

ContattiPiù

Gestore di contatti avanzato

Carlo Onofrio N86003692
Lino Strina N86003662
Anno Accademico 2021/2022

Docente: Silvio Barra

Indice

1	Introduzione	3
1.1	Descrizione del progetto	3
2	Progettazione concettuale	4
2.1	Class Diagram	4
2.2	Dizionario delle classi	5
2.3	Dizionario delle associazioni	6
2.4	Dizionario dei vincoli	7
3	Progettazione logica	8
3.1	Schema Logico	8
4	Progettazione fisica SQL	9
4.1	Definizione tabelle	9
4.1.1	Creazione schema	9
4.1.2	Indirizzo	9
4.1.3	Contatto	9
4.1.4	Alloggio	10
4.1.5	Telefono	10
4.1.6	Recapito	10
4.1.7	Gruppo	11
4.1.8	Aggregazione	11
4.1.9	Email	11
4.1.10	MessagingPR	11
4.2	Vincoli di Integrità referenziale	12
4.2.1	Integrità referenziale Contatto	12
4.2.2	Integrità referenziale Alloggio	12
4.2.3	Integrità referenziale Aggregazione	12
4.2.4	Integrità referenziale Recapito	12
4.2.5	Integrità referenziale Email	12
4.2.6	Integrità referenziale MessagingPR	12
4.3	Implementazione vincoli di N-Upla	13
4.3.1	Controlla Formato Email	13
4.4	Implementazione vincoli Interrelazionali	13
4.4.1	Controlla Telefono Reindirizzato	13
4.4.2	Controlla Eliminazione Telefono	14
4.4.3	Controllo Ordine Inserimento recapiti	15
4.4.4	Controllo eliminazione recapiti posseduti	16

Capitolo 1

Introduzione

1.1 Descrizione del progetto

ContattiPiú é un gestore di contatti avanzato. É fornito di una base di dati capace di gestire i contatti ed informazioni ad essi correlate, i suoi indirizzi fisici, numeri di telefono, gruppi a cui appartiene, i suoi indirizzi email e servizi di messaggistica.

