DML(SELECT), 연산자


DQL

DQL

DQL - SELECT

- 데이터를 검색(추출)하기 위해 사용되는 언어
- DQL은 DML에 속해 있으며 DML중 SELECT만을 따로 의미
- 데이터를 조회한 결과를 Result Set이라고 함
- Result Set은 0개 이상의 행을 포함
- Result Set은 특정한 기준에 의해 정렬이 가능
- 특정 컬럼이나 특정 행을 조회할 수 있으며, 여러 테이블의 특정 행과 컬럼을 조회하는 것도 가능

DQL

SELECT 기본 작성법

SELECT 컬럼명 [, 컬럼명,…] FROM 테이블명 WHERE 조건식;

1. SELECT
   - 조회하고자 하는 컬럼명을 기술
   - 여러 컬럼을 조회하는 경우 컬럼은 쉼표로 구분
   - 모든 컬럼을 조회하는 경우 컬럼명 대신 '*' 기호를 사용가능
   - 조회 결과는 기술한 컬럼명 순으로 표시
2. FROM
   - 조회 대상 컬럼이 포함된 테이블명을 기술
3. WHERE
   - 행을 선택하는 조건을 기술
   - 여러 개의 제한 조건을 포함 할 수 있으며, 논리연산자로 연결
   - 조건을 만족시키는 행들만 Result Set에 포함
   - 조건없이 모든 행을 검색하려는 경우 WHERE절은 생략이 가능

DQL

SELECT 사용 예시 – 기본 1

EMPLOYEE 테이블에서 직원들의 전체 정보 조회
SELECT * FROM EMPLOYEE;

EMP_ID	EMP_NAME	EMP_NO	EMAIL	PHONE	DEPT_CODE	JOB_CODE	SAL_LEVEL	SALARY	BONUS	MANAGER_ID	HIRE_DATE	ENT_DATE	ENT_YN
200	선동일	621235-1985634	sun_di@kh.or.kr	01099546325	D9	J1	S1	8000000	0.3	(null)	90/02/06	(null)	N
201	송송기	631156-1548654	song_jk@kh.or.kr	01045686656	D9	J2	S1	6000000	(null)	200	01/09/01	(null)	N
202	노옹철	861015-1356452	no_hc@kh.or.kr	01066656263	D9	J2	S4	3700000	(null)	201	01/01/01	(null)	N
203	송은희	631010-2653546	song_eh@kh.or.kr	01077607879	D6	J4	S5	2800000	(null)	204	96/05/03	(null)	N
204	유재식	660508-1342154	yoo_js@kh.or.kr	01099999129	D6	J3	S4	3400000	0.2	200	00/12/29	(null)	N
205	정중하	770102-1357951	jung_jh@kh.or.kr	01036654875	D6	J3	S4	3900000	(null)	204	99/09/09	(null)	N
206	박나라	630709-2054321	pack_nr@kh.or.kr	01096935222	D5	J7	S6	1800000	(null)	207	08/04/02	(null)	N
207	하이유	690402-2040612	ha_iy@kh.or.kr	01036654488	D5	J5	S5	2200000	0.1	200	94/07/07	(null)	N
208	김해술	870927-1313564	kim_hs@kh.or.kr	01078634444	D5	J5	S5	2500000	(null)	207	04/04/30	(null)	N
209	심봉선	750206-1325546	sim_bs@kh.or.kr	0113654485	D5	J3	S4	3500000	0.15	207	11/11/11	(null)	N
210	윤은해	650505-2356985	youn_eh@kh.or.kr	0179964233	D5	J7	S5	2000000	(null)	207	01/02/03	(null)	N
211	전형돈	830807-1121321	jun_hd@kh.or.kr	01044432222	D8	J6	S5	2000000	(null)	200	12/12/12	(null)	N
212	장쯔위	780923-2234542	jang_zw@kh.or.kr	01066682224	D8	J6	S5	2550000	0.25	211	15/06/17	(null)	N
213	하동운	621111-1785463	ha_dh@kh.or.kr	01158456632	(null)	J6	S5	2320000	0.1	(null)	99/12/31	(null)	N
214	방명수	856795-1313513	bang_ms@kh.or.kr	01074127545	D1	J7	S6	1380000	(null)	200	10/04/04	(null)	N
215	대북혼	881130-1050911	dae_bh@kh.or.kr	01088808584	D5	J5	S4	3760000	(null)	(null)	17/06/19	(null)	N
216	차태연	770808-1364897	cha_ty@kh.or.kr	01064643212	D1	J6	S5	2780000	0.2	214	13/03/01	(null)	N
217	전지연	770808-2665412	jun_jy@kh.or.kr	01033624442	D1	J6	S4	3660000	0.3	214	07/03/20	(null)	N
218	이오리	870427-2232123	loo_or@kh.or.kr	01022306545	(null)	J7	S5	2890000	(null)	(null)	16/11/28	(null)	N
219	임시환	660712-1212123	im_sw@kh.or.kr	(null)	D2	J4	S6	1550000	(null)	(null)	99/09/09	(null)	N
220	이중석	770823-1113111	lee_js@kh.or.kr	(null)	D2	J4	S5	2490000	(null)	(null)	14/09/18	(null)	N
221	유하진	800808-1123341	yoo_hj@kh.or.kr	(null)	D2	J4	S5	2480000	(null)	(null)	94/01/20	(null)	N
222	이태림	760918-2854697	lee_tr@kh.or.kr	01033000002	D8	J6	S5	2436240	0.35	100	97/09/12	17/09/12	Y

