create database xyz;

--- Tables ---

Create table parking\_lot (id numeric PRIMARY KEY, number\_of\_blocks numeric, address varchar(500), zip varchar(10), operating\_company\_na varchar(100));

Create table block\_(id numeric PRIMARY KEY , parking\_lot\_id numeric , block\_code varchar(3), number\_of\_floo numeric, CONSTRAINT fk\_block\_parking\_lot FOREIGN KEY (parking\_lot\_id) REFERENCES parking\_lot(id));

desc block\_ ;

Create table floor\_(id numeric PRIMARY KEY ,block\_id numeric,floor\_number numeric,number\_of\_wings numeric , CONSTRAINT fk\_floor\_block FOREIGN KEY (block\_id) REFERENCES block\_(id));

Create table parking\_slot(id numeric PRIMARY KEY,floor\_id numeric ,slot\_number numeric,wing\_code char(1), CONSTRAINT fk\_parking\_slot\_floor FOREIGN KEY (floor\_id) REFERENCES floor\_(id));

Create table customer(id numeric PRIMARY KEY,name varchar(10),vehical\_number varchar(20),registration\_date date,contact\_number numeric,cost numeric(10,2));

desc customer;

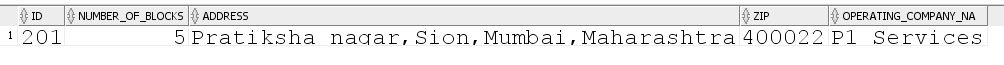
Create table parking\_slot\_reservation(id numeric PRIMARY KEY,customer\_id,start\_timestamp varchar(5),duration\_in\_minutes numeric,booking\_date date,parking\_slot\_id numeric, CONSTRAINT fk\_psr\_customer FOREIGN KEY (customer\_id) REFERENCES customer(id),CONSTRAINT fk\_psr\_parking\_slot FOREIGN KEY (parking\_slot\_id) REFERENCES parking\_slot(id));

Create table parking\_slip(id numeric PRIMARY KEY,parking\_slot\_reservatio numeric,actual\_entry\_time varchar(5),actual\_exit\_time varchar(5),basic\_cost numeric(10,2),penalty numeric(10,2),total\_cost numeric(10,2),CONSTRAINT fk\_pslip\_psr FOREIGN KEY (parking\_slot\_reservatio) REFERENCES parking\_slot\_reservation(id));

--- Insertion ---

insert into parking\_lot values(201,5,'Pratiksha nagar,Sion,Mumbai,Maharashtra','400022','P1 Services');

select \* from parking\_lot;



insert into block\_ values(01,201,'A-1',1);

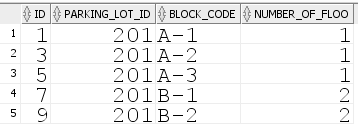
insert into block\_ values(03,201,'A-2',1);

insert into block\_ values(05,201,'A-3',1);

insert into block\_ values(07,201,'B-1',2);

insert into block\_ values(09,201,'B-2',2);

select \* from block\_;



insert into floor\_ values(101,01,1,3);

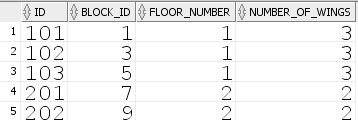
insert into floor\_ values(102,03,1,3);

insert into floor\_ values(103,05,1,3);

insert into floor\_ values(201,07,2,2);

insert into floor\_ values(202,09,2,2);

select \* from floor\_;



insert into parking\_slot values(01,101,01,'A');

insert into parking\_slot values(02,101,02,'A');

insert into parking\_slot values(03,102,03,'A');

insert into parking\_slot values(04,102,04,'A');

insert into parking\_slot values(05,103,05,'A');

insert into parking\_slot values(06,103,06,'A');

insert into parking\_slot values(07,201,07,'B');

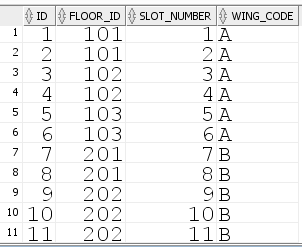
insert into parking\_slot values(08,201,08,'B');

insert into parking\_slot values(09,202,09,'B');

insert into parking\_slot values(10,202,10,'B');

insert into parking\_slot values(11,202,11,'B');

select \* from parking\_slot;



insert into customer values(01,'bhavya','MH100200','06/10/2022','932745372',100);

insert into customer values(02,'mansi','GJ100200','08/11/2022','9743834845',200);

insert into customer values(03,'ishwari','UP100200','06/09/2022','8948989998',100);

insert into customer values(04,'prachi','DL100200','10/10/2021','9848409858',100);

insert into customer values(05,'saumya','MH100321','03/10/2022','9893898349',200);

insert into customer values(06,'shreya','GJ100675','09/11/2022','9489483943',200);

insert into customer values(07,'muskan','UP100887','09/09/2022','7832883988',300);

