```
SET ECHO       off;
SET FEEDBACK   On;
SET VERIFY     On;
SET TERMOUT    On;
SET SERVEROUT  ON SIZE 500000;
-- ************************************************************************
-- PROGRAM HISTORY                                                      *
--                                                                      *
-- WCUNEX_STATS_C.SQL                                                   *
-- THIS PROGRAM PROCESSES THE DATA IN TEMP_UNIT_EXPO THAT HAS NOT ALREADY   *
-- BEEN PROCESSED BY WCUNEX_STATS_1111.  PERIOD SPLITS ARE
FORCED/ELIMINATED*
-- BASED ON AVAILABLE CRITERIA.  DATA IS MOVED TO UNIT_STATS_EXPOSURE       *
--                                                                      *
--                                                                      *
-- PROJECT.TASK     DATE          TECH.     ACTION                      *
-- ------------     -----------    -----     ------------------------------*
-- PLI32502.12477   30-JUN-2009    JPC       INITIAL CREATION              *
--                                                                      *
-- PLI32502.15895   18-SEP-2009    JPC       Altered logic to incl. 0990,   *
--                                 0900 while using 1111          *
--                                                                      *
-- PLI32502.16827   04-JAN-2010    JPC       Set troubleshooting displays   *
--                                                                      *
-- PLI32502.16866   06-JAN-2010    JPC       Add sqlcode for error-trapping.*
--                                 Add WCX synonym to all elements*
--                                 based in PK_WCUNEX_PACK        *
--                                                                      *
-- PLI32502.17366   23-FEB-2010    JPC       Fixed date assignment for non- *
--                                 anniversary split periods      *
--                                                                      *
-- PLI32502.17531   08-MAR-2010    JPC       Changed split determination;   *
--                                 added handling for 9752        *
--                                                                      *
-- PLI32502.18839   06-JUL-2010    JPC       Fix sequencing in WCUNEX3_09C  *
--                                                                      *
-- PLI32502.19870   05-JAN-2011    JPC       Add exposure transmit indicator*
--                                                                      *
-- PLI32502.21102   23-MAY-2011    JPC       Remove sequence; clear variable*
--                                                                      *
-- PLI32502.21180   07-JUN-2011    JPC       Replace missing END-IF         *
--                                                                      *
-- PLI32502.21210   10-JUN-2011    JPC       Add Determine_Split_Period     *
-- PLI32502.21257   20-APR-2012    JPC       Add diagnostics                *
-- PLI32502.23617   08-AUG-2012    JPC       Remove repetitive displays     *
-- PLI32502.23844   21-SEP-2012    JPC       Add NJ to anniv date logic     *
--                                                                      *
-- PLI32502.24578   23-JAN-2013    JPC       deviated px correction         *
-- PLI32502.27146   18-MAR-2014    JPC       Move DETERMINE_AFTERMOD (12b)  *
-- PLI32502.22646   11-APR-2014    JPC       Do not report 0063 with 0 px   *
-- PLI32502.22838   24-APR-2014    JPC       Recombine 0935 and 0936        *
-- PLI32502.21882   25-APR-2014    JPC       Modify split dates for MN      *
-- PLI32052.26375   25-APR-2014    JPC       Fix splits for PA              *
-- PLI32502.27310   27-MAY-2014    JPC       expand if/then logic for split *
--                                          indicator *
-- PLI32502.27714   29-MAY-2014    JPC       Various NY fixes               *
```

```
-- PLI32502.28654   20-NOV-2014    JPC       CA 0930 corrections           *
-- PLI32502.30030   20-AUG-2015    JPC       add premium check to GET_POLICY*
-- ********************************************************************************
-- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
-- DECLARE SECTION
-- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
DECLARE
      WS_COUNT NUMBER;
      WS_COUNT2 NUMBER;
      WS_COUNT3 NUMBER;
      WS_UNPL_ID UNEX.UNEX_UNPL_WC_NO%TYPE;
      WS_POLICY TUNX.POLICY_NUMBER%TYPE;
      WS_CLASS UNEX.UNEX_WC_EXIT_CLASS_CODE%TYPE;
      WS_HOLD_CLASS UNEX.UNEX_WC_EXIT_CLASS_CODE%TYPE;
      WS_NETPX UNEX.UNEX_WC_EXIT_NET_PREM%TYPE;
      WS_STDPX TUNX.STANDARD_PREMIUM%TYPE;
      WS_WRTPX TUNX.WRITTEN_PREMIUM%TYPE;
      WS_MODPX UNEX.UNEX_WC_EXIT_MOD_NET_PREM%TYPE;
      WS_PAYROLL TUNX.PAYROLL_AMOUNT%TYPE;
      WS_STATE_NO UNEX.UNEX_STAT_GB_STATE_NO%TYPE;
      WS_UNEX_WC_NO UNEX.UNEX_WC_NO%TYPE;
      HOLD_UNEX_ID UNEX.UNEX_WC_NO%TYPE;
      WS_SPLITNO UNEX.UNEX_WC_SPLIT_PERIOD_IND%TYPE;
      WS_SPLIT_MARKER UNEX.UNEX_WC_SPLIT_PERIOD_IND%TYPE;
      WS_AFTERMOD UNEX.UNEX_WC_RECORD_TYPE%TYPE;
      WS_MODVAL UNEX.UNEX_WC_EXP_MOD%TYPE;
      WS_MODEFF UNEX.UNEX_WC_EXP_MOD_EFF_DATE%TYPE;
      WS_BURT UNEX.UNEX_WC_BURT_RATE%TYPE;
      WS_BURT_EFF UNEX.UNEX_WC_BURT_EFF_DATE%TYPE;
      WS_HOLD_MODVAL UNEX.UNEX_WC_EXP_MOD%TYPE;
      WS_TOTAL_PX       NUMBER;
      WS_TOTAL_EXPOSURE NUMBER;
      WS_MINEFF DATE;
      WS_MAXEFF DATE;
      WS_MINMOD DATE;
      WS_MAXMOD DATE;
      WS_POLEFF DATE;
      WS_EXPOEFF DATE;
      WS_EXPOEFF2 DATE;
      WS_ANNDATE DATE;
      WS_REPORTDATE DATE;
      VAR_POLICY_IN RATP.POLICY_NUM%TYPE;
      VAR_DATE_IN DATE;
      WS_SECTION  VARCHAR2 (12) ;
      VAR_START   VARCHAR2 (8) ;
      VAR_STOP    VARCHAR2 (8) ;
      VAR_BATCH   VARCHAR2 (1) ;
      WS_PROC_IND VARCHAR2 (1) ;
      WS_1111_IND NUMBER;
      WS_XMIT     unex.UNEX_XMIT_IND%type;
      WS_SQLCODE  NUMBER;
      STOP_PROCESSING varchar2(1) := 'N';
      -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
      -- DECLARE CURSORS
      -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
          CURSOR GET_POLICY
```

```
112         IS
113           SELECT DISTINCT
114             MAX (UNPL.UNPL_WC_NO)          AS UNPLID,
115             TUNX.POLICY_NUMBER             AS POL_1 ,
116             MAX (UNPL.UNPL_WC_POLY_EFF_DATE) AS POLEFF,
117             TUNX.ANNIV_DTE                 AS ANNDATE
118           FROM
119             TEMP_UNIT_EXPO TUNX,
120             UNIT_STATS_POLICY UNPL
121           WHERE
122             TUNX.POLICY_NUMBER                                        =
                UNPL.UNPL_WC_POLY_NUMBER
123           AND MONTHS_BETWEEN (WS_REPORTDATE, LAST_DAY
              (UNPL.UNPL_WC_POLY_EFF_DATE)) = 18
124           AND TUNX.STATE_NUMBER                                       =  WS_STATE_NO
125           GROUP BY
126             TUNX.POLICY_NUMBER,
127             TUNX.ANNIV_DTE
128           HAVING
129             SUM(MOD_PREMIUM) > 0
130           ORDER BY
131             TUNX.POLICY_NUMBER DESC ;