DQL

SELECT 사용 예시 – 기본 2

EMPLOYEE 테이블에서 직원들의 정보 중 사원번호, 이름, 급여정보 출력

```
SELECT    EMP_ID,
          EMP_NAME,
          SALARY
FROM EMPLOYEE;
```

EMP_ID	EMP_NAME	SALARY
200	선동일	8000000
201	송송기	6000000
202	노옹절	3700000
203	송은희	2800000
204	유재식	3400000
205	정중하	3900000
206	박나라	1800000
207	하이유	2200000
208	김해술	2500000
209	심봉선	3500000
210	윤은해	2000000
211	전형돈	2000000
212	장쯔위	2550000
213	하농운	2320000
214	방명수	1380000
215	대북혼	3760000
216	차태연	2780000
217	전지연	3660000
218	이오리	2890000
219	임시환	1550000
220	이중석	2490000
221	유하진	2480000
222	이태림	2436240

DQL

SELECT 사용 예시 – 기본 3

EMPLOYEE 테이블에서
직급코드가 J5인 직원들의 정보 중 사원번호, 이름, 직급코드, 급여정보 출력

```
SELECT    EMP_ID,
          EMP_NAME,
          JOB_CODE,
          SALARY
FROM EMPLOYEE
WHERE JOB_CODE='J5';
```

EMP_ID	EMP_NAME	JOB_CODE	SALARY
207	하이유	J5	2200000
208	김해술	J5	2500000
215	대북혼	J5	3760000

DQL

SELECT 사용 예시 – 산술연산

컬럼 값에 대해 산술연산한 결과를 조회할 수 있음

```
SELECT EMP_NAME, SALARY*12, (SALARY+(SALARY*BONUS))*12
FROM EMPLOYEE;
```

EMP_NAME	SALARY*12	(SALARY+(SALARY*BONUS))*12
선동일	96000000	124800000
송송기	72000000	(null)
노옹철	44400000	(null)
송은희	33600000	(null)
유재식	40800000	48960000
정중하	46800000	(null)
박나라	21600000	(null)
하이유	26400000	29040000
김해술	30000000	(null)
심봉선	42000000	48300000
윤은해	24000000	(null)
전형돈	24000000	(null)
장쯔위	30600000	38250000
하동운	27840000	30624000
방명수	16560000	(null)
대북혼	45120000	(null)
차태연	33360000	40032000
전지연	43920000	57096000

DQL

SELECT 사용 예시 – 컬럼별칭

AS 키워드를 이용하여 컬럼 별칭을 지을 수 있음

SELECT EMP_NAME AS 이름, SALARY*12 “연봉(원)” FROM EMPLOYEE;

이름	연봉(원)
선동일	96000000
송송기	72000000
노옹절	44400000
송은희	33600000
유재식	40800000
정중하	46800000
박나라	21600000
하이유	26400000
김해술	30000000
심봉선	42000000
윤은해	24000000
전형돈	24000000
장쯔위	30600000

