```
CREATE OR REPLACE PACKAGE REVISORI.estrai_istanze_new
AS
    --    qta_col   NUMBER := 9;
    --    qta_mon   NUMBER := 3;

    -- TYPES
    QuantiTentativi   NUMBER := 100;

    TYPE revisore_typ IS RECORD
    (
        cognome                   VARCHAR2 (500 BYTE),
        nome                      VARCHAR2 (500 BYTE),
        sesso                     CHAR (1 BYTE),
        pec                       VARCHAR2 (500 BYTE),
        codice_fiscale            VARCHAR2 (16 BYTE),
        numero_protocollo         VARCHAR2 (4000 BYTE),
        data_protocollo           DATE,
        flag_giovane_inesperto    NUMBER (1),
        flag_revisore_uscente     NUMBER (1),
        id_mdas                   NUMBER
    );

    TYPE lista_revisore_typ IS TABLE OF revisore_typ;

    TYPE numero_estratto_typ IS RECORD
    (
        numero    NUMBER
    );

    TYPE numeri_estratti_typ IS TABLE OF numero_estratto_typ;

    TYPE cf_estratto_typ IS RECORD
    (
        codice_fiscale    VARCHAR2 (16 BYTE)
    );

    TYPE cf_estratti_typ IS TABLE OF cf_estratto_typ;


    PROCEDURE crea_lista (p_id_avviso        IN     NUMBER,
                          p_codice_fiscale   IN     VARCHAR2,
                          p_lista_revisori      OUT lista_revisore_typ,
                          p_messaggio           OUT VARCHAR2);

    PROCEDURE estrazione (p_urna              IN     lista_revisore_typ,
                          p_numeri_estratti   IN OUT numeri_estratti_typ,
                          p_id_avviso         IN     NUMBER,
                          p_nrc_reali         IN     NUMBER,
                          p_nmrc_reali        IN     NUMBER,
                          p_n_giovani_reali   IN     NUMBER,
                          p_codice_fiscale    IN     VARCHAR2,
                          p_estratto             OUT revisore_typ,
                          p_errore               OUT VARCHAR2);
```

```
END estrai_istanze_new;
/


CREATE OR REPLACE PACKAGE BODY REVISORI.estrai_istanze_new
AS

    PROCEDURE estrazione (p_urna              IN     lista_revisore_typ,
                          p_numeri_estratti   IN OUT numeri_estratti_typ,
                          p_id_avviso         IN     NUMBER,
                          p_nrc_reali         IN     NUMBER,
                          p_nmrc_reali        IN     NUMBER,
                          p_n_giovani_reali   IN     NUMBER,
                          p_codice_fiscale    IN     VARCHAR2,
                          p_estratto             OUT revisore_typ,
                          p_errore               OUT VARCHAR2)
    IS
        LE_ERRORE   EXCEPTION;
    --        l_nr_max_femmine           BOOLEAN := FALSE;
    --        l_nr_max_maschi            BOOLEAN := FALSE;
    --        l_nr_max_n_giovani         BOOLEAN := FALSE;
    --        l_nr_max_giovani           BOOLEAN := FALSE;
    --        l_return                   NUMBER;
    --        --
    --        l_nrc                      NUMBER;
    --        l_nmrc_min                 NUMBER;
    --        l_nmrc_max                 NUMBER;
    --        l_ngiovrc_min              NUMBER;
    --        l_ngiovrc_max              NUMBER;
    --        --
    --        --        ln_app                     NUMBER;
    --        ln_numero_righe_estratte   NUMBER;
    --        ln_conta                   NUMBER;
    --        l_indice                   NUMBER;
    --        l_continuo                 BOOLEAN;
    --        l_numero_libero            BOOLEAN;
    --        ln_numero_riga             NUMBER;
    BEGIN
        --        DBMS_OUTPUT.put_line ('ESTRAZIONE');
        NULL;
    EXCEPTION
        WHEN le_errore
        THEN
            NULL;
        WHEN OTHERS
        THEN
            p_errore := 'estrazione ' || SQLERRM;
    END estrazione;

    PROCEDURE pulisci_urna (p_urna                      IN OUT
lista_revisore_typ,
                            p_codice_fiscale_estratto   IN     VARCHAR,
                            p_return                       OUT NUMBER,
                            p_messaggio                    OUT VARCHAR2)
```

```
    IS
        l_urna_appoggio   lista_revisore_typ;
        l_indice          NUMBER;
        l_conta           NUMBER;
    --        l_mantengo        BOOLEAN;
    BEGIN
        p_return := 1;
        p_messaggio := NULL;
        l_urna_appoggio := lista_revisore_typ ();
        l_indice := p_urna.COUNT ();
        l_conta := 0;

        --        DBMS_OUTPUT.put_line ('Quanti in urna ' || l_indice);


        FOR i IN 1 .. l_indice
        LOOP
            IF p_urna (i).codice_fiscale <> p_codice_fiscale_estratto
            THEN
                l_urna_appoggio.EXTEND;
                l_conta := l_conta + 1;
                l_urna_appoggio (l_conta).cognome := p_urna (i).cognome;
                l_urna_appoggio (l_conta).nome := p_urna (i).nome;
                l_urna_appoggio (l_conta).sesso := p_urna (i).sesso;
                l_urna_appoggio (l_conta).pec := p_urna (i).pec;
                l_urna_appoggio (l_conta).id_mdas := p_urna (i).id_mdas;
                l_urna_appoggio (l_conta).codice_fiscale :=
                    p_urna (i).codice_fiscale;
                l_urna_appoggio (l_conta).numero_protocollo :=
                    p_urna (i).numero_protocollo;
                l_urna_appoggio (l_conta).data_protocollo :=
                    p_urna (i).data_protocollo;
                l_urna_appoggio (l_conta).flag_giovane_inesperto :=
                    p_urna (i).flag_giovane_inesperto;
            END IF;
        END LOOP;

        --        DBMS_OUTPUT.put_line ('Mantengo: ' || l_urna_appoggio.COUNT
());
        p_urna := l_urna_appoggio;
    EXCEPTION
        WHEN OTHERS
        THEN
            p_return := -1;
            p_messaggio := 'Errore PULISCI_URNA ' || SQLERRM;
    END PULISCI_URNA;


    PROCEDURE crea_lista (p_id_avviso        IN     NUMBER,
                          p_codice_fiscale   IN     VARCHAR2,
                          p_lista_revisori      OUT lista_revisore_typ,
                          p_messaggio           OUT VARCHAR2)
    IS
        CURSOR urna_a (p_id_avviso IN NUMBER, p_numero_riga IN NUMBER)
```

```
IS
    WITH
        selezione
        AS
            (SELECT ROW_NUMBER ()
                        OVER (ORDER BY sog_codice_fiscale DESC)
                        numero_riga,
                    a.*
               FROM v_consumo_sorteggiabili_istanz a
              WHERE     id_pub_avviso = p_id_avviso
                    AND (   a.SOG_CODICE_FISCALE <> p_codice_fiscale
                         OR p_codice_fiscale IS NULL))
    SELECT *
      FROM selezione
     WHERE numero_riga = p_numero_riga OR p_numero_riga IS NULL;

CURSOR urna_b (p_id_avviso IN NUMBER, p_numero_riga IN NUMBER)
IS
    WITH
        selezione
        AS
            (SELECT ROW_NUMBER () OVER (ORDER BY sog_cf DESC)
                        numero_riga,
                    a.*
               FROM v_consumo_sorteggiabili_elenc a
              WHERE     id_pub_avviso = p_id_avviso
                    AND (   a.SOG_CF <> p_codice_fiscale
                         OR p_codice_fiscale IS NULL)
                    --    questo aggiunto x eliminare quelli che già
                    --    stanno in urna A
                    AND NOT EXISTS
                            (SELECT 1
                               FROM v_consumo_sorteggiabili_istanz b
                              WHERE     id_pub_avviso = p_id_avviso
                                    AND a.sog_cf =
                                        b.sog_codice_fiscale))
    SELECT *
      FROM selezione
     WHERE numero_riga = p_numero_riga OR p_numero_riga IS NULL;

--
LE_ERRORE                   EXCEPTION;
LE_DISASTRO                 EXCEPTION;
--
l_nrc_reali                 NUMBER;
l_n_maschi_estratti         NUMBER;
l_n_femmine_estratte        NUMBER;
l_n_giovani_estratti        NUMBER;
l_n_esperti_estratti        NUMBER;
lb_elimini_femmine          BOOLEAN;
lb_elimini_maschi           BOOLEAN;
lb_elimini_giovani          BOOLEAN;
lb_elimini_esperti          BOOLEAN;
lb_continuo                 BOOLEAN;
```

```
        --        l_numero_libero           BOOLEAN;
        l_nrc                       NUMBER;
        l_na0                       NUMBER;
        --
        ls_app                      VARCHAR2 (200);
        ls_app1                     VARCHAR2 (200);
        l_messaggio                 VARCHAR2 (200);
        l_quanti                    NUMBER;
        ln_numero_riga_estratta     NUMBER;
        l_conta                     NUMBER;
        ln_numTentativi             NUMBER;
        --        l_QuantiTentativi           NUMBER;
        ln_app                      NUMBER;
        --        ln_app1                     NUMBER;
        ln_revisori_urna            NUMBER;

        l_return                    NUMBER;
        r_v_consumo_sort_elenc      v_consumo_sorteggiabili_elenc%ROWTYPE;
        l_lista_revisori            lista_revisore_typ;
        l_urna                      lista_revisore_typ;
        --        l_urna_appoggio         lista_revisore_typ;
        --        l_numero_riga_estratta    numeri_estratti_typ;
        l_codici_fiscali            cf_estratti_typ;
        l_codice_fiscale_estratto   VARCHAR (16);
        lr_urna_a                   urna_a%ROWTYPE;
        lr_urna_b                   urna_b%ROWTYPE;
    BEGIN
--        DBMS_OUTPUT.put_line (' crea_lista ');

        IF (p_id_avviso IS NULL)
        THEN
            p_messaggio := 'SQL00003 parametro Codice Avviso è vuoto ';
            ls_app := p_messaggio;
            RAISE LE_ERRORE;
        END IF;

       --        l_QuantiTentativi := 0;

       <<INIZIO>>
        --
        --  Numero palline da estrarre nella seduta di sorteggio corrente
        l_nrc := utility_estrazione.get_nrc (p_id_avviso, p_codice_fiscale);
        --  Numero di revisori presenti nell'URNA A
        l_na0 := utility_estrazione.get_na0 (p_id_avviso);
        --Inizializzo la lista revisori
        l_lista_revisori := lista_revisore_typ ();
        l_nrc_reali := 0;
        l_urna := lista_revisore_typ ();
        l_quanti := 0;

        l_n_maschi_estratti := 0;
        l_n_femmine_estratte := 0;
        l_n_giovani_estratti := 0;
        l_n_esperti_estratti := 0;
```

```
        ln_numero_riga_estratta := 0;
        --        l_numero_riga_estratta := numeri_estratti_typ ();
        l_codici_fiscali := cf_estratti_typ ();


        IF p_codice_fiscale IS NOT NULL
        THEN
            --            DBMS_OUTPUT.put_line ('SI revisore uscente');
            --            crea_errore ('CREA_LISTA', 'Passo 1');
            BEGIN
                SELECT a.*
                  INTO r_v_consumo_sort_elenc
                  FROM v_consumo_sorteggiabili_elenc a
                 WHERE     a.id_pub_avviso = p_id_avviso
                       AND a.SOG_CF = p_codice_fiscale;


                l_quanti := l_quanti + 1;
                l_lista_revisori.EXTEND ();
                l_lista_revisori (l_quanti).cognome :=
                    r_v_consumo_sort_elenc.SOG_COGNOME;
                l_lista_revisori (l_quanti).nome :=
                    r_v_consumo_sort_elenc.SOG_NOME;
                l_lista_revisori (l_quanti).sesso :=
                    r_v_consumo_sort_elenc.SOG_SESSO;
                l_lista_revisori (l_quanti).pec :=
                    r_v_consumo_sort_elenc.SOG_PEC;
                l_lista_revisori (l_quanti).id_mdas :=
                    r_v_consumo_sort_elenc.id_mdas;
                l_lista_revisori (l_quanti).codice_fiscale :=
                    p_codice_fiscale;
                l_lista_revisori (l_quanti).numero_protocollo := NULL;
                l_lista_revisori (l_quanti).data_protocollo := NULL;
                l_lista_revisori (l_quanti).flag_giovane_inesperto :=
                    r_v_consumo_sort_elenc.flag_giovane_inesperto;
                l_lista_revisori (l_quanti).flag_revisore_uscente := 1;
                --                DBMS_OUTPUT.put_line ('Ho estratto il nome Nr°
' || l_quanti);
                --
                l_nrc_reali := l_nrc_reali + 1;

                IF (r_v_consumo_sort_elenc.SOG_SESSO = 'M')
                THEN
                    l_n_maschi_estratti := l_n_maschi_estratti + 1;
                ELSE
                    l_n_femmine_estratte := l_n_femmine_estratte + 1;
                END IF;

                IF (r_v_consumo_sort_elenc.flag_giovane_inesperto = 1)
                THEN
                    l_n_giovani_estratti := l_n_giovani_estratti + 1;
                ELSE
                    l_n_esperti_estratti := l_n_esperti_estratti + 1;
                END IF;
```

```
            EXCEPTION
                WHEN OTHERS
                THEN
                    --                    DBMS_OUTPUT.put_line (SQLERRM);
                    ls_app := 'ERRORE 3 ' || SQLERRM;
                    RAISE LE_ERRORE;
            END;
        END IF;


        --
        --  Devo decidere se pesco tutti i nomi dall'URNA A e se l'URNA A è
uguale ai revisori da estrarre  - 1° caso
        --              o se l'URNA A non è sufficiente e quindi dovrò pescare
dall'URNA B   - 2° caso
        --  Alla fine di questo IF ho una variable URNA da cui devo poter
estrarre i nomi.
