TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

ĐẠI HỌC QUỐC GIA TP.HCM



**BÁO CÁO ĐỒ ÁN**

Đề tài: **THIẾT KẾ HỆ THỐNG**

**QUẢN LÝ VÉ MÁY BAY**

Giảng viên hướng dẫn : **Cô Đỗ Thị Minh Phụng**

Sinh viên thực hiện : **Nhóm 7**

Lê Văn Pa – 15520596

Lã Ngô Mỹ Linh – 17520683

Đặng Văn Công Hoàng – 17520503

Mai Trung Tín – 15520894

***TP. Hồ Chí Minh, ngày 28 tháng 6 năm 2019***

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

ĐẠI HỌC QUỐC GIA TP.HCM



**BÁO CÁO ĐỒ ÁN**

Đề tài: **THIẾT KẾ HỆ THỐNG**

**QUẢN LÝ VÉ MÁY BAY**

Giảng viên hướng dẫn : **Cô Đỗ Thị Minh Phụng**

Sinh viên thực hiện : **Nhóm 7**

Lê Văn Pa – 15520596

Lã Ngô Mỹ Linh – 17520683

Đặng Văn Công Hoàng – 17520503

Mai Trung Tín – 15520894

***TP. Hồ Chí Minh, ngày 28 tháng 6 năm 2019***

MỞ ĐẦU

Trên thế giới hiện nay, tin học là một ngành phát triển không ngừng. Thời kỳ công nghiệp đòi hỏi đòi hỏi thông tin nhanh chóng chính xác. Có thể nói tin học đã chiếm một vị trí quan trọng trong cuộc sống hàng ngày, trong các ngành khoa học kỹ thuật. Trong những năm gần đây cùng với sự phát triển chung của ngành tin học thế giới và khu vực, ngành tin học nước ta đó cú bước phát triển nhất định.

Việc đưa tin học vào công việc quản lý, nhằm giảm bớt sức lao động của con người, tiết kiệm thời gian, độ chính xác cao và tiện lợi rất nhiều so với việc làm quản lý giất tờ bằng thủ công như trước đây. Tin học hóa nhằm thu hẹp không gian lưu trữ, tránh được thất lạc dữ liệu, tự động hóa và cụ thể hóa các thông tin theo nhu cầu của con người. Còn ở Việt Nam ngành tin học tuy còn non trẻ nhưng cũng đang từng bước bứt phá được những thành công trong việc áp dụng tin học vào việc phat triển kinh tế – xã hội, góp phần vào việc phát triển xã hội và khẳng định mình ở thị trường quốc tế.

Chính vì vậy khi em được nhận đề tài xây dựng chương trình quản lý vé máy bay dùng hệ quản trị CSDL Oracle, với những kiến thức đã được học và quá trình tìm hiểu, đặc biệt là sự hướng dẫn rất đỗi nhiệt tình của cô giáo, Ths. Đỗ Thị Minh Phụng, chúng em đã hoàn thành được đề tài của mình với một bài toán quản lý. Trong quá trình làm bài chắc chắn em sẽ mắc phải những thiếu sót nên rất mong nhận được sự đóng góp ý kiến của cô để em có cơ hội sửa sai, phát triển trong thực tế và từng bước hoàn thiện mình.

LỜI CÁM ƠN

Lời đầu tiên, nhóm xin đặc biệt gửi lời cảm ơn đến Cô Đỗ Thị Minh Phụng (Giáo viên môn Hệ quản trị cơ sở dữ liệu). Cô đã cung cấp kiến thức, chỉ bảo và đóng góp những ý kiến quý báu giúp nhóm hoàn thành được đồ án môn học của mình.

Xuất phát từ mục đích học tập, tìm hiểu sâu hơn các kiến thức về các hệ quản trị cơ sở dữ liệu, cũng như tìm hiểu thêm về quy trình nghiệp vụ của các rạp chiếu phim, nhóm chúng em đã thực hiện đồ án “Thiết kế hệ thống quản lý vé máy bay”. Trong quá trình thực hiện đồ án, dựa trên những kiến thức được Cô cung cấp trên trường kết hợp với việc tự tìm hiểu những công cụ và kiến thức mới, nhóm đã cố gắng thực hiện đồ án một cách tốt nhất. Tuy nhiên, đồ án chưa hoàn thiện và còn nhiều sai sót nhưng nó là kết quả của sự nỗ lực của các thành viên trong nhóm, sự giúp đỡ của tất cả bạn bè và Cô.

Nhóm rất mong nhận sự góp ý từ phía Cô nhằm rút ra những kinh nghiệm quý báu và hoàn thiện vốn kiến thức để nhóm có thể tiếp tục hoàn thành những đồ án khác trong tương lai. Xin chân thành cảm ơn quý Cô!

Nhóm sinh viên thực hiện

TÓM TẮT BÁO CÁO

Nội dung bài báo cáo này là kết quả quá trình thực hiện đồ án của các thành viên trong nhóm trong vòng 4 tháng. Thời gian từ ngày 28 tháng 02 năm 2019 đến ngày 28 tháng 06 năm 2019

Bài báo cáo bao gồm các chương sau:

* Chương 1: Tổng quan đề tài
* Chương 2: Phân tích, thiết kế chức năng
* Chương 3: Thiết kế CSDL
* Chương 4: Xây dựng và quản lý giao tác
* Chương 5: Xử lý truy xuất đồng thời
* Chương 6: Thiết kế giao diện
* Chương 7: Tống kết

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Giảng viên hướng dẫn

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH 1](#_Toc12042045)

[DANH MỤC BẢNG BIỂU VÀ TỪ VIẾT TẮT 2](#_Toc12042046)

[Chương 1. TỔNG QUAN ĐỀ TÀI 3](#_Toc12042047)

[1.1. Đặt vấn đề 3](#_Toc12042048)

[1.2. Giải pháp 3](#_Toc12042049)

[1.3. Công nghệ sử dụng 3](#_Toc12042050)

[Chương 2. PHÂN TÍCH, THIẾT KẾ CHỨC NĂNG 4](#_Toc12042051)

[2.1. Phân tích yêu cầu 4](#_Toc12042052)

[2.1.1. Yêu cầu chức năng 4](#_Toc12042053)

[2.1.2. Yêu cầu phi chức năng 4](#_Toc12042054)

[2.2. Thiết kế chức năng 5](#_Toc12042055)

[Chương 3. THIẾT KẾ CƠ SỞ DỮ LIỆU 6](#_Toc12042056)

[3.1. Mô hình thực thể - mối kết hợp (ERD) 6](#_Toc12042057)

[3.2. Mô hình quan hệ 6](#_Toc12042058)

[3.3. Mô tả các bảng 7](#_Toc12042059)

[3.4. Ràng buộc toàn vẹn 9](#_Toc12042060)

[Chương 4. XÂY DỰNG VÀ QUẢN LÝ GIAO TÁC 10](#_Toc12042061)

[4.1. Triggers 10](#_Toc12042062)

[4.1.1. Cơ sở lý thuyết 10](#_Toc12042063)

[4.1.2. Danh sách các trigger 10](#_Toc12042064)

[4.1.3. Mô tả một số trigger 11](#_Toc12042065)

[4.2. Stored procedures 11](#_Toc12042066)

[4.2.1. Cơ sở lý thuyết 11](#_Toc12042067)

[4.2.2. Danh sách các stored procedure 11](#_Toc12042068)

[4.2.3. Mô tả một số stored procedure 13](#_Toc12042069)

[4.3. Functions 13](#_Toc12042070)

[4.3.1. Cơ sở lý thuyết 13](#_Toc12042071)

[4.3.2. Danh sách các function 14](#_Toc12042072)

[4.3.3. Mô tả một số function 14](#_Toc12042073)

[Chương 5. XỬ LÝ TRUY XUẤT ĐỒNG THỜI 15](#_Toc12042074)

[5.1. Các mức cô lập 15](#_Toc12042075)

[5.1.1. Read Uncommitted 15](#_Toc12042076)

[5.1.2. Read Committed 15](#_Toc12042077)

[5.1.3. Repeatable Read 16](#_Toc12042078)

[5.1.4. Serializable 17](#_Toc12042079)

[5.2. Lost update 17](#_Toc12042080)

[5.3. Dirty read 18](#_Toc12042081)

[5.4. Unrepeatable read 18](#_Toc12042082)

[5.5. Phantom 20](#_Toc12042083)

[5.6. Deadlock 21](#_Toc12042084)

[Chương 6. THIẾT KẾ GIAO DIỆN 24](#_Toc12042085)

[6.1. Danh sách các màn hình 24](#_Toc12042086)

[6.2. Sơ đồ liên kết màn hình 25](#_Toc12042087)

[6.3. Mô tả chức năng từng màn hình 25](#_Toc12042088)

[Chương 7. TỔNG KẾT 32](#_Toc12042089)

[7.1. Kết quả đạt được 32](#_Toc12042090)

[7.2. Hạn chế 32](#_Toc12042091)

[7.3. Hướng phát triển 33](#_Toc12042092)

[BẢNG PHÂN CÔNG CÔNG VIỆC 34](#_Toc12042093)

[TÀI LIỆU THAM KHẢO 35](#_Toc12042094)

DANH MỤC HÌNH ẢNH

[Hình 1: Mô hình ERD 6](#_Toc12042095)

[Hình 2: Cấu trúc của trigger trong Oracle 10](#_Toc12042096)

[Hình 3: Cấu trúc của stored procedure trong Oracle 11](#_Toc12042097)

[Hình 4: Cấu trúc của function trong Oracle 14](#_Toc12042098)

[Hình 5: Nguyên nhân gây deadlock 22](file:///C:\Users\KaiPa\Desktop\baocao.docx#_Toc12042099)

[Hình 6: Sơ đồ liên kết màn hình 25](#_Toc12042100)

DANH MỤC BẢNG BIỂU VÀ TỪ VIẾT TẮT

Danh mục từ viết tắt:

CSDL – Cơ sở dữ liệu

HQT CSDL – Hệ quản trị cơ sở dữ liệu

RBTV – Ràng buộc toàn vẹn

**Danh mục bảng biểu:**

[Bảng 1: Yêu cầu CSDL 4](#_Toc12042101)

[Bảng 2: Thiết kế chức năng 5](#_Toc12042102)

[Bảng 3: mô tả bảng DANGNHAP 7](#_Toc12042103)

[Bảng 4: Mô tả bảng HANGMAYBAY 7](#_Toc12042104)

[Bảng 5: Mô tả bảng SANBAY 7](#_Toc12042105)

[Bảng 6: Mô tả bảng CHUYENBAY 8](#_Toc12042106)

[Bảng 7: Mô tả bảng KHACHHANG 8](#_Toc12042107)

[Bảng 8: Mô tả bảng VE 9](#_Toc12042108)

[Bảng 9: Mô tả bảng CHITIETVE 9](#_Toc12042109)

[Bảng 10: Mô tả bảng KHUYENMAI 9](#_Toc12042110)

[Bảng 11: Danh sách trigger 11](#_Toc12042111)

[Bảng 12: Danh sách stored procedure 13](#_Toc12042112)

[Bảng 13: Danh sách function 14](#_Toc12042113)

[Bảng 14: Danh sách các màn hình 24](#_Toc12042114)

# TỔNG QUAN ĐỀ TÀI

## Đặt vấn đề

Việc tin học hóa quy trình, công cụ quản lý vé máy bay cũng như các chuyến bay, khách hàng ngày càng trở thành mối quan tâm của nhiều doanh nghiệp, cơ quan, tổ chức. Lợi ích của các hệ thống quản lý này đem lại là vô cùng to lớn. Nó khắc phục được những yếu điểm của phương pháp quản lý truyền thống bằng thủ công và mở ra một môi trường làm việc hiện đại, tự động cho doanh nghiệp, tổ chức. Từ đó, phương pháp này đem lại nhiều hiệu quả về mặt thời gian, chi phí và nhân lực cho doanh nghiệp.

Ngày nay nhu cầu du lịch của mọi người ngày càng cao, nhu cầu sử dụng dịch vụ hàng không theo đó cũng tăng vọt. Điều đó dẫn tới quy mô CSDL cần lưu trữ của hãng hàng không ngày càng lớn làm cho việc quản lý CSDL truyền thống không còn hiệu quả, gấy tiêu tốn thời gian, tiền bạc, công sức.

## Giải pháp

Để giải quyết vẫn đề trên, nhóm chúng em quyết định thiết kế phần mềm Quản lý vé máy bay, sử dụng HQT CSDL Oracle và các công nghệ của nó. Kèm theo việc áp dụng các kỹ thuật xử lý các giao tác xảy ra đồng thời và giải quyết các vấn đề khác của HQT CSDL, chúng em mong muốn phần mềm này sẽ được đưa vào sử dụng trong thực tiễn.

