

## 40. DBMS ဆိုတဲ့ (၇)

\* ဤအားလုံး

JDBC API

Database

[datetime]

(ii) yyyy-MM-dd hh:mm:ss

getDate()  $\Rightarrow$  java.sql.Date yyyy-MM-dd

getTimestamp()  $\Rightarrow$  java.sql.Timestamp yyyy-MM-dd hh:mm:ss,

# 41. 외부키(foreign key) 활용

myapp-users

user_id	name	email	pwd	created_at
1	aaa	-	-	-
2	bbb	-	-	-
3	ccc	-	-	-

myapp-projects

project_id	Title	description	startdate	enddate	members
101	p1	-	-	-	1,3
102	p2	-	-	-	1,2
103	p3	-	-	-	1,2,3

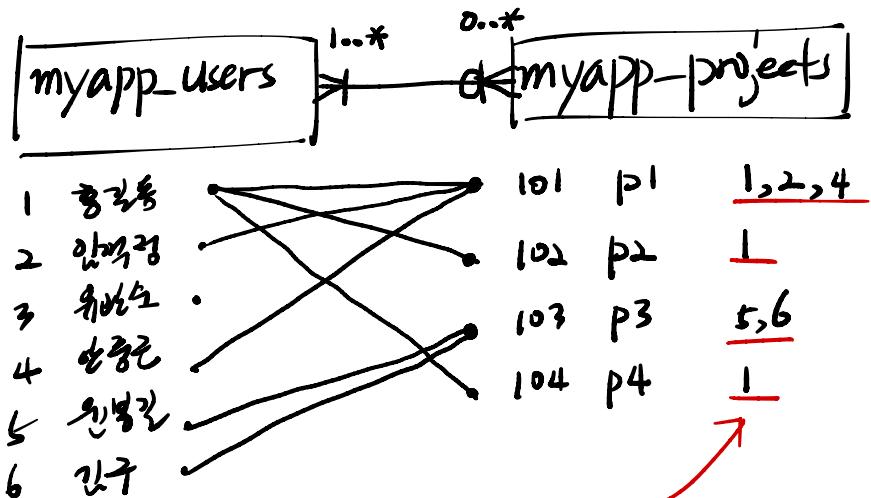
1번은 1번

프로젝트에 참여하는  
사람은 반드시 있는  
필수값입니다!

↑  
필수값이 까지다!  
↳ 필수값입니다!

## 41. 외부키(foreign key) 활용

1) 데이터의 데이터간에 관계를 짜는  
(cf 대 cf 관계)



(cf 대 cf)  
(cf 대 cf 관계)

외부 키와 함께 관계의 흐름을  
제어하는 관계를 데이터에 막을 수 있다

관련된 관계를 데이터에 막을 수 있다

Foreign key 활용  
+  
관련된 관계  
관련 관계로 관계를 막을 수 있다

제어?

\* 데이터를 관리하는 틀

- 1 흐름
- 2 일정
- 3 주제

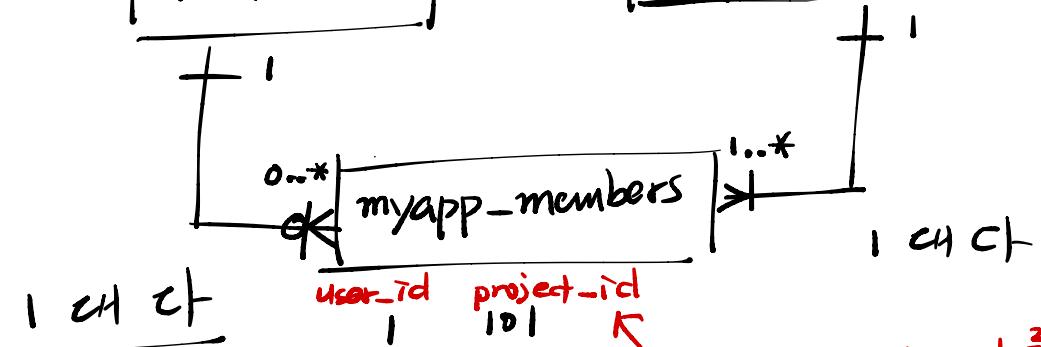
myapp-users	
1	2

- 1 흐름
- 2 일정
- 3 주제

myapp-projects	
101	p1
102	p2
103	p3

\* 외부키 (foreign key)

