

# 관계모델과 기본 SQL

Database Laboratory

# 차례

---

- ▶ 데이터베이스 관리시스템
- ▶ 릴레이션 모델 (관계모델)
- ▶ SQL의 기본개념
  - ▶ DDL(Data Definition Language)
    - ▶ CREATE
    - ▶ DROP
    - ▶ ALTER
  - ▶ DCL(Data Control Language)
    - ▶ GRANT
    - ▶ REVOKE
  - ▶ DML(Data Manipulation Language)

# 데이터베이스 관리 시스템 (DBMS)

---

- ▶ **DBMS**는 특정 분야 관련 정보를 저장/관리한다
  - ▶ 상호 관련 있는 데이터의 모임
  - ▶ 데이터를 액세스하기 위한 프로그램의 집합
  - ▶ 사용하기 편하고 효율적인 환경을 제공
- ▶ **Database 응용분야:**
  - ▶ Banking: transactions
  - ▶ Airlines: reservations, schedules
  - ▶ Universities: registration, grades
  - ▶ Sales: customers, products, purchases
  - ▶ Online retailers: order tracking, customized recommendations
  - ▶ Manufacturing: production, inventory, orders, supply chain
  - ▶ Human resources: employee records, salaries, tax deductions
- ▶ 데이터베이스는 일반적으로 그 크기가 매우 크다고 가정한다.

# DB시스템의 목적

DBMS는 전통적인 OS 가 지원하는 파일 처리 시스템의 아래와 같은 문제점을 처리하기 위해 개발 되었음.

- ▶ 데이터의 중복과 불일치
- ▶ 데이터 액세스상의 어려운 점
- ▶ 데이터의 독립성 - 여러 파일과 포맷
- ▶ 무결성 문제
- ▶ 갱신의 원자성
- ▶ 여러 사용자에게 의한 동시 액세스
- ▶ 보안 문제



# 대학 데이터베이스예제

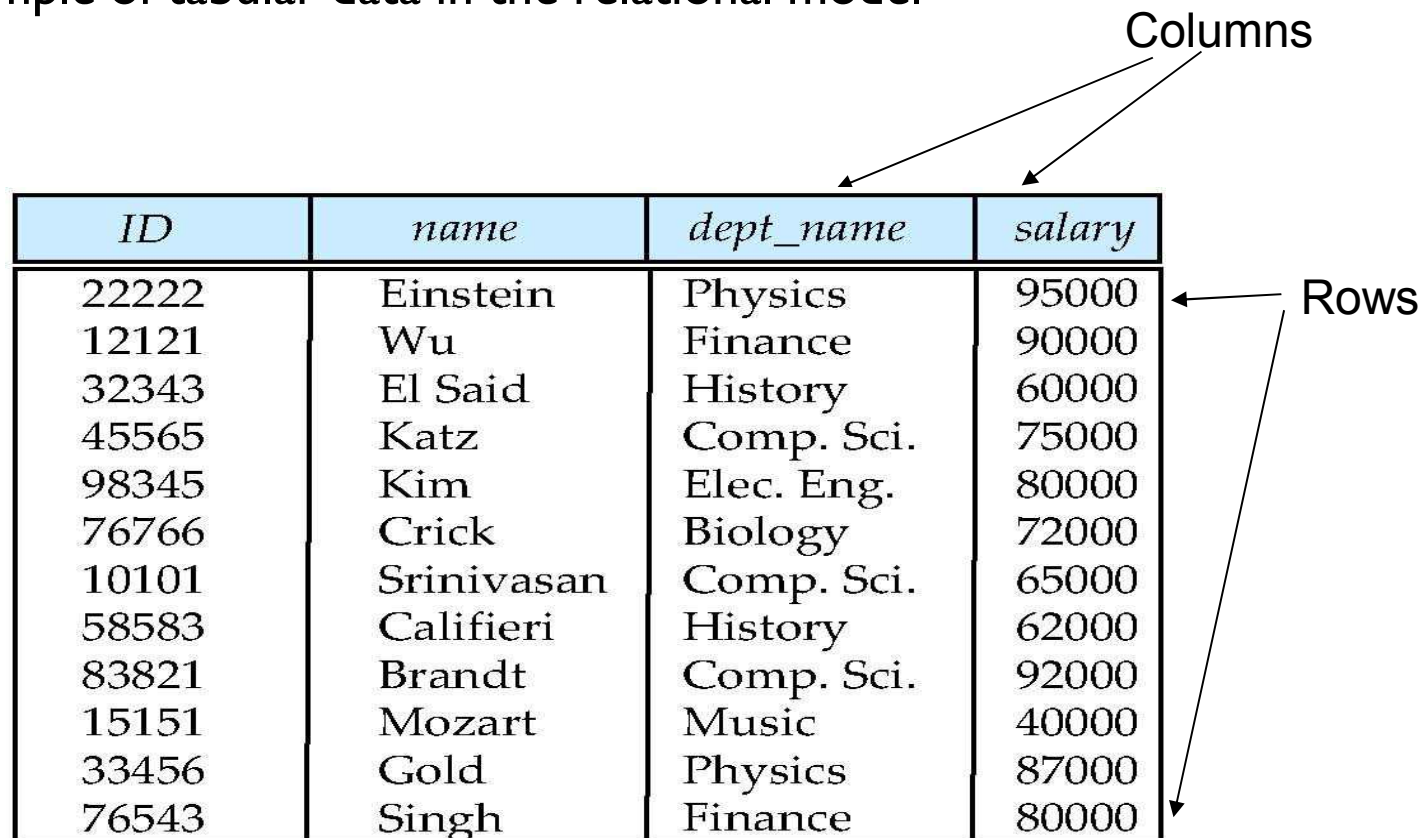
## ▶ 응용 예제

- ▶ 새로운 학생, 강사(instructor), 코스(course) 입력하기
- ▶ 각 코스에 학생 등록하고, 클래스 명단 생성하기
- ▶ 학생에게 학점(grade) 부여하고, 평점(grade point average, GPA) 계산하고, 성적 증명서(transcript) 생성하기



# 릴레이션 모델

- ▶ Example of tabular data in the relational model



The diagram illustrates a table with four columns and ten rows. Two arrows labeled 'Columns' point to the top row, specifically to the 'dept\_name' and 'salary' columns. Two arrows labeled 'Rows' point to the right side of the table, specifically to the first and last rows.

