```sql
USE SpeedingTickets;
GO

-- 1. Display a list of all tickets submitted within a given month of the current year.
-- The month will be supplied to the routine as a word (e.g. April)
DROP PROCEDURE usp_GetTicketsbyMonth;
GO

CREATE PROCEDURE usp_GetTicketsbyMonth
    @month VARCHAR(10)
AS
BEGIN

    SELECT * FROM Ticket WHERE FORMAT(Date, 'MMMM') = @month AND year(Date)= year(getdate());
END;

EXEC dbo.usp_GetTicketsbyMonth March;


--2.Display a list of the top 10 tickets that have had the most activity in the form
--of comments. A start date and end date will be supplied to the routine in the
--form ‘yyyy-mm-dd’.


-- Adding extra information to the table 'Comment'
INSERT INTO Comment (TicketID, TechnicianID, UserID, Comment) VALUES
    (4,2,4,'Comment 1'), (4,2,4,'Comment 2'),  (4,2,4,'Comment 3'),  (6,4,6,'Comment 1'), (6,4,6,'Comment 2'), (6,4,6,'Comment 3'),
    (6,4,6,'Comment 4'), (7,5,3,'Comment 1'),  (7,5,3,'Comment 2'),  (8,5,20,'Comment 1'), (8,5,20,'Comment 2'),(8,5,20,'Comment 3'),
    (9,2,4,'Comment 1'), (9,2,4,'Comment 2'), (9,2,4,'Comment 3'), (9,2,4,'Comment 4'), (10,1,1,'Comment 1'), (10,1,1,'Comment 2'),
    (10,1,1,'Comment 3'),  (11,2,3,'Comment 1'),   (11,2,3,'Comment 2'), (11,2,3,'Comment 3'), (11,2,3,'Comment 4'), (11,2,3,'Comment 5'),
    (11,2,3,'Comment 6'),  (17,4,6,'Comment 1'),  (17,4,6,'Comment 2');
```

```sql


CREATE PROCEDURE usp_Top10Tickets
    @startDate DATE, @endDate DATE
AS
BEGIN
    SELECT TOP 10 C.TicketID, count(C.TicketID) AS numberOfTickets
        FROM Comment C
        INNER JOIN Ticket T
        ON C.TicketID = T.TicketID
        WHERE Date BETWEEN @startDate AND @endDate
        GROUP BY C.TicketID
        ORDER BY numberOfTickets
        DESC ;
END;

EXEC dbo.usp_Top10Tickets '2020-03-01','2020-04-10';
GO


--3. Display a list of tickets for a particular category, ordered by descending date,
--with the corresponding date displayed in the format ( Month dddd, yyyy) e.g.
--November 21st, 2009. A category name will be supplied to the routine.

DROP PROCEDURE usp_GetTicketsByCategory;
GO

CREATE PROCEDURE usp_GetTicketsByCategory
    @categoryName VARCHAR(10)

AS
BEGIN

    DECLARE @dateToDisplay DATETIME;
    DECLARE @ticketNumber INT = 0;
    DECLARE @ticketToDisplay INT;
    DECLARE @toDisplay VARCHAR(6) = 'th';


    SELECT TOP 1 @dateToDisplay = T.Date, @ticketToDisplay = T.TicketID,
        @ticketNumber = T.TicketID
        FROM Ticket T
        RIGHT JOIN Category C
        ON T.CategoryID = C.CategoryID
        WHERE C.Name = @categoryName AND T.TicketID > @ticketNumber
        ORDER BY T.TicketID, T.Date
        DESC ;

    WHILE @@ROWCOUNT > 0
        BEGIN
                IF day(@dateToDisplay) = '01' OR day(@dateToDisplay) = '21' OR day(@dateToDisplay) = '31'
                    BEGIN
                            SET @toDisplay = 'st';

                    END

                ELSE
                    BEGIN
                                IF day(@dateToDisplay) = '02' OR day(@dateToDisplay) = '22'
                                        BEGIN
                                                 SET @toDisplay = 'nd';

                                         END
                                ELSE
                                        IF day(@dateToDisplay) = '03' OR day(@dateToDisplay) = '23'
                                        BEGIN
                                                 SET @toDisplay = 'rd';

                                         END

                                        ELSE
                                            BEGIN
                                                    SET @toDisplay = 'th';
                                            END


                    END

                PRINT 'Ticket #: ' + cast(@ticketNumber AS VARCHAR)  + ' ------> The issued date is: ' + format(@
dateToDisplay,'MMMM') + ' '+ cast(day(@dateToDisplay) AS VARCHAR)  + @toDisplay + ', '+ cast(year(@dateToDisplay) AS VARCHAR);

            SELECT TOP 1 @dateToDisplay = T.Date, @ticketToDisplay = T.TicketID,
            @ticketNumber = T.TicketID
            FROM Ticket T
            RIGHT JOIN Category C
            ON T.CategoryID = C.CategoryID
            WHERE C.Name = @categoryName AND T.TicketID > @ticketNumber
            ORDER BY T.TicketID, T.Date
            DESC ;

        END
END;

EXEC dbo.usp_GetTicketsByCategory 'Hardware'; -- The categories are: Hardware, Software, Phone, Internet, Other
GO


--4. Return the total number of active tasks for a given support staff member. An
--employee number will be supplied to the routine. The routine should also
--return zero (0), if there are no active tasks for that person, and negative one
--(-1), if the support staff member could not be found.

