1)Register for an event by providing user ID and event ID. First check if the event exists. If not print out a message saying wrong event ID. Then check whether the event is full. If so, put the user on the wait list with the next position and print out a message saying you are on waitlist and the waitlist position. Otherwise register the user with the event and print a message you are registered.

Comment- It registers for an event by providing user ID and event ID and checks if the event exists. If not print out a message saying wrong event ID. Then its check whether the event is full. If so, put the user on the wait list with the next position and print out a message saying you are on waitlist and the waitlist position Otherwise procedure register the user with the event and print a message you are registered.

```
create or replace PROCEDURE ParticipantRegister(user_id in integer, event_id in integer)
IS

t_location LOCATIONS.LDESC%type;
t_eventid event.eid%type;

t_event event%rowtype;

t_participantnumber integer;
t_maxp_event integer;  -- Max participant

t_max_waitrank integer;
n_waitlist integer;
t_userid integer;

check_dup_p integer;
check_dup_w integer;


BEGIN

  -- get user and event id
  select * into t_event from event where eid = event_id and ecflag ='0';
  select userid into t_userid from usertable where userid = user_id;


    -- get user and event id -- check whether participant exists or not?
  check_dup_p := 0;
  check_dup_w := 0;
  select count(*) into check_dup_p from participant
  where eid = event_id and userid = user_id and pcflag ='0';
```

```
select count(*) into check_dup_w from waitlist
where eid = event_id and userid = user_id;

if check_dup_p >= 1 or check_dup_w >= 1 then
     dbms_output.put_line('You have already registered for the event before.');
else

   -- check the event id and user id are not null.
   if t_event.eid is not null and t_userid is not null then



     select count(*) into t_participantnumber from participant
     where EID = t_event.eid and pcflag ='0';

     select lcap into t_maxp_event from locations where lid = t_event.lid;
 --  dbms_output.put_line('p: ' || t_participantnumber);
 --  dbms_output.put_line('m: ' || t_maxp_event);
     if t_participantnumber = t_maxp_event then


        select max(wrank) into t_max_waitrank from waitlist where eid = t_event.eid;

        -- Set waitlist = 0 for the first waitlist
        if t_max_waitrank is null then
          t_max_waitrank := 0;
        end if;

        -- insert waitlist rank + 1 record
        insert into waitlist values(user_id,event_id,t_max_waitrank+1);

        -- send the message saying that the user is on the waitlist
        n_waitlist := t_max_waitrank+1;

        insert into message values(msg_seq.nextval,user_id, 'The event ID:' || event_id ||
        ' is full. User ID: ' || user_id
        || ' You have been added in the waitlist rank : ' ||
        n_waitlist, systimestamp);

        dbms_output.put_line('The event ID:' || event_id ||
        ' is full. User ID: ' || user_id
        || ' You have been added in the waitlist rank : ' ||
        n_waitlist);

     else
```

```
        insert into participant values(pid_seq.nextval,0,'','0',event_id,user_id);
        insert into message values(msg_seq.nextval,user_id, 'The event ID:' || event_id ||
        ' is available. User ID : '|| user_id ||
        ' You have been registered as a participant.', systimestamp);

        dbms_output.put_line('The event ID:' || event_id ||
        ' is available. User ID : '|| user_id ||
        ' You have been registered as a participant.');

      end if;

    end if;
  end if;

  exception
  when no_data_found then
    dbms_output.put_line('Wrong user or event ID.');

END;
```

Test case 1: register an event that is available:
By running
exec ParticipantRegister(7,6);
Check participant table for user id 7, event 6
Select * from participant where userid = 7 and eid = 6;

User 6 has been registered for Event 6.

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
50	0	(null)	0	6	7

Test case 2: User register in to the unavailable event.
By running

exec ParticipantRegister(10,8); -- user 10 want to register for event 8.

You will get the message saying that you have been put in to the waitlist + ranking.

```
PL/SQL procedure successfully completed.

The event ID:8 is full. User ID: 10 You have been added in the waitlist rank : 4
```

Check wait list table
Select * from waitlist where eid = 8;

We will see that user id 10 has been put in the wait list with rank 4.