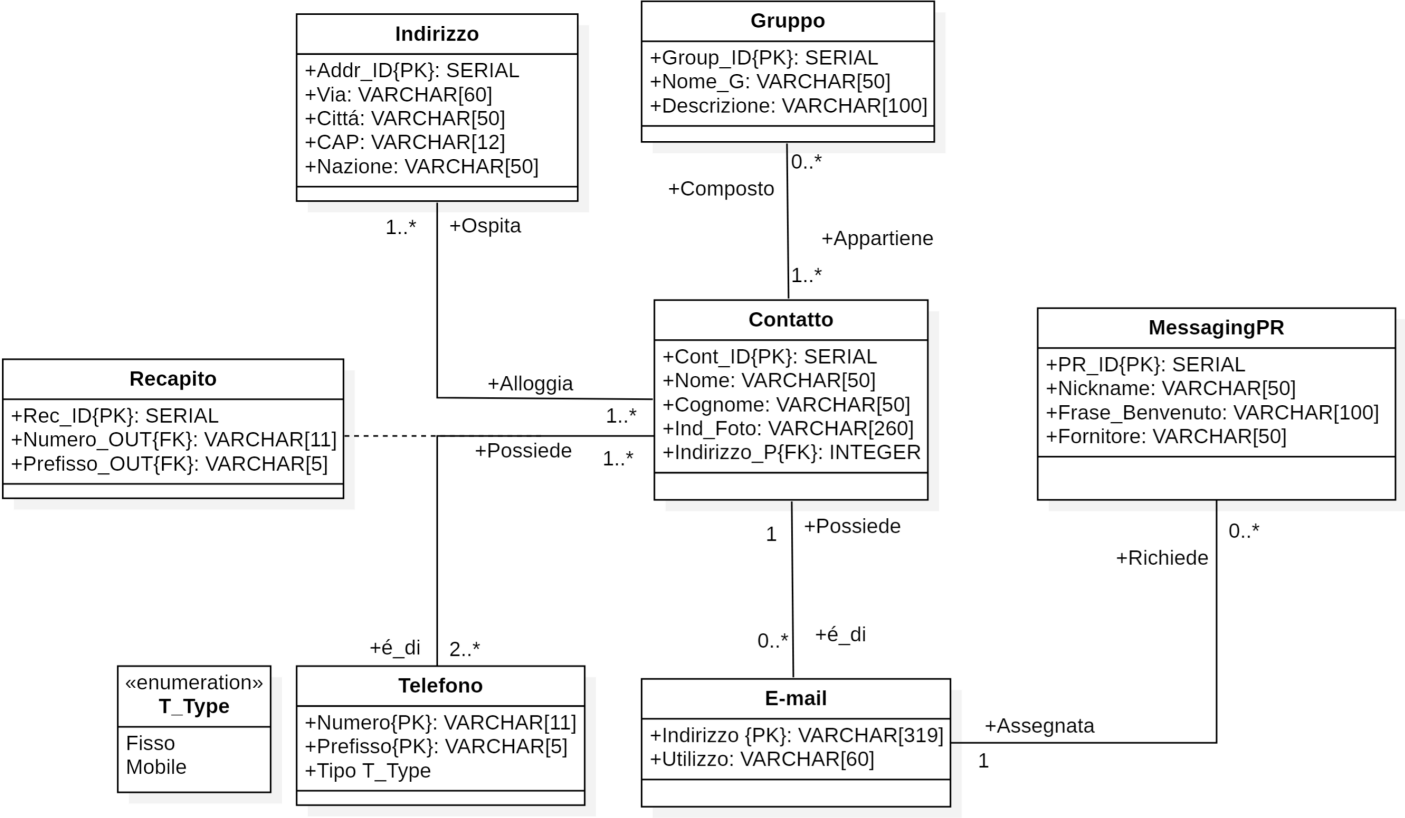
Ogni contatto dovrà essere fornito di almeno due numeri di telefono, fisso e mobile. Il telefono fisso verrà reindirizzato ad un telefono mobile, analogamente il telefono mobile potrà essere reindirizzato ad un telefono fisso. Ogni contatto dovrà essere fornito di almeno un indirizzo fisico, che sarà riconosciuto come primario. Ogni contatto può appartenere ad uno o più gruppi di cui fanno parte altri contatti. Ogni contatto può avere indirizzi email e, nel caso, uno o più providers di messaggistica ad esso collegato. I contatti possono condividere numeri di telefono ed indirizzi fisici, ma non indirizzi email.

Capitolo 2

Progettazione concettuale

2.1 Class Diagram

In fase di progettazione concettuale siamo giunti a produrre un Class Diagram già ristrutturato, non abbiamo quindi trovato necessario inserire una sezione di ristrutturazione.



