Ravendb

line 1: 1st Trần Thanh Trung  
line 2: *dept. name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 4th Trần Thạnh Phong  
line 2: *dept. name of organization*  
*(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCIDline 1: 2nd Nguyễn Thành Đạt  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

lline 1: 3rd Nguyễn Trung Nguyên  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

Dự án khởi tạo và thiết lập một hệ cơ sở dữ liệu phân tán lưu trữ thông tin của một cơ sở kinh doanh giày với các thao tác trên cơ sỡ dữ liệu như truy vấn, thiết lập các mức cô lập, xử lý các vấn đề liên quan đến khả năng đồng bộ trong vấn đề truy vấn phân tán. Đồng thời tìm và đưa ra phương pháp để tối ưu một câu truy vấn dựa trên cơ sở dữ liệu được tạo.

# Thiết kế CSDL phân tán.

## DDL & DML

Tạo các bảng thông tin chứa dữ liệu cần thiết để xây dựng hệ cơ sở dữ liệu của một cửa hàng bán giày như :GIAY, CHINHANH, KHACHHANG, NHANVIEN, QLKHO, QLBH, HOADON, CTHD. Các bảng được tạo đồng thời tại cả hai chi nhánh dưới hình thức giống nhau.

**Tạo bảng GIAY**

Graphical user interface, text, application, email

Description automatically generated

**Tạo bảng CHINHANH**

**Graphical user interface

Description automatically generated with low confidence**

**Tạo bảng KHACHHANG**

**Text

Description automatically generated**

**Tạo bảng NHANVIEN**

**Graphical user interface, text, application, email

Description automatically generated**

**Tạo bảng G\_QLKHO**

**Graphical user interface, text, application

Description automatically generated**

**Tạo bảng G\_QLBH**

**Graphical user interface, text, application, email

Description automatically generated**

**Tạo bảng HOADON**

**Graphical user interface, text, application

Description automatically generated**

**Tạo bảng CTHD**

Graphical user interface, text, application

Description automatically generated

Sau đó, tiến hành viết và cài đặt các TRIGGER như tạo CTHD, Update CTHD, Update tình trạng hàng. Bên cạnh đó tiến hành cài đặt các FUNCTION và Procedure cần thiết.

Để hoàn thành cơ sở dữ liệu, tiến hành INSERT các giá trị vào hai chi nhánh.

## Thực hiện 10 câu truy vấn.

Trên cơ sỡ dữ liệu đã tạo, tiến hành 10 câu truy vấn để kiểm tra tính hoàn thiện của dữ liệu.

Các câu truy vấn được thực hiện bao gồm:

**Câu 1: Đưa ra thông tin tất cả giày tại cả 2 chi nhánh với số lượng dưới 10 sắp xếp giảm dần.**

**SELECT** G. \*, QLK.SOLUONG  
**FROM** CN1.GIAY G **JOIN**(**SELECT** \* **FROM** CN1.G\_QLKHO  
**UNION  
SELECT** \* **FROM** CN2.G\_QLKHO@DBL\_CN2) QLK  
**ON** G.MAGIAY = QLK.MAGIAY  
**WHERE** QLK.SOLUONG < 10  
**ORDER BY** QLK.SOLUONG **DESC**;

**Kết quả trả về:**

Table

Description automatically generated

**Câu 2: Tìm những mẫu giày với tình trạng là “Con hang” và có số lượng lớn hơn 20 còn lại trong kho ở cả 02 chi nhánh.**

**SELECT** G.\*  
**FROM** CN1.GIAY G **JOIN**(**SELECT** \*  
**FROM** CN1.G\_QLKHO  
**UNION  
SELECT** \*  
**FROM** CN2.G\_QLKHO@DBL\_CN2) QLK

**ON** G.MAGIAY = QLK.MAGIAY  
**WHERE** QLK.SOLUONG > 20  
**AND** QLK.MAGIAY **IN** (**SELECT** QLBH.MAGIAY  
**FROM** CN1.GIAY G **JOIN**(**SELECT** \*  
**FROM** CN1.G\_QLBH  
**UNION  
SELECT** \*  
**FROM** CN2.G\_QLBH@DBL\_CN2) QLBH  
**ON** G.MAGIAY = QLBH.MAGIAY  
**WHERE** QLBH.TINHTRANG = 'Con  
hang');

**Kết quả trả về:**

Graphical user interface, application, table, Excel

Description automatically generated

**CÂU 3: Tìm những mẫu giày được phân phố tại chi nhánh 2 nhưng không có tại chi nhánh 1 với tình trạng “Con hang”.**

**SELECT** G.\*  
**FROM** CN1.GIAY G **JOIN** CN2.G\_QLBH@DBL\_CN2 QLBH  
**ON** G.MAGIAY = QLBH.MAGIAY  
**WHERE** QLBH.TINHTRANG = 'Con hang'  
**MINUS  
SELECT** G.\*  
**FROM** CN1.GIAY G **JOIN** CN1.G\_QLBH QLBH  
**ON** G.MAGIAY = QLBH.MAGIAY

