|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **修改日期** | **版本** | **修改原因** | **修改人姓名** | **立案單號** |
| 2017/02/16 | 1 | Created | 劉文明 | **170216000399** |
| 2017/2/21 | 2 | 增加查詢資料的錯誤訊息處理 | 劉文明 | **170216000399** |
| 2017/2/23 | 3 | 1. 增加刪檔機制，避免批次重跑資料dup 2. SQL調整排除團險件的理賠紀錄 3. 原有的判斷(OK001 : 溢付扣回)，增加比對的原因判斷    1. 是否有預付金給付扣回(OK002)    2. 是否有人工修正(OK003) | 劉文明 | **170216000399** |
| 2017/3/6 | 4 | 1. 增加是否預付金給付(OK004) | 劉文明 | **170216000399** |
| 2017/3/14 | 5 | 1. 增加人工修正整張保單沖回的判斷 | 劉文明 | **170313000418** |
| 2017/3/22 | 6 | 1. 增加比對分類    1. OK005:人工整筆沖回但含有非身故給付項目    2. OK006:非當月份的溢付扣回 | 劉文明 | **170313000418** |
| 2017/4/13 | 7 | 1. 問題處理:解決部分案件未正確分類到OK003及OK005 2. 要排除的理賠給付項目改以代碼中文對照設定處理 | 劉文明 | **170411000332** |

**一、程式功能概述**

|  |  |
| --- | --- |
| **程式功能** | IFRS帳務明細比對作業 |
| **程式名稱** | AAZZ\_B011.java |
| **作業方式** | BATCH |
| **概要說明** | 配合IFRS專案，比對理賠紀錄與帳務資料，將不符合案件產出提供精算評價科確認。 |
| **需求單位** | 精算評價一科 |
| **作業單位** | 壽險資訊二科 |
| **作業平台** | ■一般 □平板電腦 □手機 |
| **使用對象** | ■員工(UCBean) □客戶(CustomerBean) |

**二、程式流程圖**

****

**三、相關檔案**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **項次** | **中文說明** | **檔案名稱** | **查詢** | **新增** | **修改** | **刪除** |
|  | 理賠紀錄檔 | DTAAB001 | ■ | □ | □ | □ |
|  | 理賠溢付件人工修正檔 | DTAAY001 | ■ | □ | □ | □ |
|  | 交易會計分錄明細檔 | DTDKF001 | ■ | □ | □ | □ |
|  | 理賠預付金檔 | DTAAI001 | ■ | □ | □ | □ |
|  | 人工修正案件受理檔 | DTAIZ000 | ■ | □ | □ | □ |
|  | 人工修正案件明細檔 | DTAIZ001 | ■ | □ | □ | □ |
|  | 理賠保單明細檔 | DTAAB100 | ■ | □ | □ | □ |
|  | IFRS帳務比對結果檔 | DTAAZZ03 | ■ | ■ | ■ | ■ |

**四、相關模組**

|  |  |  |
| --- | --- | --- |
| **項次** | **中文說明** | **程式名稱** |
| 1. | 異常訊息記錄模組 | batch.ErrorLog |
| 2. | 批次作業件數記錄模組 | batch.CountManager |
| 3. |  |  |

**五、批次基本資料**

|  |  |
| --- | --- |
| **作業名稱** | JAAAMZ001 |
| **業務別** | AA |
| **次系統名稱** | ZZ |
| **處理週期** | 月 |
| **分批處理的件數** | 100 |

**六、傳入參數**

|  |  |  |  |
| --- | --- | --- | --- |
| **項次** | **說明** | **資料型態** | **備註** |
| 1 | 帳務日期\_起日 | DATE | 必傳 |
| 2 | 帳務日期\_迄日 | DATE | 必傳 |
| 3 | 受理編號 | STRING | 可不傳入 |
| 4 | 險別代號 | STRING | 可不傳入 |

