TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

NGÔN NGỮ ĐỊNH NGHĨA VÀ THAO TÁC DỮ LIỆU

GV: Đỗ Bá Lâm
Email: lamdb@soict.hust.edu.vn
Viện Công nghệ thông tin và Truyền thông,
Trường Đại học Bách Khoa Hà Nội

Nội dung

Các ngôn ngữ
dữ liệu

Đại số quan hệ

Structured
Query Language

Nội dung

Các ngôn ngữ dữ liệu

Đại số quan hệ

Structured Query Language

Ví dụ

- Tìm tên của các sinh viên nào sống ở Bundoora
  - Tìm các bộ của bảng Student có Suburb = Bundoora
  - Đưa ra các giá trị của thuộc tính Name của các bộ này

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Ví dụ (2)

- Tìm các sinh viên đăng ký khoá học có mã số 113
  - Tìm các giá trị SID trong bảng Enrol có trường course tương ứng là 113
  - Đưa các bộ của bảng Student có Id trong các giá trị tìm thấy ở trên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

SID	Course
3936	101
1108	113
8507	101

Course

No	Name	Dept
113	BCS	CSCE
101	MCS	CSCE

Phân loại ngôn ngữ truy vấn

- Ngôn ngữ cấp thấp
  - Đại số quan hệ
    - Một câu hỏi = một tập các phép toán trên các quan hệ
    - Ví dụ: $(\sigma_{suburb="Bundoora"}(Student))$
  - Tính toán vị từ
    - Một câu hỏi = một tập mô tả của các bộ mong muốn
    - Ví dụ: {s | s ∈ Student ^ s[Suburb] = ‘‘Bundoora’’}
- Ngôn ngữ cấp cao
  - QBE: câu truy vấn đơn giản, được thiết lập thông qua giao diện đồ họa
  - SQL: ngôn ngữ cho phép định nghĩa, thao tác, và quản lý dữ liệu

Nội dung

Các ngôn ngữ
dữ liệu

Đại số quan hệ

Structured
Query Language

Tổng quan

- Gồm các phép toán tương ứng với các thao tác trên các quan hệ
- Mỗi phép toán
  - Đầu vào: một hay nhiều quan hệ
  - Đầu ra: một quan hệ
- Biểu thức đại số quan hệ = chuỗi các phép toán
- Kết quả thực hiện một biểu thức đại số là một quan hệ
- Được cài đặt trong phần lớn các hệ CSDL hiện nay

Phân loại các phép toán

- Phép toán quan hệ
  - Phép chiếu (*projection*)
  - Phép chọn (*selection*)
  - Phép kết nối (*join*)
  - Phép chia (*division*)
- Phép toán tập hợp
  - Phép hợp (*union*)
  - Phép giao (*intersection*)
  - Phép trừ (*difference*)
  - Phép tích đề-các (*cartesian product*)

Phép chiếu

- Đ/n: Lựa chọn một số thuộc tính từ một quan hệ
- Cú pháp: $\prod A1, A2, \ldots (R)$

- Ví dụ: đưa ra danh sách tên của tất cả các sinh viên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

$\prod name(Student)$

Kết quả

Name
Robert
Glen
Norman
Mary

Phép chọn

- Đ/n: Lựa chọn các bộ trong một quan hệ thoả mãn điều kiện cho trước
- Cú pháp: $\sigma_{<condition>}(R)$

- Ví dụ: đưa ra danh sách những sinh viên sống ở Bundoora

$\sigma_{suburb="Bundoora"}(Student)$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Kết quả

Id	Name	Suburb
3936	Glen	Bundoora
8507	Norman	Bundoora

Ví dụ - chọn và chiếu

- Ví dụ: đưa ra tên những sinh viên sống ở Bundoora

$$\prod name(\sigma_{suburb="Bundoora"}(Student))$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

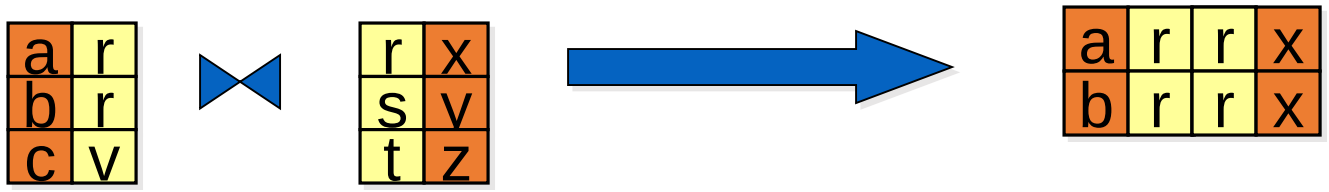
Kết quả

Name
Glen
Norman

Phép kết nối

- Đ/n: ghép các bộ từ 2 quan hệ thoả mãn điều kiện kết nối
- Cú pháp:

$$R_1 \triangleright\!\triangleleft_{<join\_condition>} R_2$$



- Ví dụ: đưa ra danh sách các sinh viên và khoá học

$$Student \triangleright\!\triangleleft_{Id=SID} Enrol$$

Student ⋈ Id=SID Enrol

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

SID	Course
3936	101
1108	113
8507	101

Kết quả

SID	Id	Name	Suburb	Course
1108	1108	Robert	Kew	113
3936	3936	Glen	Bundoora	101
8507	8507	Norman	Bundoora	101

Ví dụ - chọn, chiếu và kết nối

- Đưa ra ***tên*** của các sinh viên sống ở Bundoora và ***mã khoá học*** mà sinh viên đó đăng ký

$$\prod Name, Course(\sigma_{Suburb="Bundoora"}(Student \rhd\lhd_{Id=SID} Enrol))$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

