

## 목차

1. LIKE
  - 1.1 정확하게 찾고 싶은 문자 표현
  - 1.2 Wildcard character(%,\_ ) 사용하는 표현
  - 1.3 특정 단어를 원하지 않을 경우 표현
2. 필드 결합 표현
3. 공백 제거 표현
4. 문자 함수
5. 날짜 함수

# 1.LIKE

## 1.1 정확하게 찾고 싶은 문자를 표현한 경우

```
SELECT FIRST_NAME, LAST_NAME  
FROM EMPLOYEES  
WHERE LAST_NAME LIKE 'Mallin';
```

	FIRST_NAME	LAST_NAME
1	Jason	Mallin

'' 사이에 찾고 싶은 문자를 정확하게 입력하여 문자를 찾는다. '' 사이에선 대소문자가 구별되기 때문에 구별하여 정확히 표현해야 한다. EMPLOYEES 참조 테이블에서 LAST\_NAME 이 'Mallin'인 것을 탐색하여 FIRST\_NAME, LAST\_NAME 열을 함께 출력하였다. 총 1가지 행만 탐색된 것을 보아 LAST\_NAME이 Mallin인 사람은 한 명만 있다는 것을 알 수 있다.

## 1.LIKE

### 1.2

Wildcard character: 문자열에서 특정한 문자 또는 문자들을 대체하기 위해 사용하는 문자

\*Wildcard character 중 '%'를 사용하는 경우

- 1) 뒤에 나오는 문자들을 알 수 없는 경우      2) 앞에 나오는 문자들을 알 수 없는 경우

```
SELECT COUNTRY_ID,COUNTRY_NAME,REGION_ID
FROM COUNTRIES
WHERE COUNTRY_NAME LIKE 'C%';
```

	COUNTRY_ID	COUNTRY_NAME	REGION_ID
1	CA	Canada	2
2	CN	China	3

COUNTRIES 참조 테이블에서 'C'뒤에 글자가 있는 COUNTRY\_NAME 에 대하여 COUNTRY\_ID,REGION\_ID,COUNTRY\_NAME 을 검색하였더니 2가지의 행이 검색되었다.

```
SELECT DEPARTMENT_NAME,DEPARTMENT_ID
FROM DEPARTMENTS
WHERE DEPARTMENT_NAME LIKE '%e';
```

	DEPARTMENT_NAME	DEPARTMENT_ID
1	Executive	90
2	Finance	100

DEPARTMENTS 참조 테이블에서 'e' 앞에 글자가 있는 DEPARTMENT\_NAME 에 대하여 DEPARTMENT\_NAME,DEPARTMENT\_ID를 검색하였더니 2가지의 행이 검색되었다.

# 1.LIKE

\* Wildcard character 중 '%'를 사용하는 경우

3)앞뒤에 나오는 문자들을 알수 없는 경우

```
SELECT EMPLOYEE_ID, JOB_ID
FROM JOB_HISTORY
WHERE JOB_ID LIKE '%ACC%';
```

	EMPLOYEE_ID	JOB_ID
1	101	AC_ACCOUNT
2	200	AC_ACCOUNT

JOB\_HISTORY 참조 테이블에서 JOB\_ID에 'ACC'가 포함된 행에 대하여 EMPLOYEE\_ID와 JOB\_ID를 출력하였더니 2가지의 행이 검색되었다.

4)시작과 끝의 한글자만 아는 경우

```
SELECT JOB_ID, JOB_TITLE
FROM JOBS
WHERE JOB_ID LIKE 'A%R';
```

	JOB_ID	JOB_TITLE
1	AC_MGR	Accounting Manager

JOB\_HISTORY 참조 테이블에서 JOB\_ID에서 A로 시작해서 R로 끝나는 행에 대하여 JOB\_ID와 JOB\_TITLE을 검색하였더니 1가지 행이 검색되었다.

# 1.LIKE

\*Wildcard character 중 '\_'를 사용하는 경우

1)뒤에 나오는 한글자를 알수 없는 경우

```
SELECT REGION_ID,REGION_NAME
FROM REGIONS
WHERE REGION_NAME LIKE 'Asi_';
```

	REGION_ID	REGION_NAME
1	3	Asia

REGIONS 참조 테이블에서 REGION\_NAME 이 Asi\_로 마지막 한글자만 모르는 형태로 이에 해당되는 행에 대하여 REGION\_ID,REGION\_NAME을 출력한 결과 1 행이 나왔고, REGION\_NAME이 Asia 임을 알 수 있다.

