Relazione progetto di Laboratorio

Corso di Basi di Dati

Università degli studi di Udine, A.A. 2024-2025

Daniele De Martin
Massimiliano Di Marco
Enrico Peressin
Michele Vecchiato

PROGETTAZIONE E IMPLEMENTAZIONE
DI UNA BASE DI DATI PER LA GESTIONE DI UNA BANCA

Indice

1. Raccolta e analisi dei requisiti

1.1. Richiesta Originale

> Si vuole progettare una base di dati di supporto ad alcune delle attività di una banca.
> La banca è organizzata in un certo numero di filiali. Ogni filiale si trova in una determinata città ed è identificata univocamente da un nome (si noti che in una città vi possono essere più filiali). La banca tiene traccia dei risultati (attivi) conseguiti da ciascuna filiale.
> Ai clienti della banca è assegnato un codice che li identifica univocamente. La banca tiene traccia del nome del cliente e della sua residenza. I clienti possono possedere uno o più conti e possono chiedere prestiti. A un cliente può essere associato un particolare dipendente della banca, che segue personalmente tutte le pratiche del cliente (si tenga presente che non tutti i clienti godono di tale privilegio e che ad un dipendente della banca possono essere associati zero, uno o più clienti).
> I dipendenti della banca sono identificati da un codice. La banca memorizza nome e recapito telefonico di ogni dipendente, il nome delle persone a suo carico e il codice dell'eventuale capo. La banca tiene inoltre traccia della data di assunzione di ciascun dipendente e dell'anzianità aziendale di ciascun dipendente (da quanto tempo tale dipendente lavora per la banca).
> La banca offre due tipi di conto: conto corrente (con la possibilità di emettere assegni, ma senza interessi) e conto di risparmio (senza la possibilità di emettere assegni, ma con interessi). Un conto può essere posseduto congiuntamente da più clienti e un cliente può possedere più conti. Ogni conto è caratterizzato da un numero che lo identifica univocamente. Per ogni conto, la banca tiene traccia del saldo corrente e della data dell'ultima operazione eseguita da ciascuno dei possessori (un'operazione può essere eseguita congiuntamente da più possessori). Ogni conto di risparmio è caratterizzado da un tasso di interesse, mentre ogni conto corrente è caratterizzato da uno scoperto accordato al cliente.
> Un prestito (ad esempio, un mutuo) viene emesso da una specifica filiale e può essere attribuito a uno o più clienti congiuntamente. Ogni prestito è identificato univocamente da un codice numerico. Ogni prestito è caratterizzato da un ammontare e da un insieme di rate per la restutuzione del prestito. Ogni rata di un dato prestito è contraddistinta da un numero d'ordine (prima rata, seconda rata...). Di ogni rata vengono memorizzati anche la data e l'ammontare.

1.2. Analisi dei Requisiti

1.2.1. Assunzioni

Al fine di proseguire con la progettazione concettuale, sono state effettuate le seguenti assunzioni:

- Gli **attivi** sono la somma della liquidità dei conti meno la somma dei prestiti erogati. Sono relativi alla singola filiale.
- Un **cliente** può avere conti in filiali diverse e ogni conto è associato ad una singola filiale.
- I **prestiti** sono legati al conto, non al cliente.

- Un **dipendente** non può gestire se stesso.
- Un **dipendente** lavora in una sola filiale con la possibilità di gestire clienti al di fuori della propria filiale.
- Il **capo** di un dipendente è l'unico responsabile della filiale in cui il dipendente lavora.
- Nei **conti cointestati** i clienti non possono essere seguiti da dipendenti (gestori) diversi.
- In caso di **ri-assunzione** di un dipendente, si tiene conto solo dell'ultima assunzione per il calcolo dell'anzianità.
- Tutte le **rate** di un determinato prestito hanno lo stesso ammontare e devono essere pagate in ordine.

1.2.2. Glossario

Per chiarire il significato e le relazioni dei termini chiave definite nei requisiti viene fornito un glossario esplicativo:

Termine	Descrizione
Filiale	Unità operativa della banca situata in una determinata città. È gestita da un unico capo.
Cliente	Persona fisica con almeno un conto aperto nella banca.
Conto	Servizio di gestione del denaro che permette diverse operazioni. Può essere esclusivamente di tipo corrente o di risparmio.
Conto Corrente	Tipo di conto caratterizzato da uno scoperto.
Conto di risparmio	Tipo di conto caratterizzato da un tasso di interesse.
Dipendente	Persona fisica che lavora in una certa filiale della banca.
Gestore	Dipendente che prende in carico le pratiche di uno o più clienti.
Operazione	Transazioni bancarie effettuate su un conto da uno o più intestatari. Sono operazioni l'apertura di un conto, il prelievo, il pagamento elettronico (bancomat) e il versamento.
Prestito	Somma di denaro concessa alla banca a un cliente.

Tabella 1: *Glossario dei termini chiave*

2. Progettazione Concettuale

2.1. Costruzione dello schema Entità Relazione

L'analisi dei requisiti ha portato alla definizione di un insieme di entità e relazioni che costituiranno il modello concettuale della base di dati.

- L'entità ***FILIALE*** rappresenta un'unità operativa della banca situata in una determinata città. La chiave primaria è il *Nome*, mentre gli altri attributi sono *Città* e *Indirizzo*. Inoltre, per ogni filiale è presente l'attributo derivato *Attivi*, che rappresenta l'ammontare totale della liquidità della filiale e viene calcolato sulla base dei conti, prestiti e rate ad esso associati.

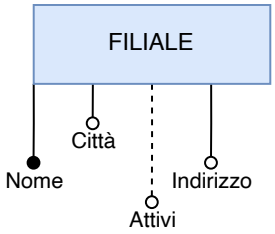

Figura 1: *Entità FILIALE*

- L'entità ***CLIENTE*** rappresenta una persona fisica che ha aperto nella banca almeno un conto. Essa è caratterizzata da un *codice univoco* assegnato dalla banca ad ogni cliente e dal *codice fiscale*, entrambi questi attributi possono essere due chiavi primarie differenti in quanto sono univoche per ogni cliente. Gli altri attributi servono per tenere traccia dell'anagrafica del cliente: *Nome*, *Cognome*, *Telefono*, *Data di nascita* e *Residenza*.

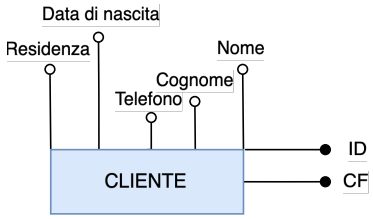

Figura 2: *Entità CLIENTE*

- L'entità ***DIPENDENTE*** è caratterizzata da un codice univoco *ID* che funge da chiave primaria. *Nome*, *Cognome*, *Numero di telefono*, *Data di assunzione* sono gli altri attributi che la descrivono. È stato scelto di tenere traccia dell'anzianità aziendale sulla base della data di assunzione.
  La qualifica di capo viene descritta da una specializzazione parziale di *DIPENDENTE*, chiamata *CAPO*.

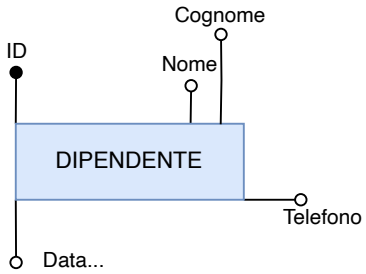

Figura 3: *Entità DIPENDENTE*

- L'entità ***CAPO*** rappresenta il capo di una filiale. Essendo una specializzazione dell'entità *DIPENDENTE*, eredita tutti gli attributi di quest'ultima. Un capo è univoco per ogni filiale.