USERID	EID	WRANK
1	8	1
2	8	2
3	8	3
10	8	4

Test case 3: register with user who has already been registered
By running
exec ParticipantRegister(7,6);
Also user who is already put in the waitlist.
exec ParticipantRegister(10,8);

```
You have already registered for the event before.
```

2. Cancel an event registration by providing user ID and event ID. Please check whether the event exists and the user has indeed registered. If the user is registered, delete the registration and register the first person on the waitlist (if the wait list is not empty). Generate a message (insert into message table) for the current user saying event canceled. If a user on top of wait list is now registered, generate a message to that user saying that he or she is now registered with the event.

Comment- It cancels an event registration by providing user ID and event ID and also checks whether the event exists and the user has indeed registered the user registered can delete the registration and register the first person on the waitlist (if the wait list is not empty). And it can also generate a message (insert into message table) for the current user saying event cancelled and a user on top of wait list can be registered and procedure can generate a message to the user saying that he or she is now registered with the event.

```
create or replace PROCEDURE RegisterCancellation(user_id in integer, event_id in integer)
IS
t_location LOCATIONS.LDESC%type;
t_eventid event.eid%type;
t_event event%rowtype;
t_participantnumber integer;
t_maxp_event integer;  -- Max participant
t_max_waitrank integer;
n_waitlist integer;
t_userid integer;

t_waitlist waitlist%rowtype;
t_waitlist_check integer;
```

```
BEGIN

  -- get user and event id -- check whether participant exists or not?
  select userid, eid into t_userid, t_eventid from participant
  where eid = event_id and userid = user_id and pcflag ='0';

  --select userid into t_userid from usertable where userid = user_id;
  --select eid into t_eventid from event where eid = event_id;

  -- check the event id and user id are not null.
  if t_eventid is not null and t_userid is not null then

    select * into t_event from event where eid = event_id;

    dbms_output.put_line('User/event cancel request: ' || user_id
       || ' : ' || event_id);

    t_participantnumber := 0;
    select count(*) into t_participantnumber from participant
    where EID = t_eventid and pcflag ='0';

    select lcap into t_maxp_event from locations where lid = t_event.lid;
    --dbms_output.put_line('p: ' || t_participantnumber);
    --dbms_output.put_line('m: ' || t_maxp_event);

    -- the event is full
    if t_participantnumber = t_maxp_event then

       -- select the first rank in waitlist.
       --dbms_output.put_line('Test');
       select count(*) into t_waitlist_check from waitlist where eid = event_id;

       if t_waitlist_check > 0 then

          select * into t_waitlist from waitlist where eid = event_id
          and wrank = 1;

          dbms_output.put_line('First rank in wait list id: ' || t_waitlist.userid);


          -- update the participant cancel flag
          UPDATE participant
          SET pcflag = '1'
          WHERE eid = event_id and userid = user_id;
```

```
        -- insert participant from waitlist no.1
        insert into participant values(pid_seq.nextval,0,'','0',event_id,t_waitlist.userid);

        delete from waitlist where userid = t_waitlist.userid and eid = event_id;

        -- update another waitlist in the same event -1 rank
        UPDATE waitlist
        SET wrank = wrank -1
        WHERE eid = event_id;

        -- sending the message to the new participant.
        insert into message values(msg_seq.nextval,t_waitlist.userid,
        'You have been registered for the event ID :' || event_id
        , systimestamp);

        dbms_output.put_line('User : ' || t_waitlist.userid ||
        ', you have been registered for the event ID :' || event_id);

    -- no waitlist and event is full... just cancel participant.
    else

        UPDATE participant
        SET pcflag = '1'
        WHERE eid = event_id and userid = user_id;

        insert into message values(msg_seq.nextval,user_id, 'User ID : '
        ||user_id || ' You have been cencelled.'
        || ' from the event ID : '|| event_id, systimestamp);

        dbms_output.put_line('User ID : '||user_id || ' You have been cencelled.'
        || ' from the event ID : '|| event_id);

    end if;

-- else the event is available no wait list.
else

    UPDATE participant
    SET pcflag = '1'
    WHERE eid = event_id and userid = user_id;

    insert into message values(msg_seq.nextval,user_id, 'User ID : '
    ||user_id || ' You have been cencelled.'
    || ' from the event ID : '|| event_id, systimestamp);

    dbms_output.put_line('User ID : '||user_id || ' You have been cencelled.'
```

```
      || ' from the event ID : '|| event_id);

   end if;

 end if;

 exception
 when no_data_found then
    dbms_output.put_line('Wrong user or event ID.');

END;
```

Test case 1
Cancel the registration (Normal case with no waitlist in the event)
Cancel userid 1 from the event 1.

