**Dirty read in SQL**

There are mainly four types of common concurrency problems: dirty read, lost read, non-repeatable read and phantom reads.

**Explanation –**  
If we have a ticket booking system and One Customer is trying to book a ticket at that time available number of the ticket is 10, before completing the payment, the Second Customer wants to book a ticket that time this 2nd transaction will show the second customer that the number of the available tickets is 9. The twist is here if the first customer does not have sufficient fund in his debit card or in his wallet then the 1st transaction will Rollback, that time 9 seat available which is read by the 2nd transaction is *Dirty Read*.

**Example:**  
**Available ticket:** For 1st customer

* **1st Step –**
* **Input:**
* -- Transaction 1

Select \*from Bus\_ticket;

**Output:**

|  |  |  |
| --- | --- | --- |
| ID | Bus\_Name | Available\_Seat |
| 1 | KA0017 | 10 |

* **2nd Step –**  
  Booking time for 1st customer
* **Input:**
* --Transaction 1
* BEING Transaction
* UPDATE Bus Ticket set Available Seat=9
* WHERE ID=1
* --Payment for Transaction 1
* Wait for Delay '00.00.30'

Rollback transaction

**Available ticket:** For 2nd customer while 1st customer is paying for the ticket.

* **3rd Step –**
* **Input:**
* -- Transaction 1
* set transaction isolation level read uncommitted

Select \* from Bus ticket where ID=1;

**Output:**

|  |  |  |
| --- | --- | --- |
| ID | Bus\_Name | Available\_Seat |
| 1 | KA0017 | 9 |

Note that during the payment of 1st customer 2nd transaction read it 9 seat is available if some how 1st transaction Rollback then available seat 9 that is Dirty read data. After rollback of the 1st transaction available seat is 10 again. 2nd and 3rd step happening at the same time.

Actually available seat after rollback of transaction 1:

|  |  |  |
| --- | --- | --- |
| ID | Bus\_Name | Available\_Seat |
| 1 | KA0017 | 10 |