        --
        IF (   l_nrc <= l_na0
            OR (l_nrc - 1 <= l_na0 AND p_codice_fiscale IS NOT NULL))
        THEN
--            DBMS_OUTPUT.put_line (' Caso Urna A >= da sorteggiare ');

            OPEN urna_a (p_id_avviso, NULL);

            l_conta := 0;

            LOOP
                FETCH urna_a INTO lr_urna_a;

                EXIT WHEN urna_a%NOTFOUND;
                l_conta := l_conta + 1;
                l_urna.EXTEND;
                l_urna (l_conta).cognome := lr_urna_a.sog_cognome;
                l_urna (l_conta).nome := lr_urna_a.sog_nome;
                l_urna (l_conta).sesso := lr_urna_a.sog_sesso;
                l_urna (l_conta).pec := lr_urna_a.sog_pec;
                l_urna (l_conta).id_mdas := lr_urna_a.id_mdas;

                l_urna (l_conta).codice_fiscale :=
                    lr_urna_a.sog_codice_fiscale;
                l_urna (l_conta).numero_protocollo :=
                    lr_urna_a.numero_protocollo;
                l_urna (l_conta).data_protocollo := lr_urna_a.data_protocollo;
                l_urna (l_conta).flag_giovane_inesperto :=
                    lr_urna_a.flag_giovane_inesperto;
            --
            END LOOP;

            CLOSE urna_a;
        ELSIF (   l_nrc > l_na0
               OR (l_nrc - 1 > (l_na0) AND p_codice_fiscale IS NOT NULL))
        THEN
            --  In questo caso i revisori da estrarre sono di più dei revisori
```

```
nell'URNA A
            --      oppure, nel caso ci sia un revisore uscente, i revisori da
estrarre -1 sono
            --      di revisori nell'URNA A
--            DBMS_OUTPUT.put_line (' Caso Urna A < da sorteggiare ');

            OPEN urna_a (p_id_avviso, NULL);

            LOOP
                FETCH urna_a INTO lr_urna_a;

                EXIT WHEN urna_a%NOTFOUND;
                l_quanti := l_quanti + 1;
                --                ln_numero_riga_estratta :=
ln_numero_riga_estratta + 1;
                l_lista_revisori.EXTEND ();
                --                l_numero_riga_estratta.EXTEND ();
                --                l_numero_riga_estratta
(l_quanti_estratti).numero :=
                --                    lr_urna_a.numero_riga;
                l_lista_revisori (l_quanti).cognome := lr_urna_a.sog_cognome;
                l_lista_revisori (l_quanti).nome := lr_urna_a.sog_nome;
                l_lista_revisori (l_quanti).sesso := lr_urna_a.sog_sesso;
                l_lista_revisori (l_quanti).pec := lr_urna_a.sog_pec;
                l_lista_revisori (l_quanti).id_mdas := lr_urna_a.id_mdas;
                l_lista_revisori (l_quanti).codice_fiscale :=
                    lr_urna_a.sog_codice_fiscale;
                l_lista_revisori (l_quanti).numero_protocollo :=
                    lr_urna_a.numero_protocollo;
                l_lista_revisori (l_quanti).data_protocollo :=
                    lr_urna_a.data_protocollo;
                l_lista_revisori (l_quanti).flag_giovane_inesperto :=
                    lr_urna_a.flag_giovane_inesperto;
                l_lista_revisori (l_quanti).flag_revisore_uscente := 0;
                --
                l_nrc_reali := l_nrc_reali + 1;

                IF (lr_urna_a.SOG_SESSO = 'M')
                THEN
                    l_n_maschi_estratti := l_n_maschi_estratti + 1;
                ELSE
                    l_n_femmine_estratte := l_n_femmine_estratte + 1;
                END IF;

                IF (lr_urna_a.flag_giovane_inesperto = 1)
                THEN
                    l_n_giovani_estratti := l_n_giovani_estratti + 1;
                ELSE
                    l_n_esperti_estratti := l_n_esperti_estratti + 1;
                END IF;
            END LOOP;

            CLOSE urna_a;
```

```
            OPEN urna_b (p_id_avviso, NULL);

            l_conta := 0;

            LOOP
                FETCH urna_b INTO lr_urna_b;

                EXIT WHEN urna_b%NOTFOUND;
                l_conta := l_conta + 1;
                l_urna.EXTEND;
                l_urna (l_conta).cognome := lr_urna_b.sog_cognome;
                l_urna (l_conta).nome := lr_urna_b.sog_nome;
                l_urna (l_conta).sesso := lr_urna_b.sog_sesso;
                l_urna (l_conta).pec := lr_urna_b.sog_pec;
                l_urna (l_conta).id_mdas := lr_urna_b.id_mdas;
                l_urna (l_conta).codice_fiscale := lr_urna_b.SOG_CF;
                l_urna (l_conta).numero_protocollo := NULL;
                l_urna (l_conta).data_protocollo := NULL;
                l_urna (l_conta).flag_giovane_inesperto :=
                    lr_urna_b.flag_giovane_inesperto;
            END LOOP;

            CLOSE urna_b;
        ELSE
            DBMS_OUTPUT.put_line (' ERRORE ');
            ls_app :=
                'ERRORE 1 - improbabile caso con numero palline ed urna
inconguenti';
            RAISE LE_ERRORE;
        END IF;

        IF (l_nrc > l_lista_revisori.COUNT ())
        THEN
            --            DBMS_OUTPUT.put_line (' Proseguo con il sorteggio ');
            NULL;
        END IF;

        --      SORTEGGIO INIZIO

        WHILE l_nrc - l_nrc_reali > 0
        LOOP
            l_return :=
                elimina_femmine_da_urna (p_id_avviso,
                                         p_codice_fiscale,
                                         l_n_femmine_estratte, --compreso il
revisore uscente
                                         l_n_maschi_estratti, --compreso il
revisore uscente
                                         l_n_giovani_estratti, --compreso il
revisore uscente
                                         l_n_esperti_estratti --compreso il
revisore uscente
                                                              );
```

```
            IF l_return = -1                                        --  Errore
            THEN
                NULL;
            ELSIF l_return = 1                                  --  Devo eliminare
            THEN
                lb_elimini_femmine := TRUE;
            --

            ELSE                                            --  NON devo eliminare
                lb_elimini_femmine := FALSE;
            END IF;

            l_return :=
                elimina_maschi_da_urna (p_id_avviso,
                                        p_codice_fiscale,
                                        l_n_femmine_estratte, --compreso il
revisore uscente
                                        l_n_maschi_estratti, --compreso il
revisore uscente
                                        l_n_giovani_estratti, --compreso il
revisore uscente
                                        l_n_esperti_estratti --compreso il
revisore uscente
                                                            );

            IF l_return = -1                                        --  Errore
            THEN
                NULL;
            ELSIF l_return = 1                                  --  Devo eliminare
            THEN
                lb_elimini_maschi := TRUE;
            ELSE                                            --  NON devo eliminare
                lb_elimini_maschi := FALSE;
            END IF;

            l_return :=
                elimina_esperti_da_urna (p_id_avviso,
                                         p_codice_fiscale,
                                         l_n_femmine_estratte, --compreso il
revisore uscente
                                         l_n_maschi_estratti, --compreso il
revisore uscente
                                         l_n_giovani_estratti, --compreso il
revisore uscente
                                         l_n_esperti_estratti --compreso il
revisore uscente
                                                             );

            IF l_return = -1                                        --  Errore
            THEN
                NULL;
            ELSIF l_return = 1                                  --  Devo eliminare
            THEN
```

```
            lb_elimini_esperti := TRUE;
        --
        ELSE                                          --  NON devo eliminare
            lb_elimini_esperti := FALSE;
        END IF;


        l_return :=
            elimina_giovani_da_urna (p_id_avviso,
                                     p_codice_fiscale,
                                     l_n_femmine_estratte, --compreso il
revisore uscente
                                     l_n_maschi_estratti, --compreso il
revisore uscente
                                     l_n_giovani_estratti, --compreso il
revisore uscente
                                     l_n_esperti_estratti --compreso il
revisore uscente
                                                          );


        IF l_return = -1                                        --  Errore
        THEN
            NULL;
        ELSIF l_return = 1                                  --  Devo eliminare
        THEN
            lb_elimini_giovani := TRUE;
        --
        ELSE                                          --  NON devo eliminare
            lb_elimini_giovani := FALSE;
        END IF;

        ln_revisori_urna := l_urna.COUNT ();
        --            DBMS_OUTPUT.put_line (
        --                ' L''urna ha ' || ln_revisori_urna || '
revisori');

        lb_continuo := TRUE;
        ln_numTentativi := 0;

        WHILE lb_continuo
        LOOP
            ln_numTentativi := ln_numTentativi + 1;

            IF ln_numTentativi > 100
            THEN
                --                    DBMS_OUTPUT.put_line ('Sei entrato in
loop');
                EXIT;
            END IF;

            SELECT TRUNC (DBMS_RANDOM.VALUE (1, ln_revisori_urna + 1))
              INTO ln_app
              FROM DUAL;
```

```
                --                DBMS_OUTPUT.put_line ('Estratta riga: ' ||
ln_app);
                l_codice_fiscale_estratto := l_urna (ln_app).codice_fiscale;
                --                DBMS_OUTPUT.put_line (
                --                       'Palline estratte fino ad ora: '
                --                    || l_codici_fiscali.COUNT);

                --                l_numero_libero := TRUE;
                lb_continuo := FALSE;

                IF     l_urna (ln_app).flag_giovane_inesperto = 1
                   AND lb_elimini_giovani
                THEN
                    lb_continuo := TRUE;
                END IF;

                --
                IF     l_urna (ln_app).flag_giovane_inesperto = 0
                   AND lb_elimini_esperti
                THEN
                    lb_continuo := TRUE;
                END IF;

                --
                IF l_urna (ln_app).sesso = 'M' AND lb_elimini_maschi
                THEN
                    lb_continuo := TRUE;
                END IF;

                --
                IF l_urna (ln_app).sesso = 'F' AND lb_elimini_femmine