## Công nghệ sử dụng

* HQT CSDL: Oracle SQL Developer
* Ngôn ngữ xây dựng phần mềm: Java
* IDE xây dựng phần mềm: Netbean
* Công cụ quản lý mã nguồn: Github
* Công cụ thiết kế ERD:

# PHÂN TÍCH, THIẾT KẾ CHỨC NĂNG

## Phân tích yêu cầu

### Yêu cầu chức năng

* Yêu cầu CSDL phải quản lý được các đối tượng sau:

|  |  |
| --- | --- |
| Đối tượng | Mô tả |
| Khách hàng | Lưu các thông tin cơ bản của khách hàng, để phân biệt từng khách hàng với nhau |
| Nhân viên hỗ trợ | Mỗi khi khách hàng cần tra vé, tra chuyến bay hay đặt vé chuyến bay thì có nhân viên hỗ trợ cho khách hàng. Phần mềm cho phép nhiều nhân viên đăng nhập vào cùng một lúc |
| Chuyến bay | Lưu thông tin chuyến bay |
| Vé máy bay | Thông tin vé có chưa thông tin chuyến bay và thông tin khách hàng cũng như nhân viên hỗ trợ |
| Khuyến mãi | Lưu thông tin chương trình khuyến mãi hoặc mã khuyến mãi (coupon, voucher,…) |
| Sân bay | Lưu thông tin sân bay |

Bảng 1: Yêu cầu CSDL

* Yêu cầu phần mềm cho phép người dùng tạo chuyến bay, tạo khách hàng, và các đối tượng khác theo như yêu cầu.
* Có tính năng tìm kiếm chuyến bay, đặt vé / hủy vé máy bay.
* Cho phép truy xuất lại thông tin chi tiết vé đã đặt, xem lịch sử chuyến bay và lịch sử vé.
* Tạo thống kê, báo cáo theo năm, tháng, quý.
* Có thể sao lưu và phục hồi dữ liệu.

### Yêu cầu phi chức năng

* Yêu cầu giao diện: tiện dụng, thân thiện, sinh động và tương tác cao.
* Yêu cầu hiệu suất:
* Thời gian cho phép để hệ thống phản hồi lại thông tin đã tiếp nhận yêu cầu xử lý từ phía người sử dụng là 5 giây.
* Thời gian cho phép để trả kết quả tìm kiếm thông tin là dưới 7 giây.
* Phần mềm sử dụng hợp lý tài nguyên hệ thống để tránh treo máy, tràn bộ nhớ ảnh hưởng các chương trình khác.
* Cho phép nhiều người sử dụng cùng lúc mà không bị xung đột dữ liệu.
* Yêu cầu an toàn:
* Hệ thống có thể sao lưu, phục hồi lại cơ sở dữ liệu khi có sự cố.
* Phần mềm chạy trên hệ thống không xung đột với các phần mềm khác.

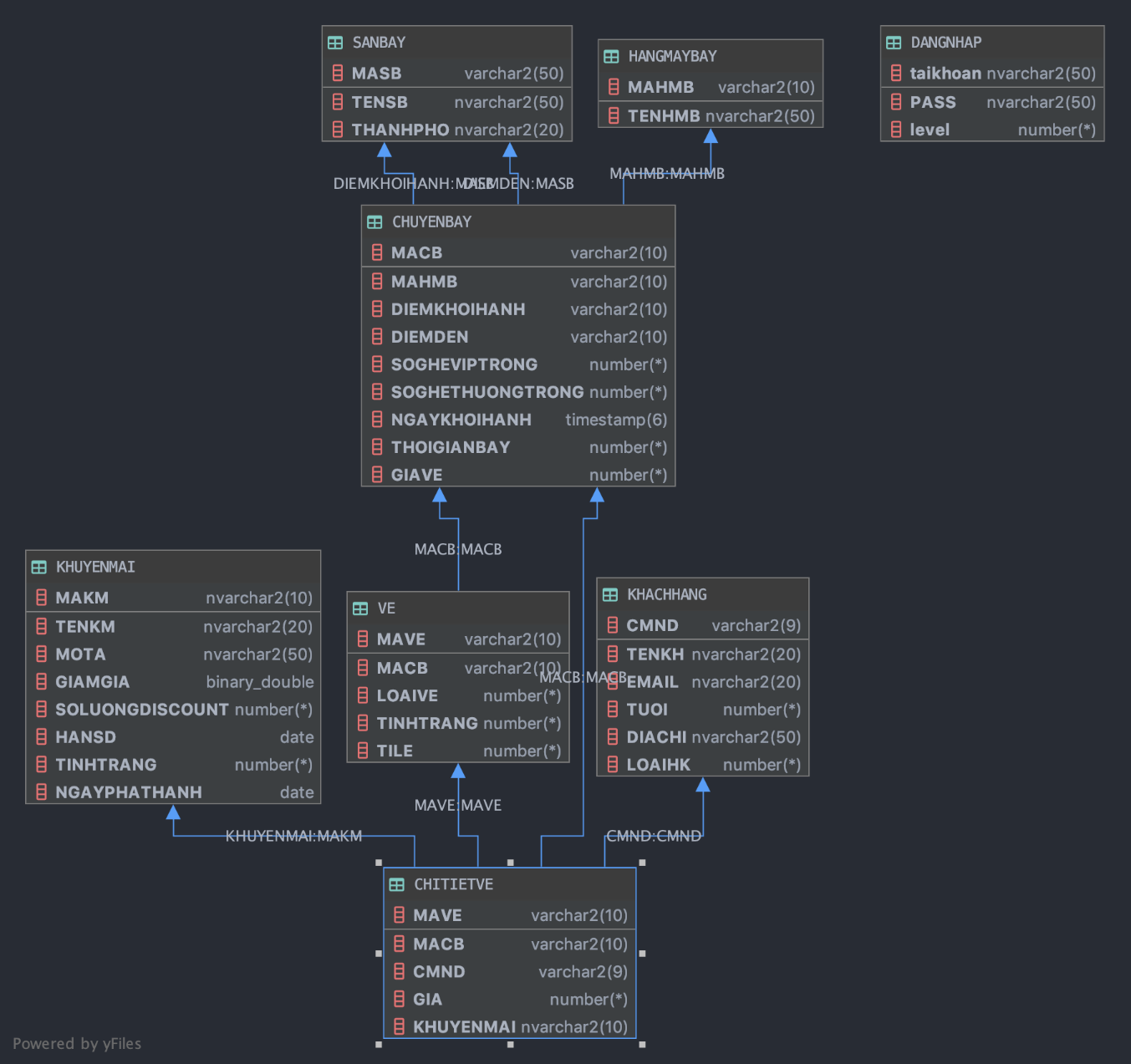
## Thiết kế chức năng

|  |  |
| --- | --- |
| Tên chức năng | Mô tả |
| Tạo chuyến bay | Tạo chuyến bay mới với thời gian khởi hành, địa điểm đi, địa điểm đến, thời gian bay, số ghế và một số thông tin khác |
| Đặt vé | Đặt vé khi khách hàng yêu cầu, thông tin vé cần có thông tin chuyến bay, loại vé và giá trị để tính giá vé |
| Tra cứu vé | Sau khi đặt vé, nếu khách hàng có nhu cầu kiểm tra lại thì dùng chức năng này để truy xuất thông tin vé. Trong thông tin vé có đầy đủ thông tin khách hàng, thông tin chuyến bay và thông tin vé |
| Thống kê | Thống kê số lượng khách/chuyến bay trong tuần/tháng/năm hoặc trong một khoảng thời gian nhất định. |
| Khuyến mãi | Áp dụng khuyến mãi khi mua vé, có 2 loại là khuyến mãi theo khoảng thời gian và giảm giá theo voucher. |

Bảng 2: Thiết kế chức năng

# THIẾT KẾ CƠ SỞ DỮ LIỆU

## Mô hình thực thể - mối kết hợp (ERD)



Hình 1: Mô hình ERD

## Mô hình quan hệ

* **DANGNHAP**: TAIKHOAN, PASS, LEVEL
* **HANGMAYBAY**: MAHMB, TENHMB
* **SANBAY**: MASB, TENSB, THANHPHO
* **CHUYENBAY**: MACB, MAHMB, DIEMKHOIHANH, DIEMDEN, SOGHEVIPTRONG, SOGHETHUONGTRONG, NGAYKHOIHANH,THOIGIANBAY, GIAVE
* **KHACHHANG**: CMND, TENKH, EMAIL, TUOI, DIACHI, LOAIKH
* **VE**: MAVE, MACB, LOAIVE, TINHTRANG, TILE
* **CHITIETVE**: MAVE, MACB, CMND, GIA, KHUYENMAI
* **KHUYENMAI**: MAKM, TENKM, MOTA, GIAMGIA, SOLUONGDISCOUNT, HANSD, TINHTRANG, NGAYPHATHANH

## Mô tả các bảng

* Bảng DANGNHAP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | EMAIL | NVARCHAR(50) | x | Email nhân viên, dùng như là username |
| 2 | PASS | NVARCHAR(50) |  | Mật khẩu truy cập |
| 3 | LEVEL | INT |  | Cấp độ nhân viên |

Bảng 3: mô tả bảng DANGNHAP

* Bảng HANGMAYBAY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MAHMB | NVARCHAR(10) | x | Mã hãng máy bay |
| 2 | TENHMB | NVARCHAR(20) |  | Tên hãng máy bay |

Bảng 4: Mô tả bảng HANGMAYBAY

* Bảng SANBAY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MASB | NVARCHAR(10) | x | Mã sân bay |
| 2 | TENSB | NVARCHAR(20) |  | Tên sân bay |

Bảng 5: Mô tả bảng SANBAY

* Bảng CHUYENBAY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MACB | NVARCHAR(10) | x | Mã chuyến bay |
| 2 | MAHMB | NVARCHAR(10) | x | Mã hãng máy bay |
| 3 | DIEMKHOIHANH | NVARCHAR(50) |  | Điểm khởi hành / điểm đi |
| 4 | DIEMDEN | NVARCHAR(50) |  | Điểm đến / điểm dừng cuối cùng |
| 5 | SANBAYDI | NVARCHAR(10) |  | Mã sân bay đi |
| 6 | SANBAYDEN | NVARCHAR(10) |  | Mã sân bay đến |
| 7 | SOGHEVIPTRONG | INT |  | Số ghế vip còn trống trên chuyến bay |
| 8 | SOGHETHUONGTRONG | INT |  | Số ghế thường còn trống trên chuyến bay |
| 9 | NGAYKHOIHANH | DATE |  | Ngày khởi hành |
| 10 | THOIGIANBAY | INT |  | Thời gian bay |
| 11 | GIAVE | MONEY |  | Giá vé gốc loại thường |

Bảng 6: Mô tả bảng CHUYENBAY

* Bảng KHACHHANG

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | CMND | VARCHAR2(9) | x | CMND |
| 2 | TENKH | NVARCHAR(20) | x | Tên khách hàng |
| 3 | EMAIL | NVARCHAR(20) |  | Email khách hàng |
| 4 | TUOI | INT |  | Tuổi khách hàng |
| 5 | DIACHI | NVARCHAR(50) |  | Địa chỉ |
| 6 | LOAIHK | INT |  | Loại khách hàng (trẻ em, người lớn) |

Bảng 7: Mô tả bảng KHACHHANG

* Bảng VE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MAVE | NVARCHAR(10) | x | Mã vé |
| 2 | MACB | NVARCHAR(10) | x | Mã chuyến bay |
| 3 | LOAIVE | bit |  | Loại vé (thường, vip) |
| 4 | TINHTRANG | bit |  | Tình trạng (còn trống hoặc không) |
| 5 | TILE | INT |  | Tỉ lệ giá vé |

Bảng 8: Mô tả bảng VE

* Bảng CHITIETVE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MAVE | NVARCHAR(10) | x | Mã vé |
| 2 | MACB | NVARCHAR(10) | x | Mã chuyến bay |
| 3 | CMND | NVARCHAR(9) | x | CMND khách hàng |
| 4 | GIA | INT |  | Giá vé cuối cùng |
| 5 | KHUYENMAI | NVARCHAR(10) | x | Mã khuyến mãi |

Bảng 9: Mô tả bảng CHITIETVE

* Bảng KHUYENMAI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Kiểu dữ liệu | null | Mô tả |
| 1 | MAKM | NVARCHAR(10) | x | Mã khuyến mãi |
| 2 | TENKM | NVARCHAR(20) |  | Tên khuyến mãi |
| 3 | MOTA | NVARCHAR(50) |  | Mô tả |
| 4 | GIAMGIA | BINARY\_DOUBLE |  | Số tiền giảm |
| 5 | SOLUONGDISCOUNT | INT |  | Số lần giảm giá |
| 6 | HANSD | DATE |  | Hạn sử dụng |
| 7 | TINHTRANG | INT |  | Tình trạng (hợp lệ hay không) |
| 8 | NGAYPHATHANH | DATE |  | Ngày phát hành khuyến mãi |

Bảng 10: Mô tả bảng KHUYENMAI

## Ràng buộc toàn vẹn

|  |  |
| --- | --- |
| STT | Nội dung RBTV |
| 2 | Khách hàng đặt vé phải trên 18 tuổi |
| 3 | Thời gian mỗi chuyến bay kéo dài ít nhất là 30 phút |
| 4 | Khách hàng loại “trẻ em” phải đảm bảo dưới 18 tuổi và nếu dưới 14 tuổi thì không có CMND |
| 5 | Một chuyến bay có điểm đến khác với điểm đi, ngày về phải lớn hơn ngày khởi hành |

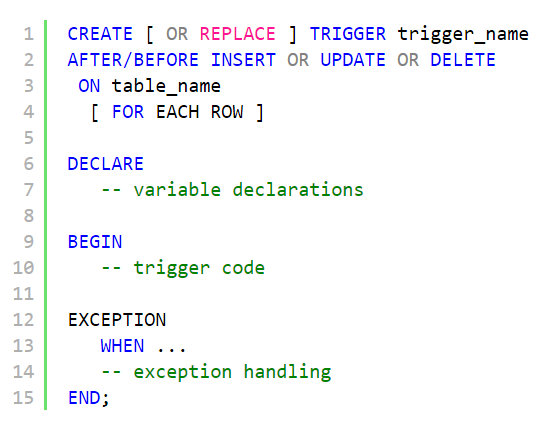
Bảng 11: Các ràng buộc toàn vẹn

# XÂY DỰNG VÀ QUẢN LÝ GIAO TÁC

## Triggers

### Cơ sở lý thuyết

Trigger có thể hiểu đơn giản là một thủ tục SQL được thực thi ở phía server khi có một sự kiện như Insert, Delete, hay Update. Tuy nhiên khác với Stored Procedure, Trigger hoàn toàn không có tham số. Chúng ta không thể gọi thực hiện trực tiếp Trigger bằng lệnh EXECUTE như Strore Procedure hoặc bằng bấy kỳ một lệnh nào khác. Thay vào đó Trigger sẽ được thực hiện một cách tự động. Trigger được lưu trữ và quản lý trong Server Database, được dùng trong trường hợp ta muốn kiểm tra các ràng buộc toàn vẹn trong Database.



