**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**



**BÁO CÁO ĐỒ ÁN THỰC HÀNH MÔN HỌC**

**HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**ĐỀ TÀI:**

**XÂY DỰNG PHẦN MỀM**

**QUẢN LÝ CỬA HÀNG TIỆN LỢI**

Nhóm sinh viên thực hiện:

1. Đỗ Sĩ Đạt MSSV: 21521932

2. Hoàng Gia Lộc MSSV: 21521086

3. Vũ Thanh Doan MSSV: 21520191

4. Võ Hồng Kim Anh MSSV: 21520597

**TP. HỒ CHÍ MINH, NĂM 2023**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2023*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**MỤC LỤC**

[LỜI CẢM ƠN 4](#_Toc137525355)

[TÓM TẮT ĐỀ TÀI 5](#_Toc137525356)

[TÓM TẮT ĐỀ TÀI 6](#_Toc137525357)

[CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN 7](#_Toc137525358)

[1.1. Phát biểu bài toán 7](#_Toc137525359)

[1.2. Mục tiêu của đề tài 7](#_Toc137525360)

[1.3. Yêu cầu chức năng 8](#_Toc137525361)

[1.4. Yêu cầu phi chức năng 9](#_Toc137525362)

[1.5. Công cụ sử dụng 9](#_Toc137525363)

[1.6. Thiết kế mô hình quan hệ. 9](#_Toc137525364)

[1.6.1. Mô hình thực thể mối kết hợp 10](#_Toc137525365)

[1.6.2. Mô hình dữ liệu quan hệ 11](#_Toc137525366)

[1.6.3. Bảng thuyết minh quan hệ và và thuộc tính 11](#_Toc137525367)

[1.7. Mô tả ràng buộc toàn vẹn 16](#_Toc137525368)

[1.7.1. Ràng buộc toàn vẹn khoá chính 16](#_Toc137525369)

[1.7.2. Ràng buộc toàn vẹn khoá ngoại 17](#_Toc137525370)

[1.7.3. Ràng buộc liên thuộc tính 18](#_Toc137525371)

[1.7.4. Ràng buộc miền giá trị 19](#_Toc137525372)

[1.7.5. Ràng buộc liên bộ, liên quan hệ 20](#_Toc137525373)

[1.7.6. Ràng buộc liên thuộc tính, liên quan hệ 20](#_Toc137525374)

[1.7.7. Ràng buộc do thuộc tính tổng hợp 20](#_Toc137525375)

[CHƯƠNG 2. XÂY DỰNG CÁC GIAO TÁC 22](#_Toc137525376)

[2.1. Trigger 22](#_Toc137525377)

[2.1.1. Định nghĩa trigger 22](#_Toc137525378)

[2.1.2. Danh sách các trigger trong đồ án 23](#_Toc137525379)

[2.1.3. Mô tả một số trigger tiêu biểu 26](#_Toc137525380)

[2.1.3.1. Trigger trên bảng nhân viên 26](#_Toc137525381)

[2.1.3.2. Trigger trên bảng tài khoản 27](#_Toc137525382)

[2.1.3.3. Trigger trên bảng sản phẩm 28](#_Toc137525383)

[2.1.3.4. Trigger trên bảng chi tiết kho 30](#_Toc137525384)

[2.1.3.5. Trigger trên bảng chi tiết hoá đơn 32](#_Toc137525385)

[2.1.3.6. Trigger trên bảng hoá đơn. 34](#_Toc137525386)

[2.2. Stored Procedure 35](#_Toc137525387)

[2.2.1. Định nghĩa Stored Procedure 35](#_Toc137525388)

[2.2.2. Danh sách các Stored Procedure có trong đồ án 36](#_Toc137525389)

[2.2.3. Một số Stored procedure tiêu biểu 43](#_Toc137525390)

[2.2.3.1. Thêm khách hàng 43](#_Toc137525391)

[2.2.3.2. Thêm chi tiết kho 44](#_Toc137525392)

[2.2.3.3. Lấy thông tin TOP 10 sản phẩm bán chạy nhất 45](#_Toc137525393)

[CHƯƠNG 3. XỬ LÝ GIAO TÁC ĐỒNG THỜI 47](#_Toc137525394)

[3.1. Các mức cô lập trong Oracle 47](#_Toc137525395)

[3.1.1. Read commited 47](#_Toc137525396)

[3.1.2. Serializable 47](#_Toc137525397)

[3.2. Cơ chế khoá 49](#_Toc137525398)

[3.2.1. Định nghĩa 49](#_Toc137525399)

[3.2.2. Các loại khoá 49](#_Toc137525400)

[3.2.3. Cơ chế khóa tác động đến sự tương tác giữa việc đọc và ghi dữ liệu 49](#_Toc137525401)

[3.2.4. Các chế độ khoá trong Oracle 50](#_Toc137525402)

[3.3. Transaction 50](#_Toc137525403)

[3.3.1. Định nghĩa 50](#_Toc137525404)

[3.3.2. Các tính chất của transaction 50](#_Toc137525405)

[3.3.3. Cấu trúc của một transaction 51](#_Toc137525406)

[3.3.4. Transaction controll 52](#_Toc137525407)

[3.3.5. Hàng đợi transaction 54](#_Toc137525408)

[3.4. Deadlock 55](#_Toc137525409)

[3.4.1. Định nghĩa 55](#_Toc137525410)

[3.4.2. Xử lý deadlock trong oracle 55](#_Toc137525411)

[3.5. Các vấn đề xảy ra khi truy xuất dữ liệu đồng thời 57](#_Toc137525412)

[3.5.1. Mất dữ liệu cập nhật (Lost Update) 57](#_Toc137525413)

[3.5.2. Đọc dữ liệu chưa commit(Dirty Read) 63](#_Toc137525414)

[3.5.3. Đọc dữ liệu không lặp lại (Non – repeatable read) 66](#_Toc137525415)

[3.5.4. Bóng ma(Phantom read) 70](#_Toc137525416)

[CHƯƠNG 4. THIẾT KẾ GIAO DIỆN 74](#_Toc137525417)

[4.1. Môi trường phát triển và triển khai hệ thống 74](#_Toc137525418)

[4.1.1. Môi trường phát triển 74](#_Toc137525419)

[4.1.2. Môi trường triển khai 74](#_Toc137525420)

[4.2. Màn hình giao diện 75](#_Toc137525421)

[4.2.1. Màn hình đăng nhập 75](#_Toc137525422)

[4.2.2. Màn hình trang chủ 75](#_Toc137525423)

[4.2.3. Màn hình chức năng quản lý nhân viên 77](#_Toc137525424)

[4.2.4. Màn hình chức năng quản lý sản phẩm 79](#_Toc137525425)

[4.2.5. Màn hình chức năng quản lý khách hàng 82](#_Toc137525426)

[4.2.6. Màn hình chức năng quản lý hóa đơn 85](#_Toc137525427)

[4.2.7. Màn hình chức năng quản lý kho 86](#_Toc137525428)

[4.2.8. Màn hình chức năng quản lý bán hàng 88](#_Toc137525429)

[CHƯƠNG 5. KẾT LUẬN 90](#_Toc137525430)

[5.1. Bảng phân công công việc 90](#_Toc137525431)

[5.2. Kết quả đạt được 90](#_Toc137525432)

[5.2.1. Lý thuyết 90](#_Toc137525433)

[5.2.2. Công nghệ 90](#_Toc137525434)

[5.3. Hướng phát triển 90](#_Toc137525435)

[5.3.1. Về hệ thống 90](#_Toc137525436)

[5.3.2. Về con người 91](#_Toc137525437)

[DANH MỤC TÀI LIỆU THAM KHẢO 92](#_Toc137525438)

# LỜI CẢM ƠN

Lời đầu tiên, nhóm chúng em xin gửi lời cảm ơn chân thành đến toàn thể giảng viên trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM cũng như là nhà trường vì đã trang bị những kiến thức cơ bản làm nền tảng để nhóm chúng em thực hiện đồ án môn học này.

Đặc biệt hơn nữa, chúng em xin gửi lời cảm ơn chân thành đến cô Thái Bảo Trân và thầy Nguyễn Hồ Duy Tri đã giúp đỡ chúng em trong suốt quá trình học tập và thực hiện đồ án, sự quan tâm, giảng dạy tận tình của thầy cô đã hỗ trợ chúng em rất nhiều trong thời gian vừa qua và qua đó chúng em đã học hỏi được nhiều kiến thức hơn. Một lần nữa nhóm xin gửi lời cảm ơn sâu sắc đến quý thầy cô.

Trong quá trình nghiên thực hiện đồ án “Xây dựng phần mềm Quản lý Cửa hàng tiện lợi”, nhóm chúng em đã kết hợp giữa những kiến thức căn bản về những gì được cô và thầy trao đổi và truyền đặt trên lớp để cố gắng hoàn thiện đoàn một cách tốt nhất có thể. Mặc dù đã vận dụng tối đa những gì đã học được nhưng chúng em vẫn khó có thể tránh khỏi những sai sót. Chính vì vậy, nhóm em rất mong nhận được sự góp ý từ phía cô và thầy để có thể hoàn thiện một cách tốt nhất có thể. Qua đó cũng tích lũy và học hỏi kinh nghiệm để làm hành trang cho tương lai.

Lời cuối cùng, nhóm chúng em xin chúc quý thầy cô thật nhiều sức khỏe và niềm vui để có thể tiếp tục giảng dạy về truyền đạt thật nhiều kiến thức bổ ích đến cho những sinh viên khác. Chúng em xin chân thành cảm ơn!

# TÓM TẮT ĐỀ TÀI

*(Tiếng Việt)*

Trong thời đại hiện đại ngày nay, cửa hàng tiện lợi đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày của chúng ta. Với sự phát triển không ngừng của công nghệ và sự thay đổi trong thói quen tiêu dùng, cửa hàng tiện lợi đã tiến xa hơn khái niệm truyền thống của một nơi mua sắm đơn giản, việc quản lý một cửa hàng tiện lợi trở nên phức tạp và đòi hỏi sự chuyên nghiệp và hiệu quả. Để đáp ứng nhu cầu này, nhóm đã xây dựng một phần mềm quản lý cửa hàng tiện lợi.

Phần mềm quản lý cửa hàng tiện lợi không chỉ đơn thuần là một phần mềm, mà là một giải pháp toàn diện giúp tối ưu hóa mọi khía cạnh của hoạt động cửa hàng. Từ việc quản lý hàng tồn kho, đặt hàng, bán hàng, thanh toán, đến quản lý nhân viên, v.v. Phần mềm này mang đến sự tổ chức và hiệu quả cho cửa hàng tiện lợi, với mục đích xây dựng được một phần mềm hoàn thiện để quản lý và nhân viên trong cửa hàng có một phần mềm quản lý thông tin, hoạt động mua bán của cửa hàng. Với nhiều tính năng dễ sử dụng và tối ưu, phần mềm này cung cấp các chức năng quản lý bao gồm quản lý nhân viên, quản lý sản phẩm, quản lý khách hàng, quản lý hóa đơn, quản lý kho và bán hàng.

Việc sử dụng phần mềm quản lý đã được tối ưu, chuyên biệt hóa và phân tách rõ các tính năng với những người dùng khác nhau. Quản lý sẽ tiến hành đăng nhập vào phần mềm và được cấp quyền quản lý để thực hiện các nhiệm vụ quản lý bao gồm: nhân viên, tài khoản, sản phẩm, khách hàng, hóa đơn, chi tiết hoá đơn và kho, chi tiết kho. Đối với nhân viên, sẽ được phân quyền để quản lý sản phẩm, hóa đơn, khách hàng, hoá đơn, chi tiết hoá đơn, kho và chi tiết kho. Mỗi một nhân viên khi vào làm sẽ được quản lý cấp 1 tài khoản và mật khẩu trong suốt quá trình làm việc.

Đồ án này sẽ tập trung vào việc phát triển các tính năng cốt lõi của phần mềm quản lý cửa hàng tiện lợi và đảm bảo tính ổn định, bảo mật và hiệu suất của phần mềm. Đồ án sẽ được thực hiện thông qua việc xác định yêu cầu, thiết kế giao diện, phát triển mã nguồn và kiểm thử để đảm bảo rằng phần mềm đáp ứng được các yêu cầu của chủ cửa. Chính vì vậy việc sử dụng phần mềm quản lý cửa hàng tiện lợi, các doanh nghiệp có thể tiết kiệm thời gian, công sức trong việc quản lý và tối ưu hóa quy trình kinh doanh. Từ đó nâng cao hiệu suất và mở rộng cơ hội phát triển của cửa hàng tiện lợi.

# TÓM TẮT ĐỀ TÀI

*(Tiếng Anh)*

In today's modern era, convenience stores have become an integral part of our daily lives. With the continuous development of technology and changes in consumers habits, convenience stores have evolved beyond the traditional concept of a simple shopping place. Now, the management of a convenience store has become complicated and requires professionalism and efficiency. To meet this demand, our team has developed a comprehensive convenience store management system.

The convenience store management system is not just a software, but a holistic solution that optimizes all aspects of store operations. It covers inventory management, ordering, sales, payment, employee management, and more. The system brings organization and efficiency to the convenience store, aiming to establish a complete application system that facilitates smooth and convenient communication among all store employees. With user-friendly and optimized features, the system provides management functions including employee management, product management, customer management, invoice management, logistics management, and sales management.

The use of the optimized and specialized management system separates different user roles. Managers will log into the system and be granted management rights to perform tasks related to employees, products, customers, invoices, and logistics. On the other hand, employees will be assigned permissions to manage products, invoices, customers, sales activities, and logistics. Each employee will be given one account and password upon starting their job, which they will use throughout their employment.

This project will focus on developing the core features of the convenience store management application and ensuring its stability, security, and performance. The project will involve requirements determination, interface design, source code development, and testing to ensure that the software meets the requirements of store owners and provides maximum benefits to users. By utilizing the convenience store management system, companies can save time and effort in managing and optimizing their business processes. Thereby improving efficiency and expanding the development opportunities of the convenience store.