132       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
133       CURSOR GET_EXPO_A
134       IS
135           SELECT DISTINCT CLASS_CODE         AS CLASSCODE,
136               EXPOSURE_EFFECTIVE           AS EXPOEFF  ,
137               TUNX.EXPOSURE_EFFECTIVE2     AS EXPOEFF2 ,
138               TUNX.USE_MOD_IND             AS AFTER_MOD,
139               NVL (TUNX.MODIFIER_VALUE, 0) AS MODVAL   ,
140               TUNX.MULTIPLIER AS BURT_RATE,
141               TUNX.MULTIPLIER_EFFECTIVE    AS BURT_EFF
142             FROM TEMP_UNIT_EXPO TUNX
143            WHERE TUNX.POLICY_NUMBER   = WS_POLICY
144               AND TUNX.STATE_NUMBER = WS_STATE_NO
145               AND SUBSTR (TUNX.CLASS_CODE, 1, 1) BETWEEN ('0') AND ('9')
146               AND TUNX.CLASS_CODE NOT IN ('0932', '9749')
147           GROUP BY TUNX.CLASS_CODE         ,
148               TUNX.EXPOSURE_EFFECTIVE  ,
149               TUNX.MODIFIER_VALUE      ,
150               TUNX.USE_MOD_IND         ,
151               TUNX.MULTIPLIER          ,
152               TUNX.MULTIPLIER_EFFECTIVE,
153               TUNX.EXPOSURE_EFFECTIVE2
154           ORDER BY TUNX.CLASS_CODE;
155       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
156       CURSOR GET_EXPO_B
157       IS
158           SELECT CLASS_CODE                AS CLASSCODE,
159               TUNX.USE_MOD_IND             AS AFTER_MOD,
160               NVL (TUNX.MODIFIER_VALUE, 0) AS MODVAL   ,
161               --decode(tunx.class_code, '9807', 0, TUNX.MULTIPLIER) AS BURT_RATE,
162               TUNX.MULTIPLIER AS BURT_RATE,
163               DECODE (CLASS_CODE, '0900', EXPOSURE_EFFECTIVE, '9740',
                  EXPOSURE_EFFECTIVE, '9741', EXPOSURE_EFFECTIVE,
                  TUNX.EXPOSURE_EFFECTIVE2) AS BURT_EFF
```

```
164             FROM TEMP_UNIT_EXPO TUNX
165             WHERE TUNX.POLICY_NUMBER   = WS_POLICY
166                 AND TUNX.STATE_NUMBER = WS_STATE_NO
167                 AND SUBSTR (TUNX.CLASS_CODE, 1, 1) BETWEEN ('0') AND ('9')
168                 AND TUNX.CLASS_CODE NOT IN ('0932', '9749')
169             GROUP BY TUNX.CLASS_CODE   ,
170                 TUNX.MODIFIER_VALUE,
171                 TUNX.USE_MOD_IND   ,
172                 TUNX.MULTIPLIER    ,
173                 DECODE (CLASS_CODE, '0900', EXPOSURE_EFFECTIVE, '9740',
                    EXPOSURE_EFFECTIVE, '9741', EXPOSURE_EFFECTIVE,
                    TUNX.EXPOSURE_EFFECTIVE2);
174         -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
175         CURSOR GET_EXPO_C
176         IS
177             SELECT CLASS_CODE                   AS CLASSCODE,
178                 TUNX.USE_MOD_IND               AS AFTER_MOD,
179                 MAX (NVL (TUNX.MODIFIER_VALUE, 0)) AS MODVAL   ,
180                 TUNX.MULTIPLIER AS BURT_RATE,
181                 TUNX.POLICY_EFF                AS BURT_EFF
182             FROM TEMP_UNIT_EXPO TUNX
183             WHERE TUNX.POLICY_NUMBER   = WS_POLICY
184                 AND TUNX.STATE_NUMBER = WS_STATE_NO
185                 AND SUBSTR (TUNX.CLASS_CODE, 1, 1) BETWEEN ('0') AND ('9')
186                 AND TUNX.CLASS_CODE NOT IN ('0932', '9749')
187             GROUP BY TUNX.CLASS_CODE,
188                 TUNX.USE_MOD_IND,
189                 TUNX.MULTIPLIER ,
190                 TUNX.POLICY_EFF;
191
192         -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
193         CURSOR GET_TOTALS
194         IS
195             SELECT SUM (DISTINCT STANDARD_PREMIUM) AS STDPX,
196                 SUM (DISTINCT MOD_PREMIUM)      AS MODPX,
197                 SUM (DISTINCT NET_PREMIUM)      AS NETPX,
198                 SUM (DISTINCT WRITTEN_PREMIUM)  AS WRTPX,
199                 SUM (DISTINCT PAYROLL_AMOUNT)   AS PYRLL
200             FROM TEMP_UNIT_EXPO TUNX
201             WHERE TUNX.POLICY_NUMBER        = WS_POLICY
202                 AND TUNX.CLASS_CODE        = WS_CLASS
203                 AND TUNX.EXPOSURE_EFFECTIVE = WS_EXPOEFF
204                 AND TUNX.STATE_NUMBER      = WS_STATE_NO
205             GROUP BY TUNX.EXPOSURE_EFFECTIVE ;
206         -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
207         CURSOR GET_TOTALS_B
208         IS
209             SELECT SUM (DISTINCT STANDARD_PREMIUM) AS STDPX,
210                 SUM (DISTINCT MOD_PREMIUM)      AS MODPX,
211                 SUM (DISTINCT NET_PREMIUM)      AS NETPX,
212                 SUM (DISTINCT WRITTEN_PREMIUM)  AS WRTPX,
213                 SUM (DISTINCT PAYROLL_AMOUNT)   AS PYRLL
214             FROM TEMP_UNIT_EXPO TUNX
215             WHERE TUNX.POLICY_NUMBER           = WS_POLICY
216                 AND TUNX.CLASS_CODE           = WS_CLASS
217                 and tunx.EXPOSURE_EFFECTIVE2    = WS_BURT_EFF
```

```
218                 AND TUNX.STATE_NUMBER          = WS_STATE_NO
219                 AND NVL (TUNX.MODIFIER_VALUE, 0) = WS_MODVAL
220             GROUP BY SPLIT_IND;
221       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
222       CURSOR GET_TOTALS_C
223       IS
224             SELECT ROUND (SUM (TUNX.STANDARD_PREMIUM)) AS STDPX,
225                 ROUND (SUM (TUNX.MOD_PREMIUM))       AS MODPX,
226                 ROUND (SUM (TUNX.NET_PREMIUM))       AS NETPX,
227                 ROUND (SUM (TUNX.WRITTEN_PREMIUM))   AS WRTPX,
228                 ROUND (SUM (TUNX.PAYROLL_AMOUNT))    AS PYRLL
229               FROM TEMP_UNIT_EXPO TUNX
230               WHERE TUNX.POLICY_NUMBER   = WS_POLICY
231                 AND TUNX.CLASS_CODE   = WS_CLASS
232                 AND TUNX.STATE_NUMBER = WS_STATE_NO
233                 group by state_abbrv;
234       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
235       CURSOR FIND_STATES
236       IS
237             SELECT STAT_GB_STATE_NO STATE_NO
238               FROM STAT
239               WHERE STAT_GB_STATE_NO NOT IN ('72', '71', '70', '64', '63', '62', '60', '58', '57', '56',
                  '55', '53', '51')
240             ORDER BY STAT_GB_STATE_NAME DESC;
241       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
242       --  GET LOGICALS
243       -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
244    BEGIN
245          SELECT TO_CHAR (SYSDATE, 'HH24:MI:SS')
246          INTO VAR_START
247          FROM DUAL;
248
249       WS_SECTION := 'WCUNEX3_00A';
250        SELECT UPPER (UETB.UETB_DP_VALUE) INPUT_SRC
251          INTO VAR_BATCH
252          FROM USER_EXIT_TABLE UETB
253         WHERE UETB.UETB_DP_FIELD_NAME      = 'RPT_OPT'
254             AND UETB.UETB_DP_PROGRAM_NAME = 'WCUNEX_STATS'
255             AND UETB.UETB_DP_USER_NAME    = USER;
256       --
257       WS_SECTION := 'WCUNEX3_00B';
258        SELECT UPPER (UETB.UETB_DP_VALUE) REPORT_DTE
259          INTO VAR_DATE_IN
260          FROM USER_EXIT_TABLE UETB
261         WHERE UETB.UETB_DP_FIELD_NAME      = 'ENDDATE'
262             AND UETB.UETB_DP_PROGRAM_NAME = 'WCUNEX_STATS'
263             AND UETB.UETB_DP_USER_NAME    = USER;
264       --
265       WS_SECTION := 'WCUNEX3_00C';
266        SELECT UPPER (UETB.UETB_DP_VALUE) POL_NUMBER
