

BRANCH_ID	VARCHAR	Mã chi nhánh nhân viên làm việc
-----------	---------	---------------------------------

Bảng 2. Thông tin bảng Employees

- CUSTOMERS

Thuộc tính	Kiểu dữ liệu	Mô tả
CUS_ID	VARCHAR	Mã khách hàng
CUS_NAME	VARCHAR	Tên khách hàng
CUS_PHONE	VARCHAR	Số điện thoại khách hàng
CUS_GENDER	VARCHAR	Giới tính
CUS_ADDRESS	VARCHAR	Địa chỉ khách hàng
BRANCH_ID	VARCHAR	Mã chi nhánh khác hàng mua

Bảng 3. Thông tin bảng Customer

- PRODUCTS

Thuộc tính	Kiểu dữ liệu	Mô tả
PRODUCT_ID	VARCHAR	Mã sản phẩm
PRODUCT_NAME	VARCHAR	Tên sản phẩm
PRODUCT_PRICE	NUMBER	Giá sản phẩm
PRODUCT_DESCRIPTION	VARCHAR	Mô tả sản phẩm
PRODUCT_TYPE	VARCHAR	Loại sản phẩm
WARRANTY_PERIOD	NUMBER	Thời gian bảo hành sản phẩm

Bảng 4. Thông tin bảng Products

- WAREHOUSE_MANAGER

Thuộc tính	Kiểu dữ liệu	Mô tả
BRANCH_ID	VARCHAR	Mã chi nhánh
PRODUCT_ID	VARCHAR	Mã sản phẩm
DATE_UPDATE	DATE	Địa chỉ chi nhánh

QUANTITY	NUMBER	Số lượng sản phẩm trong kho
MANUFACTUREDATE	DATE	Ngày sản xuất sản phẩm

Bảng 5. Thông tin bảng Warehouse_manager

- WAREHOUSE_SALER

Thuộc tính	Kiểu dữ liệu	Mô tả
BRANCH_ID	VARCHAR	Mã chi nhánh
PRODUCT_ID	VARCHAR	Mã sản phẩm
PRODUCT_STATUS	NUMBER	Tình trạng sản phẩm

Bảng 6. Thông tin bảng Warehouse_Saler

- ORDERS

Thuộc tính	Kiểu dữ liệu	Mô tả
ORDER_ID	NUMBER	Mã đơn hàng
CUS_ID	VARCHAR	Mã khách mua hàng
ORDER_DATE	DATE	Ngày đặt hàng
BRANCH_ID	VARCHAR	Mã chi nhánh
EMPLOYEE_ID	VARCHAR	Mã nhân viên
TOTAL_PRICE	NUMBER	Tổng tiền hóa đơn

Bảng 7. Thông tin bảng Orders

- ORDER_DETAILS

Thuộc tính	Kiểu dữ liệu	Mô tả
ORDER_ID	NUMBER	Mã đơn hàng
PRODUCT_ID	VARCHAR	Mã sản phẩm
QUANTITY	NUMBER	Số lượng sản phẩm
PRICE	NUMBER	Giá sản phẩm

Bảng 8. Thông tin bảng Order_details

C. Kiến trúc phân mảnh

1) Mô tả phân mảnh

- Quan hệ BRANCHES là phân mảnh ngang chính.
- Quan hệ EMPLOYEES, ORDERS, ORDER_DETAILS là phân mảnh ngang dẫn xuất.
- Quan hệ WAREHOUSE được phân mảnh hỗn hợp thành WAREHOUSE_MANAGER và WAREHOUSE_SALER.
- Quan hệ CUSTOMERS, PRODUCTS được nhân bản tại tất cả các chi nhánh.

2) Quan hệ trên 2 chi nhánh

- Quan hệ BRANCHES là phân mảnh ngang chính:

$CN1.BRANCHES = \delta_{(BRANCH_NAME = "Chi\ nhánh\ 1")} BRANCHES$
 $CN2.BRANCHES = \delta_{(BRANCH_NAME = "Chi\ nhánh\ 2")} BRANCHES$

Hình 2. Quan hệ Branches

- Quan hệ EMPLOYEES, ORDERS, ORDER_DETAILS là phân mảnh ngang dẫn xuất:

$CN1.EMPLOYEES = EMPLOYEES \bowtie_{BRANCH_ID} CN1.BRANCHES$
 $CN2.EMPLOYEES = EMPLOYEES \bowtie_{BRANCH_ID} CN2.BRANCHES$
 $CN1.ORDER_ID = ORDER_ID \bowtie_{EMPLOYEE_ID} CN1.EMPLOYEES$

Hình 3. Quan hệ Employees và Order

$CN2.ORDER_ID = ORDER_ID \bowtie_{EMPLOYEE_ID} CN2.EMPLOYEES$
 $CN1.ORDER_DETAILS = ORDER_DETAILS \bowtie_{ORDER_ID} CN1.ORDER_ID$
 $CN2.TRANSACTION = ORDER_DETAILS \bowtie_{ORDER_ID} CN2.ORDER_ID$

Hình 4. Quan hệ Order và Order_details

- Quan hệ WAREHOUSE được phân mảnh hỗn hợp:

$CN1.WAREHOUSE_MANAGER =$
 $\pi_{(BRANCH_ID, PRODUCT_ID, DATE_UPDATE, QUANTITY, MANUFACTUREDATE)}$
 $(WAREHOUSE \bowtie_{BRANCH_ID} CN1.BRANCHES)$

 $CN2.WAREHOUSE_MANAGER =$
 $\pi_{(BRANCH_ID, PRODUCT_ID, DATE_UPDATE, QUANTITY, MANUFACTUREDATE)}$
 $(WAREHOUSE \bowtie_{BRANCH_ID} CN1.BRANCHES)$

 $CN1.WAREHOUSE_SALER =$
 $\pi_{(BRANCH_ID, PRODUCT_ID, PRODUCT_STATUS)}(WAREHOUSE \bowtie_{BRANCH_ID}$
 $CN1.BRANCHES)$

 $CN2.WAREHOUSE_SALER =$
 $\pi_{(BRANCH_ID, PRODUCT_ID, PRODUCT_STATUS)}(WAREHOUSE \bowtie_{BRANCH_ID}$
 $CN1.BRANCHES)$

Hình 5. Quan hệ Warehouse

- Quan hệ CUSTOMERS, PRODUCTS được nhân bản tất cả chi nhánh.

D. Kiến trúc phân quyền

1) Tài khoản DIRECTOR

- Có quyền Connect.
- Được phép xem, truy vấn tất cả các bảng ở 2 chi nhánh.
- Được phép thêm, sửa, xóa ở các bảng: BRANCHES, PRODUCTS, CUSTOMERS, EMPLOYEES, WAREHOUSE_MANAGER và WAREHOUSE_SALER ở cả 2 chi nhánh.

2) Tài khoản WAREHOUSE_MANAGER

- Có quyền Connect.
- Có quyền xem, truy vấn ở bảng BRANCHES tại chi nhánh mình quản lý.
- Có quyền xem, thêm, sửa ở bảng PRODUCTS tại chi nhánh mình quản lý.
- Có quyền xem, thêm, xóa, sửa ở bảng WAREHOUSE_MANAGER và WAREHOUSE_SALER tại chi nhánh mình quản lý.

3) Tài khoản EMPLOYEE

- Có quyền Connect.
- Có quyền xem, truy vấn ở tất cả các bảng trừ WAREHOUSE_MANAGER tại chi nhánh mình làm việc

III. THỰC HIỆN TRUY VẤN TRÊN

MÔI TRƯỜNG PHÂN TÁN

1) Cho biết thông tin sản phẩm (PID, PNAME, PRICE) có số lượng bán nhiều nhất và có giá bán lớn hơn 5.000.000 VNĐ

```
SELECT P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE,
       SUM(OD.QUANTITY) AS TOTAL
FROM CHINHANH1.PRODUCTS P
INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE
HAVING P.PRODUCT_PRICE > 5000000
ORDER BY TOTAL DESC;
```

Kết quả:

- Chi nhánh 1:

```
SQL> SELECT P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE, SUM(OD.QUANTITY) AS TOTAL
2 FROM CHINHANH1.PRODUCTS P
3 INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
4 GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE
5 HAVING P.PRODUCT_PRICE > 5000000
6 ORDER BY TOTAL DESC;
```

PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE
TOTAL		
P1	Tivi LED 4K 55 inch	12000000
12		
P3	Tu Lanh Side by Side	18000000
8		
P11	Dieu Hoa Inverter 2HP	10500000
7		
PRODUCT_ID PRODUCT_NAME PRODUCT_PRICE		
TOTAL		
P4	May Giat Front Load 8kg	9500000
5		
P2	May Lanh Inverter 1.5HP	8500000
5		
P10	May Pha Ca Phe Espresso	5500000
4		

6 rows selected.

