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

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN** **LÝ THUYẾT**

**BÁO CÁO**

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**Lớp: 20VP**

**20126038 – Nguyễn Hồ Trung Hiếu**

**20126041 – Nguyễn Huỳnh Mẫn**

**20126045 – Vũ Hoài Nam**

**20126062 – Thiều Vĩnh Trung**

**Môn học: Hệ quản trị cơ sở dữ liệu**

Thành phố Hồ Chí Minh – 2022

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

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN LÝ THUYẾT**

**BÁO CÁO**

**BÁO CÁO ĐỒ ÁN CUỐI KỲ**

**| Giáo viên hướng dẫn |**

**Cô Hồ Thị Hoàng Vy**

**Cô Phạm Thị Bạch Huệ**

**Môn học: Hệ quản trị cơ sở dữ liệu**

Thành phố Hồ Chí Minh – 2022

MỤC LỤC

[MỤC LỤC 3](#_Toc129460297)

[THÔNG TIN NHÓM 4](#_Toc129460298)

[CHỨC NĂNG HỆ THỐNG VÀ TÌNH HUỐNG TRANH CHẤP 5](#_Toc129460299)

[I. Các chức năng của hệ thống 5](#_Toc129460300)

[Chức năng cho DÙNG CHUNG 5](#_Toc129460301)

[Phân hệ đối tác 5](#_Toc129460302)

[Phân hệ khách hàng 5](#_Toc129460303)

[Phân hệ tài xế 6](#_Toc129460304)

[Phân hệ nhân viên 6](#_Toc129460305)

[Phân hệ quản trị 6](#_Toc129460306)

[II. Xác định tình huống tranh chấp 7](#_Toc129460307)

[THIẾT KẾ CSDL 15](#_Toc129460308)

[1. Phân hệ quản trị 15](#_Toc129460309)

[2. Phân hệ khách hàng 16](#_Toc129460310)

[3. Phân hệ đối tác 18](#_Toc129460311)

[4. Phân hệ tài xế 21](#_Toc129460312)

[LƯỢC ĐỒ QUAN HỆ VÀ SCHEMA 22](#_Toc129460313)

THÔNG TIN NHÓM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Công việc** | **% Hoàn thành** |
| **1** | 20126038 | Nguyễn Hồ Trung Hiếu | Thiết kế database, phân quyền, tìm tình huống tranh chấp | 100% |
| **2** | 20126041 | Nguyễn Huỳnh Mẫn | Thiết kế database, phân quyền, tìm tình huống tranh chấp | 100% |
| **3** | 20126045 | Vũ Hoài Nam | Thiết kế database, thiết kế prototype, tìm tình huống tranh chấp | 100% |
| **4** | 20126062 | Thiều Vĩnh Trung | Thiết kế database, báo cáo, phân quyền, tìm tình huống tranh chấp | 100% |

CHỨC NĂNG HỆ THỐNG VÀ TÌNH HUỐNG TRANH CHẤP

## Các chức năng của hệ thống

### Chức năng cho DÙNG CHUNG

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| ALL1 | Đăng nhập | Đăng nhập vào hệ thống dựa vào tài khoản và mật khẩu. |
| ALL2 | Đăng xuất | Bấm nút đăng xuất khỏi tài khoản |
| ALL3 | Cập nhật mật khẩu | Cập nhật lại mật khẩu mới cho tài khoản |

### Phân hệ đối tác

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| DT1 | Đăng ký tài khoản | Đăng ký thông tin qua website |
| DT2 | Quản lý cửa hàng | Cập nhật thông tin và trạng thái của cửa hàng |
| DT3 | Quản lý đơn hàng | Thay đổi trạng thái của đơn hàng và xác nhận đơn với tài xế |
| DT4 | Quản lý chi nhánh | Cập nhật thông tin cụ thể của từng chi nhánh (địa chỉ,…) |
| DT5 | Quản lý thực đơn | Thêm, xóa, sửa thực đơn |
| DT6 | Xem và ký hợp đồng | Được phép xem hợp đồng và có thể tái ký hợp đồng |

### Phân hệ khách hàng

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| KH1 | Quản lý thông tin cá nhân | Cho phép người dùng cập nhật, chỉnh sửa thông tin cá nhân của mình như họ tên, số điện thoại, địa chỉ,… |
| KH2 | Xem danh sách cửa hàng | Xem danh sách các cửa hàng đang được hỗ trợ và sẵn sàng nhận đơn hàng. Có thể tìm kiếm cửa hàng theo địa điểm, tên cửa hàng,… |
| KH3 | Xem danh sách món | Xem danh sách các món ăn được cung cấp bởi cửa hàng |
| KH4 | Đặt món | Đặt món từ thực đơn của cửa hàng đã chọn. Người dùng chọn các món ăn yêu thích của mình, cung cấp địa chỉ giao hàng, lựa chọn phương thức thanh toán và hoàn tất đơn hàng. |
| KH5 | Xem và hủy đơn hàng | Xem thông tin về các đơn hàng đã đặt, bao gồm các món ăn đã chọn, địa chỉ giao hàng, phương thức thanh toán,… Người dùng cũng có thể hủy đơn hàng khi đơn hàng ở tình trạng chờ xác nhận. |
| KH6 | Đăng ký tài khoản | Đăng ký tài khoản để sử dụng các dịch vụ của hệ thống. Người dùng cung cấp thông tin cá nhân, tên đăng nhập và mật khẩu để đăng ký tài khoản. |
| KH7 | Quản lý các đánh giá về món | Xem, chỉnh sửa hoặc xóa các đánh giá của mình để chia sẻ trải nghiệm của mình với cộng đồng người dùng khác. |

### Phân hệ tài xế

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| TX1 | Quản lý thông tin cá nhân | Quản lý thông tin cá nhân của mình, bao gồm họ tên, CMND, điện thoại, địa chỉ, biển số xe, khu vực hoạt động, email và thông tin tài khoản ngân hàng để nhận tiền. |
| TX2 | Xem danh sách đơn hàng | Xem danh sách đơn hàng hiện có theo khu vực mà họ đã đăng ký và có thể chọn đơn hàng để phục vụ. |
| TX3 | Xem lịch sử giao hàng | Xem lịch sử giao hàng của mình, bao gồm các thông tin về ngày giao hàng, địa chỉ giao hàng và thông tin vận chuyển, phí vận chuyển được nhận ứng với từng đơn hàng. |
| TX4 | Xem và cập nhật khu vực hoạt động | Xem và cập nhật khu vực mà họ có thể hoạt động trong đó bao gồm các quận/huyện, thành phố |
| TX5 | Cập nhật quá trình đơn hàng (đã nhận, đang giao, đã giao) | Cập nhật trạng thái của đơn hàng mà họ đã nhận, từ khi đơn hàng được xử lý đến khi đơn hàng được giao thành công. Các trạng thái thường gặp là "đã nhận", "đang giao" và "đã giao". |

### Phân hệ nhân viên

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| NV1 | Xem hợp đồng | Xem thông tin về các hợp đồng mà đối tác đã ký kết với công ty, bao gồm ngày bắt đầu, ngày kết thúc, giá trị hợp đồng và các điều khoản và điều kiện khác. |
| NV2 | Duyệt hợp đồng | Duyệt hợp đồng. Nếu duyệt, nhân viên sẽ thông báo thời gian hiệu lực của hợp đồng đến đối tác. |
| NV3 | Gửi thông báo gia hạn hợp đồng | Khi hợp đồng của đối tác sắp hết hạn, nhân viên có thể gửi thông báo yêu cầu gia hạn cho đối tác. |

