SQL Pogramming

- Day 6 -

2023. 04

목차

Day 1. 데이터베이스와 SQL
Day 2. 테이블 / 인덱스
Day 3. DDL / DML / DCL / TCL
Day 4. SELECT 기본문형 익히기1
Day 5. SELECT 기본문형 익히기2
Day 6. 서브쿼리 / 스칼라쿼리
Day 7. 뷰 / 인라인뷰
Day 8. 내장함수 일반
Day 9. 내장함수 CASE
Day 10. 조인 기본
Day 11. 조인 활용1
Day 12. 조인 활용2
Day 13. 데이터 압축하기1
Day 14. 데이터 압축하기2
Day 15. 데이터 늘리기1
Day 16. 데이터 늘리기2
Day 17. 인덱스 이해하기
Day 18. SELECT 중요성
Day 19. 분석함수1
Day 20. 분석함수2
Day 21. 분석함수3
Day 22. 실전연습1
Day 23. 실전연습2
Day 24. 프로시저 만들기1
Day 25. 프로시저 만들기2
Day 26. SQL 리뷰하기

Day 6. 서브쿼리 / 스칼라쿼리

■ SUB-QUERY(서브쿼리) 정의

▶ 하나의 SQL문 안에 포함되어 있는 또 하나의 SQL문을 말함

▶ 서브쿼리는 메인쿼리에 포함되는 종속적인 관계임

▶ 문법적으로 구분하는 가장 쉬운 방법은 SQL문이 ( )로 묶어져 있음

▶ 서브쿼리의 위치에 따라 다음과 같이 나눌 수 있음

① WHERE절 → Nested SUB-QUERY

② FROM절 → INLINE-VIEW

③ SELECT절 → SCALAR-QUERY

▶ 서브쿼리는 메인쿼리의 컬럼을 모두 사용할 수 있지만 메인쿼리는 서브쿼리의 컬럼을 사용할 수 없음 (예외적으로 인라인뷰에 정의된 컬럼은 메인쿼리에서 사용 가능)

▶ 큰 개념의 조인에 포함시킬 수 있으나 보통은 구분함

▶ 서브쿼리를 사용해야 할 때 잘못 판단하여 조인 방식을 사용하는 경우

→ 결과는 부서(1) 레벨이고, 사원(M) 테이블에서 체크해야 할 조건이 존재한다고 하면, 이런 상황에서 조인을 사용하면 결과 집합은 사원(M) 레벨이 되어 버림.