* 1. **七、程式內容**初始：
     1. 件數歸零：
        1. 讀取件數、
        2. 處理件數、
        3. 錯誤件數。
     2. $執行日期: 系統日期
  2. 傳入參數檢核：
     1. 帳務日期的檢核
        1. 帳務日期\_起日:
           1. 需檢核是否為正確日期格式，若有誤，顯示錯誤訊息 “帳務起日不為合理日期” ，程式結束。
        2. 帳務日期\_迄日:
           1. 需檢核是否為正確日期格式，若有誤，顯示錯誤訊息“帳務迄日不為合理日期”，程式結束。
        3. 帳務日期\_起日 需小 於帳務日期\_迄日 :
           1. 若不符合，顯示錯誤訊息 “帳務起日需 <= 帳務迄日，請確認。”，程式結束。
  3. 刪除檔案: //\*\*將帳務日期\_起日、帳務日期\_迄日符合的資料刪除
     1. 刪除IFRS帳務比對結果檔DTAAZZ03,BY參數
        1. 帳務日期\_起日: 同批次傳入帳務日期\_起日
        2. 帳務日期\_迄日 : 同批次傳入帳務日期\_迄日
        3. 若刪除異常，
           1. 顯示錯誤訊息“刪除IFRS帳務比對結果檔DTAAZZ03失敗”，程式結束。
        4. 若刪不到資料屬正常,繼續處理下一STEP
  4. //\*\*分別讀取理賠紀錄與會計分錄明細資料，以受理編號加總金額後比對，將金額不相符的資料再做判斷。SQL語法請參閱附錄一，測試環境請以此語法撈取資料。
  5. (AA\_REC)讀取理賠紀錄檔DTAAB001，BY查詢條件
     1. 帳務日期 : 介於 帳務日期\_起日 ， 帳務日期\_迄日 之間。
     2. //\*\*財會不列入理賠給付款的要排除
     3. 理賠給付代碼: not in (代碼中文對照(AA,AAZZB011\_BYPASS\_CLAMCODE)所有設定的資料)
     4. //\*\*給付狀態要拆正常給付跟(協議、削減)為0的狀態
     5. 給付狀態:in ('1','4','7') 或 (in ('2','3') 且給付金額 <> 0)
     6. //\*\*排除SG轉檔件
     7. 受理編號 : not like 'OL%' and not like ‘\_\_\_\_\_\_\_\_\_\_\_G\_\_’
     8. 以受理編號 group by
     9. 查詢欄位
        1. 受理編號
        2. sum(給付金額) as AA\_AMT
  6. (DK\_REC)讀取會計分錄明細檔DTDKF001，BY查詢條件
     1. 帳務日期 : 介於 帳務日期\_起日 ， 帳務日期\_迄日 之間。
     2. 會計子目 : '5125000' (理賠給付款)
     3. 交易種類 : like 'AA\_\_\_\_' (理賠相關交易)
     4. 摘要 : not like '\_\_\_\_\_\_\_\_\_\_\_G\_\_'
     5. //\*\* memo欄位對應理賠系統是放受理編號
     6. 以摘要(memo) group by
     7. 查詢欄位
        1. 摘要(memo)
        2. //\*\* 若借貸方 = 'CR' 負向，則金額加總時要用負數金額
        3. sum(case 借貸方 = 'CR' then –1\*金額 else 金額 end) as DK\_AMT
  7. //\*\*將AA\_REC與DK\_REC分別用受理編號比對，先判斷以AA\_REC有(DK\_REC可能沒有)的案件,再判斷以DK\_REC為主(AA\_REC可能沒有)
  8. (AA\_BIG)查詢AA\_REC left join DK\_REC on AA\_REC.受理編號 = DK\_REC.摘要，by查詢條件
     1. AA\_REC.AA\_AMT – DK\_REC.DK\_AMT <> 0 or DK\_REC.memo is null
     2. 查詢欄位
        1. AA\_REC.aply\_no as ta\_aply
        2. AA\_REC.aa\_amt,DK\_REC.memo as tb\_aply
        3. DK\_REC.dk\_amt
        4. AA\_REC.aa\_amt - DK\_REC.dk\_amt as ta\_amt
  9. (DK\_BIG)查詢AA\_REC right join DK\_REC on AA\_REC.受理編號 = DK\_REC.摘要，by查詢條件
     1. AA\_REC.AA\_AMT – DK\_REC.DK\_AMT <> 0 or AA\_REC.aply\_no is null
     2. 查詢欄位
        1. AA\_REC.aply\_no as ta\_aply
        2. AA\_REC.aa\_amt,DK\_REC.memo as tb\_aply
        3. DK\_REC.dk\_amt
        4. AA\_REC.aa\_amt - DK\_REC.dk\_amt as tb\_amt
  10. //\*\*將兩者差異資料合併
  11. (t\_dist)查詢AA\_BIG union DK\_BIG 所有欄位
  12. //\*\*去看又沒有做溢付扣回
  13. (chk\_ovrp)查詢t\_dist
      1. left join 理賠紀錄檔DTAAB001 on t\_dist.受理編號 = DTAAB001.受理編號
      2. left join理賠溢付件人工修正檔DTAAY001 on DTAAB001的受理編號、保單號碼、事故者ID、險別代號、理賠給付代碼相同
