

2. 데이터 제한과 정렬

- ◆ 본 과정을 마치면 다음을 할 수 있어야 합니다.
 - 질의에 의해 검색되는 행을 제한할 수 있습니다.
 - 질의에 의해 검색되는 행을 정렬할 수 있습니다.




Selection을 사용하여 행을 제한

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKES	MANAGER		30
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20
...				

“... 부서 10의 모든
종업원을 검색”



EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20

선택된 행 제한

- ◆ WHERE 절을 사용하여 리턴되는 행을 제한합니다.

```
SELECT  [DISTINCT] { *, column [alias], ... }  
FROM    table  
[WHERE  condition(s)];
```

- ◆ WHERE 절은 FROM 절 다음에 옵니다.

WHERE 절 사용

```
SQL> SELECT  ename, job, deptno  
2  FROM      emp  
3  WHERE     job='CLERK';
```

ENAME	JOB	DEPTNO
SMITH	CLERK	20
ADAMS	CLERK	20
JAMES	CLERK	30
MILLER	CLERK	10

문자 스트링과 날짜

- ◆ 문자 스트링과 날짜 값은 단일 인용부호(' ')로 둘러싸여 있습니다.
- ◆ 문자 값은 대/소문자를 구분하고, 날짜 값은 날짜 형식을 구분합니다.
- ◆ 디폴트 날짜 형식은 'DD-MON-YY' 입니다.

```
SQL> SELECT  ename, job, deptno  
2  FROM      emp  
3  WHERE      ename='JAMES';
```

비교 연산자

Operator	Description
=	같다
>	보다 크다
>=	보다 크거나 같다
<	보다 작다
<=	보다 작거나 같다
≠	같지 않다.

비교 연산자 사용

```
SQL> SELECT  ename, sal, comm  
2 FROM      emp  
3 WHERE     sal <= comm;
```

ENAME	SAL	COMM
MARTIN	1250	1400

다른 비교연산자

Operator	Description
BETWEEN ... AND ...	두 값의 사이(포함하는)
IN(list)	어떤 값의 목록과 일치
LIKE	문자 패턴과 일치
IS NULL	NULL 값
IS NAN	Not a Number 값
IS INFINITE	무한대 값

BETWEEN 연산자 사용

- ◆ 값의 범위에 해당하는 행을 디스플레이 하기 위해 BETWEEN 연산자를 사용합니다.

```
SQL> SELECT  ename, sal  
2  FROM      emp  
3  WHERE     sal BETWEEN 1000 AND 1500;
```

ENAME	SAL	Lower limit	Higher limit
WARD	1250		
MARTIN	1250		
TURNER	1500		
ADAMS	1100		
MILLER	1300		

IN 연산자 사용

- ◆ 목록에 있는 값들과 비교하기 위해서 IN 연산자를 사용합니다.

```
SQL> SELECT empno, ename, sal, mgr  
2 FROM emp  
3 WHERE mgr IN (7902, 7566, 7788);
```

EMPNO	ENAME	SAL	MGR
7369	SMITH	800	7902
7788	SCOTT	3000	7566
7876	ADAMS	1100	7788
7902	FORD	3000	7566

LIKE 연산자 사용

- ◆ 검색 스트링 값에 대한 와일드 카드 검색을 위해서 LIKE 연산자를 사용합니다.
- ◆ 검색 조건은 리터럴 문자나 숫자를 포함할 수 있습니다.
 - % 는 문자가 없거나 또는 하나 이상을 나타냅니다.
 - _ 는 하나의 문자를 나타냅니다.

```
SQL> SELECT  ename  
2 FROM      emp  
3 WHERE     ename LIKE 'S%';
```

LIKE 연산자 사용

- ◆ 패턴 일치 문자를 조합할 수 있습니다.

```
SQL> SELECT  ename  
2 FROM      emp  
3 WHERE     ename LIKE '_A%';
```

ENAME
WARD
MARTIN
JAMES

- ◆ “%”나 “_”에 대해서 검색하기 위해서 ESCAPE 식별자를 사용할 수 있습니다.

