



ĐẠI HỌC ĐÀ NẴNG – ĐẠI HỌC KINH TẾ  
KHOA THỐNG KÊ TIN HỌC

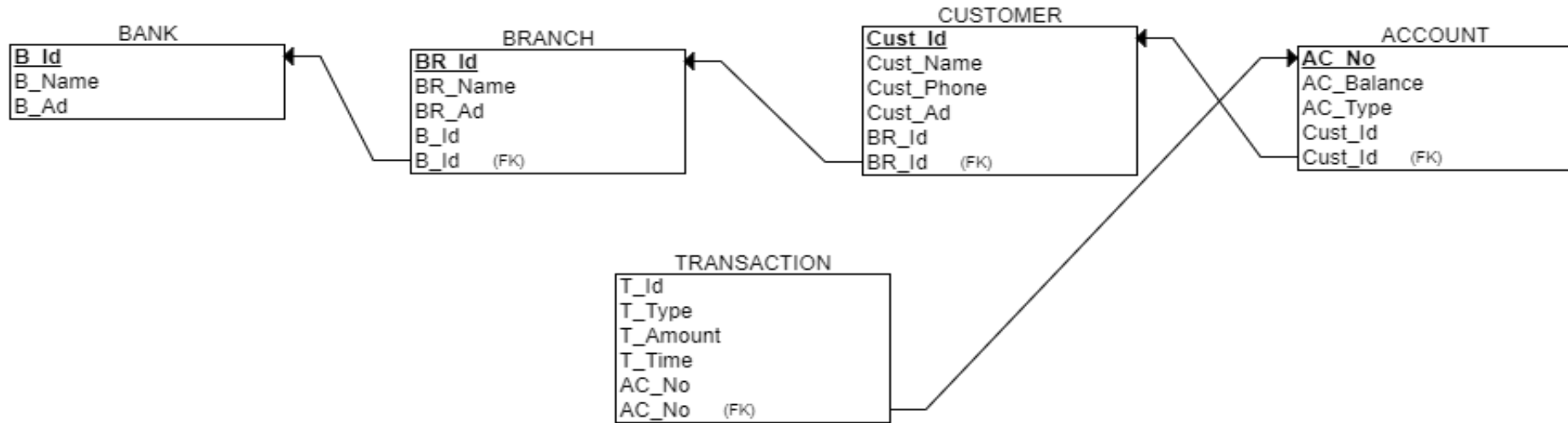
# NGÔN NGỮ SQL

*Giảng viên: Cao Thị Nhâm*





# BANKING Database Diagram





# Ngôn ngữ SQL

---

- SQL (Structured Query Language) – là ngôn ngữ dùng để tương tác với cơ sở dữ liệu
- Chia thành 2 nhóm lệnh:
  - DDL (Data Definition Language)
    - Tạo/chỉnh sửa/xóa CSDL, bảng, index, view,...
  - DML (Data Manipulation Language)
    - Thêm, sửa, xóa và lấy dữ liệu

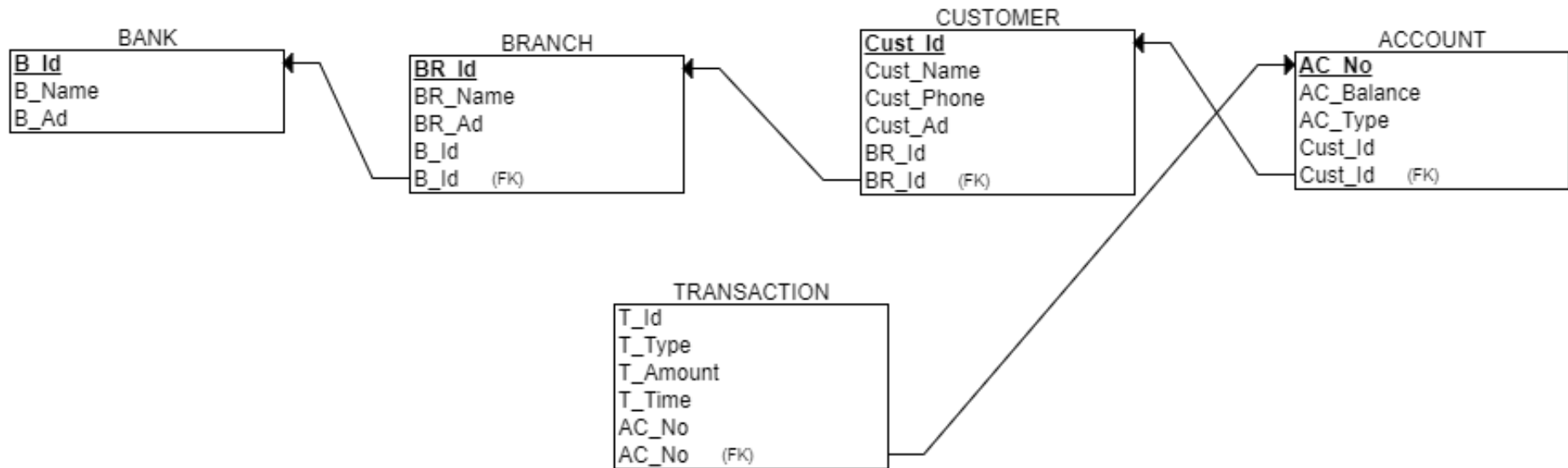
# DML

(DATA MANIPULATION LANGUAGE)





