Music Shop

Gruppo 10 **Mat:108801**

Sedoni Enrico

Indice

Descrizione della realtà da analizzare

Il database serve per gestire una catena nota di negozi nazionale che si occupano della vendita e del noleggio di cd musicali nelle varie città di interesse.

In particolare interessa gestire una catena di filiali che sono in grado di rifornirsi a vicenda. Ogni filiale espone dei cd musicali.

Delle filiali si memorizzano: il codice (univoco), il numero di telefono e l'indirizzo (composto da: via, cap, città, num civico e provincia).

Le filiali devono avere la possibilità di rifornirsi a vicenda e di rifornirsi con il magazzino di riferimento. Occorre anche tener presente che se il magazzino è situato nella stessa provincia della filiale e si avvia una procedura di rifornimento tra filiali, allora la filiale deve dare precedenza al magazzino e si deve rifornire da quest'ultimo.

Dei magazzini si memorizzano: il codice (univoco) e l'indirizzo (composto da: via, cap, città, num civico e provincia).

Ogni filiale è rifornita da un solo magazzino, mentre un magazzino rifornisce più filiali.

Ogni filiale possiede del personale: dirigenti e dipendenti. Un dirigente dirige una ed una sola filiale e una filiale è diretta da un solo dirigente. Un dipendente invece lavora in un unica filiale, mentre in una filiale lavorano più dipendenti. Dei dipendenti e dei dirigenti si memorizzano: costo orario (un dirigente costerà di più di un dipendente), nome, cognome, codice fiscale, telefono e i dati di residenza. Un dirigente inoltre è caratterizzato dall'anzianità, che indica il numero di anni in cui si è occupato di dirigere la filiale. Deve essere quindi possibile calcolare il costo totale del personale per ogni filiale.

Una filiale, oltre a vendere e noleggiare cd musicali, può anche sponsorizzare eventi musicali per aumentare la propria notorietà e quella della catena. In particolare una filiale sponsorizza molti eventi e un evento è sponsorizzato da una sola filiale.

Un evento è caratterizzato da: località, data e nome.

Dei cd musicali interessa memorizzare: il nome, la durata, il numero dei brani e il prezzo cad. Un negozio espone quindi dei cd musicali ai clienti. Interessa gestire anche il numero di copie disponibili alla vendita per le varie filiali.

Viene gestito anche il genere musicale dei cd, del genere si memorizza: il nome (univoco) e l'anno di popolarità in cui era in voga il genere musicale.

Un cd musicale può aderire a molti generi musicali e ad un genere musicale possono appartenere molti cd musicali.

Interessano gestire anche le band che hanno scritto i cd, delle quali si memorizzano nome e data di fondazione.

Per band e cd musicali occorre gestire il fatto che i nomi di entrambi non sono univoci.

E' necessario inoltre gestire le case discografiche, in particolare una band può far parte di una casa discografica e una casa discografica può seguire molti artisti differenti. La casa discografica è identificata da: nome (univoco) e anno di fondazione.

Infine per il cd musicale deve essere anche gestito lo studio di registrazione che si è occupato di produrre il cd. Dello studio di registrazione di memorizzano: nome e e l'indirizzo (composto da: via, cap, città, num civico e provincia). Un cd quindi è registrato da uno studio di registrazione e uno studio di registrazione registra molti cd, occorre tener memorizzato anche il costo della registrazione del cd.

E' inoltre necessario gestire i clienti che acquistano i cd musicali con i propri rispettivi carrelli di acquisto. Dei clienti si memorizza: codice fiscale, nome, cognome e telefono. Il cliente acquista un carrello in una certa data, il quale contiene dei cd musicali. Il carrello è identificato da un numero (unico se associato a un determinato cliente) e un importo totale.

Inoltre la nota catena di negozi mette a disposizione dei buoni sconto per i vari clienti. Il buono sconto è identificato da un codice (univoco), un attributo booleano che indica se è valido oppure no e il valore dello sconto in percentuale. Il carrello può essere quindi soggetto a un buono sconto, che una volta utilizzato perde di validità.

Un cliente deve poter anche prendere in prestito dei cd musicali per poterli ascoltare e decidere se acquistarli successivamente oppure no. In particolare interessa gestire il vincolo che un cliente può prendere in prestito un determinato cd una ed una sola volta, dopodichè, se lo vuole, è costretto ad acquistarlo la volta sucessiva.

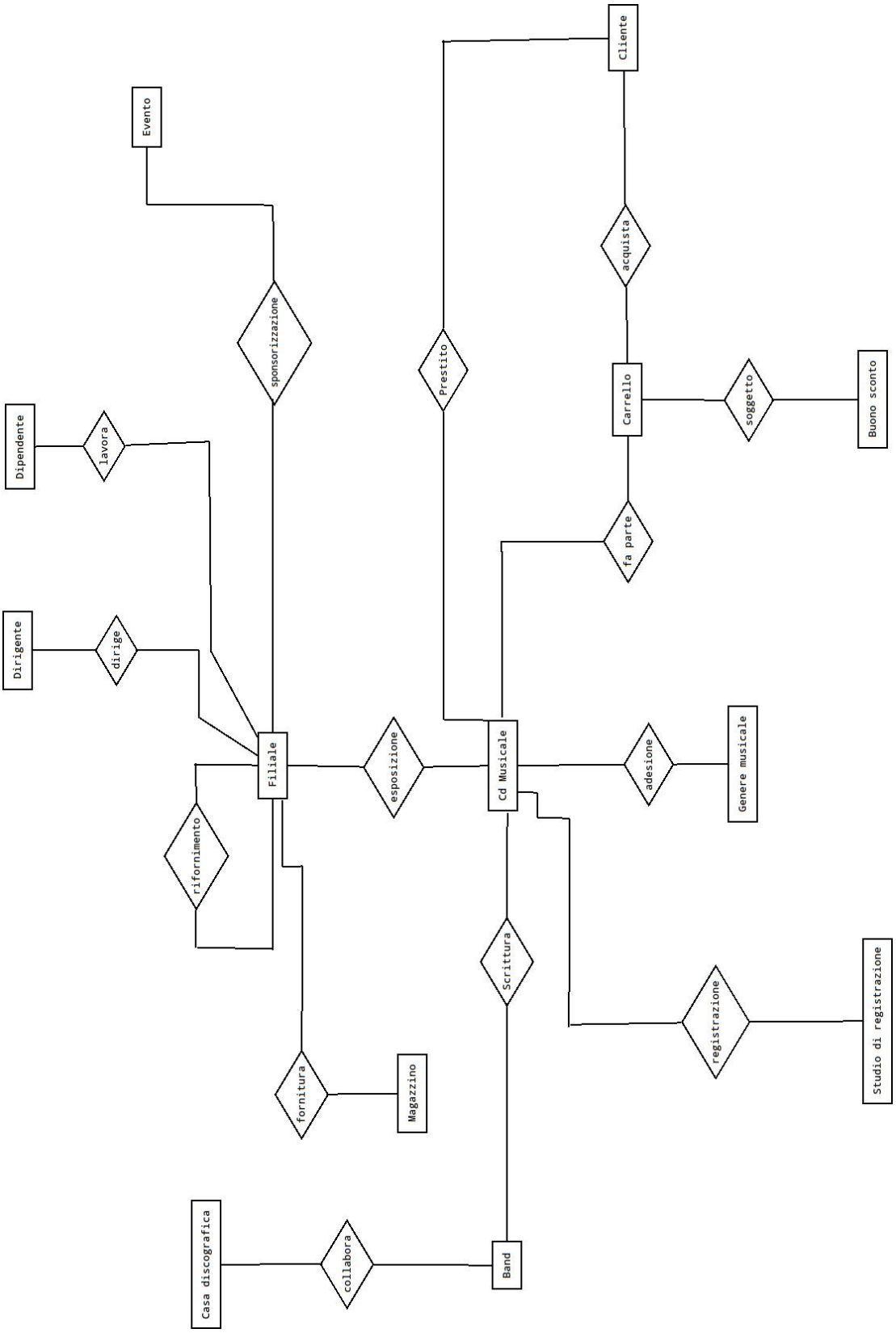
Del prestito si memorizzano anche: costo e data.

Glossario

Termine	Descrizione	Sinonimi	Legami
Casa discografica	- Nome - Anno fondazione		Band
Band	- Codice - Nome - Data di fondazione	Artista	Casa discografica, cd musicale
Studio di registrazione	- Nome - Via - Citta - CAP - Num civico - Provincia		Cd musicale
Cd musicale	- Nome - Num brani - Durata - Prezzo cad.		Band, studio di registrazione, genere musicale, filiale, carrello, cliente
Genere musicale	- Nome - Popolarità(anno)		Cd musicale
Magazzino	- Codice - Via - Citta - CAP - Num civico - Provincia		Fialiale
Filiale	- Codice - Telefono - Via - Citta - CAP	negozio	Magazzino, cd musicale, dirigente, dipendente, evento

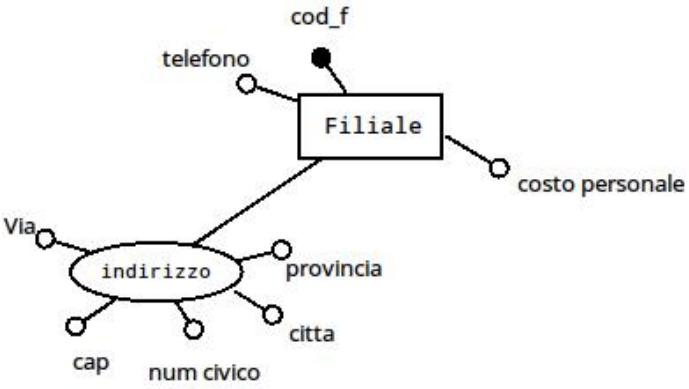
	- Num civico - Provincia		
Dirigente	- Codice fiscale - Nome - Cognome - Telefono - Via - Citta - CAP - Num civico - Provincia - Costo orario - Anzianità		Filiale
Dipendente	- Codice fiscale - Nome - Cognome - Telefono - Via - Citta - CAP - Num civico - Provincia - Costo orario		Filiale
Evento	- Nome - Data - Località		FIliale
Cliente	- Codice fiscale - Nome - Cognome - Telefono		Cd musicale
Buono sconto	- Codice - Sconto		Carrello
Carrello	- Numero		Cliente, buono sconto

Schema Scheletro



Progettazione Concettuale

“[..]Delle filiali si memorizzano: il codice (univoco), il numero di telefono e l’indirizzo (composto da: via, cap, città, num civico e provincia)[...]”

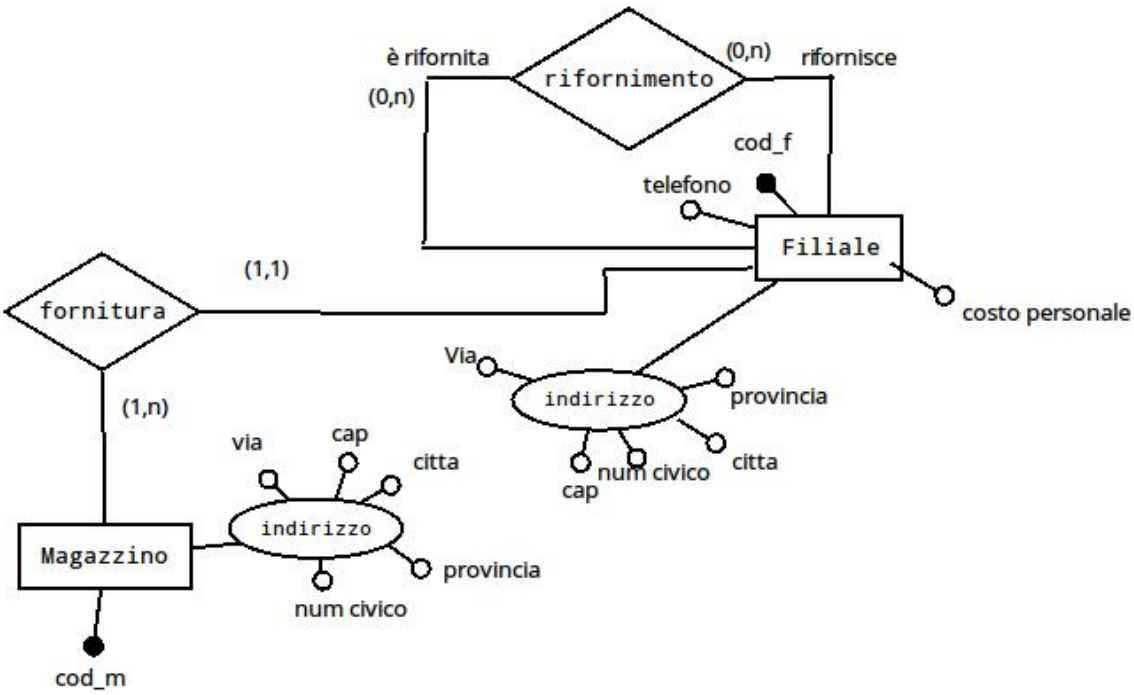


L’indirizzo è stato rappresentato come attributo composto per comodità. Verrà poi gestito in una fase sucessiva. Si è deciso anche di inserire oltre agli attributi indicati dal testo il dato derivato “costo personale”. Siccome è una grossa multinazionale, le filiali potrebbero essere in gran numero, quindi il dato derivato potrebbe risultare conveniente. In una fase sucessiva progettuale ne verrà studiata l’effettiva convenienza.

“[…]Le filiali devono avere la possibilità di rifornirsi a vicenda e di rifornirsi con il magazzino di riferimento. Occorre anche tener presente che se il magazzino è situato nella stessa provincia della filiale e si avvia una procedura di rifornimento tra filiali, allora la filiale deve dare precedenza al magazzino e si deve rifornire da quest’ultimo.

Dei magazzini si memorizzano: il codice (univoco) e l’indirizzo (composto da: via, cap, città, num civico e provincia).

Ogni filiale è rifornita da un solo magazzino, mentre un magazzino rifornisce più filiali. […]”



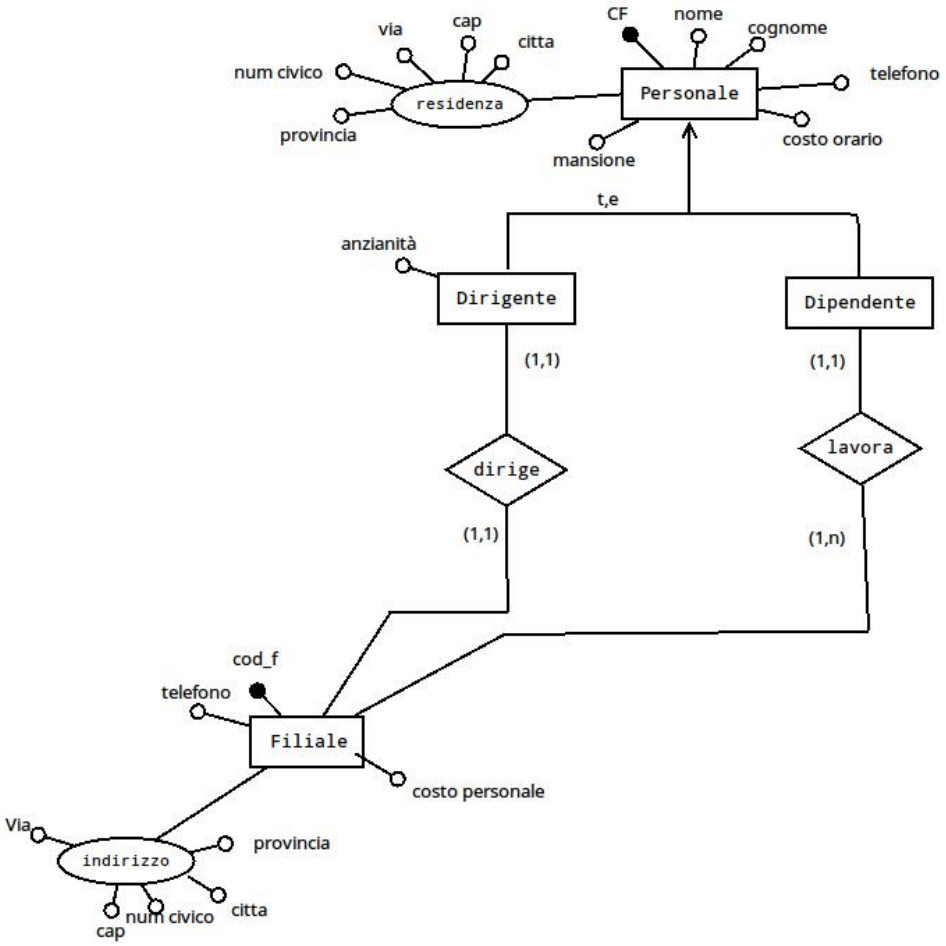
Si è deciso di rappresentare il rifornimento tra le filiali come un auto-associazione N:M. Una filiale può rifornire molte filiali e può essere rifornita da molte filiali. Ciò però non vieta che una filiale possa rifornirsi da sola, questo vincolo verrà gestito poi in una fase sucessiva di progettazione (trigger).

La relazione tra la filiale e il magazzino invece è una semplice 1:N. Un magazzino rifornisce molte filiali, mentre un filiale è rifornita da uno ed un solo magazzino.

Per l’entità magazzino si è addottata la stessa strategia dell’entità filiale per rappresentare l’indirizzo.

La gestione delle procedure di rifornimento tra filiali e tra filiale-magazzino avverrà in una fase sucessiva di proggetto, siccome è un vincolo non imponibile in questa fase.