1. AS는 생략가능(공백으로 구분)
2. 숫자 혹은 특수문자가 있는 경우 반드시 “”를 사용해야 함

DQL

SELECT 사용 예시 – 리터럴

임의로 지정한 문자열을 SELECT 절에 사용하면, 테이블에 존재하는 데이터처럼 사용이 가능

SELECT EMP_NAME , SALARY, '원' AS 단위 FROM EMPLOYEE;

EMP_NAME	SALARY	단위
선동일	8000000	원
송송기	6000000	원
노옹철	3700000	원
송은희	2800000	원
유재식	3400000	원
정중하	3900000	원
박나라	1800000	원
하이유	2200000	원
김해술	2500000	원
심봉선	3500000	원
윤은해	2000000	원
전형돈	2000000	원
장쯔위	2550000	원

※ 리터럴은 Result Set의 모든 행에 반복 표시

DQL

SELECT 사용 예시 – DISTINCT

컬럼에 포함된 중복 값을 한번씩만 표시하고자 할 때 사용

```
SELECT JOB_CODE
FROM EMPLOYEE;
```

JOB_CODE
J1
J2
J2
J4
J3
J3
J7
J5
J5
J3
J7
J6
J6
J6
J7

```
SELECT DISTINCT JOB_CODE
FROM EMPLOYEE;
```

JOB_CODE
J2
J7
J3
J6
J5
J1
J4

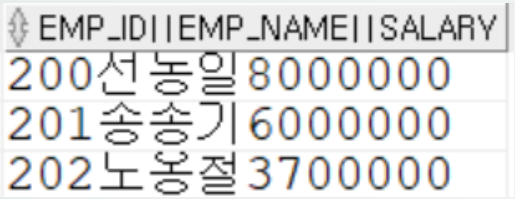
연산자

연산자

SELECT 연산자 - 연결연산자

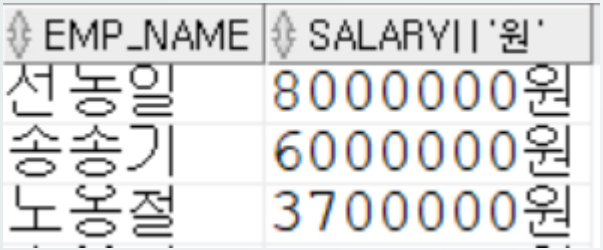
연결 연산자인 '||'를 사용하여 여러 컬럼을 하나의 컬럼인 것처럼 연결하거나, 컬럼과 리터럴을 연결할 수 있다.

SELECT EMP_ID || EMP_NAME || SALARY FROM EMPLOYEE;



EMP_ID||EMP_NAME||SALARY
200선농일8000000
201송송기6000000
202노옹절3700000

SELECT EMP_NANE, SALARY || '원' FROM EMPLOYEE;



EMP_NAME	SALARY||'원'
선농일	8000000원
송송기	6000000원
노옹절	3700000원

연산자

SELECT 연산자 – 논리 연산자

여러 개의 제한 조건 결과를 하나의 논리 결과로 만들어 준다.

연산자	설명
AND	여러 조건이 동시에 TRUE일 경우 TRUE
OR	여러 조건들 중 하나라도 TRUE일 경우 TRUE
NOT	조건에 대한 반대로 반환(NULL은 예외)

```
SELECT EMP_NAME, DEPT_CODE, SALARY FROM EMPLOYEE
WHERE DEPT_CODE='D6' AND SALARY>2000000
```

	EMP_NAME	DEPT_CODE	SALARY
1	송은희	D6	2800000
2	유재식	D6	3400000
3	정 중하	D6	3900000

연산자

SELECT 연산자 – 비교 연산자

표현식 사이의 관계를 비교하기 위해 사용하고, 비교 결과는 논리 결과(TRUE, FALSE, NULL 중 하나가 된다
단, 비교하는 두 컬럼 값/표현식은 서로 동일한 데이터 타입이어야 한다.

