندارد.

۲- مصرف حافظه زیاد

این عملگرها در عمل چه کاربردی دارند؟



آیا عملگرهای Outer Join خاصیت جابجایی دارند؟





## نیم تفریق (Semi Minus) □

$$R_1 \text{ SEMIMINUS } R_2 = R_1 \text{ MINUS } (R_1 \text{ SEMIJOIN } R_2) \quad \square$$

عملکرد □

$R_4 (S\#, \dots, CITY, P\#, \dots, W)$

S1	C1	P1	5
S1	C1	P3	4
S2	C2	P2	6
S4	C4	P4	7
S5	C5	P5	10
S3	C3	?	?
S6	C6	?	?

در بدنه  $R_5$ : تاپل‌های پیوند نشدنی از رابطه چپ □

$$H_{R_5} = H_{R_1} \quad \blacksquare$$

S SEMIMINUS P





بخش هفتم: عملیات در پایگاه داده رابطه‌ای

## عملگر تقسیم (Divide) □

□ مفروضند رابطه‌های:

$$\left[ \begin{array}{l} \overbrace{R_1(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)}^{X \quad Y} \\ R_2(B_1, B_2, \dots, B_m) \end{array} \right.$$

□ شرط عمل:

$$R_3(X) := R_1(X, Y) \div R_2(Y) \longrightarrow H_{R_2} \subseteq H_{R_1} \quad \blacksquare$$

□ عملکرد:

$$H_{R_3} = X = H_{R_1} - H_{R_2}^{-1}$$

۲- در بدنه  $R_3$ : بخش  $X$  از تاپلهایی از  $R_1$  که حاوی تمام مقادیر  $Y$  از  $R_2$  باشند.



بخش هفتم: عملیات در پایگاه داده رابطه‌ای

□ کاربرد این عملگر، زمانی است که بخواهیم همه ی حالت های یک اتفاق را بررسی نماییم.

□ ابتدا بخشی را که شامل همه می شود را پیدا می نماییم (مقسوم علیه)، سپس بخش دیگر را بر آن تقسیم می کنیم. با این شرط که حتما می بایست، صفت های مندرج در مقسوم علیه در مقسوم هم وجود داشته باشد، خروجی شامل صفت های باقی مانده خواهد بود.

□ مثال: مشخصات دانشجویانی که همه درسهای ارایه شده توسط استاد شماره ۱۰۰ را اخذ کرده اند:

$$\text{Temp} \leftarrow \Pi_{\langle \text{COID} \rangle} (\sigma_{\text{PID}=100}(\text{STCOT}))$$

$$\text{STCOT} \div \text{Temp}$$

□ ضرب و تقسیم جبر رابطه‌ای لزوماً عکس هم نیستند.

□ **تمرین:** عملگر تقسیم را در SQL شبیه‌سازی کنید.



$$R_1(S\#, P\#) \div R_2(P\#) = R_3(S\#)$$

S1	P1	P1	S1
S1	P2	P2	
S1	P3	P3	
S2	P1		
S2	P2		
S3	P1		

$$R_1(S\#, P\#) \div R_4(P\#) = R_5(S\#)$$

S1	P1	P1	S1
S1	P2	P2	S2
S1	P3		
S2	P1		
S2	P2		
S3	P1		



## عملگر گسترش - EXTEND

صفت یا صفاتی را به عنوان (heading) یک رابطه اضافه می‌کند. حاصل، رابطه دیگری است.

**EXTEND STUD ADD STADDRESS**

**STUD (STID, ..., STD, STADDRESS)**

در SQL با ALTER TABLE پیاده‌سازی شده ولی ALTER ستون(هایی) را به همان جدول اضافه می‌کند.

با این عملگر می‌توانیم یک ستون محاسبه‌شده به رابطه اضافه نماییم.



## عملگر تلخیص – SUMMARIZE

- تاپل‌های رابطه را گروه‌بندی می‌کند به نحوی که مقدار صفت (صفات) گروه‌بندی در هر گروه یکسان باشد؛ معمولاً با یک یا چند تابع جمعی استفاده می‌شود.
- این عملگر در SQL با GROUP BY پیاده‌سازی شده است.