Hình 2: Cấu trúc của trigger trong Oracle

### Danh sách các trigger

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên trigger | Kích hoạt | Mô tả |
| 1 | UD\_GHE | Sau khi update trên bảng VE | Cập nhật tình trạng ghế sau khi khách hàng hủy vé |
| 2 | KT\_VE | Trước khi insert hoặc update trên bảng CHITIETVE | Kiểm tra và tính giá vé |
| 3 | GHE\_CB | Sau khi insert trên bảng VE | Cập nhật số lượng ghế sau khi thêm vé |
| 4 | CK\_TG\_CB | Sau khi insert trên bảng chuyến bay | Kiểm tra RBTV : Thời gian bay tối thiểu là 30 phút |
| 5 | CK\_TUOI\_KH | Sau khi inseret hoặc update trên bảng KHACHHANG | Kiểm tra RBTV : khách hàng đặt vé phải trên 18 tuổi |

Bảng 12: Danh sách trigger

### Mô tả một số trigger

Tên Trigger: KT\_VE

Thao tác: Insert or Update

Trên bảng: CHITIETVE

Nội dung: Khi thêm mới hoặc cập nhật bảng CHITIETVE thì cập nhật giá vé nếu tình trạng vé khác 0.

Code:

|  |
| --- |
| CREATE OR REPLACE FUNCTION UP\_GIA (V\_MAVE CHITIETVE.MAVE%TYPE)  RETURN CHITIETVE.GIA%TYPE  IS  V\_GIA CHITIETVE.GIA%TYPE;  BEGIN  SELECT CB.GIAVE \* ((100 - KM.GIAMGIA) / 100) \* (VE.TYLE / 100)  INTO V\_GIA  FROM CHITIETVE CT, VE, CHUYENBAY CB, KHUYENMAI KM  WHERE (CT.MAVE = V\_MAVE) AND  (CT.MAVE = VE.MAVE) AND  (VE.MACB = CB.MACB) AND  (CT.MAKM = KM.MAKM);    RETURN V\_GIA;    END; |
| CREATE OR REPLACE TRIGGER KT\_VE  BEFORE INSERT OR UPDATE  ON CHITIETVE  FOR EACH ROW  DECLARE  V\_TINHTRANG VE.TINHTRANG%TYPE;  BEGIN  SELECT VE.TINHTRANG FROM CHITIETVE CT JOIN VE ON CT.MAVE = VE.MAVE  INTO V\_TINHTRANG  WHERE CT.MAVE = :NEW.MAVE;  IF (V\_TINHTRANG != 0)  THEN  BEGIN  UPDATE CHITIETVE CT  SET CT.GIA = UP\_GIA(:NEW.MAVE)  WHERE CT.MAVE = :NEW.MAVE;  DBMS\_OUTPUT.PUT('THANH CONG!');  END;  ELSE  BEGIN  DBMS\_OUTPUT.PUT('LOI: TINH TRANG VE = 0');  ROLLBACK;  END;  END IF;    END; |

Thực hiện:

[B1] Kiểm tra tình trạng vé.

[1.1] Nếu tình trạng vé khác 0, sang bước 2.

[1.2] Ngược lại, bỏ qua.

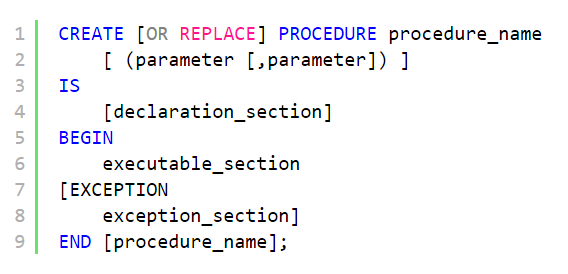
[B2] Cập nhật giá vé, sử dụng hàm UP\_GIA().

## Stored procedures

### Cơ sở lý thuyết

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.

Stored procudure sẽ giúp các ứng dụng nhìn minh bạch hơn, nghĩa là khi ta định nghĩa các thao tác xử lý vào một Stored thì công việc của các ngôn ngữ lập trình khác chỉ quan tâm đến tên thủ tục, các tham số truyền vào chứ không cần biết nó thực hiện như thế nào. Mỗi thủ tục sẽ có các mức độ truy cập, nghĩa là ta có thể cấp quyền sử dụng cho một User nào đó trong hệ quản trị.



Hình 3: Cấu trúc của stored procedure trong Oracle

### Danh sách các stored procedure

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên procedure | Tham số | Mô tả |
| 1 | danhsachchuyenbay | Điểm đi, điểm đến, số vé, loại vé, ngày khởi hành | Truy vấn danh sách chuyến bay |
| 2 | danhsachchuyenbaykh | Điểm đi, điểm đến, hãng máy bay, số vé, loại vé, ngày khởi hành | Truy vấn danh sách chuyến bay khứ hồi |
| 3 | capnhattienve | Mã vé, số tiền mới | Cập nhật số tiền mới cho một mã vé cụ thể |
| 4 | danhsachchuyenbay | Điểm đi, điểm đến, số vé, loại vé, ngày khởi hành | Truy vấn danh sách chuyến bay (tính năng tìm chuyến bay trong chức năng đặt vé) |
| 5 | danhsachchuyenbaykh | Điểm đi, điểm đến, số vé, loại vé, ngày khởi hành, hãng máy bay | Truy vấn danh sách chuyến bay khứ hồi |
| 6 | danhsachvecua1cb | Mã chuyến bay | Truy vấn danh sách vé của 1 chuyến bay cụ thể |
| 7 | doanhthutheocb | Mã chuyến bay | Tính tổng doanh thu của một chuyến bay cụ thể |
| 8 | doanhthutheonam | Năm bao nhiêu | Tính tổng doanh thu của một năm cụ thẻ |
| 9 | huyve | Mã vé | Hủy đặt vé |
| 10 | insert\_khuyen\_mai | Tên khuyến mãi, mô tả, giảm giá, hạn sử dụng, s1, tình trạng | Tạo mã khuyến mãi mới |
| 11 | khachhangphothong |  |  |
| 12 | Khachhangvip |  |  |
| 13 | laychuyenbaybangid | Mã chuyến bay | Truy vấn chuyến bay |
| 14 | Soluongvecuahang  mbtheongay | Ngày, mã hãng máy bay | Truy vấn số lượng vé của một hãng máy bay trong ngày cụ thể |
| 15 | suachuyenbay | Mã chuyến bay, mã hãng máy bay, điểm đi, điểm đến, ngày khỏi hành, thời gian bay, giá vé | Cập nhật thông tin cho chuyến bay |
| 16 | suagiakhuyenmai | Mã khuyến mãi, giảm giá | Cập nhật thông tin cho khuyến mãi |
| 17 | themchuyenbay | Mã chuyến bay, mã hãng máy bay, điểm đi, điểm đến, ngày khỏi hành, thời gian bay, giá vé | Thêm chuyến bay mới |
| 18 | Thongtinkhachhang  motchuyenbay | Mã chuyến bay | Truy vấn thông tin của tất cả hành khách trong một chuyến bay cụ thể |

Bảng 13: Danh sách stored procedure

### Mô tả một số stored procedure

* **Mua vé:**

Tên: insert\_ve()

Nội dung: Thực hiện khi khách hàng mua vé.

Tham số đầu vào: p\_macb in VARCHAR2, p\_sove in number, p\_makh in varchar2, p\_loaive in number, p\_km in nvarchar2

Tham số đầu ra: không có.

Script:

procedure insert\_ve(p\_macb in VARCHAR2,

p\_sove in number,

p\_makh in varchar2,

p\_loaive in number,

p\_km in nvarchar2)

as

p\_loaihk number;

v\_dem integer := 0;

v\_thanhtien number;

type arr\_type IS VARRAY (5) OF Ve.mave%

type;

mave\_arr arr\_type := arr\_type();

cursor c1 is select ve.mave

from ve

where MACB = p\_macb

and LOAIVE = p\_loaive

and TINHTRANG = 0;

counter integer := 0;

begin

select LOAIHK into p\_loaihk from KHACHHANG where CMND = p\_makh;

for n in c1

loop

counter := counter + 1;

mave\_arr.extend;

mave\_arr(counter) := n.MAVE;

v\_thanhtien := fn\_thanhtien(p\_macb, p\_loaihk, p\_km, mave\_arr(counter));

-- DBMS\_OUTPUT.put\_line(v\_thanhtien);

insert into CHITIETVE(MAVE, MACB, CMND, GIA, KHUYENMAI)

values (mave\_arr(counter), p\_macb, p\_makh, v\_thanhtien, p\_km);

update ve

set tinhtrang = 1

where mave = mave\_arr(counter);

end loop;

commit;

exception

WHEN DUP\_VAL\_ON\_INDEX THEN

DBMS\_OUTPUT.put\_line('trung khoa chinh');

WHEN OTHERS THEN

DBMS\_OUTPUT.put\_line('loi gi do');

ROLLBACK;

end insert\_ve;

Bước thực hiện:

1. Lấy loại khách hàng của khách hàng.
2. Thực hiện lặp.
3. Thêm mã vé vào mảng vé đã được khai báo trước đó.
4. Gọi hàm tính tiền để tính tổng tiền vé.
5. Thêm những vé trong mảng vào bảng CHITIETVE.
6. Cập nhật tình trạng vé thành 1( đã đặt) tương ứng.

* **Thêm chuyến bay**

Tên: themchuyenbay()

Nội dung: Tạo chuyến bay vô hệ thống.

Tham số đầu vào: p\_macb varchar2, p\_mahmb varchar2, p\_diemkh nvarchar2, p\_diemden nvarchar2, p\_ngaykh timestamp, p\_tgbay number, p\_giave number

Tham số đầu ra: không có.

Script:

create or replace procedure themchuyenbay

(

p\_macb varchar2,

p\_mahmb varchar2,

p\_diemkh nvarchar2,

p\_diemden nvarchar2,

p\_ngaykh timestamp,

p\_tgbay number,

p\_giave number

)

as

v\_dem number := 50;

v\_mave nvarchar2(10);

v\_soghe varchar2(2) := 0;

ex\_notfound exception ;

begin

insert into CHUYENBAY(macb, mahmb, diemkhoihanh, diemden, ngaykhoihanh, thoigianbay, giave)

values(p\_macb,p\_mahmb,p\_diemkh,p\_diemden,p\_ngaykh,p\_tgbay,p\_giave);

while v\_dem >= 1

loop

v\_mave := p\_macb||v\_dem;

if v\_dem <= 10 then

insert into VE(mave, macb, loaive, tinhtrang, tile,SOGHE) values (v\_mave, p\_macb,1,0,2,v\_soghe);

end if;

if v\_dem > 10 then

insert into VE(mave, macb, loaive, tinhtrang, tile,SOGHE) values (v\_mave, p\_macb,0,0,1,v\_soghe);

end if;

v\_soghe := v\_soghe + 1;

v\_dem := v\_dem - 1;

end loop;

commit;

exception

when others then

DBMS\_OUTPUT.put\_line('1 khong them chuyen bay duoc');

raise\_application\_error(-20343,'khong them chuyen bay duoc',true);

rollback ;

end;

Bước thực hiện:

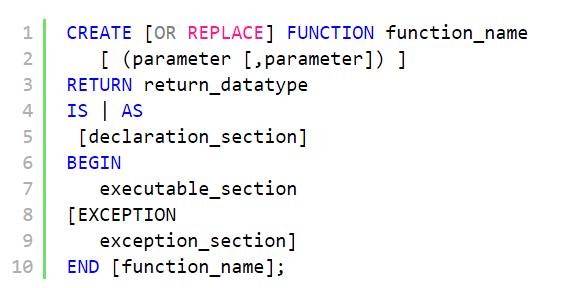
1. Thêm 1 chuyến bay mới vào bảng CHUYENBAY.
2. Thực hiện vòng lặp.
3. Gán mã vé là mã chuyến bay và số thứ tự.
4. 10 vé đầu tiên sẽ được thêm vào bảng VE với loại vé là vé VIP, còn lại là loại phổ thông.

## Functions

### Cơ sở lý thuyết

Function là một kiểu chương trình con, nó bao gồm một nhóm các lệnh PL/SQL. Mỗi thủ tục trong Oracle có tên riêng của nó mà nó có thể được gọi. Đơn vị chương trình con này được lưu trữ như là một đối tượng cơ sở dữ liệu.

Các hàm sẽ giúp đơn giản hóa chương trình và có thể sử dụng nhiều lần. Ví dụ bạn cần viết một loạt các câu lệnh SQL phức tạp lên đến hàng ngàn dòng thì có thể phân tích và chia nhỏ thành nhiều hàm. Nếu nói đến gom nhóm và sử dụng nhiều lần thì stored procedure cũng làm được, tuy nhiên procedure sẽ không có giá trị trả về, còn function thì sẽ có giá trị trả về.