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

## A Sample Relational Database

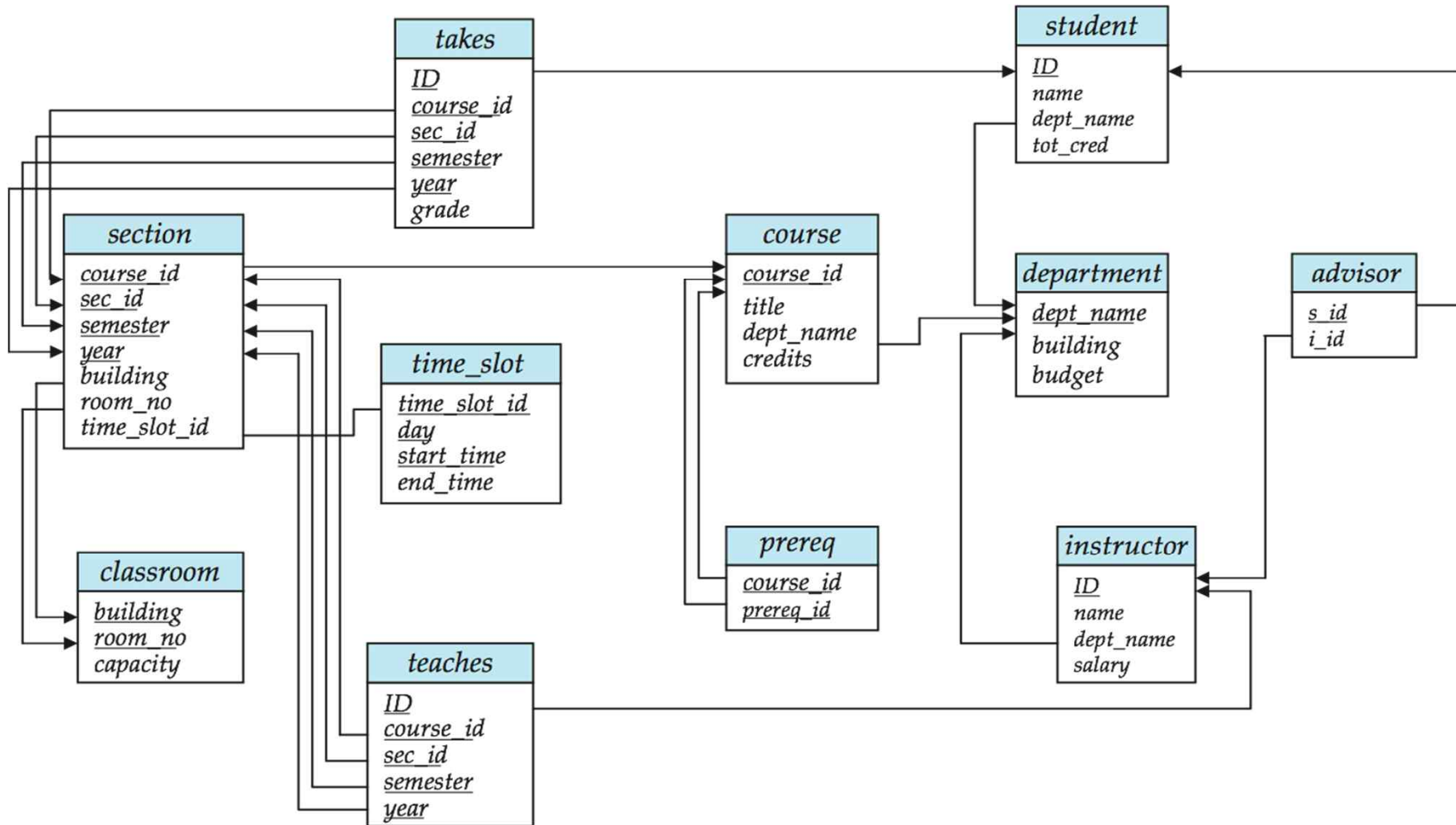
<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.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

(a) The *instructor* table

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

(b) The *department* table

# 대학 데이터베이스 설계





# SQL

---

## ▶ 기본개념

- ▶ SQL(Structure Query Language)은 구조화된 질의 언어라는 의미를 가지고 있는 일종의 프로그래밍 언어로써 관계형 데이터베이스를 제어하고 데이터를 추출하는데 이용
- ▶ C++, JAVA, etc.
  - ▶ 입출력과 화면인터페이스, 네트워크 기능들에 집중
- ▶ SQL은 ANSI-SQL이라는 국제적인 표준이 있고 Oracle, Sybase, Microsoft, mySQL과 같은 관계형 데이터베이스의 벤더들은 별도의 SQL 언어를 가지고 있다

# SQL

---

## ▶ SQL의 종류

- ▶ DDL (Data Definition Language)
  - ▶ 데이터와 그 구조를 정의
- ▶ DCL (Data Control Language)
  - ▶ 데이터베이스의 사용자 권한을 정의
- ▶ DML(Data Manipulation Language)
  - ▶ 데이터의 검색과 수정

## ▶ SQL 문을 이용하는 방법

- ▶ 쿼리 실행기를 통한 직접 실행
- ▶ ODBC나 ADO등의 방식을 이용하면 비주얼 베이직이나 ASP와 같은 기타 프로그래밍에서도 SQL문을 실행 시킬 수 있다.