      3. BY 查詢條件
         1. DTAAB001.受理編號 is not null
      4. 查詢欄位
         1. DTAAB001.受理編號
         2. DTAAB001.保單號碼
         3. DTAAB001.險別代號
         4. DTAAB001.理賠給付代碼
         5. DTAAB001.事故者ID
         6. DTAAB001.理賠給付名稱
         7. (DTAAY001.理賠差額 – DTAAY001.溢付餘額 ) as tot\_amt
  14. //\*\*將溢付資料以受理編號合併加總
  15. (t\_fin)查詢chk\_ovrp
      1. Group by 受理編號
      2. 查詢欄位
         1. aply\_no
         2. sum(tot\_amt) as over\_pay\_amt
  16. //\*\*將溢付總額資料併入查詢結果
  17. 查詢t\_dist left join t\_fin on 受理編號相同
      1. 查詢欄位
         1. T\_dist所有欄位
         2. T\_fin.over\_pay\_amt
  18. 若查無資料，屬正常
      1. 顯示訊息，“理賠紀錄與會計分錄明細資料查無資料”
  19. 若查詢異常，
      1. 顯示錯誤訊息,“讀取理賠紀錄與會計分錄明細資料有誤”
      2. set 批次回傳碼 為錯誤。
      3. 程式結束
  20. 若查詢無誤，依序逐筆處理每筆資料：
      1. 讀取件數++
      2. Set $錯誤碼 = 空白
      3. IF處理當筆.OVER\_PAY\_AMT <> 0 且 處理當筆.DIFF\_AMT <> 0 且 處理當筆.OVER\_PAY\_AMT +處理當筆.DIFF\_AMT = 0
         1. $錯誤碼 = ‘OK001’
      4. ELSE
         1. $錯誤碼 = ‘CK001’
      5. END IF
      6. //\*\*檢核是否為非當月份的溢付扣回:SQL請參閱[附錄五](#MARK5)
      7. IF $錯誤碼 = ‘CK001’
         1. 讀取理賠紀錄檔DTAAB001 left join 理賠溢付檔DTAAY001 on DTAAB001.受理編號 = DTAAY001.原受理編號相同
         2. 查詢條件
            1. DTAAB001受理編號 : 同處理當筆受理編號
         3. 查詢欄位
            1. DTAAY001. 理賠差額 – DTAAY001. 溢付餘額
         4. 若查無資料
            1. $溢付扣回金額 = 0
         5. 若有資料
            1. $溢付扣回金額 = 查詢結果金額
         6. IF 處理當筆.taAply=AA\_APLY\_NULL 且處理當筆.DK\_AMT > 0 且 處理當筆.DK\_AMT = $溢付扣回金額
            1. $錯誤碼 = ‘OK006’
         7. ELSE
            1. $錯誤碼 = ‘CK001’
         8. END IF
      8. END IF
      9. //\*\*若錯誤碼 = CK001 再判斷是否為其他原因造成比對不符
      10. //\*\*比對是否為預付金給付扣回:SQL請參閱[附錄二](#MARK2)
      11. IF $錯誤碼 = ‘CK001’
          1. 讀取理賠預付金檔DTAAI001 left join 理賠紀錄檔DTAAB001 on AAI001.受理編號 = AAB001.預付金受理編號且保單號碼、險別代號、事故者ID相同
          2. BY查詢條件
             1. AAI001.受理編號: 處理當筆.TB\_APLY
             2. AAB001.受理編號 不為空值
             3. AAI001.預付金額 + AAB001.給付金額 = 0
          3. 查詢欄位
             1. COUNT(\*) as CNT
          4. 若查詢筆數 > 0 (CNT > 0)
             1. Set $錯誤碼 = ‘OK002’
          5. 若查詢筆數 = 0
             1. Set $錯誤碼 = ‘CK001’
          6. 若讀取異常
             1. 訊息中文＝“預付金扣回讀取預付金檔錯誤，受理編號:”+處理當筆TB\_APLY
             2. CALL Batch.ErrorLog(異常訊息記錄模組)，記錄錯誤訊息。
             3. 錯誤件數++
             4. 繼續處理下一筆
      12. END IF
      13. //\*\*檢查是否有當月處理的人工修正:以理賠保單明細檔的所有保單去看有沒有於人工修正系統已資服覆核的案件，SQL請參閱[附錄三](#MARK3)
      14. IF $錯誤碼 = ‘CK001’
          1. IF 處理當筆的TA\_APLY,AA\_AMT,TB\_APLY,DK\_AMT,DIFF\_AMT不為空值，只有OVER\_PAY\_AMT是空值
             1. (AI\_FIX)先讀取人工修正案件明細檔DTAIZ001 left join 人工修正案件受理檔DTAIZ000 on 案件編號相同，BY查詢條件