SID	Course
3936	101
1108	113
8507	101

Kết quả

Name	Course
Glen	101
Norman	101

Phép kết nối tự nhiên

- Đ/n: là phép kết nối với điều kiện bằng trên các thuộc tính trùng tên
- Ví dụ:

Takes

SID	SNO
1108	21
1108	23
8507	23
8507	29

*

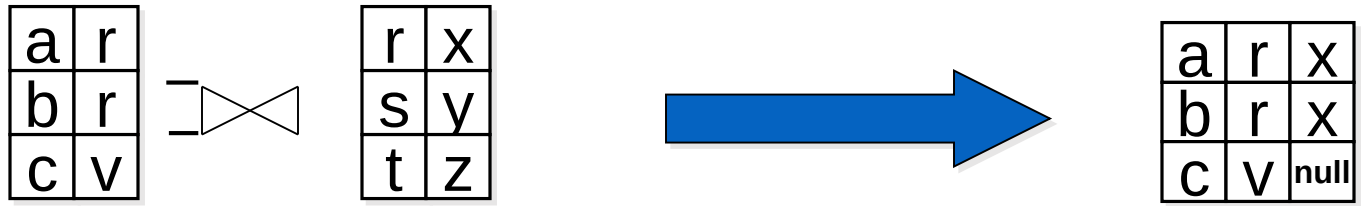
Enrol

SID	Course
3936	101
1108	113
8507	101

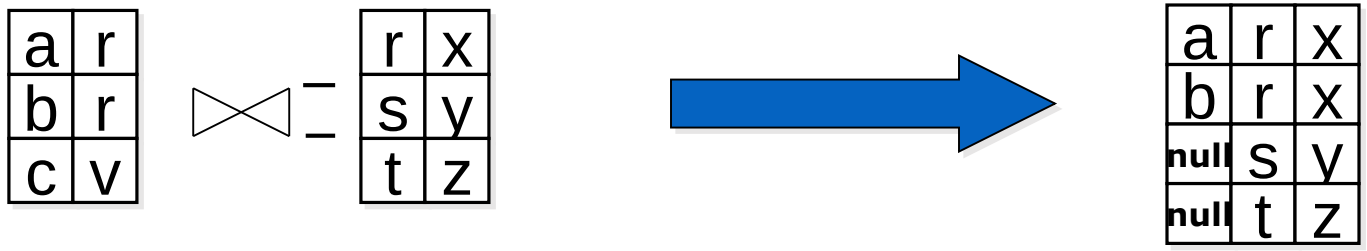
SID	SNO	Course
1108	21	113
1108	23	113
8507	23	101
8507	29	101

Phép kết nối ngoài

- Phép kết nối ngoài trái



- Phép kết nối ngoài phải



Ví dụ về phép kết nối ngoài

- Đưa ra danh sách mã số các sinh viên và mã khoá học mà sinh viên đó đăng ký nếu có

Student

ID	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Id = SID

Enrol

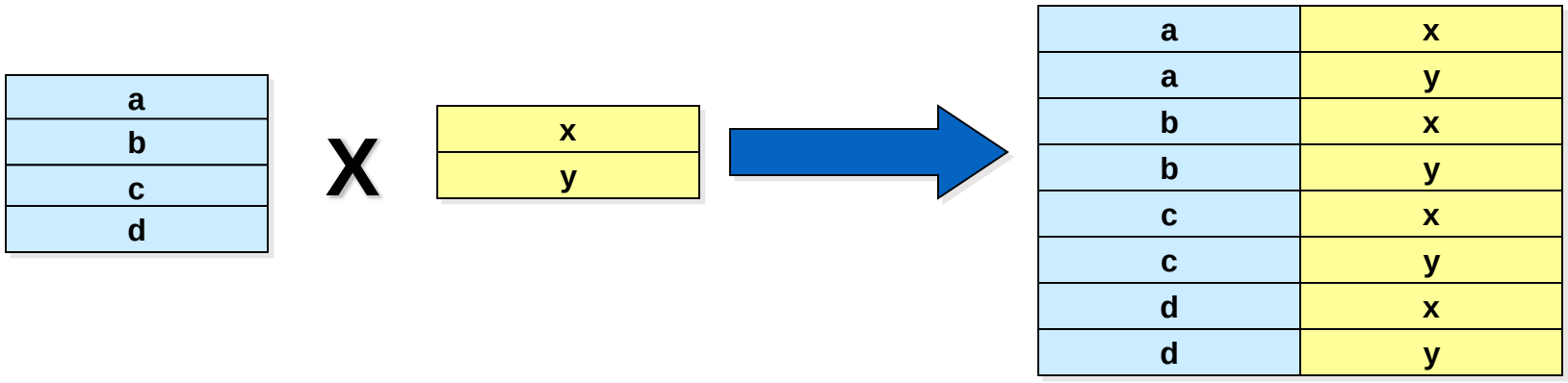
SID	Course
3936	101
1108	113
8507	101

Kết quả

ID	Name	Suburb	SID	Course
1108	Robert	Kew	1108	113
3936	Glen	Bundoora	3936	101
8507	Norman	Bundoora	8507	101
8452	Mary	Balwyn	null	**null**

Phép tích đề-các

- Đ/n: là kết nối giữa từng bộ của quan hệ thứ nhất và mỗi bộ của quan hệ thứ hai
- Cú pháp: R_1 x R_2