# DDL (Data Definition Language)

---

- ▶ 역할

- ▶ 데이터 정의 언어로써 Table, Index, View, Stored Procedure 등과 같은 데이터베이스의 개체들을 생성, 수정, 삭제하는데 사용된다

- ▶ 종류

- ▶ CREATE
  - ▶ DROP
  - ▶ ALTER

# DDL (Data Definition Language)

---

## ▶ CREATE문

### ▶ CREATE TABLE문

- ▶ 데이터베이스의 정보를 검색하고 수정하기 이전에 해야 할 일이 이러한 정보를 저장하는 개체를 만드는 일이다

```
CREATE TABLE 테이블이름 (  
    열_이름 열의데이터타입,  
    열_이름 열의데이터타입 NULL or NOT NULL,  
    [PRIMARY KEY(열_이름 리스트),]  
    [UNIQUE(열_이름 리스트),]  
    [FOREIGN KEY(열_이름 리스트),  
        REFERENCE 기본테이블 [( )]]  
    ... );
```

# DDL (Data Definition Language)

## ▶ CREATE 문 (테이블 생성/데이터 입력)

```
use [20115190] --test 데이터베이스를 사용하겠다는 쿼리--]

/*student 테이블 생성 및 테이블 구조*/
CREATE TABLE student(
    ID int NOT NULL PRIMARY KEY,
    NAME VARCHAR(10) NOT NULL,
    dept_name VARCHAR(20) NOT NULL,
    tot_cred INT NULL,
    UNIQUE (NAME)
)

--student 테이블에 데이터 추사
INSERT INTO student
VALUES (00128, 'zhang', 'Comp.sci', 102);

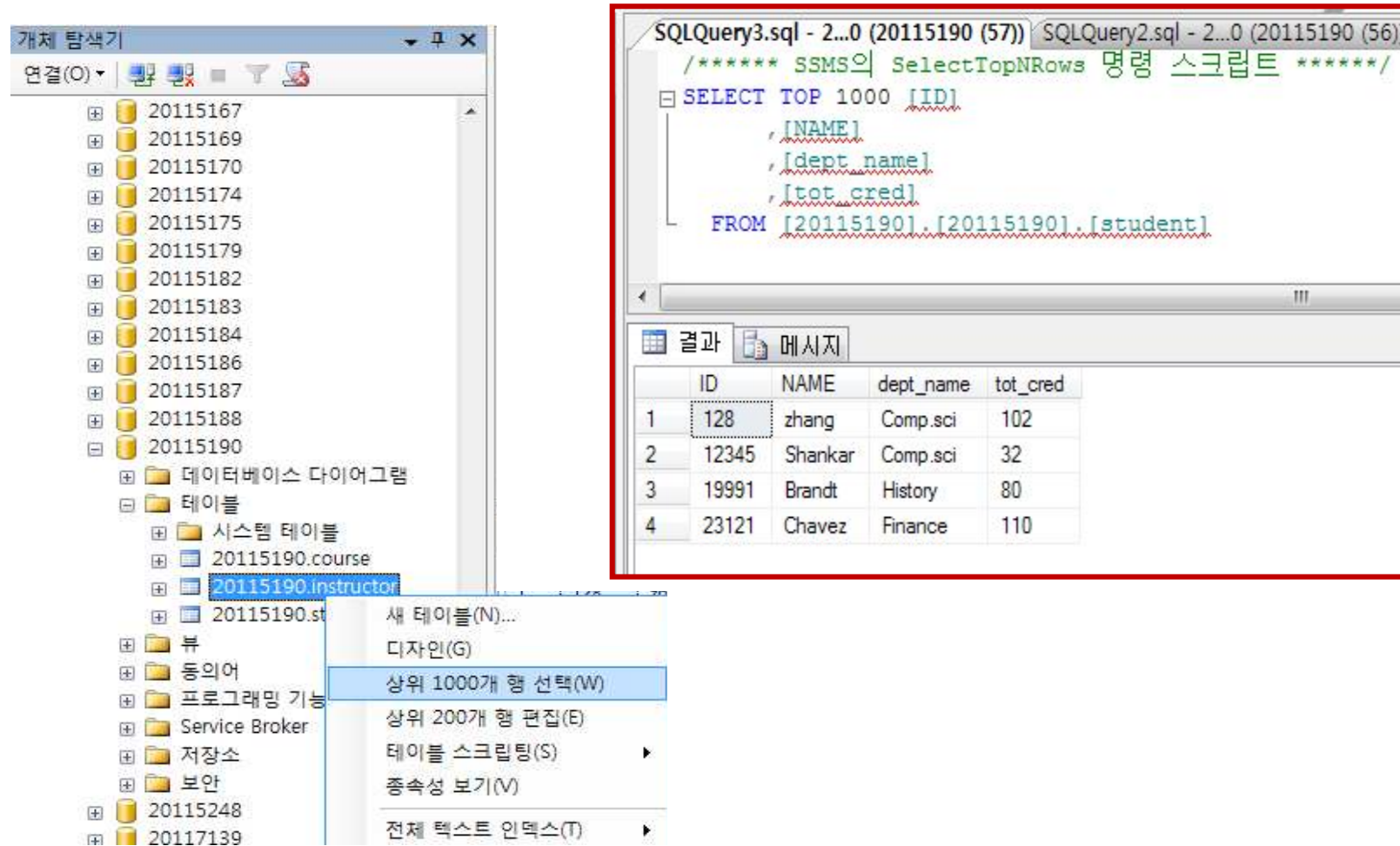
INSERT INTO student
VALUES (12345, 'Shankar', 'Comp.sci', 32);

INSERT INTO student
VALUES (19991, 'Brandt', 'History', 80);

INSERT INTO student
VALUES (23121, 'Chavez', 'Finance', 110);
```

# DDL (Data Definition Language)

## ▶ CREATE 문 (결과 보기)



The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Database Explorer' pane displays a tree view of the database structure. The '20115190' database is selected, and the '20115190.instructor' table is highlighted. A context menu is open over the '20115190.instructor' table, showing options such as '새 테이블(N)...', '디자인(G)', '상위 1000개 행 선택(W)', '상위 200개 행 편집(E)', '테이블 스크립팅(S)', '종속성 보기(V)', and '전체 텍스트 인덱스(T)'. The '상위 1000개 행 선택(W)' option is selected.