**SUMMARIZE STCOT BY (STID) ADD AVG(GRADE) AS AVER**

- برای این پرسش‌ها، اول عنوان (Heading) رابطه جواب را تعیین می‌کنیم.
- به جای AVG می‌توانیم از توابع جمع و یا گروهی دیگر مانند MIN (حداقل)، MAX (حداکثر)، SUM (جمع) و یا COUNT (شمارشگر تاپل‌ها) استفاده کنیم.



## عملگر GROUP

عملگر GROUP پیشنهاد Date است، برای تبدیل رابطه نرمال به غیرنرمال (در SQL، NEST است).

عکس آن UNGROUP (در SQL، UNNEST) است.

SP GROUP (P#, QTY) AS NNPQTY

NNSP (S#, NNPQTY[P#, QTY])

S1	P1	50
	P2	70
	P3	60
S2	P1	100
	P2	150

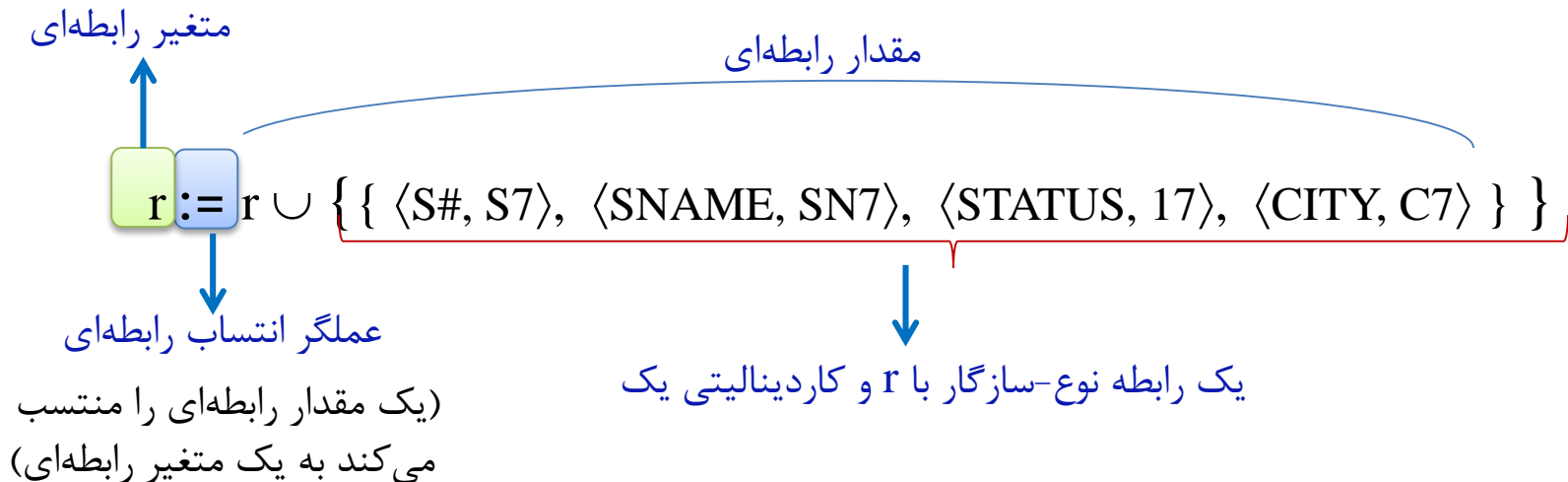
با استفاده از UNGROUP، رابطه نرمال SP را می‌توانیم مجدداً به دست آوریم.

NNSP UNGROUP NNPQTY



از لحاظ تئوریک می‌توان عملیات ذخیره‌سازی را هم با عملگرهای جبر رابطه‌ای انجام داد. □

عمل	عملگر	
درج	U	$r \leftarrow r \cup E$
حذف	-	$r \leftarrow r - E$
به‌هنگام سازی	اول - بعد U	





## مقایسه دو رابطه

دو رابطه  $R_1$  و  $R_2$  مقایسه‌شدنی (قابل قیاس) هستند، هر گاه نوع-سازگار باشند ( $H_{R_2} = H_{R_1}$ )

در مقایسه رابطه  $R_1$  با  $R_2$ ، بدنه  $R_1$  با بدنه  $R_2$  مقایسه می‌شود از نظر هم‌مجموعگی، زیرمجموعگی و زیرمجموعگی

$$\Pi_{\langle \text{STID} \rangle}(\text{STT}) * \Pi_{\langle \text{STID} \rangle}(\text{SCR})$$

$$* \in \{ \subset, \supset, \subseteq, \supseteq, =, \neq \}$$

پاسخ عمل مقایسه: یا  $T$  یا  $F$ . به طور مثال در رابطه فوق:

- اگر  $\supset$  باشد، پاسخ  $T$  است اگر حداقل یک دانشجو باشد که درسی انتخاب نکرده باشد.
- اگر  $\subset$  باشد، پاسخ  $T$  است اگر حداقل در یک عمل ذخیره‌سازی در این DB قاعده جامعیت  $C2$  رعایت نشده باشد (حذف از دانشجو و یا درج در انتخاب درس).





