Eserciziario di Basi di Dati

Sara Foresti
Eros Pedrini
Sabrina De Capitani di Vimercati

Indice

Prefazione

Questo testo contiene una collezione di esercizi rivolti ad un corso universitario di Basi di Dati. I corsi di riferimento sono quelli tenuti presso il *Dipartimento di Tecnologie dell'Informazioni di Crema* (*Università degli Studi di Milano*) dalla *Professoressa Pierangela Samarati*.

Lo scopo principale del lavoro è quello di permettere agli studenti di un corso di Basi di Dati di esercitarsi sugli argomenti introdotti nelle lezioni e di preparasi all'esame scritto. Ogni Capitolo presenta i richiami alla parte di teoria necessari per la completa comprensione degli esercizi proposti. Crediamo, quindi, che possa costituire un utile strumento di verifica, apprendimento e approfondimento, non solo per gli studenti, ma anche per chiunque sia interessato al mondo delle basi di dati.

Tutti gli esercizi presenti in questo testo sono risolti; alcuni presentano una soluzione completamente commentata e vengono proposti dopo i richiami di teoria, gli altri presentano solamente la soluzione di riferimento. Va inoltre osservato che per alcune tipologie di esercizi (ad esempio quelli di interrogazione e di progetto) non sempre esiste una soluzione univoca; per questa tipologia di esercizi è stata proposta una delle possibili soluzioni in modo tale che i lettori possano confrontare, con spirito critico, la loro soluzione con quella suggerita.

Il testo è strutturato logicamente in due parti, che vanno a coprire i due corsi di *Basi di Dati del Dipartimento di Tecnologie dell'Informazioni di Crema*: *Basi di Dati: Elementi* e *Basi di Dati: Complementi*. La prima parte, che include i Capitoli 1, 2 e 3, copre il corso di *Basi di Dati: Elementi*. Il *Capitolo 1* tratta dell'*Algebra Relazionale*. Il *Capitolo 2* tratta dello *Structured Query Language* (*SQL*). Infine il *Capitolo 3* tratta la *Progettazione di Basi di Dati*, attraverso il *Modello Entità-Relazione*.

Nella seconda parte, viene fornita una panoramica sui principali argomenti del corso di *Basi di Dati: Complementi*. Il *Capitolo 4* tratta della *Gestione delle Transazioni* coprendo i seguenti argomenti: *Ripresa a Caldo*, *Serializzabilità* e *Timestamp*. Il *Capitolo 5* tratta i principali argomenti di *Supporto alle decisioni* (*Aggregazioni ROLAP* in *SQL* e *Data Mining*).

Vorremmo rivolgere uno speciale ringraziamento alla professoressa ***Pierangela Samarati*** *ed al professor* ***Stefano Paraboschi****, i cui commenti e suggerimenti ci hanno aiutato a migliorare questo eserciziario.*

Capitolo 1

Algebra Relazionale

Questo Capitolo è dedicato all'*Algebra Relazionale*, cioè al linguaggio procedurale che utilizza una simbologia algebrica. L'algebra relazionale opera sul modello relazionale ed espressioni algebriche producono come risultato delle relazioni.
Si noti che, nel seguito del Capitolo, si fa riferimento ad una notazione algebrica *pura*. Di conseguenza, le relazioni in analisi *non hanno chiave*. Non è infatti possibile che una relazione abbia due (o più) tuple uguali, visto che gli operatori algebrici non mantengono i duplicati.

1.1 Algebra Relazionale: Operatori

L'algebra relazionale si basa su nove operatori, che si distinguono in operatori *di base*, *derivati* e *aggiunti*.
Gli operatori di base sono:

- selezione (σ)
- proiezione (π)
- unione ($\cup$)
- differenza ($-$)
- prodotto cartesiano ($\times$)

Gli operatori derivati sono:

- intersezione ($\cap$)
- join ($\bowtie$)

Gli operatori aggiunti sono:

- ridenominazione (ρ)
- assegnamento ($:=$)

Si ricordi che dato uno schema relazionale $R(a_1, 1_2, \ldots, a_n)$ si definisce:

- *grado di R* ($gr(R)$) il numero n di attributi di R;
- *cardinalità di R* ($|R|$) il numero di tuple (righe) di R.

Per comodità di esposizione, l'ordine seguito nella presentazione degli operatori è il seguente: operatori aggiunti, di base e, in fine, derivati.

1.1.1 Operatori Aggiunti

Ridenominazione Questo operatore consente di cambiare il nome ad uno o più attributi nella relazione operando.
La sintassi è la seguente:

$$\rho_{\,b_1, b_2, \ldots, b_n \leftarrow a_1, a_2, \ldots, a_n}(\mathrm{R})$$

dove, R è la relazione operando, a_i (con $i = 1, 2, \ldots, n$) rappresentano i nomi degli attributi che si vuole rinominare in R; b_i (con $i = 1, 2, \ldots, n$) sono i *nuovi* nomi da assegnare, nell'ordine, agli attributi.
Il risultato di questa operazione è una relazione *senza nome*, avente lo stesso schema di R a meno degli attributi rinominati (quindi con lo stesso grado), e con la stessa cardinalità di R.

Assegnamento Questo operatore consente di dare un nome al risultato di una espressione algebrica.
La sintassi è la seguente:

$$\mathrm{NOME} := \mathrm{R}$$

Il risultato di questa operazione è una relazione con nome NOME e schema, grado e cardinalità pari a quelli dell'operando R.
Questo operatore viene in genere impiegato per definire delle viste, al fine di risolvere interrogazioni complesse.

1.1.2 Operatori di Base

Selezione Questo operatore consente di estrarre un sottoinsieme delle tuple (righe) della relazione operando, basandosi su una condizione booleana di filtro specificata.

La sintassi è la seguente:

$$\sigma_{\text{Condizione}}(\text{R})$$

dove *Condizione* è un'espressione booleana ($\wedge$, $\vee$, $\neg$), di condizioni atomiche della forma A *op* B. A e B possono essere attributi o costanti, mentre *op* è un operatore di confronto ($=$, $\neq$, $>$, $<$, $\leq$, $\geq$).
Il risultato di questa operazione è una relazione *senza nome*, avente lo stesso schema e grado di R, ma composta da tutte e sole le tuple di R che soddisfano la condizione specificata. Di conseguenza la cardinalità della relazione risultante è un valore compreso fra zero e la cardinalità di R.

Proiezione Questo operatore consente di estrarre un sottoinsieme degli attributi (colonne) della relazione operando.
La sintassi è la seguente:

$$\pi_{a_1, a_2, \ldots, a_n}(\text{R})$$

dove $a_1, a_2, \ldots, a_n$ rappresenta un sottoinsieme degli attributi di R.
Il risultato di questa operazione è una relazione *senza nome*, avente come schema $(a_1, a_2, \ldots, a_n)$, cioè i soli attributi specificati per la proiezione. Il grado è pari al numero degli attributi oggetto della proiezione. La cardinalità della relazione risultante è pari a zero se R è vuota, è un valore compreso fra uno e la cardinalità di R altrimenti. Si noti che, in algebra relazionale, una relazione *non* contiene mai tuple duplicate, quindi nemmeno il risultato di un'espressione algebrica può contenerli. Questo è il motivo per cui, in seguito ad una proiezione, la cardinalità di una relazione può diminuire.

Unione Questo operatore consente di calcolare l'unione insiemistica fra due relazioni, non mantenendo, quindi, i duplicati.
La sintassi è la seguente:

$$\text{R} \cup \text{S}$$

dove R e S sono due relazioni definite sullo *stesso* insieme di attributi (quindi sullo stesso schema).
Il risultato di questa operazione è una relazione *senza nome*, avente come schema lo stesso schema di R e di S e quindi anche lo stesso grado. La cardinalità minima della relazione risultante è la cardinalità della più grande fra R e S. La cardinalità massima della relazione risultante è la somma delle cardinalità dei due operandi R e S.

Differenza Questo operatore consente di calcolare la differenza insiemistica fra due relazioni, cioè il risultato contiene le tuple della prima relazione operando che non compaiono anche nella seconda.

La sintassi è la seguente:

$$R - S$$

dove R e S sono due relazioni definite sullo *stesso* insieme di attributi (quindi sullo stesso schema).
Il risultato di questa operazione è una relazione *senza nome*, avente come schema lo stesso schema di R e di S e quindi anche lo stesso grado. La cardinalità minima della relazione risultante è zero. La cardinalità massima della relazione risultante è invece pari alla cardinalità di R.

Prodotto Cartesiano Questo operatore consente di calcolare il prodotto cartesiano fra due relazioni, cioè il risultato contiene tutte le possibili coppie di tuple, dove la prima appartiene al primo operando e la seconda al secondo.
La sintassi è la seguente:

$$R \times S$$

dove R e S sono due relazioni. Considerando $R(a_1, a_2, \ldots, a_n)$ e $S(b_1, b_2, \ldots, b_m)$, la relazione risultato ha il seguente schema: $(a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m)$.
Si noti che, affinché l'operazione possa essere eseguita, gli schemi di R e S devono essere *disgiunti*; cioè non contenere attributi con lo stesso nome. Il risultato di questa operazione è una relazione *senza nome*, avente come schema la concatenazione dei due schemi e quindi avente come grado la somma dei gradi dei due operandi. La cardinalità è pari invece al prodotto fra le cardinalità dei due operandi.

1.1.3 Operatori Derivati

Intersezione Questo operatore consente di calcolare l'intersezione insiemistica fra due relazioni.
La sintassi è la seguente:

$$R \bigcap S$$

dove R e S sono due relazioni definite sullo *stesso* insieme di attributi (quindi sullo stesso schema).
Questo operatore si dice *derivato* perché si può ottenre a partire dall'operatore di base *differenza* come segue:

$$R \bigcap S \equiv R - (R - S)$$

Il risultato di questa operazione è una relazione *senza nome*, avente come schema lo stesso schema di R e di S e quindi anche lo stesso grado. La cardinalità della relazione risultante è compresa fra zero e la cardinalità della più piccola fra R e S.

Join Esistono due diversi tipi di join: *theta join* e *join naturale*. Entrambi sono ottenuti dalla composizione di un prodotto cartesiano e una selezione.

Theta join La sintassi è la seguente:

$$R \bowtie_{Condizione} S$$

dove *Condizione* è una condizione booleana che può essere applicata al prodotto cartesiano di R e S. Affinché l'operazione possa essere eseguita, gli schemi di R e S devono essere *disgiunti*.
Questo operatore si dice *derivato* perché si può ottenere dagli operatori di base *selezione* e *prodotto cartesiano* come segue:

$$R \bowtie_{Condizione} S \equiv \sigma_{Condizione}(R \times S)$$

Il risultato di questa operazione è una relazione *senza nome*, avente come schema la concatenazione dei due schemi e quindi avente come grado la somma dei gradi dei due operandi. La cardinalità della relazione risultante è compresa fra zero e il prodotto delle cardinalità dei due operandi.

Un tipo particolare di theta join è l'*equi join*, dove la condizione è composta da una o più condizioni di uguaglianza, composte tra loro da $\wedge$ (and logici). In generale, le condizioni di uguaglianza, coinvolgono un attributo di R e uno di S.

Join Naturale La sintassi è la seguente:

$$R \bowtie S$$

Affinché l'operazione possa essere eseguita, R e S devono avere almeno un attributo comune. In caso contrario, l'operazione degenera in un prodotto cartesiano.
Questo operatore si dice *derivato* perché si può ottenere dagli operatori di base *proiezione*, *selezione* e *prodotto cartesiano* come segue:

$$R \bowtie S \equiv \pi_{A_1, A_2, \ldots, A_n}(\sigma_{Condizione}(R \times S))$$

dove *Condizione* è ottenuta ponendo uguali gli attributi comuni a R e S, e l'insieme di attributi $A_1, A_2, \ldots, A_n$, presenti nella proiezione, sono tutti gli attributi di R e S a meno dei possibili duplicati: nel risultato viene mantenuta solo una copia degli attributi comuni.
Il risultato di questa operazione è una relazione *senza nome*, avente come schema l'unione dei due schemi, a meno degli attributi comuni, e quindi avente come grado la somma dei gradi dei due operandi, meno il numero di attributi comuni. La cardinalità della relazione risultante è compresa fra zero e il prodotto delle cardinalità dei due operandi.

