



고급 SQL



Database Laboratory

# 차례

---

- ▶ 예시 데이터
- ▶ 기본적인 질의 구조 및 연산자
- ▶ 집합연산자 & 조인 (JOIN)
- ▶ 집계 함수 (Aggregate Functions)
- ▶ 중첩 부질의 (Nested Subqueries)

# 예시 데이터

## ▶ department

```
CREATE TABLE department (  
    dept_name VARCHAR(20),  
    building VARCHAR(15),  
    budget INT,  
    PRIMARY KEY(dept_name)  
);
```

dept_name	building	budget
Comp.Sci.	Taylor	100000
Biology	Watson	90000
Elec.Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

# 예시 데이터

## ▶ instructor

```
CREATE TABLE instructor (  
  id          VARCHAR(5),  
  name        VARCHAR(20) NOT NULL,  
  dept_name   VARCHAR(20),  
  salary      INT,  
  PRIMARY KEY(id)  
);
```

id	name	dept_name	salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp.Sci.	75000
98345	Kim	Elec.Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp.Sci.	65000
58583	Clifieri	History	62000
83821	Brandt	Comp.Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

# 예시 데이터

## ▶ student

```
CREATE TABLE student (  
    id          VARCHAR(5),  
    name        VARCHAR(20) NOT NULL,  
    dept_name   VARCHAR(20),  
    tot_cred    INT,  
    PRIMARY KEY(id)  
);
```

id	name	dept_name	tot_cred
00128	Zhang	Comp.Sci.	102
12345	Shankar	Comp.Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp.Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp.Sci.	58
76653	Aoi	Elec.Eng.	60
98765	Bourikas	Elec.Eng.	98

# 기본적인 질의 구조 및 연산자

---

## ▶ 기본 구조

```
SELECT ... FROM ... WHERE ...
```

## ▶ SELECT 절

- ▶ 질의의 결과로 바라는 컬럼명

## ▶ FROM 절

- ▶ 질의 평가에 대한 모든 테이블

## ▶ WHERE 절

- ▶ 질의 결과가 만족해야 하는 조건

# 기본적인 질의 구조 및 연산자

## ▶ SELECT 절

### ▶ 질의의 결과로 바라는 컬럼명

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT * FROM department
```

	dept_name	building	budget
1	Biology	Watson	90000
2	Comp.Sci.	Taylor	100000
3	Elec.Eng.	Taylor	85000
4	Finance	Painter	120000
5	History	Painter	50000
6	Music	Packard	80000
7	Physics	Watson	70000

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT dept_name FROM department
```

	dept_name
1	Biology
2	Comp.Sci.
3	Elec.Eng.
4	Finance
5	History
6	Music
7	Physics

# 기본적인 질의 구조 및 연산자

## ▶ FROM 절

- ▶ 질의 평가에 대한 모든 테이블

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT * FROM student, instructor
```

결과 메시지

	id	name	dept_name	tot_cred	id	name	dept_name	salary
1	00128	Zhang	Comp.Sci.	102	10101	Srinivasan	Comp.Sci.	65000
2	12345	Shankar	Comp.Sci.	32	10101	Srinivasan	Comp.Sci.	65000
3	19991	Brandt	History	80	10101	Srinivasan	Comp.Sci.	65000
4	23121	Chavez	Finance	110	10101	Srinivasan	Comp.Sci.	65000
5	44553	Peltier	Physics	56	10101	Srinivasan	Comp.Sci.	65000
6	45678	Levy	Physics	46	10101	Srinivasan	Comp.Sci.	65000
7	54321	Williams	Comp.Sci.	54	10101	Srinivasan	Comp.Sci.	65000
8	55739	Sanchez	Music	38	10101	Srinivasan	Comp.Sci.	65000
9	70557	Snow	Physics	0	10101	Srinivasan	Comp.Sci.	65000
10	76543	Brown	Comp.Sci.	58	10101	Srinivasan	Comp.Sci.	65000
11	76653	Aoi	Elec.Eng.	60	10101	Srinivasan	Comp.Sci.	65000
12	98765	Bourikas	Elec.Eng.	98	10101	Srinivasan	Comp.Sci.	65000
13	00128	Zhang	Comp.Sci.	102	12121	Wu	Finance	90000
14	12345	Shankar	Comp.Sci.	32	12121	Wu	Finance	90000
15	19991	Brandt	History	80	12121	Wu	Finance	90000
16	23121	Chavez	Finance	110	12121	Wu	Finance	90000
17	44553	Peltier	Physics	56	12121	Wu	Finance	90000
18	45678	Levy	Physics	46	12121	Wu	Finance	90000



# 기본적인 질의 구조 및 연산자

## ▶ WHERE 절

- ▶ 질의 결과가 만족해야 하는 조건

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT * FROM student  
WHERE tot_cred < 100
```