Hình 4: Cấu trúc của function trong Oracle

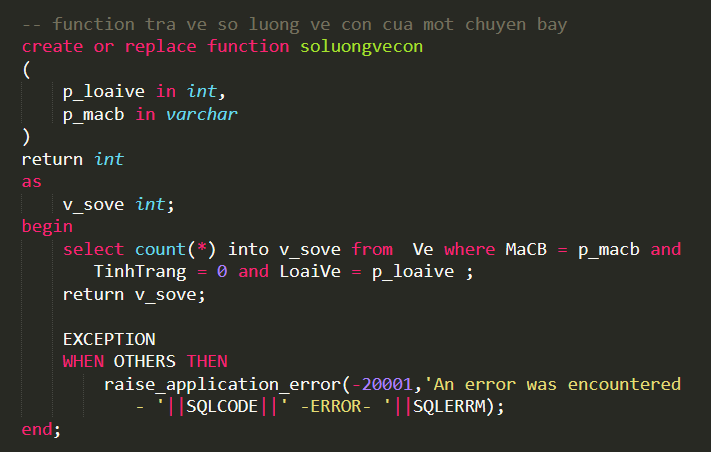
### Danh sách các function

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên function | Tham số | Mô tả |
| 1 | soluongvecon | Mã chuyến bay, loại vé | Trả về số lượng vé còn lại của một chuyến bay |

Bảng 14: Danh sách function

### Mô tả một số function

* Tên: soluongvecon()
* Nội dung: kiểm tra số lượng vé chưa được mua của một chuyến bay cụ thể
* Tham số đầu vào: mã chuyến bay, loại vé
* Kiểu dữ liệu trả về: int
* Code SQL:



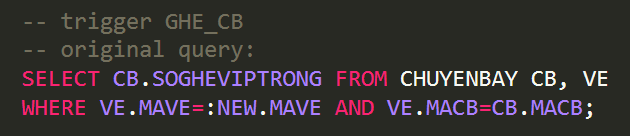
* Thực hiện:

Truy vấn số lượng vé trong bảng vé với mã chuyến bay và loại vé trùng với tham số đầu vào và tình trạng vé = 0 (tức chưa đặt)

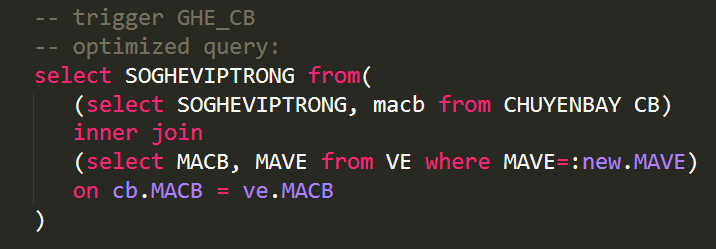
## Tối ưu hóa một số câu truy vấn

* Trigger GHE\_CB

Câu truy vấn ban đầu:

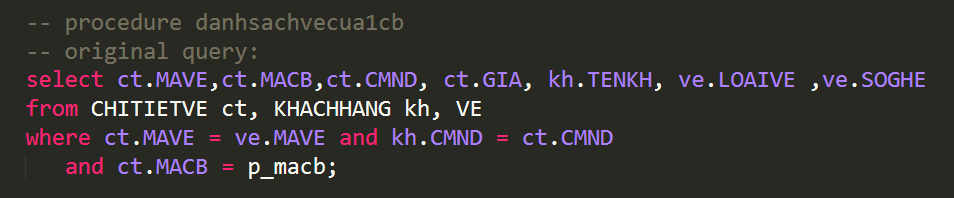


Câu truy vấn sau khi tối ưu hóa:

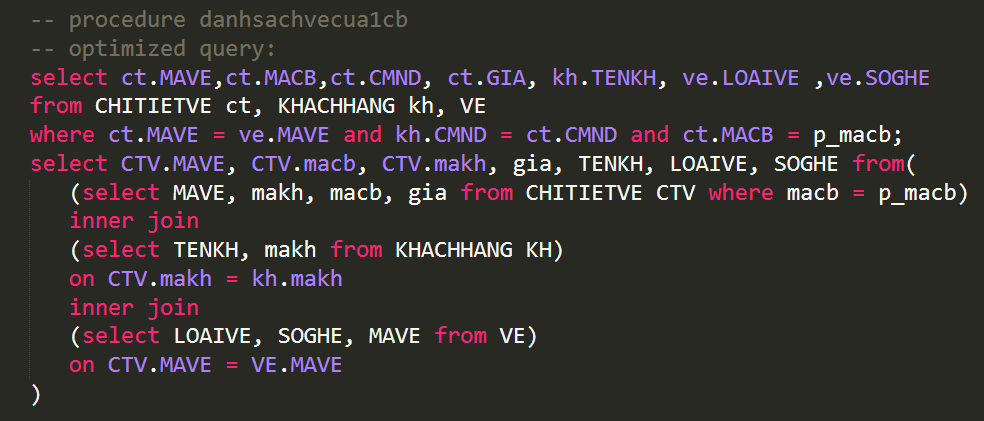


* Procedure danhsachvecua1cb

Câu truy vấn ban đầu:

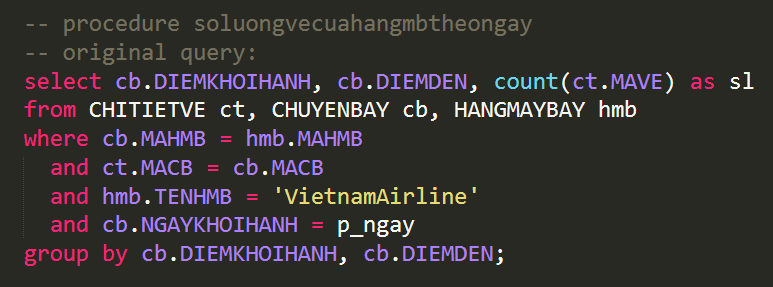


Câu truy vấn sau khi tối ưu hóa:

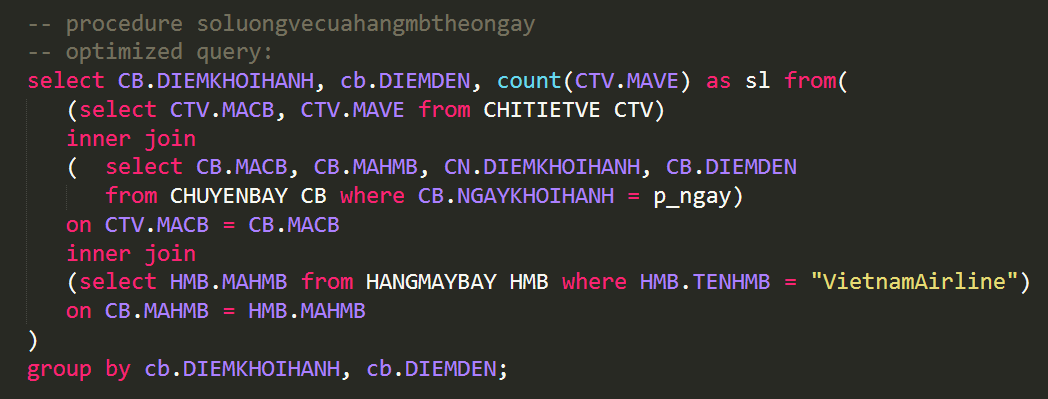


* Procedure soluongvecuahangmbtheongay

Câu truy vấn ban đầu:



Câu truy vấn sau khi tối ưu hóa:



# XỬ LÝ TRUY XUẤT ĐỒNG THỜI

## Các mức cô lập

### Read Uncommitted

Khi transaction thực hiện ở mức này, các truy vấn vẫn có thể truy nhập vào các bản ghi đang được cập nhật bởi một transaction khác và nhận được dữ liệu tại thời điểm đó mặc dù dữ liệu đó chưa được commit (uncommited data). Nếu vì lý do nào đó transaction ban đầu rollback lại những cập nhật, dữ liệu sẽ trở lại giá trị cũ. Khi đó transaction thứ hai nhận được dữ liệu sai.

* Đặc điểm:
* Không thiết lập Shared Lock trên những đơn vị dữ liệu cần đọc. Do đó không phải chờ khi đọc dữ liệu (kể cả khi dữ liệu đang bị lock bởi giao tác khác)
* Vẫn tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác
* Ưu điểm
* Tốc độ xử lý rất nhanh
* Không cản trở những giao tác khác thực hiện việc cập nhật dữ liệu
* Nhược điểm
* Có khả năng xảy ra mọi vấn đề khi xử lý đồng thời : Dirty Reads, Unrepeatable Reads, Phantoms, Lost Updates

### Read Committed

Đây là mức isolation mặc định, nếu bạn không đặt gì cả thì transaction sẽ hoạt động ở mức này. Transaction sẽ không đọc được dữ liệu đang được cập nhật mà phải đợi đến khi việc cập nhật thực hiện xong. Vì thế nó tránh được dirty read như ở mức trên.

* Đặc điểm

Đây là mức độ cô lập mặc định của SQL Server

* Tạo Shared Lock trên đơn vị dữ liệu được đọc, Shared Lock được giải phóng ngay sau khi đọc xong dữ liệu
* Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác
* Ưu điểm
* Giải quyết vấn đề Dirty Reads.
* Shared Lock được giải phóng ngay, không cần phải giữ cho đến hết giao tác nên không cản trở nhiều đến thao tác cập nhật của các giao tác khác.
* Nhược điểm
* Chưa giải quyết được vấn đề Unrepeatable Reads, Phantoms,Lost Updates
* Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)

### Repeatable Read

Mức isolation này hoạt động nhứ mức read commit nhưng nâng thêm một nấc nữa bằng cách ngăn không cho transaction ghi vào dữ liệu đang được đọc bởi một transaction khác cho đến khi transaction khác đó hoàn tất.

* Đặc điểm
* Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác => Các giao tác khác phải chờ đến khi giao tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này .

(Repeatable Read = Read Committed + Giải quyết Unrepeatable Reads).

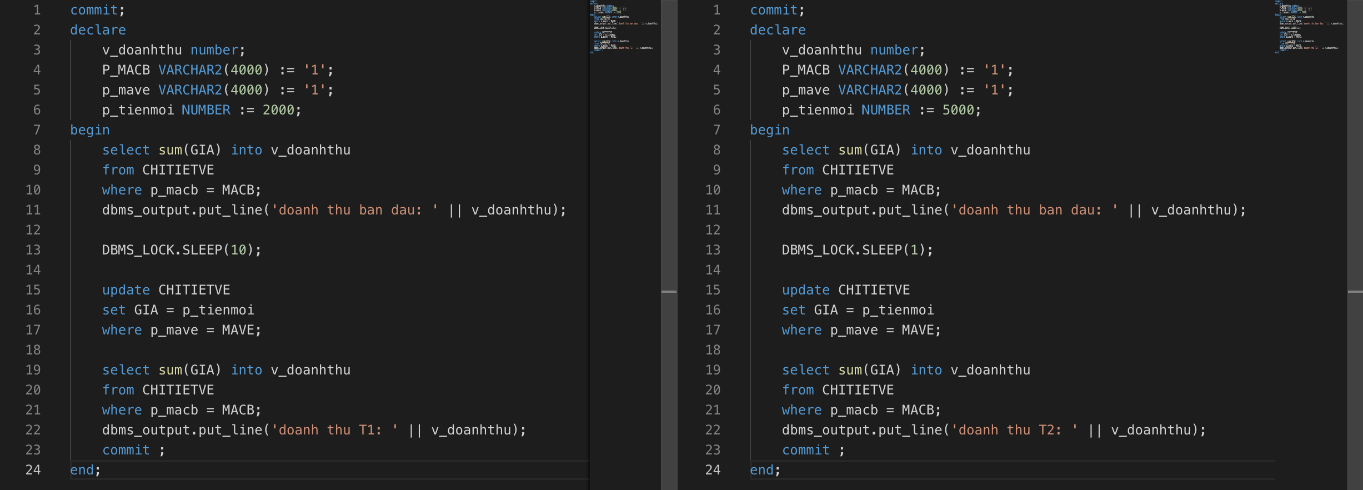
* Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.
* Ưu điểm
* Giải quyết vấn đề Dirty Reads và Unrepeatable Reads
* Nhược điểm
* Chưa giải quyết được vấn đề Phantoms, do vẫn cho phép insert những dòng dữ liệu thỏa điều kiện thiết lập shared lock
* Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)
* Shared lock được giữ đến hết giao tác ==> cản trở việc cập nhật dữ liệu của các giao tác khác

### Serializable

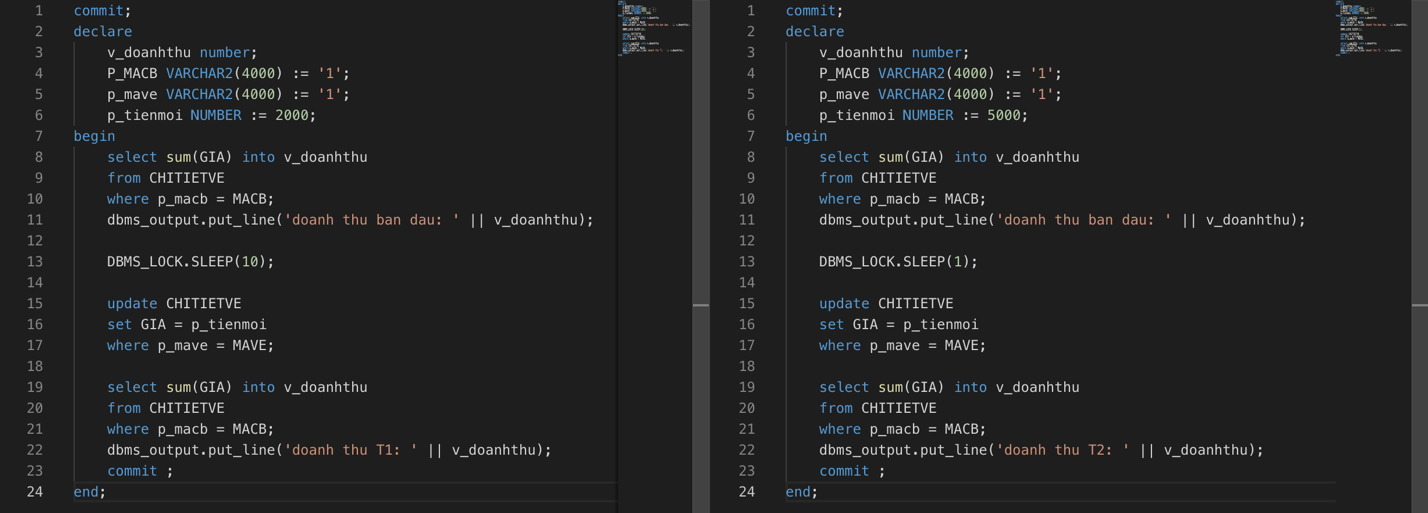
Mức isolation này tăng thêm một cấp nữa và khóa toàn bộ dải các bản ghi có thể bị ảnh hưởng bởi một transaction khác, dù là UPDATE/DELETE bản ghi đã có hay INSERT bản ghi mới.