Hình 6. Kết quả câu truy vấn 1 trên chi nhánh 1

- Chi nhánh 2:

```
SQL> SELECT P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE, SUM(OD.QUANTITY) AS TOTAL
2 FROM CHINHANH2.PRODUCTS P
3 INNER JOIN CHINHANH2.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
4 GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME, P.PRODUCT_PRICE
5 HAVING P.PRODUCT_PRICE > 5000000
6 ORDER BY TOTAL DESC;
```

PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE
TOTAL		
P3	Tu Lanh Side by Side	18000000
17		
P1	Tivi LED 4K 55 inch	12000000
16		
P2	May Lanh Inverter 1.5HP	8500000
11		
PRODUCT_ID PRODUCT_NAME PRODUCT_PRICE		
TOTAL		
P11	Dieu Hoa Inverter 2HP	10500000
3		
P4	May Giat Front Load 8kg	9500000
3		
P10	May Pha Ca Phe Espresso	5500000
2		

Hình 7. Kết quả câu truy vấn 1 trên chi nhánh 2

2) In ra khách hàng có số lượng đơn hàng nhiều nhất

```
SELECT O.CUS_ID, C.CUS_NAME, COUNT(O.ORDER_ID) AS TOTAL
FROM CHINHANH1.ORDERS O
INNER JOIN CHINHANH1.CUSTOMERS C ON O.CUS_ID = C.CUS_ID
GROUP BY O.CUS_ID, C.CUS_NAME
ORDER BY TOTAL DESC;
```

Kết quả:

- Chi nhánh 1:

```
SQL> SELECT O.CUS_ID, C.CUS_NAME, COUNT(O.ORDER_ID) AS TOTAL
2 FROM CHINHANH1.ORDERS O
3 INNER JOIN CHINHANH1.CUSTOMERS C ON O.CUS_ID = C.CUS_ID
4 GROUP BY O.CUS_ID, C.CUS_NAME
5 ORDER BY TOTAL DESC;
```

CUS_ID	CUS_NAME	TOTAL
KH110	Huynh Thi Kieu Diem	4
KH216	Vo Thi Kim Ngan	1
KH103	Pham Thi Bao Anh	1
KH104	Tran Thi Kim Anh	1
KH105	Ha Van Hung	1
KH106	Bui Bao Bich	1
KH107	Nguyen Thi Thuy	1
KH108	Nguyen Van Thanh	1
KH109	Nguyen Hai Tu	1
KH111	Nguyen Thanh Huyen	1
KH112	Le Van An	1
CUS_ID CUS_NAME TOTAL		
KH113	Tran Thi Mai	1
KH117	Do Manh Tung	1
KH118	Trinh Ngoc Linh	1
KH119	Vo Van Long	1
KH206	Vo Van Duc	1
KH214	Tran Thi Minh Chau	1
KH101	Nguyen Van An	1
KH102	Tran Phuoc Thinh	1

19 rows selected.

Hình 8. Kết quả câu truy vấn 2 trên chi nhánh 1

- Chi nhánh 2:

```
SQL> SELECT O.CUS_ID, C.CUS_NAME, COUNT(O.ORDER_ID) AS TOTAL
2 FROM CHINHANH2.ORDERS O
3 INNER JOIN CHINHANH2.CUSTOMERS C ON O.CUS_ID = C.CUS_ID
4 GROUP BY O.CUS_ID, C.CUS_NAME
5 ORDER BY TOTAL DESC;
```

CUS_ID	CUS_NAME	TOTAL
KH212	Nguyen Thi Ngoc Tram	4
KH216	Vo Thi Kim Ngan	2
KH211	Pham Van An	2
KH214	Tran Thi Minh Chau	2
KH101	Nguyen Van An	1
KH215	Nguyen Van Duc	1
KH217	Tran Van Hung	1
KH206	Vo Van Duc	1
KH111	Nguyen Thanh Huyen	1
KH213	Le Van Long	1

10 rows selected.

Hình 9. Kết quả câu truy vấn 2 trên chi nhánh 2

3) Liệt kê sản phẩm được bán nhiều nhất ở 2 chi nhánh

```
SELECT P.PRODUCT_ID, P.PRODUCT_NAME
FROM CHINHANH1.PRODUCTS P
INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME
HAVING SUM(OD.QUANTITY) =
(
    SELECT MAX(SUM(OD.QUANTITY))
    FROM CHINHANH1.ORDER_DETAILS OD
    GROUP BY OD.PRODUCT_ID
)
UNION ALL
SELECT P.PRODUCT_ID, P.PRODUCT_NAME
FROM CHINHANH2.PRODUCTS@DIRECTOR_LINK P
INNER JOIN CHINHANH2.ORDER_DETAILS@DIRECTOR_LINK OD
ON P.PRODUCT_ID = OD.PRODUCT_ID
GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME
HAVING SUM(OD.QUANTITY) =
(
    SELECT MAX(SUM(OD.QUANTITY))
    FROM CHINHANH2.ORDER_DETAILS@DIRECTOR_LINK OD
    GROUP BY OD.PRODUCT_ID
);
```

Kết quả:

```
SQL> SELECT P.PRODUCT_ID, P.PRODUCT_NAME
2 FROM CHINHANH1.PRODUCTS P
3 INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
4 GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME
5 HAVING SUM(OD.QUANTITY) = (SELECT MAX(SUM(OD.QUANTITY)) FROM CHINHANH1.ORDER_DETAILS OD GROUP BY
OD.PRODUCT_ID)
6 UNION ALL
7 SELECT P.PRODUCT_ID, P.PRODUCT_NAME
8 FROM CHINHANH2.PRODUCTS@DIRECTOR_LINK P
9 INNER JOIN CHINHANH2.ORDER_DETAILS@DIRECTOR_LINK OD ON P.PRODUCT_ID = OD.PRODUCT_ID
10 GROUP BY P.PRODUCT_ID, P.PRODUCT_NAME
11 HAVING SUM(OD.QUANTITY) = (SELECT MAX(SUM(OD.QUANTITY)) FROM CHINHANH2.ORDER_DETAILS@DIRECTOR_LINK
12 OD GROUP BY OD.PRODUCT_ID);
PRODUCT_ID PRODUCT_NAME
-----
P1         Tivi LED 4K 55 inch
P18        May Say Toc Ion
P2         Tu Lanh Side by Side
```

Hình 10. Kết quả câu truy vấn 3 trên cả 2 chi nhánh

4) Liệt kê danh sách khách hàng mua sản phẩm P1 ở cả 2 chi nhánh

```
SELECT O.CUS_ID
FROM CHINHANH1.ORDER_DETAILS OD
JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
WHERE OD.PRODUCT_ID = 'P1' AND O.BRANCH_ID = 'CN1'
INTERSECT
SELECT O.CUS_ID
FROM CHINHANH2.ORDER_DETAILS@EMPLOYEE_LINK OD
JOIN CHINHANH2.ORDERS@EMPLOYEE_LINK O ON OD.ORDER_ID = O.ORDER_ID
WHERE OD.PRODUCT_ID = 'P1' AND O.BRANCH_ID = 'CN2';
```

Kết quả:

```
SQL> SELECT O.CUS_ID
2 FROM CHINHANH1.ORDER_DETAILS OD
3 JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
4 WHERE OD.PRODUCT_ID = 'P1' AND O.BRANCH_ID = 'CN1'
5 INTERSECT
6 SELECT O.CUS_ID
7 FROM CHINHANH2.ORDER_DETAILS@EMPLOYEE_LINK OD
8 JOIN CHINHANH2.ORDERS@EMPLOYEE_LINK O ON OD.ORDER_ID = O.ORDER_ID
9 WHERE OD.PRODUCT_ID = 'P1' AND O.BRANCH_ID = 'CN2';
CUS_ID
-----
KH101
KH111
KH214
KH206
```