2.2 Dizionario delle classi

Classe	Descrizione	Attributi
Contatto	Descrive il contatto	Cont_ID{PK} : <i>SERIAL</i> ⇔ Identifica univocamente il contatto. Nome : <i>VARCHAR[50]</i> ⇔ Nome associato al contatto. Cognome : <i>VARCHAR[50]</i> ⇔ Cognome associato al contatto. Ind_Foto : <i>VARCHAR[260]</i> ⇔ Indirizzo locale sulla macchina della foto assegnata al contatto. Indirizzo_P{FK} : <i>INTEGER</i> ⇔ Indirizzo fisico principale del contatto.
Indirizzo	Descrive gli indirizzi del contatto	Addr_ID{PK} : <i>SERIAL</i> ⇔ Identifica univocamente l'indirizzo. Via : <i>VARCHAR[60]</i> ⇔ Via dell'indirizzo. Città : <i>VARCHAR[50]</i> ⇔ Città dell'indirizzo. CAP : <i>VARCHAR[12]</i> ⇔ CAP dell'indirizzo. Nazione : <i>VARCHAR[50]</i> ⇔ Nazione dell'indirizzo.
Telefono	Descrive i numeri telefonici del contatto	Numero{PK} : <i>VARCHAR[11]</i> ⇔ Numero di telefono del contatto. Prefisso{PK} : <i>VARCHAR[5]</i> ⇔ Prefisso del numero di telefono. Tipo : <i>T_Type</i> ⇔ Tipo di telefono: mobile o fisso.
Recapito	Descrive il reindirizzamento telefonico dal contatto	Rec_ID{PK} : <i>SERIAL</i> ⇔ Identifica univocamente il recapito. Numero_OUT{FK} : <i>VARCHAR[11]</i> ⇔ Numero di telefono a cui reindirizzare. Prefisso_OUT{FK} : <i>VARCHAR[5]</i> ⇔ Prefisso a cui reindirizzare.
Gruppo	Descrive i gruppi dei quali il contatto fa parte	Group_ID{PK} : <i>SERIAL</i> ⇔ Identifica univocamente il gruppo. Nome_G : <i>VARCHAR[50]</i> ⇔ Nome del gruppo. Descrizione : <i>VARCHAR[100]</i> ⇔ Descrizione del gruppo.
Email	Descrive gli indirizzi Email del contatto	Indirizzo{PK} : <i>VARCHAR[319]</i> ⇔ Indirizzo email del contatto. Utilizzo : <i>VARCHAR[60]</i> ⇔ Utilizzo dell'indirizzo email.
MessagingPR	Descrive il provider di messaggistica del contatto	PR_ID{PK} : <i>SERIAL</i> ⇔ Identifica univocamente il provider di messaggistica. Nickname : <i>VARCHAR[50]</i> ⇔ Nickname del contatto sul provider di messaggistica. Frase_Benvenuto : <i>VARCHAR[100]</i> ⇔ Frase di benvenuto del contatto sul provider di messaggistica. Fornitore : <i>VARCHAR[50]</i> ⇔ Fornitore di messaggistica.

2.3 Dizionario delle associazioni

La seguente tabella descrive le associazioni, per gruppo di tabelle coinvolte, e il loro scopo.

Associazione	Tabelle coinvolte
Alloggio	Indirizzo [1..*] ruolo(<i>Ospita</i>): Indica gli indirizzi fisici che ospitano ai contatti. Contatto [1..*] ruolo(<i>Alloggia</i>): Indica i contatti che alloggiano ad indirizzi fisici .
Recapito	Telefono [2..*] ruolo(<i>Posseduto</i>): Indica i telefoni che sono posseduti da un contatto. Contatto [1..*] ruolo(<i>Possiede</i>): Indica i contatti che posseggono i telefoni.
Assegnazione Email	Email [0..*] ruolo(<i>Posseduto</i>): Indica gli indirizzi email che sono posseduti da un contatto. Contatto [1] ruolo(<i>Possiede</i>): Indica il contatto che possiede un indirizzo email.
Aggregazione	Gruppo [0..*] ruolo(<i>Composto</i>): Indica i gruppi che sono composti da contatti. Contatto [1..*] ruolo(<i>Appartiene</i>): Indica i contatti che appartengono ai gruppi.
Assegnazione MessagingPr	Email [1] ruolo(<i>Assegnata</i>): Indica l'indirizzo email che é assegnato ad un provider di messaggistica. MessagingPr [0..*] ruolo(<i>Richiede</i>): Indica il provider di messaggistica che richiedono un indirizzo email.

2.4 Dizionario dei vincoli

Vincoli	Tipologia	Descrizione
Controllo formato Email	N-upla	Controlla che il formato dell'indirizzo email sia corretto.
Controllo Telefono Reindirizzato	Interrelazionale	Controlla che il telefono reindirizzato da un fisso sia mobile e nel caso da mobile sia fisso.
Controllo eliminazione Telefono	Interrelazionale	Automatismo Controlla se nessun' altro utente sta utilizzando il numero da eliminare e procede quindi all'eliminazione del numero dal database. Inoltre controlla che l'utente al quale si sta tentando di cancellare questo numero non lo stia utilizzando come numero di reindirizzamento da fisso a mobile
Controllo ordine inserimento recapiti	Interrelazionale	Controlla che i primi due recapiti vengano inseriti nell'ordine corretto.
Controllo eliminazione recapiti posseduti	Interrelazionale	Impedisce l'eliminazione di recapiti nel caso in cui ne rimangano solo due.