On the right, the 'SQLQuery3.sql - 2...0 (20115190 (57))' window displays the following SQL query:

```
SELECT TOP 1000 [ID]
, [NAME]
, [dept_name]
, [tot_cred]
FROM [20115190].[20115190].[student]
```

Below the query, the '결과' (Results) tab shows the query results in a table format:

ID	NAME	dept_name	tot_cred	
1	128	zhang	Comp.sci	102
2	12345	Shankar	Comp.sci	32
3	19991	Brandt	History	80
4	23121	Chavez	Finance	110

# DDL (Data Definition Language)

- ▶ ALTER 문 – column 추가
  - ▶ 생성된 개체를 수정할 때 사용되는 DDL문

The screenshot shows a SQL IDE with three tabs: 'SQLQuery4.sql - ... (20115190 (58))\*', 'SQLQuery3.sql - 2...0 (20115190 (57))', and 'SQLQuery2.sql - 2...0 (20115190 (57))'. The active tab 'SQLQuery4.sql' contains the following SQL code:

```
ALTER TABLE student
ADD e_mail varchar(200) NULL -- student 테이블에 e_mail칼럼 추가

ALTER TABLE student
ADD Homepage varchar(200) NULL -- student 테이블에 Homepage칼럼 추가

select * from student
```

Below the code editor, the '결과' (Results) tab is active, displaying a table with 7 columns: ID, NAME, dept\_name, tot\_cred, e\_mail, and Homepage. The table contains 4 rows of data. A red rectangle highlights the 'e\_mail' and 'Homepage' columns for all rows, showing they are all NULL.

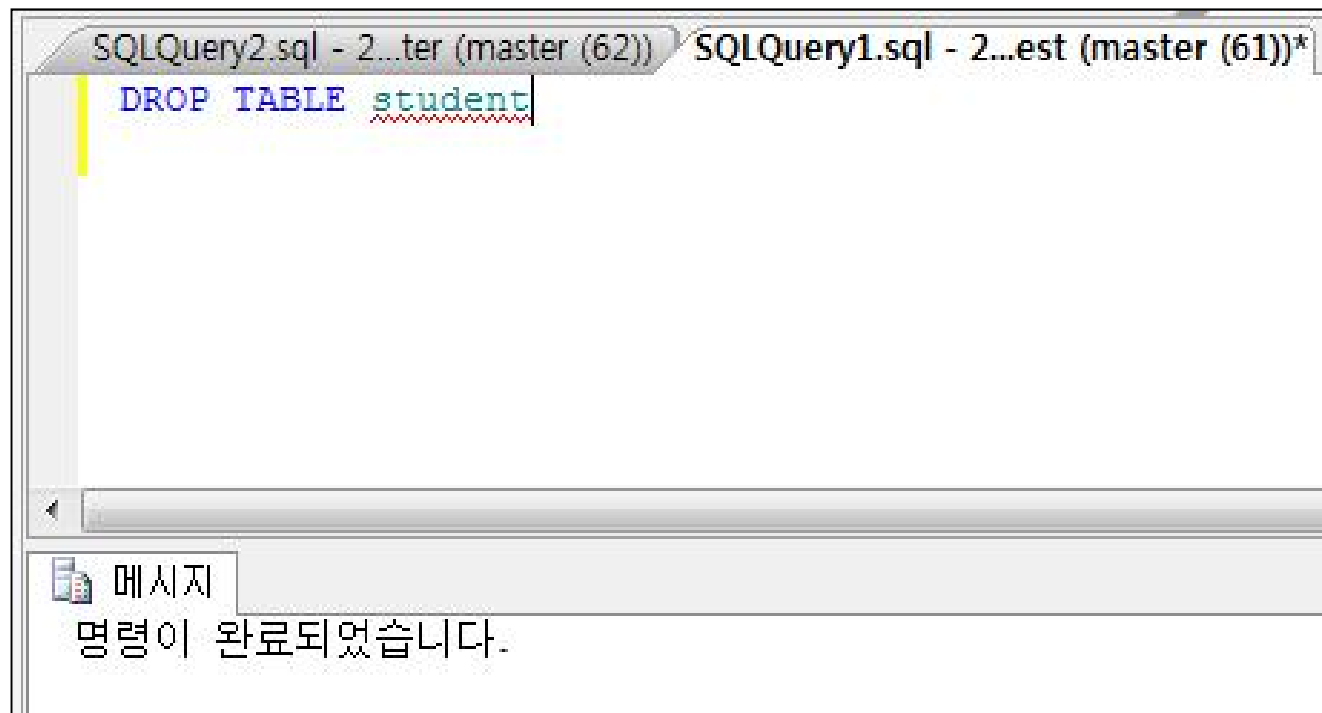
	ID	NAME	dept_name	tot_cred	e_mail	Homepage
1	128	zhang	Comp.sci	102	NULL	NULL
2	12345	Shankar	Comp.sci	32	NULL	NULL
3	19991	Brandt	History	80	NULL	NULL
4	23121	Chavez	Finance	110	NULL	NULL

# DDL (Data Definition Language)

---

## ▶ DROP 문

- ▶ 생성된 테이블을 삭제하는데 사용





# DCL (Data Control Language)

## ▶ GRANT문

- ▶ 테이블이나 뷰 등에 대해 접근하는 권한을 설정하는 대표적인 DCL문 이다
- ▶ 'Product\_Manager'라는 사용자에게 Company 테이블 접근 권한을 주는 DCL문

```
CREATE LOGIN Product_Manager WITH PASSWORD = 'password'  
USE test  
sp_ADDUSER 'Product_Manager'
```

메시지  
명령이 완료되었습니다.

```
SQLQuery2.sql - 2...inistrator (52))*  
USE test  
GRANT INSERT, UPDATE, DELETE  
ON Company  
TO Product_Manager
```

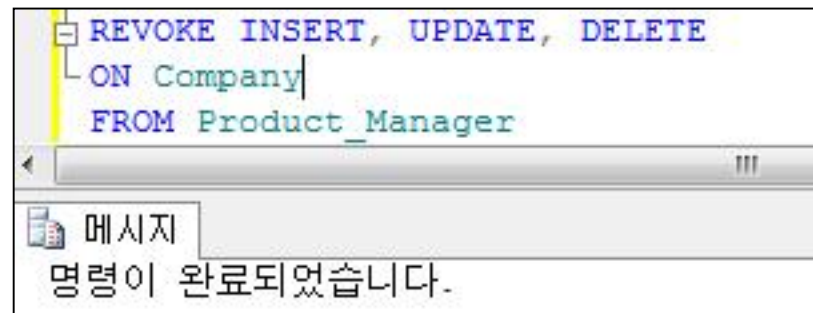
메시지  
명령이 완료되었습니다.