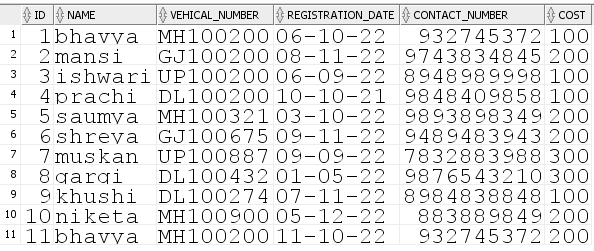
insert into customer values(08,'gargi','DL100432','01/05/2022','9876543210',300);

insert into customer values(09,'khushi','DL100274','07/11/2022','8984838848',100);

insert into customer values(10,'niketa','MH100900','05/12/2022','883889849',200);

insert into customer values(11,'bhavya','MH100200','11/10/2022','932745372',200);

select \* from customer;



insert into parking\_slot\_reservation values(201,01,'01:00',60,'06/10/2022',01);

insert into parking\_slot\_reservation values(202,02,'02:30',120,'08/11/2022',02);

insert into parking\_slot\_reservation values(203,03,'03:30',60,'06/09/2022',03);

insert into parking\_slot\_reservation values(204,04,'04:00',60,'10/10/2021',04);

insert into parking\_slot\_reservation values(205,05,'05:00',120,'03/10/2022',05);

insert into parking\_slot\_reservation values(206,06,'06:00',120,'09/11/2022',06);

insert into parking\_slot\_reservation values(207,07,'07:00',180,'09/09/2022',07);

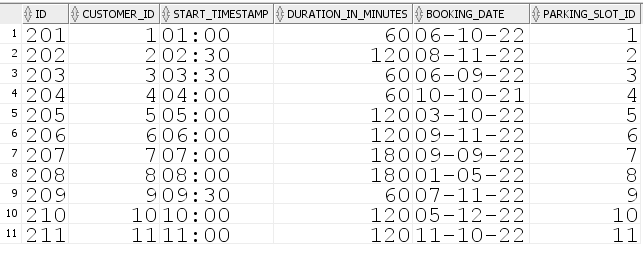
insert into parking\_slot\_reservation values(208,08,'08:00',180,'01/05/2022',08);

insert into parking\_slot\_reservation values(209,09,'09:30',60,'07/11/2022',09);

insert into parking\_slot\_reservation values(210,10,'10:00',120,'05/12/2022',10);

insert into parking\_slot\_reservation values(211,11,'11:00',120,'11/10/2022',11);

select \* from parking\_slot\_reservation;



insert into parking\_slip values(01,201,'01:00','02:00',100,NULL,100);

insert into parking\_slip values(02,202,'02:30','4:30',200,NULL,200);

insert into parking\_slip values(03,203,'3:30','4:30',100,NULL,100);

insert into parking\_slip values(04,204,'4:00','5:00',100,NULL,100);

insert into parking\_slip values(05,205,'5:00','7:00',200,NULL,200);

insert into parking\_slip values(06,206,'6:00','8:00',200,NULL,200);

insert into parking\_slip values(07,207,'7:00','10:00',300,NULL,300);

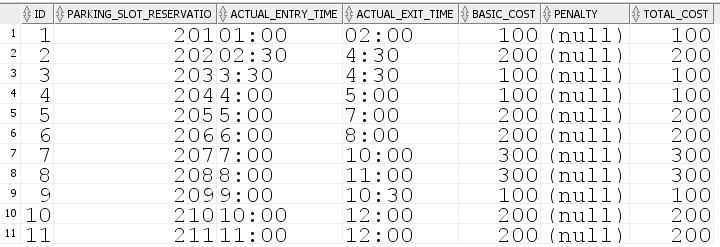
insert into parking\_slip values(08,208,'8:00','11:00',300,NULL,300);

insert into parking\_slip values(09,209,'9:00','10:30',100,NULL,100);

insert into parking\_slip values(10,210,'10:00','12:00',200,NULL,200);

insert into parking\_slip values(11,211,'11:00','12:00',200,NULL,200);

select \* from parking\_slip;