2)앞에 나오는 한글자를 알수 없는 경우

```
SELECT CITY,STATE_PROVINCE
FROM LOCATIONS
WHERE CITY LIKE '_oronto';
```

	CITY	STATE_PROVINCE
1	Toronto	Ontario

LOCATIONS 참조 테이블에서 CITY가 앞의 한글자를 제외하고 oronto 인 행에 대하여 CITY와 STATE\_PROVINCE를 출력하였더니 1 가지의 행이 검색되었다. Toronto 가 CITY 임을 알 수 있다.

# 1.LIKE

## 1.3 특정 단어를 원하지 않는 경우

```
SELECT DEPARTMENT_ID, DEPARTMENT_NAME  
FROM DEPARTMENTS  
WHERE DEPARTMENT_NAME NOT LIKE '%e%';
```

	DEPARTMENT_ID	DEPARTMENT_NAME
1	10	Administration
2	30	Purchasing
3	50	Shipping
4	60	IT
5	110	Accounting
6	170	Manufacturing
7	180	Construction
8	190	Contracting
9	210	IT Support
10	220	NOC
11	270	Payroll

DEPARTMENTS 참조테이블에서  
DEPARTMENT\_NAME에 e 가 들어가지 않는  
행에 대하여 DEPARTMENT\_ID,  
DEPARTMENT\_NAME 을 검색하였더니 총 11  
가지의 행이 검색되었다.

## 2.필드 결합 표현

### 1) 여러 열을 하나로 삽입하기

```
SELECT FIRST_NAME || LAST_NAME  
FROM EMPLOYEES;
```

FIRST_NAME  LAST_NAME	
1 EllenAbel	96 WinstonTaylor
2 SundarAnde	97 SigalTobias
3 MozheAtkinson	98 PeterTucker
4 DavidAustin	99 OliverTuvault
5 HermannBaer	100 Jose ManuelUrman
6 ShelliBaida	101 PeterVargas
7 AmitBanda	102 ClaraVishney
8 ElizabethBates	103 ShantaVollman
9 SarahBell	104 AlanaWalsh
10 DavidBernstein	105 MatthewWeiss
11 LauraBissot	106 JenniferWhalen
12 HarrisonBloom	107 EleniZlotkey
13 AlexisBull	
14 AnthonyCabrio	
15 GeraldCambrault	
16 NanetteCambrault	
--	

EMPLOYEES 참조테이블에서 FIRST\_NAME  
과 LAST\_NAME의 열을 합쳐서 하나의 열로  
출력한 결과 총 107 개의 행이 출력되었다.

### 2)문자를 삽입해서 별칭과 함께 결합하기

```
SELECT EMPLOYEE_ID || '의 입사일: ' || START_DATE AS START_INFO  
FROM JOB_HISTORY;
```

START_INFO	
1	101의 입사일: 97/09/21
2	101의 입사일: 01/10/28
3	102의 입사일: 01/01/13
4	114의 입사일: 06/03/24
5	122의 입사일: 07/01/01
6	176의 입사일: 06/03/24
7	176의 입사일: 07/01/01
8	200의 입사일: 95/09/17
9	200의 입사일: 02/07/01
10	201의 입사일: 04/02/17

JOB\_HISTORY 참조테이블에서 EMPLOYEE\_ID  
와 START\_DATE 를 하나의 열로 합쳐서 출력  
하는데, 열의 이름이 EMPLOYEE\_ID ||  
START\_DATE 가 아니라 별칭인 START\_INFO  
로 출력하였고 총 10개의 행이 출력되었다.

### 3.공백 제거 표현

\*FROM DUAL 은 오라클에서 제공하는 간단한 연산 메모장 역할을 하는 테이블이다.

#### 1)오른쪽 공백 제거하기

#### 2)왼쪽 공백 제거하기

#### 3)양쪽 공백 제거하기

```
SELECT RTRIM('SQL DEVELOPER ')  
AS REVIEW FROM DUAL;
```

REVIEW
1 SQL DEVELOPER

DUAL로 생성한 테이블에 'SQL DEVELOPER ' 의 오른쪽 끝의 공백을 제거하고 REVIEW라는 열의 이름으로 출력하였다.

```
SELECT RTRIM(' ERROR:404')  
AS REVIEW FROM DUAL;
```

REVIEW
1 ERROR:404

DUAL로 생성한 테이블에 ' ERROR:404' 의 왼쪽 끝의 공백을 제거하고 REVIEW라는 열의 이름으로 출력하였다.

```
SELECT RTRIM(' 컴퓨터공학과 4학년 ')  
AS REVIEW FROM DUAL;
```

REVIEW
1 컴퓨터공학과 4학년

DUAL로 생성한 테이블에 ' 컴퓨터공학과 4학년 ' 의 양쪽 끝의 공백을 제거하고 REVIEW라는 열의 이름으로 출력하였다.