이렇게 되면 원하는 결과를 만들기 위해 GROUP BY나 DISTINCT를 추가해 결과를 다시 조직(1) 레벨로 만들어야 함

■ SUB-QUERY(서브쿼리) 종류 - 반환 데이터의 형태 분류

▶ SINGLE ROW(단일 행) 서브쿼리

○ 서브쿼리의 실행 결과가 항상 1건 이하인 서브쿼리를 의미

○ 단일 행 서브쿼리는 단일 행 비교 연산자와 함께 사용됨

○ 단일 행 비교 연산자에는 =, <, <=, >, >=, <> 이 있음

▶ MULTI ROW(다중 행) 서브쿼리

○ 서브쿼리의 실행결과가 여러 건인 서브쿼리를 의미

○ 다중 해 서브쿼리는 다중 행 비교 연산자와 함께 사용됨

▶ MULTI COLUMN(다중 컬럼) 서브쿼리

○ 서브쿼리의 실행 결과로 여러 컬럼을 반환함

○ 메인쿼리의 조건 절에 여러 컬럼을 동시에 비교할 수 있음

○ 서브쿼리와 메인쿼리에서 비교하고자 하는 컬럼 개수와 컬럼 위치가 동일해야 함

■ SUB-QUERY(서브쿼리) 종류 - 동작하는 방식 분류

▶ 비연관 서브쿼리

○ 서브쿼리가 메인쿼리 컬럼을 가지고 있지 않은 형태

▶ 연관 서브쿼리

○ 서브쿼리가 메인쿼리 컬럼을 사용하는 형태

■ 단일행 SUB-QUERY(서브쿼리) 사용 예제

```
SELECT OUTBOUND_DATE
      ,INVOICE_NO
      ,OUT_TYPE_DIV
  FROM LO_OUT_M
 WHERE INVOICE_NO = (
                     SELECT MAX(INVOICE_NO)
                       FROM LO_OUT_D
                      WHERE ITEM_NM   LIKE '% 호박죽'
                        AND ORDER_QTY = 10
                    );
```

Day 6. 서브쿼리 / 스칼라쿼리

■ 다중행 SUB-QUERY(서브쿼리) 사용 예제

```
SELECT OUTBOUND_DATE
      ,INVOICE_NO
      ,OUT_TYPE_DIV
  FROM LO_OUT_M
 WHERE INVOICE_NO IN (
                      SELECT INVOICE_NO
                        FROM LO_OUT_D
                       WHERE ITEM_NM   LIKE '% 호박죽 '
                         AND ORDER_QTY = 10
                     );
```

Day 6. 서브쿼리 / 스칼라쿼리

■ 다중컬럼 SUB-QUERY(서브쿼리) 사용 예제

```
SELECT OUTBOUND_DATE
      ,INVOICE_NO
      ,OUT_TYPE_DIV
  FROM LO_OUT_M
 WHERE (INVOICE_NO, OUT_TYPE_DIV) IN (
                                      SELECT INVOICE_NO, OUT_TYPE_DIV_D
                                        FROM LO_OUT_D
                                       WHERE ITEM_NM   LIKE '%호박죽'
                                         AND ORDER_QTY > 100
                                     );
```

Day 6. 서브쿼리 / 스칼라쿼리

■ 연관 SUB-QUERY(서브쿼리) 사용 예제

```
SELECT OUTBOUND_DATE
      ,INVOICE_NO
      ,OUT_TYPE_DIV
  FROM LO_OUT_M M1
 WHERE M1.OUTBOUND_DATE = TO_DATE('20190903', 'YYYY-MM-DD ' )
   AND M1.INVOICE_NO    IN (SELECT S1.INVOICE_NO
                              FROM LO_OUT_D S1
                             WHERE S1.INVOICE_NO = M1.INVOICE_NO
                               AND S1.ORDER_QTY  > 100
                           );
```

■ 연관 SUB-QUERY(서브쿼리) 사용 예제

▶ EXISTS 서브쿼리

○ 항상 연관 서브쿼리로 사용됨

○ 아무리 조건을 만족하는 건이 여러 건이더라도 조건을 만족하는 1건만 찾으면 추가적인 검색을 진행하지 않음

○ 현업에서 조건을 만족하는지 여부를 묻는 로직이 많이 사용됨

○ EXISTS 연산자의 왼쪽에는 컬럼명이나 상수가 표시되지 않음

```
SELECT OUTBOUND_DATE
      ,INVOICE_NO
      ,OUT_TYPE_DIV
  FROM LO_OUT_M M1
 WHERE M1.OUTBOUND_DATE = TO_DATE('20190903', 'YYYY-MM-DD')
   AND EXISTS (SELECT 1
                 FROM LO_OUT_D S1
                WHERE S1.INVOICE_NO = M1.INVOICE_NO
                  AND S1.ORDER_QTY  > 100
               );
```