- 다른 테이블의 pk를 가리킨다
- 같은 pk를 사용할 수 있다



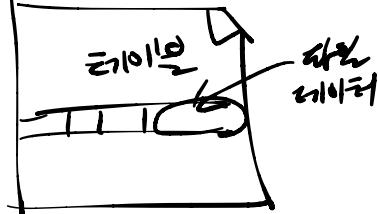
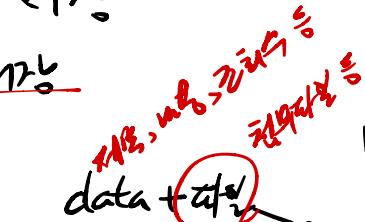
1 CH C1

	user_id	project_id
1	101	
1	103	
2		101
2		102
2		103

다른 테이블의 pk를 → 가리킨다  
"Foreign key"

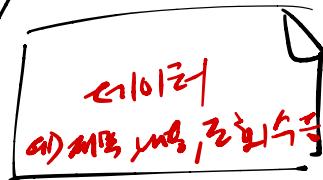
\* DB의 특성 짚기

① 데이터가 구조화된



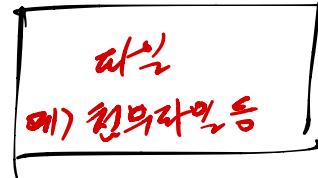
② 자료를 분리해서 저장

데이터베이스의  
구조화된



구조화된  
구조화된

구조화된



데이터베이스가  
구조화된 자료로  
구조화된 자료로



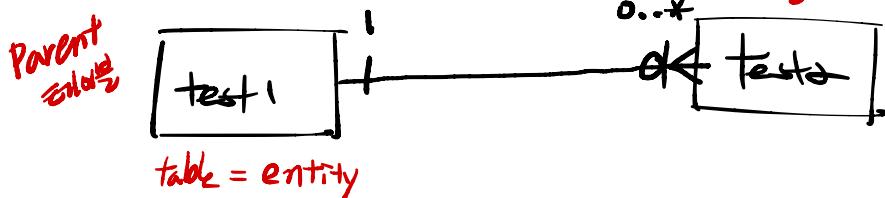
자료는 데이터  
구조화된 자료



자료는 알고 쓰는  
모의 구조화



44. child table  
(Foreign key)



ERD  
Information Engineering  $\rightarrow$  IEs  
(IE  $\rightarrow$  DB)

pk	no	title	content	rdt
1	aaa	-	-	.
2	bbb	-	-	.
3	ccc	-	-	.
4	ddd	-	-	.
5	eee	-	-	.

row = record

pk	fno	filepath	bno	fk	test1 (no)
1	101	a1.gif	1		
2	102	a2.gif	1		
3	103	a3.gif	1		
4	104	b1.gif	5		
5	105	b2.gif	5		
6	106	x.gif	00		test1의 no 00은 오류.

\* from → where → select → order by  
①

```
mysql> select * from room;
+----+-----+-----+-----+
| rno | loc      | name   | qnty  |
+----+-----+-----+-----+
| 1  | 강남      | 501    | 30    |
| 2  | 강남      | 502    | 30    |
| 3  | 강남      | 503    | 30    |
| 4  | 종로      | 301    | 30    |
| 5  | 종로      | 302    | 30    |
| 6  | 종로      | 303    | 30    |
| 7  | 서초      | 301    | 30    |
| 8  | 서초      | 302    | 30    |
| 9  | 서초      | 501    | 30    |
| 10 | 서초     | 601    | 30    |
+----+-----+-----+-----+
```

select rno, name, concat(loc, ' ', name) as name2  
① from room  
order by loc asc, name2 asc;