Operatore	*Grado*	*Card. Min.*	*Card. Max*
$\rho_{\ b_1,\ldots,b_n \leftarrow a_1,\ldots,a_n}(\mathrm{R})$	gr(R)	|R|	|R|
NOME := R	gr(R)	|R|	|R|
$\sigma_{\ \mathrm{Condizione}}(\mathrm{R})$	gr(R)	0	|R|
$\pi_{\ a_1, a_2, \ldots, a_n}(\mathrm{R})$	n	1	|R|
$\mathrm{R} \bigcup \mathrm{S}$	gr(R)	max(|R|,|S|)	|R|+|S|
$\mathrm{R} - \mathrm{S}$	gr(R)	0	|R|
$\mathrm{R} \times \mathrm{S}$	gr(R)+gr(S)	$\|\mathrm{R}\| \cdot \|\mathrm{S}\|$	$\|\mathrm{R}\| \cdot \|\mathrm{S}\|$
$\mathrm{R} \bigcap \mathrm{S}$	gr(R)	0	min(|R|,|S|)
$\mathrm{R} \bowtie_{\mathrm{Condizione}} \mathrm{S}$	gr(R)+gr(S)	0	$\|\mathrm{R}\| \cdot \|\mathrm{S}\|$
$\mathrm{R} \bowtie \mathrm{S}$	gr(R)+gr(S)−C[1]	0	$\|\mathrm{R}\| \cdot \|\mathrm{S}\|$

Tabella 1.1: Tabella riassuntiva degli operatori (si suppongono R e S entrambe non vuote)

1.2 Equivalenza fra Espressioni

Una interrogazione, a parità di risultato, può essere scritta in forme diverse; se queste forme non dipendono dagli schemi delle relazioni coinvolte, si dicono equivalenti fra di loro.
Le principali trasformazioni di equivalenza sono di seguito elencate. Si noti che: R, S e T sono espressioni algebriche; c_i sono condizioni; a_i sono singoli attributi; A_i sono insiemi di attributi.

- Atomizzazione delle selezioni:
  $\sigma_{\ c_1 \wedge c_2}(\mathrm{R}) \equiv \sigma_{\ c_1}(\sigma_{\ c_2}(\mathrm{R}))$

- Idempotenza delle proiezioni:
  $\pi_{\ A_1}(\mathrm{R}) \equiv \pi_{\ A_1}(\pi_{\ A_1, A_2}(\mathrm{R}))$

- Anticipazione delle selezioni rispetto al join:
  $\sigma_{\ c}(\mathrm{R} \bowtie \mathrm{S}) \equiv \mathrm{R} \bowtie_c \mathrm{S}$
  dove c è una condizione che riguarda solo attributi in S

- Anticipazione delle proiezioni rispetto al join:
  $\pi_{\ A_1, A_2}(\mathrm{R} \bowtie \mathrm{S}) \equiv \mathrm{R} \bowtie \pi_{\ A_2}(\mathrm{S})$
  dove A_1 è l'insieme di tutti gli attributi di R, mentre A_2 è il sottoinsieme di attributi di S coinvolti nel join.

- Inglobamento della selezione in un prodotto cartesiano:
  $\sigma_{\ c}(\mathrm{R} \times \mathrm{S}) \equiv \mathrm{R} \bowtie_c \mathrm{S}$

- Distributività della selezione rispetto all'unione:
  $\sigma_{\ c}(\mathrm{R} \bigcup \mathrm{S}) \equiv \sigma_{\ c}(\mathrm{R}) \bigcup \sigma_{\ c}(\mathrm{S})$

[1] C è la cardinalità dell'insieme $\Psi(R) \cap \Psi(S)$, dove $\Psi(X)$ è una funzione che applicata su una relazione X restituisce l'insieme di tutti gli attributi di X; $\Psi(X) = \{x_1, x_2, \ldots, x_n\}$.

- Distributività della selezione rispetto alla differenza:
  $\sigma_c(R - S) \equiv \sigma_c(R) - \sigma_c(S)$
- Distributività della selezione rispetto all'intersezione:
  $\sigma_c(R \cap S) \equiv \sigma_c(R) \cap \sigma_c(S)$
- Distributività della proiezione rispetto all'unione:
  $\pi_A(R \cup S) \equiv \pi_A(R) \cup \pi_A(S)$
- Distributività del join rispetto all'unione:
  $R \bowtie (S \cup T) \equiv (R \bowtie S) \cup (R \bowtie T)$
- Corrispondenza fra operatori insiemistici e selezioni:
  - $\sigma_{c_1 \vee c_2}(R) \equiv \sigma_{c_1}(R) \cup \sigma_{c_2}(R)$
  - $\sigma_{c_1 \wedge c_2}(R) \equiv \sigma_{c_1}(R) \cap \sigma_{c_2}(R) \equiv \sigma_{c_1}(R) \bowtie \sigma_{c_2}(R)$
  - $\sigma_{c_1 \wedge \neg c_2}(R) \equiv \sigma_{c_1}(R) - \sigma_{c_2}(R)$
- Commutatività e associatività di tutti gli operatori binari (tranne la differenza):
  - $R \; op \; (S \; op \; T) \equiv (R \; op \; S) \; op \; T$
  - $R \; op \; S \equiv S \; op \; R$

  dove *op* può essere $\cup$, $\cap$, $\times$ o $\bowtie$.

1.3 Esercizi Svolti e Commentati

In questa sezione vengono presentati alcuni esercizi svolti.

1.3.1 Festa

Sia dato lo schema relazionale:

FESTA(Codice, Costo, NomeRistorante)
REGALO(NomeInvitato, CodiceFesta, Regalo)
INVITATO(Nome, Indirizzo, Telefono)

Interrogazione 1

> Determinare il nome dei ristoranti in cui non c'è mai stato un invitato di nome 'Marco'.

Per la risoluzione di questa interrogazione, è necessario per prima cosa determinare il nome dei ristoranti nei quali si è svolta una festa (*NomeRistorante*) e poi determinare il nome dei ristoranti dove si è svolta almeno una festa dove almeno un invitato ha nome 'Marco'. Il risultato finale è ottenuto dalla sottrazione insiemistica fra i due insiemi trovati.

Come prima cosa, si individua il nome di tutti i ristoranti dove si è svolta almeno una festa.

$$F := \pi_{\text{NomeRistorante}}(\text{FESTA})$$

Si selezionano poi, fra le partecipazioni alle feste, solo quelle che riguardano un invitato di nome 'Marco'.

$$M := \sigma_{\text{NomeInvitato='Marco'}}(\text{REGALO})$$

Attraverso il join fra M e FESTA, si ottengono tutti i dati delle feste dove almeno un invitato ha nome 'Marco'. Dai dati, si estrae quindi il nome del ristorante. La vista RM contiene quindi i nomi dei ristoranti dove si è svolta almeno una festa cui ha partecipato un invitato di nome 'Marco'.

$$RM := \pi_{\text{NomeRistorante}}(M \bowtie_{\text{Codice=CodiceFesta}} \text{FESTA})$$

Facendo poi la differenza fra tutti i nomi di ristorante e i nomi di ristorante dove si è svolta almeno una festa cui ha partecipato un invitato di nome 'Marco', si ottengono i nomi di ristorante in cui non c'è mai stato un invitato di nome 'Marco'.

$$F - RM$$

Si noti che, per ottenere il risultato corretto, è necessario eseguire la proiezione su *NomeRistorante* prima di effettuare la sottrazione e questo perché altrimenti la sottrazione non potrebbe essere eseguita a causa dei differenti schemi degli operandi;

Interrogazione 2

> Determinare il nome degli invitati che hanno partecipato a tutte le feste avvenute nel ristorante 'Il Tulipano'.

Per la risoluzione di questa interrogazione, si individuano i nomi degli invitati che *non hanno partecipato ad almeno una* festa al ristorante 'Il Tulipano'. Sottraendo poi questo gruppo dall'insieme di tutti gli invitati a feste tenutesi a 'Il Tulipano', si ottiene quanto richiesto dall'interrogazione.

Come prima cosa si individua il *Codice* di tutte le feste che si sono svolte a 'Il Tulipano'.

$$\text{FT} := \pi_{\text{Codice}}(\sigma_{\text{NomeRistorante='Il Tulipano'}}(\text{FESTA}))$$

Si esegue poi il prodotto cartesiano fra il nome di tutti gli invitati a una qualsiasi festa e il codice di tutte le feste avvenute a 'Il Tulipano'. L'insieme così ottenuto contiene *tutte* le possibili coppie (*NomeInvitato*, *CodiceFesta*). In altre parole, questo insieme contiene sia gli inviti realmente avvenuti, sia gli inviti che sono sono stati formulati.

$$\text{TUTTI} := \pi_{\text{NomeInvitato}}(\text{REGALO}) \times \rho_{\text{CodiceFesta}\leftarrow\text{Codice}}(\text{FT})$$

Si estraggono poi tutte le coppie di inviti (*NomeInvitato*, *CodiceFesta*) realmente avvenute (in qualsiasi ristorante).

$$\text{REALI} := \pi_{\text{NomeInvitato,CodiceFesta}}(\text{REGALO})$$

È possibile a questo punto, attraverso una sottrazione, individuare le coppie di inviti (*NomeInvitato*, *CodiceFesta*), per feste tenutesi a 'Il Tulipano', che non sono avvenute. Dal risultato della sottrazione, attraverso una proiezione, si estraggono i nomi degli invitati che non hanno partecipato ad almeno una delle feste tenutesi a 'Il Tulipano'.

$$\text{NRIS} := \pi_{\text{NomeInvitato}}(\text{TUTTI} - \text{REALI})$$

Ora si toglie dall'insieme di tutti gli invitati ad almeno una festa (REALI), coloro che non hanno partecipato ad almeno una delle feste tenutesi a 'Il Tulipano'.

$$\pi_{\text{NomeInvitato}}(\text{REALI}) - \text{NRIS}$$

1.3.2 Ditta

Sia dato lo schema relazionale:

PERSONA(CF, Nome, Cognome, Stipendio)
LAVORA(CFPersona, PIvaDitta, Anzianità)
DIRIGE(CFPersona, PIvaDitta)
DITTA(PIva, Denominazione, NumeroDipendeti)

Interrogazione 1

> Determinare nome e cognome dei dirigenti che percepiscono lo stipendio più alto.

Per la soluzione di questa interrogazione è necessario individuare i dati anagrafici e lo stipendio di tutti i dirigenti. Si selezionano poi i dirigenti per cui esiste un altro dirigente con stipendio maggiore, ovvero i dirigenti che *non* percepiscono il massimo stipendio. Tramite una sottrazione si ottiene infine il risultato voluto.

Come primo passo nella soluzione, si individuano i dati di interesse dei dirigenti, ovvero nome, cognome e stipendio, per mezzo di un join tra le DIRIGE e PERSONA. Si crea poi una vista equivalente, con gli attributi rinominati.

$$\text{DIR} := \pi_{\text{Nome,Cognome,Stipendio}}(\text{DIRIGE} \bowtie_{\text{CF=CFPersona}} \text{PERSONA})$$

$$\text{DIR1} := \rho_{\text{n,c,s}\leftarrow\text{Nome,Cognome,Stipendio}}(\text{DIR})$$

Si compongono, quindi, le due viste, contenenti gli stessi dati, attraverso un join, che restituisce le coppie di dirigenti per cui lo stipendio del primo è inferiore a quello del secondo. Estraendo a questo punto i soli dati corrispondenti a quelli presenti nella tabella DIR, si ottengono i dati dei dirigenti che non percepiscono lo stipendio massimo.

$$\text{NRIS} := \pi_{\text{Nome,Cognome,Stipendio}}(\text{DIR} \bowtie_{\text{Stipendio<s}} \text{DIR1})$$

Come ultimo passo, attraverso una sottrazione fra tutti i dirigenti e coloro che non percepiscono lo stipendio massimo, si ottiene il risultato finale all'interrogazione.

$$\pi_{\text{Nome,Cognome}}(\text{DIR} - \text{NRIS})$$

Interrogazione 2

> Determinare la denominazione delle ditte in cui esiste almeno un lavoratore che percepisce uno stipendio maggiore di quello di almeno un dirigente della stessa ditta.