Ví dụ phép tích đề-các

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

X

Sport

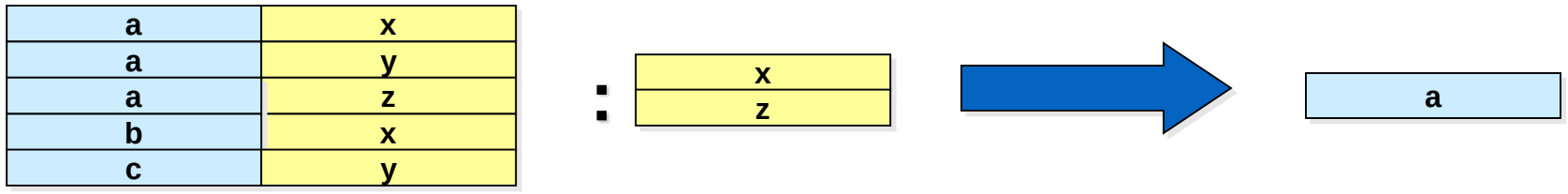
SportID	Sport
05	Swimming
09	Dancing

Student_Sport

Id	Name	Suburb	SportID	Sport
1108	Robert	Kew	05	Swimming
3936	Glen	Bundoora	05	Swimming
8507	Norman	Bundoora	05	Swimming
8452	Mary	Balwyn	05	Swimming
1108	Robert	Kew	09	Dancing
3936	Glen	Bundoora	09	Dancing
8507	Norman	Bundoora	09	Dancing
8452	Mary	Balwyn	09	Dancing

Phép chia

- Đ/n: cho R_1 và R_2 lần lượt là các quan hệ n và m ngôi. Kết quả của phép chia R_1 cho R_2 là một quan hệ (n-m) ngôi
- Cú pháp: $R_1 : R_2$



Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

:

Course

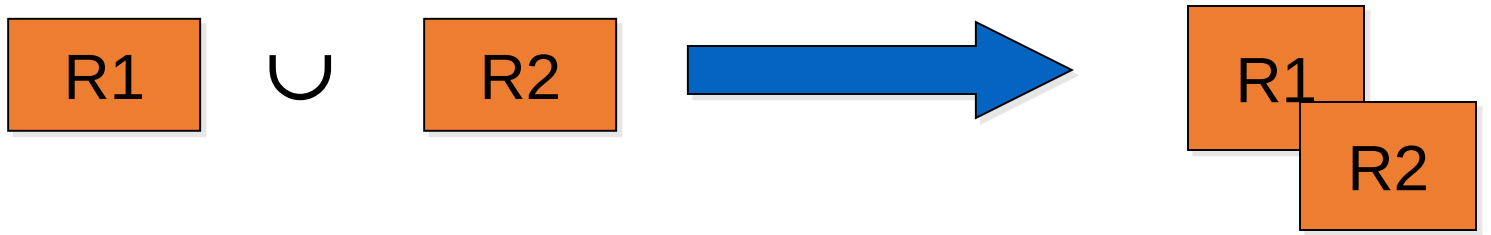
Course
BCS
MCS

Kết quả

Name
Database

Phép hợp

- Đ/n: gồm các bộ thuộc ít nhất một trong hai quan hệ đầu vào. Hai quan hệ khả hợp được nếu có cùng số lượng thuộc tính, và thuộc tính xác định trên cùng miền giá trị
- Cú pháp: $R_1 \cup R_2$



Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∪

Subject2

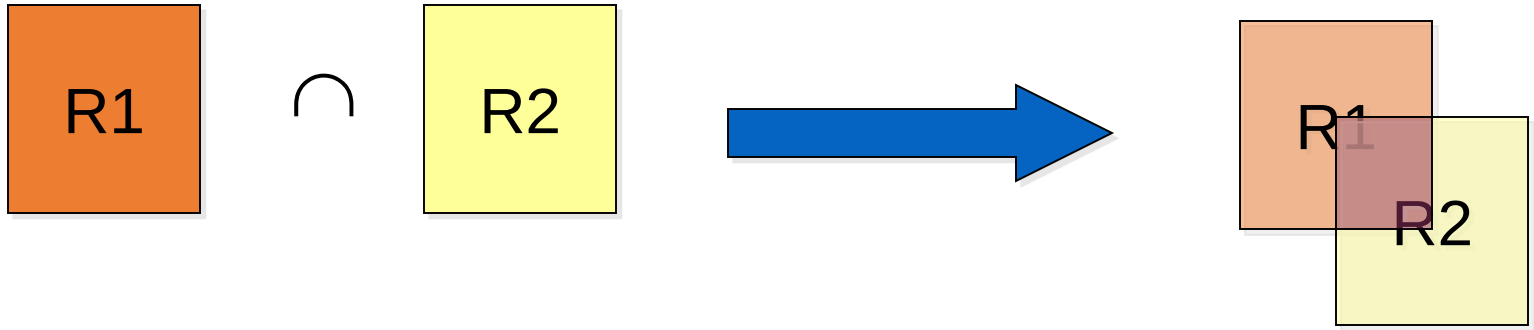
Name	Course
DataMining	MCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS
DataMining	MCS
Writing	BCS