                THEN
                    lb_continuo := TRUE;
                END IF;
            --                IF l_codici_fiscali.COUNT () > 0
            --                THEN
            --                    FOR i IN 1 .. l_codici_fiscali.COUNT ()
            --                    LOOP
            --                        DBMS_OUTPUT.put_line (
            --                               ' Riga i: '
            --                            || i
            --                            || ' -> '
            --                            || l_codici_fiscali
(i).CODICE_FISCALE);
            --                        ln_app1 := l_codici_fiscali
(i).CODICE_FISCALE;
            --
            --                        IF l_codici_fiscali (i).CODICE_FISCALE =
            --                           l_codice_fiscale_estratto
            --                        THEN
            --                            lb_continuo := TRUE;
            --                        END IF;
            --                    END LOOP;
```

```
            --                 END IF;
            --                 IF l_numero_libero
            --                 THEN
            --                     ln_numero_riga_estratta :=
ln_numero_riga_estratta + 1;
            --                     l_numero_riga_estratta.EXTEND ();
            --                     l_numero_riga_estratta
(ln_numero_riga_estratta).numero :=
            --                         ln_app;
            --                 END IF;
            END LOOP;



            ln_numero_riga_estratta := ln_numero_riga_estratta + 1;
            l_codici_fiscali.EXTEND ();
            l_codici_fiscali (ln_numero_riga_estratta).codice_fiscale :=
                l_codice_fiscale_estratto;
            --
            l_lista_revisori.EXTEND;
            l_quanti := l_quanti + 1;
            l_lista_revisori (l_quanti).cognome := l_urna (ln_app).cognome;
            l_lista_revisori (l_quanti).nome := l_urna (ln_app).nome;
            l_lista_revisori (l_quanti).sesso := l_urna (ln_app).sesso;
            l_lista_revisori (l_quanti).pec := l_urna (ln_app).pec;
            l_lista_revisori (l_quanti).id_mdas := l_urna (ln_app).id_mdas;
            l_lista_revisori (l_quanti).codice_fiscale :=
                l_codice_fiscale_estratto;
            --                 l_urna (ln_app).codice_fiscale;
            ls_app1 := l_lista_revisori (l_quanti).codice_fiscale;
            l_lista_revisori (l_quanti).numero_protocollo :=
                l_urna (ln_app).numero_protocollo;
            l_lista_revisori (l_quanti).data_protocollo :=
                l_urna (ln_app).data_protocollo;
            l_lista_revisori (l_quanti).flag_giovane_inesperto :=
                l_urna (ln_app).flag_giovane_inesperto;

            -- DA ELIMINARE
            l_nrc_reali := l_nrc_reali + 1;

            IF (l_urna (ln_app).sesso = 'M')
            THEN
                l_n_maschi_estratti := l_n_maschi_estratti + 1;
            ELSE
                l_n_femmine_estratte := l_n_femmine_estratte + 1;
            END IF;

            IF (l_urna (ln_app).flag_giovane_inesperto = 1)
            THEN
                l_n_giovani_estratti := l_n_giovani_estratti + 1;
            ELSE
                l_n_esperti_estratti := l_n_esperti_estratti + 1;
            END IF;
```

```
            -- FINE DA ELIMINARE
            pulisci_urna (l_urna,
                          l_codice_fiscale_estratto,
                          l_return,
                          l_messaggio);

            IF l_return = -1                        --  Errore in PULISCI_URNA
            THEN
                NULL;
            END IF;
        END LOOP;

        --      DBMS_OUTPUT.put_line ('Alla fine ho estratto '||
l_lista_revisori.COUNT ()|| ' revisori ');
        --      SORTEGGIO FINE
        --      Testo se la rosa è ben formata e, in tal caso, anche se non
rispetta i vincoli di genere ed esperienze la prendo per buona
        --

        l_return :=
            FLAG_ROSA_DA_SCARTARE (p_id_avviso,
                                   p_codice_fiscale,
                                   l_n_femmine_estratte, --compreso il revisore
uscente
                                   l_n_maschi_estratti, --compreso il revisore
uscente
                                   l_n_giovani_estratti, --compreso il revisore
uscente
                                   l_n_esperti_estratti --compreso il revisore
uscente
                                                        );

        IF l_return = 1
        THEN
            GOTO INIZIO;                            --  Da rifare il sorteggio
        ELSIF l_return = 0
        THEN
            NULL;                                                -- TUTTO OK
        ELSE
            ls_app :=
                SUBSTR ('CREA_LISTA FLAG_ROSA_DA_SCARTARE - ' || SQLERRM,
                        1,
                        200);
            RAISE LE_ERRORE;
        END IF;

        p_lista_revisori := l_lista_revisori;
    --
    EXCEPTION
        WHEN LE_DISASTRO
        THEN
            p_messaggio :=
                   ' Creazione rosa fallita dopo '
                || QuantiTentativi
```

```
                || ' tentativi per ID_AVVISO='
                || TO_CHAR (p_id_avviso);
        --            crea_errore ('CREA_LISTA', p_messaggio);
        WHEN LE_ERRORE
        THEN
            p_messaggio := ls_app;
        --            crea_errore ('CREA_LISTA', ls_app);
        WHEN OTHERS
        THEN
            p_messaggio := ' SQL00001 ' || SQLERRM;
    --          crea_errore ('CREA_LISTA', p_messaggio);
    END crea_lista;
END estrai_istanze_new;
/



CREATE OR REPLACE
FUNCTION elimina_esperti_da_urna (
    p_avviso                        IN NUMBER,
    p_revisore_uscente              IN VARCHAR2,
    p_numero_femmine_gia_estratte   IN NUMBER,  --compreso il revisore uscente
    p_numero_maschi_gia_estratti    IN NUMBER,  --compreso il revisore uscente
    p_numero_giovani_gia_estratti   IN NUMBER,  --compreso il revisore uscente
    p_numero_esperti_gia_estratti   IN NUMBER   --compreso il revisore uscente
                                        )
    RETURN NUMBER
IS
    v_risposta                    NUMBER;
    v_get_nrc                     NUMBER;
    v_get_na0                     NUMBER;
    v_get_na0_GIOVANI              NUMBER;
  --  v_get_na0_ESPERTI             NUMBER;
    --v_get_nB0                     NUMBER;
    v_get_nB0_GIOVANI             NUMBER;
  --  v_get_nB0_ESPERTI             NUMBER;
    v_flag_rev_uscente_not_null   INTEGER;
    V_GET_NGRC_MIN                NUMBER;
    V_GET_NGRC_MAX                NUMBER;
    V_GET_NERC_MIN                NUMBER;
    V_GET_NERC_MAX                NUMBER;
    V_FLAG_REV_USCENTE_ESPERTO    INTEGER;
  --  V_FLAG_REV_USCENTE_GIOVANE    INTEGER;
    v_numero_gia_estratti         number;
    v_numero_na0_rev_uscente      number;
BEGIN

    SELECT utility_estrazione.GET_NRC (p_avviso, p_revisore_uscente)INTO
v_get_nrc FROM DUAL;                                --numero di palline da
estrarre

    --
    --
   -- SELECT utility_estrazione.GET_NB0 (p_avviso, p_revisore_uscente) INTO
```

```
v_get_nB0 FROM DUAL;

    SELECT utility_estrazione.GET_NA0 (p_avviso) INTO v_get_na0 FROM DUAL;
----NUMERO PALLINE URNA A prima della seduta di sorteggio corrente

    --nell'urna A non è presente il revisore uscente
    --

    SELECT NVL2 (p_revisore_uscente, 1, 0) INTO v_flag_rev_uscente_not_null FROM
DUAL;
    SELECT utility_estrazione.GET_NA0_GIOVANI (P_AVVISO) INTO v_get_na0_GIOVANI
FROM DUAL;
  --  SELECT utility_estrazione.GET_NA0_ESPERTI (P_AVVISO) INTO
v_get_na0_ESPERTI FROM DUAL;
    SELECT utility_estrazione.GET_NB0_GIOVANI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_GIOVANI FROM DUAL;
  --  SELECT utility_estrazione.GET_NB0_ESPERTI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_ESPERTI FROM DUAL;
    SELECT utility_estrazione.GET_NGIOVRC_MIN (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGRC_MIN FROM DUAL;
    SELECT utility_estrazione.GET_NGIOVRC_MAX (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGRC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NGRC_MIN INTO V_GET_NERC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NGRC_MAX INTO V_GET_NERC_MIN FROM DUAL;

    SELECT 1 INTO V_FLAG_REV_USCENTE_ESPERTO FROM DUAL;

  --  SELECT 0 INTO V_FLAG_REV_USCENTE_GIOVANE FROM DUAL;
  v_numero_gia_estratti:=p_numero_esperti_gia_estratti +
p_numero_giovani_gia_estratti;--numero palline già estratte
  v_numero_na0_rev_uscente:=v_get_na0 + v_flag_rev_uscente_not_null;--numero
palline urna A prima seduta sorteggio corrente + l'eventuale pallina del
revisore uscente


    IF (    ((v_numero_na0_rev_uscente) <= v_get_nrc)
        AND ((v_numero_gia_estratti) <
             v_numero_na0_rev_uscente))
    THEN
        v_risposta := 0;
    --caso 1
    ELSE
        IF (    ((v_numero_na0_rev_uscente) <= v_get_nrc)
            AND NOT ((  v_numero_gia_estratti) <
                      v_numero_na0_rev_uscente))
        THEN
            --caso 2



            IF (p_numero_esperti_gia_estratti = V_GET_NeRC_MAX) --ho già
raggiunto il numero massimo di esperti
            THEN
                IF (v_get_nB0_GIOVANI  >
```

```
                    p_numero_giovani_gia_estratti) --ho ancora giovani a
disposizione
                THEN
                    v_risposta := 1;
                ELSE                        -- non ho più giovani a disposizione
                    v_risposta := 0;
                END IF;
            ELSE
                v_risposta := 0;
            END IF;
        ELSE
            IF (    NOT ((v_numero_na0_rev_uscente) <=
                         v_get_nrc)
                AND ((  v_numero_gia_estratti) <
                     v_numero_na0_rev_uscente))
            THEN
                --se l'urna A (+ revisore uscente) è capiente e la pallina che
sto per estrarre, la devo estrarre dall'urna A


                IF (p_numero_esperti_gia_estratti = V_GET_NeRC_MAX) --ho già
raggiunto il numero massimo di esperti
                THEN
                    IF (v_get_na0_GIOVANI  >
                        p_numero_giovani_gia_estratti)
                    THEN
                        v_risposta := 1;
                    ELSE
                        v_risposta := 0;
                    END IF;
                ELSE
                    v_risposta := 0;
                END IF;
            --caso 3

            ELSE
                IF (    NOT ((v_numero_na0_rev_uscente) <=
                             v_get_nrc)
                    AND NOT ((  v_numero_gia_estratti) <
                             v_numero_na0_rev_uscente))
                THEN
                    --caso impossibile
                    --vorrebbe dire che ho già fatto le estrazioni necessarie e
