```sql
/* Assignment 4
   Author: Cheolsoon Im
   Student number: w0439657*/

/* create and use TicketDatabase (drop if necessary) */
USE master;

GO

DROP DATABASE IF EXISTS TicketDatabase;

CREATE DATABASE TicketDatabase;

GO

USE TicketDatabase;

GO

/* Create 9 tables (drop if necessary)*/


DROP TABLE IF EXISTS Communication;

CREATE TABLE Communication
(
    CommunicationID   CHAR(12)     NOT NULL,
    Description       VARCHAR(256) NOT NULL,
    TicketID          CHAR(11)     NOT NULL,
    UserID            CHAR(6)     NULL,
    RequestingStaffID CHAR(7)     NOT NULL,
    RespondingStaffID CHAR(7)     NULL,
    PRIMARY KEY (CommunicationID)
);


DROP TABLE IF EXISTS Hardware;

CREATE TABLE Hardware
(
    HardwareID     CHAR(9)     NOT NULL,
    Description    VARCHAR(80) NOT NULL,
    Warranty       TINYINT      NOT NULL,
    DateOfPurchase DATE         NOT NULL,
    Make           VARCHAR(32)  NOT NULL,
    Model          VARCHAR(32)  NOT NULL,
```

```
47      SerialNumber   VARCHAR(32)  NOT NULL,
48      PRIMARY KEY (HardwareID)
49  );
50
51
52  DROP TABLE IF EXISTS Staff;
53
54  CREATE TABLE Staff
55  (
56      StaffID   CHAR(7)    NOT NULL,
57      FirstName VARCHAR(32) NOT NULL,
58      LastName  VARCHAR(32) NOT NULL,
59      PRIMARY KEY (StaffID)
60  );
61
62
63  DROP TABLE IF EXISTS Status;
64
65  CREATE TABLE Status
66  (
67      StatusID    CHAR(2)    NOT NULL,
68      Description VARCHAR(80) NOT NULL,
69      PRIMARY KEY (StatusID)
70  );
71
72
73  DROP TABLE IF EXISTS Task;
74
75  CREATE TABLE Task
76  (
77      TaskID      CHAR(10)     NOT NULL,
78      Description VARCHAR(256) NOT NULL,
79      TicketID    CHAR(11)     NOT NULL,
80      StaffID     CHAR(7)     NULL,
81      StatusID    CHAR(2)     NOT NULL,
82      PRIMARY KEY (TaskID)
83  );
84
85
86  DROP TABLE IF EXISTS Ticket;
87
88  CREATE TABLE Ticket
89  (
90      TicketID        CHAR(11)     NOT NULL,
91      Description     VARCHAR(256) NOT NULL,
92      WhetherHardware TINYINT      NOT NULL,
```

```
 93     DateGenerated   DATETIME2(7)    NOT NULL,
 94     UserID          CHAR(6)      NULL,
 95     StaffID         CHAR(7)      NULL,
 96     AssignedStaffID CHAR(7)      NULL,
 97     CategoryID      CHAR(5)      NOT NULL,
 98     PRIMARY KEY (TicketID)
 99 );
100
101
102 DROP TABLE IF EXISTS Category;
103
104 CREATE TABLE Category
105 (
106     CategoryID  CHAR(5)    NOT NULL,
107     Description VARCHAR(80) NOT NULL,
108     PRIMARY KEY (CategoryID)
109 );
110
111
112 DROP TABLE IF EXISTS [User]
113
114 CREATE TABLE [User]
115 (
116     UserID    CHAR(6)    NOT NULL,
117     FirstName VARCHAR(32) NOT NULL,
118     LastName  VARCHAR(32) NOT NULL,
119     PRIMARY KEY (UserID)
120 );
121
122
123 DROP TABLE IF EXISTS HardwareTicket
124
125 CREATE TABLE HardwareTicket
126 (
127     HardwareID  CHAR(9)   NOT NULL,
128     TicketID    CHAR(11)   NOT NULL,
129     PRIMARY KEY (HardwareID,TicketID)
130 );
131
132
133 /* Create 14 foreign keys */
134 ALTER TABLE Task
135     DROP CONSTRAINT IF EXISTS FK__Task__TicketID;
136 ALTER TABLE Task
137     ADD CONSTRAINT FK__Task__TicketID FOREIGN KEY (
    TicketID) REFERENCES Ticket (TicketID);
```

```
138
139 ALTER TABLE Communication
140     DROP CONSTRAINT IF EXISTS FK__Communication__TicketID;
141 ALTER TABLE Communication
142     ADD CONSTRAINT FK__Communication__TicketID FOREIGN KEY
     (TicketID) REFERENCES Ticket (TicketID);
143
144 ALTER TABLE Ticket
145     DROP CONSTRAINT IF EXISTS FK__Ticket__CategoryID;
146 ALTER TABLE Ticket
147     ADD CONSTRAINT FK__Ticket__CategoryID FOREIGN KEY (
