🔴 데이터베이스

eXERD

논리 모드 - 사람이 인식하기 쉽게 (한글이나 알아보기 쉬운 단어로 표현)

물리 모드 - 데이터베이스 모드 (실제 컬럼명)

스키마 : 식별자 이름

관계를 맺는 이유 : 데이터의 유효성을 검사하기 위해

관계	특징
식별 관계	• 부모테이블 기본키(PK)가 자식테이블의 외래키(FK)이자 기본키(PK)로 사용되는 관계 • 자식테이블의 행을 추가할 때 부모테이블의 참조 행이 없다면 자식테이블의 행을 추가 할 수 없음 • 실선으로 나타냄
비식별 관계	• 부모테이블 기본키(PK)가 자식테이블의 일반컬럼이나 외래키(FK)컬럼에 저장되는 관계 • 자식테이블의 행을 추가할 때 부모테이블의 참조 행이 없어도 자식테이블의 행을 추가 할 수 있음 • 점선으로 나타냄

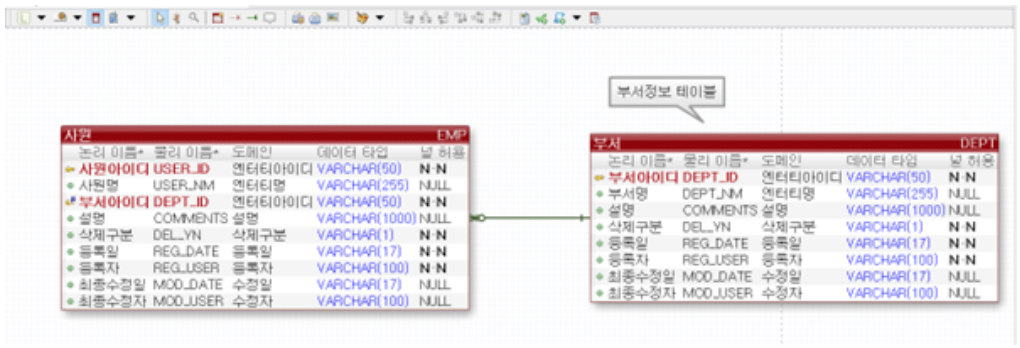
<단축키>

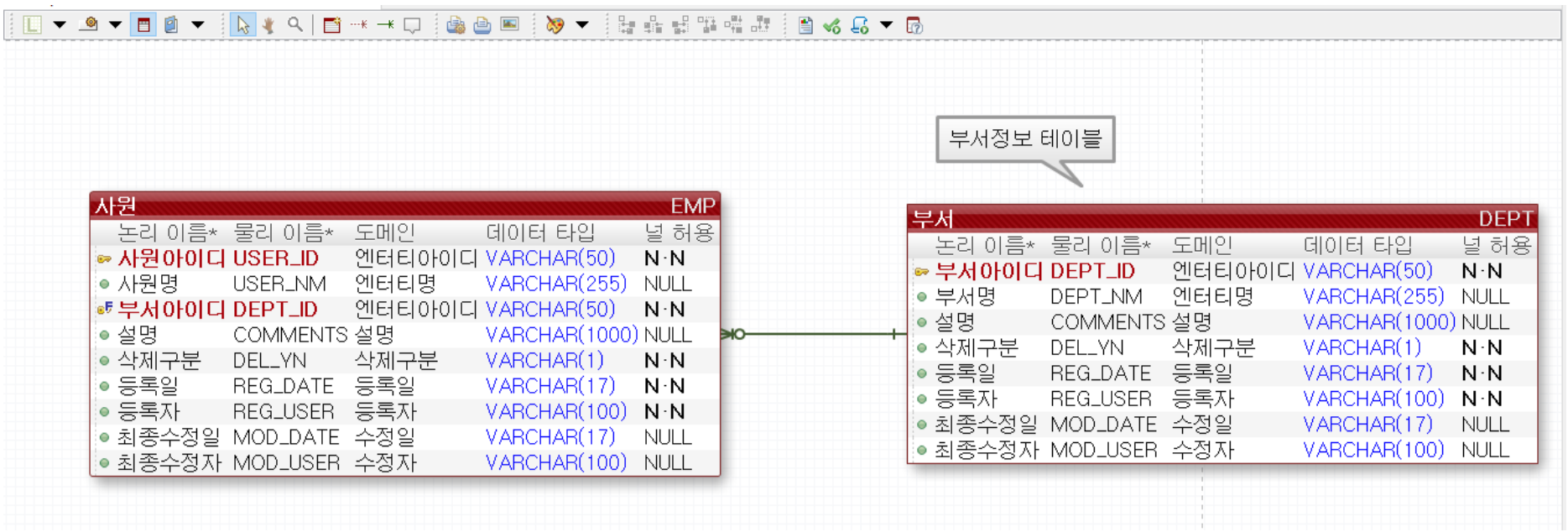
다른 테이블에서 같은 컬럼 추가 : Ctrl + 마우스로 이동

컬럼 추가 : Ctrl + Enter

PK 컬럼 추가 : Ctrl + Shift + Enter

정리 : Ctrl + Shift + F





- eXERD 테이블들과 iip 데이터 연관 확인해보기
-

DBeaver

인터페이스 아이디(INTERFACE_ID) : PK 컬럼 (중복 불가)

ex) F@00000092 -> 고객이 알지 못함 / 시스템에 따른 인터페이스 아이디

인터페이스 명(INTERFACE_NM) : 화면에도 보임 / 일반적인 인터페이스 명

ex) A_B_001에 짝 지어진 이름

인티그레이션 아이디(INTEGRATION_ID) : 고객이 식별하는 인터페이스 아이디

ex)A_B_001 ->고객이 원하는 아이디

INTERFACE_ID	INTERFACE_NM	INTEGRATION_ID
F@00000092	temp_test	temp_test
F@00000090	MRRTHDKT00	MRRTHDKT00