267          INTO VAR_POLICY_IN
268          FROM USER_EXIT_TABLE UETB
269         WHERE UETB.UETB_DP_FIELD_NAME      = 'DATAOPT'
270             AND UETB.UETB_DP_PROGRAM_NAME = 'WCUNEX_STATS'
271             AND UETB.UETB_DP_USER_NAME    = USER;
272       --
```

```
273     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
274     WS_REPORTDATE := VAR_DATE_IN;
275     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
276     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
277     --
278     -- MAIN PROGRAM
279     --
280     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
281     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
282     FOR REC1 IN FIND_STATES
283     LOOP
284         WS_SECTION  := 'WCUNEX3_00D';
285
286
287
288         WS_STATE_NO := REC1.STATE_NO;
289         WS_HOLD_CLASS := null;
290         --
291         FOR REC2 IN GET_POLICY
292         LOOP
293             WS_SECTION    := 'WCUNEX3_01 ';
294             WS_UNPL_ID    := REC2.UNPLID;
295             WS_POLICY     := REC2.POL_1;
296             WS_ANNDATE    := REC2.ANNDATE;
297             WS_POLEFF     := REC2.POLEFF;
298             VAR_POLICY_IN := WS_POLICY;
299
300             -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
301             -- AN ANNIVERSARY DATE IS A LEGITIMATE CAUSE FOR A SPLIT PERIOD.
302             -- EACH PERIOD NEEDS TO BE TOTALLED SEPERATELY
303             -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
304             IF (WS_ANNDATE  IS NOT NULL AND WS_STATE_NO <> '22') THEN -- ANNDATE
                IF
305                 FOR REC3 IN GET_EXPO_A
306                 LOOP
307                     WS_HOLD_MODVAL := WS_MODVAL;
308                     WS_SECTION     := 'WCUNEX3_02 ';
309                     WS_CLASS       := REC3.CLASSCODE;
310                     WS_EXPOEFF     := REC3.EXPOEFF;
311                     WS_EXPOEFF2    := REC3.EXPOEFF2;
312                     WS_AFTERMOD    := REC3.AFTER_MOD;
313                     WS_MODVAL      := REC3.MODVAL;
314                     WS_BURT        := REC3.BURT_RATE;
315                     WS_BURT_EFF    := REC3.BURT_EFF;
316                     --
317                     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
318                     WS_SECTION := 'WCUNEX3_03 ';
319                      SELECT MIN (TUNX.EXPOSURE_EFFECTIVE) AS MINEFF,
320                          MAX (TUNX.EXPOSURE_EFFECTIVE)  AS MAXEFF,
321                          MIN (TUNX.MOD_EFFECTIVE)       AS MINMOD,
322                          MAX (TUNX.MOD_EFFECTIVE)       AS MAXMOD
323                       INTO WS_MINEFF,
324                          WS_MAXEFF ,
325                          WS_MINMOD ,
326                          WS_MAXMOD
327                       FROM TEMP_UNIT_EXPO TUNX
```

```
328                      WHERE TUNX.POLICY_NUMBER = WS_POLICY
329                          AND TUNX.CLASS_CODE = WS_CLASS;
330                      --
331                      -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
332                                              WS_SPLIT_MARKER := 'N';
333                      IF WS_STATE_NO              in ('20','29') THEN
334                          IF WS_MINEFF       = WS_EXPOEFF OR WS_HOLD_CLASS <>
                         WS_CLASS THEN
335                              WS_SECTION := 'WCUNEX3_04A';
336                              WS_SPLITNO := 0;
337                              WS_MODEFF  := WS_ANNDATE;
338                          ELSE
339                              WS_SECTION      := 'WCUNEX3_05A';
340                              WS_SPLITNO      := 1;
341                              WS_SPLIT_MARKER := 'Y';
342
343                              SELECT ADD_MONTHS (WS_ANNDATE, 12) INTO WS_MODEFF
                                 FROM DUAL;
344                          END IF;
345                      ELSE -- WS_STATE_NO <> '20'
346                          IF WS_MINEFF       = WS_EXPOEFF OR WS_HOLD_CLASS <>
                         WS_CLASS THEN
347                              WS_SECTION := 'WCUNEX3_04B';
348                              WS_SPLITNO := 0;
349                              WS_MODEFF  := WS_MINEFF;
350
351
352                          ELSE
353                              WS_SECTION      := 'WCUNEX3_05B';
354                              WS_SPLITNO      := 1;
355                              WS_SPLIT_MARKER := 'Y';
356                      --
357                               SELECT ADD_MONTHS (WS_ANNDATE, 12)
358                               INTO WS_MODEFF
359                               FROM DUAL;
360
361                      --
362                              WS_BURT_EFF := WS_MODEFF;
363
364                          END IF;
365                      END IF;
366                      -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
367                      WS_SECTION                := 'WCUNEX3_06 ';
368                      IF WS_STATE_NO              = '20' THEN
369                          WS_HOLD_CLASS       := WS_CLASS;
370                          IF WS_CLASS          = WS_HOLD_CLASS THEN
371                              IF WS_MINEFF <> WS_EXPOEFF2 THEN
372                                  SELECT ADD_MONTHS (WS_ANNDATE, 12) INTO
                                     WS_MODEFF FROM DUAL;
373
374
375                                  WS_SPLITNO      := 1;
376                                  WS_SPLIT_MARKER := 'Y';
377                              ELSE
378                                  WS_SECTION := 'WCUNEX3_07 ';
379                                  WS_SPLITNO := 0;
```

```
                        WS_MODEFF  := WS_ANNDATE;

                    END IF;
                END IF;
            END IF; --WS_STATE_NO = '20'

            -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
            --
            FOR REC4 IN GET_TOTALS
            LOOP
                WS_SECTION := 'WCUNEX3_08 ';
                WS_STDPX   := REC4.STDPX;
                WS_MODPX   := REC4.MODPX;
                WS_NETPX   := REC4.NETPX;
                WS_WRTPX   := REC4.WRTPX;
                WS_PAYROLL := REC4.PYRLL;
                WS_AFTERMOD :=   WCX.DETERMINE_AFTERMOD
                (WS_AFTERMOD, WS_CLASS, WS_STATE_NO, WS_STDPX);
                -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
                WS_SECTION := 'WCUNEX3_09 ';
                -- ONLY DE AND PA REQUIRE THESE CODES TO SPLIT; ALL
                OTHERS STATES COMBINE
                IF WS_CLASS NOT IN ('0900', '0063', '9740', '9741', '0277', '9722',
                '9752','0935', '0936') THEN
                    WCX.TEMP_UPDATE_TUNX (WS_SPLITNO, WS_POLICY,
                    WS_EXPOEFF, WS_SQLCODE) ;
                    COMMIT;
                    if ws_sqlcode not in (0,1) THEN
                        dbms_output.put_line('ERROR     : '||WS_SECTION||'
                        '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                        '||WS_SPLIT_MARKER);
