```
use MIST353VehicleServiceKhalfani;

if (OBJECT_ID('procAnalysisOfCarWithProblems') is not null)
    drop procedure procAnalysisOfCarWithProblems;

if (OBJECT_ID('procFindServiceAvailability') is not null)
    drop procedure procFindServiceAvailability;

if (OBJECT_ID('procAddTimeSlotToAppointment') is not null)
    drop procedure procAddTimeSlotToAppointment;

if (OBJECT_ID('procFindServiceAvailability2') is not null)
    drop procedure procFindServiceAvailability2;

if (OBJECT_ID('procRemoveAppointment') is not null)
    drop procedure procRemoveAppointment;

if (OBJECT_ID('procMakeAppointment') is not null)
drop procedure procMakeAppointment;

if (OBJECT_ID('trgIncreaseAppointmentSlotAvailability') is not null)
drop trigger trgIncreaseAppointmentSlotAvailability;


go

create procedure procAnalysisOfCarWithProblems
(
    @startDate date,
    @endDate date,
    @numberofTimeServiced int
)
as
begin
    select V.vehicleId, C.CustomerName

    from Appointment A join Vehicle V
    on A.VehicleId = V.VehicleId
    join Customer C
    on C.CustomerId = V.CustomerId
    where AppointmentId in

    (select distinct ATP.AppointmentId
    from AppointmentSlot APS join
    AppointmentTimePeriod ATP
    on APS.AppointmentSlotId = ATP.AppointmentSlotId

    where APS.AppointmentDate between @startDate and @endDate)

    group by V.VehicleId, C.CustomerName
    having count(V.vehicleId) >= @numberofTimeServiced
```

```
end

/*
execute procAnalysisOfCarWithProblems
@startDate = '10/1/2016',
    @endDate = '10/31/2016',
    @numberofTimeServiced = 2;


select * from AppointmentSlot;
select * from Appointment;
select * from AppointmentTimePeriod

*/

go

create procedure procFindServiceAvailability
(
    @StartDate date,
    @EndDate date
)
as
begin
    select AppointmentDate, AppointmentStartTime
    from AppointmentSlot
    where SlotsLeft > 0
    and AppointmentDate between @StartDate and @EndDate;
end

go

create procedure MakeTimeSlot
(
    @appointmentId int,
    @appointmentSlotId int
)
as
begin

insert AppointmentTimePeriod
(AppointmentId, AppointmentSlotId)
values
(@appointmentId, @appointmentSlotId)

end

go

create procedure procAddTimeSlotToAppointment
(
    @appointmentId int,
    @appointmentSlotId int,
    @userFriendlyErrorMessage varchar(100) output
)
as
begin

    begin try
    insert into AppointmentTimePeriod
    (AppointmentId, AppointmentSlotId)
    values
    (@appointmentId, @appointmentSlotId)
    set @userFriendlyErrorMessage = 'Successful';
    end try
    begin catch
        set @appointmentId = -1;
        if (ERROR_MESSAGE() like '%chValidSlot%')
        set @userFriendlyErrorMessage = 'No Appointment Available';
        end catch
    end
    /*
    declare @userFriendlyErrorMessage varchar(100);
    execute procAddTimeSlotToAppointment
    @appointmentId = 1009,
    @appointmentSlotId = 101,
    @userFriendlyErrorMessage = @userFriendlyErrorMessage output;
    print @userFriendlyErrorMessage;
    */
go

create procedure procFindServiceAvailability2
(
    @StartDate date = null,
    @EndDate date = null
)
as
begin

    if (@StartDate is null)
    set @StartDate = GETDATE();

    select AppointmentSlotId, AppointmentDate, AppointmentStartTime
    from AppointmentSlot
    where SlotsLeft > 0
    and AppointmentDate between @StartDate and  IsNull(@EndDate, AppointmentDate );
end
/*
execute procFindServiceAvailability2
    --input parameters
    @StartDate date = '10-10-2016',
    @EndDate date = '10-14-2016';
*/
```

```sql
go

create procedure procRemoveAppointment
(
    @appointmentId int,
    @userFriendlyErrorMessage varchar(100)
)
as
begin

    begin try
    delete from Appointment
    where AppointmentId = @appointmentId
    set @userFriendlyErrorMessage = 'Successful';
    end try
    begin catch
        set @appointmentId = -1;
        if (ERROR_MESSAGE () like '%AppointmentId%')
            set @userFriendlyErrorMessage = 'Appointment ID is invalid';
        end catch


end

go

create procedure procMakeAppointment
(
    @vehicleId int,
    @userFriendlyErrorMessage varchar(100) output
)
as
begin

    declare @appointmentId int

    --error catching

    begin try
        insert into Appointment
        (BookedDate, VehicleId)
        values
        (getDate(),@vehicleId)

        set @appointmentId =
            IDENT_CURRENT('Appointment')
        set @userFriendlyErrorMessage = 'Succesful';

    end try

    begin catch
        set @appointmentId = -1;
        if (ERROR_MESSAGE () like '%fkAppointmentToVehicle%')
        set @userFriendlyErrorMessage = 'vehicle Id Invalid';

    end catch

    return @appointmentId;
end

/*
delcare @appointmentId int, @userFriendlyErrorMessage varchar (100);
execute @appointmentId = procMakeAppointment
    @vehicleId = 16,
    @userFriendlyErrorMessage = @userFriendlyErrorMessage out;
print @appointmentId;
print @userFriendlyErrorMessage
*/

go

create trigger trgDecreaseAppointmentSlotAvailability
on AppointmentTimePeriod
after delete
as
begin


    update AppointmentSlot
    set SlotsLeft = SlotsLeft - 1
    from AppointmentSlot A join deleted D
    on A.AppointmentSlotId = D.AppointmentSlotId;

end;

go

create trigger trIncreaseAppointmentSlotAvailability
on AppointmentTimePeriod
after delete
as
begin
    declare @appointmentSlotId int;

    select @appointmentSlotId = AppointmentSlotId
    from deleted;

    update AppointmentSlot
    set SlotsLeft = SlotsLeft + 1
    where AppointmentSlotId = @appointmentSlotId;

end;
```