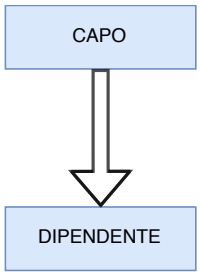

Figura 4: *Entità CAPO*

- L'entità ***CONTO*** serve per identificare un servizio della banca messo a disposizione per il cliente. Ogni entità viene identificata univocamente da un attributo *IBAN*, un attributo *Saldo* tiene traccia dell'ammontare di denaro presente sul conto. La banca inoltre mette a disposizione due tipi di conto, quindi l'entità Conto è stata specializzata in due sottoentità: *CONTO CORRENTE* e *CONTO DI RISPARMIO*. La specializzazione è totale e disgiunta.
  - L'entità ***CONTO CORRENTE*** è una specializzazione dell'entità *CONTO* pertanto ne eredita tutti gli attributi e tutte le relazioni, la chiave primaria è quindi quella dell'entità *CONTO*. L'attributo che lo caratterizza è *Scoperto* che indica quanto la banca può concedere di debito nei confronti del cliente.
  - L'entità ***CONTO DI RISPARMIO*** è una specializzazione dell'entità *CONTO* pertanto ne eredita tutti gli attributi e tutte le relazioni, la chiave primaria è quindi quella dell'entità di *CONTO*. L'attributo che lo caratterizza è *Tasso d'interesse* che indica il valore di rendita mensile del conto.

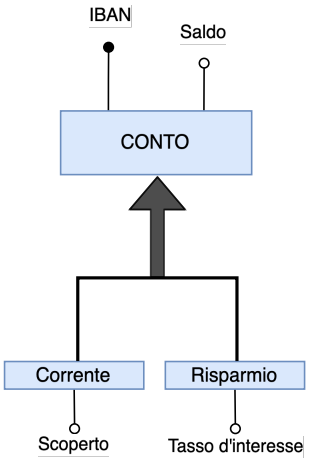

Figura 5: *Entità CONTO*

- L'entità ***PRESTITO*** costituisce il servizio creditizio della banca. Essa è caratterizzata da un codice univoco che funge da chiave primaria. L'attributo *Ammontare* fornisce l'informazione relativa alla somma di denaro prestata, mentre l'attributo *Inizio* registra la data in cui il prestito ha avuto origine. L'attributo *Somma rate* è un attributo derivato, che tiene traccia dell'importo saldato dal cliente. L'attributo *Mensilità* indica il numero di rate complessive del prestito.

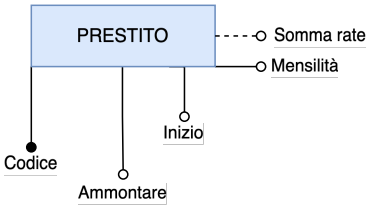

Figura 6: *Entità PRESTITO*

- L'entità ***RATA*** è una entità debole ed ha il compito di rappresentare ogni singolo pagamento periodico associato a un determinato prestito. L'identificazione univoca di ciascuna rata è garantita da una chiave primaria composta, costituita dal suo numero (indicante la "posizione" della rata nella sequenza dei pagamenti) e dalla chiave esterna che fa riferimento all'entità *PRESTITO*. Tra gli attributi figurano inoltre la *Data scadenza*, ossia il giorno entro cui la rata deve essere corrisposta, e la *Data pagamento*, che riporta il momento in cui il versamento è stato effettuato. Infine, l'attributo *Ammontare* specifica l'importo dovuto per quella singola rata.

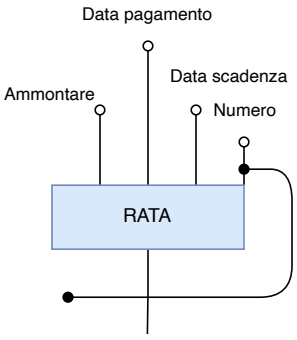

Figura 7: *Entità RATA*

- La relazione ***È CAPO*** collega l'entità *CAPO* con l'entità *FILIALE*, definendo il legame tra il capo di una filiale e la filiale stessa. La cardinalità di (1,1) tra la relazione e l'entità *FILIALE* indica che ogni filiale ha un solo capo, mentre la cardinalità di (0,1) tra la relazione e l'entità *CAPO* indica che un dipendente può essere al più capo di una sola filiale.

Figura 8: *Relazione è capo*

- La relazione ***LAVORA*** collega l'entità *DIPENDENTE* con l'entità *FILIALE*. La cardinalità di (1,1) tra la relazione e l'entità *DIPENDENTE* indica che ogni dipendente lavora in una e in una sola filiale, mentre la cardinalità di (1,N) tra la relazione e l'entità *FILIALE* indica che in una filiale lavorano uno o più dipendenti.

Figura 9: *Relazione lavora*

- La relazione ***DI*** collega l'entità *DIPENDENTE* con l'entità *CAPO*. La cardinalità di (1,N) tra la relazione e l'entità *CAPO* indica che un capo dirige uno o più dipendenti, mentre la cardinalità di (1,1) tra la relazione e l'entità *DIPENDENTE* indica che un dipendente ha uno e un solo capo.

Figura 10: *Relazione di*

- La relazione ***È COMPOSTO*** collega l'entità *PRESTITO* con l'entità *RATA*, dando forma al legame logico tra un finanziamento e i singoli pagamenti previsti per il suo rimborso. Dal lato di *RATA*, la cardinalità è di (1,1), poiché ogni rata è necessariamente associata ad uno e un solo prestito essendo *RATA* un'entità debole. Dal lato di Prestito, invece, la cardinalità è di (1,N), poiché un singolo prestito può essere suddiviso in una o più rate.

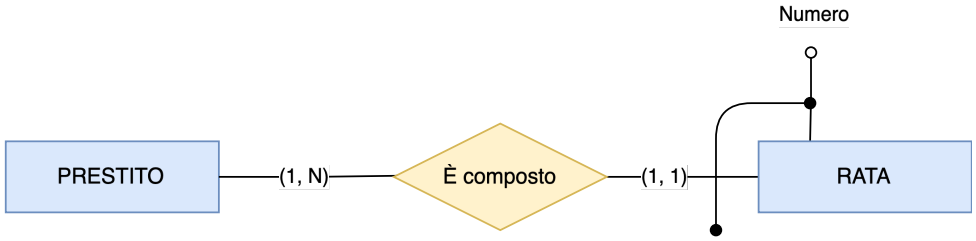

Figura 11: *Relazione è composto*

- La relazione ***È ASSOCIATO*** collega l'entità Conto con l'entità *PRESTITO*, definendo il legame tra esso e il conto bancario a cui è associato. Dal lato di *PRESTITO*, la cardinalità è (1,1), poiché ogni prestito deve fare riferimento a un solo conto bancario. Dal lato di *CONTO* la cardinalità è (0,N), infatti un conto non necessariamente ha prestiti associati.

Figura 12: *Relazione è associato*

- La relazione ***POSSIEDE*** collega le entità *CLIENTE* e *CONTO*. Un cliente deve possedere almeno un conto e più clienti possono possedere lo stesso conto (caso di conto cointestato), da cui deriva la cardinalità (1,N) della relazione sul lato di *CLIENTE*. D'altro canto un *CONTO* deve essere posseduto da almeno un cliente e più conti possono fare riferimento allo stesso cliente (caso in cui uno stesso cliente ha aperto più conti con la banca), da cui deriva la cardinalità (1,N) della relazione sul lato di *CONTO*.
  Gli attributi *Operazione* e *Data* sulla relazione indicano l'ultima operazione svolta e la data in cui è stata effettuata. Nel caso di operazione congiunta di più clienti gli attributi *Operazione/Data* vengono aggiornati per entrambi.

