sqld 시험 정리 2과목 part2

카테고리	SQLD
체크 표시됨	☐
태그	notion

서브쿼리

-하나의 SQL 문안에 포함되어 있는 또 다른 SQL 문을 말함.

-반드시 괄호로 묶어야 함.

ex) SELECT 안에 SELECT 문, INSERT, UPDATE,DELETE 안의 SELECT 문

서브쿼리 사용 가능한 곳

1. SEELCET 절
2. FROM 절
3. WHERE 절
4. HAVING 절
5. ORDER BY 절
6. 기타 DML(INSERT,DELETE,UPDATE)절

서브 쿼리 종류

1. 동작하는 방식에 따라

1) UN-CORREALATED(비연관) 서브쿼리

- 서브쿼리가 메인쿼리 컬럼을 가지고 있지 않은 형태의 서브쿼리
- 메인쿼리에 서브쿼리가 실행된 결과 값을 제공하기 위한 목적으로 사용.

2)CORRELATED(연관) 서브쿼리

- 서브쿼리가 메인쿼리 컬럼을 가지고 있는 형태의 서브 쿼리
- 일반적으로 메인쿼리가 먼저 수행된 후에 서브쿼리에서 조건이 맞는지 확인하고자 할 때 사용

2. 위치에 따라

1) 스칼라 서브쿼리

- SELECT에 사용하는 서브쿼리
- 서브쿼리 결과를 마치 하나의 컬럼처럼 사용하기 위해 주로 사용

2) 인라인뷰

- FROM 절에 사용하는 서브쿼리
- 서브쿼리 결과를 테이블처럼 사용하기 위해 주로 사용

3) WHERE 절 서브쿼리

- 가장 일반적인 서브쿼리
- 비교 상수 자리에 값을 전달하기 위한 목적으로 주로 사용(상수항의 대체)
- 리턴 데이터의 형태에 따라 단일행 서브쿼리, 다중행 서브쿼리, 다중컬럼 서브쿼리, 상호연관 서브쿼리로 구분

WHERE 절 서브쿼리 종류

1) 단일행 서브쿼리

- 서브쿼리 결과가 1개의 행이 리턴되는 형태
- 단일행 서브쿼리 연산자 종류

2) 다중행 서브쿼리

- 서브쿼리 결과가 여러 행이 리턴되는 형태
- =,>,< 와 같은 비교 연산자 사용불가(여러 값이랑 비교할 수 없는 연산자들)
- 서브쿼리 결과를 하나로 요약하거나 다중행 서브쿼리 연산자를 사용

3) 다중컬럼 서브쿼리

- 서브쿼리 결과가 여러 컬럼이 리턴되는 형태
- 메인쿼리와의 비교 컬럼이 2개 이상
- 대소 비교 전달 불가(두 값을 동시에 묶어 대소비교 할 수 없음)

4) 상호연관 서브쿼리

- 메인쿼리와 서브쿼리의 비교를 수행하는 형태
- 비교할 집단이나 조건은 서브쿼리에 명시(메인쿼리절에는 서브쿼리 칼럼이 정의되지 않았기 때문에 에러 발생)

****상호연관 서브쿼리 연산 순서**

1) 메인쿼리 테이블 READ

2) 메인쿼리 WHERE 절 확인(SAL 확인)

3) 서브쿼리 테이블 READ

4) 서브쿼리 WHERE 절 확인(다시 E1.DEPTNO 요구)

5) E1. DEPTNO 값을 서브쿼리의 DEPTNO 컬럼과 비교하여 조건절 완성

6) 위 조건에 성립하는 행의 그룹 연산 결과 확인(AVG(SAL)

7) 위 결과를 메인쿼리에 전달하여 해당 조건을 만족하는 행만 추출

*상호연관 서브쿼리 사용 시 GROUP BY 생략 가능

인라인뷰(Inline View)

-쿼리 안의 뷰의 형태로 테이블처럼 조회할 데이터를 정의하기 위해 사용

-테이블명이 존재하지 않기 때문에 다른 테이블과 조인 시 반드시 테이블 별칭 명시 (단독으로 사용하는 경우는 불필요)