F@00000091	PRDYHDKT10	PRDYHDKT10

인터페이스명	인터페이스ID
(환경)광주광역시_대구광역시…	A007_B005_B006_D…
(농업)강원도_경기도_DB연계	A001_B001_B002_D…
교통정보 조회	A004_B009_B001_AP…

INTERFACE_ID	INTERFACE_NM	INTEGRATION_ID
F@00000092	temp_test	temp_test
F@00000090	MRRTHDKT00	MRRTHDKT00
F@00000091	PRDYHDKT10	PRDYHDKT10

인터페이스명	인터페이스ID
(환경)광주광역시_대구광역시…	A007_B005_B006_D…
(농업)강원도_경기도_DB연계	A001_B001_B002_D…
교통정보 조회	A004_B009_B001_AP…

<단축키>

쿼리문 작성 후 실행 : Ctrl + Enter

전체 스크롤 : Ctrl + A

전체 실행 : Alt + X

<DBeaver 쿼리문>

❗INSERT 문

1) **INSERT INTO** EMP1 ('uhaha','uhaha','2');

2) **INSERT INTO** EMP1 (
EMP_ID,
EMP_NM,
DEPT_ID
) **VALUES**(
'whoana',
'whoana',
'1'
);

	EMP_ID	EMP_NM	DEPT_ID
1	whoana	whoana	1
2	uhaha	uhaha	2
3	ming	ming	3
4	hong	hong	4

❗SELECT 문

1) **SELECT** *
FROM DEPT1 a; —모든 컬럼을 DEPT1에서 조회

	DEPT_ID	DEPT_NM
1	1	경영
2	2	지원
3	3	IT
4	4	개발
5	5	지원

2) **ORDER BY** DEPT_ID **ASC**, EMP_NM **DESC** —아이디를 오름차순, 이름을 내림차순

EMP_ID	EMP_NM	DEPT_ID
whoana	whoana	1
whoana	whoana	10
kyubin	kyubin	11
kkl	kkl	12
kkb	kkb	13