### Phân hệ quản trị

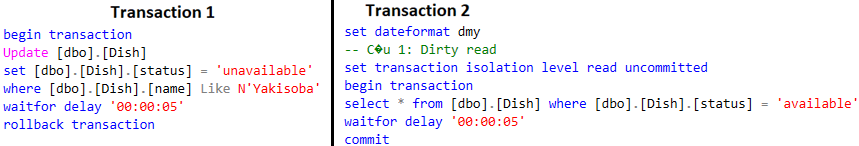
|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| QT1 | Quản lí người dùng | - Cập nhật thông tin tài khoản  - Thêm/xóa/sửa tài khoản admin và nhân viên  - Khóa và kích hoạt tài khoản |
| QT2 | Cập nhật quyền người dùng | - Cấp quyền thao tác trên dữ liệu  - Cấp quyền thao tác trên giao diện |

## Xác định tình huống tranh chấp

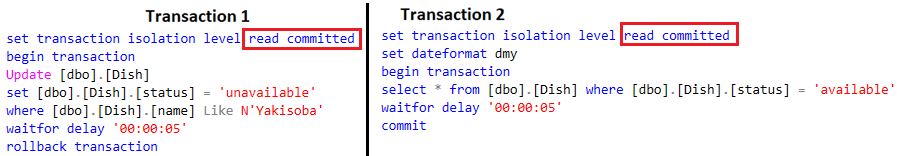
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Chức năng 1** | **Người dùng** | **Chức năng 2** | **Người dùng** | **Lỗi tranh chấp** |
| **1** | Đặt món | Khách hàng 1 | Đặt món | Khách hàng 1 | **Dirty Read**: Khi khách hàng A đặt 1 món X thì số lượng món X giảm xuống 1, thì cùng lúc đó khách hàng B đang đọc với số lượng món hàng X-1. Nhưng sau đó, giao dịch của đơn hàng khách A bị lỗi → rollback. Làm cho khách B đọc sai dữ liệu. |
| **2** | Nhận đơn | Tài xế 1 | Nhận đơn | Tài xế 2 | **Dirty Read:** Khi một tài xế A bấm nhận đơn hàng X, thì trong danh sách đơn hàng - đơn hàng X đã nhận. Tài xế B khi xem danh sách thì không thấy đơn hàng X, nhưng trong quá trình tài xế A chọn bị lỗi hệ thống và bị rollback → Tài xế B không xem được đơn X. |
| **3** | Đặt món | Khách hàng | Cập nhật món | Đối tác | **Dirty Read:** Đối tác cập nhật số lượng món X (VD: từ 10 lên 15), thì lúc này khách hạng sẽ xem được món X là 15. Tuy nhiên, trong quá trình cập nhật của đối tác bị lỗi → rollback → khách hàng đọc sai dữ liệu món. |
| **4** | Xác nhận hợp đồng | Nhân viên 1 | Xem hợp đồng | Nhân viên 2 | **Dirty Read:** Khi một nhân viên A bấm xác nhận hợp đồng X, thì trong hợp đồng – hợp đồng X đã xác nhận. Nhân viên B khi xem danh sách thì thấy hợp đồng X đã xác nhận, nhưng trong quá trình nhân viên A xác nhận bị lỗi hệ thống và bị rollback → Nhân viên B không xác nhận được hợp đồng X. |
| **5** | Cập nhật lại đơn hàng | Tài xế | Thống kê thu nhập | Đối tác | **Unrepeatable:** Khi đối tác xem tổng thu nhập của mình trên tất cả chi nhánh (mang tính realtime, kể cả những đơn hàng chưa được xác nhận). Sau đó có một đơn hàng được cập nhật quá trình đã giao. Tiếp theo đối tác muốn vào một chi nhánh để xem tổng thu nhập của một chi nhánh cụ thể thì thấy tổng thu nhập của chi nhánh đó đã được thay đổi so với lần kiểm tra trên tất cả chi nhánh của đối tác. |
| **6** | Thống kê đánh giá | Đối tác | Đánh giá món | Khách hàng | **Unrepeatable:** Trong transaction A, khách hàng tạo một đơn hàng với những tùy chọn X,Y,Z. Đối tác thấy đơn hàng mới, thực hiện xác nhận đơn hàng. Trong lúc đơn hàng chưa xác nhận thì khách hàng bỏ bớt món trong đơn hàng của mình nên sau đó đối tác đã xác nhận đơn hàng với số lượng món và giá tiền khác với ban đầu. |
| **7** | Đặt món | Khách hàng | Xóa tùy chọn món | Đối tác | **Unrepeatable:** Trong 1 transaction tạo đơn hàng với tùy chọn món là A, tên món là B, cùng lúc đó 1 transaction khác cập nhật giá tùy chọn món A, tên món B. Khi tạo đơn hàng với món A và B → lỗi unrepeated vì giá trước khi transaction B thực hiện và giá ban đầu khác nhau. |
| **8** | Cập nhật đơn hàng | Đối tác | Cập nhật đơn hàng | Tài xế | **Unrepeatable:** Tài xế A chọn đơn hàng X trong khu vực hoạt động của mình 🡪 tài xế update nhận đơn hàng để giao. Cùng lúc đó đối tác chuyển đơn hàng sang một chi nhánh khác khu vực hoạt động của tài xế. Tài xế update không được giá trị ID của mình nên sẽ bị lỗi. |
| **9** | Thống kê số lượng đơn hàng | Đối tác | Đặt hàng | Khách hàng | **Phantom:** Trong 1 transaction tính thu nhập của tháng và các ngày. Trong lúc đó khách hàng thêm 1 đơn hàng mới vào tháng hiện tại → Thu nhập của tháng không bằng tổng thu nhập các ngày trong tháng. |
| **10** | Theo dõi thu nhập | Tài xế | Xử lý đơn hàng | Tài xế | **Phantom:** Trong 1 transaction lấy lịch sử đơn hàng và tính tổng thu nhập tháng này của tài xế, có 1 đơn hàng mới vừa được hoàn thành → Lịch sử đơn hàng không có đơn hàng đó, nhưng tổng thu nhập thì lại có phí của đơn hàng đó. |
| **11** | Quản lý số liệu | Đối tác | Xử lý đơn hàng | Đối tác | **Phantom:** Trong 1 transaction tính tổng thu nhập tháng này và tổng thu nhập ngày hôm nay, có 1 đơn hàng được xử lý trong ngày hôm nay → thu nhập tháng không tính đơn hàng đó nhưng thu nhập ngày thì lại có. |
| **12** | Đặt món | Khách hàng | Xóa tùy chọn món | Đối tác | **Phantom:** Trong 1 transaction tạo đơn hàng với tùy chọn món là A, tên món là B, cùng lúc đó 1 transaction khác xóa mất tùy chọn món A, tên món B. Khi tạo đơn hàng với tùy chọn món A, tên món B → Lỗi phantom vì dòng dữ liệu đó đã bị mất. |
| **13** | Xác nhận đơn hàng | Tài xế 1 | Xác nhận đơn hàng | Tài xế 2 | **Lost update:** Một tài xế chọn nhận đơn hàng, nhưng cùng lúc đó một tài xế khác cũng chọn đơn hàng này và lưu trữ vào cơ sở dữ liệu. Khi xem lại thông tin đơn hàng, chỉ một trong hai cập nhật tình trạng mới nhất được lưu trữ trong cơ sở dữ liệu, gây ra sự cố trong quá trình xử lý đơn hàng. |
| **14** | Hủy đơn hàng | Khách hàng | Xác nhận đơn hàng | Đối tác | **Lost update:** Khi khách hàng đặt món và gửi yêu cầu đặt hàng cho đối tác, đối tác tiếp nhận yêu cầu và thực hiện xác nhận đơn hàng. Trong khi đang chờ xác nhận từ đối tác, khách hàng quyết định hủy đơn hàng và gửi yêu cầu hủy đơn hàng cho đối tác, cùng lúc đó đối tác bấm xác nhận đơn → Gây ra sự cố xử lý dữ liệu |
| **15** | Cập nhật hợp đồng | Nhân viên 1 | Cập nhật hợp đồng | Nhân viên 2 | **Lost update:** Hai nhân viên đang thao tác trên cùng một hợp đồng của đối tác. Nhân viên A thực hiện chỉnh sửa thông tin hợp đồng, sau đó nhân viên B cũng thực hiện chỉnh sửa thông tin trên cùng hợp đồng  → Gây ra sự cố xử lý dữ liệu |
| **16** | Đặt món | Khách hàng | Đặt món | Khách hàng | **Lost update:** Hai khách hàng đồng thời thực hiện đặt món X và đặt hàng trên hệ thống quản trị cơ sở dữ liệu. Tuy nhiên, số lượng sản phẩm X chỉ còn 1 trong kho, vì vậy chỉ có thể bán được cho một khách hàng → Gây ra sự cố xử lý dữ liệu |

