# 쿼리의 논리적 순서

FROM //테이블의 존재, 해당권한이 있는지를 체크한다.

ON

JOIN

WHERE

GROUP BY //SELECT보다 빨리 실행되기 때문에 ALIAS지정 해놓은거 못쓴다.

CUBE | ROLLUP

HAVING

SELECT //일단 다 가져와서 SELECT를 하기 때문에 컬럼하나를 가져오나 다가지고오나 IO비용은 거의같다.

DISTINCT

ORDER BY //SELECT보다 늦게 실행되기 때문에 ALIAS 지정 해놓은거 사용가능

TOP

## 계층형 쿼리의 순서

### <문법>

START WITH 블라블라

CONNECT BY 블라블라

### <이해>

한 개의 테이블에서 내부적으로 정의한 순서에 의해 구조화하는 방법

START WITH는 계층구조가 어느행으로부터 시작할지 설정

CONNECT BY는 각각의 행들이 어떤 계층구조의 조건으로 연결되어야 하는지 설정

### <노하우>

1. WHERE보다 CONNECT BY가 먼저 실행되기 때문에 어차피 걸 조건이라면 WHERE보다 START WITH에 거는 것이 성능이 좋다.
2. 매우 실수하기 쉬운 것.

CONNECT BY 조건에 키들을 다 넣어줘서 유일하게 해줘야 한다. 그렇지 않으면 카티션 발생한다.

# 뷰

자주 사용하는 쿼리문들을 매번 작성하기 보다는 뷰로 만들어놓고 참조하면 편리하다.

가상테이블이기 때문에 물리적으로 구현되어 있지 않다.

생성 : CREATE VIEW 이름

AS 서브쿼리

(WITH CHECK OPTION 제약조건명

WITH READ ONLY)

컴파일 : ALTER VIEW 이름 COMPILE

수정 : REPLACE

삭제 : DROP VIEW 이름

# 조인

같은 행동이나 ANSI JOIN이냐 ORACLE JOIN이냐에 따라 문법이 다르다.

## 오라클조인

### <내부조인>

#### 동등조인

양쪽의 데이터가 정확하게 같은것만 가져올 때 사용.

기호로 =를 사용한다. 예) WHERE a.dept\_id = b.dept\_id

#### 세미조인

서브쿼리로 조건에 만족하는 데이터 그룹을 따로 만들어서 합칠 때 사용.

IN과 EXISTS 사용. EXISTS뒤에는 값의 리스트가 아닌 서브쿼리만 올 수 있다.

IN과 EXISTS의 차이 :

IN : 뒤에 포함되어있는 조건을 건다.

EXISTS : 하나의 행마다 서브쿼리를 실행해서 하나라도 존재하면 참이 되어서 레코드를 출력

#### 안티조인

세미조인과 반대개념. NOT IN 과 NOT EXISTS 사용.

예) 판매된 제품목록에 없는 인기없는 제품 리스트 뽑기

SELECT P.,ProductNumber, p.ProductName

FROM Products As P

WHERE not EXISTS

(SELECT \*

FROM Order\_Details AS od

WHERE od.ProductNumber = p.ProductNumber);

#### 셀프조인

동등한 테이블을 사용해 조인

#### 외부조인

어느 한쪽 테이블에는 값이 없거나 해당 로우가 아예 없더라도 데이터를 가져온다.

오라클에서는 **참조되는 쪽(데이터가 없는 테이블)에 (+)**를 붙여준다.

외부조인은 조건에 해당하는 조인 조건 모두에 (+)를 붙여야 한다.

한번에 한 테이블에만 외부조인을 할 수 있다.

#### 카타시안 조인

조인조건이 없는 조인. 그 결과의 수는 두 테이블 건수의 곱.

## ANSI 조인

조인조건이 WHERE절이 아닌 FROM 절에 들어간다.

#### 내부조인

FROM 테이블A INNER JOIN 테이블B ON 조건절

#### 외부조인

기준테이블 LEFT OUTER JOIN 참조테이블 ON 조건절

기준테이블 RIGHT OUTER JOIN 참조테이블 ON 조건절

#### 풀아우터조인

ANSI조인에만 제공하는 기능이다. 양쪽 다 이빨빠지게 조회한다.

## 조인의 수행원리

### NESTED LOOP JOIN

이너 테이블에서 JOIN 컬럼이 인덱스에 걸려있지 않으면 굉장한 비효율이 발생된다.

한건마다 풀스캔을 해야되기 때문에. 때문에 대량의 데이터에는 적합하지 않다.

### SORT MERGE JOIN

JOIN컬럼을 기준으로 sorting을 시키고 난 뒤에 조인을 실행한다.

예) 조인하려고 하는데 멤버테이블이 굉장한 대용량이라고 치자.

그럴 때 그룹 테이블을 build input으로 삼아서 hash 영역에 저장해놓는다.

멤버테이블이 읽히면서 조인이 되는 원리.

해쉬영역으로 올라갈때는 join컬럼을 기준으로 hash function이 적용되기 때문에 key컬럼에 중복값이 없을수록 성능에 유리하다.

해쉬영역에 들어가는 테이블의 크기가 충분히 작아야 한다. 해쉬영역을 벗어날 정도로 큰 테이블이라면 오히려 디스크 영역을 사용하기 때문에 성능에 몹시 불리하게 된다.