3) **WHERE** DEPT_NM **LIKE** '%지' —’지’로 끝나는 것

	DEPT_ID	DEPT_NM

4) **WHERE** DEPT_NM **LIKE** '%지%' —’지’가 들어가는 것

	DEPT_ID	DEPT_NM
1	2	지원
2	5	지원

5) **WHERE** DEPT_NM **LIKE** '지%' —’지’로 시작하는 것

	DEPT_ID	DEPT_NM
1	2	지원
2	5	지원

6) **WHERE** DEPT_ID = '3' —아이디가 3인 것

7) **SELECT**

```
a.DEPT_ID,
a.DEPT_NM
FROM DEPT1 a
WHERE a.DEPT_ID > '3';
```

	DEPT_ID	DEPT_NM
1	4	개발
2	5	지원

❗INNER JOIN 문

1) **SELECT**

```
a.EMP_ID,
a.emp_nm,
b.DEPT_ID,
b.DEPT_NM
FROM EMP1 a
INNER JOIN DEPT1 b
ON a.DEPT_ID = b.DEPT_ID
AND a.EMP_ID = 'whoana'
ORDER BY a.EMP_NM ASC;
```

EMP_ID	EMP_NM	DEPT_ID	DEPT_NM
whoana	whoana	1	경영

DOCKER

CMD 창에서 명령어

Docker –v : 버전 확인

Docker ps : 어떤 파일이 있는지 확인

Docker images : 이미지 파일을 확인

이미지 가져오기 : docker pull jaspeen/oracle-xe-11g

컨테이너 실행 (id : system / passwd : oracle) :

docker run –name oracle11g –d –p 8080:8089 –p 1521:1521 jaspeen/oracle-xe-11g

Sqlplus 접속 : docker exec –it oracle11g sqlplus

```
C:₩Windows₩system32>docker exec -it frosty_perlman sqlplus

SQL*Plus: Release 11.2.0.2.0 Production on Wed Jan 18 00:34:14 2023

Copyright (c) 1982, 2011, Oracle.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
```

버전 확인 : SQL> select instance_name, version, status from v$instance;

```
SQL> select instance_name, version, status from v$instance;

INSTANCE_NAME    VERSION           STATUS
---------------- ----------------- ------------
XE               11.2.0.2.0        OPEN
```

컨테이너 bash 접속 : docker exec –it oracle11g bash

```
C:₩Windows₩system32>docker exec -it frosty_perlman bash
root@91e5a92d9efb:/# ls -l
total 92
drwxr-xr-x     1 root root 4096 Nov 26  2015 bin
drwxr-xr-x     2 root root 4096 Apr 10  2014 boot
drwxr-xr-x     5 root root  340 Jan 18 00:28 dev
-rwxr-xr-x     1 root root 3002 Nov 26  2015 entrypoint.sh
drwxr-xr-x     1 root root 4096 Jan 18 00:30 etc
drwxr-xr-x     2 root root 4096 Apr 10  2014 home
drwxr-xr-x     1 root root 4096 Oct 28  2015 lib
drwxr-xr-x     2 root root 4096 Oct 28  2015 lib64
drwxr-xr-x     2 root root 4096 Oct 28  2015 media
drwxr-xr-x     2 root root 4096 Apr 10  2014 mnt
drwxr-xr-x     2 root root 4096 Oct 28  2015 opt
-rwxr-xr-x     1 root root  824 Nov 26  2015 oracle-install.sh
dr-xr-xr-x  2113 root root    0 Jan 18 00:28 proc
drwx------     1 root root 4096 Jan 18 00:34 root
drwxr-xr-x     1 root root 4096 Oct 28  2015 run
drwxr-xr-x     1 root root 4096 Nov 26  2015 sbin
drwxr-xr-x     2 root root 4096 Oct 28  2015 srv
dr-xr-xr-x    11 root root    0 Jan 18 00:28 sys
drwxrwxrwt     1 root root 4096 Jan 18 00:30 tmp
drwxr-xr-x     1 root root 4096 Nov 26  2015 u01
drwxr-xr-x     1 root root 4096 Nov 10  2015 usr
drwxr-xr-x     1 root root 4096 Nov 10  2015 var
root@91e5a92d9efb:/#
```