# CƠ SỞ DỮ LIỆU MẪU





# Các loại thao tác

---

- Thêm mới

- INSERT

- SELECT...INTO

- Sửa

- UPDATE

- Xóa

- DELETE

- TRUNCATE

- SELECT



# Lệnh SELECT

**SELECT** [ALL | DISTINCT] danh\_sách\_cột

[INTO [tên\_bảng\_mới]]

**FROM** {tên\_bảng | tên\_view}

\*

cột\_1, cột\_2, ..., cột\_n

[WHERE điều\_kiện\_1]

[GROUP BY danh\_sách\_cột\_1]

[HAVING điều\_kiện\_2]

[ORDER BY danh\_sách\_cột\_2 [ASC | DESC]]



# Các phép toán

---

- Các phép toán toán học

- $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\%$

- Phép so sánh

- $>$ ,  $<$ ,  $\geq$ ,  $\leq$ ,  $\neq$  ( $<>$ ). Có thể kết hợp với ANY, ALL

- Các phép toán logic

- AND, OR, NOT

- Phép toán liên quan đến khoảng

- BETWEEN, NOT BETWEEN

- Phép toán liên quan đến giá trị rời rạc

- IN, NOT IN

- Phép toán liên quan đến chuỗi

- LIKE, NOT LIKE





# Phép toán toán học

---

- Phép toán toán học

```
SELECT ac_balance, ac_balance*0.1 "Tiền lãi"  
FROM ACCOUNT
```



# Phép so sánh

- Phép so sánh: >, >=, <, <=, <>, =

```
SELECT ...  
FROM ...  
WHERE tên_cột <phép_so_sánh> giá_trị
```

- Ví dụ

```
SELECT ac_no, ac_balance  
from account  
where ac_balance > 60000000
```



	ac_no	ac_balance
1	1000000001	88118000
2	1000000002	188372000
3	1000000004	157231000
4	1000000005	178232000
5	1000000007	332183000
6	1000000008	393221000
7	1000000009	172923000
8	1000000011	227449000
9	1000000012	184002000



# Phép toán BETWEEN/NOT BETWEEN

- Kiểm tra giá trị của cột thuộc/không thuộc trong khoảng cho trước

```
SELECT ...  
FROM ... NOT BETWEEN  
WHERE tên_cột BETWEEN giá_trị_1 AND giá_trị_2
```

- Ví dụ

```
SELECT ac_no, ac_balance  
from account  
where ac_balance BETWEEN 60000000 AND 100000000
```



	ac_no	ac_balance
1	1000000001	88118000
2	1000000014	94311000
3	1000000022	69486000
4	1000000029	85747000
5	1000000034	86609000
6	1000000039	83933000
7	1000000047	77659000
8	1000000048	99830000
9	1000000049	91027000
10	1000000050	95035000



# Phép toán IN/NOT IN

- Kiểm tra giá trị của cột có nằm trong danh sách giá trị (rời rạc) cho trước hay không

```
SELECT ...  
FROM ...  
WHERE tên_cột IN/NOT IN(gt1, gt2, gt3, ...)
```

- Ví dụ

```
SELECT cust_id, cust_name  
from customer  
where br_id in ('VT009', 'VB002', 'VT006')
```



	cust_id	cust_name
1	000001	Hà Công Lục
2	000006	Nguyễn Quang Công Minh
3	000008	Nguyễn Lê Minh Quân
4	000012	Hứa Văn Đại
5	000015	Trương Quang Hòa
6	000028	Trương Nhật Minh
7	000030	Nguyễn Văn Hoàng Long
8	000033	Lương Minh Hiếu



# Phép toán LIKE/NOT LIKE

- Tìm kiếm giá trị gần đúng (chỉ áp dụng cho dữ liệu dạng chữ)

```
SELECT ...  
FROM ...  
WHERE tên_cột LIKE/NOT LIKE 'pattern'
```