Hình 11. Kết quả câu truy vấn 4 trên cả 2 chi nhánh

5) Liệt kê danh sách sản phẩm còn hàng ở chi nhánh 1 nhưng không còn hàng ở chi nhánh 2

```
SELECT P.PRODUCT_ID, P.PRODUCT_NAME
FROM CHINHANH1.PRODUCTS P
INNER JOIN CHINHANH1.WAREHOUSE_MANAGER WM ON P.PRODUCT_ID = WM.PRODUCT_ID
WHERE WM.QUANTITY > 0
MINUS
SELECT P.PRODUCT_ID, P.PRODUCT_NAME
FROM CHINHANH2.PRODUCTS@WAREHOUSE_MANAGER_LINK P
INNER JOIN CHINHANH2.WAREHOUSE_MANAGER@WAREHOUSE_MANAGER_LINK WM
ON P.PRODUCT_ID = WM.PRODUCT_ID
WHERE WM.QUANTITY > 0;
```

Kết quả:

```
SQL> SELECT P.PRODUCT_ID, P.PRODUCT_NAME
2 FROM CHINHANH1.PRODUCTS P
3 INNER JOIN CHINHANH1.WAREHOUSE_MANAGER WM ON P.PRODUCT_ID = WM.PRODUCT_ID
4 WHERE WM.QUANTITY > 0
5 MINUS
6 SELECT P.PRODUCT_ID, P.PRODUCT_NAME
7 FROM CHINHANH2.PRODUCTS@WAREHOUSE_MANAGER_LINK P
8 INNER JOIN CHINHANH2.WAREHOUSE_MANAGER@WAREHOUSE_MANAGER_LINK WM ON P.PRODUCT_ID = WM.PRODUCT_ID
9 WHERE WM.QUANTITY > 0;
PRODUCT_ID PRODUCT_NAME
-----
P11         Dieu Hoa Inverter 2HP
P16         Den LED Trang Tri
P7          May Xay Ca Phes
```

Hình 12. Kết quả câu truy vấn 5 trên cả 2 chi nhánh

6) Tìm khách hàng có giới tính nữ đã mua tất cả sản phẩm có giá trên 12.000.000 VNĐ vào ngày 20/10

```
SELECT DISTINCT C.CUS_ID, C.CUS_NAME
FROM CHINHANH1.CUSTOMERS C
WHERE C.CUS_GENDER = 'Nu'
AND NOT EXISTS (
SELECT *
FROM CHINHANH1.PRODUCTS P
WHERE P.PRODUCT_PRICE >= 12000000
AND NOT EXISTS (
SELECT *
FROM CHINHANH1.ORDER_DETAILS OD
JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
WHERE O.CUS_ID = C.CUS_ID
AND OD.PRODUCT_ID = P.PRODUCT_ID
AND TO_CHAR(O.ORDER_DATE, 'DD-MM') = '20-10'
)
);
```

Kết quả:

- Tại chi nhánh 1:

```
SQL> SELECT DISTINCT C.CUS_ID, C.CUS_NAME
2 FROM CHINHANH1.CUSTOMERS C
3 WHERE C.CUS_GENDER = 'Nu'
4 AND NOT EXISTS (
5 SELECT *
6 FROM CHINHANH1.PRODUCTS P
7 WHERE P.PRODUCT_PRICE >= 12000000
8 AND NOT EXISTS (
9 SELECT *
10 FROM CHINHANH1.ORDER_DETAILS OD
11 JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
12 WHERE O.CUS_ID = C.CUS_ID
13 AND OD.PRODUCT_ID = P.PRODUCT_ID
14 AND TO_CHAR(O.ORDER_DATE, 'DD-MM') = '20-10'
15 )
16 );
CUS_ID CUS_NAME
-----
KH113  Tran Thi Mai
```

Hình 13. Kết quả câu truy vấn 6 trên chi nhánh 1

- Tại chi nhánh 2:

```
SQL> SELECT DISTINCT C.CUS_ID, C.CUS_NAME
2 FROM CHINHANH2.CUSTOMERS C
3 WHERE C.CUS_GENDER = 'Nu'
4 AND NOT EXISTS (
5 SELECT *
6 FROM CHINHANH2.PRODUCTS P
7 WHERE P.PRODUCT_PRICE >= 12000000
8 AND NOT EXISTS (
9 SELECT *
10 FROM CHINHANH2.ORDER_DETAILS OD
11 JOIN CHINHANH2.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
12 WHERE O.CUS_ID = C.CUS_ID
13 AND OD.PRODUCT_ID = P.PRODUCT_ID
14 AND TO_CHAR(O.ORDER_DATE, 'DD-MM') = '20-10'
15 )
16 );
CUS_ID CUS_NAME
-----
KH212  Nguyen Thi Ngoc Tram
KH214  Tran Thi Minh Chau
```

Hình 14. Kết quả câu truy vấn 6 trên chi nhánh 2

7) In ra danh sách số lượng nhân viên của 2 chi nhánh

```
SELECT E1.BRANCH_ID AS MaCN, COUNT(E1.EMP_ID) AS SoLuongNhanVien
FROM CHINHANH1.EMPLOYEES E1
WHERE E1.BRANCH_ID = 'CN1'
GROUP BY E1.BRANCH_ID
UNION ALL
SELECT E2.BRANCH_ID AS MaCN, COUNT(E2.EMP_ID) AS SoLuongNhanVien
FROM CHINHANH2.EMPLOYEES@DIRECTOR_LINK E2
WHERE E2.BRANCH_ID = 'CN2'
GROUP BY E2.BRANCH_ID;
```

Kết quả:

```
SQL> SELECT E1.BRANCH_ID AS MaCN, COUNT(E1.EMP_ID) AS SoLuongNhanVien
2 FROM CHINHANH1.EMPLOYEES E1
3 WHERE E1.BRANCH_ID = 'CN1'
4 GROUP BY E1.BRANCH_ID
5 UNION ALL
6 SELECT E2.BRANCH_ID AS MaCN, COUNT(E2.EMP_ID) AS SoLuongNhanVien
7 FROM CHINHANH2.EMPLOYEES@DIRECTOR_LINK E2
8 WHERE E2.BRANCH_ID = 'CN2'
9 GROUP BY E2.BRANCH_ID;
```

MaCN	SoLuongNhanVien
CN1	22
CN2	21

Hình 15. Kết quả câu truy vấn 7 trên 2 chi nhánh

8) Liệt kê danh sách các sản phẩm ở CN1 có khách hàng đặt hàng từ ngày 05/01/2023 đến nay và số lượng tồn kho > 0

```
SELECT DISTINCT P.PRODUCT_ID, P.PRODUCT_NAME, WM.QUANTITY, O.ORDER_DATE
FROM CHINHANH1.PRODUCTS P
INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
INNER JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
INNER JOIN CHINHANH1.WAREHOUSE_MANAGER WM ON P.PRODUCT_ID = WM.PRODUCT_ID
WHERE O.ORDER_DATE >= TO_DATE('25-01-2023', 'DD-MM-YYYY')
AND WM.QUANTITY > 0;
```

Kết quả:

- Tại chi nhánh 1:

```
SQL> SELECT DISTINCT P.PRODUCT_ID, P.PRODUCT_NAME, WM.QUANTITY, O.ORDER_DATE
2 FROM CHINHANH1.PRODUCTS P
3 INNER JOIN CHINHANH1.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
4 INNER JOIN CHINHANH1.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
5 INNER JOIN CHINHANH1.WAREHOUSE_MANAGER WM ON P.PRODUCT_ID = WM.PRODUCT_ID
6 WHERE O.ORDER_DATE >= TO_DATE('25-01-2023', 'DD-MM-YYYY') AND WM.QUANTITY > 0;
```