Per poter risolvere questa interrogazione è per prima cosa necessario individuare tutte le informazioni di interesse sia per i lavoratori sia per i dirigenti. Successivamente, si confrontano tra loro gli stipendi di dirigenti e lavoratori per selezionare le coppie che rispettano le condizioni poste dall'interrogazione.

Come primo passo, si individuano i dati di interesse per i dirigenti, cioè *Denominazione*, *Stipendio* e *PIva*. A tale scopo è necessario comporre le tabelle DIRIGE, PERSONA e DITTA, per mezzo di una operazione di join.

$$\text{DIR} := \text{DIRIGE} \bowtie_{\text{CF=CFPersona}} \text{PERSONA} \bowtie_{\text{PIva=PIvaDitta}} \text{DITTA}$$

Si crea, quindi, una tabella analoga, ma con gli attributi DENOMINZIONE, STIPENDIO e PIVA, ridenominati.

$$\text{SDIR} := \rho_{\text{d}_1,\text{s}_1,\text{p}_1 \leftarrow \text{Denominazione,Stipendio,PIva}}(\pi_{\text{Denominazione,Stipendio,PIVA}}(\text{DIR}))$$

La stessa operazione viene fatta per i lavoratori, utilizzando la tabella LAVORA al posto della tabella DIRIGE.

$$\text{LAV} := \text{LAVORA} \bowtie_{\text{CF=CFPersona}} \text{PERSONA} \bowtie_{\text{PIva=PIvaDitta}} \text{DITTA}$$

$$\text{SLAV} := \rho_{\text{d}_2,\text{s}_2,\text{p}_2 \leftarrow \text{Denominazione,Stipendio,PIva}}(\pi_{\text{Denominazione,Stipendio,PIva}}(\text{LAV}))$$

Una volta acquisite tutte le informazioni di interesse per dirigenti e lavoratori, si compongono le viste SDIR ed SLAV in modo da formare tutte le possibili coppie (*Dirigente*, *Lavoratore*) tali per cui la ditta per cui lavorano è la stessa e lo stipendio del primo è inferiore a quello del secondo. Delle coppie selezionate si estrae la denominazione della ditta.

$$\pi_{\text{d}_1}(\text{SDIR} \bowtie_{\text{p}_1=\text{p}_2 \wedge \text{s}_1<\text{s}_2} \text{SLAV})$$

1.3.3 Computer

Sia dato lo schema relazionale:

COMPUTER(Codice, Descrizione, Marca)
RIPARAZIONE(CodiceComputer, Data, Guasto, Costo)

Interrogazione 1

> Determinare i dati dei computer di marca 'Compaq' che hanno subito almeno tre riparazioni.

Per risolvere questa interrogazione è necessario individuare i codici dei computer di marca 'Compaq', e per ciascun codice, verificare se la tabella RIPARAZIONE contiene almeno 3 tuple dove il CODICECOMPUTER è uguale a quello considerato. A tale scopo so creano

delle terne di computer 'Compaq' che hanno subito delle riparazione e si scelgono solo le terne che riguardano lo stesso computer ma che fanno riferimento a riparazioni avvenute in date diverse.

Per prima cosa, si selezionano i computer di marca 'Compaq' e si individuano le tuple nella tabella RIPARAZIONE che li riguardano, attraverso una semplice operazione di join.

$$\text{CP} := \sigma_{\text{Marca='Compaq'}}(\text{COMPUTER})$$

$$\text{CR} := \pi_{\text{Codice,Data}}(\text{CP} \bowtie_{\text{Codice=CodiceComputer}} \text{RIPARAZIONE})$$

Le tuple nella vista CR (Computer Riparati) vengono composte in modo da creare una vista T (Terne) contenente tutte le triple di riparazioni a computer, compresi i duplicati. Tra queste tuple, per individuare i computer 'Compaq' riparati almeno tre volte, si selezionano quelle dove tutti e tre i computer hanno lo stesso codice, ma la data di riparazione è diversa. Ovvero si selezionano le terne di riparazioni che riguardano lo stesso computer ma che fanno riferimento a tre riparazioni diverse.
Delle tuple selezionate si estrae il codice del computer.

$$\text{T} := (\text{CR} \times (\rho_{c_1,d_1 \leftarrow \text{Codice,Data}}(\text{CR}) \times \rho_{c_2,d_2 \leftarrow \text{Codice,Data}}(\text{CR})))$$

$$\pi_{\text{Codice}}(\sigma_{\text{Codice}=c_1 \wedge \text{Codice}=c_2 \wedge \text{Data} \neq d_1 \wedge \text{Data} \neq d_2 \wedge d_1 \neq d_2}(\text{T}))$$

Interrogazione 2

> Determinare i codici dei computer di marca 'IBM' che hanno subito delle riparazioni esclusivamente per malfunzionamento alla scheda video (segnalato con *Guasto*='video').

Per risolvere questa interrogazione è importante notare che, se un computer ha subito solo riparazioni per malfunzionamenti alla scheda video, significa che non ha subito mai riparazioni per guasti di altro tipo. Si individuano allora i computer riparati per guasti di tipo non video e si sottraggono all'insieme di tutti i pc. A questo punto, è sufficiente restringere ulteriormente la scelta ai computer di marca 'IBM'.

Come primo passo selezioniamo tutte le riparazioni per guasti non alla scheda video dalla tabella che contiene l'elenco di tutte le riparazioni effettuate.

$$\text{NV} := \sigma_{\text{Guasto} \neq \text{'video'}}(\text{RIPARAZIONE})$$

A questo punto, si sottrae dall'insieme di tutti i computer, quelli che hanno subito riparazioni per guasti diversi da 'video'. È importante notare a questo proposito che la proiezione sull'attributo *CodiceComputer precede* la sottrazione. In caso contrario, anziché ottenere i computer che hanno subito *solo* riparazioni per guasti alla scheda video, si otterrebbero i

computer che hanno subito *anche* riparazioni per guasti alla scheda video. Infatti, si sottrarrebbe all'insieme di tutte le riparazioni quello delle riparazioni che non riguardano la scheda video, ottenendo tutti i record di riparazioni per guasti 'video', che non è il risultato richiesto.

$$R := \pi_{\text{CodiceComputer}}(\text{RIPARAZIONE}) - \pi_{\text{CodiceComputer}}(\text{NV})$$

Per finire, si selezionano i computer di marca 'IBM' e, tramite un'operazione di join, si compone il risultato con R, ottenendo il risultato richiesto.

$$\pi_{\text{Codice}}(\sigma_{\text{Marca='IBM'}}(\text{COMPUTER}) \bowtie_{\text{Codice=CodiceComputer}} R)$$

1.3.4 Sentenze

Sia dato lo schema relazionale:
PERSONA(CF, Nome, Cognome, DataNascita, CittàNascita, CittàResidenza)
CONDANNA(CFPersona, CFGiudice, Data, TipoReato, TipoCondanna, Durata)
GIUDICE(CF, Nome, Cognome, Tribunale, AnnoIngressoMagistratura)

Interrogazione 1

> Determinare il nome e cognome delle persone condannate nel 2002 all'ergastolo per uxoricidio e che hanno subito almeno una condanna per furto.

Si mostra in questo esercizio come è possibile comporre in diversi modi le condizioni che si vuole imporre.

Per prima cosa si selezionano i codici fiscali delle persone condannate per furto.

$$\text{LADRI} := \pi_{\text{CFPersona}}(\sigma_{\text{TipoReato='Furto'}}(\text{CONDANNA}))$$

Si selezionano poi le condanne inflitte, per qualsiasi ragione, durante l'anno 2002.

$$\text{C2002} := \sigma_{\text{Data}\geq\text{01-01-2002}\wedge\text{Data}\leq\text{31-12-2002}}(\text{CONDANNA})$$

Si selezionano poi, fra le condanne inflitte nel 2002, quelle che sono dovute ad uxoricidio e che hanno portato all'ergastolo.

$$\text{UXORICIDI} := \pi_{\text{CFPersona}}(\sigma_{\text{TipoReato='Uxoricidio'}\wedge\text{TipoCondanna='Ergastolo'}}(\text{C2002}))$$

Per individuare le persone che sono sia state condannate per furto, sia all'ergastolo nel 2002 per uxoricidio, è sufficiente intersecare il risultato delle due viste precedenti (LADRI, UXORICIDI). Si recuperano poi i dati anagrafici delle persone che soddisfano le condizioni specificate attraverso un join del risultato con la tabella PERSONA.

$$\pi_{\text{Nome,Cognome}}((\text{LADRI} \cap \text{UXORICIDI}) \bowtie_{\text{CFPersona=CF}} \text{PERSONA})$$

Interrogazione 2

> Determinare, per ciascun giudice del tribunale di Milano, la massima durata delle condanne che ha emesso.

Per risolvere questa interrogazione, una volta individuati tutti i giudici di Milano con le condanne inferte da ciascuno, per individuare la condanna di maggior durata, si selezionano quelle condanne per cui ne esiste una, inferta dallo stesso giudice, ma di durata più breve. Si toglie poi dall'insieme di tutte le condanne di tutti i giudici il sottoinsieme così calcolato.

Si selezionano quindi come prima cosa tutti i giudici che operano nel tribunale di Milano (MI).

$$\text{MI} := \sigma_{\text{Tribunale='Milano'}}(\text{GIUDICE})$$

Per ciascun giudice del tribunale di Milano si individuano le durate delle condanne da lui inferte, attraverso un'opportuna operazione di join.

$$\text{C} := \pi_{\text{CFGiudice,Durata}}(\text{MI} \bowtie_{\text{CF=CFGiudice}} \text{CONDANNA})$$

A questo punto si individuano tutte le coppie di giudici con la durata di una loro condanna e si selezionano le coppie che fanno riferimento allo stesso giudice e per cui la durata del primo elemento nella coppia è inferiore a quella del secondo. Si ottiene quindi, per ciascun giudice, l'elenco delle durate non massime. Attraverso un'operazione di sottrazione si ottiene quindi la durata massima di condanna per ciascun giudice del tribunale di Milano.

$$\text{C} - \pi_{\text{CFGiudice,Durata}}(\text{C} \bowtie_{\text{CFGiudice=CF} \wedge \text{Durata<d}} \rho_{\text{CF,d} \leftarrow \text{CFGiudice,Durata}}(\text{C}))$$

1.3.5 Missioni

Sia dato lo schema relazionale:

MISSIONE(Codice, Città, DataPartenza, Scopo, DurataPrevista)
AGENTE(Codice, Nome, Cognome, Specializzazione)
PARTECIPA(CodiceMissione, CodiceAgente, Ruolo)

Interrogazione 1

> Determinare il codice delle missioni che hanno la minima durata prevista.

Per individuare le missioni di durata minima, estraiamo per prima cosa gli attributi di interesse dalla tabella MISSIONI, cioè *Codice* e *DurataPrevista*.

$$\text{M} := \pi_{\text{Codice,DurataPrevista}}(\text{MISSIONE})$$

Si valutano quindi tutte le coppie di missioni, selezionando solo le missioni per cui ne esiste un'altra con durata inferiore. Dato che le missioni così selezionate sono quelle di durata non minima, il complemento rispetto all'insieme di tutte le missioni contiene l'elenco delle missioni di durata minima.

$$\pi_{\text{Codice}}(\text{M}) - \pi_{\text{Codice}}(\text{M} \bowtie_{\text{DurataPrevista}>\text{d}} \rho_{\text{c,d}\leftarrow\text{Codice,DurataPrevista}}(\text{M}))$$

Interrogazione 2

> Determinare il codice degli agenti che hanno partecipato con lo stesso ruolo ad almeno due missioni iniziate nell'anno 2002.