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
1	0	(null)	0	1	1
2	0	(null)	0	1	2

by running
set serveroutput on;
exec RegisterCancellation(1,1);
After execute the program we will get message.

```
PL/SQL procedure successfully completed.

User/event cancel request: 1 : 1
User ID : 1 You have been cencelled. from the event ID : 1
```

And user 1 has been flagged as cancel in participant table

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
1	0	(null)	1	1	1
2	0	(null)	0	1	2

Test case 2: Cancellation the participant in the event that has another waitlist.
Example: event 8 has full participants and waitlist
Participant table

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
6	0	(null)	0	8	6
7	0	(null)	0	8	7
8	0	(null)	0	8	8
9	0	(null)	0	8	9

waitlist table

USERID	EID	WRANK
1	8	1
2	8	2
3	8	3
10	8	4

Test Cancel user id 6 from the event 8.

By exec RegisterCancellation(6,8);

The system return message says that

```
User/event cancel request: 6 : 8
First rank in wait list id: 1
User : 1, you have been registered for the event ID :8
```

Check the participant table
User id 1 has been registered.

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
52	0	(null)	0	8	1

Then check the waitlist table for event 8.
All waitlist for event 8 have been moved up 1 position.

USERID	EID	WRANK
2	8	1
3	8	2
10	8	3

Check the message table to prove that user id 1 has been informed that he/she has been registered for event 8.

56	1	You have been registered for the event ID :8	17-MAY-16 09.49.04.865000000 PM

3. Enter a review for an event. The user provides a numerical rating (1 to 5) as well as some comment. To prevent abuse, the feature needs to check that the user has registered for the event and only registered users can enter reviews.

Comment- It allows registered users of the events to write reviews and numerical ratings (1 to 5)

```
create or replace PROCEDURE UserEventReview(user_id in integer, event_id in integer
, event_rate in integer, event_comment in varchar)
IS

t_location LOCATIONS.LDESC%type;
t_eventid event.eid%type;
t_event event%rowtype;
t_participantnumber integer;
t_maxp_event integer;  -- Max participant
t_max_waitrank integer;
n_waitlist integer;
t_userid integer;

BEGIN

  -- get user and event id -- check whether participant exists or not?
  select userid, eid into t_userid, t_eventid from participant
  where eid = event_id and userid = user_id and pcflag ='0';

  -- check the event id and user id are not null.
  --dbms_output.put_line('Check!!');
  if t_eventid is not null and t_userid is not null then

    select count(*) into t_participantnumber from participant
    where EID = t_event.eid and pcflag ='0';


    UPDATE participant
          SET prating = event_rate, PCOMMENT = event_comment
          WHERE eid = event_id and userid = user_id and pcflag ='0';
    --dbms_output.put_line('Update!!');
  end if;

  exception
  when no_data_found then
     dbms_output.put_line('Wrong user or event ID. Or user is not registered for the event');

END;
```

Test case 1:
User id 2 want to rate the event 6

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
10	0	(null)	0	6	2

We can test this function by:
exec UserEventReview(2, 6, 5, 'Great Event !!');

```
PL/SQL procedure successfully completed.
```

Then check the participant table to see the update.

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
10	5	Great Event !!	0	6	2

Test case 2:
The participant who is not in the event will not be able to vote the event.

PID	PRATING	PCOMMENT	PCFLAG	EID	USERID
10	5	Great Event !!	0	6	2
11	0	(null)	0	6	3
12	0	(null)	0	6	4
13	0	(null)	0	6	5
50	0	(null)	0	6	7

As No user 1 in the table, we can test this case by
exec UserEventReview(1, 6, 3, 'Event ? !!');
The system will return error message as below:

```
PL/SQL procedure successfully completed.

Wrong user or event ID. Or user is not registered for the event
```