Z000.資服覆核日期 :Between帳務日期\_起日 and 帳務日期\_迄日

Z001.保單號碼 in (以處理當筆受理編號查詢理賠保單明細檔DTAAB100所有的保單號碼)

Z000.異動種類 like ‘AA%’

查詢欄位: (DISTINCT 底下所有欄位)

Z001.保單號碼

Z001.輸入日期 轉西元年月日

Z001.異動種類

Z000.退補表示

Z000.退補金額

Z000.傳票序號

Z000.傳票日期

Z000.制票單位

Z000.修正類別

Z000.資服覆核日期 轉西元年月日

* + - * 1. 再讀取AI\_FIX所有資料

讀取欄位

將所有退補金額加總

* + - * 1. 若讀取異常,

訊息中文＝“讀取人工修正紀錄檔錯誤，受理編號:”+處理當筆TA\_APLY

CALL Batch.ErrorLog(異常訊息記錄模組)，記錄錯誤訊息。

錯誤件數++

繼續處理下一筆

* + - * 1. 若無符合資料 (查詢結果為null)

SET $人工修正總金額 = 0

* + - * 1. 若有資料

SET $人工修正總金額 = 查詢出的退補金額加總

* + - * 1. //\*\*diffamt不為零的判斷處理: 部分理賠紀錄沖回，有理賠紀錄
        2. IF 處理當筆.DIFF\_AMT + $人工修正總金額 = 0 且處理當筆.DIFF\_AMT <> 0

Set $錯誤碼 = ‘OK003’

* + - * 1. ELSE

//\*\*diffamt=0且taAply=AA\_APLY\_NULL的處理:整筆沖回找不到理賠紀錄

IF處理當筆.DIFF\_AMT = 0 且 處理當筆.taAply=AA\_APLY\_NULL

IF 處理當筆.DK\_AMT = $人工修正總金額

Set $錯誤碼 = ‘OK003’

ELSE

Set $錯誤碼 = ‘CK001’

END IF

ELSE

Set $錯誤碼 = ‘CK001’