Day 6. 서브쿼리 / 스칼라쿼리

■ SCALAR-QUERY(스칼라쿼리) 정의

- ▶ 한 행, 한 컬럼(1Row, 1Column)만을 반환하는 서브쿼리
- ▶ 데이터가 추출되지 않아도 됨 (NULL 데이터 추출)
- ▶ 결과 건수만큼 반복적으로 수행됨
  - ○ 동일한 입력 값이 들어오면 수행하지 않음
  - ○ MULTI BUFFER에 저장한 값을 이용하여 리턴함
- ▶ 컬럼(Column)을 쓸 수 있는 모든 대부분의 곳에서 사용할 수 있음
- ▶ 일종의 함수이므로 중첩해서 사용할 수 있음
- ▶ OUTPUT이 두 개 이상 나오거나, OUTPUT의 데이터 유형이 맞지 않는 경우 Syntax 에러가 발생함
- ▶ 함수의 특징을 가짐. 기능적으로 많은 INPUT이 있더라도 OUTPUT은 일반적으로 하나만 나온다는 것. 성능면에서는 전체 데이터를 일일이 모든 건 수만큼 수행해야 함
- ▶ 대량의 데이터 처리 시, 스칼라쿼리를 남발하는 경우는 조인의 장점이나 집합적 개념을 적용하기 힘들므로, 같은 결과를 얻을 수 있다면 가능하면 스칼라쿼리가 아니라 조인으로 대체하는 것이 좋음

■ SCALAR-QUERY(스칼라쿼리) 위치

- ▶ SELECT LIST 항목
- ▶ 함수의 인자
- ▶ WHERE 절의 조건
- ▶ ORDER BY 절
- ▶ CASE 조건 절
- ▶ CASE 결과 절
- ▶ HAVING절

■ SCALAR-QUERY(스칼라쿼리) 사용 예제

```
SELECT M1.OUTBOUND_DATE
      ,M1.INVOICE_NO
      ,M1.OUT_TYPE_DIV
      ,(
        SELECT SUM(S1.ORDER_QTY) AS SUM_ORDER_QTY
          FROM LO_OUT_D S1
         WHERE S1.INVOICE_NO = M1.INVOICE_NO
       ) AS SUM_QTY
  FROM LO_OUT_M M1
 WHERE M1.INVOICE_NO LIKE '3467247026%';
```

주문 마스터 정보 (A_OUT_M)

BRAND_CD	INVOICE_NO	OUTBOUND_DATE	OUT_TYPE_DIV	ORDER_NM
1001	#01	2023-01-03	M11	윤현수
	#02	2023-01-03	M11	전정훈
	#03	2023-01-04	M12	고선주
	#04	2023-01-05	M12	최재원
	#05	2023-01-05	M21	권민재
2001	#01	2023-01-03	M11	강민규
	#07	2023-01-04	M21	김민기
	#08	2023-01-04	M22	김민기
	#09	2023-01-04	M22	조승완
	#10	2023-01-05	M22	진효인

상품 마스터 정보 (A_ITEM)

BRAND_CD	ITEM_CD	ITEM_NM	QTY_IN_BOX
1001	A	상품A	2
	B	상품B	2
	C	상품C	2
	D	상품D	3
	E	상품E	3
2001	A	상품A	2
	B	상품B	2
	C	상품C	2
	D	상품D	3
	E	상품E	3

주문 디테일 정보 (A_OUT_D)

BRAND_CD	INVOICE_NO	LINE_NO	ITEM_CD	ORDER_QTY
1001	#01	1	A	1
	#02	1	B	1
		2	C	3
	#03	1	B	2
	#04	1	A	1
		2	D	1
		3	E	2
	#05	1	C	5
2001	#01	1	A	1
		2	B	2
	#07	1	E	1
	#08	1	C	1
	#09	1	B	3
		2	D	1
	#10	1	E	1

■ 출고일자가 1월 3일인 인보이스에 대한 주문디테일 정보를 표시해 줘!

■ 1001 브랜드이고 출고유형이 M1로 시작하는 인보이스에 대한 주문디테일 정보를 표시해 줘!

■ [브랜드] & [인보이스]별로 총 주문수량이 3 이상인 인보이스의 주문마스터 정보를 표시해 줘!