# DCL (Data Control Language)

---

## ▶ REVOKE문

- ▶ GRANT문이 Product\_Manager에게 권한을 설정한다면 REVOKE 문은 설정했던 권한을 삭제한다



# DML(Data Manipulation Language)

---

- ▶ select문
  - ▶ 데이터 전체 검색
  - ▶ 특정 컬럼 검색
- ▶ where문
  - ▶ 조건문 검색
  - ▶ 비교 / 논리 / 산술 연산자
- ▶ SELECT ~ INTO
- ▶ 중복된 행 제거하기: DISTINCT
- ▶ 수치연산 함수
- ▶ 문자열 함수 : + , LEFT, LEN, LOWER, etc.
- ▶ 집합 함수 : AVG, SUM, COUNT, MAX/MIN, GROUP BY, HAVING
- ▶ Insert문
- ▶ update문

# DML(Data Manipulation Language)

---

## ▶ SELECT 문

- ▶ 테이블이나 뷰에서 데이터를 보여주는데 사용된다
- ▶ 테이블에 추가된 데이터를 검색할 때 사용한다
- ▶ keyword : SELECT, FROM, WHERE 등

```
SELECT select_list           /* column 제한 */  
[INTO new_table]  
FROM table_source  
[WHERE search_condition]    /* row 제한 */  
[GROUP BY group_by_expression]  
[HAVING search_condition]  
[ORDER BY order_expression [ASC | DESC] ]
```

# DML(Data Manipulation Language)

## ▶ SELECT 문 예제

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery7.sql - ... (20115190 (61))\* Sc

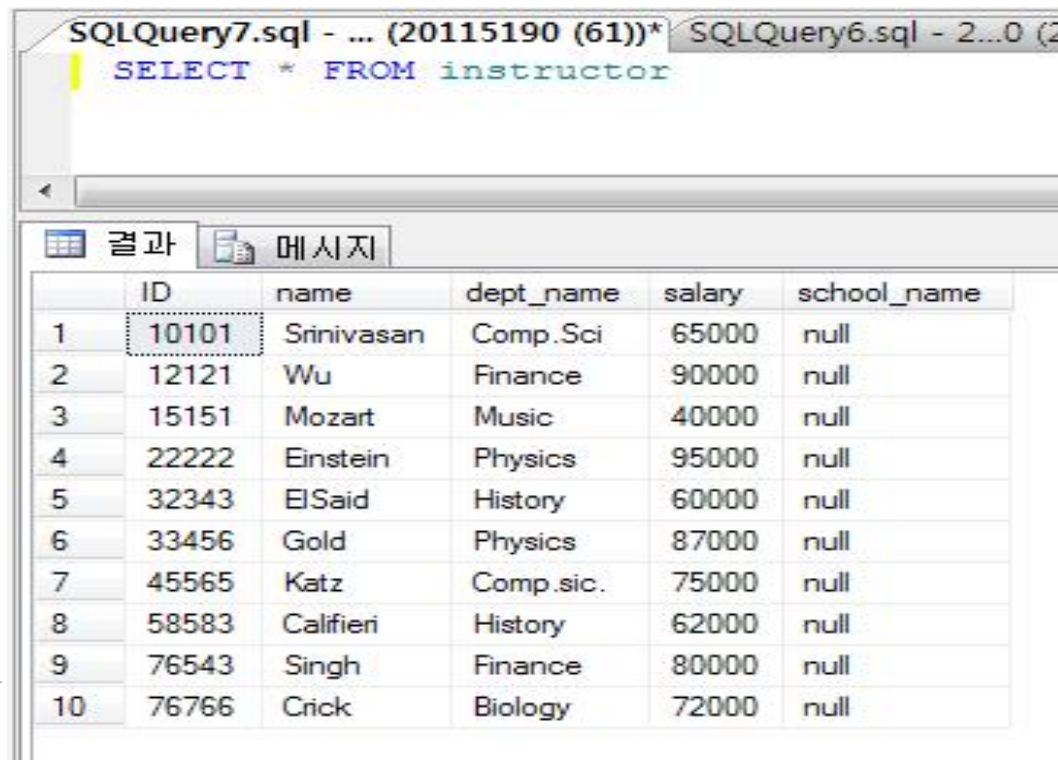
```
SELECT ID,salary
FROM instructor
WHERE salary>=72000
ORDER BY(salary) ASC
```

	ID	salary
1	76766	72000
2	45565	75000
3	76543	80000
4	33456	87000
5	12121	90000
6	22222	95000

# DML(Data Manipulation Language)

## ▶ 데이터 전체 검색

- ▶ instructor 테이블에서 모든 column 데이터를 출력하고자 할 때 ‘\*’ 이용
  - ▶ WHERE 조건이 있으면 row 들을 제한
  - ▶ ‘\*’는 실제 쿼리가 실행 될 때 column list 로 변경된다



The screenshot shows a SQL query window with the following content:

```
SQLQuery7.sql - ... (20115190 (61))* SQLQuery6.sql - 2...0 (2)
SELECT * FROM instructor
```

Below the query window, there is a tabbed interface with '결과' (Results) and '메시지' (Messages) tabs. The '결과' tab is active, displaying a table with the following data:

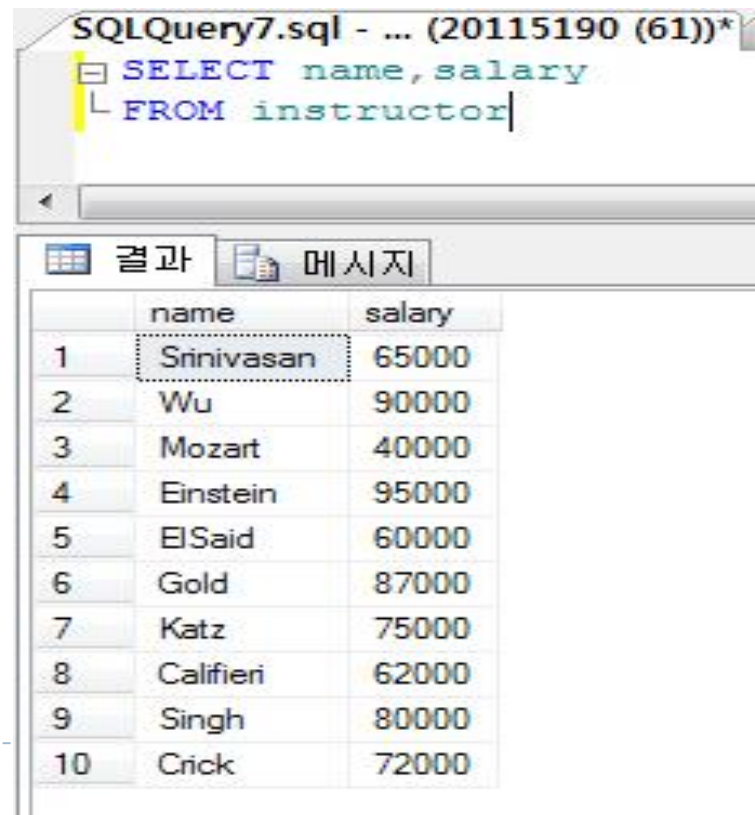
	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.sic.	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