\* from → where → color by → order by  
②

```
mysql> select * from room;
+----+-----+-----+-----+
| rno | loc      | name    | qnty   |
+----+-----+-----+-----+
| 1  | 강남      | 501    | 30     |
| 2  | 강남      | 502    | 30     |
| 3  | 강남      | 503    | 30     |
| 4  | 종로      | 301    | 30     |
| 5  | 종로      | 302    | 30     |
| 6  | 종로      | 303    | 30     |
| 7  | 서초      | 301    | 30     |
| 8  | 서초      | 302    | 30     |
| 9  | 서초      | 501    | 30     |
| 10 | 서초      | 601    | 30     |
+----+-----+-----+-----+
```

```
select rno, name, concat(loc, ' ', name) as name2
① from room
order by loc asc, name2 asc;
```

\* from → where → select → order by

③ ↑ (가장 최종적인 결과 표시)

mysql> select \* from room;

(  
① select rno, name, concat(loc, '-', name) as name2  
from room  
order by loc asc, name2 asc;  
② ↑ (가장 최종적인 결과 표시)  
③ — (가장 최종적인 결과 표시)

rno	loc	name	qnty	name2 (loc - name)
1	강남	501	30	강남-501
2	강남	502	30	강남-502
3	강남	503	30	강남-503
4	종로	301	30	종로-301
5	종로	302	30	종로-302
6	종로	303	30	종로-303
7	서초	301	30	서초-301
8	서초	302	30	서초-302
9	서초	501	30	서초-501
10	서초	601	30	서초-601

\* from → where → select → order by

④  
③ select rno, name, concat(loc, ' ', name) as name2  
② from room  
① order by loc asc, name2 asc;

rno	loc	name	qnty	name2
1	강남	501	30	강남-501
2	강남	502	30	강남-502
3	강남	503	30	강남-503
7	서초	301	30	서초-301
8	서초	302	30	서초-302
9	서초	501	30	서초-501
10	서초	601	30	서초-601
4	종로	301	30	종로-301
5	종로	302	30	종로-302
6	종로	303	30	종로-303

\* from → where → select → order by → 결과 추출 ⑤

rno	name	name2
1	501	강남-501
2	502	강남-502
3	503	강남-503
7	301	서초-301
8	302	서초-302
9	501	서초-501
10	601	서초-601
4	301	종로-301
5	302	종로-302
6	303	종로-303

③ select rno, name, concat(loc, '-', name) as name2  
① from room  
④ order by loc asc, name2 asc;

select command  
선택한 결과의 값을  
결과 데이터로 추출한다

\* JOIN - Cross Join = cartesian join

board1

bno	title	content
1	제목1	내용
2	제목2	내용
3	제목3	내용

attach\_file1

fno	filepath	bno
101	a1.gif	1
102	a2.gif	1
103	c1.gif	3

select board1.bno, title, content, fno, filepath, attach\_file1.bno  
from board1 cross join attach\_file1;

bno	title	content	fno	filepath	bno
3	제목3	내용	101	a1.gif	1
2	제목2	내용	101	a1.gif	1
1	제목1	내용	101	a1.gif	1
3	제목3	내용	102	a2.gif	1
2	제목2	내용	102	a2.gif	1
1	제목1	내용	102	a2.gif	1
3	제목3	내용	103	c1.gif	3
2	제목2	내용	103	c1.gif	3
1	제목1	내용	103	c1.gif	3

\* JOIN - natural join  $\Rightarrow$  같은 이름의 컬럼 값을 기준으로 데이터를 합친다  
 board1 attach\_file

bno	title	content		fno	filepath	bno
1	제목1	내용	a	101	a1.gif	1
2	제목2	내용	a	102	a2.gif	1
3	제목3	내용	a	103	c1.gif	3

```
select b.bno, title, content, fno, filepath, a.bno
  -> from board1 b natural join attach_file1 a;
```

bno	title	content	fno	filepath	bno
1	제목1	내용	101	a1.gif	1
1	제목1	내용	102	a2.gif	1
3	제목3	내용	103	c1.gif	3

\* JOIN - natural join  $\Rightarrow$  하기!

```
select * from board2;
```

no	title	content	o
1	제목1	내용	o
2	제목2	내용	o
3	제목3	내용	o

```
select * from attach_file2;
```

no	filepath	bno
101	a1.gif	1
102	a2.gif	1
103	c1.gif	3

select b.no, title, content, a.no, filepath, bno  
-> from board2 b natural join attach\_file2 a;  
Empty set (0.00 sec)

내부집합을 허용하지 않기 때문에 조인할 수 있다.