PRODUCT_ID	PRODUCT_NAME	QUANTITY	ORDER_DATE
P1	Tivi LED 4K 55 inch	500	20-OCT-23
P14	Tu Say Quan Ao	480	20-OCT-23

Hình 16. Kết quả câu truy vấn 8 trên chi nhánh 1

- Tại chi nhánh 2:

```
SQL> SELECT DISTINCT P.PRODUCT_ID, P.PRODUCT_NAME, WM.QUANTITY, O.ORDER_DATE
2 FROM CHINHANH2.PRODUCTS P
3 INNER JOIN CHINHANH2.ORDER_DETAILS OD ON P.PRODUCT_ID = OD.PRODUCT_ID
4 INNER JOIN CHINHANH2.ORDERS O ON OD.ORDER_ID = O.ORDER_ID
5 INNER JOIN CHINHANH2.WAREHOUSE_MANAGER WM ON P.PRODUCT_ID = WM.PRODUCT_ID
6 WHERE O.ORDER_DATE >= TO_DATE('25-01-2023', 'DD-MM-YYYY') AND WM.QUANTITY > 0;
```

PRODUCT_ID	PRODUCT_NAME	QUANTITY	ORDER_DATE
P1	Tivi LED 4K 55 inch	50	20-OCT-23
P2	May Lanh Inverter 1.5HP	30	20-OCT-23
P3	Tu Lanh Side by Side	480	20-OCT-23
P20	Quat Dung Cao Cap	290	20-OCT-23

Hình 17. Kết quả câu truy vấn 8 trên chi nhánh 2

9) In ra danh sách top 3 mã sản phẩm có số lượng tồn kho nhiều nhất cả 2 chi nhánh

```
SELECT PRODUCT_ID, PRODUCT_NAME, SUM(TOTAL_QUANTITY) AS TOTAL_QUANTITY
FROM
(
  (
    SELECT WM.PRODUCT_ID, PRODUCT_NAME, SUM(WM.QUANTITY)
    AS TOTAL_QUANTITY
    FROM CHINHANH1.WAREHOUSE_MANAGER WM
    INNER JOIN CHINHANH1.PRODUCTS P ON WM.PRODUCT_ID = P.PRODUCT_ID
    GROUP BY WM.PRODUCT_ID, PRODUCT_NAME
    HAVING SUM(WM.QUANTITY) > 0
    ORDER BY WM.PRODUCT_ID
    FETCH FIRST 3 ROWS ONLY
  )
  UNION ALL
  (
    SELECT WM.PRODUCT_ID, PRODUCT_NAME, SUM(WM.QUANTITY)
    AS TOTAL_QUANTITY
    FROM CHINHANH2.WAREHOUSE_MANAGER@WAREHOUSE_MANAGER_LINK WM
    INNER JOIN CHINHANH2.PRODUCTS@WAREHOUSE_MANAGER_LINK P
    ON WM.PRODUCT_ID = P.PRODUCT_ID
    GROUP BY WM.PRODUCT_ID, PRODUCT_NAME
    HAVING SUM(WM.QUANTITY) > 0
    ORDER BY WM.PRODUCT_ID
    FETCH FIRST 3 ROWS ONLY
  )
)
GROUP BY PRODUCT_ID, PRODUCT_NAME
ORDER BY TOTAL_QUANTITY DESC
FETCH FIRST 3 ROWS ONLY;
```

Kết quả:

```
SQL> SELECT PRODUCT_ID, PRODUCT_NAME, SUM(TOTAL_QUANTITY) AS TOTAL_QUANTITY
2 FROM
3 (
4 (SELECT WM.PRODUCT_ID, PRODUCT_NAME, SUM(WM.QUANTITY) AS TOTAL_QUANTITY
5 FROM CHINHANH1.WAREHOUSE_MANAGER WM
6 INNER JOIN CHINHANH1.PRODUCTS P ON WM.PRODUCT_ID = P.PRODUCT_ID
7 GROUP BY WM.PRODUCT_ID, PRODUCT_NAME
8 HAVING SUM(WM.QUANTITY) > 0
9 ORDER BY WM.PRODUCT_ID
10 FETCH FIRST 3 ROWS ONLY)
11 UNION ALL
12 (SELECT WM.PRODUCT_ID, PRODUCT_NAME, SUM(WM.QUANTITY) AS TOTAL_QUANTITY
13 FROM CHINHANH2.WAREHOUSE_MANAGER@WAREHOUSE_MANAGER_LINK WM
14 INNER JOIN CHINHANH2.PRODUCTS@WAREHOUSE_MANAGER_LINK P ON WM.PRODUCT_ID = P.PRODUCT_ID
15 GROUP BY WM.PRODUCT_ID, PRODUCT_NAME
16 HAVING SUM(WM.QUANTITY) > 0
17 ORDER BY WM.PRODUCT_ID
18 FETCH FIRST 3 ROWS ONLY)
19 )
20 GROUP BY PRODUCT_ID, PRODUCT_NAME
21 ORDER BY TOTAL_QUANTITY DESC
22 FETCH FIRST 3 ROWS ONLY;
```

PRODUCT_ID	PRODUCT_NAME	TOTAL_QUANTITY
P10	May Pha Ca Pha Espresso	560
P1	Tivi LED 4K 55 inch	550
P12	Binh Dun Nuoc Sieu Toc	420

Hình 18. Kết quả câu truy vấn 9 trên cả hai chi nhánh

10) Tính tổng doanh thu của cả hai chi nhánh

```
SELECT O.BRANCH_ID, SUM(O.TOTAL_PRICE) AS TOTAL_PRICE
FROM CHINHANH1.ORDERS O
GROUP BY O.BRANCH_ID
UNION ALL
SELECT O.BRANCH_ID, SUM(O.TOTAL_PRICE) AS TOTAL_PRICE
FROM CHINHANH2.ORDERS@DIRECTOR_LINK O
GROUP BY O.BRANCH_ID;
```

Kết quả:

```
SQL> SELECT O.BRANCH_ID, SUM(O.TOTAL_PRICE) AS TOTAL_PRICE
2 FROM CHINHANH1.ORDERS O
3 GROUP BY O.BRANCH_ID
4 UNION ALL
5 SELECT O.BRANCH_ID, SUM(O.TOTAL_PRICE) AS TOTAL_PRICE
6 FROM CHINHANH2.ORDERS@DIRECTOR_LINK O
7 GROUP BY O.BRANCH_ID;
```

BRANCH_ID	TOTAL_PRICE
CN1	610550000
CN2	735600000

Hình 19. Kết quả câu truy vấn 10 trên cả hai chi nhánh

IV. FUNCTION, PROCEDURE, TRIGGER TRÊN

MÔI TRƯỜNG PHÂN TÁN

A. Procedure

```
SET SERVEROUTPUT ON;
CREATE OR REPLACE PROCEDURE changeEmployeeSalary (empID
    CHINHANH1.EMPLOYEES.EMP_ID%TYPE, sal NUMBER)
AS
    dem NUMBER;
BEGIN
    SELECT COUNT(*) INTO dem
    FROM CHINHANH1.EMPLOYEES Emp1
    WHERE Emp1.EMP_ID = empID;

    IF (dem > 0) THEN
        UPDATE CHINHANH1.EMPLOYEES Emp1
        SET Emp1.EMP_SALARY = sal
        WHERE Emp1.EMP_ID = empID;

    ELSE
        SELECT COUNT(*) INTO dem
        FROM CHINHANH2.EMPLOYEES@DIRECTOR_LINK Emp2
        WHERE Emp2.EMP_ID = empID;

        IF (dem > 0) THEN
            UPDATE CHINHANH2.EMPLOYEES@DIRECTOR_LINK Emp2
            SET EMP_SALARY = sal
            WHERE emp2.EMP_ID = empID;

        ELSE
            DBMS_OUTPUT.PUT_LINE('Employee not found in any branch.');
```

Hình 20. Tạo Procedure changeEmployeeSalary

B. Trigger

Khi có thay đổi trong chi tiết hóa đơn thì tính lại tổng tiền cho hóa đơn.

