Top 3 most complex interactions of transactions

- *add_team_to_fantasy_league:*
    - This endpoint adds a fantasy team to a fantasy league. It first checks that the value of the team is less than or equal to the budget of the fantasy league. The value of the team is calculated by the sum of all its player values.
    - A Non-Repeatable Read could occur if the league budget changes while this endpoint is running. For example, if another endpoint alters either the value of the team or the budget of the league after this endpoint queries one of these two amounts, there could end up being a fantasy team in a fantasy league that has more money than it is allowed.
- *add_player_to_fantasy_team:*
    - This endpoint adds a player to a fantasy team. It first checks if the team can afford the player, meaning that the value of the player is less than or equal to the balance of the team. If the team can afford the player, the player is added, and the value is subtracted from the team balance.
    - A Non-Repeatable Read could occur if two players are being added to the same team at the same time, both queries will pass the check where the team balance is high enough to purchase each player, and then both player values will be deducted from the team balance. This could lead to a fantasy team having a negative balance.
    - A similar Non-Repeatable Read could occur in this same scenario. A player can only be added to a team if the number of current players on the team is less than 11. If a team has 10 players, and there are two simultaneous calls to this endpoint, both queries would pass the check where there are less than the max amount of players on the team, and both would be added. This would cause a team to have more than the maximum allowed number of players, which would affect their total score and balance.
    - A Dirty Read could occur if a very high value player (who's too expensive for the team) and a low value player are being added to the same team at the same time. The high value player would make the team balance temporarily go negative, and then the low value player will not be able to get added because it read the team balance as negative. The high value player transaction gets canceled and the team balance gets rolled back to what it was.
- *remove_player_from_fantasy_team:*
    - This endpoint removes a player from the fantasy team and refunds the value of the player back to the team.
    - A Non-Repeatable read could occur if two players are removed at the same time. The endpoint first queries the current team balance, then adds the player value to that balance and updates it. If the call to remove the second player gets the team balance before it is updated with the value of the first player, the new balance will only account for the value of the second player.

Solution

To prevent these potential concurrency issues, we added a "for share" clause at the end of each query to acquire a shared lock on the selected rows during a transaction. This prevents other transactions from accessing the data until the initial transaction is committed and releases the lock. For instance, if two simultaneous calls to add_team_to_fantasy_league are made, the database will pick one of the transactions to carry out, and the other transaction cannot access the data currently being accessed by the initial transaction until it is complete.