## \* JOIN - join ~ using ('주소', ...)

```
select * from board4;
```

bno	title	content
1	제목1	내용
2	제목2	내용
3	제목3	내용

```
select * from attach_file4;
```

fno	title	bno
1	a1.gif	1
2	a2.gif	1
3	c1.gif	3

이런 조인 방식을 ~준으로  
이연 조인 방식을 지정할 수 있다.

```
select b.bno, b.title, content, a.fno, a.title, a.bno
from board4 b join attach_file4 a using (bno);
```

bno	title	content	fno	title	bno
1	제목1	내용	1	a1.gif	1
1	제목1	내용	2	a2.gif	1
3	제목3	내용	3	c1.gif	3

\* JOIN - join ~ using (bno, ...)  $\Rightarrow$  헷갈기!

↑ 같은 이름을 가진 컬럼이 있다.

```
select * from board5;
```

no	title	content
1	제목1	내용
2	제목2	내용
3	제목3	내용

```
select * from attach_file5;
```

fno	filepath	bno
1	a1.gif	1
2	a2.gif	1
3	c1.gif	3



board 테이블에  
bno 컬럼이  
없다!  
or!

✓ select no, title, content, fno, filepath, bno  
  -> from board5 b join attach\_file5 a using (bno);  
ERROR 1054 (42S22): Unknown column 'bno' in 'from clause'