Selezioniamo per prima cosa le missioni la cui data di partenza si colloca nel 2002. In particolare, si estrae il codice di tali missioni.

$$\text{M2000} := \pi_{\text{Codice}}(\sigma_{\text{DataPartenza}\geq\text{01-01-2002}\wedge\text{DataPartenza}\leq\text{31-12-2002}}(\text{MISSIONE}))$$

Per individuare gli agenti che hanno partecipato a queste missioni, si fa il join della vista M2000 con la tabella PARTECIPA, in modo da avere tutte le informazioni di interesse. Per comodità nella scrittura dell'interrogazione, si computano due viste, P1 e P2, tra loro uguali, ma con gli attributi rinominati.

$$\text{P} := \pi_{\text{CodiceMissione,CodiceAgente,Ruolo}}(\text{PARTECIPA} \bowtie_{\text{CodiceMissione=Codice}} \text{M2000})$$

$$\text{P1} := \rho_{\text{cm1,ca1,r1}\leftarrow\text{CodiceMissione,CodiceAgente,Ruolo}}(\text{P})$$

$$\text{P2} := \rho_{\text{cm2,ca2,r2}\leftarrow\text{CodiceMissione,CodiceAgente,Ruolo}}(\text{P})$$

Tramite un'operazione di join, si calcolano tutte le coppie di agenti che hanno partecipato a missioni nel 2002, mantenendo solo le coppie riguardanti lo stesso agente, con lo stesso ruolo, ma due missioni diverse. Delle coppie selezionate, si mantiene solo il codice dell'agente come richiesto.

$$\pi_{\text{ca1}}(\text{P1} \bowtie_{\text{cm1}\neq\text{cm2}\wedge\text{r1=r2}\wedge\text{ca1=ca2}} \text{P2})$$

1.4 Esercizi con Soluzioni

1.4.1 Schemi, gradi e cardinalità

Esercizio 1

Date le seguenti relazioni:
R(a, b, c)
S(a, b)
T(c, d)
U(a, b, c)
V(a, b, d)
Z(a, d)

Indicare lo schema, il grado e le cardinalità minima e massima delle seguenti espressioni, considerando che nessuna delle relazioni date è vuota:

1. $R \bigcup S$
2. $\rho_{a,b \leftarrow c,d}(T) - S$
3. $S \bowtie T$
4. $\pi_{b}(\sigma_{a=1 \wedge a=7}(R))$
5. $S \times T$
6. $\pi_{a,d}(V) - Z$
7. $\rho_{c \leftarrow d}(V) \bigcup U$
8. $\sigma_{c=4}(Z)$
9. $V \bowtie Z$
10. $V \times Z$

Soluzione

	ESPRESSIONE	SCHEMA	GRADO	CARD. MIN	CARD. MAX
1	$R \bigcup S$	Non Applicabile: le due relazioni hanno diverso schema			
2	$\rho_{a,b \leftarrow c,d}(T) - S$	(a, b)	2	0	$\|T\|$
3	$S \bowtie T$	(a, b, c, d)	4	$\|S\| \cdot \|T\|$	$\|S\| \cdot \|T\|$
4	$\pi_{b}(\sigma_{a=1 \wedge a=7}(R))$	(b)	1	0	0
5	$S \times T$	(a, b, c, d)	4	$\|S\| \cdot \|T\|$	$\|S\| \cdot \|T\|$
6	$\pi_{a,d}(V) - Z$	(a, d)	2	0	$\|V\|$
7	$\rho_{c \leftarrow d}(V) \bigcup U$	(a, b, c)	3	$\max(\|V\|, \|U\|)$	$\|V\|+\|U\|$
8	$\sigma_{c=4}(Z)$	Non Applicabile: c non fa parte di Z			
9	$V \bowtie Z$	(a, b, d)	3	0	$\|V\|$
11	$V \times Z$	Non Applicabile: V e Z hanno attributi comuni			

La quarta espressione produce come risultato una relazione *vuota*. Non è infatti possibile che l'attributo a assuma il valore 1 e il valore 7 contemporaneamente nella stessa tupla.

La cardinalità massima del risultato della sesta espressione è pari a $|V|$, che è la cardinalità massima del primo operando della sottrazione, cioè $\pi_{a,d}(V)$.

La cardinalità massima della nona espressione è pari a $|V|$, e non $|V| \cdot |Z|$ perché gli attributi coinvolti nel join naturale sono a e d, ovvero tutti gli attributi che compongono Z, quindi ciascuna tupla in V viene abbinata con al più una tupla in Z.

Esercizio 2

Date le seguenti relazioni:
R(a, b, c)
S(a, b, c)
T(e, f)
U(a, b, c)
V(a, b, c)
Z(a, d)

Indicare lo schema, il grado e le cardinalità minima e massima delle seguenti espressioni, considerando che nessuna delle relazioni date è vuota:

1. $R \bigcup S$
2. $\rho_{b' \leftarrow b}(T)$
3. $\pi_{f}(T)$
4. $R - T$
5. $\pi_{b}(\sigma_{b=5}(R))$
6. $U - V$
7. $U \bigcap V$
8. $V \bowtie Z$
9. $\rho_{a',d' \leftarrow a,b}(U)$
10. $\sigma_{(a<100) \vee (a>90)}(U)$

Soluzione

	ESPRESSIONE	SCHEMA	GRADO	CARD. MIN	CARD. MAX
1	$R \bigcup S$	(a, b, c)	3	max(|R|, |S|)	|R|+|S|
2	$\rho_{b' \leftarrow b}(T)$	Non Applicabile: b non fa parte di T			
3	$\pi_{f}(T)$	(f)	1	1	|T|
4	$R - T$	Non Applicabile: le due relazioni hanno diverso schema			
5	$\pi_{b}(\sigma_{b=5}(R))$	(b)	1	0	1
6	$U - V$	(a, b, c)	3	0	|U|
7	$U \bigcap V$	(a, b, c)	3	0	min(|U|, |V|)
8	$V \bowtie Z$	(a, b, c, d)	4	0	$\|V\| \cdot \|Z\|$
9	$\rho_{a',d' \leftarrow a,b}(U)$	(a', d', c)	3	|U|	|U|
10	$\sigma_{(a<100) \vee (a>90)}(U)$	(a, b, c)	3	|U|	|U|

La cardinalità massima del risultato della quinta espressione è uno perché, la selezione per b=5 restituisce le tuple con solo questo valore per b, la proiezione quindi su b, eliminando i duplicati, restituisce solo il valore 5.

1.4.2 Equivalenza fra relazioni

Esercizio

Siano date le relazioni

R(a, b)

S(a, b, c)

indicare il risultato delle seguenti operazioni, motivando opportunamente il risultato in termini di equivalenza tra le relazioni.

1. $R \bowtie \emptyset$
2. $R \bowtie (\sigma_{C}(R))$
3. $R \cup (\sigma_{C}(R))$
4. $R \cup \pi_{a}(R)$
5. $R \bowtie (\pi_{a}(R))$
6. $S \cap (\sigma_{C}(S))$
7. $S \times \emptyset$
8. $(\sigma_{C1}(S)) \cup (\sigma_{C2}(S))$
9. $(\pi_{a, b}(S)) \bowtie S$
10. $(\pi_{a, b}(S)) \bowtie (\sigma_{C}(S))$

Soluzione

ESPRESSIONE	SOLUZIONE	MOTIVAZIONE
$R \bowtie \emptyset$	$\emptyset$	Nel secondo operando non ci sono tuple, quindi non è possibile generare nessuna coppia con le tuple nel primo operando.
$R \bowtie (\sigma_{C}(R))$	$\sigma_{C}(R)$	Dato che lo schema dei due operandi è lo stesso, l'operazione è equivalente a una intersezione fra gli stessi. Visto che il secondo operando è un sottoinsieme del primo, il risultato finale è composto dalle tuple del medesimo.
$R \bigcup (\sigma_{C}(R))$	R	Visto che il secondo operando è un sottoinsieme del primo, il risultato finale è composto dalle tuple dell'insieme più grande, cioè R.
$R \bigcup \pi_{a}(R)$	Non Applicabile: le due relazioni hanno diverso schema.	
$R \bowtie (\pi_{a}(R))$	R	Ciascuna tupla in R viene associata alla tupla in $\pi_{a}(R)$ con lo stesso valore per a. Dato che gli attributi comuni vengono riportati solo una volta nel risultato, la soluzione corrisponde a R stessa.
$S \bigcap (\sigma_{C}(S))$	$\sigma_{C}(S)$	Visto che il secondo operando è un sottoinsieme del primo, il risultato finale è composto dalle tuple del medesimo.
$S \times \emptyset$	$\emptyset$	Nel secondo operando non ci sono tuple, quindi non è possibile generare nessuna coppia con le tuple nel primo operando.
$(\sigma_{C1}(S)) \bigcup (\sigma_{C2}(S))$	$\sigma_{C1 \vee C2}(S)$	Anticipando l'unione rispetto alla selezione, le due condizioni di selezione vengono messe in or fra di loro perché sia le tuple che rispettano una condizione che quelle che rispettano l'altra fanno parte del risultato finale.
$(\pi_{a, b}(S)) \bowtie S$	S	Ciascuna tupla in S viene associata alla tupla in $\pi_{a,b}(S)$ con lo stesso valore per a e per b. Dato che gli attributi comuni vengono riportati solo una volta nel risultato, la soluzione corrisponde a S stessa.
$(\pi_{a, b}(S)) \bowtie (\sigma_{C}(S))$	$\sigma_{C}(S)$	Se si posticipa la selezione rispetto al join, l'argomento della selezione è l'espressione di cui al punto precedente.

1.4.3 Supermercato

Sia dato lo schema relazionale:
SUPERMERCATO(Codice, Nome, Telefono, Via, Città, CAP)
ACQUISTO(CodiceSupermercato, CodiceProdotto, CodiceTransazione)
PRODOTTO(Codice, Nome, Costo, CodiceScaffaleSupermercato)
TRANSAZIONE(Codice, Data, CodiceCliente, NomeCliente, CittàResidenzaCliente)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome e l'indirizzo dei supermercati presso i quali sono stati acquistate delle patate sia da clienti residenti a 'Milano' sia da clienti residenti a 'Crema'.
2. Determinare, per ogni transazione, il nome del prodotto più costoso.

Interrogazione 1

$$\text{RCREMA} := \sigma_{\text{CittàResidenzaCliente='Crema'}}(\text{TRANSAZIONE})$$

$$\text{RMILANO} := \sigma_{\text{CittàResidenzaCliente='Milano'}}(\text{TRANSAZIONE})$$

$$\text{V1} := \pi_{\text{CodiceProdotto,CodiceSupermercato}}(\text{ACQUISTO} \bowtie_{\text{CodiceTransizione=Codice}} \text{RMILANO})$$

$$\text{V2} := \pi_{\text{CodiceSupermercato}}(\sigma_{\text{Nome='patata'}}(\text{PRODOTTO} \bowtie_{\text{Codice=CodiceProdotto}} \text{V1}))$$

$$\text{V3} := \pi_{\text{CodiceProdotto,CodiceSupermercato}}(\text{ACQUISTO} \bowtie_{\text{CodiceTransizione=Codice}} \text{RCREMA})$$

$$\text{V4} := \pi_{\text{CodiceSupermercato}}(\sigma_{\text{Nome='patata'}}(\text{PRODOTTO} \bowtie_{\text{Codice=CodiceProdotto}} \text{V3}))$$

$$\text{SUPCRMI} := (\text{V2} \cap \text{V4})$$

$$\textit{RISULTATO} := \pi_{\text{Nome,Via,Città,CAP}}(\text{SUPCRMI} \bowtie_{\text{CodiceSupermercato=Codice}} \text{SUPERMERCATO})$$

Interrogazione 2

$$\text{PA} := \pi_{\text{CodiceProdotto,Nome,CodiceTransazione,Costo}}(\text{PRODOTTO} \bowtie_{\text{CodiceProdotto=Codice}} \text{ACQUISTO})$$

$$\text{PA2} := \rho_{\text{cp,n,ct,c}\leftarrow\text{CodiceProdotto,Nome,CodiceTransazione,Costo}}(\text{PA})$$

$$\text{PMC} := \text{PA} \bowtie_{\text{CodiceTransazione=ct}\wedge\text{Costo<c}} \text{PA2}$$

$RISULTATO :=$
$\pi_{\text{CodiceTransazione,CodiceProdotto,Nome}}(\text{PA}) - \pi_{\text{CodiceTransazione,CodiceProdotto,Nome}}(\text{PMC})$

1.4.4 Arredamento

Sia dato lo schema relazionale:

MOBILE(Codice, Linea, Dimensione, Colore, Costo)
CLIENTE(CF, Nome, Cognome, CittàResidenza)
ORDINE(CFCliente, CodiceMobile, DataOrdine, DataConsegna, ModalitàTrasporto)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare, per ciascuna linea, il mobile più costoso.
2. Determinare il codice fiscale dei clienti che hanno effettuato solo ordini con modalità di trasporto a 'carico del cliente'.

