ORACLE®

DB Management

DQL 활용

DQL

- DQL
  - ✓ Data Query Language
  - ✓ 데이터 질의어
  - ✓ 정의된 데이터베이스에서 데이터를 조회하는 역할을 수행하는 SQL문
  - ✓ 테이블(Table)이나 뷰(View) 등의 데이터베이스 객체에서 원하는 행(Row)을 조회하는 기능을 담당

- DQL 종류
  - ① DQL에는 한 가지 쿼리문만 존재함
  - ② SELECT문을 이용해 테이블에서 원하는 내용만 조회할 수 있음

SELECT

- 기본 구문

```
SELECT 칼럼1, 칼럼2, ...
FROM 테이블_이름
[WHERE 조회_조건]
[GROUB BY 그룹_칼럼
   [HAVING 그룹_조건]]
[ORDER BY 정렬_칼럼]
```

- 수행 순서

```
⑤  SELECT 칼럼1, 칼럼2, ...
①  FROM 테이블_이름
②  [WHERE 조회_조건]
③  [GROUB BY 그룹_칼럼
④     [HAVING 그룹_조건]]
⑥  [ORDER BY 정렬_칼럼]
```

ORACLE

SELECT

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- CODE, MODEL 조회하기

```
SELECT CODE, MODEL FROM PRODUCT_TBL;
```

CODE	MODEL
A1	WING
A2	LOCK
B1	WHEEL
B2	ARM
C1	BODY
C2	HEAD

SELECT

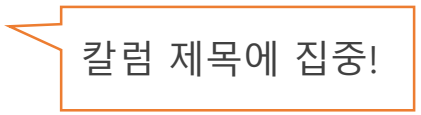
- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- CODE, MODEL 칼럼의 별명을 지정한 뒤 조회하기

```
SELECT CODE AS 코드, MODEL AS 모델 FROM PRODUCT_TBL;
```

코드	모델
A1	WING
A2	LOCK
B1	WHEEL
B2	ARM
C1	BODY
C2	HEAD



SELECT

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- 모든 칼럼(CODE, MODEL, CATEGORY, PRICE, AMOUNT, MANUFACTURED) 조회하기

```
SELECT * FROM PRODUCT_TBL;
```

중요!
실무에서는 * 금지

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

SELECT

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- 테이블에 별명을 지정한 뒤 모든 칼럼 조회하기

```
SELECT P.CODE, P.MODEL, P.CATEGORY, P.PRICE, P.AMOUNT, P.MANUFACTURED FROM PRODUCT_TBL P;
```

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

테이블 별명을 P로 지정

DISTINCT

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- CATEGORY의 중복을 제거해서 조회하기

```
SELECT DISTINCT CATEGORY FROM PRODUCT_TBL;
```

CATEGORY
MAIN
SUB

중복 제거!
DISTINCT

WHERE

- WHERE
  - ✓ 테이블의 데이터 중에서 원하는 데이터만 선택적으로 조회하고자 할 때 사용하는 절
  - ✓ 칼럼 이름, 연산자, 상수 값, 표현식 등을 결합하여 다양한 형태로 작성

- WHERE절에서 사용 가능한 데이터의 타입
  - ① 문자 타입 : 문자는 작은 따옴표(')로 묶어서 작성
  - ② 숫자 타입 : 숫자는 그냥 작성
  - ③ 날짜 타입 : 날짜는 작은 따옴표(')로 묶어서 작성하거나 TO_DATE/TO_TIMESTAMP 함수로 작성
  - ④ NULL : IS NULL 또는 IS NOT NULL

- 상수 값은 대/소문자를 구분하므로 주의해야 함
  - ✓ WHERE MODEL = 'A120'과
    WHERE MODEL = 'a120'은 다른 조건식이므로 다른 결과가 조회됨

WHERE

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- PRICE가 300 이상인 제품의 MODEL과 PRICE 조회하기

```
SELECT MODEL, PRICE FROM PRODUCT_TBL WHERE PRICE >= 300;
```

MODEL	PRICE
LOCK	300
BODY	500
HEAD	400

좋은 WHERE절 작성하기

1. (가능하면) 인덱스가 설정된 칼럼을 조건으로 사용할 것
   - ✓ 인덱스 : 빠른 검색을 위하여 특정 칼럼에 설정하는 데이터베이스 객체
   - ✓ 검색할 때 자주 사용하는 칼럼은 인덱스가 설정되어 있는 것이 매우 유리함
   - ✓ 단, 내용이 자주 변하는 칼럼은 인덱스를 설정하지 않는 것이 좋음
   - ✓ PK로 설정한 칼럼은 자동으로 인덱스가 추가되므로 조건으로 사용하면 유리함
   - ✓ UNIQUE 칼럼은 자동으로 인덱스가 추가되므로 조건으로 사용하면 유리함

