

Phase 5: Apex Programming (Developer)

Apex Trigger – Update Show Available Seats

- Object: Booking__c
- Purpose: Automatically updates the available seats of a show whenever a booking is created or updated.
- When a new booking is inserted, the trigger reduces the available seats in the related Show__c record.
- When a booking is updated, it adjusts the available seats based on the change in Seats_Booked__c.
- Result: Ensures the Show__c.Available_Seats__c always reflects the correct number of seats available.

❖ Apex Trigger Code

trigger BookingTrigger_UpdateSeats on Booking__c (after insert, after update, after delete) {

 // Map to track seat changes per Show

 Map<Id, Decimal> seatChangeMap = new Map<Id, Decimal>();

 // Insert / Update

 if (Trigger.isInsert || Trigger.isUpdate) {

 for (Booking__c b : Trigger.new) {

 if (b.Show__c != null && b.Seats_Booked__c != null) {

 Decimal change = b.Seats_Booked__c

 // For update, subtract old seats

 if (Trigger.isUpdate && Trigger.oldMap.containsKey(b.Id)) {

 Decimal oldSeats = Trigger.oldMap.get(b.Id).Seats_Booked__c != null

 ? Trigger.oldMap.get(b.Id).Seats_Booked__c

```

        : 0;

        change = b.Seats_Booked__c - oldSeats;

    }

    if (!seatChangeMap.containsKey(b.Show__c)) seatChangeMap.put(b.Show__c,
0);

    seatChangeMap.put(b.Show__c, seatChangeMap.get(b.Show__c) + change);

    }

    }

    }

// Delete

if (Trigger.isDelete) {

    for (Booking__c b : Trigger.old) {

        if (b.Show__c != null && b.Seats_Booked__c != null) {

            Decimal change = -b.Seats_Booked__c;

            if (!seatChangeMap.containsKey(b.Show__c)) seatChangeMap.put(b.Show__c,
0);

            seatChangeMap.put(b.Show__c, seatChangeMap.get(b.Show__c) + change);

        }

    }

}

// Update Show__c Available_Seats__c

if (!seatChangeMap.isEmpty()) {

```

```
List<Show__c> showsToUpdate = [SELECT Id, Available_Seats__c FROM
Show__c WHERE Id IN :seatChangeMap.keySet()];

for (Show__c s : showsToUpdate) {

    Decimal current = s.Available_Seats__c != null ? s.Available_Seats__c : 0;

    Decimal newAvailable = current - seatChangeMap.get(s.Id);

    s.Available_Seats__c = newAvailable >= 0 ? newAvailable : 0;

}

update showsToUpdate;

}

}
```

```

File • Edit • Debug • Test • Workspace • Help • < • >
BookingTrigger_UpdateSeats.apxt
Code Coverage: None • API Version: 64
Go To

1 trigger BookingTrigger_UpdateSeats on Booking__c (after insert, after update, after delete) {
2
3     // Map to track seat changes per Show
4     Map<Id, Decimal> seatChangeMap = new Map<Id, Decimal>();
5
6     // Insert / Update
7     if (Trigger.isInsert || Trigger.isUpdate) {
8         for (Booking__c b : Trigger.new) {
9             if (b.Show__c != null && b.Seats_Booked__c != null) {
10                 Decimal change = b.Seats_Booked__c;
11
12                 // For update, subtract old seats
13                 if (Trigger.isUpdate && Trigger.oldMap.containsKey(b.Id)) {
14                     Decimal oldSeats = Trigger.oldMap.get(b.Id).Seats_Booked__c != null
15                         ? Trigger.oldMap.get(b.Id).Seats_Booked__c
16                         : 0;
17                     change = b.Seats_Booked__c - oldSeats;
18                 }
19             }
20         }
21     }
22 }
23
24 }
25
26 // Delete
27 if (Trigger.isDelete) {
28     for (Booking__c b : Trigger.old) {
29         if (b.Show__c != null && b.Seats_Booked__c != null) {
30             Decimal change = -b.Seats_Booked__c;
31             if (!seatChangeMap.containsKey(b.Show__c)) seatChangeMap.put(b.Show__c, 0);
32             seatChangeMap.put(b.Show__c, seatChangeMap.get(b.Show__c) + change);
33         }
34     }
35 }
36
37 // Update Show__c Available_Seats__c
38 if (!seatChangeMap.isEmpty()) {
39     List<Show__c> showsToUpdate = [SELECT Id, Available_Seats__c FROM Show__c WHERE Id IN :seatChangeMap.keySet()];
40     for (Show__c s : showsToUpdate) {
41         Decimal current = s.Available_Seats__c != null ? s.Available_Seats__c : 0;
42         Decimal newAvailable = current - seatChangeMap.get(s.Id);
43         s.Available_Seats__c = newAvailable >= 0 ? newAvailable : 0;
44     }
45     update showsToUpdate;
46 }
47 }