1. **Giải quyết tình huống tranh chấp**
2. **Dirty Read**
3. *Tình huống 1*

Khi khách hàng A đặt 1 món X thì số lượng món X giảm xuống 1, thì cùng lúc đó khách hàng B đang đọc với số lượng món hàng X-1. Nhưng sau đó, giao dịch của đơn hàng khách A bị lỗi → rollback. Làm cho khách B đọc sai dữ liệu.

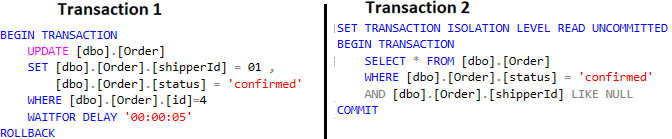


**→** Để giải quyết vấn đề này, ta chỉ cần bỏ READ UNCOMMITTED và sử dụng mức độ cô lập mặc định của hệ thống (READ COMMITTED)

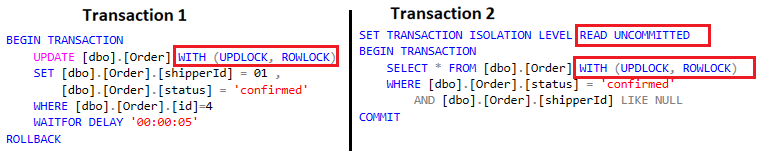


1. *Tình huống 2*

Khi một tài xế A bấm nhận đơn hàng X, thì trong danh sách đơn hàng - đơn hàng X đã nhận. Tài xế B khi xem danh sách thì không thấy đơn hàng X, nhưng trong quá trình tài xế A chọn bị lỗi hệ thống và bị rollback → Tài xế B không xem được đơn X.

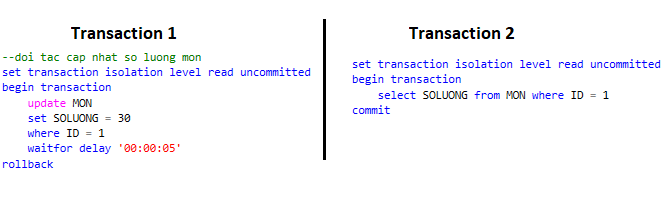


→ Để giải quyết tình huống tranh chấp này, ta có thể sử dụng cơ chế locking để đảm bảo rằng đơn hàng X chỉ được tài xế A đang xử lý truy cập vào. Đồng thời Sử dụng UPDLOCK và ROWLOCK trong Transaction 2 cũng đảm bảo rằng chỉ có một tài xế được phép truy cập vào đơn hàng X cùng một lúc.

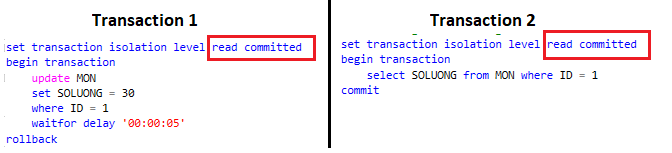


1. *Tình huống 3*

Đối tác cập nhật số lượng món X (VD: từ 10 lên 15), thì lúc này khách hạng sẽ xem được món X là 15. Tuy nhiên, trong quá trình cập nhật của đối tác bị lỗi → rollback → khách hàng đọc sai dữ liệu món.

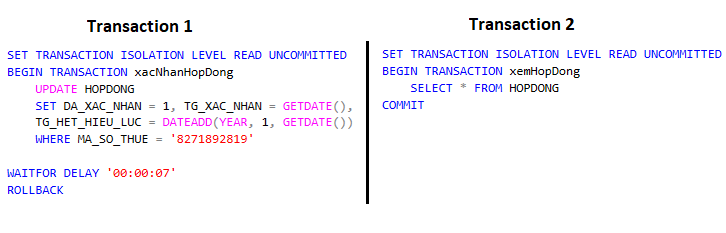


→ Ta sử dụng READ COMMITTED để giải quyết tình huống Dirty Read hoặc có thể không cần phải set lại, vì mặc định của hệ thống đã là READ COMMITTED

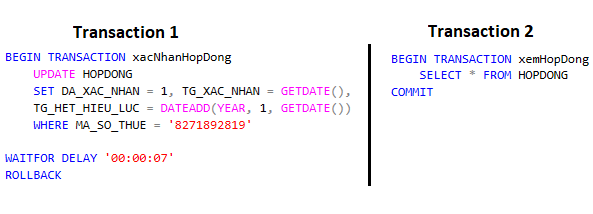


1. *Tình huống 4*

Khi một nhân viên A bấm xác nhận hợp đồng X, thì trong hợp đồng – hợp đồng X đã xác nhận. Nhân viên B khi xem danh sách thì thấy hợp đồng X đã xác nhận, nhưng trong quá trình nhân viên A xác nhận bị lỗi hệ thống và bị rollback → Nhân viên B không xác nhận được hợp đồng X.

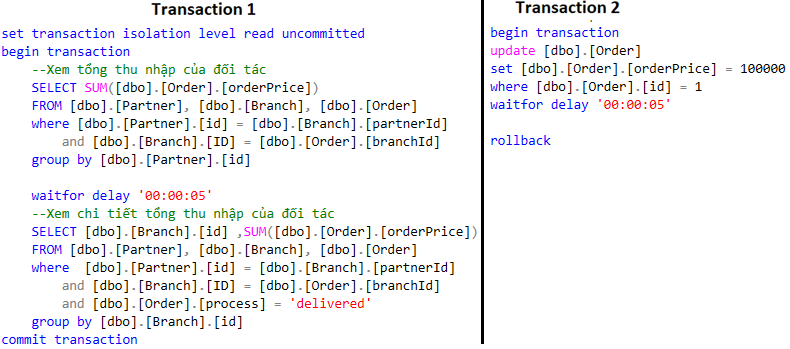
****

**→** Để giải quyết vấn đề này, ta chỉ cần bỏ READ UNCOMMITTED và sử dụng mức độ cô lập mặc định của hệ thống (READ COMMITTED)



1. **Unrepeatable**
2. *Tình huống 1*

Khi đối tác xem tổng thu nhập của mình trên tất cả chi nhánh (mang tính realtime, kể cả những đơn hàng chưa được xác nhận). Sau đó có một đơn hàng được cập nhật đơn giá (tăng hoặc giảm). Tiếp theo đối tác muốn vào một chi nhánh để xem tổng thu nhập của một chi nhánh cụ thể thì thấy tổng thu nhập của chi nhánh đó đã được thay đổi so với lần kiểm tra trên tất cả chi nhánh của đối tác.