“[...]Ogni filiale possiede del personale: dirigenti e dipendenti. Un dirigente dirige una ed una sola filiale e una filiale è diretta da un solo dirigente. Un dipendente invece lavora in un unica filiale, mentre in una filiale lavorano più dipendenti. Dei dipendenti e dei dirigenti si memorizzano: costo orario (un dirigente costerà di più di un dipendente), nome, cognome, codice fiscale, telefono e i dati di residenza. Un dirigente inoltre è caratterizzato dall’anzianità, che indica il numero di anni in cui si è occupato di dirigere la filiale. Deve essere quindi possibile calcolare il costo totale del personale per ogni filiale.[...]”



Si è deciso di gestire il personale con una gerarchia siccome dipendenti e dirigenti possiedono la maggior parte degli attributi in comune (tranne l’anzianità che appartiene solamente ai dirigenti). In particolare si è utilizzata una gerarchia di tipo totale-esclusiva:

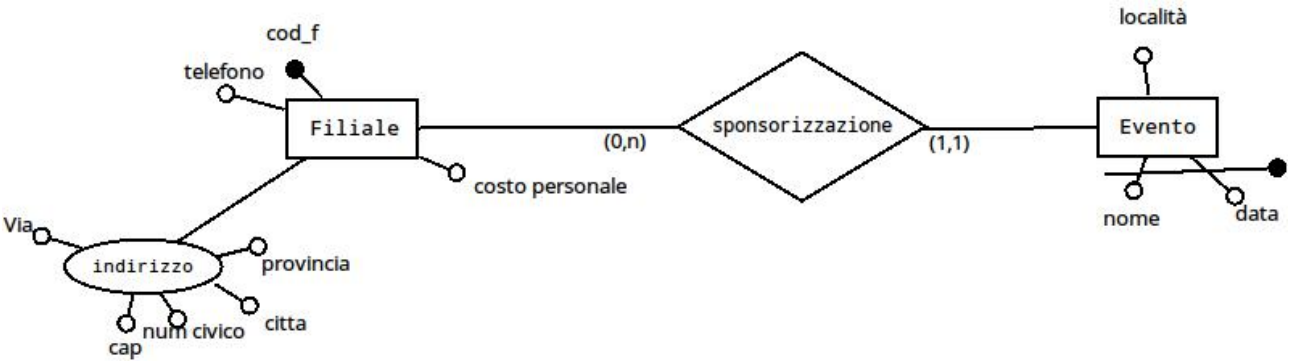
- Totale, tutto il personale è suddiviso in dipendenti e dirigenti, non interessano gestire altre categorie di personale
- Esclusiva, se si è dipendenti non si può essere dirigenti e viceversa.

La filiale quindi è in relazione 1:1 con il dirigente siccome una filiale è diretta da un solo dirigente e un dirigente dirige una ed una sola filiale alla volta. Per quanto

riguarda la cardinalità tra filiale e dipendente è 1:N siccome un dipendente lavora per una sola filiale e in una filiale lavorano molti dipendenti.

“[...]Una filiale, oltre a vendere e noleggiare cd musicali, può anche sponsorizzare eventi musicali per aumentare la propria notorietà e quella della catena. In particolare una filiale sponsorizza molti eventi e un evento è sponsorizzato da una sola filiale.

Un evento è caratterizzato da: località, data e nome.[...]”



Si è scelta come chiave primaria dell’entità “evento” la coppia nome-data, si è supposto infatti che questi due insieme determinino univocamente l’evento. Per la maggior parte degli eventi moderni questa supposizione viene rispettata.

Per quanto riguarda la cardinalità dell’associazione è 1:N, ovvero una filiale può sponsorizzare molti eventi, mentre un evento è sponsorizzato da una sola filiale.

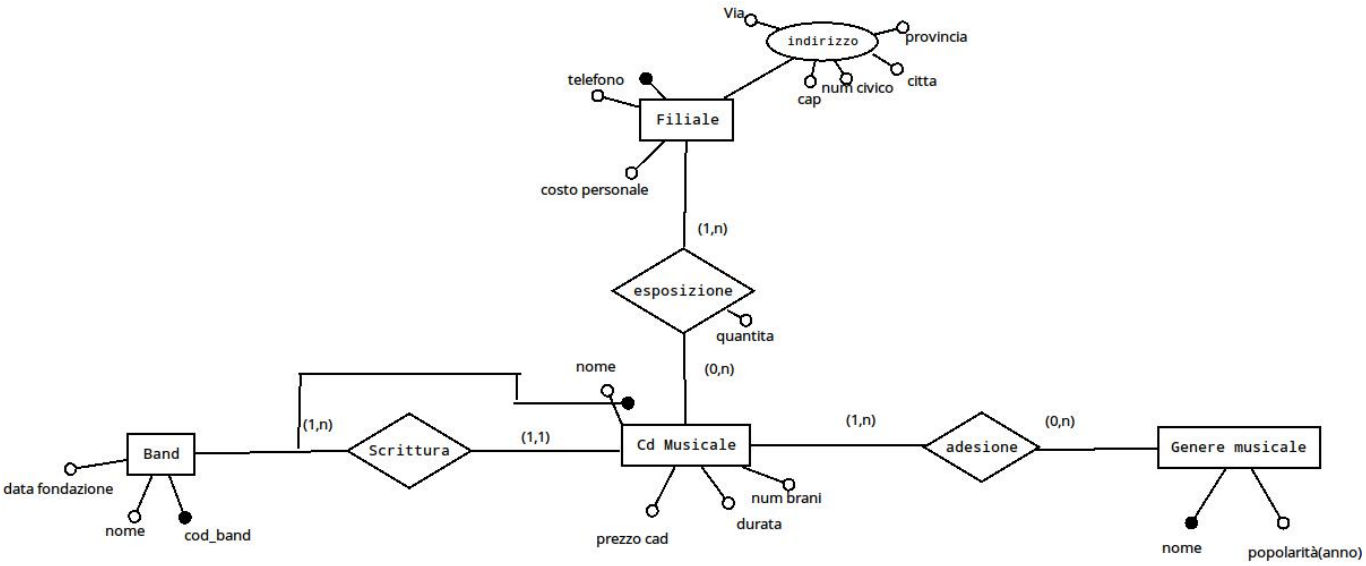
“[...]Dei cd musicali interessa memorizzare: il nome, la durata, il numero dei brani e il prezzo cad. Un negozio espone quindi dei cd musicali ai clienti. Interessa gestire anche il numero di copie disponibili alla vendita per le varie filiali.

Viene gestito anche il genere musicale dei cd, del genere si memorizza: il nome (univoco) e l’anno di popolarità in cui era in voga il genere musicale.

Un cd musicale può aderire a molti generi musicali e ad un genere musicale possono appartenere molti cd musicali.

Interessano gestire anche le band che hanno scritto i cd, delle quali si memorizzano nome e data di fondazione.

Per band e cd musicali occorre gestire il fatto che i nomi di entrambi non sono univoci.[...]”



Si è deciso di indentificare univocamente l’entità band attraverso un codice, in questo modo si è riusciti a identificare univocamente anche il cd musicale componendo il nome del cd con la chiave esterna “cod_band” dell’entità band, derivata dalla relazione che intercorre tra le due entità. Un altra soluzione possibile era quella di identificare l’entità “cd musicale” con un codice anch’esso, ma si è preferito tenere la chiave composta, decisamente più significativa di un codice.

Il cd è quindi in relazione 1:N con la band siccome una band scrive molti cd e un cd è scritto da una sola band. Per semplicità si è supposto che il cd sia scritto da una sola band, il database non prende in considerazione quei cd scritti da più artisti con particolari collaborazioni (caso raro, ma possibile nella realtà).

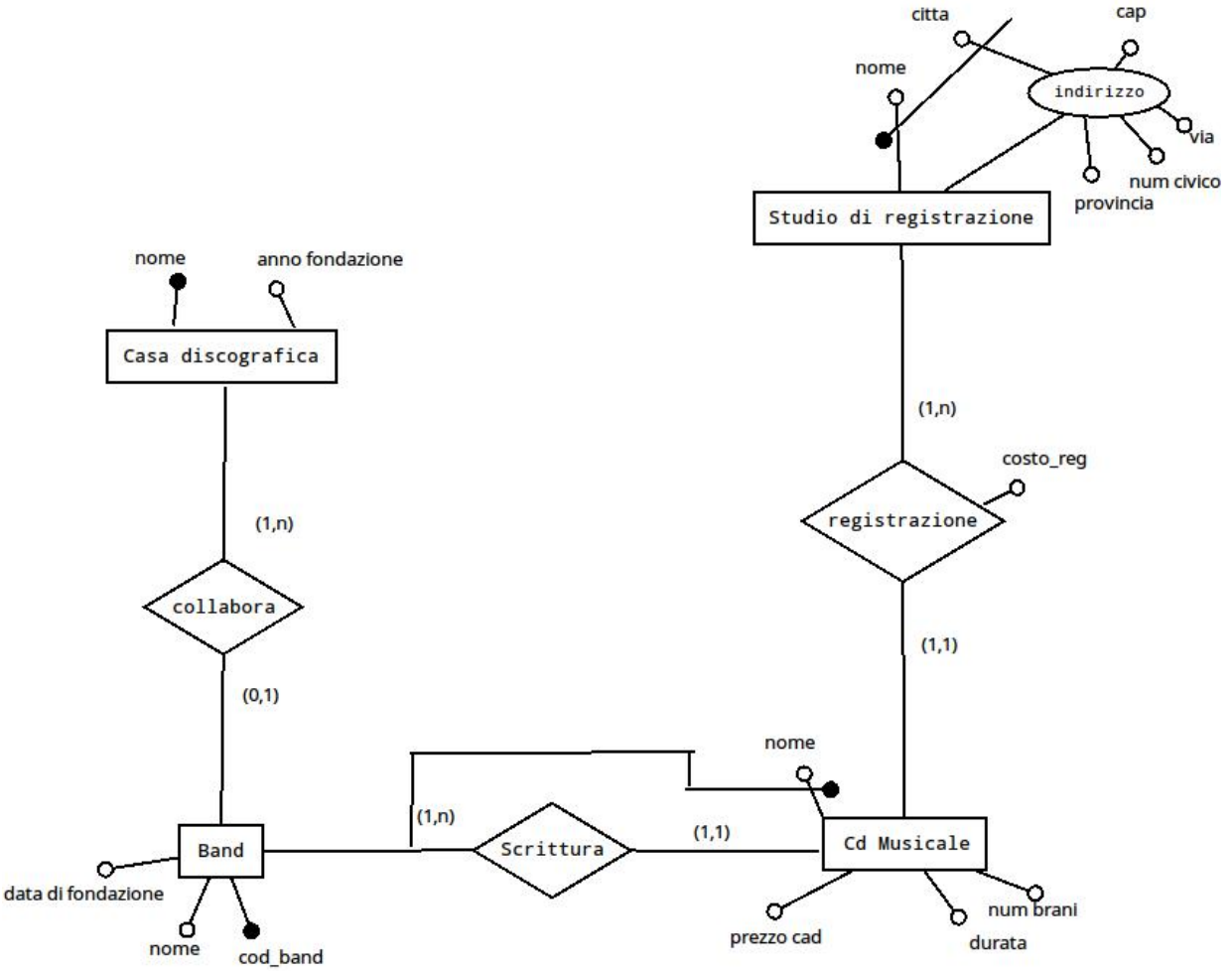
L’entità “cd musicale” indica il cd generale e non la copia del prodotto specifico messa in vendita in negozio, anche per questo motivo non si è deciso di utilizzare un codice, che avrebbe potuto far intendere il prodotto specifico.

La filiale è quindi in relazione N:M con il cd, infatti una filiale espone molti cd e un cd può essere esposto in più filiali alla volta. Nella relazione inoltre è stata salvata la quantita di cd che ha in esposizione la filiale, quando la quantità sarà bassa verrà avviata una procedura per il rifornimento del cd.

Infine il cd musicale è in relazione N:M con il genere musicale, siccome un cd può appartenere a molti generi contemporaneamente e ad un genere appartengono molti cd.

“[...]E’ necessario inoltre gestire le case discografiche, in particolare una band può far parte di una casa discografica e una casa discografica segue molti artisti differenti. La casa discografica è identificata da: nome (univoco) e anno di fondazione.

Infine per il cd musicale deve essere anche gestito lo studio di registrazione che si è occupato di produrre il cd. Dello studio di registrazione di memorizzano: nome e e l'indirizzo (composto da: via, cap, città, num civico e provincia). Un cd quindi è registrato da uno studio di registrazione e uno studio di registrazione registra molti cd, occorre tener memorizzato anche il costo della registrazione del cd.[...]"



Si è deciso di selezionare come chiave primaria dello studio di registrazione la coppia città-nome, si è quindi supposto che non esistano due studi con lo stesso nome localizzati nella stessa città.

Il cd è quindi in relazione 1:N con lo studio di registrazione poichè un cd è registrato in un solo studio di registrazione e uno studio di registrazione produce molti cd musicali. Nella relazione tra le due entità si è salvato il costo di registrazione del cd.

L'entità "band" è in relazione con la casa discografica 1:N, infatti una band può appartenere oppure no ad una casa discografica ed una casa discografica segue molte band differenti.

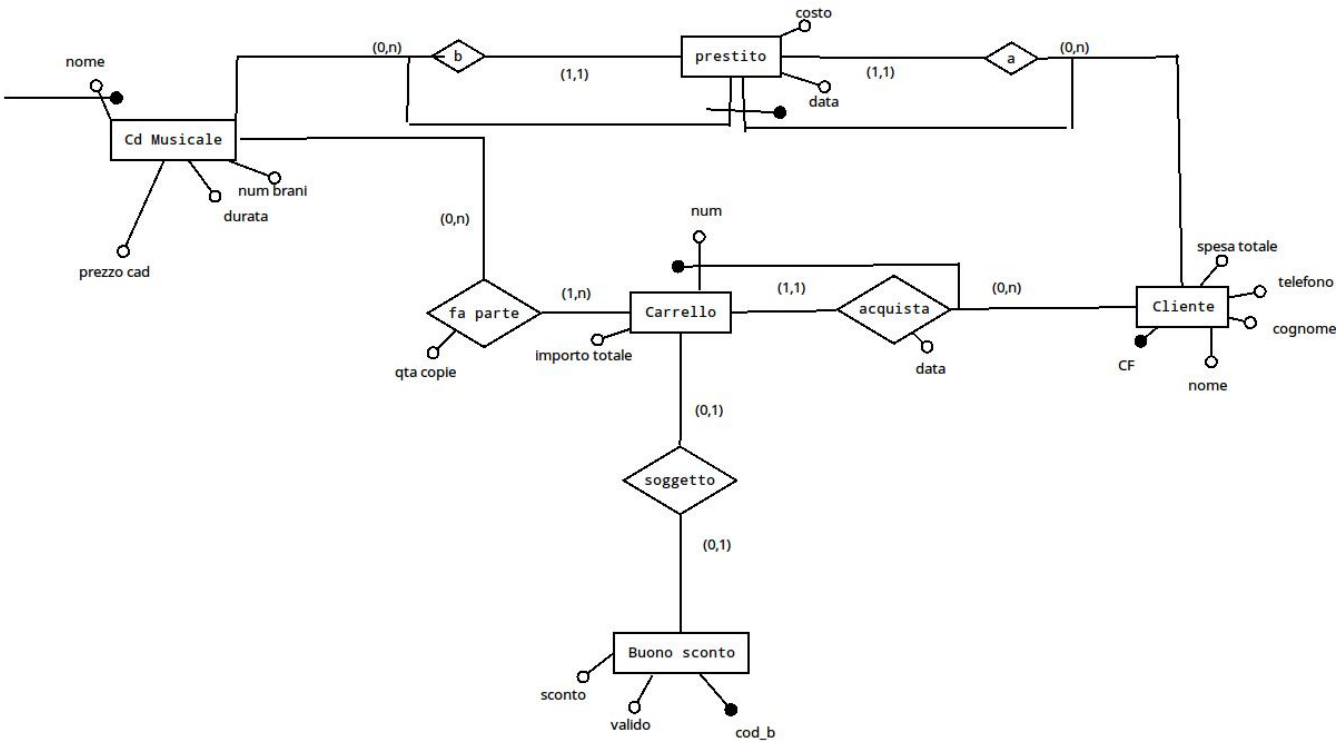
"[...]E' inoltre necessario gestire i clienti che acquistano i cd musicali con i propri rispettivi carrelli di acquisto. Dei clienti si memorizza: codice fiscale, nome, cognome e telefono. Il cliente acquista un carrello in una certa data, il quale contiene dei cd

musicali. Il carrello è identificato da un numero (unico se associato a un determinato cliente) e un importo totale.

Inoltre la nota catena di negozi mette a disposizione dei buoni sconto per i vari clienti. Il buono sconto è identificato da un codice (univoco) e il valore dello sconto(da). Il carrello può essere quindi soggetto a un buono sconto, che una volta utilizzato perde di validità.

Un cliente deve poter anche prendere in prestito dei cd musicali per poterli ascoltare e decidere se acquistarli successivamente oppure no. In particolare interessa gestire il vincolo che un cliente può prendere in prestito un determinato cd una ed una sola volta, dopodichè, se lo vuole, è costretto ad acquistarlo la volta sucessiva.

