Function

1) To find screen id for a movie show in a particular theatre.

```
create function f1(thid in shows.th_id%type,sdt in shows.sdatetime%type,mov in
shows.mov_name%type) return number
 is
 sid number(2);
begin
select screen_id into sid from shows where sdatetime=sdt and th_id=thid and mov_name=mov;
return sid;
end;
/


--calling program
declare
   m shows.mov_name%type;
   th shows.th_id%type;
   sdt1 shows.sdatetime%type;
   scid shows.screen_id%type;
begin
   m:='Tu Jhuthi Main Makkar';
   th:=100;
   sdt1:=to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm');
    scid:=f1(th,sdt1,m);
   dbms_output.put_line(scid);
end;
/
```

Output:

```
10
```

Procedure

1) Procedure to find movie show date and time for a particular movie at a particular theatre location.

```
create procedure p1(thid in shows.th_id%type,mov in shows.mov_name%type)
 is
        sdatetime shows.sdatetime%type;
        cursor c1 is select to_char(sdatetime,'dd-mon-yyyy hh:mi:ss am') as sdatetime from shows
where mov_name=mov and th_id=thid;
        r1 c1%rowtype;
        type t1 is table of r1%type;
        l1 t1;
begin
        open c1;
        fetch c1 bulk collect into l1;
        close c1;
        dbms_output.put_line('Show date and time for movie '||mov||' are');
       for i in 1..l1.count
            loop
             dbms_output.put_line(l1(i).sdatetime);
            end loop;
end;
/

declare
        m shows.mov_name%type;
        th shows.th_id%type;
        sdt1 shows.sdatetime%type;
begin
        m:='Tu Jhuthi Main Makkar';
```

```
        th:=100;
      p1(th,m);
end;
/
```

Output:

```
Show date and time for movie Tu Jhuthi Main Makkar are
20-apr-2023 12:00:00 pm
21-apr-2023 09:30:00 pm
20-apr-2023 05:00:00 pm
```

2) Procedure for showing seat details for a particular movie show on a given selected date and time.

```
--for silver seats
create procedure p2(thid in silver_seat.th_id%type,sdt in silver_seat.sdatetime%type,sid in silver_seat.screen_id%type)
is
cursor c1 is select * from silver_seat where sdatetime=sdt and th_id=thid and screen_id=sid;
        r1 c1%rowtype;
        type t1 is table of r1%type;
        l1 t1;
        p silver_price.price%type;
begin
        select price into p from silver_price where sdatetime=sdt and th_id=thid and screen_id=sid;
        open c1;
        fetch c1 bulk collect into l1;
        close c1;
       dbms_output.put_line('Screen Id: '||sid);
```

```
            dbms_output.put_line('Silver Seats details');
            dbms_output.put_line('Price: '||p);

          for i in 1..l1.count
            loop
            dbms_output.put_line('Seat number: '||l1(i).silver_seatnum||' Screen:
'||l1(i).screen_id||' Theatre: '||l1(i).th_id||' Status: '||l1(i).avail);
          end loop;
end;
/

--for gold seats
create procedure p3(thid in gold_seat.th_id%type,sdt in gold_seat.sdatetime%type,sid in
gold_seat.screen_id%type)
is
 cursor c1 is select * from gold_seat where sdatetime=sdt and th_id=thid and screen_id=sid;
            r1 c1%rowtype;
            type t1 is table of r1%type;
            l1 t1;
            p gold_price.price%type;
begin
            select price into p from gold_price where sdatetime=sdt and th_id=thid and screen_id=sid;
            open c1;
            fetch c1 bulk collect into l1;
            close c1;
           dbms_output.put_line('Screen Id: '||sid);
           dbms_output.put_line('Gold Seats details');
           dbms_output.put_line('Price: '||p);

          for i in 1..l1.count
            loop
```

```
            dbms_output.put_line('Seat number: '||l1(i).gold_seatnum||' Screen: '||l1(i).screen_id||'
Theatre: '||l1(i).th_id||' Status: '||l1(i).avail);
        end loop;
end;
/
```

--calling program

```
declare
            m shows.mov_name%type;
            th shows.th_id%type;
            sdt1 shows.sdatetime%type;
            scid shows.screen_id%type;
begin
            m:='Tu Jhuthi Main Makkar';
            th:=100;
            sdt1:=to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm');
            scid:=f1(th,sdt1,m);
             p2(th,sdt1,scid);
            dbms_output.put_line('----------');
            p3(th,sdt1,scid);
end;
/
```