--- PL SQL Blocks ---

-- trigger calculting the total cost after adding penalty and placing the value in parking\_slip

create or replace trigger after\_insert\_parking\_slip

after insert on parking\_slip

for each row

declare

v\_total\_cost number(10,2);

begin

v\_total\_cost := :new.basic\_cost + coalesce(:new.penalty, 0);

update parking\_slip

set total\_cost = v\_total\_cost

where id = :new.id;

commit;

end;

/

DROP TRIGGER after\_insert\_parking\_slip;

-- raise alert for updating the registartion after 1 year

create or replace trigger b\_u\_registration\_date

before update of registration\_date on customer

for each row

begin

if months\_between(sysdate, :new.registration\_date) > 12 then

raise\_application\_error(-20002, 'Cannot update registration date for records older than a year');

end if;

end;

/

-- trigger on block table after updating the id of parking slot

create or replace trigger A\_U\_BLOCK

after update of PARKING\_LOT\_ID on BLOCK\_

for each row

declare

V\_OPERATING\_COMPANY\_NA PARKING\_LOT.OPERATING\_COMPANY\_NA%type;

begin

select OPERATING\_COMPANY\_NA into V\_OPERATING\_COMPANY\_NA

from PARKING\_LOT

where id = :new.PARKING\_LOT\_ID;

update PARKING\_LOT

set OPERATING\_COMPANY\_NA = V\_OPERATING\_COMPANY\_NA

where id = :new.PARKING\_LOT\_ID;

end;

/

-- trigger on parking\_lot before deleting te parking lot it check if the paraking slot is empty or not

create or replace trigger before\_delete\_parking\_lot

before delete on parking\_lot

for each row

declare

v\_count number;

begin

select count(\*)

into v\_count

from parking\_slot

where floor\_id in (select id from floor\_ where block\_id in (select id from block\_ where parking\_lot\_id = :old.id));

if v\_count > 0 then

raise\_application\_error(-20003, 'Cannot delete parking lot with associated parking slots');

end if;

end;

/

-- trigger in order to update the parking lot once a block is deleted.

create or replace trigger AFTER\_DELETE\_BLOCK

after delete on BLOCK\_

for each row

begin

update PARKING\_LOT

set NUMBER\_OF\_BLOCKS = NUMBER\_OF\_BLOCKS - 1

where id = :old.PARKING\_LOT\_ID;

end;

/

-- procedure

-- to check availability for car parking in parking slot

create or replace procedure check\_parking\_avl(parking\_lot\_id in number) as

v\_count number;

begin

select count(\*)

into v\_count

from parking\_slot

where floor\_id in (select id from floor\_ where block\_id in (select id from block\_ where parking\_lot\_id = parking\_lot\_id));

if v\_count > 0 then

raise\_application\_error(-20003, 'Cannot delete parking lot with associated parking slots');

end if;

end check\_parking\_avl;

/

set serveroutput on;

declare

v\_parking\_lot\_id number := 201;

begin

check\_parking\_avl(v\_parking\_lot\_id);

end;

/

-- function

--Get information of customer based on customer ID

create or replace function get\_customer\_details(customer\_id in number) return varchar2 is

v\_details varchar2(4000);

begin

select 'Customer ID: ' || to\_char(id) || chr(10) ||

'Name: ' || name || chr(10) ||

'Registration Date: ' || to\_char(registration\_date, 'DD/MM/YYYY') || chr(10) ||

'Contact Number: ' || to\_char(contact\_number) || chr(10) ||

'Cost: ' || to\_char(cost, 'FM999999990.00') into v\_details

from customer

where id = customer\_id;

return v\_details;

exception

when no\_data\_found then

return 'Customer with ID ' || to\_char(customer\_id) || ' not found.';

end get\_customer\_details;

/

set serveroutput on;

declare

customer\_id number := 1;

customer\_details varchar2(4000);

begin

customer\_details := get\_customer\_details(customer\_id);

dbms\_output.put\_line(customer\_details);

end;

/

-- calculate the total amount obtained from each year

select sum(cost) as total\_amount\_2022

from customer

where extract(year from registration\_date) = 2022;



-- which block has max no of reservations

select b.block\_code, count(psr.id) as reservation\_count

from block\_ b

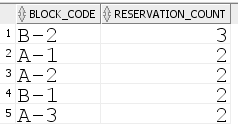
join floor\_ f on b.id = f.block\_id

join parking\_slot ps on f.id = ps.floor\_id

join parking\_slot\_reservation psr on ps.id = psr.parking\_slot\_id

group by b.block\_code

order by reservation\_count desc;



-- select a customer find the no of reservation

select c.name as customer\_name, count(psr.id) as reservation\_count

from customer c

left join parking\_slot\_reservation psr on c.id = psr.customer\_id

where c.name = 'bhavya'

group by c.name;