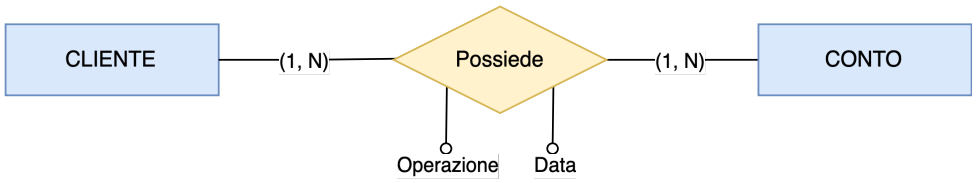

Figura 13: *Relazione possiede*

- La relazione ***GESTISCE*** collega *DIPENDENTE* e *CLIENTE*. Un sottoinsieme dei dipendenti può seguire le pratiche di un certo numero di clienti della banca, da cui ne deriva la cardinalità (0,N) della relazione sul lato di *DIPENDENTE*. D'altro canto un *CLIENTE* può avere al più un solo gestore che segue le sue attività nella banca, da cui ne deriva la cardinalità (0,1) della relazione sul lato di *CLIENTE*.

Figura 14: *Relazione gestisce*

- La relazione ***CONTIENE*** collega *FILIALE* a *CONTO* in quanto ogni *CONTO* deve fare riferimento ad una e una sola *FILIALE*. Una filiale può contenere uno o più conti (anche zero se la filiale è appena stata aperta), da cui ne deriva la cardinalità (0,N) della relazione sul lato di filiale. D'altro canto un *CONTO* deve essere associato ad una e una sola *FILIALE*, da cui ne deriva la cardinalità (1,1) della relazione sul lato di *CONTO*.

Figura 15: *Relazione contiene*

2.2. Scelte particolari

- La specializzazione non totale *CAPO - DIPENDENTE* ci permette di inserire la molteplicità (1,1) nella relazione *È CAPO* e di non dover tenere la molteplicità (0,1) nel caso in cui *DIPENDENTE* non avesse avuto la specializzazione. Favorisce inoltre una maggiore chiarezza nella relazione *DI*.
- La specializzazione totale di *CONTO* è dovuta alla presenza dei diversi attributi che caratterizzazno le due specializzazioni.
- La scelta di assegnare il ruolo di entità a *RATA* è dovuta alla numerosità degli attributi e alla gestione dell'ammontare dei prestiti. Avendo un numero seriale non è univoco, è necessario che una parte della chiave sia il codice del prestito.

2.3. Schema Concettuale

Dopo le analisi fatte, lo schema concettuale nel modello Entità Relazione è il seguente:

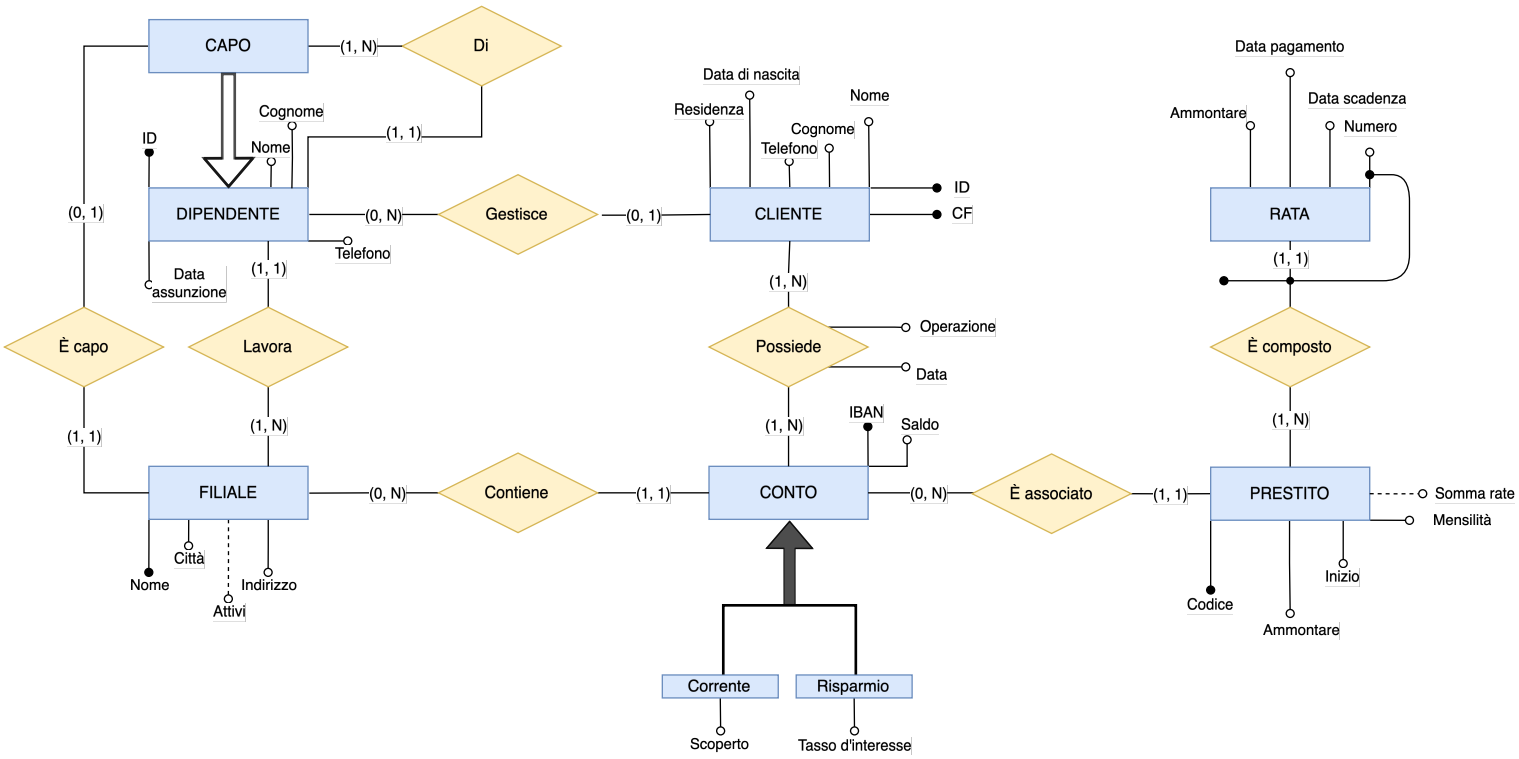

Figura 16: *Schema concettuale nel modello Entità Relazione*

2.4. Analisi dei cicli

2.4.1. Ciclo *DIPENDENTE - FLIALE - CAPO*:

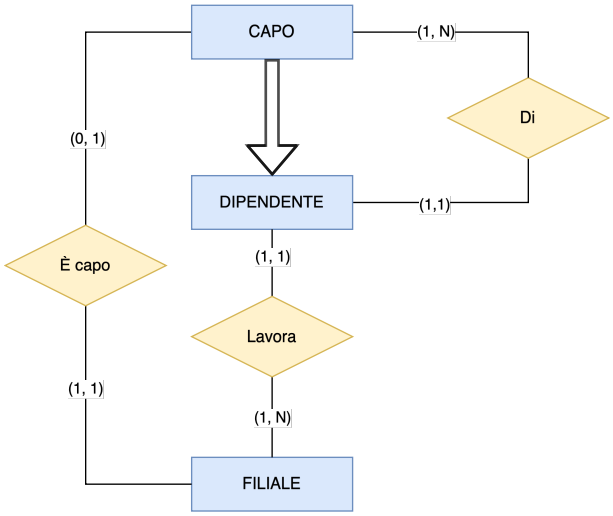

Figura 17: *Ciclo DIPENDENTE - FILIALE - CAPO*

Questo ciclo è problematico in quanto potrebbe accadere che il capo di una filiale non lavori presso la filiale di cui è responsabile. È necessario imporre dei vincoli di integrità per evitare che ciò accada.