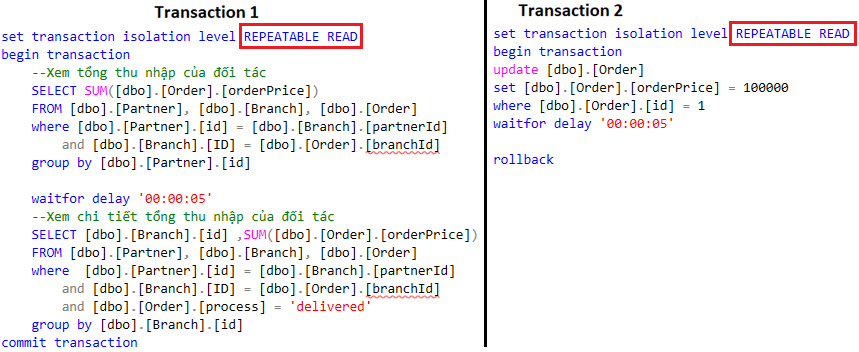


→ Sử dụng REPEATABLE READ →Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác => Các giao tác khác phải chờ đến khi giao

tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này .

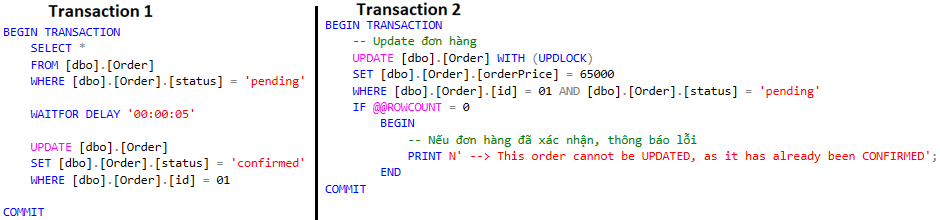
(Repeatable Read = Read Committed + Giải quyết Unrepeatable Reads)

→ Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến hết giao tác.



1. *Tình huống 2*

Trong transaction A, khách hàng tạo một đơn hàng với những tùy chọn X,Y,Z. Đối tác thấy đơn hàng mới, thực hiện xác nhận đơn hàng. Trong lúc đơn hàng chưa xác nhận thì khách hàng bỏ bớt món trong đơn hàng của mình nên sau đó đối tác đã xác nhận đơn hàng với số lượng món và giá tiền khác với ban đầu.



→ Sử dụng cơ chế khóa để tránh tranh chấp giữa các 2 transaction.

→ Trong transaction 1, chúng ta sử dụng khóa UPDLOCK để khóa bảng DONHANG khi chúng ta đọc dữ liệu. Điều này sẽ ngăn chặn các transaction khác cập nhật hoặc đọc dữ liệu trong DONHANG khi transaction này đang được thực hiện. Đồng thời, chúng ta sử dụng ROWLOCK để đảm bảo rằng chỉ có một hàng trong DONHANG được khóa tại một thời điểm. Việc này sẽ giúp tránh các lỗi liên quan đến Unrepeatable Read.

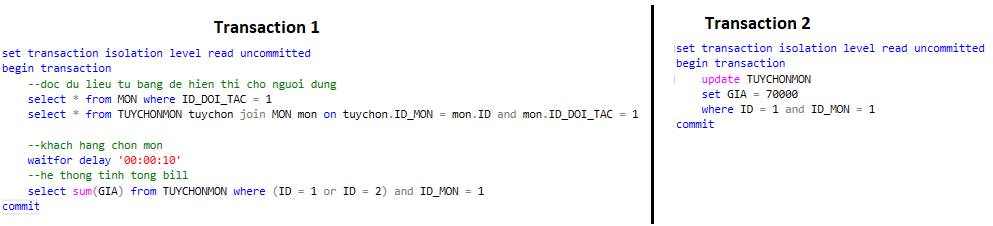
→ Trong transaction 2, chúng ta cũng sử dụng khóa UPDLOCK để khóa hàng được cập nhật. Điều này sẽ ngăn chặn các transaction khác cập nhật hàng này khi transaction này đang được thực hiện.

Text

Description automatically generated with medium confidence

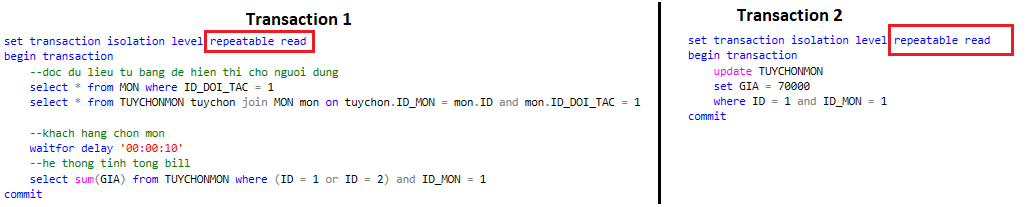
1. *Tình huống 3*

Trong 1 transaction tạo đơn hàng với tùy chọn món là A, tên món là B, cùng lúc đó 1 transaction khác cập nhật giá tùy chọn món A, tên món B. Khi tạo đơn hàng với món A và B → lỗi unrepeated vì giá trước khi transaction B thực hiện và giá ban đầu khác nhau.



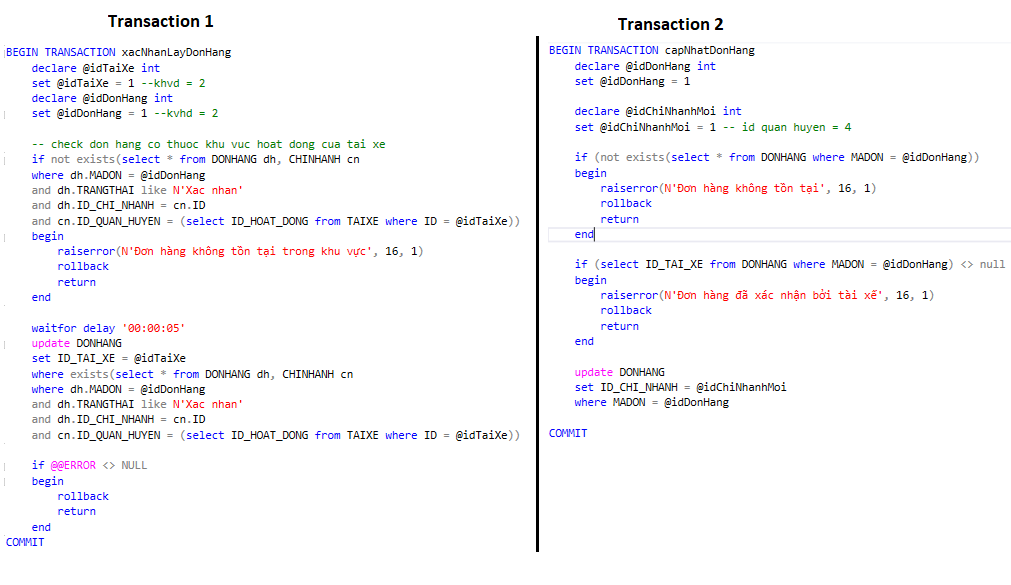
→ Sử dụng REPEATABLE READ →Tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác => Các giao tác khác phải chờ đến khi giao

tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này .



