SQL典例　　序号表

序号表 生成方式

序号表 维护

序号表 应用实例

如何得到一条语句所影响的行数？

用found_rows()函数

判断Select得到的行数

用row_count()函数判断

Update或Delete影响的行数

```
select  *  from  SC
where       grade > = 60
select      found_rows()
```

```
delete  from  SC
where       grade >  100
select      row_count()
```

序号表Nums的生成方式

```
delimiter $$
create procedure `seqNumsInsert_1`()
begin
    declare  i  int;   set  i = 1;
    while  i <= 100  do
            insert into  seqNums  values (i);
            set  i = i+1;
    end while;
end $$
delimiter;
call  ` seqNumsInsert_1 `;
```

序号表Nums的生成方式

```
delimiter $$
create procedure `seqNumsInsert_2`()
begin
    declare  i  int;   set  i = 1;
    insert into  seqNums  values(1);
    while  i <= 50  do
            insert into  seqNums  select  sn + i  from  seqNums;
            set  i = i * 2;
    end while;
    insert into  seqNums  select  sn + i  from  seqNums  where  sn + i <= 100;
end $$
delimiter ;
call  ` seqNumsInsert_2 `;
```

序号表Nums的生成方式

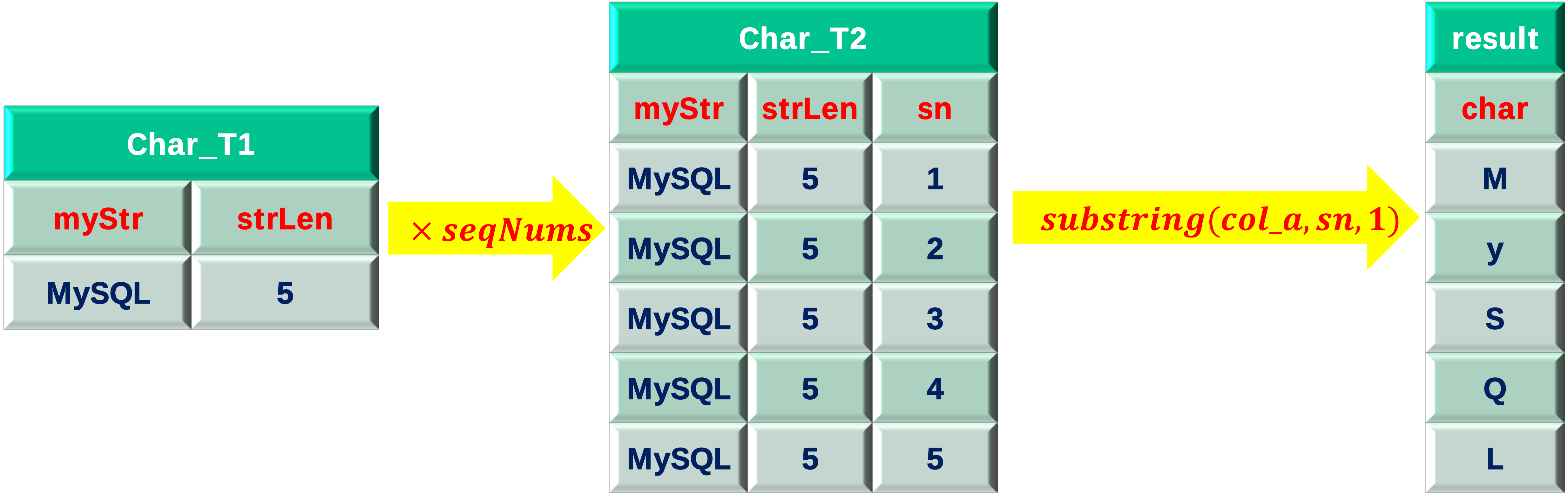
```
delimiter $$
create procedure `seqNumsInsert_2`()
begin
    declare  n  int;   set  n = 100;
    with  recursive  Nums(i)  as (
            select   1
            union all
            select   i+1  from  Num  where  i < n )
    select  i  into  seqNums  from  Nums ;
end $$
delimiter ;
call  ` seqNumsInsert_2 `;
```

