13강 계좌 상세보기를 위한 쿼리 공부

1. 주소 설계

> 계좌 상세보기
> /account/1
> /account/1?gubun=all
> /account/1?gubun=withdraw
> /account/1?gubun=deposit

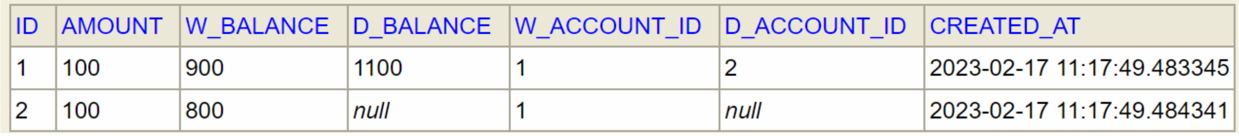
스프링 컨트롤러 쿼리스트링 디폴트 값 받는법

```
@GetMapping("/account/{id}")
public String detail(@PathVariable int id, @RequestParam(name = "gubun",
defaultValue = "all") String gubun) {
    // 1. 인증체크

    // 2. 레파지토리 호출 (메서드를 3개 or 마이바티스 동적쿼리)

    return "account/detail";
}
```

2. 기본 쿼리 연습

2.1 출금내역 보기

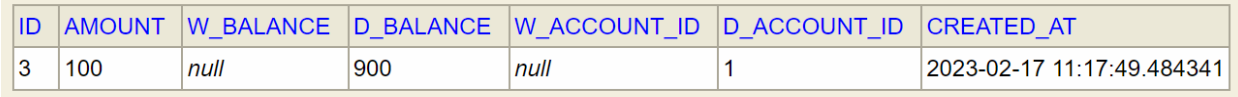
```
select * from history_tb where w_account_id = 1
```



ID	AMOUNT	W_BALANCE	D_BALANCE	W_ACCOUNT_ID	D_ACCOUNT_ID	CREATED_AT
1	100	900	1100	1	2	2023-02-17 11:17:49.483345
2	100	800	*null*	1	*null*	2023-02-17 11:17:49.484341

2.2 입금내역 보기

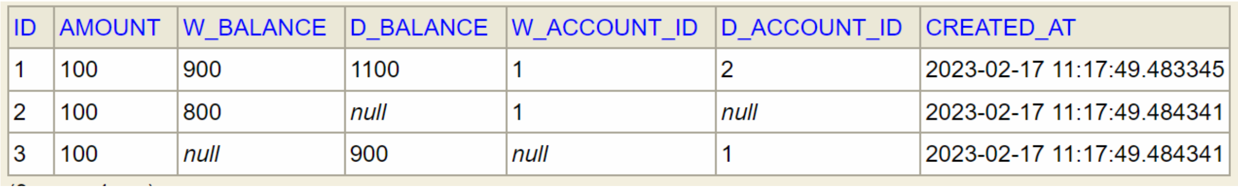
```
select * from history_tb where d_account_id = 1
```



ID	AMOUNT	W_BALANCE	D_BALANCE	W_ACCOUNT_ID	D_ACCOUNT_ID	CREATED_AT
3	100	*null*	900	*null*	1	2023-02-17 11:17:49.484341

2.3 입출금내역 보기

```
select * from history_tb where w_account_id = 1 or d_account_id = 1
```

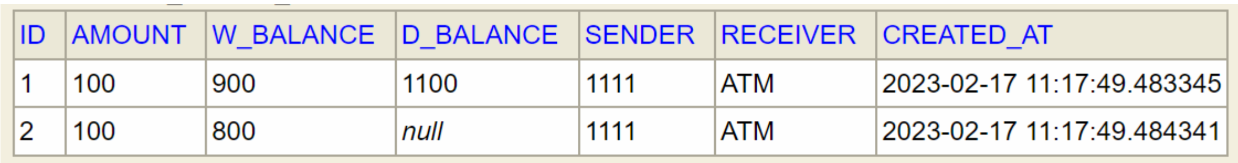


ID	AMOUNT	W_BALANCE	D_BALANCE	W_ACCOUNT_ID	D_ACCOUNT_ID	CREATED_AT
1	100	900	1100	1	2	2023-02-17 11:17:49.483345
2	100	800	*null*	1	*null*	2023-02-17 11:17:49.484341
3	100	*null*	900	*null*	1	2023-02-17 11:17:49.484341