* Đặc điểm
* Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác => Các giao tác khác phải chờ đến khi giao tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này.
* Không cho phép Insert những dòng dữ liệu thỏa mãn điều kiện thiết lập Shared Lock (sử dụng Key Range Lock) ==> Serializable = Repeatable Read + Giải quyết Phantoms
* Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.
* Ưu điểm
* Giải quyết thêm được vấn đề Phantoms
* Nhược điểm
* Phải chờ nếu đơn vị dữ liệu cần đọc đang được giữ khoá ghi (xlock)
* Cản trở nhiều đến việc cập nhật dữ liệu của các giao tác khác

## Lost update

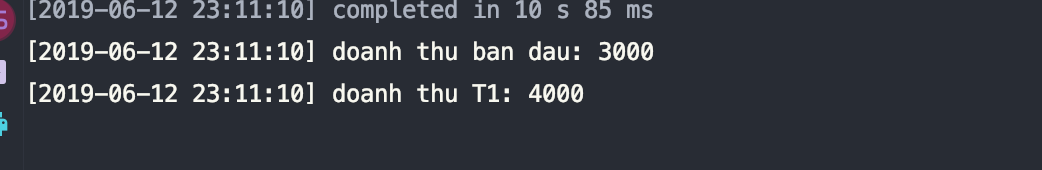


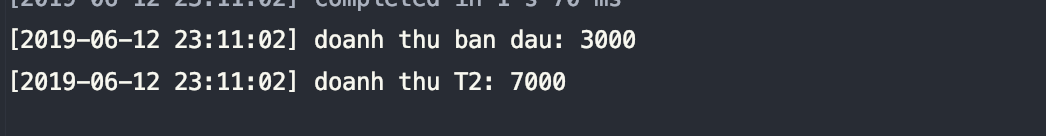
* Tình huống: Khi cập nhật tổng giá tiền của 1 vé, T1 cập nhật lên 2000. T2 cập nhật lên 5000. Giả sử T1 bị delay 10s. Thì T2 thực hiện update trước. sau đó T1 lấy lại giá trị cũ để cập nhật. Trong trường hợp này. Cả hai đều tăng giá tiền. Nhưng sau khi cập nhật, dẫn đến kết quả tổng doanh thu sai. Mất dữ liệu cập nhật của T2.



* Nguyên nhân: T1 và T2 đọc giá trị doanh thu ban đầu. Tuy nhiên T2 không nhận biết được T1 đã cập nhật doanh thu mới trong khi T2 vẫn sử dụng giá trị doanh thu đọc ban đầu. Vì vậy dữ liệu doanh thu đã bị cập nhật sai.
* Hướng giải quyết :

Cách 1: Khi sử dụng Read uncommitted hoặc Read committed, ta chuyển thành Update lock, từ đó T2 sẽ phải chờ T1 hoàn tất để nhả khóa.





## Dirty read

* Tình huống:

|  |  |  |
| --- | --- | --- |
| T1 | T2 | Giải thích |
| SELECT MaCB, GiaVe FROM ChuyenBay WHERE MaCB IN (‘100’, ‘200’, ‘300’);   |  |  | | --- | --- | | ChuyenBay | | | MaCB | GiaVe | | 100 | 1200000 | | 200 | 1000000 | |  | T1 thực hiện câu truy vấn cho chuyến bay ‘100’, ‘200’, ‘300’. Không tìm thấy chuyến bay ‘300’. |
| INSERT INTO ChuyenBay (MaCB, GiaVe) VALUES (300, 1300000); |  | T1 thêm 1 dòng dữ liệu vào bảng ChuyenBay, nhưng chưa commit. |
|  | SELECT MaCB, GiaVe FROM ChuyenBay WHERE MaCB IN (‘100’, ‘200’, ‘300’);   |  |  | | --- | --- | | ChuyenBay | | | MaCB | GiaVe | | 100 | 1200000 | | 200 | 1000000 | | 300 | 1300000 | | T2 thực hiện câu truy vấn cho chuyến bay ‘100’, ‘200’, ‘300’. |
|  | UPDATE ChuyenBay  SET GiaVe=990000  WHERE MaCB=’100’; | T2 cập nhật giá vé của chuyến bay ‘100’. |
| ROLLBACK; |  | T1 hủy giao tác. |
|  | COMMIT; | T2 commit dữ liệu. |
|  | SELECT MaCB, GiaVe FROM ChuyenBay WHERE MaCB IN (‘100’, ‘200’, ‘300’);   |  |  | | --- | --- | | ChuyenBay | | | MaCB | GiaVe | | 100 | 990000 | | 200 | 1000000 | | T2 thực hiện lại truy vấn cho chuyến bay ‘100’, ‘200’, ‘300’, nhưng không tìm thấy ‘300’. Dữ liệu ban đầu T2 đọc chưa được commit mà bị hủy bỏ nên ra kết quả dữ liệu không chính xác. |

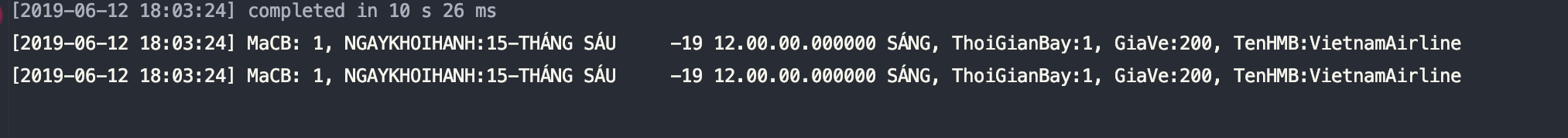
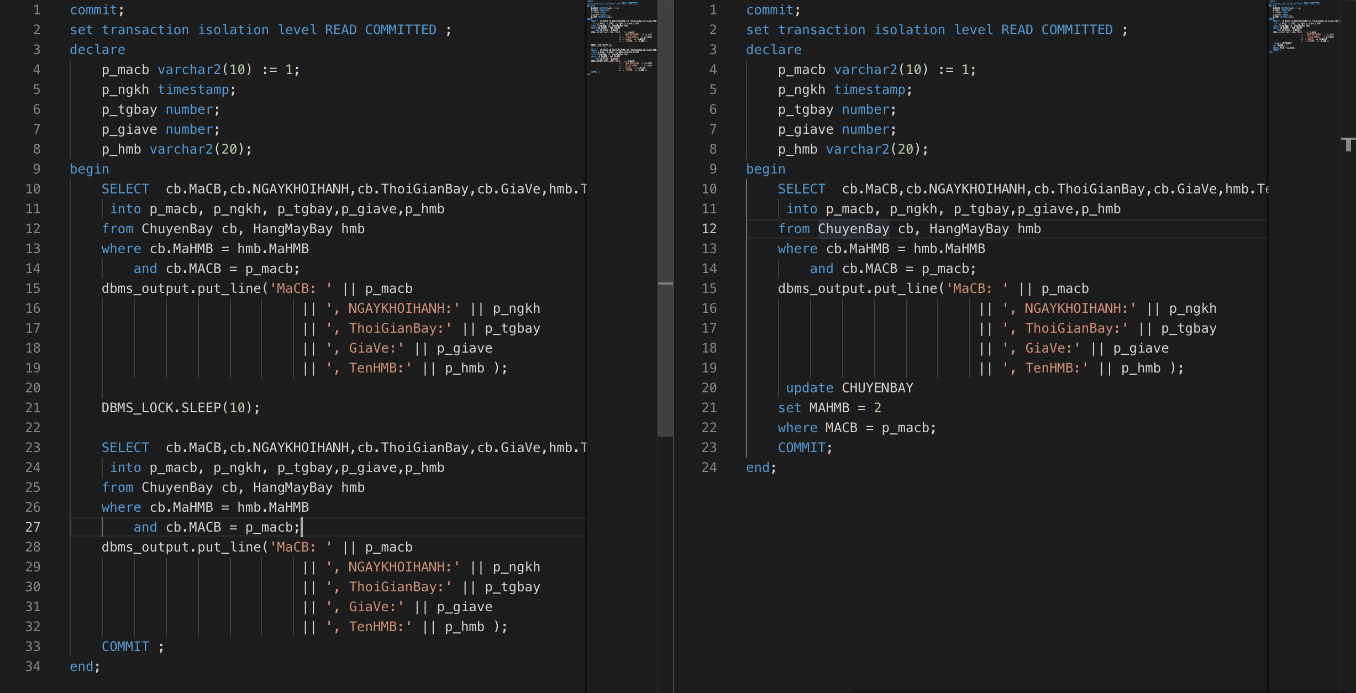
* Nguyên nhân: Khi T1 cập nhật dữ liệu, T2 có thể đọc được dữ liệu vừa cập nhật của T1, tuy nhiên dữ liệu đó chưa được T1 commit (chế độ Read Uncommitted). Sau đó T1 hủy giao tác (rollback) nên dữ liệu T2 đọc được là vô nghĩa (dữ liệu rác). -> Xảy ra hiện tượng Dirty Read.
* Hướng giải quyết: Sử dụng mức cô lập Read Committed, Repeatable Read hoặc Serializable để giải quyết vấn đề này.
* Kết quả:

|  |  |  |
| --- | --- | --- |
| T1 | T2 | Giải thích |
| SELECT MaCB, GiaVe FROM ChuyenBay WHERE MaCB IN (‘100’, ‘200’, ‘300’);   |  |  | | --- | --- | | ChuyenBay | | | MaCB | GiaVe | | 100 | 1200000 | | 200 | 1000000 | |  | T1 thực hiện câu truy vấn cho chuyến bay ‘100’, ‘200’, ‘300’. Không tìm thấy chuyến bay ‘300’. |
| INSERT INTO ChuyenBay (MaCB, GiaVe) VALUES (300, 1300000); |  | T1 thêm 1 dòng dữ liệu vào bảng ChuyenBay, nhưng chưa commit. |
|  | SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  --SET TRANSACTION ISOLATION LEVEL REPEATABLE READ/ SERIALIZABLE;  SELECT MaCB, GiaVe FROM ChuyenBay WHERE MaCB IN (‘100’, ‘200’, ‘300’);   |  |  | | --- | --- | | ChuyenBay | | | MaCB | GiaVe | | 100 | 1200000 | | 200 | 1000000 | | T2 đặt mức cô lập Read Committed (hoặc Repeatable Read hoặc Serializable).  T2 thực hiện câu truy vấn cho chuyến bay ‘100’, ‘200’, ‘300’, không tìm thấy chuyến bay ‘300’ vì dữ liệu chưa được T1 commit. |

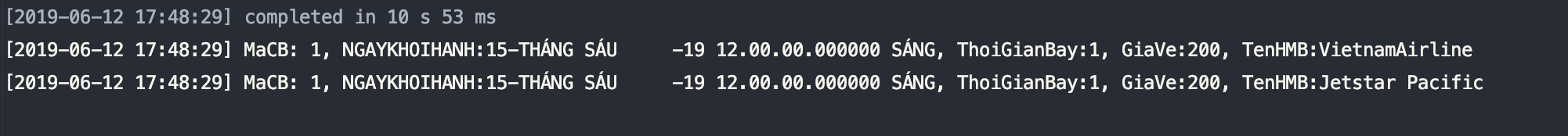
* Tuy nhiên, Oracle sử dụng cơ chế Read Committed làm cơ chế mặc định (default) nên đã giải quyết được vấn đề Dirty Read.