Phép giao

- Đ/n: gồm các bộ thuộc cả hai quan hệ đầu vào
- Cú pháp: $R_1 \cap R_2$



Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∩

Subject2

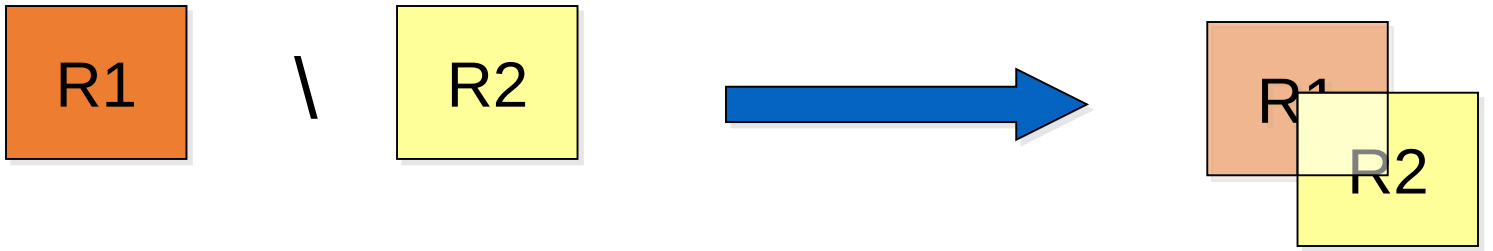
Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	MCS

Phép trừ

- Đ/n: gồm các bộ thuộc quan hệ thứ nhất nhưng không thuộc quan hệ thứ hai. Hai quan hệ phải là khả hợp
- Cú pháp: $R_1 \setminus R_2$



Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

\

Subject2

Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Database	BCS
Algebra	MCS

Ví dụ

- Phép giao và phép trừ

r (A B C)

a_1	b_1	c_1
a_1	b_1	c_2
a_1	b_2	c_2
a_2	b_2	c_2
a_3	b_2	c_2

s (A B C)

a_1	b_1	c_1
a_1	b_2	c_1
a_1	b_2	c_2

$r \cap s = g$ (A B C)

a_1	b_1	c_1
a_1	b_2	c_2

$r - s = t$ (A B C)

a_1	b_1	c_2
a_2	b_2	c_2
a_3	b_2	c_2

Ví dụ Phép chiếu

$$\prod_X (r) = \{ t[X] \mid t \in r \}$$

$X = \{ A, B \}$; $Y = \{ C \}$

r	(A	B	C)
	a_1	b_1	c_1
	a_1	b_1	c_2
	a_1	b_2	c_2
	a_2	b_2	c_2
	a_3	b_2	c_2

$\Pi_X(r) = s_1$ (A B)

a_1	b_1
a_1	b_2
a_2	b_2
a_3	b_2

$\Pi_Y(r) = s_2$ (C)

c_1
c_2

Ví dụ Phép chọn

$$\sigma_F(r) = \{ t \mid t \in r \wedge F(t) = \text{đúng} \}$$

r	(A	B	C)
	a_1	b_1	c_1
	a_1	b_1	c_2
	a_1	b_2	c_2
	a_2	b_2	c_2
	a_3	b_2	c_2

$\sigma_{A=a1}(r) = r_1$

(A	B	C)
a_1	b_1	c_1
a_1	b_2	c_2
a_1	b_1	c_2

$\sigma_{A=a1 \wedge C=c2}(r) = r_2$

(A	B	C)
a_1	b_1	c_2
a_1	b_2	c_2

Bài tập 1

- Cho CSDL gồm 3 quan hệ sau: S (Các hãng cung ứng), P(các mặt hàng), SP(số lượng cung ứng)

S (S# SNAME STATUS CITY)

S#	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Black	30	Paris

SP (S# P# QTY)

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S2	P1	300
S2	P2	400
S3	P2	200

P (P# PNAME COLOR WEIGHT CITY)

P#	PNAME	COLOR	WEIGHT	CITY
P1	Nut	red	12	London
P2	Bolt	green	17	Paris
P3	Screw	blue	17	Rom
P4	Screw	red	14	London

Bài tập 1 – Yêu cầu

- Biểu diễn câu hỏi truy vấn bằng ngôn ngữ đại số quan hệ
  1. Đưa ra danh sách các mặt hàng màu đỏ
  2. Cho biết S# của các hãng cung ứng mặt hàng 'P1' hoặc 'P2'
  3. Liệt kê S# của các hãng cung ứng cả hai mặt hàng 'P1' và 'P2'
  4. Đưa ra S# của các hãng cung ứng ít nhất một mặt hàng màu đỏ
  5. Đưa ra S# của các hãng cung ứng tất cả các mặt hàng.
- Tính kết quả của các câu truy vấn

Bài tập 1 – Đáp án

- Đưa ra danh sách các mặt hàng màu đỏ:

$$\sigma_{COLOR = 'red'}(P)$$

- Cho biết S# của các hãng cung ứng mặt hàng 'P1' hoặc 'P2':

$$\Pi_{S\#}(\sigma_{P\# = 'P1' \vee P\# = 'P2'}(SP))$$

- Liệt kê S# của các hãng cung ứng cả hai mặt hàng 'P1' và 'P2':

$$\Pi_{S\#}(\sigma_{P\# = 'P1'}(SP)) \cap \Pi_{S\#}(\sigma_{P\# = 'P2'}(SP))$$

- Đưa ra S# của các hãng cung ứng ít nhất một mặt hàng màu đỏ:

$$\Pi_{S\#}(SP * \sigma_{COLOR = 'red'}(P))$$

- Đưa ra S# của các hãng cung ứng tất cả các mặt hàng:

$$\Pi_{S\#,P\#}(SP) \div \Pi_{P\#}(P)$$

Nội dung

Các ngôn ngữ
dữ liệu

Đại số quan hệ

Structured
Query Language

SQL (*Structured Query Language*)

- 1975: SEQUEL
  - System-R
- 1976: SEQUEL2
- 1978/79: SQL
  - System-R
- 1986: chuẩn SQL-86
- 1989: chuẩn SQL-89
- 1992: chuẩn SQL-92 (SQL2)
- 1996: chuẩn SQL-96
- 1999: chuẩn SQL-99 (SQL3)
- Newers: 2003, 2006, 2008, 2011, 2016.