- ‘Pattern’ – kí tự và các kí tự đại diện

- %: đại diện cho chuỗi bất kì (kể cả chuỗi trống)
- \_(gạch dưới): đại diện cho một kí tự bất kì
- []: đại diện cho một kí tự nằm dấu ngoặc
- [^]: đại diện cho một kí tự không nằm trong dấu ngoặc

```
SELECT cust_name, cust_ad  
FROM customer  
WHERE cust_ad LIKE N'%Quảng Nam'
```



	cust_name	cust_ad
1	Trần Văn Thiện Thanh	19 ĐƯỜNG SỐ 1, ĐIỆN QUANG, ĐIỆN BẢN, QUẢNG NAM
2	Nguyễn Đức Duy	Giáo Tây, Đại Hòa, Đại Lộc, Quảng Nam
3	Lê Tấn Anh Quốc	THÔN THANH QUYẾT 1, ĐIỆN THẮNG TRUNG, ĐIỆN BẢN, Q...
4	Trưởng Quang Hòa	294 HUỖNH THỨC KHÁNG, AN XUÂN, TAM KỶ, QUẢNG NAM
5	Dương Ngọc Long	01 THOẠI NGỌC HẦU, TAM KỶ, QUẢNG NAM
6	Lương Minh Hiếu	ĐỘI 7, THÔN 8, TAM LỘC, PHÚ NINH, QUẢNG NAM



# Phép toán logic

- AND, OR: dùng để kết hợp nhiều điều kiện lại với nhau
- NOT: dùng để phủ định một điều kiện

```
SELECT ...  
FROM ...  
WHERE (điều_kiện_1) AND/OR (điều_kiện_2)...
```

- Ví dụ

Tìm những khách ở chi  
nhánh VT009 và có số điện  
thoại bắt đầu bằng 098



```
SELECT cust_name, cust_ad  
FROM customer  
WHERE br_id = 'VT009' AND cust_phone like '098%'
```



# Những hàm trong câu truy vấn

- Đặt hàm trong các thành phần của truy vấn. Cách dùng hàm:

**Tên\_hàm ([danh\_sách\_tham\_số])**

- Một số hàm thông dụng

Chuỗi	Số	Thời gian	Gộp
LEN	FLOOR	DAY, MONTH, YEAR	COUNT
LOWER, UPPER	SQRT	DATEPART	SUM
LEFT	ROUND	DATENAME	MIN, MAX
RIGHT	RAND	GETDATE	SUM
SUBSTRING		DATEDIFF	AVG
RTRIM, LTRIM, TRIM		DATEADD	



# Hàm thời gian

---

- Lấy ngày giờ hiện tại `GETDATE()`
- Lấy thành phần ngày trong tháng/tháng/năm của dữ liệu dạng thời gian `DAY(giá_trị)` `MONTH(giá_trị)` `YEAR(giá_trị)`
- Lấy thành phần thời gian dạng text (ngày trong tuần, tháng) `DATENAME(giá_trị)`





# Hàm thời gian...

- Lấy thành phần của một giá trị thời gian

- `DATEPART(format, giá_trị)`

- `format` – định dạng thành phần cần lấy

- Ví dụ

```
SELECT DATEPART(yy, GETDATE())
```



	(No column name)
1	2020

year	yy, yyyy
quarter	qq, q
month	mm, m
dayofyear	dy, y
day	dd, d
week	wk, ww
weekday	dw
hour	hh
minute	mi, n
second	ss, s
millisecond	ms
microsecond	mcs
nanosecond	ns



# Hàm thời gian...

- Trừ hai mốc thời gian theo đơn vị cho trước

- `DATEDIFF(format, date_1, date_2)`

- `Date_2 - date_1`

- Ví dụ

```
SELECT DATEDIFF(YY, t_date, GETDATE())  
from transactions
```



	(No column name)
1	9
2	4
3	8
4	10
5	10
6	5
7	6
8	8
9	5
10	3
11	8



# Hàm thời gian...

- Cộng giá trị cho một mốc thời gian

```
SELECT DATEADD(format, giá_trị, cột/ngày_tháng)
```

- Ví dụ

```
select  getdate() "Thời gian hiện tại",  
        dateadd(YYYY, 1, getdate()) "Một năm sau"
```



	Thời gian hiện tại	Một năm sau
1	2020-08-25 14:46:43.283	2021-08-25 14:46:43.283



# Các hàm gộp

---

- Có 2 loại gộp:
  - Gộp đơn giản (toàn bộ dữ liệu trả về)
  - Gộp theo nhóm



# Hàm gộp toàn bộ dữ liệu

```
SELECT tên_hàm1(tên_cột_i), tên_hàm2(tên_cột_j), ...  
FROM ...
```

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lực	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

`SELECT COUNT(CUST_NAME) FROM CUSTOMER`

	(No column name)
1	9

`SELECT COUNT(*) FROM CUSTOMER`

	(No column name)
1	10



# Gộp theo nhóm dữ liệu

```
SELECT danh_sách_cột_1, hàm_gộp_1(tên_cột_i), hàm_gộp_2(tên_cột_j), ...  
FROM ...  
[WHERE ...]  
GROUP BY danh_sách_cột_2
```

- danh\_sách\_cột\_1, danh\_sách\_cột\_2: có các cột giống nhau nhưng thứ tự có thể khác

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trường Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

```
SELECT BR_ID, COUNT(*) SoLuongKH  
FROM CUSTOMER  
GROUP BY BR_ID
```

