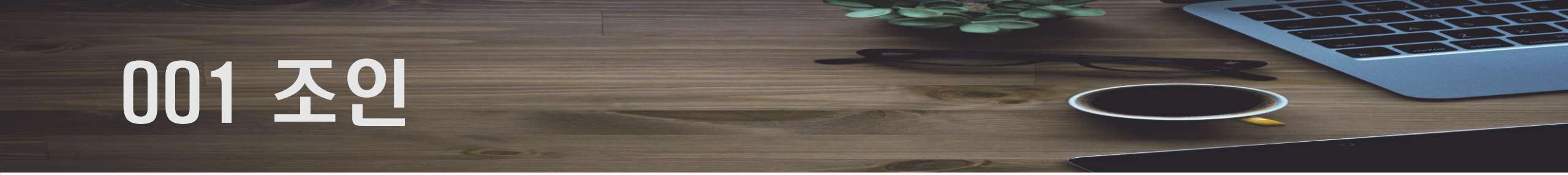
# 6장 조인

강사 금영석

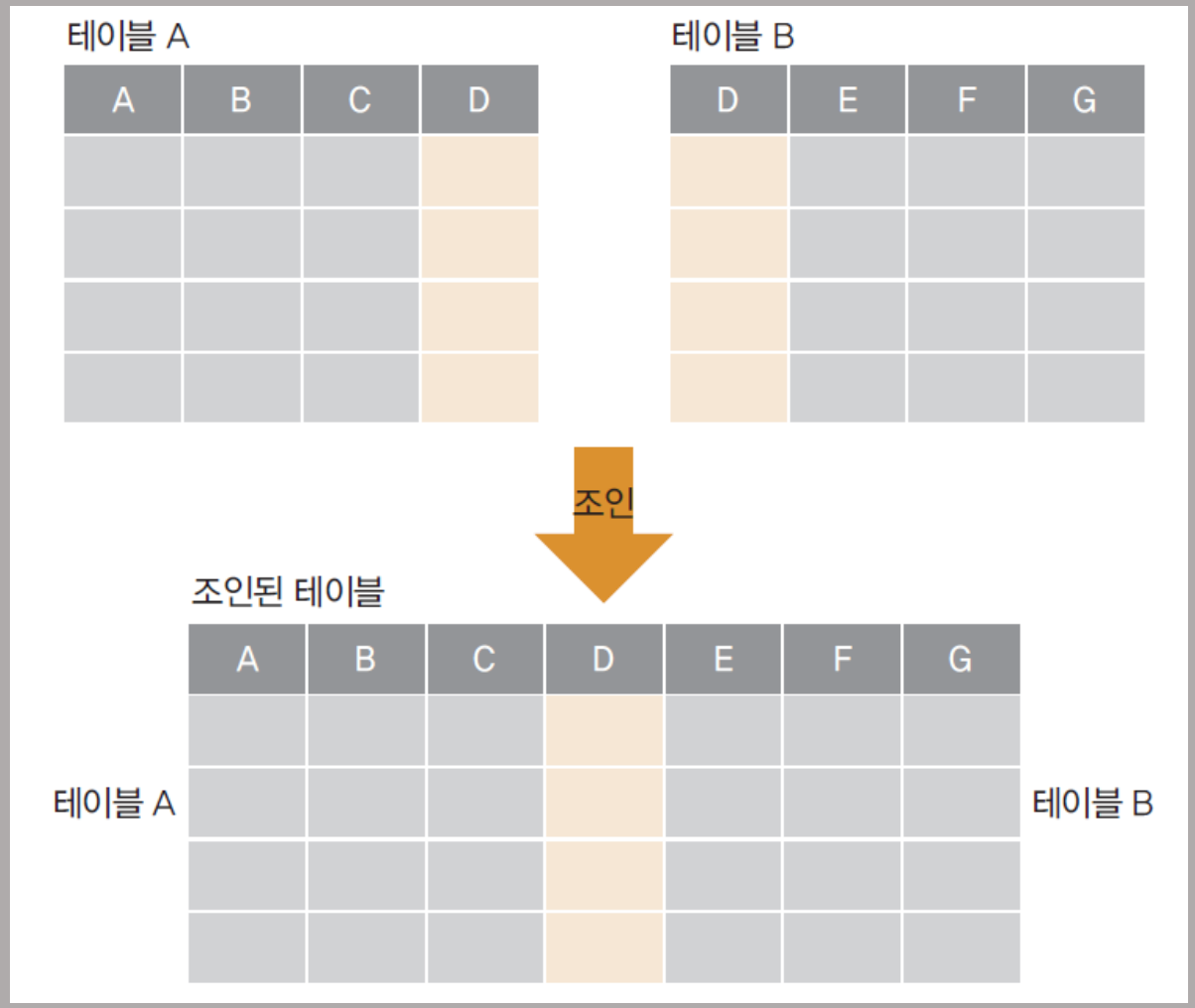
# CONTENT



# 001 조인

- 우리는 계속해서 언급하는 '관계형 데이터베이스' 라는 명칭은 테이블들이 관계(Relationship)를 맺고 조작되는 원리에서 유래한다.
- 테이블에는 각 유형에 맞는 데이터가 저장되어 있고 테이블들은 특정한 규칙에 따라 상호 관계를 맺는다.
- 데이터는 테이블에 흩어져 저장되어 있으므로 사용자가 원하는 형태로 데이터를 조작하려면 특별한 방법이 필요한데 이를 위해 사용하는 기법이 조인이다.
- 조인은 한 개 이상의 테이블과 테이블을 서로 연결하여 사용하는 기법을 말한다.

# 001 조인



# 001 조인

## ❖조인 기법의 종류

- 실무에서 가장 많이 쓰는 기법은 동등 조인, 외부 조인, 자체 조인이다.

| 조인 기법 | 설명 |
|---|---|
| 곱집합(Cartesian product) | 가능한 모든 행을 조인한다. |
| 동등 조인(equi join) | 조인 조건이 정확히 일치하는 경우에 결과를 출력한다. |
| 비동등 조인(non equi join) | 조인 조건이 정확히 일치하지 않는 경우에 결과를 출력한다. |
| 외부 조인(outer join) | 조인 조건이 정확히 일치하지 않아도 모든 결과를 출력한다. |
| 자체 조인(self join) | 자체 테이블에서 조인하고자 할 때 사용한다. |

# 002 동등 조인

❖동등 조인

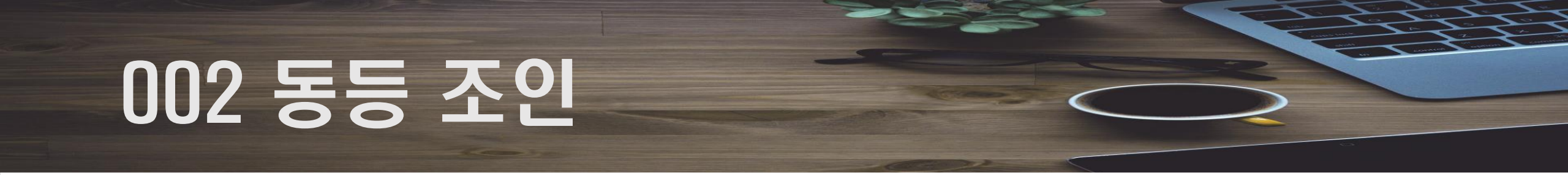
- 동등 조인은 조인 조건이 정확히 일치하는 경우에 결과를 출력한다.

```
SELECT [테이블 이름1].[열 이름1], [테이블 이름2].[열 이름2]....
FROM [테이블 이름1], [테이블 이름2]
WHERE [테이블 이름1].[열 이름1] = [테이블 이름2].[열 이름2]
```

# 002 동등 조인

```
SELECT *
FROM student, department
WHERE student.dept_id = department.dept_id;
```

| STU_ID | RESIDENT_ID | YEAR | DEPT_ID | DEPT_ID_1 | DEPT_NAME | OFFICE |
|---|---|---|---|---|---|---|
| 1292001 | 900424 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292002 | 900305 | 3 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292003 | 991021 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292004 | 930504 | 4 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292006 | 961101 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292007 | 920214 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292009 | 931224 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292010 | 980824 | 1 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292011 | 970625 | 1 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292012 | 940721 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |



# 002 동등 조인

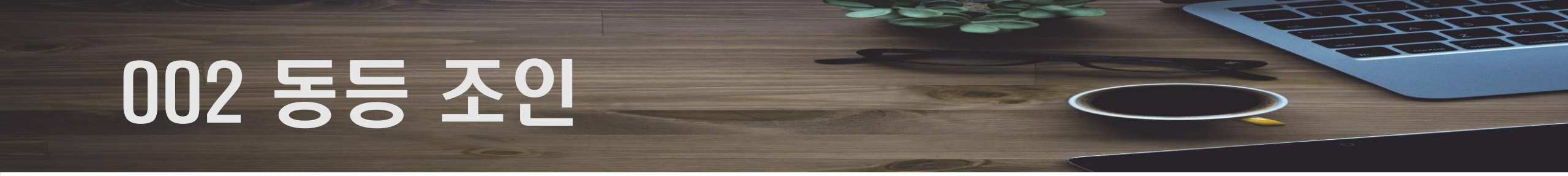
❖이름이 긴 테이블은 별칭을 이용하면 편리하게 사용할 수 있다.

```
SELECT *
FROM [테이블 이름1] [별칭1], [테이블 이름2] [별칭2]
WHERE [별칭1].[열 이름1] = [별칭2].[열 이름2];
```

# 002 동등 조인

```
SELECT *
FROM student s, department d
WHERE s.dept_id = d.dept_id;
```

| STU_ID | RESIDENT_ID | YEAR | DEPT_ID | DEPT_ID_1 | DEPT_NAME | OFFICE |
|---|---|---|---|---|---|---|
| 1292001 | 900424 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292002 | 900305 | 3 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292003 | 991021 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292004 | 930504 | 4 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292006 | 961101 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292007 | 920214 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292009 | 931224 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 1292010 | 980824 | 1 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 1292011 | 970625 | 1 | 2 | 2 | 전기전자공학부 | 403호 |
| 1292012 | 940721 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |



# 002 동등 조인

❖테이블 3개를 연결할 경우

```
SELECT *
FROM [테이블 이름1] [별칭1], [테이블 이름2] [별칭2], [테이블 이름3] [별칭3]
WHERE [별칭1].[열 이름1] = [별칭2].[열 이름2]
AND [별칭2].[열 이름3] = [별칭3].[열 이름4];
```

# 002 동등 조인

```
CREATE TABLE teacher (
    name varchar2(20) not null,
    office varchar2(50) not null
);
```

```
insert into teacher values('홍길동', '302호');
insert into teacher values('이순신', '403호');
insert into teacher values('유관순', '303호');
```

# 002 동등 조인

```
select * from student s, department d, teacher t
WHERE s.dept_id = d.dept_id AND d.office = t.office;
```

| STU_ID | RESIDENT_ID | DEPT_ID | YEAR | DEPT_ID_1 | DEPT_NAME | OFFICE | NAME | OFFICE_1 |
|---|---|---|---|---|---|---|---|---|
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292003 | 991021 | 3 | 4 | 3 | 데이터 싸이언스 | 303호 | 유관순 | 303호 |
| 1292006 | 961101 | 3 | 2 | 3 | 데이터 싸이언스 | 303호 | 유관순 | 303호 |
| 1292009 | 931224 | 3 | 4 | 3 | 데이터 싸이언스 | 303호 | 유관순 | 303호 |
| 1292012 | 940721 | 3 | 2 | 3 | 데이터 싸이언스 | 303호 | 유관순 | 303호 |
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292001 | 900424 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292004 | 930504 | 1 | 4 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292007 | 920214 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292010 | 980824 | 1 | 1 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |

# 003 외부 조인

❖외부 조인

- 외부 조인은 조건을 만족하지 않는 행도 모두 출력하기 위한 조인 기법이다.
- 데이터의 양이 적은 쪽에 (+) 기호를 사용한다.

```
SELECT [테이블 이름1].[열 이름1], [테이블 이름2].[열 이름2].....
FROM [테이블 이름1], [테이블 이름2]
WHERE [테이블 이름1].[열 이름1] = [테이블 이름2].[열 이름2](+);
```

# 003 외부 조인

```
DELETE FROM teacher WHERE name = '유관순';
```

```
select * from student s, department d, teacher t
WHERE s.dept_id = d.dept_id AND d.office = t.office;
```

303호 데이터 누락

| STU_ID | RESIDENT_ID | DEPT_ID | YEAR | DEPT_ID_1 | DEPT_NAME | OFFICE | NAME | OFFICE_1 |
|---|---|---|---|---|---|---|---|---|
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292001 | 900424 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292004 | 930504 | 1 | 4 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292007 | 920214 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292010 | 980824 | 1 | 1 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |

# 003 외부 조인

```
select * from student s, department d, teacher t
WHERE s.dept_id = d.dept_id AND d.office = t.office(+);
```

| STU_ID | RESIDENT_ID | DEPT_ID | YEAR | DEPT_ID_1 | DEPT_NAME | OFFICE | NAME | OFFICE_1 |
|---|---|---|---|---|---|---|---|---|
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292002 | 900305 | 2 | 3 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292005 | 970105 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292008 | 960305 | 2 | 2 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292011 | 970625 | 2 | 1 | 2 | 전기전자공학부 | 403호 | 이순신 | 403호 |
| 1292001 | 900424 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292004 | 930504 | 1 | 4 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292007 | 920214 | 1 | 3 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292010 | 980824 | 1 | 1 | 1 | 컴퓨터 과학부 | 302호 | 홍길동 | 302호 |
| 1292003 | 991021 | 3 | 4 | 3 | 데이터 싸이언스 | 303호 | (null) | (null) |
| 1292006 | 961101 | 3 | 2 | 3 | 데이터 싸이언스 | 303호 | (null) | (null) |
| 1292009 | 931224 | 3 | 4 | 3 | 데이터 싸이언스 | 303호 | (null) | (null) |
| 1292012 | 940721 | 3 | 2 | 3 | 데이터 싸이언스 | 303호 | (null) | (null) |

# 003 외부 조인

- 외부 조인은 연결하는 방법에 따라 레프트 아웃터 조인(Left Outer Join), 라이트 아웃터 조인(Right Outer Join)이라고 부르기도 한다.
- 오라클  데이터 베이스에서는 (+) 가 붙지 않는 쪽을 기준으로 부른다.
- 예를 들어 A = B(+) 라면 오른쪽에 null이 생성되어 왼쪽이 기준이 되기 때문에 레프트 아웃터 조인이라고 부른다.

# 004 자체 조인

❖자체 조인
- 자기 자신의 데이터와 조인하는 방식이다.
- 자기 자신의 테이블을 조인하는 것을 자체 조인(Self Join) 이라고 한다.
- 자체 조인을 사용하려면 별칭을 사용해야 한다.

```
SELECT [테이블 별칭1].[열 이름1], [테이블 별칭2].[열이름],...
FROM [테이블 이름1] [별칭 1], [테이블 이름1] [별칭2]
WHERE [테이블 별칭1].[열 이름1] = [테이블 별칭2].[열 이름2]
```

# 004 자체 조인

```
SELECT A.stu_id, A.year, B.stu_id || '학번 ' || B.year || '학년'
FROM student A, student B
WHERE A.stu_id  = B.stu_id;
```

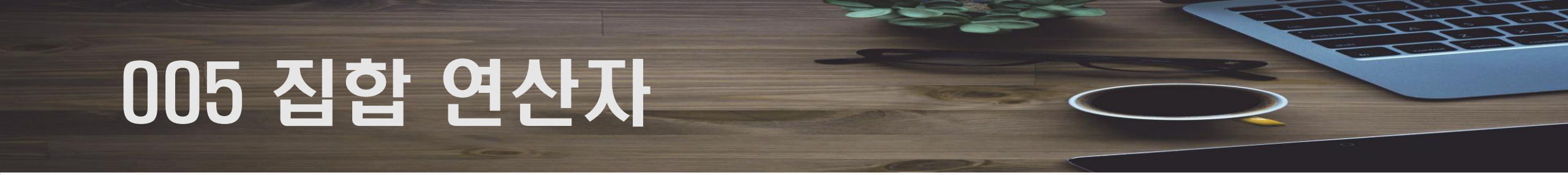
| | STU_ID | YEAR | B.STU_ID\|\|'학번'\|\|B.YEAR\|\|'학년' |
|---|---|---|---|
| 1 | 1292001 | 3 | 1292001학번 3학년 |
| 2 | 1292002 | 3 | 1292002학번 3학년 |
| 3 | 1292003 | 4 | 1292003학번 4학년 |
| 4 | 1292004 | 4 | 1292004학번 4학년 |
| 5 | 1292005 | 2 | 1292005학번 2학년 |
| 6 | 1292006 | 2 | 1292006학번 2학년 |
| 7 | 1292007 | 3 | 1292007학번 3학년 |
| 8 | 1292008 | 2 | 1292008학번 2학년 |
| 9 | 1292009 | 4 | 1292009학번 4학년 |
| 10 | 1292010 | 1 | 1292010학번 1학년 |
| 11 | 1292011 | 1 | 1292011학번 1학년 |
| 12 | 1292012 | 2 | 1292012학번 2학년 |