Bối cảnh: ORDERS, ORDER_DETAILS

Bảng tầm hưởng:

	Insert	Update	Delete
ORDERS	-	-	-
ORDER_DETAILS	+	+(quality, price)	+

Bảng 9. Bảng tầm ảnh hưởng của ràng buộc toàn vẹn

```
CREATE OR REPLACE TRIGGER UPDATE_TOTAL_PRICE
BEFORE INSERT OR UPDATE OR DELETE ON CHINHANH1.ORDER_DETAILS
FOR EACH ROW
DECLARE
    v_total_price NUMBER;
BEGIN
    SELECT NVL(SUM(QUANTITY * PRICE), 0) INTO v_total_price
    FROM CHINHANH1.ORDER_DETAILS
    WHERE ORDER_ID = :NEW.ORDER_ID;

    SELECT NVL(SUM(QUANTITY * PRICE), 0) INTO old_v_total_price
    FROM CHINHANH1.ORDER_DETAILS
    WHERE ORDER_ID = :OLD.ORDER_ID;

    IF INSERTING THEN
        UPDATE CHINHANH1.ORDERS
        SET TOTAL_PRICE = v_total_price + :NEW.PRICE* :NEW.QUANTITY
        WHERE ORDER_ID = :NEW.ORDER_ID;

    ELSIF UPDATING THEN
        UPDATE CHINHANH1.ORDERS
        SET TOTAL_PRICE = v_total_price + :NEW.PRICE* :NEW.QUANTITY
        WHERE ORDER_ID = :NEW.ORDER_ID;

        UPDATE CHINHANH1.ORDERS
        SET TOTAL_PRICE = old_v_total_price - :OLD.PRICE* :OLD.QUANTITY
        WHERE ORDER_ID = :OLD.ORDER_ID;

    ELSE
        UPDATE CHINHANH1.ORDERS
        SET TOTAL_PRICE = v_total_price - :OLD.PRICE* :OLD.QUANTITY
        WHERE ORDER_ID = :NEW.ORDER_ID;

    END IF;
END;
```

Hình 21. Tạo Trigger Update_Total_Price

V. CÁC MỨC CỘ LẬP TRONG MÔI TRƯỜNG PHẦN TẤN

A. Lost update

Mô tả tình huống: Nhân viên 1 đang thay đổi thông tin của khách hàng thì có nhân viên 2 đến thay đổi thông tin cũng của chính khách hàng đó nhưng với dữ liệu khác. Từ đó, dẫn đến việc thông tin của nhân viên 2 ghi đè lên trên thông tin của nhân viên còn lại.

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;
T1	UPDATE CHINHANH1.CUSTOMERS SET CUS_PHONE = '0310101010' WHERE CUS_ID = 'KH105';	
T2		UPDATE CHINHANH1.CUSTOMERS@DIRECTOR_LINK SET CUS_PHONE = '0359297916' WHERE CUS_ID = 'KH105';
T3	COMMIT;	
T4		COMMIT;
T5	SELECT CUS_PHONE FROM CHINHANH1.CUSTOMERS WHERE CUS_ID = 'KH105';	
Kết quả	0359297916 CUS_PHONE đã bị ghi đè	

Cách khắc phục:

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE

T1	UPDATE CHINHANH1.CUSTOMERS SET CUS_PHONE = '0310101010' WHERE CUS_ID = 'KH105';	
T2		UPDATE CHINHANH1.CUSTOMERS@DIRECTOR_LINK SET CUS_PHONE = '0359297916' WHERE CUS_ID = 'KH105';
T3	COMMIT;	
T4		COMMIT;
T5	SELECT CUS_PHONE FROM CHINHANH1.CUSTOMERS WHERE CUS_ID = 'KH105';	
Kết quả	0359297916 CUS_PHONE đã bị ghi đè	Error report - ORA-08177: can't serialize access for this transaction ORA-02063: preceding line from OFF1

B. Non-repeatable

Mô tả tình huống: Giám đốc CN1 xem thông tin nhân viên lần 1 hoàn tất thì giám đốc CN2 truy cập vào hệ thống để thay đổi thông tin khách hàng. Sau đó, giám đốc CN1 quay lại để kiểm tra thông tin thì nhận thấy có sự thay đổi so với lần xem đầu tiên.

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;
T1	SELECT * FROM CHINHANH1.EMPLOYEES WHERE EMP_ID = 'NV122';	
T2		UPDATE CHINHANH1.EMPLOYEES@DIRECTOR_LINK SET EMP_SALARY =

		25000000 WHERE EMP_ID = 'NV122';
T3		COMMIT;
T4	SELECT * FROM CHINHANH1.EMPL OYEES WHERE EMP_ID = 'NV122';	SELECT * FROM CHINHANH1.EMPLO YEES@ DIRECTOR_LINK WHERE EMP_ID = 'NV122';
Kết quả	25000000 Kết quả truy xuất khác với ban đầu	25000000

Cách khắc phục:

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE
T1	SELECT * FROM CHINHANH1.EMPL OYEES WHERE EMP_ID = 'NV122';	
T2		UPDATE CHINHANH1.EMPLO YEES@ DIRECTOR_LINK SET EMP_SALARY = 30000000 WHERE EMP_ID = 'NV122';
T3		COMMIT;
T4	SELECT * FROM CHINHANH1.EMPL OYEES WHERE EMP_ID = 'NV122';	SELECT * FROM CHINHANH1.EMPLO YEES@ DIRECTOR_LINK WHERE EMP_ID = 'NV122';
T5	COMMIT;	
T6	SELECT * FROM CHINHANH1.EMPL OYEES WHERE EMP_ID = 'NV122';	
Kết quả	30000000	30000000

C. Deadlock

Mô tả tình huống: Nhân viên 1 thay đổi thông tin trên sản phẩm A trong lúc đó nhân viên 2 thay đổi thông tin trên sản phẩm B. Sau đó, nhân viên 1 chuyển sang thay đổi thông tin trên sản phẩm B còn nhân viên 2 cũng làm ngược lại.

Tim e	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;
T1	UPDATE CHINHANH1.PROD UCTS SET PRODUCT_PRICE = 10000000 WHERE PRODUCT_ID = 'P2';	
T2		UPDATE CHINHANH1.PRODU CTS@ DIRECTOR_LINK SET PRODUCT_PRICE = 7000000 WHERE PRODUCT_ID = 'P5';
T3	UPDATE CHINHANH1.PROD UCTS SET PRODUCT_PRICE = 5000000 WHERE PRODUCT_ID = 'P5';	
T4		UPDATE CHINHANH1.PRODU CTS@ DIRECTOR_LINK SET PRODUCT_PRICE = 11000000 WHERE PRODUCT_ID = 'P2';
T5	DEADLOCK	
T6	COMMIT;	
T7		COMMIT;
T8	SELECT * FROM CHINHANH1.PROD UCTS WHERE PRODUCT_ID = 'P2' OR PRODUCT_ID = 'P5';	

Kết quả	ERROR at line 2: ORA-00060: deadlock detected while waiting for resource	ORA-08177: can't serialize access for this transaction ORA-02063: preceding line from DIRECTOR_LINK
---------	--	---

Cách khắc phục: Với vấn đề Deadlock, hệ quản trị cơ sở dữ liệu có cơ chế phát hiện và xử lý.