- جبر رابطه‌ای **زبانی** است از نظر رابطه‌ای **کامل** (Relational Completeness) یعنی هر رابطه معتبر متصور از مجموعه رابطه‌های ممکن را می‌توان به کمک یک عبارت جبر رابطه‌ای بیان کرد.
- جبر رابطه‌ای ضابطه تشخیص کامل بودن زبان‌های رابطه‌ای است.
- اگر هر رابطه‌ای را که با جبر رابطه‌ای می‌توان نشان داد، با زبانی مدعی کامل بودن رابطه‌ای بتوان نشان داد، آن زبان از نظر رابطه‌ای **کامل** است.
- کاربردهای جبر رابطه‌ای:
  - عملیات بازیابی
  - عملیات ذخیره‌سازی
  - تعریف انواع رابطه‌های مشتق (رابطه مجازی، لحظه‌ای و ...) مثال: تعریف دید (View) در SQL
  - ....



□ برای نوشتن یک پرسش (Query):

۱- از چه رابطه‌هایی استفاده کنیم.

۲- از چه عملگرهایی استفاده کنیم (حتی‌الامکان با کمترین تعداد عملگر)

۳- چه ترتیبی از عملگرها استفاده کنیم.

□ روش‌های اجرای عملگر Join در DBMS کدامند؟



حساب رابطه‌ای شاخه‌ای است از منطق ریاضی، منطق مسندات.

حساب رابطه‌ای و جبر رابطه‌ای معادلند. یعنی هر رابطه‌ای را که بتوان با یک عبارت جبر رابطه‌ای نوشت، می‌توان با عبارتی از حساب رابطه‌ای هم نوشت و برعکس.

حساب رابطه‌ای حالت **توصیفی** دارد ولی جبر رابطه‌ای حالت **دستوری** دارد.

↓  
Prospective

دستورات عملیاتی به سیستم می‌دهیم.

↓  
Descriptive

به کمک عبارات منطقی، شرایط ناظر  
به رابطه را برای سیستم توصیف می‌کنیم.

حساب رابطه‌ای هم ضابطه تشخیص زبان‌های رابطه‌ای کامل است.



بخش هفتم: عملیات در پایگاه داده رابطه‌ای

متغیر تاپلی (Tuple Variable) یا متغیر طیفی (Range Variable):

متغیری است که مقادیر آن تاپل‌های یک رابطه است (هر لحظه یک تاپل).

**RANGVAR SX RANGES OVER S;**

**RANGVAR PX RANGES OVER P;**

**RANGVAR SPX RANGES OVER SP;**

**RANGVAR C2X RANGES OVER (S WHERE CITY='C2');**



طیف مقادیرش تاپل‌هایی از S است که شرط را داشته باشند.



## سورها (Quantifiers) □

□ سور وجودی  $\text{EXISTS } X (F)$ : حداقل یک مقدار برای متغیر  $X$  وجود دارد به نحوی که به ازای آن، فرمول  $F$  به درست ارزیابی شود.

□ سور همگانی (عمومی)  $\text{FOR ALL } X (F)$ : به ازای تمام مقادیر متغیر  $X$ ، فرمول  $F$  به درست ارزیابی می‌شود.

با فرض اینکه  $X$  از مجموعه اعداد صحیح مثبت مقدار می‌گیرد.



$\text{EXISTS } X (X < 10)$  حاصل ارزیابی: TRUE

$\text{FOR ALL } X (X < 10)$  حاصل ارزیابی: FALSE



یک فرمول خوش ساخت (WFF) به صورت زیر تعریف می‌شود:

□ اگر  $R$  یک رابطه و  $T$  یک متغیر تاپلی تعریف شده روی  $R$  باشد، آنگاه  $R(T)$  یک فرمول اتمی است.

[ $R(T)$  یعنی،  $T$  یک عنصر (تاپلی) از  $R$  است.]