**WHERE** QLBH.TINHTRANG = 'Con hang';

**Kết quả trả về:**

Table

Description automatically generated

**CÂU 4: Tìm khách hàng mua hàng nhiều nhất trong tháng 10 tại chi nhánh 1.**

**SELECT** HD.MAKH, **COUNT**(**DISTINCT** CT.MAGIAY)  
**FROM** CN1.HOADON HD **JOIN** CN1.CTHD CT  
**ON** HD.MAHD = CT.MAHD  
**WHERE TO\_CHAR**(HD.NGAYHD, 'MM') = 10  
**GROUP BY** HD.MAKH  
**HAVING COUNT**(**DISTINCT** CT.MAGIAY) >= **ALL**(**SELECT COUNT**(**DISTINCT** CT.MAGIAY)  
**FROM** CN1.HOADON HD **JOIN** CN1.CTHD CT  
**ON** HD.MAHD = CT.MAHD  
**WHERE TO\_CHAR**(HD.NGAYHD, 'MM') = 10  
**GROUP BY** HD.MAKH);

**Kết quả trả về:**

Text

Description automatically generated

**CÂU 5: Tìm nhân viên ở chi nhánh 2 có số lượng hóa đơn nhiều nhất trong cả hai chi nhánh.**

**SELECT** HD.MANV, **COUNT**(HD.MAHD)  
**FROM**(**SELECT** \*  
**FROM** CN1.NHANVIEN  
**UNION  
SELECT** \*  
**FROM** CN2.NHANVIEN@DBL\_CN2) NV  
**JOIN**(**SELECT** \*  
**FROM** CN1.HOADON  
**UNION  
SELECT** \*  
**FROM** CN2.HOADON@DBL\_CN2) HD  
**ON** NV.MANV = HD.MANV  
**GROUP BY** HD.MANV  
**HAVING COUNT**(HD.MAHD) >= **ALL**(**SELECT COUNT**(HD.MAHD)  
**FROM**(**SELECT** \*  
**FROM** CN1.NHANVIEN  
**UNION  
SELECT** \*  
**FROM** CN2.NHANVIEN@DBL\_CN2) NV  
**JOIN**(**SELECT** \*  
**FROM** CN1.HOADON  
**UNION  
SELECT** \*  
**FROM** CN2.HOADON@DBL\_CN2) HD  
**ON** NV.MANV = HD.MANV  
**GROUP BY** HD.MANV);

**Kết quả trả về:**

**A picture containing calendar

Description automatically generated**

**CÂU 6: Với mỗi màu sắc giày, liệt kê số lượng mua của khách hành trên 18 tuổi.**

**SELECT** MAUSAC, **SUM**(SOLUONG)  
**FROM** GIAY G,  
(**SELECT** \*  
**FROM** CTHD, HOADON HD  
**WHERE** CTHD.MAHD=HD.MAHD  
**UNION  
SELECT** \*  
**FROM** CN1.CTHD@DBL\_CN1 CTHD, CN1.HOADON@DBL\_CN1 HD  
**WHERE** CTHD.MAHD=HD.MAHD) HD  
**WHERE** G.MAGIAY=HD.MAGIAY  
**AND** HD.MAKH **IN** (**SELECT** MAKH  
**FROM** (**SELECT** MAKH, TUOI  
**FROM** KHACHHANG  
**UNION  
SELECT** MAKH, TUOI  
**FROM** CN1.KHACHHANG@DBL\_CN1)  
**WHERE** TUOI **BETWEEN** 18 **AND** 25)  
**GROUP BY** MAUSAC;

**Kết quả trả về:**

Table

Description automatically generated

**Câu 7: Tìm các hóa đơn mua hàng trong tháng 7 với tổng số lượng giày có trên 5 sản phẩm, sắp xếp theo thứ tự giảm dần ở cả 02 chi nhánh.**

**SELECT** A.MAHD, A.TSL  
**FROM**(**SELECT** HD.MAHD, **SUM**(HD.SOLUONG) TSL  
**FROM**(**SELECT** CTHD.MAHD, NGAYHD, MAGIAY, SOLUONG  
**FROM** CN1.HOADON@DBL\_CN1 HD,CN1.CTHD@DBL\_CN1 CTHD  
**WHERE** HD.MAHD=CTHD.MAHD  
**UNION  
SELECT** CTHD.MAHD, NGAYHD, MAGIAY, SOLUONG  
**FROM** CN2.HOADON HD, CN2.CTHD CTHD  
**WHERE** HD.MAHD=CTHD.MAHD) HD  
**WHERE TO\_CHAR**(HD.NGAYHD, 'MM') = 7  
**GROUP BY** HD.MAHD  
**HAVING SUM**(HD.SOLUONG) > 5) A  
**ORDER BY** A.TSL **DESC**;