	BR_ID	SoLuongKH
1	VB001	1
2	VB002	1
3	VB003	1
4	VB004	2
5	VB005	1
6	VT006	1
7	VT009	1
8	VT010	1
9	VT011	1



# Gộp dữ liệu theo nhóm

- Trong trường hợp muốn lọc bản ghi và muốn sử dụng hàm gộp trong điều kiện lọc thì dùng thêm HAVING

```
SELECT danh_sách_cột_1, hàm_gộp_1(tên_cột_i), hàm_gộp_2(tên_cột_j), ...  
FROM ...  
[WHERE ...]  
GROUP BY danh_sách_cột_2  
HAVING điều_kiện
```

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

```
SELECT BR_ID, COUNT(*) SoLuongKH  
FROM CUSTOMER  
GROUP BY BR_ID  
HAVING COUNT(*) > 1
```

	BR_ID	SoLuongKH
1	VB004	2



# Nối bảng

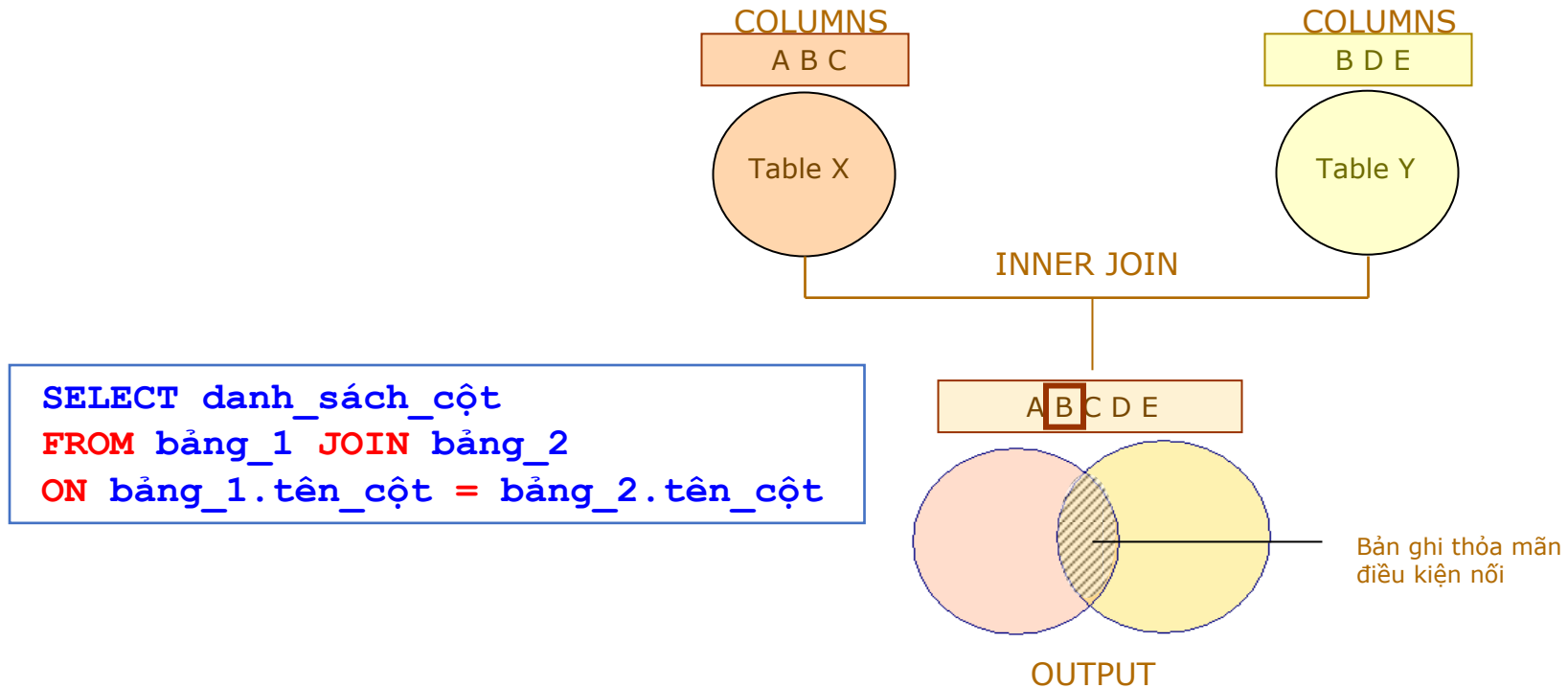
---

- Lấy dữ liệu từ nhiều bảng
- Các loại nối
  - Nối trong
  - Nối ngoài
  - Nối chéo
  - Tự nối





# Nối trong





# Ví dụ phép nối trong

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

	BR_id	BR_name	BR_ad	B_id
1	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
2	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
3	VB003	Vietcombank Thái Ng...	44 Trần Nhân Tông - tp Thái N...	BFTVVNVX07
4	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07

```
select *  
from CUSTOMER join BRANCH on CUSTOMER.br_id = BRANCH.BR_id
```