Capitolo 3

Progettazione logica

Lo schema logico viene rappresentato indicando le chiavi primarie in grassetto e le chiavi esterne sottolineate.

3.1 Schema Logico

Indirizzo	$\left(\textbf{Addr_ID}, \text{Via}, \text{Città}, \text{CAP}, \text{Nazione} \right)$
Contatto	$\left(\textbf{Cont_ID}, \text{Nome}, \text{Cognome}, \text{Ind_Foto}, \underline{\text{Indirizzo_P}} \right)$ $\text{Indirizzo_P} \rightarrow \text{Indirizzo.Addr_ID}$
Alloggio	$\left(\underline{\text{Cont_ID}}, \underline{\text{Addr_ID}} \right)$ $\text{Cont_ID} \rightarrow \text{Contatto.Cont_ID}$ $\text{Addr_ID} \rightarrow \text{Indirizzo.Addr_ID}$
Telefono	$\left(\textbf{Numero}, \textbf{Prefisso}, \text{Tipo} \right)$
Recapito	$\left(\textbf{Rec_ID}, \text{Numero}, \text{Prefisso}, \underline{\text{Numero_OUT}}, \underline{\text{Prefisso_OUT}}, \underline{\text{Cont_ID}} \right)$ $\text{Numero} \rightarrow \text{Telefono.Numero}$ $\text{Prefisso} \rightarrow \text{Telefono.Prefisso}$ $\text{Numero_OUT} \rightarrow \text{Telefono.Numero}$ $\text{Prefisso_OUT} \rightarrow \text{Telefono.Prefisso}$ $\text{Cont_ID} \rightarrow \text{Contatto.Cont_ID}$
Gruppo	$\left(\textbf{Group_ID}, \text{Nome_G}, \text{Descrizione} \right)$
Aggregazione	$\left(\underline{\text{Group_ID}}, \underline{\text{Cont_ID}} \right)$ $\text{Group_ID} \rightarrow \text{Gruppo.Group_ID}$ $\text{Cont_ID} \rightarrow \text{Contatto.Cont_ID}$
Email	$\left(\textbf{Indirizzo}, \text{Utilizzo}, \underline{\text{Cont_ID}} \right)$ $\text{Cont_ID} \rightarrow \text{Contatto.Cont_ID}$
MessagingPR	$\left(\textbf{PR_ID}, \text{Nickname}, \text{Frase_Benvenuto}, \text{Fornitore}, \underline{\text{Indirizzo}} \right)$ $\text{Indirizzo} \rightarrow \text{Email.Indirizzo}$

Capitolo 4

Progettazione fisica SQL

4.1 Definizione tabelle

4.1.1 Creazione schema

```
CREATE DATABASE ProgettoBasi;  
  
CREATE SCHEMA ContattiPiu AUTHORIZATION uvmzphxd;
```

4.1.2 Indirizzo

```
CREATE TABLE Indirizzo(  
  
    Addr_ID SERIAL NOT NULL,  
  
    Via VARCHAR(60) NOT NULL,  
  
    Città VARCHAR(50) NOT NULL,  
  
    CAP VARCHAR(12) NOT NULL,  
  
    Nazione VARCHAR(50) NOT NULL,  
  
PRIMARY KEY (Addr_ID),  
  
UNIQUE (Via,Città,CAP,Nazione)  
  
);
```

4.1.3 Contatto

```
CREATE TABLE Contatto(  
  
    Cont_ID SERIAL NOT NULL,  
  
    Nome VARCHAR(50) NOT NULL,  
  
    Cognome VARCHAR(50) NOT NULL,  
  
    Ind_Foto VARCHAR(260),  
  
    Indirizzo_P INTEGER NOT NULL,  
  
FOREIGN KEY (Indirizzo_P) REFERENCES Indirizzo (Addr_ID)  
ON DELETE SET NULL ON UPDATE CASCADE,  
  
PRIMARY KEY (Cont_ID)  
  
);
```