**Kết quả trả về:**

Table

Description automatically generated

**CÂU 8: Tìm mẫu giày có số lượng bán ít nhất trong năm 2021 ở cả 02 chi nhánh.**

**SELECT** MAGIAY, **SUM**(TSL)  
**FROM**(**SELECT** MAGIAY,**SUM**(SOLUONG) TSL  
**FROM** CN1.CTHD@DBL\_CN1 CT1, CN1.HOADON@DBL\_CN1 HD1  
**WHERE** CT1.MAHD=HD1.MAHD  
**AND TO\_CHAR**(HD1.NGAYHD, 'YYYY')=2021  
**GROUP BY** MAGIAY  
**UNION  
SELECT** MAGIAY, **SUM**(SOLUONG) TSL  
**FROM** CN2.CTHD CT2, CN2.HOADON HD2  
**WHERE** CT2.MAHD=HD2.MAHD  
**AND TO\_CHAR**(HD2.NGAYHD, 'YYYY')=2021  
**GROUP BY** MAGIAY) CT  
**GROUP BY** MAGIAY  
**HAVING SUM**(TSL) <= **ALL**(**SELECT SUM**(TSL)  
**FROM**(**SELECT** MAGIAY,**SUM**(SOLUONG) TSL  
**FROM** CN1.CTHD@DBL\_CN1 CT1, CN1.HOADON@DBL\_CN1 HD1  
**WHERE** CT1.MAHD=HD1.MAHD  
**AND TO\_CHAR**(HD1.NGAYHD, 'YYYY')=2021  
**GROUP BY** MAGIAY  
**UNION  
SELECT** MAGIAY, **SUM**(SOLUONG) TSL

**FROM** CN2.CTHD CT2, CN2.HOADON HD2  
**WHERE** CT2.MAHD=HD2.MAHD  
**AND TO\_CHAR**(HD2.NGAYHD, 'YYYY')=2021  
**GROUP BY** MAGIAY) CT  
**GROUP BY** MAGIAY);

**Kết quả trả về:**

Calendar

Description automatically generated

**CÂU 9: Tìm mẫu giày được mua bởi cả khách hàng nam và nữ ở cả 02 chi nhánh.**

(**SELECT DISTINCT** HD.MAGIAY  
**FROM** (**SELECT** \*  
**FROM** CN2.KHACHHANG  
**UNION  
SELECT** \*  
**FROM** CN1.KHACHHANG@DBL\_CN1) KH  
**JOIN** (**SELECT** \*  
**FROM** CN2.HOADON **JOIN** CN2.CTHD **ON**HOADON.MAHD=CTHD.MAHD  
**UNION  
SELECT** \*  
**FROM** CN1.HOADON@DBL\_CN1 **JOIN**CN1.CTHD@DBL\_CN1 **ON** HOADON.MAHD=CTHD.MAHD) HD  
**ON** KH.MAKH = HD.MAKH  
**WHERE** KH.GIOITINH = 'Nu'

**INTERSECT  
SELECT DISTINCT** HD.MAGIAY  
**FROM** (**SELECT** \*  
**FROM** CN2.KHACHHANG  
**UNION  
SELECT** \*  
**FROM** CN1.KHACHHANG@DBL\_CN1) KH  
**JOIN** (**SELECT** \*  
**FROM** CN2.HOADON **JOIN** CN2.CTHD **ON**HOADON.MAHD=CTHD.MAHD  
**UNION  
SELECT** \*  
**FROM** CN1.HOADON@DBL\_CN1 **JOIN**CN1.CTHD@DBL\_CN1 **ON** HOADON.MAHD=CTHD.MAHD) HD  
**ON** KH.MAKH = HD.MAKH  
**WHERE** KH.GIOITINH = 'Nam') A,  
(**SELECT** HD.MAGIAY, **SUM**(HD.SOLUONG) SL  
**FROM** (**SELECT** \*  
**FROM** CN2.KHACHHANG  
**UNION  
SELECT** \*  
**FROM** CN1.KHACHHANG@DBL\_CN1) KH  
**JOIN** (**SELECT** \*  
**FROM** CN2.HOADON **JOIN** CN2.CTHD **ON**HOADON.MAHD=CTHD.MAHD  
**UNION  
SELECT** \*  
**FROM** CN1.HOADON@DBL\_CN1 **JOIN**CN1.CTHD@DBL\_CN1 **ON** HOADON.MAHD=CTHD.MAHD) HD  
**ON** KH.MAKH = HD.MAKH  
**WHERE** KH.GIOITINH = 'Nu'

**GROUP BY** HD.MAGIAY) B,  
(**SELECT** HD.MAGIAY, **SUM**(HD.SOLUONG) SL  
**FROM** (**SELECT** \*  
**FROM** CN2.KHACHHANG  
**UNION  
SELECT** \*  
**FROM** CN1.KHACHHANG@DBL\_CN1) KH  
**JOIN** (**SELECT** \*  
**FROM** CN2.HOADON **JOIN** CN2.CTHD **ON**HOADON.MAHD=CTHD.MAHD  
**UNION  
SELECT** \*  
**FROM** CN1.HOADON@DBL\_CN1 **JOIN**CN1.CTHD@DBL\_CN1 **ON** HOADON.MAHD=CTHD.MAHD) HD  
**ON** KH.MAKH = HD.MAKH  
**WHERE** KH.GIOITINH = 'Nam'  
**GROUP BY** HD.MAGIAY) C  
**WHERE** A.MAGIAY=B.MAGIAY **AND** A.MAGIAY=C.MAGIAY;