                    end if;
                    --
                    --
                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                    =)
                    WS_SECTION    := 'WCUNEX3_10 ';
                    IF WS_BURT_EFF = WS_EXPOEFF2 THEN
                        SELECT UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                        INTO WS_UNEX_WC_NO FROM DUAL;
                        --
                        (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                        (=)=(=)
                        WS_SECTION := 'WCUNEX3_11 ';

                        IF NVL (WS_MODVAL, 0) = 0 THEN
                            WS_MODPX      := WS_NETPX;
                        END IF;
                        --
                        (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                        (=)=(=)
                        -- INSERT INTO UNIT_STATS_EXPOSURE
                        --
                        (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                        (=)=(=)
                        --
```

```
421                     WS_SECTION := 'WCUNEX3_11A';
422                             WS_TOTAL_PX := WCX.CALCULATE_TTL_PREM
                                (WS_NETPX, WS_MODPX);
423                     --
424                     --
425                     WS_SECTION := 'WCUNEX3_11b';
426                     WS_TOTAL_EXPOSURE:=
                        WCX.CALCULATE_TTL_EXPOSURE (WS_TOTAL_PX,
                        WS_PAYROLL);
427                     --
428                     WS_XMIT :=
                        WCX.GET_EXPO_XMIT(WS_CLASS,WS_NETPX,WS_STATE_
                        NO);
429                         WS_MODEFF :=
                            WCX.Determine_Split_Period(WS_CLASS,WS_POLICY,WS
                            _SPLITNO,WS_STATE_NO);
430
431                     IF WS_MODVAL       > 0 THEN
432                         WS_SECTION := 'WCUNEX3_11C';
433                         WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                            WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                            WS_CLASS, WS_MODVAL, WS_MODEFF,
                            WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                            WS_MODPX, WS_SPLITNO, WS_BURT,
                            WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                            WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                            WS_SECTION, WS_XMIT, WS_SQLCODE) ;
434
435                              if ws_sqlcode not in (0,1) THEN
436                                  dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                    '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                    '||WS_SPLIT_MARKER);
437                              END IF;
438                     ELSIF WS_MODVAL    = 0 AND WS_CLASS NOT IN ('0990',
                        '9874', '9885', '9886', '9887', '9889') THEN
439                         WS_SECTION := 'WCUNEX3_11D';
440                         WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                            WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                            WS_CLASS, WS_MODVAL, WS_MODEFF,
                            WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                            WS_MODPX, WS_SPLITNO, WS_BURT,
                            WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                            WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                            WS_SECTION, WS_XMIT, WS_SQLCODE) ;
441
442                              if ws_sqlcode not in (0,1) THEN
443                                  dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                    '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                    '||WS_SPLIT_MARKER);
444                              END IF;
445
446                     ELSIF WS_MODVAL    = 0 AND WS_CLASS IN ('0990', '9874',
                        '9885', '9886', '9887', '9889') THEN
447                         WS_SECTION := 'WCUNEX3_11e';
```

```
448                                     WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                        WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                        WS_CLASS, WS_MODVAL, WS_MODEFF,
                                        WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                                        WS_MODPX, WS_SPLITNO, WS_BURT,
                                        WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                        WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                                        WS_SECTION, WS_XMIT, WS_SQLCODE) ;
449
450                                      if ws_sqlcode not in (0,1) THEN
451                                         dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                            '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                            '||WS_SPLIT_MARKER);
452                                      END IF;
453                                 ELSE
454                                      WS_SECTION := 'WCUNEX3_11K';
455                                      IF WS_SQLCODE NOT IN (0,1) THEN
456                                         DBMS_OUTPUT.PUT_LINE('ERROR     :
                                            '||WS_SECTION||' '||WS_UNPL_ID||' '||WS_CLASS||'
                                            '||WS_SPLITNO||' '||WS_SPLIT_MARKER);
457                                      END IF;
458                                 END IF;
459                                 --
460                            END IF; --(IF WS_BURT_EFF     = WS_EXPOEFF2 THEN)
461
462                             --
                                (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                                (=)
463                        ELSE --(WS_CLASS = '0900', '0063', '9740', '9741','0277','9722','9752')
464                             WS_SECTION         := 'WCUNEX3_09A';
465
466                             IF WS_STATE_NO NOT IN ('07', '37') THEN
467                                  SELECT SUM (DISTINCT TUNX.STANDARD_PREMIUM) AS
                                     STDPX,
468                                      SUM (DISTINCT TUNX.MOD_PREMIUM)      AS MODPX,
469                                      SUM (DISTINCT TUNX.NET_PREMIUM)      AS NETPX,
470                                      SUM (DISTINCT TUNX.WRITTEN_PREMIUM)   AS WRTPX,
471                                      SUM (DISTINCT TUNX.PAYROLL_AMOUNT)    AS PYRLL
472                                    INTO WS_STDPX,
473                                      WS_MODPX ,
474                                      WS_NETPX ,
475                                      WS_WRTPX ,
476                                      WS_PAYROLL