1. *Tình huống 4*

Tài xế A chọn đơn hàng X trong khu vực hoạt động của mình 🡪 tài xế update nhận đơn hàng để giao. Cùng lúc đó đối tác chuyển đơn hàng sang một chi nhánh khác khu vực hoạt động của tài xế. Tài xế update không được giá trị ID của mình nên sẽ bị lỗi.

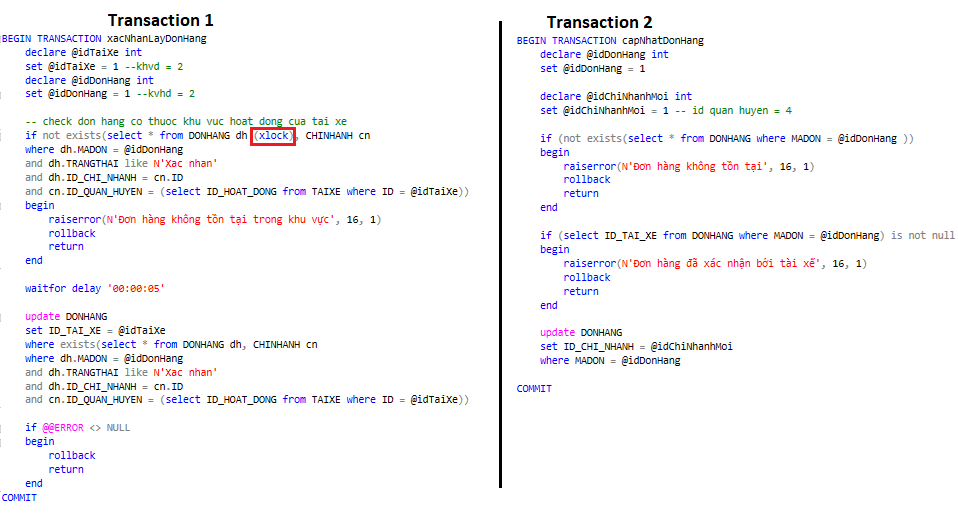


→ Xin khóa XLOCK trên đơn vị dữ liệu để đọc

- Những thao tác khác khi cập nhật trên cùng đơn vị dữ liệu này sẽ phải đợi

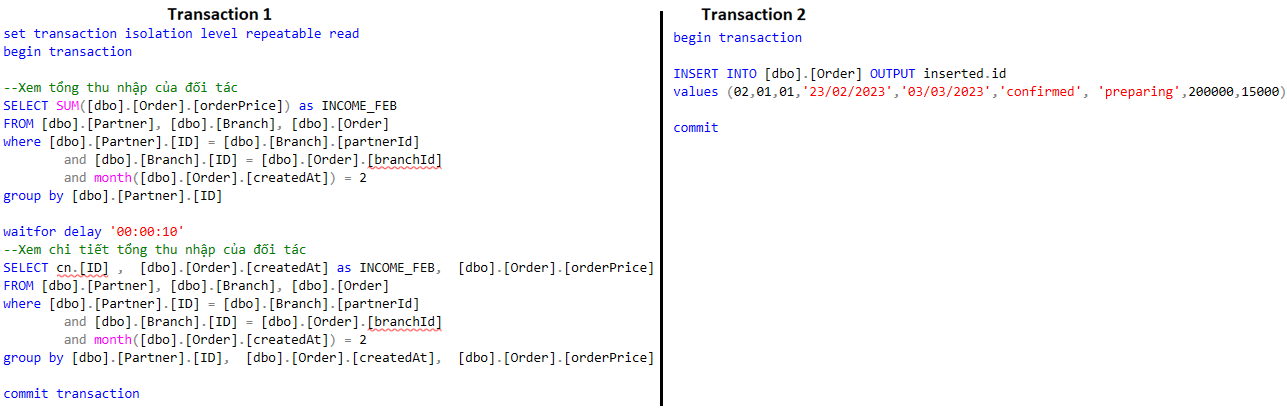
- Khi select lại lần 2 dữ liệu ko thay đổi, đảm bảo tính consistency của giao tác

- Chỉ nhả khóa khi hết giao tác, lúc này các giao tác khác trong hàng đợi có thể tiến hành thực thi.



1. **Phantom**
2. *Tình huống 1*

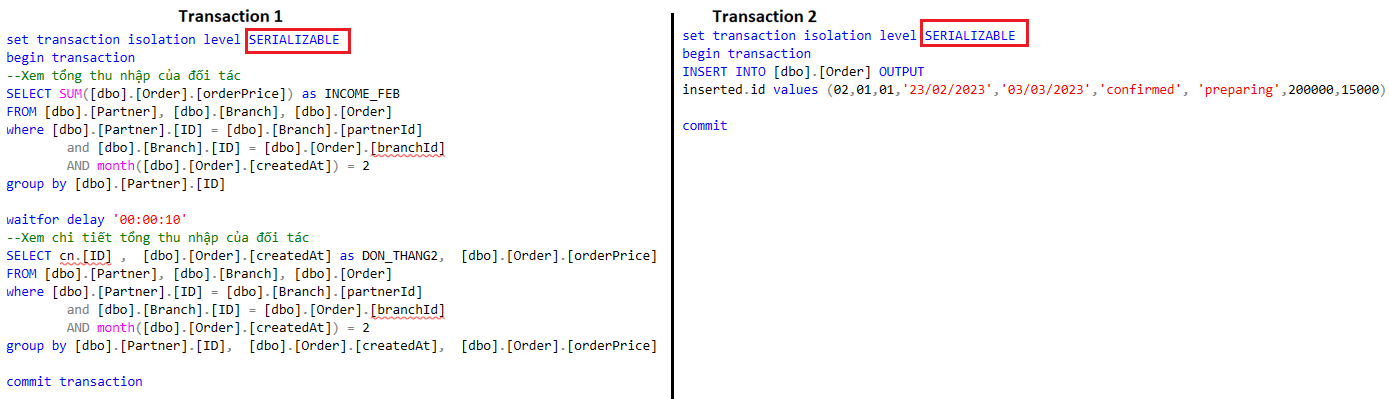
Trong 1 transaction tính thu nhập của tháng và các ngày. Trong lúc đó khách hàng thêm 1 đơn hàng mới vào tháng hiện tại → Thu nhập của tháng không bằng tổng thu nhập các ngày trong tháng.



→ Sử dụng SERIALIZABLE để 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 → Giải quyết được vấn đề Phantom.

→ Tạo Exclusive Lock trên đơn vị dữ liệu được ghi, Exclusive Lock được giữ cho đến

hết giao tác.



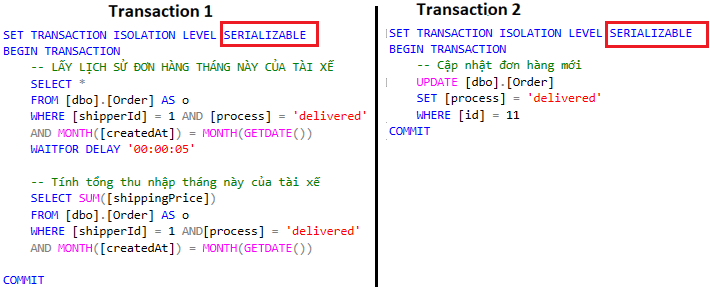
1. *Tình huống 2*

Trong 1 transaction lấy lịch sử đơn hàng và tính tổng thu nhập tháng này của tài xế, có 1 đơn hàng mới vừa được hoàn thành → Lịch sử đơn hàng không có đơn hàng đó, nhưng tổng thu nhập thì lại có phí của đơn hàng đó.

