IT214
DATABASE MANAGEMENT SYSTEM

PROJECT SUBMISSION

Database Title : Stored Procedures

SQL Queries

Team Details

NAME	ID
Unnathi Machiraju	201501140
Riya Talwar	201501154
Nikisha Patel	201501157
Brinda Jena	201501158

1. **Booking is done if there is availability of seats in any of the buses on the desired route and schedule.**

```
CREATE OR REPLACE FUNCTION BOOKING(deptime time without time zone,arrtime time
without time zone,dep_date date,ori varchar,dest varchar,ch integer,ad integer,bm char,tt
varchar(20)) RETURNS VOID AS $$
DECLARE
        sum int;
        id int;
        f int;
        j record;
        y record;
        flag int;
        x record;
        r int;
        s int;
        d int;
        sid int;
        nid int;
        c1 int;
        c2 int;
BEGIN
        sum=ch+ad;
        f :=0;
        flag :=0;
        c1 :=0;
        c2 :=0;
        select max(booking_id) into id from booking_details;
        nid=id+1;
        select route_id into d from route where origin=ori and destination=dest;
        select schedule_id into sid from bus_schedule where departure_time=deptime and
arrival_time=arrtime and departure_date=dep_date;
        for j in select bus_number,schedule_id,seats_available from "HAVING" where
schedule_id=sid and route_id=d
        LOOP
                if((j.seats_available > sum or j.seats_available=sum) and f=0) then
                        INSERT INTO booking_details
VALUES(nid,tt,bm,j.bus_number,sid,d,ch,ad);
                        nid=nid+1;
                        for x in select * from ALL_STOPS(j.bus_number,j.schedule_id)
                        loop
                                c1=c1+1;
                                c2=0;
                                if((x.ALL_STOPS=ori or flag=1 )and x.ALL_STOPS!=dest) then
                                        flag :=1;
```

```
                                        for y in select * from
ALL_STOPS(j.bus_number,j.schedule_id)
                                        loop
                                                c2=c2+1;
                                                if(c2>c1) then
                                                select route_id into r from route where
origin=x.ALL_STOPS and destination=y.ALL_STOPS;
                                                select bus_schedule.schedule_id into s from
"HAVING" join bus_schedule on("HAVING".schedule_id=bus_schedule.schedule_id)where
bus_number= j.bus_number and route_id=r and departure_date=dep_date;
                                                UPDATE "HAVING" set
seats_available=seats_available-sum where route_id=r and bus_number=j.bus_number and
schedule_id=s;
                                                end if;
                                        end loop;
                                end if;
                        end loop;
                        raise notice 'Booking done';
                        f :=1;
                end if;
        END LOOP;
        if(f=0) then
                raise notice 'Seats not available';
        end if;
        RETURN ;
END $$ language plpgsql;

select BOOKING('03:50:00','08:00:00','2017-11-10','Valsad','Vadodara',0,1,'A','Cash');
```

2. Display journey hours corresponding to the routes covered by various buses.

```
CREATE OR REPLACE FUNCTION JOURNEY_HOURS(ori varchar(20),dest varchar(20),bno
integer,d date) RETURNS bus_schedule.departure_time%type AS $$
DECLARE
        k time without time zone;
        ans time without time zone;
        dtime bus_schedule.departure_time%type;
        atime bus_schedule.arrival_time%type;
        ddate bus_schedule.departure_date%type;
        adate bus_schedule.arrival_date%type;
        diffdate INT;
BEGIN
        k := '24:00:00';
        ans :='00:00:00';
        select s.departure_time into dtime from route as r join  "HAVING" as h on ( h.route_id =
r.route_id) join bus_schedule as s on (s.schedule_id = h.schedule_id) where departure_date=d
AND origin=ori AND destination=dest AND bus_number=bno;
        select s.arrival_time into atime from route as r join  "HAVING" as h on ( h.route_id =
r.route_id) join bus_schedule as s on (s.schedule_id = h.schedule_id) where departure_date=d
AND origin=ori AND destination=dest AND bus_number=bno;
        select s.departure_date into ddate from route as r join  "HAVING" as h on ( h.route_id =
r.route_id) join bus_schedule as s on (s.schedule_id = h.schedule_id) where departure_date=d
AND origin=ori AND destination=dest AND bus_number=bno;
        select s.arrival_date into adate from route as r join  "HAVING" as h on ( h.route_id =
r.route_id) join bus_schedule as s on (s.schedule_id = h.schedule_id) where departure_date=d
AND origin=ori AND destination=dest AND bus_number=bno;
        select date_part('day',age(adate,ddate)) into diffdate;
        if(diffdate=0)then
                ans :=atime-dtime;
        elsif(atime>=dtime)then
                ans :=k + atime - dtime;
        elsif(atime<dtime)then
                ans :=k - dtime + atime;
        END if;
        RETURN ans;
END $$ LANGUAGE plpgsql;
select * from journey_hours('Rajkot','Surat',6,'2017-11-10');
```