477                                    FROM TEMP_UNIT_EXPO TUNX
478                                   WHERE TUNX.POLICY_NUMBER   = WS_POLICY
479                                      AND TUNX.CLASS_CODE   = WS_CLASS
480                                      AND TUNX.STATE_NUMBER = WS_STATE_NO ;
481
482                                 -- if there is a split, these class codes need to be on the second
                                    period only
483                                 --
                                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                                    (=)=(=)
484                                 WS_SECTION := 'WCUNEX3_09B';
485
```

```
486                     WCX.TEMP_UPDATE_TUNX (WS_SPLITNO, WS_POLICY,
                        WS_EXPOEFF, WS_SQLCODE) ;
487                     COMMIT;
488                      IF WS_SQLCODE NOT IN (0,1) THEN
489                         DBMS_OUTPUT.PUT_LINE('ERROR     :
                            '||WS_SECTION||' '||WS_UNPL_ID||' '||WS_CLASS||'
                            '||WS_SPLITNO||' '||WS_SPLIT_MARKER);
490                      END IF;
491                     --
492                     IF WS_POLICY = VAR_POLICY_IN THEN
493                        SELECT
                           UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                           INTO WS_UNEX_WC_NO FROM DUAL;
494                     END IF;
495                     IF WS_BURT_EFF     = WS_EXPOEFF2 THEN
496                        WS_SECTION := 'WCUNEX3_10B';
497                        --
498                        SELECT
                           UNIT_STATS_EXPOSURE_SEQUENCE.CURRVAL
                           INTO WS_UNEX_WC_NO FROM DUAL;
499
500                        WS_SECTION          := 'WCUNEX3_10C';
501                        HOLD_UNEX_ID         := WS_UNEX_WC_NO;
502                        WS_SECTION          := 'WCUNEX3_10D';
503                        IF WS_POLICY         = VAR_POLICY_IN THEN
504                           WS_UNEX_WC_NO := HOLD_UNEX_ID;
505                        END IF;
506                        IF WS_SPLITNO              = 0 THEN
507                           WS_SECTION          := 'WCUNEX3_11e';
508                           IF NVL (WS_MODVAL, 0) = 0 THEN
509                              WS_MODPX      := WS_NETPX;
510                           END IF;
511                           --
                              (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
                              =(=)=(=)=(=)=(=)
512                           -- INSERT INTO UNIT_STATS_EXPOSURE
513                           --
                              (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
                              =(=)=(=)=(=)=(=)
514                           WS_SECTION := 'WCUNEX3_11f';
515                           --
516                           WS_TOTAL_PX := WCX.CALCULATE_TTL_PREM
                              (WS_NETPX, WS_MODPX);
517
518                           --
519                           WS_SECTION := 'WCUNEX3_11g';
520                           WS_TOTAL_EXPOSURE:=
                              WCX.CALCULATE_TTL_EXPOSURE
                              (WS_TOTAL_PX, WS_PAYROLL);
521                           --
522                           WS_SECTION := 'WCUNEX3_11h1';
523                           --
524                           IF WS_CLASS              IN ('0900', '0063', '9740', '9741',
                              '0277',  '9722', '9752', '0935', '0936') then   -- 9807
                              removed from list per hdr15683
525                              IF WS_SPLIT_MARKER = 'Y' THEN
```

```
526                                         WS_SPLITNO := 0;
527                                     END IF;
528                                 END IF;
529                             WS_MODEFF :=
                                WCX.Determine_Split_Period(WS_CLASS,WS_POLICY,WS
                                _SPLITNO,WS_STATE_NO);
530                                 --
531                                WS_SECTION := 'WCUNEX3_11h2';
532                             WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                WS_CLASS, WS_MODVAL, WS_MODEFF,
                                WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                                WS_MODPX, WS_SPLITNO, WS_BURT,
                                WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                                WS_SECTION, WS_XMIT, WS_SQLCODE) ;
533
534                              if ws_sqlcode not in (0,1) THEN
535                                 dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                    '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                    '||WS_SPLIT_MARKER);
536                              END IF;
537
538                                 --
                                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
                                    =(=)=(=)=(=)=(=)
539                             END IF; -- WS_SPLITNO  = 0
540                         END IF;       -- WS_BURT_EFF = WS_EXPOEFF2
541                     ELSE                -- STATE NO = 07 OR 37
542                         WS_SECTION := 'WCUNEX3_09C';
543
544                         WS_MODEFF  := WS_ANNDATE;
545
546                         --
547                         SELECT UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                            INTO WS_UNEX_WC_NO FROM DUAL;
548                         --
549                         WS_XMIT :=
                            WCX.GET_EXPO_XMIT(WS_CLASS,WS_NETPX,WS_STATE_
                            NO);
550                             WS_MODEFF :=
                                WCX.Determine_Split_Period(WS_CLASS,WS_POLICY,WS
                                _SPLITNO,WS_STATE_NO);
551
552                         WCX.TEMP_INSERT_UNEX (WS_UNEX_WC_NO,
                            WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                            WS_CLASS, WS_MODVAL, WS_MODEFF, WS_BURT_EFF,
                            WS_PAYROLL, WS_NETPX, WS_MODPX, WS_SPLITNO,
                            WS_BURT, WS_AFTERMOD, WS_XMIT, ws_sqlcode) ;
553                         COMMIT;
554
555                     END IF; -- STATE NO = 07 OR 37
556                 END IF;       -- WS_CLASS <> '0900'
557                 -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
558             END LOOP; -- GET_TOTALS
559         END LOOP;       -- GET_EXPO_A
```

```
560                 WS_SECTION := 'WCUNEX3_11i';
561               ELSE --ANNDATE IF (ANNDATE IS NULL)
562
563                 -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
564                 -- THESE ARE THE POLICIES THAT DO NOT HAVE AN ANNIVERSARY DATE
                    BUT HAVE
565                 -- STILL BEEN SPLIT.  THE LOGIC BELOW DETERMINES IF THE SPLIT IS
                    NEEDED
566                 -- AND MERGES RECORDS WHERE IT IS NOT.