    CategoryID) REFERENCES Category (CategoryID);
148
149 ALTER TABLE Task
150     DROP CONSTRAINT IF EXISTS FK__Task__StatusID
151 ALTER TABLE Task
152     ADD CONSTRAINT FK__Task__StatusID FOREIGN KEY (
    StatusID) REFERENCES Status (StatusID);
153
154 ALTER TABLE Ticket
155     DROP CONSTRAINT IF EXISTS FK__Ticket__UserID;
156 ALTER TABLE Ticket
157     ADD CONSTRAINT FK__Ticket__UserID FOREIGN KEY (UserID
    ) REFERENCES [User] (UserID)
158     ON DELETE SET NULL ON UPDATE SET NULL;
159
160 ALTER TABLE Ticket
161     DROP CONSTRAINT IF EXISTS FK__Ticket__StaffID;
162 ALTER TABLE Ticket
163     ADD CONSTRAINT FK__Ticket__StaffID FOREIGN KEY (
    StaffID) REFERENCES Staff (StaffID)
164     ON DELETE SET NULL ON UPDATE SET NULL;
165
166 ALTER TABLE Ticket
167     DROP CONSTRAINT IF EXISTS FK__Ticket__AssignedStaffID;
168 ALTER TABLE Ticket
169     ADD CONSTRAINT FK__Ticket__AssignedStaffID FOREIGN KEY
     (AssignedStaffID) REFERENCES Staff (StaffID)
170
171 ALTER TABLE Task
172     DROP CONSTRAINT IF EXISTS FK__Task__StaffID;
173 ALTER TABLE Task
174     ADD CONSTRAINT FK__Task__StaffID FOREIGN KEY (StaffID
    ) REFERENCES Staff (StaffID);
175
176 ALTER TABLE Communication
```

```
177     DROP CONSTRAINT IF EXISTS
    FK__Communication__RequestingStaffID;
178 ALTER TABLE Communication
179     ADD CONSTRAINT FK__Communication__RequestingStaffID
    FOREIGN KEY (RequestingStaffID) REFERENCES Staff (StaffID
    );
180
181 ALTER TABLE Communication
182     DROP CONSTRAINT IF EXISTS
    FK__Communication__RespondingStaffID;
183 ALTER TABLE Communication
184     ADD CONSTRAINT FK__Communication__RespondingStaffID
    FOREIGN KEY (RespondingStaffID) REFERENCES Staff (StaffID
    );
185
186 ALTER TABLE Communication
187     DROP CONSTRAINT IF EXISTS FK__Communication__UserID;
188 ALTER TABLE Communication
189     ADD CONSTRAINT FK__Communication__UserID FOREIGN KEY (
    UserID) REFERENCES [User] (UserID)
190     ON DELETE SET NULL ON UPDATE SET NULL;
191
192 ALTER TABLE HardwareTicket
193     DROP CONSTRAINT IF EXISTS FK__HardwareTicket__TicketID
    ;
194 ALTER TABLE HardwareTicket
195     ADD CONSTRAINT FK__HardwareTicket__TicketID FOREIGN
    KEY (TicketID) REFERENCES Ticket (TicketID)
196     ON DELETE CASCADE ON UPDATE CASCADE ;
197
198 ALTER TABLE HardwareTicket
199     DROP CONSTRAINT IF EXISTS
    FK__HardwareTicket__HardwareID;
200 ALTER TABLE HardwareTicket
201     ADD CONSTRAINT FK__HardwareTicket__HardwareID FOREIGN
    KEY (HardwareID) REFERENCES Hardware (HardwareID);
202
203 /*Add check constraints*/
204 ALTER TABLE Ticket
205     ADD CONSTRAINT CK__Ticket__WhetherHardware CHECK (
    WhetherHardware = 0 OR WhetherHardware = 1);
206
207 ALTER TABLE Hardware
208     ADD CONSTRAINT CK__Hardware__Warranty CHECK (Warranty
     = 0 OR Warranty = 1);
209
```

```
210 ALTER TABLE Hardware
211     ADD CONSTRAINT CK__Hardware__Make CHECK (len(Make) > 0
    );
212
213 ALTER TABLE Hardware
214     ADD CONSTRAINT CK__Hardware_DateOfPurchase CHECK (
    DateOfPurchase < GETDATE());
215
216 /*Add default constraints*/
217 ALTER TABLE [User]
218     ADD CONSTRAINT DF__User__FirstName DEFAULT ('') FOR
    FirstName;
219
220 ALTER TABLE [User]
221     ADD CONSTRAINT DF__User__LastName DEFAULT ('') FOR
    Lastname;
222
223 ALTER TABLE Staff
224     ADD CONSTRAINT DF__Staff__FirstName DEFAULT ('') FOR
    FirstName;
225
226 ALTER TABLE Staff
227     ADD CONSTRAINT DF__Staff__LastName DEFAULT ('') FOR
    Lastname;
228
229 /* Add unique constraints serial number */
230 ALTER TABLE Hardware