## Unrepeatable read

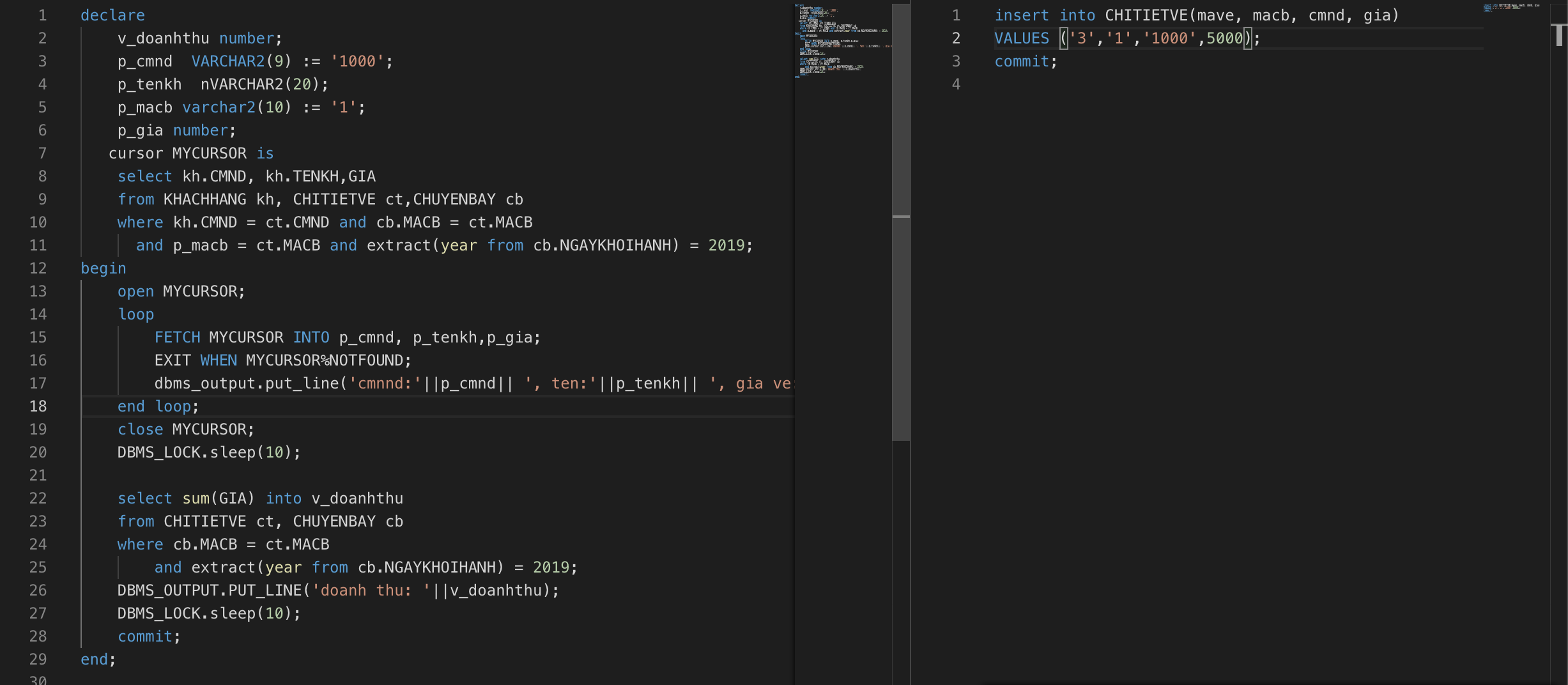
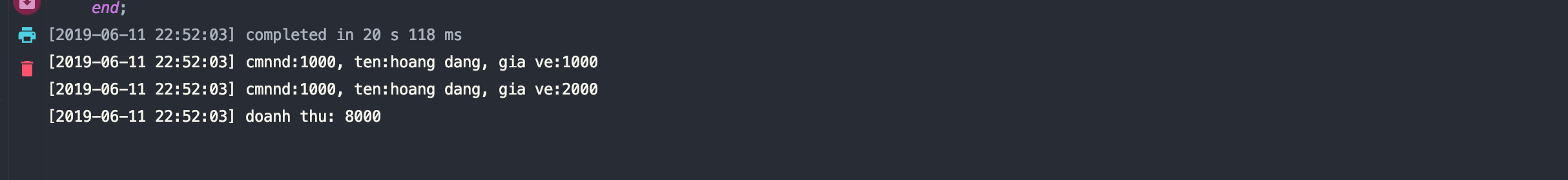
* Tình huống: Khi thay đổi hãng máy bay của một chuyến bay. Ban đầu T1 đọc thông tin chuyến bay, sau đó delay. T2 vào thay đổi thông tin chuyến bay, sau đó commit lên CSDL. T1 trở lại không thể đọc lại dữ liệu nhưng không thể nhìn thấy dữ liệu ban đầu.

**

* Nguyên nhân: T1 thiết lập mức cô lập read committed nên mỗi câu lệnh SELECT trong cùng 1 thao tác sẽ thiết lập snapshot của chính nó dù những câu lệnh này đọc trên những đơn vị dữ liệu giống nhau.
* Hướng giải quyết : Dùng mức cô lập Serializable.Khi chạy dòng lệnh select sẽ tạo Share lock và giữ Share lock này đến hết giao tác. Khi T2 vào update thông tin. T2 phải đợi T1 thực hiện xong mới update. → Dữ liệu đúng.
* Kết quả:



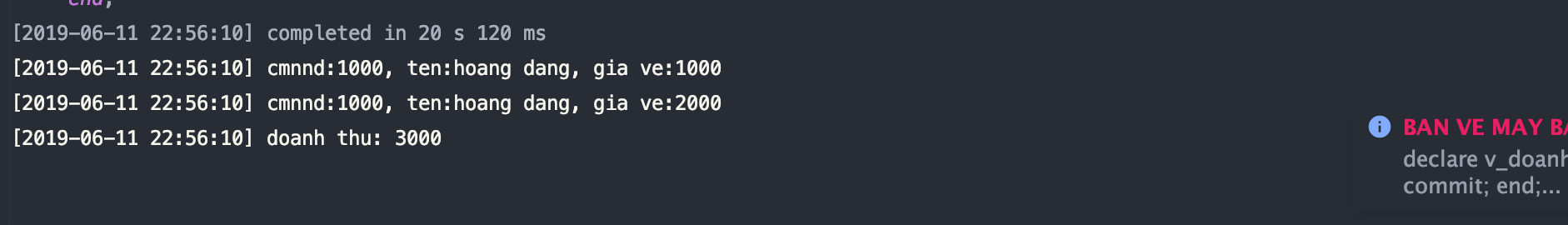
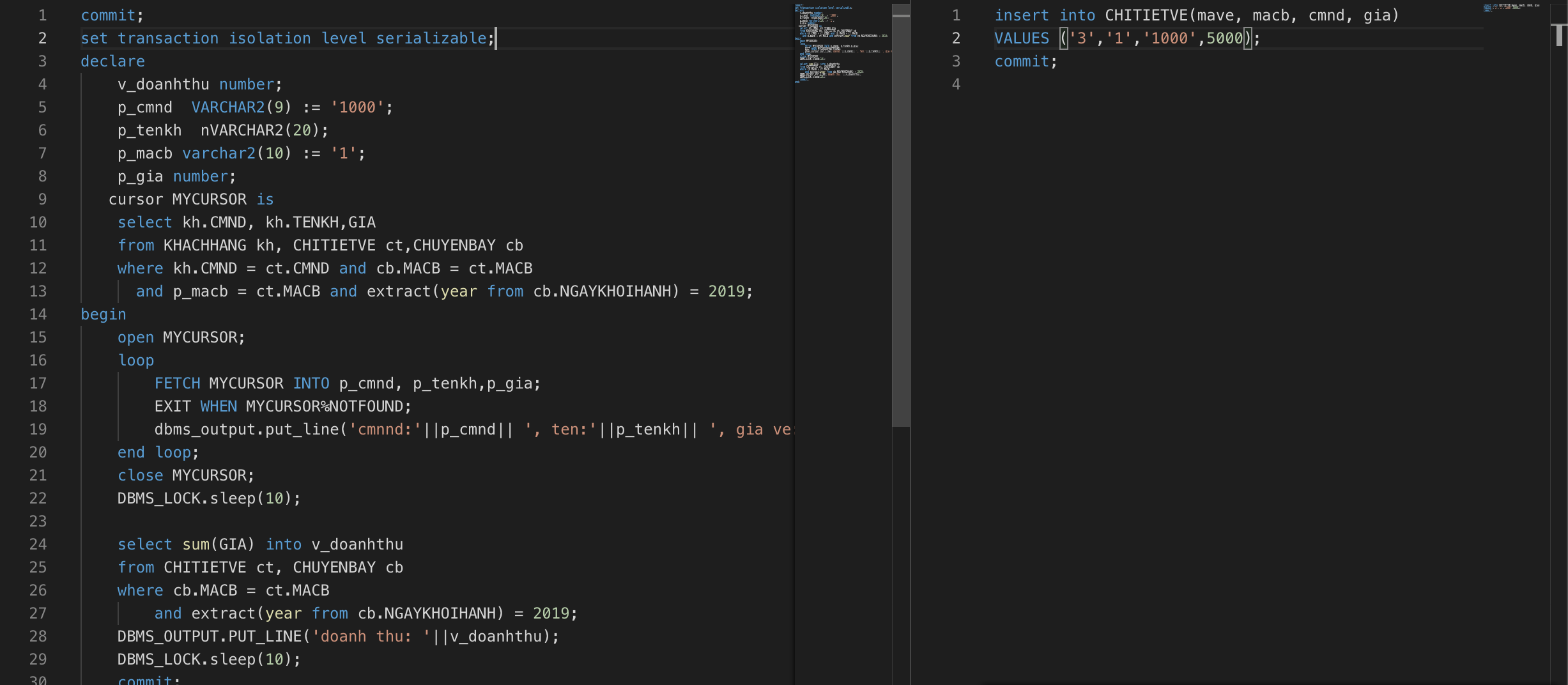
## Phantom

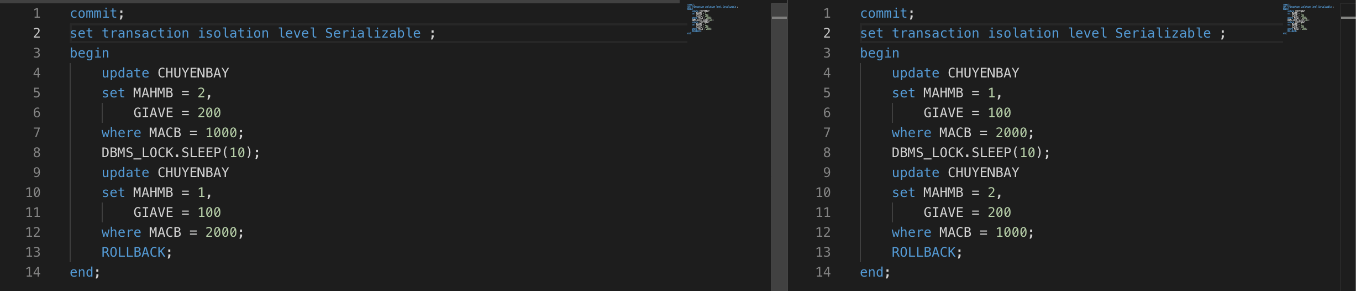
* Tình huống: Trong khi T1 đang xem danh sách vé và tính doanh thu, T2 thực hiện đặt vé (thêm vé mới). Điều này làm T1 nhìn thấy bị sai -> Xảy ra hiện tượng Phantom.

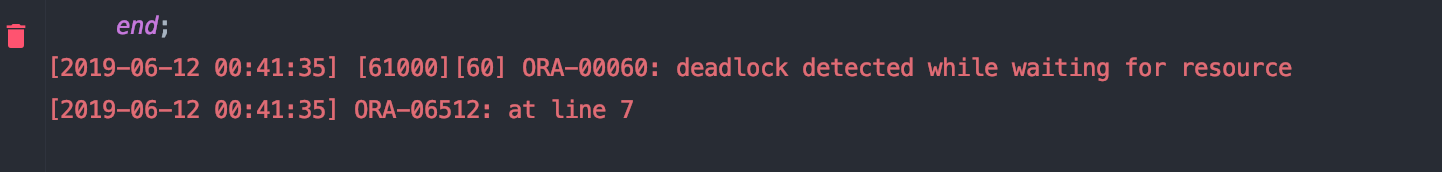
Tại đây phải có 3 dòng vé. Tuy nhiên T1 chỉ thấy có 2 dòng vé và tổng doanh thu bị sai.

* Nguyên nhân: T1 thiết lập mức cô lập read committed nên khi tính tổng doanh thu thì sẽ tính cả số tiền vừa thu được từ việc bán thức ăn ở T2.
* Hướng giải quyết : Dùng mức cô lập Serializable. T2 phải chờ đến khi T1 thực hiện xong rồi mới thực hiện.
* Kết quả: Danh sách vé sẽ không tính việc bán vé ở T2. Vì lúc này T1 dùng dữ liệu trước khi T2 thực thi.

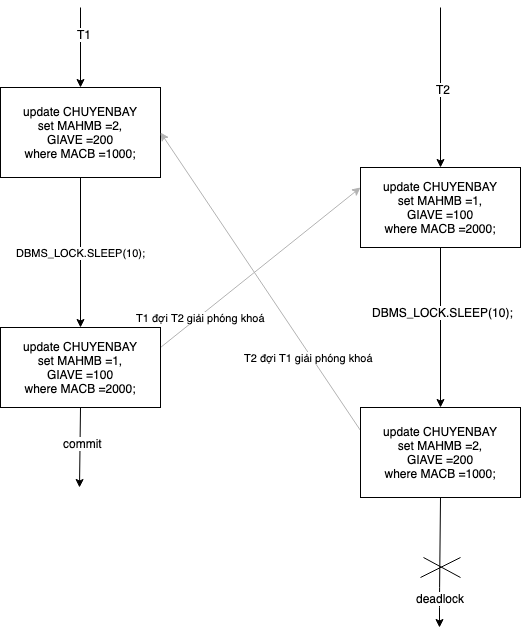
## Deadlock

* Tình huống: Trong khi T1 đang xem cập nhật chuyến bay 1000 sau đó cập nhật tiếp chuyến bay 1000 , thì T2 thực hiện cập nhật lần lượt chuyến bay số 2000 và 1000. Điều này khiến cho T1 chờ tài nguyên của T2. T2 chờ tài nguyên của T1. → xảy ra deadlock.