<테이블 이름 규칙>

TSU0301

1) T : table을 의미

2) TSU0301 → 공통코드 : 화면에서 쓰는 한글명들을 코드화

Ex) 송신, 수신 같은 것을 코드화 / 1번은 송신, 2번은 수신 / 송수신은 데이터가 2개 뿐

3) 송수신처럼 데이터가 별로 없을 때 테이블을 만들기보다 공통코드에 집어넣음

4) SU : support

6) AN : analysis (인터페이스 분석)

7) IM : 개발 관련(데이터 처리흐름, 앱 처리방식 등)

	LEVEL1	LEVEL2	CD	LEVEL1_NM	LEVEL2_NM	NM	NM2
1	AN	10	0	분석	IO구분	INPUT	INPUT
2	AN	10	1	분석	IO구분	OUTPUT	OUTPUT
3	AN	11	0	분석	데이터출력형식	xml	xml
4	AN	11	1	분석	데이터출력형식	json	json
5	AN	11	2	분석	데이터출력형식	delimiter	delimiter
6	AN	11	3	분석	데이터출력형식	fixedlength	fixedlength
7	AN	12	0	분석	배포상태	배포중	배포중
8	AN	12	1	분석	배포상태	배포실패	배포실패
9	AN	12	2	분석	배포상태	배포성공	배포성공
10	AN	13	L	분석	정렬기준	왼쪽정렬	왼쪽
11	AN	13	R	분석	정렬기준	오른쪽정렬	오른쪽
12	AN	14	0	분석	인터페이스유형(현I	Routing	Routing
13	AN	14	1	분석	인터페이스유형(현I	Mapping	Mapping
14	AN	14	2	분석	인터페이스유형(현I	FilePut	FilePut
15	AN	14	3	분석	인터페이스유형(현I	FileGet	FileGet
16	CO	01	0	공통	결재아이템유형	요건라이프	요건라이프사
17	CO	01	1	공통	결재아이템유형	장애대장처	장애대장처리
18	IM	01	0	개발	DATA처리흐름	단방향	단방향
19	IM	01	1	개발	DATA처리흐름	양방향	양방향

Level 1 : 대분류 / Level 2 : 소분류, 구체적인 분류, 최종 분류

JOIN문 실습

INNER JOIN	• 기준 테이블과 조인 테이블 모두 데이터가 존재해야 조회됨 • 반드시 필요한 값이 있을 때에는 사용 • 서로 키 값이 있는 테이블을 조인할 때 사용
LEFT OUTER JOIN	• 기준 테이블에만 데이터가 존재해도 조회됨 • 왼쪽테이블은 데이터가 무조건 있어야 하고 오른쪽테이블은 데이터가 없어도 됨

- 코드 테이블

	INTERFACE_NM	INTEGRATION_ID	APP_PR_METHOD	DATA_PR_METHOD	DATA_PR_DIR
1	temp_test	temp_test	0	0	0
2	MRRTHDKT00	MRRTHDKT00	0	0	0
3	PRDYHDKT10	PRDYHDKT10	0	0	0
4	ATAASAAW71	ATAASAAW71	1	0	0
5	ATABAABT01	ATABAABT01	0	1	1
6	ATACCACT01	ATACCACT01	1	0	0
7	ATADEADT01	ATADEADT01	0	1	1
8	ATAEAAES01	ATAEAAES01	1	0	0
9	ATAFMAFS01	ATAFMAFS01	0	1	1