Del prestito si memorizzano anche: costo e data.“



Si è deciso di identificare il carrello univocamene con la coppia num-cf, dove il codice ficale (cf) deriva dal cliente, che è in relazione con il carrello. In questo modo uno stesso cliente può acquistare più carrelli, a patto che il carrello abbia un numero differente.

Un vincolo certamente particolare, ma gestibile abbastanza semplicemente, del database è quello del prestito, infatti come dice il testo “[...]un cliente può prendere in prestito un determinato cd una ed una sola volta, dopodichè, se lo vuole, è costretto ad acquistarlo la volta sucessiva”, si è deciso quindi di reificare la relazione N:M tra cliente e cd musicale. La chiave primaria del prestito sarà quindi la coppia cf-(nome,cod_band), ovvero la composizione della chiave del cd musicale e quella del cliente. In questo modo un cliente può prendere in prestito un determinato cd una ed una sola volta.

Del cliente inoltre si è deciso di tenere memorizzata anche la spesa totale, che è un dato derivato. La spesa totale è ricavata dalla somma di tutti gli importi dei carrelli

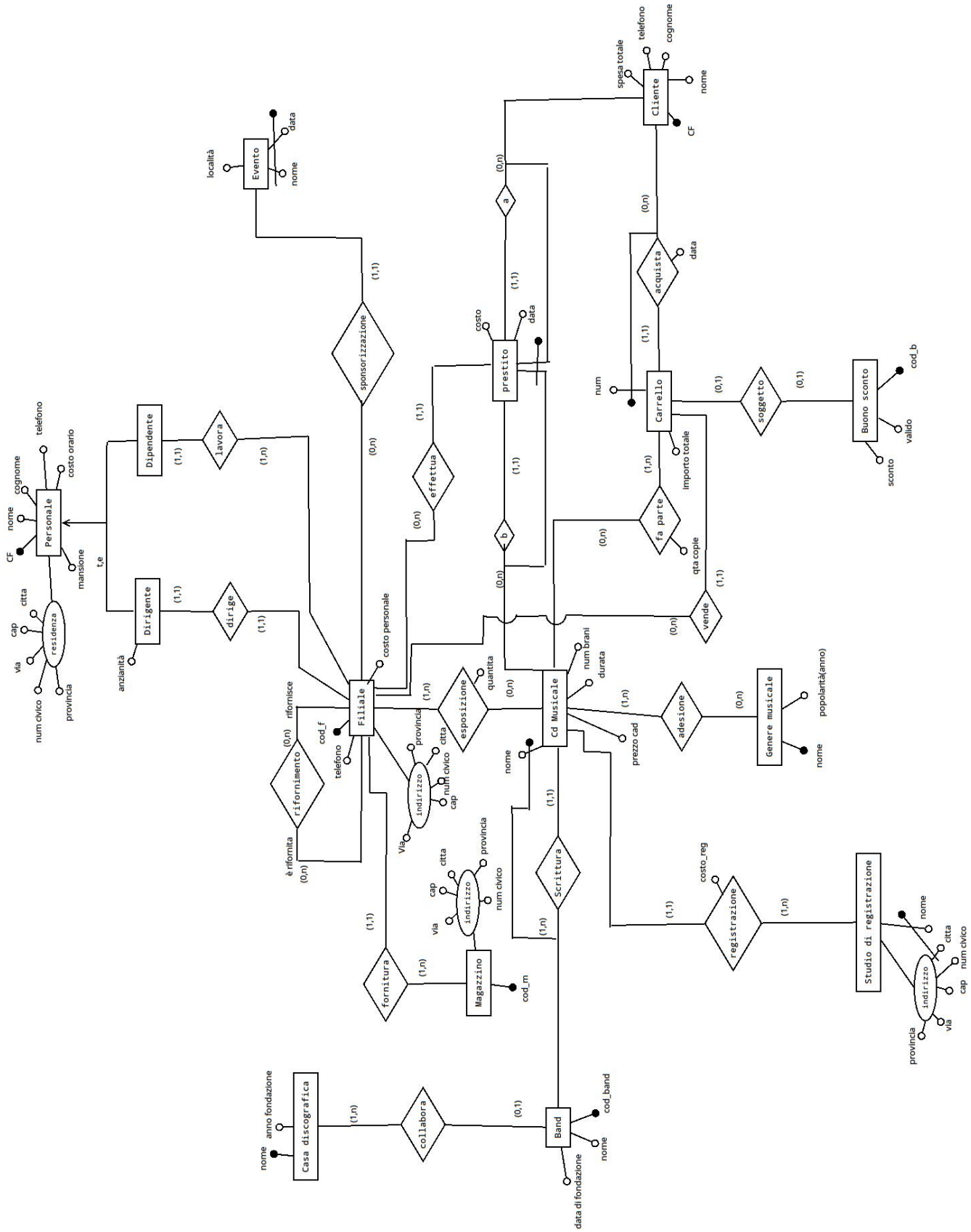
acquistati (senza considerare gli sconti su questi) più la somma di tutti i prestiti effettuati. Anche l’attributo “importo totale” del carrello è un dato derivato, che si ricava dalla somma di tutti i prezzi dei cd acquistati per la quantità di copie rispettive di ognuno di essi.

Per quanto riguarda le relazioni è interessante quella che intercorre tra il buono sconto ed il carrello.

Il carrello è in relazione 1:1 con il buono sconto, infatti un carrello può essere soggetto ad un buono sconto, mentre un buono sconto può essere utilizzato su un solo carrello. La catena mette a disposizione molti buoni sconto alla volta, quindi si potranno avere diversi buoni sconto non utilizzati sui carrelli. Inoltre un buono sconto una volta utilizzato perde la validità e non può essere più utilizzato, la relazione impone già questo vincolo, siccome si è utilizzata come chiave primaria il codice del buono, tuttavia si è deciso di inserire comunque anche l’attributo booleano sul buono sconto chiamato “valido”, siccome la catena per qualche motivo potrebbe voler invalidare diversi buoni sconto.

Si è infine deciso di inserire una associazione tra la filiale ed il carrello e una tra la filiale ed il prestito (non rappresentate in figura), chiamate nello schema er “effettua” e “vende”. Ne è stato necessario l’inserimento per recuperare il codice della filiale in cui sono effettivamente avvenuti il prestito del cd o l’acquisto del carrello da parte del cliente, in modo da diminuire le rispettive quantità di cd disponibili in esposizione all’interno delle filiali.

Schema Er Completo



Schema logico

La progettazione logica del database avviene in diversi passaggi:

1) Eliminazione delle gerarchie ISA

2) Selezione delle chiavi primarie ed eliminazione delle chiavi esterne

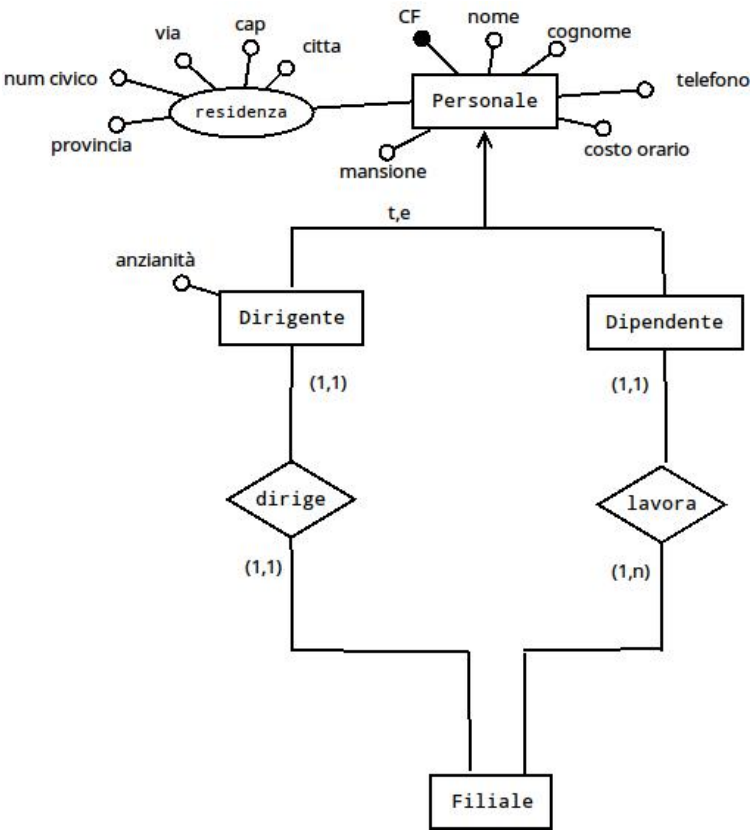
3) Trasformazione degli attributi composti e multipli

4) Traduzione delle entità e delle associazioni in schemi di relazioni

5) Verifica della normalizzazione

Eliminazione delle gerarchie ISA

L'eliminazione delle gerarchie è la prima fase del progetto logico. Nel database è presente una sola generalizzazione, ovvvero quella del personale, che lo suddivide in dirigenti e dipendenti.



La generalizzazione è di tipo totale-esclusiva. Esistono quindi tre vie:

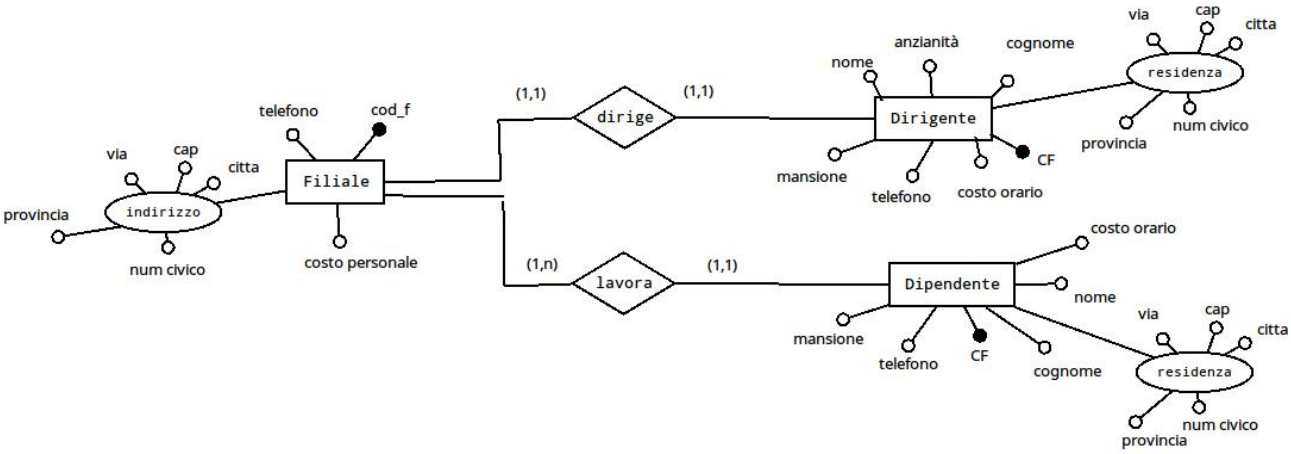
1) Mantenimento delle entità, tutte le entità vengono mantenute e le entità figlie vengono associate al padre ed identificate esternamente tramite l'associazione

2) Collasso verso l’alto, in questo caso le entità figlie vengono riunite nell’entità padre, i loro attributi diventano opzionali nel padre e si fa uso di selettori

3) Collasso verso il basso, si elimina l’entità padre trasferendone tutti gli attributi e le associazioni nelle entità figlie

Si è scelto il collasso verso il basso, siccome l’entità padre non presenta alcuna associazione con altre entità. Era possibile scegliere anche il mantenimento delle entità, non avrebbe comportato differenze sostanziali. Al contrario il collasso verso l’alto era certamente il più sconveniente.

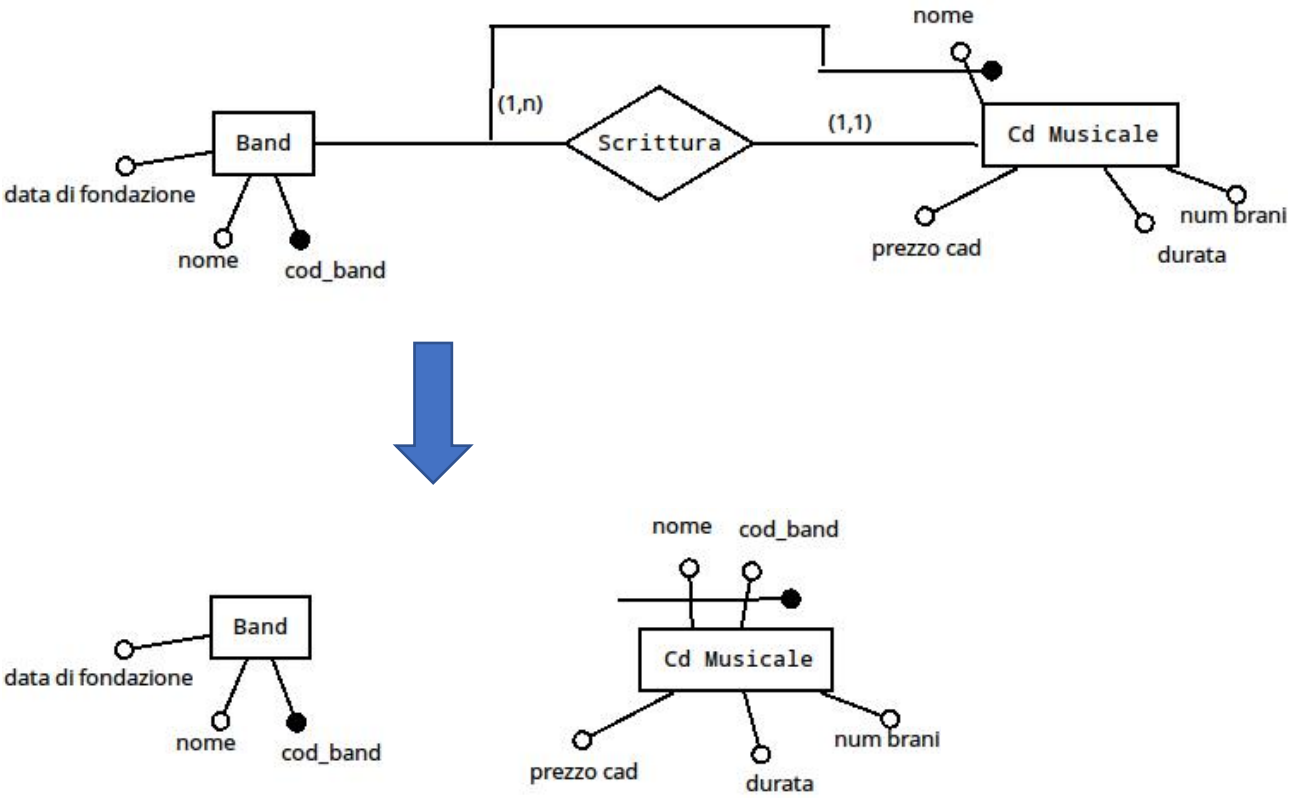
Lo schema viene quindi trasformato in questo modo:



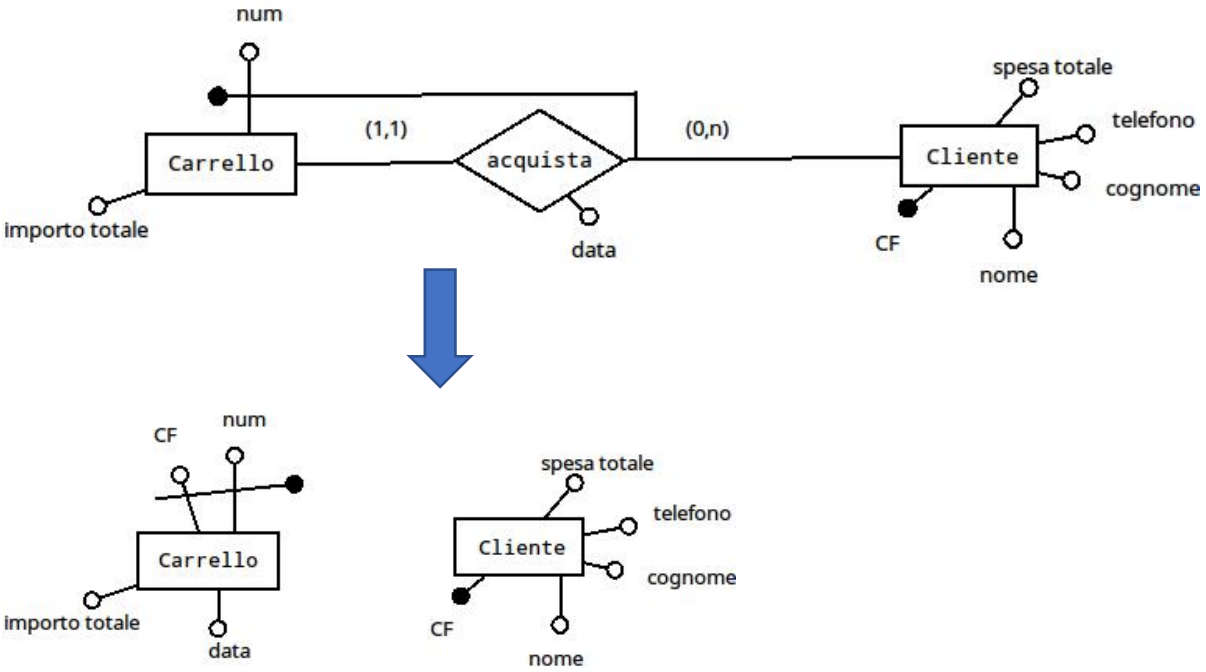
Selezione delle chiavi primarie ed eliminazione delle chiavi esterne

Di seguito verranno eliminate le chiavi esterne presenti nello schema er.

