# Print 'NOT confirmed' based on the reservation status, of a particular passenger.

SQL> declare

1. sn passenger\_details.serial\_no%type;
2. r passenger\_details.reservation\_status%type; 4
3. begin
4. dbms\_output.put\_line('Enter the serial number of the passenger to view their reservation status: ');
5. sn:=&sn;
6. select reservation\_status into r from passenger\_details where serial\_no=sn;
7. if (r='Waitlisted') then
8. dbms\_output.put\_line('NOT confirmed'); 11
9. else
10. dbms\_output.put\_line('Confirmed');
11. end if;
12. end;

16 /

Enter value for sn: 4 old 7: sn:=&sn;

new 7: sn:=4;

Enter the serial number of the passenger to view their reservation status: NOT confirmed

PL/SQL procedure successfully completed.

# Print the total seats available for a particular train and for a particular class.

SQL> declare

1. tn total\_seat\_count.train\_no%type;
2. cls total\_seat\_count.class%type;
3. seat\_tot total\_seat\_count.total\_seats%type; 5
4. begin
5. dbms\_output.put\_line('To find the total number of seats follow: ');
6. dbms\_output.put\_line('Enter the train number: ');
7. tn:=&tn;
8. dbms\_output.put\_line('Enter the class: ');
9. cls:='&cls';
10. select total\_seats into seat\_tot from total\_seat\_count
11. where train\_no=tn and class=cls;
12. dbms\_output.put\_line('The total seats available for this particular train are '||seat\_tot);
13. end;

16 /

Enter value for tn: 12755 old 9: tn:=&tn;

new 9: tn:=12755; Enter value for cls: 1A old 11: cls:='&cls';

new 11: cls:='1A';

To find the total number of seats follow: Enter the train number:

Enter the class:

The total seats available for this particular train are 1 PL/SQL procedure successfully completed.

1. Write a cursor for the following.

# Retrieve the passenger details for “x” train number and given journey date.

SQL> declare

* 1. tn ticket.train\_number%type;
  2. doj ticket.date\_of\_journey%type;
  3. cursor p\_cur is select p.name,p.age,p.reservation\_status,t.transactionid from passenger\_details p ,ticket t
  4. where p.pnr\_no=t.pnr\_no
  5. and train\_number=&tn and date\_of\_journey='&doj';
  6. p\_details p\_cur%rowtype; 8

1. begin
2. open p\_cur;
3. loop
4. fetch p\_cur into p\_details ;
5. exit when p\_cur%notfound;
6. end loop;
7. dbms\_output.put\_line('PASSENGER DETAILS ARE');
8. dbms\_output.put\_line('Name: '||p\_details.name);
9. dbms\_output.put\_line('Age: '||p\_details.age);
10. dbms\_output.put\_line('Reservation Status: '||p\_details.reservation\_status);
11. dbms\_output.put\_line('Transaction ID: '||p\_details.transactionid); 20
12. close p\_cur;
13. end;

23 /

Enter value for tn: 15430

Enter value for doj: 04-DEC-19

old 6: and train\_number=&tn and date\_of\_journey='&doj';

new 6: and train\_number=5430 and date\_of\_journey='04-DEC-19'; PASSENGER DETAILS ARE

Name: Lauryn Age: 20

Reservation Status: Confirmed Transaction ID: 890117641211095

PL/SQL procedure successfully completed.

# Display the train name(once) and the substation names.

SQL> declare

1. cursor t\_cur is select t.name,t.train\_number,tt.from\_station,tt.to\_station
2. from train t ,ticket tt where t.train\_number=tt.train\_number; 4

5 t\_details t\_cur%rowtype; 6

1. begin
2. open t\_cur;
3. loop
4. fetch t\_cur into t\_details;
5. exit when t\_cur%notfound;
6. dbms\_output.put\_line('Train Name: '||t\_details.name);
7. dbms\_output.put\_line('From station: '||t\_details.from\_station||' To station: '||t\_details.to\_station);
8. dbms\_output.put\_line(' ');
9. end loop;
10. close t\_cur;
11. end;

18 /

Train Name: Katpadi Express

From station: Chennai To station: Katpadi

Train Name: Shatabdi Express

From station: Amritsar To station: Delhi

Train Name: Krishna Express

From station: Vellore To station: Tirupati

Train Name: Krishna Express

From station: Vellore To station: Tirupati

Train Name: Charminar Express

From station: Rajkot To station: Dhanbaad

Train Name: Rajdhani Express

From station: Bombay To station: Kovalam

Train Name: Rajdhani Express

From station: Chennai To station: Pune

PL/SQL procedure successfully completed.