Interrogazione 1

$\text{M} := \pi_{\text{Codice,Linea,Costo}}(\text{MOBILE})$

$\text{MMC} := \text{M} \bowtie_{\text{Linea=l} \wedge \text{Costo<cos}} \rho_{\text{c,l,cos} \leftarrow \text{Codice,Linea,Costo}}(\text{M})$

$RISULTATO :=$
$\pi_{\text{Codice,Linea}}(\text{M}) - \pi_{\text{Codice,Linea}}(\text{MMC})$

Interrogazione 2

$\pi_{\text{CFCliente}}(\text{ORDINE}) - \pi_{\text{CFCliente}}(\sigma_{\text{ModalitàTrasporto} \neq \text{'carico del cliente'}}(\text{ORDINE}))$

1.4.5 Gommista

Sia dato lo schema relazionale:

PNEUMATICO(Codice, Materiale, Dimensione, CodiceProduttore)
PRODUTTORE(Codice, Nome, Via, Città)
ACQUISTO(CFCliente, CodicePneumatico, DataVendita, Costo)
CLIENTE(CF, Nome, Telefono, Via, Città)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il codice fiscale dei clienti che nel 2003 hanno acquistato almeno due volte lo stesso pneumatico.
2. Determinare il codice dei produttori che producono solo pneumatici realizzati con 'gomma secca' oppure con 'gomma sintetica'.

Interrogazione 1

$$\text{ACQUISTIPERIODO} := \sigma_{\text{DataVendita}\geq\text{01-01-2003} \wedge \text{DataVendita}\leq\text{31-12-2003}}(\text{ACQUISTO})$$

$$\text{V1} := \pi_{\text{CFCliente,CodicePneumatico,DataVendita}}(\text{ACQUISTIPERIODO})$$

$$\text{V1B} := \rho_{\text{CF,Codice,Data}\leftarrow\text{CFCliente,CodicePneumatico,DataVendita}}(\text{V1})$$

$$\textit{RISULTATO} := \pi_{\text{CFCliente}}(\text{V1} \bowtie_{\text{CFCliente=CF} \wedge \text{CodicePneumatico=Codice} \wedge \text{DataVendita} \neq \text{Data}} \text{V1B})$$

Interrogazione 2

$$\text{CP} := \pi_{\text{CodiceProduttore}}(\sigma_{\text{Materiale}\neq\text{'gomma secca'} \wedge \text{Materiale}\neq\text{'gomma sintetica'}}(\text{PNEUMATICO}))$$

$$\textit{RISULTATO} := \rho_{\text{CodiceProduttore}\leftarrow\text{Codice}}(\pi_{\text{Codice}}(\text{PRODUTTORE})) - \text{CP}$$

1.4.6 Calendari

Sia dato lo schema relazionale:

CALENDARIO(Codice, Prezzo, Tipo, AnnoCalendario, CodiceProduttore)
PRODUTTORE(Codice, Nome, Indirizzo)
FOTOGRAFIA(Numero, NomeFotografo, MeseDelCalendario, CodiceCalendario)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare, per ogni produttore, il codice del calendario di tipo 'da tavolo' di prezzo maggiore.
2. Determinare il codice dei calendari del 2003 che contengono solo fotografie scattate dal fotografo 'Sferrante'.

Interrogazione 1

$$\text{DT1} := \pi_{\text{Codice,Prezzo,CodiceProduttore}}(\sigma_{\text{Tipo='da tavolo'}}(\text{CALENDARIO}))$$

$$\text{DT2} := \rho_{\text{c,p,cp}\leftarrow\text{Codice,Prezzo,CodiceProduttore}}(\text{DT1})$$

$$\text{PRMIN} := \pi_{\text{CodiceProduttore,Codice}}(\text{DT1} \bowtie_{\text{CodiceProduttore=cp} \wedge \text{Prezzo<p}} \text{DT2})$$

$$\textit{RISULTATO} := \pi_{\text{CodiceProduttore,Codice}}(\text{DT1}) - \text{PRMIN}$$

Interrogazione 2

$\text{CAL03} := \pi_{\text{Codice}}(\sigma_{\text{AnnoCalendario=2003}}(\text{CALENDARIO}))$

$\text{CALNOSF} := \sigma_{\text{NomeFotografo}\neq\text{'Sferrante'}}(\text{FOTOGRAFIA}) \bowtie_{\text{CodiceCalendario=Codice}} \text{CAL03}$

RISULTATO :=
CAL03 − CALNOSF

1.4.7 Biscotti

Sia dato lo schema relazionale:
BISCOTTO(Codice, Nome, Forma, NomeCreatore, NomeDittaProduttrice)
INGREDIENTE(Codice, Nome, Tipologia)
COMPOSIZIONE(CodiceBiscotto, CodiceIngrediente, Quantità)

Nota: L'attributo *Quantità* è espresso in percentuale.

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il codice dei biscotti che contengono lo 0,3% di 'farina di riso' e lo 0,3% di 'zucchero integrale di canna'.
2. Determinare il nome dei biscotti di forma 'circolare' creati da 'MangiaTutti' e che contengono almeno l'1,6% di 'cioccolato in gocce'.

Interrogazione 1

$\text{CI} := \text{COMPOSIZIONE} \bowtie_{\text{CodiceIngrediente=Codice}} \text{INGREDIENTE}$

$\text{BF} := \pi_{\text{CodiceBiscotto}}(\sigma_{\text{Nome='farina di riso'}\wedge\text{Quantità=0.3}}(\text{CI}))$

$\text{BZ} := \pi_{\text{CodiceBiscotto}}(\sigma_{\text{Nome='zucchero di canna'}\wedge\text{Quantità=0.3}}(\text{CI}))$

RISULTATO :=
$\text{BF} \cap \text{BZ}$

Interrogazione 2

$\text{CI} := \text{COMPOSIZIONE} \bowtie_{\text{CodiceIngrediente=Codice}} \text{INGREDIENTE}$

$\text{BC} := \pi_{\text{CodiceBiscotto}}(\sigma_{\text{Nome='cioccolato in gocce'}\wedge\text{Quantità}\geq 1.6}(\text{CI}))$

RISULTATO :=
$\pi_{\text{Nome}}(\sigma_{\text{Forma='circolare'}\wedge\text{NomeCreatore='MangiaTutti'}}(\text{BISCOTTO} \bowtie_{\text{Codice=CodiceBiscotto}} \text{BC}))$

1.4.8 Rimborso

Sia dato lo schema relazionale:

FONDO(Codice, Importo, DataInizioErogazione, Scadenza, MatricolaTitolare)
DIPENDENTE(Matricola, Nome, Cognome, Posizione)
PARTECIPA(MatDipendente, CodiceFondo)
RIMBORSOSPESA(MatDipendente, CodiceFondo, Data, Importo, Motivazione)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare, per ciascun fondo, il nome e cognome dei dipendenti che hanno chiesto l'ultimo rimborso spesa.
2. Determinare la matricola dei dipendenti che hanno chiesto almeno un rimborso spesa superiore a 3000 Euro e che non hanno mai chiesto un rimborso con motivazione 'viaggio aereo'.

Interrogazione 1

$\text{RS0} := \pi_{\text{MatDipendente,CodiceFondo,Data}}(\text{RIMBORSOSPESA})$

$\text{RS1} := \rho_{\text{m,c,d}\leftarrow\text{MatDipendente,CodiceFondo,Data}}(\text{RS0})$

$\text{R} := \pi_{\text{MatDipendente,CodiceFondo,Data}}(\text{RS0} \bowtie_{\text{CodiceFondo=c}\wedge\text{Data<d}} \text{RS1})$

RISULTATO :=
$\pi_{\text{Nome,Cognome}}((\text{RS0} - \text{R}) \bowtie_{\text{MatDipendente=Matricola}} \text{DIPENDENTE})$

Interrogazione 2

$\text{DIP0} := \sigma_{\text{Importo>3000}}(\text{RIMBORSOSPESA})$

$\text{DIP1} := \sigma_{\text{Motivazione='viaggio aereo'}}(\text{RIMBORSOSPESA})$

RISULTATO :=
$\pi_{\text{MatDipendente}}(\text{DIP0}) - \pi_{\text{MatDipendente}}(\text{DIP1})$

1.4.9 Conferenze

Sia dato lo schema relazionale:
PERSONA(CF, Nome, Cognome, Affiliazione, Indirizzo, CittàResidenza)
CONFERENZA(Nome, Data, Indirizzo, Città)
ISCRIZIONE(NomeConferenza, DataConferenza, CFPersona, Data, Importo)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome ed il cognome delle persone che si sono iscritte ad almeno due conferenze diverse (indipendentemente dall'anno in cui le conferenze si sono tenute).
2. Determinare il nome delle conferenze del 2002 per le quali tutti gli iscritti vivono in una città diversa da quella dove ha avuto luogo la conferenza.

Interrogazione 1

$$\text{I0} := \pi_{\text{NomeConferenza,CF,Nome,Cognome}}(\text{ISCRIZIONE} \bowtie_{\text{CFPersona=CF}} \text{PERSONA})$$

$$\text{I1} := \rho_{\text{nc,cf}_1\text{,n,c}\leftarrow\text{NomeConferenza,CF,Nome,Cognome}}(\text{I0})$$

$$\textit{RISULTATO} := \pi_{\text{Nome,Cognome}}(\text{I0} \bowtie_{\text{CF=cf}_1 \wedge \text{NomeConferenza} \neq \text{nc}} \text{I1})$$

Interrogazione 2

$$\text{IP} := \pi_{\text{CF,NomeConferenza,DataConferenza,cr}}(\rho_{\text{cr}\leftarrow\text{CittàResidenza}}(\text{PERSONA}) \bowtie_{\text{CF=CFPersona}} \text{ISCRIZIONE})$$

$$\text{IP1} := \sigma_{\text{Data}\geq\text{01-01-2002} \wedge \text{Data}\leq\text{31-12-2002}}(\text{CONFERENZA}) \bowtie_{\text{Nome=NomeConferenza} \wedge \text{Data=DataConferenza}} \text{IP}$$

$$\textit{RISULTATO} := \pi_{\text{NomeConferenza}}(\text{IP1}) - \pi_{\text{NomeConferenza}}(\sigma_{\text{Città=cr}}(\text{IP1}))$$

1.4.10 Donut

Sia dato lo schema relazionale:
TIPODONUT(Codice, Nome, DataPrimaProduzione)
CLIENTE(CF, Nome, Cognome, CittàResidenza, DataNascita)
ACQUISTO(CFCliente, CodiceDonut, DataAcquisto)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome e cognome dei clienti che hanno acquistato donut esattamente nello stesso giorno della loro prima produzione.