-WHERE 절 서브쿼리와 다르게 서브쿼리 결과를 메인 쿼리의 어느 절에서도 사용할 수 있음

-인라인뷰의 결과와 메인쿼리 테이블과 조인할 목적으로 주로 사용

-모든 연산자 사용 가능

스칼라 서브쿼리

-SELECT 절에 사용하는 쿼리로, 마치 하나의 컬럼처럼 표현하기 위해 사용(단, 하나의 출력 대상만 표현 가능)

-각 행마다 스칼라 서브쿼리 결과가 하나여야 함(단일행 서브쿼리 형태)

-조인의 대체 연산

-스칼라 서브쿼리를 사용한 조인 처리 시 OUTER JOIN이 기본(값이 없더라도 생략되지 않고 NULL로 출력)

서브 쿼리 주의 사항

-특별한 경우(TOP-N 분석 등)을 제외하고는 서브 쿼리절에 ORDER BY 절을 사용 불가

-단일 행 서브쿼리와 다중 행 서브쿼리에 따라 연산자의 선택이 중요

집합 연산자

-SELECT문 결과를 하나의 집합으로 간주, 그 집합에 대한 합집합, 교집합, 차집합 연산

-SELECT 문과 SELECT 문 사이에 집합 연산자 정의

-두 집합의 컬럼이 동일하게 구성되어야 함(각 컬럼의 데이터 타입과 순서 일치 필요)

-전체 집합의 데이터타입과 첫번째 집합에 의해 결정됨

합집합

-두 집합의 총 합(전체) 출력

-UNION 과 UNION ALL 사용 가능

1) UNION

-중복된 데이터는 한 번만 출력

-중복된 데이터를 제거하기 위해 내부적으로 정렬 수행

-중복된 데이터가 없을 경우는 UNION 사용 대신 UNION ALL 사용(불필요한 정렬 발생할 수 있으므로)

2)UNION ALL

-중복된 데이터도 전체 출력

교집합

-두 집합 사이에 INTERSECT

-두집합의 교집합(공통으로 있는 행) 출력

차집합

-두 집합 사이에 MINUS 전달

-두 집합의 차집합(한 쪽 집합에만 존재하는 행) 출력

-A-B와 B-A는 다르므로 집합의 순서 주의!

집합 연산자 사용시 주의 사항

1. 두 집합의 컬럼 수 일치
2. 두 집합의 컬럼 순서 일치
3. 두 집합의 각 컬럼의 데이터 타입 일치
4. 각 컬럼의 사이즈는 달라도 됨
5. 개별 SELECT 문에 ORDER BY 전달 불가(GROUP BY 전달 가능)

그룹함수

-숫자함수 중 여러값을 전달하여 하나의 요약값을 출력하는 다중행 함수

-수학/통계 함수들(기술통계 함수)

-GROUP BY 절에 의해 그룹별 연산 결과를 리턴 함

-반드시 한 컬럼만 전달

-NULL은 무시하고 연산

COUNT

-행의 수를 세는 함수

-대상 컬럼은 * 또는 단 하나의 컬럼만 전달 가능(* 사용 시 모든 컬럼의 값이 널일 때만 COUNT 제외)

-문자, 숫자, 날짜 컬럼 모두 전달 가능

-행의 수를 세는 경우 NOT NULL 컬럼을 찾아 세는 것이 좋음(PK 컬럼)

VARIANCE / STDDEV

- 분산과 표준편차
- 표준편차는 분산의 루트값

GROUP BY FUNCTION

-GROUP BY 절에 사용하는 함수

1. **GROUPING SETS(A,B, ...)**

-A 별, B 별 그룹 연산 결과 출력

-나열 순서 중요하지 x

-기본 출력에 전체 총계는 출력되지 x

-NULL 혹은 () 사용하여 전체 총 합 출력 가능

2. **ROLLUP(A,B)**

-A 별, (A,B)별, 전체 그룹 연산 결과 출력

-나열 대상의 순서가 중요함

-기본적으로 전체 총계가 출력됨

3. **CUBE(A,B)**

-A 별, B 별, (A,B)별, 전체 그룹 연산 결과 출력됨

-그룹으로 묶을 대상의 나열 순서 중요하지 x

-기본적으로 전체 총계가 출력됨

윈도우 함수(WINDOW FUNCTION)