4.1.4 Alloggio

```
CREATE TABLE Alloggio(  
    Cont_ID INTEGER NOT NULL,  
    Addr_ID INTEGER NOT NULL,  
  
    FOREIGN KEY (Cont_ID) REFERENCES Contatto (Cont_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
  
    FOREIGN KEY (Addr_ID) REFERENCES Indirizzo (Addr_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
  
    PRIMARY KEY (Cont_ID, Addr_ID)  
);
```

4.1.5 Telefono

```
CREATE TYPE T_TYPE AS ENUM ('Fisso', 'Mobile');  
  
CREATE TABLE Telefono(  
    Numero VARCHAR(11) NOT NULL,  
    Prefisso VARCHAR(5) NOT NULL,  
    Tipo T_TYPE NOT NULL,  
  
    PRIMARY KEY (Numero, Prefisso)  
);
```

4.1.6 Recapito

```
CREATE TABLE Recapito(  
    Rec_ID SERIAL NOT NULL,  
    Cont_ID INTEGER NOT NULL,  
    Numero VARCHAR(11) NOT NULL,  
    Prefisso VARCHAR(5) NOT NULL,  
    Numero_OUT VARCHAR(11),  
    Prefisso_OUT VARCHAR(5),  
  
    FOREIGN KEY (Cont_ID) REFERENCES Contatto (Cont_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
  
    FOREIGN KEY (Numero, Prefisso) REFERENCES Telefono (Numero, Prefisso),  
  
    FOREIGN KEY (Numero_OUT, Prefisso_OUT) REFERENCES Telefono (Numero, Prefisso),  
  
    PRIMARY KEY (Rec_ID),  
  
    UNIQUE (Cont_ID, Numero, Prefisso)  
);
```

4.1.7 Gruppo

```
CREATE TABLE Gruppo(  
    Group_ID SERIAL NOT NULL,  
    Nome_G VARCHAR(50) NOT NULL,  
    Descrizione VARCHAR(100),  
    PRIMARY KEY (Group_ID)  
);
```

4.1.8 Aggregazione

```
CREATE TABLE Aggregazione(  
    Group_ID INTEGER NOT NULL,  
    Cont_ID INTEGER NOT NULL,  
    FOREIGN KEY (Group_ID) REFERENCES Gruppo (Group_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    FOREIGN KEY (Cont_ID) REFERENCES Contatto (Cont_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    PRIMARY KEY (Group_ID, Cont_ID)  
);
```

4.1.9 Email

```
CREATE TABLE Email(  
    Indirizzo VARCHAR(319) NOT NULL,  
    Utilizzo VARCHAR(60),  
    Cont_ID SERIAL NOT NULL,  
    FOREIGN KEY (Cont_ID) REFERENCES Contatto (Cont_ID)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    PRIMARY KEY (Indirizzo)  
);
```

4.1.10 MessagingPR

```
CREATE TABLE MessagingPR(  
    PR_ID SERIAL NOT NULL,  
    Nickname VARCHAR(50) NOT NULL,  
    Frase_Benvenuto VARCHAR(100),  
    Fornitore VARCHAR(50),  
    Indirizzo VARCHAR(319) NOT NULL,  
    FOREIGN KEY (Indirizzo) REFERENCES Email (Indirizzo)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    PRIMARY KEY (PR_ID),  
    UNIQUE (PR_ID, Indirizzo)  
);
```

4.2 Vincoli di Integrità referenziale

4.2.1 Integrità referenziale Contatto

```
ALTER TABLE Contatto

ADD CONSTRAINT FK_Contatto_Indirizzo FOREIGN KEY (Indirizzo_P)
REFERENCES Indirizzo(Addr_ID)
ON DELETE SET NULL ON UPDATE CASCADE;
```

4.2.2 Integrità referenziale Alloggio

```
ALTER TABLE Alloggio

ADD CONSTRAINT FK_Alloggio_Indirizzo FOREIGN KEY (Addr_ID)
REFERENCES Indirizzo(Addr_ID)
ON DELETE CASCADE ON UPDATE CASCADE;

ADD CONSTRAINT FK_Alloggio_Contatto FOREIGN KEY (Cont_ID)
REFERENCES Contatto(Cont_ID)
ON DELETE CASCADE ON UPDATE CASCADE;
```