결과		메시지		
	id	name	dept_name	tot_cred
1	12345	Shankar	Comp.Sci.	32
2	19991	Brandt	History	80
3	44553	Peltier	Physics	56
4	45678	Levy	Physics	46
5	54321	Williams	Comp.Sci.	54
6	55739	Sanchez	Music	38
7	70557	Snow	Physics	0
8	76543	Brown	Comp.Sci.	58
9	76653	Aoi	Elec.Eng.	60
10	98765	Bourikas	Elec.Eng.	98

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT * FROM student  
WHERE tot_cred > 100
```

결과		메시지		
	id	name	dept_name	tot_cred
1	00128	Zhang	Comp.Sci.	102
2	23121	Chavez	Finance	110

# 기본적인 질의 구조 및 연산자

## ▶ 재명명 연산

### ▶ 테이블과 컬럼명을 재명명하는 연산

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT i.name, s.name  
FROM instructor AS i, student AS s
```

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT i.name, s.name  
FROM instructor AS i, student AS s  
WHERE i.salary < 70000 AND s.tot_cred > 100
```

	결과	메시지
	name	name
1	Srinivasan	Zhang
2	Wu	Zhang
3	Mozart	Zhang
4	Einstein	Zhang
5	El Said	Zhang
6	Gold	Zhang
7	Katz	Zhang
8	Clifieri	Zhang
9	Singh	Zhang
10	Crick	Zhang
11	Brandt	Zhang
12	Kim	Zhang
13	Srinivasan	Shankar
14	Wu	Shankar
15	Mozart	Shankar
16	Einstein	Shankar
17	El Said	Shankar
18	Gold	Shankar

	결과	메시지
	name	name
1	Srinivasan	Zhang
2	Mozart	Zhang
3	El Said	Zhang
4	Clifieri	Zhang
5	Srinivasan	Chavez
6	Mozart	Chavez
7	El Said	Chavez
8	Clifieri	Chavez

# 기본적인 질의 구조 및 연산자

## ▶ 스트링 연산

- ▶ 문자열 비교를 위한 문자열 매칭 연산자 (LIKE)
  - ▶ percent (%): The % character matches any substring.
  - ▶ underscore (\_): The \_ character matches any character.

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT * FROM department  
WHERE dept_name LIKE '%i%'
```

	dept_name	building	budget
1	Biology	Watson	90000
2	Comp.Sci.	Taylor	100000
3	Finance	Painter	120000
4	History	Painter	50000
5	Music	Packard	80000
6	Physics	Watson	70000

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT * FROM student  
WHERE name LIKE '_h%'
```

	id	name	dept_name	tot_cred
1	00128	Zhang	Comp.Sci.	102
2	12345	Shankar	Comp.Sci.	32
3	23121	Chavez	Finance	110

# 기본적인 질의 구조 및 연산자

## ▶ 데이터 출력의 순서화

- ▶ 각 컬럼에 대해 내림차순(desc) 또는 오름차순(asc)으로 정렬하여 출력
- ▶ 오름차순이 기본 값