- 이름 테이블

	LEVEL1	LEVEL2	COMMENTS	CD	NM	NM2
1	IM	01	개발-DATA처리흐름-단방향	0	단방향	단방향
2	IM	01	개발-DATA처리흐름-양방향	1	양방향	양방향
3	IM	02	개발-APP처리방식-동기	0	동기	동기
4	IM	02	개발-APP처리방식-비동기	1	비동기	비동기
5	IM	12	개발-DATA처리방식-배치	0	배치	배치
6	IM	12	개발-DATA처리방식-온라인	1	온라인	온라인
7	IM	12	개발-DATA처리방식-온라인전송	4	온라인전송	온라인전송
8	IM	12	개발-DATA처리방식-온라인조회	3	온라인조회	온라인조회
9	IM	12	개발-DATA처리방식-요청	2	요청	요청
10	IM	12	개발-DATA처리방식-전체계좌	5	전계좌	전계좌

<코드 테이블의 코드에 맞게 이름 테이블에서 이름을 가져오기>

```
SELECT
    a.INTERFACE_NM
    ,a.INTEGRATION_ID
    ,a.APP_PR_METHOD --APP처리방식
    ,cd1.NM AS APP_PR_METHOD_NM
    ,a.DATA_PR_METHOD --데이터처리방식
    ,cd2.NM AS DATA_PR_METHOD_NM
    ,a.DATA_PR_DIR --데이터처리방향
    ,cd3.NM AS DATA_PR_DIR_NM
    FROM TAN0201 a
LEFT OUTER JOIN TSU0301 cd1 ON a.APP_PR_METHOD = cd1.CD AND cd1.LEVEL1 = 'IM' AND cd1.LEVEL2 = '02'
LEFT OUTER JOIN TSU0301 cd2 ON a.DATA_PR_METHOD = cd2.CD AND cd2.LEVEL1 = 'IM' AND cd2.LEVEL2 = '12'
LEFT OUTER JOIN TSU0301 cd3 ON a.DATA_PR_DIR = cd3.CD AND cd3.LEVEL1 ='IM' AND cd3.LEVEL2 ='01'
      //WHERE a.DEL_YN = 'N' --무조건 있어야함(실무에서)
      //AND a.REG_USER ='iip'
     //AND a.INTEGRATION_ID LIKE 'U%'
    //AND a.APP_PR_METHOD_NM = '0' --항상 코드 값으로 날라옴(한글로 안날라옴) / LIKE문을 썼을 때 느리니까 WHERE문은 코드 값으로
    //ORDER BY a.MOD_DATE DESC; --최근 수정된 기준으로
```

- 결과

	ID	APP_P	APP_PR_METHOD_NM	DATA	DATA_PR_METHOD_NM	DATA_PI	DATA_PR_DIR_NM
1		0	동기	0	배치	0	단방향
2		0	동기	0	배치	0	단방향
3		0	동기	0	배치	0	단방향
4		1	비동기	0	배치	0	단방향
5		0	동기	1	온라인	1	양방향

<SQL 문>

1. UNION : 두 개의 테이블을 하나로 만드는 연산 / 두 개 테이블의 컬럼 수, 컬럼 데이터 형식이 모두 일치해야한다.
    - UNION : 두 개의 테이블을 합치면서 중복 데이터를 제거 / 정렬을 발생 시킴
    - UNION ALL : 두 개의 테이블을 합치면서 중복제거 없이 전부 보여줌
2. 시퀀스 : 일련의 연속적인 사건들 / 사건이나 행동 등의 순서
3. COMMIT / ROLLBACK
    - COMMIT : 보류 중인 모든 데이터 변경사항을 영구적으로 적용 / 현재 트랜잭션 종료
    - ROLLBACK : 보류 중인 모든 데이터 변경사항을 폐기 / 현재 트랜잭션 종료 / 직전 커밋 직후의 단계로 회귀
4. DROP / DELETE / TURNCATE
    - DROP : 테이블 삭제 / 삭제 후 되돌릴 수 없음
    - DELETE : 레코드(데이터)만 삭제 / 테이블 용량은 줄어 들지 않음 / 삭제 후 되돌릴 수 있음
    - TURNCATE : 용량이 줄어들고 인덱스 등도 모두 삭제 / 테이블이 삭제되지는 않지만 데이터가 한번에 삭제 됨 / 삭제 후 되돌릴 수 없음