non ha senso invocare la funzione
                    --errore
                    v_risposta := 0;
                --caso 4

                ELSE
                    v_risposta := 0;                                --altri cvasi
                END IF;
            END IF;
        END IF;
    END IF;
```

```
    RETURN v_risposta;
EXCEPTION
    WHEN OTHERS
    THEN
        DBMS_OUTPUT.PUT_LINE (SQLERRM);
        RETURN v_risposta;
END;
/



-- End of DDL Script for Function REVISORI.ELIMINA_ESPERTI_DA_URNA

-- Start of DDL Script for Function REVISORI.ELIMINA_FEMMINE_DA_URNA
-- Generated 09-giu-2025 15:07:45 from REVISORI@ggi_istanze_produzione_apex51

CREATE OR REPLACE
FUNCTION elimina_femmine_da_urna (
    p_avviso                        IN NUMBER,
    p_revisore_uscente              IN VARCHAR2,
    p_numero_femmine_gia_estratte   IN NUMBER,  --compreso il revisore uscente
    p_numero_maschi_gia_estratti    IN NUMBER,  --compreso il revisore uscente
    p_numero_giovani_gia_estratti   IN NUMBER,  --compreso il revisore uscente
    p_numero_esperti_gia_estratti   IN NUMBER   --compreso il revisore uscente
)
    RETURN NUMBER
IS
    v_risposta                    NUMBER;
    v_get_nrc                     NUMBER;
    v_get_na0                     NUMBER;
    v_get_na0_MASCHI              NUMBER;
    v_get_nB0                     NUMBER;
    v_get_nB0_MASCHI              NUMBER;
    v_flag_rev_uscente_not_null   INTEGER;
    V_GET_NMRC_MIN                NUMBER;
    V_GET_NMRC_MAX                NUMBER;
    V_GET_NFRC_MAX                NUMBER;
    V_SESSO_REVISORE_USCENTE      VARCHAR2(1);
    V_FLAG_REV_USCENTE_MASCHIO    INTEGER;
    v_numero_gia_estratti         number;
    v_numero_na0_rev_uscente      number;
BEGIN
    --
    -- dato l'avviso e il numero di femmine, di maschi, di giovani, di esperti
già estratti
    -- restituisco 1 o 0 se l'urna di riferimento va privata delle femmine,
prima di procedere con l'estrazione

    SELECT utility_estrazione.GET_NRC (p_avviso, p_revisore_uscente)INTO
v_get_nrc FROM DUAL;--numero di palline da estrarre
    SELECT utility_estrazione.GET_NA0 (p_avviso) INTO v_get_na0 FROM DUAL;
--NUMERO PALLINE URNA A prima della seduta di sorteggio corrente
    SELECT utility_estrazione.GET_NB0 (p_avviso, p_revisore_uscente) INTO
```

```
v_get_nB0 FROM DUAL;--NUMERO PALLINE URNA B prima della seduta di sorteggio
corrente
    --nell'urna A non è presente il revisore uscente
    SELECT NVL2 (p_revisore_uscente, 1, 0) INTO v_flag_rev_uscente_not_null FROM
DUAL;
    SELECT utility_estrazione.GET_NA0_MASCHI (P_AVVISO) INTO v_get_na0_MASCHI
FROM DUAL;--NUMERO PALLINE maschi URNA A prima della seduta di sorteggio
corrente
    SELECT utility_estrazione.GET_NB0_MASCHI (P_AVVISO, p_revisore_uscente) INTO
v_get_nB0_MASCHI FROM DUAL;--NUMERO PALLINE maschi URNA B prima della seduta di
sorteggio corrente
    SELECT utility_estrazione.GET_NMRC_MIN (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MIN FROM DUAL;--numero minimo maschi da estrarre
    SELECT utility_estrazione.GET_NMRC_MAX (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MAX FROM DUAL;--numero massimo maschi da estrarre
    SELECT v_get_nrc - V_GET_NMRC_MIN INTO V_GET_NFRC_MAX FROM DUAL;--numero
massimo femmine da estrarre
    SELECT utility_estrazione.GET_SESSO_REVISORE_USCENTE(P_AVVISO,
p_revisore_uscente) INTO V_SESSO_REVISORE_USCENTE FROM dual;
    SELECT DECODE (V_SESSO_REVISORE_USCENTE, 'M', 1,NULL,0, 0) INTO
V_FLAG_REV_USCENTE_MASCHIO FROM DUAL;
    v_numero_gia_estratti:=p_numero_femmine_gia_estratte +
p_numero_maschi_gia_estratti;--numero palline già estratte
    v_numero_na0_rev_uscente:=v_get_na0 + v_flag_rev_uscente_not_null;--numero
palline urna A prima seduta sorteggio corrente + l'eventuale pallina del
revisore uscente


    IF (v_numero_na0_rev_uscente <= v_get_nrc AND v_numero_gia_estratti <
v_numero_na0_rev_uscente)
    --se l'urna A (+ revisore uscente) non è capiente o è appena capiente e la
pallina che sto per estrarre, la devo estrarre dall'urna A
    THEN
        --non pongo restrizioni se ho un numero di palline insufficiente o
appena sufficiente a coprire il mio obiettivo sulla numerosità di estrazione
        v_risposta := 0;
    ELSE
        IF (v_numero_na0_rev_uscente <= v_get_nrc AND NOT(v_numero_gia_estratti
< v_numero_na0_rev_uscente))
        --se l'urna A (+ revisore uscente) non è capiente o è appena capiente e
la pallina che sto per estrarre, la devo estrarre dall'urna B
        THEN
            IF (p_numero_femmine_gia_estratte = V_GET_NFRC_MAX)
            --ho già raggiunto il numero massimo di femmine
            THEN
                IF (v_get_nB0_MASCHI - V_FLAG_REV_USCENTE_MASCHIO >
p_numero_maschi_gia_estratti)

                THEN
                    --ho ancora maschi a disposizione
                    v_risposta := 1;
                ELSE
                    -- non ho più maschi a disposizione
                    v_risposta := 0;
```

```
                END IF;
            ELSE
                --non ho raggiunto il numero massimo di femmine
                v_risposta := 0;
            END IF;
        ELSE
            IF (NOT (v_numero_na0_rev_uscente <= v_get_nrc) AND
v_numero_gia_estratti < v_numero_na0_rev_uscente)
            THEN
                --se l'urna A (+ revisore uscente) è capiente e la pallina che
sto per estrarre, la devo estrarre dall'urna A
                IF (p_numero_femmine_gia_estratte = V_GET_NFRC_MAX)
                THEN
                    --ho già raggiunto il numero massimo di femmine
                    IF (v_get_na0_MASCHI - V_FLAG_REV_USCENTE_MASCHIO >
                        p_numero_maschi_gia_estratti)
                    THEN
                        --ho ancora maschi a disposizione
                        v_risposta := 1;
                    ELSE
                        -- non ho più maschi a disposizione
                        v_risposta := 0;
                    END IF;
                ELSE
                    --non ho raggiunto il numero massimo di femmine
                    v_risposta := 0;
                END IF;
            ELSE
                IF (NOT(v_numero_na0_rev_uscente <= v_get_nrc) AND NOT (
v_numero_gia_estratti < v_numero_na0_rev_uscente))
                THEN
                    --caso impossibile
                    --vorrebbe dire che ho già fatto le estrazioni necessarie e
non ha senso invocare la funzione
                    --errore
                    v_risposta := 0;
                --caso 4

                ELSE
                    v_risposta := 0;                            --altri cvasi
                END IF;
            END IF;
        END IF;
    END IF;

    RETURN v_risposta;
EXCEPTION
    WHEN OTHERS
    THEN
        DBMS_OUTPUT.PUT_LINE (SQLERRM);
        RETURN v_risposta;
END;
/
```

```
-- End of DDL Script for Function REVISORI.ELIMINA_FEMMINE_DA_URNA

-- Start of DDL Script for Function REVISORI.ELIMINA_GIOVANI_DA_URNA
-- Generated 09-giu-2025 15:07:45 from REVISORI@ggi_istanze_produzione_apex51

CREATE OR REPLACE
FUNCTION elimina_giovani_da_urna (
    p_avviso                        IN NUMBER,
    p_revisore_uscente              IN VARCHAR2,
    p_numero_femmine_gia_estratte   IN NUMBER,  --compreso il revisore uscente
    p_numero_maschi_gia_estratti    IN NUMBER,  --compreso il revisore uscente
    p_numero_giovani_gia_estratti   IN NUMBER,  --compreso il revisore uscente
    p_numero_esperti_gia_estratti   IN NUMBER   --compreso il revisore uscente
                                        )
    RETURN NUMBER
IS
    v_risposta                    NUMBER;
    v_get_nrc                     NUMBER;
    v_get_na0                     NUMBER;
    v_get_na0_GIOVANI              NUMBER;
    v_get_na0_ESPERTI             NUMBER;
    v_get_nB0                     NUMBER;
    v_get_nB0_GIOVANI              NUMBER;
    v_get_nB0_ESPERTI             NUMBER;
    v_flag_rev_uscente_not_null   INTEGER;
    V_GET_NGRC_MIN                NUMBER;
    V_GET_NGRC_MAX                NUMBER;
    V_GET_NERC_MIN                NUMBER;
    V_GET_NERC_MAX                NUMBER;
    V_FLAG_REV_USCENTE_ESPERTO    INTEGER;
    V_FLAG_REV_USCENTE_GIOVANE    INTEGER;
BEGIN

    SELECT utility_estrazione.GET_NRC (p_avviso, p_revisore_uscente)INTO
v_get_nrc FROM DUAL;                                  --numero di palline da
estrarre

    --
    --
    SELECT utility_estrazione.GET_NB0 (p_avviso, p_revisore_uscente) INTO
v_get_nB0 FROM DUAL;

    SELECT utility_estrazione.GET_NA0 (p_avviso) INTO v_get_na0 FROM DUAL;
----NUMERO PALLINE URNA A prima della seduta di sorteggio corrente

    --nell'urna A non è presente il revisore uscente
    --

    SELECT NVL2 (p_revisore_uscente, 1, 0) INTO v_flag_rev_uscente_not_null FROM
DUAL;
    SELECT utility_estrazione.GET_NA0_GIOVANI (P_AVVISO) INTO v_get_na0_GIOVANI
FROM DUAL;
```

```
    SELECT utility_estrazione.GET_NA0_ESPERTI (P_AVVISO) INTO v_get_na0_ESPERTI
FROM DUAL;
    SELECT utility_estrazione.GET_NB0_GIOVANI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_GIOVANI FROM DUAL;
    SELECT utility_estrazione.GET_NB0_ESPERTI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_ESPERTI FROM DUAL;
    SELECT utility_estrazione.GET_NGIOVRC_MIN (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGRC_MIN FROM DUAL;
    SELECT utility_estrazione.GET_NGIOVRC_MAX (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGRC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NGRC_MIN INTO V_GET_NERC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NGRC_MAX INTO V_GET_NERC_MIN FROM DUAL;

    SELECT 1 INTO V_FLAG_REV_USCENTE_ESPERTO FROM DUAL;

    SELECT 0 INTO V_FLAG_REV_USCENTE_GIOVANE FROM DUAL;


    IF (    ((v_get_na0 + v_flag_rev_uscente_not_null) <= v_get_nrc)
        AND ((p_numero_esperti_gia_estratti + p_numero_giovani_gia_estratti) <
             v_get_na0 + v_flag_rev_uscente_not_null))
    THEN
        v_risposta := 0;
    --caso 1
    ELSE
        IF (    ((v_get_na0 + v_flag_rev_uscente_not_null) <= v_get_nrc)
            AND NOT ((  p_numero_esperti_gia_estratti +
p_numero_giovani_gia_estratti) <
                     v_get_na0 + v_flag_rev_uscente_not_null))
        THEN
            --caso 2



            IF (p_numero_giovani_gia_estratti = V_GET_NGRC_MAX) --ho già
raggiunto il numero massimo di GIOVANI
            THEN
                IF (v_get_nB0_ESPERTI - V_FLAG_REV_USCENTE_ESPERTO >
                    p_numero_esperti_gia_estratti) --ho ancora ESPERTI a
disposizione
                THEN
                    v_risposta := 1;
                ELSE                        -- non ho più ESPERTI a disposizione
                    v_risposta := 0;
                END IF;
            ELSE
                v_risposta := 0;