2.4.2. Ciclo *DIPENDENTE - CLIENTE - CONTO - FILIALE*:

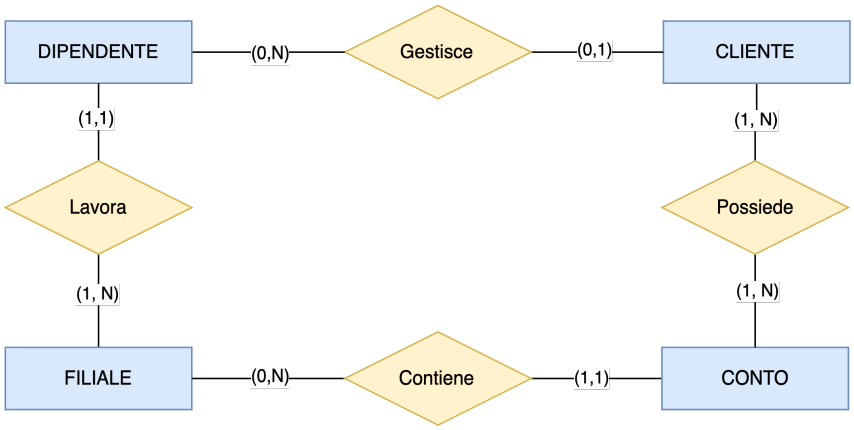

Figura 18: *Ciclo DIPENDENTE - CLIENTE - CONTO - FILIALE*

Questo ciclo non genera problemi di inconsistenza, in quanto a un cliente è permesso avere un gestore che lavora presso una certa filiale e avere più conti aperti in filiali diverse.

2.5. Vincoli d'integrità

Alcuni vincoli non possono essere catturati tramite il modello ER, vengono riportati di seguito e saranno tenuti in considerazione nella sezione di implementazione fisica:

- Il capo di una filiale deve lavorare nella filiale in cui è responsabile.
- Due clienti con gestore differente non possono avere un conto condiviso.
- Un dipendente non può gestire se stesso.
- Le rate vanno pagate in ordine cronologico.
- La somma dell'importo delle rate deve corrispondere all'ammontare del prestito.

3. Progettazione Logica

Nel processo di ottimizzazione delle prestazioni, nell'analisi delle ridondanze e nella semplificazione dello schema ER concettuale in vista della sua ristrutturazione, sono stati presi in considerazione volumi di dati stimati sulla base di una banca reale di riferimento, Intesa Sanpaolo S.p.A

3.1. Tabella dei volumi

Nome	Costrutto	Volume
Cliente	Entità	15.000.000
Conto	Entità	12.000.000
Conto Corrente	Entità	10.000.000
Conto Risparmio	Entità	2.000.000
Dipendente	Entità	100.000
Filiale	Entità	3.000
Prestito	Entità	7.000.000
Contiene	Relazione	12.000.000
Di	Relazione	100.000
È associato	Relazione	7.000.000
È capo	Relazione	3.000
È composto	Relazione	7.000.000
Gestisce	Relazione	10.000.000
Lavora	Relazione	100.000
Possiede	Relazione	19.000.000

Tabella 2: *Tabella dei volumi*

3.1.1. Considerazioni

Il numero di clienti, conti, dipendenti e filiali è stato stimato sulla base di dati reali dell' Intesa Sanpaolo. Abbiamo ipotizzato un numero di prestiti sulla base di una proporzione realistica rispetto ai conti. Per distinguere tra conti correnti e conti di risparmio, abbiamo utilizzato la percentuale media nazionale, applicandola al numero totale di conti. I volumi per le relazioni sono state calcolate tenendo conto delle cardinalità e della natura dei legami tra le entità coinvolte, in modo da mantenere coerenza con il modello concettuale.

3.2. Analisi delle ridondanze

3.2.1. Studio dell'attributo derivato *Attivi* di *FILIALE*

Il primo blocco di operazioni coinvolge l'attributo derivato *Attivi* che produce una ridondanza ed è derivabile da altre entità, nel nostro caso da *CONTO, PRESTITO* e *RATA*. Ipotizziamo delle operazioni e le loro relative frequenze che vanno a coinvolgere questo attributo e osserviamo se è conveniente eliminarlo o mantenerlo.

Operazione 1:

Interrogazione per leggere il valore *attivi* di ogni filiale con frequenza di una volta al giorno

Con attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Filiale	Entità	3000	Lettura

Tabella 3: *Operazione 1*

$$\text{op1: } (1 \text{ lettura}) \cdot 3.000$$

Per leggere il valore *attivi* di ogni filiale, è necessario eseguire una lettura della tabella *FILIALE* e leggere l'attributo derivato *attivi*.

Senza attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Filiale	Entità	3000	Lettura
Contiene	Relazione	12.000.000	Lettura
Conto	Entità	12.000.000	Lettura
È associato	Relazione	7.000.000	Lettura
Prestito	Entità	7.000.000	Lettura

Tabella 4: *Operazione 1*

$$\text{op1: } ((3 \text{ letture } \{\text{Filiale, contiene, conto}\} \cdot 4000) + (2 \text{ letture } \{\text{è associato}, \text{Prestito}\} \cdot 2333)) \cdot 3000$$

Senza l'attributo *attivi*, per calcolare gli *attivi* di ogni filiale vengono lette le 3.000 righe della tabella *FILIALE*. Poi, per ogni filiale, si risale ai conti che possiede: in media sono circa 4.000. Vengono poi effettuati altri 4.000 accessi alla tabella Conto per ottenere i saldi.

Lo stesso vale per i prestiti: per ogni filiale si leggono in media 4.000 righe nella tabella Associato e poi si accede a Prestito per recuperarne l'importo.

In totale quindi, come si vede dalla tabella, bisognerà leggere interamente le relazioni *contiene*, *è associato* e tutte le entità Conto e Prestito.

Operazione 2

Inserimento di un conto nella base di dati con frequenza 150 volte al giorno

Con attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Conto	Entità	150	Scrittura
Contiene	Relazione	150	Scrittura
Possiede	Relazione	150	Scrittura
Filiale	Entità	150	Scrittura
Filiale	Entità	150	Lettura

Tabella 5: *Operazione 2*

$$\text{op2: } (4 \text{ scrittura}\{\text{conto, contiene, possiede, filiale}\} + 1 \text{ lettura}\{\text{filiale}\}) \cdot 150$$

Per inserire un conto bisogna scrivere nell'entità Conto e nelle due relazioni Contiene e Possiede, poichè un conto deve avere un cliente che lo possiede e il conto deve essere contenuto da una filiale. Infine bisogna leggere e scrivere nell'entità Filiale per aggiornare l'attributo *Attivi* con il saldo del conto appena inserito.

Senza attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Conto	Entità	150	Scrittura
Contiene	Relazione	150	Scrittura
Possiede	Relazione	150	Scrittura

Tabella 6: *Operazione 2*

$$\text{op2: } (3 \text{ scrittura}\{\text{conto, contiene, possiede}\}) \cdot 150$$

La logica è come quella vista sopra, con l'eccezione che non serve aggiornare l'attributo *Attivi*, che non è presente.

Operazione 3

Inserimento di una operazione in possiede con frequenza 1.000.000 al giorno

Con attributo *Attivi*:

Nome	Costrutto	Accessi	Tipo
Possiede	Relazione	1.000.000	Scrittura
Possiede	Relazione	1.000.000	Lettura
Conto	Entità	1.000.000	Scrittura
Conto	Entità	1.000.000	Lettura
Filiale	Entità	1.000.000	Scrittura
Filiale	Entità	1.000.000	Lettura
Contiene	Relazione	1.000.000	Lettura

Tabella 7: *Operazione 3*

$$\text{op3: } (3 \text{ scrittura}\{\text{Possiede, conto, filiale}\} + 4 \text{ letture}\{\text{Possiede, conto, filiale, contiene}\}) \cdot 1.000.000$$

Poichè la relazione Possiede contiene l'attributo operazione, ogni volta che un'operazione viene eseguita bisogna aggiornare l'attributo, ciò comporta una lettura e una scrittura. dopodiché, bisogna anche in questo caso aggiornare il saldo del conto che fa riferimento a quella tupla in possiede, dopodiché solamente leggere la relazione Contiene per individuare la filiale in cui quel conto ha sede e aggiornare quindi l'attributo *Attivi* della filiale.