**CHƯƠNG 1.** **GIỚI THIỆU TỔNG QUAN**

* 1. **Phát biểu bài toán**

Trong xã hội ngày càng phát triển, đời sống con người trở nên ngày càng bận rộn, thời gian đã trở thành một tài nguyên vô cùng quý giá. Vì thể, cửa hàng tiện lợi là một giải pháp mang đến sự linh hoạt cần thiết cho khách hàng. Đây là nơi cung cấp đa dạng các sản phẩm từ thực phẩm tươi sống, hàng tiêu dùng, đồ uống, báo chí, đồ điện tử và nhiều hơn thế nữa. Một số cửa hàng tiện lợi hoạt động 24/7, cho phép khách hàng mua sắm và tiếp cận dịch vụ bất kể thời gian nào trong ngày. Điều này rất hữu ích cho những người có lịch trình bận rộn, những người làm việc theo ca hoặc cần mua hàng vào ban đêm. Khách hàng có thể tìm thấy hầu hết những mặt hàng cần thiết hàng ngày trong một cửa hàng duy nhất. Điều này cũng giúp tiết kiệm công sức của khách hàng khi không phải đi đến nhiều nơi khác nhau để mua sắm.

Cửa hàng tiện lợi là một loại hình kinh doanh bán lẻ phổ biến, có nhiều sản phẩm và dịch vụ khác nhau như một siêu thị thu nhỏ. Dù quy mô không lớn như siêu thị hay trung tâm mua sắm nhưng các yêu tố để phần mềm có thể hoạt động không kém phần phức tạp. Cần phải giải quyết các vấn đề về quản lý, vận hành và tối ưu hoạt động kinh doanh. Việc điều phối và quản lý hiệu quả tất cả các hoạt động này đòi hỏi sự tổ chức tốt, khả năng xử lý dữ liệu lớn, và khả năng ra quyết định nhanh chóng. Nhằm đáp ứng các yêu cầu như trên, phần mềm quản lý cửa hàng tiện lợi giúp đơn giản hóa quy trình kinh doanh, tối ưu hóa năng suất và giảm thiểu sai sót. Do đó, việc sử dụng một ứng dụng quản lý tích hợp tất cả các chức năng sẽ giúp các doanh nghiệp giải quyết được vấn đề một cách hiệu quả và nhanh chóng hơn, đem lại một sự đầu tư xứng đáng.

Tóm lại, Phần mềm quản lý cửa hàng tiện lợi có ý nghĩa quan trọng trong việc nâng cao tính hiệu quả và cạnh tranh của các doanh nghiệp trong thị trường bán lẻ hiện nay.

* 1. **Mục tiêu của đề tài**

Xây dựng được một phần mềm ứng dụng hoàn thiện để quản lý và tất cả những nhân viên trong cửa hàng có một phần mềm trao đổi một cách dễ dàng và thuận tiện. Ứng dụng sẽ giới thiệu những chức năng tối ưu nhất và dễ dàng sử dụng đối với tất cả mọi người. Phần mềm quản lý cửa hàng tiện lợi được xây dựng là một phần mềm quản lý tích hợp, lưu trữ thông tin tập trung. Được thiết kế bao gồm các chức năng cơ bản như quản lý nhân viên, quản lý tài khoản, quản lý sản phẩm, quản lý kho, chi tiết kho, quản lý khách hàng, quản lý hoá đơn và các chi tiết hoá đơn.

Người dùng được chia thành 2 loại: Quản lý và Nhân viên. Phần mềm yêu cầu các loại tài khoản khác nhau để chuyên biệt hóa và phân tách rõ nhiệm vụ của từng nhân viên. Quản lý sẽ đảm nhận nhiệm vụ đăng nhập vào phần mềm và sẽ được phân quyền quản lý: nhân viên, tài khoản, khách hàng, sản phẩm, hoá đơn, chi tiết hoá đơn, kho, chi tiết kho. Nhân viên được phân quyền quản lý: sản phẩm, hoá đơn, khách hàng, kho.

Các tính năng chính của phần mềm bao gồm:

* Quản lý thông tin nhân viên
* Quản lý thông tin tài khoản
* Quản lý thông tin sản phẩm
* Quản lý thông tin khách hàng
* Quản lý thông tin hóa đơn
* Quản lý thông tin chi tiết hóa đơn
* Quản lý thông tin chi tiết kho
  1. **Yêu cầu chức năng**
* Chức năng lưu trữ thông tin
* Thông tin của sản phẩm: Mã sản phẩm, tên sản phẩm, giá sản phẩm, số lượng sản phẩm, phân loại sản phẩm, ngày sản xuất, hạn sử dụng, mô tả sản phẩm.
* Thông tin của hóa đơn: Mã hóa đơn, ngày mua hàng, thông tin khách hàng (nếu có), hình thức thanh toán, trị giá hóa đơn.
* Thông tin của nhân viên: Mã nhân viên, họ tên nhân viên, số điện thoại nhân viên, ngày sinh nhân viên, số Căn cước công dân của nhân viên, địa chỉ, chức vụ, lương, ngày vào làm.
* Thông tin lưu kho: Mã lô hàng, mã nhân viên, ngày nhập, tên nhà cung cấp.
* Thông tin của khách mua hàng: Mã khách hàng, số điện thoại khách hàng, họ tên khách hàng, ngày đăng ký.
* Chức năng thêm, tra cứu, cập nhật, xoá
* Thêm, tra cứu, cập nhật, xoá thông tin của từng sản phẩm
* Thêm, tra cứu, cập nhật, xoá thông tin của từng tài khoản
* Thêm, tra cứu, tìm kiếm, xoá thông tin của nhân viên
* Thêm, tra cứu, tìm kiếm, xoá thông tin của khách hàng
* Thêm, tra cứu, tìm kiếm, xoá thông tin của các lô hàng trong kho
* Thêm, tra cứu, tìm kiếm, xoá thông tin của các chi tiết kho kho
* Thêm, tra cứu thông tin của hóa đơn
* Thêm, tra cứu thông tin của các chi tiết hoá đơn
* Chức năng tính toán
* Tính tổng giá trị hóa đơn, giá của chi tiết hoá đơn.
* Tự động tính số lượng của sản phẩm, khi có các thao tác dữ liệu liên quan đến thuộc tính số lượng trong bảng chi tiết hoá đơn và chi tiết kho
* Chức năng thống kê, báo cáo
  + Thống kê, báo cáo doanh thu của cửa hàng theo thời gian
  1. **Yêu cầu phi chức năng**
* Tính bảo mật : Phần mềm cần có các tính năng bảo mật để đảm bảo an toàn cho dữ liệu của cửa hàng, bao gồm đăng nhập an toàn, quản lý quyền truy cập, mã hóa dữ liệu.
* Tính nhất quán dữ liệu: Phần mềm cần đảm bảo tính nhất quán dữ liệu giữa các phân hệ, chẳng hạn dữ liệu về sản phẩm, kho hàng, doanh thu, đơn hàng, khách hàng, và các thông tin khác.
* Tính hỗ trợ khách hàng: Phần mềm cần hỗ trợ tính năng quản lý khách hàng, chẳng hạn quản lý thông tin khách hàng, lịch sử mua hàng, chương trình khuyến mãi, điểm thưởng, đánh giá và phản hồi của khách hàng. Điều này giúp cửa hàng tiện lợi có thể tương tác và tạo dựng mối quan hệ tốt với khách hàng, từ đó tăng cường sự hài lòng và trung thành của khách hàng.
* Tính quản lý kho hàng: Phần mềm cần hỗ trợ quản lý kho hàng, bao gồm nhập xuất hàng hóa, kiểm kê, theo dõi tồn kho, và đồng bộ hoạt động với các cửa hàng khác nếu có.
  1. **Công cụ sử dụng**

Trong quá trình thực hiện đề tài, nhóm đã sử dụng một số công cụ để phục vụ cho việc triển khai đề tài, bao gồm:

* Hệ điều hành: Windows 11.
* Công cụ quản lý CSDL: SQL Developer.
* Công cụ xây dựng ứng dụng: Visual Studio Code, Apache Netbeans.
* Công cụ quản lý mã nguồn: Github.
  1. **Thiết kế mô hình quan hệ.**
     1. **Mô hình thực thể mối kết hợp**

**A picture containing text, screenshot, diagram, line

Description automatically generated**

Mô hình thực thể mối kết hợp

* + 1. **Mô hình dữ liệu quan hệ**

SANPHAM(MaSP, TenSP, GiaSP, SoLuong, PhanLoai, MoTa, GiaGiam)

NHANVIEN(MaNV, TenNV, SDTNV, NgaySinh, CCCD, DiaChi, ChucVu, Luong, NgayVL)

KHACHHANG(MaKH, SDTKH, TenKH, NgayDK)

HOADON(MaHD, NgayMua, MaNV, MaKH, HinhThuc, TriGia)

KHO(MaLoHang, MaNV, NgayNhap, TenNhaCungCap)

TAIKHOAN(username, password, MaNV)

CHITIETKHO(MaLoHang, MaSP, SoLuong)

CHITIETHOADON(MaHD, MaSP, SoLuong, Gia)

* + 1. **Bảng thuyết minh quan hệ và và thuộc tính**

**SANPHAM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MASP | Mã sản phẩm | Nvarchar2(30) | Not null | Khóa chính |
| TENSP | Tên sản phẩm | Nvarchar2(100) | Not null |  |
| GIASP | Giá sản phẩm | Int |  |  |
| SOLUONG | Số lượng | Int |  |  |
| PHANLOAI | Phân loại | Nvarchar2(50) |  |  |
| MOTA | Mô tả | Nvarchar2(500) |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ SANPHAM

**NHANVIEN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MANV | Mã nhân viên | Nvarchar2(10) | Not null | Khóa chính |
| TENNV | Họ tên nhân viên | Nvarchar2(50) | Not null |  |
| SDTNV | Số điện thoại | Nvarchar2(20) | Not null |  |
| NGAYSINH | Ngày sinh | Date |  |  |
| CCCD | Căn cước công dân | Nvarchar2(20) | Not null |  |
| DIACHI | Địa chỉ | Nvarchar2(200) |  |  |
| CHUCVU | Chức vụ | Nvarchar2(50) |  |  |
| LUONG | Lương | Int |  |  |
| NGAYVL | Ngày vào làm | Date |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ NHANVIEN

**KHACHHANG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MAKH | Mã khách hàng | Nvarchar2(15) | Not null | Khóa chính |
| SDTKH | Số điện thoại | Nvarchar2(20) | Not null |  |
| TENKH | Họ tên | Nvarchar2(50) | Not null |  |
| NGAYDK | Ngày đăng ký | Date |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ KHACHHANG

**HOADON**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MAHD | Mã hóa đơn | Nvarchar2(10) | Not null | Khóa chính |
| NGAYMUA | Ngày lập hóa đơn | Date | Not null |  |
| MANV | Mã nhân viên lập hóa đơn | Nvarchar2(10) | Not null | Khóa ngoại tham chiếu đến MANV của bảng NHANVIEN |
| MAKH | Mã khách hàng mua hóa đơn | Nvarchar2(20) |  | Khóa ngoại tham chiếu đến MAKH của bảng KHACHHANG |
| HINHTHUC | Hình thức thanh toán | Nvarchar2(50) |  |  |
| TRIGIA | Trị giá của hóa đơn | Int |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ HOADON

**KHO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MALOHANG | Mã lô hàng | Nvarchar2(10) | Not null | Khóa chính |
| MANV | Mã nhân viên nhập kho | Nvarchar2(10) | Not null | Khóa ngoại tham chiếu đến MANV của bảng NHANVIEN |
| NGAYNHAP | Ngày nhập lô hàng | Date |  |  |
| TENNHACUNGCAP | Tên nhà cung cấp | Nvarchar2(100) |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ KHO

**TAIKHOAN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| USERNAME | Tên đăng nhập | Nvarchar2(20) | Not null | Khóa chính |
| PASSWORD | Mật khẩu | Varchar(50) | Not null |  |
| MANV | Mã nhân viên | Nvarchar2(10) | Not null | Khóa ngoại tham chiếu đến MANV của bảng NHANVIEN |

Bảng thuyết minh quan hệ và thuộc tính quan hệ TAIKHOAN

**CHITIETKHO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MALOHANG | Mã lô hàng | Nvarchar2(10) | Not null | Khóa chính, khóa ngoại tham chiếu đến MALOHANG của bảng KHO |
| MASP | Mã sản phẩm | Nvarchar2(30) | Not null | Khóa chính, khóa ngoại tham chiếu đến MASP của bảng SANPHAM |
| SOLUONG | Số lượng sản phẩm trong lô hàng | Int |  |  |

Bảng thuyết minh quan hệ và thuộc tính quan hệ CHITIETKHO

**CHITIETHOADON**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| MAHD | Mã hóa đơn | Nvarchar2(10) | Not null | Khóa chính, khóa ngoại tham chiếu đến MAHD của bảng HOADON |
| MASP | Mã sản phẩm | Nvarchar2(30) | Not null | Khóa chính, khóa ngoại tham chiếu đến MASP của bảng SANPHAM |
| SOLUONG | Số lượng | Int | Not null |  |
| GIA | Giá sản phẩm | Int | Not null |  |

Bảng thuyết minh quan hệ và thuộc tính của quan hệ CHITIETHOADON

* 1. **Mô tả ràng buộc toàn vẹn**

### Ràng buộc toàn vẹn khoá chính

**RB1**: Thuộc tính MASP trong quan hệ SANPHAM là duy nhất.

* Nội dung: ∀sp1, sp2 ∈ SANPHAM: sp1. MASP ≠ sp2. MASP
* Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB1** | Thêm | Xóa | Sửa |
| SANPHAM | + | - | + (MASP) |

**RB2**: Thuộc tính MANV trong quan hệ NHANVIEN là duy nhất.

* Nội dung: ∀nv1, nv2 ∈ NHANVIEN: nv1. MANV ≠ nv2. MANV
* Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB2** | Thêm | Xóa | Sửa |
| NHANVIEN | + | - | + (MANV) |