**Kết quả trả về:**

A picture containing text, receipt

Description automatically generated

**CÂU 10: Tìm khách hàng đã mua hết tất cả loại giày có giá lớn hơn 500.000 tại chi nhánh 1.**

**SELECT DISTINCT**(MAKH)  
**FROM** CN1.HOADON@DBL\_CN1 HD1, CN1.CTHD@DBL\_CN1 CT1,  
CN2.GIAY G1  
**WHERE** HD1.MAHD=CT1.MAHD  
**AND** CT1.MAGIAY=G1.MAGIAY  
**AND** THUONGHIEU **NOT IN**(**SELECT DISTINCT**(THUONGHIEU)  
**FROM** CN2.GIAY G  
**WHERE** GIATIEN > 500000  
**AND NOT EXISTS**(**SELECT** \*  
**FROM** CN1.CTHD@DBL\_CN1 CT2, CN1.HOADON@DBL\_CN1 HD2  
**WHERE** G.MAGIAY=CT2.MAGIAY  
**AND** HD2.MAHD=CT2.MAHD  
**AND** HD2.MAHD=HD1.MAHD));

**Kết quả trả về:**

Text

Description automatically generated

# HÀM, THỦ TỤC, RÀNG BUỘC TOÀN VẸN TRUY VẤN TRÊN MÔI TRƯỜNG PHÂN TÁN.

Trong phần này, nhóm sẽ trình bày về các ràng buộc, hàm cần được cài đặt trong môi trường phân tán để hoàn thiện hóa khả năng cập nhập, truy vấn, đọc và ghi dữ liệu trong môi trường phân tán của dữ liệu với sự truy vấn từ những máy khác nhau.

## Hàm, thủ tục trong môi trường phân tán.

Tiến hành cài đặt một số trigger, hàm và thủ tục như sau:

* **Trigger:**

**CREATE OR REPLACE TRIGGER** Update\_CTHD

**AFTER UPDATE ON** CTHD

**FOR EACH ROW**

**DECLARE**

V\_GIATIEN GIAY.GiaTien%**TYPE**;

DIFF NUMBER;

CN CHINHANH.MACN%**TYPE**;

**BEGIN**

**SELECT** GIAY.GIATIEN **INTO** V\_GIATIEN **FROM** GIAY

**WHERE** MAGIAY = :NEW.MAGIAY;

**SELECT** MACN **INTO** CN **FROM** HOADON

**WHERE** MAHD=:**NEW**.MAHD;

DIFF := :**NEW**.SOLUONG - :**OLD**.SOLUONG;

**UPDATE** HOADON

**SET** ThanhTien = ThanhTien + V\_GIATIEN \* DIFF

**WHERE** MaHD = :**NEW**.MaHD;

**IF** (CN = 'CN01') **THEN**

**UPDATE** G\_QLKHO

**SET** SOLUONG = SOLUONG - DIFF

**WHERE** MAGIAY = :**NEW**.MAGIAY;

**ELSE**

**UPDATE** CN2.G\_QLKHO@DBL\_CN2

**SET** SOLUONG=SOLUONG - DIFF

**WHERE** MAGIAY = :**NEW**.MAGIAY;

**END IF**;

**END**;

* **Function:**

**CREATE OR REPLACE FUNCTION** TongHD(v\_MaNV **IN** VARCHAR2)

**RETURN number IS**

tong1 number := 0;

tong2 number := 0;

**BEGIN**

**select COUNT**(MAHD) **into** tong1 **FROM** HOADON **WHERE** MANV=v\_MaNV;

**select COUNT**(MAHD) **into** tong2 **FROM** CN2.HOADON@DBL\_CN2 **WHERE** MANV=v\_MaNV;

RETURN tong1+tong2;

**END;**

**set serveroutput on size 30000;**

* **Procedure:**

**create or replace procedure** NhapKho(v\_MaGiay **in** varchar2,v\_SoLuong **in** Number)

**As**

dem int;

**Begin**

**select count**(MaGiay) **into** dem **from** TX\_QLKHO **where** MaGiay = v\_MaGiay;

**if**(dem>0) **then**

**update** G\_QLKHO

**set** SoLuong = SoLuong + v\_SoLuong, NgayNhapKho = **CURRENT\_DATE**

**where** MaGiay = v\_MaGiay;

**else**

**insert into** G\_QLKHO **values** ('CN01', v\_MaGiay, v\_SoLuong, **CURRENT\_DATE**);

**insert into** G\_QLBH **values** ('CN01', v\_MaGiay, 'Con Hang', 0);

**end if**;

**COMMIT**;

**End**;

## **Ràng buộc toàn vẹn.**

Cài đặt ràng buộc toàn vẹn cho giới tính của khách hàng chỉ có thể là “Nam” hoặc “Nu”.