Output:

```
Screen Id: 10
Silver Seats details
Price: 50
Seat number: A1 Screen: 10 Theatre: 100 Status: 1
Seat number: A2 Screen: 10 Theatre: 100 Status: 1
Seat number: A3 Screen: 10 Theatre: 100 Status: 1
Seat number: A4 Screen: 10 Theatre: 100 Status: 1
Seat number: A5 Screen: 10 Theatre: 100 Status: 1
Seat number: A6 Screen: 10 Theatre: 100 Status: 1
Seat number: A7 Screen: 10 Theatre: 100 Status: 1
-------------
Screen Id: 10
Gold Seats details
Price: 100
Seat number: B1 Screen: 10 Theatre: 100 Status: 1
Seat number: B2 Screen: 10 Theatre: 100 Status: 1
Seat number: B3 Screen: 10 Theatre: 100 Status: 1
Seat number: B4 Screen: 10 Theatre: 100 Status: 1
Seat number: B5 Screen: 10 Theatre: 100 Status: 1
Seat number: B6 Screen: 10 Theatre: 100 Status: 1
Seat number: B7 Screen: 10 Theatre: 100 Status: 1
Seat number: B8 Screen: 10 Theatre: 100 Status: 1
```

Triggers

1) Trigger to update seat availability whenever it is booked for a show at a given date and time for a particular theatre.

```
--for silver ticket

create or replace trigger t1

   after insert on silverticket for each row

begin

   update silver_seat set avail=0 where sdatetime=:new.sdatetime and screen_id=:new.screen_id
and th_id=:new.th_id and silver_seatnum=:new.silver_seatnum;

 end;

/


--for gold ticket

create or replace trigger t2

   after insert on goldticket for each row

begin

   update gold_seat set avail=0 where sdatetime=:new.sdatetime and screen_id=:new.screen_id and
th_id=:new.th_id and gold_seatnum=:new.gold_seatnum;

 end;

/


--**************************

insert into silverticket values(1,50,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy
hh:mi:ss pm'), 10, 100,'A1',2);

select * from silverticket;

select silver_seatnum,screen_id,th_id,to_char(sdatetime,'dd-mon-yyyy hh:mi:ss pm') as
sdatetime,avail from silver_seat where screen_id=10;

insert into goldticket values(2,50,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy
hh:mi:ss pm'), 10, 100,'B1',2);

select * from goldticket;
```

select gold_seatnum,screen_id,th_id,to_char(sdatetime,'dd-mon-yyyy hh:mi:ss pm') as sdatetime,avail from gold_seat where screen_id=10;

Output:

1 row(s) inserted

TICKET_ID	NPRICE	BK_DT_TIME	BK_TYPE	SDATETIME	SCREEN_ID	TH_ID	SILVER_SEATNUM	CID
1	50	19-APR-23	online	20-APR-23	10	100	A1	2

SILVER_SEATNUM	SCREEN_ID	TH_ID	SDATETIME	AVAIL
A1	10	100	20-04-2023 12:00 pm	0
A2	10	100	20-04-2023 12:00 pm	1
A3	10	100	20-04-2023 12:00 pm	1
A4	10	100	20-04-2023 12:00 pm	1
A5	10	100	20-04-2023 12:00 pm	1
A6	10	100	20-04-2023 12:00 pm	1
A7	10	100	20-04-2023 12:00 pm	1

1 row(s) inserted

TICKET_ID	NPRICE	BK_DT_TIME	BK_TYPE	SDATETIME	SCREEN_ID	TH_ID	GOLD_SEATNUM	CID
2	100	19-APR-23	online	20-APR-23	10	100	B1	2

GOLD_SEATNUM	SCREEN_ID	TH_ID	SDATETIME	AVAIL
B1	10	100	20-04-2023 12:00 pm	0
B2	10	100	20-04-2023 12:00 pm	1
B3	10	100	20-04-2023 12:00 pm	1
B4	10	100	20-04-2023 12:00 pm	1
B5	10	100	20-04-2023 12:00 pm	1
B6	10	100	20-04-2023 12:00 pm	1
B7	10	100	20-04-2023 12:00 pm	1