**RB3**: Thuộc tính MAKH trong quan hệ KHACHHANG là duy nhất.

* Nội dung: ∀kh1, kh2 ∈ KHACHHANG: kh1.MAKH ≠ kh2.MAKH
* Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB3** | Thêm | Xóa | Sửa |
| KHACHHANG | + | - | + (MAKH) |

**RB4**: Thuộc tính MAHD trong quan hệ HOADON là duy nhất.

* Nội dung: ∀hd1, hd2 ∈ HOADON: hd1. MAHD ≠ hd2. MAHD
* Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB4** | Thêm | Xóa | Sửa |
| HOADON | + | - | + (MAHD) |

**RB5**: Thuộc tính MALOHANG trong quan hệ KHO là duy nhất.

* Nội dung: ∀lh1, lh2 ∈ KHO: lh1. MALOHANG ≠ lh2. MALOHANG
* Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB5** | Thêm | Xóa | Sửa |
| KHO | + | - | + (MALOHANG) |

### Ràng buộc toàn vẹn khoá ngoại

**RB6:** Mỗi hoá đơn phải do một nhân viên tạo ra

* • Nội dung: ∀hd ∈ HOADON, ∃nv ∈ NHANVIEN: hd.MANV = nv.MANV

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB6** | Thêm | Xóa | Sửa |
| HOADON | + | - | +(MANV) |
| NHANVIEN | - | + | - |

**RB7:** Mỗi kho phải là do một nhân viên đảm nhận việc nhận hàng ở lần nhập kho đó.

• Nội dung: ∀k ∈ KHO, ∃nv ∈ NHANVIEN: k.MANV = nv.MANV

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB7** | Thêm | Xóa | Sửa |
| KHO | + | - | +(MANV) |
| NHANVIEN | - | + | - |

**RB8:** Mỗi tài khoản phải có một nhân viên quản lý tài khoản đó.

• Nội dung: ∀tk ∈ TAIKHOAN, ∃nv ∈ NHANVIEN: tk.MANV = nv.MANV

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB8** | Thêm | Xóa | Sửa |
| TAIKHOAN | + | - | +(MANV) |
| NHANVIEN | - | + | - |

**RB9:** Mỗi chi tiết kho phải thuộc một lô hàng và có một mã sản phẩm nằm trong bảng sản phẩm

• Nội dung: ∀ctk ∈ CHITIETKHO, ∃k ∈ KHO, ∃sp ∈ SANPHAM :

ctk.MALOHANG = k. MALOHANG ∧ ctk.MASP = sp. MASP

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB9** | Thêm | Xóa | Sửa |
| CHITIETKHO | + | - | +( MALOHANG, MASP) |
| KHO | - | + | - |
| SANPHAM | - | + | - |

**RB10:** Mỗi chi tiết hoá đơn phải thuộc một hoá đơn và có một nhân viên tạo ra hoá đơn đó.

• Nội dung: ∀cthd ∈ CHITIETHOADON, ∃hd ∈ HOADON, ∃nv ∈ NHANVIEN : cthd.MAHD = hd. MAHD ∧ cthd.MANV = nv.MANV

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB10** | Thêm | Xóa | Sửa |
| CHITIETKHO | + | - | +( MAHD, MANV) |
| HOADON | - | + | - |
| NHANVIEN | - | + | - |

### Ràng buộc liên thuộc tính

**RB11:** Ngày sinh của một nhân viên phải nhỏ hơn ngày vào làm của nhân viên đó.

• Nội dung: ∀nv ∈ NHANVIEN: nv.NGAYSINH < nv.NGAYVL

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB11** | Thêm | Xóa | Sửa |
| NHANVIEN | + | - | +(NGAYSINH, NGAYVL) |

### Ràng buộc miền giá trị

**RB12:** Lương của nhân viên phải nhỏ hơn hoặc bằng 25 triệu

• Nội dung: ∀nv ∈ NHANVIEN: nv.LUONG <= 25.000.000

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB12** | Thêm | Xóa | Sửa |
| NHANVIEN | + | - | +(LUONG) |

**RB13:** Số lượng sản phẩm của một sản phẩm phải lớn hơn hoặc bằng 0

• Nội dung: ∀sp ∈ SANPHAM, ∀ctk ∈ CHITIETKHO: sp.SOLUONG >= 0, ctk.SOLUONG >= 0

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB13** | Thêm | Xóa | Sửa |
| SANPHAM | + | - | +( SOLUONG) |
| CHITIETKHO | + | - | +( SOLUONG) |

**RB14:** Giá của một sản phẩm phải lớn hơn 0

• Nội dung: ∀sp ∈ SANPHAM: sp.GIASP > 0

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB14** | Thêm | Xóa | Sửa |
| SANPHAM | + | - | +( GIA) |

### Ràng buộc liên bộ, liên quan hệ

**RB15:** Số lượng tài khoản phải nhỏ hơn số lượng nhân viên

• Nội dung: ∀tk ∈ TAIKHOAN, ∀nv ∈ NHANVIEN, COUNT(tk) (MATK) <= COUNT(nv) (MANV).

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB15** | Thêm | Xóa | Sửa |
| NHANVIEN | - | + | - |
| TAIKHOAN | + | - | - |

### Ràng buộc liên thuộc tính, liên quan hệ

**RB16:** Ngày tạo hoá đơn phải lớn hơn ngày vào làm của nhân viên và ngày đăng ký của khách hàng

• Nội dung: ∀hd ∈ HOADON, ∃nv ∈ NHANVIEN, ∃kh ∈ KHACHHANG: hd.NGAYHD > (nv.NGAYVL ∧ kh.NGAYDK)

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB16** | Thêm | Xóa | Sửa |
| HOADON | + | - | +(NGAYHD) |
| NHANVIEN | - | - | +(NGAYVL) |
| KHACHHANG | - | - | +(NGAYDK) |

### Ràng buộc do thuộc tính tổng hợp

**RB17:** Số lượng một sản phẩm trong quan hệ sản phẩm sẽ bằng số lượng chi tiết kho sản phẩm đó vừa nhập vào cộng với số lượng đang có

• Nội dung: ∀sp ∈ SANPHAM:

sp.SOLUONG = (ctk ∈ CHITIETKHO: ctk.MASP = sp.MASP) (ctk.SOLUONG + sp.SOLUONG);

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB17** | Thêm | Xóa | Sửa |
| CHITIETKHO | + | + | +(SOLUONG) |
| SANPHAM | - | - | +(SOLUONG) |

**RB18:** Số lượng trong quan hệ sản phẩm sẽ giảm xuống khi một chi tiết hoá đơn được thêm vào, giá của chi tiết hoá đơn sẽ bằng số lượng sản phẩm \* giá của sản phẩm, và trị giá hoá đơn sẽ bằng trị giá hoá đơn hiện tại cộng với giá của chi tiết hoá đơn

• Nội dung: ∀cthd ∈ CHITIETHOADON, ∃hd ∈ HOADON, ∃sp ∈ SANPHAM:

sp.SOLUONG = (cthd.MASP = sp.MASP)(:sp.SOLUONG – cthd.SOLUONG)

cthd.GIA = (cthd.MASP = sp.MASP)(cthd.SOLUONG\*sp.GIA)

hd.TRIGIA = (hd.MAHD = cthd.MAHD) (hd.TRIGIA + cthd.GIA)

• Bảng tầm ảnh hưởng:

|  |  |  |  |
| --- | --- | --- | --- |
| **RB18** | Thêm | Xóa | Sửa |
| HOADON | - | - | +(TRIGIA) |
| SANPHAM | - | - | +( SOLUONG) |
| CHITIETHOADON | + | + | +(SOLUONG) |

**CHƯƠNG 2. XÂY DỰNG CÁC GIAO TÁC**

* 1. **Trigger**

## Định nghĩa trigger

Hiểu đơn giản thì Trigger là một stored procedure không có tham số. Trigger thực thi một cách tự động khi một trong ba câu lệnh Insert, Update, Delete làm thay đổi dữ liệu trên bảng có chứa trigger.

* + - **Có 2 thời điểm xảy ra trigger là trước ( BEFORE ) khi và sau**

**( AFTER )** khi thực hiện các hành động INSERT, UPDATE, DELETE.

+ Ta có câu lệnh chạy trigger trước khi INSERT, UPDATE, DELETE:

BEFORE INSERT OR UPDATE OR DELETE ON [TÊN BẢNG]

+ Ta có câu lệnh chạy trigger sau khi INSERT, UPDATE, DELETE:

AFTER INSERT OR UPDATE OR DELETE ON [TÊN BẢNG]

* + - **Trigger được sử dụng để:**

+ Thực hiện kiểm tra các ràng buộc phức tạp trong dữ liệu.

+ Có thể sử dụng trigger để tự động hoá việc tính toán cho các thuộc tính trong quan hệ.

* + - **Cú pháp tạo một trigger trong Oracle**

|  |
| --- |
| CREATE [OR REPLACE] TRIGGER tên trigger  BEFORE | AFTER INSERT OR UPDATE OR DELETE  ON tên quan hệ  FOR EACH ROW  DECLARE  --- khai báo các biến để sử dụng  BEGIN  --- các câu lệnh  EXCEPTION  --- các câu lệnh xử lý trường hợp ngoại lệ  END; |

* + - **Một vài lưu ý khi dùng trigger:**

+ Trong TRIGGER không có SELECT mà chỉ có SELECT ... INTO

+ :NEW.thuoc\_tinh và :OLD.thuoc\_tinh. Trong đó, :NEW là một biến ảo chứa các giá trị khi một hàng mới được thêm vào, cập nhật và :OLD là một biến ảo chứa các giá trị cũ trước khi hàng đó bị thay đổi, xoá đi.

+ Nếu kiểm tra không thoả điều kiện và muốn phát ra một lỗi thì dùng câu lệnh:

raise\_application\_error (-20000, ‘Thong bao loi’);

## Danh sách các trigger trong đồ án

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên Trigger** | **Thao tác** | **Bảng** | **Nội dung trigger** |
|  | Trg\_NGS\_NGVL\_LUONG\_NV | INSERT, UPDATE | NHANVIEN | - Đảm bảo ngày sinh luôn nhỏ hơn ngày vào làm của một nhân viên.  - Lương của nhân viên không quá 20 triệu |
|  | Trg\_KTRA\_SLTK | INSERT | TAIKHOAN | Số lượng tài khoản phải nhỏ hơn hoặc bằng số lượng nhân viên |
|  | Trg\_KTRA\_SL\_GIA\_SANPHAM | INSERT, UPDATE | SANPHAM | - Khi thêm một sản phẩm số lượng phải bằng không  - Khi cập nhật một sản phẩm thì số lượng phải lớn hơn không  - Và giá của một sản phẩm phải lớn hơn không |
|  | Trg\_AUTO\_TANG\_GIAM\_SL\_SANPHAM\_NHAPKHO | UPDATE, INSERT, DELETE | CHITIETKHO | - Khi thêm hoặc cập nhật một chi tiết kho: số lượng sản phẩm phải > 0  - Khi thêm một chi tiết kho thì số lượng của sản phẩm có mã sản phẩm tương ứng trong bảng sản phẩm phải tăng lên tương ứng với số lượng chi tiết kho nhập vào.  - Khi cập nhật một sản phẩm thì số lượng sẽ tăng giảm trong bảng sản phẩm, tuỳ theo số lượng cập nhật  - Khi xoá một sản phẩm thì số lượng trong bảng sản phẩm sẽ giảm xuống. |
|  | Trg\_AUTO\_GIAM\_SL\_SANPHAM\_TINHGIA\_TRIGIA\_HD | INSERT | CHITIETHOADON | - Khi thêm một chi tiết hoá đơn  + số lượng sản phẩm có mã sản phẩm tương ứng trong bảng sản phẩm sẽ giảm xuống ứng với số lượng chi tiết hoá đơn đó  + giá chi tiết hoá đơn sẽ được tự động tính bằng giá sản phẩm nhân số lượng sản phẩm trong chi tiết hoá đơn  + tự động tăng giá trị hoá đơn = giá trị hoá đơn hiện tại cộng với giá trị chi tiết hoá đơn mới thêm vào |
|  | Trg\_ KTRA\_NGHD\_NGVL\_NV | INSERT | HOADON | - Khi thêm một hoá đơn thì phải kiểm tra ngày vào làm của nhân viên và ngày đăng ký của khách hàng có lớn hơn ngày hoá đơn hay không |

## Mô tả một số trigger tiêu biểu

### Trigger trên bảng nhân viên

**Tên trigger:** Trg\_NGS\_NGVL\_LUONG\_NV

**Diễn giải:** Khi thêm hoặc cập nhật một nhân viên thì sẽ kiểm tra lương của nhân viên phải nhỏ hơn 20 triệu và ngày vào làm phải lớn hơn ngày sinh

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_NGS\_NGVL\_LUONG\_NV  BEFORE INSERT OR UPDATE ON NHANVIEN  FOR EACH ROW  DECLARE  BEGIN  IF(:NEW.LUONG > 20000000 AND :NEW.CHUCVU = 'Nhân Viên') THEN RAISE\_APPLICATION\_ERROR(-20000, 'Luong cua nhan vien khong duoc lon hon 20 trieu'); END IF;  IF(:NEW.NGAYSINH > :NEW.NGAYVL) THEN RAISE\_APPLICATION\_ERROR(-20000, 'Ngay sinh cua nhan vien phai < ngay vao lam'); END IF;  END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_NGS\_NGVL\_LUONG\_NV , để trường hợp thực hiện trigger này là trước khi thêm và cập nhật cho nhân viên.
2. Kiểm tra thuộc tính lương, nếu lương lớn hơn 20 triệu thì sẽ báo lỗi.
3. Kiểm tra ngày sinh và ngày vào làm của nhân viên, nếu ngày sinh lớn hơn thì sẽ báo lỗi và kết thúc trigger.

### Trigger trên bảng tài khoản

**Tên trigger:** Trg\_KTRA\_SLTK

**Diễn giải:** Khi thêm một tài khoản thì sẽ kiểm tra số lượng nhân viên so với số lượng tài khoản, nếu số lượng tài khoản lớn hơn số lượng nhân viên thì sẽ báo lỗi.