IS NULL 연산자 사용

- ◆ IS NULL 연산자로 null 값을 테스트 합니다.

```
SQL> SELECT  ename, mgr  
2  FROM      emp  
3  WHERE     mgr IS NULL;
```

ENAME	MGR
-----	-----
KING	

IS NAN, IS INFINITE

- ◆ 10g 버전에 추가된 연산자 입니다.
- ◆ NAN : Not a Number 값을 테스트 합니다.
- ◆ IS INFINITE : 무한대 값을 테스트 합니다.
- ◆ BINARY_FLOAT 데이터 타입에서 `binary_float_nan`과 `binary_float_infinity` 로 값이 입력되어 있을 경우 사용합니다.
- ◆ BINARY_DOUBLE 데이터 타입에서 `binary_double_nan`과 `binary_double_infinite` 로 값이 입력되어 있을 경우 사용합니다.

```
SQL> CREATE TABLE binary_test
  2  (bin_float BINARY_FLOAT,  ➡ 32 bit 부동 소수점 표현
  3  bin_double BINARY_DOUBLE); ➡ 64 bit 부동 소수점 표현
```

```
SQL> INSERT INTO binary_test
  2  VALUES(39.5f, 15.7d);
```

```
SQL> INSERT INTO binary_test VALUES(
  2  BINARY_FLOAT_INFINITY,  ➡ 무한대값을 저장
  3  BINARY_DOUBLE_INFINITY);
```

```
SQL> INSERT INTO binary_test
  2  VALUES (binary_float_nan, binary_double_nan); ➡ not a number
```


논리 연산자

Operator	Description
AND	양쪽 컴포넌트의 조건이 TRUE 이면 TRUE를 리턴합니다.
OR	한쪽 컴포넌트의 조건만 TRUE이면 TRUE를 리턴합니다.
NOT	이후의 조건이 FALSE이면 TRUE를 리턴합니다.

AND 연산자 사용

- ◆ AND는 양쪽의 조건이 참이어야 TRUE를 리턴합니다.

```
SQL> SELECT empno, ename, job, sal  
2 FROM emp  
3 WHERE sal >= 1100  
4 AND job = 'CLERK';
```

EMPNO	ENAME	JOB	SAL
7876	ADAMS	CLERK	1100
7934	MILLER	CLERK	1300

OR 연산자 사용

- ◆ OR는 한쪽의 조건이 참이면 TRUE를 리턴합니다.

```
SQL> SELECT empno, ename, job, sal  
2 FROM emp  
3 WHERE sal >= 1100  
4 OR job = 'CLERK';
```

EMPNO	ENAME	JOB	SAL
7369	SMITH	CLERK	800
7499	ALLEN	SALESMAN	1600
7521	WARD	SALESMAN	1250

...

14 개의 행이 선택되었습니다.

NOT 연산자 사용

- NOT 연산자는 뒤의 조건에 반대되는 결과를 리턴합니다.

```
SQL> SELECT ename, job  
2 FROM emp  
3 WHERE job NOT IN ('CLERK', 'MANAGER', 'ANALYST');
```

ENAME	JOB
ALLEN	SALESMAN
WARD	SALESMAN
MARTIN	SALESMAN
KING	PRESIDENT
TURNER	SALESMAN

우선순위 규칙

우선순위	연산자
1	모든 비교 연산자
2	NOT
3	AND
4	OR

우선순위의 규칙 적용

```
SQL> SELECT  ename, job, sal  
2  FROM      emp  
3  WHERE     job='SALESMAN'  
4  OR        job='PRESIDENT'  
5  AND       sal>1500;
```

ENAME	JOB	SAL
ALLEN	SALESMAN	1600
WARD	SALESMAN	1250
MARTIN	SALESMAN	1250
KING	PRESIDENT	5000
TURNER	SALESMAN	1500

우선순위의 규칙 적용



우선순위를 강제로 바꾸기 위해서 괄호를 사용합니다.

```
SQL> SELECT  ename, job, sal  
2  FROM      emp  
3  WHERE      (job='SALESMAN'  
4  OR         job='PRESIDENT')  
5  AND        sal>1500;
```