D. Phantom read

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;	ALTER SESSION SET ISOLATION_LEVEL = READ COMMITTED;
T1	SELECT COUNT(*) FROM CHINHANH1.EMPLOYEES; COUNT(*)	
T2		DELETE FROM CHINHANH1.EMPLOYEES@DIRECTOR_LINK WHERE EMP_ID = 'NV123';
T3		COMMIT;
T4	SELECT COUNT(*) FROM CHINHANH1.EMPLOYEES;	
T5	COUNT(*) 22	

Cách khắc phục:

Ti me	Chi nhánh 1	Chi nhánh2
T0	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;	ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;
T1	SELECT COUNT(*) FROM CHINHANH1.EMPLOYEES; COUNT(*) 23	

T2		DELETE FROM CHINHANH1.EMPLOYEES@DIRECTOR_LINK WHERE EMP_ID = 'NV123';
T3		COMMIT;
T4	SELECT COUNT(*) FROM CHINHANH1.EMPLOYEES; COUNT(*) 23	
T5	COMMIT;	
T6	SELECT COUNT(*) FROM CHINHANH1.EMPLOYEES;	
Kết quả	COUNT(*) 22	

VI. THỰC HIỆN TỐI ƯU HÓA TRUY VẤN TRÊN MÔI TRƯỜNG PHÂN TÁN

1) Câu truy vấn đơn giản chưa tối ưu

Cho biết thông tin những sản phẩm (PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS) ở chi nhánh “TP Hồ Chí Minh” có ngày bán ra là 10/2023 và có số lượng bán lớn hơn 2

```
SELECT DISTINCT P.PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
FROM CHINHANH1.PRODUCTS P, CHINHANH1.WAREHOUSE_SALER WS,
CHINHANH1.ORDERS O, CHINHANH1.ORDER_DETAILS OD, CHINHANH1.BRANCHES B
WHERE B.BRANCH_ID = WS.BRANCH_ID
AND WS.PRODUCT_ID = P.PRODUCT_ID
AND P.PRODUCT_ID = OD.PRODUCT_ID
AND OD.ORDER_ID = O.ORDER_ID
AND B.BRANCH_NAME = 'Chi nhánh 1'
AND EXTRACT (YEAR FROM ORDER_DATE) = 2023
AND EXTRACT (MONTH FROM ORDER_DATE) = 10
AND QUANTITY > 2;
```

Hình 22. Câu truy vấn chưa tối ưu

2) Explain query câu truy vấn đơn giản chưa tối ưu

```
SELECT /*+ GATHER_PLAN_STATISTICS*/
DISTINCT P.PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
FROM CHINHANH1.PRODUCTS P, CHINHANH1.WAREHOUSE_SALER WS,
CHINHANH1.ORDERS O, CHINHANH1.ORDER_DETAILS OD, CHINHANH1.BRANCHES B
WHERE B.BRANCH_ID = WS.BRANCH_ID
AND WS.PRODUCT_ID = P.PRODUCT_ID
AND P.PRODUCT_ID = OD.PRODUCT_ID
AND OD.ORDER_ID = O.ORDER_ID
AND B.BRANCH_NAME = 'Chi nhánh 1'
AND EXTRACT (YEAR FROM ORDER_DATE) = 2023
AND EXTRACT (MONTH FROM ORDER_DATE) = 10
AND QUANTITY > 2;
SELECT *
FROM TABLE (DBMS_XPLAN.display_cursor(format => 'ALLSTATS LAST'));
```

Hình 23. Explain query câu truy vấn đơn giản chưa tối ưu

```
SQL> SELECT /*+ GATHER_PLAN_STATISTICS*/
2 DISTINCT P.PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
3 FROM CHINHANH1.PRODUCTS P, CHINHANH1.WAREHOUSE_SALER WS,
4 CHINHANH1.ORDERS O, CHINHANH1.ORDER_DETAILS OD, CHINHANH1.BRANCHES B
5 WHERE B.BRANCH_ID = WS.BRANCH_ID
6 AND WS.PRODUCT_ID = P.PRODUCT_ID
7 AND P.PRODUCT_ID = OD.PRODUCT_ID
8 AND OD.ORDER_ID = O.ORDER_ID
9 AND B.BRANCH_NAME = 'Chi nhánh 1'
10 AND EXTRACT (YEAR FROM ORDER_DATE) = 2023
11 AND EXTRACT (MONTH FROM ORDER_DATE) = 10
12 AND QUANTITY > 2;

PRODUCT_ID PRODUCT_NAME PRODUCT_STATUS
-----
P14 Tu Say Quan Ao 1

SQL> SELECT *
2 FROM TABLE (DBMS_XPLAN.display_cursor(format => 'ALLSTATS LAST'));
```

```
PLAN_TABLE_OUTPUT
SQL_ID 9fkavq6824uw5, child number 0
SELECT /*+ GATHER_PLAN_STATISTICS*/ DISTINCT P.PRODUCT_ID,
PRODUCT_NAME, PRODUCT_STATUS FROM CHINHANH1.PRODUCTS P,
CHINHANH1.WAREHOUSE_SALER WS, CHINHANH1.ORDERS O,
CHINHANH1.ORDER_DETAILS OD, CHINHANH1.BRANCHES B WHERE B.BRANCH_ID =
WS.BRANCH_ID AND WS.PRODUCT_ID = P.PRODUCT_ID AND P.PRODUCT_ID =
OD.PRODUCT_ID AND OD.ORDER_ID = O.ORDER_ID AND B.BRANCH_NAME
= 'Chi nhánh 1' AND EXTRACT (YEAR FROM ORDER_DATE) = 2023 AND
EXTRACT (MONTH FROM ORDER_DATE) = 10 AND QUANTITY > 2

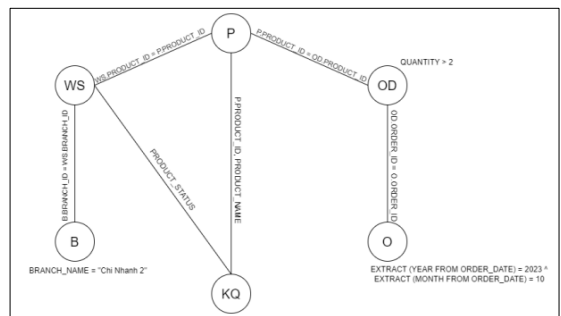
PLAN_TABLE_OUTPUT
Plan hash value: 2163768099

-----
| Id | Operation | Name | Starts | E-Rows | A
--
| Rows | A-Time | Buffers | OMem | 1Mem | Used-Mem |
-----
PLAN_TABLE_OUTPUT
| 0 | SELECT STATEMENT | | | 1 | |
| 1 | 00:00:00.01 | 26 | | | |
| 1 | HASH UNIQUE | | | 1 | 1 |
| 1 | 00:00:00.01 | 26 | 881K | 881K |
|* 2 | HASH JOIN | | | 1 | 1 |
| 1 | 00:00:00.01 | 26 | 1572K | 1301K (0) |
|* 3 | HASH JOIN SEMI | | | 1 | 1 |
| 20 | 00:00:00.01 | 20 | 1538K | 1538K | 454K (0) |
```

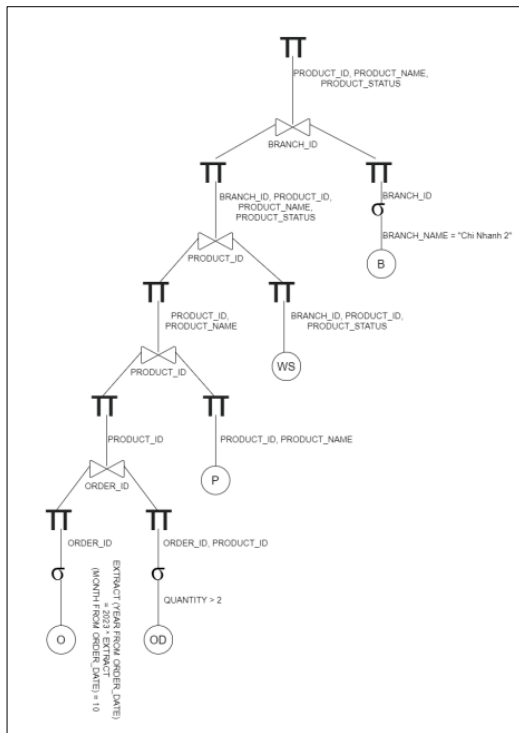
```
PLAN_TABLE_OUTPUT
| 4 | MERGE JOIN CARTESIAN | | | 1 | 1 |
| 20 | 00:00:00.01 | 14 | | | |
| 5 | MERGE JOIN | | | 1 | 1 |
| 1 | 00:00:00.01 | 8 | | | |
|* 6 | TABLE ACCESS BY INDEX ROWID | ORDERS | 1 | 1 |
| 2 | 00:00:00.01 | 2 | | | |
| 7 | INDEX FULL SCAN | SYS_C009999 | 1 | 22 |
PLAN_TABLE_OUTPUT
20 | 00:00:00.01 | 1 | | | |
|* 8 | SORT JOIN | 6 | 2048 | 2048 | 2048 (0) | 2 | 23 |
| 1 | 00:00:00.01 | 6 | | | |
|* 9 | TABLE ACCESS FULL | ORDER_DETAILS | 1 | 23 |
| 23 | 00:00:00.01 | 6 | | | |
| 10 | BUFFER SORT | | 1 | 20 |
| 20 | 00:00:00.01 | 6 | 73728 | 73728 |
PLAN_TABLE_OUTPUT
| 11 | TABLE ACCESS FULL | WAREHOUSE_SALER | 1 | 20 | |
| 20 | 00:00:00.01 | 6 | | | |
|* 12 | TABLE ACCESS FULL | BRANCHES | 1 | 1 |
| 1 | 00:00:00.01 | 6 | | | |
| 13 | TABLE ACCESS FULL | PRODUCTS | 1 | 1 |
| 20 | 00:00:00.01 | 6 | | | |
PLAN_TABLE_OUTPUT
Predicate Information (identified by operation id):
2 - access("WS"."PRODUCT_ID"="P"."PRODUCT_ID" AND "P"."PRODUCT_ID"="OD"."PROD
UCT_ID")
3 - access("B"."BRANCH_ID"="WS"."BRANCH_ID")
6 - filter(((EXTRACT(YEAR FROM INTERNAL_FUNCTION("ORDER_DATE"))=2023 AND EXTRA
CT(MONTH FROM
PLAN_TABLE_OUTPUT
INTERNAL_FUNCTION("ORDER_DATE"))=10))
8 - access("OD"."ORDER_ID"="O"."ORDER_ID")
filter("OD"."ORDER_ID"="O"."ORDER_ID")
9 - filter("QUANTITY">2)
12 - filter("B"."BRANCH_NAME"='Chi nhánh 1')
Note
- this is an adaptive plan
48 rows selected.
```