Các thành phần của SQL

- Ngôn ngữ định nghĩa dữ liệu (***D**ata **D**efinition **L**anguage*)
  - Cấu trúc các bảng CSDL
  - Các mối liên hệ của dữ liệu
  - Quy tắc, ràng buộc áp đặt lên dữ liệu
- Ngôn ngữ thao tác dữ liệu (***D**ata **M**anipulation **L**anguage*)
  - Thêm, xoá, sửa và truy vấn dữ liệu trong CSDL
- Ngôn ngữ quản lý dữ liệu (***D**ata **C**ontrol **L**anguage*)
  - Thay đổi cấu trúc của các bảng dữ liệu
  - Khai báo bảo mật thông tin
  - Quyền hạn của người dùng trong khai thác CSDL

Nội dung

- Phần 1. SQL Language – Basic
- Phần 2. SQL Language – Advanced

Reference: https://www.tutorialspoint.com/sql/index.htm

Phần 1. SQL LANGUAGE - BASIC

- Datatypes
- Create, Drop, Select Database
- Create, Insert Table
- Where
- And & Or
- Update
- Delete
- Like
- Top
- Order by
- Group by
- Distinct

Các kiểu dữ liệu

- **char(n).** Xâu kí tự độ dài cố định, với tối đa n kí tự. Thêm dấu cách vào bên phải để đủ số kí tự.
- **varchar(n).** Xâu kí tự độ dài thay đổi, với tối đa n kí tự
- **int,** Số nguyên
- **smallint.** Số nguyên
- **numeric(p,d).** Số dấu phẩy động, với độ chính xác xác định bởi p kí tự trong đó có d kí tự bên phải dấu thập phân. Ví dụ numeric(3,1) cho phép lưu trữ chính xác 44.5 nhưng không phải 444.5 hay 0.32
- **real, double precision.** Số thực
- **float(n).** Số dấu phẩy động, với số chữ số tối đa bên phải dấu thập phân là n
- **date.** Ngày tháng năm
- **time.** Giờ phút giây

Kiểu dữ liệu

Kiểu dữ liệu	Từ	Tới
int	-2,147,483,648	2,147,483,647
smallint	-32,768	32,767
numeric	-10^38 +1	10^38 -1
real	-3.40E + 38	3.40E + 38
float	-1.79E + 308	1.79E + 308
date	Lưu ngày như June 30, 1991	
time	Lưu thời gian như 12:30 P.M	

CHAR và VARCHAR

- **char(n).**
  - Xâu kí tự độ dài cố định, xác định bởi người dùng
  - Các kí tự cách được thêm vào bên phải (nếu cần) để đạt đủ độ dài
- **varchar(n).**
  - Xâu kí tự độ dài thay đổi, với tối đa n kí tự
  - Có thêm 1 hoặc 2 byte tiền tố xác định số kí tự

Value	CHAR(4)	Storage Required	VARCHAR(4)	Storage Required
''	' '	4 bytes	''	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes

CREATE DATABASE

- **CREATE DATABASE:** tạo mới một CSDL
- Cú pháp

```
CREATE DATABASE DatabaseName;
```

- Lưu ý: tên CSDL phân biệt nhau
- Ví dụ

```
SQL> CREATE DATABASE testDB;
SQL> SHOW DATABASES;
```

DROP DATABASE

- **DROP DATABASE:** xóa một CSDL đã có
- Cú pháp

```
DROP DATABASE DatabaseName;
```

- Lưu ý: tên CSDL phân biệt nhau
- Ví dụ:

```
SQL> DROP DATABASE testDB;
SQL> SHOW DATABASES;
```

USE DATABASE

- **USE:** lựa chọn một CSDL đã có để thao tác
- Cú pháp:

```
USE DatabaseName;
```

Note: database name should be unique within the RDBMS.

- Example:

```
SQL> SHOW DATABASES;
SQL> USE mysql;
```

Create Table

- **CREATE TABLE:** tạo một bảng mới.
- Cú pháp

```
CREATE TABLE table_name(
   column1 datatype,
   column2 datatype,
    .....
   columnN datatype,
   PRIMARY KEY( one or more columns )
);
```

- Ví dụ:

```
SQL> CREATE TABLE CUSTOMERS(
   ID          INT                NOT NULL,
   NAME        VARCHAR (20)       NOT NULL,
   AGE         INT                NOT NULL,
   ADDRESS     CHAR (25) ,
   SALARY      DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

Desc Table

```
SQL> DESC CUSTOMERS;
+---------+---------------+------+-----+---------+-------+
| Field   | Type          | Null | Key | Default | Extra |
+---------+---------------+------+-----+---------+-------+
| ID      | int(11)       | NO   | PRI |         |       |
| NAME    | varchar(20)   | NO   |     |         |       |
| AGE     | int(11)       | NO   |     |         |       |
| ADDRESS | char(25)      | YES  |     | NULL    |       |
| SALARY  | decimal(18,2) | YES  |     | NULL    |       |
+---------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Drop Table

- **DROP TABLE:** xóa bảng dữ liệu và các dữ liệu của nó.
- Cú pháp

```
DROP TABLE table_name;
```

- Ví dụ:

```
SQL> DROP TABLE CUSTOMERS;
Query OK, 0 rows affected (0.01 sec)

SQL> DESC CUSTOMERS;
ERROR 1146 (42S02): Table 'TEST.CUSTOMERS' doesn't exist
```

Insert Query

- **INSERT INTO:** thêm các dòng/hàng vào trong bảng
- Cú pháp:

```
INSERT INTO TABLE_NAME (column1, column2, column3,...columnN)
VALUES (value1, value2, value3,...valueN);
```