Text

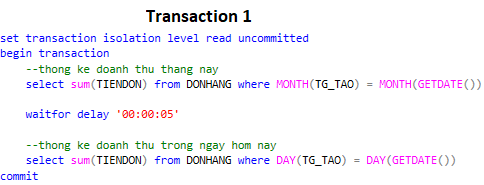
Description automatically generated

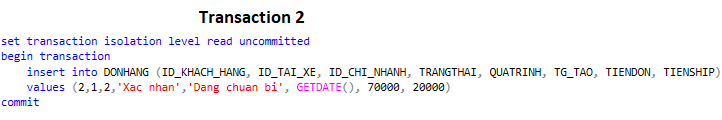
→ Sử dụng SERIALIZABLE để tạo Shared Lock trên đơn vị dữ liệu được đọc và giữ shared lock này đến hết giao tác => Các giao tác khác phải chờ đến khi giao tác này kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này → Giải quyết được vấn đề Phantom.



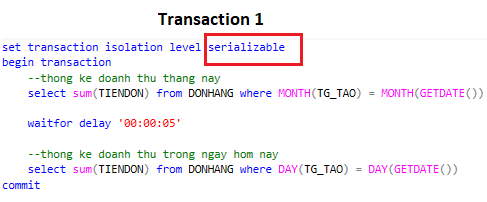
1. *Tình huống 3*

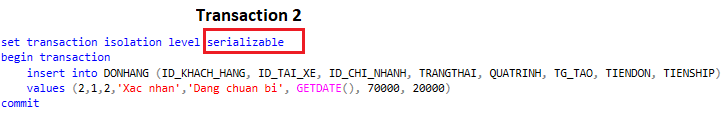
Trong 1 transaction tính tổng thu nhập tháng này và tổng thu nhập ngày hôm nay, có 1 đơn hàng được xử lý trong ngày hôm nay → thu nhập tháng không tính đơn hàng đó nhưng thu nhập ngày thì lại có.





**→ Hướng giải quyết**: Sử dụng ISOLATION LEVEL SERIALIZABLE ở cả 2 transaction





1. *Tình huống 4*

Trong 1 transaction tạo đơn hàng với tùy chọn món là A, tên món là B, cùng lúc đó 1 transaction khác xóa mất tùy chọn món A, tên món B. Khi tạo đơn hàng với tùy chọn món A, tên món B → Lỗi phantom vì dòng dữ liệu đó đã bị mất.

Text

Description automatically generated

→ **Hướng giải quyết:** Sử dụng ISOLATION LEVEL SERIALIZABLE để cho các thao tác chạy tuần tự. Giao tác nào vào trước sẽ chạy trước, những thao tác khác phải đợi

=> Giải quyết được Phantom

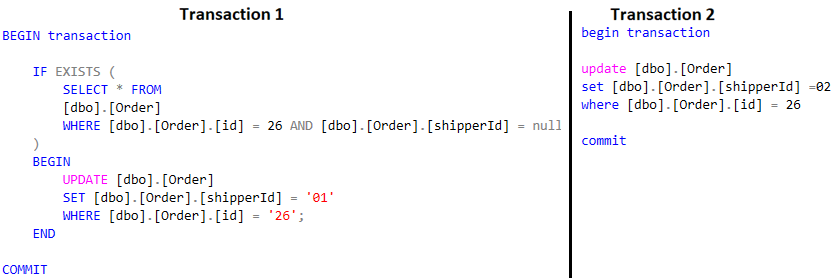
→ Không sử dụng khóa vì thao tác như Delete hay Insert vẫn có thể chen vào được.

Graphical user interface, text, email

Description automatically generated

1. **Lost update**
2. *Tình huống 1*

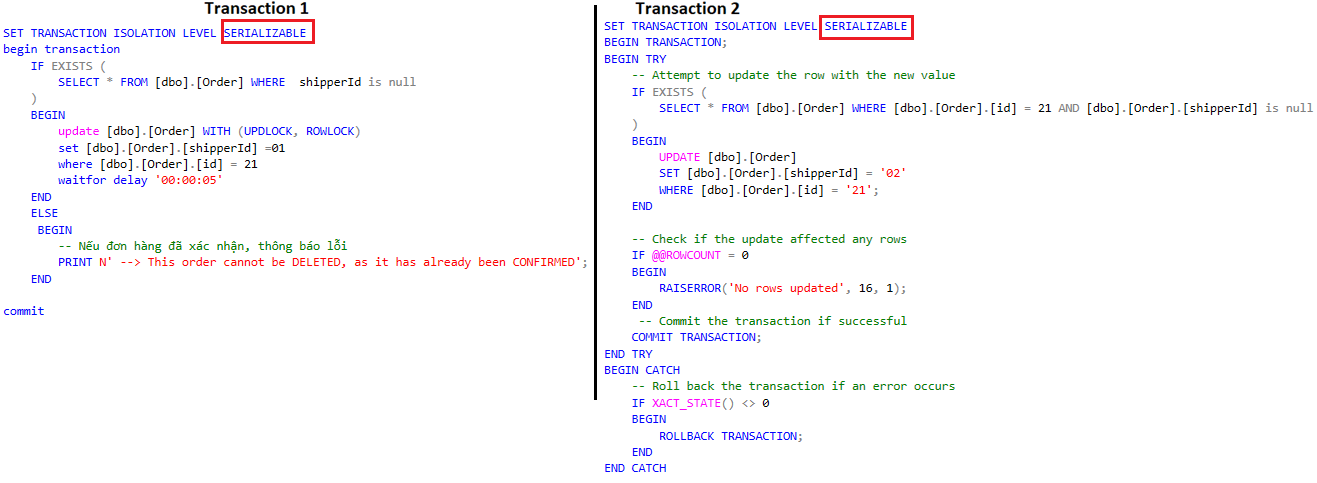
Một tài xế chọn nhận đơn hàng, nhưng cùng lúc đó một tài xế khác cũng chọn đơn hàng này và lưu trữ vào cơ sở dữ liệu. Khi xem lại thông tin đơn hàng, chỉ một trong hai cập nhật tình trạng mới nhất được lưu trữ trong cơ sở dữ liệu, gây ra sự cố trong quá trình xử lý đơn hàng.



→ **Hướng giải quyết:** Ta có thể xin khóa uplock trên những dataset cần truy cập để. Ở đây chỉ xin uplock trên một hàng mà câu truy vấn quan tâm đến mà không phải lock toàn bảng → Để tránh việc các giao tác khác cần truy cập đến dataset khác trong bảng mà không xuất hiện Lost Update

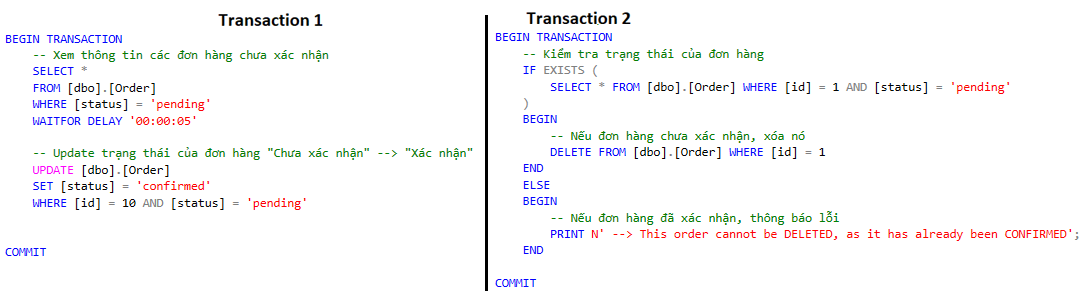
- Cần thêm một vài dòng code ở tran 2 để khi không truy cập được vào dòng cần update dữ liệu (không được cấp khóa), thì raise error và rollback