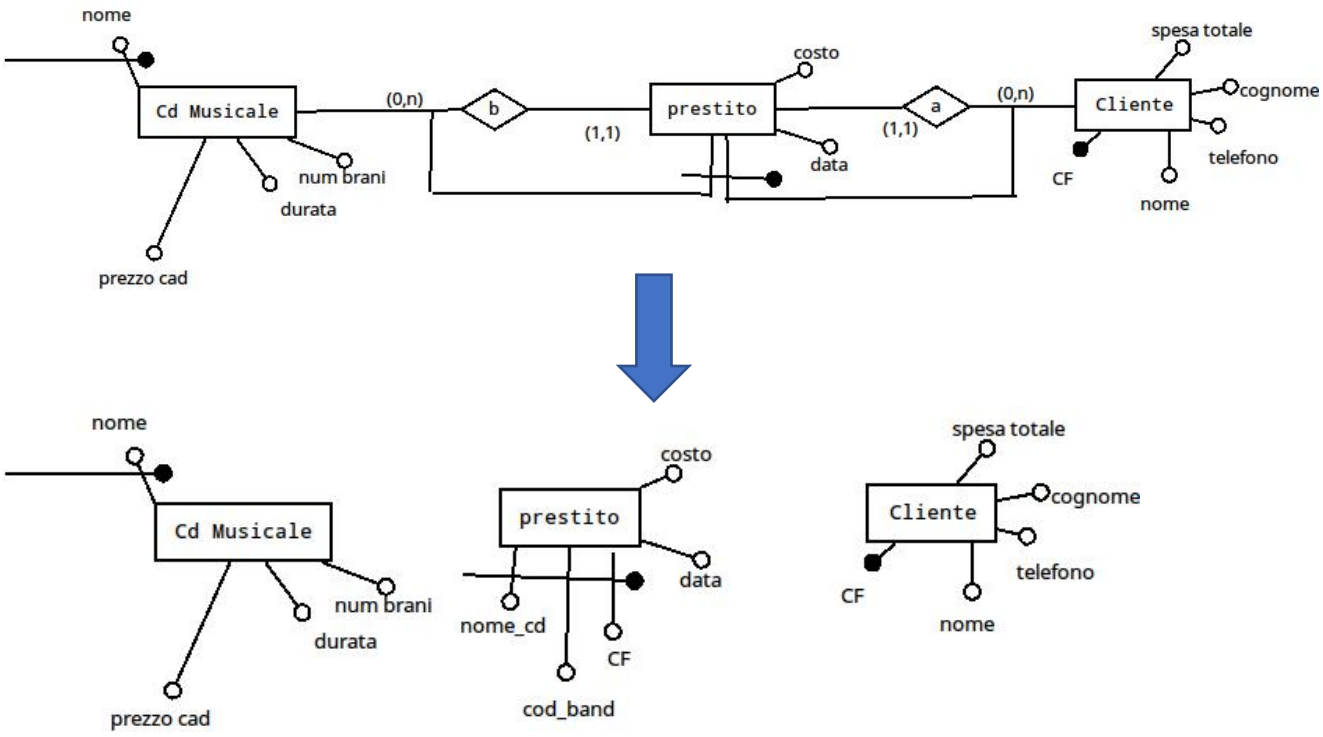
L’entità “Cd musicale” ha una componente di identificazione esterna dall’entità band



L’entità “carrello” ha una componente di identificazione esterna dall’entità “cliente”



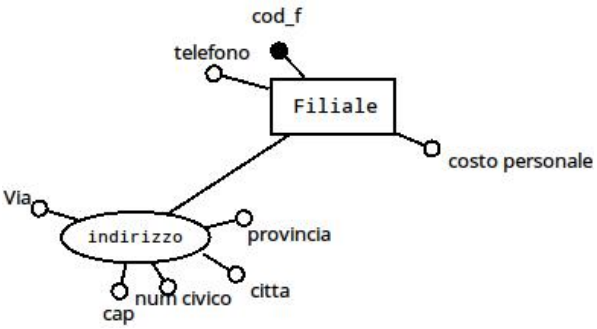
L’entità prestito ha delle componenti di identificazione esterna derivate dalle entità “cliente” e “Cd musicale”

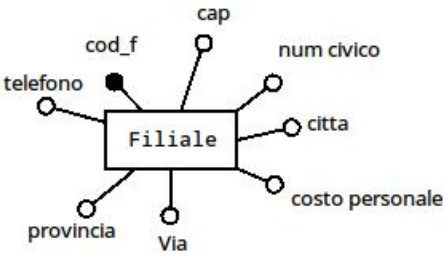


Trasformazione degli attributi composti e multipli

Come terza fase della progettazione logica è ora necessario eliminare gli attributi composti e multipli, siccome non è possibile tradurli nello schema relazionale.

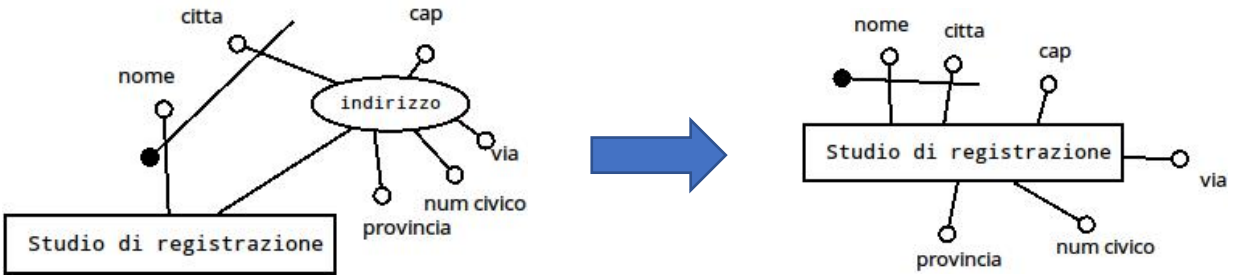
Si è eliminato l’attributo composto indirizzo dell’entità filiale e scomposto negli attributi più semplici



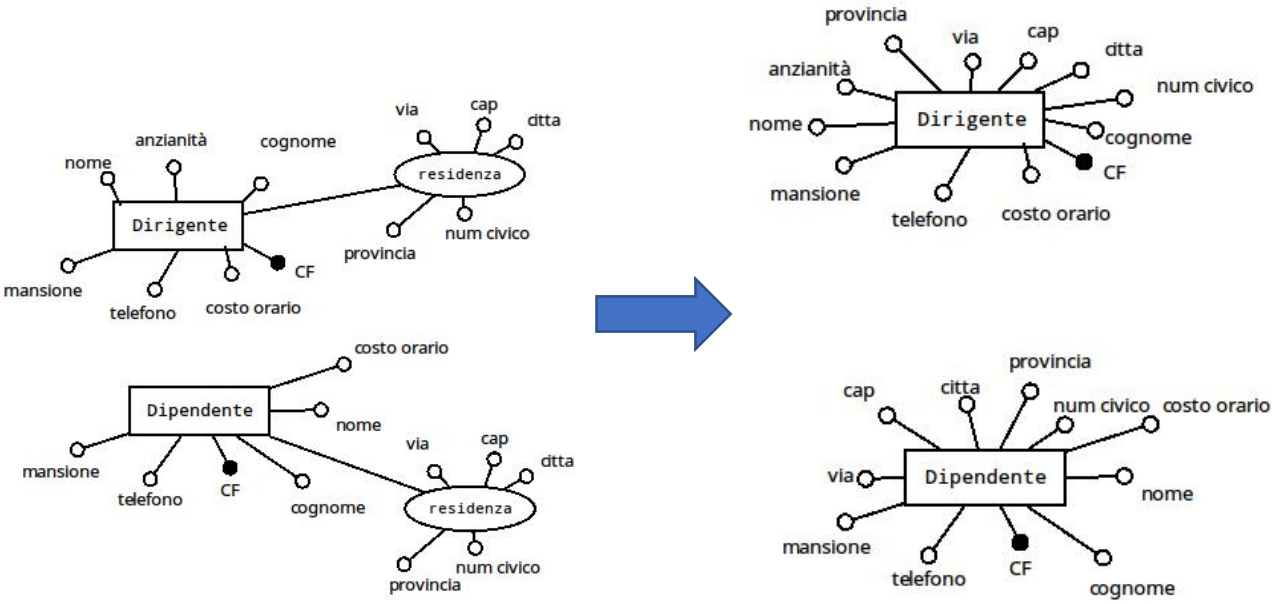


Si è eliminato l’attributo composto indirizzo dell’entità magazzino e scomposto negli attributi più semplici

Si è eliminato l’attributo composto indirizzo dell’entità “studio di registrazione” e scomposto negli attributi più semplici

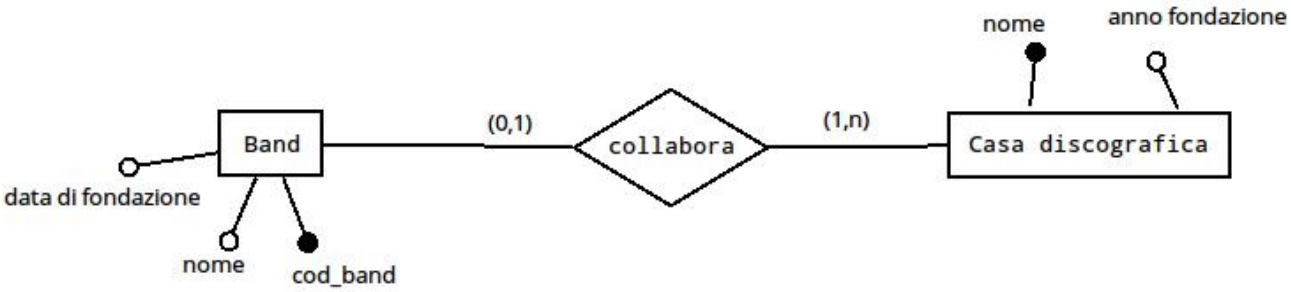


Si sono eliminati gli attributi composti “residenza” dalle entità “dipendente” e “dirigente” e sono stati scomposti negli attributi più semplici



Traduzione delle entità e delle associazioni in schemi di relazioni

In questa sezione si traducono le entità e le associazioni che sono state analizzate fino ad ora nello schema relazionale

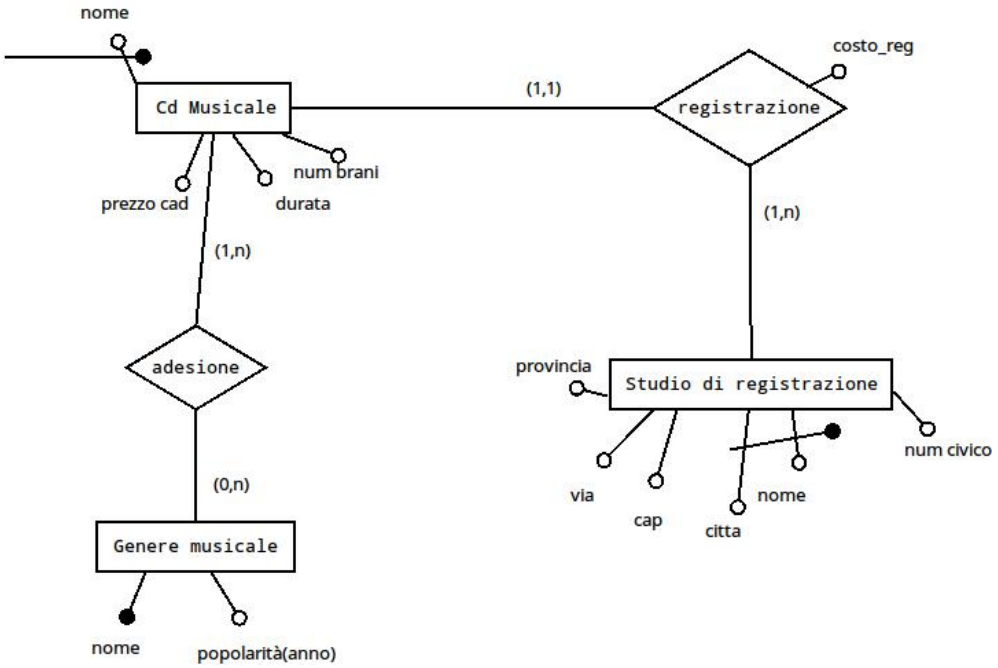


Siccome l'associazione "collabora" tra le entità "band" e "casa discografica" è 1:N è stata possibile accorparla nell'entità band

CASA_DISCOGRAFICA(<u>nome</u>, anno_fondazione)

BAND(<u>cod_band</u>, nome, data_fondazione, nome_c)

FK: nome_c REFERENCES CASA_DISCOGRAFICA



Siccome l'associazione "registrazione" è 1:N è stata accorpata nell'entità "Cd musicale" .

L'associazione "adesione" invece è N:M quindi è stata utilizzata la traduzione standard a tre entità.

Inoltre è stata accorpata l'associazione "scrittura" di tipo 1:N che collega l'entità band con l'entità "cd musicale" all'interno di quest'ultima (in figura non è stata mostrata).

STUDIO_DI_REGISTRAZIONE(<u>nome</u>, via, cap, <u>citta</u>, num_civico, provincia)

CD_MUSICALE(<u>nome</u>, <u>cod_band</u>, nome_studio, prezzo_cad, durata, num_brani, costo_reg, citta)

FK: cod_band REFERENCES BAND

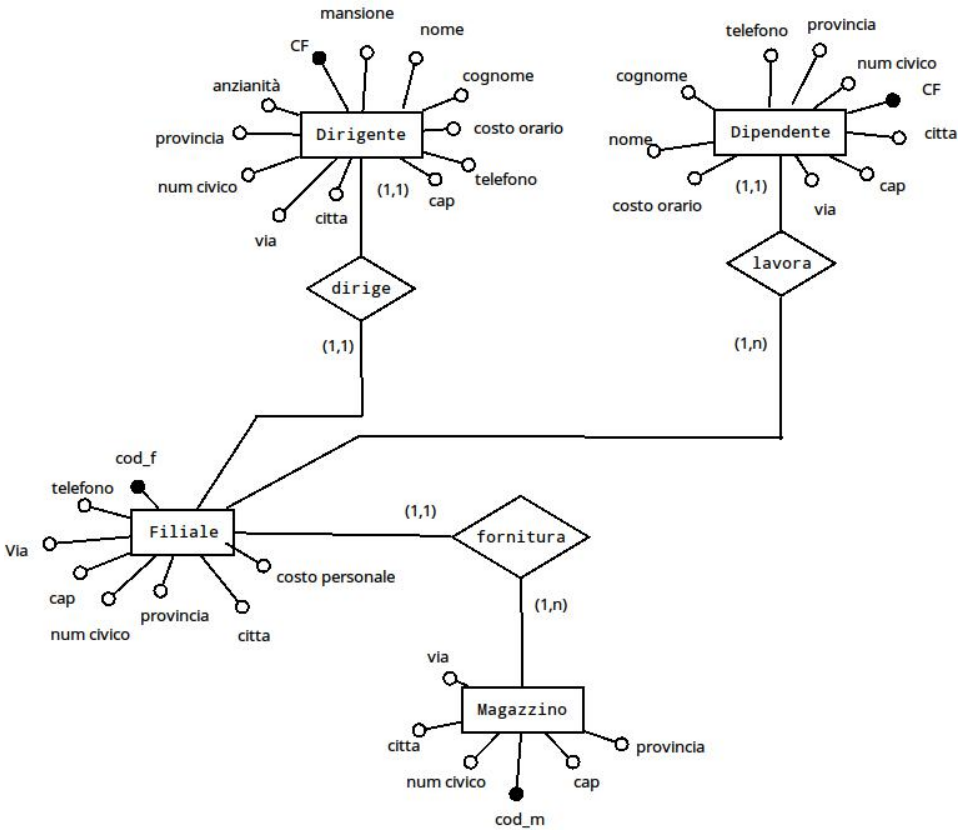
FK: nome_studio, citta REFERENCES STUDIO_DI_REGISTRAZIONE NOT NULL

GENERE_MUSICALE(<u>nome</u>, popolarità)

ADESIONE(<u>nome</u>, <u>cod_band</u>, <u>nome_cd</u>)

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: nome REFERENCES GENERE_MUSICALE



L’associazione “fornitura” che lega il magazzino con la filiale è stata accorpata nell’entità filiale siccome è di tipo 1:N.

L’associazio “dirige” invece è di tipo 1:1, si è deciso quindi di utilizzare la traduzione standard a tre tabelle.

L’associazione “lavora” invece ha cardinalità 1:N, è stata quindi accorpata nell’entità dipendente.

