**SELECT 연산자와 표현식**

사칙연산 : +, -, \*, /

- 날짜형 데이터도 +, - 가능

문자열 결합연산자 : ||

(예, ‘a’ || ‘b’ > ‘ab’)

**SELECT** 1+1 , 1-1

**FROM** **DUAL**;

**SELECT** 1+1 plus\_test, 1-1 minus\_test

**FROM** **DUAL** ;

plus\_test와 minus\_test는 **ALIAS(별칭)**. **AS** 를 앞에 붙이기도 함

**SELECT** first\_name || ' ' || last\_name **AS** full\_name

**FROM** EMPLOYEES;

- 임시 컬럼명 설정이라 보면 된다. 조회결과 별칭 붙이기

**비교 연산자**

= 동등 연산자, 값이 같으면 참

!=, <> 비동등 연산자, 값이 틀리면 참

>, < 부등호 연산자. 예, 4 > 3 (참), 4 < 3 (거짓)

>= , <= 부등호 연산자

BETWEEN…AND 대상값 BETWEEN 값1 AND 값2

(대상값 >= 값1 AND 대상값 <= 값2)

AND AND 연산자

OR OR 연산자

NOT NOT 연산자

**SELECT** \*

**FROM** employees

**WHERE** salary **BETWEEN** 3000

**AND** 5000

**ORDER BY** salary;

**SELECT** \*

**FROM** employees

**WHERE** **NOT** ( salary = 2500)

**ORDER** **BY** salary;

**NULL 연산자**

* IS NULL, IS NOT NULL

**SELECT** \*

**FROM** 테이블명

**WHERE** 컬럼 **IS NULL**

**ORDER BY** 기준컬럼;

**LIKE : 문자열 비교**

last\_name **like** 'da%' : ‘da’로 시작하는 모든 데이터 값

- last\_name **like** '%d‘ : “d”로 끝나는 모든 데이터 값

- '%' 는 모든 문자를 의미

**SELECT** \*

**FROM** employees

**WHERE** phone\_number **LIKE '%9'**;

**SELECT** \*

**FROM** employees

**WHERE** phone\_number **LIKE '%124%'** ;

**IN** : 컬럼 IN (값1, 값2, 값3, …) > OR와 동일한 동작

**SELECT** \*

**FROM** employees

**WHERE** JOB\_ID **IN** ('IT\_PROG', 'AD\_VP', 'FI\_ACCOUNT');

(Job\_id 컬럼에 (~) 값이 있는 것들만 보여줘라)

**SELECT** \*

**FROM** employees

**WHERE** JOB\_ID **NOT IN** ('IT\_PROG', 'AD\_VP', 'FI\_ACCOUNT');

**기타 연산자**

집합(SET) 연산자 : UNION, UNION ALL, INTERSECT, MINUS, …

EXISTS

**SQL 표현식 (expression)**

**CASE 표현식**

IF ~ THEN ~ ELSE 로직을 구현한 표현식

단순형 / 검색형

반환하는 값의 데이터 유형이 모두 같아야 함 ( 값1,값2,~,값n 까지)

THEN과 ELSE 다음은 값 뿐만 아니라 표현식이 올 수 있음

ELSE는 생략이 가능

**1) 단순형**

**CASE** expr **WHEN** 비교표현식1 **THEN** 값1

**WHEN** 비교표현식2 **THEN** 값2 ….

**ELSE** 값n

**END**

expr이 비교표현식1 과 같으면 값1, 비교표현식2와 같으면 값2를 반환

어느 비교표현식과도 같지 않으면 값n을 반환

**2) 검색형**

**CASE** **WHEN** 조건1 **THEN** 값1

**WHEN** 조건2 **THEN** 값2 ….

**ELSE** 값n

**END**

조건1을 만족하면 값1, 조건2를 만족하면 값2를 반환

어느 조건도 만족하지 않으면 ELSE 다음의 값n을 반환

**단순형** 예시

**SELECT** country\_id

,country\_name

,**CASE** region\_id **WHEN** 1 **THEN** '유럽'

**WHEN** 2 **THEN** '아메리카'

**WHEN** 3 **THEN** '아시아'

**WHEN** 4 **THEN** '중동 및 아프리카'

**END** **AS** region\_name

**FROM** countries;

**검색형** 예1

**SELECT** employee\_id, first\_name, last\_name, salary, job\_id

,**CASE** **WHEN** salary **BETWEEN** 1 **AND** 5000 **THEN** '낮음'

**WHEN** salary **BETWEEN** 5001 **AND** 10000 **THEN** '중간'

**WHEN** salary **BETWEEN** 10000 **AND** 15000 **THEN** '높음'

**ELSE** '최상위'

**END** salary\_rank

**FROM** employees;

**의사컬럼 (pseudocolumn)**

테이블에 있는 컬럼 처럼 동작하지만 실제 컬럼은 아닌 가상 컬럼

· **ROWNUM** 의사컬럼

쿼리 수행 후 조회된 각 로우에 대해 그 순서를 가리키는 수를 반환

주로 where 절에서 쿼리 결과로 반환되는 로우 수를 제어할 때 사용

입력된 순으로 컬럼이 순서가 정해지는 것 같다.

**SELECT** employee\_id, first\_name, last\_name, **rownum**

**FROM** employees;

**SELECT** employee\_id, first\_name, last\_name, rownum

**FROM** employees

WHERE **rownum <= 5**;

- 파이썬의 .head() 처럼 쓸 수 있다.

**TIP**

SELECT \*

FROM EMPLOYEES

WHERE **1=1**

-- AND JOB\_ID = "~"

AND JOB\_ID = "~~";

AND JOB\_ID = "~~~"

;

여러 조건을 쉽게 바꾸면서 보기 위해서 이렇게 쓰는 경우가 있다! (TIP)