序号表Nums的生成方式

```
create table  digits (digit  int)
insert into  digits (digit)   values  (0), (1), (2), (3), (4), (5), (6), (7), (8), (9)


select              D3.digit * 100 +
                    D2.digit * 10 +
                    D1.digit  + 1 as sn
into                seqNums
from                digits  as  D1
      cross join    digits  as  D2
      cross join    digits  as  D3
```

将字符串每个字符单独作为一行输出



Char_T1	
myStr	strLen
MySQL	5

$\times$ *seqNums*

Char_T2		
myStr	strLen	sn
MySQL	5	1
MySQL	5	2
MySQL	5	3
MySQL	5	4
MySQL	5	5

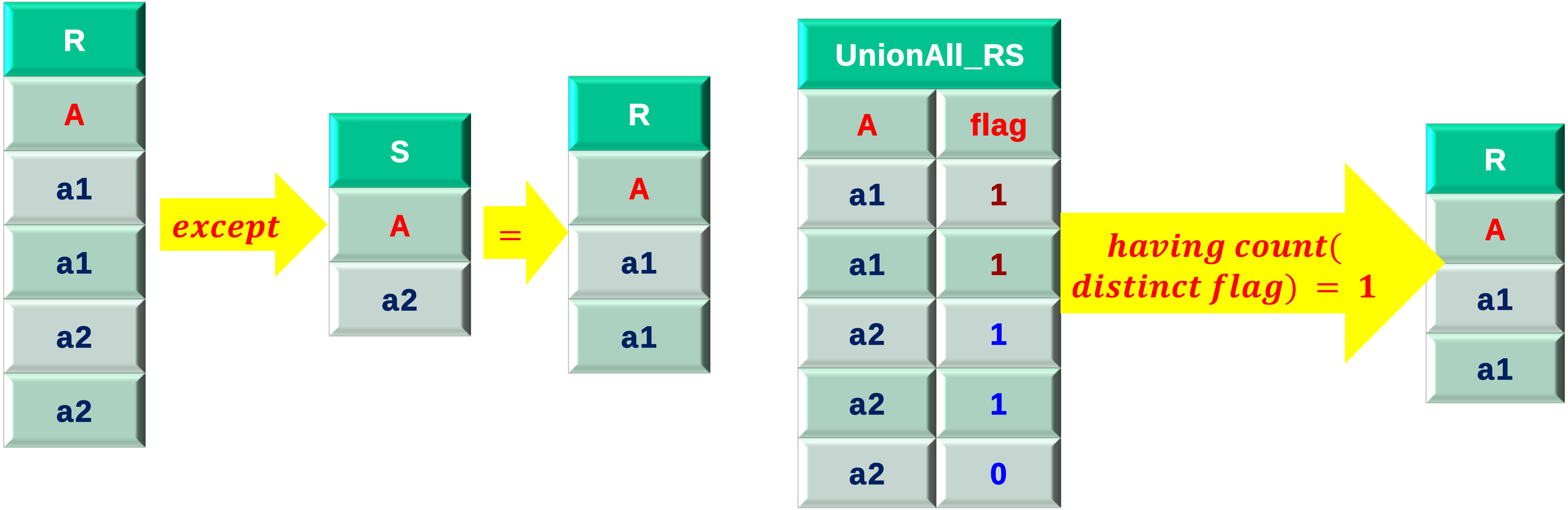
$substring(col\_a, sn, 1)$

result
char
M
y
S
Q
L

将字符串每个字符单独作为一行输出

```
select  substring(col_a, sn, 1)
from
       ( select  col_a, sn
         from   ( select  'MySQL'  myStr,
                          len('MySQL')  strLen )  as  Char_T1,
                seqNums
         where   strLen >= sn )  as  Char_T2
```