Senza attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Possiede	Relazione	1.000.000	Scrittura
Possiede	Relazione	1.000.000	Lettura
Conto	Entità	1.000.000	Scrittura
Conto	Entità	1.000.000	Lettura

Tabella 8: *Operazione 3*

$$\text{op3: } (2 \text{ scritture}\{\text{Possiede, conto}\} + 2 \text{ letture}\{\text{Possiede, conto}\}) \cdot 1.000.000$$

La logica è la stessa di prima, ma non serve aggiornare l'attributo *Attivi* della filiale, quindi non serve leggere e scrivere nell'entità Filiale.

Operazione 4

Aggiornamento di tutti i prestiti con frequenza di una volta al mese.

Con attributo *attivi*:

Nome	Costrutto	Accessi	Tipo
Rata	Entità	233.333	Scrittura
È composto	Relazione	233.333	Lettura
Prestito	Entità	233.333	Lettura
Prestito	Entità	233.333	Scrittura
È associato	Relazione	233.333	Lettura
Contiene	Relazione	233.333	Lettura
Filiale	Entità	233.333	Lettura
Filiale	Entità	233.333	Scrittura

Tabella 9: *Operazione 4*

$$\text{op4: } (3 \text{ scritture}\{\text{Rata, Prestito, Filiale}\} +$$

$$5 \text{ Letture}\{\text{è composto, Prestito, è associato, Contiene, Filiale}\}) \cdot 7.000.000 \cdot \frac{1}{30}$$

⚠️

Abbiamo considerato l'aggiornamento mensile delle rate e quindi questo comporta la scrittura della rata che viene saldata in quel mese e da cui poi bisogna risalire al prestito a cui essa fa riferimento tramite la relazione *è composto*, aggiornare il prestito di riferimento, dopodiché tramite la relazione *è associato* ricavare l'iban del conto a cui è associato, poter quindi leggere in Contiene la filiale in cui quel prestito fa riferimento e quindi operare un aggiornamento dell'attributo attivi della filiale.

Senza attributo attivi:

Nome	Costrutto	Accessi	Tipo
Rata	Entità	233.333	Scrittura
È composto	Relazione	233.333	Lettura
Prestito	Entità	233.333	Lettura
Prestito	Entità	233.333	Scrittura

Tabella 10: *Operazione 4*

$$\text{op4: } (2 \text{ scritture}\{\text{Rata, Prestito}\} + 2 \text{ Letture}\{\text{è composto, Prestito}\}) \cdot 7.000.000 \cdot \tfrac{1}{30}$$

Anche in questo caso la logica rimane la stessa, ma non serve aggiornare l'attributo *Attivi* della filiale, quindi non serve leggere e scrivere nell'entità Filiale e nelle relazioni *è associato* e *contiene*.

$$\text{Totale con attributo attivi : } 13.274.017$$

$$\text{Totale senza attributo attivi : } 45.865.567$$

Questa analisi ci suggerisce che la conservazione dell'attributo derivato attivi sia utile e quindi lo manterremo nel nostro schema ER ristrutturato.

3.2.2. Studio dell'attributo derivato *Somma rate* di *PRESTITO*

Il secondo blocco di operazioni riguardano la ridondanza introdotta dall'attributo derivato somma rate dell'entità Prestito che misura il numero di rate che sono state pagate. Anche in questo caso è il caso di un attributo derivato secondo funzioni aggregative e le entità che sono coinvolte sono Rata e Prestito. Possiamo considerare due operazioni che sono:

Operazione 1

Inserimento di una rata una volta al mese per ogni prestito della banca

Con attributo ridondante *Somma rate*:

Nome	Costrutto	Accessi	Tipo
Rata	Entità	7.000.000	Scrittura
Prestito	Entità	7.000.000	Lettura
Prestito	Entità	7.000.000	Scrittura
è composto	Relazione	7.000.000	Lettura

Tabella 11: *Operazione 1*

$$\text{op1: } (2 \text{ scritture}\{\text{Rata, Prestito}\} + 2 \text{ Letture}\{\text{Prestito, è composto}\}) \cdot 7.000.000 \cdot \frac{12}{365}$$

Senza attributo ridondante *Somma rate*:

Nome	Costrutto	Accessi	Tipo
Rata	Entità	7.000.000	Scrittura

Tabella 12: *Operazione 1*

$$\text{op1: } (1 \text{ scrittura}\{\text{Rata}\}) \cdot 7.000.000 \cdot \frac{12}{365}$$

In questo caso, l'operazione di inserimento di una rata comporta la scrittura della rata e la scrittura del prestito, senza la necessità di leggere il prestito per aggiornare l'attributo somma rate.

Operazione 2

Lettura del valore della somma delle rate pagate per ogni prestito con frequenza di 2 volte all'anno.

Per questa analisi abbiamo dovuto introdurre un' ulteriore ipotesi, ovvero il numero medio di rate presenti nella nostra base di dati per ogni prestito. Abbiamo supposto questo numero essere 12, che equivale ad un anno di rate pagate.

Con attributo ridondante *Somma rate*:

Nome	Costrutto	Accessi	Tipo
Prestito	Entità	7.000.000	Lettura

Tabella 13: *Operazione 2*

$$\text{op2: } (1 \text{ lettura}\{\text{Prestito}\}) \cdot 7.000.000 \cdot \frac{2}{365}$$

Senza attributo ridondante *Somma rate*:

Nome	Costrutto	Accessi	Tipo
Prestito	Entità	7.000.000	Lettura
è associato	Entità	7.000.000	Lettura
Rata	Entità	7.000.000	Lettura

Tabella 14: *Operazione 2*

$$\text{op2: } (2 \text{ letture}\{\text{Prestito, è associato}\} + 1 \text{ lettura}\{\text{Rata}\} \cdot 12) \cdot 7.000.000 \cdot \tfrac{2}{365}$$

$$\text{Totale con ridondanza: } \quad 1.419.178,08$$

$$\text{Totale senza ridondanza: } \quad 997.960,27$$

⚠

Per questa ridondanza abbiamo concluso quindi che l'attributo somma rate possa essere rimosso e non essere utilizzato nello schema ER ristrutturato.

3.3. Selelezione delle chiavi primarie

Nell'entità *CLIENTE* abbiamo scelto come chiave primaria l'attributo *ID* rispetto a *Codice Fiscale* per mantenere una linearità con l'entità *DIPENDENTE* la quale è identificata a sua volta da un codice identificativo. In tutti gli altri casi la chiave candidata a essere primaria era unica.

3.4. Rimozione delle specializzazioni

Per le analisi fatte in precedenza siamo giunti alla conclusione che il blocco *CAPO-DI-DIPENDENTE* può essere «compresso», riducendo la complessità visiva e pratica del problema, eliminando la specializzazione capo e la relativa relazione *DI*, sostituendo il tutto con un nuovo attributo derivato posto nell'entità *DIPENDENTE*: *Id capo*. Di conseguenza viene anche cambiato il riferimento della relazione *È CAPO* che non farà più riferimento all'entità *CAPO* in quanto è stata eliminata ma bensì a *DIPENDENTE* mantenendo le cardinalità invariate. Non c'è perdita di informazione in quanto il nuovo attributo *ID capo* viene ricavato dalle relazioni *LAVORA* ed *È CAPO*. Per ricavare il capo di un certo dipendente posso andare a vedere la filiale in cui lavora (che è unica per le cardinalità della relazione), tale filiale sarà gestita da uno e un solo capo (deducibile dalle cardinalità della relazione è capo). Si può quindi, in maniera univoca, ricavare il capo di un certo dipendente passando attraverso le relazioni e salvare il dato di interesse nell'attributo *ID capo*.