```

```

File • Edit • Debug • Test • Workspace • Help • < • >
BookingTrigger_UpdateSeats.apxt
Code Coverage: None • API Version: 64
Go To

22 }
23
24 }
25
26 // Delete
27 if (Trigger.isDelete) {
28     for (Booking__c b : Trigger.old) {
29         if (b.Show__c != null && b.Seats_Booked__c != null) {
30             Decimal change = -b.Seats_Booked__c;
31             if (!seatChangeMap.containsKey(b.Show__c)) seatChangeMap.put(b.Show__c, 0);
32             seatChangeMap.put(b.Show__c, seatChangeMap.get(b.Show__c) + change);
33         }
34     }
35 }
36
37 // Update Show__c Available_Seats__c
38 if (!seatChangeMap.isEmpty()) {
39     List<Show__c> showsToUpdate = [SELECT Id, Available_Seats__c FROM Show__c WHERE Id IN :seatChangeMap.keySet()];
40     for (Show__c s : showsToUpdate) {
41         Decimal current = s.Available_Seats__c != null ? s.Available_Seats__c : 0;
42         Decimal newAvailable = current - seatChangeMap.get(s.Id);
43         s.Available_Seats__c = newAvailable >= 0 ? newAvailable : 0;
44     }
45     update showsToUpdate;
46 }
47 }

```

```

File • Edit • Debug • Test • Workspace • Help • < • >
BookingTrigger_UpdateSeats.apxt
Code Coverage: None • API Version: 64
Go To

29 if (b.Show__c != null && b.Seats_Booked__c != null) {
30     Decimal change = -b.Seats_Booked__c;
31     if (!seatChangeMap.containsKey(b.Show__c)) seatChangeMap.put(b.Show__c, 0);
32     seatChangeMap.put(b.Show__c, seatChangeMap.get(b.Show__c) + change);
33 }
34 }
35 }
36
37 // Update Show__c Available_Seats__c
38 if (!seatChangeMap.isEmpty()) {
39     List<Show__c> showsToUpdate = [SELECT Id, Available_Seats__c FROM Show__c WHERE Id IN :seatChangeMap.keySet()];
40     for (Show__c s : showsToUpdate) {
41         Decimal current = s.Available_Seats__c != null ? s.Available_Seats__c : 0;
42         Decimal newAvailable = current - seatChangeMap.get(s.Id);
43         s.Available_Seats__c = newAvailable >= 0 ? newAvailable : 0;
44     }
45     update showsToUpdate;
46 }
47 }

```

❖ Minimal Test Class for Booking Trigger

- To verify that the BookingTrigger_UpdateSeats trigger works correctly.
- When a booking is inserted, the show's available seats decrease.
- When a booking is updated, the available seats adjust correctly.
- When a booking is deleted, the seats are restored.

@isTest

```
private class BookingTriggerTest {
```

```
    @isTest static void testBookingTrigger() {
```

```
        // Create Show
```

```
        Show__c s = new Show__c(Name='Test Show', Available_Seats__c=50);
```

```
        insert s;
```

```
        // Insert Booking
```

```
        Booking__c b = new Booking__c(Name='B1', Show__c=s.Id,
        Seats_Booked__c=5);
```

```
        insert b;
```

```
        // Verify seats decreased
```

```
        s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
```

```
        System.assertEquals(45, s.Available_Seats__c);
```

```
        // Update Booking
```

```
        b.Seats_Booked__c = 8;
```

```
        update b;
```

```
        s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
```

```
        System.assertEquals(42, s.Available_Seats__c);
```

```
        // Delete Booking
```

```
        delete b;
```

```
        s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
```

```
        System.assertEquals(50, s.Available_Seats__c);
```

```
    } }
```

Search Setup

★

+

🏠

?

⚙️

🔔

👤

ct Manager

SETUP

Apex Classes

BookingTriggerTest

Apex Class Detail

Edit

Delete

Download

Run Test

Show Dependencies

Name	BookingTriggerTest	Status	Active
Namespace Prefix		Created By	Dhanshree Gawhale
Last Modified By	Dhanshree Gawhale		9/26/2025, 2:57 AM

Class Body

Class Summary

Version Settings

Trace Flags

```
1  @isTest
2  private class BookingTriggerTest {
3      @isTest static void testBookingTrigger() {
4          // Create Show
5          Show__c s = new Show__c(Name='Test Show', Available_Seats__c=50);
6          insert s;
7
8          // Insert Booking
9          Booking__c b = new Booking__c(Name='B1', Show__c=s.Id, Seats_Booked__c=5);
10         insert b;
11
12         // Verify seats decreased
13         s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
14         System.assertEquals(45, s.Available_Seats__c);
15
16         // Update Booking
17         b.Seats_Booked__c = 8;
18         update b;
19         s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
20         System.assertEquals(42, s.Available_Seats__c);
21
22         // Delete Booking
23         delete b;
24         s = [SELECT Available_Seats__c FROM Show__c WHERE Id = :s.Id];
```