숙제 > ENP테이블과 REG_DATE (VARCHAR(14 or 17)) , DEPT테이블에 REG_DATE (VARCHAR(14 or 17)) 컬럼을 추가하고 sysdate로 문자열로 변환해서 insert문 작성해서 emp와 dept 데이터를 추가로 하나 넣어보기

- 도커 시간 변경 : ln -snf /usr/share/zoneinfo/Asia/Seoul /etc/localtime
- 도커 시간 확인 : date

```
//초까지만(14자리)
INSERT INTO EMP (
        EMP_ID,
        EMP_NM,
        DEPT_ID,
        EMP_REG_DATE
) VALUES(
        'whoana',
        'whoana',
        '10',
        to_char(SYSDATE, 'YYYYMMDDHH24MISS')
);

INSERT INTO DEPT (
        DEPT_ID
        ,DEPT_NM
        ,DEPT_REG_DATE
)VALUES (
        '6'
        ,'총무'
        ,to_char(SYSDATE,'YYYYMMDDHH24MISS')
);

//밀리초까지(17자리)
INSERT INTO EMP (
        EMP_ID,
        EMP_NM,
        DEPT_ID,
        EMP_REG_DATE
) VALUES(
        'kyubin',
        'kyubin',
```

```
        '11',
        to_char(SYSTIMESTAMP, 'YYYYMMDDHH24MISSFF3')
);

INSERT INTO DEPT (
        DEPT_ID
        ,DEPT_NM
        ,DEPT_REG_DATE
)VALUES (
        '11'
        ,'기획'
        ,to_char(SYSDATE,'YYYYMMDDHH24MISS')
);
```

2023.01.27

테이블(시스템)을 이해하자

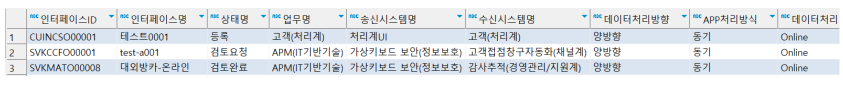
	테이블명
인터페이스	TAN0201
사용자	TSU0101
공통코드	TSU0301

- 물리 이름이 ID로 끝나는 컬럼은 연계 테이블이 존재한다.
- 정보가 들어가 있는 컬럼은 TSU0301(공통코드)에 존재한다.
- 등록일, 등록자명 등이 들어있는 데이터는 TSU0101(사용자)에 존재한다.

<숙제>

1. 인터페이스 리스트 조회

goodluck/인터페이스조회-공통코드조인.sql at master · whoana/goodluck
신입사원교육. Contribute to whoana/goodluck development by creating an account on GitHub.
https://github.com/whoana/goodluck/blob/master/resources/sql/인터페이스조회-공통코드조인.sql

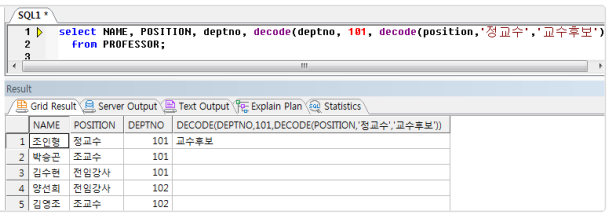


2. 인터페이스 상세 조회

2023.01.31

<decode 함수>

[SQL] 조건문 - DECODE, CASE~WHEN~THEN
조건문 - DECODE DECODE(A, B, '1', null) :: A 가 B 일 경우 '1'을, 아닐 경우 null(생략 가능) DECODE(A, B, '1', '2') :: A 가 B 일 경우 '1'을, 아닐 경우 '2' DECODE(A, B, '1', C, '2', '3') :: A 가 B 일 경우 '1'을, A 가 C 일 경우
https://data-make.tistory.com/20