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_KTRA\_SLTK  BEFORE INSERT ON TAIKHOAN  FOR EACH ROW  DECLARE  SLTK INT;  SLNV INT;  BEGIN  SELECT COUNT(USERNAME) INTO SLTK FROM TAIKHOAN;  SELECT COUNT(MANV) INTO SLNV FROM NHANVIEN;  IF(SLTK > SLNV) THEN RAISE\_APPLICATION\_ERROR(-20000, 'So Luong tai khoan phai be hon so luong nhan vien'); END IF;  END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_KTRA\_SLTK, để trường hợp thực hiện trigger này là trước khi thêm một tài khoản. Tạo hai biến SLTK và SLNV.
2. Đếm số lượng tài khoản và số lượng nhân viên sau đó gán giá trị đó lần lượt vào hai biến SLTK và SLNV
3. Kiểm tra SLTK và SLNV, nếu SLTK lớn hơn SLNV thì sẽ báo lỗi

### Trigger trên bảng sản phẩm

**Tên trigger:** Trg\_KTRA\_SL\_GIA\_SANPHAM

**Diễn giải:** Khi thêm hoặc cập nhật một sản phẩm thì sẽ kiểm tra giá của một sản phẩm, đảm bảo giá phải lớn hơn không. Khi thêm một sản phẩm thì thuộc tính số lượng phải bằng không. Khi cập nhật một sản phẩm gì giá nó phải lớn hơn không

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_KTRA\_SL\_GIA\_SANPHAM  BEFORE INSERT OR UPDATE ON SANPHAM  FOR EACH ROW  BEGIN  IF(INSERTING AND :NEW.SOLUONG != 0) THEN RAISE\_APPLICATION\_ERROR(-20000, 'SO LUONG SAN PHAM phai = 0, VUI LONG NHAP LAI');  ELSIF(UPDATING AND :NEW.SOLUONG < 0 ) THEN RAISE\_APPLICATION\_ERROR(-20000, 'SO LUONG SAN PHAM khong duoc < 0, VUI LONG NHAP LAI');  END IF;  IF :NEW.GIASP < 1 THEN RAISE\_APPLICATION\_ERROR(-20000, 'GIA SAN PHAM KHONG DUOC < 1');  END IF;  END; END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_KTRA\_SL\_GIA\_SANPHAM, để trường hợp thực hiện trigger này là trước khi thêm hoặc cập nhật một sản phẩm.
2. Kiểm tra đối với trường hợp thêm thì số lượng có bằng không hay khác không. Nếu khác không thì sẽ báo lỗi.
3. Kiểm tra đối với trường hợp cập nhật thì số lượng lớn hơn hay bé hơn không. Nếu bé hơn không thì sẽ báo lỗi.
4. Kiểm tra giá của một sản phẩm, đảm bảo giá phải lớn hơn 0, nếu bé hơn 1 thì sẽ báo lỗi

### Trigger trên bảng chi tiết kho

**Tên trigger:** Trg\_AUTO\_TANG\_GIAM\_SL\_SANPHAM\_NHAPKHO

**Diễn giải:** Khi thêm, xoá, cập nhật một chi tiết kho thì sẽ tự động tăng, giảm số lượng trong bảng sản phẩm tương ứng với số lượng chi tiết kho.

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_AUTO\_TANG\_GIAM\_SL\_SANPHAM\_NHAPKHO  AFTER INSERT OR UPDATE OR DELETE ON CHITIETKHO  FOR EACH ROW  DECLARE  SLSP SANPHAM.SOLUONG%TYPE;  TONG SANPHAM.SOLUONG%TYPE;  BEGIN  IF(:NEW.SOLUONG < 0) THEN RAISE\_APPLICATION\_ERROR(-20000, 'SO LUONG SAN PHAM < 0, VUI LONG NHAP LAI');  END IF;  IF INSERTING THEN  SELECT SOLUONG INTO SLSP FROM SANPHAM WHERE :NEW.MASP = SANPHAM.MASP;  UPDATE SANPHAM SET SOLUONG = SLSP + :NEW.SOLUONG WHERE :NEW.MASP = SANPHAM.MASP;  ELSIF UPDATING THEN  SELECT SOLUONG INTO SLSP FROM SANPHAM WHERE :NEW.MASP = SANPHAM.MASP;  TONG := SLSP - :OLD.SOLUONG;  TONG := TONG + :NEW.SOLUONG;  UPDATE SANPHAM SET SOLUONG = TONG WHERE :NEW.MASP = MASP;  ELSIF DELETING THEN  SELECT SOLUONG INTO SLSP FROM SANPHAM WHERE :OLD.MASP = SANPHAM.MASP;  UPDATE SANPHAM SET SOLUONG = SLSP - :OLD.SOLUONG WHERE :OLD.MASP = MASP;  END IF;  END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_AUTO\_TANG\_GIAM\_SL\_SANPHAM\_NHAPKHO, để trường hợp thực hiện trigger này là sau khi thêm, xoá, cập nhật một chi tiết kho. Tạo hai biến SLSP(Số lượng sản phẩm) và TONG (tổng)
2. Kiểm tra số lượng nhập vào. Nếu số lượng bé hơn 0 thì sẽ báo lỗi.
3. Đối với trường hợp thêm một chi tiết kho, thì sẽ lấy số lượng sản phẩm (có mã sản phẩm tương ứng với mã sản phẩm chi tiết kho) hiện tại trong bảng sản phẩm gán vào biến SLSP. Sau đó, cập nhật lại số lượng sản phẩm trong bảng sản phẩm với số lượng bằng SLSP + số lượng sản phẩm chi tiết kho mới thêm vào.
4. Đối với trường hợp cập nhật một chi tiết kho, thì sẽ lấy số lượng sản phẩm (có mã sản phẩm tương ứng với mã sản phẩm chi tiết kho) hiện tại trong bảng sản phẩm gán vào biến SLSP. Sau đó, gán giá trị cho biến TONG bằng số lượng sản phẩm trong bảng sản phẩm, trừ đi số lượng trước khi cập nhật trong bảng chi tiết kho và cộng với số lượng mới cập nhật vào bảng chi tiết kho. Cuối cùng là cập nhật lại số lượng trong bảng sản phẩm, SOLUONG = TONG.
5. Đối với trường hợp xoá một chi tiết kho, thì sẽ lấy số lượng sản phẩm trong bảng sản phẩm trừ đi số lượng của chi tiết kho bị xoá.

### Trigger trên bảng chi tiết hoá đơn

**Tên trigger:** Trg\_AUTO\_GIAM\_SL\_SANPHAM\_TINHGIA\_TRIGIA\_HD

**Diễn giải:** Khi thêm một chi tiết hoá đơn thì sẽ tự động giảm số lượng trong bảng sản phẩm tương ứng với số lượng chi tiết hoá đơn, giá của một chi tiết hoá đơn sẽ được tính bằng số lượng sản phẩm nhân với giá của sản phẩm, và trị giá của hoá đơn sẽ bằng trị giá hiện tại của hoá đơn cộng với giá của chi tiết hoá đơn vừa thêm vào.

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_AUTO\_GIAM\_SL\_SANPHAM\_TINHGIA\_TRIGIA\_HD  BEFORE INSERT ON CHITIETHOADON  FOR EACH ROW  DECLARE  SLSPBD SANPHAM.SOLUONG%TYPE;  GIA\_MASP SANPHAM.GIASP%TYPE;  TG HOADON.TRIGIA%TYPE;  BEGIN  SELECT SOLUONG, GIASP INTO SLSPBD, GIA\_MASP FROM SANPHAM WHERE :NEW.MASP = SANPHAM.MASP;  SELECT TRIGIA INTO TG FROM HOADON WHERE HOADON.MAHD = :NEW.MAHD;    UPDATE SANPHAM SET SOLUONG = SLSPBD - :NEW.SOLUONG WHERE :NEW.MASP = SANPHAM.MASP;    :NEW.GIA := :NEW.SOLUONG \* GIA\_MASP;  UPDATE HOADON SET TRIGIA = TG + :NEW.GIA WHERE :NEW.MAHD = HOADON.MAHD;  END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_AUTO\_GIAM\_SL\_SANPHAM\_TINHGIA\_TRIGIA\_HD, để trường hợp thực hiện trigger này trước khi thêm một chi tiết hoá đơn. Tạo ba biến SLSPBD (Số lượng sản phẩm ban đầu) và GIA\_MASP (giá của sản phẩm) và TG(trị giá)
2. Gán số lượng, giá sản phẩm, trị giá hoá đơn lần lượt vào các biến SLSPBD, GIA\_MASP, TG.
3. Tính giá cho chi tiết hoá đơn bằng số lượng nhân với giá của sản phẩm.
4. Cập nhật trị giá của hoá đơn bằng trị giá ban đầu cộng với giá của chi tiết hoá đơn. Kết thúc trigger

### Trigger trên bảng hoá đơn.

**Tên trigger:** Trg\_ KTRA\_NGHD\_NGVL\_NV

**Diễn giải:** Khi thêm một hoá đơn , trigger này sẽ kiểm tra sự hợp lệ của ngày mua, đảm bảo rằng ngày mua lớn hơn ngày vào làm của nhân viên và ngày đăng ký của khách hàng.

**Mã lệnh PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE TRIGGER Trg\_ KTRA\_NGHD\_NGVL\_NV  BEFORE INSERT ON HOADON  FOR EACH ROW  DECLARE  NGAYVLAM DATE;  NGDK DATE;  BEGIN  SELECT NGAYVL INTO NGAYVLAM FROM NHANVIEN WHERE :NEW.MANV = MANV;  SELECT NGAYDK INTO NGDK FROM KHACHHANG WHERE :NEW.MAKH = MAKH;  IF(NGAYVLAM > :NEW.NGAYMUA ) THEN RAISE\_APPLICATION\_ERROR(-20000, 'TAI SAO NHAN VIEN CO THE NHAP HOA DON TRUOC KHI DI LAM'); END IF;  IF(NGDK > :NEW.NGAYMUA) THEN RAISE\_APPLICATION\_ERROR(-20000, 'NGAY MUA PHAI LON HON NGAYDK CUA KHACHHANG '); END IF;  END; |

**Các bước thực hiện:**

1. Tạo tên cho trigger Trg\_ KTRA\_NGHD\_NGVL\_NV, để trường hợp thực hiện trigger này trước khi thêm một hoá đơn. Tạo hai biến NGAYVLAM (ngày vào làm của nhân viên) và NGDK (ngày đăng ký của khách hàng).
2. Gán ngày vào làm của nhân viên và ngày đăng ký của khách hàng vào 2 biến NGAYVLAM và NGDK.
3. Kiểm tra tính hợp lệ của ngày mua, nếu ngày mua lớn hơn ngày vào làm và ngày đăng ký là hợp lệ, không thì sẽ báo lỗi. Kết thúc trigger

# Stored Procedure

## Định nghĩa Stored Procedure

Procedure hay còn gọi là thủ tục, nó dùng để gom một nhóm lệnh SQL cùng xử lý một mục đích cụ thẻ nào đó, sau đó đặt cho nó một cái tên và khai bao tham số truyền vào để sau này mỗi khi cần sử dụng thì ta chỉ cần gọi tên và truyền tham số là được.

* Procedure là một khối độc lập của một chương trình có thể được lưu trữ trong cơ sở dữ liệu
* Gọi đến các Procedure này có thể được thực hiện bằng cách đề cập đến tên của chúng.
* Nó chủ yếu được sử dụng để thực hiện một quá trình bên trong PL/SQL.
* Nó có thể có khối lồng nhau, hoặc nó có thể được xác định và lồng vào bên trong các khối hoặc các gói khác.
* Các giá trị có thể được thông qua vào các thủ tục hoặc lấy từ các thủ tục thông qua các tham số.
* Những thông số cần được đưa vào khai báo để sử dụng.
* **Cú pháp tạo một stored procedure trong oracle.**

|  |
| --- |
| CREATE [OR REPLACE] PROCEDURE “tên procedure”(“tên tham số một” IN/OUT “kiểu dữ liệu của tham số 1”, “tên tham số hai” IN/OUT “kiểu dữ liệu của tham số 2”,…)  IS  --- khai báo các biến để sử dụng  BEGIN  --- các câu lệnh  EXCEPTION  WHEN  --- các câu lệnh xử lý trường hợp ngoại lệ  END; |

* **Một số lưu ý khi viết procedure**

• IN: Đây là tham số mặc định, tham số này là sẽ là dữ liệu truyền vào.

• OUT: Tham số này sẽ là dữ liệu truyền ra.

• IN OUT: Là tham số đặc, vừa là tham số đầu vào vừa là tham số đầu ra, và thường giá trị ban đầu bị thay đổi.