	CUST_ID	CUST_NAME	BR_ID	BR_id	BR_name	BR_ad	B_id
1	000004	Trần Văn Thiện Thanh	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
2	000005	Nguyễn Đức Duy	VB005	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07
3	000007	Trần Trương Thiện Nguyên	VB001	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
4	000008	Nguyễn Lê Minh Quân	VB002	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
5	000009	Nguyễn Văn Linh	VB003	VB003	Vietcombank Thái Nguyên	44 Trần Nhân Tông - tp Thái Nguyên	BFTVVNVX07
6	000010	Đặng Nhật Phong	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07



# Nối ngoài

---

- Kết quả trả về bao gồm tất cả bản ghi của một bảng và mỗi bản ghi kết nối với bảng còn lại
- Hiển thị giá trị NULL cho những ô không thỏa mãn điều kiện nối
- Có 3 loại:
  - Nối trái
  - Nối phải
  - Nối 2 phía



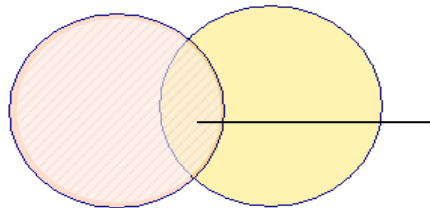
# Nối ngoài...

- Nối trái



```
SELECT danh_sách_cột  
FROM bảng_1 LEFT OUTER JOIN bảng_2  
ON bảng_1.tên_cột = bảng_2.tên_cột
```

A B C D E



Tất cả bản ghi từ bảng X nối với bảng Y, nếu không thỏa mãn điều kiện nối thì để giá trị NULL



# Ví dụ nối trái

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

	BR_id	BR_name	BR_ad	B_id
1	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
2	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
3	VB003	Vietcombank Thái Ng...	44 Trần Nhân Tông - tp Thái N...	BFTVVNVX07
4	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07

```
select *  
from CUSTOMER left outer join BRANCH on CUSTOMER.br_id = BRANCH.BR_id
```

	CUST_ID	CUST_NAME	BR_ID	BR_id	BR_name	BR_ad	B_id
1	000001	Hà Công Lục	VT009	NULL	NULL	NULL	NULL
2	000002	Trần Đức Quý	VT010	NULL	NULL	NULL	NULL
3	000003	Lê Quang Phong	VT011	NULL	NULL	NULL	NULL
4	000004	Trần Văn Thiện Thanh	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	000005	Nguyễn Đức Duy	VB005	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07
6	000006	NULL	VT006	NULL	NULL	NULL	NULL
7	000007	Trần Trương Thiện Nguyên	VB001	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
8	000008	Nguyễn Lê Minh Quân	VB002	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
9	000009	Nguyễn Văn Linh	VB003	VB003	Vietcombank Thái Nguyên	44 Trần Nhân Tông - tp Thái Nguyên	BFTVVNVX07
10	000010	Đặng Nhật Phong	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07



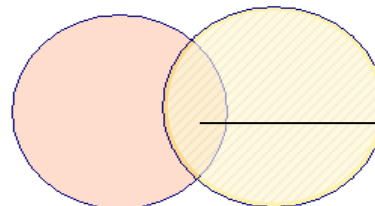
# Nối ngoài...

- Nối phải



```
SELECT danh_sách_cột  
FROM bảng_1 RIGHT OUTER JOIN bảng_2  
ON bảng_1.tên_cột = bảng_2.tên_cột
```

A B C D E



OUTPUT

Tất cả bản ghi từ bảng Y nối với bảng X, nếu không thỏa mãn điều kiện nối thì để giá trị NULL



# Ví dụ nối phải

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

	BR_id	BR_name	BR_ad	B_id
1	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
2	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
3	VB003	Vietcombank Thái Ng...	44 Trần Nhân Tông - tp Thái N...	BFTVVNVX07
4	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07

```
select *  
from CUSTOMER right outer join BRANCH on CUSTOMER.br_id = BRANCH.BR_id
```

	CUST_ID	CUST_NAME	BR_ID	BR_id	BR_name	BR_ad	B_id
1	000007	Trần Trương Thiện Nguyên	VB001	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
2	000008	Nguyễn Lê Minh Quân	VB002	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
3	000009	Nguyễn Văn Linh	VB003	VB003	Vietcombank Thái Nguyên	44 Trần Nhân Tông - tp Thái Nguyên	BFTVVNVX07
4	000004	Trần Văn Thiện Thanh	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	000010	Đặng Nhật Phong	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
6	000005	Nguyễn Đức Duy	VB005	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07



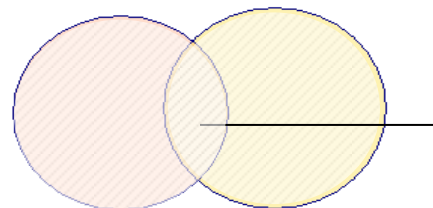
# Nối ngoài...