\* JOIN - join ~ on 헷  
(inner join)

```
select * from board5;
```

no	title	content
1	제목1	내용
2	제목2	내용
3	제목3	내용

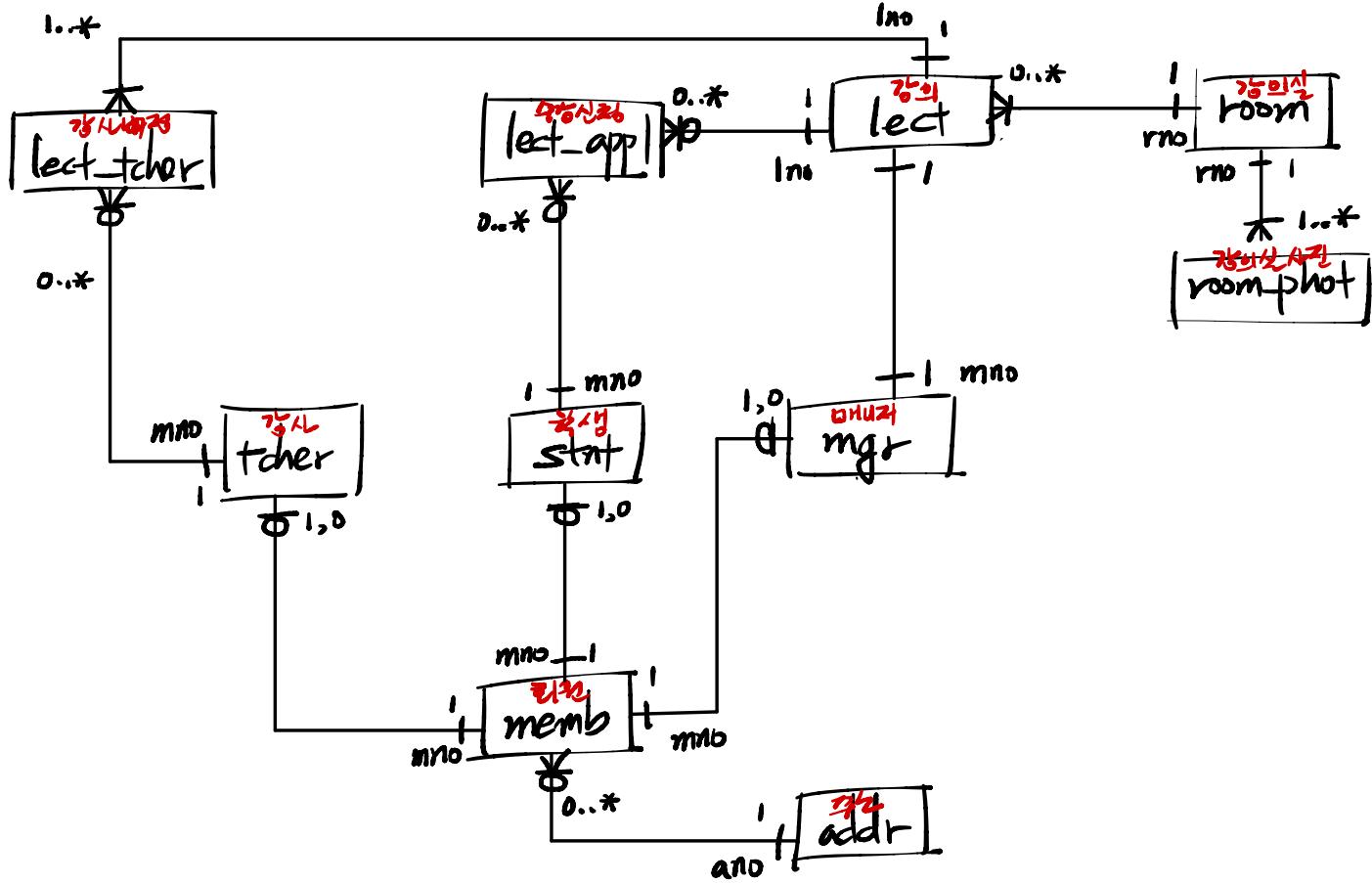
```
select * from attach_file5;
```

fno	filepath	bno
1	a1.gif	1
2	a2.gif	1
3	c1.gif	3

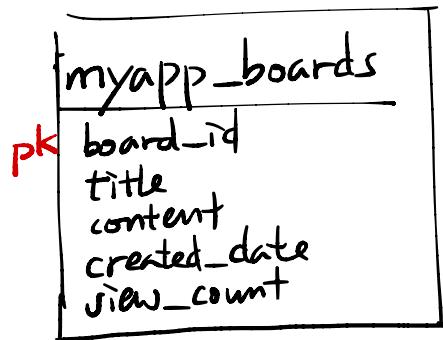
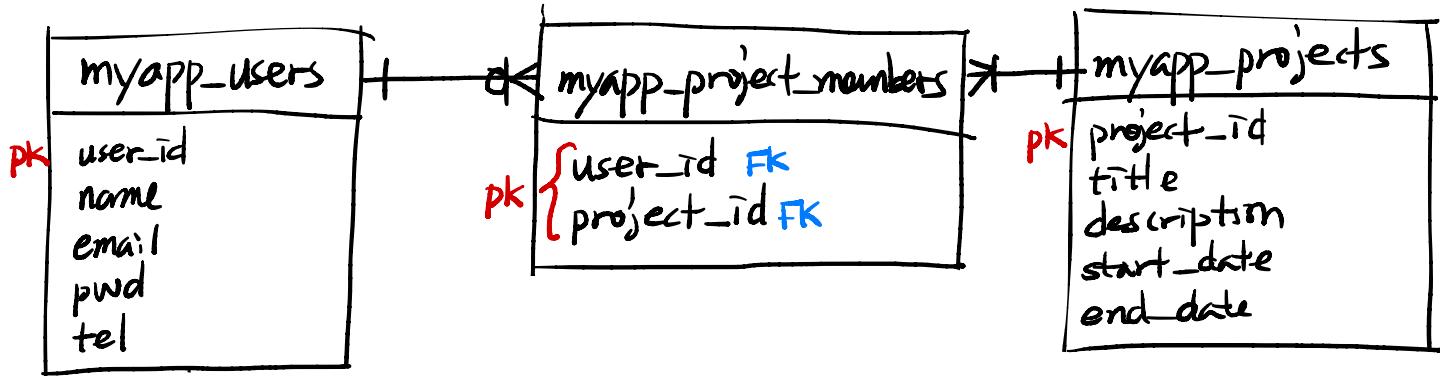
```
select no, title, content, fno, filepath, bno  
from board5 b join attach_file5 a on b.no=a.bno;
```