231     ADD CONSTRAINT AK__Hardware__SerialNumber UNIQUE (
    SerialNumber);
232
233 /* Add appropriate indexes to non-keys used for searching
    records */
234 CREATE INDEX IX__User__FirstNameLastName
235     ON [User] (FirstName, LastName);
236
237 CREATE INDEX IX__Staff__FirstNameLastName
238     ON Staff (FirstName, LastName);
239
240 GO
241
242 INSERT INTO Category(CategoryID, Description )
243 VALUES ('c0000','Hardware'),
244        ('c0001','Software'),
245        ('c0002','Network'),
246        ('c0003','Other'),
247        ('c0004','Unknown');
```

```
248
249 GO
250
251 INSERT INTO Hardware(HardwareID, Description, Warranty,
252                        DateOfPurchase, Make, Model,
    SerialNumber)
253 VALUES ('h00000000','CPU', 1, '2019-01-01' ,'Samsung','XA'
    ,'XA1121'),
254        ('h00000001','Hard Drive',1,'2020-01-01','Intel','
    KT','KT2020'),
255        ('h00000002','Power Supply',0,'2015-01-01','ASUS','
    UP','UP2323'),
256        ('h00000003','Moniter',0,'2019-01-01','LG','RS','
    RS2322'),
257        ('h00000004','Cooler',1,'2011-01-01','Dell','DD','
    DD2898');
258
259 GO
260
261 INSERT INTO Staff(StaffID, FirstName, LastName)
262 VALUES ('s000000','Abbey','Hacker'),
263        ('s000001','Brown','Hacker'),
264        ('s000002','Cathy','Hacker'),
265        ('s000003','Donald','Hacker'),
266        ('s000004','Eric','Hacker'),
267        ('s000005','Foster','Hacker'),
268        ('s000006','Gurleen','Hacker'),
269        ('s000007','Herold','Hacker'),
270        ('s000008','Ingrid','Hacker'),
271        ('s000009','Jonah','Hacker');
272
273 GO
274
275 INSERT INTO Status(StatusID, Description)
276 VALUES ('s0','Proposed'),
277        ('s1','Transferred to related department'),
278        ('s2','Action approved'),
279        ('s3','In progress'),
280        ('s4','Done');
281
282 GO
283
284 INSERT INTO [User](UserID, FirstName, LastName)
285 VALUES ('u0000','Jane','Smith'),
286        ('u0001','Jack','Bauer'),
287        ('u0002','Cheolsoon','Im'),
```

```
288         ('u0003','Kevin','Im'),
289         ('u0004','Eunice','Im'),
290         ('u0005','Jack','Welch'),
291         ('u0006','Tom','Scott'),
292         ('u0007','Rick','Boose'),
293         ('u0008','Robert','Jones'),
294         ('u0009','Steve','Kaufman');
295
296 GO
297
298 INSERT INTO Ticket(TicketID, Description, DateGenerated,
    WhetherHardware,
299                     UserID, StaffID, AssignedStaffID,
    CategoryID)
300 VALUES
301     ('t0000000000','Software freezes', '2020-02-22 16:30:
    00' ,0, NULL, 's000000','s000003','c0001'),
302     ('t0000000001','Cannot install new software.', '2020-
    01-30 16:30:00' ,0, NULL, 's000000','s000004','c0001'),
303     ('t0000000002','Cannot find printers', '2020-01-30 16:
    00:00' ,0, NULL, 's000001','s000004','c0002'),
304     ('t0000000003','Hardware- cannot identify hard drive',
    '2020-04-30 16:30:00' , 1, NULL, 's000002','s000004','
    c0000'),
305     ('t0000000004','Hardware','2020-03-01 16:30:00' , 1, '
    u0000', NULL, NULL,'c0000'),
306     ('t0000000005','Hardware','2020-03-03 16:30:00' , 1, '
    u0000', NULL, 's000005', 'c0000'),
307     ('t0000000006','Hardware','2020-03-02 12:30:00' , 1, '
    u0000', NULL, NULL,'c0000'),
308     ('t0000000007','Hardware','2020-03-20 16:30:00' , 1, '
    u0001', NULL, 's000005', 'c0000'),
309     ('t0000000008','Hardware','2020-01-30 13:30:00' , 1, '
    u0002', 's000003', NULL, 'c0003'),
310     ('t0000000009','Hardware','2020-01-30 16:30:00' , 1, '
    u0003', 's000005', NULL, 'c0003'),