Hình 24. Kết quả Explain câu truy vấn đơn giản chưa tối ưu

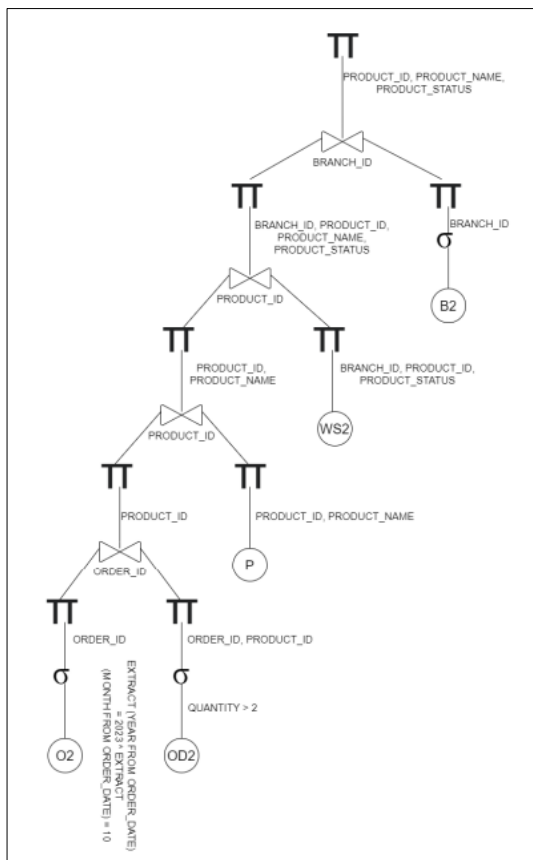
3) Tối ưu hóa câu truy vấn cục bộ, phân tán



Hình 25. Đồ thị truy vấn



Hình 26. Câu truy vấn tối ưu trên môi trường tập trung



Hình 27. Câu truy vấn tối ưu trên môi trường phân tán

4) Viết lại câu Query trên môi trường phân tán

```
SELECT DISTINCT PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
FROM
((
  SELECT BRANCH_ID, E.PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
  FROM
  ((
    SELECT C.PRODUCT_ID, PRODUCT_NAME
    FROM
    ((
      SELECT PRODUCT_ID
      FROM
      ((
        SELECT ORDER_ID
        FROM CHINHANH1.ORDERS
        WHERE EXTRACT(YEAR FROM ORDER_DATE) = 2023
        AND EXTRACT(MONTH FROM ORDER_DATE) = 10
      ) A INNER JOIN (
        SELECT ORDER_ID, PRODUCT_ID
        FROM CHINHANH1.ORDER_DETAILS
        WHERE QUANTITY > 2
      ) B ON A.ORDER_ID = B.ORDER_ID
    ) C INNER JOIN (
      SELECT PRODUCT_ID, PRODUCT_NAME
      FROM CHINHANH1.PRODUCTS
    ) D ON C.PRODUCT_ID = D.PRODUCT_ID
    ) E INNER JOIN (
      SELECT BRANCH_ID, PRODUCT_ID, PRODUCT_STATUS
      FROM CHINHANH1.WAREHOUSE_SALER
    ) F ON E.PRODUCT_ID = F.PRODUCT_ID
    ) G INNER JOIN (
      SELECT BRANCH_ID
      FROM CHINHANH1.BRANCHES
      WHERE BRANCH_NAME = 'Chi nhánh 1'
    ) H ON G.BRANCH_ID = H.BRANCH_ID);
  )
```

Hình 28. Câu truy vấn sau khi được viết lại trên môi trường phân tán

5) Explain query trên môi trường phân tán

```
SELECT /*+ GATHER_PLAN_STATISTICS */
DISTINCT PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
FROM
((
  SELECT BRANCH_ID, E.PRODUCT_ID, PRODUCT_NAME, PRODUCT_STATUS
  FROM
  ((
    SELECT C.PRODUCT_ID, PRODUCT_NAME
    FROM
    ((
      SELECT PRODUCT_ID
      FROM
      ((
        SELECT ORDER_ID
        FROM CHINHANH1.ORDERS
        WHERE EXTRACT(YEAR FROM ORDER_DATE) = 2023
        AND EXTRACT(MONTH FROM ORDER_DATE) = 10
      ) A INNER JOIN (
        SELECT ORDER_ID, PRODUCT_ID
        FROM CHINHANH1.ORDER_DETAILS
        WHERE QUANTITY > 2
      ) B ON A.ORDER_ID = B.ORDER_ID
    ) C INNER JOIN (
      SELECT PRODUCT_ID, PRODUCT_NAME
      FROM CHINHANH1.PRODUCTS
    ) D ON C.PRODUCT_ID = D.PRODUCT_ID
    ) E INNER JOIN (
      SELECT BRANCH_ID, PRODUCT_ID, PRODUCT_STATUS
      FROM CHINHANH1.WAREHOUSE_SALER
    ) F ON E.PRODUCT_ID = F.PRODUCT_ID
    ) G INNER JOIN (
      SELECT BRANCH_ID
      FROM CHINHANH1.BRANCHES
      WHERE BRANCH_NAME = 'Chi nhánh 1'
    ) H ON G.BRANCH_ID = H.BRANCH_ID);
  )
```

Hình 29. Explain query sau khi được viết lại trên môi trường phân tán

VII. LỜI CẢM ƠN

Đầu tiên, nhóm chúng em xin gửi lời cảm ơn và lòng biết ơn sâu sắc nhất tới giảng viên Thái Bảo Trân và giảng viên Nguyễn Minh Nhựt – những người đã giảng dạy và chia sẻ rất nhiều kiến thức cũng như các ví dụ thực tiễn trong các bài giảng. Thầy đã hướng dẫn cho chúng em làm bài tập, sửa chữa và đóng góp nhiều ý kiến quý báu giúp chúng em hoàn thành tốt báo cáo môn học của mình.

Bộ môn Cơ sở dữ liệu phân tán là môn học thú vị, vô cùng bổ ích và có tính thực tế cao. Tuy nhiên, do vốn kiến thức chuyên môn còn nhiều hạn chế và khả năng tiếp thu thực tế còn nhiều ngỡ ngàng. Mặc dù chúng em đã cố gắng hết sức nhưng chắc chắn bài báo cáo khó có thể tránh khỏi những thiếu sót và nhiều chỗ còn chưa chính xác, chúng em rất mong nhận được sự góp ý, chỉ bảo thêm của Thầy nhằm hoàn thiện những kiến thức của mình để nhóm chúng em có thể dùng làm hành trang thực hiện tiếp các đề tài khác trong tương lai cũng như là trong học tập và làm việc sau này.