3. 조인 쿼리 작성

3.1 출금내역 보기

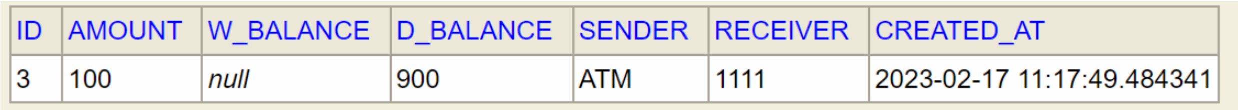
```
SELECT ht.id, ht.amount, ht.w_balance, ht.d_balance, wat.number sender, 'ATM'
receiver, ht.created_at
FROM history_tb ht
INNER JOIN account_tb wat
ON ht.w_account_id = wat.id
WHERE ht.w_account_id = 1
```



ID	AMOUNT	W_BALANCE	D_BALANCE	SENDER	RECEIVER	CREATED_AT
1	100	900	1100	1111	ATM	2023-02-17 11:17:49.483345
2	100	800	*null*	1111	ATM	2023-02-17 11:17:49.484341

3.2 입금내역 보기

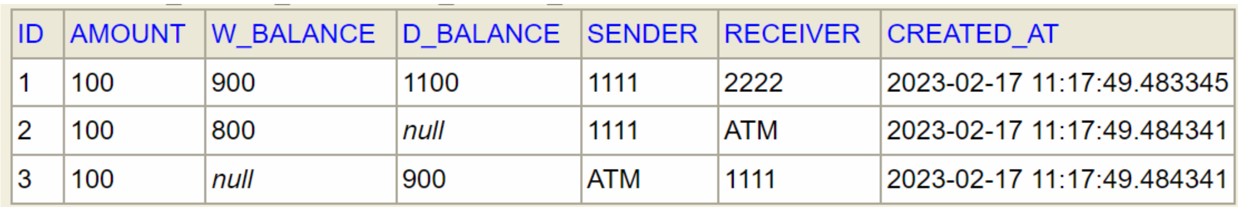
```
SELECT ht.id, ht.amount, ht.w_balance, ht.d_balance,  'ATM' sender, dat.number
receiver, ht.created_at
FROM history_tb ht
INNER JOIN account_tb dat
ON ht.d_account_id = dat.id
WHERE ht.d_account_id = 1
```



ID	AMOUNT	W_BALANCE	D_BALANCE	SENDER	RECEIVER	CREATED_AT
3	100	*null*	900	ATM	1111	2023-02-17 11:17:49.484341

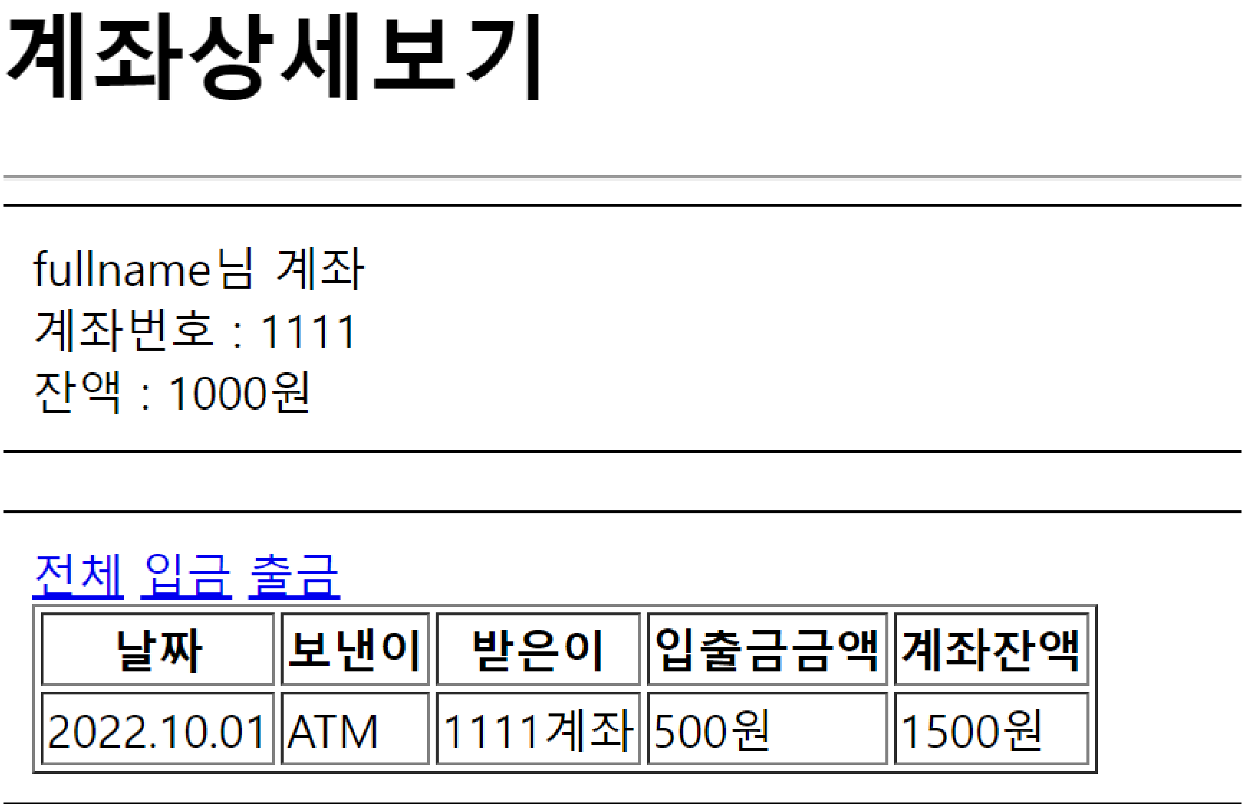
3.3 입출금내역 보기

```
SELECT ht.id, ht.amount, ht.w_balance, ht.d_balance, nvl(wat.number, 'ATM') sender,
nvl(dat.number, 'ATM') receiver, ht.created_at
FROM history_tb ht
LEFT OUTER JOIN account_tb wat
ON ht.w_account_id = wat.id
LEFT OUTER JOIN account_tb dat
ON ht.d_account_id = dat.id
WHERE ht.w_account_id = 1 or ht.d_account_id = 1
```