## Danh sách các Stored Procedure có trong đồ án

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên** | **Tham số đầu vào** | **Tham số đầu ra** | **Ý nghĩa** |
|  | INSERT\_SANPHAM | ma in sanpham.masp%type, ten in sanpham.tensp%type, gia in sanpham.giasp%type, sl in sanpham.soluong%type, pl in sanpham.phanloai%type, nsx in sanpham.nsx%type, hsd in sanpham.hsd%type, mt in sanpham.mota%type |  | Thêm sản phẩm |
|  | UPDATE\_SANPHAM | ma in sanpham.masp%type, ten in sanpham.tensp%type, gia in sanpham.giasp%type, sl in sanpham.soluong%type, pl in sanpham.phanloai%type, nsxuat in sanpham.nsx%type, hsdung in sanpham.hsd%type, mt in sanpham.mota%type |  | Cập nhật sản phẩm |
|  | DELETE\_SANPHAM | ma in sanpham.masp%type |  | Xoá sản phẩm |
|  | SEARCH\_SANPHAM | text\_search in nvarchar2 | dssp out sys\_refcursor | Tìm kiếm sản phẩm |
|  | DOANHTHU | fr\_date in date,  end\_date in date, | tong out number | Tính doanh thu |
|  | INSERT\_KHACHHANG | ma in khachhang.makh%type,  ten in khachhang.tenkh%type,  sdt in khachhang.sdtkh%type |  | Thêm khách hàng |
|  | UPDATE\_KHACHHANG | ma in khachhang.makh%type,  ten in khachhang.tenkh%type,  sdt in khachhang.sdtkh%type |  | Cập nhật khách hàng |
|  | DELETE\_KHACHHANG | ma in khachhang.makh%type |  | Xóa khách hàng |
|  | SEARCH\_KHACHHANG | text\_search in nvarchar2 | dskh out sys\_refcursor | Tìm kiếm khách hàng |
|  | INSERT\_HOADON | mhd in hoadon.mahd%type, ngm in hoadon.ngaymua%type mnv in nhanvien.manv%type, mkh in khachhang.makh%type, h\_thuc in hoadon.hinhthuc%type |  | Tạo hóa đơn |
|  | SEARCH\_HOADON | text\_search in nvarchar2 | dshd out sys\_refcursor | Tìm kiếm hóa đơn |
|  | INSERT\_KHO\_LOHANG | mlh in kho.malohang%type, mnv in nhanvien.manv%type, tncc in kho.tennhacungcap%type |  | Thêm lô hàng vào quan hệ kho |
|  | UPDATE\_LOHANG | mlh in kho.malohang%type, tncc in kho.tennhacungcap%type, mnv in nhanvien.manv%type |  | Cập nhật lô hàng trong quan hệ kho |
|  | DELETE\_LOHANG | mlh in kho.malohang%type |  | Xoá lô hàng |
|  | SEARCH\_LOHANG | Text\_search in nvarchar2 | dslh out sys\_refcursor | Tìm kiếm lô hàng |
|  | INSERT\_CHITIETKHO | malh1 in chitietkho.malohang%type,  masp1 in sanpham.masp%type,  soluong in chitietkho.soluong%type |  | Thêm chi tiết kho |
|  | UPDATE\_CTK | malh in chitietkho.malohang%type, sl in chitietkho.soluong%type |  | Cập nhật số lượng chi tiết kho |
|  | DELETE\_CTK | mlh in kho.malohang%type, msp in sanpham.masp%type |  | Xoá một chi tiết kho |
|  | INSERT\_NV | mnv in nhanvien.manv%type, ten in nhanvien.tennv%type, sdt in nhanvien.sdtnv%type, ns in nhanvien.ngaysinh%type, cccd in nhanvien.cccd%type, dc in nhanvien.diachi%type, cv in nhanvien.chucvu%type, l in nhanvien.luong%type, nvl in nhanvien.ngayvl%type |  | Thêm nhân viên |
|  | DELETE\_NV | mnv in nhanvien.manv%type |  | Xóa nhân viên |
|  | UPDATE\_NV | mnv in nhanvien.manv%type, ten in nhanvien.tennv%type, sdt in nhanvien.sdtnv%type, ns in nhanvien.ngaysinh%type, cccdi in nhanvien.cccd%type, dc in nhanvien.diachi%type, cv in nhanvien.chucvu%type, l in nhanvien.luong%type, nvl in nhanvien.ngayvl%type |  | Cập nhật nhân viên |
|  | SEARCH\_NHANVIEN | text\_search in nvarchar2 | dsnv out sys\_refcursor | Tìm kiếm nhân viên |
|  | INSERT\_CHITIETHOADON | mhd in hoadon.mahd%type, msp in sanpham.masp%type, sl in chitiethoadon.soluong%type |  | Thêm một chi tiết hoá đơn |
|  | SEARCH\_CTHD | text\_search in nvarchar2 | dscthd out sys\_refcursor | Tìm kiếm chi tiết hoá đơn |
|  | INSERT\_TAIKHOAN | usr in taikhoan.username%type, pass in taikhoan.password%type, mnv in taikhoan.manv%type |  | Thêm một tài khoản |
|  | DELETE\_TK | mtk in taikhoan.username%type |  | Xoá tài khoản |
|  | SEARCH\_TK | mtk in taikhoan.username%type | dstk out sys\_refcursor | Tìm kiếm tài khoản |
|  | SLKH\_MUAHANG | fr\_date in date, end\_date in date | tong out number | Thống kê số lượng khách hàng mua hàng |
|  | TOP10SP\_BANCHAY | fr\_date in date, end\_date in date | top10 out sys\_refcursor | Thống kê 10 sản phẩm bán chạy nhất |

## Một số Stored procedure tiêu biểu

### Thêm khách hàng

* **Tên store procedure:** INSERT\_KHACHHANG
* **Tham số truyền vào:** ma in khachhang.makh%type, ten in khachhang.tenkh%type, sdt in khachhang.sdtkh%type
* **Ý nghĩa:** thêm một khách hàng mới vào cơ sở dữ liệu
* **Các bước thực hiện:**
  + Nhập vào các tham số cần thiết cho procedure, và tạo một biến LUUSDT có kiểu dữ liệu giống kiểu dữ liệu của KHACHHANG.SDTKH và gán cho nó một giá trị ‘TONTAI’
  + Tạo một khối lệnh lồng trong stored procedure hiện tại, câu lệnh đầu tiên trong khối này là kiểm tra xem có SDTKH nào trùng với SDT vừa nhập vào hay không. Nếu có thì sẽ lưu vào LUUSDT, không thì sẽ chạy vào EXCEPTION NO\_DATA\_FOUND và INSERT KHACHHANG. Kết thúc khối lệnh
  + Ngoài lệnh lồng sẽ có một câu lệnh kiểm tra xem LUUSDT có khác với giá trị ban đầu là ‘KHONGTONTAI’ hay không, nếu khác thì đồng nghĩa là đã có một SDTKH tồn tại trong quan hệ khách hàng được gán vào đó và nó sẽ báo lỗi khách hàng đã tồn tại.
* **Mã PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE PROCEDURE INSERT\_KHACHHANG(  ma in khachhang.makh%type,  ten in khachhang.tenkh%type,  sdt in khachhang.sdtkh%type )  IS  LUUSDT KHACHHANG.SDTKH%TYPE := 'KHONGTONTAI';  BEGIN  BEGIN  SELECT SDTKH INTO LUUSDT FROM KHACHHANG WHERE SDTKH = SDT;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  INSERT INTO KHACHHANG VALUES(MA, SDT, TEN , SYSDATE);  END;  IF LUUSDT != ' KHONGTONTAI ' THEN  RAISE\_APPLICATION\_ERROR(-20000, 'SDT KH NAY DA TON TAI');  END IF;  END; |

### Thêm chi tiết kho

**- Tên store procedure:** THEM\_CHITIETKHO

**- Tham số truyền vào:**

malh1 in chitietkho.malohang%type,

masp1 in sanpham.masp%type,

soluong in chitietkho.soluong%type

**- Ý nghĩa:** thêm vào một chi tiết kho

**- Các bước thực hiện:**

* + Nhập vào các tham số cần thiết cho procedure
  + Kiểm tra xem mã lô hàng và mã sản phẩm có tồn tại trong quan hệ kho và sản phẩm hay không, nếu không tồn tại thì sẽ báo lỗi.
  + Tồn tại thì sẽ bắt đầu insert chi tiết kho
* **Mã PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE PROCEDURE INSERT\_CHITIETKHO(MALH1 IN CHITIETKHO.MALOHANG%TYPE, MASP1 IN SANPHAM.MASP%TYPE, SOLUONG IN CHITIETKHO.SOLUONG%TYPE)  IS  CHECK\_MLH CHITIETKHO.MALOHANG%TYPE := 'kotontai';  CHECK\_MSP SANPHAM.MASP%TYPE := 'kotontai';  BEGIN  BEGIN  SELECT MALOHANG INTO CHECK\_MLH FROM KHO WHERE MALOHANG = MALH1;  SELECT MASP INTO CHECK\_MSP FROM SANPHAM WHERE MASP = MASP1;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  IF(CHECK\_MLH = 'kotontai') THEN RAISE\_APPLICATION\_ERROR(-20000, 'MALOHANG BAN NHAP VAO KHONG DUNG');  ELSIF (CHECK\_MSP = 'kotontai') THEN RAISE\_APPLICATION\_ERROR(-20000, 'MASP BAN NHAP VAO KHONG DUNG');  END IF;  END;  INSERT INTO CHITIETKHO VALUES(MALH1, MASP1, SOLUONG);  END; |

### Lấy thông tin TOP 10 sản phẩm bán chạy nhất

**- Tên store procedure:** TOP10SP\_BANCHAY

**- Tham số truyền vào:**

fr\_date in date, end\_date in date, top10 out sys\_refcursor

**- Ý nghĩa:** lấy thông tin top 10 sản phẩm bán chạy nhất trong một khoảng thời gian

**- Các bước thực hiện:**

* + Nhập vào các tham số cần thiết cho procedure
  + Tìm top 10 với câu lệnh select
  + Xuất trả thông tin ra con trỏ top10
* **Mã PL/SQL:**

|  |
| --- |
| CREATE OR REPLACE PROCEDURE TOP10SP\_BANCHAY(FR\_DATE IN DATE, END\_DATE IN DATE, TOP10 OUT SYS\_REFCURSOR)  IS  BEGIN  OPEN TOP10  FOR SELECT SANPHAM.MASP, TENSP, SUM(CHITIETHOADON.SOLUONG)  FROM HOADON, CHITIETHOADON, SANPHAM  WHERE CHITIETHOADON.MAHD = HOADON.MAHD AND SANPHAM.MASP = CHITIETHOADON.MASP  AND NgayMua BETWEEN FR\_DATE AND END\_DATE  GROUP BY SANPHAM.MASP, TENSP  ORDER BY SUM(CHITIETHOADON.SOLUONG) DESC  FETCH FIRST 10 ROWS ONLY;  CLOSE TOP10;  END; |

**CHƯƠNG 3. XỬ LÝ GIAO TÁC ĐỒNG THỜI**

# Các mức cô lập trong Oracle

## Read commited

Trong Read Committed, mọi truy vấn được thực hiện bởi một transaction chỉ nhìn thấy dữ liệu đã được commit. Mức cô lập này là mặc định. Nó phù hợp với môi trường cơ sở dữ liệu trong đó ít transaction có khả năng xung đột.

* Xung đột ghi trong Read Committed:

• Trong Read Committed, một xung đột ghi xảy ra khi transaction cố gắng thay đổi một dòng đã được cập nhật bởi một transaction khác mà chưa được commit hay rollback.

• Transaction ngăn chặn sửa đổi dòng được gọi là blocking transaction.Transaction Read Committed chờ blocking transaction kết thúc và giải phóng khóa rồi transaction Read Committed mới lấy được khóa và thực hiện thay đổi.

## Serializable

Ở mức cô lập tuần tự (serializable isolation level), transaction chỉ thấy những thay đổi đã được commit trước thời điểm transaction có mức cô lập tuần tự này bắt đầu và các thay đổi được thực hiện bởi chính transaction này.

* Một transaction tuần tự hoạt động trong một môi trường chỉ một mình nó như thể không có người dùng nào khác đang sửa đổi dữ liệu trong cơ sở dữ liệu. mức cô lập tuần tự phù hợp với các môi trường:

• Với cơ sở dữ liệu lớn và các transaction ngắn chỉ cập nhật một vài dòng.

• Trường hợp có 2 transaction đồng thời sửa đổi cùng 1 dòng là tương đối thấp.

• Trường hợp các transaction tương đối dài chủ yếu chỉ đọc

* Ở mức cô lập tuần tự, Bất kỳ dòng nào được đọc bởi transaction tuần tự được đảm bảo giống nhau khi đọc lại. Bất kỳ truy vấn nào cũng được đảm bảo trả về cùng kết quả trong suốt thời gian transaction đó hoạt động, do đó, những thay đổi được thực hiện bởi transaction khác sẽ không hiển thị đối với truy vấn của transaction tuần tự bất kể những thay đổi của transaction khác đã chạy được bao lâu. Transaction tuần tự (serializable) ngăn ngừa được Dirty reads, Nonrepeatable reads, Phantom reads.
* Trong cơ sở dữ liệu Oracle, nếu transaction Tuần tự thực hiện thay đổi 1 dòng nào đó mà dòng này đã được thay đổi bởi một transaction khác đã commit sau khi transaction tuần tự bắt đầu thực thi thì sẽ xuất hiện 1 lỗi đó là: ORA-08177: can't serialize access for this transaction

# Cơ chế khoá

## Định nghĩa

Cơ chế khoá là một phần quan trọng trong quản lý giao dịch trong cơ sở dữ liệu. Nó được sử dụng để đảm bảo tính nhất quán và đồng thời xử lý các vấn đề giữa các giao dịch đồng thời.

## Các loại khoá

Hệ quản trị có thể thực thi một vài cơ chế khóa, tùy vào các thao tác khác nhau lên dữ liệu. Nhìn chung, hệ quản trị sử dụng 2 loại chính: exclusive locks (khóa độc quyền) và share locks (khóa chia sẻ). Tại một thời điểm, chỉ có 1 excusive lock được thực thi trên 1 đơn vị dữ liệu, nhưng nhiều share locks có thể cùng thực thi trên 1 đơn vị dữ liệu.

## Cơ chế khóa tác động đến sự tương tác giữa việc đọc và ghi dữ liệu

* Một dòng dữ liệu chỉ bị khóa khi nó đang được thay đổi bởi việc hành động ghi. Khi có 1 câu truy vấn cập nhật dữ liệu của 1 dòng, hệ quản trị sẽ khóa dòng đó lại.
* Một hành động ghi lên 1 dòng dữ liệu sẽ ngăn chặn các hành động khác cùng đồng thời ghi lên dòng dữ liệu đó.
* Hành động đọc dữ liệu không ngăn cản hành động ghi: Vì hành động đọc không khóa đơn vị dữ liệu nó đang đọc, nên hành động ghi có thể thao tác lên đơn vị dữ liệu này. Chỉ có một trường hợp ngoại lệ là câu lệnh Select … for update sẽ lock đơn vị dữ liệu nó đang đọc.
* Hành động ghi dữ liệu không ngăn cản hành động đọc: Khi một đơn vị dữ liệu đang được thao tác bởi hành động ghi, hệ quản trị sẽ sử dụng phiên bản trước đó của đơn vị dữliệu để trả về cho hành động đọc.
* Cơ chế khoá giúp đảm bảo tính nhất quán dữ liệu trong môi trường đa giao dịch. Nó đảm bảo rằng các giao dịch được thực hiện một cách an toàn và không gây xung đột khi truy cập vào cùng một tài nguyên cùng một lúc. Tuy nhiên, quá nhiều khoá có thể dẫn đến hiện tượng deadlock khi các giao dịch chờ đợi lẫn nhau vô hạn. Do đó, việc thiết kế và tinh chỉnh cơ chế khoá là rất quan trọng để đảm bảo hiệu suất và tính sẵn sàng của cơ sở dữ liệu.
* Sử dụng chế độ khóa cấp càng thấp, dữ liệu có thể được truy cập bởi càng nhiều user. Ngược lại, nếu sử dụng mức khóa càng cao, dữ liệu càng hạn chế người truy cập.