Một lần nữa, nhóm xin gửi đến thầy, bạn bè lời cảm ơn đặc biệt chân thành và tốt đẹp nhất.

TÀI LIỆU THAM KHẢO

[1] Các file báo cáo BTL1 của anh chị khóa trên do giảng viên Nguyễn Minh Nhựt cung cấp

[2] Tài liệu thực hành: **“Database Link Virtual Machine”** và **“Database Radmin VPN Real Enviroment”** do giảng viên Nguyễn Minh Nhựt cung cấp

[3] Tài liệu tham khảo **“PROCEDURE đơn giản tập trung”** do giảng viên Nguyễn Minh Nhựt cung cấp

[4] Các bài giảng lý thuyết về môn Cơ sở dữ liệu phân tán do giảng viên Thái Bảo Trân cung cấp

[5] Tài liệu tham khảo **“EXPLAIN QUERY ORACLE”** do giảng viên Nguyễn Minh Nhựt cung cấp

```
SOL> SELECT /*+ GATHER_PLAN_STATISTICS */
2  DISTINCT PRODUCT_ID, PRODUCT_NAME, PRODUCT STATUS
3  FROM
4  ((
5  SELECT BRANCH_ID, E.PRODUCT_ID, PRODUCT_NAME, PRODUCT STATUS
6  FROM
7  ((
8  SELECT C.PRODUCT_ID, PRODUCT NAME
9  FROM
10  ((
11  SELECT PRODUCT_ID
12  FROM
13  ((
14  SELECT ORDER_ID
15  FROM CHINHANH1.ORDERS
16  WHERE EXTRACT(YEAR FROM ORDER_DATE) = 2023
17  AND EXTRACT(MONTH FROM ORDER_DATE) = 10
18  ) A INNER JOIN (
19  SELECT ORDER_ID, PRODUCT_ID
20  FROM CHINHANH1.ORDER_DETAILS
21  WHERE QUANTITY > 2
22  ) B ON A.ORDER_ID = B.ORDER_ID)
23  ) C INNER JOIN (
24  SELECT PRODUCT_ID, PRODUCT NAME
25  FROM CHINHANH1.PRODUCTS
26  ) D ON C.PRODUCT_ID = D.PRODUCT_ID)
27  ) E INNER JOIN (
28  SELECT BRANCH_ID, PRODUCT_ID, PRODUCT STATUS
29  FROM CHINHANH1.WAREHOUSE_SALER
30  ) F ON E.PRODUCT_ID = F.PRODUCT_ID)
31  ) G INNER JOIN (
32  SELECT BRANCH_ID FROM CHINHANH1.BRANCHES
33  WHERE BRANCH_NAME = 'Chi nhánh 1'
34  ) H ON G.BRANCH_ID = H.BRANCH_ID);

PRODUCT_ID PRODUCT NAME PRODUCT STATUS
-----
P14 Tu Say Quan Ao 1

SQL> SELECT *
2  FROM TABLE(DBMS_XPLAN.display_cursor(format=>'ALLSTATS LAST'));

PLAN_TABLE_OUTPUT
-----
SQL_ID a9auzmy2y6qql, child number 0
SELECT /*+ GATHER_PLAN_STATISTICS */
DISTINCT PRODUCT_ID,
PRODUCT NAME, PRODUCT STATUS FROM (( SELECT BRANCH_ID,
E.PRODUCT_ID, SELECT C.PRODUCT_ID, PRODUCT NAME FROM ((
SELECT PRODUCT_ID FROM
SELECT ORDER_ID FROM CHINHANH1.ORDERS WHERE EXTRACT(YEAR
FROM ORDER_DATE) = 2023 AND EXTRACT(MONTH FROM ORDER_DATE) = 10 ) A
INNER JOIN ( SELECT ORDER_ID, PRODUCT_ID FROM CHINHANH1.ORDER_DETAILS
WHERE QUANTITY > 2 ) B ON A.ORDER_ID = B.ORDER_ID ) C INNER
JOIN ( SELECT PRODUCT FROM CHINHANH1.PRODUCTS ) D ON C.PRODUCT_ID = D.PRODUCT_ID ) E INNER JOIN (
SELECT BRANCH_ID, PRODUCT_ID, PRODUCT STATUS FROM CHINHANH1.WAREHOUSE_SALER ) F ON E.PRODUCT_ID = F.PRODUCT_ID ) G INNER JOIN (
SELECT BRANCH_ID FROM CHINHANH1.BRANCHES WHERE BRANCH_NAME = 'Chi nhánh 1' ) H ON G.BRANCH_ID = H.BRANCH_ID

Plan hash value: 623686595

-----
PLAN_TABLE_OUTPUT
-----

| Id | Operation | Name | Starts | E-Rows |
|----|-----|-----|-----|-----|
| 0 | SELECT STATEMENT | | 1 | 1 |
| 1 | HASH UNIQUE | | 1 | 1 |

PLAN_TABLE_OUTPUT
-----
| 1 | 00:00:00.01 | 18 | 881K | 881K | |
| 2 | NESTED LOOPS SEMI | | 1 | 1 |
| 1 | 00:00:00.01 | 18 | | | |
| 3 | HASH JOIN | | 1 | 1 |
| 1 | 00:00:00.01 | 16 | 1399K | 1399K | 411K (0) |
| 4 | NESTED LOOPS | | 1 | 1 |
| 1 | 00:00:00.01 | 10 | | | |

PLAN_TABLE_OUTPUT
-----
| 5 | NESTED LOOPS | | 1 | 1 | |
| 1 | 00:00:00.01 | 9 | | | |
| 6 | MERGE JOIN | | 1 | 1 |
| 1 | 00:00:00.01 | 8 | | | |
| 7 | TABLE ACCESS BY INDEX ROWID | ORDERS | 1 | 1 |
| 2 | 00:00:00.01 | 2 | | | |
| 8 | INDEX FULL SCAN | SYS_C009999 | 1 | 22 |
| 20 | 00:00:00.01 | 1 | | | |

PLAN_TABLE_OUTPUT
-----
| 9 | SORT JOIN | | 2 | 23 | |
| 1 | 00:00:00.01 | 6 | 2048 | 2048 | 2048 (0) |
| 10 | TABLE ACCESS FULL | ORDER_DETAILS | 1 | 23 |
| 23 | 00:00:00.01 | 6 | | | |
| 11 | INDEX UNIQUE SCAN | SYS_C009977 | 1 | 1 |
| 1 | 00:00:00.01 | 1 | | | |
| 12 | TABLE ACCESS BY INDEX ROWID | PRODUCTS | 1 | 1 |

PLAN_TABLE_OUTPUT
-----
| 13 | TABLE ACCESS FULL | WAREHOUSE_SALER | 1 | 20 | |
| 20 | 00:00:00.01 | 6 | | | |
| 14 | TABLE ACCESS BY INDEX ROWID | BRANCHES | 1 | 1 |
| 1 | 00:00:00.01 | 2 | | | |
| 15 | INDEX UNIQUE SCAN | SYS_C009971 | 1 | 1 |
| 1 | 00:00:00.01 | 1 | | | |

PLAN_TABLE_OUTPUT
-----

Predicate Information (identified by operation id):
-----
3 - access("PRODUCT_ID"="PRODUCT_ID")
7 - filter((EXTRACT(YEAR FROM INTERNAL_FUNCTION("ORDER_DATE"))=2023 AND EXTRA
CT(MONTH FROM
INTERNAL_FUNCTION("ORDER_DATE"))=10))
9 - access("ORDER_ID"="ORDER_ID")
filter("ORDER_ID"="ORDER_ID")
10 - filter("QUANTITY">2)
11 - access("PRODUCT_ID"="PRODUCT_ID")
14 - filter("BRANCH_NAME"='Chi nhánh 1')
15 - access("BRANCH_ID"="BRANCH_ID")

Note
-----
- this is an adaptive plan

PLAN_TABLE_OUTPUT
-----

58 rows selected.
```

Hình 30: Kết quả truy vấn Explain câu truy vấn trên

