

- ROWNUM은 테이블에 존재하지 않고, 테이블에서 가져온 데이터를 이용해서 번호를 매기는 방식으로 결과는 테이블에서 가장 먼저 가져올 수 있는 데이터들을 꺼내서 번호를 붙인다.
- BNO 217번 데이터는 3번째로 꺼내진 데이터이다.

```
SELECT /*+ FULL(tb1_board) */
       rownum rn, bno, title
FROM   tb1_board
WHERE  bno > 0
ORDER BY bno;
```

스크립트 출력 x | 자동 추적 x | 스크립트

SQL | 350개의 행이 인출됨(

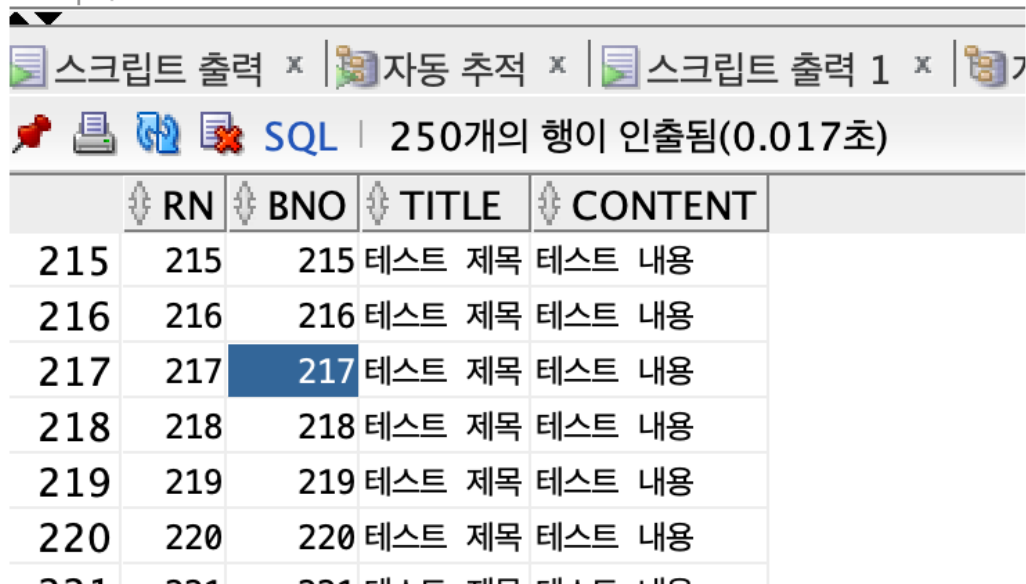
	RN	BNO	TITLE
211	529	211	테스트 제목
212	530	212	테스트 제목
213	531	213	테스트 제목
214	532	214	테스트 제목
215	1	215	테스트 제목
216	2	216	테스트 제목
217	3	217	테스트 제목
218	4	218	테스트 제목
219	5	219	테스트 제목
220	6	220	테스트 제목

INDEX를 이용한 접근 시 ROWNUM

ROWNUM의 의미가 테이블에서 데이터를 가져오면서 붙는 번호라는 사실은 결국 문제는 테이블에 어떤 순서로 접근하는가에 따라서 ROWNUM값은 바뀔 수 있다는 것이다.

- 위의 경우는 우선 FULL로 접근해서 217번 데이터를 찾았고 이후에 정렬을 하는데 이미 데이터는 다 가져온 상태이므로 ROWNUM에는 아무런 영향을 주지 않는다.

```
SELECT /*+ INDEX_ASC(tb1_board pk_board) */  
       rownum rn, bno, title, content  
FROM tb1_board;
```




스크립트 출력 x | 자동 추적 x | 스크립트 출력 1 x |

SQL | 250개의 행이 인출됨(0.017초)


	RN	BNO	TITLE	CONTENT
215	215	215	테스트 제목	테스트 내용
216	216	216	테스트 제목	테스트 내용
217	217	217	테스트 제목	테스트 내용
218	218	218	테스트 제목	테스트 내용
219	219	219	테스트 제목	테스트 내용
220	220	220	테스트 제목	테스트 내용
...

- 가장 먼저 찾은 데이터부터 ROWNUM이 1부터 시작

 즉, ROWNUM은 ORDER BY전에 부여되며, ORDER BY는 맨 나중에 실행된다.

- ROWNUM을 변경하기 위해 DML을 사용할 수 없다.

- 주로 <, <= 사용하며 >, >= 인 경우 **ROWNUM**은 동작하지 않는다.

- ROWNUM = 1은 사용 가능 하지만 **ROWNUM = 2**인 경우는 데이터가
추출되지 않는다.  *ROWNUM 조건을 제외한 나머지 조건을 만족하는 레코드에 붙는 순번이다.*

ROWNUM은 WHERE절을 만족하는 레코드에 붙이는 순번이므로 해석해 보면,
ROWNUM = 2는, 처음 한 건 추출해서 ROWNUM이 2인지 비교하는 것이다.
하지만 처음 레코드는 ROWNUM이 1이며, 조건에 맞지 않으므로 버린다.

그 다음 레코드 선택 후 또 ROWNUM이 2인지 비교하지만 전 레코드를
버렸기 때문에 새로 추출되는 레코드는 ROWNUM이 1이다. 2가 아니다.

그러기 때문에 또 버리게 되고 그 다음 레코드를 추출한다. 이 과정을 반복하면
ROWNUM = 2는 도달할 수 없는 값이 됨을 알 수 있다.)

또 ROWNUM 값이 실제로 할당되는 방법에 대해서도 많은 사람들이 오해를 하고 있습니다. ROWNUM 값은 쿼리의 조건절이 처리되고 난 이후, 그리고 sort, aggregation이 수행되기 이전에 할당됩니다. ~~또~~ 또 ROWNUM 값은 할당된 이후에만 증가 (increment) 됩니다. 따라서 아래 쿼리는 로우를 반환하지 않습니다.