            END IF;
        ELSE
            IF (    NOT ((v_get_na0 + v_flag_rev_uscente_not_null) <=
                         v_get_nrc)
                AND ((  p_numero_esperti_gia_estratti +
p_numero_giovani_gia_estratti) <
                     v_get_na0 + v_flag_rev_uscente_not_null))
```

```
            THEN
                --se l'urna A (+ revisore uscente) è capiente e la pallina che
sto per estrarre, la devo estrarre dall'urna A


                IF (p_numero_GIOVANI_gia_estratti = V_GET_NGRC_MAX) --ho già
raggiunto il numero massimo di GIOVANI
                THEN
                    IF (v_get_na0_ESPERTI - V_FLAG_REV_USCENTE_ESPERTO >
                        p_numero_esperti_gia_estratti)
                    THEN
                        v_risposta := 1;
                    ELSE                    -- non ho più femmine a disposizione
                        v_risposta := 0;
                    END IF;
                ELSE
                    v_risposta := 0;
                END IF;
            --caso 3

            ELSE
                IF (    NOT ((v_get_na0 + v_flag_rev_uscente_not_null) <=
                             v_get_nrc)
                    AND NOT ((  p_numero_esperti_gia_estratti
                              + p_numero_giovani_gia_estratti) <
                             v_get_na0 + v_flag_rev_uscente_not_null))
                THEN
                    --caso impossibile
                    --vorrebbe dire che ho già fatto le estrazioni necessarie e
non ha senso invocare la funzione
                    --errore
                    v_risposta := 0;
                --caso 4

                ELSE
                    v_risposta := 0;                                --altri cvasi
                END IF;
            END IF;
        END IF;
    END IF;

    RETURN v_risposta;
EXCEPTION
    WHEN OTHERS
    THEN
        DBMS_OUTPUT.PUT_LINE (SQLERRM);
        RETURN v_risposta;
END;
/



-- End of DDL Script for Function REVISORI.ELIMINA_GIOVANI_DA_URNA
```

```
-- Start of DDL Script for Function REVISORI.ELIMINA_MASCHI_DA_URNA
-- Generated 09-giu-2025 15:07:46 from REVISORI@ggi_istanze_produzione_apex51

CREATE OR REPLACE
FUNCTION elimina_maschi_da_urna (
    p_avviso                        IN NUMBER,
    p_revisore_uscente              IN VARCHAR2,
    p_numero_femmine_gia_estratte   IN NUMBER,  --compreso il revisore uscente
    p_numero_maschi_gia_estratti    IN NUMBER,  --compreso il revisore uscente
    p_numero_giovani_gia_estratti   IN NUMBER,  --compreso il revisore uscente
    p_numero_esperti_gia_estratti   IN NUMBER   --compreso il revisore uscente
                                        )
    RETURN NUMBER
IS
    v_risposta                    NUMBER;
    v_get_nrc                     NUMBER;
    v_get_na0                     NUMBER;
    v_get_na0_MASCHI              NUMBER;
    v_get_na0_FEMMINE             NUMBER;
    v_get_nB0                     NUMBER;
    v_get_nB0_MASCHI              NUMBER;
    v_get_nB0_FEMMINE             NUMBER;
    v_flag_rev_uscente_not_null   INTEGER;
    V_GET_NMRC_MIN                NUMBER;
    V_GET_NMRC_MAX                NUMBER;
    V_GET_NFRC_MIN                NUMBER;
    V_GET_NFRC_MAX                NUMBER;
    V_SESSO_REVISORE_USCENTE      VARCHAR2 (1);
    V_FLAG_REV_USCENTE_FEMMINA    INTEGER;
    V_FLAG_REV_USCENTE_MASCHIO    INTEGER;
BEGIN

    SELECT utility_estrazione.GET_NRC (p_avviso, p_revisore_uscente)INTO
v_get_nrc FROM DUAL;                                --numero di palline da
estrarre

    --
    --
    SELECT utility_estrazione.GET_NB0 (p_avviso, p_revisore_uscente) INTO
v_get_nB0 FROM DUAL;

    SELECT utility_estrazione.GET_NA0 (p_avviso) INTO v_get_na0 FROM DUAL;
----NUMERO PALLINE URNA A prima della seduta di sorteggio corrente

    --nell'urna A non è presente il revisore uscente
    --

    SELECT NVL2 (p_revisore_uscente, 1, 0) INTO v_flag_rev_uscente_not_null FROM
DUAL;
    SELECT utility_estrazione.GET_NA0_MASCHI (P_AVVISO) INTO v_get_na0_MASCHI
FROM DUAL;
    SELECT utility_estrazione.GET_NA0_FEMMINE (P_AVVISO) INTO v_get_na0_FEMMINE
FROM DUAL;
    SELECT utility_estrazione.GET_NB0_MASCHI (P_AVVISO, p_revisore_uscente) INTO
```

```
v_get_nB0_MASCHI FROM DUAL;
    SELECT utility_estrazione.GET_NB0_FEMMINE (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_FEMMINE FROM DUAL;
    SELECT utility_estrazione.GET_NMRC_MIN (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MIN FROM DUAL;
    SELECT utility_estrazione.GET_NMRC_MAX (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NMRC_MIN INTO V_GET_NFRC_MAX FROM DUAL;
    SELECT v_get_nrc - V_GET_NMRC_MAX INTO V_GET_NFRC_MIN FROM DUAL;
    SELECT utility_estrazione.GET_SESSO_REVISORE_USCENTE(P_AVVISO,
p_revisore_uscente) INTO V_SESSO_REVISORE_USCENTE FROM dual;

    SELECT DECODE (V_SESSO_REVISORE_USCENTE, 'F', 1,NULL,0, 0) INTO
V_FLAG_REV_USCENTE_FEMMINA FROM DUAL;

    SELECT DECODE (V_SESSO_REVISORE_USCENTE, 'M', 1,NULL,0, 0) INTO
V_FLAG_REV_USCENTE_MASCHIO FROM DUAL;


    IF (    ((v_get_na0 + v_flag_rev_uscente_not_null) <= v_get_nrc)
        AND ((p_numero_femmine_gia_estratte + p_numero_maschi_gia_estratti) <
             v_get_na0 + v_flag_rev_uscente_not_null))
    THEN
        --se l'urna A (+ revisore uscente) non è capiente o è appena capiente e
la pallina che sto per estrarre, la devo estrarre dall'urna A
        v_risposta := 0;
    --caso 1
    ELSE
        IF (    ((v_get_na0 + v_flag_rev_uscente_not_null) <= v_get_nrc)
            AND NOT ((  p_numero_femmine_gia_estratte
                      + p_numero_maschi_gia_estratti) <
                     v_get_na0 + v_flag_rev_uscente_not_null))
        THEN
            --se l'urna A (+ revisore uscente) non è capiente o è appena
capiente e la pallina che sto per estrarre, la devo estrarre dall'urna B
            --caso 2



            IF (p_numero_maschi_gia_estratti = V_GET_NMRC_MAX) --ho già
raggiunto il numero massimo di maschi
            THEN
                IF (v_get_nB0_FEMMINE - V_FLAG_REV_USCENTE_FEMMINA >
                    p_numero_femmine_gia_estratte) --ho ancora femmine a
disposizione
                THEN
                    v_risposta := 1;
                ELSE                        -- non ho più femmine a disposizione
                    v_risposta := 0;
                END IF;
            ELSE
                v_risposta := 0;
            END IF;
        ELSE
```

```
            IF (    NOT ((v_get_na0 + v_flag_rev_uscente_not_null) <=
                         v_get_nrc)
                AND ((  p_numero_femmine_gia_estratte
                      + p_numero_maschi_gia_estratti) <
                     v_get_na0 + v_flag_rev_uscente_not_null))
            THEN
                --se l'urna A (+ revisore uscente) è capiente e la pallina che
sto per estrarre, la devo estrarre dall'urna A


                IF (p_numero_maschi_gia_estratti = V_GET_NMRC_MAX) --ho già
raggiunto il numero massimo di maschi
                THEN
                    IF (v_get_na0_FEMMINE - V_FLAG_REV_USCENTE_FEMMINA >
                        p_numero_femmine_gia_estratte) --ho ancora femmine a
disposizione
                    THEN
                        v_risposta := 1;
                    ELSE                    -- non ho più femmine a disposizione
                        v_risposta := 0;
                    END IF;
                ELSE
                    v_risposta := 0;
                END IF;
            --caso 3

            ELSE
                IF (    NOT ((v_get_na0 + v_flag_rev_uscente_not_null) <=
                             v_get_nrc)
                    AND NOT ((  p_numero_femmine_gia_estratte
                              + p_numero_maschi_gia_estratti) <
                             v_get_na0 + v_flag_rev_uscente_not_null))
                THEN
                    --caso impossibile
                    --vorrebbe dire che ho già fatto le estrazioni necessarie e
non ha senso invocare la funzione
                    --errore
                    v_risposta := 0;
                --caso 4

                ELSE
                    v_risposta := 0;                                --altri cvasi
                END IF;
            END IF;
        END IF;
    END IF;

    RETURN v_risposta;
EXCEPTION
    WHEN OTHERS
    THEN
        DBMS_OUTPUT.PUT_LINE (SQLERRM);
        RETURN v_risposta;
END;
```

```
/



-- End of DDL Script for Function REVISORI.ELIMINA_MASCHI_DA_URNA

-- Start of DDL Script for Function REVISORI.FLAG_ROSA_DA_SCARTARE
-- Generated 09-giu-2025 15:07:46 from REVISORI@ggi_istanze_produzione_apex51

CREATE OR REPLACE
FUNCTION flag_rosa_da_scartare (
    p_avviso                        IN NUMBER,
    p_revisore_uscente              IN VARCHAR2,
    p_numero_femmine_estratte   IN NUMBER,  --compreso il revisore uscente
    p_numero_maschi_estratti    IN NUMBER,  --compreso il revisore uscente
    p_numero_giovani_estratti   IN NUMBER,  --compreso il revisore uscente
    p_numero_esperti_estratti   IN NUMBER   --compreso il revisore uscente
)
    RETURN NUMBER
IS
    v_get_nrc                     NUMBER;
    v_get_na0                     NUMBER;
    v_get_na0_MASCHI              NUMBER;
    v_get_nB0                     NUMBER;
    v_get_nB0_MASCHI              NUMBER;
    v_flag_rev_uscente_not_null   INTEGER;
    V_GET_NMRC_MIN                NUMBER;
    V_GET_NMRC_MAX                NUMBER;
    V_GET_NFRC_MAX                NUMBER;
    V_GET_NFRC_MIN                number;
    V_SESSO_REVISORE_USCENTE      VARCHAR2(1);
    V_FLAG_REV_USCENTE_MASCHIO    INTEGER;
    v_numero_na0_rev_uscente      number;
    V_GET_NGIOVRC_MIN number;
    V_GET_NGIOVRC_MAX number;
    V_GET_NEXPRC_MAX number;
    V_GET_NEXPRC_MIN number;
    v_get_na0_femmine number;
    V_FLAG_REV_USCENTE_FEMMINA number;
    v_get_na0_GIOVANI number;
    v_get_na0_esperti number;
    v_get_nB0_FEMMINE number;
    v_get_nB0_giovani number;
    v_get_nB0_esperti number;


BEGIN
    SELECT utility_estrazione.GET_NRC (p_avviso, p_revisore_uscente)INTO
v_get_nrc FROM DUAL;--numero di palline da estrarre
    SELECT utility_estrazione.GET_NA0 (p_avviso) INTO v_get_na0 FROM DUAL;
--NUMERO PALLINE URNA A prima della seduta di sorteggio corrente
    SELECT utility_estrazione.GET_NB0 (p_avviso, p_revisore_uscente) INTO
v_get_nB0 FROM DUAL;--NUMERO PALLINE URNA B prima della seduta di sorteggio
corrente
```

```
    SELECT NVL2 (p_revisore_uscente, 1, 0) INTO v_flag_rev_uscente_not_null FROM
DUAL;
    SELECT utility_estrazione.GET_NA0_MASCHI (P_AVVISO) INTO v_get_na0_MASCHI
FROM DUAL;--NUMERO PALLINE maschi URNA A prima della seduta di sorteggio
corrente
    SELECT utility_estrazione.GET_NB0_MASCHI (P_AVVISO, p_revisore_uscente) INTO
v_get_nB0_MASCHI FROM DUAL;--NUMERO PALLINE maschi URNA B prima della seduta di
sorteggio corrente