* Nguyên nhân:



Hình 5: Nguyên nhân gây deadlock

* Hướng giải quyết : Để giải quyết deadlock, hệ quản trị sẽ tự động rollback transaction có kích cỡ nhỏ hơn. Kích cỡ của 1 transaction được xác định bởi số dòng insert, update or delete. Trong trường hợp này, 2 transaction có kích cỡ bằng nhau nên hệ quản trị sẽ rollback transaction T2 thực hiện sau. Để ngăn ngừa deadlock, ta giảm mức cô lập xuống Repeatable Read.
* Kết quả: Danh sách vé sẽ không tính việc bán vé ở T2. Vì lúc này T1 dùng dữ liệu trước khi T2 thực thi.

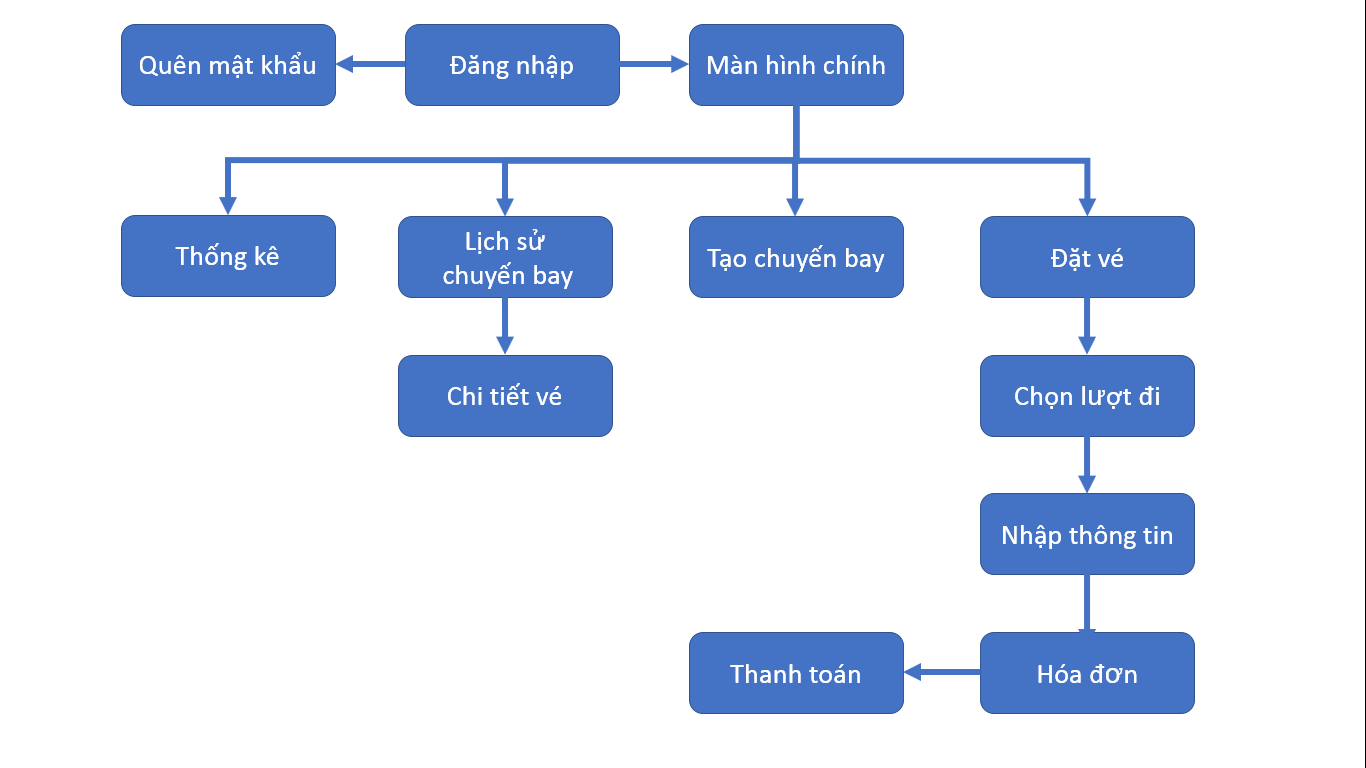
# THIẾT KẾ GIAO DIỆN

## Danh sách các màn hình

|  |  |  |
| --- | --- | --- |
| STT | Tên màn hình | Mô tả chức năng |
| 1 | Đăng nhập | Đăng nhập vào màn hình chính |
| 2 | Quên mật khẩu | Lấy lại mật khẩu cho tài khoản |
| 3 | Man hình chính | Màn hình chính để truy cập các tính năng chính của phần mềm |
| 4 | Đặt vé | Nhập thông tin để tìm kiếm chuyến bay |
| 5 | Chọn lượt đi | Chọn chuyến bay phù hợp |
| 6 | Nhập thông tin | Nhập thông tin từng khách hàng |
| 7 | Hóa đơn | Xuất hóa đơn sau khi xác định được chuyến bay và thông tin khách hàng |
| 8 | Thanh toán | In ra thông tin thanh toán |
| 9 | Tạo chuyến bay | Dùng cho nhân viên tạo chuyến bay mới |
| 10 | Thống kê | Thống kê theo từng năm, tháng,… |
| 11 | Lịch sử chuyến bay | Nhập mã chuyến bay để xem lịch sử chuyến bay |
| 12 | Chi tiết vé | Hiện thông tin chi tiết vé đã đặt |

Bảng 15: Danh sách các màn hình

## Sơ đồ liên kết màn hình

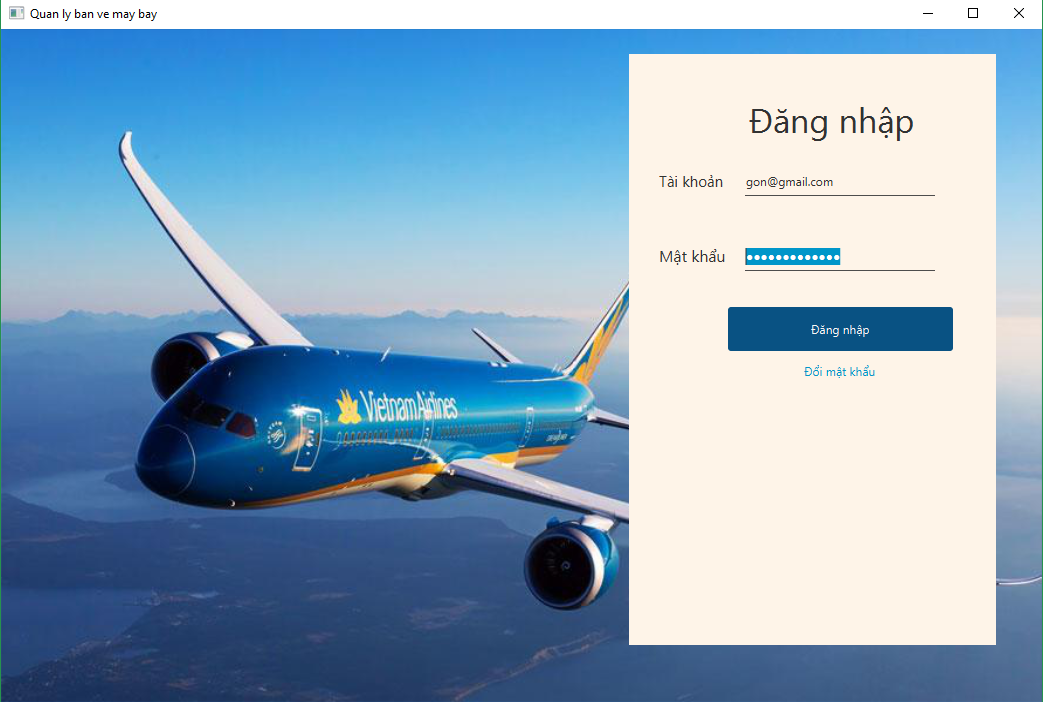


Hình 6: Sơ đồ liên kết màn hình

## Mô tả chức năng từng màn hình

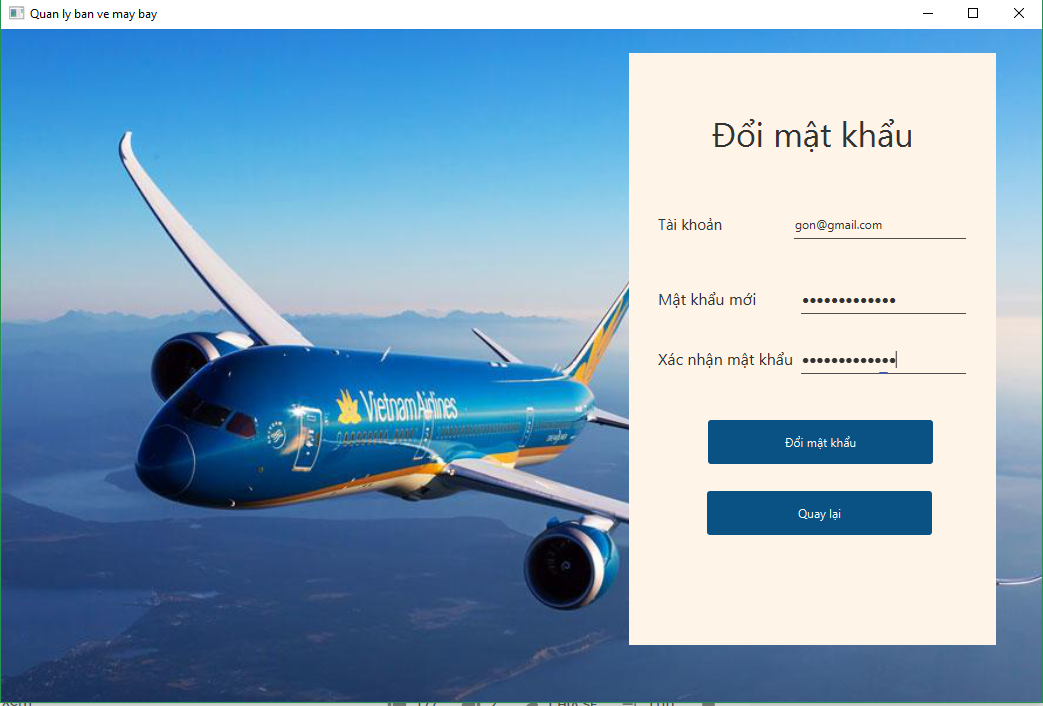
* Màn hình đăng nhập

Chức năng: Đăng nhập bằng tài khoản nhân viên. Bấm nút đăng nhập để vào màn hình chính, click vào “Đổi mật khẩu” nếu nhân viên quên mật khẩu.

****

* Màn hình đổi mật khẩu

Chức năng: đổi mật khẩu nếu nhân viên quên mật khẩu. Bấm vào nút đổi mật khẩu để thực hiện, bấm nút quay lại để trở về màn hình đăng nhập.

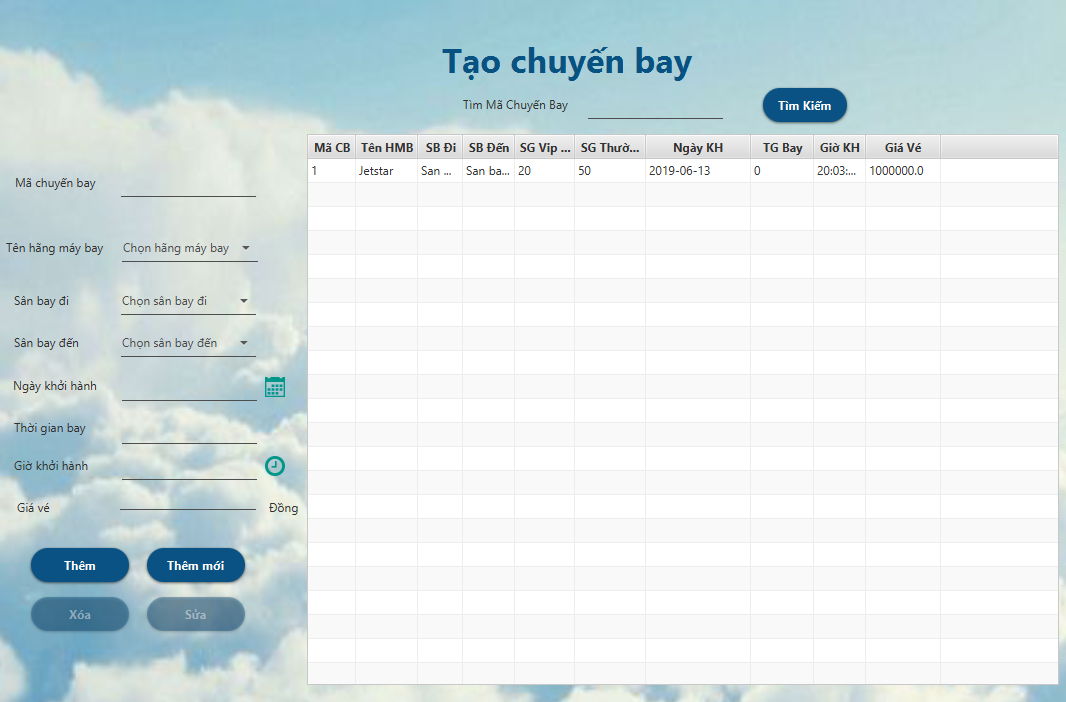


* Màn hình chính

Chức năng: trên màn hình có các tùy chọn Đặt vé, Tạo chuyến bay, Thống kê, Lịch sử chuyến bay, tương ứng với các chức năng của phần mềm.