```
SQLQuery1.sql - ...\AC9HWUser (52))*  
SELECT * FROM student  
ORDER BY tot_cred
```

	id	name	dept_name	tot_cred
1	70557	Snow	Physics	0
2	12345	Shankar	Comp.Sci.	32
3	55739	Sanchez	Music	38
4	45678	Levy	Physics	46
5	54321	Williams	Comp.Sci.	54
6	44553	Peltier	Physics	56
7	76543	Brown	Comp.Sci.	58
8	76653	Aoi	Elec.Eng.	60
9	19991	Brandt	History	80
10	98765	Bourikas	Elec.Eng.	98
11	00128	Zhang	Comp.Sci.	102
12	23121	Chavez	Finance	110

```
SQLQuery1.sql - ...\AC9HWUser (52))*  
SELECT * FROM student  
ORDER BY tot_cred DESC
```

	id	name	dept_name	tot_cred
1	23121	Chavez	Finance	110
2	00128	Zhang	Comp.Sci.	102
3	98765	Bourikas	Elec.Eng.	98
4	19991	Brandt	History	80
5	76653	Aoi	Elec.Eng.	60
6	76543	Brown	Comp.Sci.	58
7	44553	Peltier	Physics	56
8	54321	Williams	Comp.Sci.	54
9	45678	Levy	Physics	46
10	55739	Sanchez	Music	38
11	12345	Shankar	Comp.Sci.	32
12	70557	Snow	Physics	0

# 집합연산자

## ▶ 기본 구조

```
SELECT ...  
[UNION | UNION ALL | INTERSECT | EXCEPT]  
SELECT ...
```

- ▶ 두 SELECT문의 컬럼 개수와 데이터 타입은 일치해야 함.
- ▶ 검색 결과의 헤더는 앞쪽 SELECT문에 의해 결정된다.

연산자	의미	결과
UNION	합집합	중복을 제거한 결과의 합을 검색
UNION ALL		중복을 포함한 결과의 합을 검색
INTERSECT	교집합	양쪽 모두에서 포함된 행을 검색
EXCEPT	차집합	첫번째 검색 결과에서 두번째 검색 결과를 제외한 나머지를 검색

# 집합연산자

## ▶ UNION, UNION ALL (합집합)

### ▶ 결과의 합을 검색

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT name  
FROM instructor  
UNION  
SELECT name  
FROM student
```

결과 메시지

(23개 행이 영향을 받음)

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT name  
FROM instructor  
UNION ALL  
SELECT name  
FROM student
```

결과 메시지

(24개 행이 영향을 받음)

결과 메시지

	name
1	Aoi
2	Bourikas
3	Brandt
4	Brown
5	Chavez
6	Clifieri
7	Crick
8	Einstein
9	El Said
10	Gold
11	Katz
12	Kim
13	Levy
14	Mozart
15	Peltier
16	Sanchez
17	Shankar
18	Singh
19	Snow
20	Sriniva...
21	Williams
22	Wu
23	Zhang

결과 메시지

	name
7	Katz
8	Clifieri
9	Singh
10	Crick
11	Brandt
12	Kim
13	Zhang
14	Shankar
15	Brandt
16	Chavez
17	Peltier
18	Levy
19	Williams
20	Sanchez
21	Snow
22	Brown
23	Aoi
24	Bourikas

# 집합연산자

---

## ▶ INTERSECT (교집합)

- ▶ 양쪽 모두에서 포함된 데이터를 검색

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT name
FROM instructor
INTERSECT
SELECT name
FROM student
```

결과 메시지

	name
1	Brandt

# 집합연산자

## ▶ EXCEPT (차집합)

- ▶ 첫번째 검색 결과에서 두번째 검색 결과를 제외한 나머지를 검색

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT name  
FROM instructor  
EXCEPT  
SELECT name  
FROM student
```

결과		메시지
	name	
1	Clifier	
2	Crick	
3	Einstein	
4	El Said	
5	Gold	
6	Katz	
7	Kim	
8	Mozart	
9	Singh	
10	Sriniv...	
11	Wu	



# 조인 (JOIN)

---

- ▶ 카티전 곱 (Cartesian Product)
- ▶ INNER JOIN
- ▶ OUTER JOIN(LEFT, RIGHT)

# 조인 (JOIN)

## ▶ 카티전 곱 (Cartesian Product)

- ▶ 두 테이블에 대해서 가능한 모든 경우의 수를 계산

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT *
```

```
FROM student, department
```

