Subquery

- SQL문 안에 포함되어 있는 또 다른 SQL문으로, 알려지지 않은 기준을 이용한 검색을 위해 사용.
- 주의 사항
  - 단일 행(Single row) 비교 연산자는 Subquery 결과가 반드시 1건 이하여야 한다.
    - =, <, ⇐, <, >, >=, <>
  - 복수 행(Multiple row) 비교 연산자는 Subquery 결과 건수와 상관없다.
    - IN, ALL, ANY, SOME, EXISTS
  - ORDER BY 절을 사용할 수 없다.
    - ☞ SELECT 절에서 오직 한 개만 올 수 있기 때문에 Main query 마지막 문장에 위치해야 한다.
- 사용 가능한 위치
  - SELECT 문의 GROUP BY 제외한 나머지 부분
  - INSERT 문의 VALUES 절
  - UPDATE 문의 SET 절

Data Manipulation Language, SQL, Database

Single Row Subquery

- 단일 행 연산자(=, <, □, <, >, >=, <>)를 사용하는 Subquery
- 결과 건수는 반드시 1건 이하여야 한다.

```
SELECT PLAYER_NAME, BACK_NO
FROM PLAYER
WHERE HEIGHT <= (SELECT AVG(HEIGHT)
                 FROM PLAYER)
ORDER BY BACK_NO;
```

Multi Row Subquery

- 다중 행 비교 연산자(IN, ALL, ANY, SOME)을 사용하는 Subquery.
- Subquery 결과가 2건 이상 반환될 경우 사용.

```
SELECT TEAM_ID, PLAYER_NAME, BACK_NO
FROM PLAYER
WHERE TEAM_ID IN (SELECT TEAM_ID --IN이 아닌 =로 사용하면 Error. "동명이인"의 경우!
                  FROM PLAYER
                  WHERE PLAYER_NAME = '홍길동')
ORDER BY TEAM_NAME;
```

Multi Column Subquery

- Subquery 결과로 여러 개의 칼럼이 반환되어 Main query의 조건과 동시에 비교되는 Subquery.
- SQL Server 미지원

- 예시

소속팀별 키가 가장 작은 사람들의 정보 출력

```
SELECT TEAM_ID, PLAYER_NAME, HEIGHT
FROM PLAYER
WHERE (TEAM_ID, HEIGHT) IN (SELECT TEAM_ID, MIN(HEIGHT) --각각의 칼럼에 매칭되어 비교
                            FROM PLAYER
                            GROUP BY TEAM_ID)
ORDER BY TEAM_ID, PLAYER_NAME;
```

Correlated Subquery

- Subquery 내에 Mainquery 칼럼이 사용된 Subquery.
- 여러 만족 조건이 있더라도 추가 검색없이 단 1건만 찾으려면 EXISTS를 사용한다.

- 예시1

자신이 속한 팀의 평균 키보다 작은 선수들의 정보 출력

```
SELECT T.TEAM_NAME, MAIN_P.PLAYER_NAME, MAIN_P.HEIGHT
FROM PLAYER MAIN_P, TEAM T
WHERE MAIN_P.TEAM_ID = T.TEAM_ID
      AND MAIN_P.HEIGHT < (SELECT AVG(SUB_P.HEIGHT)
                           FROM PLAYER SUB_P
                           WHERE SUB_P.TEAM_ID = MAIN_P.TEAM_ID
                                 AND SUB_P.HEIGHT IS NOT NULL
                           GROUP BY MAIN_P.HEIGHT)
ORDER BY MAIN_P.PLAYER_NAME;
```

- 예시2

'20120501' ~ '20120502' 사이의 경기가 있는 경기장 조회 (EXISTS 이용)

```
SELECT STADIUM_ID ID, STADIUM_NAME
FROM STADIUM ST
WHERE EXISTS (SELECT 1 --임의의 가상 칼럼
              FROM SCHEDULE SC
              WHERE ST.STADIUM_ID = SC.STADIUM_ID
                    AND SC.SCHE_DATE BETWEEN '20120501' AND '20120502');
```

기타 Subquery

Scalar Subquery

- SELECT 절에서 사용하는 Subquery
- 1 Row by 1 Column 반환 Subquery.
- 단일 행 Subquery이기 때문에 2건 이상 반환되면 Error!
- 예시

소속팀별 선수의 평균키 출력

```
SELECT PLAYER_NAME, HEIGHT, ROUND((SELECT AVG(HEIGHT)
                                   FROM PLAYER SUB_P
                                   WHERE SUB_P.TEAM_ID = MAIN_P.TEAM_ID), 2)
FROM PLAYER MAIN_P;
```

Inline View

- FROM 절에서 사용하는 Subquery
- 임시로 동적 생성된 Table
  - ☞ Subquery의 칼럼은 Mainquery에서 사용할 수 없으나, Inline View는 테이블이기 때문에 Inline View의 칼럼은 SQL문을 자유롭게 참조할 수 있다!
  - ☞ ORDER BY 절 또한 사용할 수 있다!
- 예시

Position이 “MF” 선수들의 소속팀명과 선수명 조회

```
SELECT T.TEAM_NAME, P.PLAYER_NAME
FROM (SELECT TEAM_ID, PLAYER_NAME
      FROM PLAYER
      WHERE POSITION = 'MF') P,
     TEAM T
WHERE P.TEAM_ID = T.TEAM_ID;
```

TOP-N query

Inline View에 먼저 정렬을 수행하고, 정렬된 결과 중에서 WHERE 절에서 ROWNUM을 사용하여 일부 데이터를 추출하는 Subquery

```

```

From:
https://wiki.ledyx.xyz/ - **Ledyx WIKI**

Permanent link:
https://wiki.ledyx.xyz/doku.php?id=relational_database:subquery

Last update: **2021/02/07 04:16**