567                 -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
568                 -- DETERMINE IF PERIOD SPLIT DUE TO HAVING A MOD OR RATE CHANGE
569                 WS_SECTION := 'WCUNEX3_11j';
570                   SELECT COUNT (DISTINCT MODIFIER_VALUE)     ,
571                       COUNT (DISTINCT MULTIPLIER_EFFECTIVE),
572                       COUNT (DISTINCT MULTIPLIER)
573                    INTO WS_COUNT,
574                       WS_COUNT2,
575                       WS_COUNT3
576                    FROM TEMP_UNIT_EXPO TUNX
577                    WHERE TUNX.CAT_CODE         = 'MANUAL'
578                       AND TUNX.POLICY_NUMBER = WS_POLICY
579                       AND TUNX.STATE_NUMBER  = WS_STATE_NO;
580                 --
581                     WS_SECTION := 'WCUNEX3_MN22';
582                     IF WS_COUNT >1 AND WS_COUNT2 > 1 AND WS_COUNT3 >1 or
583                      WS_COUNT >1 AND WS_COUNT2 > 1 AND WS_COUNT3 >=1 or
584                      WS_COUNT >1 AND WS_COUNT2 >= 1 AND WS_COUNT3 >1 or
585                      WS_COUNT >=1 AND WS_COUNT2 > 1 AND WS_COUNT3 >1
                         THEN
586                        WS_SPLIT_MARKER := 'Y';
587                        ELSE
588                        WS_SPLIT_MARKER := 'N';
589                     END IF;
590                 --
591                 IF WS_SPLIT_MARKER = 'N'  or (WS_ANNDATE IS NOT NULL AND
                    WS_STATE_NO = '22') THEN  -- COUNT IF
592                     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
593                     -- IN THIS CASE THE SPLIT IS UNJUSTIFIED OR OCCURS IN
                        MINNESOTA (WHERE
594                     -- SPLITS ARE NOT ACCEPTED)
595                     -- *** 22-JUN-2009: MN splits are OK for mod changes, not anniversary dates
                        ***
596                     -- *** 24-APR-2014: By 05-OCT-2011, MN splits were NOT allowed for mod
                        changes
597                     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
598                         WS_SECTION     := 'WCUNEX3_12a1';
599
600                     FOR REC5 IN GET_EXPO_C
601                     LOOP
602                         WS_SECTION     := 'WCUNEX3_12a2';
603                         WS_HOLD_MODVAL := WS_MODVAL;
604                         WS_CLASS       := REC5.CLASSCODE;
605                         WS_AFTERMOD    := REC5.AFTER_MOD;
606                         WS_MODVAL      := REC5.MODVAL;
607                         WS_BURT        := REC5.BURT_RATE;
608                         WS_BURT_EFF    := REC5.BURT_EFF;
```

```
609                     WS_EXPOEFF     := WS_BURT_EFF;
610                     WS_MODEFF      := WS_EXPOEFF;
611
612                     WS_SPLITNO     := 0;
613                     -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
614                     WS_SECTION := 'WCUNEX3_12b';
615                     FOR REC6   IN GET_TOTALS_C
616                     LOOP
617                          WS_STDPX   := REC6.STDPX;
618                          WS_MODPX   := REC6.MODPX;
619                          WS_NETPX   := REC6.NETPX;
620                          WS_WRTPX   := REC6.WRTPX;
621                          WS_PAYROLL := REC6.PYRLL;
622                          WS_AFTERMOD :=   WCX.DETERMINE_AFTERMOD
                             (WS_AFTERMOD, WS_CLASS, WS_STATE_NO, WS_STDPX);
623
624                          --
625                          --
                             (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                             =)
626                          WS_SECTION := 'WCUNEX3_13 ';
627                          WCX.RESET_PERIOD_SPLIT (WS_CLASS, WS_POLICY,
                             WS_EXPOEFF, ws_sqlcode) ;
628                            IF WS_SQLCODE NOT IN (0,1) THEN
629                                DBMS_OUTPUT.PUT_LINE('ERROR     : '||WS_SECTION||'
                                   '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                   '||WS_SPLIT_MARKER);
630                            END IF;
631                          --
632                          WS_SECTION := 'WCUNEX3_14 ';
633                           SELECT UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                              INTO WS_UNEX_WC_NO FROM DUAL;
634                          --
                             (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                             =)
635                          WS_SECTION          := 'WCUNEX3_15 ';
636
637                         if WS_STATE_NO <> '20' then
638                          IF NVL (WS_MODVAL, 0) = 0 THEN
639                                WS_MODPX      := WS_NETPX;
640                          END IF;
641                         end if;
642                          --
                             (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                             =)
643                          -- INSERT INTO UNIT_STATS_EXPOSURE
644                          --
                             (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                             =)
645                          WS_SECTION := 'WCUNEX3_15A';
646                           SELECT COUNT (DISTINCT UNEX_UNPL_WC_NO) AS IND_1111
647                             INTO WS_1111_IND
648                             FROM UNIT_STATS_EXPOSURE UNEX
649                            WHERE UNEX.UNEX_WC_REPORT_DATE       =
                               WS_REPORTDATE
650                                AND UNEX.UNEX_WC_EXIT_CLASS_CODE = '1111'
```

```
651                                 AND UNEX.UNEX_STAT_GB_STATE_NO   = WS_STATE_NO
652                                 AND UNEX_UNPL_WC_NO              = WS_UNPL_ID;
653
654                             --
655                              SELECT DECODE (WS_1111_IND, 0, 'N', 'Y')
656                              INTO WS_PROC_IND
657                              FROM DUAL;
658                             --
659                             WS_TOTAL_PX := WCX.CALCULATE_TTL_PREM (WS_NETPX,
                                WS_MODPX);
660                             --
661                             WS_SECTION := 'WCUNEX3_15B';
662
663                                 WS_TOTAL_EXPOSURE:=
                                    WCX.CALCULATE_TTL_EXPOSURE (WS_TOTAL_PX,
                                    WS_PAYROLL);
664
665                                 WS_XMIT :=
                                    WCX.GET_EXPO_XMIT(WS_CLASS,WS_NETPX,WS_STATE_
                                    NO);
666                             --
667                             WS_SECTION    := 'WCUNEX3_15C';
668
669                             IF WS_PROC_IND = 'Y' AND WS_CLASS IN ('0900', '0990','1111')
                                THEN
670                                 WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                    WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                    WS_CLASS, WS_MODVAL, WS_MODEFF, WS_BURT_EFF,
                                    WS_PAYROLL, WS_NETPX, WS_MODPX, WS_SPLITNO,
                                    WS_BURT, WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                    WS_TOTAL_EXPOSURE, WS_TOTAL_PX, WS_SECTION,
                                    WS_XMIT, WS_SQLCODE) ;
671
672                                     if ws_sqlcode not in (0,1) THEN
673                                         dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                            '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                            '||WS_SPLIT_MARKER);
674                                     END IF;
675                                 ELSIF WS_PROC_IND = 'N' THEN
676                                 IF WS_STATE_no in ('04') and SUBSTR (WS_CLASS, 1, 4) IN
                                    ('0930') THEN
677                                                                     WS_MODVAL          := 0;
678                                                                 END
                                                                    IF;
679                                 WS_SECTION    := 'WCUNEX3_15D';
680                                     WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                        WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                        WS_CLASS, WS_MODVAL, WS_MODEFF,