결과 메시지

	id	name	dept_name	tot_cred	dept_name	building	budget
1	00128	Zhang	Comp.Sci.	102	Biology	Watson	90000
2	12345	Shankar	Comp.Sci.	32	Biology	Watson	90000
3	19991	Brandt	History	80	Biology	Watson	90000
4	23121	Chavez	Finance	110	Biology	Watson	90000
5	44553	Peltier	Physics	56	Biology	Watson	90000
6	45678	Levy	Physics	46	Biology	Watson	90000
7	54321	Williams	Comp.Sci.	54	Biology	Watson	90000
8	55739	Sanchez	Music	38	Biology	Watson	90000
9	70557	Snow	Physics	0	Biology	Watson	90000
10	76543	Brown	Comp.Sci.	58	Biology	Watson	90000
11	76653	Aoi	Elec.Eng.	60	Biology	Watson	90000
12	98765	Bourikas	Elec.Eng.	98	Biology	Watson	90000
13	00128	Zhang	Comp.Sci.	102	Comp.Sci.	Taylor	100000
14	12345	Shankar	Comp.Sci.	32	Comp.Sci.	Taylor	100000
15	19991	Brandt	History	80	Comp.Sci.	Taylor	100000
16	23121	Chavez	Finance	110	Comp.Sci.	Taylor	100000
17	44553	Peltier	Physics	56	Comp.Sci.	Taylor	100000
18	45678	Levy	Physics	46	Comp.Sci.	Taylor	100000
19	54321	Williams	Comp.Sci.	54	Comp.Sci.	Taylor	100000
20	55739	Sanchez	Music	38	Comp.Sci.	Taylor	100000
21	70557	Snow	Physics	0	Comp.Sci.	Taylor	100000
22	76543	Brown	Comp.Sci.	58	Comp.Sci.	Taylor	100000
23	76653	Aoi	Elec.Eng.	60	Comp.Sci.	Taylor	100000

쿼리가 실행되었습니다.

(local)(10.50 SP2) DESKTOP-7CIAC9HWUser (52) test 00:00:00 84개의 행

# 조인 (JOIN)

## ▶ INNER JOIN (WHERE 절을 이용)

- ▶ 공통 컬럼의 값이 같은 데이터만을 매치하여, 결과로 공통 컬럼을 포함하는 데이터의 집합의 결과

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT *  
FROM student AS s, department AS d  
WHERE s.dept_name = d.dept_name
```

결과 메시지

	id	name	dept_name	tot_cred	dept_name	building	budget
1	00128	Zhang	Comp.Sci.	102	Comp.Sci.	Taylor	100000
2	12345	Shankar	Comp.Sci.	32	Comp.Sci.	Taylor	100000
3	19991	Brandt	History	80	History	Painter	50000
4	23121	Chavez	Finance	110	Finance	Painter	120000
5	44553	Peltier	Physics	56	Physics	Watson	70000
6	45678	Levy	Physics	46	Physics	Watson	70000
7	54321	Williams	Comp.Sci.	54	Comp.Sci.	Taylor	100000
8	55739	Sanchez	Music	38	Music	Packard	80000
9	70557	Snow	Physics	0	Physics	Watson	70000
10	76543	Brown	Comp.Sci.	58	Comp.Sci.	Taylor	100000
11	76653	Aoi	Elec.Eng.	60	Elec.Eng.	Taylor	85000
12	98765	Bourikas	Elec.Eng.	98	Elec.Eng.	Taylor	85000

# 조인 (JOIN)

- ▶ INNER JOIN (INNER JOIN과 ON을 이용)
  - ▶ 공통 컬럼의 값이 같은 데이터만을 매치하여, 결과로 공통 컬럼을 포함하는 데이터의 집합의 결과

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT *  
FROM student AS s INNER JOIN department AS d  
ON s.dept_name = d.dept_name
```

