

# database-5

☰ 태그	
📅 날짜	@2023년 6월 12일

## SQL로 테이블에 있는 데이터를 조회하는 기본 문법

### 내용

- select
- project attributes
- selection condition
- join condition (join은 차후 영상에서 더 자세히 다룹니다)
- AS로 별칭짓기
- DISTINCT 키워드
- LIKE로 조회하기
- (asterisk) 사용하기
- where절 없는 select

### 조회

SELECT statement

- id가 9인 임직원의 이름과 직군

SELECT attribute(s) FROM table(s) [WHERE condition(s)];

SELECT name, position FROM employee WHERE id = 9;

projection attribute

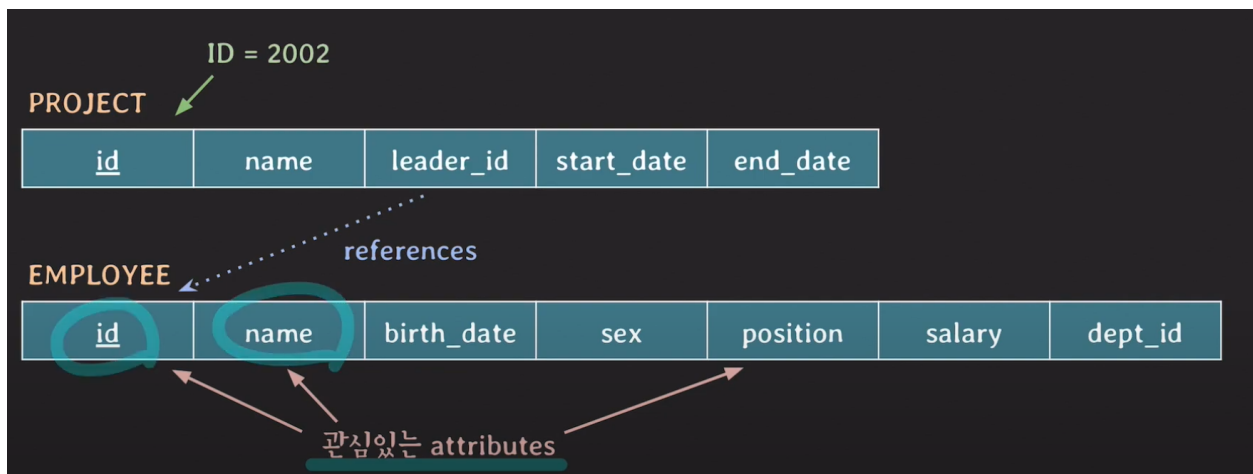
selection condition

```

+-----+-----+
| name   | position |
+-----+-----+
| HENRY  | HR       |
+-----+-----+
1 row in set (0.01 sec)

```

- project 2002를 리딩(leading)하고 있는 임직원의 id와 이름과 직국



```

SELECT employee.id, employee.name, position FROM project, employee
WHERE project.id = 2002 and project.leader_id = employee.id;
      selection condition      join condition

```

```

+----+-----+-----+
| id | name   | position |
+----+-----+-----+
| 13 | JISUNG | PO       |
+----+-----+-----+
1 row in set (0.00 sec)

```

employee.id, employee.name 로 작성한 이유는 project와 employee의 attribute명이 중복되기 때문에 명확하게 작성

## AS

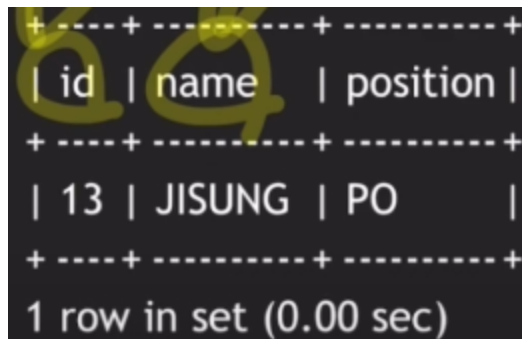
- 테이블이나 attribute에 별칭(alias)을 붙일 때 사용한다.
- AS는 생략 가능하다.

```
SELECT employee.id, employee.name, position FROM project, employee
      WHERE project.id = 2002 and project.leader_id = employee.id;
```

이름을 여러번 쓰기 번거롭다.

AS 로 별칭

```
SELECT E.id, E.name, position FROM project AS P, employee AS E
      WHERE P.id = 2002 and P.leader_id = E.id;
```



id	name	position
13	JISUNG	PO

1 row in set (0.00 sec)

```
SELECT E.id AS leader_id, E.name AS leader_name, position
      FROM project AS P, employee AS E
      WHERE P.id = 2002 and P.leader_id = E.id;
```

아래도 동일하게 작동

```
SELECT E.id leader_id, E.name leader_name, position
      FROM project P, employee E
      WHERE P.id = 2002 and P.leader_id = E.id;
```

```

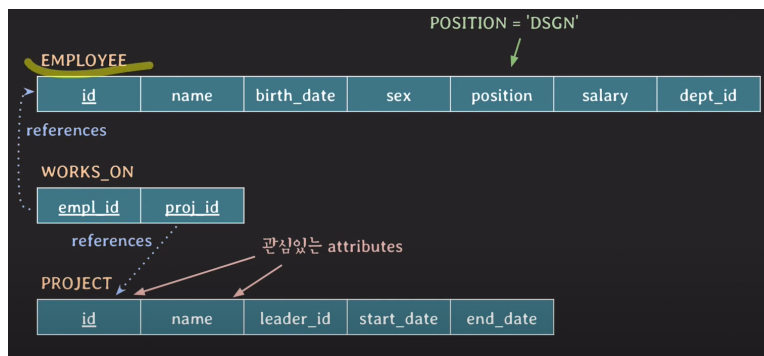
+-----+-----+-----+
| leader_id | leader_name | position |
+-----+-----+-----+
| 13        | JISUNG      | PO       |
+-----+-----+-----+
1 row in set (0.00 sec)