2) Trigger to insert allover shows details in box office and website table whenever it is inserted in shows table.

```
create or replace trigger t3 after insert or update on shows for each row
begin
  insert into box_office values(:new.mov_name,:new.sdatetime,:new.screen_id,:new.th_id);
end;
/

create or replace trigger t4 after insert or update on shows for each row
begin
  insert into website values(:new.mov_name,:new.sdatetime,:new.screen_id,:new.th_id);
end;
/

--**************************

insert into shows values('Tu Jhuthi Main Makkar', to_date('21-Apr-2023 9:00:00 am','dd-mon-yyyy hh:mi:ss pm'), '2.5 hrs', 10, 101);

select mov_name,to_char(sdatetime,'dd-mon-yyyy hh:mi:ss pm') as sdatetime,runtime,screen_id,th_id from shows;

select mov_name,to_char(sdatetime,'dd-mon-yyyy hh:mi:ss pm') as sdatetime,screen_id,th_id from box_office;

select mov_name,to_char(sdatetime,'dd-mon-yyyy hh:mi:ss pm') as sdatetime,screen_id,th_id from website;
```

Output:

1 row(s) inserted.

MOV_NAME	SDATETIME	RUNTIME	SCREEN_ID	TH_ID
Tu Jhuthi Main Makkar	20-04-2023 12:00 pm	2.5 hrs	10	100

MOV_NAME	SDATETIME	SCREEN_ID	TH_ID
Tu Jhuthi Main Makkar	20-04-2023 12:00 pm	10	100

MOV_NAME	SDATETIME	SCREEN_ID	TH_ID
Tu Jhuthi Main Makkar	20-04-2023 12:00 pm	10	100

3) Trigger to check if same seatnum is not being added in silverticket and goldticket table for already booked seat number.

--for silver seats

```
create trigger t5 before insert on silverticket for each row
declare
t_count number:=0;
begin
select count(1) into t_count from silverticket where silver_seatnum=:new.silver_seatnum and sdatetime=:new.sdatetime and th_id=:new.th_id and screen_id=:new.screen_id;
if t_count>0
then
raise_application_error(-20202,'Silver seatnum '||:new.silver_seatnum||' already booked');
end if;
end;
/
```

--for gold seats

```
create trigger t6 before insert on goldticket for each row
declare
```

```
g_count number:=0;

begin

select count(1) into g_count from goldticket where gold_seatnum=:new.gold_seatnum and
sdatetime=:new.sdatetime and th_id=:new.th_id and screen_id=:new.screen_id;

if g_count>0

then

raise_application_error(-20202,'Gold seatnum '||:new.gold_seatnum||' already booked');

end if;

end;

/


--**************************

insert into silverticket values(1,50,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy
hh:mi:ss pm'), 10, 100,'A1',2);

insert into goldticket values(2,100,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy
hh:mi:ss pm'), 10, 100,'B1',2);
```

Output:

```
ORA-20202: Silver seatnum A1 already booked ORA-06512: at
"SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T5", line 7 ORA-06512: at "SYS.DBMS_SQL", line
1721


ORA-20202: Gold seatnum B1 already booked ORA-06512: at
"SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T6", line 7 ORA-06512: at "SYS.DBMS_SQL", line
1721
```

4) Check that the seat numbers should not exceed specified capacity of a screen in a theatre.

--for silver seats

```
create trigger t7 before insert on silver_seat for each row
declare
g_count number:=0;
n screen.silver_seats%type;
begin
select silver_seats into n from screen where th_id=:new.th_id and screen_id=:new.screen_id;
select count(*) into g_count from silver_seat where sdatetime=:new.sdatetime and
th_id=:new.th_id and screen_id=:new.screen_id;
if g_count>=n
then
raise_application_error(-20202,'Maximum Silver seats capacity exceeded.');
end if;
end;
/
```

--for gold seats

```
create trigger t8 before insert on gold_seat for each row
declare
g_count number:=0;
n screen.gold_seats%type;
begin
select gold_seats into n from screen where th_id=:new.th_id and screen_id=:new.screen_id;
select count(*) into g_count from gold_seat where sdatetime=:new.sdatetime and th_id=:new.th_id
and screen_id=:new.screen_id;
if g_count>=n
then
raise_application_error(-20202,'Maximum gold seats capacity exceeded.');
end if;
end;
```

/

--**********

insert into silver_seat values('A8', 10, 100, to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'),1);

insert into gold_seat values('B9', 10, 100,to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'),1);

insert into gold_seat values('B10', 10, 100,to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'),1);

insert into gold_seat values('B10', 10, 100,to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'),1);

Output:

```
ORA-20202: Maximum Silver seats capacity exceeded. ORA-06512: at
"SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T7", line 9 ORA-06512: at "SYS.DBMS_SQL", line
1721


1 row(s) inserted.


1 row(s) inserted.


ORA-20202: Maximum gold seats capacity exceeded. ORA-06512: at
"SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T8", line 9 ORA-06512: at "SYS.DBMS_SQL", line
1721
```