MAGAZZINO(<u>cod_m</u>, via, cap, citta, num_civico, provincia)

DIRIGENTE(<u>cf</u>, nome, cognome, costo_orario, telefono, via, cap, citta, num_civico, provincia, anzianità)

FILIALE(<u>cod_f</u>, costo_personale, telefono, via, cap, citta, num_civico, provincia, cod_m)

FK: cod_m REFERENCES MAGAZZINO NOT NULL

DIRIGE(<u>cod_f</u>, cf_dir)

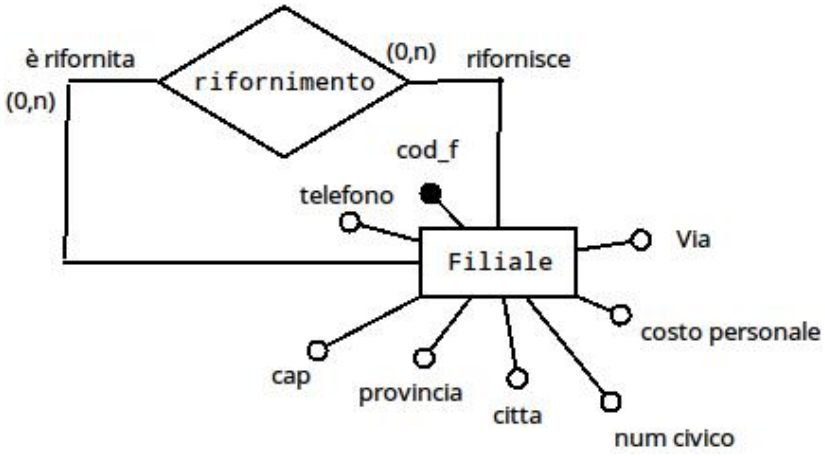
FK: cod_f REFERENCES FILIALE

FK: cf_dir REFERENCES DIRIGENTE

AK: cf_dir

DIPENDENTE(<u>cf</u>, nome, cognome, costo_orario, telefono, via, cap, citta, num_civico, provincia, cod_f)

FK: cod_f REFERENCES FILIALE NOT NULL



L’auto_associazione “rifornimento” dell’entità filiale è di tipo N:M, si è deciso quindi di utilizzare la traduzione standard a due tabelle.

(Ho ripetuto la traduzione dell’entità filiale per chiarezza)

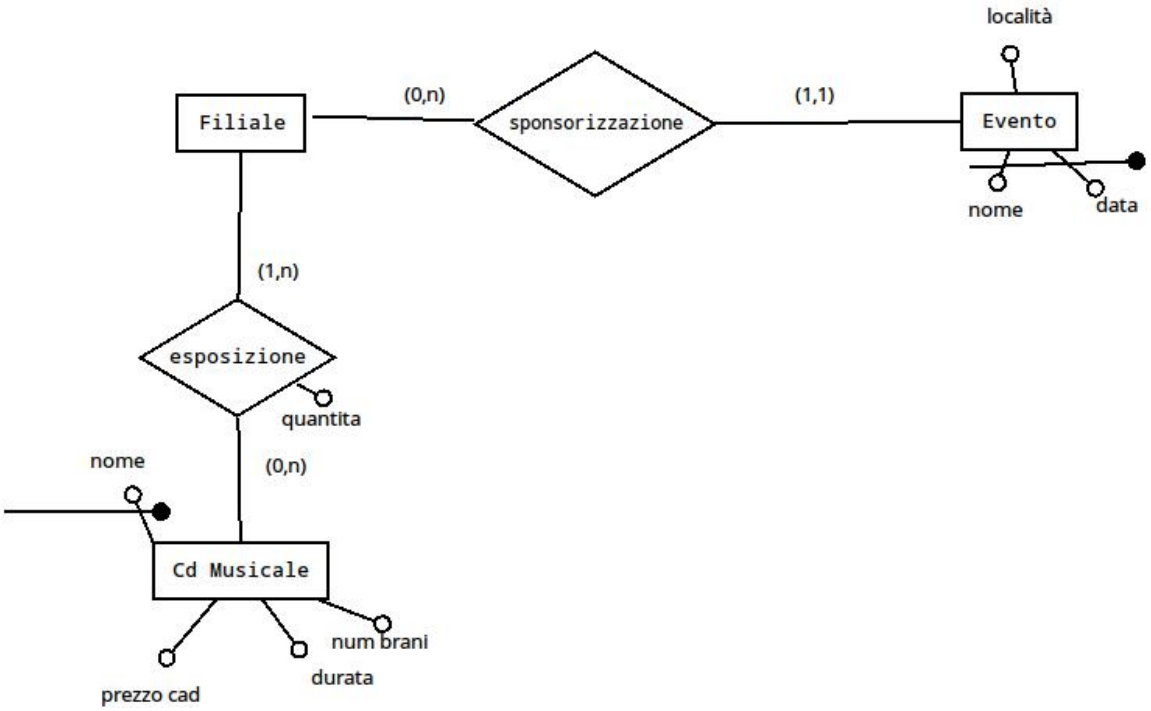
FILIALE(<u>cod_f</u>, costo_personale, telefono, via, cap, citta, num_civico, provincia, cod_m)

FK: cod_m REFERENCES MAGAZZINO NOT NULL

RIFORNIMENTO(<u>cod_fornitore</u>, <u>cod_rifornisce</u>)

FK: cod_fornitore REFERENCES FILIALE

FK: cod_rifornisce REFERENCES FILIALE



L’associazione “sponsorizza” è di tipo 1:N, è stato quindi possibile accorparla nell’entità evento.

L’associazione “eposizione” invece è di tipo N:M, è stato quindi necessario utilizzare la traduzione standard.

Di seguito sono riportate solamente le traduzioni dell’entità “evento” ed “esposizione” siccome le entità “filiale” e “cd musicale” erano state già tradotte in precedenza.

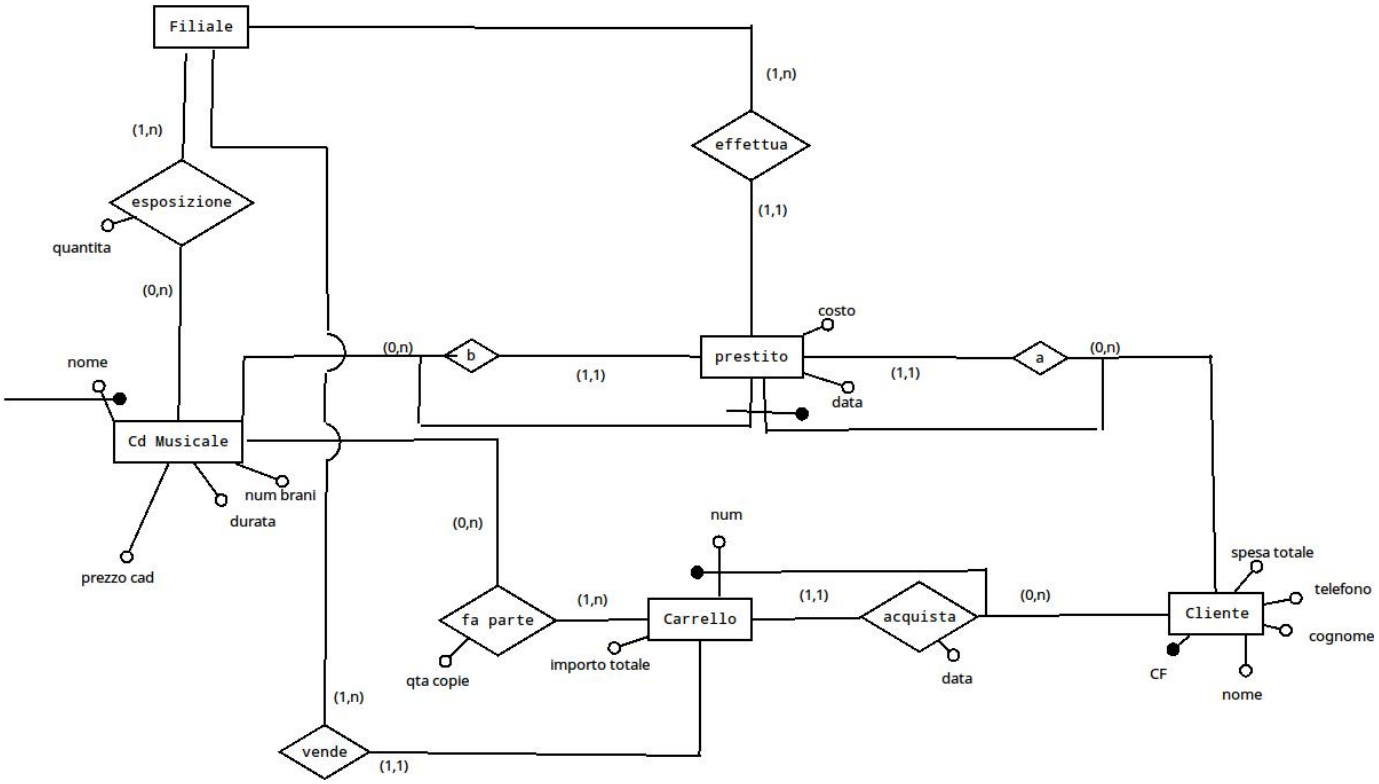
EVENTO(<u>nome</u>, <u>data</u>, località, cod_f)

FK: cod_f REFERENCES FILIALE NOT NULL

ESPOSIZIONE(<u>nome_cd</u>, <u>cod_band</u>, <u>cod_f</u>, quantita)

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: cod_f REFERENCES FILIALE



Le associazioni “vende” e “effettua” siccome sono di tipo 1:N sono state accorpate rispettivamente nelle entità “carrello” e “prestito”.

Nell’entità prestito inoltre sono state accorpate anche le associazioni “a” e “b” che collegano il prestito con il cliente e con il cd musicale, ciò è stato possibile perchè entrambe sono di tipo 1:N.

L’associazione “acquista” è anch’essa di tipo 1:N e quindi viene accorpata nell’entità carrello.

Infine l’associazione “fa parte” è di tipo N:M, viene quindi utilizzata la traduzione standard.

CLIENTE(<u>cf</u>, nome, cognome, telefono, spesa_totale)

PRESTITO(data, <u>cf</u>, <u>nome_cd</u>, <u>cod_band,</u> costo, cod_f)

FK: cf REFERENCES CLIENTE

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: cod_f REFERENCES FILIALE NOT NULL

CARRELLO(<u>num, cf</u>, data, importo_totale, cod_f)

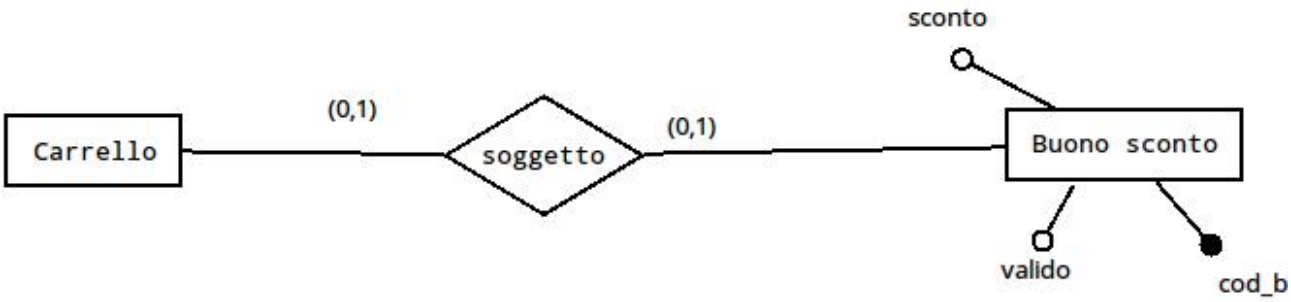
FK: cf REFERENCES CLIENTE

FK: cod_f REFERENCES FILIALE NOT NULL

FA_PARTE(<u>num, cf, nome_cd, cod_band,</u> qta_copie)

FK: num, cf REFERENCES CARRELLO

FK: nome_cd, cod_band REFERENCES CD_MUSICALE



L'associazione "soggetto" è di tipo 1:1, la partecipazione di entrambe le entità è opzionale.

L'unica traduzione possibile in questo caso è quella standard a tre entità (la traduzione dell'entità carrello è stata omessa perchè è stata già mostrata precedentemente).

BUONO_SCONTO(<u>cod_b</u>, sconto, valido)

SOGGETTO(<u>cod_b</u>, num, cf)

FK: num, cf REFERENCES CARRELLO

FK: cod_b REFERENCES BUONO_SCONTO

AK: num, cf

Lo schema logico finale risulterà quindi essere il seguente:

CASA_DISCOGRAFICA(nome, anno_fondazione)

BAND(cod_band, nome, data_fondazione, nome_c)

FK: nome_c REFERENCES CASA_DISCOGRAFICA

STUDIO_DI_REGISTRAZIONE(nome, via, cap, citta, num_civico, provincia)

CD_MUSICALE(nome, cod_band, nome_studio, prezzo_cad, durata, num_brani, costo_reg, citta)

FK: cod_band REFERENCES BAND

FK: nome_studio, citta REFERENCES STUDIO_DI_REGISTRAZIONE NOT NULL

GENERE_MUSICALE(nome, popolarità)

ADESIONE(nome, cod_band, nome_cd)

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: nome REFERENCES GENERE_MUSICALE

MAGAZZINO(cod_m, via, cap, citta, num_civico, provincia)
DIRIGENTE(cf, nome, cognome, costo_orario, telefono, via, cap, citta, num_civico, provincia, anzianità)
FILIALE(cod_f, costo_personale, telefono, via, cap, citta, num_civico, provincia, cod_m)
FK: cod_m REFERENCES MAGAZZINO NOT NULL
DIRIGE(cod_f, cf_dir)

FK: cod_f REFERENCES FILIALE

FK: cf_dir REFERENCES DIRIGENTE

AK: cf_dir

DIPENDENTE(cf, nome, cognome, costo_orario, telefono, via, cap, citta, num_civico, provincia, cod_f)
FK: cod_f REFERENCES FILIALE NOT NULL

RIFORNIMENTO(cod_fornitore, cod_rifornisce)

FK: cod_fornitore REFERENCES FILIALE

FK: cod_rifornisce REFERENCES FILIALE

EVENTO(nome, data, località, cod_f)

FK: cod_f REFERENCES FILIALE NOT NULL

ESPOSIZIONE(nome_cd, cod_band, cod_f, quantita)

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: cod_f REFERENCES FILIALE

CLIENTE(cf, nome, cognome, telefono, spesa_totale)

PRESTITO(data, cf, nome_cd, cod_band, costo, cod_f)

FK: cf REFERENCES CLIENTE

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

FK: cod_f REFERENCES FILIALE NOT NULL

BUONO_SCONTO(cod_b, sconto, valido)

CARRELLO(num, cf, data, importo_totale, cod_f)

FK: cf REFERENCES CLIENTE

FK: cod_f REFERENCES FILIALE NOT NULL

FA_PARTE(num, cf, nome_cd, cod_band, qta_copie)

FK: num, cf REFERENCES CARRELLO

FK: nome_cd, cod_band REFERENCES CD_MUSICALE

SOGGETTO(cod_b, num, cf)

FK: num, cf REFERENCES CARRELLO

FK: cod_b REFERENCES BUONO_SCONTO

AK: num, cf

Verifica della normalizzazione

Il database risulta essere in forma normale.

Studio dati derivati

Nel progetto sono presenti diversi dati derivati, si è deciso di limitare lo studio a due soli di questi. In particolare si è studiato il dato derivato “importo totale” presente nell’entità carrello e “costo personale” presente nell’entità filiale.

1) Il dato derivato “importo totale” è calcolato come la somma dei prezzi di tutti i prodotti moltiplicato le quantità rispettive e tolto il valore di un buono sconto se utilizzato.

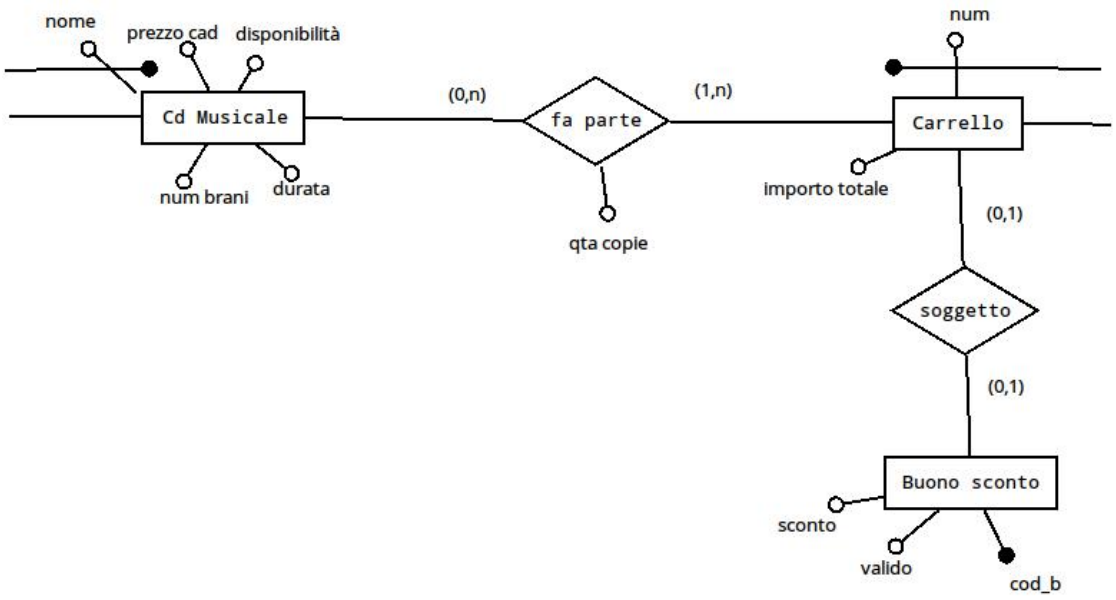


Tabella dei volumi

Si è considerata una gamma di scelta di 10000 cd differenti, in media un carrello contiene 2.5 cd e un cd fa parte di 3 carrelli (con 30000 record nell’associazione).