## 4.문자함수: UPPER,REPLACE,INITCAP,CONCAT

### 1)UPPER

```
SELECT JOB_ID,JOB_TITLE  
FROM JOBS  
WHERE UPPER(JOB_TITLE)='PRESIDENT';
```

JOB_ID	JOB_TITLE
AD_PRES	President

JOBS 참조테이블에서 JOB\_TITLE이 대소문자가 섞여있는 형태이어서, UPPER 함수로 모두 대문자로 변환한 뒤 PRESIDENT에 해당되는 열에 대하여 JOB\_ID,JOB\_TITLE을 검색하였더니 1가지 열이 검색되었다. 검색할시에만 대문자로 바꾸는 것으로 원래 테이블에는 영향이 가지 않는다.

### 2)REPLACE

```
SELECT REPLACE('바다로 떠나요','바다','산') "CHANGES"  
FROM DUAL;
```

CHANGES
1 산로 떠나요

DUAL로 생성한 테이블에 '바다로 떠나요'에서 바다를 산으로 바꾼 형태를 CHANGE 라는 열의 이름으로 REPLACE 함수를 활용하여 출력하였다.

### 3)INITCAP

```
SELECT INITCAP('Say hello to ewha') "CHANGES"  
FROM DUAL;
```

CHANGES
1 Say Hello To Ewha

DUAL로 생성한 테이블에서 'Say hello to ewha' 문장을 단어의 첫부분만 대문자로 되도록 INITCAP 함수를 활용하여 CHANGES 열의 이름으로 출력하였다.

### 4)CONCAT

```
SELECT CONCAT('k-','pop')  
FROM DUAL;
```

CONCAT('K-','POP')
1 k-pop

DUAL로 생성한 테이블에서 'k-'와'pop' 문장을 CONCAT 함수를 활용하여 'k-pop'의 단어로 이었고 이를 출력하였다.

# 문자함수: INSTR,SUBSTR

## 5)INSTR

```
SELECT INSTR('강원도,독도','도')
FROM DUAL;
```

INSTR('강원도,독도','도')
3

DUAL로 생성한 테이블에서 '강원도, 독도'의 문장에서 INSTR 함수를 활용하여 '도'가 처음 찾아지는 위치를 출력하였다. '도'가 여러 개 있을 경우 모든 개수의 '도'를 탐색하지 못하는 것을 알 수 있다.

## 6)SUBSTR

```
SELECT SUBSTR('자료구조,컴퓨터구조',1)
FROM DUAL;
```

SUBSTR('자료구조,컴퓨터구조',1)
1 자료구조,컴퓨터구조

DUAL로 생성한 테이블에서 '자료구조,컴퓨터구조'의 문장을 SUBSTR 함수를 통해 1의 값으로 처음부터 출력하였다. -1 이라면 마지막 값만을 출력한다.

# 날짜함수: CURRENT\_DATE, CURRENT\_TIMESTAMP, MONTHS\_BETWEEN

## 1)CURRENT\_DATE

```
SELECT SESSIONTIMEZONE, CURRENT_DATE
FROM DUAL;
```

SESSIONTIMEZONE	CURRENT_DATE
1 Asia/Seoul	21/05/20

DUAL로 생성한 테이블에 SESSIONTIMEZONE( 현재 시간의 위치), CURRENT\_DATE( 현재 날짜) 를 출력하였다.

## 2)CURRENT\_TIMESTAMP

```
SELECT SESSIONTIMEZONE, CURRENT_TIMESTAMP
FROM DUAL;
```

SESSIONTIMEZONE	CURRENT_TIMESTAMP
1 Asia/Seoul	21/05/20 22:36:12.623000000 ASIA/SEOUL

DUAL로 생성한 테이블에 SESSIONTIMEZONE( 현재 시간의 위치), CURRENT\_TIMESTAMP( 현재 시/분/초 시간) 을 출력하였다.

## 3)MONTHS\_BETWEEN

```
SELECT MONTHS_BETWEEN('21/05/20','18/03/03')
FROM DUAL;
```

MONTHS_BETWEEN('21/05/20','18/03/03')
1 38.5483870967741935483870967741935483871

DUAL로 생성한 테이블에 MONTHS\_BETWEEN 함수를 활용하여 21/05/20 과 18/03/03 사이의 몇 달이 있는지 출력하였다. 날짜에서 '/'는 한국권, '-'미국권의 표현이다.