311     ('t0000000010','Hardware','2020-05-03 16:30:00' , 1, '
    u0000', NULL, 's000005', 'c0000'),
312     ('t0000000011','Hardware','2020-05-02 12:30:00' , 1, '
    u0000', NULL, NULL,'c0000'),
313     ('t0000000012','Hardware','2020-06-20 16:30:00' , 1, '
    u0001', NULL, 's000005', 'c0000'),
314     ('t0000000013','Hardware','2020-07-30 13:30:00' , 1, '
    u0002', 's000003', NULL, 'c0003'),
315     ('t0000000014','Hardware','2020-08-30 16:30:00' , 1, '
    u0003', 's000005', NULL, 'c0003');
```

```
316 ;
317
318 GO
319
320
321 INSERT INTO Task(TaskID, Description, TicketID, StaffID,
    StatusID)
322 VALUES  ('t000000000','Replace it with the new one.','
    t0000000000','s000007','s4'),
323         ('t000000001','Run the diagnostics I.','
    t0000000000',NULL,'s0'),
324         ('t000000002','Pass it to other person.','
    t0000000001',NULL,'s4'),
325         ('t000000003','Need to ask outside expert I.','
    t0000000002',NULL,'s0'),
326         ('t000000004','Run the diagnostics II.','
    t0000000009','s000008','s0'),
327         ('t000000005','Run the diagnostics III.','
    t0000000009','s000009','s1'),
328         ('t000000006','Need to ask outside expert II.','
    t0000000009','s000001','s2'),
329         ('t000000007','Need to ask outside expert III.','
    t0000000009','s000002','s2'),
330         ('t000000008','Run the diagnostics IV.','
    t0000000009','s000001','s4'),
331         ('t000000009','Run the diagnostics V.','
    t0000000009','s000002','s4'),
332         ('t000000010','Run the diagnostics III.','
    t0000000009','s000009','s1'),
333         ('t000000011','Need to ask outside expert II.','
    t0000000009','s000001','s2'),
334         ('t000000012','Need to ask outside expert III.','
    t0000000009','s000002','s2'),
335         ('t000000013','Run the diagnostics IV.','
    t0000000009','s000001','s3'),
336         ('t000000014','Run the diagnostics V.','
    t0000000009','s000002','s4');
337
338 GO
339
340 INSERT INTO HardwareTicket(HardwareID, TicketID)
341 VALUES ('h00000000','t0000000000'),
342        ('h00000000','t0000000001'),
343        ('h00000002','t0000000000'),
344        ('h00000003','t0000000003'),
345        ('h00000003','t0000000004'),
```

```
346         ('h00000003','t0000000005'),
347         ('h00000003','t0000000006'),
348         ('h00000002','t0000000006'),
349         ('h00000004','t0000000000'),
350         ('h00000004','t0000000002');
351
352 GO
353
354 INSERT INTO Communication(CommunicationID, Description,
    TicketID,
355                            UserID, RequestingStaffID,
    RespondingStaffID)
356 VALUES
357     ('c00000000000','initial comment. Explain in detail.',
    't0000000000',NULL,'s000000','s000000'),
358     ('c00000000001','Is this your first time having this
    problem?','t0000000000','u0000','s000000',NULL),
359     ('c00000000002','Have you tried to fix it?','
    t0000000000','u0000','s000001','s000000'),
360     ('c00000000003','What is your email address?','
    t0000000001',NULL,'s000001','s000000'),
361     ('c00000000004','It will take more than a week. Is it
    okay?','t0000000002','u0001','s000002','s000000'),
362     ('c00000000005','It will take more than a month. Is it
     okay?','t0000000002',NULL,'s000008',NULL),
363     ('c00000000006','What is your phone number?','
    t0000000003','u0001','s000008',NULL),
364     ('c00000000007','What is your skill level?','
    t0000000004','u0001','s000009','s000000'),
365     ('c00000000008','Do you understand my explanation?','
    t0000000005',NULL,'s000009','s000000'),