# DML(Data Manipulation Language)

---

## ▶ 특정 컬럼 검색

- ▶ SELECT문 다음에 특정 column을 명시하면 명시된 column만 출력
- ▶ 여러 개의 column을 출력하기 위해서는 쉼표(,)로 column을 연결



The screenshot shows a SQL query window titled 'SQLQuery7.sql - ... (20115190 (61))\*'. The query text is 'SELECT name, salary FROM instructor'. Below the query, there are two tabs: '결과' (Results) and '메시지' (Messages). The '결과' tab is active, displaying a table with 10 rows of data. The first row is highlighted with a dashed border.

	name	salary
1	Srinivasan	65000
2	Wu	90000
3	Mozart	40000
4	Einstein	95000
5	ElSaid	60000
6	Gold	87000
7	Katz	75000
8	Califieri	62000
9	Singh	80000
10	Crick	72000

# DML(Data Manipulation Language)

- ▶ column 명 또는 table에 별칭주기

SQLQuery8.sql - 2...0 (20115190 (63)) | SQLQuery7.sql - ... (20115190 (61))\* | SQLQu

```
SELECT * FROM instructor AS 교사
```

```
SELECT name AS 성명, ID AS '인증 번호' FROM instructor
```

결과 메시지

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.sic	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

성명 인증 번호

1	Srinivasan	10101
2	Wu	12121
3	Mozart	15151
4	Einstein	22222
5	ElSaid	32343
6	Gold	33456
7	Katz	45565
8	Califieri	58583
9	Singh	76543
10	Crick	76766

SQLQuery7.sql - ... (20115190 (61))\* | SQLQuery6.sql - 2...0 (20115190 (60)) | SQLQuery5.sql - 2...0 (20115190 (60))

```
SELECT 교사.name AS 성명, 교사.ID AS '인증 번호' FROM instructor AS 교사
```

결과 메시지

	성명	인증 번호
1	Srinivasan	10101
2	Wu	12121
3	Mozart	15151
4	Einstein	22222
5	ElSaid	32343
6	Gold	33456
7	Katz	45565
8	Califieri	58583
9	Singh	76543
10	Crick	76766



# DML(Data Manipulation Language)

---

## ▶ 조건 문 검색

- ▶ WHERE절 이하에 검색 조건을 명시하면 전체 테이블에서 대상이 되는 Row를 제한해서 원하는 Row만 가져온다

## ▶ 검색 조건

- ▶ 컬럼내의 내용을 수식(비교연산자,산술연산자,논리연산자)을 이용하여 비교

# DML(Data Manipulation Language)

---

## ▶ 비교 연산자

- ▶ 두 표현식을 비교하여 연산결과가 참(true)인 Row만 쿼리의 적용을 받도록 제한한다

비교연산	설명
=	A 와 B가 같다
>	A 가 B보다 크다
<	A 가 B보다 작다
>=	A 가 B보다 크거나 같다
<=	A 가 B보다 작거나 같다
<> or !=	A 와 B가 같지 않다

# DML(Data Manipulation Language)

## ▶ 비교 연산자 예

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery7.sql - ... (20115190 (61))\* SQLQ

```
SELECT name  
FROM instructor  
WHERE dept_name = 'physics'
```

결과 메시지

	name
1	Einstein
2	Gold

SQLQuery7.sql - ... (20115190 (61))

```
SELECT name  
FROM instructor  
WHERE salary > 87000
```

결과 메시지

	name
1	Wu
2	Einstein

# DML(Data Manipulation Language)

## ▶ 산술 연산자

- ▶ +(더하기), -(빼기), \*(곱하기), /(나누기), %(나머지)

### Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery7.sql - ... (20115190 (61))*		SQLQuery5.sql - 2...0 (20115	
<pre>SELECT salary * 0.9 as 'salary without tax' FROM instructor WHERE name = 'Wu'</pre>			
결과 메시지			
salary without tax			
1	81000.0		

# DML(Data Manipulation Language)

## ▶ 논리 연산자

- ▶ 논리 연산은 조건식이 여러 개인 경우 이러한 조건식을 AND, OR, NOT과 같은 논리 연산으로 묶어서 조건식을 비교하는 것

연산자	의미
ALL	모든 비교집합이 TRUE 인 경우 true
AND	두 개의 부울 식이 모두 TRUE인 경우 true
ANY	비교 집합 중 어느 하나가 TRUE 인 경우 true
BETWEEN	비교 연산자 범위 안에 있는 경우 true
EXISTS	하위 쿼리에 행이 포함된 경우 true
IN	피연산자가 식 목록 중 하나와 동일한 경우 true
LIKE	피연산자가 패턴과 일치하는 경우
NOT	연산자의 값을 반대로
OR	한 개의 부울 식이 TRUE인 경우 true
SOME	비교 집합 중 일부가 TRUE인 경우 true

# DML(Data Manipulation Language)

## ▶ 논리 연산자의 예

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

## ▶ AND

```
SQLQuery7.sql - ... (20115190 (61))* SQLQuery5.sql - 2...0 (20115190 (59))
```

```
SELECT name  
FROM instructor  
WHERE dept_name = 'physics' AND salary > 87000
```

결과	메시지
name	
1	Einstein

## ▶ OR

```
SQLQuery7.sql - ... (20115190 (61))* SQLQuery5.sql - 2...0 (20115190 (59))
```

```
SELECT name  
FROM instructor  
WHERE dept_name = 'physics' OR dept_name = 'Music'
```