Nội dung:

∀𝑛∈𝐾𝐻𝐴𝐶𝐻𝐻𝐴𝑁𝐺(𝑛.𝐺𝐼𝑂𝐼𝑇𝐼𝑁𝐻 𝐼𝑁 {′𝑁𝑎𝑚′,′𝑁𝑢′})

Bối cảnh: quan hệ KHACHHANG

Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **THÊM** | **XÓA** | **SỬA** |
| **KHACHHANG** | **+** | **-** | **+(GIOITINH)** |

**ALTER TABLE** KHACHHANG **ADD CONSTRAINT** ck\_kh **CHECK** (GIOITINH='Nam' **OR** GIOITINH='Nu')

# Các mức cô lập trong môi trường phân tán.

Tìm hiểu, thiết lập và áp dụng việc cài đặt mức cô lập trong việc thực hiện truy vấn trong phân tán để giải quyết 1 số lỗi có thể xảy ra.

## Thực hiện các mức cô lập trên môi trường phân tán.

* **Lost update**

Tình trạng này xảy ra khi có nhiều hơn một giao tác cùng thực hiện cập nhật trên 1 đơn vị dữ liệu. Khi đó, tác dụng của giao tác cập nhật thực hiện sau sẽ đè lên tác dụng của thao tác cập nhật trước.

-- Read committed

Máy 1:

**select** \* **from** CN1.G\_QLBH **where** MAGIAY = 'G01';

**UPDATE** CN1.G\_QLBH **SET** KHUYENMAI = 10 **WHERE** MAGIAY = 'G01';

**UPDATE** CN2.G\_QLBH@DBL\_CN2 **SET** KHUYENMAI = 10 **WHERE** MAGIAY = 'G01';

Máy 2:

**UPDATE** CN1.G\_QLBH@DBL\_CN1 **SET** KHUYENMAI = 20 **WHERE** MAGIAY = 'G01';

**UPDATE** CN2.G\_QLBH **SET** KHUYENMAI = 20 **WHERE** MAGIAY = 'G01';

Máy 1: commit;

1. Máy 2: commit;
2. Ban đầu set khuyến mãi của “G01” là 0.
3. Graphical user interface, text, application, email

   Description automatically generated
4. Website

   Description automatically generated with low confidence
5. Sau đó set KHUYENMAI của G01 là 10 nhưng chưa commit. Sau khi update cả 2 máy, thì khuyến mãi của “G01” đã thay đổi bằng 20, ở đây ta có thể thấy dữ liệu đã update ở máy 1 đã bị mất do dữ liệu của máy 2 ghi đè lên.

Graphical user interface, text, application, email

Description automatically generated

**Khắc phục:**

-- Serializable

Máy 1:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**UPDATE** CN1.G\_QLBH **SET** KHUYENMAI = 0 **WHERE** MAGIAY = 'G01';

**UPDATE** CN2.G\_QLBH@DBL\_CN2 **SET** KHUYENMAI = 0 **WHERE** MAGIAY = 'G01';

MÁY 2:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**UPDATE** CN1.G\_QLBH@DBL\_CN1 **SET** KHUYENMAI = 20 **WHERE** MAGIAY = 'G01';

**UPDATE** CN2.G\_QLBH **SET** KHUYENMAI = 20 **WHERE** MAGIAY = 'G01';

MÁY 1: COMMIT;

MÁY 2: COMMIT;

Để khắc phục tình trạng này, ta SET TRANSACTION ISOLATION LEVEL SERIALIZABLE cho cả 2 máy. Sau khi update dữ liệu cả 2 máy và máy 1 commit, máy 2 sẽ hiện thông báo lỗi.

Graphical user interface, text, application, email

Description automatically generated

Mặc dù máy 2 đã commit KHUYENMAI lên 20% nhưng thông tin vẫn được giữ như đã commit ở máy 1.

Graphical user interface, text, application, email

Description automatically generated

* **Deadlock:** 
  1. Deadlock là hiện tượng mà một hay nhiều lệnh trong database tranh chấp tài nguyên với nhau mà trong đó một lệnh giữ tài nguyên mà lệnh còn lại cần. Điều này dẫn đến việc không có lệnh nào có thể kết thúc để tiến hành giải phóng tài nguyên.

-- Read committed

MÁY 1:

**UPDATE** CN1.KHACHHANG **SET** DIACHI = 'Quan 2, TP HCM' **WHERE** MAKH = 'KH01';

MÁY 2:

**UPDATE** CN1.KHACHHANG@DBL\_CN1 **SET** DIACHI = 'Quan Go Vap, TP HCM' **WHERE** MAKH = 'KH02';

MÁY 1:

**UPDATE** CN1.KHACHHANG **SET** DIACHI = 'Quan 3, TP HCM' **WHERE** MAKH = 'KH02';

MÁY 2:

**UPDATE** CN1.KHACHHANG@DBL\_CN1 **SET** DIACHI = 'TP Da Lat, Lam Dong' **WHERE** MAKH = 'KH01';

MÁY 1: COMMIT;

1. MÁY 2: COMMIT;

Hai máy update dữ liệu địa chỉ, máy 1 update khách hàng “KH01”, máy 2 update khách hàng “KH02”. Sau đó máy 1 update khách hàng “KH02”, máy 2 update khách hàng “KH01”. Lúc này máy 1 xảy ra deadlock.