366     ('c00000000009','What is your address?','t0000000005',
    'u0002','s000009',NULL),
367     ('c00000000010','What is your email address?','
    t0000000010',NULL,'s000001','s000000'),
368     ('c00000000011','It will take more than a week. Is it
    okay?','t0000000010','u0001','s000002','s000000'),
369     ('c00000000012','It will take more than a month. Is it
     okay?','t0000000010',NULL,'s000008',NULL),
370     ('c00000000013','What is your phone number?','
    t0000000009','u0001','s000008',NULL),
371     ('c00000000014','What is your skill level?','
    t0000000009','u0001','s000009','s000000'),
372     ('c00000000015','Do you understand my explanation?','
    t0000000008',NULL,'s000009','s000000'),
373     ('c00000000016','What is your address?','t0000000007',
```

```
373 'u0002','s000009',NULL);
374
375 GO
376
377
378 -- OK/Recheck done 1. Display a list of all tickets
    submitted within a given month of the current year.
379 -- The month will be supplied to the routine as a word (e.
    g. April)
380
381 -- 1st try
382 DROP PROCEDURE If EXISTS usp_GetTicketsForAGivenMonth;
383 GO
384
385 CREATE PROCEDURE usp_GetTicketsForAGivenMonth
386     @currentMonth INT
387 AS
388 BEGIN
389     SELECT TicketID FROM Ticket WHERE   MONTH(
    DateGenerated) = @currentMonth;
390 END;
391 GO
392
393 EXEC usp_GetTicketsForAGivenMonth 4;
394
395 -- practice for 2nd try
396 SELECT CAST(FORMAT(DateGenerated,'MMMM') AS CHAR) FROM
    Ticket;
397
398
399 -- 1. Final Solution
    ------------------------------------------------------------
    --------------------------
400 DROP PROCEDURE If EXISTS
    usp_GetTicketsForAGivenMonthAsChar;
401 GO
402
403 CREATE PROCEDURE usp_GetTicketsForAGivenMonthAsChar
404     @currentMonthChar CHAR(32)
405 AS
406 BEGIN
407     SELECT TicketID FROM Ticket WHERE FORMAT(DateGenerated
    ,'MMMM') = @currentMonthChar;
408 END;
409 GO
410
```

```
411 EXEC usp_GetTicketsForAGivenMonthAsChar 'March';
412 GO
413 EXEC usp_GetTicketsForAGivenMonthAsChar 'December';
414 GO
415 ------------------------------------------------------------
    ------------------------------------------------------
416
417 -- OK/Recheck done 2. Display   a   list    of  the top 10
        tickets that    have    had the most    activity    in
        the form
418 -- of   comments.   A   start   date    and end date
    will    be  supplied    to  the routine in  the form
419 -- ‘yyyy-mm-dd’.
420 -- I assummed "communicatons" as "comments."
421
422 -- 2. Final Solution
     ------------------------------------------------------
    ---------------------------------
423 DROP PROCEDURE IF EXISTS usp_GetMostCommunicationTickets;
424 GO
425
426 CREATE PROCEDURE usp_GetMostCommunicationTickets
427     @startDate DATE, @endDate DATE
428 AS
429 BEGIN
430     SELECT TOP 10 T.TicketID, COUNT(C.TicketID) AS NumCom
431     FROM Ticket T
432     LEFT JOIN Communication C ON C.TicketID = T.TicketID
433     WHERE DateGenerated > @startDate AND DateGenerated <@
    endDate
434     GROUP BY T.TicketID
435     ORDER BY NumCom DESC
436 END;
437 GO
438
439 EXEC usp_GetMostCommunicationTickets '2020-01-03', '2020-
    05-04';
440 GO
441 ------------------------------------------------------------
    ------------------------------------------------------
442
443 -- Recheck done 3. Display  a   list    of  tickets for a
        particular  category,   ordered by  descending  date
    ,   with
444 -- the  corresponding   date    displayed   in  the format
        (Month  dddd,   yyyy)   e.g.    November    21st, 2009
```

```
444 .