- Thêm câu truy vấn: SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

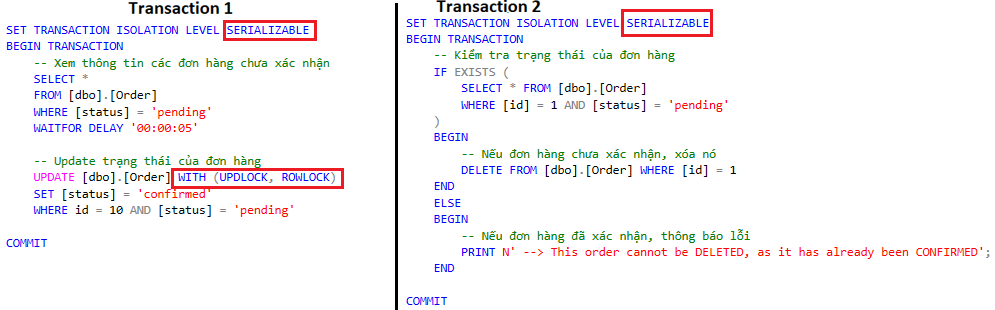


1. *Tình huống 2*

Khi khách hàng đặt món và gửi yêu cầu đặt hàng cho đối tác, đối tác tiếp nhận yêu cầu và thực hiện xác nhận đơn hàng. Trong khi đang chờ xác nhận từ đối tác, khách hàng quyết định hủy đơn hàng và gửi yêu cầu hủy đơn hàng cho đối tác, cùng lúc đó đối tác bấm xác nhận đơn → Gây ra sự cố xử lý dữ liệu

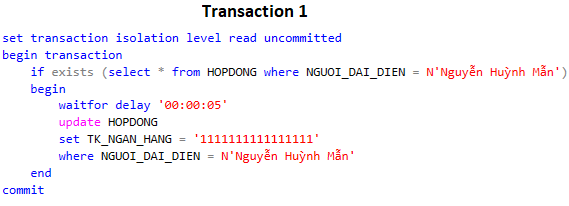


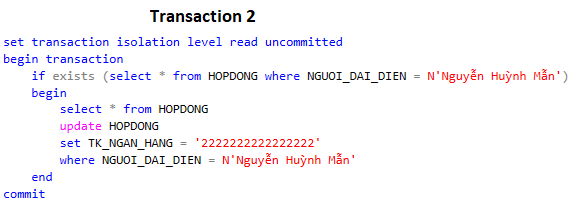
→ **Hướng giải quyết:** Dùng SERIALIZABLE cho cả 2 transaction và sử dụng thêm WITH(UPDLOCK, ROWLOCK) để đảm bảo rằng chỉ có 1 transaction được cập nhật đơn hàng đó.



1. *Tình huống 3*

Hai nhân viên đang thao tác trên cùng một hợp đồng của đối tác. Nhân viên A thực hiện chỉnh sửa thông tin hợp đồng, sau đó nhân viên B cũng thực hiện chỉnh sửa thông tin trên cùng hợp đồng → Gây ra sự cố xử lý dữ liệu



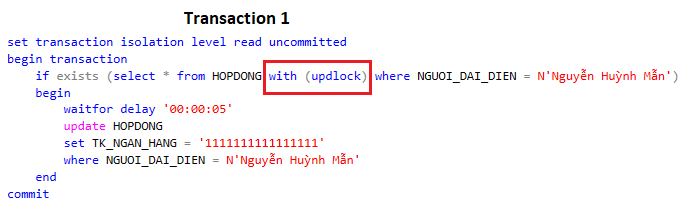


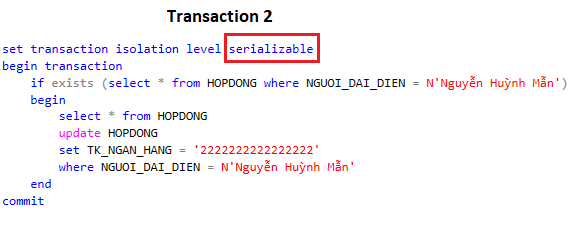
→ **Hướng giải quyết:**

**-** Ở transaction 1: Sử dụng khóa UPDLOCK khi đọc ghi trên cùng đơn vị dữ liệu

=> Những thao tác khác khi đọc ghi trên đơn vị dữ liệu này sẽ phải đợi. Giao tác đang giữ khóa UPDLOCK sau đó sẽ nâng cấp lên XLOCK và tiến hành update.

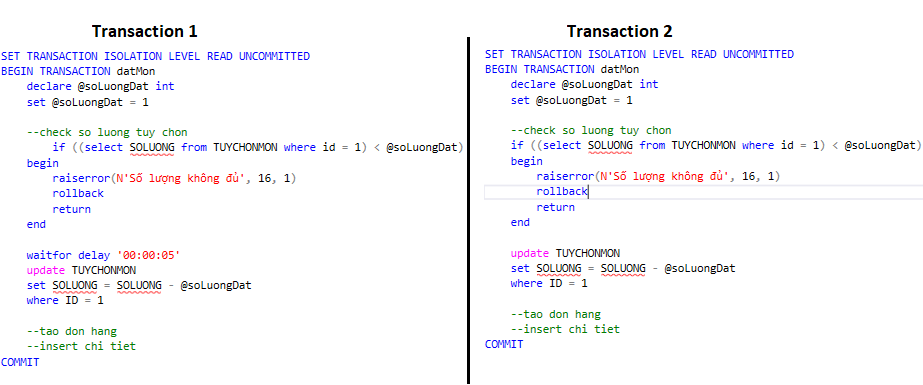
- Ở transaction 2: Sử dụng SERIALIZABLE để đảm bảo rằng chỉ có 1 transaction được cập nhật đơn hàng đó.





1. *Tình huống 4*

Hai khách hàng đồng thời thực hiện đặt món X và đặt hàng trên hệ thống quản trị cơ sở dữ liệu. Tuy nhiên, số lượng sản phẩm X chỉ còn 1 trong kho, vì vậy chỉ có thể bán được cho một khách hàng → Gây ra sự cố xử lý dữ liệu

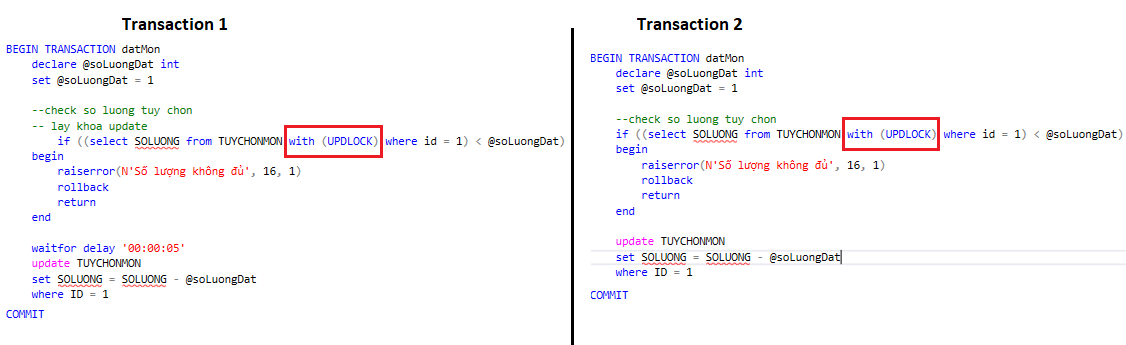


→ **Hướng giải quyết:** Sử dụng khóa UPDLOCK khi đọc ghi trên cùng đơn vị dữ liệu

🡺 Những thao tác khác khi đọc ghi trên đơn vị dữ liệu này sẽ phải đợi.