2. Determinare il codice dei donut acquistati solo da clienti residenti a San Francisco e nati dopo il 1971.

Interrogazione 1

$A := \text{TIPODONUT} \bowtie_{\text{Codice=CodiceDonut}} \text{ACQUISTO}$

$A1 := \pi_{\text{CFCliente}}(\sigma_{\text{DataPrimaProduzione=DataAcquisto}}(A))$

RISULTATO :=
$\pi_{\text{Nome,Cognome}}(A1 \bowtie_{\text{CFCliente=CF}} \text{CLIENTE})$

Interrogazione 2

$NA := \pi_{\text{CF}}(\sigma_{\text{DataNascita<01-01-1972} \vee \text{CittàResidenza} \neq \text{`San Francisco'}}(\text{CLIENTE}))$

RISULTATO :=
$\pi_{\text{CodiceDonut}}(\text{ACQUISTO}) - \pi_{\text{CodiceDonut}}(NA \bowtie_{\text{CFCliente=CF}} \text{ACQUISTO})$

1.4.11 Ambulatorio

Sia dato lo schema relazionale:

MEDICO(Codice, Nome, Cognome, Specializzazione, Città, Telefono)
PAZIENTE(CodiceSSN, Nome, Cognome, DataNascita, Città, Telefono)
VISITA(CodiceMedico, CodicePaziente, Data, Diagnosi, CodiceMedicinale)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome, cognome e codice sanitario del paziente più vecchio.

2. Determinare il nome e cognome dei medici che hanno visitato tutti i pazienti.

Interrogazione 1

$P := \rho_{\text{cod,n,c,d} \leftarrow \text{CodiceSSN,Nome,Cognome,DataNascita}}(\text{PAZIENTE})$

$P2 := \rho_{\text{cod}_1,\text{n}_1,\text{c}_1,\text{d}_1 \leftarrow \text{cod,n,c,d}}(P)$

RISULTATO :=
$\pi_{\text{cod,n,c}}(P) - \pi_{\text{cod,n,c}}(P \bowtie_{\text{d}>\text{d}_1} P2)$

Interrogazione 2

$$P := \rho_{n,c,d,ct,t \leftarrow Nome,Cognome,DataNascita,Città,Telefono}(PAZIENTE)$$

$$TC := \pi_{Codice,Nome,Cognome,CodiceSSN}(MEDICO \times P)$$

$$MP := \pi_{Codice,Nome,Cognome,CodiceSSN}(MEDICO \bowtie_{Codice=CodiceMedico} \rho_{CodiceSSN \leftarrow CodicePaziente}(VISITA))$$

RISULTATO :=
$$\pi_{Nome,Cognome}(MP) - \pi_{Nome,Cognome}(TC - MP)$$

1.4.12 Orologi

Sia dato lo schema relazionale:
PRODUTTORE(Nome, Nazione)
NEGOZIO(Codice, Città)
OROLOGIO(Codice, Tipo, PrezzoProd, AnnoProduzione, NomeProduttore)
FORNITURA(NomeProduttore, CodiceNegozio, CodiceOrologio, PrezzoVendita)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare le città in cui ci sono negozi che vendono 'cronografi' con produttore di sede svizzera.
2. Determinare i produttori che servono tutti i negozi di orologi di Milano.

Interrogazione 1

$$PS := \sigma_{Nazione='Svizzera'}(PRODUTTORE)$$

$$C := \sigma_{Tipo='cronografo'}(OROLOGIO)$$

$$CS := \pi_{Codice}(PS \bowtie_{Nome=NomeProduttore} C)$$

RISULTATO :=
$$\pi_{Città}(NEGOZIO \bowtie_{Codice=CodiceNegozio} FORNITURA \bowtie_{CodiceOrologio=c} \rho_{c \leftarrow Codice}(CS))$$

Interrogazione 2

$$TC := \rho_{NomeProduttore \leftarrow Nome}(\pi_{Nome,Codice}(PRODUTTORE \times \sigma_{Città='Milano'}(NEGOZIO)))$$

$$FM := \pi_{NomeProduttore,Codice}(FORNITURA \bowtie_{CodiceNegozio=Codice} \sigma_{Città='Milano'}(NEGOZIO))$$

RISULTATO :=
$\pi_{\text{NomeProduttore}}(\text{FM}) - \pi_{\text{NomeProduttore}}(\text{TC} - \text{FM})$

1.4.13 Mondiali

Sia dato lo schema relazionale:

SQUADRA(Nazione, AnnoEdizione, CT, IDCapitano)
GIOCATORE(ID, Cognome, Nome)
CONVOCAZIONE(IDGiocatore, Nazione, AnnoEdizione, Ruolo, NumGoalSegnati)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare nome e cognome dei capitani che sono stati convocati con il ruolo di attaccante.
2. Determinare, per ciascuna edizione del mondiale, il cognome del giocatore capocannoniere, cioè del giocatore che ha segnato il maggior numero di goal.

Interrogazione 1

$\text{C} := \rho_{\text{id,n,ae} \leftarrow \text{IdGiocatore,Nazione,AnnoEdizione}}(\sigma_{\text{Ruoolo='attaccante'}}(\text{CONVOCAZIONE}))$

RISULTATO :=
$\pi_{\text{Nome,Cognome}}(\text{C}) \bowtie_{\text{id=IdCapitano} \wedge \text{n=Nazione} \wedge \text{ae=AnnoEdizione}} \text{SQUADRA}$

Interrogazione 2

$\text{CO} := \pi_{\text{IdGiocatore,NumGoalSegnati,AnnoEdizione}}(\text{CONVOCAZIONE})$

$\text{CO1} := \rho_{\text{Id1,NGoal1,Anno1} \leftarrow \text{IdGiocatore,NumGoalSegnati,AnnoEdizione}}(\text{CO})$

$\text{CO2} := \rho_{\text{Id2,NGoal2,Anno2} \leftarrow \text{IdGiocatore,NumGoalSegnati,AnnoEdizione}}(\text{CO})$

$\text{CC} := \text{CO1} - \pi_{\text{Id1,NGoal1,Anno1}}(\text{CO1} \bowtie_{\text{Anno1=Anno2} \wedge \text{NGoal1<NGoal2}} \text{CO2})$

RISULTATO :=
$\pi_{\text{Cognome}}(\text{GIOCATORE} \bowtie_{\text{Id=Id1}} \text{CC})$

1.4.14 SportClub

Sia dato lo schema relazionale:

SOCIO(NumTessera, Classe, Nome, Cognome, AnnoNascita)
CAMPO(Codice, Superficie, Lunghezza, Larghezza)
PRENOTAZIONE(Progressivo, NumTesseraSocio, CodiceCampo, Data, Ora)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il numero delle tessera dei soci di classe 'Platinum' che hanno prenotato tutti i campi.
2. Determinare il progressivo delle prenotazioni che riguardano i campi di lunghezza massima.

Interrogazione 1

$\text{SP} := \pi_{\text{NumTessera}}(\sigma_{\text{Classe='Platinum'}}(\text{SOCIO}))$

$\text{C} := \pi_{\text{Codice}}(\text{CAMPO})$

$\text{TUTTI} := \text{SP} \times \text{C}$

$\text{VERI} := \pi_{\text{NumTesseraSocio,CodiceCampo}}(\text{PRENOTAZIONE})$

$\text{VERI1} := \rho_{\text{NumTessera,Codice}\leftarrow\text{NumTesseraSocio,CodiceCampo}}(\text{VERI}) \bowtie \text{SP}$

RISULTATO :=
$\pi_{\text{NumTessera}}(\text{VERI1}) - \pi_{\text{NumTessera}}(\text{TUTTI} - \text{VERI1})$

Interrogazione 2

$\text{C} := \rho_{\text{C,S,Lu,La}\leftarrow\text{Codice,Superficie,Lunghezza,Larghezza}}(\text{CAMPO})$

$\text{CLMIN} := \text{CAMPO} \bowtie_{\text{Lunghezza}<\text{Lu}} \text{C}$

$\text{CPG} := \pi_{\text{Codice}}(\text{CAMPO}) - \pi_{\text{Codice}}(\text{CLMIN})$

RISULTATO :=
$\pi_{\text{Progressivo}}(\text{PRENOTAZIONE} \bowtie_{\text{CodiceCampo=Codice}} \text{CPG})$

1.4.15 Yogurt

Sia dato lo schema relazionale:

YOGURT(Codice, NomeTipo, Confezione, Peso, PrezzoConsigliato)
INGREDIENTE(Codice, Nome, ValoreNutritivo, CostoAlKg, Tipologia)
COMPOSIZIONE(CodiceYogurt, CodiceIngrediente, Quantità)

Nota: Il peso e la quantità sono espressi in grammi.

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il codice di tutti gli ingredienti impiegati nella produzione di yogurt confezionati in 'vasetti' da 500 grammi.
2. Determinare il codice degli ingredienti che vengono utilizzati nella produzione di un solo tipo di yogurt.

Interrogazione 1

$$\text{VAS} := \sigma_{\text{Peso=500} \wedge \text{Confezione='vasetto'}}(\text{YOGURT})$$

$$\text{I} := \pi_{\text{CodiceYogurt}}(\text{VAS} \bowtie_{\text{Codice=CodiceYogurt}} \text{COMPOSIZIONE})$$

$$\textit{RISULTATO} := \pi_{\text{CodiceI}}(\text{I} \bowtie_{\text{CodiceIngrediente=Codice}} \text{INGREDIENTE})$$

Interrogazione 2

$$\text{YC1} := \pi_{\text{NomeTipo,CodiceIngrediente}}(\text{YOGURT} \bowtie_{\text{Codice=CodiceYogurt}} \text{COMPOSIZIONE})$$

$$\text{YC2} := \rho_{\text{nt,ci} \leftarrow \text{NomeTipo,CodiceIngrediente}}(\text{YC1})$$

$$\textit{RISULTATO} := \pi_{\text{CodiceIngrediente}}(\text{YC1}) - \pi_{\text{CodiceIngrediente}}(\text{YC1} \bowtie_{\text{NomeTipo} \neq \text{nt} \wedge \text{CodiceIngrediente=ci}} \text{YC2})$$

1.4.16 Aereo

Sia dato lo schema relazionale:

VOLO(Codice, Data, NomeCompagnia, Partenza, Destinazione, DistanzaMiglia)
PRENOTAZIONE(CodiceVolo, Data, CFCliente)
CLIENTE(CF, Cognome, Nome)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome delle compagnie aeree che effettuano il volo più lungo.
2. Determinare il nome della compagnia aerea che ha programmato almeno due voli nello stesso giorno con diversa destinazione.

Interrogazione 1

$V1 := \pi_{\text{NomeCompagnia,DistanzaMiglia}}(\text{VOLO})$

$V2 := \rho_{\text{n,d}\leftarrow\text{NomeCompagnia,DistanzaMiglia}}(V1)$

RISULTATO :=
$\pi_{\text{NomeCompagnia}}(\ V1 - V1 \bowtie_{\text{DistanzaMiglia<d}} V2)$

Interrogazione 2

$V := \rho_{\text{cv,d,nc,p,dest,dm}\leftarrow\text{Codice,Data,NomeCompagnia,Partenza,Destinazione,DistanzaMiglia}}($
VOLO)

RISULTATO :=
$\pi_{\text{NomeCompagnia}}(\text{VOLO} \bowtie_{\text{Codice}\neq\text{c}\wedge\text{Data}=\text{d}\wedge\text{NomeCompagnia}=\text{nc}\wedge\text{Destinazione}\neq\text{dest}} V)$

1.4.17 Fattorino

Sia dato lo schema relazionale:

FATTORINO(Codice, Nome, Cognome)
CLIENTE(CF, Nome, Cognome, Città)
CONSEGNA(CodiceFattorino, CFCliente, Data, Mezzo)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare quali ditte hanno effettuato consegne usando solo furgoni.
2. Determinare il nome e cognome dei fattorini che effettuato almeno due consegne per lo stesso cliente.