결과	메시지
name	
1	Mozart
2	Einstein
3	Gold

# DML(Data Manipulation Language)

## ▶ 범위 검색

- ▶ 어떤 범위의 데이터들을 검색하기 위해서는 비교 연산자를 AND 나 OR 로 연결시켜서 검색할 수 있다
- ▶ 간단한 표현법 **BETWEEN**

The image shows two side-by-side SQL Developer query windows. The left window displays the following SQL query:

```
SELECT name
FROM instructor
WHERE salary >= 65000 AND salary <= 72000
```

The right window displays the same query but using the BETWEEN operator:

```
SELECT name
FROM instructor
WHERE salary BETWEEN 65000 AND 72000
```

A red arrow points from the left window to the right window, indicating a transformation or comparison. Below the query text in both windows, there is a '결과' (Results) tab showing the output of the query. The results are as follows:

	name
1	Srinivasan
2	Crick

# DML(Data Manipulation Language)

## ▶ Like 연산자

- ▶ 어떤 특정한 자료를 찾을 때 그 이름을 정확히 기억을 한다는 것은 어려운 일 중에 하나이다

와일드 카드	의미	실 예
%	복수개의 문자열	LIKE 'C%' : C로 시작하는 데이터 검색 LIKE '%i' : n로 끝나는 데이터 검색
_	단일 문자	LIKE '_omp.Sci' : omp.Sci로 끝나고 전체길이가 8자리 문자열
[]	[]안에 있는 한 개의 문자	LIKE '[A-D]omp.Sci' : M에서 Z 사이의 한 문자로 시작하고 ton으로 끝나는 단어 검색
[^]	[]안에 있는 문자 제외	LIKE 'C[^e]%' : C로 시작하고 두 번째 문자가 e 가 아닌 모든 단어 검색



# DML(Data Manipulation Language)

## ▶ 중복된 row 제거하기 : DISTINCT

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery11.sql - ... (20115190 (62))\*

```
SELECT DISTINCT dept_name  
FROM instructor
```

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

# DML(Data Manipulation Language)

## ▶ 테이블 복사하기 : SELECT ~ INTO

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

```
SQLQuery11.sql - ... (20115190 (62))* DBLAB-...
SELECT * INTO instructor_copy
FROM instructor
WHERE salary <=60000|
```

메시지

(2개 행이 영향을 받음)

결과

```
SQLQuery11.sql - ... (20115190 (62))* DBLAB-WINSERVER...51
```

```
SELECT *
FROM instructor_copy
```

	ID	name	dept_name	salary	school_name
1	15151	Mozart	Music	40000	null
2	32343	ElSaid	History	60000	null

# DML(Data Manipulation Language)

## ▶ 문자열 함수

+ (두개 이상의 문자열을 연결하는 연산자)

LEFT(왼쪽에서부터 지정된 문자 수에서 시작하는 문자열의 일부를 반환)

LEN(문자열에서 실제 문자열의 개수 반환)

LOWER(문자열을 소문자로 변환) , etc.

## Course 테이블

	course_id	title	dept_name	credits
1	BIO-101	Intro. to Biology	Biology	4
2	BIO-199	Computational Biology	Biology	3
3	BIO-301	Genetics	Biology	4
4	CS-101	Intro. to Computer Science	Comp.Sci	4
5	CS-190	Game Design	Comp.Sci	4
6	CS-315	Robotics	Comp.Sci	3
7	CS-319	Image Processing	Comp.Sci	3
8	CS-347	Database System Concepts	Comp.Sci	3
9	EE-181	Intro. to Digital System	Ekec.Eng	3
10	FIN-201	Investment Banking	Finance	3
11	HIS-251	World History	History	3
12	MU-199	Music Video Production	Music	3
13	PHY-101	Physical Principles	Physics	4

SQLQuery17.sql - ...0 (20115190 (55)) SQLQuery11.s	
SELECT title + ',' + dept_name	
FROM course	
결과 메시지	
(열 이름 없음)	
1	Intro. to Biology,Biology
2	Computational Biology,Biology
3	Genetics,Biology
4	Intro. to Computer Science,Comp.Sci
5	Game Design,Comp.Sci
6	Robotics,Comp.Sci
7	Image Processing,Comp.Sci
8	Database System Concepts,Comp.Sci
9	Intro. to Digital System,Ekec.Eng
10	Investment Banking,Finance
11	World History,History
12	Music Video Production,Music
13	Physical Principles,Physics

# DML(Data Manipulation Language)

---

- ▶ 집합 함수(Aggregate Function)
  - ▶ AVG
  - ▶ SUM
  - ▶ COUNT
  - ▶ MAX/MIN
  - ▶ GROUP BY 와 HAVING
  - ▶ HAVING

# DML(Data Manipulation Language)

## ▶ AVG

- ▶ 입력한 표현식에서 NULL 값을 무시하고 해당 모든 column의 평균값을 구한다
- ▶ 인수
  - ▶ ALL : 모든 값에 집계함수를 적용한다
  - ▶ DISTINCT : 값의 발생횟수에 상관없이 UNIQUE 한 값들만 AVG를 수행

## Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery17.sql - ...0 (20115190 (55)) S	
SELECT AVG(salary) FROM instructor	
결과 메시지	
(열 이름 없음)	
1	72600

# DML(Data Manipulation Language)

## ▶ SUM

- ▶ 모든 값의 총합을 구하며 ALL, DISTINCT의 인수 사용

### Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

SQLQuery17.sql -...0 (20115190 (55))	
<pre>SELECT SUM(salary) FROM instructor</pre>	
결과	메시지
(열 이름 없음)	
1	726000



# DML(Data Manipulation Language)

## ▶ COUNT

- ▶ 검색된 결과의 전체 카운트 수를 알아내는 함수이다
- ▶ 리스트의 전체 개수; 예) 게시판에 올라온 총 목록 수

Course 테이블

	course_id	title	dept_name	credits
1	BIO-101	Intro. to Biology	Biology	4
2	BIO-199	Computational Biology	Biology	3
3	BIO-301	Genetics	Biology	4
4	CS-101	Intro. to Computer Science	Comp.Sci	4
5	CS-190	Game Design	Comp.Sci	4
6	CS-315	Robotics	Comp.Sci	3
7	CS-319	Image Processing	Comp.Sci	3
8	CS-347	Database System Concepts	Comp.Sci	3
9	EE-181	Intro. to Digital System	Elec.Eng	3
10	FIN-201	Investment Banking	Finance	3
11	HIS-251	World History	History	3
12	MU-199	Music Video Production	Music	3
13	PHY-101	Physical Principles	Physics	4