□ اگر  $T_i$  یک متغیر تاپلی روی رابطه  $R$  و  $A$  یک صفت از  $R$  باشد و  $T_j$  یک متغیر تاپلی بر روی  $S$  و  $B$  یک

صفت از  $S$  باشد، آنگاه  $T_i.A \text{ theta } T_j.B$  یک فرمول اتمی است (theta یک از عملگرهای متعارف مقایسه‌ای است).

□  $T_i.A \text{ theta } C$  و  $C \text{ theta } T_j.B$  نیز که در آن  $C$  یک مقدار ثابت است، فرمول اتمی هستند.

□ اگر  $F_1$  و  $F_2$  فرمول باشند، آنگاه  $(F_1 \text{ AND } F_2)$ ،  $(F_1 \text{ OR } F_2)$ ،  $\text{NOT}(F_1)$  نیز فرمول هستند.

□ اگر  $F$  یک فرمول و  $T$  یک متغیر تاپلی باشد، آنگاه  $\text{EXISTS } T(F)$  و  $\text{FORALL } T(F)$  نیز فرمول هستند.



اگر  $X$  یک متغیر تاپلی روی رابطه  $R(A_1, A_2, \dots, A_n)$  باشد در اینصورت شکل کلی عبارت حساب رابطه‌ای



بدین صورت است:

(target-items) [WHERE F]

$\{X | R(X) \wedge F(X)\}$  یا  $\{X | F(X)\}$  یا  $\{X | X \in R \wedge F(X)\}$

که در آن target-items فهرستی از صفات متغیر تاپلی  $X$  به صورت  $X.A_1, X.A_2, \dots, X.A_n$  و  $F$  یک فرمول خوش ساخت است. در واقع مجموعه تاپل هایی را نشان می دهد که شرط  $F$  در مورد آنها صادق است.

❑ ST.STID      شماره تمام دانشجویان در رابطه STT

❑ ST.STID WHERE ST.STDEID='D11'      شماره دانشجویان گروه آموزشی D11

$\{ST .STID | ST \in STT \wedge ST. STDEID='D11'\}$

❑ (ST.STID, ST.STL) WHERE EXISTS STCO (ST.STID=STCO.STID AND STCO.COID='COM11')



شماره دانشجویی و مقطع تحصیلی آنهایی که درس COM11 را انتخاب کرده‌اند.



## حساب رابطه‌ای – عبارت حساب رابطه‌ای (ادامه)

بخش هفتم: عملیات در پایگاه داده رابطه‌ای

۴۰



❑ SX.S# شماره همه تهیه کنندگان

❑ SX.SNAME WHERE SX.CITY='C2' AND SX.STATUS > 15

نام تهیه کنندگان شهرستان C2 که وضعیت آنها بزرگتر از 15 باشد.

$\{ \langle \text{SX.SNAME} \rangle \mid \text{SX} \in \text{S} \wedge \text{SX.CITY} = \text{'C2'} \wedge \text{SX.STATUS} > 15 \}$

$\{ \text{SX.SNAME} \mid \text{S}(\text{SX}) \wedge \text{SX.CITY} = \text{'C2'} \wedge \text{SX.STATUS} > 15 \}$

❑ شماره و نام دانشجویان دختر دانشکده فیزیک

$\{ \text{s.STID}, \text{s.SName} \mid \text{STT}(\text{s}) \wedge \text{s.gender} = \text{'F'} \wedge (\exists \text{d})(\text{department}(\text{d}) \wedge \text{d.name} = \text{'Maths'} \wedge \text{d.deptId} = \text{s.deptNo}) \}$

❑ مثال‌های بیشتر در کتاب‌های مرجع .





یادآوری: بین این دو سور روابط زیر وجود دارد.