2. 등호(=) 왼쪽은 (가능하면) 가공하지 않고 그대로 사용할 것
   - ✓ 인덱스가 설정된 칼럼을 함수 등으로 가공하면 인덱스를 사용할 수 없음
   - ✓ 예시) CODE 칼럼이 VARCHAR2 타입이고 PK로 설정된 경우
   - ✓ WHERE CODE = '1'
     - ➢ 상수 값을 CODE의 타입과 맞춰서 작성하면 아무 문제가 없음
   - ✓ WHERE CODE = 1
     - ✓ 상수 값을 CODE의 타입과 다르게 지정하면 오라클이 아래와 같이 TO_NUMBER 함수를 사용하여 타입을 맞춤(묵시적 형 변환)
     - ✓ WHERE TO_NUMBER(CODE) = 1
     - ✓ 위 WHERE절은 CODE의 인덱스를 사용할 수 없어서 성능이 떨어지게 됨

ORDER BY

- ORDER BY
  - ✓ 조회 결과를 정렬하고자 할 때 사용하는 절
  - ✓ 오름차순 정렬과 내림차순 정렬로 구분
  - ✓ 오름차순 정렬은 ASC, 내림차순 정렬은 DESC

- 오름차순 정렬 순서
  - ✓ 타입이 다른 데이터가 섞여 있는 경우 아래 기준으로 동작
  - ① 문자
    - ➢ 영문 : 알파벳 순
    - ➢ 한글 : 가나다 순
  - ② 숫자
    - ➢ 작은 숫자를 먼저 출력
  - ③ 날짜
    - ➢ 과거 날짜를 먼저 출력
  - ④ NULL

- 내림차순 정렬은 오름차순 정렬의 역순

ORDER BY

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- AMOUNT순으로 오름차순 정렬하여 조회하기

```
SELECT * FROM PRODUCT_TBL ORDER BY AMOUNT ASC;
```

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A2	LOCK	SUB	300	1	20/02/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01
A1	WING	MAIN	200	2	20/01/01
B2	ARM	MAIN	50	2	20/04/01
B1	WHEEL	SUB	100	4	20/03/01

ORDER BY

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- 1차로 CATEGORY순으로 내림차순 정렬 후 2차로 AMOUNT순으로 내림차순 정렬하여 조회하기

```
SELECT * FROM PRODUCT_TBL ORDER BY CATEGORY DESC, AMOUNT DESC;
```

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
B1	WHEEL	SUB	100	4	20/03/01
A2	LOCK	SUB	300	1	20/02/01
C2	HEAD	SUB	400	1	20/06/01
A1	WING	MAIN	200	2	20/01/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01

GROUP BY

- GROUP BY
  - ✓ 특정 칼럼 값을 기준으로 전체 행(ROW)을 서브 그룹으로 그룹화
  - ✓ 통계를 내는 경우에 주로 사용됨
  - ✓ GROUP BY절에 명시하지 않는 칼럼은 SELECT절에서 조회가 불가능함
  - ✓ 둘 이상의 칼럼을 이용해 다중 그룹화 진행이 가능함

- GROUP BY 작성 방법
  - ① GROUP BY 이전에 가능한 모든 조건은 WHERE절로 처리(그룹화 할 대상의 개수를 줄이기 위함)
  - ② SELECT문의 실행 순서에 의해 SELECT절에서 지정한 칼럼의 별명은 GROUP BY절에서 사용이 불가능함
  - ③ SELECT절에서 각종 통계 함수를 사용할 수 있음
    - ➢ SUM / AVG / MAX / MIN / COUNT 함수
  - ④ SELECT문에서 조회하고자 하는 칼럼은 반드시 GROUP BY절에 명시되어 있어야 함

GROUP BY

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- CATEGORY와 CATEGORY별 AMOUNT의 합계 조회하기

```
SELECT CATEGORY, SUM(AMOUNT) FROM PRODUCT_TBL GROUP BY CATEGORY;
```