SQLQuery17.sql - ...0 (20115190 (55)) SQL	
SELECT COUNT (*) FROM course WHERE course_id LIKE 'B%'	
결과	메시지
(열 이름 없음)	
1	3

# DML(Data Manipulation Language)

- ▶ MAX / MIN
  - ▶ 특정 열의 최대값 / 최소값

Instructor 테이블

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	58583	Califieri	History	62000	null
9	76543	Singh	Finance	80000	null
10	76766	Crick	Biology	72000	null

```
SQLQuery17.sql - ...0 (20115190 (55))
SELECT MAX(salary)
FROM instructor
```

결과 메시지

(열 이름 없음)

1 95000

```
SQLQuery17.sql - ...0 (20115190 (55))
SELECT MIN(salary)
FROM instructor
```

결과 메시지

(열 이름 없음)

1 40000



# DML(Data Manipulation Language)

## ▶ GROUP BY 와 HAVING

- ▶ GROUP BY : 한 특정열의 값들을 UNIQUE 한 값에 따라 그룹을 짓는 연산자
- ▶ HAVING : GROUP BY 절에 의해 만들어진 소그룹에 대한 조건이 적용됨

### Course 테이블

	course_id	title	dept_name	credits
1	BIO-101	Intro. to Biology	Biology	4
2	BIO-199	Computational Biology	Biology	3
3	BIO-301	Genetics	Biology	4
4	CS-101	Intro. to Computer Science	Comp.Sci	4
5	CS-190	Game Design	Comp.Sci	4
6	CS-315	Robotics	Comp.Sci	3
7	CS-319	Image Processing	Comp.Sci	3
8	CS-347	Database System Concepts	Comp.Sci	3
9	EE-181	Intro. to Digital System	Elec.Eng	3
10	FIN-201	Investment Banking	Finance	3
11	HIS-251	World History	History	3
12	MU-199	Music Video Production	Music	3
13	PHY-101	Physical Principles	Physics	4

SQLQuery17.sql - ...0 (20115190 (55))

```
SELECT course_id  
FROM course  
GROUP BY course_id
```

결과 메시지

	course_id
1	BIO-101
2	BIO-199
3	BIO-301
4	CS-101
5	CS-190
6	CS-315
7	CS-319
8	CS-347
9	EE-181
10	FIN-201
11	HIS-251
12	MU-199
13	PHY-101

SQLQuery17.sql - ...0 (20115190 (55)) SQL

```
SELECT course_id  
FROM course  
GROUP BY course_id  
HAVING course_id LIKE 'CS%'
```

결과 메시지

	course_id
1	CS-101
2	CS-190
3	CS-315
4	CS-319
5	CS-347

# DML(Data Manipulation Language)

## ▶ INSERT 문

- ▶ 테이블에 어떠한 특정한 row 를 추가할 때 사용한다

The screenshot displays a SQL IDE interface with two main panes. The top pane shows a SQL script with three INSERT statements into the 'instructor' table. The bottom-left pane shows the execution messages, indicating that each row was successfully inserted. The bottom-right pane shows the result of a SELECT query, displaying a table with 13 rows of instructor data.

```
SQLQuery18.sql - ...0 (20115190 (57)) DBLAB-WINSEVE...0115190.course SQLQuery17.sql - ...0 (20115190 (55)) SQLQuery11.sql - ... (20115190 (62))*
```

```
INSERT INTO instructor (ID, name, dept_name, salary, school_name) VALUES ('51844', 'Sam', 'Comp.Sci', '90000', 'null')
INSERT INTO instructor VALUES ('55190', 'John', 'Music', '50000', 'null')
INSERT INTO instructor VALUES ('53484', 'Jane', 'Finance', '67000', 'null')
```

메시지

(1개 행이 영향을 받음)  
(1개 행이 영향을 받음)  
(1개 행이 영향을 받음)

```
SQLQuery18.sql - ...0 (20115190 (57)) DBLAB-WINSEVE...0
```

```
SELECT * FROM instructor
```

결과 메시지

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	51844	Sam	Comp.Sci	90000	null
9	53484	Jane	Finance	67000	null
10	55190	John	Music	50000	null
11	58583	Califieri	History	62000	null
12	76543	Singh	Finance	80000	null
13	76766	Crick	Biology	72000	null

# DML(Data Manipulation Language)

## ▶ UPDATE 문

- ▶ 테이블이나 뷰의 특정 row 에서 데이터를 업데이트 할 때 사용

SQLQuery18.sql - ...0 (20115190 (57)) DBLAB-WINSERVE...0

```
SELECT * FROM instructor
```

결과 메시지

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	51844	Sam	Comp.Sci	90000	null
9	53484	Jane	Finance	67000	null
10	55190	John	Music	50000	null
11	58583	Califieri	History	62000	null
12	76543	Singh	Finance	80000	null
13	76766	Crick	Biology	72000	null



SQLQuery19.sql - ... (20115190 (58))\* DBLAB-WINSERVE...

```
UPDATE instructor  
SET name = 'Wang'  
WHERE name = 'John'  
  
SELECT * FROM instructor
```

결과 메시지

	ID	name	dept_name	salary	school_name
1	10101	Srinivasan	Comp.Sci	65000	null
2	12121	Wu	Finance	90000	null
3	15151	Mozart	Music	40000	null
4	22222	Einstein	Physics	95000	null
5	32343	ElSaid	History	60000	null
6	33456	Gold	Physics	87000	null
7	45565	Katz	Comp.Sci	75000	null
8	51844	Sam	Comp.Sci	90000	null
9	53484	Jane	Finance	67000	null
10	55190	Wang	Music	50000	null
11	58583	Califieri	History	62000	null
12	76543	Singh	Finance	80000	null
13	76766	Crick	Biology	72000	null