- Ví dụ:

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (1, 'Ramesh', 32, 'Ahmedabad', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)
VALUES (2, 'Khilan', 25, 'Delhi', 1500.00 );
```

Bảng mẫu

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Select Query

- **SELECT**: truy vấn dữ liệu trong CSDL, và trả về kết quả dưới dạng bảng
- Cú pháp

```
SELECT column1, column2, columnN FROM table_name;
```

```
SELECT ID, NAME, SALARY FROM CUSTOMERS;
```

ID	NAME	SALARY
1	Ramesh	2000.00
2	Khilan	1500.00
3	kaushik	2000.00
4	Chaitali	6500.00
5	Hardik	8500.00
6	Komal	4500.00
7	Muffy	10000.00

```
SELECT * FROM CUSTOMERS;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Where Clause

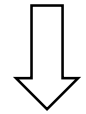
- **WHERE:** cho phép xác định điều kiện trong truy vấn dữ liệu. Điều kiện trên một bảng hoặc nhiều bảng.
- Cú pháp

```
SELECT column1,.., columnN
FROM table_name
WHERE [condition]
```

- Toán tử: >, <, =, LIKE, NOT,….
- Ví dụ

```
SQL> SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE SALARY > 2000;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	SALARY
4	Chaitali	6500.00
5	Hardik	8500.00
6	Komal	4500.00
7	Muffy	10000.00

AND and OR

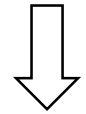
- **AND** & **OR**: cho phép kết hợp nhiều điều kiện
- Ví dụ cú pháp

```
SELECT column1, ,columnN
FROM table_name
WHERE [condition1] AND...AND
  [conditionN];
```

- Ví dụ

```
SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE SALARY > 2000 AND age < 25;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	SALARY
6	Komal	4500.00
7	Muffy	10000.00

Update Query

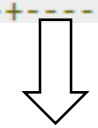
- **UPDATE:** sửa đổi, cập nhật dữ liệu đã có trong bảng
- Cú pháp

```
UPDATE table_name
SET column1 = value1,...,
    columnN = valueN
WHERE [condition];
```

- Ví dụ:

```
UPDATE CUSTOMERS
SET ADDRESS = 'Pune'
WHERE ID = 6;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	Pune	4500.00
7	Muffy	24	Indore	10000.00

Delete Query

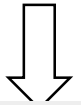
- **DELETE :** xóa bản ghi đã có trong bảng
- Cú pháp

```
DELETE FROM table_name
WHERE [condition];
```

- Ví dụ

```
DELETE FROM CUSTOMERS
WHERE ID = 6;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
7	Muffy	24	Indore	10000.00

Like Clause

- **LIKE** : cho phép so sánh các giá trị tương đồng, sử dụng các kí tự đại diện
  - %: không, một, hoặc nhiều kí tự
  - _: một kí tự
- Mẫu cú pháp

```
SELECT column FROM table_name
WHERE column LIKE '_XXXX%'
```

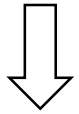
- Ví dụ

```
SELECT * FROM CUSTOMERS
WHERE SALARY LIKE '200%'
```

- Điều kiện khác

```
WHERE SALARY LIKE '%200%'
WHERE SALARY LIKE '%2'
WHERE SALARY LIKE '_00%'
WHERE SALARY LIKE '2_%_%'
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
3	kaushik	23	Kota	2000.00

Top Clause

- **TOP:** chỉ hiển thị TOP N hoặc X % bản ghi từ kết quả truy vấn
- Cú pháp

```
SELECT TOP number|percent
column_name(s)
FROM table_name
WHERE [condition]
```

- SQL Server

```
SELECT TOP 3 * FROM CUSTOMERS;
```

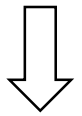
- MySQL

```
SELECT * FROM CUSTOMERS LIMIT 3;
```

- Oracle

```
SELECT * FROM CUSTOMERS
WHERE ROWNUM <= 3;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00

Order By

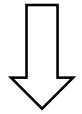
- **ORDER BY:** sắp xếp các bản ghi theo thứ tự tang dần hoặc giảm dần, dựa trên một hoặc nhiều cột. Thứ tự mặc định là tăng dần.
- Cú pháp

```
SELECT column-list
FROM table_name
[WHERE condition]
[ORDER BY column1, column2, .. columnN] [ASC | DESC];
```

- Ví dụ:

```
SELECT * FROM CUSTOMERS
ORDER BY NAME DESC;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
7	Muffy	24	Indore	10000.00
6	Komal	22	MP	4500.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
5	Hardik	27	Bhopal	8500.00
4	Chaitali	25	Mumbai	6500.00

Group By

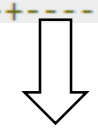
- **GROUP BY** được sử dụng để tổ chức/sắp xếp kết quả thành các nhóm.
- Cú pháp

```
SELECT column1, column2
FROM table_name
WHERE [ conditions ]
GROUP BY column1, column2
```

- Ví dụ:

```
SELECT NAME, SUM(SALARY) FROM
CUSTOMERS
GROUP BY NAME;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



NAME	SUM(SALARY)
Chaitali	6500.00
Hardik	8500.00
kaushik	2000.00
Khilan	1500.00
Komal	4500.00
Muffy	10000.00
Ramesh	2000.00

Distinct

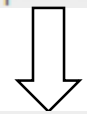
- **DISTINCT:** cho phép loại bỏ các kết quả trùng lặp
- Syntax

```
SELECT DISTINCT column1,...columnN
FROM table_name
WHERE [condition]
```