END IF

* + - * 1. END IF
      1. END IF
    1. END IF
    2. //\*\*檢查是否有非當月處理的人工修正:SQL請參閱[附錄三](#MARK3)
    3. IF $錯誤碼 = ‘CK001’
       1. $查詢起日 = 帳務日期\_迄日 + 1 天
       2. $查詢迄日 = 系統日期
       3. 作法同上述**檢查是否有當月處理的人工修正件**處理，只是用$查詢起日跟$查詢迄日去查資料
    4. END IF
    5. //\*\*檢查是否有當月人工整筆沖回但含有非身故給付項目:SQL請參閱[附錄六](#MARK6)
    6. IF $錯誤碼 = ‘CK001’
       1. (AI\_FIX)先讀取人工修正案件明細檔DTAIZ001 left join 人工修正案件受理檔DTAIZ000 on 案件編號相同，BY查詢條件
          1. Z000.資服覆核日期 :Between帳務日期\_起日 and 帳務日期\_迄日
          2. Z001.保單號碼 in (以處理當筆受理編號查詢理賠保單明細檔DTAAB100所有的保單號碼)
          3. Z000.異動種類 like ‘AA%’
          4. 查詢欄位: (DISTINCT 底下所有欄位)

Z001.保單號碼

Z001.輸入日期 轉西元年月日

Z001.異動種類

Z000.退補表示

Z000.退補金額

Z000.傳票序號

Z000.傳票日期

Z000.制票單位

Z000.修正類別

Z000.資服覆核日期 轉西元年月日

* + - 1. (AI\_SUM\_AMT)再讀取AI\_FIX所有資料
         1. 讀取欄位

將所有退補金額加總 as $AI\_AMT

* + - 1. (DK\_SUM\_AMT)
         1. 讀取會計分錄檔DTDKF001，by查詢條件

備註:同處理當筆受理編號

借貸方 DB\_CR : ‘CR’

對應檔代號 : ‘DTAAB010’

讀取欄位:

將所有紀錄的金額加總 as $DK\_SUM

* + - 1. 查詢AI\_SUM\_AMT left join DK\_SUM\_AMT on $AI\_AMT = $DK\_SUM
         1. 查詢欄位

計算符合條件的件數 as $CNT

* + - 1. 若讀取異常,
         1. 訊息中文＝“讀取人工修正紀錄檔錯誤，受理編號:”+處理當筆TA\_APLY
         2. CALL Batch.ErrorLog(異常訊息記錄模組)，記錄錯誤訊息。
         3. 錯誤件數++
         4. 繼續處理下一筆
      2. 若有資料
         1. IF $CNT > 1

Set $錯誤碼 = ‘OK005’

* + - * 1. ELSE

Set $錯誤碼 = ‘CK001’

* + - * 1. END IF
    1. END IF
    2. //\*\*檢查是否有非當月人工整筆沖回但含有非身故給付項目
    3. IF $錯誤碼 = ‘CK001’
       1. $查詢起日 = 帳務日期\_迄日 + 1 天
       2. $查詢迄日 = 系統日期
       3. 作法同上述**檢查是否有當月人工整筆沖回但含有非身故給付項目**處理，只是用$查詢起日跟$查詢迄日去查資料
    4. END IF
    5. //\*\*比對是否為預付金給付:SQL請參閱[附錄四](#MARK4)
    6. IF $錯誤碼 = ‘CK001’
       1. 讀取理賠預付金檔DTAAI001
       2. BY查詢條件
          1. AAI001.受理編號: 處理當筆.TB\_APLY
          2. 以處理當筆.TB\_APLY 做group by
       3. 查詢欄位
          1. AAI001.受理編號
          2. Sum(AAI001.預付金額 ) as $預付金總額
       4. 若讀取異常
          1. 訊息中文＝“預付金給付讀取預付金紀錄檔錯誤，受理編號:”+處理當筆TB\_APLY
          2. CALL Batch.ErrorLog(異常訊息記錄模組)，記錄錯誤訊息。
          3. 錯誤件數++
          4. 繼續處理下一筆
       5. 若查無資料，屬正常
          1. Set $預付金額 = 0
       6. 若有資料
          1. Set $預付金額 = $預付金總額
       7. IF 處理當筆.TA\_APLY=‘AA\_APLY\_NULL’且 處理當筆.DK\_AMT\_SUM > 0 且 $預付金額 = 處理當筆.DK\_AMT\_SUM
          1. Set $錯誤碼 = ‘OK004’
       8. ELSE
          1. Set $錯誤碼 = ‘CK001’
       9. END IF
    7. END IF
    8. 將資料寫入IFRS帳務比對結果檔DTAAZZ03,格式如[FORMAT(A)](#FORMATA)
    9. 若新增有誤,
       1. 訊息中文＝“新增DTAAZZ03異常，受理編號:”+理賠受理編號 + “,帳務摘要:” + 帳務摘要
       2. CALL Batch.ErrorLog(異常訊息記錄模組)，記錄錯誤訊息。
       3. 錯誤件數++
       4. 繼續處理下一筆
    10. 若新增無誤，
        1. 處理件數++
  1. 程式結束，需記錄下列件數：
     1. CALL batch.CountManager(批次作業件數記錄模組)，記錄讀取件數、團險排除件數、處理件數及錯誤件數。