결과 메시지

	id	name	dept_name	tot_cred	dept_name	building	budget
1	00128	Zhang	Comp.Sci.	102	Comp.Sci.	Taylor	100000
2	12345	Shankar	Comp.Sci.	32	Comp.Sci.	Taylor	100000
3	19991	Brandt	History	80	History	Painter	50000
4	23121	Chavez	Finance	110	Finance	Painter	120000
5	44553	Peltier	Physics	56	Physics	Watson	70000
6	45678	Levy	Physics	46	Physics	Watson	70000
7	54321	Williams	Comp.Sci.	54	Comp.Sci.	Taylor	100000
8	55739	Sanchez	Music	38	Music	Packard	80000
9	70557	Snow	Physics	0	Physics	Watson	70000
10	76543	Brown	Comp.Sci.	58	Comp.Sci.	Taylor	100000
11	76653	Aoi	Elec.Eng.	60	Elec.Eng.	Taylor	85000
12	98765	Bourikas	Elec.Eng.	98	Elec.Eng.	Taylor	85000

# 조인 (JOIN)

## ▶ OUTER JOIN(LEFT)

- ▶ 왼쪽의 테이블이 기준

SQLQuery1.sql - ...\AC9HWUser (52))\*

```
SELECT i.name AS 교수, s.name AS 학생  
FROM instructor AS i LEFT JOIN student AS s  
ON i.dept_name = s.dept_name
```



결과



메시지

(26개 행이 영향을 받음)

17	Katz	Brown
18	Clifieri	Brandt
19	Singh	Chavez
20	Crick	NULL
21	Brandt	Zhang
22	Brandt	Shankar

# 조인 (JOIN)

- ▶ OUTER JOIN(RIGHT)
  - ▶ 오른쪽의 테이블이 기준

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT i.name AS 교수, s.name AS 학생  
FROM instructor AS i RIGHT JOIN student AS s  
ON i.dept_name = s.dept_name
```

결과 메시지

	교수	학생
1	Srinivasan	Zhang
2	Katz	Zhang
3	Brandt	Zhang
4	Srinivasan	Shankar
5	Katz	Shankar
6	Brandt	Shankar
7	El Said	Brandt
8	Clifieri	Brandt

결과 메시지

(25개 행이 영향을 받음)

# 집성 함수 (Aggregate Functions)

---

- ▶ AVG
- ▶ MIN
- ▶ MAX
- ▶ SUM
- ▶ COUNT
- ▶ GROUP BY, HAVING 절

# 집성 함수 (Aggregate Functions)

## ▶ AVG

### ▶ 평균값

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT AVG(salary) FROM instructor
```

결과		메시지
(열 이름 없음)		
1	74833	

## ▶ MIN

### ▶ 최소값

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT MIN(salary) FROM instructor
```

결과		메시지
(열 이름 없음)		
1	40000	



# 집성 함수 (Aggregate Functions)

## ▶ MAX

### ▶ 최대값

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT MAX(salary) FROM instructor
```

결과		메시지
(열 이름 없음)		
1	95000	

## ▶ SUM

### ▶ 모두 더한 값

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT SUM(salary) FROM instructor
```

결과		메시지
(열 이름 없음)		
1	898000	

# 집성 함수 (Aggregate Functions)

## ▶ COUNT

### ▶ 데이터의 개수

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT COUNT(salary) FROM instructor
```

결과		메시지	
(열 이름 없음)			
1	12		

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT COUNT(salary) FROM instructor  
WHERE salary > 70000
```

결과		메시지	
(열 이름 없음)			
1	8		

# 집성 함수 (Aggregate Functions)

- ▶ GROUP BY, HAVING 절
  - ▶ GROUP BY : 특정 컬럼으로 그룹 짓는 연산자
  - ▶ HAVING : 그룹 지어진 데이터에 대한 조건을 부여

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT dept_name FROM instructor  
GROUP BY dept_name
```

결과 메시지

	dept_name
1	Biology
2	Comp.Sci.
3	Elec.Eng.
4	Finance
5	History
6	Music
7	Physics