연산자	설명
=	같다
> , <	크다 / 작다
>= , =<	크거나 같다/ 작거나 같다
<> , != , ^=	같지 않다
BETWEEN AND	특정 범위에 포함되는지 비교
LIKE / NOT LIKE	문자 패턴 비교
IS NULL / IS NOT NULL	NULL 여부 비교
IN / NOT IN	비교 값 목록에 포함/미포함 여부 비교

연산자

SELECT 연산자 – 비교 연산자 – BETWEEN AND

비교하려는 값이 지정한 범위(경계 포함 → 이상/이하)에 포함되면 TRUE 리턴

[급여가 350만원이상 600만원 이하인 직원의 이름과 급여 출력]

```
SELECT EMP_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY BETWEEN
3500000 AND 6000000;
```

또는

```
SELECT EMP_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY >= 3500000
AND SALARY <= 6000000;
```

EMP_NAME	SALARY
송송기	6000000
노옹절	3700000
정숭하	3900000
심봉선	3500000
대북혼	3760000
전지연	3660000

연산자

SELECT 연산자 – 비교 연산자 – LIKE – 1

비교하려는 값이 지정한 특정 패턴을 만족시키면 TRUE를 리턴하는 연산자
'%'와 '_'를 와일드카드로 사용할 수 있다.

※ 와일드카드 : 아무 글자나 대체할 수 있는 문자
% : 글자 수 제한없이 대체
_ : 한 글자를 대체

[성이 전씨인 직원의 이름과 급여 출력]

```
SELECT EMP_NAME, SALARY
FROM EMPLOYEE
WHERE EMP_NAME LIKE '전%';
```

또는

```
SELECT EMP_NAME, SALARY
FROM EMPLOYEE
WHERE EMP_NAME LIKE '전_ _';
```

EMP_NAME	SALARY
전형돈	2000000
전지연	3660000

연산자

SELECT 연산자 – 비교 연산자 – LIKE – 2

[EMAIL 중 '_' 앞자리가 3자리인 직원 조회]

```
SELECT EMP_NAME, EMAIL
FROM EMPLOYEE
WHERE EMAIL LIKE '_ _ _ _%';
```

※ 와일드카드 문자와 패턴의 특수문자가 동일한 경우 어떤 것을 패턴으로 결정하는지 구분할 수 없음

EMP_NAME	EMAIL
선동일	sun di@kh.or.kr
송송기	song jk@kh.or.kr
노옹철	no hc@kh.or.kr
송은희	song eh@kh.or.kr
유재식	yoo js@kh.or.kr
정중하	jung jh@kh.or.kr
박나라	pack nr@kh.or.kr
하이유	ha iy@kh.or.kr
김해술	kim hs@kh.or.kr
심봉선	sim bs@kh.or.kr
윤은해	youn eh@kh.or.kr
전형돈	jun hd@kh.or.kr
장쯔위	jang zw@kh.or.kr
하동운	ha dh@kh.or.kr
방명수	bang ms@kh.or.kr
대북혼	dae bh@kh.or.kr
차태연	cha ty@kh.or.kr

연산자

SELECT 연산자 – 비교 연산자 – LIKE – 3

[EMAIL 중 '_' 앞자리가 3자리인 직원 조회]

```
SELECT EMP_NAME, EMAIL
FROM EMPLOYEE
WHERE
EMAIL LIKE '_ _ _ #_%' ESCAPE '#';
```

※ 와일드 카드를 동일하게 사용하고,
데이터 앞에 임의의 특수문자를 붙이고,
ESCAPE옵션을 붙이면 해당 문자
뒤의 '_'의 경우 와일드 카드가 아닌 문자패턴
으로 처리

EMP_NAME	EMAIL
선동일	sun di@kh.or.kr
유재식	yoo js@kh.or.kr
김해술	kim hs@kh.or.kr
심봉선	sim bs@kh.or.kr
전형돈	jun hd@kh.or.kr
대북혼	dae bh@kh.or.kr
차태연	cha ty@kh.or.kr
전지연	jun jy@kh.or.kr
이오리	loo or@kh.or.kr
이승석	lee js@kh.or.kr
유하진	yoo hj@kh.or.kr
이태림	lee tr@kh.or.kr

연산자

SELECT 연산자 – 비교 연산자 – NOT LIKE

['이'씨 성이 아닌 직원의 이름, 이메일 조회]

```
SELECT EMP_NAME, EMAIL
FROM EMPLOYEE
WHERE EMP_NAME NOT LIKE '이%';
```