5) Trigger to check that ticket id for a show cannot be same for silver and gold ticket.

```
create trigger t9 before insert on silverticket for each row
declare
g_count number:=0;
begin
select count(1) into g_count from goldticket where ticket_id=:new.ticket_id and
sdatetime=:new.sdatetime and th_id=:new.th_id and screen_id=:new.screen_id;
if g_count>0
then
raise_application_error(-20202,'Silver ticket id ane gold ticket id cannot be same for a movie show');
end if;
end;
/
```

```
create trigger t10 before insert on goldticket for each row
declare
g_count number:=0;
begin
select count(1) into g_count from silverticket where ticket_id=:new.ticket_id and
sdatetime=:new.sdatetime and th_id=:new.th_id and screen_id=:new.screen_id;
if g_count>0
then
raise_application_error(-20202,'Silver ticket id ane gold ticket id cannot be same for a movie show');
end if;
end;
/
```

```
--**************************
```

insert into goldticket values(3,100,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'), 10, 100,'B2',3);

insert into silverticket values(3,100,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'), 10, 100,'A2',1);

insert into goldticket values(1,100,sysdate,'online',to_date('20-Apr-2023 12:00:00 pm','dd-mon-yyyy hh:mi:ss pm'), 10, 100,'B3',3);

Output:

```
1 row(s) inserted.
```

```
 ORA-20202: Silver ticket id ane gold ticket id cannot be same for a movie
show ORA-06512: at "SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T9", line 7 ORA-06512: at
"SYS.DBMS_SQL", line 1721
```

```
ORA-20202: Silver ticket id ane gold ticket id cannot be same for a movie show
ORA-06512: at "SQL_OUKXPLQNLDYJARMZZXAZUFRDU.T10", line 7 ORA-06512: at
"SYS.DBMS_SQL", line 1721
```

6) Trigger to insert adshown data in adtemp which store data of all ads for a particular date,theatre and movie.

create or replace trigger AdRevenue

after insert or update

on AdsShown

for each row

declare

mname varchar2(100);

begin

select mov_name into mname from shows where th_id=:new.th_id and sdatetime = :new.sdatetime and screen_id = :new.screen_id;

insert into Adtemp values(:new.th_id,:new.sdatetime,:new.ad_id , mname);

```
end AdRevenue;
/

-------------------------------------------------------------------------------------------------------------------------------
declare
cnt number;
amount AdvertisementDetails.per_show_price%type;
adrev AdvertisementRevenue.revenue%type;

cursor c1 is select * from Adtemp;
r1 c1%rowtype;

begin
for r1 in c1
loop
exit when c1%notfound;

select per_show_price into amount from AdvertisementDetails where th_id = r1.th_id and ad_id = r1.ad_id and mov_name = r1.mov_name;

select count(sdatetime) into cnt from Adtemp where th_id = r1.th_id and ad_id = r1.ad_id and mov_name = r1.mov_name and sdatetime = r1.sdatetime group by th_id , ad_id , mov_name , sdatetime;

adrev := amount * cnt;

insert into AdvertisementRevenue values(r1.th_id , r1.sdatetime , r1.mov_name , r1.ad_id , adrev);

end loop;
end;
/
```

7) Trigger and function which adds ticket revenue, movie revenue and website revenue whenever a ticket is booked.