ID	AMOUNT	W_BALANCE	D_BALANCE	SENDER	RECEIVER	CREATED_AT
1	100	900	1100	1111	2222	2023-02-17 11:17:49.483345
2	100	800	*null*	1111	ATM	2023-02-17 11:17:49.484341
3	100	*null*	900	ATM	1111	2023-02-17 11:17:49.484341

3.4 계좌 상세보기



계좌상세보기

fullname님 계좌
계좌번호 : 1111
잔액 : 1000원

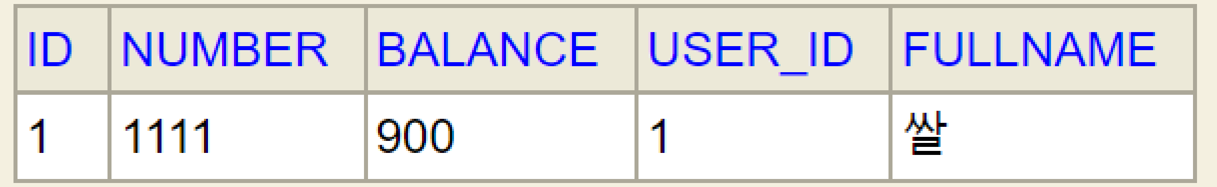
전체 입금 출금

날짜	보낸이	받은이	입출금금액	계좌잔액
2022.10.01	ATM	1111계좌	500원	1500원

3.4.1 계좌상세보기 계좌정보

- INNER JOIN 해도 되지만, 스칼라 서브쿼리 연습

```
select at.id, at.number, at.balance, at.user_id, (select fullname from user_tb ut
where ut.id = at.user_id) fullname
from account_tb at
where at.id = 1
```

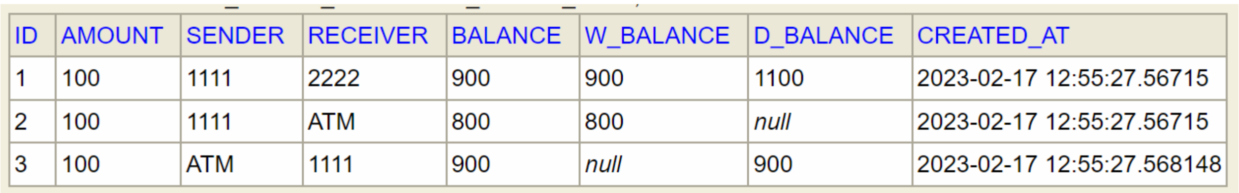


ID	NUMBER	BALANCE	USER_ID	FULLNAME
1	1111	900	1	쌀

3.4.2 계좌상세보기 트랜잭션 히스토리 알고리즘

> 입출금 모두 보기
>
> 출금 계좌 잔액이 null 이면 임금 계좌 잔액을 보여준다.
>
> 입금 계좌 잔액이 null 이면 출금 계좌 잔액을 보여준다.
>
> 둘 다 아니면 출금 계좌 잔액을 보여준다.