- Nối 2 phía



```
SELECT danh_sách_cột  
FROM bảng_1 FULL OUTER JOIN bảng_2  
ON bảng_1.tên_cột = bảng_2.tên_cột
```

Output columns: **A B C D E**



Tất cả bản ghi từ bảng X và bảng Y. Bản ghi giống nhau chỉ xuất hiện 1 lần

OUTPUT





# Ví dụ nổi đầy đủ

	CUST_ID	CUST_NAME	BR_ID
1	000001	Hà Công Lục	VT009
2	000002	Trần Đức Quý	VT010
3	000003	Lê Quang Phong	VT011
4	000004	Trần Văn Thiện Thanh	VB004
5	000005	Nguyễn Đức Duy	VB005
6	000006	NULL	VT006
7	000007	Trần Trương Thiện Nguyên	VB001
8	000008	Nguyễn Lê Minh Quân	VB002
9	000009	Nguyễn Văn Linh	VB003
10	000010	Đặng Nhật Phong	VB004

	BR_id	BR_name	BR_ad	B_id
1	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
2	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
3	VB003	Vietcombank Thái Ng...	44 Trần Nhân Tông - tp Thái N...	BFTVVNVX07
4	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07

```
select *  
from CUSTOMER full outer join BRANCH on CUSTOMER.br_id = BRANCH.BR_id
```

	CUST_ID	CUST_NAME	BR_ID	BR_id	BR_name	BR_ad	B_id
1	000001	Hà Công Lục	VT009	NULL	NULL	NULL	NULL
2	000002	Trần Đức Quý	VT010	NULL	NULL	NULL	NULL
3	000003	Lê Quang Phong	VT011	NULL	NULL	NULL	NULL
4	000004	Trần Văn Thiện Thanh	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07
5	000005	Nguyễn Đức Duy	VB005	VB005	Vietcombank Bắc Ninh	98 Phan Huy Chú	BFTVVNVX07
6	000006	NULL	VT006	NULL	NULL	NULL	NULL
7	000007	Trần Trương Thiện Nguyên	VB001	VB001	Vietcombank Thái Bình	12 Lý Bôn - tp Thái Bình	BFTVVNVX07
8	000008	Nguyễn Lê Minh Quân	VB002	VB002	Vietcombank Nam Định	04 Trần Hưng Đạo - tp Nam Định	BFTVVNVX07
9	000009	Nguyễn Văn Linh	VB003	VB003	Vietcombank Thái Nguyên	44 Trần Nhân Tông - tp Thái Nguyên	BFTVVNVX07
10	000010	Đặng Nhật Phong	VB004	VB004	Vietcombank Hà Nội	32 Cầu Giấy	BFTVVNVX07



# Truy vấn lồng

---

- Là gì?
  - Đặt truy vấn bên trong một truy vấn khác
  - Truy vấn con bao giờ cũng đặt trong dấu ngoặc tròn ( )
- Truy vấn con có thể đặt trong mệnh đề
  - SELECT
  - FROM
  - WHERE
  - GROUP BY
  - HAVING



# Truy vấn lồng...

- Ví dụ:

```
select cust_name, br_id
from customer
where br_id = ( select customer.br_id
                from customer join branch on customer.Br_id = branch.BR_id
                where cust_name = N'Hà Công Lực')
```



	cust_name	br_id
1	Hà Công Lực	VT009
2	Trương Quang Hòa	VT009
3	Trương Nhật Minh	VT009
4	Nguyễn Văn Hoàng Long	VT009
5	Lương Minh Hiếu	VT009




# Truy vấn lồng...


- Trong phép so sánh có thể bổ sung thêm từ khóa ANY/SOME và ALL

```
SELECT ...  
FROM ...  
WHERE tên_cột <phép_so_sánh> ALL/SOME(SELECT...FROM...)
```

```
SELECT AC_NO, ac_balance  
from account  
where ac_balance > all (select ac_balance  
                        from account join customer on account.cust_id = customer.Cust_id  
                        where br_id = 'VB004')
```



	ac_balance
1	157231000
2	-6107000



	AC_NO	ac_balance
1	1000000002	188372000
2	1000000005	178232000
3	1000000007	332183000
4	1000000008	393221000
5	1000000009	172923000
6	1000000011	227449000
7	1000000012	184002000
8	1000000013	229384000



# Nối giữa truy vấn con và truy vấn cha

```
select cust_name, br_id
from customer c1
where EXISTS(select cust_id, count(ac_no)
              from account
              where c1.Cust_id = account.cust_id
              group by cust_id
              having count(ac_no) >= 2)
```



# Thêm mới

- INSERT đầy đủ

```
INSERT INTO tên_bảng VALUES(giá_trị_1, giá_trị_2, giá_trị_3,...)
```

- Giá\_trị\_1, giá\_trị\_2,... theo đúng thứ tự cột

- INSERT khuyết thiếu

```
INSERT INTO tên_bảng(cột_1, cột_2, cột_3,...)  
VALUES(giá_trị_1, giá_trị_2, giá_trị_3,...)
```

- Giá\_trị\_1, giá\_trị\_2,...theo đúng thứ tự của cột\_1, cột\_2,...