-서로 다른 행의 비교나 연산을 위해 만든 함수

-GROUP BY 를 쓰지 않고 그룹 연산 가능

-LAG, LEAD, SUM, AVG, MIN, MAX, COUNT, RANK

*****PARTITION BY 절: 출력할 총 데이터 수 변화 없이 그룹연산 수행할 GROUP BY 컬럼**

*****ORDER BY 절**

-RANK의 경우 필수(정렬 컬럼 및 정렬 순서에 따라 순위 변화)

-SUM, AVG, MIN, MAX, COUNT 등은 누적값 출력 시 사용

*****ROWS[RANGE BEETWEEN A AND B**

-연산 범위 설정

-ORDER BY 절 필수

- **PARTITON BY, ORDER BY, ROWS... 절 전달 순서 중요(ORDER BY를 PARTITON BY 전에 사용 불가)**

그룹 함수의 형태

-SUM, COUNT, AVG, MIN, MAX 등

-OVER 절을 사용하여 윈도우 함수로 사용 가능

-반드시 연산할 대상을 그룹함수의 입력값으로 전달

1)SUM OVER()

-전체 총 합, 그룹별 총 합 출력 가능

**** 윈도우 함수의 연산 범위 : 집계 연산 시 행의 범위 설정 가능**

1. **ROWS, RANGE 차이**

1) ROWS: 값이 같더라도 각 행씩 연산

2) RANGE: 같은 값의 경우 하나의 RANGE로 묶어서 동시 연산(DEFAULT)

2. **BETWEEN A AND B**

A) 시작점 정의

-CURRENT ROW: 현재행부터

-UNBOUNDED PRECEDING: 처음부터(DEFAULT)

-N PRECEDING: N 이전부터

B) 마지막 시점 정의

-CURRENT ROW: 현재행까지(DEFAULT)

-UNBOUNDED FOLLOWING: 마지막까지

-N FOLLOWING: N 이후까지

순위 관련 함수

1) RANK(순위)

1-1) RANK WITHIN GROUP

-특정값에 대한 순위 확인(RANK WITHIN)

-윈도우함수가 아닌 일반함수

2-2)RANK() OVER()

-전체 중/특정 그룹 중 값의 순위 확인

-ORDER BY 절 필수

-순위를 구할 대상을 ORDER BY 절에 명시(여러 개 나열 가능)

-그룹 내 순위 구할 시 PARTITION BY 절 사용

3) DENSE_RANK

-누적순위

-값이 같을 때 동일한 순위 부여 후 다음 순위가 바로 이어지는 순위 부여 방식

ex) 1등이 5명이더라도 그 다음 순위가 2등

4) ROW_NUMBER

-연속된 행 번호

-동일한 순서를 인정하지 않고 단순히 순서대로 나열한대로의 순서 값 리턴

- **LAG, LEAD**

-행 순서대로 각각 이전 값(LAG), 이후 값(LEAD) 가져오기

-ORDER BY 절 필수

- **FIRST_VALUE, LAST_VALUE**

-정렬 순서대로 정해진 범위에서의 처음 값, 마지막 값 출력

-순서와 범위 정의에 따라 최솟값과 최댓값 리턴 가능

-PARTITON BY, ORDER BY 절 생략 가능

- **NTILE**

-행을 특정 컬럼 순서에 따라 정해진 수의 그룹으로 나누기 위한 함수

-그룹 번호가 리턴됨

-ORDER BY 필수

-PARTITON BY를 사용하여 특정 그룹을 또 원하는 수 만큼 그룹 분리 가능

-총 행의 수가 명확히 나눠지지 않을 때 앞 그룹의 크기가 더 크게 분리됨

ex) 14명 3개 그룹 분리 시 → 5, 5,4 로 나뉨