ENAME	JOB	SAL
ALLEN	SALESMAN	1600
KING	PRESIDENT	5000

집합 연산자

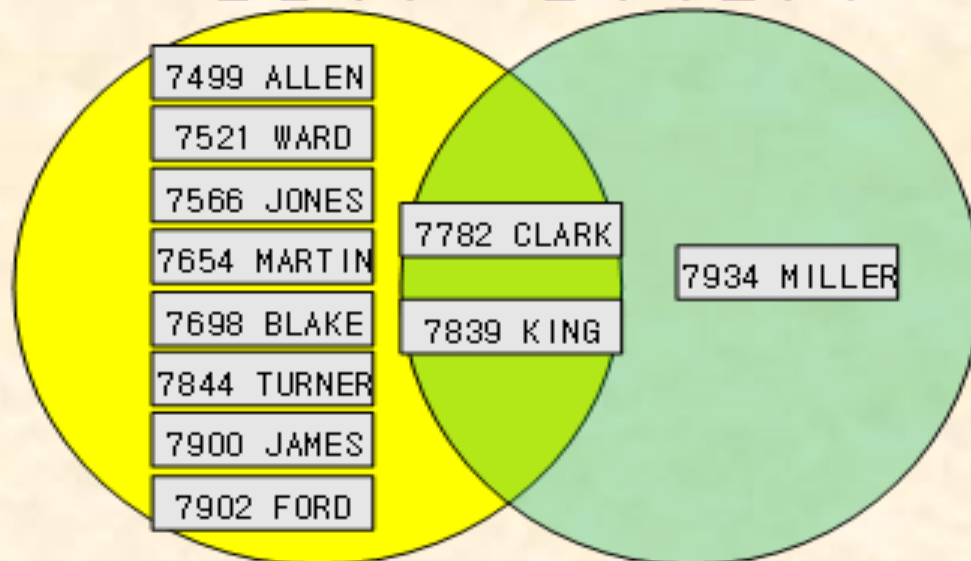
Operator	Description
UNION	중복된 정보는 한번만 보여줌
UNION ALL	모든 데이터를 보여줌
INTERSECT	중복된 데이터를 보여줌
MINUS	첫 번째 쿼리에만 있는 데이터를 보여줌

UNION 연산자 사용

- ◆ 첫 번째 쿼리와 두 번째 쿼리의 중복된 정보는 한 번만 보여줍니다.

```
SQL> SELECT empno, ename FROM emp WHERE hiredate LIKE '81%'  
2 UNION  
3 SELECT empno, ename FROM emp WHERE deptno = 10;
```

81년 입사자 10번 부서 근무자



EMPNO	ENAME

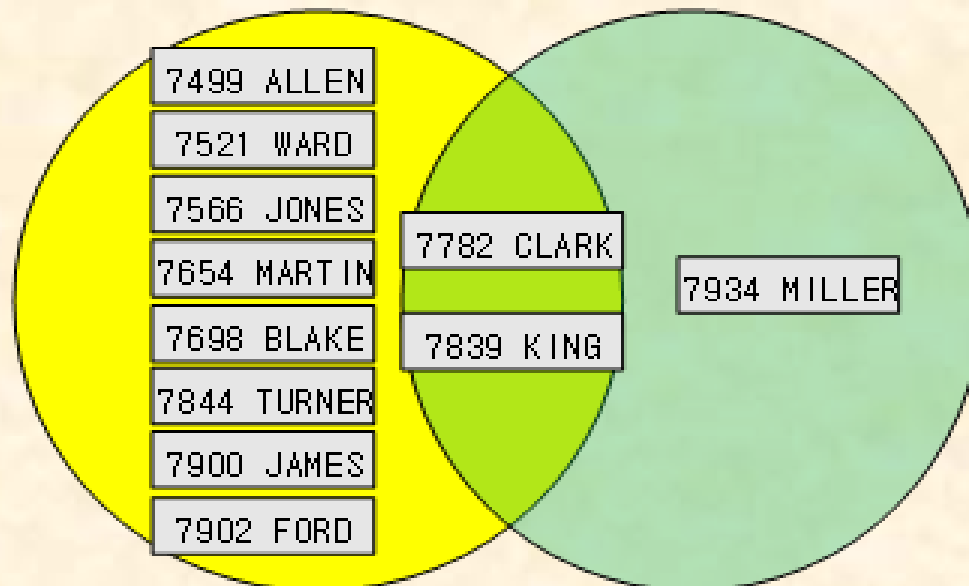
7499	ALLEN
7521	WARD
7566	JONES
7654	MARTIN
7698	BLAKE
7782	CLARK
7839	KING
7844	TURNER
7900	JAMES
7902	FORD
7934	MILLER