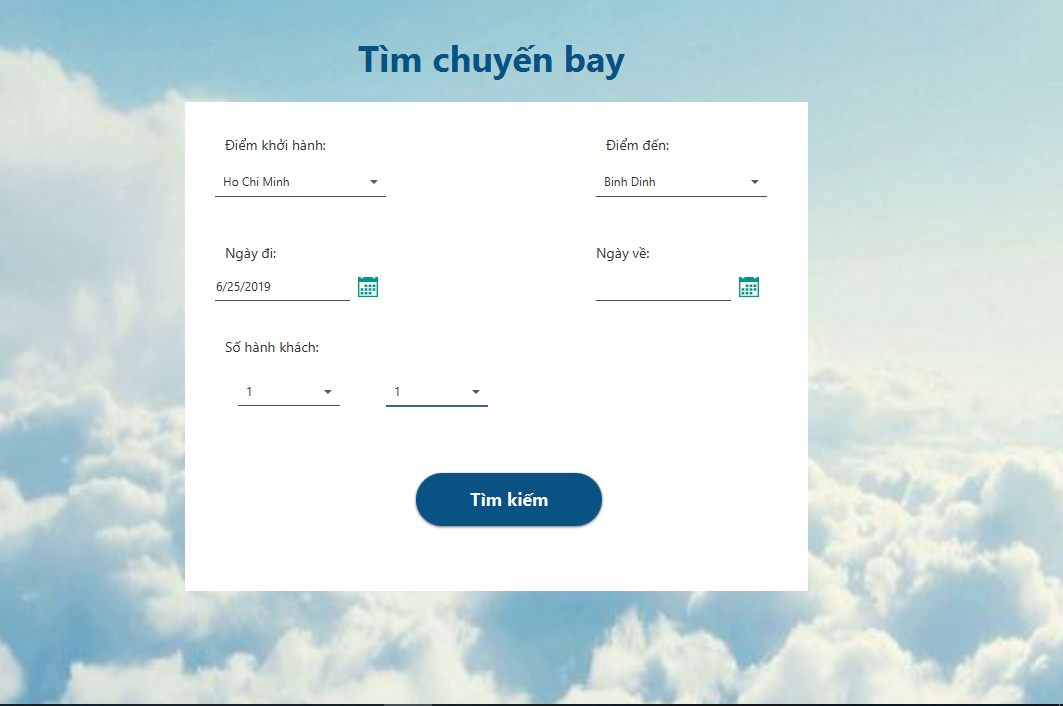
* Màn hình tạo chuyến bay

Chức năng: tạo chuyến bay mới, tìm kiếm và chỉnh sửa các chuyến bay đã tạo. nhập mã chuyến bay và nhấn nút tìm kiếm để tìm kiếm chuyến bay, nhấn nút thêm sau khi điền thông tin để tạo chuyến bay mới, ấn nút xóa để xóa chuyến bay đã tạo.



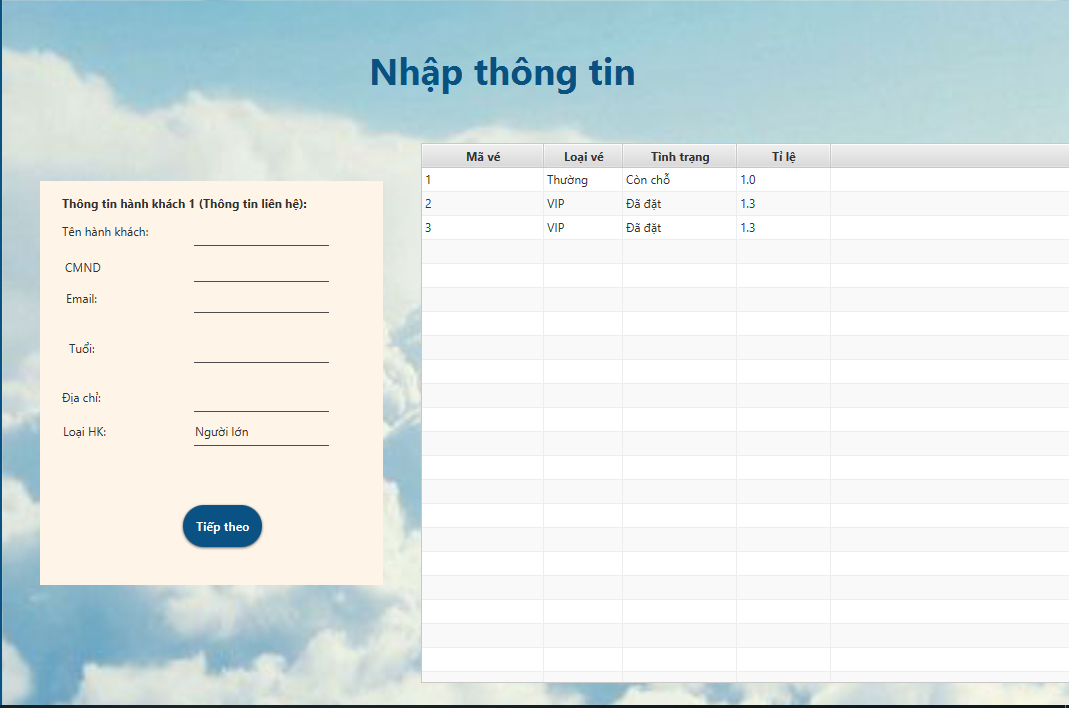
* Màn hình tìm kiếm chuyến bay

Chức năng: nhập đầy đủ các thông tin chuyến bay mình cần và ấn nút Tìm kiếm, phần mềm sẽ truy vấn ra các chuyến bay phù hợp để ta lựa chọn. Sau khi tìm kiếm sẽ chuyển tới màn hình chọn chuyến bay.



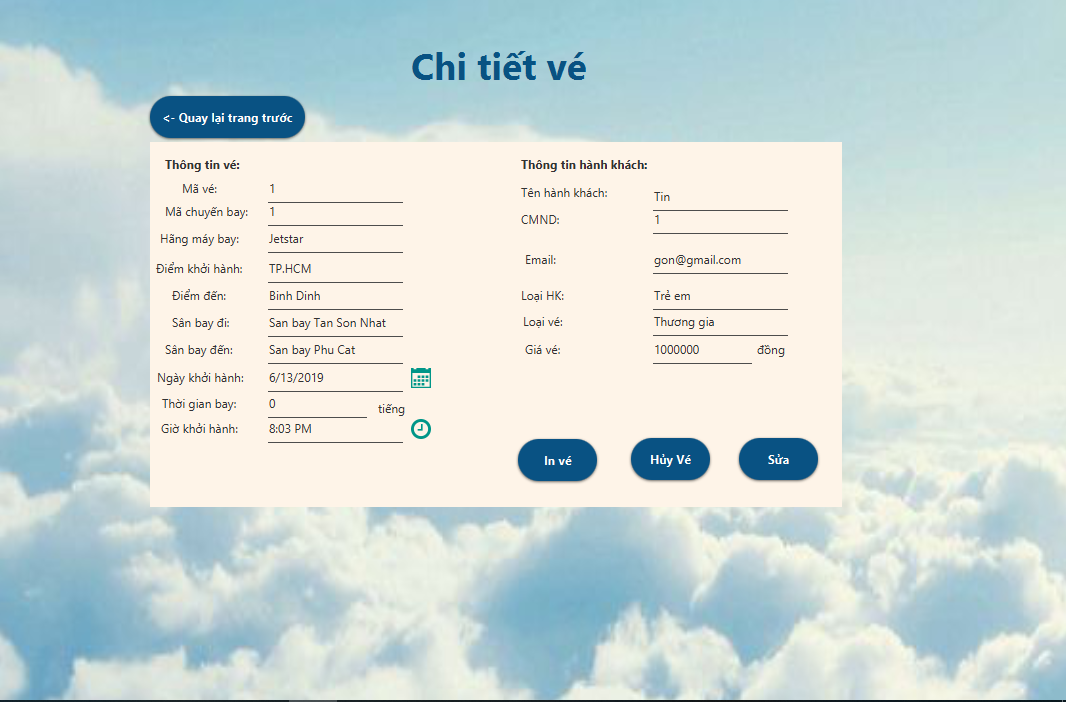
* Màn hình đặt vé

Chức năng: nhập thông tin từng khách hàng sau khi đã chọn chuyến bay. Nếu ở màn hình tìm kiếm chuyến bay, người dùng nhập số lượng khách hàng nhiều hơn 1, thì ở bước này ta nhập lần lượt thông tin từng khách hàng. Sau mỗi khách hàng, nhấn nút Tiếp theo đế nhập thông tin khách hàng kế tiếp. Bảng thông tin vé kế bên cho thấy ghế nào đã được đặt.



* Màn hình chi tiết vé

Chức năng: hiện thông tin chi tiết, đây là tính năng xem lại vé đã đặt cho khách hàng. Nhấn nút In vé để xuất file chi tiết vé dạng \*.pdf. nhấn nút Sửa để điều chỉnh thông tin vé. Nhấn nút Xóa để hủy đặt vé.



* Màn hình lịch sử bán vé

Chức năng: xem lịch sử bán vé của một chuyến bay cụ thể. Nhập mã chuyến bay vào và nhấn Tìm để tìm kiếm, nhấn Xem tất cả để xem tất cả vé. Nhấn vào Xem chi tiết để chuyển tới màn hình chi tiết vé.



* Màn hình zzz

# TỔNG KẾT

## Kết quả đạt được

Trong học kỳ vừa qua, nhóm đã tìm hiểu vận dụng kiến thức về hệ quản trị cơ sở dữ liệu và đã đạt được một số kết quả như sau.

* Nắm vững kiến thức và có thể vận dụng, cài đặt stored procedure, trigger, function, transaction trong nhiều hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, Oracle.
* Trang bị thêm kiến thức về các chế độ khóa và mức cô lập trong Oracle.
* Giải quyết được các trường hợp xử lý đồng thời trên hệ quản trị trong môi trường giả lập.
* Xây dựng được phần mềm “Quản lý rạp vé máy bay” với các chức năng:
* Tạo chuyến bay
* Đặt vé
* Tra cứu vé
* Thống kê
* Khuyến mãi

Các chức năng hoạt động như ý muốn và có thể hỗ trợ nhiều người dùng đồng thời mà không gặp vấn đề về xung đột dữ liệu.

## Hạn chế

Do thời gian hạn chế, lượng công việc lớn, cùng với năng lực nhân lực vẫn chưa đủ kinh nghiệm, nhóm vẫn chưa thiết kế được một giao diện đẹp và mang lại trải nghiệm tốt nhất cho người dùng. Các câu lệnh P/L SQL vẫn chưa thực sự tối ưu.

Về vấn đề bảo mật, phần mềm cũng chưa có các cơ chế bảo mật tránh mối nguy hiểm như chèn mã độc vào câu lệnh SQL (SQL Injection). Hơn nữa mật khẩu của nhân viên vẫn lưu dưới dạng text, chưa được mã hóa tránh việc ăn cắp thông tin, cướp quyền truy cập tài khoản. Vấn đề sao lưu, phục hồi dữ liệu cũng chưa thể hiện được trong đồ án.

Ngoài ra, nhóm chưa xây dựng được mobile app và phiên bản web của hệ thống quản lý vé máy bay. Nếu làm được điều này thì hệ thống này sẽ tiếp cận tới nhiều người dùng hơn, và khiến nó trở nên tiện dụng hơn khi người dùng không cần phải có một chiếc máy tính thì mới có thể sử dụng được.

## Hướng phát triển

Để khắc phục các nhược điểm kể trên, ta cần giải quyết các vấn đề sau:

* Tối ưu hóa câu lệnh P/L SQL
* Mã hóa mật khẩu nhân viên
* Dùng các kỹ thuật phòng tránh SQL Injection
* Thiết kế lại UI/UX
* Xây dựng tính năng sao lưu, phục hồi
* Xây dựng phiên bản trên thiết bị di động
* Xây dựng phiên bản web

# BẢNG PHÂN CÔNG CÔNG VIỆC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Công việc | Hoàng | Tín | Linh | Pa |
| Tìm hiểu tổng quan đề tài | x | x | x | x |
| Phân tích yêu cầu | x | x | x | x |
| Thiết kế chức năng | x | x | x | x |
| Thiết kế CSDL | x | x | x | x |
| Xây dựng ràng buộc toàn vẹn |  |  |  | x |
| Xây dựng trigger |  |  | x |  |
| Xây dựng stored procedure | x |  | x |  |
| Xây dựng function | x |  | x |  |
| Tối ưu hóa câu truy vấn |  |  |  | x |
| Các mức cô lập |  |  |  | x |
| Giải quyết lost update | x |  | x |  |
| Giải quyết dirty read |  |  | x |  |
| Giải quyết unrepeatable read | x |  |  |  |
| Giải quyết phantom | x |  |  |  |
| Giải quyết deadlock | x |  |  |  |
| Thiết kế giao diện |  | x |  |  |
| Xây dựng giao diện phần mềm |  | x |  |  |
| Xây dựng chức năng phần mềm |  | x |  |  |
| Lập trình báo biểu | x | x |  |  |
| Kết nối CSDL |  | x |  |  |
| Chuẩn bị dữ liệu ban đầu | x | x | x | x |
| Tổng kết |  |  |  | x |
| Tổng hợp và viết báo cáo |  |  |  | x |

# TÀI LIỆU THAM KHẢO

* Documentation của Oracle: <https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/data-concurrency-and-consistency.html>
* Tối ưu hóa câu truy vấn : <https://sqlvietnam.wordpress.com/2016/01/09/thu-thuat-toi-uu-hoa-sql/>
* Isolation levels : <http://www.sqlviet.com/blog/cac-mc-isolation-level>