Account Register – Crashing & Single Day Limitation – GL Account 1025

06/25/2026 9:53 am EDT

Account Register – Crashing & Single Day Limitation – GL Account [00163534, 00163757]

When attempting to run and export the Account Register report, the process is extremely limited in scope and frequently crashes.

- The report will only allow one day at a time.
- Even when pulling data for a single day, the report sometimes crashes before completing.

Emeka tested and confirmed severe performance issues when accessing the Account Register List for certain GL accounts. Investigation determined that the issue was not related to the front-end application but rather to the SQL query executed by the

dbo.Account_Register_List stored procedure on the backend.

The issue appears to be caused by stored procedure performance when querying large volumes of `GL_Register` data. The optimized version is currently performing much better in the customer's environment.

A modified version of the query was developed and tested on the customer's environment, focusing on improving execution efficiency while maintaining the existing functionality and result set.

After implementing the changes, the response time for generating the Account Register List improved significantly, and the slow performance that had originally prompted the ticket is no longer present.

```
aLTER PROC [dbo].[Account_Register_List]
  @account_code  varchar(30),
  @branch_code   varchar(30),
  @category_code  varchar(30),
  @dfromdate     datetime,
  @dthrudate     datetime,
  @accounttype   varchar(30),
  @registertype  varchar(30)
```

```
AS

BEGIN

  SET NOCOUNT ON;


  DECLARE

    @account_id     int,

    @branch_id      int,

    @category_id    int,

    @registertype_id int,

    @sql          nvarchar(max);


  EXEC Get_Account_Id  @account_code,  @account_id  OUTPUT;

  EXEC Get_Branch_Id   @branch_code,   @branch_id   OUTPUT;

  EXEC Get_Category_Id @category_code, @category_id OUTPUT;


  SELECT @registertype_id = ISNULL(Register_Type_Id, 0)

  FROM SS_Register_Type

  WHERE Type_Code = @registertype;


  IF @dfromdate IS NOT NULL

    SET @dfromdate = DATEADD(DAY, DATEDIFF(DAY, 0, @dfromdate), 0);


  IF @dthrudate IS NOT NULL

    SET @dthrudate = DATEADD(DAY, DATEDIFF(DAY, 0, @dthrudate), 0);


  IF @accounttype = 'BANK'
```

```
BEGIN
  SET @sql = N'
  SELECT
    ISNULL(pymt.Check_Number, '' ''),
    reg.Date,
    reg.Credit_Or_Debit,
    rt.Type_Code,
    ISNULL(reg.Reference, '' ''),
    (CASE reg.Type_CVEO
      WHEN ''C'' THEN cu.Customer_Name
      WHEN ''V'' THEN vd.Company_Name
      ELSE reg.Other_Name
     END) as Name,
    reg.Amount,
    reg.Memo,
    reg.Status,
    gl.Account_Code,
    gl.Description,
    reg.Splits,
    ISNULL(reg.Register_Number, 0),
    ISNULL(pymt.Check_Id, 0),
    br.Branch_Code,
    ct.Category_Code,
    reg.Register_Id
  FROM GL_Register reg
  INNER JOIN SS_Register_Type rt
```

```
   ON rt.Register_Type_Id = reg.Register_Type_Id
INNER JOIN GL_Account gl
   ON gl.Account_Id = reg.Offset_Account_Id
INNER JOIN AR_Branch br
   ON br.Branch_Id = reg.Branch_Id
INNER JOIN AR_Category ct
   ON ct.Category_Id = reg.Category_Id
LEFT JOIN AR_Customer cu
   ON cu.Customer_Id = reg.Customer_Id
LEFT JOIN AP_Vendor vd
   ON vd.Vendor_Id = reg.Vendor_Id
OUTER APPLY
(
   SELECT TOP (1)
      p.Check_Number,
      p.Check_Id
   FROM
   (
      SELECT Check_Number, Check_Id, 1 AS sort_key
      FROM AP_Check
      WHERE Register_Id = reg.Register_Id

      UNION ALL

      SELECT Check_Number, Check_Id, 2 AS sort_key
      FROM AP_Check
```

```
        WHERE Void_Register_Id = reg.Register_Id
      ) p
      ORDER BY p.sort_key
    ) pymt
    WHERE reg.Account_Id = @account_id';
END
ELSE
BEGIN
    SET @sql = N'
    SELECT
      Check_No = '''',
      reg.Date,
      reg.Credit_Or_Debit,
      rt.Type_Code,
      ISNULL(reg.Reference, '' ''),
      Name = CASE reg.Type_CVEO
            WHEN ''C'' THEN cu.Customer_Name
            WHEN ''V'' THEN vd.Company_Name
            WHEN ''E'' THEN (emp.Last_Name + '', '' + emp.First_Name)
            ELSE reg.Other_Name
          END,
      reg.Amount,
      reg.Memo,
      reg.Status,
      gl.Account_Code,
      gl.Description,
```

```
        reg.Splits,
        ISNULL(reg.Register_Number, 0),
        0,
        br.Branch_Code,
        ct.Category_Code,
        reg.Register_Id
    FROM GL_Register reg
    INNER JOIN SS_Register_Type rt
        ON rt.Register_Type_Id = reg.Register_Type_Id
    INNER JOIN GL_Account gl
        ON gl.Account_Id = reg.Offset_Account_Id
    INNER JOIN AR_Branch br
        ON br.Branch_Id = reg.Branch_Id
    INNER JOIN AR_Category ct
        ON ct.Category_Id = reg.Category_Id
    LEFT JOIN SY_Employee emp
        ON emp.Employee_Id = reg.Employee_Id
    LEFT JOIN AR_Customer cu
        ON cu.Customer_Id = reg.Customer_Id
    LEFT JOIN AP_Vendor vd
        ON vd.Vendor_Id = reg.Vendor_Id
    WHERE reg.Account_Id = @account_id';
END

IF @branch_id > 1
    SET @sql += N' AND reg.Branch_Id = @branch_id';
```

```
IF @category_id > 1
  SET @sql += N' AND reg.Category_Id = @category_id';

IF @registertype_id > 0
  SET @sql += N' AND reg.Register_Type_Id = @registertype_id';

IF @dfromdate IS NOT NULL AND @dthrudate IS NULL
BEGIN
  SET @sql += N' AND reg.Date >= @dfromdate';
END
ELSE IF @dfromdate IS NULL AND @dthrudate IS NOT NULL
BEGIN
  SET @sql += N' AND reg.Date < DATEADD(DAY, 1, @dthrudate)';
END
ELSE IF @dfromdate IS NOT NULL AND @dthrudate IS NOT NULL
BEGIN
  SET @sql += N' AND reg.Date >= @dfromdate
          AND reg.Date < DATEADD(DAY, 1, @dthrudate)';
END

SET @sql += N' ORDER BY reg.Date, reg.Register_Number
        OPTION (RECOMPILE);';

EXEC sp_executesql
  @sql,
```

```
      N'@account_id int,
       @branch_id int,
       @category_id int,
       @registertype_id int,
       @dfromdate datetime,
       @dthrudate datetime',
      @account_id = @account_id,
      @branch_id = @branch_id,
      @category_id = @category_id,
      @registertype_id = @registertype_id,
      @dfromdate = @dfromdate,
      @dthrudate = @dthrudate;

   RETURN;
END
GO
```