```
SUM(NVL(COL1,0))
```

※ 집계함수는 컬럼간위로 연산을 실시함. (집계함수가 아닌 함수는 ROW간위로 연산실시)

여기서 COL1 컬럼은 SUM()함수에 의하여 연산이 된다. 연산중 NULL값이 있어 전체 결과가 NULL값이 될까 두려워 이렇게 사용했다면 이것은 잘못된 생각이다.

집계함수의 특징.

SUM()함수에서 단일 컬럼이 연산이 될 때에는 NULL값은 연산의 대상에서 제외된다. NULL값이 연산을 하지 않았으므로 결과는 절대 NULL값이 되지 않는다. 위 처럼 NVL()함수를 사용하여 NULL값을 가진 경우 '0'으로 바꾸어도 결과는 동일하다. 그러니 SUM()함수 내에 NVL()를 쓰는 것은 불필요한 연산만 증가시키는 것이다.
'0'을 더하는 것도 연산이기 때문이다. 예를 들어 SUM()함수를 이용할 Row가 10만이고 그중 NULL값을 가진 경우가 9만이라고 한다면 불필요한 연산을 9만번이나 더 수행하는 것이 된다.

사실 이와 같은 잘못을 저지르는 이유는 NULL값의 연산에 대한 무지보다는 최종으로 추출되는 결과가 NULL값인 겨우에 화면에 공백이 추출되어 보이므로 이를 '0'으로 채우고 싶어서 그렇게 한 경우일 것이다.
그렇다면 이런 경우는 아래와 같이 하여야 한다.

```
NVL(SUM(COL1),0)
```

바로 위 문장은 앞선 위의 경우와 큰 차이가 없어 보이지만 처리할 일의 양에는 큰 차이가 난다.
조금전에 가정했듯이 Row 10만 건이 있다가 했을 때 전체 값이 NULL인 경우에 앞선 예는 10만 번이나 '0'을 더하는 연산을 하여 '0'을 출력하지만 바로 위의 예는 한번도 연산을 하지 않고 '0'을 출력하게 된다.

이번에는 두 컬럼의 연산이 있는 경우를 살펴보도록 하겠다.
예를 들어 아래와 같이 데이터가 들어 있다고 했을 경우 어떤 결과가 나오는지 확인해 보자.

ROW	COL1	COL2
1	NULL	NULL
2	1	NULL
3	1	1

1+1=2 NULL+NULL+NVLL =NULL NULL+NULL+2=2

* 실행과 결과

```
SELECT SUM(COL1), SUM(COL1+NULL), SUM(COL1+COL2) FROM 테이블명
--------------------------------------------------------------------------------
2 NULL 2
```