사용법	뜻
DECODE(A, B, ‘1’, null)	A = B → 1 / A ≠ B → null
DECODE(A, B, ‘1’, ‘2’)	A = B → 1 / A ≠ B → 2
DECODE(A, B, ‘1’, C, ‘2’, ‘3’)	A = B → 1 / A = C → 2 / A ≠ B && A ≠ C → 3
DECODE(A, B, DECODE(C, D, ‘1’, null))	A = B → C = D → 1 / A = B → C ≠ D → null
DECODE(A, B, DECODE(C, D, ‘1’, ‘2’))	A = B → C = D → 1 / A = B → C ≠ D → 2

- INNER JOIN (무조건 있는 데이터)

ex) 인터페이스 ID, 인티그레이션 ID

- LEFT OUTER JOIN (있을 수도 없을 수도 있는 것)

  ex) 코드값(TSU0301)

*리스트 조회 할 때 ORDER BY를 무조건 준다 → 가독성을 위해

<SQL문 실습>

1. MAX() 함수

```
SELECT *
FROM TAN0201
WHERE
    REG_DATE = (SELECT max(REG_DATE) FROM TAN0201);   --값이 하나만 표현 됨(GROUP BY를 주지 않으면)
```

→ 등록일이 가장 큰(최신)인 데이터를 보여줌

	INTERFACE_ID	INTERFACE_NM	INTEGRATION_ID	REG_DATE	REG_USER
1	F@00009549	등록테스트_20230126_03	NISCTPO00001	20230126151611578	TEST201

2. MAX(), MIN() 함수

```
SELECT max(DATA_PR_METHOD), MIN(DATA_PR_METHOD) FROM TAN0201;
```

→ 데이터방식 중 가장 큰 값과 가장 작은 값을 보여줌

	MAX(DATA_PR_METHOD)	MIN(DATA_PR_METHOD)
1	4	0

```
SELECT
    max(DATA_PR_METHOD)
    ,MIN(DATA_PR_METHOD)
    ,max(REG_DATE)
    ,min(REG_DATE)
FROM TAN0201;
```

	MAX(DATA_PR_METHOD)	MIN(DATA_PR_METHOD)	MAX(REG_DATE)	MIN(REG_DATE)
1	4	0	20230126151611578	20200709145414

3. GROUP BY() 함수

```
-- 2022년 이후로 등록된 일자별 인터페이스 건수를 구해보기
SELECT
    SUBSTR(REG_DATE,1,8)  --GROUP BY에 들어간 절은 결과로 나올 수 있음
    ,COUNT(INTERFACE_ID)
    ,max(REG_USER) --같은 날 등록한 사람 중 문자가 더 큰 사람
    ,MOD_USER  --그룹함수가 아니기 때문에 오류가 남
        FROM TAN0201
        WHERE REG_DATE >= '20220101' || '000000000' --뒤의 '0'들은 시간.분.초.밀리를 표현
    GROUP BY SUBSTR(REG_DATE,1,8), MOD_USER
    ORDER BY 1 DESC;
```

	SUBSTR(REG_DATE,1,8)-	COUNT(INTERFACE_ID)	MAX(REG_USER)	MOD_USER
1	20230126	3	TEST201	TEST201
2	20230125	1	shl	shl
3	20230120	1	TEST201	TEST201
4	20230120	2	shl	shl

→ SUBSTR(REG_DATE,1,8)은 REG_DATE의 문자열을 1번부터 8번까지 끊어서 보여주도록하는 함수

4. JOIN

- 인터페이스 : **SELECT** * **FROM** tan0201; → 9,152개
- 담당자 : **SELECT** * **FROM** tan0219; → 82개

1) INNER JOIN

```
SELECT a.INTERFACE_ID , b.USER_ID
  FROM TAN0201 a
INNER JOIN TAN0219 b ON a.INTERFACE_ID = b.INTERFACE_ID;
```

	INTERFACE_ID	USER_ID
1	F@00000381	eai
2	F@00000381	user1
3	F@00000381	user1
4	F@00000396	eai