주문 마스터 정보 (A_OUT_M)

BRAND_CD	INVOICE_NO	OUTBOUND_DATE	OUT_TYPE_DIV	ORDER_NM
1001	#01	2023-01-03	M11	윤현수
	#02	2023-01-03	M11	전정훈
	#03	2023-01-04	M12	고선주
	#04	2023-01-05	M12	최재원
	#05	2023-01-05	M21	권민재
2001	#01	2023-01-03	M11	강민규
	#07	2023-01-04	M21	김민기
	#08	2023-01-04	M22	김민기
	#09	2023-01-04	M22	조승완
	#10	2023-01-05	M22	진효인

상품 마스터 정보 (A_ITEM)

BRAND_CD	ITEM_CD	ITEM_NM	QTY_IN_BOX
1001	A	상품A	2
	B	상품B	2
	C	상품C	2
	D	상품D	3
	E	상품E	3
2001	A	상품A	2
	B	상품B	2
	C	상품C	2
	D	상품D	3
	E	상품E	3

주문 디테일 정보 (A_OUT_D)

BRAND_CD	INVOICE_NO	LINE_NO	ITEM_CD	ORDER_QTY
1001	#01	1	A	1
	#02	1	B	1
		2	C	3
	#03	1	B	2
	#04	1	A	1
		2	D	1
		3	E	2
	#05	1	C	5
2001	#01	1	A	1
		2	B	2
	#07	1	E	1
	#08	1	C	1
	#09	1	B	3
		2	D	1
	#10	1	E	1

■ [브랜드] & [상품]별 주문수량 합계를 표시하되 상품명은 스칼라쿼리를 이용해 표시해 줘!

Day 6. 서브쿼리 / 스칼라쿼리

실전문제① ▶ 트랜잭션 테이블의 레코드 건수 적게 읽기			
《테이블》	■ LO_OUT_M(출고주문)	■ 읽고 싶은 테이블이 있으면 모두	■
《조건》	■ OUTBOUND_DATE(출고일자)	▶ 2019년 6월 15일 이후 10일간	
《정렬》	■		
《특징》	■ 출고 데이터가 발생한 출고일자를 순서대로 표시해 보기 ■ 다른 사람들이 작성한 SQL과 비교해 보기		

결과 ▼ 총 건수 : 7건

OUTBOUND_DATE
2019/06/17
2019/06/18
2019/06/19
2019/06/20
2019/06/21
2019/06/24
2019/06/25

Day 6. 서브쿼리 / 스칼라쿼리

실전문제② ▶ 스칼라쿼리를 활용한 컬럼 연계 최대값 구하기			
《테이블》	■ LO_OUT_M(출고주문)	■ LO_OUT_D(출고주문상세)	■
《조건》	■ OUTBOUND_DATE(출고일자) ■ OUTBOUND_NO(출고번호)	▶ 2019년 6월 3일 ▶ D190603-897353 ~ D190603-897360	
《정렬》	■		
《특징》	■ LO_OUT_M 테이블을 메인 쿼리에서 사용하고 LO_OUT_D 테이블을 스칼라 쿼리에서 사용하기 ■ 조건에 해당하는 INVOICE_NO(송장번호)에 대해 ORDER_QTY(출고수량)이 가장 큰 LINE_NO(송장라인번호)를 함께 구하기		

결과 ▼ 총 건수 : 5건

INVOICE_NO	OUT_TYPE_DIV	OUT_BOX_DIV	MAX_ORDER_QTY	MAX_LINE_NO
346724722524	M11	D1	10	1
346724717893	M11	D1	10	1
346724738226	M12	D1	14	2
346724778734	M12	D1	4	3
346724776870	M12	D1	10	1

Thank you !

ASETEC Location http://www.asetec.co.kr

본사. 경기도 성남시 분당구 성남대로 331번길 8, 킨스타워 2201호 TEL.031.609.7000 FAX.031.609.7009
부산. 부산광역시 해운대구 센텀동로 99 TEL.051.506.6352 FAX.051.504.8794