Concetto	Tipo	Volume
Cd Musicale	E	10000
Fa_parte	R	30000
Carrello	E	12000
Soggetto	R	3000
Buono sconto	E	6000

Tabella delle operazioni

Sono state considerate due operazioni:

1) Visualizzare importo totale di un carrello
2) Dato cd aggiungerlo a un carrello (dato cd e carrello)
3) Utilizzo di un buono sconto in un carrello (dato carrello e codice buono)

Si è supposto che l'operazione di visualizzazione dell'importo totale avvenga con minore frequenza rispetto all'aggiunta di un cd nel carrello.

Operazione	Tipo	Frequenza
Op1	I	400/g
Op2	I	500/g
Op3	I	50/g

Tabella accessi con dato derivato

Si suppone che i dati del cd siano già dati (prezzo compreso), nell'operazione 2 non sarà quindi necessario andare ad effettuare una lettura sull'entità cd.

Si suppone invece di non conoscere fin da subito il valore dello sconto del buono, sarà quindi necessaria un lettura extra sul buono per recuperarlo.

Operazione	Concetto	Accessi	Tipo
Op1	Carrello	1	L
Op2	Fa_parte	1	S
Op2	Carrello	1	L
Op2	Carrello	1	S
Op3	Buono Sconto	1	L
Op3	Soggetto	1	S
Op3	Carrello	1	L
Op3	Carrello	1	S

Costo: 400+2500+300 = 3200/g

Tabella accessi senza dato derivato

Operazione	Concetto	Accessi	Tipo
Op1	Carrello	1	L
Op1	Fa_parte	2.5	L
Op1	Cd Musicale	2.5	L
Op1	Soggetto	0.25	L
Op1	Buono sconto	0.25	L
Op2	Fa_parte	1	S
Op3	Soggetto	1	S

Costo: 2600+1000+100 = 3700/g

In questo caso conviene tenere il dato derivato “importo totale”.

2) Il dato derivato “costo personale” dell’entità “filiale” è calcolato come la somma dei costi orari di tutti i dipendenti e dirigenti che lavorano in quella fialiale. Lo studio sul dato derivato è stato effettuato sullo schema er seguente, che rappresenta la forma d’implementazione scelta.

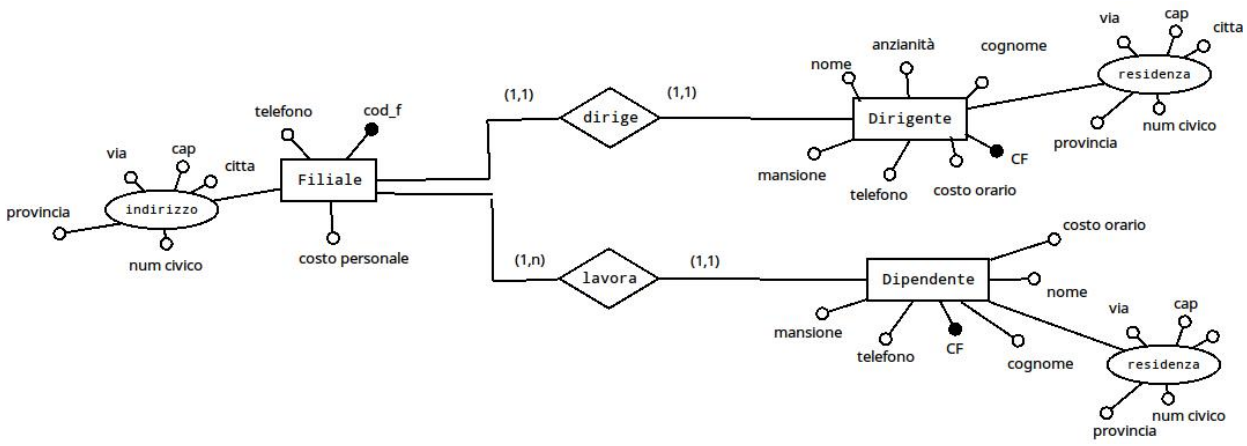


Tabella dei volumi

Si è considerata una catena di negozi abbastanza famosa con 200 filiali differenti sparse per la nazione. Ogni filiale è seguita da un dirigente diverso, ci saranno quindi 200 dirigenti. In una filiale lavorano più dipendenti, ma un dipendente lavora per una e una sola filiale. Sono stati considerati in media 30 dipendenti per filiale.

Concetto	Tipo	Volume
Filiale	E	200
Dirige	R	200
Dirigente	E	200
Dipendente	E	6000
Lavora	R	6000

Tabella delle operazioni

Sono state considerate tre operazioni:

1) Modifica del costo orario di un dipendente, dato codice fiscale
2) Modifica del costo orario di un dirigente, dato codice fiscale
3) Controllo sul costo del personale di una determinata filiale

Sono tutte operazioni che avvengono abbastanza di rado in una azienda, ma quando si parla di una catena di negozi queste aumentano all'aumentare delle sedi e dei dipendenti, è stato quindi necessario tenerne conto.

Operazione	Tipo	Frequenza
Op1	I	10/g
Op2	I	2/g
Op3	I	5/g

Tabella accessi con dato derivato

Operazione	Concetto	Accessi	Tipo
Op1	Dipendente	1	L
Op1	Dipendente	1	S
Op1	Lavora	1	L
Op1	Filiale	1	L

Op1	Filiale	1	S
Op2	Dirigente	1	L
Op2	Dirigente	1	S
Op2	Dirige	1	L
Op2	Filiale	1	L
Op2	Filiale	1	S
Op3	Filiale	1	L

Costo: 70+14+5 = 89/g

Tabella accessi senza dato derivato

Operazione	Concetto	Accessi	Tipo
Op1	Dipendente	1	L
Op1	DIpendente	1	S
Op2	Dirigente	1	L
Op2	Dirigente	1	S
Op3	Filiale	1	L
Op3	Dirige	1	L
Op3	Dirigente	1	L
Op3	Lavora	30	L
Op3	Dipendente	30	L

Costo: 30+6+315 = 351/g

Come mostra lo studio appena svolto, inaspettatamente, conviene tenere il dato derivato in questo caso.

Query di creazione

Di seguito si presentano le query per la creazione dell'intero database. Si evidenziano in verde i vincoli imposti non esprimibili durante la fase di progettazione.

```
CREATE TABLE CASA_DISCOGRAFICA(
        nome VARCHAR(30) PRIMARY KEY,
        anno_fondazione INTEGER NOT NULL

);



CREATE TABLE BAND(
        cod_band INTEGER PRIMARY KEY,
        nome VARCHAR(30) NOT NULL,
        data_fondazione date NOT NULL,
        nome_c VARCHAR(30),
        FOREIGN KEY (nome_c) REFERENCES CASA_DISCOGRAFICA(nome)
        ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE STUDIO_DI_REGISTRAZIONE(
        nome VARCHAR(30) NOT NULL,
        via VARCHAR(60) NOT NULL,
        cap CHAR(5) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        num_civico INTEGER NOT NULL,
        provincia VARCHAR(60) NOT NULL,
        PRIMARY KEY(nome,citta),

        CHECK(num_civico > 0 )
);
```

```
CREATE TABLE CD_MUSICALE(
        nome VARCHAR(60) NOT NULL,
        cod_band INTEGER NOT NULL,
        nome_studio VARCHAR(60) NOT NULL,
        prezzo_cad INTEGER NOT NULL,
        durata TIME NOT NULL,
        num_brani INTEGER NOT NULL,
        costo_reg DECIMAL(10,3) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        PRIMARY KEY(nome, cod_band),
        FOREIGN KEY (nome_studio, citta) REFERENCES
STUDIO_DI_REGISTRAZIONE(nome, citta)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cod_band) REFERENCES BAND(cod_band)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(prezzo_cad > 0),
        CHECK(durata > '00:02:00' ),
        CHECK(num_brani > 0)


);


CREATE TABLE GENERE_MUSICALE(
        nome VARCHAR(60) PRIMARY KEY,
        popolarita INTEGER NOT NULL,
        CHECK(popolarita > 1500)


);

CREATE TABLE ADESIONE(
        genere VARCHAR(60) NOT NULL,
        cod_band INTEGER NOT NULL,
        nome_cd VARCHAR(60) NOT NULL,
```

```
        PRIMARY KEY(genere, cod_band, nome_cd),

        FOREIGN KEY (nome_cd, cod_band) REFERENCES CD_MUSICALE(nome,
cod_band)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (genere) REFERENCES GENERE_MUSICALE(nome)
        ON DELETE CASCADE ON UPDATE CASCADE


);


CREATE TABLE MAGAZZINO(
        cod_m INTEGER PRIMARY KEY,
        via VARCHAR(60) NOT NULL,
        cap CHAR(5) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        num_civico INTEGER NOT NULL,
        provincia VARCHAR(60) NOT NULL,

        CHECK(num_civico > 0 )

);


CREATE TABLE DIRIGENTE(
        cf CHAR(16) PRIMARY KEY,
        nome VARCHAR(60) NOT NULL,
        cognome VARCHAR(60) NOT NULL,
        costo_orario DECIMAL(10,3) NOT NULL,
        telefono VARCHAR(10),
        via VARCHAR(60) NOT NULL,
        cap CHAR(5) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        num_civico INTEGER NOT NULL,
        provincia VARCHAR(60) NOT NULL,
```

```
        anzianita INTEGER NOT NULL,

        CHECK(num_civico > 0 ),
        CHECK(anzianita > 0 ),
        CHECK(costo_orario > 0)



);

CREATE TABLE FILIALE(
        cod_f INTEGER PRIMARY KEY,
        costo_personale DECIMAL(15,3) NOT NULL,
        telefono VARCHAR(10) NOT NULL,
        via VARCHAR(60) NOT NULL,
        cap CHAR(5) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        num_civico INTEGER NOT NULL,
        provincia VARCHAR(60) NOT NULL,
        cod_m INTEGER NOT NULL,
        FOREIGN  KEY (cod_m) REFERENCES MAGAZZINO(cod_m)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(num_civico > 0),
        CHECK(costo_personale >= 0)


);

CREATE TABLE DIRIGE(
        cod_f INTEGER PRIMARY KEY,
        cf_dir CHAR(16) NOT NULL,
        UNIQUE(cf_dir),
        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cf_dir) REFERENCES DIRIGENTE(cf)
```

```
        ON DELETE CASCADE ON UPDATE CASCADE


);

CREATE TABLE DIPENDENTE(
        cf CHAR(16) PRIMARY KEY,
        nome VARCHAR(60) NOT NULL,
        cognome VARCHAR(60) NOT NULL,
        costo_orario DECIMAL(10,3) NOT NULL,
        telefono VARCHAR(10),
        via VARCHAR(60) NOT NULL,
        cap CHAR(5) NOT NULL,
        citta VARCHAR(60) NOT NULL,
        num_civico INTEGER NOT NULL,
        provincia VARCHAR(60) NOT NULL,
        cod_f INTEGER NOT NULL,

        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(num_civico > 0 ),
        CHECK(costo_orario > 0)



);


CREATE TABLE RIFORNIMENTO(
        cod_fornitore INTEGER NOT NULL,
        cod_rifornisce INTEGER NOT NULL,
        PRIMARY KEY(cod_fornitore, cod_rifornisce),

        FOREIGN KEY (cod_fornitore) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,
```

```
        FOREIGN KEY (cod_rifornisce) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE EVENTO(
        nome VARCHAR(60)  NOT NULL,
        data DATE NOT NULL,
        localita VARCHAR(60) NOT NULL,
        cod_f INTEGER NOT NULL,

        PRIMARY KEY(nome, data),
        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(LENGTH(localita) >= 2),
        CHECK(LENGTH(nome) >= 3)

);

CREATE TABLE ESPOSIZIONE(
        nome_cd VARCHAR(60) NOT NULL,
        cod_band INTEGER NOT NULL,
        cod_f INTEGER NOT NULL,
        quantita INTEGER,
        PRIMARY KEY(nome_cd, cod_band, cod_f),

        FOREIGN KEY (nome_cd, cod_band) REFERENCES CD_MUSICALE(nome,
cod_band)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK (quantita >= 0)

);
```

```
CREATE TABLE CLIENTE(
        cf CHAR(16) PRIMARY KEY,
        nome VARCHAR(60) NOT NULL,
        cognome VARCHAR(60) NOT NULL,
        telefono VARCHAR(10),
        spesa_totale DECIMAL(10,3) NOT NULL


);

CREATE TABLE PRESTITO(
        data date NOT NULL,
        cf CHAR(16) NOT NULL,
        nome_cd VARCHAR(60) NOT NULL,
        cod_band INTEGER NOT NULL,
        costo DECIMAL (10,3) NOT NULL,
        cod_f INTEGER NOT NULL,
        PRIMARY KEY(cf, nome_cd, cod_band),

        FOREIGN KEY (nome_cd, cod_band) REFERENCES CD_MUSICALE(nome,
cod_band)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cf) REFERENCES CLIENTE(cf)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(costo > 0)

);

CREATE TABLE BUONO_SCONTO(
        cod_b INTEGER PRIMARY KEY,
        sconto DECIMAL(3,2),
```

```
        valido BOOLEAN NOT NULL,


        CHECK(sconto <= 1 and sconto > 0)

);


CREATE TABLE CARRELLO(
        num INTEGER NOT NULL,
        cf CHAR(16) NOT NULL,
        data date NOT NULL,
        importo_totale DECIMAL(10,3),

        cod_f INTEGER NOT NULL,
        PRIMARY KEY(num, cf),

        FOREIGN KEY (cf) REFERENCES CLIENTE(cf)
        ON DELETE CASCADE ON UPDATE CASCADE,

        FOREIGN KEY (cod_f) REFERENCES FILIALE(cod_f)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(importo_totale >= 0),
        CHECK(num > 0)

);



CREATE TABLE FA_PARTE(
        num INTEGER NOT NULL,
        cf CHAR(16) NOT NULL,
        nome_cd VARCHAR(60) NOT NULL,
        cod_band INTEGER NOT NULL,
        qta_copie INTEGER NOT NULL,
```

```
        PRIMARY KEY(num, cf, nome_cd, cod_band),

        FOREIGN KEY (num, cf) REFERENCES CARRELLO(num, cf)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (nome_cd, cod_band) REFERENCES CD_MUSICALE(nome,
cod_band)
        ON DELETE CASCADE ON UPDATE CASCADE,

        CHECK(qta_copie > 0)

);

CREATE TABLE SOGGETTO(

        cod_b INTEGER PRIMARY KEY,
        num INTEGER,
        cf CHAR(16),

        UNIQUE(num, cf),

        FOREIGN KEY (num, cf) REFERENCES CARRELLO(num, cf)
        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (cod_b) REFERENCES BUONO_SCONTO(cod_b)
        ON DELETE CASCADE ON UPDATE CASCADE

);
```

Triggers e stored procedure

Di seguito sono implementati i trigger principali necessari al corretto funzionamento del database. Sono inotre implementate due stored procedure “extra” che gestiscono il rifornimento dei cd.

/* aggiornamento del "costo personale" di una filiale dopo “l’affidamento” a un dirigente */

```
CREATE OR REPLACE FUNCTION aggiorna_costo_personale() RETURNS trigger AS $$
                DECLARE
                costo DECIMAL(10,3);
                BEGIN
                        costo = (SELECT costo_orario from DIRIGENTE WHERE cf =
NEW.cf_dir);
                        UPDATE FILIALE
                                SET costo_personale = costo_personale + costo
                                WHERE cod_f = NEW.cod_f;
                 RETURN NEW;
                END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER COSTO_PERSONALE
        AFTER INSERT ON DIRIGE
        FOR EACH ROW EXECUTE PROCEDURE aggiorna_costo_personale();
```

/* aggiornamento del "costo personale" di una filiale dopo l’assunzione di un dipendente */

```
CREATE OR REPLACE FUNCTION aggiorna_costo_personale2() RETURNS trigger AS $$

                BEGIN

                        UPDATE FILIALE
                                SET costo_personale = costo_personale + NEW.costo_orario
                                WHERE cod_f = NEW.cod_f;
```

```
                RETURN NEW;
                END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER COSTO_PERSONALE2
        AFTER INSERT ON DIPENDENTE
        FOR EACH ROW EXECUTE PROCEDURE aggiorna_costo_personale2();
```

/* aggiornamento del "costo personale" di una filiale dopo il licenziamento di un dipendente */

```
CREATE OR REPLACE FUNCTION riduci_costo_personale2() RETURNS trigger AS $$

                BEGIN

                        UPDATE FILIALE
                                SET costo_personale = costo_personale - OLD.costo_orario
                                WHERE cod_f = OLD.cod_f;
                 RETURN OLD;
                END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER RIDUCI_COSTO_PERSONALE2
        AFTER DELETE ON DIPENDENTE
        FOR EACH ROW EXECUTE PROCEDURE riduci_costo_personale2();
```