- Example:

```
SELECT DISTINCT SALARY
FROM CUSTOMERS
ORDER BY SALARY;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



SALARY
1500.00
2000.00
4500.00
6500.00
8500.00
10000.00

Phần 2. SQL LANGUAGE - ADVANCED

- NULL
- ALTER TABLE
- CONSTRAINTS
- JOIN
- UNION, MINUS, INTERSECT
- ALIAS
- Sub Queries
- Functions

NULL

- **NULL:** giá trị NULL trong bảng đề cập tới giá trị bị bỏ trống, không có giá trị nào.
- Cú pháp

```
CREATE TABLE CUSTOMERS(
   ID          INT              NOT NULL,
   NAME        VARCHAR (20)     NOT NULL,
   AGE         INT              NOT NULL,
   ADDRESS     CHAR (25) ,
   SALARY      DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

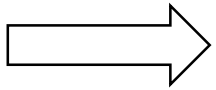
NULL – Ví dụ

- Example

```
SELECT  ID, NAME, AGE, ADDRESS, SALARY
FROM CUSTOMERS
WHERE SALARY IS NOT NULL;
```

- Try with: `WHERE SALARY IS NULL;`

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	
7	Muffy	24	Indore	



ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00

Alter Table

- **ALTER TABLE:** cho phép thêm, sửa, xóa các cột trong bảng đã tạo ra. Ngoài ra lệnh này cũng cho phép thêm hoặc xóa các ràng buộc.
- Cú pháp

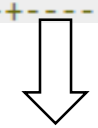
```
ALTER TABLE table_name
ADD column_name datatype;
```

- Ví dụ:

```
ALTER TABLE CUSTOMERS
ADD SEX char(1);

ALTER TABLE CUSTOMERS
DROP SEX;
```

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



ID	NAME	AGE	ADDRESS	SALARY	SEX
1	Ramesh	32	Ahmedabad	2000.00	NULL
2	Ramesh	25	Delhi	1500.00	NULL
3	kaushik	23	Kota	2000.00	NULL
4	kaushik	25	Mumbai	6500.00	NULL
5	Hardik	27	Bhopal	8500.00	NULL
6	Komal	22	MP	4500.00	NULL
7	Muffy	24	Indore	10000.00	NULL

Ràng buộc - Constraint

- Ràng buộc là các luật, quy định áp dụng lên các cột trong bảng. Chúng được sử dụng để giới hạn các kiểu dữ liệu, miền giá trị của dữ liệu.
  1. *NOT NULL Constraint – Đảm bảo ràng một cột không có giá trị NULL*
  2. *DEFAULT Constraint – Cung cấp giá trị mặc định cho một cột*
  3. *UNIQUE Constraint – Đảm bảo ràng tất cả các giá trị trong một cột là khác nhau*
  4. *PRIMARY Key – Khóa chính trong một bảng*
  5. *FOREIGN Key – Khóa ngoài tham chiếu tới bảng khác*
  6. *CHECK Constraint – Kiểm tra giá trị trong bảng có thỏa mãn điều kiện hay không.*
  7. *INDEX – Tạo chỉ mục cho bảng*
- Ví dụ:

```
ALTER TABLE CUSTOMERS DROP PRIMARY KEY ;
ALTER TABLE CUSTOMERS ADD PRIMARY KEY (ID);
```

Khóa ngoại/khóa ngoài (Foreign Key)

- Khóa ngoại: là khóa được sử dụng đề liên kết hai bảng với nhau
- Ví dụ:

```
CREATE TABLE CUSTOMERS(
   ID   INT              NOT NULL,
   NAME VARCHAR (20)     NOT NULL,
   AGE  INT              NOT NULL,
   ADDRESS  CHAR (25) ,
   SALARY   DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

```
CREATE TABLE ORDERS (
   ID     INT          NOT NULL,
   DATE  DATETIME,
   CUSTOMER_ID INT,
   AMOUNT double,
   PRIMARY KEY (ID)
);
```

Tạo Foreign key

- Tạo khóa ngoại trong định nghĩa bảng

```
CREATE TABLE Orders (
  OrderID int NOT NULL,
  CustomerID int,
  PRIMARY KEY (OrderID),
  CONSTRAINT FK_CustomerID FOREIGN KEY (CustomerID) REFERENCES
                                                  Customers(ID)
);
```

- Tạo khóa ngoại trong bảng đã có

```
ALTER TABLE Orders
ADD CONSTRAINT FK_CustomerID FOREIGN KEY (CustomerID)
REFERENCES Customers(ID)
;
```

Join

- **Joins:** kết nối các bản ghi từ nhiều bảng với nhau.
- Ví dụ:

```
SELECT ID, NAME, AGE, AMOUNT
FROM CUSTOMERS, ORDERS
WHERE  CUSTOMERS.ID =
ORDERS.CUSTOMER_ID;
```

ID	NAME	AGE	AMOUNT
3	kaushik	23	3000
3	kaushik	23	1500
2	Khilan	25	1560
4	Chaitali	25	2060

Table 1 – CUSTOMERS Table

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Table 2 – ORDERS Table

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

Union

- **UNION** cho phép kết hợp kết quả từ hai hay nhiều câu truy vấn SELECT, mà không trả về các bản ghi trùng lặp. Các câu truy vấn SELECT phải thoả mãn:
  - Cùng số lượng các cột được truy vấn
  - Cùng miền giá trị
- UNION ALL: cho phép trả về các bản ghi trùng lặp
- Cú pháp