- INSERT dữ liệu từ câu truy vấn

```
INSERT INTO tên_bảng(...)  
SELECT ... FROM ...
```

- Cột trong lệnh truy vấn phải tương ứng với cột trong *tên\_bảng*



# Cập nhật

- Cú pháp

```
update tên_bảng  
set tên_cột_1 = giá_trị_1,  
    tên_cột_2 = giá_trị_2, ...  
from [tên_bảng JOIN...]  
where điều_kiện
```



# Cập nhật...

---

- Ví dụ

```
update customer  
set cust_name = branch.BR_id + 'no name'  
from customer join branch on customer.Br_id = Branch.BR_id  
where cust_name is null
```





# Xóa

---

- Lần lượt từng bản ghi

```
DELETE [FROM] tên_bảng  
[WHERE điều_kiện]
```

- Toàn bộ dữ liệu trong bảng

```
TRUNCATE TABLE tên_bảng
```

# DDL

(DATA DEFINITION LANGUAGE)





# Các đối tượng

---

- Cơ sở dữ liệu
- Bảng
- Ràng buộc



# Kiểu dữ liệu

Kiểu số	Kiểu chữ	Kiểu thời gian	Kiểu khác
BIGINT	CHAR	DATETIME	VARBINARY
INT	NCHAR	SMALLDATETIME	
SMALLINT	VARCHAR	DATE	
TINYINT	NVARCHAR	TIME	
BIT		DATETIME2	
NUMERIC		DATETIMEOFFSET	
MONEY			
SMALLMONEY			
FLOAT			
REAL			

(Microsoft SQL Server 2010 trở đi)



# KIỂU DỮ LIỆU...

---

- CHAR và VARCHAR

- CHAR(n): độ dài cố định
- VARCHAR(n): độ dài không cố định

- Ví dụ:

- CHAR (3) → Nhập: "AB" → Giá trị lưu trong CSDL: "AB " (thêm dấu cách cuối chuỗi)
- VARCHAR(3) → Nhập: "AB" → Giá trị lưu trong CSDL: "AB"

- NCHAR/NVARCHAR và CHAR/VARCHAR

- "N": lưu trữ text dạng Unicode (2 byte lưu 1 kí tự)



# Tạo cơ sở dữ liệu

CREATE DATABASE tên\_db

Bank  
Bank\_log  
master  
mastlog  
model  
model\_msdbdata  
model\_msdblog  
model\_replicatedmaster  
model\_replicatedmaster\_log  
modellog  
MSDBData  
MSDBLog  
QLKH  
QLKH\_log

```
CREATE DATABASE database_name
[ CONTAINMENT = { NONE | PARTIAL } ]
[ ON
    [ PRIMARY ] <filespec> [ ,...n ]
    [ , <filegroup> [ ,...n ] ]
    [ LOG ON <filespec> [ ,...n ] ]
]
[ COLLATE collation_name ]
[ WITH <option> [ ,...n ] ]
[;]

<option> ::=
{
    FILESTREAM ( <filestream_option> [ ,...n ] )
    | DEFAULT_FULLTEXT_LANGUAGE = { lcid | language_name | language_alias }
    | DEFAULT_LANGUAGE = { lcid | language_name | language_alias }
    | NESTED_TRIGGERS = { OFF | ON }
    | TRANSFORM_NOISE_WORDS = { OFF | ON }
    | TWO_DIGIT_YEAR_CUTOFF = <two_digit_year_cutoff>
    | DB_CHAINING { OFF | ON }
    | TRUSTWORTHY { OFF | ON }
    | PERSISTENT_LOG_BUFFER=ON ( DIRECTORY_NAME='<Filepath to folder on DAX formatted volume>' )
}

<filestream_option> ::=
{
    NON_TRANSACTED_ACCESS = { OFF | READ_ONLY | FULL }
    | DIRECTORY_NAME = 'directory_name'
}

<filespec> ::=
{
    (
        NAME = logical_file_name ,
        FILENAME = { 'os_file_name' | 'filestream_path' }
        [ , SIZE = size [ KB | MB | GB | TB ] ]
        [ , MAXSIZE = { max_size [ KB | MB | GB | TB ] | UNLIMITED } ]
        [ , FILEGROWTH = growth_increment [ KB | MB | GB | TB | % ] ]
    )
}

<filegroup> ::=
{
    FILEGROUP filegroup_name [ [ CONTAINS FILESTREAM ] [ DEFAULT ] | CONTAINS MEMORY_OPTIMIZED_DATA ]
    <filespec> [ ,...n ]
}
```



# Tạo bảng

```
CREATE TABLE tên_bảng  
(  
    tên_cột_1 kiểu_dữ_liệu,  
    tên_cột_2 kiểu_dữ_liệu,  
    ....,  
    primary key(tên_cột_x, ...),  
    foreign key (tên_cột_y) references tên_bảng_1,  
    ....  
)
```

```
create table Branch  
(  
    BR_id CHAR(5),  
    BR_name nvarchar(50),  
    BR_ad    nvarchar(150),  
    B_id     char(10),  
    primary key(BR_id),  
    foreign key (B_id) references Bank  
)
```