/* aggiornamento del "costo personale" di una filiale dopo la rimozione di un dirigente */

```
CREATE OR REPLACE FUNCTION riduci_costo_personale() RETURNS trigger AS $$
```

```
            DECLARE
            costo DECIMAL(10,3);
            BEGIN
                costo = (SELECT costo_orario from DIRIGENTE WHERE cf =
OLD.cf_dir);
                UPDATE FILIALE
                    SET costo_personale = costo_personale - costo
                    WHERE cod_f = OLD.cod_f;
             RETURN OLD;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER RIDUCI_COSTO_PERSONALE
        AFTER DELETE ON DIRIGE
        FOR EACH ROW EXECUTE PROCEDURE riduci_costo_personale();

/* trigger che tiene aggiornato il dato derivato costo personale di una filiale alla
modifica del costo orario di un dipendente */
CREATE OR REPLACE FUNCTION aggiorna_costop_dip() RETURNS trigger AS $$

            DECLARE
            costo DECIMAL(10,3);

            BEGIN
                costo = NEW.costo_orario - OLD.costo_orario;
                UPDATE FILIALE
                SET costo_personale = costo_personale + costo
                WHERE cod_f = NEW.cod_f;

             RETURN NEW;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER AGGIORNA_COSTOP_DIP
        AFTER UPDATE ON DIPENDENTE
        FOR EACH ROW EXECUTE PROCEDURE  aggiorna_costop_dip() ;
```

```
/* trigger che tiene aggiornato il dato derivato costo personale di una filiale alla
modifica del costo orario di un dirigente */

CREATE OR REPLACE FUNCTION aggiorna_costop_dir() RETURNS trigger AS $$

        DECLARE
        costo DECIMAL(10,3);
        codf INTEGER; /* codice filiale in cui lavora il dirigente */

        BEGIN

            /* recupero la filiale del dirigente */
            codf = (select cod_f from dirige d where d.cf_dir = NEW.cf);
            costo = NEW.costo_orario - OLD.costo_orario;
            UPDATE FILIALE
            SET costo_personale = costo_personale + costo
            WHERE cod_f = codf;

         RETURN NEW;
        END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER AGGIORNA_COSTOP_DIR
    AFTER UPDATE ON DIRIGENTE
    FOR EACH ROW EXECUTE PROCEDURE  aggiorna_costop_dir() ;


/* trigger che aggiorna la spesa totale del cliente dopo l'effettuo di un prestito */

CREATE OR REPLACE FUNCTION aggiorna_spesa_totale_p() RETURNS trigger AS $$

        BEGIN
            UPDATE CLIENTE
                SET spesa_totale = spesa_totale + NEW.costo
                WHERE cf = NEW.cf;
         RETURN NEW;
        END;
```

```
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER AGGIORNA_SPESA_TOTALE_P
        AFTER INSERT ON PRESTITO
        FOR EACH ROW EXECUTE PROCEDURE aggiorna_spesa_totale_p();




/* trigger che aggiorna la spesa totale di un cliente in base ai prodotti inseriti
all'interno di un carrello acquistato */

CREATE OR REPLACE FUNCTION aggiorna_spesa_totale_acq() RETURNS trigger AS $$
                DECLARE
                prezzo DECIMAL(10,3);

                BEGIN

                prezzo= (select prezzo_cad
                                from cd_musicale
                                where nome = NEW.nome_cd and cod_band =
NEW.cod_band);



                        UPDATE CLIENTE
                                SET spesa_totale = spesa_totale + (prezzo*NEW.qta_copie)
                                WHERE cf = NEW.cf;
                 RETURN NEW;
                END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER AGGIORNA_SPESA_TOTALE_ACQ
        AFTER INSERT ON FA_PARTE
        FOR EACH ROW EXECUTE PROCEDURE aggiorna_spesa_totale_acq();
```

```
/* trigger che diminuisce la quantita di cd disponibili di una filiale dopo un prestito */

CREATE OR REPLACE FUNCTION riduce_quantita_cd_pr() RETURNS trigger AS $$

        BEGIN

            UPDATE ESPOSIZIONE
                SET quantita = quantita - 1
                WHERE cod_f = NEW.cod_f and  nome_cd = NEW.nome_cd
                        and cod_band = NEW.cod_band;
         RETURN NEW;
        END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER RIDUCE_QT_CD_PR
        AFTER INSERT ON PRESTITO
        FOR EACH ROW EXECUTE PROCEDURE riduce_quantita_cd_pr();



/* trigger che diminuisce la quantita di cd disponibili di una filiale dopo l'inserimento di
questi all'interno di un carrello */

CREATE OR REPLACE FUNCTION riduce_quantita_cd_acq() RETURNS trigger AS $$
        DECLARE
            codf INTEGER;


        BEGIN



            codf = (select distinct cod_f
                    from carrello c
```

```
                    where NEW.num = c.num and NEW.cf = c.cf

                    );


                UPDATE ESPOSIZIONE
                    SET quantita = quantita - NEW.qta_copie
                    WHERE cod_f = codf and  nome_cd = NEW.nome_cd
                        and cod_band = NEW.cod_band;
            RETURN NEW;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER RIDUCE_QT_CD_ACQ
        AFTER INSERT ON FA_PARTE
        FOR EACH ROW EXECUTE PROCEDURE riduce_quantita_cd_acq();



/* trigger che aggiorna il dato derivato “importo totale” di un carrello in base ai cd
inseriti all’interno di quest’ultimo  */
CREATE OR REPLACE FUNCTION aumenta_importo_totale() RETURNS trigger AS $$
            DECLARE
                prezzo INTEGER;


            BEGIN

                prezzo = (select prezzo_cad
                        from cd_musicale
                        where nome = NEW.nome_cd and cod_band =
NEW.cod_band);


                UPDATE CARRELLO
                    SET importo_totale = importo_totale +
(prezzo*NEW.qta_copie)
```

```
                    WHERE num = NEW.num and cf = NEW.cf;
            RETURN NEW;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER AUMENTA_IMPORTO_TOTALE
        AFTER INSERT ON FA_PARTE
        FOR EACH ROW EXECUTE PROCEDURE  aumenta_importo_totale() ;
```

/* trigger che verifica la validità di un buono se utilizzato in un carrello. Se il buono viene utilizzato correttamente allora perderà la validità */

```
CREATE OR REPLACE FUNCTION controllo_buono() RETURNS trigger AS $$

            DECLARE
            valido BOOLEAN;
            prezzo_sconto DECIMAL(10,3);

            BEGIN

            valido = (select b.valido
                    from buono_sconto b
                    where b.cod_b = NEW.cod_b);
            prezzo_sconto = ((select importo_totale
                            from carrello c
                            where cf = NEW.cf and num = NEW.num) * (select sconto
                                                                    from buono_sconto
                                                                    where cod_b = NEW.cod_b));

                    IF valido = false THEN RAISE EXCEPTION 'buono non valido';
            ELSE IF valido = true THEN

                            UPDATE CARRELLO
```

```
                    SET importo_totale = importo_totale - prezzo_sconto
                    WHERE num = NEW.num and cf = NEW.cf;


                    UPDATE BUONO_SCONTO
                    SET valido = 'false'
                    WHERE cod_b = NEW.cod_b;


                END IF;
                END IF;
            RETURN NEW;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER CONTROLLO_BUONO
        AFTER INSERT OR UPDATE ON SOGGETTO
        FOR EACH ROW EXECUTE PROCEDURE  controllo_buono() ;

/* una filiale non può rifornirsi da sola */
CREATE OR REPLACE FUNCTION controllo_rif() RETURNS trigger AS $$

            BEGIN
                IF (NEW.cod_fornitore = NEW.cod_rifornisce) THEN RAISE
EXCEPTION 'la filiale non può rifornirsi da sola';
                END IF;

            RETURN NEW;
            END;
    $$ LANGUAGE 'plpgsql';

CREATE TRIGGER CONTROLLO_RIF
        AFTER INSERT ON RIFORNIMENTO
        FOR EACH ROW EXECUTE PROCEDURE controllo_rif();
```

/* stored procedure per il rifornimento standard con il magazzino di un determinato cd in una filiale. */

```
CREATE OR REPLACE FUNCTION rifornimento(quant INTEGER, nomecd VARCHAR, codband INTEGER, codf INTEGER) RETURNS INTEGER AS $$
		DECLARE
		Q INTEGER;
		BEGIN


		UPDATE ESPOSIZIONE
		SET quantita = quantita + quant
		WHERE cod_f = codf and nome_cd = nomecd and cod_band = codband;


		Q = (select quantita from esposizione WHERE cod_f = codf and nome_cd = nomecd and cod_band = codband );
		RETURN Q;
		END;
		$$ LANGUAGE 'plpgsql';
```

/* stored procedure per il rifornimento di cd tra due determinate filiali. E’ utile quando si ha una necessità urgente di determinati cd e il magazzino è saturo di richieste di rifornimento. Lo scambio di cd tra due filiali è certamente più veloce quindi del rifornimento magazzino-filiale. E’ necessario inoltre tenere a mente determinati vincoli imposti dal testo:

1) La filiale può rifornirsi con un altra filiale solo se non è presente un magazzino nella provincia in cui è situata, in quel caso infatti il rifornimento del cd avverrebbe comunque in tempi sufficientemente brevi

2) La filiale scelta per il rifornimento deve trovarsi all’interno della lista delle filiali di rifornimento per la filiale (cioè all’interno dell’auto associazione “rifornimento”)

3) La filiale di rifornimento scelta deve avere una sufficiente disponibilità di quel cd musicale

*/

```
CREATE OR REPLACE FUNCTION rifornimento_filiali(quant INTEGER, nomecd VARCHAR, codband INTEGER, codf INTEGER, codf2 INTEGER) RETURNS INTEGER AS $$
		DECLARE
```

```
codM INTEGER;
provM VARCHAR(60);
q2 INTEGER; /* quantita cd disponibili della filiale scelta per il rifornimento*/
q_n INTEGER; /* nuova quantita di cd disponibili per la filiale */
BEGIN

codM = (select cod_m from filiale f where f.cod_f = codf);
provM = (select provincia from magazzino m where m.cod_m =codM );

IF provM = (select provincia from filiale f where f.cod_f = codf) THEN RAISE EXCEPTION 'effettuare la procedura di rifornimento standard, il magazzino è sufficientemente vicino alla filiale per fornire il prodotto in tempi brevi';
END IF;

q2 = (select quantita from esposizione e where e.cod_f = codf2 and nome_cd = nomecd and cod_band = codband);

/* la filiale scelta per il rifornimento deve essere presente nella lista di filiali di rifornimento per quella determinata filiale (dentro l'associazione "rifornimento") .
*/

IF (select count(*) from rifornimento r where r.cod_rifornisce = codf and r.cod_fornitore = codf2) < 1 THEN RAISE EXCEPTION 'la filiale scelta di rifornimento non è presente nella lista di filiali di rifornimento disponibili';
END IF;

IF (q2 - quant) < 0 THEN RAISE EXCEPTION 'quantita di cd disponibili della filiale di rifornimento non sufficiente';
ELSE
    UPDATE ESPOSIZIONE
    SET quantita = quantita + quant
    WHERE nome_cd = nomecd and cod_band = cod_band and cod_f = codf;

    UPDATE ESPOSIZIONE
    SET quantita = quantita - quant
```

```
                    WHERE nome_cd = nomecd and cod_band = cod_band and cod_f =
codf2;

                END IF;


                q_n = (select quantita from esposizione e where nome_cd = nomecd and
cod_band = cod_band and cod_f = codf);
                RETURN q_n;
                END;
                $$ LANGUAGE 'plpgsql';
```