```

## DISTINCT :

select 결과에서 중복되는 tuple은 제외하고 싶을때 사용

- 디자이너들이 참여하고 있는 프로젝트들의 ID와 이름



works\_on이 emp와 project 의 연결고리 역할

```

SELECT P.id, P.name
  FROM employee AS E, works_on AS W, project AS P
 WHERE E.position = 'DSGN' and E.id = W.empl_id and W.proj_id = P.id;

```

```

+-----+-----+
| id    | name                                     |
+-----+-----+
| 2002   | 확장성 있게 백엔드 리팩토링           |
| 2003   | 홈페이지 UI 개선                       |
| 2003   | 홈페이지 UI 개선                       |
+-----+-----+

```

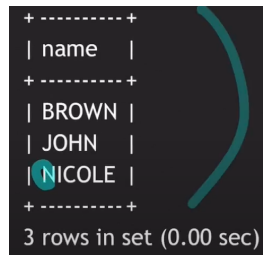
중복된 튜플이 나오게 되는데 이 중복 튜플을 제거하기 위해 사용된다.

```
SELECT DISTINCT P.id, P.name
FROM employee AS E, works_on AS W, project AS P
WHERE E.position = 'DSGN' and E.id = W.empl_id and W.proj_id = P.id;
```

## LIKE

- 이름이 N으로 시작하거나 N으로 끝나는 임직원들의 이름을 알고 싶다.

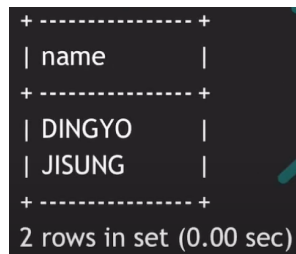
```
SELECT name FROM employee WHERE name LIKE 'N%' or name LIKE '%N';
```



```
+-----+
| name   |
+-----+
| BROWN  |
| JOHN   |
| NICOLE |
+-----+
3 rows in set (0.00 sec)
```

- 이름에 NG가 들어가는 임직원

```
SELECT name FROM employee WHERE name LIKE '%NG%';
```



```
+-----+
| name   |
+-----+
| DINGYO |
| JISUNG |
+-----+
2 rows in set (0.00 sec)
```

- 이름이 J로 시작하는, 총 네 글자의 이름을 가지는 임직원의 이름

```
SELECT name FROM employee WHERE name LIKE 'J____';
```

+	-----	+
	name	
+	-----	+
	JANE	
	JOHN	
+	-----	+

- %로 시작하거나 \_로 끝나는 프로젝트 이름을 찾고 싶다면?

```
SELECT name FROM project WHERE name LIKE '\%' or name LIKE '%\_';
```

// \ escape 문자

LIKE 정리		
항목	설명	
LIKE	문자열 pattern matching에 사용	
reserved character	%	0개 이상의 임의의 개수를 가지는 문자들을 의미
	_	하나의 문자를 의미
escape character	\	예약 문자를 escape시켜서 문자 본연의 문자로 사용하고 싶을 때 사용

## \*(asterisk)

: 선택된 tuples의 모든 attributes를 보여주고 싶을 때 사용한다

- id가 9인 임직원의 모든 attributes를 알고 싶다.

```
SELECT * FROM employee WHERE id = 9;
```

id	name	birth_date	sex	position	salary	dept_id
9	HENRY	1982-05-20	M	HR	82000000	1002

1 row in set (0.00 sec)

```
mysql> SELECT *  
-> FROM project, employee  
-> WHERE project.id = 2002 and project.leader_id = employee.id;
```

id	name	leader_id	start_date	end_date	id	name	birth_date	sex	position	salary	dept_id
2002	리팩토링	13	2022-01-23	2022-03-23	13	JISUNG	1989-07-07	M	PO	90000000	1005

1 row in set (0.00 sec)

## SELECT without WHERE

: 테이블에 있는 모든 tuple를 반환한다.

- 모든 임직원의 이름과 생일을 알고 싶다

```
SELECT name, birth_date FROM employee;
```

```
+-----+-----+
| name  | birth_date |
+-----+-----+
| MESSI | 1987-02-01 |
| JANE  | 1996-05-05 |
|      | ...        |
| SAM   | 1992-08-04 |
+-----+-----+
14 rows in set (0.00 sec)
```

## 주의사항

- SELECT로 조회할 때 조건들을 포함해서 조회한다면 이 조건들과 관련된 attributes에 index가 걸려 있어야 한다. 그렇지 않다면 데이터가 많아질수록 조회 속도가 느려진다.
  - `SELECT * FROM employee WHERE position = 'dev_back';`
- 오늘 내용은 MySQL 기준이다.
  - 다른 RDBMS의 SQL 문법은 다를 수 있다.
- SELECT와 관련하여 대표적으로 중요한 기본기들을 담고있다.