UNION ALL 연산자 사용

- ◆ 첫 번째 쿼리와 두 번째 쿼리 모든 정보를 보여줍니다.
중복된 정보를 모두 보여줍니다.

```
SQL> SELECT empno, ename FROM emp WHERE hiredate LIKE '81%'  
2  UNION ALL  
3  SELECT empno, ename FROM emp WHERE deptno = 10;
```

81년 입사자 10번 부서 근무자



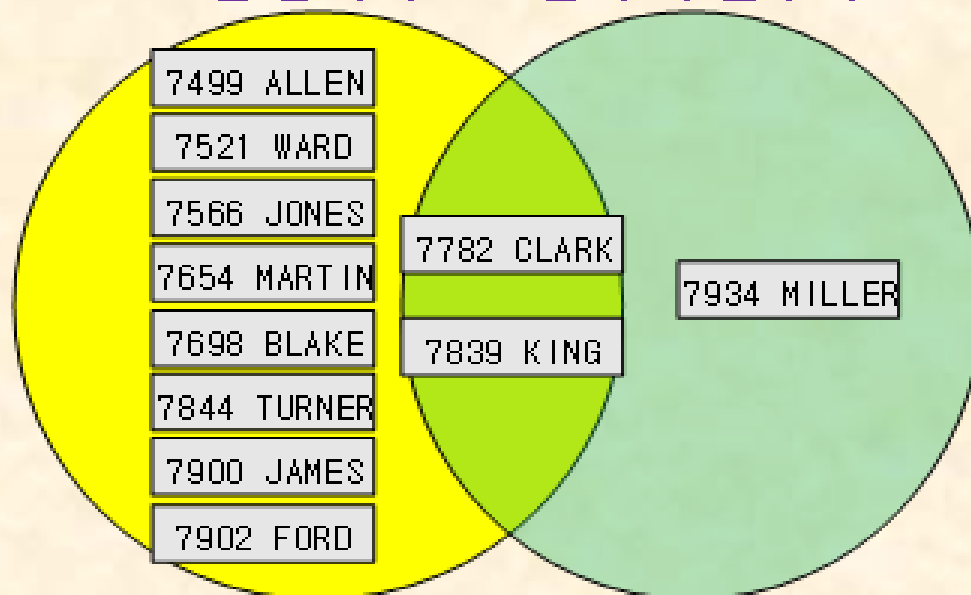
EMPNO	ENAME
7499	ALLEN
7521	WARD
7566	JONES
7654	MARTIN
7698	BLAKE
7782	CLARK
7839	KING
7844	TURNER
7900	JAMES
7902	FORD
7782	CLARK
7839	KING
7934	MILLER

MINUS 연산자 사용

- ◆ 두 번째 쿼리에는 없고 첫 번째 쿼리에만 있는 데이터를 보여줍니다.

```
SQL> SELECT empno, ename FROM emp WHERE hiredate LIKE '81%'  
2 MINUS  
3 SELECT empno, ename FROM emp WHERE deptno = 10;
```

81년 입사자 10번 부서 근무자



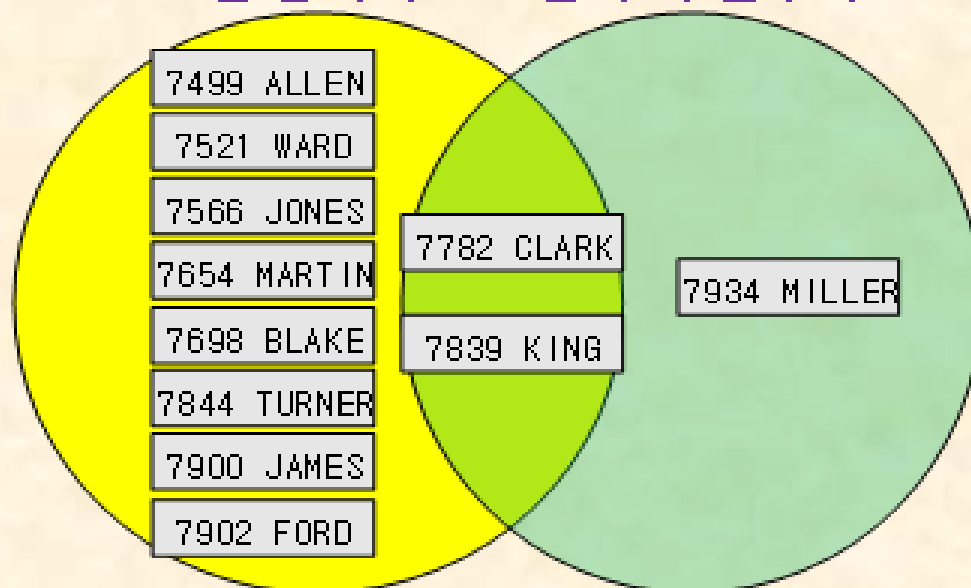
EMPNO	ENAME
7499	ALLEN
7521	WARD
7566	JONES
7654	MARTIN
7698	BLAKE
7844	TURNER
7900	JAMES
7902	FORD