Graphical user interface, application, Word

Description automatically generated

**Khắc phục:**

-- Serializable

MÁY 1:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**UPDATE** CN1.KHACHHANG **SET** DIACHI = 'Quan 2, TP HCM' **WHERE** MAKH = 'KH01';

MÁY 2:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**UPDATE** CN1.KHACHHANG@DBL\_CN1 **SET** DIACHI = 'Quan Go Vap, TP HCM' **WHERE** MAKH = 'KH02';

MÁY 1:

**UPDATE** CN1.KHACHHANG **SET** DIACHI = 'Quan 3, TP HCM' **WHERE** MAKH = 'KH02';

MÁY 2:

**UPDATE** CN1.KHACHHANG@DBL\_CN1 **SET** DIACHI = 'TP Da Lat, Lam Dong' **WHERE** MAKH = 'KH01';

MÁY 1: COMMIT;

MÁY 2: COMMIT;

Để khắc phục tình trạng trên, ta SET TRANSACTION ISOLATION LEVEL SERIALIZABLE cho cả 2 máy. Thực hiện lại các bước như trên, máy 1 update khách hàng “KH01”, máy 2 update khách hàng “KH02”. Sau đó máy 1 update khách hàng “KH02”, máy 2 update khách hàng “KH01”. Lúc này máy 1 xảy ra deadlock. Máy 1 commit, máy 2 sẽ báo lỗi không update được. Máy 2 commit để hoàn tất giao tác.

Graphical user interface, application

Description automatically generated

Tuy nhiên máy 2 đã thay đổi thông tin của KH02 và KH03 những trong danh sách khác hàng, thông tin vẫn được giữ như ở lần commit của máy 1.

Graphical user interface, application, table

Description automatically generated

* **Non-repeatable:**

Là hiện tượng thông tin cập nhập chưa được commit thì một truy vấn khác thực hiện đè lên làm mất thông tin được cập nhật trước đó.

-- Read committed

MÁY 1:

SELECT \* FROM CN1.G\_QLBH WHERE MAGIAY = ‘G02’;

MÁY 2:

**UPDATE** CN2.G\_QLBH **SET** KHUYENMAI = 20 **WHERE** MAGIAY = ‘G02’;

**UPDATE** CN1.G\_QLBH@DBL\_CN1 **SET** KHUYENMAI = 20 **WHERE** MAGIAY = ‘G02’;

COMMIT;

MÁY 1:

SELECT \* FROM CN1.G\_QLBH WHERE MAGIAY = ‘G02’;

1. COMMIT;

Đầu tiên truy vấn ở CN1 G02 với thông tin KHUYENMAI =10

Graphical user interface, application

Description automatically generated

Tuy nhiên ở chi nhánh 2 thì máy lại update thông tin khuyến mãi của G02 là 20, gây ra mất thông tin trước đó.

Graphical user interface, text, application

Description automatically generated Điều này làm thông tin truy vấn ở máy 1 cũng thay đổi.

Graphical user interface, text, application, email

Description automatically generated

**Khắc phục:**

-- Serializable

MÁY 1:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**SELECT** \* **FROM** CN1.G\_QLBH **WHERE** MAGIAY = 'G03';

MÁY 2:

**SET TRANSACTION ISOLATION LEVEL** SERIALIZABLE;

**UPDATE** CN2.G\_QLBH **SET** KHUYENMAI = 15 **WHERE** MAGIAY = 'G03';

**UPDATE** CN1.G\_QLBH@DBL\_CN1 **SET** KHUYENMAI = 15 **WHERE** MAGIAY = 'G03';

COMMIT;

MÁY 1:

**SELECT** \* **FROM** CN1.G\_QLBH **WHERE** MAGIAY = 'G03';

COMMIT;

Để khắc phục tình trạng trên, ta SET TRANSACTION ISOLATION LEVEL SERIALIZABLE cho cả 2 máy. Truy vấn tới “G02” có khuyến mãi 20. Sau đó máy 2 update khuyến mãi là 1. Sau đó ta truy vấn lại “G02”, khuyến mãi lúc này vẫn là 20.

Graphical user interface, application

Description automatically generated

Chỉ khi nào máy 1 commit thì khuyến mãi ở máy 2 mới được cập nhật.

Graphical user interface, text, application

Description automatically generated

* **Phantom read:**

Phantom read problem xảy ra khi một giao dịch (transaction) đọc một số biến từ buffer và khi nó đọc cùng một biến sau đó, nó phát hiện ra rằng biến đó không tồn tại.

-- Read committed

MÁY 1:

SELECT \* FROM CN1.HOADON;

MÁY 2:

TAOHD('NV01','KH01');

MÁY 1:

SELECT \* FROM CN1.HOADON;

1. COMMIT;

Ở máy 1, ta truy vấn hóa đơn chi nhánh 1.