```
SELECT column1 [, column2 ]
FROM table1 [, table2 ] [WHERE condition]
UNION
SELECT column1 [, column2 ]
FROM table1 [, table2 ] [WHERE condition]
```

Union

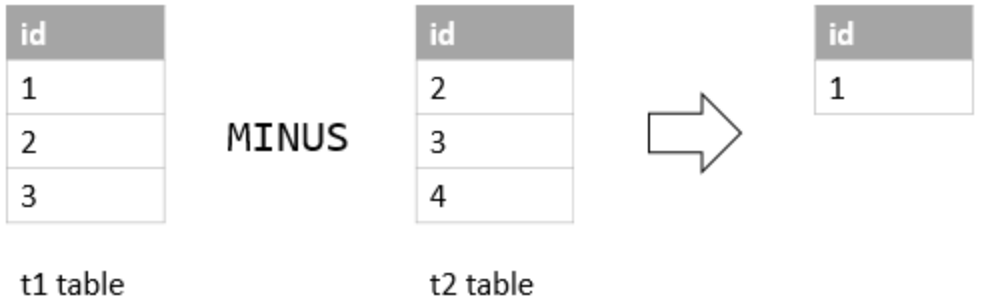
```
SELECT  ID, NAME  FROM CUSTOMERS
WHERE AGE>=25
UNION
SELECT  ID, NAME  FROM CUSTOMERS
WHERE SALARY >=5000
```

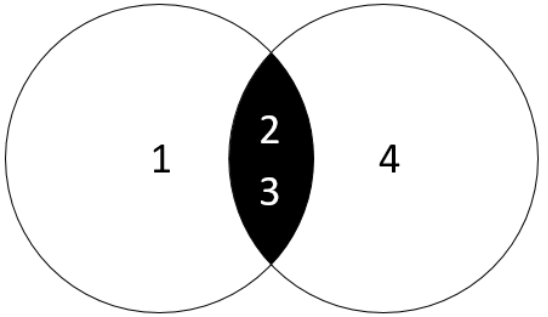
Table 1 – CUSTOMERS Table is as follows.

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

MINUS, INTERSECT

- **MINUS** trả về các bản ghi/dòng có trong câu truy vấn SELECT thứ nhất, nhưng không thuộc câu truy vấn SELECT thứ hai
  - Oracle hỗ trợ MINUS
  - SQL Server, PostgreSQL hỗ trợ EXCEPT
  - MySQL không hỗ trợ MINUS
- **INTERSECT** trả về các bản ghi/dòng có trong cả hai truy vấn SELECT
  - Kết quả trùng lặp bị loại bỏ
  - MySQL không hỗ trợ INTERSECT





ALIAS

- Chúng ta có thể đổi tên cho một cột hoặc một bảng tạm thời bằng cách đưa ra tên khác, được gọi là Alias – bí danh.
  - Tên bảng, cột thực sự không bị thay đổi trong CSDL
  - Tên bí dạnh của cột không được dùng trong WHERE, HAVING. Được dùng trong ORDER BY
- Cú pháp

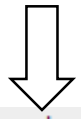
```
SELECT column_name AS alias_name
FROM table_name
WHERE [condition];
```

- Ví dụ

```
SELECT  ID AS CUSTOMER_ID, NAME AS
CUSTOMER_NAME
FROM CUSTOMERS
WHERE SALARY IS NOT NULL;
```

Table 1 – CUSTOMERS Table is as follows.

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00



CUSTOMER_ID	CUSTOMER_NAME
1	Ramesh
2	Khilan
3	kaushik
4	Chaitali
5	Hardik
6	Komal
7	Muffy

Sub Queries

- Sub Queries: là một câu truy vấn nằm bên trong câu truy vấn khác, được đặt ở mệnh đề Where
  - Lệnh ORDER BY không được sử dụng trong câu truy vấn bên trong
  - Câu truy vấn bên trong thường trả về một cột/thuộc tính
- Cú pháp

```
SELECT column_name…
FROM   table…
WHERE  column_name OPERATOR
     (SELECT column_name [, column_name ]
     FROM table [WHERE])
```

Operators

- IN: cho phép kiểm tra một giá trị có nằm trong một tập hợp các giá trị hay không
- Cú pháp:

```
SELECT column1, column2,…
FROM table1, table2,…
WHERE column1 IN (‘value1’, ‘value2’,…);
```

- NOT IN: ngược lại của IN
- EXISTS: kiểm tra câu truy vấn subquery có trả về hàng/bản ghi hay không

```
WHERE EXISTS (subquery)
```

- NOT EXISTS: ngược lại của EXISTS

Functions

- Hàm tính toán trên nhóm các bản ghi
  - MAX/MIN
  - SUM
  - AVG
  - COUNT
- Đưa ra nhân viên có lương cao nhất

```
SELECT sname  FROM  CUSTOMERS
WHERE Salary ≥ ALL(SELECT Salary FROM CUSTOMERS)
```

- Đếm số lượng nhân viên

```
SELECT COUNT(ID)  FROM CUSTOMERS
```

- Đưa ra lương cao nhất

```
SELECT MAX(Salary) FROM  CUSTOMERS
```

Table 1 – CUSTOMERS Table is as follows.

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Sub Queries – Ví dụ

```
SELECT *
    FROM CUSTOMERS
    WHERE ID IN (SELECT CUSTOMER_ID
          FROM ORDERS) ;
```

Table 1 – CUSTOMERS Table

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Table 2 – ORDERS Table

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

Cấu trúc câu truy vấn

SELECT [DISTINCT] <column 1>, <column 2>, …
FROM <table 1>,<table 2>, …
[**WHERE** <condition>]
[**GROUP BY** <column 1>, <column 2>, …
[**HAVING** <condition>]]
[**ORDER BY** <column 1> [**ASC**|**DESC**]]