## Các chế độ khoá trong Oracle

* Hệ quản trị Oracle sử dụng 2 chế độ khóa chính cho cơ sở dữ liệu nhiều người dùng:

• Exclusive lock mode: Transaction nhận được exclusive lock khi nó có nhu cầu thay đổi dữ liệu. Các transaction khác sẽ không thể thao tác lên đơn vị dữ liệu này cho đến khi transaction đang giữ exclusive lock nhả khóa.

• Share lock mode: Chế độ này cho phép chia sẻ cùng 1 đơn vị dữ liệu. Nhiều giao tác đọc có thể chia sẻ dữ liệu cho nhau, mỗi giao tác giữ một share lock để ngăn hành động ghi vào cùng thời điểm của các giao tác ghi. Nhiều giao tác có thể được cấp share lock trên cùng đơn vị dữ liệu.

# Transaction

## Định nghĩa

Transaction là một tiến trình xử lý có xác định điểm đầu và điểm cuối, được chia nhỏ thành các operation (phép thực thi) , tiến trình được thực thi một cách tuần tự và độc lập các operation đó theo nguyên tắc hoặc tất cả đều thành công hoặc một operation thất bại thì toàn bộ tiến trình thất bại. Nếu việc thực thi một operation nào đó bị fail (hỏng) đồng nghĩa với việc dữ liệu phải rollback (trở lại) trạng thái ban đầu.

## Các tính chất của transaction

Một transaction đòi hỏi phải có 4 tính chất ACID. ACID là viết tắt của cụm từ Atomicity (nguyên tử), Consitency (nhất quán), Isolation (Cô lập), và Durability (Lâu bền).

* Atomicity: Mọi thay đổi về mặt dữ liệu phải được thục hiện trọn vẹn khi transaction thực hiện thành công hoặc không có bất kì sự thay đổi nào về mặt dữ liệu nếu có xẩy ra sự cố.
* Consistency: Sau khi một transaction kết thúc thì tất cả dữ liệu phải được nhất quán dù thành công hay thất bại.
* Isolation: Các transaction khi đông thời thực thi trên hệ thống thì không có bất kì ảnh hưởng gì tời nhau.
* Durability: Sau khi một transaction thành công thì tác dụng mà nó tạo ra phải bền vững trong cơ sở dữ liệu cho dù hệ thống có xẩy ra lỗi.

## Cấu trúc của một transaction

* Một transaction gồm 1 hay nhiều câu truy vấn, có thể bao gồm DDL,DML.
* Có bắt đầu và kết thúc.
* **Bắt đầu transaction:**

▪ Transaction bắt đầu khi câu lệnh SQL đầu tiên được thực thi, bao gồm DDL,DML hoặc lệnh SET TRANSACTION.

▪ TRANSACTION NAME: Đặt tên cho transaction, bắt đầu transasction.

▪ SET TRANSACTION NAME <Tên transaction>

▪ Khi 1 transaction mới bắt đầu , hệ quản trị oracle sẽ gán nó vào **undo data segment** ( ghi nhận lại các thao tác của transaction trước khi commit, để có thể rollback khi có lỗi).

* **Kết thúc transaction:**

▪ Transaction có thể kết thúc trong nhiều trường hợp:

• Gặp lệnh COMMIT hoặc ROLLBACK mà không có savepoint.

• Gặp các câu lệnh DDL như create, drop, rename, alter.

• User ngắt kết nối đến hệ quản trị đột ngột, transaction sẽ tự động commit.

• Các ứng dụng đang kết nối đến hệ quản trị bị dừng đột ngột,

transaction sẽ tự động rollback.

• Một transaction đang thực thi là 1 transaction đã bắt đầu nhưng chưa được COMMIT hoặc ROLLBACK.

|  |
| --- |
| BEGIN  SET TRANSACTION NAME ‘UPDATE\_GIASP’;  UPDATE SANPHAM SET GIASP = 10000 WHERE MASP = ‘SP003’;  COMMIT; -- OR ROLLBACK  END; |

## Transaction controll

Gồm các lệnh để quản lý sự thay đổi của DML lên database, gồm 1 số lệnh chính:

* SET TRANSACTION ISOLATION [SERIALIZABLE | READ COMMITED]. Trong Oracle chỉ có 2 mức cô lập này.
* SAVEPOINT: Xác định 1 điểm trong transaction để rollback về khi có sự cố.
* COMMIT: Kết thúc transaction, lưu thay đổi vĩnh viễn, xóa tất cả SAVEPOINT, mở transaction locks.
* ROLLBACK: phục hồi lại dữ liệu trước khi thay đổi. Mở tất cả khoá, xoá toàn bộ checkpoint. Kết thúc một transaction.
* ROLLBACK TO SAVEPOINT: Việc ROLLBACK lại Savepoint trong transaction chưa được COMMIT có nghĩa là hoàn tác lại mọi thay đổi được thực hiện sau Savepoint , điều này không có nghĩa là sẽ Rollback lại toàn bộ Transaction.

|  |
| --- |
| BEGIN  SET TRANSACTION NAME ‘UPDATE\_GIASP3’;  SET TRANSACTION ISOLATION READ COMMITTED    UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP003’;  SAVEPOINT sp1;  UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP004’;  SAVEPOINT sp2;  UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP005’;  SAVEPOINT sp3;  COMMIT; -- OR ROLLBACK  EXCEPTION WHEN DUP\_VAL\_ON\_INDEX  THEN  ROLLBACK;  dbms\_output.put\_line(‘co loi DUP\_VAL’);  END; |

## Hàng đợi transaction

Transaction đang đợi tài nguyên bị khóa sẽ bị block, nó sẽ xếp vào hàng đợi của transaction đang giữ tài nguyên đó. Transaction đang giữ tài nguyên phải commit hoặc rollback để transaction bị khóa tiếp tục thực thi.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Session 1** | **Session 2** | **Giải thích** |
|  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP004’; |  | Session 1 bắt đầu giao tác cập nhật giá sản phẩm của mã sản phẩm ‘SP004’. Khoá dòng ‘SP004’ |
|  | SAVEPOINT sp1; |  | Session 1 tạo một savepoint có tên sp1 |
|  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP005’; |  | Session 1 tiếp tục cập nhật giá sản phẩm của mã sản phẩm ‘SP005’. Khoá dòng ‘SP005’ |
|  |  | UPDATE SANPHAM SET GIASP = 20000 WHERE MASP = ‘SP005’; | Session 2 bắt đầu giao tác cập nhật giá sản phẩm của mã sản phẩm ‘SP004’. Nhưng khoá dòng ‘SP004’ đang do giao tác session 1 giữ. Session 2 xếp hàng đợi cho đến khi giao tác session 1 hoàn tất và nhả khoá để có thể cập nhật |
|  | ROLLBACK TO SAVEPOINT sp1; |  | Session 1 rollback việc cập nhật lương  Cho mã sản phẩm ‘SP005’. Session 2 vẫn chưa được thực hiện cập nhật vì giao tác của session 1 chưa hoàn tất. |
|  | Commit; |  | Session 1 kết thúc giao tác và session có thể vào cập nhật cho sản phẩm ‘SP005’; |
|  |  | Commit; | Session 2 kết thúc giao tác của mình sau khi cập nhật giá trị cho mã sản phẩm ‘SP005’ |

# Deadlock

## Định nghĩa

Deadlock là tình huống khi một hay nhiều người dùng đang chờ một đơn vị dữ liệu mà đang bị các người dùng khác, cũng đang chờ đơn vị dữ liệu khác nhả lock để thực hiện.

## Xử lý deadlock trong oracle

Hệ quản trị Oracle tự động phát hiện và xử lý deadlock bẳng cách rollback một transaction gây deadlock, giải phóng 1 dòng đang bị khóa. Cơ sở dữ liệu sẽ trả về thông báo cho transaction bị rollback.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Session 1** | **Session 2** | **Giải thích** |
|  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP004’; |  | Session 1 bắt đầu giao tác, cập nhật giá sản phẩm của mã sản phẩm ‘SP004’. |
|  |  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP005’; | Session 2 bắt đầu giao tác, cập nhật giá sản phẩm của mã sản phẩm ‘SP005’. |
|  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP005’; |  | Session 1 thực hiện cập nhật giá sản phẩm của mã sản phẩm ‘SP005’. Nhưng lúc này session 2 đang giữ khoá của mã sản phẩm ‘SP005’ nên session 1 phải đợi. |
|  |  | UPDATE SANPHAM SET GIASP = 15000 WHERE MASP = ‘SP004’; | Session 2 thực hiện cập nhật giá sản phẩm của mã sản phẩm ‘SP004’. Nhưng lúc này session 1 đang giữ khoá của mã sản phẩm ‘SP004’ nên session 2 phải đợi. |

Cả 2 session chờ đợi lẫn nhau, không seesion nào thực thi được nên xảy ra deadlock

# Các vấn đề xảy ra khi truy xuất dữ liệu đồng thời

## Mất dữ liệu cập nhật (Lost Update)

1. **Mô tả tình huống:**

Ngày 28/5/2023 khi lục soát kho thì hai nhân viên Lộc và Doan thấy còn một bị kẹo mentos. Hai nhân viên Lộc và Doan cùng chia nhau đếm số lượng cây kẹo mentos có trong bị kẹo. Nhân viên Lộc đếm được 25 cây kẹo và nhân viên Doan đếm được 35 cây kẹo, tổng số cây kẹo mentos đếm được là 60 cây. Số lượng cây kẹo mentos trong quan hệ SANPHAM hiện tại có 20 cây. Nhân viên Lộc và Doan bắt đầu cập nhật số cây kẹo của mình đã đếm được vào cột số lượng trong quan hệ SANPHAM.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Doan** | **Giải thích** |
|  | SELECT SOLUONG FROM SANPHAM WHERE TENSP = ‘KEO MENTOS’; |  | Nhân viên Lộc truy vấn số lượng cây kẹo mentos đang có trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Lộc thấy số lượng cây kẹo mentos có 20 cây. |
|  |  | SELECT SOLUONG FROM SANPHAM WHERE TENSP = ‘KEO MENTOS’; | Nhân viên Doan truy vấn số lượng cây kẹo mentos đang có trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Doan thấy số lượng cây kẹo mentos có 20 cây. |
|  | SL\_KEO\_CN := SOLUONG + 25; |  | Nhân viên Lộc gán giá trị cho biến SL\_KEO\_CN = số lượng kẹo đang có trong SANPHAM + số lượng kẹo vừa đếm được = 45 |
|  |  | SL\_KEO\_CN := SOLUONG + 35; | Nhân viên Doan gán giá trị cho biến SL\_KEO\_CN = số lượng kẹo đang có trong SANPHAM + số lượng kẹo vừa đếm được = 55 |
|  | UPDATE SANPHAM SET SOLUONG = SL\_KEO\_CN WHERE TENSP = ‘KEO MENTOS’; |  | Nhân viên Lộc cập nhật số lượng cây kẹo mentos. Sau khi cập nhật thì số lượng cây kẹo = 45. |
|  |  | UPDATE SANPHAM SET SOLUONG = SL\_KEO\_CN WHERE TENSP = ‘KEO MENTOS’; | Nhân viên Doan cập nhật số lượng cây kẹo mentos. Sau khi cập nhật thì số lượng cây kẹo = 55. |
|  | COMMIT; |  | Nhân viên Lộc commit giao tác thì câu lệnh update của Doan mới bắt đầu chạy và ảnh hưởng lên dữ liệu |
|  |  | COMMIT; | Nhân viên Doan commit, kết thúc giao tác cập nhật của mình. |

1. **Vấn đề, nguyên nhân và giải pháp:**

* **Vấn đề xảy ra**: khi cả hai nhân viên cùng gán giá trị cho biến SL\_KEO\_CN và cùng cập nhật đồng thời, số lượng kẹo của nhân viên Lộc vừa cập nhật thì đã bị số lượng kẹo của nhân viên Doan vừa cập nhật ghi đè lên. Sau đó nhân viên Lộc đọc lại dữ liệu số lượng kẹo thì thấy số lượng kẹo không nhất quán
* **Nguyên nhân**: cả hai nhân viên khi vào cập nhật dữ liệu đồng thời,

nhưng không có bất kỳ chế độ khoá nào được thiết lập để đảm bảo cho hai giao tác của hai nhân viên được thực hiện tuần tự, không để trường hợp mất nhất quán dữ liệu xảy ra.