445 -- A    category    name    will    be  supplied    to
    the routine.
446
447 DROP PROCEDURE IF EXISTS usp_GetTicketsForAGivenCategory;
448 GO
449
450 CREATE PROCEDURE usp_GetTicketsForAGivenCategory
451     @givenCategory CHAR(32)
452 AS
453 BEGIN
454     DECLARE @dayClassification CHAR(32), @dayValue INT, @
    dateGenerated DATETIME2(7) = NUll;
455     SET @dayClassification ='TH';
456
457     WHILE @@ROWCOUNT >0
458     BEGIN
459         SELECT  @dayValue = DATENAME(DAY,DateGenerated
    ) , @dateGenerated = DateGenerated FROM Ticket;
460     --    SELECT @dayValue = DATENAME(DAY,DateGenerated)
    FROM Ticket;
461         PRINT '@dayValue:  '+ STR(@dayValue);
462
463
464         IF @dayValue=30
465         BEGIN
466             SET @dayClassification ='ST';
467             PRINT 'Hello';
468         END
469         ELSE
470         BEGIN
471             SET @dayClassification ='RD';
472         END
473     END
474     SELECT FORMAT(DateGenerated,'MMMM')+' '+DATENAME(DAY,
    DateGenerated)+@dayClassification+DATENAME(YEAR,
    DateGenerated), TicketID FROM Ticket WHERE CategoryID = @
    givenCategory ORDER BY DateGenerated DESC
475
476 END;
477
478 GO
479
480 EXEC usp_GetTicketsForAGivenCategory 'c0000';
481 GO
482
```

```
483 -- New try similar to Chinook one  ==> Succeed!! DONE
    except DESC
484 -- Needs to find out how to DESC!!!!
485 DROP PROCEDURE IF EXISTS usp_GetTicketsForAGivenCatetory2;
486 GO
487
488 CREATE PROCEDURE usp_GetTicketsForAGivenCatetory2
489     @categoryInput CHAR(5)
490 AS
491 BEGIN
492     DECLARE @dayNumber INT;
493     DECLARE @yearNumber INT;
494     DECLARE @monthName CHAR(10);
495 --    DECLARE @dateGenerated DATETIME2 = CAST(0 AS
    DATETIME2);
496     DECLARE @dateGenerated DATETIME2(7) = CONVERT(
    DATETIME2(7),'01/01/1900 00:00', 103);
497     DECLARE @ticketID CHAR(11);
498     DECLARE @line VARCHAR(90);
499
500     SELECT TOP 1 @dayNumber = DATENAME(DAY,DateGenerated
    ), @yearNumber = DATENAME(YEAR, DateGenerated),
501                  @monthName = FORMAT(DateGenerated,'MMMM'
    ),@ticketID = TicketID, @dateGenerated = DateGenerated
502     FROM Ticket
503     WHERE DateGenerated > @dateGenerated AND CategoryID
     = @categoryInput
504     ORDER BY DateGenerated DESC;
505     WHILE @@ROWCOUNT >0
506     BEGIN
507         SET @line = 'Ticket ID:' + @ticketID +' is
    generated on  '+@monthName;
508
509         IF @dayNumber = 21 OR @dayNumber = 1
510             SET @line += STR(@dayNumber) +'st';
511         ELSE IF @dayNumber = 22 OR @dayNumber =2
512             SET @line += STR(@dayNumber) +'nd';
513         ELSE IF @dayNumber = 23 OR @dayNumber =3
514             SET @line += STR(@dayNumber) + 'rd';
515         ELSE
516             SET @line += STR(@dayNumber) +'th';
517         SET @line += ',' + STR(@yearNumber);
518         PRINT @line;
519         SELECT TOP 1 @dayNumber = DATENAME(DAY,
    DateGenerated), @yearNumber = DATENAME(YEAR, DateGenerated
    ),  @ticketID = TicketID,
```

```
520                     @monthName = FORMAT(DateGenerated,'MMMM'
    ), @dateGenerated = DateGenerated
521             FROM Ticket
522             WHERE DateGenerated > @dateGenerated AND
    CategoryID = @categoryInput
523             ORDER BY DateGenerated;
524         END
525 END;
526 GO
527
528 EXEC usp_GetTicketsForAGivenCatetory2 'c0000';
529 GO
530
531
532 --with clause practice
533 with reversedTicket as
534         (
535             SELECT TOP 100 TicketID as TicketIDreversedOrder