    SELECT utility_estrazione.GET_NMRC_MIN (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MIN FROM DUAL;--numero minimo maschi da estrarre
    SELECT utility_estrazione.GET_NMRC_MAX (P_AVVISO, p_revisore_uscente) INTO
V_GET_NMRC_MAX FROM DUAL;--numero massimo maschi da estrarre
    SELECT v_get_nrc - V_GET_NMRC_MIN INTO V_GET_NFRC_MAX FROM DUAL;--numero
massimo femmine da estrarre
    SELECT v_get_nrc - V_GET_NMRC_MAX INTO V_GET_NFRC_MIN FROM DUAL;--numero
minimo femmine da estrarre
    SELECT utility_estrazione.GET_SESSO_REVISORE_USCENTE(P_AVVISO,
p_revisore_uscente) INTO V_SESSO_REVISORE_USCENTE FROM dual;
    SELECT DECODE (V_SESSO_REVISORE_USCENTE, 'M', 1,NULL,0, 0) INTO
V_FLAG_REV_USCENTE_MASCHIO FROM DUAL;
    SELECT DECODE (V_SESSO_REVISORE_USCENTE, 'F', 1,NULL,0, 0) INTO
V_FLAG_REV_USCENTE_FEMMINA FROM DUAL;
    v_numero_na0_rev_uscente:=v_get_na0 + v_flag_rev_uscente_not_null;--numero
palline urna A prima seduta sorteggio corrente + l'eventuale pallina del
revisore uscente
    SELECT utility_estrazione.GET_NGIOVRC_MIN (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGIOVRC_MIN FROM DUAL;--numero minimo giovani da estrarre
    SELECT utility_estrazione.GET_NGIOVRC_MAX (P_AVVISO, p_revisore_uscente)
INTO V_GET_NGIOVRC_MAX FROM DUAL;--numero massimo giovani da estrarre
    SELECT v_get_nrc - v_GET_NGIOVRC_MIN INTO V_GET_NEXPRC_MAX FROM
DUAL;--numero massimo esperti da estrarre
    SELECT v_get_nrc - v_GET_NGIOVRC_MAX INTO V_GET_NEXPRC_MIN FROM
DUAL;--numero minimo esperti da estrarre
    SELECT utility_estrazione.GET_NA0_FEMMINE (P_AVVISO) INTO v_get_na0_femmine
FROM DUAL;--NUMERO PALLINE femmine URNA A prima della seduta di sorteggio
corrente
    SELECT utility_estrazione.GET_NA0_GIOVANI (P_AVVISO) INTO v_get_na0_GIOVANI
FROM DUAL;
    SELECT utility_estrazione.GET_NA0_ESPERTI (P_AVVISO) INTO v_get_na0_ESPERTI
FROM DUAL;
    SELECT utility_estrazione.GET_NB0_FEMMINE (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_FEMMINE FROM DUAL;--NUMERO PALLINE femmine URNA B prima della
seduta di sorteggio corrente
    SELECT utility_estrazione.GET_NB0_GIOVANI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_giovani FROM DUAL;--NUMERO PALLINE giovani URNA B prima della
seduta di sorteggio corrente
    SELECT utility_estrazione.GET_NB0_ESPERTI (P_AVVISO, p_revisore_uscente)
INTO v_get_nB0_esperti FROM DUAL;--NUMERO PALLINE esperti URNA B prima della
seduta di sorteggio corrente


    if
        p_numero_maschi_estratti < V_GET_NMRC_MIN or p_numero_maschi_estratti >
V_GET_NMRC_MAX or
```

```
        p_numero_femmine_estratte < V_GET_NFRC_MIN or p_numero_femmine_estratte
> V_GET_NFRC_MAX or
        p_numero_giovani_estratti < V_GET_NGIOVRC_MIN or
p_numero_giovani_estratti > V_GET_NGIOVRC_MAX or
        p_numero_esperti_estratti < V_GET_NEXPRC_MIN or
p_numero_esperti_estratti > V_GET_NEXPRC_MAX
    then
       --allora significa che la rosa non rispetta le quote
        IF
            (v_numero_na0_rev_uscente >= v_get_nrc )
            --pesco tutto dall'urna A
        THEN
            if
                v_get_na0_MASCHI + V_FLAG_REV_USCENTE_MASCHIO >= V_GET_NMRC_MIN
and
                v_get_na0_femmine + V_FLAG_REV_USCENTE_FEMMINA >= V_GET_NFRC_MIN
and
                v_get_na0_GIOVANI >=V_GET_NGIOVRC_MIN and
                v_get_na0_ESPERTI + v_flag_rev_uscente_not_null
>=V_GET_NEXPRC_MIN
                --composizione urna A consente la costituzione di una rosa
rispettantele quote
            then
                return 1;
            else
                --composizione urna A non consente la costituzione di una rosa
rispettantele quote
                return 0;
            end if;
        else
            --pesco tutta l'urna A e in parte dall'urna B, quindi la mia
valutazione deve avvenire sull'insieme di tutte le palline (A+B)
            if
                v_get_nB0_MASCHI>=V_GET_NMRC_MIN - (v_get_na0_MASCHI +
V_FLAG_REV_USCENTE_MASCHIO) and
                v_get_nB0_FEMMINE >=V_GET_NFRC_MIN-(v_get_na0_femmine +
V_FLAG_REV_USCENTE_FEMMINA) and
                v_get_nB0_giovani>=V_GET_NGIOVRC_MIN-v_get_na0_GIOVANI and
                v_get_nB0_esperti>=V_GET_NEXPRC_MIN-(v_get_na0_ESPERTI +
v_flag_rev_uscente_not_null)
                -- composizione urna b +urna a   consente la costituzione di una
rosa rispettantele quote
            then
                return 1;
            else
                -- composizione urna b +urna a  non consente la costituzione di
una rosa rispettantele quote
                return 0;
            end if;

        end if;
    else
  --allora significa che la rosa rispetta le quote
        return 0;
```

```
end if;

EXCEPTION
    WHEN OTHERS
    THEN
        DBMS_OUTPUT.PUT_LINE (SQLERRM);
        RETURN -1;
END;
/



CREATE OR REPLACE
PACKAGE utility_estrazione AS


FUNCTION GET_SESSO_REVISORE_USCENTE(
      ID_AVVISO IN NUMBER
     , p_codice_fiscale IN VARCHAR2

    ) RETURN varchar2;
FUNCTION GET_NA0(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;
FUNCTION GET_NA0_MASCHI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;
    FUNCTION GET_NA0_FEMMINE(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;
FUNCTION GET_NA0_GIOVANI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;

FUNCTION GET_NA0_ESPERTI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;

FUNCTION GET_NB0(
      ID_AVVISO IN NUMBER
     , p_codice_fiscale IN VARCHAR2

    ) RETURN NUMBER;
FUNCTION GET_NB0_MASCHI(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;
    FUNCTION GET_NB0_GIOVANI(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;
    FUNCTION GET_NB0_ESPERTI(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
```

```
    ) RETURN NUMBER;
    FUNCTION GET_NB0_FEMMINE(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;


FUNCTION GET_NRI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;

FUNCTION GET_NMRI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;

FUNCTION GET_NGIOVRI(
      ID_AVVISO IN NUMBER
    ) RETURN NUMBER;

  FUNCTION GET_NTOT(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale   IN     VARCHAR2
    ) RETURN NUMBER;


    FUNCTION GET_NRC(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;

    FUNCTION GET_NMRC_MIN(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;/*NUMERO MINIMO DI SOGGETTI MASCHI DA ESTRARRE NELLA ROSA
CORRENTE*/

        FUNCTION GET_NMRC_MAX(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;/*NUMERO MASSIMO DI SOGGETTI MASCHI DA ESTRARRE NELLA ROSA
CORRENTE*/
 FUNCTION GET_NMRC_MINMAX(
      ID_AVVISO IN NUMBER
      ,QUOTA_GENERE IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;
FUNCTION GET_NGIOVRC_MINMAX(
      ID_AVVISO IN NUMBER
      ,QUOTA_GIOV IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;


     FUNCTION GET_NGIOVRC_MIN(
```

```
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;/*NUMERO MINIMO DI SOGGETTI MASCHI DA ESTRARRE NELLA ROSA
CORRENTE*/

        FUNCTION GET_NGIOVRC_MAX(
      ID_AVVISO IN NUMBER
      , p_codice_fiscale IN VARCHAR2
    ) RETURN NUMBER;/*NUMERO MASSIMO DI SOGGETTI MASCHI DA ESTRARRE NELLA ROSA
CORRENTE*/

FUNCTION GET_FLAG_SOST_PARZ(ID_AVVISO IN NUMBER) return number;

END ;
/



CREATE OR REPLACE
PACKAGE BODY utility_estrazione
AS
    FUNCTION GET_SESSO_REVISORE_USCENTE (ID_AVVISO          IN NUMBER,
                                         p_codice_fiscale   IN VARCHAR2)
        RETURN VARCHAR2
    IS
        T   VARCHAR2 (1);
    BEGIN
        SELECT MAX (a.sog_sesso)
          INTO T
          FROM v_consumo_sorteggiabili_elenc a
         WHERE a.id_pub_avviso = id_avviso AND a.SOG_CF = p_codice_fiscale;

        RETURN T;
    END GET_SESSO_REVISORE_USCENTE;



    FUNCTION GET_NA0 ( --NUMERO PALLINE URNA A prima della seduta di sorteggio
corrente
                              --nell'urna A non è presente il revisore uscente
    ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO;

        RETURN T;
    END GET_NA0;

    FUNCTION GET_NB0 ( --NUMERO PALLINE URNA B prima della seduta di sorteggio
```

```
corrente
                              --nell'urna B non è presente il revisore uscente
    ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT /*+ NO_CPU_COSTING */
               COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_elenc a
         WHERE     id_pub_avviso = id_avviso
               AND (   a.SOG_CF < p_codice_fiscale
                    OR a.SOG_CF > p_codice_fiscale
                    OR p_codice_fiscale IS NULL)
               AND NOT EXISTS
                       (SELECT 1
                          FROM v_consumo_sorteggiabili_istanz b
                         WHERE     id_pub_avviso = id_avviso
                               AND a.sog_cf = b.sog_codice_fiscale);

        --        SELECT COUNT (*)
        --          INTO T
        --          FROM v_consumo_sorteggiabili_elenc a
        --         WHERE     id_pub_avviso = id_avviso
        --               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale
IS NULL)
        --               --    questo aggiunto x eliminare quelli che già
        --               --    stanno in urna A
        --               AND NOT EXISTS
        --                       (SELECT 1
        --                          FROM v_consumo_sorteggiabili_istanz b
        --                         WHERE     id_pub_avviso = id_avviso
        --                               AND a.sog_cf = b.sog_codice_fiscale);

        RETURN T;
    END GET_NB0;

    FUNCTION GET_NB0_MASCHI ( --NUMERO PALLINE MASCHI URNA B prima della seduta
di sorteggio corrente
                              --nell'urna B non è presente il revisore uscente
    ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN

        SELECT /*+ NO_CPU_COSTING */
               COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_elenc a
         WHERE     id_pub_avviso = id_avviso
               AND A.sog_sesso = 'M'
               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale IS NULL)
```

```
              --    questo aggiunto x eliminare quelli che già
              --    stanno in urna A
              AND NOT EXISTS
                      (SELECT 1
                         FROM v_consumo_sorteggiabili_istanz b
                        WHERE     id_pub_avviso = id_avviso
                              AND a.sog_cf = b.sog_codice_fiscale);

        RETURN T;
    END GET_NB0_MASCHI;

    FUNCTION GET_NB0_ESPERTI ( --NUMERO PALLINE GIOVANI URNA B prima della
seduta di sorteggio corrente
                              --nell'urna B non è presente il revisore uscente
                              ID_AVVISO          IN NUMBER,
                              p_codice_fiscale   IN VARCHAR2)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT /*+ NO_CPU_COSTING */
               COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_elenc a
         WHERE     id_pub_avviso = id_avviso
               AND A.flag_giovane_inesperto = 0
               AND (   a.SOG_CF < p_codice_fiscale
                    OR a.SOG_CF > p_codice_fiscale
                    OR p_codice_fiscale IS NULL)
               AND NOT EXISTS
                       (SELECT 1
                          FROM v_consumo_sorteggiabili_istanz b
                         WHERE     id_pub_avviso = id_avviso
                               AND a.sog_cf = b.sog_codice_fiscale);

        --        SELECT COUNT (*)
        --          INTO T
        --          FROM v_consumo_sorteggiabili_elenc a
        --         WHERE     id_pub_avviso = id_avviso
        --               AND A.flag_giovane_inesperto = 0
        --               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale
IS NULL)
        --               --    questo aggiunto x eliminare quelli che già
        --               --    stanno in urna A
        --               AND NOT EXISTS
        --                       (SELECT 1
        --                          FROM v_consumo_sorteggiabili_istanz b
        --                         WHERE     id_pub_avviso = id_avviso
        --                               AND a.sog_cf = b.sog_codice_fiscale);

        RETURN T;
    END GET_NB0_ESPERTI;

    FUNCTION GET_NB0_GIOVANI ( --NUMERO PALLINE GIOVANI URNA B prima della
```

```
seduta di sorteggio corrente
                              --nell'urna B non è presente il revisore uscente
                              ID_AVVISO          IN NUMBER,
                              p_codice_fiscale   IN VARCHAR2)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT /*+ NO_CPU_COSTING */
               COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_elenc a
         WHERE     id_avviso = id_pub_avviso
               AND A.flag_giovane_inesperto = 1
               AND NOT EXISTS
                       (SELECT 1
                          FROM v_consumo_sorteggiabili_istanz b
                         WHERE     A.id_pub_avviso = id_pub_avviso
                               AND b.sog_codice_fiscale = a.sog_cf)
               AND (   p_codice_fiscale < a.SOG_CF
                    OR p_codice_fiscale > a.SOG_CF
                    OR p_codice_fiscale IS NULL);

        --        SELECT COUNT (*)
        --          INTO T
        --          FROM v_consumo_sorteggiabili_elenc a
        --         WHERE     id_pub_avviso = id_avviso
        --               AND A.flag_giovane_inesperto = 1
        --               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale
IS NULL)
        --               --    questo aggiunto x eliminare quelli che già
        --               --    stanno in urna A
        --               AND NOT EXISTS
        --                       (SELECT 1
        --                          FROM v_consumo_sorteggiabili_istanz b
        --                         WHERE     id_pub_avviso = id_avviso
        --                               AND a.sog_cf = b.sog_codice_fiscale);

        RETURN T;
    END GET_NB0_GIOVANI;


    FUNCTION GET_NB0_FEMMINE ( --NUMERO PALLINE FEMMINE URNA B prima della
seduta di sorteggio corrente
                              --nell'urna B non è presente il revisore uscente
                              ID_AVVISO          IN NUMBER,
                              p_codice_fiscale   IN VARCHAR2)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT /*+ NO_CPU_COSTING */
               COUNT (*)
          INTO T
```

```
          FROM v_consumo_sorteggiabili_elenc a
         WHERE     id_pub_avviso = id_avviso
               AND A.sog_sesso = 'F'
               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale IS NULL)
               AND NOT EXISTS
                       (SELECT 1
                          FROM v_consumo_sorteggiabili_istanz b
                         WHERE     id_pub_avviso = id_avviso
                               AND a.sog_cf = b.sog_codice_fiscale);

        --        SELECT COUNT (*)
        --          INTO T
        --          FROM v_consumo_sorteggiabili_elenc a
        --         WHERE     id_pub_avviso = id_avviso
        --               AND A.sog_sesso = 'F'
        --               AND (a.SOG_CF <> p_codice_fiscale OR p_codice_fiscale
IS NULL)
        --               --    questo aggiunto x eliminare quelli che già
        --               --    stanno in urna A
        --               AND NOT EXISTS
        --                       (SELECT 1
        --                          FROM v_consumo_sorteggiabili_istanz b
        --                         WHERE     id_pub_avviso = id_avviso
        --                               AND a.sog_cf = b.sog_codice_fiscale);

        RETURN T;
    END GET_NB0_FEMMINE;


    FUNCTION GET_NA0_ESPERTI (ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO AND A.flag_giovane_inesperto = 0;

        RETURN T;
    END GET_NA0_ESPERTI;

    FUNCTION GET_NA0_GIOVANI (ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO AND A.flag_giovane_inesperto = 1;

        RETURN T;
    END GET_NA0_GIOVANI;
```

```
    FUNCTION GET_NA0_MASCHI ( --NUMERO PALLINE MASCHI URNA A prima della seduta
di sorteggio corrente
                              --nell'urna A non è presente il revisore uscente
    ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO AND A.sog_sesso = 'M';

        RETURN T;
    END GET_NA0_MASCHI;

    FUNCTION GET_NA0_FEMMINE ( --NUMERO PALLINE FEMMINE URNA A prima della
seduta di sorteggio corrente
                              --nell'urna A non è presente il revisore uscente
                              ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO AND A.sog_sesso = 'F';

        RETURN T;
    END GET_NA0_FEMMINE;

    FUNCTION GET_NRI (                            --NUMERO ROSA STORICA IDONEI
                      --va a guardare se per lo stesso avviso vi sono altri
sorteggi con sorteggiati idonei
                      ID_AVVISO IN NUMBER)
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT COUNT (*)
          INTO T
          FROM CONSUMO_SORTEGGIO             A,
               CONSUMO_SORTEGGIATI           B,
               CONSUMO_PUBBLICAZIONE_AVVISO  C
         WHERE     A.ID = B.ID_CONSUMO_SORTEGGIO
               AND (   B.FLAG_INCOMPATIB_INCARICO IS NULL
                    OR B.FLAG_INCOMPATIB_INCARICO = 0)
               AND A.ID_CONSUMO_PUBBL_AVVISO = C.ID
               AND C.ID = ID_AVVISO;

        RETURN T;
    END GET_NRI;
```

```
FUNCTION GET_NMRI (                  /*numero maschi ROSA storica IDONEI*/
                   ID_AVVISO IN NUMBER)
    RETURN NUMBER
IS
    T   NUMBER;
BEGIN
    SELECT NVL (SUM (DECODE (b.sesso, 'M', 1, 0)), 0)
      INTO T
      FROM CONSUMO_SORTEGGIO             A,
           CONSUMO_SORTEGGIATI           B,
           CONSUMO_PUBBLICAZIONE_AVVISO  C
     WHERE     A.ID = B.ID_CONSUMO_SORTEGGIO
           AND (   B.FLAG_INCOMPATIB_INCARICO IS NULL
                OR B.FLAG_INCOMPATIB_INCARICO = 0)
           AND A.ID_CONSUMO_PUBBL_AVVISO = C.ID
           AND C.ID = ID_AVVISO;

    RETURN T;
END GET_NMRI;

FUNCTION GET_NGIOVRI (    /*numero GIOVANI INESPERTI ROSA storica IDONEI*/
                       ID_AVVISO IN NUMBER)
    RETURN NUMBER
IS
    T   NUMBER;
BEGIN
    SELECT NVL (SUM (D.FLAG_GIOVANE_INESPERTO), 0)
      INTO T
      FROM CONSUMO_SORTEGGIO             A,
           CONSUMO_SORTEGGIATI           B,
           CONSUMO_PUBBLICAZIONE_AVVISO  C,
           (  SELECT TRIM (UPPER (sog_cf))    sog_cf,
                     DECODE (
                         SUM (DECODE (FASCIA, 'GIOVANI/INESPERTI', 1, 0)),
                         0, 0,
                         1)                   FLAG_GIOVANE_INESPERTO
                FROM consumo_elenco_fvg
            GROUP BY TRIM (UPPER (sog_cf))) D
     WHERE     A.ID = B.ID_CONSUMO_SORTEGGIO
           AND (   B.FLAG_INCOMPATIB_INCARICO IS NULL
                OR B.FLAG_INCOMPATIB_INCARICO = 0)
           AND A.ID_CONSUMO_PUBBL_AVVISO = C.ID
           AND C.ID = ID_AVVISO
           AND D.sog_cf = TRIM (UPPER (CODICE_FISCALE));

    RETURN T;
END GET_NGIOVRI;

FUNCTION GET_NTOT (ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2
                                                                    )
    RETURN NUMBER
IS
    --restituisce il numero di persone che devono uscire dal sorteggio
```

```
(compresi i revisori uscenti,
        --compresi quelli che finiscono automaticamente nella rosa per via del
numero di palline dell'urna)
        T                         NUMBER;
        NUMERO_ISTANZE_VAL_POS    NUMBER;
        FLAG_PRIMO_SORTEGGIO      NUMBER;
        TIPO_ORGANO               VARCHAR2 (10);

        OBIETTIVO                 VARCHAR2 (20);

    BEGIN
        SELECT COUNT (*)
          INTO NUMERO_ISTANZE_VAL_POS
          FROM v_consumo_sorteggiabili_istanz a
         WHERE a.id_pub_avviso = ID_AVVISO;

        SELECT DECODE (COUNT (*), 0, 1, 0)
          INTO FLAG_PRIMO_SORTEGGIO
          FROM consumo_sorteggio, consumo_sorteggiati
         WHERE     consumo_sorteggiati.id_consumo_sorteggio =
                   consumo_sorteggio.id
               AND CONSUMO_SORTEGGIO.id_consumo_pubbl_avviso = ID_AVVISO;

        -- SELECT DECODE(COUNT(*),0,0,1) INTO FLAG_PRIMO_SORTEGGIO FROM
consumo_sorteggio WHERE CONSUMO_SORTEGGIO.id_consumo_pubbl_avviso=ID_AVVISO;

        SELECT d.tipo_organo
          INTO TIPO_ORGANO
          FROM consumo_pubblicazione_avviso c, v_consumo_dati_enti d
         WHERE     c.ente_codice = d.ente_codice
               AND c.id = ID_AVVISO
               AND (c.data_pubblicazione BETWEEN d.data_inizio_validita
                                             AND d.data_fine_validita)
               AND c.tipo_ente = d.ente_tipo;

        -- SELECT c.tipo_organo INTO TIPO_ORGANO FROM
CONSUMO_PUBBLICAZIONE_AVVISO C WHERE C.ID = ID_AVVISO;



        SELECT c.obiettivo
          INTO OBIETTIVO
          FROM consumo_pubblicazione_avviso c
         WHERE c.id = ID_AVVISO;




        SELECT CASE
                   WHEN     p_codice_fiscale IS NULL
                        AND FLAG_PRIMO_SORTEGGIO = 1
                        AND TIPO_ORGANO = 'MON'
                        AND NUMERO_ISTANZE_VAL_POS = 1
```

```
                    THEN
                        1
                    WHEN     p_codice_fiscale IS NULL
                         AND FLAG_PRIMO_SORTEGGIO = 1
                         AND TIPO_ORGANO = 'MON'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 1)
                    THEN
                        3
                    WHEN     p_codice_fiscale IS NULL
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'MON'
                         AND NUMERO_ISTANZE_VAL_POS = 1
                    THEN