→ 82개 (양쪽 테이블에 모두 데이터가 있어야만 조인)

2) LEFT OUTER JOIN

```
SELECT a.INTERFACE_ID , b.USER_ID
  FROM TAN0201 a
LEFT OUTER JOIN TAN0219 b ON a.INTERFACE_ID = b.INTERFACE_ID;
```

	INTERFACE_ID	USER_ID
79	F@00000437	[NULL]
80	F@00000438	iip
81	F@00000438	iip
82	F@00000439	[NULL]

→ 9,197개 (왼쪽 테이블에만 데이터가 있으면 조인 가능)

3) RIGHT OUTER JOIN

```
SELECT a.INTERFACE_ID , b.USER_ID
  FROM TAN0201 a
RIGHT OUTER JOIN TAN0219 b ON a.INTERFACE_ID = b.INTERFACE_ID ;
```

	INTERFACE_ID	USER_ID
1	F@00000381	eai
2	F@00000381	user1
3	F@00000381	user1
4	F@00000396	eai

→ 82개 (오른쪽 테이블에만 데이터가 있으면 조인 가능)

4) FULL OUTER JOIN

```
SELECT a.INTERFACE_ID , b.USER_ID
  FROM TAN0201 a
FULL OUTER JOIN TAN0219 b ON a.INTERFACE_ID = b.INTERFACE_ID;
```

	INTERFACE_ID	USER_ID
1	F@00000381	user1
2	F@00000381	user1
3	F@00000381	eai
4	F@00000396	user1

→9,172개 (모든 데이터들을 다 조인)

5. UNION

```
SELECT '삭제된', INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID ='F@00000487'
UNION
SELECT '안삭제된', INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID IN ('F@000004867','F@00000508');
```

	'삭제된'	INTERFACE_ID
1	삭제된	F@00000487
2	안삭제된	F@00000508

→ 중복 제거

```
SELECT INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID ='F@00000487'
UNION ALL
SELECT INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID IN ('F@00000487','F@00000508');
```

	INTERFACE_ID
1	F@00000487
2	F@00000487
3	F@00000508

→ 중복 제거하지 않음

6. MINUS

```
SELECT INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID ='F@00000487'
MINUS
SELECT INTERFACE_ID FROM TAN0201 WHERE INTERFACE_ID IN ('F@00000487','F@00000508');
```

	INTERFACE_ID

→ 같은 데이터를 빼줌

7. INSERT문

```
-- INSERT 실습
INSERT INTO tsu0101
SELECT
    'kyulee' AS USER_ID
    ,PASSWORD
    ,USER_NM
    ,CELL_PHONE
    ,PHONE
    ,EMAIL
    ,COMPANY_NM
    ,DEPARTMENT_NM
    ,POSITION_NM
    ,ROLE_ID
    ,DEL_YN
    ,REG_DATE
    ,REG_USER
    ,MOD_DATE
    ,MOD_USER
FROM tsu0101
WHERE USER_ID = 'whoana';
```

	USER_ID	PASSWORD	USER_NM
1	TEST201	TEST201	TEST201
2	TEST202	TEST202	TEST202
3	kyulee	whoana	whoana

▼ 기억하면 좋은 내용

테이블 조회 = 다이어그램으로 (02.인터페이스연관정보가 중요)

<개발 관련 커뮤니티>

꿈꾸는 개발자, DBA 커뮤니티 구루비

조건에 만족하는 값들의 사이 를 조회 하는 쿼리 확인 부탁드립니다 52 서브쿼리는 limit 없이 외부에서 잘라도 될까요? 100 헥사곤 위치정보로 거리 구하기 (Presto SQL) 268 AWS RDS for Oracle 계정 복사 257 datafile 삭제 983 Unique 제약 조건 질문 1,024 두 날짜 사이 해당되는 날짜 불러오기 (level, connect by) 1,156 스케줄링 질문 1,096 NOT EXISTS 두번 사용 시 오류가 발생합니다.

http://www.gurubee.net/