Trường hợp race condition chung trong hệ thống ngân hàng khi không có các cơ chế đồng bộ (synchronization) là: **Thực hiện các giao dịch đồng thời,** cụ thể: khi một khách hàng giao dịch giữa hai tài khoản, ngân hàng cần đảm bảo rằng số tiền được trừ khỏi tài khoản nguồn và được ghi có vào tài khoản đích. Nếu hai giao dịch chuyển tiền xảy ra cùng lúc, có thể xảy ra race condition, dẫn đến việc một tài khoản bị ghi nợ hai lần hoặc một tài khoản không nhận được tiền**.**

Các trường hợp cụ thể:

Trường hợp race condition	**Mô tả**	**Tác hại**
1. Cập nhật thông tin tài khoản	Khi có nhiều người cùng lúc thực hiện các tác vụ cập nhật thông tin trên cùng một tài khoản.	Thông tin bị cập nhật không chính xác, dữ liệu bị ghi đè lên nhau → truy xuất sai dữ liệu.
2. Chuyển khoản	Khi có hai giao dịch chuyển khoản cùng một lúc	Cập nhật số dư sai -> ảnh hưởng trực tiếp đến tài sản và trải nghiệm người dùng. Hai giao dịch cập nhật số dư tài khoản cùng một lúc có thể dẫn đến việc số dư tài khoản không chính xác.
3. Rút tiền	Khi có nhiều người cùng lúc thực hiện thao tác rút tiền trên cùng một tài khoản ngân hàng. Ví dụ: Giả sử có hai giao dịch A và B cùng lúc cập nhật số dư tài khoản của một khách hàng. Giao dịch A rút 100.000 đồng từ tài khoản, trong khi giao dịch B nạp 200.000 đồng vào tài khoản. Nếu giao dịch A được thực hiện trước giao dịch B, thì số dư tài khoản sẽ bị cập nhật sai thành -100.000 đồng	Tương tự như trên

4. Xử lý giao dịch thẻ	Khi một khách hàng sử dụng thẻ ATM để rút tiền từ tài khoản của họ. Khi khách hàng nhập mã PIN và số tiền rút, máy ATM gửi yêu cầu đến hệ thống ngân hàng để xác minh giao dịch. Nếu có hai yêu cầu rút tiền cùng một lúc từ cùng một thẻ, có thể xảy ra trường hợp một yêu cầu được chấp thuận mặc dù khách hàng không thực hiện giao dịch.	Tương tự như trên
5. Sử dụng hệ thống thanh toán	Giả sử hai cửa hàng bán lẻ sử dụng cùng một hệ thống thanh toán để xử lý các giao dịch thẻ. Nếu có hai giao dịch thẻ cùng một lúc từ cùng một khách hàng, thì có thể xảy ra trường hợp một giao dịch bị tính phí hai lần.	Tương tự như trên

Cách phòng ngừa race condition trong ngân hàng:

Để phòng ngừa race condition trong ngân hàng, cần áp dụng các biện pháp sau:

- **Sử dụng khóa:** Khóa là một cơ chế đồng bộ hóa ngăn hai hoặc nhiều luồng truy cập cùng một tài nguyên cùng một lúc. Việc sử dụng khóa có thể giúp đảm bảo rằng chỉ có một giao dịch được thực hiện tại một thời điểm.
- **Sử dụng phiên bản:** Phiên bản là một kỹ thuật cho phép theo dõi và quản lý các thay đổi đối với dữ liệu. Việc sử dụng phiên bản có thể giúp đảm bảo rằng chỉ có phiên bản mới nhất của dữ liệu được sử dụng.
- **Thiết kế hệ thống cẩn thận:** Hệ thống ngân hàng cần được thiết kế cẩn thận để tránh các tình huống có thể dẫn đến race condition. Ví dụ, cần tránh sử dụng các biến chung cho nhiều luồng và cần đảm bảo rằng các giao dịch được thực hiện theo thứ tự chính xác.

Các phương pháp chuyên môn để giải quyết race condition:

1. Các phương pháp busy waiting:
    a. Kiểm tra luân phiến (turn)
    b. Biến cờ hiệu (flag)
    c. Peterson
    d. Beckery
    e. Mutex lock
    f. Spin lock

b) Các phương pháp sleep_wakeup:
    a. Semaphore
    b. Critical region
    c. Monitor

So sánh các 2 cơ chế phổ biến – Semaphore và Mutex Lock:

Tiêu chí	**Mutex Lock**	**Semaphore**
Mục đích	Đảm bảo rằng chỉ có một luồng truy cập vào tài nguyên bảo vệ tại một thời điểm.	Đảm bảo đồng bộ hóa và kiểm soát truy cập đến một tài nguyên có thể được chia sẻ bởi nhiều luồng.
Đặc điểm	Có thể được sử dụng để bảo vệ truy cập đến dữ liệu hoặc các tài nguyên động.	Có thể kiểm soát số lượng luồng có thể truy cập cùng một lúc vào một tài nguyên.
Kiểm soát	Có hai trạng thái: khóa (lock) và mở (unlock). Luồng phải khóa mutex trước khi truy cập vào tài nguyên và mở mutex sau khi hoàn thành truy cập.	Có thể được khởi tạo với một giá trị ban đầu và có thể được tăng hoặc giảm bằng cách sử dụng các hàm đợi (wait) và phát (post).
Cơ chế bảo vệ	Mutex sử dụng cơ chế khóa (lock) và mở (unlock) để đảm bảo rằng chỉ một luồng có thể truy cập vào tài nguyên tại một thời điểm.	Semaphore sử dụng biến đếm để kiểm soát số lượng luồng có thể truy cập cùng một lúc vào một tài nguyên.
Ứng dụng khi	Chỉ có một luồng được phép truy cập vào tài nguyên hoặc phần của mã tại một thời điểm. Cần bảo vệ truy cập đến dữ liệu hoặc tài nguyên động, chẳng hạn như biến, danh sách liên kết, hoặc cấu trúc dữ liệu.	Cần kiểm soát truy cập đồng thời của nhiều luồng vào một tài nguyên. Cần giới hạn số lượng luồng được phép truy cập vào một tài nguyên cùng một lúc. Cần quản lý tài nguyên có thể được chia sẻ bởi nhiều luồng như kết nối cơ sở dữ liệu, slot trong bộ nhớ đệm, hoặc giới hạn số lượng luồng được phép thực hiện một

		hoạt động cùng một lúc.
Hạn chế sử dụng	Mutex thích hợp cho các tài nguyên có thể được truy cập bởi một luồng duy nhất tại một thời điểm.	Semaphore thích hợp cho các tài nguyên có thể được chia sẻ bởi nhiều luồng nhưng cần kiểm soát số lượng luồng có thể truy cập cùng một lúc.

Race condition là một lỗi logic nghiêm trọng có thể xảy ra trong các hệ thống phần mềm, bao gồm cả hệ thống ngân hàng. Các ngân hàng cần áp dụng các biện pháp phòng ngừa để tránh race condition, chẳng hạn như sử dụng khóa, phiên bản và thiết kế hệ thống cẩn thận.