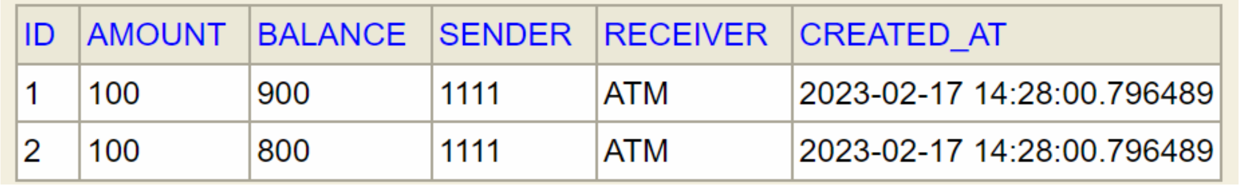
```
SELECT
ht.id,
ht.amount,
nvl(wat.number, 'ATM') sender,
nvl(dat.number, 'ATM') receiver,
case when ht.w_account_id = 1 then ht.w_balance
     when ht.d_account_id = 1 then ht.d_balance
     end as balance,
ht.w_balance,
ht.d_balance,
ht.created_at
FROM history_tb ht
LEFT OUTER JOIN account_tb wat
ON ht.w_account_id = wat.id
LEFT OUTER JOIN account_tb dat
ON ht.d_account_id = dat.id
WHERE ht.w_account_id = 1 or ht.d_account_id = 1;
```



ID	AMOUNT	SENDER	RECEIVER	BALANCE	W_BALANCE	D_BALANCE	CREATED_AT
1	100	1111	2222	900	900	1100	2023-02-17 12:55:27.56715
2	100	1111	ATM	800	800	*null*	2023-02-17 12:55:27.56715
3	100	ATM	1111	900	*null*	900	2023-02-17 12:55:27.568148

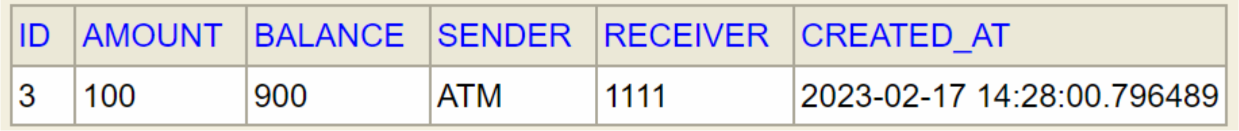
3.4.3 계좌상세보기 트랜잭션 히스토리 완성 쿼리

```
SELECT ht.id, ht.amount, ht.w_balance balance, wat.number sender, 'ATM' receiver,
ht.created_at
FROM history_tb ht
INNER JOIN account_tb wat
ON ht.w_account_id = wat.id
WHERE ht.w_account_id = 1
```



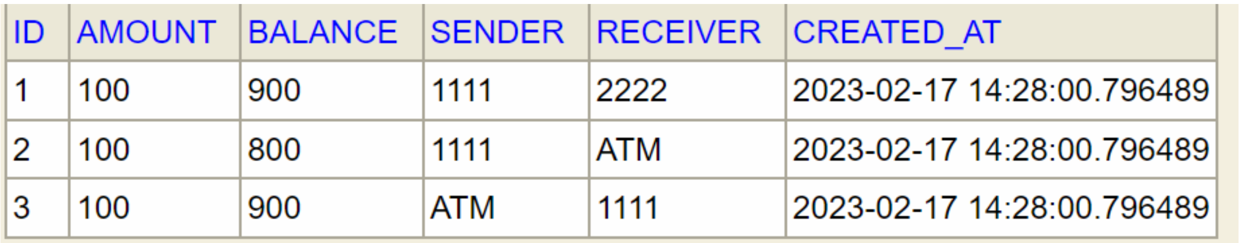
ID	AMOUNT	BALANCE	SENDER	RECEIVER	CREATED_AT
1	100	900	1111	ATM	2023-02-17 14:28:00.796489
2	100	800	1111	ATM	2023-02-17 14:28:00.796489

```
SELECT ht.id, ht.amount, ht.d_balance balance, 'ATM' sender, dat.number receiver,
ht.created_at
FROM history_tb ht
INNER JOIN account_tb dat
ON ht.d_account_id = dat.id
WHERE ht.d_account_id = 1
```



ID	AMOUNT	BALANCE	SENDER	RECEIVER	CREATED_AT
3	100	900	ATM	1111	2023-02-17 14:28:00.796489

```
SELECT
ht.id,
ht.amount,
case when ht.w_account_id = 1 then ht.w_balance
     when ht.d_account_id = 1 then ht.d_balance
     end as balance,
nvl(wat.number, 'ATM') sender,
nvl(dat.number, 'ATM') receiver,
ht.created_at
FROM history_tb ht
LEFT OUTER JOIN account_tb wat
ON ht.w_account_id = wat.id
LEFT OUTER JOIN account_tb dat
ON ht.d_account_id = dat.id
WHERE ht.w_account_id = 1 or ht.d_account_id = 1;
```



ID	AMOUNT	BALANCE	SENDER	RECEIVER	CREATED_AT
1	100	900	1111	2222	2023-02-17 14:28:00.796489
2	100	800	1111	ATM	2023-02-17 14:28:00.796489
3	100	900	ATM	1111	2023-02-17 14:28:00.796489