1. Sau đó máy 2 tạo thêm một hóa đơn. Máy 1 truy vấn lại hóa đơn, thấy hóa đơn được tạo thêm ở máy 2, mặc dù máy 1 vẫn còn trong giao tác.
2. Graphical user interface, text, application, email

   Description automatically generated

**Khắc phục:**

-- Serializable

MÁY 1:

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

SELECT \* FROM CN1.HOADON;

MÁY 2:

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

TAOHD('NV01','KH01');

MÁY 1:

SELECT \* FROM CN1.HOADON;

1. COMMIT;

Để khắc phục tình trạng trên, ta SET TRANSACTION ISOLATION LEVEL SERIALIZABLE cho cả 2 máy. Ta truy vấn hóa đơn chi nhánh 1.

Graphical user interface, application, Word

Description automatically generated

Sau đó máy 2 tạo thêm một hóa đơn. Máy 1 truy vấn lại hóa đơn, thấy hóa đơn vẫn giữ nguyên dữ liệu như cũ.

Graphical user interface, application

Description automatically generated

Lúc này máy 1 commit để hoàn thành giao tác và truy vấn lại hóa đơn, hóa đơn mới được tạo ở máy 2 sẽ xuất hiện.

Graphical user interface, text, application, email

Description automatically generated

# Tối ưu hóa truy vấn trên môi trường phân tán

Cho biết những nhân viên(Mã nhân viên, Họ tên, SODT) và tên chi nhánh của kho có lương trên 5 triệu đã những đôi giày có giá từ 1 triệu cho các khách hàng Nam

## Câu truy vấn ban đầu chưa được tối ưu:

**Text

Description automatically generated with medium confidence**

Kết quả

**Table

Description automatically generated**

Kiểm tra ngữ nghĩa của truy vấn

**Background pattern

Description automatically generated**

Đồ thị truy vấn liên thông nên câu truy vấn đúng ngữ nghĩa

## Phân rã câu truy vấn ban đầu

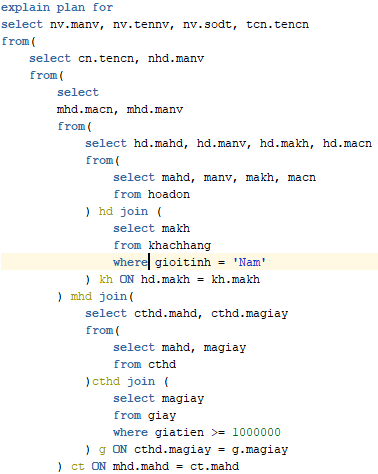
A screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface, application, Teams

Description automatically generated

## Câu truy vấn đại số quan hệ sau khi tối ưu hóa toàn cục

CODE:

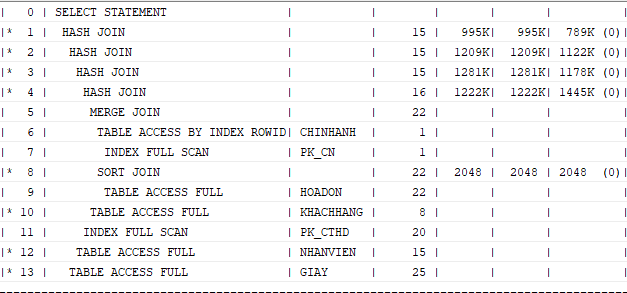


Text

Description automatically generated

## Đánh giá thông qua EXPLAIN plan

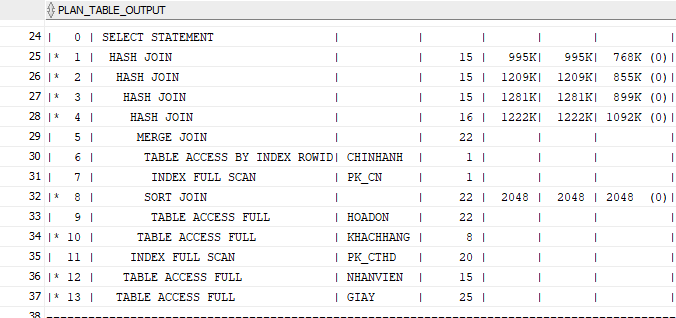
Chưa tối ưu hóa



A screenshot of a computer

Description automatically generated with medium confidence

Sau khi tối ưu

A screenshot of a computer

Description automatically generated with medium confidence

Nhận xét:

## Lược đồ phân mảnh

Quan hệ CHINHANH phân mảnh ngang chính theo tên chi nhánh:

*CN1 = σ TenCN = 'Quan 9, TPHCM' CHINHANH*

*CN2 = σ TenCN ='Quan 1, TPHCM' CHINHANH*

Quan hệ G\_QLKHO, G\_QLBH, HOADON, MAHOADON, KHACHHANG, NHANVIEN phân mảnh ngang dẫn xuất như sau:

*QLK1 = G\_QLKHO ⋉MACN CN1*

*QLK2 = G\_QLKHO ⋉MACN CN2*

*QLBH1 = G\_QLBH ⋉MACN CN1*

*QLBH2 = G\_QLBH ⋉MACN CN2*

*HD1 = HOADON ⋉MACN CN1*

*HD2 = HOADON ⋉MACN CN2*

*CT1 = HD1 ⋉MAHD CN1*

*CT2 = HD2 ⋉MAHD CN2*

*KH1 = KHACHHANG ⋉MACN CN1*

*KH2 = KHACHHANG ⋉MACN CN2*

*NV1 = NHANVIEN* *⋉MACN CN1*

*NV2 = NHANVIEN* *⋉MACN CN2*

Quan hệ KHACHHANG phân mảnh hỗn hợp như sau:

*KH1A = π TENKH, DIACHI, SODT (KHACHHANG ⋉ MACN CN1)*

*KH1B= π MAKH, TUOI, GIOITINH (KHACHHANG ⋉ MACN CN1)*

*KH2A = π TENKH, DIACHI, SODT (KHACHHANG ⋉ MACN CN2)*

*KH2B= π π MAKH, TUOI, GIOITINH (KHACHHANG ⋉ MACN CN2)*

Quan hệ GIAY, NHANVIEN được nhân bản tại tất cả chi nhánh

## Cây truy vấn dựa vào lược đồ phân mảnh

Graphical user interface, application, Teams

Description automatically generated

**your paper not being published.**

# CHƯƠNG 5: CƠ CHẾ NHÂN BẢN TRONG ORACLE

## Khái niệm

Nhân bản cơ sở dữ liệu Oracle không là gì ngoài một quy trình được sử dụng để tạo cơ sở dữ liệu giống hệt cơ sở dữ liệu Oracle hiện có. Kỹ thuật nhân bản được sử dụng để tạo một bản sao của Cơ sở dữ liệu hiện có để thực hiện các hoạt động thử nghiệm khác nhau của DBA như sao lưu và phục hồi.

Cách thức hoạt động:

|  |  |
| --- | --- |
| Phương pháp nhân bản | Cách thức hoạt động |
| Recovery Manager (RMAN) Backup | Kết nối các phiên bản Oracle nguồn và đích  Sao chép các tệp cơ sở dữ liệu bằng tính năng sao chép RMAN  Phục hồi và mở cơ sở dữ liệu nhân bản |
| Staging Areas | Sao lưu từng tệp cơ sở dữ liệu và lưu trữ nó trong khu vực tổ chức  Chuyển từng tệp sao lưu từ nguồn đến đích  Khôi phục từng tệp sao lưu vào các vị trí đã chỉ định  Phục hồi và mở cơ sở dữ liệu nhân bản |
| An Existing Backup | Tạo cơ sở dữ liệu nhân bản tại thời điểm cụ thể hoặc SCN  Xác thực các bản sao lưu trước khi thực hiện thao tác sao chép  Chuyển các tệp nhật ký làm lại được lưu trữ cần thiết sang máy chủ đích  Phục hồi và mở cơ sở dữ liệu nhân bản |

## Cơ chế nhân bản của RMAN

### Cơ chế nhân bản của Recovery Manager (RMAN) Backup

- Tạo một control file cho cơ sở dữ liệu nhân bản

- Khôi phục các data file đích vào cơ sở dữ liệu nhân bản

- Thực hiện phục hồi không đầy đủ bằng cách sử dụng tất cả các bản sao lưu incremental có sẵn và các archived redo log files

- Tắt và khởi động lại instance phụ trợ

- Mở cơ sở dữ liệu nhân bản với tùy chọn RESETLOGS

- Tạo các online redo log files

- Tạo một định danh cơ sở dư liệu mới, duy nhất (DBID) cho cơ sở dữ liệu nhân bản

### Các bước tiến hành

-Tạo tệp mật khẩu trên máy đích

- Thiết lập kết nối giữa máy chủ và máy đích (tnsnames.ora, sqlnet.ora)

- Tạo các thư mục cho các tệp cơ sở dữ liệu

- Lấy bản sao lưu RMAN từ máy chủ và sao chép nó vào máy đích.

- Khởi động cơ sở dữ liệu đích ở chế độ Nomount

- Chạy lệnh nhân bản cơ sở dữ liệu RMAN

Link video tham khảo: <https://vimeo.com/12847611?embedded=true&source=video_title&owner=3403610>

# TÀI LIỆU THAM KHẢO

1. Slide bài giảng và hướng dẫn thực hành do giảng viên cung cấp.

2. Tài liệu tham khảo giáo viên cung cấp.

3. Tài liệu tạo hàm: <https://docs.oracle.com/cd/B19306_01/server.102/b14200/statements_5009.htm>

4. Tài liệu tạo thủ tục: <https://docs.oracle.com/cd/B19306_01/server.102/b14200/statements_6009.htm>

5. Tài liệu về nhân bản cơ sở dữ liệu:

<https://docs.oracle.com/en/enterprise-manager/cloud-control/enterprise-manager-cloud-control/13.3.1/emlcm/cloning-oracle-databases-and-pluggable-databases.html#GUID-DDDC435F-C89B-482C-91CC-34C03C53099D>

<https://www.youtube.com/watch?v=g6aVyKYoIsI&t=554s>