FORMAT（A）：　　[ＢＣＡＫ](#A)！！！

|  |  |
| --- | --- |
| 目標欄位說明 | 來源欄位說明 |
| 執行日期 | $執行日期 |
| 帳務日期\_起日 | 同批次傳入 |
| 帳務日期\_迄日 | 同批次傳入 |
| 理賠受理編號 | IF 處理當筆.TA\_APLY是null 或 空值  SET ‘AA\_APLY\_NULL’  ELSE  SET 處理當筆.TA\_APLY  END IF |
| 理賠案件金額總合 | 處理當筆.AA\_AMT，若為null則set為0 |
| 帳務摘要 | IF 處理當筆.TB\_APLY是null 或 空值  SET ‘DK\_APLY\_NULL’  ELSE  SET 處理當筆.TB\_APLY  END IF |
| 帳務案件金額總合 | 處理當筆.DK\_AMT，若為null則set為0 |
| 金額差異 | 處理當筆.DIFF\_AMT，若為null則set為0 |
| 溢付扣回金額 | 處理當筆.OVER\_PAY\_AMT，若為null則set為0 |
| 比對不符代碼 | $錯誤碼 |

附錄一

|  |
| --- |
| with AA\_REC as (  select aply\_no ,sum(pay\_amt) as AA\_amt  from dbaa.dtaab001  where (clam\_amt\_code not in ('DCZ1' ,'JAZ3','CBW2','CBW9','JAJ1','JAL1','JAL2','JAZ4','JAZ5','EBX3','DEX4','BKZ2','BKZ5','FAX5','DBX6','GDXC','GCX7','BJ04', 'BJ05','JIXH','BJ03' ) )  and ACNT\_DATE BETWEEN '2017-01-01' and '2017-01-31'  and (pay\_sts in ('1','4','7') or (pay\_sts in ('2','3') and pay\_amt <> 0 ) )  and aply\_no not like 'OL%'  group by aply\_no  )  , DK\_REC as (  select F001.memo ,sum(case db\_cr when 'CR' then -amt else amt end) as DK\_amt  from dbdk.dtdkf001 F001  where acnt\_Date between '2017-01-01' and '2017-01-31'  and acnt\_code='5125000'  and TRN\_KIND like 'AA\_\_\_\_'  group by F001.memo  )  , AA\_BIG as (  select AA\_REC.aply\_no as ta\_aply,AA\_REC.aa\_amt,DK\_REC.memo as tb\_aply,DK\_REC.dk\_amt,AA\_REC.aa\_amt - DK\_REC.dk\_amt as ta\_amt  from AA\_REC left join DK\_REC  on AA\_REC.aply\_no = DK\_REC.memo  where AA\_REC.aa\_amt - DK\_REC.dk\_amt <> 0 or DK\_REC.memo is null  ), DK\_BIG as (  select AA\_REC.aply\_no as ta\_aply,AA\_REC.aa\_amt,DK\_REC.memo as tb\_aply,DK\_REC.dk\_amt,AA\_REC.aa\_amt - DK\_REC.dk\_amt as tb\_amt  from AA\_REC right join DK\_REC  on AA\_REC.aply\_no = DK\_REC.memo  where AA\_REC.aa\_amt - DK\_REC.dk\_amt <> 0 or AA\_REC.aply\_no is null  ) , t\_merg as (  select AA\_big.\* from AA\_big  union  select DK\_big.\* from DK\_big  ) , t\_dist as (  select distinct t\_merg.ta\_aply,t\_merg.aa\_amt,t\_merg.tb\_aply,t\_merg.dk\_amt,t\_merg.ta\_amt as diff\_amt  from t\_merg  ), chk\_ovrp as (  select distinct a.aply\_no ,a.policy\_no,a.prod\_id,a.clam\_amt\_code,a.ocr\_id,a.clam\_amt\_name,(b.pay\_amt\_diff-b.bal\_amt) as tot\_amt  from t\_dist  left join dbaa.dtaab001 a on t\_dist.ta\_aply = a.aply\_no  left join dbaa.dtaay001 b on a.aply\_no = b.ori\_aply\_no and a.policy\_no = b.policy\_no and a.ocr\_id = b.ocr\_id and a.prod\_id = b.prod\_id and a.clam\_amt\_code=b.clam\_amt\_code  where a.aply\_no is not null  ), t\_fin as (  select chk\_ovrp.aply\_no ,sum(tot\_amt) as over\_pay\_amt  from chk\_ovrp  group by chk\_ovrp.aply\_no  )  select t\_dist.\*,t\_fin.over\_pay\_amt  from t\_dist  left join t\_fin on t\_dist.ta\_aply = t\_fin.aply\_no  with ur |