3. If user enters origin and destination, the date he wishes to travel on and also the number of adults and children whose booking he wants to make. Display the

corresponding total fare for all the bus schedules available on that date on that route so that the user can decide which schedule he should select.

```
CREATE OR REPLACE FUNCTION CALCULATE_FARE(ori varchar(20),dest varchar(20),d
date,no_of_adult integer,no_of_child integer)RETURNS SETOF record AS $$
DECLARE
	cf INT;
	af INT;
	totalfare INT;
	k record;
	mytable rec%ROWTYPE;

BEGIN
	FOR k IN select f.bus_number,s.schedule_id,r.origin,r.destination,child_fare,adult_fare
from route as r join  "HAVING" as h on ( h.route_id = r.route_id) join has_fare_for as f on
(h.bus_number = f.bus_number AND h.route_id = f.route_id) join bus_schedule as s on
(s.schedule_id = h.schedule_id) where departure_date=d AND origin=ori AND destination=dest
AND seats_available>=no_of_child+no_of_adult
	LOOP
		totalfare := (k.child_fare*no_of_child) + (k.adult_fare*no_of_adult);
		mytable.schedule_id :=k.schedule_id;
		mytable.bus_number :=k.bus_number;
		mytable.origin :=k.origin;
		mytable.destination :=k.destination;
		mytable.total_fare :=totalfare;
		RETURN NEXT mytable;
	END LOOP;
	RETURN;
END $$ LANGUAGE plpgsql;

SELECT *from CALCULATE_FARE('Ahmedabad','Rajkot','2017-11-10',2,1) AS ans(schedule_id
integer,bus_number integer,origin varchar(20),destination varchar(20),total_fare integer);
```

4. Display the bus number which covers maximum cities

```
CREATE OR REPLACE FUNCTION BUS_HAVING_MAX_STOPS()RETURNS SETOF integer
AS $$
DECLARE
        j INT;
        k record;
        maximum INT;
        busno INT;
        ans integer;
BEGIN
        maximum=0;
        FOR j IN select count(city) from(select bus_number,origin as city from "HAVING" natural
join route union select bus_number,destination as city from "HAVING" natural join route)as d
group by bus_number
        LOOP
                if(j>maximum) then
                        maximum := j;

                END if;
        END LOOP;
        FOR k IN select bus_number,count(city) as stops from(select bus_number,origin as city
from "HAVING" natural join route union select bus_number,destination as city from "HAVING"
natural join route)as d group by bus_number
        LOOP
                if(k.stops=maximum)then
                      ans := k.bus_number;
                         return next ans;
                end if;
        end loop;
        return ;
END $$ LANGUAGE plpgsql;

SELECT *from BUS_HAVING_MAX_STOPS();
```

5. Given a bus number and departure date it displays all the stops of that bus from the actual origin and destination in sequence

```
CREATE OR REPLACE FUNCTION ALL_STOPS(bno integer,sid integer) RETURNS SETOF
varchar AS $$
DECLARE
        j record;
        min time without time zone;
        strt varchar(20);
        k varchar(20);
        d date;
BEGIN
        min := '23:23:59';
        select departure_date into d from bus_schedule where schedule_id=sid;
        for j IN select departure_time,arrival_time,origin,destination from "HAVING" natural join
route natural join bus_schedule where bus_number=bno and departure_date=d order by
arrival_time
        LOOP
                if(j.departure_time < min)then
                        min=j.departure_time;
                        strt=j.origin;
                end if;
        END LOOP;
        return next strt;
        for k in select destination from (select distinct destination,arrival_time from "HAVING"
natural join route natural join bus_schedule where bus_number=bno and departure_date=d
order by arrival_time) as a
        loop
         return next k;
        end loop;
      return;
END $$ LANGUAGE plpgsql;

select * from ALL_STOPS(2,209);
```