INTERSECT 연산자 사용

- 첫 번째 쿼리와 두 번째 쿼리에서 중복된 데이터를 보여줍니다.

```
SQL> SELECT empno, ename FROM emp WHERE hiredate LIKE '81%'  
2 INTERSECT  
3 SELECT empno, ename FROM emp WHERE deptno = 10;
```

81년 입사자 10번 부서 근무자



EMPNO	ENAME

7782	CLARK
7839	KING

ORDER BY 절을 이용한 정렬

◆ ORDER BY 절은 SELECT 문장의 가장 뒤에 옵니다.

- ASC: 오름차순, 디폴트
- DESC: 내림차순

```
SQL> SELECT   ename, job, deptno, hiredate
2  FROM      emp
3  ORDER BY   hiredate;
```

ENAME	JOB	DEPTNO	HIREDATE
SMITH	CLERK	20	80/12/17
ALLEN	SALESMAN	30	81/02/20

...

14 개의 행이 선택되었습니다.

내림차순 정렬

```
SQL> SELECT   ename, job, deptno, hiredate  
2  FROM      emp  
3  ORDER BY   hiredate DESC;
```

ENAME	JOB	DEPTNO	HIREDATE
ADAMS	CLERK	20	87/05/23
SCOTT	ANALYST	20	87/04/19
MILLER	CLERK	10	82/01/23
FORD	ANALYST	20	81/12/03

...

14 개의 행이 선택되었습니다.

여러 변경에 의한 정렬

```
SQL> SELECT empno, ename, sal*12 annsal  
2 FROM emp  
3 ORDER BY annsal;
```

EMPNO	ENAME	ANNSAL
7369	SMITH	9600
7900	JAMES	11400
7876	ADAMS	13200
7521	WARD	15000

...

14 개의 행이 선택되었습니다.

Quiz 1

- SELECT ename , sal , deptno "Department"
FROM emp ORDER BY _____;
이 query는 부서별로 sort하기를 얻는다. 밑줄 친 부분에 들어갈
적당한 것은? (복수 선택)
A. ename
B. department no
C. deptno
D. Department
E. "Department"
F. 3

Quiz 2

□ 다음 QUERY의 결과는 ?

```
SELECT ename FROM emp
```

```
WHERE ((( ename like '%S%' and ename like  
'%T%') and ename like '%P%') or ename like  
'%R%');
```

A. ename에 'S'자나, 'T'자나, 'P'자나, 'R'자가 있는 경우

B. ename에 'S','T'가 있으면서 'P'자나, 'R'자가 있는 경우

C. ename에 'S','T','P'가 모두 있는 경우나 'R'자가 있는 경우

Quiz 3

□ 다음 QUERY의 결과는?

SELECT ename FROM emp WHERE ename like
'_x%';

A. ename이 'x'로 시작하는 사원들

B. ename이 'x'로 끝나는 사원들

C. ename의 중간에 'x'가 들어간 사원들

D. ename의 두번째 문자가 'x'인 사원들

Quiz 4

- 다음 문장을 실행했을 때 결과로 나올 수 있는 것은?

```
SELECT name FROM s_customer  
WHERE name LIKE '%X\_Y%' ESCAPE '\';
```

- A. THE X_Y SPORTS
- B. THE X\&Y SPORTS
- C. THE X&Y SPORTS
- D. THE X_Y SPORTS

Quiz 5

- 다음 중 모든 datatype에 대하여 사용가능한 연산자는(세 가지)?
- A. +
 - B. IS NULL
 - C. NOT
 - D. BETWEEN ... AND ...
 - E. LIKE