4.2.3 Integrità referenziale Aggregazione

```
ALTER TABLE Aggregazione

ADD CONSTRAINT FK_Aggregazione_Contatto FOREIGN KEY (Cont_ID)
REFERENCES Contatto(Cont_ID)
ON DELETE CASCADE ON UPDATE CASCADE;

ADD CONSTRAINT FK_Aggregazione_Gruppo FOREIGN KEY (Group_ID)
REFERENCES Gruppo(Group_ID)
ON DELETE CASCADE ON UPDATE CASCADE;
```

4.2.4 Integrità referenziale Recapito

```
ALTER TABLE Recapito

ADD CONSTRAINT FK_Recapito_Contatto FOREIGN KEY (Cont_ID)
REFERENCES Contatto(Cont_ID)
ON DELETE CASCADE ON UPDATE CASCADE;
```

4.2.5 Integrità referenziale Email

```
ALTER TABLE Email

ADD CONSTRAINT FK_Email_Contatto FOREIGN KEY (Cont_ID)
REFERENCES Contatto (Cont_ID)
ON DELETE CASCADE ON UPDATE CASCADE;
```

4.2.6 Integrità referenziale MessagingPR

```
ALTER TABLE MessagingPR

ADD CONSTRAINT FOREIGN KEY FK_MessagingPR_Email (Indirizzo)
REFERENCES Email (Indirizzo)
ON DELETE CASCADE ON UPDATE CASCADE;
```

4.3 Implementazione vincoli di N-Upla

4.3.1 Controlla Formato Email

```
ALTER TABLE Email  
  
ADD CONSTRAINT Controlla_Formato_Email CHECK (Indirizzo LIKE '%@%.%');
```

4.4 Implementazione vincoli Interrelazionali

4.4.1 Controlla Telefono Reindirizzato

```
/* Il trigger controlla che il tipo di numero reindirizzato sia sempre esatto sia  
   in modifica che inserimento*/  
  
CREATE OR REPLACE FUNCTION controllo_telefono_reindirizzato() RETURNS TRIGGER AS  
    $controllo_telefono_reindirizzato$  
  
DECLARE  
  
Tipo_in Telefono.tipo%TYPE;  
  
Tipo_out Telefono.tipo%TYPE;  
  
BEGIN  
  
SELECT tel.tipo INTO Tipo_in  
  
FROM Telefono AS tel  
  
WHERE NEW.numero = tel.numero AND NEW.prefisso = tel.prefisso;  
  
SELECT tel.tipo INTO Tipo_out  
  
FROM Telefono AS tel  
  
WHERE NEW.Numero_OUT = tel.Numero AND NEW.Prefisso_OUT = tel.Prefisso;  
  
IF(Tipo_in = Tipo_out) THEN  
  
RAISE EXCEPTION 'Errore : i tipi non devono essere uguali';  
  
ELSEIF((Tipo_in = 'Fisso') AND (Tipo_out IS NULL)) THEN  
  
RAISE EXCEPTION 'Errore : un numero fisso deve avere necessariamente un mobile  
    di reindirizzamento';  
  
END IF;  
  
RETURN NULL;  
  
END;  
  
$controllo_telefono_reindirizzato$ LANGUAGE plpgsql;  
  
CREATE OR REPLACE TRIGGER CONTROLLO_TELEFONO_REINDIRIZZATO AFTER INSERT OR UPDATE  
    ON RECAPITO  
  
FOR EACH ROW EXECUTE FUNCTION controllo_telefono_reindirizzato();
```