                        3
                    WHEN     p_codice_fiscale IS NULL
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'MON'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 1)
                    THEN
                        3
                    WHEN     p_codice_fiscale IS NOT NULL
                         AND FLAG_PRIMO_SORTEGGIO = 1
                         AND TIPO_ORGANO = 'MON'
                         AND NUMERO_ISTANZE_VAL_POS = 1
                    THEN
                        2 -- in questo caso nella rosa finisce il revisore
uscente e l'unico revisore che ha presentato istanza
                    WHEN     p_codice_fiscale IS NOT NULL
                         AND FLAG_PRIMO_SORTEGGIO = 1
                         AND TIPO_ORGANO = 'MON'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 1)
                    THEN
                        3                        --compreso il revisore uscente
                    WHEN     p_codice_fiscale IS NOT NULL
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'MON'
                         AND NUMERO_ISTANZE_VAL_POS = 1
                    THEN
                        3                        --compreso il revisore uscente
                    WHEN     p_codice_fiscale IS NOT NULL
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'MON'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 1)
                    THEN
                        3                        --compreso il revisore uscente

                    --WHEN FLAG_PRIMO_SORTEGGIO=1 AND TIPO_ORGANO='COL' AND
NUMERO_ISTANZE_VAL_POS=3 THEN 3
                    --WHEN FLAG_PRIMO_SORTEGGIO=1 AND TIPO_ORGANO='COL' AND
NOT(NUMERO_ISTANZE_VAL_POS=3) THEN 9
                    --WHEN FLAG_PRIMO_SORTEGGIO=0 AND TIPO_ORGANO='COL' AND
NUMERO_ISTANZE_VAL_POS=3 THEN 9
                    -- WHEN FLAG_PRIMO_SORTEGGIO=0 AND TIPO_ORGANO='COL' AND
NOT(NUMERO_ISTANZE_VAL_POS=3) THEN 9
```

```
                    WHEN     NVL (OBIETTIVO, 'NUOVA_COST') = 'NUOVA_COST'
                         AND FLAG_PRIMO_SORTEGGIO = 1
                         AND TIPO_ORGANO = 'COL'
                         AND NUMERO_ISTANZE_VAL_POS = 3
                    THEN
                        3
                    WHEN     NVL (OBIETTIVO, 'NUOVA_COST') = 'NUOVA_COST'
                         AND FLAG_PRIMO_SORTEGGIO = 1
                         AND TIPO_ORGANO = 'COL'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 3)
                    THEN
                        9
                    WHEN     NVL (OBIETTIVO, 'NUOVA_COST') = 'NUOVA_COST'
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'COL'
                         AND NUMERO_ISTANZE_VAL_POS = 3
                    THEN
                        9
                    WHEN     NVL (OBIETTIVO, 'NUOVA_COST') = 'NUOVA_COST'
                         AND FLAG_PRIMO_SORTEGGIO = 0
                         AND TIPO_ORGANO = 'COL'
                         AND NOT (NUMERO_ISTANZE_VAL_POS = 3)
                    THEN
                        9
                    WHEN NVL (OBIETTIVO, 'NUOVA_COST') = 'SOST_PARZ'
                    THEN
                        3




                END
          INTO T
          FROM DUAL;

        /*SICCOME LA LEGGE PARLA DI UN TERZO PER GENERE, SI ASSUME CHE NTOT SIA
MULTIPLO DI 3*/
        RETURN T;
    END GET_NTOT;

    FUNCTION GET_NRC ( /*NUMERO PALLINE DA ESTRARRE NELLA SEDUTA DI SORTEGGIO
CORRENTE*/
                      /*NUMERO TOTALE DI SOGGETTI DA ESTRARRE NELLA ROSA
CORRENTE*/
                                                --compreso il revisore uscente
                      ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2
                                                                      )
        RETURN NUMBER
    IS
        T   NUMBER;
    BEGIN
        SELECT GET_NTOT (ID_AVVISO, p_codice_fiscale) - GET_NRI (ID_AVVISO)
          INTO T
          FROM DUAL;
```

```
        RETURN T;
    END GET_NRC;



    FUNCTION GET_NMRC_MIN ( /*NUMERO MINIMO DI MASCHI DELLA ROSA corrente al
termine della seduta sorteggio corrente*/
                           ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2
                                                                          )
        RETURN NUMBER
    IS
        T   NUMBER;
        Q   NUMBER;
    BEGIN
        IF GET_FLAG_SOST_PARZ (ID_AVVISO) = 0
        THEN
            Q := 1 / 3;

            SELECT CEIL (GET_NMRC_MINMAX (ID_AVVISO, Q, p_codice_fiscale))
              INTO T
              FROM DUAL;


            RETURN T;
        ELSE                          --se c'è sostituzione parziale
            SELECT DECODE (
                       SIGN (b.numero_minimo_maschi - GET_NMRI (id_avviso)),
                       1, b.numero_minimo_maschi - GET_NMRI (id_avviso),
                       0)
              INTO T
              FROM consumo_pubblicazione_avviso a, consumo_d_tipo_sost_parz b
             WHERE a.tipo_sost_parz = b.id AND a.id = id_avviso;

            RETURN T;
        END IF;
    END GET_NMRC_MIN;



    FUNCTION GET_NGIOVRC_MIN ( /*NUMERO MINIMO DI GIOVANI DELLA ROSA corrente al
termine della seduta sorteggio corrente*/
                              ID_AVVISO          IN NUMBER,
                              p_codice_fiscale   IN VARCHAR2
                                                    )
        RETURN NUMBER
    IS
        T   NUMBER;
        Q   NUMBER;
    BEGIN
        IF GET_FLAG_SOST_PARZ (ID_AVVISO) = 0
        THEN
            SELECT --decode(d.tipo_organo,'MON',0,1/9) INTO Q
```

```
                DECODE (d.tipo_organo, 'MON', 0, 1 / 3)
          INTO Q
          FROM consumo_pubblicazione_avviso c, v_consumo_dati_enti d
         WHERE     d.ente_codice = c.ente_codice
                AND c.id = ID_AVVISO
                AND (c.data_pubblicazione BETWEEN d.data_inizio_validita
                                              AND d.data_fine_validita)
                AND c.tipo_ente = d.ente_tipo;

        SELECT CEIL (GET_NGIOVRC_MINMAX (ID_AVVISO, Q, p_codice_fiscale))
          INTO T
          FROM DUAL;


        RETURN T;
    ELSE                         --se c'è sostituzione parziale
        SELECT DECODE (
                   SIGN (
                       b.numero_minimo_giovani - GET_NGIOVRI (id_avviso)),
                   1, b.numero_minimo_giovani - GET_NGIOVRI (id_avviso),
                   0)
          INTO T
          FROM consumo_pubblicazione_avviso a, consumo_d_tipo_sost_parz b
         WHERE a.tipo_sost_parz = b.id AND a.id = id_avviso;

        RETURN T;
    END IF;
END GET_NGIOVRC_MIN;

    FUNCTION GET_NGIOVRC_MAX ( /*NUMERO MASSIMO DI GIOVANI DELLA ROSA corrente
al termine della seduta sorteggio corrente*/
                              ID_AVVISO          IN NUMBER,
                              p_codice_fiscale   IN VARCHAR2
                                                    )
        RETURN NUMBER
    IS
        T   NUMBER;
        Q   NUMBER;
    BEGIN
        IF GET_FLAG_SOST_PARZ (ID_AVVISO) = 0
        THEN
            SELECT -- decode(d.tipo_organo,'MON',0,2/3) INTO Q
                   DECODE (d.tipo_organo, 'MON', 0, 1 / 3)
              INTO Q
              FROM consumo_pubblicazione_avviso c, v_consumo_dati_enti d
             WHERE     d.ente_codice = c.ente_codice
                   AND c.id = ID_AVVISO
                   AND (c.data_pubblicazione BETWEEN d.data_inizio_validita
                                                 AND d.data_fine_validita)
                   AND c.tipo_ente = d.ente_tipo;
            SELECT FLOOR (
                       GET_NGIOVRC_MINMAX (ID_AVVISO, Q, p_codice_fiscale))
              INTO T
              FROM DUAL;
```

```
        RETURN T;
    ELSE                          --se c'è sostituzione parziale
        SELECT DECODE (
                   SIGN (
                       b.numero_massimo_giovani - GET_NGIOVRI (id_avviso)),
                   1, b.numero_massimo_giovani - GET_NGIOVRI (id_avviso),
                   0)
          INTO T
          FROM consumo_pubblicazione_avviso a, consumo_d_tipo_sost_parz b
         WHERE a.tipo_sost_parz = b.id AND a.id = id_avviso;

        RETURN T;
    END IF;
END GET_NGIOVRC_MAX;



FUNCTION GET_NMRC_MAX (ID_AVVISO IN NUMBER, p_codice_fiscale IN VARCHAR2
                                                                      )
    RETURN NUMBER
IS
    T   NUMBER;
    Q   NUMBER;
BEGIN
    IF GET_FLAG_SOST_PARZ (ID_AVVISO) = 0
    THEN
        Q := 2 / 3;

        SELECT FLOOR (GET_NMRC_MINMAX (ID_AVVISO, Q, p_codice_fiscale))
          INTO T
          FROM DUAL;


        RETURN T;
    ELSE                          --se c'è sostituzione parziale
        SELECT DECODE (
                   SIGN (b.numero_massimo_maschi - GET_NMRI (id_avviso)),
                   1, b.numero_massimo_maschi - GET_NMRI (id_avviso),
                   0)
          INTO T
          FROM consumo_pubblicazione_avviso a, consumo_d_tipo_sost_parz b
         WHERE a.tipo_sost_parz = b.id AND a.id = id_avviso;

        RETURN T;
    END IF;
END GET_NMRC_MAX;



FUNCTION GET_NMRC_MINMAX (ID_AVVISO          IN NUMBER,
                          QUOTA_GENERE       IN NUMBER,
                          p_codice_fiscale   IN VARCHAR2
```

```
                                                    )
    RETURN NUMBER
IS
    T   NUMBER;
    A   NUMBER;
    B   NUMBER;
    C   NUMBER;
BEGIN
    SELECT   GET_NTOT (ID_AVVISO, p_codice_fiscale) * QUOTA_GENERE
           - GET_NMRI (ID_AVVISO)
      INTO A
      FROM DUAL;

    SELECT GET_NRC (ID_AVVISO, p_codice_fiscale) INTO B FROM DUAL;

    SELECT DECODE (SIGN (A), -1, 0, A) INTO C FROM DUAL;

    SELECT DECODE (SIGN (C - B), 1, B, C) INTO T FROM DUAL;


    RETURN T;
END GET_NMRC_MINMAX;



FUNCTION GET_NGIOVRC_MINMAX (ID_AVVISO          IN NUMBER,
                             QUOTA_GIOV         IN NUMBER,
                             p_codice_fiscale   IN VARCHAR2
                                                  )
    RETURN NUMBER
IS
    T   NUMBER;
    A   NUMBER;
    B   NUMBER;
    C   NUMBER;
BEGIN
    SELECT   GET_NTOT (ID_AVVISO, p_codice_fiscale) * QUOTA_GIOV
           - GET_NGIOVRI (ID_AVVISO)
      INTO A
      FROM DUAL;

    SELECT GET_NRC (ID_AVVISO, p_codice_fiscale) INTO B FROM DUAL;

    SELECT DECODE (SIGN (A), -1, 0, A) INTO C FROM DUAL;

    SELECT DECODE (SIGN (C - B), 1, B, C) INTO T FROM DUAL;


    RETURN T;
END GET_NGIOVRC_MINMAX;


FUNCTION GET_FLAG_SOST_PARZ (ID_AVVISO IN NUMBER)
    RETURN NUMBER
```

```
    IS
        V_FLAG   NUMBER (1);
    BEGIN
        SELECT DECODE (a.obiettivo, 'SOST_PARZ', 1, 0)
          INTO V_FLAG
          FROM consumo_pubblicazione_avviso a
         WHERE a.id = ID_AVVISO;

        RETURN V_FLAG;
    END GET_FLAG_SOST_PARZ;



END;
/
```