# Display the fare details of a particular train(use basic exceptions)

SQL> declare

1. tn train\_ticket\_fare.train\_no%type;
2. cursor fare\_cur is select \* from train\_ticket\_fare
3. where train\_no=&tn;
4. f\_details fare\_cur%rowtype; 6

7

1. begin
2. open fare\_cur;
3. loop
4. fetch fare\_cur into f\_details;
5. exit when fare\_cur%notfound;
6. end loop;

14

1. dbms\_output.put\_line('FARE DETAILS');
2. dbms\_output.put\_line('Base Fare: '||f\_details.base\_fare);
3. dbms\_output.put\_line('Reservation Charge: '||f\_details.reservation\_charge);
4. dbms\_output.put\_line('Superfast Charge: '||f\_details.superfast\_charge);
5. dbms\_output.put\_line('Other Charge: '||f\_details.other\_charge);
6. dbms\_output.put\_line('Taktal Charge: '||f\_details.taktal\_charge);
7. dbms\_output.put\_line('Service Tax: '||f\_details.service\_tax); 22
8. exception
9. when no\_data\_found then
10. dbms\_output.put\_line('Train fare details not availbale'); 26

27 end;

28 /

Enter value for tn: 11625 old 4: where train\_no=&tn;

new 4: where train\_no=11625; FARE DETAILS

Base Fare: 250

Reservation Charge: 20

Superfast Charge: 34

Other Charge: 10

Taktal Charge: 4

Service Tax: 10

PL/SQL procedure successfully completed.

# Write a cursor to update the reservation status of the passengers (generate seat number, if seats have reached maximum, put waiting list number (30% of total seats), if waiting list number reaches maximum, put PQWL (10%of total seats), RAC-20%) . SQL> declare

1. pnr ticket.pnr\_no%type;
2. seat total\_seat\_count.total\_seats%type;
3. rs passenger\_details.reservation\_status%type;
4. tno total\_seat\_count.train\_no%type;
5. cursor seat\_cur is select total\_seats from total\_seat\_count where train\_no=&tno;
6. wl number;
7. pqwl number;
8. temp varchar2(100);
9. tem varchar2(100);

11

1. begin
2. tno:=&tno;
3. pnr:=&pnr; 15 wl:=0;

16 pqwl:=0; 17

1. open seat\_cur;
2. loop
3. fetch seat\_cur into seat;
4. exit when seat\_cur%notfound;
5. end loop; 23

24 select pnr\_no into pnr from ticket where train\_number=tno; 25

26 if seat>100 then 27 wl:= wl+1;

1. temp:='wl'||wl;
2. dbms\_output.put\_line('Seats have reached maximum,you are now added in waiting list');
3. update passenger\_details set reservation\_status=temp where pnr\_no=pnr; 31
4. if wl >30 then
5. pqwl:=pqwl+1;
6. tem:='pqwl'||pqwl;
7. dbms\_output.put\_line('Waiting list reached maximum.You are now in PQWL');
8. update passenger\_details set reservation\_status= tem where pnr\_no=pnr;
9. end if; 38
10. else
11. dbms\_output.put\_line('Seats Allocated');
12. update passenger\_details set reservation\_status='Confirmed' where pnr\_no=pnr;
13. end if;
14. end; 44 /

Enter value for tno: 12755

old 6: cursor seat\_cur is select total\_seats from total\_seat\_count where train\_no=&tno; new 6: cursor seat\_cur is select total\_seats from total\_seat\_count where train\_no=2755;

Enter value for tno: 12755 old 13: tno:=&tno;

new 13: tno:=12755;

Enter value for pnr: 1198326700 old 14: pnr:=&pnr;

new 14: pnr:=1198326700;

Seats Allocated

PL/SQL procedure successfully completed.