- Giao tác đang giữ khóa UPDLOCK sau đó sẽ nâng cấp lên XLOCK và tiến hành update

- Cuối cùng nhả khóa khi commit giao tác => Giao tác khác có thể xin khóa UPDLOCK và tiến hành → update như thường → Không còn Lost Update.



THIẾT KẾ CSDL

## Phân hệ quản trị

Text

Description automatically generated

- Khi truy cập vào website, giao diện đăng nhập vào hệ thống sẽ hiển thị đầu tiên

Text

Description automatically generated with medium confidence

- Giao diện quản lý tài khoản của nhân viên:

Table

Description automatically generated

- Giao diện quản lý các đối tác:

Graphical user interface, table

Description automatically generated

## Phân hệ khách hàng

- Khi khách hàng đăng nhập vào hệ thống, sẽ được chọn chi nhánh cửa hàng để đặt món

Diagram

Description automatically generated

- Sau khi chọn chi nhánh, phần giao diện thực đơn sẽ hiển thị ra tương ứng với chi nhánh đã chọn. Tại đây, khách hàng có thể xem qua danh sách món, chi tiết các món, đánh giá,… và tiến hành đặt món yêu thích:

Table

Description automatically generatedA picture containing diagram

Description automatically generated

- Giao diện chi tiết món và đánh giá món ăn. Ở đây, khách hàng có thể sẽ được tên món, mức giá, mô tả chi tiết và các lượt đánh giá từ những khách hàng khác

- Các món đã chọn sẽ hiển thị ra giao diện cùng với kích cỡ, số lượng, tổng tiền

- Khách hàng có thể thực hiện thanh toán bằng tiền mặt hoặc banking khi đặt hàng

Graphical user interface

Description automatically generated

- Khách hàng được xem lại lịch sử các đơn hàng đã đặt trước đó. Đồng thời xem đơn hàng hiện tại, tình trạng của đơn hàng:

Graphical user interface, table

Description automatically generated

## Phân hệ đối tác

- Sau khi đăng nhập, giao diện chính sẽ hiển thị các thực đơn của cửa hàng đối tác quản lý. Tại đây, đối tác có thể thêm thực đơn mới hoặc thêm các món mới vào thực đơn

Graphical user interface

Description automatically generated with medium confidence

- Giao diện chi tiết món và thông tin chi nhánh :

A picture containing graphical user interface

Description automatically generated A picture containing text

Description automatically generated

- Ở chi tiết đơn hàng, đối tác được chỉnh sửa, cập nhật lại tên món, mô tả, giá tiền,… Theo dõi được các đánh giá từ khách hàng

Graphical user interface, table

Description automatically generated with medium confidence

- Giao diện Order sẽ hiển thị các lịch sử giao dịch trong ngày (các chi tiết về đơn hàng, trạng thái,…)

Graphical user interface, website

Description automatically generated

Graphical user interface, application, table

Description automatically generated

- Thống kê đơn hàng sẽ được hiển thị tại giao diện Analysis

Graphical user interface, application, table

Description automatically generated

## Phân hệ tài xế

- Tài xế nhận các đơn giao hàng thông qua giao diện Order. Danh sách các đơn hàng và chi tiết đơn được thể hiển rõ ở giao diện này sau đó tài xế được chọn các đơn hàng phù hợp

Graphical user interface, email

Description automatically generated

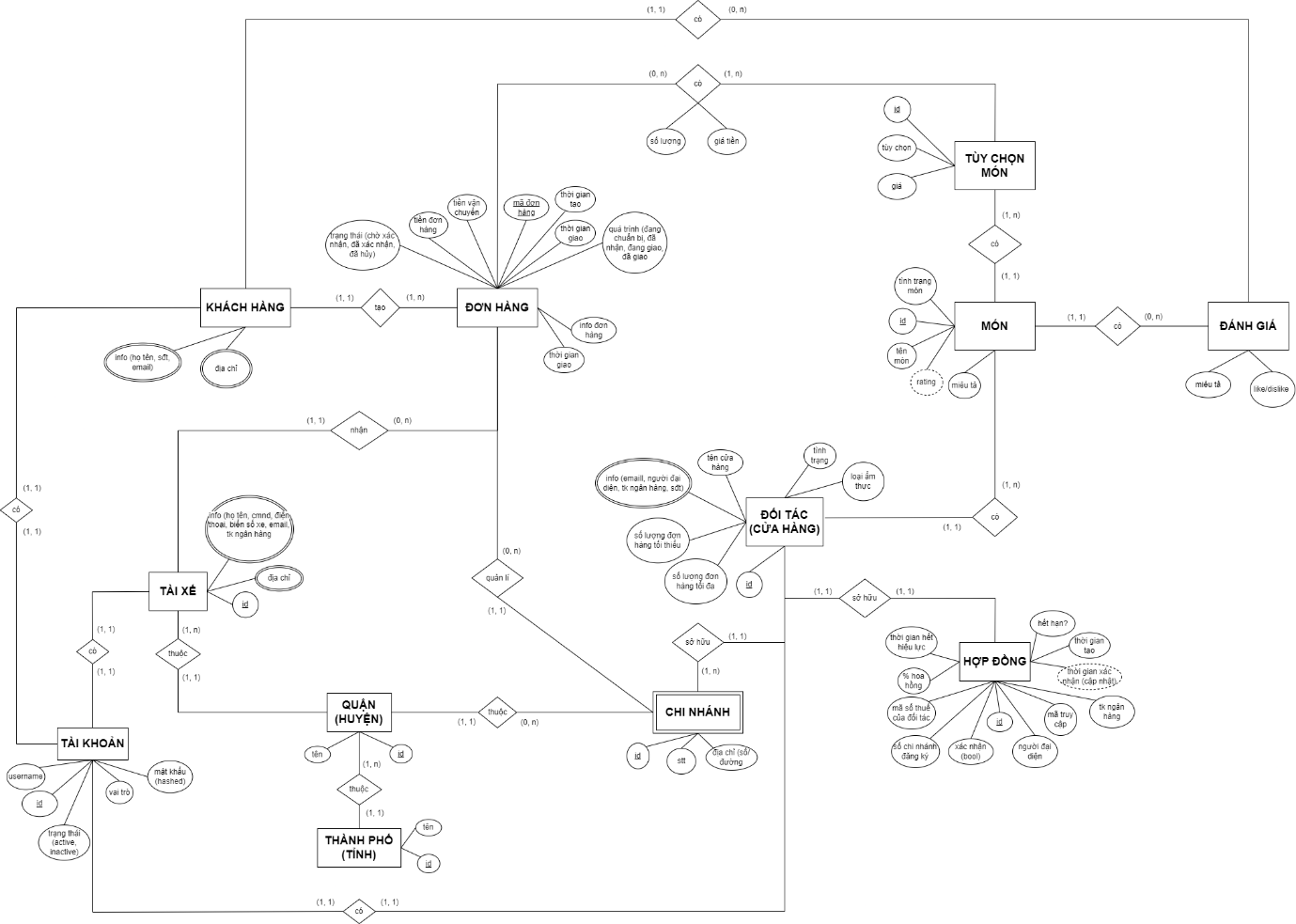
- Lịch sử giao hàng trong ngày và đơn hàng đang nhận sẽ hiển thị tại đây:

A screenshot of a computer

Description automatically generated

LƯỢC ĐỒ QUAN HỆ VÀ SCHEMA

1. **Lược đồ quan hệ**



1. **Schema**

Diagram, engineering drawing

Description automatically generated