Query di inserimento

Di seguito sono riportate le query per l'inserimento dei dati puramente a scopo dimostrativo.

```
INSERT INTO CASA_DISCOGRAFICA VALUES('Bertelsmann Music Group', '1987');
INSERT INTO CASA_DISCOGRAFICA VALUES('EMI','1931');
INSERT INTO CASA_DISCOGRAFICA VALUES('Sony Music','1991');
INSERT INTO CASA_DISCOGRAFICA VALUES('PolyGram','1962');
INSERT INTO CASA_DISCOGRAFICA VALUES('Universal Music Group','1934');

INSERT INTO BAND VALUES('234','Beatles','2000-11-22','Bertelsmann Music Group');
INSERT INTO BAND VALUES('4254','Anthrax','2001-3-2','EMI');
INSERT INTO BAND VALUES('12','Blink 182','1987-5-25','Bertelsmann Music Group');
INSERT INTO BAND VALUES('53','Bon jovi','2005-1-11','Universal Music Group');
INSERT INTO BAND VALUES('863','Chumbawamba','2000-3-4','EMI');
INSERT INTO BAND VALUES('48','The cure','2005-5-6','Universal Music Group');
INSERT INTO BAND VALUES('3','Depeche mode','2006-6-4','Bertelsmann Music Group');
INSERT INTO BAND VALUES('93','The doors','2007-11-3','EMI');
```

```
INSERT INTO BAND VALUES('354','Eagles','1988-1-22','Bertelsmann Music Group');
INSERT INTO BAND VALUES('111','Evereything but the girl','2015-3-5','PolyGram');
INSERT INTO BAND VALUES('232','Foo fighters','2012-3-4','PolyGram');
INSERT INTO BAND VALUES('222','Genesis','2013-2-1','Bertelsmann Music Group');
INSERT INTO BAND VALUES('331','Green day','2000-11-2','Bertelsmann Music Group');
INSERT INTO BAND VALUES('89','Jamiroquai','2004-3-3','Sony Music');
```

```
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Lemon
studio','Roma','41019','Soliera','23','Modena');
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Donkey','Croce
lama','41019','Modena','43','Modena');
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Rock studio','Papa
giovanni','41019','Limidi','33','Modena');
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Ross','Carpi
ravarino','41019','Carpi','234','Modena');
INSERT INTO STUDIO_DI_REGISTRAZIONE
VALUES('Veritas','Freccia','75938','Agrigento','123','Reggio Emilia');
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Riverdale
studio','Arcobaleno','98538','Salerno','444','Bologna');
INSERT INTO STUDIO_DI_REGISTRAZIONE VALUES('Lake
studio','Mulo','41019','Campogalliano','333','Modena');
```

```
INSERT INTO CD_MUSICALE VALUES('La voce del padrone','234','Lemon
studio','7','01:00:00','6','234','Soliera');
INSERT INTO CD_MUSICALE VALUES('Una donna per amico','234','Lemon
studio','8','01:00:00','6','234','Soliera');
INSERT INTO CD_MUSICALE VALUES('Ovunque proteggi ','4254','Lemon
studio','8','01:00:00','7','435','Soliera');
INSERT INTO CD_MUSICALE VALUES('I buoni e i cattivi','4254','Lemon
studio','9','01:20:00','7','555','Soliera');
INSERT INTO CD_MUSICALE VALUES('WoW','12','Lemon
studio','20','00:40:00','7','444','Soliera');
INSERT INTO CD_MUSICALE VALUES('A sangue
freddo','12','Donkey','12','01:00:00','7','324','Modena');
INSERT INTO CD_MUSICALE
VALUES('Dreamland','53','Donkey','5','00:40:00','8','643','Modena');
```

```
INSERT INTO CD_MUSICALE VALUES('A Race with the
Devil','53','Donkey','5','00:40:00','9','325','Modena');

INSERT INTO CD_MUSICALE
VALUES('Carboni','863','Donkey','6','00:45:00','9','545','Modena');

INSERT INTO CD_MUSICALE VALUES('Darwin!','863','Rock
studio','8','00:45:00','9','342','Limidi');

INSERT INTO CD_MUSICALE VALUES('Pompa','48','Rock
studio','10','00:45:00','9','355','Limidi');

INSERT INTO CD_MUSICALE VALUES('Il tuffatore','48','Rock
studio','11','00:47:00','7','354','Limidi');

INSERT INTO CD_MUSICALE VALUES('Reset','3','Rock
studio','12','00:47:00','8','666','Limidi');

INSERT INTO CD_MUSICALE VALUES('Latin Lover','3','Rock
studio','12','00:47:00','8','454','Limidi');

INSERT INTO CD_MUSICALE VALUES('Il vile','93','Ross','13','00:47:00','7','234','Carpi');

INSERT INTO CD_MUSICALE VALUES('Kinotto','93','Ross','15','00:48:00','7','565','Carpi');

INSERT INTO CD_MUSICALE VALUES('Non e per
sempre','354','Veritas','12','01:47:00','7','575','Agrigento');

INSERT INTO CD_MUSICALE
VALUES('Rockmantico','354','Veritas','6','00:43:00','7','345','Agrigento');

INSERT INTO CD_MUSICALE VALUES('Ossigeno','111','Riverdale
studio','5','00:40:00','8','345','Salerno');

INSERT INTO CD_MUSICALE VALUES('Kanaglia','232','Riverdale
studio','6','00:57:00','8','123','Salerno');

INSERT INTO CD_MUSICALE VALUES('Fatti Sentire','222','Lake
studio','6','00:47:00','8','765','Campogalliano');

INSERT INTO CD_MUSICALE VALUES('Peter Pan','331','Lake
studio','7','00:27:00','6','1111','Campogalliano');

INSERT INTO CD_MUSICALE VALUES('Everything Is Love','89','Lake
studio','8','00:47:00','6','455','Campogalliano');


INSERT INTO GENERE_MUSICALE VALUES('Hard Rock','1980');

INSERT INTO GENERE_MUSICALE VALUES('Pop','2010');

INSERT INTO GENERE_MUSICALE VALUES('Heavy Metal','1980');

INSERT INTO GENERE_MUSICALE VALUES('Blues','1960');




INSERT INTO ADESIONE VALUES('Hard Rock','234','La voce del padrone');

INSERT INTO ADESIONE VALUES('Hard Rock','234','Una donna per amico');
```

```
INSERT INTO ADESIONE VALUES('Hard Rock','4254','Ovunque proteggi ');
INSERT INTO ADESIONE VALUES('Hard Rock','4254','I buoni e i cattivi');
INSERT INTO ADESIONE VALUES('Hard Rock','12','WoW');
INSERT INTO ADESIONE VALUES('Hard Rock','12','A sangue freddo');
INSERT INTO ADESIONE VALUES('Hard Rock','53','Dreamland');
INSERT INTO ADESIONE VALUES('Heavy Metal','53','A Race with the Devil');
INSERT INTO ADESIONE VALUES('Heavy Metal','863','Carboni');
INSERT INTO ADESIONE VALUES('Heavy Metal','863','Darwin!');
INSERT INTO ADESIONE VALUES('Heavy Metal','48','Pompa');
INSERT INTO ADESIONE VALUES('Blues','48','Il tuffatore');
INSERT INTO ADESIONE VALUES('Blues','3','Reset');
INSERT INTO ADESIONE VALUES('Blues','3','Latin Lover');
INSERT INTO ADESIONE VALUES('Blues','93','Il vile');
INSERT INTO ADESIONE VALUES('Pop','93','Kinotto');
INSERT INTO ADESIONE VALUES('Pop','354','Non e per sempre');
INSERT INTO ADESIONE VALUES('Pop','354','Rockmantico');
INSERT INTO ADESIONE VALUES('Pop','111','Ossigeno');
INSERT INTO ADESIONE VALUES('Pop','232','Kanaglia');
INSERT INTO ADESIONE VALUES('Pop','222','Fatti Sentire');
INSERT INTO ADESIONE VALUES('Hard Rock','331','Peter Pan');
INSERT INTO ADESIONE VALUES('Hard Rock','89','Everything Is Love');


INSERT INTO MAGAZZINO VALUES('1','Cascata','41019','Modena','12','Modena');
INSERT INTO MAGAZZINO VALUES('2','Roma','34244','Salerno','3','Salerno');
INSERT INTO MAGAZZINO VALUES('3','Palladio','75938','Rolo','23','Palermo');


INSERT INTO DIRIGENTE
VALUES('ABC','Daniele','Bruco','70','3557349485','Roma','41019','Soliera','32','Modena
','15');
INSERT INTO DIRIGENTE
VALUES('EFG','Matteo','Aguzzo','75','3348385839','Cielo','23542','Milano','43','Milano','
18');
INSERT INTO DIRIGENTE
VALUES('HIJ','Luigi','Sedo','75','3285434567','Sole','32454','Arezzo','23','Arezzo','20');
```

```
INSERT INTO FILIALE
VALUES('10','0','059635483','Roma','41019','Modena','5','Modena','1');

INSERT INTO FILIALE
VALUES('11','0','059837493','Sole','23424','Palermo','33','Palermo','2');

INSERT INTO FILIALE
VALUES('12','0','059281293','Calo','75938','Rolo','56','Palermo','2');



INSERT INTO DIRIGE VALUES('10','ABC');

INSERT INTO DIRIGE VALUES('11','EFG');

INSERT INTO DIRIGE VALUES('12','HIJ');


INSERT INTO DIPENDENTE
VALUES('QWE','Paolo','Tarozzi','25','123456788','croce','41019','Soliera','2','Modena','1
0');

INSERT INTO DIPENDENTE VALUES('RTY','Enrico','Sedoni','35','384293920','Croce
lama','41019','Soliera','257','Modena','10');

INSERT INTO DIPENDENTE
VALUES('UUI','Roberto','Pulizo','20','3456086234','Reti','23454','Roma','32','Roma','11');

INSERT INTO DIPENDENTE
VALUES('IOP','Giulio','Sternieri','25','098765432','Viazzolo','41019','Soliera','33','Moden
a','11');

INSERT INTO DIPENDENTE VALUES('ASD','Isabella','Marchi','33','3928304933','Luigi
galvani','75938','Correggio','12','Reggio Emilia','12');

INSERT INTO DIPENDENTE VALUES('FGH','Elisa','Carducci','31','3928342333','Paolo
Sarti','75938','Correggio','32','Reggio Emilia','12');



INSERT INTO RIFORNIMENTO VALUES('11','12');

INSERT INTO RIFORNIMENTO VALUES('12','11');



INSERT INTO EVENTO VALUES('Rock in modena','2010-02-05','Modena','10');

INSERT INTO EVENTO VALUES('Blues insieme','2011-03-02','Modena','10');

INSERT INTO EVENTO VALUES('Metal in palermo','2010-05-12','Palermo','11');

INSERT INTO EVENTO VALUES('Music festival','2012-02-22','Salerno','12');

INSERT INTO EVENTO VALUES('Fosh Fest','2013-01-12','Modena','10');
```

```
INSERT INTO ESPOSIZIONE VALUES('La voce del padrone','234','10','15');
INSERT INTO ESPOSIZIONE VALUES('Una donna per amico','234','10','15');
INSERT INTO ESPOSIZIONE VALUES('Ovunque proteggi ','4254','10','15');
INSERT INTO ESPOSIZIONE VALUES('I buoni e i cattivi','4254','10','15');
INSERT INTO ESPOSIZIONE VALUES('WoW','12','10','15');
INSERT INTO ESPOSIZIONE VALUES('Dreamland','53','10','15');
INSERT INTO ESPOSIZIONE VALUES('Carboni','863','10','15');
INSERT INTO ESPOSIZIONE VALUES('Darwin!','863','10','15');
INSERT INTO ESPOSIZIONE VALUES('Reset','3','10','15');
INSERT INTO ESPOSIZIONE VALUES('Il vile','93','10','15');
INSERT INTO ESPOSIZIONE VALUES('Kinotto','93','10','15');
INSERT INTO ESPOSIZIONE VALUES('Rockmantico','354','10','15');
INSERT INTO ESPOSIZIONE VALUES('Peter Pan','331','10','15');

INSERT INTO ESPOSIZIONE VALUES('La voce del padrone','234','11','15');
INSERT INTO ESPOSIZIONE VALUES('Una donna per amico','234','11','15');
INSERT INTO ESPOSIZIONE VALUES('Ovunque proteggi ','4254','11','15');
INSERT INTO ESPOSIZIONE VALUES('WoW','12','11','15');
INSERT INTO ESPOSIZIONE VALUES('Dreamland','53','11','15');
INSERT INTO ESPOSIZIONE VALUES('Carboni','863','11','15');
INSERT INTO ESPOSIZIONE VALUES('Darwin!','863','11','15');
INSERT INTO ESPOSIZIONE VALUES('Reset','3','11','15');
INSERT INTO ESPOSIZIONE VALUES('Kinotto','93','11','15');
INSERT INTO ESPOSIZIONE VALUES('Rockmantico','354','11','15');
INSERT INTO ESPOSIZIONE VALUES('Peter Pan','331','11','15');

INSERT INTO ESPOSIZIONE VALUES('La voce del padrone','234','12','15');
INSERT INTO ESPOSIZIONE VALUES('Una donna per amico','234','12','15');
INSERT INTO ESPOSIZIONE VALUES('I buoni e i cattivi','4254','12','15');
INSERT INTO ESPOSIZIONE VALUES('WoW','12','12','15');
INSERT INTO ESPOSIZIONE VALUES('Dreamland','53','12','15');
```

```
INSERT INTO ESPOSIZIONE VALUES('Carboni','863','12','15');
INSERT INTO ESPOSIZIONE VALUES('Everything Is Love','89','12','15');
INSERT INTO ESPOSIZIONE VALUES('Kinotto','93','12','15');
INSERT INTO ESPOSIZIONE VALUES('Rockmantico','354','12','15');
INSERT INTO ESPOSIZIONE VALUES('Peter Pan','331','12','15');


INSERT INTO CLIENTE VALUES('ABCD','Luigi','Sanfilippo','09867764','0');
INSERT INTO CLIENTE VALUES('BCDE','Rulo','Pastulu','5372934','0');
INSERT INTO CLIENTE VALUES('FGHI','Francesco','Sanfilippo','5454555','0');
INSERT INTO CLIENTE VALUES('GFDT','Mario','Bianchi','67523473','0');
INSERT INTO CLIENTE VALUES('JEMCC','Paolo','Rossi','848473649','0');
INSERT INTO CLIENTE VALUES('SKEKS','Oca','Pule','73736393','0');
INSERT INTO CLIENTE VALUES('ETNSM','Lorella','Casella','09867334','0');
INSERT INTO CLIENTE VALUES('OWNCJC','Matteo','Arizona','33434223','0');
INSERT INTO CLIENTE VALUES('KMDKNE','SIngh','India','56234223','0');
INSERT INTO CLIENTE VALUES('GXNWLD','Enrico','Lupo','1233445','0');


INSERT INTO PRESTITO VALUES('2010-05-06','ABCD','Peter Pan','331','5','12');
INSERT INTO PRESTITO VALUES('2010-05-06','ABCD','Rockmantico','354','5','12');
INSERT INTO PRESTITO VALUES('2011-03-06','BCDE','Peter Pan','331','5','12');
INSERT INTO PRESTITO VALUES('2013-08-11','BCDE','I buoni e i cattivi','4254','5','12');
INSERT INTO PRESTITO VALUES('2010-05-06','FGHI','La voce del padrone','234','5','11');
INSERT INTO PRESTITO VALUES('2010-05-06','FGHI','Reset','3','5','11');

INSERT INTO BUONO_SCONTO VALUES('1234','0.10','true');
INSERT INTO BUONO_SCONTO VALUES('456','0.10','true');
INSERT INTO BUONO_SCONTO VALUES('789','0.15','true');
INSERT INTO BUONO_SCONTO VALUES('3421','0.15','true');
INSERT INTO BUONO_SCONTO VALUES('3245','0.20','true');
INSERT INTO BUONO_SCONTO VALUES('12345','0.20','true');


INSERT INTO CARRELLO VALUES('1','GFDT','2009-08-07','0','10');
```

```
INSERT INTO CARRELLO VALUES('2','JEMCC','2002-03-07','0','10');
INSERT INTO CARRELLO VALUES('3','SKEKS','2008-03-11','0','11');
INSERT INTO CARRELLO VALUES('4','ETNSM','2008-06-22','0','10');
INSERT INTO CARRELLO VALUES('5','OWNCJC','2008-06-07','0','10');
INSERT INTO CARRELLO VALUES('6','KMDKNE','2005-12-07','0','11');
INSERT INTO CARRELLO VALUES('1','GXNWLD','2008-03-17','0','12');
INSERT INTO CARRELLO VALUES('2','ABCD','2009-05-27','0','12');
INSERT INTO CARRELLO VALUES('3','BCDE','2008-06-07','0','12');
INSERT INTO CARRELLO VALUES('4','KMDKNE','2001-06-07','0','12');



INSERT INTO FA_PARTE VALUES('1','GFDT','Peter Pan','331','3');
INSERT INTO FA_PARTE VALUES('1','GFDT','WoW','12','3');

INSERT INTO FA_PARTE VALUES('2','JEMCC','Peter Pan','331','2');
INSERT INTO FA_PARTE VALUES('2','JEMCC','WoW','12','2');

INSERT INTO FA_PARTE VALUES('3','SKEKS','La voce del padrone','234','1');
INSERT INTO FA_PARTE VALUES('3','SKEKS','Una donna per amico','234','1');

INSERT INTO FA_PARTE VALUES('4','ETNSM','Dreamland','53','4');
INSERT INTO FA_PARTE VALUES('4','ETNSM','Peter Pan','331','4');

INSERT INTO FA_PARTE VALUES('5','OWNCJC','La voce del padrone','234','1');
INSERT INTO FA_PARTE VALUES('5','OWNCJC','WoW','12','1');

INSERT INTO FA_PARTE VALUES('6','KMDKNE','Peter Pan','331','1');
INSERT INTO FA_PARTE VALUES('6','KMDKNE','WoW','12','1');

INSERT INTO FA_PARTE VALUES('1','GXNWLD','I buoni e i cattivi','4254','2');

INSERT INTO FA_PARTE VALUES('2','ABCD','Peter Pan','331','1');

INSERT INTO FA_PARTE VALUES('3','BCDE','WoW','12','2');
```

INSERT INTO FA_PARTE VALUES('4','KMDKNE','La voce del padrone','234','1');

INSERT INTO SOGGETTO VALUES('1234','1','GFDT');
INSERT INTO SOGGETTO VALUES('456','2','JEMCC');
INSERT INTO SOGGETTO VALUES('789','3','BCDE');
INSERT INTO SOGGETTO VALUES('12345','1','GXNWLD');

Query di interrogazione

1) Selezionare la filiale con il maggior numero di dipendenti

SELECT cod_f

FROM dipendente

GROUP BY cod_f

HAVING count(*) >= ALL(SELECT count(*) FROM dipendente GROUP BY cod_f)

2) Selezionare il nome e il prezzo dei cd musicali delle band che fanno parte della casa discografica “EMI” e che sono stati venduti ad almeno un cliente

SELECT distinct cd.nome, cd.prezzo_cad

FROM cd_musicale cd, band b, fa_parte f

WHERE cd.cod_band = b.cod_band and nome_c = 'EMI' and f.nome_cd = cd.nome and f.cod_band = cd.cod_band

3) Selezionare il nome del cd e il rispettivo nome della band con il maggior numero di vendite

SELECT b.nome, f.nome_cd

FROM fa_parte f, band b

```
WHERE f.cod_band = b.cod_band
GROUP BY f.nome_cd, b.nome
HAVING count(*) >= ALL(SELECT count(*)
                       FROM fa_parte f2, band b2
                       WHERE f2.cod_band = b2.cod_band
                        GROUP BY f2.nome_cd, b2.nome)
```

4) Selezionare i generi musicali con anno di popolarità tra il 1980 e il 2018

```
SELECT nome
FROM genere_musicale
WHERE popolarita BETWEEN '1980' and '2018'
```

5) Selezionare le case discografiche per le quali tutte le band che seguono sono state fondate dopo il ‘2000-01-01’

```
SELECT nome
FROM casa_discografica c
WHERE NOT EXISTS(SELECT *
                 FROM band b
                 WHERE b.nome_c = c.nome
                  and  b.data_fondazione <= '2000-01-01' )
```

6) Selezionare il nome dei cd e il rispettivo nome della band dei cd che non sono mai stati presi in prestito

```
SELECT cd.nome, b.nome
FROM  cd_musicale cd, band b
WHERE b.cod_band = cd.cod_band and NOT EXISTS (SELECT *
                                               FROM prestito p
```

WHERE p.nome_cd = cd.nome and p.cod_band = cd.cod_band)

7) Selezionare per ogni filiale il numero totale di cd venduti

SELECT cod_f, sum(qta_copie)

FROM carrello c, fa_parte f

WHERE c.cf = f.cf and c.num = f.num

GROUP BY cod_f

8) Selezionare il nome dei cd che sono stati sia comprati che presi in prestito almeno una volta

SELECT cd.nome

FROM cd_musicale cd

WHERE EXISTS(SELECT * FROM prestito p

WHERE p.nome_cd = cd.nome and p.cod_band = cd.cod_band and EXISTS(SELECT *

FROM fa_parte f
WHERE f.nome_cd = cd.nome and f.cod_band = cd.cod_band))

9) Selezionare il nome dei cd musicali prodotti da studi di registazione della provincia di Modena e con costo di registrazione superiore a 300 euro

SELECT cd.nome, cd.nome_studio

FROM cd_musicale cd, studio_di_registrazione s

WHERE cd.nome_studio = s.nome and s.provincia = 'Modena' and cd.costo_reg > '300'

10) Selezionare i carrelli sottoposti a buoni sconto superiori del 10% (0,1)

SELECT c.num, c.cf

FROM carrello c, soggetto s, buono_sconto b

WHERE c.num = s.num and c.cf = s.cf and b.cod_b = s.cod_b and b.sconto > '0.10'

Query di modifica

Modificare il costo orario del dipendente con codice fiscale “RTY” a 40 euro l’ora

```
UPDATE DIPENDENTE
SET costo_orario = '40'
WHERE cf = 'RTY'
```

Modificare il numero di telefono in “059545434” del cliente con cf = “GFDT”

```
UPDATE CLIENTE
SET telefono = '0595454340'
WHERE cf = 'GFDT'
```

Modifica il prezzo_cad a 20 euro del cd con titolo “Peter Pan” della band con codice “331”

```
UPDATE CD_MUSICALE
SET prezzo_cad = '20'
WHERE nome = 'Peter Pan' and cod_band = '331'
```

Query di eliminazione

Eliminare il buono sconto con codice “3421”

```
DELETE FROM buono_sconto
WHERE cod_b = '3421'
```

Eliminare l’evento con nome “Metal in palermo”

```
DELETE FROM EVENTO
WHERE nome = 'Metal in palermo'
```

Eliminare il dipendente con codice fiscale “RTY”

DELETE FROM DIPENDENTE

WHERE cf = 'RTY'

Progettazione fisica

Di seguito viene presentato un esempio di progettazione fisica:

Query analizzata:

SELECT nome

FROM cd_musicale

WHERE prezzo_cad = '5' and durata = '00:40:00'

Dati:

NT = 10000

NB = 1000

Indici:

prezzo_cad, indice clustered. NKp = 100 e NFp = 120

durata, indice unclustered. NKd = 50 e NFd = 130

Fp = 1/NKp = 1/100

Fd = 1/NKd = 1/50

E = ⌈Fp*Fd*NT ⌉ = 2

Costi:

Cseq = 1000

Cp = ⌈Fp*NFp⌉+ ⌈Fp*NB⌉ = ⌈(1/100)*120⌉+ ⌈(1/100)*1000⌉ = 2+10 = 12

Cd = ⌈Fd*NFd⌉+ ⌈Fd*NT⌉= ⌈(1/50)*130⌉+ ⌈(1/50)*10000⌉ = 3+200 = 203

Risultato:

Per velocizzare la query conviene costruire l’indice sul prezzo_cad