Successivamente la specializzazione di *CONTO* è stata ristrutturata aggiungendo due nuove relazioni: *TIPO-CORRENTE* e *TIPO-RISPARMIO* che legano rispettivamente le entità *CORRENTE* e *RISPARMIO*. Gli attributi delle tre relazioni coinvolte nella specializzazione sono rimasti invariati. Le cardinalità delle due nuove relazioni sono (0,1) dal lato di *CONTO* in quanto un conto è sicuramente di uno dei due tipi e sicuramente non di entrambi, e dal lato di *CORRENTE* e *RISPARMIO* è (1,1) in quanto i due tipi di conto esistono e sono associate a uno e un solo conto. Le chiavi primarie di *CORRENTE* e di *RISPARMIO*

sono delle chiavi primarie legate alla relazione con conto, ne ereditano quindi la chiave primaria *IBAN*. Da notare il fatto che l'insieme degli *IBAN* di *CORRENTE* deve essere disgiunto dall'insieme *IBAN* di *RISPARMIO* (non esiste un conto che è sia corrente che di risparmio in quanto la specializzazione originariamente era disgiunta).

3.5. Schema ER ristrutturato

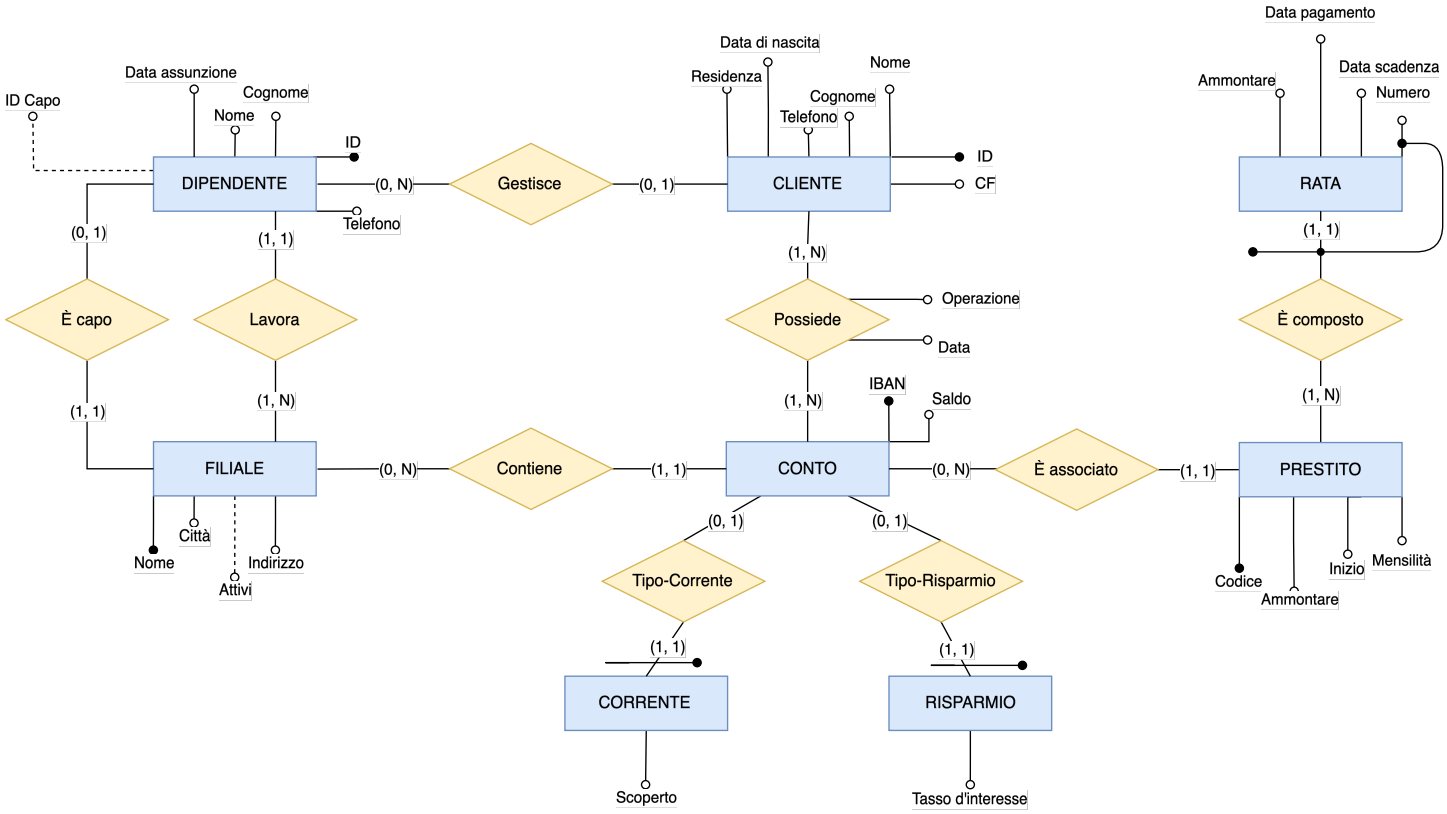

Figura 19: *Schema ER ristrutturato*

3.6. Schema Logico

CLIENTE (ID, CF, Residenza, DataDiNascita, Telefono, Cognome, Nome, *Gestore*)
- *CF*: UNIQUE

CONTO (*IBAN*, Saldo, *FilialeAppartenenza*)
- FilialeAppartenenza: NOT NULL

CONTO_CORRENTE (*IBAN*, Scoperto)
- *IBAN*: UNIQUE

CONTO_RISPARMIO (*IBAN*, TassoInteresse)
- *IBAN*: UNIQUE

POSSIEDE (*Cliente, Conto*, Operazione, Data)

PRESTITO (Codice, Ammontare, Inizio, Tipo, *ContoAssociato*)
- *ContoAssociato*: NOT NULL

RATA (Numero, *CodicePrestito*, Ammontare, DataPagamento, DataScadenza)

FILIALE (Nome, Città, Attivi(derivato), Indirizzo, *Capo*)
- *Capo* NOT NULL

DIPENDENTE (ID, Nome, Cognome, Telefono, DataAssunzione, *IDCapo* (derivato), *Filiale*)
- *Filiale* NOT NULL

Legenda: Le chiavi primarie sono sottolineate e le chiavi esterne sono in corsivo.

3.6.1. Chiavi esterne

- *Gestore* è chiave esterna di *CLIENTE* rispetto a *DIPENDENTE*
- *FilialeAppartenenza* è chiave esterna di *CONTO* rispetto a *FILIALE*
- *IBAN* è chiave esterna di *CONTO CORRENTE, CONTO RISPARMIO* rispetto a *CONTO*
- *Conto* chiave esterna di *POSSIEDE* rispetto a *CONTO*.
- *Cliente* è chiave esterna di *POSSIEDE* rispetto a *CLIENTE*
- *ContoAssociato* è chiave esterna di *PRESTITO* rispetto a CONTO
- *Capo* è chiave esterna di *FILIALE* rispetto a *DIPENDENTE*
- *CodicePrestito* è chiave esterna di *RATA* rispetto a *PRESTITO*
- *IDCapo* è chiave esterna di *DIPENDENTE* rispetto a DIPENDENTE
- *Filiale* è chiave esterna di *CLIENTE* rispetto a *FILIALE*

4. Popolamento del database

4.1. Creazione delle tabelle

Per ogni entità è stata creata una tabella nello schema fisico dove gli attributi dell'entità corrispondono ai campi della tabella. I campi della tabella sono stati opportunamente dichiarati in base al tipo di dato e aggiunto eventuali controlli sul loro valore per avere coerenza logica con quanto richiesto tramite l'utilizzo della condizione `CHECK()`.