                                        WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                                        WS_MODPX, WS_SPLITNO, WS_BURT,
                                        WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                        WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                                        WS_SECTION, WS_XMIT, WS_SQLCODE) ;
681
```

```
682                                     if ws_sqlcode not in (0,1) THEN
683                                         dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                            '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                            '||WS_SPLIT_MARKER);
684                                     END IF;
685                                 END IF;
686                                 --
                                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                                    =)
687                             END LOOP; -- GET_TOTALS_C
688                         END LOOP;       -- REC5 LOOP
689                     ELSE                -- COUNT IF
690                         -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
691                         -- THESE POLICIES HAVE MORE THAN ONE MOD EFFECTIVE DATE
                            DUE TO A MOD CHANGE
692                         -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
693                         FOR REC7 IN GET_EXPO_B
694                         LOOP
695                             WS_SECTION     := 'WCUNEX3_16 ';
696                             WS_SPLITNO     := 0;
697                             WS_HOLD_MODVAL := WS_MODVAL;
698                             WS_CLASS       := REC7.CLASSCODE;
699                             WS_AFTERMOD    := REC7.AFTER_MOD;
700                             WS_MODVAL      := REC7.MODVAL;
701                             WS_BURT        := REC7.BURT_RATE;
702                             WS_BURT_EFF    := REC7.BURT_EFF;
703                             WS_EXPOEFF     := WS_BURT_EFF;
704                             STOP_PROCESSING := 'N' ;
705                             --
706                             WS_SECTION := 'WCUNEX3_17 ';
707                              SELECT MIN (TUNX.EXPOSURE_EFFECTIVE) AS MINEFF,
708                                  MAX (TUNX.EXPOSURE_EFFECTIVE)  AS MAXEFF,
709                                  MIN (TUNX.MOD_EFFECTIVE)       AS MINMOD,
710                                  MAX (TUNX.MOD_EFFECTIVE)       AS MAXMOD
711                                INTO WS_MINEFF,
712                                  WS_MAXEFF ,
713                                  WS_MINMOD ,
714                                  WS_MAXMOD
715                                FROM TEMP_UNIT_EXPO TUNX
716                               WHERE TUNX.POLICY_NUMBER = WS_POLICY
717                                  AND TUNX.CLASS_CODE = WS_CLASS;
718                             --
719
720                             IF WS_MODVAL           <> NVL (WS_HOLD_MODVAL, WS_MODVAL)
                                THEN -- MODVAL IF
721                                 WS_SECTION       := 'WCUNEX3_17B';
722
723                                 IF WS_MINEFF      = WS_BURT_EFF THEN
724                                                     WS_SECTION       :=
                                                    'WCUNEX3_171';
725                                     WS_SPLITNO := 0;
726                                 ELSE
727                                                     WS_SECTION       :=
                                                    'WCUNEX3_172';
728                                     WS_SPLITNO     := 1;
```

```
729                         WS_SPLIT_MARKER := 'Y';
730
731                     END IF;
732                     --
733                     --
734                     if WS_STATE_NO <> '22' then
735                     WS_MODEFF  := WS_BURT_EFF;
736                     end if;
737                     --
738                     WS_SECTION := 'WCUNEX3_17C';
739                      WS_AFTERMOD :=   WCX.DETERMINE_AFTERMOD
                        (WS_AFTERMOD, WS_CLASS, WS_STATE_NO, WS_STDPX);
740                     --
741                     WS_SECTION := 'WCUNEX3_18 ';
742                     FOR REC8   IN GET_TOTALS_B
743                     LOOP
744                         WS_STDPX   := REC8.STDPX;
745                         WS_MODPX   := REC8.MODPX;
746                         WS_NETPX   := REC8.NETPX;
747                         WS_WRTPX   := REC8.WRTPX;
748                         WS_PAYROLL := REC8.PYRLL;
749                         --
750                         WS_SECTION := 'WCUNEX3_19 ';
751
752                         WCX.RESET_PERIOD_SPLIT (WS_CLASS, WS_POLICY,
                            WS_EXPOEFF, ws_sqlcode) ;
753                             IF WS_SQLCODE NOT IN (0,1) THEN
754                                 DBMS_OUTPUT.PUT_LINE('ERROR     :
                                    '||WS_SECTION||' '||WS_UNPL_ID||' '||WS_CLASS||'
                                    '||WS_SPLITNO||'
                                    '||WS_SPLIT_MARKER);
                                        --
755                             END IF;
756                          --
757                          WS_SECTION := 'WCUNEX3_20 ';
758
759                          SELECT UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                             INTO WS_UNEX_WC_NO FROM DUAL;
760
761                         WS_SECTION          := 'WCUNEX3_21 ';
762                         IF NVL (WS_MODVAL, 0) = 0 THEN
763                               WS_MODPX      := WS_NETPX;
764                         END IF;
765                         --
                            (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                            (=)=(=)
766                         -- INSERT INTO UNIT_STATS_EXPOSURE
767                         --
                            (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                            (=)=(=)
768                         WS_SECTION := 'WCUNEX3_21A';
769                         WS_TOTAL_PX := WCX.CALCULATE_TTL_PREM
                            (WS_NETPX, WS_MODPX);
770                         --
771                         WS_SECTION := 'WCUNEX3_21B';
772                         WS_TOTAL_EXPOSURE:=
```

```
                                    WCX.CALCULATE_TTL_EXPOSURE (WS_TOTAL_PX,
                                    WS_PAYROLL);
773                                 --
774                                 WS_SECTION := 'WCUNEX3_21C';
775                                      WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                         WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                         WS_CLASS, WS_MODVAL, WS_MODEFF,
                                         WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                                         WS_MODPX, WS_SPLITNO, WS_BURT,
                                         WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                         WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                                         WS_SECTION, WS_XMIT, WS_SQLCODE) ;
776
777                                       IF WS_SQLCODE NOT IN (0,1) THEN
778                                          DBMS_OUTPUT.PUT_LINE('ERROR     :
                                             '||WS_SECTION||' '||WS_UNPL_ID||' '||WS_CLASS||'
                                             '||WS_SPLITNO||' '||WS_SPLIT_MARKER);
779                                       END IF;
780                                 --
                                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                                    =)
781                                  END LOOP; -- GET_TOTALS_B
782                            ELSE           -- MODVAL = WS_HOLD_MODVAL
783                                 WS_MODEFF         := WS_MINEFF;
784                                 WS_SECTION        := 'WCUNEX3_21D';
785                                 --
                                    (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(
                                    =)
786                                  WS_AFTERMOD :=   WCX.DETERMINE_AFTERMOD