附錄二[[BACK](#MARK2_BACK)]

|  |
| --- |
| select count(\*) as CNT  from DBAA.DTAAI001 AAI001  left join DBAA.DTAAB001 AAB001  on AAI001.APLY\_NO = AAB001.PREV\_APLY\_NO  and AAI001.POLICY\_NO = AAB001.POLICY\_NO  and AAI001.PROD\_ID = AAB001.PROD\_ID  and AAI001.OCR\_ID = AAB001.OCR\_ID  where AAI001.APLY\_NO='D220313273P001'  and AAB001.APLY\_NO IS NOT NULL  and AAI001.ADV\_AMT + AAB001.PAY\_AMT = 0  with ur |

附錄三[[BACK](#MARK3_BACK)]

|  |
| --- |
| with AI\_FIX as (  select DISTINCT Z001.POLICY\_NO,  DATE(Z001.INPUT\_DATE) AS INPUT\_DATE,  Z000.RTN\_ADD\_IND,  Z000.RTN\_ADD\_AMT,  Z000.RTN\_RCPT\_NO,  Z000.RTN\_SLIP\_DATE,  Z000.RTN\_SLIP\_DIV,  Z000.FIX\_KIND,  DATE(Z000.IT\_APRV\_DATE) AS FIN\_FIX\_DATE  from DBAI.DTAIZ001 Z001  left join DBAI.DTAIZ000 Z000 on Z001.case\_no = Z000.case\_no  where DATE(Z000.IT\_APRV\_DATE) BETWEEN ':START\_DATE' and ':END\_DATE'  and Z001.POLICY\_NO in (SELECT distinct policy\_no from dbaa.dtaab100 where aply\_no=':APLY\_NO')  and Z000.FIX\_KIND like 'AA%')  select sum(RTN\_ADD\_AMT) as SUM\_AMT  from AI\_FIX  with ur |

附錄四[[BACK](#MARK4_BACK)]

|  |
| --- |
| select AAI001.APLY\_NO,  sum(ADV\_AMT) as PRE\_PAY\_AMT  from DBAA.DTAAI001 AAI001  where AAI001.APLY\_NO=':APLY\_NO'  group by AAI001.APLY\_NO  with ur |