$\text{FOR ALL } X (F) = \text{NOT EXISTS } X (\text{NOT } F)$

$\text{EXISTS } X (F) = \text{NOT } (\text{FORALL } X (\text{NOT } F))$

$\text{FORALL } X (F) \Rightarrow \text{EXISTS } X (F)$

$\text{NOT EXISTS } X (F) \Rightarrow \text{NOT FORALL } X (F)$

بر اساس روابط فوق می‌توان روابط پیچیده دیگری را نیز استنباط کرد مانند روابط هم ارزی زیر:

$\text{FORALL } X (F \text{ AND } G) = \text{NOT EXISTS } X (\text{NOT}(F) \text{ OR } \text{NOT}(G))$

$\text{FORALL } X (F \text{ OR } G) = \text{NOT EXISTS } X (\text{NOT}(F) \text{ AND } \text{NOT}(G))$

$\text{EXISTS } X (F \text{ OR } G) = \text{NOT FORALL } X (\text{NOT}(F) \text{ AND } \text{NOT}(G))$

$\text{EXISTS } X (F \text{ AND } G) = \text{NOT FORALL } X (\text{NOT}(F) \text{ OR } \text{NOT}(G))$



هر عبارت جبر رابطه ای را می توان با استفاده از عبارات حساب رابطه ای نوشت

مثال : فرض کنید داریم:

$$[e] = \{ \langle x_1, \dots, x_n \rangle \mid F \}$$

آنگاه این دو عبارت معادل هستند:

$$[\sigma_C(e)] = \{ \langle x_1, \dots, x_n \rangle \mid F \wedge C' \}$$

که  $C'$  از  $C$  با جایگزینی هر صفت با متغیر معادل آن به دست می آید.



## پرسش و پاسخ . . .

ایمیل : [zarepour@iust.ac.ir](mailto:zarepour@iust.ac.ir)

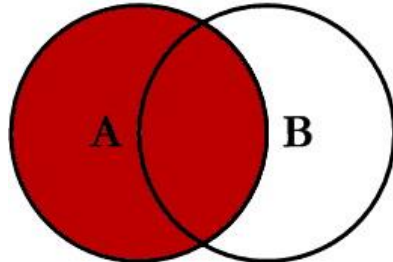
ارتباط حضوری: ساعت مشخص شده در برنامه هفتگی به عنوان رفع

اشکال دانشجویی (روزهای شنبه و دوشنبه ساعت ۹:۳۰ تا ۱۱ صبح)

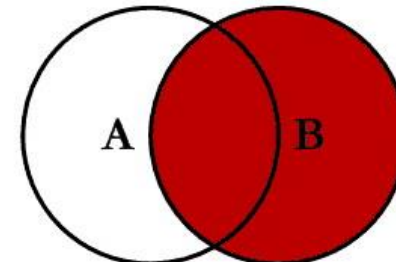
[www.ezarepour.ir](http://www.ezarepour.ir)



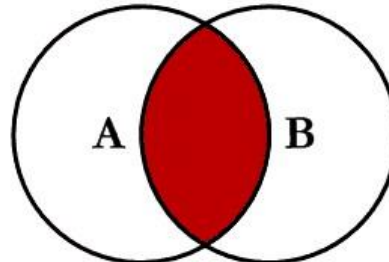
# SQL JOINS



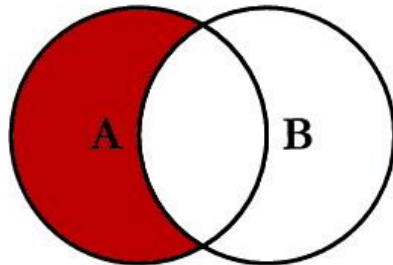
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key
```



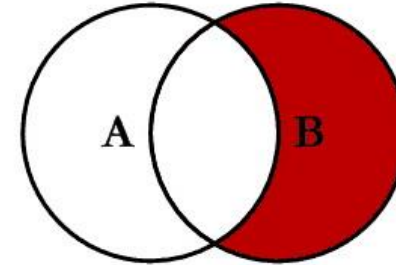
```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key
```



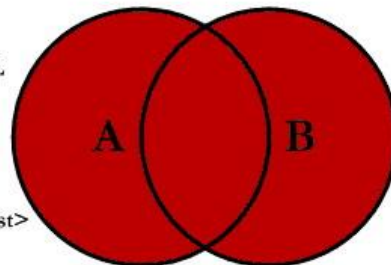
```
SELECT <select_list>  
FROM TableA A  
INNER JOIN TableB B  
ON A.Key = B.Key
```



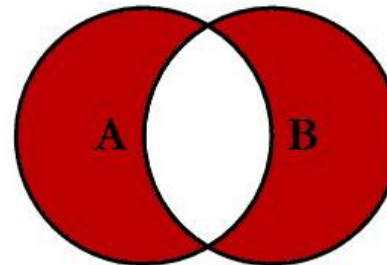
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key  
WHERE B.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL  
OR B.Key IS NULL
```