no	title	content	fno	filepath	bno
1	제목1	내용	1	a1.gif	1
1	제목1	내용	2	a2.gif	1
3	제목3	내용	3	c1.gif	3

## \* JOIN 예제 - ERD (Entity Relationship Diagram)

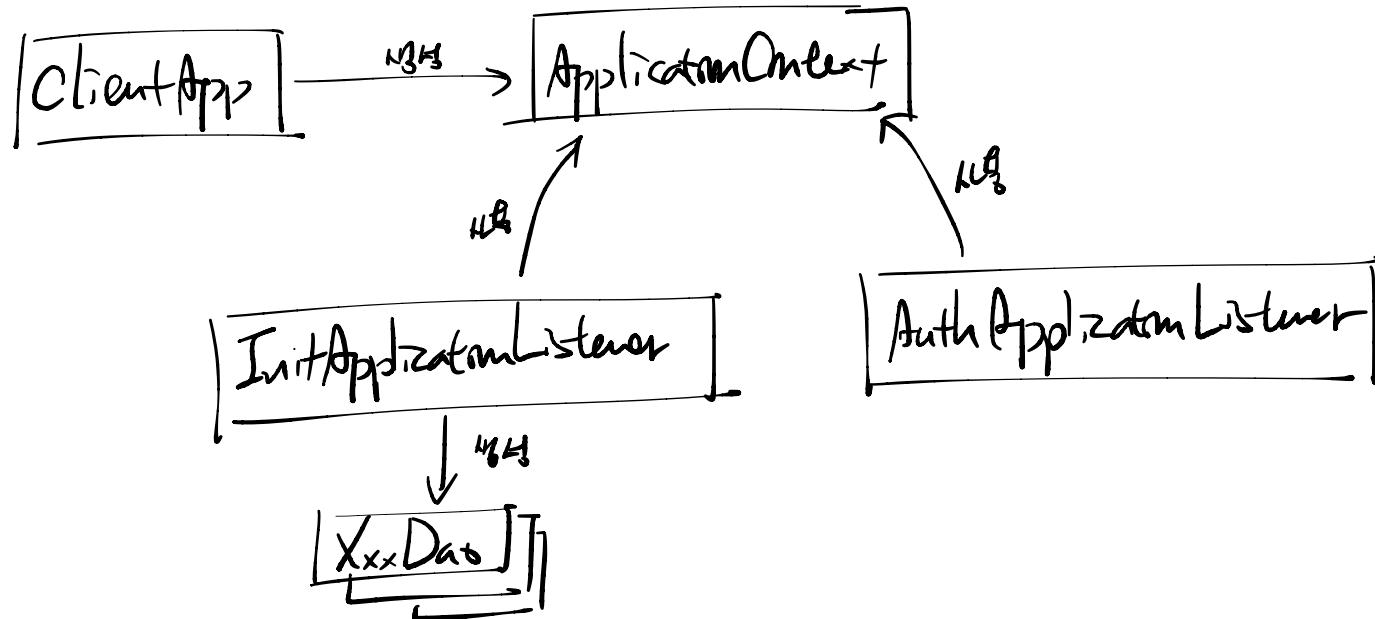


## \* myapp ERD



42. 201 / 206 22.8.861

① ApplicationListener چیست چه کاری دارد



\*  $x_1x_1\frac{1}{2} + x_2x_2\frac{1}{2}$