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT dept_name FROM instructor  
GROUP BY dept_name  
HAVING dept_name LIKE '_i%'
```

결과 메시지

	dept_name
1	Biology
2	Finance
3	History

# 중첩 부질의 (Nested Subqueries)

---

## ▶ 중첩 부질의란?

- ▶ 다른 질의 내에 내포되는 SELECT-FROM-WHERE 표현식
- ▶ 부질의의 공통적인 사용은 집합 멤버쉽, 집합 비교 및 집합 수의 테스트를 수행하는 것
- ▶ IN
- ▶ SOME & ALL
- ▶ EXISTS & NOT EXISTS
- ▶ FROM 절의 부질의

# 중첩 부질의 (Nested Subqueries)

## ▶ IN & NOT IN

### ▶ 집합 멤버쉽

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT name, salary  
FROM instructor  
WHERE dept_name IN  
(  
    SELECT dept_name  
    FROM student  
    WHERE dept_name LIKE 'F%'  
)
```

	결과	메시지
	name	salary
1	Wu	90000
2	Singh	80000

	결과	메시지
	dept_name	
1	Finance	

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT name, salary  
FROM instructor  
WHERE dept_name NOT IN  
(  
    SELECT dept_name  
    FROM student  
    WHERE dept_name LIKE 'F%'  
)
```

	결과	메시지
	name	salary
1	Srinivasan	65000
2	Mozart	40000
3	Einstein	95000
4	El Said	60000
5	Gold	87000
6	Katz	75000
7	Clifien	62000
8	Crick	72000
9	Brandt	92000
10	Kim	80000

# 중첩 부질의 (Nested Subqueries)

## ▶ SOME & ALL

### ▶ 집합 비교

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT name, salary
FROM instructor
WHERE salary < SOME
(
    SELECT salary
    FROM instructor
    WHERE name LIKE '%i%'
)
```

	salary
1	65000
2	95000
3	60000
4	62000
5	80000
6	72000
7	80000

	name	salary
1	Srinivasan	65000
2	Wu	90000
3	Mozart	40000
4	El Said	60000
5	Gold	87000
6	Katz	75000
7	Clifieri	62000
8	Singh	80000
9	Crick	72000
10	Brandt	92000
11	Kim	80000

SQLQuery1.sql - ...IAC9HWUser (52))\*

```
SELECT name, salary
FROM instructor
WHERE salary < ALL
(
    SELECT salary
    FROM instructor
    WHERE name LIKE '%i%'
)
```

	name	salary
1	Mozart	40000

# 중첩 부질의 (Nested Subqueries)

## ▶ EXISTS & NOT EXISTS

### ▶ 빈 테이블 검사

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT *  
FROM instructor  
WHERE EXISTS  
(  
    SELECT dept_name  
    FROM student  
    WHERE tot_cred > 20  
)
```

	dept_name
1	Comp.Sci.
2	Comp.Sci.
3	History
4	Finance
5	Physics
6	Physics
7	Comp.Sci.
8	Music
9	Comp.Sci.
10	Elec.Eng.
11	Elec.Eng.

	id	name	dept_name	salary
1	10101	Srinivasan	Comp.Sci.	65000
2	12121	Wu	Finance	90000
3	15151	Mozart	Music	40000
4	22222	Einstein	Physics	95000
5	32343	El Said	History	60000
6	33456	Gold	Physics	87000
7	45565	Katz	Comp.Sci.	75000
8	58583	Clifieri	History	62000
9	76543	Singh	Finance	80000
10	76766	Crick	Biology	72000
11	83821	Brandt	Comp.Sci.	92000
12	98345	Kim	Elec.Eng.	80000

# 중첩 부질의 (Nested Subqueries)

## ▶ FROM 절의 부질의

```
SQLQuery1.sql - ...IAC9HWUser (52))*  
SELECT *  
FROM (  
    SELECT DISTINCT dept_name  
    FROM student  
    ) AS tmp  
WHERE tmp.dept_name LIKE '_i%'
```

결과		메시지	
	dept_name		
1	Comp.Sci.		
2	Elec.Eng.		
3	Finance		
4	History		
5	Music		
6	Physics		

결과		메시지	
	dept_name		
1	Finance		
2	History		