--for silver ticket

```
create or replace trigger bookedsilverticket

after insert or update

on silverticket

for each row

declare

mname varchar2(100);

mvshare number;

wbshare number;

mshare number;

wshare number;

ans boolean;

begin

select mov_name into mname from shows where th_id = :new.th_id and screen_id=:new.screen_id and sdatetime = :new.sdatetime;

select mov_share_in_per into mvshare from movie_share where mov_name = mname;

if :new.bk_type = 'online' then

select webshare_in_per into wbshare from websitedetails where th_id = :new.th_id and mov_name = mname and webname = :new.webname;

elsif :new.bk_type = 'offline' then

wbshare := 0;

end if;

mshare := (mvshare/100) * (:new.nprice);

wshare := (wbshare/100) * (:new.nprice);

ans := TicketRevCheck(:new.th_id , :new.sdatetime , mname , :new.nprice , mshare , wshare , :new.webname);

if ans then

dbms_output.put_line('Successful Ticket Data Entry');

else

dbms_output.put_line('Failed Ticket Data Entry...Please Keep Note of it');
```

```
end if;
end bookedsilverticket;
/

--for gold ticket
create or replace trigger bookedgoldticket
after insert or update
on goldticket
for each row
declare
mname varchar2(100);
mvshare number;
wbshare number;
mshare number;
wshare number;
ans boolean;
begin
select mov_name into mname from shows where th_id = :new.th_id and screen_id=:new.screen_id and sdatetime = :new.sdatetime;
select mov_share_in_per into mvshare from movie_share where mov_name = mname;

if :new.bk_type = 'online' then
select webshare_in_per into wbshare from websitedetails where th_id = :new.th_id and mov_name = mname and webname = :new.webname;
elsif :new.bk_type = 'offline' then
wbshare := 0;
end if;

mshare := (mvshare/100) * :new.nprice;
wshare := (wbshare/100) * :new.nprice;
```

```
ans := TicketRevCheck(:new.th_id , :new.sdatetime , mname , :new.nprice , mshare , wshare , :new.webname);


if ans then

dbms_output.put_line('Successful Ticket Data Entry');

else

dbms_output.put_line('Failed Ticket Data Entry...Please Keep Note of it');

end if;

end bookedgoldticket;

/


--function

create or replace function TicketRevCheck(thid in theatre.th_id%type , sdt in shows.sdatetime%type , mname in movie.mov_name%type , tprice in silver_price.price%type , movsh in number, wbsh in number , wbname in silverticket.webname%type) return boolean

is

trev TicketRevenue.ticketrevenue%type;

begin

trev := tprice - movsh - wbsh;

MERGE INTO TicketRevenue

USING (SELECT th_id,mov_name,revenue_date

     FROM  TicketRevenue) tr

ON (thid = tr.th_id and mname = tr.mov_name and sdt = tr.revenue_date)

WHEN matched THEN

  UPDATE SET ticketrevenue = ticketrevenue + trev

WHEN NOT matched THEN

  INSERT VALUES(thid,mname,sdt,trev);

MERGE INTO MovieRevenue

USING (SELECT th_id,mov_name,revenue_date

     FROM  MovieRevenue) mr

ON (thid = mr.th_id and mname = mr.mov_name and sdt = mr.revenue_date)

WHEN matched THEN
```

```
  UPDATE SET movierevenue = movierevenue + movsh
WHEN NOT matched THEN
  INSERT VALUES(thid,mname,sdt,movsh);
MERGE INTO WebRevenue
USING (SELECT th_id,mov_name,revenue_date,wname
      FROM  WebRevenue) wr
ON (thid = wr.th_id and mname = wr.mov_name and sdt = wr.revenue_date and wbname =
wr.wname)
WHEN matched THEN
  UPDATE SET webrevenue = webrevenue + wbsh
WHEN NOT matched THEN
  INSERT VALUES(thid,mname,sdt,wbname,wbsh);
return true;
end TicketRevCheck;
/
```

8) Trigger and function to add concession revenue whenever a concession is ordered.

```
create or replace trigger BookedConcessionData
after insert or update
on Concession
for each row
declare
qty number(10);
ans boolean;
begin
MERGE INTO ConcessionQuantity
USING (SELECT th_id,conc_name,revenuedate
      FROM  ConcessionQuantity) cq
```

```
ON (:new.th_id = cq.th_id and :new.conc_name = cq.conc_name and :new.orderdttime =
cq.revenuedate)
WHEN matched THEN
  UPDATE SET quantity = quantity + :new.quantity
    WHEN NOT matched THEN
  INSERT VALUES(:new.th_id,:new.conc_name,:new.orderdttime,:new.quantity);

ans := ConcessionRevCheck(:new.th_id ,:new.conc_name , :new.orderdttime , :new.quantity);

if ans then
dbms_output.put_line('Successful Booked Concession Data Entry');
else
dbms_output.put_line('Failed Booked Concession Data Entry...Please Keep Note of it');
end if;
end BookedConcessionData;
/

create or replace function ConcessionRevCheck (thid in number, concname in varchar2 , revdate in
date , qty in number) return boolean
is
p number(10);
amount number(10);
begin
select conc_price into p from ConcessionDetails where th_id=thid and conc_name = concname;
amount := p * qty;

MERGE INTO ConcessionRevenue
USING (SELECT th_id,conc_name,revenuedate
      FROM  ConcessionRevenue) cr
ON (thid = cr.th_id and concname = cr.conc_name and revdate = cr.revenuedate)
WHEN matched THEN
  UPDATE SET conc_rev = conc_rev + amount
```

```
WHEN NOT matched THEN
  INSERT VALUES(thid,concname,revdate,amount);

end ConcessionRevCheck;
/
```