使用union all自定义集合差操作except



R

A
a1
a1
a2
a2

except

S

A
a2

=

R

A
a1
a1

UnionAll_RS

A	flag
a1	1
a1	1
a2	1
a2	1
a2	0

having count(distinct flag) = 1

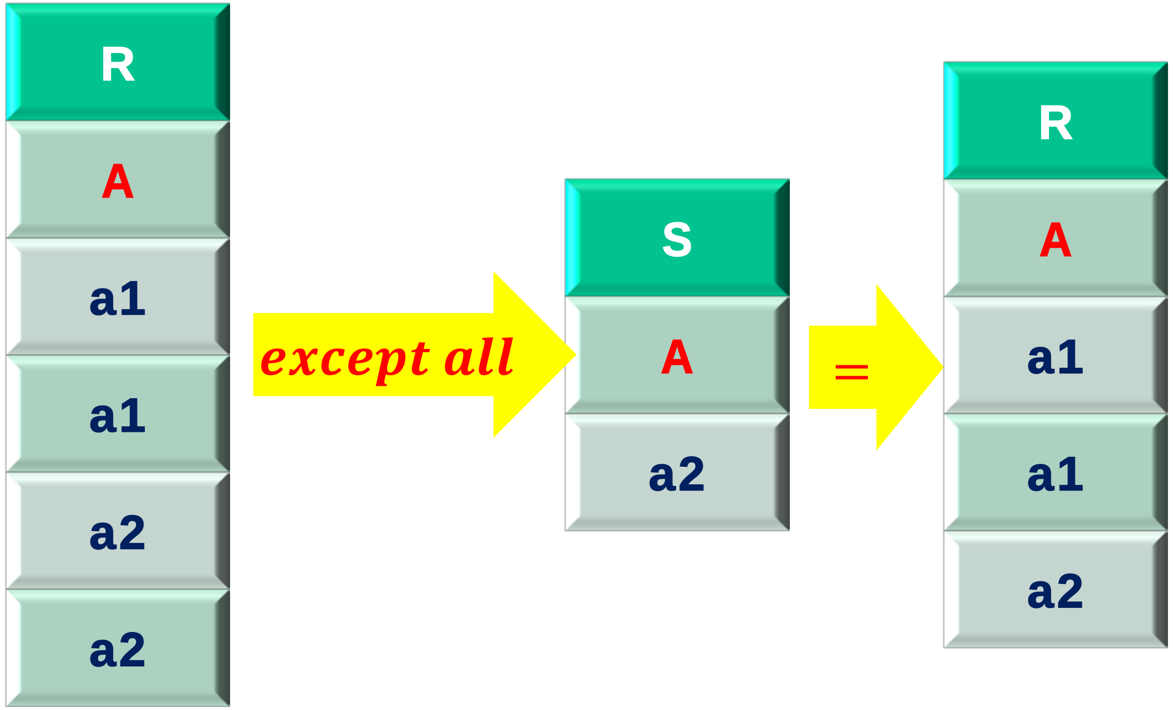
R

A
a1
a1

使用union all自定义集合差操作except

```
select      A
from        ( select    1  as  flag,  A
             from       R
            union all
            select      0,  A
            from        S )  as  UnionAll_RS
group by    A
having      count ( distinct flag ) = 1
  and       max ( flag ) = 1
```