* **Giải pháp**: Sử dụng mức cô lập SERIALIZABLE. Khi sử dụng mức cô lập SERIALIZABLE cho cả hai giao tác T1, T2 thì khi hai giao tác thực hiện đồng thời, giao tác T1 của nhân viên Lộc thực hiện xong trước và commit, cập nhật số lượng kẹo lên 45, giao tác T2 của nhân viên Doan cập nhật đồng thời sau giao tác T1 thì sẽ bị báo lỗi ( ORA-08177: can't serialize access for this transaction ), ngăn không cho vấn đề lost update xảy ra.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Doan** | **Giải thích** |
|  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  SELECT SOLUONG FROM SANPHAM WHERE TENSP = ‘KEO MENTOS’; |  | Set mức cô lập cho giao tác của nhân viên Lộc là SERIALIZABLE  Nhân viên Lộc truy vấn số lượng cây kẹo mentos đang có trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Lộc thấy số lượng cây kẹo mentos có 20 cây. |
|  |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  SELECT SOLUONG FROM SANPHAM WHERE SP = ‘KEO MENTOS’; | Set mức cô lập cho giao tác của nhân viên Doan là SERIALIZABLE  Nhân viên Doan truy vấn số lượng cây kẹo mentos đang có trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Doan thấy số lượng cây kẹo mentos có 20 cây. |
| **3.** | SL\_KEO\_CN := SOLUONG + 25; |  | Nhân viên Lộc gán giá trị cho biến SL\_KEO\_CN = số lượng kẹo đang có trong SANPHAM + số lượng kẹo vừa đếm được = 45 |
| **4.** |  | SL\_KEO\_CN := SOLUONG + 35; | Nhân viên Doan gán giá trị cho biến SL\_KEO\_CN = số lượng kẹo đang có trong SANPHAM + số lượng kẹo vừa đếm được = 55 |
| **5.** | UPDATE SANPHAM SET SOLUONG = SL\_KEO\_CN WHERE TENSP = ‘KEO MENTOS’; |  | Nhân viên Lộc cập nhật số lượng cây kẹo mentos. Sau khi cập nhật thì số lượng cây kẹo = 45. |
| **6.** |  | UPDATE SANPHAM SET SOLUONG = SL\_KEO\_CN WHERE TENSP = ‘KEO MENTOS’; | Nhân viên Doan cập nhật số lượng cây kẹo mentos. Sau khi cập nhật thì số lượng cây kẹo = 55. |
| **7.** | COMMIT; |  | Nhân viên Lộc commit giao tác. |
| **8.** |  | ORA-08177: can't serialize access for this transaction | Giao tác của nhân viên Doan sẽ tự động báo lỗi () và kết thúc giao tác |
| **9.** |  | Commit;  SELECT SOLUONG FROM SANPHAM WHERE TENSP = ‘KEO MENTOS’; | Nhân viên Doan commit và truy vấn lại dữ liệu thì thấy số lượng kẹo đã lên 45 |

## Đọc dữ liệu chưa commit(Dirty Read)

* 1. **Mô tả tình huống**

Ngày 3/6/2023 Nhân viên Lộc đang cập nhật giá cho sản phẩm ‘Khoai tây Ostar vị kim chi’, giá ban đầu là 14000, nhân viên Lộc cập nhật giá của sản phẩm này xuống 12000, đồng thời lúc đó nhân viên Doan vào đọc giá bán của sản phẩm này, thấy giá bán đã giảm xuống còn 12000 nên đã bán nó với giá 12000. Nhân viên Lộc mặc dù đã cập nhật xuống nhưng không commit mà rollback. Gây ra tình trạng không nhất quán dữ liệu. Có thể ảnh hưởng đến doanh thu cửa hàng.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Doan** | **Giải thích** |
|  | UPDATE SANPHAM SET GIASP = 12000 WHERE TENSP = ‘Khoai tây Ostar vị kim chi’ AND MASP = ‘SP001’; |  | Nhân viên Lộc cập nhật giá ‘ Khoai tây Ostar vị kim chi’ xuống 12000 từ SANPHAM. |
|  |  | SELECT GIASP FROM SANPHAM WHERE TENSP = ‘Khoai tây Ostar vị kim chi’ AND MASP = ‘SP001’; | Nhân viên Doan truy vấn giá K’hoai tây Ostar vị kim chi’ trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Doan thấy giá sản phẩm = 12000 |
|  | Rollback; |  | Nhân viên Lộc rollback kết quả của mình, trả giá của sản phẩm về giá ban đầu là 14000 |
|  |  | Commit; | Nhân viên Doan commit |

* 1. **Vấn đề, nguyên nhân, giải pháp**
     + **Vấn đề xảy ra:** khi cả hai nhân viên cùng thao tác trên một đơn vị cơ sở dữ liệu GIASP của SANPHAM, nhân viên Lộc đang vào cập nhật giá trị của GIASP thì nhân viên Doan cũng vào lấy thông tin của GIASP, nhưng sau đó nhân viên Lộc lại rollback giá trị đó, nên giá trị nhân viên Doan đã đọc là giá trị rác. Gây mất nhất quán dữ liệu
     + **Nguyên nhân:**  nhân viên Lộc và Doan cùng thao tác trên một đơn vị dữ liệu nhưng không có các chế độ khoá hợp lý để đảm bảo cho các giao tác được thực hiện tuần tự, không gây ra mất nhất quán dữ liệu.
     + **Giải pháp:** sử dụng mức cô lập SERIALIZABLE, để có một chế độ khoá phù hợp, vừa đảm bảo không xảy ra vấn đề đọc giữ liệu chưa commit và các vấn đề khác khi có nhiều giao tác xử lý đồng thời. Khi sử dụng mức cô lập SERIALIZABLE, khi nhân viên Lộc cập nhật và đã commit dữ liệu, nhưng nhân viên Doan chưa commit thì giá trị của GIASP cũng sẽ không thay đổi, khi nào nhân viên Doan commit và đọc lại giá trị thì nhân viên Doan mới thấy được sự thay đổi nếu như nhân viên Lộc commit dữ liệu, nếu rollback thì sẽ không có gì thay đổi

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Doan** | **Giải thích** |
|  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; |  | Set mức cô lập cho giao tác của nhân viên Lộc là SERIALIZABLE |
|  |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; | Set mức cô lập cho giao tác của nhân viên Doan là SERIALIZABLE |
| **3.** | UPDATE SANPHAM SET GIASP = 12000 WHERE TENSP = ‘Khoai tây Ostar vị kim chi’ AND MASP = ‘SP001’; |  | Nhân viên Lộc cập nhật giá ‘ Khoai tây Ostar vị kim chi’ xuống 12000 từ SANPHAM. |
| **4.** |  | SELECT GIASP FROM SANPHAM WHERE TENSP = ‘Khoai tây Ostar vị kim chi’ AND MASP = ‘SP001’; | Nhân viên Doan truy vấn giá K’hoai tây Ostar vị kim chi’ trong quan hệ SANPHAM. Sau khi truy vấn thì nhân viên Doan thấy giá sản phẩm = 14000 |
| **5.** | Rollback; |  | Nhân viên Lộc rollback kết quả của mình, trả giá của sản phẩm về giá ban đầu là 14000 |
| **6.** |  | Commit; | Nhân viên Doan commit. Không bị mất nhất quán dữ liệu hay bất cứ vấn đề nào của xử lý đồng thời gây ra |

## Đọc dữ liệu không lặp lại (Non – repeatable read)

* 1. **Mô tả tình huống**

Ngày 5/3 nhân viên Đạt vào đọc tên của mã sản phẩm ‘SP003’, sản phẩm này có tên là ‘Bánh otto nhân sô cô la’ để làm báo cáo thống kê, đồng thời, nhân viên Anh cũng vào cập nhật tên của mã sản phẩm ‘SP003’ thành ‘Bánh kinh đô nhân sô cô la’, nhân viên Đạt sau đó đọc lại thì thấy dữ liệu không giống như ban đầu mình đã đọc là ‘Bánh otto nhân sô cô la’ làm cho việc báo cáo thống kê không chính xác, gây phân vân cho nhân viên Đạt khi làm báo cáo.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Đạt** | **Transaction (T2)**  **Nhân Viên Anh** | **Giải thích** |
|  | SELECT TENSP FROM SANPHAM WHERE MASP = ‘SP003’; |  | Nhân viên Đạt đọc tên của mã sản phẩm ‘SP003’, và đọc được tên của sản phẩm là ‘Bánh otto nhân sô cô la’. |
|  |  | UPDATE SANPHAM SET TENSP = ‘Bánh kinh đô nhân sô cô la’ WHERE MASP = ‘SP003’; | Nhân viên Anh cập nhật tên của mã sản phẩm ‘SP003’ thành Bánh kinh đô nhân sô cô la. |
|  | SELECT TENSP FROM SANPHAM WHERE MASP = ‘SP003’; |  | Nhân viên Đạt đọc lại tên của mã sản phẩm ‘SP003’ thì lại thấy tên là ‘Bánh kinh đô nhân sô cô la’. Không nhất quán dữ liệu |
|  |  | Commit; | Nhân viên Anh commit thay đổi dữ liệu. |
|  | Commit; |  | Nhân viên Đạt commit. |

* 1. **Vấn đề xảy ra, nguyên nhân, giải pháp**
     + **Vấn đề xảy ra:** Hai nhân viên đồng thời thao tác trên một đơn vị dữ liệu, nhân viên Đạt đọc tên của một sản phẩm, đồng thời sau đó, nhân viên Anh cập nhật lại tên của sản phẩm đó. Nhân viên Đạt đọc lại lần thứ hai thì thấy tên sản phẩm đã bị thay đổi. Gây ra sai sót, không nhất quán dữ liệu
     + **Nguyên nhân:** hai giao tác thực hiện đồng thời trên một đơn vị dữ liệu nhưng không được sử dụng khoá hợp lý nên xảy ra vấn đề đọc không lặp lại
     + **Giải pháp:** Set mức độ cô lập thành SERIALIZABLE để không cho phép vấn đề đọc không lặp lại xảy ra. Khi set mức độ cô lập là SERIALIZABLE thì từ khi bắt đầu cho đến khi kết thúc việc đọc thì giao tác của nhân viên Đạt sẽ không bị ảnh hưởng bởi giao tác của nhân viên Anh nếu nhân viên Đạt chưa commit và ngược lại.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Đạt** | **Transaction (T2)**  **Nhân Viên Anh** | **Giải thích** |
|  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; |  | Set mức cô lập cho giao tác của nhân viên Đạt là SERIALIZABLE |
|  |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; | Set mức cô lập cho giao tác của nhân viên Anh là SERIALIZABLE |
| **3.** | SELECT TENSP FROM SANPHAM WHERE MASP = ‘SP003’; |  | Nhân viên Đạt đọc tên của mã sản phẩm ‘SP003’, và đọc được tên của sản phẩm là ‘Bánh otto nhân sô cô la’. |
| **4.** |  | UPDATE SANPHAM SET TENSP = ‘Bánh kinh đô nhân sô cô la’ WHERE MASP = ‘SP003’; | Nhân viên Anh cập nhật tên của mã sản phẩm ‘SP003’ thành Bánh kinh đô nhân sô cô la. |
| **5.** | SELECT TENSP FROM SANPHAM WHERE MASP = ‘SP003’; |  | Nhân viên Đạt đọc lại tên của mã sản phẩm ‘SP003’ thì lại thấy tên sản phẩm vẫn là ‘Bánh otto nhân sô cô la’. |
| **6.** |  | Commit; | Nhân viên Anh commit. Mặc dù nhân Anh commit nhưng dữ liệu thay đổi vẫn chưa ảnh hưởng đến nhân viên Đạt nếu nhân viên Đạt chưa commit |
| **7.** | Commit; |  | Nhân viên Đạt commit giao tác của mình. |

## Bóng ma(Phantom read)

* 1. **Mô tả tình huống:**

Ngày 9/6/2023 nhân viên Lộc đang làm báo cáo, kiểm kê với các hoá đơn của nhân viên Anh từ ngày 9/5/2023 đến ngày 9/6/2023. Ngày 9/6 Nhân viên Anh thêm một hoá đơn nên tập dữ liệu đó có thêm một hoá đơn được thêm vào. Và sau khi nhân viên Lộc đọc lại dữ liệu thì thấy có một dòng dữ liệu mới xuất hiện trong tập dữ liệu mà nhân viên Lộc đang thao tác.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Anh** | **Giải thích** |
| **1.** | SELECT \* FROM HOADON WHERE MANV = ‘NV005’ AND NGAYMUA > ‘9-MAY-2023’ AND NGAYMUA < ‘9-JUN-2023’; |  | Nhân viên lộc thống kê các hoá đơn của nhân viên Anh từ ngày 9/5/2023 tới ngày 9/6/2023 |
| **2.** |  | INSERT INTO HOADON(‘HD111’, ‘9-JUN-2023’, ‘NV005’, null, ‘Tien Mat’, 105000); | Nhân viên Anh thêm một hoá đơn mới vào quan hệ hoá đơn |
| **3.** | SELECT \* FROM HOADON WHERE MANV = ‘NV005’ AND NGAYMUA > ‘9-MAY-2023’ AND NGAYMUA < ‘9-JUN-2023’; |  | Nhân viên Lộc thống kê lại các hoá đơn của nhân viên anh từ ngày 9/5/2023 tới ngày 9/6/2023 thì thấy có thêm một bản ghi mới được thêm vào. |
| **4.** |  | Commit; | Nhân viên Anh commit. |
| **5.** | Commit; |  | Nhân viên Lộc commit. |