**8.**

**Write a PL/SQL procedure to.**

## List the details of passengers who has reserved next to “Mr. X”.

SQL> create or replace procedure next\_person(ser passenger\_details.serial\_no%type) is 2

1. ser\_after passenger\_details.serial\_no%type := ser+1;
2. cursor cur is select \* from passenger\_details where serial\_no=ser\_after;
3. passenger\_rec cur%rowtype; 6

7 begin 8

1. open cur;
2. loop
3. fetch cur into passenger\_rec;
4. exit when cur%notfound;
5. dbms\_output.put\_line('Name: '||passenger\_rec.name||' Age: '||passenger\_rec.age ||' Reservation Status : '||passenger\_rec.reservation\_status);
6. end loop;
7. close cur;
8. end; 17 /

Procedure created.

SQL> exec next\_person(1);

Name: Akshita Age: 19 Reservation Status : Confirmed

PL/SQL procedure successfully completed.

SQL> exec next\_person(2);

Name: Jayalata Age: 78 Reservation Status : Confirmed PL/SQL procedure successfully completed.

## PNR No. of a passengers for a given source and a destination.

SQL> create or replace procedure pnr(s ticket.from\_station%type,d ticket.to\_station%type) is

1. p ticket.pnr\_no%type;
2. begin
3. select pnr\_no into p
4. from ticket
5. where from\_station=s and to\_station=d; 7

8 dbms\_output.put\_line('PNR\_NO is : '||p); 9

10 end; 11 /

Procedure created.

SQL> exec pnr('Rajkot','Dhanbaad'); PNR\_NO is : 1098452132

PL/SQL procedure successfully completed.

**Write a PL/SQL function to.**

1. **Get the PNRNo and return the total ticket fare.**

SQL> create or replace function total\_fare(pnr ticket.pnr\_no%type)

1. return number is
2. tot ticket.total\_ticket\_fare%type; 4
3. begin
4. select total\_ticket\_fare into tot
5. from ticket
6. where pnr\_no=pnr;
7. return tot; 10 end;

11 /

Function created.

SQL> declare

1. a number;
2. begin 4 a:=&a;
3. dbms\_output.put\_line('Total ticket fare for the the given pnr number is: '||total\_fare(a));
4. end; 7 /

Enter value for a: 4879011231 old 4: a:=&a;

new 4: a:=4879011231;

Total ticket fare for the the given pnr number is: 987

PL/SQL procedure successfully completed.

## Get the Passenger name , train no and return the total journey time in hours and minutes.

SQL> create or replace function total\_time(n passenger\_details.name%type,tn ticket.train\_number%type)

1. return number is
2. tt train.traveltime%type; 4
3. begin
4. select traveltime into tt
5. from passenger\_details,ticket,train
6. where passenger\_details.pnr\_no=ticket.pnr\_no and ticket.train\_number=train.train\_number
7. and passenger\_details.name=n and ticket.train\_number=tn;
8. return tt;
9. end; 12 /

Function created.

SQL> declare

1. a number;
2. b varchar(30);
3. begin 5 a:=&a;

6 b:='&b';

1. dbms\_output.put\_line('Total travel time for the the given pnr number is: '||total\_time(b,a));
2. end; 9 /

Enter value for a: 15430 old 5: a:=&a;

new 5: a:=15430;

Enter value for b: Mahira old 6: b:='&b';

new 6: b:='Mahira';

Total travel time for the the given pnr number is: 4.2 PL/SQL procedure successfully completed.

1. **Write a Trigger for the following:**
   1. **When a passenger cancels a ticket, do the necessary process and update the cancellation history table.**

SQL> create or replace trigger update\_cancellation\_list

1. after delete on ticket
2. for each row
3. declare
4. tno ticket.train\_number%type;
5. pname passenger\_details.name%type;
6. pnr ticket.pnr\_no%type;
7. doj ticket.date\_of\_journey%type;
8. rs passenger\_details.reservation\_status%type; 10
9. begin
10. pnr:= :old.pnr\_no;
11. doj := :old.date\_of\_journey;
12. tno:= :old.train\_number; 15
13. select name into pname from passenger\_details where pnr\_no=pnr;
14. update passenger\_details
15. set reservation\_status='Cancelled'
16. where pnr\_no=pnr;
17. rs:='Booked but Cancelled';
18. dbms\_output.put\_line('Deleting person:'||pname);
19. insert into cancellation\_list values(tno,pname,pnr,doj,rs);
20. dbms\_output.put\_line('Cancel history table successfully updated');
21. end; 25 /

SQL> set serveroutput on;

SQL> delete from ticket where pnr\_no=1098452132; Deleting person:Suresh

Cancel history table successfully updated 1 row deleted.