使用union all自定义集合差操作except all



R
A
a1
a1
a2
a2

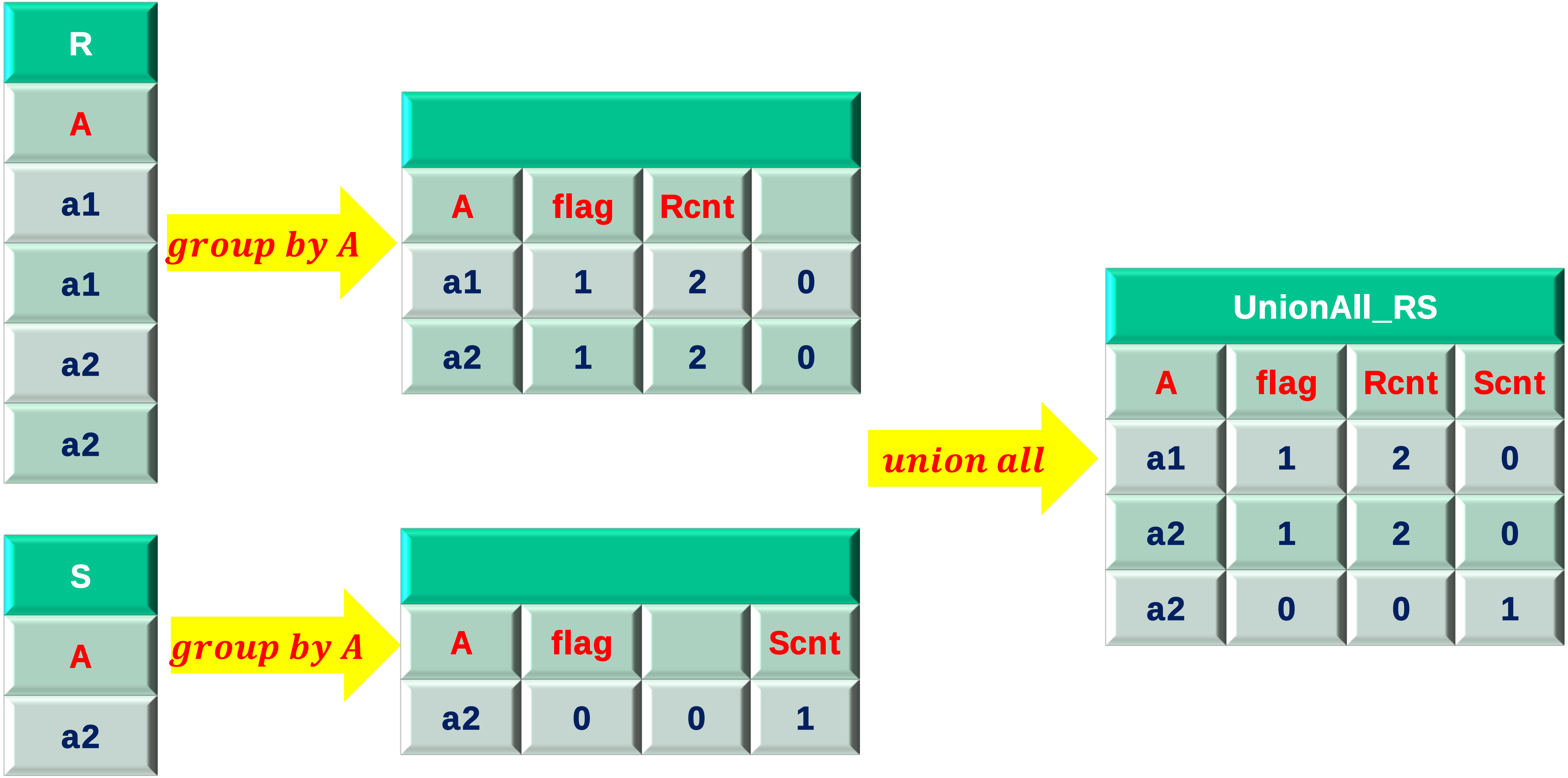
except all

S
A
a2

=

R
A
a1
a1
a2

使用union all自定义集合差操作except all



R

A
a1
a1
a2
a2

group by A

A	flag	Rcnt	
a1	1	2	0
a2	1	2	0

S

A
a2

group by A

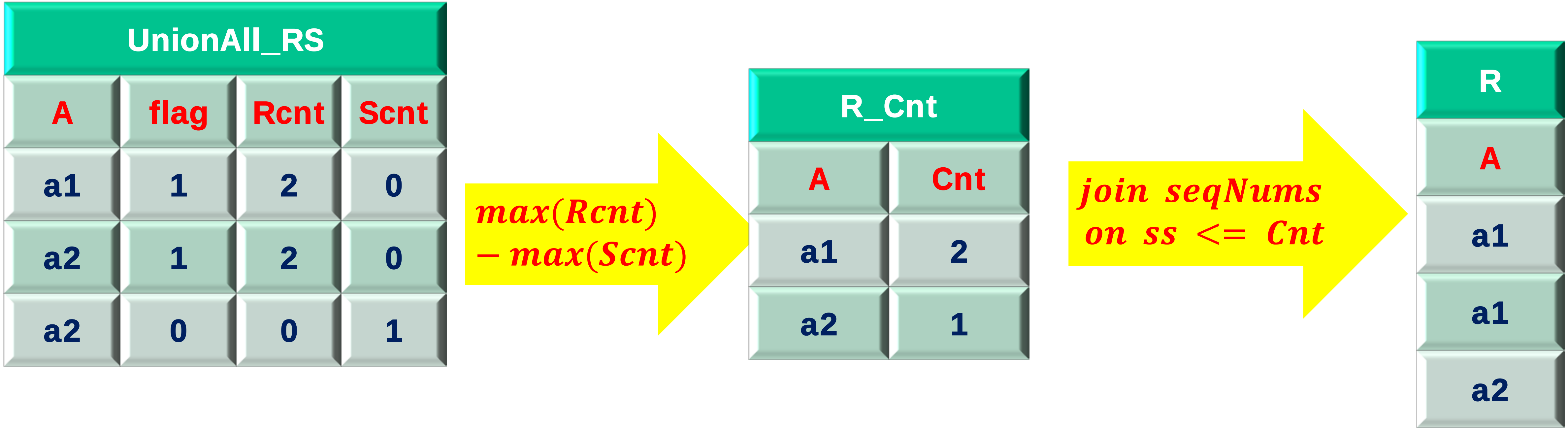
A	flag		Scnt
a2	0	0	1

union all

UnionAll_RS

A	flag	Rcnt	Scnt
a1	1	2	0
a2	1	2	0
a2	0	0	1

使用union all自定义集合差操作except all



UnionAll_RS			
A	flag	Rcnt	Scnt
a1	1	2	0
a2	1	2	0
a2	0	0	1

$max(Rcnt) - max(Scnt)$

R_Cnt	
A	Cnt
a1	2
a2	1

$join\ seqNums\ on\ ss <= Cnt$

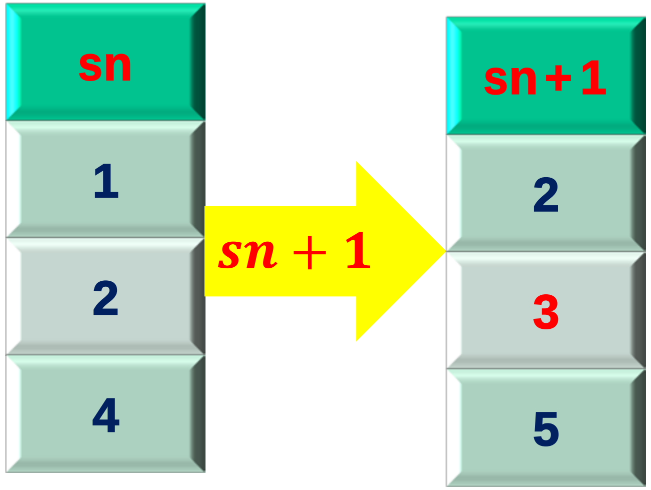
R
A
a1
a1
a2

使用union all自定义集合差操作except

```
select      A
from        ( select        A, max(Rcnt) – max(Scnt)  as  Cnt
              from          (select         1 as flag,  A,  count(*) Rcnt, 0
                             from           R
                             group by       A
                             union all
                             select         0,  A, 0, count(*) Scnt
                             from           S
                             group by   A)  UnionAll_RS
            )  R_Cnt
join        seqNums  on  sn <= Cnt
```