* 1. **Vấn đề xảy ra, nguyên nhân, kết quả**
* **Vấn đề xảy ra:** Nhân viên Lộc đang thống kê các hoá đơn của nhân viên Anh thì nhân viên Anh lại ghi thêm một hoá đơn mới vào tập các hoá đơn và nhân viên Lộc đang đọc. Sau đó, nhân viên Lộc đọc lại thì thấy có một bản ghi hoá đơn mới được nhân viên Anh thêm vào. Gây ra vấn đề không nhất quán dữ liệu trong quá trình nhân viên Lộc kiểm kê các hoá đơn của nhân viên Anh.
* **Nguyên nhân:** Hai nhân viên thực hiện đồng thời các giao tác trên một tập dữ liệu, nhưng chưa có các biện pháp ngăn ngừa vấn đề bóng ma khi xử lý đồng thời xảy ra nên đã xảy ra tình trạng bóng ma khi truy xuất dữ liệu đồng thời.
* **Giải pháp:** set mức cô lập lên mức SERIALIZABLE để đảm bảo vấn đề bóng ma không xảy ra khi truy xuất dữ liệu đồng thời, cũng như các vấn đề khác. Đảm bảo dữ liệu được nhất quán.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Transaction (T1)**  **Nhân Viên Lộc** | **Transaction (T2)**  **Nhân Viên Anh** | **Giải thích** |
| **1.** | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; |  | Set mức cô lập cho giao tác của nhân viên Lộc là SERIALIZABLE |
| **2.** |  | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE; | Set mức cô lập cho giao tác của nhân viên Anh là SERIALIZABLE |
| **3.** | SELECT \* FROM HOADON WHERE MANV = ‘NV005’ AND NGAYMUA > ‘9-MAY-2023’ AND NGAYMUA < ‘9-JUN-2023’; |  | Nhân viên Lộc thống kê các hoá đơn của nhân viên Anh từ ngày 9/5/2023 tới ngày 9/6/2023. |
| **4.** |  | INSERT INTO HOADON(‘HD111’, ‘9-JUN-2023’, ‘NV005’, null, ‘Tien Mat’, 105000); | Nhân viên Anh thêm một hoá đơn mới vào quan hệ hoá đơn |
| **5.** | SELECT \* FROM HOADON WHERE MANV = ‘NV005’ AND TRIGIA > 100.000 AND NGAYMUA > ‘9-MAY-2023’ AND NGAYMUA < ‘9-JUN-2023’; |  | Nhân viên Lộc thống kê lại các hoá đơn của nhân viên Anh từ ngày 9/5/2023 tới ngày 9/6/2023 thì vẫn là dữ liệu như trước khi nhân viên Anh chưa thêm bản ghi hoá đơn mới vào |
| **6.** |  | Commit; | Nhân viên Anh commit giao tác của mình. Mặc dù commit, nhưng dữ liệu chưa hiển thị ở giao tác của nhân viên Lộc, nên sẽ không thể xảy ra vấn đề bóng ma |
| **7.** | Commit; |  | Nhân viên Lộc commit giao tác của mình. |

# CHƯƠNG 4. THIẾT KẾ GIAO DIỆN

* 1. **Môi trường phát triển và triển khai** 
     1. **Môi trường phát triển**
* Hệ điều hành: Windows 11.
* Công cụ quản lý CSDL: SQL Developer.
* Công cụ vẽ sơ đồ phân tích thiết kế: draw.io.
* Công cụ xây dựng ứng dụng: Visual Studio Code,Apache Netbeans.
* Công cụ quản lý mã nguồn: Github.
  + 1. **Môi trường triển khai**
* Hệ điều hành: Windows 11.
* Cài đặt: SQL Developer.
  1. **Màn hình giao diện**
     1. **Màn hình đăng nhập**

**A screen shot of a tablet

Description automatically generated with low confidence**

Hình 1. Màn hình Đăng nhập

Tại đây, người dùng sẽ phải điền thông tin đăng nhập bao gồm tên đăng nhập và mật khẩu như đã cung cấp. Nếu điền sai hoặc thiếu sẽ có thông báo yêu cầu nhập lại cho đến khi nhập đúng.

Khi điền đúng thông tin đăng nhập, người dùng sẽ được chuyển đến giao diện màn hình Trang chủ. Trang Đăng nhập cũng sẽ hiển thị sau khi người dùng nhấn nút “ĐĂNG XUẤT”.

* + 1. **Màn hình trang chủ**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Hệ điều hành, phần mềm

Mô tả được tạo tự động**

Hình 2. Màn hình Trang chủ của quản lý

A picture containing text, screenshot, operating system, software

Description automatically generated

Màn hình Trang chủ của nhân viên

Màn hình TRANG CHỦ bao gồm các chức năng chính của phần mềm. Do có sự phân quyền cho người dùng nên giao diện hiển thị của quản lý và nhân viên sẽ có sự khác nhau.

* + 1. **Màn hình chức năng quản lý nhân viên**

**Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Biểu tượng máy tính

Mô tả được tạo tự động**

Màn hình quản lý nhân viên

Chức năng quản lý nhân viên: Khi nhấn vào nút “NHÂN VIÊN”, màn hình hiển thị các thông tin nhân viên đã có trong quan hệ nhân viên cùng với các nút chức năng “Thêm”, “Xóa”, Tra cứu”. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, thông tin nhân viên tương ứng sẽ hiển thị sau đó.

Ở nút “Xóa NV”, sau khi người dùng chọn thông tin nhân viên cần thực hiện chức năng và nhấn vào nút “Xóa NV”, sẽ có hộp thoại xác nhận hiện lên cho người dùng.

A screenshot of a computer

Description automatically generated

Màn hình xóa nhân viên

Tiếp đến, khi nhấn vào nút “Thêm NV”, người dùng sẽ được điều hướng đến màn hình tương ứng để có thể tiến hành thực hiện chức năng một cách cụ thể.

A screenshot of a computer

Description automatically generated

Màn hình thêm nhân viên

Nếu muốn sửa thông tin của nhân viên thì người dùng nhấn đúp chuột vào nhân viên cần sửa, màn hình sẽ hiển thị cho phép sửa thông tin nhân viên.

A screenshot of a computer

Description automatically generated

Màn hình sửa thông tin nhân viên

* + 1. **Màn hình chức năng quản lý sản phẩm**

**A screenshot of a computer

Description automatically generated**

Hình 3. Màn hình quản lý sản phẩm

Khi nhấn vào nút “SẢN PHẨM”, phần mềm sẽ hiển thị màn hình có các thông tin sản phẩm đã được thêm vào, cùng với các chức năng “Thêm”, “Xóa”, “Tra cứu”. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, sản phẩm cần tìm tương ứng sẽ hiển thị.

Ở nút “Xóa SP”, sau khi người dùng chọn thông tin sản phẩm cần thực hiện chức năng và nhấn vào nút “Xóa SP”, sẽ có hộp thoại xác nhận hiện lên cho người dùng.

A screenshot of a computer

Description automatically generated

Hình 4. Màn hình xóa sản phẩm

Tiếp đến, khi nhấn vào nút “Thêm SP”, người dùng sẽ được điều hướng đến màn hình tương ứng để có thể tiến hành thực hiện chức năng một cách cụ thể.

A screenshot of a computer

Description automatically generated

Hình 5. Màn hình thêm sản phẩm

Nếu muốn sửa thông tin của sản phẩm thì người dùng nhấn đúp chuột vào sản phẩm cần sửa, màn hình sẽ hiển thị hộp thoại để sửa thông tin sản phẩm.

A screenshot of a computer

Description automatically generated

Hình 6. Màn hình sửa thông tin sản phẩm

* + 1. **Màn hình chức năng quản lý khách hàng**

A screenshot of a computer

Description automatically generated

Hình 7. Màn hình quản lý khách hàng

Khi nhấn vào nút “KHÁCH HÀNG”, phần mềm sẽ hiển thị màn hình có các thông tin khách hàng đã được thêm vào, cùng với các chức năng “Thêm”, “Xóa”, “Tra cứu”. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, thông tin khách hàng cần tìm tương ứng sẽ hiển thị.

Ở nút “Xóa KH”, sau khi người dùng chọn thông tin khách hàng cần thực hiện chức năng và nhấn vào nút “Xóa KH”, sẽ có hộp thoại xác nhận hiện lên cho người dùng.

A screenshot of a computer

Description automatically generated

Hình 8. Màn hình xóa khách hàng

Tiếp đến, khi nhấn vào nút “Thêm KH”, người dùng sẽ được điều hướng đến màn hình tương ứng để có thể tiến hành thực hiện chức năng một cách cụ thể.

A screenshot of a computer

Description automatically generated

Hình 9. Màn hình thêm khách hàng

Nếu muốn sửa thông tin khách hàng thì người dùng nhấn đúp chuột vào khách hàng cần sửa, màn hình sẽ hiển thị hộp thoại để sửa thông tin khách hàng.

A screenshot of a computer

Description automatically generated

Hình 10. Màn hình sửa thông tin khách hàng

* + 1. **Màn hình chức năng quản lý hóa đơn**

A screenshot of a computer

Description automatically generated

*Hình 7.4.6.1 Màn hình quản lý hóa đơn*

Khi nhấn vào nút “HÓA ĐƠN”, phần mềm sẽ hiển thị các thông tin hóa đơn đã được thêm vào, cùng với chức năng “Tra cứu”. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, thông tin hóa đơn cần tìm tương ứng sẽ hiển thị.

Nếu muốn xem thông tin chi tiết của hóa đơn thì người dùng nhấn đúp vào hóa đơn cần xem thông tin chi tiết, màn hình hiển thị thông tin chi tiết của hóa đơn.

A screenshot of a computer

Description automatically generated with medium confidence

Hình 11. Màn hình chi tiết hóa đơn

* + 1. **Màn hình chức năng quản lý kho**

A screenshot of a computer

Description automatically generated with medium confidence

Hình 12. Màn hình quản lý kho

Khi nhấn vào nút “KHO”, phần mềm sẽ hiển thị màn hình có các thông tin về lô hàng đã được thêm vào, cùng với các chức năng “Thêm”, “Xóa”, “Tra cứu”. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, thông tin khách hàng cần tìm tương ứng sẽ hiển thị.

Ở nút “Xóa lô hàng”, sau khi người dùng chọn thông tin lô hàng cần thực hiện chức năng và nhấn vào nút “Xóa lô hàng”, sẽ có hộp thoại xác nhận hiện lên cho người dùng.

A screenshot of a computer

Description automatically generated

Hình 13. Màn hình xóa lô hàng

Tiếp đến, khi nhấn vào nút “Thêm lô hàng”, người dùng sẽ được điều hướng đến màn hình tương ứng để có thể tiến hành thực hiện chức năng một cách cụ thể.

A screenshot of a computer

Description automatically generated

Hình 14. Màn hình thêm lô hàng

Sau khi thêm mới một lô hàng, người dùng có thể thêm thông tin chi tiết của lô hàng đó bằng cách nhấn đúp vào lô hàng cần thêm thông tin chi tiết, người dùng sẽ được điều hướng đến màn hình tương ứng để có thể tiến hành thực hiện chức năng một cách cụ thể.

****

* + 1. **Màn hình chức năng quản lý bán hàng**

Khi nhấn vào nút “BÁN HÀNG”, phần mềm sẽ hiển thị màn hình có các chức năng để thanh toán cho khách hàng. Tại đây, người dùng có thể nhập thông tin tìm kiếm rồi nhấn Enter, thông tin sản phẩm mà khách hàng mua sẽ hiển thị sau đây. Nhân viên nhấn vào sản phẩm, có thể thay đổi số lượng tương ứng, giao diện sẽ hiển thị tổng tiền, (nhân viên có thể tùy chọn nhập số điện thoại của khách hàng thành viên). Sau đó nhấn vào nút “Thanh Toán”, giao diện sẽ xuất hiện hộp thoại “Thanh toán thành công” và tự động thêm thông tin hóa đơn vào quan hệ hoá đơn.

A screenshot of a computer

Description automatically generated

Hình 15. Màn hình quản lý bán hàng

# CHƯƠNG 5. KẾT LUẬN

* 1. **Bảng phân công công việc**

|  |  |  |
| --- | --- | --- |
| Họ và tên | MSSV | Phân công |
| Vũ Thanh Doan | 21520191 | - Làm bảng mô tả chi tiết các quan hệ. Thiết kế giao diện, cùng hoàn thành bài báo cáo |
| Võ Hồng Kim Anh | 21520597 | * Làm quy định hệ thống, ràng buộc toàn vẹn, tóm tắt đề tài. Thiết kế giao diện, cùng hoàn thành bài báo cáo |
| Hoàng Gia Lộc | 21521086 | - Code phần mềm, Xây dựng các Trigger, Stored Procedure, cùng hoàn thành bài báo cáo |
| Đỗ Sĩ Đạt | 21521932 | - Xây dựng các Trigger, Stored Procedure.  - Xử lý giao tác đồng thời.  - Làm chương 2 và 3 của báo cáo. |

* 1. **Kết quả đạt được**
     1. **Lý thuyết**

Thứ nhất, sau khi học môn hệ quản trị cơ sở dữ liệu, nhóm em đã hiểu và áp dụng được những kiến thức trong môn hệ quản trị để thực hiện một phần mềm quản lý cơ bản. Biết cách xây dựng các thao tác cho trigger vào stored procedure, xử lý các giao tác đồng thời, thiết kế một mô hình dữ liệu. Thứ hai là kết nối hệ quản trị dữ liệu oracle với java, để thực hiện các giao tác stored procedure và trigger.

Cuối cùng, đồ án đã mang lại cho nhóm một cơ hội để làm việc nhóm cùng nhau, giúp phát triển các kỹ năng làm việc nhóm.

* + 1. **Công nghệ**
* Biết cách sử dụng phần mềm Oracle SQL Developer để thực hiện xây dựng, điều khiển truy xuất, lưu trữ và bảo trì dữ liệu.
* Biết cách sử dụng figma, netbeans hỗ trợ thiết kế giao diện.
* Biết cách sử dụng hibernate, jdbc để kết nối giữa java và oracle.
  1. **Hướng phát triển**
     1. **Về hệ thống**

Nhóm dự định sẽ phát triển phần mềm thêm các tính năng như:

* Tính năng đọc mã sản phẩm bằng đầu đọc mã vạch chuyên dụng
* Tự thêm nhiều phân loại khác nhau trong sản phẩm thay vì những phân loại có sẵn.
* Trang hiển thị thông tin cá nhân của nhân viên.
* Tính năng xuất, in hóa đơn cho khách hàng.
* Tính năng báo cáo, thống kê xuất ra file excel hoặc pdf trên máy.
* Khả năng tự tương thích kích cỡ giao diện tùy vào loại máy mà người dùng sử dụng.
  + 1. **Về con người**

Cùng nhau hợp tác và phát triển thêm phần mềm trong tương lai nếu còn cùng định hướng.

# DANH MỤC TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
|  | https://viblo.asia/p/su-dung-trigger-trong-sql-qua-vi-du-co-ban-aWj538APK6m |
|  | https://www.oracletutorial.com/plsql-tutorial/oracle-trigger  https://freetuts.net/tim-hieu-procedure-trong-oracle-1432.html  https://viblo.asia/p/tim-hieu-ve-transactions-maGK73xeKj2 |