                                     (WS_AFTERMOD, WS_CLASS, WS_STATE_NO, WS_STDPX);
787                                 --
788                                 WS_SECTION := 'WCUNEX3_22 ';
789                                 FOR REC9   IN GET_TOTALS_B
790                                 LOOP
791                                      WS_STDPX   := REC9.STDPX;
792                                      WS_MODPX   := REC9.MODPX;
793                                      WS_NETPX   := REC9.NETPX;
794                                      WS_WRTPX   := REC9.WRTPX;
795                                      WS_PAYROLL := REC9.PYRLL;
796                                      --
797                                      WS_SECTION := 'WCUNEX3_23 ';
798                                      WCX.RESET_PERIOD_SPLIT (WS_CLASS, WS_POLICY,
                                         WS_EXPOEFF, ws_sqlcode) ;
799                                   IF WS_SQLCODE NOT IN (0,1) THEN
800                                       DBMS_OUTPUT.PUT_LINE('ERROR     : '||WS_SECTION||'
                                          '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                          '||WS_SPLIT_MARKER);
801                                   END IF;
802                                      --
803                                      WS_SECTION := 'WCUNEX3_24 ';
804                                       SELECT UNIT_STATS_EXPOSURE_SEQUENCE.NEXTVAL
                                          INTO WS_UNEX_WC_NO FROM DUAL;
805                                   IF WS_SQLCODE NOT IN (0,1) THEN
806                                       DBMS_OUTPUT.PUT_LINE('ERROR     : '||WS_SECTION||'
                                          '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
```

```
                                    '||WS_SPLIT_MARKER);
807                                END IF;
808
809                                WS_SECTION          := 'WCUNEX3_25 ';
810                                IF NVL (WS_MODVAL, 0) = 0 THEN
811                                     WS_MODPX      := WS_NETPX;
812                                END IF;
813                                --
                                   (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                                   (=)=(=)
814                                -- INSERT INTO UNIT_STATS_EXPOSURE
815                                --
                                   (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                                   (=)=(=)
816                                WS_SECTION := 'WCUNEX3_25A';
817                                --
818                                   WS_TOTAL_PX := WCX.CALCULATE_TTL_PREM
                                   (WS_NETPX, WS_MODPX);
819                                --
820                                   WS_TOTAL_EXPOSURE:=
                                   WCX.CALCULATE_TTL_EXPOSURE (WS_TOTAL_PX,
                                   WS_PAYROLL);
821                                --
822                                  IF WS_CLASS NOT IN ('9740','9741','0900','9752', '0935',
                                  '0936')  then
823                                   IF WS_MINEFF       = WS_BURT_EFF THEN
824                                        WS_SECTION       := 'WCUNEX3_21e';
825                                        WS_SPLITNO := 0;
826                                   ELSE
827                                        WS_SECTION       := 'WCUNEX3_21f';
828                                        WS_SPLITNO      := 1;
829                                        WS_SPLIT_MARKER := 'Y';
830                                        if WS_STATE_NO != '22' then
831                                        WS_MODEFF := WS_BURT_EFF;
832                                      end if;
833                                        WS_SECTION       := 'WCUNEX3_21f2';
834                                   END IF;
835                                   ELSE
836                                   WS_SECTION       := 'WCUNEX3_21g';
837                                    WS_BURT_EFF := WS_MINEFF;
838                                    WS_SPLITNO := 0;
839                                  END IF;
840                                     WCX.INSERT_EXPOSURES (WS_UNEX_WC_NO,
                                        WS_UNPL_ID, WS_REPORTDATE, WS_STATE_NO,
                                        WS_CLASS, WS_MODVAL, WS_MODEFF,
                                        WS_BURT_EFF, WS_PAYROLL, WS_NETPX,
                                        WS_MODPX, WS_SPLITNO, WS_BURT,
                                        WS_AFTERMOD, WS_POLICY, WS_ANNDATE,
                                        WS_TOTAL_EXPOSURE, WS_TOTAL_PX,
                                        WS_SECTION, WS_XMIT, WS_SQLCODE) ;
841
842                                      if ws_sqlcode not in (0,1) THEN
843                                          dbms_output.put_line('ERROR     : '||WS_SECTION||'
                                          '||WS_UNPL_ID||' '||WS_CLASS||' '||WS_SPLITNO||'
                                          '||WS_SPLIT_MARKER);
844                                      END IF;
```

```
                                        --
                                        (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=
                                        (=)=(=)
                                    END LOOP; -- GET_TOTALS_B/REC9
                                END IF;         -- MODVAL IF
                            END LOOP;               -- GET_EXPO_B
                        END IF;                     -- COUNT IF
                    END IF;                         -- ANNDATE IF
                END LOOP;                               -- A1 LOOP
            END LOOP;                                   -- REC1 LOOP
            -- (=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)=(=)
             SELECT TO_CHAR (SYSDATE, 'HH24:MI:SS')
               INTO VAR_STOP
               FROM DUAL;
            DBMS_OUTPUT.PUT_LINE (VAR_START) ;
            DBMS_OUTPUT.PUT_LINE (VAR_STOP) ;



EXCEPTION
WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE ('****** NO DATA FOUND ******************') ;
            DBMS_OUTPUT.PUT_LINE ('SECTION '||WS_SECTION) ;
            DBMS_OUTPUT.PUT_LINE ('SQLCODE: '||SQLCODE||' POLICY: '||WS_POLICY) ;
            DBMS_OUTPUT.PUT_LINE ('UNEX ID: '||WS_UNEX_WC_NO||' UNPL ID: '||WS_UNPL_ID||'
            REPORT DATE: '||WS_REPORTDATE) ;
            DBMS_OUTPUT.PUT_LINE ('STATE: '||WS_STATE_NO||' CLASS: '||WS_CLASS||' NET:
            '||WS_NETPX||' MOD: '||WS_MODPX||' SPLIT: '||WS_SPLITNO||' TYPE: '||WS_AFTERMOD) ;
            DBMS_OUTPUT.PUT_LINE (SQLERRM) ;
            DBMS_OUTPUT.PUT_LINE (DBMS_UTILITY.FORMAT_ERROR_BACKTRACE) ;
            DBMS_OUTPUT.PUT_LINE ('****** END OF LINE ********************') ;
WHEN TOO_MANY_ROWS THEN
            DBMS_OUTPUT.PUT_LINE ('****** TOO MANY ROWS ******************') ;
            DBMS_OUTPUT.PUT_LINE ('SECTION '||WS_SECTION) ;
            DBMS_OUTPUT.PUT_LINE ('SQLCODE: '||SQLCODE||' POLICY: '||WS_POLICY) ;
            DBMS_OUTPUT.PUT_LINE ('UNEX ID: '||WS_UNEX_WC_NO||' UNPL ID: '||WS_UNPL_ID||'
            REPORT DATE: '||WS_REPORTDATE) ;
            DBMS_OUTPUT.PUT_LINE ('STATE: '||WS_STATE_NO||' CLASS: '||WS_CLASS||' NET:
            '||WS_NETPX||' MOD: '||WS_MODPX||' SPLIT: '||WS_SPLITNO||' TYPE: '||WS_AFTERMOD) ;
            DBMS_OUTPUT.PUT_LINE (SQLERRM) ;
            DBMS_OUTPUT.PUT_LINE (DBMS_UTILITY.FORMAT_ERROR_BACKTRACE) ;
            DBMS_OUTPUT.PUT_LINE ('****** END OF LINE ********************') ;
WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE ('****** ERROR ***************************') ;
            DBMS_OUTPUT.PUT_LINE ('SECTION '||WS_SECTION) ;
            DBMS_OUTPUT.PUT_LINE ('SQLCODE: '||SQLCODE||' POLICY: '||WS_POLICY) ;
            DBMS_OUTPUT.PUT_LINE ('UNEX ID: '||WS_UNEX_WC_NO||' UNPL ID: '||WS_UNPL_ID||'
            REPORT DATE: '||WS_REPORTDATE) ;
            DBMS_OUTPUT.PUT_LINE ('STATE: '||WS_STATE_NO||' CLASS: '||WS_CLASS||' NET:
            '||WS_NETPX||' MOD: '||WS_MODPX||' SPLIT: '||WS_SPLITNO||' TYPE: '||WS_AFTERMOD) ;
            DBMS_OUTPUT.PUT_LINE (SQLERRM) ;
            DBMS_OUTPUT.PUT_LINE (DBMS_UTILITY.FORMAT_ERROR_BACKTRACE) ;
            DBMS_OUTPUT.PUT_LINE ('****** END OF LINE ********************') ;
END;
/
```

```
893  --spool off
894
```