INSERT INTO ActionsToDoInFuture (TicketID, Action) VALUES
    (2,'Task 1'),(2,'Task 2'), (2,'Task 3'),(2,'Task 4'),(4,'Task 1'),(4,'Task 2'),(4,'Task 3'),(3,'Task 1'), (3,'Task 2'),
    (5,'Task 1'),(5,'Task 2'), (12,'Task 1'),(12,'Task 2'),(12,'Task 3'),(12,'Task 4'),(6,'Task 1'),(6,'Task 2'), (6,'Task 3'),
     (17,'Task 1'),(17,'Task 2'), (18,'Task 1'),(18,'Task 2'),(7,'Task 1'),(7,'Task 2'),(7,'Task 3'),(19,'Task 1'), (19,'Task 2');



DROP FUNCTION ufn_GetTasksForTechnician;
GO
```

```sql

CREATE FUNCTION ufn_GetTasksForTechnician (@technicianID INT)
    RETURNS SMALLINT
AS
BEGIN
            DECLARE @numberOfTasks SMALLINT;
            DECLARE @technicianControl SMALLINT;

            SET @technicianControl = (SELECT TOP 1 count(TechnicianInfoID) FROM TechnicianInfo
                WHERE TechnicianInfoID = @technicianID
                GROUP BY TechnicianInfoID);

            SET @numberOfTasks = (SELECT TOP 1 sum(count(T.TicketID)) OVER ( ) AS total_count FROM Ticket T
                RIGHT JOIN ActionsToDoInFuture A
                ON T.TicketID = A.TicketID
                WHERE T.IsSolved = 'N' AND T.AssignTechnicianID = @technicianID
                GROUP BY T.TicketID);

            IF @technicianControl IS NULL
                BEGIN
                    SET @numberOfTasks = '-1';
                END

            IF @numberOfTasks IS NULL AND @technicianControl = '1'
                BEGIN
                    SET @numberOfTasks = '0';
                END

            RETURN @numberOfTasks;
END;
GO

DECLARE @result SMALLINT;
EXEC @result = ufn_GetTasksForTechnician_2 '3';
-- We have in our database TechnicianID`s: 1, 2, 4 and 5

PRINT 'The number of tasks are: ' + cast(@result AS VARCHAR);
GO


```

```sql
--5. Display a “page” of ticket information by passing, to the routine, a page
--number and the number of tickets per page. For example, passing “1,10” will
--return the first ten tickets (ordered by ticket id), but passing “2,10” will
--return next ten tickets (i.e. page 2).

DROP PROCEDURE usp_GetpageInfo;
GO

CREATE PROCEDURE usp_GetpageInfo
    @page SMALLINT, @ticketsPerPage SMALLINT
AS
BEGIN
    DECLARE @start SMALLINT;
    DECLARE @end SMALLINT;
    SET @end = (@page * @ticketsPerPage);
    SET @start = @end - @ticketsPerPage;

        SELECT * FROM (SELECT  row_number() OVER (ORDER BY TicketID DESC ) AS Row#, * FROM Ticket)
    table_2 WHERE Row# > @start  AND Row# < (@end + 1) ;
END;
GO

-- As an example I`m displaying the page #3 (with 5 rows on each page).
-- In this case the rows #11 to #15 should be displayed.
EXEC dbo.usp_GetpageInfo 3,5;
GO



```