Interrogazione 1

$\pi_{\text{NomeDitta}}(\text{CONSEGNA}) - \pi_{\text{NomeDitta}}(\sigma_{\text{Mezzo}\neq\text{'furgone'}}(\text{CONSEGNA}))$

Interrogazione 2

$\text{V} := \rho_{\text{f,c,dt,Mz}\leftarrow\text{CodiceFattorino,CFCliente,Data,Mezzo}}(\text{CONSEGNA})$

$\text{C} := \pi_{\text{CodiceFattorino}}(\text{CONSEGNA} \bowtie_{\text{CFCliente=c}\wedge\text{Data}\neq\text{dt}\wedge\text{CodiceFattorino=f}} \text{V})$

RISULTATO :=
$\pi_{\text{Nome,Cognome}}(\text{C} \bowtie_{\text{CodiceFattorino=Codice}} \text{FATTORINO})$

1.4.18 Stabilimento

Sia dato lo schema relazionale:

STABILIMENTO(Codice, Dimensione, Ubicazione)
ARTICOLO(Nome, Tipo, Prezzo)
PRODUZIONE(NomeArticolo, CodiceStabilimento, Data, Quantità)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il codice e la dimensione di tutti gli stabilimenti in cui si producono articoli di nome 'sedia' e articoli di nome 'tavolo'.

2. Determinare il codice degli stabilimenti che producono almeno due articoli diversi.

Interrogazione 1

$\text{S} := \pi_{\text{CodiceStabilimento}}(\sigma_{\text{NomeArticolo='Sedia'}}(\text{PRODUZIONE}))$

$\text{T} := \pi_{\text{CodiceStabilimento}}(\sigma_{\text{NomeArticolo='Tavolo'}}(\text{PRODUZIONE}))$

RISULTATO :=
$\pi_{\text{Codice,Dimensione}}((\text{S} \cap \text{T}) \bowtie_{\text{CodiceStabilimento=Codice}} \text{STABILIMENTO})$

Interrogazione 2

$\text{PR} := \rho_{\text{na,cs}\leftarrow\text{NomeArticolo,CodiceStabilimento}}(\text{PRODUZIONE})$

RISULTATO :=
$\pi_{\text{CodiceStabilimento}}(\text{PRODUZIONE} \bowtie_{\text{NomeArticolo}\neq\text{na}\wedge\text{CodiceStabilimento=cs}} \text{PR})$

1.4.19 Esami

Sia dato lo schema relazionale:

GRUPPO(Nome, DataConsegna, VotoProgetto)
STUDENTE(Matricola, Nome, Cognome, AnnoDiCorso, NomeGruppo)
ESAMEBD(MatricolaStudente, VotoScritto, VotoOrale)

Nota: L'assenza di voto viene indicata con il valore zero.

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome dei gruppi per i quali tutti i membri hanno ottenuto un voto superiore a 28 nello scritto e non inferiore a 25 nell'orale.
2. Determinare matricola, nome e cognome degli studenti che hanno consegnato il progetto prima del 2005 e che hanno già sostenuto lo scritto, ma non hanno ancora sostenuto l'orale.

Interrogazione 1

$\text{STD} := \pi_{\text{Matricola}}(\sigma_{\text{VotoScritto}\leq 28 \vee \text{VotoOrale}<25}(\text{ESAMEBD}))$

$\text{BAD} := \pi_{\text{NomeGruppo}}(\text{STD} \bowtie_{\text{Matricola=MatricolaStudente}} \text{STUDENTE})$

$\textit{RISULTATO} :=$
$\pi_{\text{NomeGruppo}}(\text{STUDENTE} \bowtie_{\text{Matricola=MatricolaStudente}} \text{ESAMEBD}) - \text{BAD}$

Interrogazione 2

$\text{NV} := \sigma_{\text{VotoScritto}\neq 0 \wedge \text{VotoOrale}=0}(\text{ESAMEBD})$

$\text{GP} := \rho_{\text{NomeGruppo}\leftarrow\text{Nome}}(\sigma_{\text{DataConsegna}\leq \text{31-12-2004}}(\text{GRUPPO}))$

$\textit{RISULTATO} :=$
$\pi_{\text{Matricola,Nome,Cognome}}(\text{NV} \bowtie_{\text{Matricola=MatricolaStudente}} \text{STUDENTE} \bowtie \text{GP})$

1.4.20 Telefoni

Sia dato lo schema relazionale:

TELEFONO(Codice, Marca, Tipo, Prezzo)
CLIENTE(CF, Nome, Cognome, Indirizzo, DataNascita)
NEGOZIO(Codice, Nome, Indirizzo, Telefono)
ACQUISTA(CFCliente, CodiceTel, CodiceNeg, DataAcquisto)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare il nome e cognome dei clienti che hanno acquistato un telefono di tipo 'cellulare' costato meno di 200 euro.
2. Determinare tutti i negozi per i quali esiste un cliente che ha acquistato almeno due telefoni.

Interrogazione 1

$\text{CELL} := \pi_{\text{Codice}}(\sigma_{\text{Tipo='cellulare'} \wedge \text{Prezzo}<200}(\text{TELEFONO}))$

RISULTATO :=
$\pi_{\text{Nome,Cognome}}(\text{CELL} \bowtie_{\text{Codice=CodiceTel}} \text{ACQUISTA} \bowtie_{\text{CF=CFCliente}} \text{CLIENTE})$

Interrogazione 2

$\text{AC} := \rho_{\text{CF1,CT1,CN1} \leftarrow \text{CFCliente,CodiceTel}}(\pi_{\text{CFCliente,CodiceTel,CodiceNeg}}(\text{ACQUISTA}))$

$\text{CLI} := \pi_{\text{CodiceNeg}}(\text{ACQUISTA} \bowtie_{\text{CFCliente=CF1} \wedge \text{CodiceTel} \neq \text{CT1}} \text{AC})$

RISULTATO :=
$\text{CLI} \bowtie_{\text{Codice=CodiceNeg}} \text{NEGOZIO}$

1.4.21 Università

Sia dato lo schema relazionale:

STUDENTE(Matricola, Nome, Cognome, DataNascita)
CORSOLAUREA(Codice, Nome, Descrizione)
INSEGNAMENTO(Codice, Nome, NumeroCrediti, Semestre)
ISCRIZIONE(CodiceCorsoDiLaurea, MatricolaStudente, AnnoImmatricolazione)
MANIFESTOCL(CodiceCorsoDiLaurea, CodiceInsegnamento, Fondamentale)

Si richiede di scrivere in Algebra Relazionale le seguenti interrogazioni:

1. Determinare i corsi di laurea che includono sia l'insegnamento 'Basi di Dati' sia l'insegnamento 'Sistemi Operativi' ma non l'insegnamento 'Intelligenza Artificiale'.
2. Determinare, per ogni corso di laurea, il nome e cognome dello studente più giovane.

Interrogazione 1

$\text{BD1} := \sigma_{\text{Nome='Basi di Dati'}}(\text{INSEGNAMENTO})$

$\text{BD} := \pi_{\text{CodiceCorsoDiLaurea}}(\text{MANIFESTOCL} \bowtie_{\text{CodiceInsegnamento=Codice}} \text{BD1})$

$\text{SO1} := \sigma_{\text{Nome='Sistemi Operativi'}}(\text{INSEGNAMENTO})$

SO $:= \pi_{\text{CodiceCorsoDiLaurea}}(\text{MANIFESTOCL} \bowtie_{\text{CodiceInsegnamento=Codice}} \text{SO1})$

IA1 $:= \sigma_{\text{Nome='Intelligenza Artificiale'}}(\text{INSEGNAMENTO})$

IA $:= \pi_{\text{CodiceCorsoDiLaurea}}(\text{MANIFESTOCL} \bowtie_{\text{CodiceInsegnamento=Codice}} \text{IA1})$

C $:= (\text{BD} \cap \text{SO}) - \text{IA}$

RISULTATO $:=$
C $\bowtie_{\text{CodiceCorsoDiLaurea=Codice}}$ CORSOLAUREA

Interrogazione 2

SI $:=$ STUDENTE $\bowtie_{\text{Matricola=MatricolaStudente}}$ ISCRIZIONE

STUD $:= \pi_{\text{Matricola,Nome,Cognome,DataNascita,CodiceCorsoDiLaurea}}(\text{SI})$

STUD1 $:= \rho_{\text{m,n,c,dn,cdl}\leftarrow\text{Matricola,Nome,Cognome,DataNascita,CodiceCorsoDiLaurea}}(\text{STUD})$

OS $:=$ STUD $\bowtie_{\text{CodiceCorsoDiLaurea=cdl}\wedge\text{DataNascita<dn}}$ STUD1

YS $:=$ STUD $- \pi_{\text{Matricola,Nome,Cognome,DataNascita,CodiceCorsoDiLaurea}}(\text{OS})$

RISULTATO $:=$
$\pi_{\text{Nome,Cognome,CodiceCorsoDiLaurea}}(\text{YS})$

Capitolo 2

Structured Query Language: SQL

Questo Capitolo è dedicato al *Data Manipulation Language* di SQL, cioè al linguaggio che permette di eseguire interrogazioni per recuperare e modificare informazioni contenute in una base di dati. Il capitolo espone, brevemente, anche il concetto di *viste*, che fanno parte del *Data Definition Language*. Per il linguaggio SQL, si fa riferimento allo standard SQL99.

2.1 Il linguaggio SQL

Per illustrare il linguaggio SQL si usa la notazione *BNF* (Backus Normal Form).

< >	Le parentesi angolari sono elementi sintattici che richiamano altre definizioni.
[ ]	Le parentesi quadre rappresentano un elemento opzionale.
{ }	Le parentesi graffe raggruppano elementi all'interno della definizione.
|	La barra verticale separa elementi sintattici in alternativa fra loro.
...	I puntini di sospensione indicano che l'elemento precedente può comparire un numero imprecisato di volte.

2.1.1 Interrogazioni

Le interrogazioni SQL sono caratterizzate dalla seguente sintassi:

```
SELECT    [DISTINCT]{ * | <nome attributo>
            [ { , <nome attributo> }... ] }
FROM      {<nome tabella> [ { , <nome tabella> }... |
            { JOIN <nome tabella> ON <condizione> }... ] }
[WHERE    <condizione>]
[GROUP BY {<nome attributo> [ { , <nome attributo> }... ]}]
[HAVING   <condizione>]
[ORDER BY {<nome attributo> [ASC | DESC]
            [ { , <nome attributo> [ASC | DESC] }... ]}]
```

Clausola SELECT La clausola `SELECT` elenca gli attributi che faranno parte della tabella risultato. In particolare, è possibile utilizzare il valore $*$ per indicare che verrano restituiti tutti gli attributi delle tabelle coinvolte nell'interrogazione.
Si noti che <`nome attributo`> ha la seguente forma sintattica:

```
[ <nome tabella> . ] <attributo> [ [AS] <alias> ]
```

dove <`nome tabella`> è il nome della relazione cui l'attributo appartiene ed <`alias`> è l'intestazione che l'attributo acquisisce nella tabella risultato.
La parola chiave `DISTINCT` permette la rimozione delle tuple duplicate dal risultato che, contrariamente all'*Algebra Relazionale*, non sono automaticamente eliminate.

Clausola FROM La clausola `FROM` elenca le tabelle utilizzate per l'esecuzione dell'interrogazione. In assenza di `JOIN`, le tabelle sono composte attraverso il prodotto cartesiano. Viceversa, in presenza di `JOIN`, le tuple ottenute dal prodotto cartesiano vengono filtrate dalla condizione specificata dopo la parola chiave `ON`; proprio per questo *ogni interrogazione* basata sul `JOIN` può essere scritta in modo del tutto equivalente utilizzando solo le clausole `FROM` e `WHERE`. La dichiarazione delle tabelle rimane nel `FROM`, ma le condizioni del join vengono spostate nella clausola `WHERE`.
Si noti che <`nome tabella`> ha la seguente forma sintattica:

```
<tabella> [ [AS] <alias> ]
```

dove <`alias`> diventa un sinonimo per <`tabella`>.