# 005 집합 연산자

❖집합 연산자

- 조인 기법 외에도 테이블에서 데이터를 조회 하는 방법이 한 가지 더 있는데 바로 집합 연산자(Set operators)를 이용하는 방법이다.
- 집한 연산자는 SELECT 문을 여러 개 연결하여 작성하며, 각 SELECT 문의 조회 결과를 하나로 합치거나 분리할 수 있다.
- 집한 연산자는 합집합, 교집합, 차집합의 논리와 같다.

| 종류 | 설명 | 집합 종류 |
|---|---|---|
| UNION | SELECT 문의 조회 결과의 합집합, 중복 되는 행은 한번만 출력한다. | 합집합 |
| UNION ALL | SELECT 문의 조회 결과의 합집합, 중복 되는 행도 그대로 출력한다. | 합집합 |
| INTERSECT | SELECT 문의 조회 결과의 교집합, 중복되는 행만 출력한다. | 교집합 |
| MINUS | 첫 번째 SELECT 문의 조회 결과에서 두 번째 조회 결과를 뺀다. | 차집합 |



# 005 집합 연산자

SELECT [열 이름1], [열 이름2], [열 이름3],....
FROM [테이블 이름]
[집합 연산자 (UNION, UNION ALL, INTERSECT, MINUS)]
SELECT [열 이름1], [열 이름2], [열 이름3],....
FROM [테이블 이름]
ORDER BY [열 이름] [ASC or DESC];

# 005 집합 연산자

❖집합 연산자의 규칙

- 첫번째 SELECT 문에서 기술한 열과 두 번째 SELECT 문에서 기술한 열은 왼쪽부터 순서대로 일대일로 대응하며, 열 개수와 데이터 타입이 일치 해야 한다.
- 열의 순서가 다르거나 데이터 타입이 일치하지 않으면 오류가 발생한다.
- SELECT 문에 대한 연산은 위에서 아래로 수행된다.
- ORDER BY 절은 SELECT  문의 맨 끝에 기술한다.

# 005 집합 연산자

```
SELECT dept_id FROM student
UNION
SELECT dept_id FROM department;
```

| | DEPT_ID |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

# 005 집합 연산자

```
SELECT dept_id FROM student
UNION ALL
SELECT dept_id FROM department;
```

| | DEPT_ID |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 1 |
| 5 | 2 |
| 6 | 3 |
| 7 | 1 |
| 8 | 2 |
| 9 | 3 |
| 10 | 1 |
| 11 | 2 |
| 12 | 3 |
| 13 | 1 |
| 14 | 2 |
| 15 | 3 |

# 005 집합 연산자

```
SELECT dept_id FROM student
INTERSECT
SELECT dept_id FROM department;
```

| | DEPT_ID |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

# 005 집합 연산자

```
SELECT dept_id FROM student
MINUS
SELECT dept_id FROM department;
```

# 006 실습

```
1    SELECT   *
2    FROM     student, department
3    WHERE    student.dept_id = department.dept_id
4
```

| | stu_id | resident_id | year | dept_id | dept_id | dept_name | office |
|---|---|---|---|---|---|---|---|
| 1 | 1292001 | 900424 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 2 | 1292002 | 900305 | 3 | 2 | 2 | 전기전자 공학부 | 403호 |
| 3 | 1292003 | 991021 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 4 | 1292004 | 930504 | 4 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 5 | 1292005 | 970105 | 2 | 2 | 2 | 전기전자 공학부 | 403호 |
| 6 | 1292006 | 961101 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |

# 006 실습

```
1    SELECT  *
2    FROM    student as std, department as dept
3    WHERE   std.dept_id = dept.dept_id
4
```

| | stu_id | resident_id | year | dept_id | dept_id | dept_name | office |
|---|---|---|---|---|---|---|---|
| 1 | 1292001 | 900424 | 3 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 2 | 1292002 | 900305 | 3 | 2 | 2 | 전기전자 공학부 | 403호 |
| 3 | 1292003 | 991021 | 4 | 3 | 3 | 데이터 싸이언스 | 303호 |
| 4 | 1292004 | 930504 | 4 | 1 | 1 | 컴퓨터 과학부 | 302호 |
| 5 | 1292005 | 970105 | 2 | 2 | 2 | 전기전자 공학부 | 403호 |
| 6 | 1292006 | 961101 | 2 | 3 | 3 | 데이터 싸이언스 | 303호 |

# 006 실습

```
1    SELECT  std.stu_id as '학번', dept.dept_name as 학과
2    FROM    student as std, department as dept
3    WHERE   std.dept_id = dept.dept_id
4
```

| | 학번 | 학과 |
|---|---|---|
| 1 | 1292001 | 컴퓨터 과학부 |
| 2 | 1292002 | 전기전자 공학부 |
| 3 | 1292003 | 데이터 싸이언스 |
| 4 | 1292004 | 컴퓨터 과학부 |
| 5 | 1292005 | 전기전자 공학부 |
| 6 | 1292006 | 데이터 싸이언스 |

# 006 실습

```
1    SELECT   std.stu_id as '학번', dept.dept_name as 학과
2    FROM     student as std, department as dept
3    WHERE    std.dept_id = dept.dept_id
4
```

| | 학번 | 학과 |
|---|---|---|
| 1 | 1292001 | 컴퓨터 과학부 |
| 2 | 1292002 | 전기전자 공학부 |
| 3 | 1292003 | 데이터 싸이언스 |
| 4 | 1292004 | 컴퓨터 과학부 |
| 5 | 1292005 | 전기전자 공학부 |
| 6 | 1292006 | 데이터 싸이언스 |

# 006 실습

```
1    SELECT   std.stu_id as '학번', dept.dept_name as 학과
2    FROM     student as std, department as dept
3    WHERE    std.dept_id = dept.dept_id
4    AND      dept.dept_id=2
5    OR       dept.dept_id=1
```

| | 학번 | 학과 |
|---|---|---|
| 1 | 1292002 | 전기전자 공학부 |
| 2 | 1292005 | 전기전자 공학부 |
| 3 | 1292008 | 전기전자 공학부 |
| 4 | 1292011 | 전기전자 공학부 |
| 5 | 1292001 | 컴퓨터 과학부 |
| 6 | 1292002 | 컴퓨터 과학부 |

# 006 실습

```
1    SELECT   std.stu_id as '학번'
2    FROM     student as std, department as dept
3    WHERE    std.dept_id = dept.dept_id
4    AND      dept.dept_id=2
5    OR       dept.dept_id=1
```

| | 학번 |
|---|---|
| 1 | 1292002 |
| 2 | 1292005 |
| 3 | 1292008 |
| 4 | 1292011 |
| 5 | 1292001 |
| 6 | 1292002 |
| 7 | 1292003 |

# 006 실습

```
1    SELECT   std.stu_id as '학번'
2    FROM     student as std, department as dept
3    WHERE    std.dept_id = dept.dept_id
4    AND      dept.dept_id=2
5    OR       dept.dept_id=1
```

| | 학번 |
|---|---|
| 1 | 1292002 |
| 2 | 1292005 |
| 3 | 1292008 |
| 4 | 1292011 |
| 5 | 1292001 |
| 6 | 1292002 |
| 7 | 1292003 |

# 007 QUIZ

| | stu_id | resident_id | dept_id+1 |
|---|---|---|---|
| 1 | 1292005 | 970105 | 3 |
| 2 | 1292008 | 960305 | 3 |
| 3 | 1292011 | 970625 | 3 |

# 007 QUIZ

| | 학번 | 학과 | 호수 |
|---|---|---|---|
| 1 | 1292001 | 컴퓨터 과학부 | 302호 |
| 2 | 1292002 | 전기전자 공학부 | 403호 |
| 3 | 1292003 | 데이터 싸이언스 | 303호 |
| 4 | 1292004 | 컴퓨터 과학부 | 302호 |
| 5 | 1292005 | 전기전자 공학부 | 403호 |
| 6 | 1292006 | 데이터 싸이언스 | 303호 |


수고하셨습니다.