寻找序号表中缺失的最小值

基本思想：当前序号加1如果不在序号表中，则它就是一个缺失序号



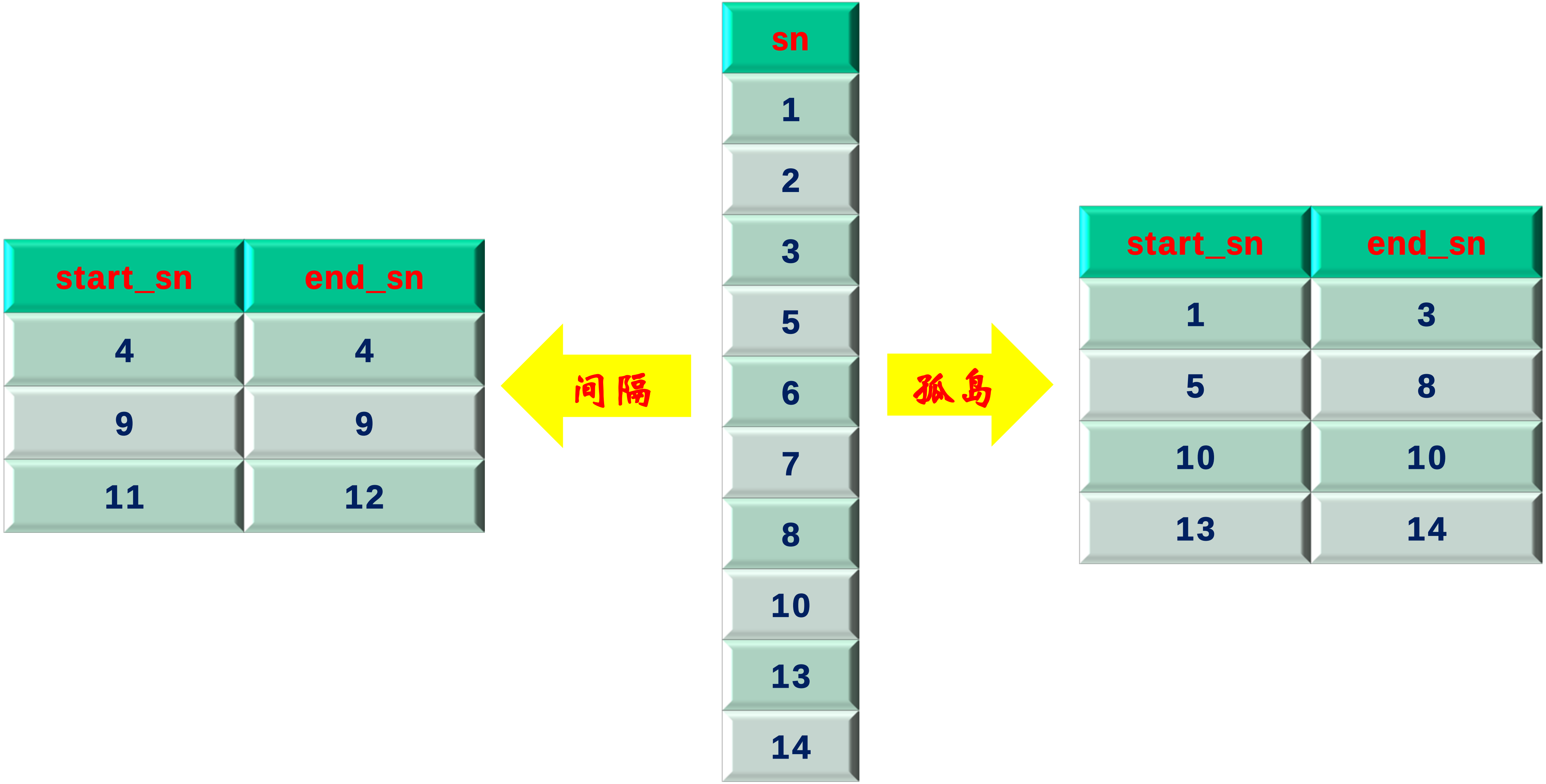
sn
1
2
4

$sn + 1$

sn+1
2
3
5

```
select      min( sn + 1 )
from        seqNums  N1
where       not exists
            ( select        *
              from          seqNums  N2
              where         N2.sn = N1.sn + 1 )
```

孤岛和间隔



sn
1
2
3
5
6
7
8
10
13
14

间隔

start_sn	end_sn
4	4
9	9
11	12

孤岛

start_sn	end_sn
1	3
5	8
10	10
13	14