536             FROM Ticket
537             ORDER BY TicketID DESC
538
539         )
540         SELECT T1.TicketIDreversedOrder, DateGenerated
541         FROM Ticket T2, reversedTicket T1
542         WHERE T1.TicketIDreversedOrder = T2.TicketID;
543
544 -- 3. Final solution
     ------------------------------------------------------
    ------------------------------------
545 DROP PROCEDURE IF EXISTS usp_GetTicketsForAGivenCatetory2;
546 GO
547
548 CREATE PROCEDURE usp_GetTicketsForAGivenCatetory2
549         @categoryInput CHAR(5)
550 AS
551 BEGIN
552
553         DECLARE @dayNumber INT;
554         DECLARE @yearNumber INT;
555         DECLARE @monthName CHAR(10);
556  --       DECLARE @dateGenerated DATETIME = CAST(0 AS DATETIME
    );
557         DECLARE @dateGenerated DATETIME2(7) = CONVERT(
    DATETIME2(7),'01/01/1900 00:00', 103);
558         DECLARE @ticketID CHAR(11);
559         DECLARE @line VARCHAR(90);
```

```
560
561     with reversedTicket as
562     (
563         SELECT TOP 100 TicketID as TicketIDreversedOrder,
    DateGenerated as DG
564         FROM Ticket
565         ORDER BY DateGenerated DESC
566     )
567     SELECT TOP 1 @dayNumber = DATENAME(DAY,DG), @
    yearNumber = DATENAME(YEAR, DG),
568                  @monthName = FORMAT(DG,'MMMM'),@ticketID
     = T1.TicketIDreversedOrder, @dateGenerated = DG
569     FROM reversedTicket T1, Ticket T2
570     WHERE DateGenerated > @dateGenerated AND CategoryID
     = @categoryInput AND T1.TicketIDreversedOrder = T2.
    TicketID
571
572     WHILE @@ROWCOUNT >0
573     BEGIN
574         SET @line = 'Ticket ID:' + @ticketID +' is
    generated on  '+@monthName;
575
576         IF @dayNumber = 21 OR @dayNumber = 1
577             SET @line += STR(@dayNumber) +'st';
578         ELSE IF @dayNumber = 22 OR @dayNumber =2
579             SET @line += STR(@dayNumber) +'nd';
580         ELSE IF @dayNumber = 23 OR @dayNumber =3
581             SET @line += STR(@dayNumber) + 'rd';
582         ELSE
583             SET @line += STR(@dayNumber) +'th';
584         SET @line += ',' + STR(@yearNumber);
585         PRINT @line;
586
587         with reversedTicket as
588             (
589                 SELECT TOP 100 TicketID as
    TicketIDreversedOrder, DateGenerated as DG
590                 FROM Ticket
591                 ORDER BY DateGenerated DESC
592             )
593         SELECT TOP 1 @dayNumber = DATENAME(DAY,DG), @
    yearNumber = DATENAME(YEAR, DG),  @ticketID = T1.
    TicketIDreversedOrder,
594                   @monthName = FORMAT(DG,'MMMM'), @
    dateGenerated = DG
595         FROM  reversedTicket T1, Ticket T2
```

```
596         WHERE DateGenerated < @dateGenerated AND
    CategoryID = @categoryInput AND T1.TicketIDreversedOrder
     = T2.TicketID
597
598     END
599 END;
600 GO
601
602 EXEC usp_GetTicketsForAGivenCatetory2 'c0000';
603 GO
604 EXEC usp_GetTicketsForAGivenCatetory2 'c0003';
605 GO
606 ------------------------------------------------------
    ------------------------------------------------------
607
608
609 -- Rechecked done/OK 4. Return  the total   number  of
    active  tasks   for a   given   support staff   member. An
        employee
610 -- number   will    be  supplied    to  the routine.
611 -- The  routine should  also
612 -- return   zero    (0), if there   are no  active  tasks
        for that    person,
613 -- and  negative    one (-1), if    the support staff
    member  could   not be  found.
614
615 --By "active", it's s0, s1, s2,s3 in my case.