# Các thao tác thay đổi cấu trúc bảng

- Dùng lệnh ALTER để:
  - Thêm/bỏ cột
  - Thêm/bỏ ràng buộc
  - Định nghĩa lại kiểu dữ liệu cho cột

```
ALTER TABLE tên_bảng DROP COLUMN tên_cột
```

```
ALTER TABLE tên_bảng ALTER COLUMN tên_cột <kiểu_dữ_liệu>
```

```
ALTER TABLE tên_bảng ADD tên_cột <kiểu_dữ_liệu>
```

```
ALTER TABLE tên_bảng ADD CONSTRAINT tên_constraint...
```

```
ALTER TABLE tên_bảng DROP CONSTRAINT tên_constraint
```





# Ràng buộc

---

- **Ràng buộc (Constraint)** là các quy tắc được áp dụng trên các cột dữ liệu của một bảng để kiểm tra tính hợp lệ của dữ liệu đầu vào
- Các loại ràng buộc
  - **NOT NULL**: bắt buộc nhập giá trị
  - **PRIMARY KEY**
  - **FOREIGN KEY**
  - **UNIQUE**: kiểm tra giá trị duy nhất
  - **CHECK**: kiểm tra giá trị nhập vào thuộc một miền cho trước



# Ràng buộc...

- NOT NULL

```
CREATE TABLE tên_bảng
(
    tên_cột_1 kiểu_dữ_liệu NOT NULL,
    tên_cột_2 kiểu_dữ_liệu,
    ....,
    primary key(tên_cột_x, ...),
    foreign key (tên_cột_y) references tên_bảng_1,
    ...
)
```

```
ALTER TABLE tên_bảng ALTER COLUMN tên_cột <kiểu_dữ_liệu> NOT NULL
```



# Ràng buộc...

- UNIQUE

```
CREATE TABLE tên_bảng
(
    tên_cột_1 kiểu_dữ_liệu UNIQUE,
    tên_cột_2 kiểu_dữ_liệu,
    ....,
    primary key(tên_cột_x, ...),
    foreign key (tên_cột_y) references tên_bảng_1,
    ...
)
```

```
CREATE TABLE tên_bảng
(
    tên_cột_1 kiểu_dữ_liệu,
    tên_cột_2 kiểu_dữ_liệu,
    ....,
    primary key(tên_cột_x, ...),
    foreign key (tên_cột_y) references tên_bảng_1,
    ....,
    constraint tên_constraint UNIQUE(tên_cột_i),
    ...
)
```

```
ALTER TABLE tên_bảng ADD CONSTRAINT tên_constraint UNIQUE(tên_cột)
```



# Ràng buộc...

- CHECK

```
CREATE TABLE tên_bảng
(
    tên_cột_1 kiểu_dữ_liệu,
    tên_cột_2 kiểu_dữ_liệu,
    ....,
    primary key(tên_cột_x, ...),
    foreign key (tên_cột_y) references tên_bảng_1,
    ...,
    constraint tên_constraint CHECK(điều_kiện),
    ...
)
```

```
CREATE TABLE tên_bảng
(
    tên_cột_1 kiểu_dữ_liệu CHECK(điều_kiện),
    tên_cột_2 kiểu_dữ_liệu,
    ....,
    primary key(tên_cột_x, ...),
    foreign key (tên_cột_y) references tên_bảng_1,
    ...
)
```

```
ALTER TABLE tên_bảng ADD CONSTRAINT tên_constraint CHECK(điều_kiện)
```

PHONGBAN		
Tên cột	Kiểu dữ liệu	Ràng buộc
MaPB	CHAR(3)	PK
TenPB	NVARCHAR(50)	
NguoiQuanLy	CHAR(3)	

THAM GIA		
Tên cột	Kiểu dữ liệu	Ràng buộc
MaNV	CHAR(3)	PK,FK
MaCT	CHAR(3)	PK,FK
SoLuongNgayCong	Numeric(4)	

CONGTRINH		
Tên cột	Kiểu dữ liệu	Ràng buộc
MaCT	CHAR(3)	PK
TenCT	NVARCHAR(50)	
NgayKC	Date	
NgayHT	Date	
DiaDiem	NVARCHAR(100)	

NHANVIEN		
Tên cột	Kiểu dữ liệu	Ràng buộc
MaNV	CHAR(3)	PK
TenNV	NVARCHAR(50)	
NgaySinh	Date	
DiaChi	NVARCHAR(100)	
GioiTinh	Numeric(1)	
Luong	Numeric(12)	
MaPB	CHAR(3)	FK

## BÀI TẬP VÍ DỤ



# KẾT QUẢ BÀI TẬP VÍ DỤ