SQL> select \* from cancellation\_list;

TRAIN\_NUMBER NAME PNR\_NO DATE\_OF\_J

RESERVATION\_STATUS

15430 Lauryn 1174520980 04-DEC-19

Booked but Cancelled

12300 Suresh 1098452132 19-NOV-22

Booked but Cancelled

## When train number is changed, update it in referencing tables.

SQL> create or replace trigger update\_tno

1. after update on train
2. for each row
3. declare
4. tno\_old train.train\_number%type;
5. tno\_new train.train\_number%type; 7
6. begin
7. tno\_old:= :old.train\_number;
8. tno\_new:= :new.train\_number;
9. update ticket
10. set train\_number=tno\_new
11. where train\_number=tno\_old;
12. update train\_ticket\_fare
13. set train\_no=tno\_new
14. where train\_no=tno\_old;
15. update train\_route
16. set train\_no=tno\_new
17. where train\_no=tno\_old;
18. dbms\_output.put\_line(‘Train Number successfully updated
19. from '||:old.train\_number||' to '||:new.train\_number ||' for all the tables.');
20. end; 23 /

Trigger created.

SQL> update train set train\_number=18562 where train\_number=15291;

Train Number successfully updated from 15291 to 18562 for all the tables.

1 row updated.

## When a passenger record is inserted reservation status should be automatically updated.

SQL> create or replace trigger update\_reservation\_status

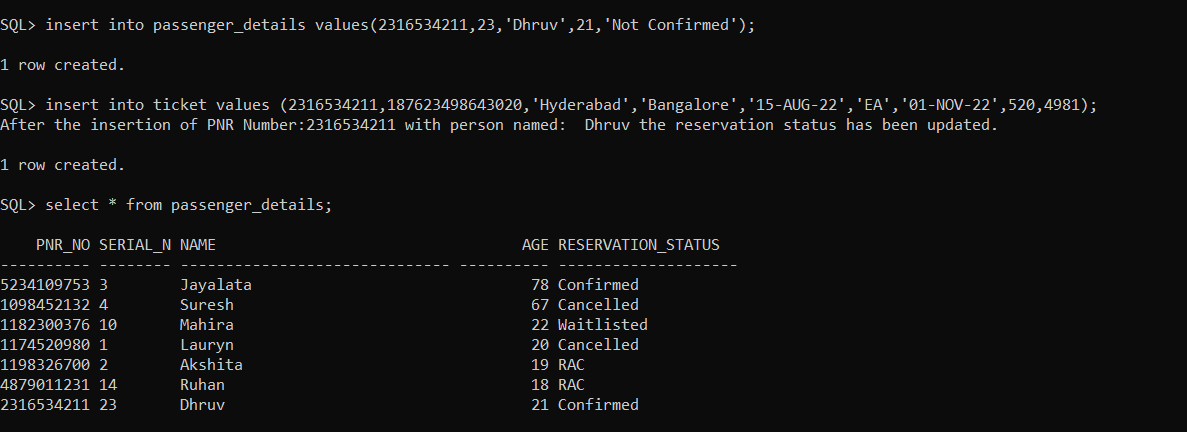
1. after insert on ticket
2. for each row 4
3. declare
4. pnr ticket.pnr\_no%type;
5. rs passenger\_details.reservation\_status%type;
6. pname passenger\_details.name%type; 9
7. begin
8. rs:='Confirmed';
9. pnr:= :new.pnr\_no;
10. select name into pname from passenger\_details where pnr\_no=pnr;
11. update passenger\_details set reservation\_status=rs where pnr\_no=pnr;
12. dbms\_output.put\_line('After the insertion of PNR Number:'||pnr||' with person named: '||pname||' the reservation status has been updated.');
13. end; 17 /

Trigger created.

SQL> insert into ticket values (2316534211,187623498643020,'Hyderabad','Bangalore','15- AUG-22','EA','01-NOV-22',520,14981);

After the insertion of PNR Number:2316534211 with person named: Dhruv the reservation status has been updated.

1 row created.



10.

## 1. Use TCL commands for your transactions. (commit, rollback, savepoint) .

SQL> commit; Commit complete.

SQL> rollback;

Rollback complete.

SQL>savepoint S;

Savepoint created.