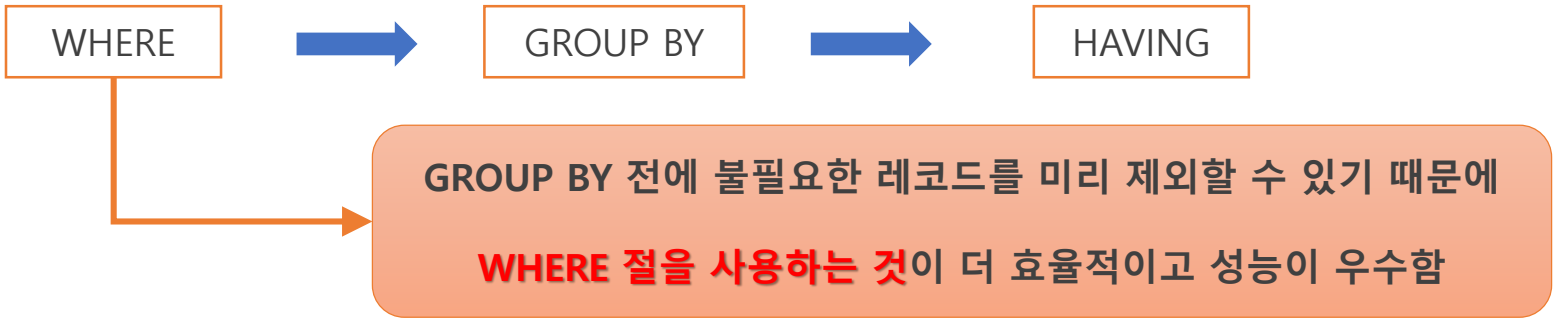
CATEGORY	SUM(AMOUNT)
MAIN	5
SUB	6

HAVING

- HAVING
  - ✓ GROUP BY절에 의해 생성된 서브 그룹을 대상으로 조건을 지정하는 절
  - ✓ 통계 함수를 이용한 조건은 HAVING절에서 처리할 수 있음
    - ➢ 예시) 평균이 XX이상인 자료 조회, 개수가 XX이상인 자료 조회 등

- HAVING절 vs WHERE절
  - ✓ HAVING절 : 그룹화된 결과를 대상으로 조건을 지정할 때 사용
  - ✓ WHERE절 : 그룹화 하지 않아도 처리할 수 있는 조건을 지정할 때 사용
  - ✓ HAVING절과 WHERE절에서 모두 처리되는 조건이 있다면 언제나 WHERE절에서 처리하는 것이 유리함

- 실행 순서



GROUP BY

- PRODUCT_TBL 테이블

CODE	MODEL	CATEGORY	PRICE	AMOUNT	MANUFACTURED
A1	WING	MAIN	200	2	20/01/01
A2	LOCK	SUB	300	1	20/02/01
B1	WHEEL	SUB	100	4	20/03/01
B2	ARM	MAIN	50	2	20/04/01
C1	BODY	MAIN	500	1	20/05/01
C2	HEAD	SUB	400	1	20/06/01

- CATEGORY와 CATEGORY별 AMOUNT의 합계 조회하기(AMOUNT의 합계가 5 이하인 경우만 조회)

```
SELECT CATEGORY, SUM(AMOUNT) FROM PRODUCT_TBL GROUP BY CATEGORY HAVING SUM(AMOUNT) <= 5;
```

CATEGORY	SUM(AMOUNT)
MAIN	5

ROWID

- ROWID
  - ✓ 오라클에서 인덱스(Index)를 생성하기 위해 내부적으로 사용하는 PSEUDO-COLUMN(의사 칼럼)
  - ✓ ROWID는 물리적인 주소(Address)를 가지고 있기 때문에 ROWID를 이용한 단일 블록 접근(Single Block Access)의 경우 가장 빠른 접근 속도를 가짐

P는 묵음!
'슈도 칼럼'으로 읽자.

- ROWID 확인

- ROWID 구조
  - ✓ 6자리 : 데이터 오브젝트 번호(Data Object Number) - 객체 번호(AAAEAb)
  - ✓ 3자리 : 상대적 파일 번호(Relative File Number) - 데이터파일에 할당되는 번호(AAE)
  - ✓ 6자리 : 블록 번호(Block Number) - 데이터 블록의 위치(AAAADN)
  - ✓ 3자리 : 행 번호(Row Number) - 데이터 블록 내 행의 위치(AAA)

ROWNUM

- ROWNUM
  - ✓ 쿼리에 의해 추출된 각 행(ROW)에 부여된 일련번호를 의미
  - ✓ 쿼리에 의해 추출된 결과 집합에 1부터 시작하는 일련번호를 행 순서대로 부여함 (WHERE절을 통해 조건에 맞는 결과 집합을 추출한 뒤에 일련번호를 부여함)
  - ✓ 순서 : FROM → WHERE → ROWNUM 부여 → SELECT → ORDER BY

- ROWNUM 특징
  - ✓ 참조할 수 있으나, 데이터베이스에 따로 저장되지 않는 PSEUDO-COLUMN(의사 칼럼)
  - ✓ 따로 저장된 데이터가 아니기 때문에 사용에 제한이 있음
  - ✓ ROWNUM이 1을 포함하는 조건은 사용 가능함
    - ➢ 가능 조건 예시)
    - ➢ WHERE ROWNUM = 1;
    - ➢ WHERE ROWNUM <= 3;
    - ➢ WHERE ROWNUM BETWEEN 1 AND 3;
  - ✓ ROWNUM이 1을 포함하지 않는 조건은 사용할 수 없음(제한이 있음)
    - ➢ 불가 조건 예시)
    - ➢ WHERE ROWNUM = 2;
    - ➢ WHERE ROWNUM >= 3;
    - ➢ WHERE ROWNUM BETWEEN 3 AND 5;
  - ✓ ROWNUM에 별명(Alias)을 주고 사용하면 모든 범위의 값을 조건으로 사용할 수 있음