616
617 -- 4. Final Solution
     ------------------------------------------------------
    ---------------
618 DROP PROCEDURE IF EXISTS usp_GetTasksForAStaff;
619 GO
620
621 CREATE PROCEDURE usp_GetTasksForAStaff
622     @givenStaff CHAR(32)
623 AS
624 BEGIN
625     DECLARE @resultStaff CHAR(32)
626     SET @resultStaff = 'DEFAULT';
627     SELECT @resultStaff= StaffID from Task WHERE @
    givenStaff =StaffID;
628     PRINT '@resultStaff= '+@resultStaff;
629
630     IF (@resultStaff = 'DEFAULT')
631     BEGIN
```

```
632         SELECT -1
633     END
634     ELSE
635     BEGIN
636         SELECT COUNT(TaskID) AS
    NumberOfActiveTasksForAStaff FROM Task WHERE StaffID = @
    givenStaff AND StatusID !='s4'
637     END
638 END;
639
640 GO
641
642 EXEC usp_GetTasksForAStaff 's000006'; -- -1 because the
    support  staff  member cannot be found
643 GO
644 EXEC usp_GetTasksForAStaff 's000007'; -- 0 because the
    support staff member has 0 active task
645 GO
646 EXEC usp_GetTasksForAStaff 's000001'; -- 3 because the
    support staff member has 3 active tasks
647 GO
648 EXEC usp_GetTasksForAStaff 's000003'; --  -1 because the
    support  staff  member cannot be found
649 GO
650 ------------------------------------------------------------
    ------------------------------------------------------
651
652 -- OK/RecheckDone 5. Display    a   “page”  of  ticket
    information by  passing,    to  the routine,    a   page
        number
653 -- and  the number  of  tickets per page. For example,
    passing "1,10" will return the first ten tickets
654 -- (ordered by ticket id) but passing "2,10" will return
    next ten tickets (i.e. page 2).
655
656
657 DROP PROCEDURE IF EXISTS usp_GetTicketsByPage;
658 GO
659
660 CREATE PROCEDURE usp_GetTicketsByPage
661     @page INT, @numberOfTickets INT
662 AS
663 BEGIN
664     SELECT TicketID FROM Ticket WHERE  @@ROWCOUNT= @
    numberOfTickets AND @page =@page
665     PRINT @@ROWCOUNT
```

```
666 END;
667
668 GO
669
670 EXEC usp_GetTicketsByPage 1,0
671
672 -- 1st stage just use numberOfTickets! (error)
673 -- DROP PROCEDURE IF EXISTS usp_GetTicketsByNumberTickets;
674 -- GO
675 --
676 -- CREATE PROCEDURE usp_GetTicketsByNumberTickets
677 --     @numberTickets INT, @page INT
678 -- AS
679 -- BEGIN
680 --     WHILE  @page < (
681 --         SELECT @@ROWCOUNT FROM Ticket
682 --     )
683 --     SELECT TOP (@numberTickets) TicketId FROM Ticket
684 -- END;
685 --
686 -- GO
687 --
688 -- SELECT @@ROWCOUNT FROM Ticket;
689 -- SELECT TicketID, ROW_NUMBER() OVER (ORDER BY TicketID)
    from Ticket;
690 --
691 -- EXEC usp_GetTicketsByNumberTickets 6, 3
692
693 -- 2nd stage
694 SELECT TOP 2 * FROM Ticket WHERE TicketID IN
695 (SELECT TicketID
696 FROM Ticket
697 ORDER BY TicketID
698 OFFSET 10 ROWS);
699
700 -- practice using numbers.
701 SELECT TOP 2 * FROM Ticket WHERE TicketID IN
702 (SELECT TicketID
703 FROM Ticket
704 ORDER BY TicketID
705 OFFSET 10 ROWS);
706
707 -- 5. Final Solution
    ------------------------------------------------------------
    -----------------------------------
708
```

```
709 DROP PROCEDURE IF EXISTS usp_GetTicketsByPage;
710 GO
711
712 CREATE PROCEDURE usp_GetTicketsByPage
713     @page INT, @numberOfTickets INT
714 AS
715 BEGIN
716     SELECT TOP (@numberOfTickets) * FROM Ticket WHERE
    TicketID IN
717         (SELECT TicketID
718         FROM Ticket
719         ORDER BY TicketID
720         OFFSET ((@page-1)*@numberOfTickets) ROWS);
721 END;
722
723 GO
724
725 EXEC usp_GetTicketsByPage 2,5;
726 GO
727 EXEC usp_GetTicketsByPage 3,4;
728 GO
729 ------------------------------------------------------------
    ------------------------------------------------------------
    --
730
```