Campi particolari che richiedevano di essere ad esempio chiave primaria, unici, o non nulli sono stati settati tramite gli appositi comandi.

Le tabelle sono state create in un ordine preciso; in particolare i vincoli di chiave esterna sono stati aggiunti solo quando tutte le tabelle coinvolte erano esistenti.

La tabella possiede è stata creata in quanto corrisponde alla relazione molti a molti tra l'entità *CONTO* e l'entità *CLIENTE*.

4.2. Modalità di generazione dei dati

Riportiamo di seguito la tabella dei volumi debitamente proporzionata sulla quale abbiamo creato i dati per il nostro database.

Nome	Costrutto	Volume
Cliente	Entità	30.000
Conto	Entità	24.000
Conto Corrente	Entità	20.000
Conto Risparmio	Entità	4.000
Dipendente	Entità	200
Filiale	Entità	6
Prestito	Entità	14.000
Contiene	Relazione	24.000
Di	Relazione	200
È associato	Relazione	14.000
È capo	Relazione	6
È composto	Relazione	14.000
Gestisce	Relazione	20.000
Lavora	Relazione	200
Possiede	Relazione	38.000

Tabella 15: *Tabella dei volumi proporzionata*

4.2.1. Dati *FILIALE*

Questi dati non richiedevano particolari attenzioni poiché non soggetti a nessun tipo di vincolo particolare.

4.2.2. Dati *DIEPNDENTE*

... To be continued by you 🥰

4.3. Creazione dei trigger

4.3.1. Trigger *FILIALE-DIPENDENTE*

Sono stati creati dei trigger per gestire le problematicità tra dipendente e filiale che non sono stati possibili catturare coi vincoli tramite lo schema relazionale.

Il manager di una filiale deve fare riferimento alla filiale che gestisce, pertanto non deve essere possibile cambiare la filiale di un manager. Il trigger controlla che su ogni inserimento o modifica nella tabella dipendente venga rispettato il vincolo appena descritto, sollevando un'eccezione in caso di problemi bloccando di conseguenza l'inserimento o la modifica.

Un altro trigger simile controlla che una volta assegnato il manager in una filiale esso lavori effettivamente in quella filiale.

4.3.2. Trigger *FILIALE-CONTO-PRESTITO-RATA*

La creazione delle rate di un prestito sono state gestite in modo automatico da un trigger il quale dopo l'inserimento di un prestito, calcola l'importo mensile di ogni rata in base all'ammontare e il numero di mensilità, creando le rate (tutte con lo stesso importo mensile) e mettendo la data di scadenza in modo coerente e sequenziale.

Un altro trigger controlla la possibilità di poter pagare una rata bloccando l'aggiornamento in caso la rata fosse già stata pagata, in caso di pagamento concesso, il trigger si occupa anche di aggiornare gli attivi della filiale corrispondente.

In modo analogo un trigger aggiorna gli attivi della filiale ogni volta che un nuovo prestito viene creato.

4.3.3. Trigger *POSSIEDE-CONTO-FILIALE*

Il calcolo degli attivi, analogamente come quello di prestiti/rate, viene fatto in automatico da un trigger ogni volta che viene aggiornato il saldo di un conto.

Per le scelte fatte nessun IBAN in conto corrente deve comparire in conto di risparmio e viceversa ma tutti gli IBAN di *CONTO CORRENTE* e di *CONTO RISPARMIO* devono comparire in *CONTO*, tale vincolo viene rispettato da due opportuni trigger.

La coerenza delle operazioni eseguibili su un determinato conto anch'essa è verificata da due appositi trigger. Viene controllato che l'operazione sia sensata sul conto (non posso aprire un conto due volte e non posso fare operazioni sul conto di risparmio) e in caso di prelievo un trigger si occupa di verificare il saldo rimanente e di aggiornarlo.

4.4. Inserimento tabelle e dati nel database

Per creare il database richiesto, popolarlo e testare le query assegnate è stata seguita una particolare logica affinché tutto venisse inserito correttamente.

Per prima cosa è stato creato il database, assegnando i volumi dei dati con valori proporzionati alla tabella dei volumi precedentemente proposta.

Vengono poi caricati nel sistema tutti i trigger utilizzati e temporaneamente disabilitati per possibili inconsistenze momentanee nell'inserimento dei dati. La modalità di generazione casuale dei dati è stata pensata in modo tale che, al termine degli inserimenti iniziali, tutto sia coerente e non ci siano errori.

Le prime tabelle popolate sono *FILIALE* e *DIPENDENTE*. Al termine del popolamento vengono eseguiti forzatamente due trigger in maniera tale da assegnare automaticamente i manager (che non erano stati inseriti) e verificare la presenza di eventuali errori (di base i dati sono stati generati consistentemente).

Estratti dei possibili gestori vengono inseriti i clienti, successivamente la tabella *CONTO* con le relative *CONTO CORRENTE* e *CONTO DI RISPARMIO*. Una volta inseriti questi dati è possibile procedere al popolamento della tabella *POSSIEDE* che gestisce tutte le connessioni tra i clienti e i loro conti.

Per la macrocategoria dei prestiti, una volta generati quest'ultimi e le relative rate andiamo, tramite apposito script, a pagare le rate che hanno una data di scadenza antecedente a quella odierna. Inseriti tutti i prestiti aggiorniamo l'attributo *attivi* della tabella *FILIALE* in maniera automatica sui dati inseriti e al termine riattiviamo tutti i trigger.

Gli script utilizzati non potevano essere sempre sostituiti dai trigger, infatti non era possibile tenerli tutti attivi e inserire tutti i valori in maniera ordinata e raggruppati per tabelle, ma avremmo dovuto fare attenzione volta per volta. Degli esempi di inserimenti di record sono presentati più avanti.

4.5. Test

Finito di popolare tutto il database ci assicuriamo tramite dei test che tutto sia perfettamente funzionante, che rispetti i requisiti che ci siamo imposti e che ci dia i risultati attesi. Questa verifica viene effettuata confrontando il risultato ottenuto dalle operazioni con i risultati attesi.

4.6. Test Dipendente-Filiale

1. Tentiamo di modificare la filiale di riferimento di un manager senza aggiornare il ruolo di managaer. Il trigger ci protegge e ci vieta l'inserimento (un dipendente non può lavorare nella filiale *A* ed essere manager della filiale *B*).
2. Simile al precedente, proviamo ad assegnare il ruolo di manager di una filiale a un dipendente che lavora presso una filiale diversa. Il trigger blocca l'azione e ci restituisce l'errore (la modifica non viene effettuata).
3. Inseriamo un nuovo dipendente: non è necessario specificare il campo manager in quanto il trigger apposito si occupa di ricercare l'id del manager nella filiale dove lavora il nuovo dipendente e assegnare il campo corrispondente.
4. Come il caso (3) ma con l'aggiunta che questo dipendente diventi manager della filiale in cui lavora. Il trigger che viene innescato sulla modifica del campo manager (che passa da −1 [non manager] a un id di filiale valido) provvede ad aggiornare il campo manager di tutti i dipendenti che lavorano nella filiale dove è appena stato modificato il manager.
5. Controlliamo una semplice operazione di rimozione di un dipendente che non è manager.

4.7. Test Prestito-Rata

1. Inseriamo un nuovo prestito. Le rate relative verranno generate in maniera automatica dal trigger che si occupa di andare a recuperare il valore di *Mensilità* e generare altrettanti record nella tabella *RATA* riempiendo in maniera adeguata tutti i campi.

2. Modifichiamo la data di pagamento di una rata, portandola da NULL a una data valida. Il controllo del trigger sarà di verificare che non ci siano rate precedenti ancora da pagare.