또는

```
SELECT EMP_NAME, EMAIL
FROM EMPLOYEE
WHERE EMP_NAME NOT LIKE '이_ _';
```

EMP_NAME	EMAIL
선동일	sun di@kh.or.kr
송중기	song jk@kh.or.kr
노옹철	no hc@kh.or.kr
송은희	song eh@kh.or.kr
유재식	yoo js@kh.or.kr
정중하	jung jh@kh.or.kr
박나라	pack nr@kh.or.kr
하이유	ha iy@kh.or.kr
김해술	kim hs@kh.or.kr
심봉선	sim bs@kh.or.kr
윤은해	youn eh@kh.or.kr
전형돈	jun hd@kh.or.kr
장쯔위	jang zw@kh.or.kr
하동운	ha dh@kh.or.kr
방명수	bang ms@kh.or.kr
대북혼	dae bh@kh.or.kr
차태연	cha ty@kh.or.kr
전지연	jun jy@kh.or.kr

연산자

SELECT 연산자 – 비교 연산자 – IS NULL / IS NOT NULL

NULL 여부를 비교하는 연산자

[관리자도 없고 부서배치도 받지 않은 직원 이름 조회]

```
SELECT EMP_NAME, MANAGER_ID, DEPT_CODE
FROM EMPLOYEE
WHERE
MANAGER_ID IS NULL
AND DEPT_CODE IS NULL;
```

EMP_N...	MANAGER_ID	DEPT_CODE
하동운	(null)	(null)
이오리	(null)	(null)

[부서배치는 받지 않았지만 보너스는 받는 직원 이름 조회]

```
SELECT EMP_NAME, BONUS, DEPT_CODE
FROM EMPLOYEE
WHERE
DEPT_CODE IS NULL
AND BONUS IS NOT NULL;
```

EMP_NAME	BONUS	DEPT_CODE
하동운	0.1	(null)

연산자

SELECT 연산자 – 비교 연산자 – IN, NOT IN

비교하려는 값 목록에 일치하는 값이 있으면 TRUE를 반환하는 연산자

[D6부서와 D9 부서원들의 이름, 부서코드, 급여 조회]

```
SELECT
EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE
WHERE DEPT_CODE IN ('D6', 'D9');
```

또는

```
SELECT
EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE
WHERE DEPT_CODE = 'D6'
OR DEPT_CODE='D9'
```

EMP_NAME	DEPT_CODE	SALARY
선동일	D9	8000000
송종기	D9	6000000
노옹철	D9	3700000
송은희	D6	2800000
유재식	D6	3400000
정중하	D6	3900000

연산자

SELECT 연산자 우선순위

여러 연산자를 사용하는 경우 우선순위를 고려해서 사용해야 함

우선순위	연산자
1	산술연산자
2	연결 연산자
3	비교연산자
4	IS NULL/IS NOT NULL, LIKE/NOT LIKE, IN/NOT IN
5	BETWEEN AND
6	논리연산자 – NOT
7	논리연산자 – AND
8	논리연산자 – OR

연산자

SELECT 연산자 우선순위

[직급코드 J7 또는 J2 인 부서원 중 급여가 2000000원 보다 많이 받는 직원의 이름, 직급코드,급여 조회]

```
SELECT EMP_NAME, JOB_CODE, SALARY
FROM EMPLOYEE
WHERE JOB_CODE='J7'
OR JOB_CODE='J2'
AND SALARY>2000000;
```

EMP_NAME	JOB_CODE	SALARY
송종기	J2	6000000
노옹철	J2	3700000
박나라	J7	1800000
윤은해	J7	2000000
방명수	J7	1380000
이오리	J7	2890000

```
WHERE JOB_CODE='J7' OR (JOB_CODE='J2' AND SALARY>2000000);
```

연산자의 우선순위로 인해서 J2직급의 급여 2000000이상인직원 또는 J7직급인 직원으로 처리

연산자

SELECT 연산자 우선순위

[직급코드 J7 또는 J2 인 부서원 중 급여가 2000000원 보다 많이 받는 직원의 이름, 직급코드,급여 조회]

```
SELECT EMP_NAME, JOB_CODE, SALARY
FROM EMPLOYEE
WHERE (JOB_CODE='J7'
OR JOB_CODE='J2')
AND SALARY>2000000;
```

EMP_NAME	JOB_CODE	SALARY
송송기	J2	6000000
노옹절	J2	3700000
이오리	J7	2890000

```
WHERE (JOB_CODE='J7' OR JOB_CODE='J2') AND SALARY>2000000;
```

OR 연산이 먼저 처리될 수 있도록 ( )를 이용하여 우선순위를 변경