附錄五[[BACK](#MARK5_BACK)]

|  |
| --- |
| select SUM(Y001.PAY\_AMT\_DIFF - Y001.BAL\_AMT) as BACK\_AMT  from DBAA.DTAAB001 A001  left join DBAA.DTAAY001 Y001  on A001.APLY\_NO = Y001.ORI\_APLY\_NO  and A001.SER\_NO = Y001.SER\_NO  and A001.POLICY\_NO = Y001.POLICY\_NO  and A001.PROD\_ID = Y001.PROD\_ID  and A001.OCR\_ID = Y001.OCR\_ID  and A001.CLAM\_AMT\_CODE = Y001.CLAM\_AMT\_CODE  where A001.APLY\_NO = ':APLY\_NO'  and Y001.ORI\_APLY\_NO IS NOT NULL  with ur |

附錄六[[BACK](#MARK6_BACK)]

|  |
| --- |
| WITH AI\_FIX  AS (SELECT DISTINCT Z001.POLICY\_NO,  DATE (Z001.INPUT\_DATE) AS INPUT\_DATE,  Z000.RTN\_ADD\_IND,  Z000.RTN\_ADD\_AMT,  Z000.RTN\_RCPT\_NO,  Z000.RTN\_SLIP\_DATE,  Z000.RTN\_SLIP\_DIV,  Z000.FIX\_KIND,  Z000.TEMP\_PRCS\_AMT,  DATE (Z000.IT\_APRV\_DATE) AS FIN\_FIX\_DATE  FROM DBAI.DTAIZ001 Z001  LEFT JOIN  DBAI.DTAIZ000 Z000  ON Z001.case\_no = Z000.case\_no  WHERE DATE(Z000.IT\_APRV\_DATE) BETWEEN ':START\_DATE' AND ':END\_DATE'  AND Z001.POLICY\_NO IN (SELECT DISTINCT policy\_no  FROM dbaa.dtaab100  WHERE aply\_no = ':APLY\_NO')  AND Z000.FIX\_KIND LIKE 'AA%')  ,DK\_FIX as (  SELECT B.CE\_SLP\_DT , B.CE\_DIV\_NO , B.CE\_SLPSEQ\_NO , B.CE\_CURR\_TYPE , B.CE\_ACC\_CD , B.CE\_SBACC\_CD , B.CE\_DLACC\_CD , B.CE\_DB\_CR, B.CE\_AMT , B.CE\_MEMO  FROM AI\_FIX  LEFT JOIN DBDK.DTDK\_B110\_SLIP\_CONTROL A  ON A.CE\_SLP\_DT =AI\_FIX.RTN\_SLIP\_DATE  AND A.CE\_SLPSEQ\_NO = AI\_FIX.RTN\_RCPT\_NO  AND A.CE\_DIV\_NO = AI\_FIX.RTN\_SLIP\_DIV  AND A.CE\_CURR\_TYPE = AI\_FIX.RTN\_CURR  LEFT JOIN DBDK.DTDK\_B111\_SLIP\_DETAIL B  ON A.CE\_SLP\_DT = B.CE\_SLP\_DT  AND A.CE\_SLPSEQ\_NO = B.CE\_SLPSEQ\_NO  AND A.CE\_DIV\_NO = B.CE\_DIV\_NO  AND A.CE\_CURR\_TYPE = B.CE\_CURR\_TYPE  WHERE A.CE\_REJAC\_DT IS NULL AND A.CE\_REJCH\_DT IS NULL AND CE\_DEL\_DT IS NULL  AND A.CE\_SLPSET\_NO=0  AND substr(B.CE\_DLACC\_CD,1,2) in ('02','03','05','09','10','11')  AND B.CE\_SBACC\_CD IN (  SELECT CE\_SBACC\_CD  FROM DBDK.DTDK\_E330\_POLICY\_PROPERTYSETTING) )  ,AI\_SUM\_AMT AS (  SELECT sum(RTN\_ADD\_AMT + TEMP\_PRCS\_AMT) AS AI\_AMT  FROM AI\_FIX)  ,DK\_SUM\_AMT AS (  SELECT sum(CE\_AMT) AS DK\_AMT  FROM DK\_FIX)  SELECT count(\*) as CNT  FROM AI\_SUM\_AMT  LEFT JOIN DK\_SUM\_AMT  ON AI\_AMT = DK\_AMT  WHERE DK\_AMT is not null  WITH UR |