4.8. Test Conto-Filiale

1. Simuliamo un versamento e un prelievo, quindi andiamo a modificare il valore del saldo dei conti. A questo punto dei trigger controllano (solo nel secondo caso) che il prelievo possa essere effettuato, quindi che il saldo sia un numero valido (non minore dello scoperto), dopodiché in entrambi i casi vengono automaticamente aggiornati gli attivi delle filiali. Lo scopo del test è comunque di verificare che il saldo venga correttamente modificato.
2. Controlliamo che il trigger che controlla la validità dei saldi funzioni, forzando la modifica di un saldo a un valore non valido. Ci attendiamo un errore.
3. Simile al primo test con il focus sull'aggiornamento degli attivi della filiale di riferimento.
4. Proviamo a inserire un IBAN valido nella tabella *CONTO* (necessario per i vincoli di chiave esterna) e poi nella tabella *CONTO CORRENTE*. Questo non dovrebbe generare problemi. Proviamo a inserire l'IBAN anche in *CONTO RISPARMIO*, il trigger vieta tale operazione e, dato che siamo all'interno di una transazione, tutti e tre gli inserimenti vengono rimossi (rollback).
5. Test di consistenza dei gestori diversi su conti cointestati

5. Query

Dopo aver verificato che anche i test restituissero i risultati attesi, abbiamo proceduto con la scrittura delle query richieste.

5.1. Query 1

> *Restituire il numero medio di rate dei prestiti associati a conti nelle filiali di Udine.*

sql
```
Query 1
SELECT AVG(mensilità) AS media_rate
  FROM prestito, conto, filiale
  WHERE prestito.conto = conto.iban
    AND conto.filiale = filiale.nome
    AND filiale.città = 'Udine';
```

La richiesta è immediata con l'utilizzo della funzione aggregata `AVG()`.

5.2. Query 2:

> *Restituire i clienti con solo conti di risparmio in filiali che hanno tra i 30 e i 32 dipendenti.*

Per comodità è stata creata una vista dove è stat fatta una selezione sulla tabella *FILIALE*, tenendo solamente quelle che rispettavano il vincolo sul numero dei dipendenti. La query poi si appoggia su questa vista per cercare i clienti che hanno almeno un conto di risparmio in queste filiali e che non hanno nessun conto corrente associato.

sql

```
Query 2
CREATE OR REPLACE VIEW filiali_3032 AS
  SELECT filiale, COUNT(*) AS n_dip
  FROM dipendente
  GROUP BY filiale
  HAVING COUNT(*) BETWEEN 30 AND 32;

SELECT cliente.id
  FROM cliente, possiede, conto, filiali_3032
  WHERE cliente.id = possiede.cliente
    AND possiede.conto = conto.iban
    AND conto.filiale = filiali_3032.filiale
    AND NOT EXISTS (
      SELECT 1
      FROM contocorrente
      WHERE contocorrente.iban = conto.iban
    );
```

5.3. Query 3:

Restituire i capi che gestiscono almeno 3 clienti che possiedono almeno 100.000€.

sql

```
Query 2
CREATE OR REPLACE VIEW clienti_ricchi AS
  SELECT cliente.id, SUM(conto.saldo) AS soldi, cliente.gestore
  FROM cliente, possiede, conto
  WHERE cliente.id = possiede.cliente
    AND conto.iban = possiede.conto
  GROUP BY cliente.id, cliente.gestore
  HAVING SUM(conto.saldo) > 100000;

SELECT DISTINCT capo
  FROM dipendente
  WHERE EXISTS (
    SELECT *
    FROM clienti_ricchi c1, clienti_ricchi c2, clienti_ricchi c3
    WHERE c1.gestore = dipendente.id
      AND c2.gestore = dipendente.id
      AND c3.gestore = dipendente.id
      AND c1.id < c2.id AND c2.id < c3.id
  );
```

La vista creata è una restrizione sui clienti che rispettano il vincolo. È stata effettuata con l'utilizzo della funzione `SUM()` poiché il saldo era relativo a tutti i conti posseduti. Per validare un capo è stato fatto il prodotto cartesiano triplo della vista e, dopo essere state selezionati solamente le righe con gestore uguale, è stato controllato che i clienti fossero tutti e tre diversi.

5.4. Query 4:

Restituire i dipendenti non capo che gestiscono esattamente 2 clienti, uno con solo conti correnti e uno son solo conti di risparmio.

sql

Query 4

```sql
CREATE OR REPLACE VIEW clienti_correnti AS
  SELECT possiede.cliente, cliente.gestore
  FROM possiede, cliente
  WHERE possiede.cliente = cliente.id
    AND NOT EXISTS (
      SELECT 1
      FROM contorisparmio
      WHERE contorisparmio.iban = possiede.conto
    );

  CREATE OR REPLACE VIEW clienti_risparmio AS
  SELECT possiede.cliente, cliente.gestore
  FROM possiede, cliente
  WHERE possiede.cliente = cliente.id
    AND NOT EXISTS (
      SELECT 1
      FROM contocorrente
      WHERE contocorrente.iban = possiede.conto
    );

 SELECT id
  FROM dipendente
  WHERE capo <> id
    AND EXISTS (
      SELECT *
      FROM clienti_correnti cc1
      WHERE cc1.gestore = dipendente.id
        AND NOT EXISTS (
          SELECT *
          FROM clienti_correnti cc2
          WHERE cc2.gestore = dipendente.id
            AND cc1.cliente <> cc2.cliente
        )
    )
    AND EXISTS (
      SELECT *
      FROM clienti_risparmio cr1
      WHERE cr1.gestore = dipendente.id
        AND NOT EXISTS (
          SELECT *
          FROM clienti_risparmio cr2
          WHERE cr2.gestore = dipendente.id
            AND cr1.cliente <> cr2.cliente
        )
    );
```

La prima (seconda) vista seleziona solamente i clienti che hanno almeno un conto corrente (di risparmio) e che non hanno nessun conto di risparmio (corrente). La query seleziona i dipendenti non

capo (con la verifica *ID* <> *Capo*) e poi controlla che esista un unico cliente nella prima vista e un unico cliente nella seconda vista.

5.5. Query 5:

> *Restituire il cliente con il prestito più alto nella filiale di Roma che non ha come gestore un dipendente con meno di 3 anni di esperienza.*

sql

```
Query 5
CREATE OR REPLACE VIEW clienti_gestiti_3 AS
 SELECT cliente.id
 FROM cliente, dipendente
 WHERE cliente.gestore = dipendente.id
   AND dipendente.data_assunzione < DATE '", data_limite, "';

CREATE OR REPLACE VIEW candidati AS
 SELECT cliente.id, prestito.ammontare
 FROM cliente, clienti_gestiti_3, possiede, prestito, conto, filiale
 WHERE cliente.id = clienti_gestiti_3.id
   AND cliente.id = possiede.cliente
   AND possiede.conto = prestito.conto
   AND prestito.conto = conto.iban
   AND conto.filiale = filiale.nome
   AND filiale.città = 'Roma';

 SELECT id, ammontare
 FROM candidati c1
 WHERE NOT EXISTS (
   SELECT 1
   FROM candidati c2
   WHERE c2.ammontare > c1.ammontare
 );
```

La prima vista ci restringe i possibili clienti a quelli che hanno un gestore assunto da almeno 3 anni. La seconda vista, a partire dalla prima, fa un ulteriore filtro prendendo i clienti solo della filiale di Roma. La query si occupa di verificare, per ogni cliente, che tra i clienti della seconda vista non ce ne sia qualcuno con saldo maggiore del proprio, in tal caso stampa il cliente.

6. Analisi dei dati

6.1. Visualizzazione dei dati

6.1.1. Distribuzione mensilità prestiti

6.1.2. Analisi attivi per anzianità gestori

6.1.3. Analisi conti cointestati

7. Conclusioni

7.1. Risultati ottenuti

7.2. Possibili miglioramenti

7.3. Considerazioni finali

(test)