4.4.2 Controlla Eliminazione Telefono

```
/* Il seguente trigger controlla se nessun'altro utente sta utilizzando il numero
   da eliminare e procedere quindi all'eliminazione del numero dal database (
   Automatismo)
Inoltre controlla che l'utente al quale si sta tentando di cancellare questo
   numero non lo stia utilizzando come numero di reindirizzamento */

CREATE OR REPLACE FUNCTION controllo_delete_telefono() RETURNS TRIGGER AS
    $controllo_delete_telefono$

BEGIN

IF(EXISTS (SELECT * FROM RECAPITO AS RE WHERE RE.NUMERO_OUT = OLD.NUMERO AND RE.
    PREFISSO_OUT=OLD.PREFISSO AND RE.CONT_ID=OLD.CONT_ID )) THEN

RAISE EXCEPTION 'Errore : Non puoi eliminare un numero che stai utilizzando come
    reindirizzamento';

ELSEIF (NOT EXISTS (SELECT * FROM RECAPITO AS RE WHERE RE.PREFISSO=OLD.PREFISSO
    AND RE.NUMERO = OLD.NUMERO AND RE.CONT_ID<>OLD.CONT_ID )) THEN

DELETE FROM TELEFONO WHERE TELEFONO.NUMERO=OLD.NUMERO AND TELEFONO.PREFISSO=OLD.
    PREFISSO;

END IF;

RETURN NULL;
END;

$controllo_delete_telefono$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER CONTROLLO_DELETE_TELEFONO AFTER DELETE ON RECAPITO
FOR EACH ROW EXECUTE FUNCTION controllo_delete_telefono();
```

4.4.3 Controllo Ordine Inserimento recapiti

```
/*Il trigger controlla che l'inserimento dei recapiti avvenga nell'ordine
   corretto: prima Fisso, poi Mobile. In oltre controlla che venga mantenuto l'
   ordine anche su aggiornamento*/

CREATE OR REPLACE FUNCTION controllo_ordine_inserimento_recapiti() RETURNS
    TRIGGER AS $controllo_ordine_inserimento_recapiti$

DECLARE

Tipo_in Telefono.tipo%TYPE;
cont INTEGER := 0;

BEGIN

SELECT tel.tipo INTO Tipo_in
FROM Telefono AS tel
WHERE NEW.numero = tel.numero AND NEW.prefisso = tel.prefisso;

SELECT COUNT (Numero) INTO cont
FROM Recapito
WHERE Cont_ID = new.Cont_ID;

IF(Cont = 0 AND Tipo_in = 'Mobile') THEN

RAISE EXCEPTION 'Inserire prima un Fisso';

END IF;

IF (Cont = 1 AND Tipo_in = 'Fisso') THEN

RAISE EXCEPTION 'Inserire un Mobile';

END IF;

RETURN NEW;

END;

$controllo_ordine_inserimento_recapiti$

LANGUAGE plpgsql;

CREATE TRIGGER CONTROLLO_ORDINE_INSERTIMENTO_RECAPITI BEFORE INSERT OR UPDATE ON
    RECAPITO

FOR EACH ROW EXECUTE FUNCTION controllo_ordine_inserimento_recapiti();
```

4.4.4 Controllo eliminazione recapiti posseduti

```
/*Il trigger impedisce l'eliminazione di recapiti nel caso in cui ne restino al
pi due. Prima viene fatto un controllo sulla chiave esterna riferita a
Contatto per assicurarsi non si tratti di una DELETE CASCADE*/

CREATE OR REPLACE FUNCTION controllo_eliminazione_recapiti_posseduti() RETURNS
    TRIGGER AS $controllo_eliminazione_recapiti_posseduti$

DECLARE

cont INTEGER := 0;

BEGIN

IF EXISTS(

SELECT *
FROM Contatto AS Cont
WHERE Cont.Cont_ID = old.Cont_ID

) THEN

SELECT COUNT (Numero) INTO cont
FROM Recapito
WHERE Cont_ID = old.Cont_ID;

IF(Cont <= 2) THEN
RAISE EXCEPTION 'Il contatto possiede solo due o meno recapiti';
END IF;

END IF;

RETURN OLD;

END;

$controllo_eliminazione_recapiti_posseduti$

LANGUAGE plpgsql;

CREATE TRIGGER CONTROLLO_ELIMINAZIONE_RECAPITI_POSSEDUTI BEFORE DELETE ON
    RECAPITO

FOR EACH ROW EXECUTE FUNCTION controllo_eliminazione_recapiti_posseduti();
```