suanlab

데이터베이스
Database

ORACLE®

DATABASE

05
정렬과 집합 연산

ORDER BY

UNION, UNION ALL, INTERSECT, MINUS

ORDER BY

- ORDER BY 키워드를 이용해 결과 테이블 내용을 사용자가 원하는 순서로 출력
- ORDER BY 키워드와 함께 정렬 기준이 되는 속성과 정렬 방식을 지정
  - 오름차순(기본): ASC / 내림차순: DESC
  - 널 값은 오름차순에서는 맨 마지막에 출력되고, 내림차순에서는 맨 먼저 출력됨
  - 여러 기준에 따라 정렬하려면 정렬 기준이 되는 속성을 차례대로 제시

```
SELECT first_name, last_name
FROM employees
ORDER BY first_name;
```

```
SELECT first_name, last_name
FROM employees
ORDER BY first_name DESC;
```

```
SELECT department_name
FROM departments
ORDER BY department_name;
```

```
SELECT department_name
FROM departments
ORDER BY department_name DESC;
```

```
SELECT country_id, city
FROM locations
ORDER BY country_id, city;
```

```
SELECT location_id, department_name
FROM departments
ORDER BY location_id DESC, department_name;
```

SQL연산자

- 조건 비교 확장을 위해서 사용되는 SQL 연산자
- BETWEEN 연산자
  - 두 값의 범위에 해당하는 데이터만 출력할 때 사용되는 확장 연산자

```
SELECT *
FROM employees
WHERE employee_id BETWEEN 120 AND 130;
```

```
SELECT *
FROM employees
WHERE salary BETWEEN 10000 AND 12000;
```

- IN 연산자
  - 여러 개의 데이터 값을 지정하여 일치하는 데이터만 출력할 때 사용

```
SELECT *
FROM employees
WHERE first_name IN ('Steven', 'John', 'Peter');
```

```
SELECT *
FROM countries
WHERE country_id IN ('US', 'IL', 'SG');
```

```
SELECT *
FROM locations
WHERE city NOT IN ('Sao Paulo', 'London', 'Southlake');
```

SQL연산자

- IS NULL 연산자
  - 특정 속성의 값이 NULL 값인지를 비교하여 데이터 조회

```
SELECT *
FROM locations
WHERE state_province IS NULL;
```

```
SELECT *
FROM employees
WHERE commission_pct IS NOT NULL;
```

- LIKE 연산자
  - 문자열 속성에서 부분적으로 일치하는 것만 출력할 때 사용

기호	설명
%	0개 이상의 문자 (문자의 내용과 개수는 상관 없음)
_	1개의 문자 (문자의 내용은 상관 없음)

기호	설명
LIKE ‘data%’	data로 시작하는 문자열 (길이 상관 없이 data로 시작)
LIKE ‘%data’	data로 끝나는 문자열 (길이 상관 없이 data로 끝남)
LIKE ‘%data%’	data가 포함된 문자열 (길이 상관 없이 data가 포함)
LIKE ‘data____’	data로 시작하는 8자리 문자열
LIKE ‘____data’	data로 끝나는 8자리 문자열

```
SELECT *
FROM locations
WHERE city LIKE 'South%';
```

```
SELECT *
FROM locations
WHERE street_address LIKE '%St';
```

```
SELECT *
FROM locations
WHERE city LIKE 'South____';
```

집합 연산자

- 연산자 앞뒤의 값을 비교하여 데이터 조회

연산자	설명
UNION	합집합 (중복 제외)
UNION ALL	합집합 (중복 포함)
MINUS	차집합
INTERSECT	교집합

```
SELECT employee_id, first_name, department_id
FROM employees
WHERE department_id = 60
UNION
SELECT employee_id, first_name, department_id
FROM employees
WHERE department_id = 100;
```

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id <= 160
UNION
SELECT employee_id, first_name
FROM employees
WHERE employee_id >= 140;
```

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id <= 160
UNION ALL
SELECT employee_id, first_name
FROM employees
WHERE employee_id >= 140;
```

집합 연산자

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id <= 160
MINUS
SELECT employee_id, first_name
FROM employees
WHERE employee_id >= 140;
```

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id <= 160
INTERSECT
SELECT employee_id, first_name
FROM employees
WHERE employee_id >= 140;
```

[실습] 정렬, 집합 연산자

- jobs 테이블에서 job_title 기준으로 정렬하여 직업 조회

- countries 테이블에서 country_name 기준 내림차순으로 정렬하여 조회

- employees 테이블에서 salary가 10000에서 12000 사이인 직원 조회

- employees 테이블에서 job_id가 'IT_PROG'와 'ST_MAN'인 직원 조회

- employees 테이블에서 manager_id가 NULL인 직원 조회

- departments 테이블에서 manager_id가 NULL이 아닌 부서 조회

[실습] 정렬, 집합 연산자

- employees 테이블에서 job_id가 'AD'로 시작하는 직원 조회

- employees 테이블의 first_name에서 'ni'를 포함하는 직원 조회

- locations 테이블에서 location_id, street_address, city에 대해 location_id가 3000 이하인 데이터와 2000 이상인 데이터를 합집합(중복포함), 차집합, 교집합 한 결과 조회


suanlab
이수안 컴퓨터 연구소
suan computer laboratory