Clausola WHERE La clausola `WHERE` è opzionale e contiene una condizione (anche complessa) che permette di definire un filtro sul risultato dell'interrogazione. Si noti che la condizione viene valutata per ciascuna tupla che concorre al risultato: se è soddisfatta, la tupla viene aggiunta al risultato, in caso contrario viene scartata.

Clausola GROUP BY La clausola `GROUP BY` è opzionale e contiene un elenco di attributi sulla base dei valori dei quali vengono raggruppate le tuple. Tutte e sole le tuple aventi lo stesso valore per gli attributi specificati fanno parte dello stesso gruppo e danno luogo ad *una sola tupla* nel risultato.
Si noti che, in presenza della clausola `GROUP BY`, la clausola `SELECT` è ristretta ai soli attributi argomento della `GROUP BY`.

Clausola HAVING La clausola `HAVING` è opzionale e viene di solito inserita solo se è presente la clausola `GROUP BY`. La clausola `HAVING` è l'equivalente della clausola `WHERE` per i gruppi, infatti consente di definire una condizione che viene valutata per ciascun gruppo di tuple che concorre al risultato.

Clausola ORDER BY La clausola `ORDER BY` è opzionale e serve per l'ordinamento delle tuple nella tabella risultato. L'ordinamento segue l'ordine lessicografico dei valori negli attributi elencati, nell'ordine in cui sono riportati nella clausola. Le opzioni `ASC` e `DESC` determinano se l'ordinamento deve essere crescente (di default) o decrescente, attributo per attributo.

Condizioni Una condizione è composta da uno o più letterali legati attraverso gli operatori logici `AND` e `OR`. Un letterale può essere considerato formato da due elementi: un operatore ed uno o più operandi. I letterali più comuni fanno uso di:

- operatori di confronto (<, >, <=, >=, =, <>), assumono la forma *operando1 operatore operando2*;
- operatore di negazione `NOT`, assume la forma *`NOT` operando*, con operando booleano;
- operatore di intervallo [`NOT`] `BETWEEN`, assume la forma *operando1 [`NOT`] `BETWEEN` operando2 `AND` operando3*, questo operatore valuta se *operando1* è compreso (non compreso) fra *operando2* ed *operando3* (estremi inclusi);
- operatore per il confronto di stringhe `LIKE`con un semplice supporto delle espressioni regolari, assume la forma *operando1 `LIKE` operando2*; a differenza dell'operatore di uguaglianza, è possibile utilizzare i metacaratteri _ e % che rappresentano rispettivamente un singolo carattere e una stringa di lunghezza arbitraria;
- operatore per la gestione dei valori nulli `IS` [`NOT`] `NULL`, assume la forma *operando `IS` [`NOT`] `NULL`* che restituisce vero oppure falso a seconda che operando sia (non sia) nullo.

Operatori di Aggregazione Gli operatori di aggregazione sono delle funzioni che, dato un gruppo di tuple (eventualmente generato dalla clausola `GROUP BY`), restituiscono un valore caratteristico del gruppo. I principali operatori di aggregazione sono di seguito elencati:

- `AVG(<attributo>)`: restituisce il valore medio per `<attributo>` calcolato sul gruppo;
- `COUNT( * | <attributo> )`: restituisce il numero di valori di `<attributo>` per il gruppo o il numero di tuple (*) nello stesso;
- `MAX(<attributo>)`: restituisce il massimo valore per `<attributo>` all'interno del gruppo;
- `MIN(<attributo>)`: restituisce il minimo valore per `<attributo>` all'interno del gruppo;
- `SUM(<attributo>)`: restituisce la somma su `<attributo>` calcolata sul gruppo.

Si noti che, `<attributo>` può essere preceduto da una fra le due parole chiave `DISTINCT` e `ALL` (di default). La prima indica che i duplicati non vengono considerati nella valutazione della funzione, mentre la seconda li considera.

2.1.2 Interrogazioni Nidificate

Per esprimere interrogazioni più complesse di quelle viste fino ad ora, lo standard SQL permette di definire delle interrogazioni (dette *nidificate*) come operandi all'interno dei letterali delle condizioni. A tale scopo, oltre agli operatori già citati, è possibile utilizzare altri operatori: `ANY`, `SOME`, `ALL`, [`NOT`] `IN` e [`NOT`] `EXISTS`.

La sintassi, per `ANY`, `SOME` e `ALL`, è la seguente:

```
<espressione> <operatore di confronto>
   {ANY | SOME | ALL} <interrogazione>
```

dove espressione può essere sia un attributo sia un valore. Il letterale risulta verificato se:

- `ANY`: almeno una delle tuple <`tupla`> risultato di <`interrogazione`> *rende vero* il letterale <`espressione`> <`operatore`> <`tupla`>;
- `SOME`: è un *alias* per `ANY`;
- `ALL`: tutte le tuple <`tupla`> risultato di <`interrogazione`> *rendono vero* il letterale <`espressione`> <`operatore`> <`tupla`>.

La sintassi, per [`NOT`] `IN`, è la seguente:

```
<espressione> [NOT] IN <interrogazione>
```

e il letterale risulta verificato se il valore di <`espressione`> compare almeno una volta (non compare mai) nel risultato di <`interrogazione`>.
La sintassi, per [`NOT`] `EXISTS`, è la seguente:

```
[NOT] EXISTS <interrogazione>
```

e il letterale risulta verificato se <`interrogazione`> restituisce almeno una tupla (ha risultato vuoto).
Si noti che un alias definito nell'interrogazione principale può essere impiegato come variabile nelle interrogazioni nidificate.

2.1.3 Interrogazioni Insiemistiche

Oltre che attraverso la nidificazione, due (o più) interrogazioni possono essere composte attraverso operatori insiemistici. Una *interrogazione insiemistica* si presenta nella seguente forma:

```
<interrogazione A>
{UNION | INTERSECT | EXCEPT} [ALL]
<interrogazione B>
```

dove <`interrogazione A`> e <`interrogazione B`> sono generiche interrogazioni SQL con il vincolo di restituire un risultato dello stesso grado e con la stessa struttura, ovvero devono avere lo stesso numero di attributi e il dominio di attributi, che occupano la stessa posizione all'interno della clausola `SELECT` di <`interrogazione A`> e <`interrogazione B`>, deve essere uguale. Il nome delle colonne del risultato di una interrogazione insiemistica è *dipendente dall'implementazione* nel caso le interrogazioni che la compongono non restituiscano esattamente attributi aventi lo stesso nome.
In dettaglio gli operatori insiemistici sono:

- `UNION`: restituisce l'unione dei risultati delle interrogazioni operandi;

- `INTERSECT`: restituisce l'intersezione dei risultati delle interrogazioni operandi;
- `EXCEPT`: restituisce la differenza insiemistica fra il risultato della prima interrogazione operando e quello della seconda.

Di default, questi tre operatori non mantengono i duplicati, per mantenerli è possibile specificare la parola chiave `ALL` dopo l'operatore.

2.1.4 Interrogazioni con Viste

Oltre alle tradizionali tabelle presenti nella base di dati, è possibile eseguire interrogazioni anche sulle *viste*. Una vista è il risultato di una interrogazione, cui è stato dato un nome. La sintassi per la definizione di una vista è la seguente:

```
CREATE VIEW <nome vista>
  [ (<nome attributo> [ { , <nome attributo> }... ]) ]
AS <interrogazione>
```

dove <`interrogazione`> è un'interrogazione SQL corretta il cui risultato ha lo stesso numero di attributi specificati dopo il nome della vista.

2.2 Esercizi Svolti e Commentati

In questa sezione vengono presentati alcuni esercizi svolti.

2.2.1 I Pittori

Sia dato lo schema relazionale:

PITTORE(Nome, DataNascita, DataMorte, LuogoNascita)
QUADRO(Titolo, NomePittore, NomeMuseo, Data)
MUSEO(Nome, Indirizzo, Città)

Nota: Se un pittore risulta ancora in vita allora la sua data di morte sarà impostata a `NULL`.

Interrogazione 1

> Determinare il nome dei pittori nati a Parigi dopo il 1968 che hanno dipinto almeno tre quadri esposti al 'MOMA' di New York.

Analizzando il testo dell'interrogazione è possibile notare che:

- Il risultato dell'interrogazione deve essere il *nome dei pittori*; di conseguenza l'argomento della clausola `SELECT` deve essere l'attributo *Nome* della tabella PITTORE.
- Le condizioni poste dalla richiesta riguardano il *luogo di nascita* (Parigi), la *data di nascita* (1968), il *nome del museo* (MOMA) e la *città* dello stesso (New York); tutte queste condizioni possono venire valutate sulla singola tupla e riguardano un suo attributo, quindi devono essere inserite nella clausola `WHERE`.
- La richiesta presenta inoltre una condizione (*almeno tre quadri*) che non può essere valutata sulla singole tuple, ma riguarda gruppi di tuple costruiti tramite la clausola `GROUP BY`; quindi la condizione di *aver dipinto almeno tre quadri* deve essere inserita nella clausola `HAVING`.

L'interrogazione risultante è la seguente:

```
SELECT   Pittore.Nome
FROM     Pittore JOIN Quadro ON Pittore.Nome = NomePittore
                 JOIN Museo ON Museo.Nome = NomeMuseo
WHERE    DataNascita > 31-12-1968
         AND LuogoNascita = 'Parigi'
         AND Museo.Nome = 'MOMA'
         AND Città = 'New York'
GROUP BY Pittore.Nome
HAVING   COUNT(*) >= 3
```

L'interrogazione precedente può essere scritta senza il `JOIN`. L'interrogazione risultante è:

```
SELECT   Pittore.Nome
FROM     Pittore, Quadro, Museo
WHERE    Pittore.Nome = NomePittore
         AND Museo.Nome = NomeMuseo
         AND DataNascita > 31-12-1968
         AND LuogoNascita = 'Parigi'
         AND Museo.Nome = 'MOMA'
         AND Città = 'New York'
GROUP BY Pittore.Nome
HAVING   COUNT(*) >= 3
```

Interrogazione 2

> Determinare i nomi dei musei che contengono almeno un quadro dipinto fra il 1300 ed il 1600 ma non contengono quadri dipinti da Leonardo da Vinci.

Analizzando il testo dell'interrogazione è importante notare che: le condizioni relative a *dipinto fra* e *dipinti da* non possono essere valutate sulla singola tupla poiché riguardano una proprietà che deve essere soddisfatta dai Musei. Questo si evince dalle parole chiavi (presenti nel testo) *contengono almeno un quadro* e *ma non contengono quadri*. Si procede, quindi, valutando separatamente le due condizioni e componendo i risultati attraverso l'operatore insiemistico di differenza (`EXCEPT`).

L'interrogazione risultante è la seguente:

```
SELECT   DISTINCT NomeMuseo
FROM     Quadro
WHERE    Data BETWEEN 01-01-1300 AND 31-12-1600
         EXCEPT
SELECT   DISTINCT NomeMuseo
FROM     Quadro
WHERE    NomePittore = 'Leonardo da Vinci'
```

In modo equivalente è possibile usare una interrogazione nidificata, con l'operatore `NOT IN`; ponendo come condizione che il nome del museo non sia tra quelli che espongono quadri dipinti da 'Leonardo da Vinci'. Si ottiene quindi:

```
SELECT   DISTINCT NomeMuseo
FROM     Quadro
WHERE    Data >= 01-01-1300
         AND Data <= 31-12-1600
         AND NomeMuseo NOT IN (
             SELECT   NomeMuseo
             FROM     Quadro
             WHERE    NomePittore = 'Leonardo da Vinci' )
```

Interrogazione 3

> Determinare i nomi dei musei che espongono almeno un quadro di un pittore ancora in vita.

Analizzando il testo dell'interrogazione si nota che: l'unica condizione da valutare riguarda la data di morte del pittore.
Si ottiene, quindi, la seguente interrogazione:

```
SELECT   DISTINCT NomeMuseo
FROM     Pittore JOIN Quadro ON Nome = NomePittore
WHERE    DataMorte IS NULL
```

2.2.2 Le Feste

Sia dato lo schema relazionale:
FESTA(Codice, Costo, NomeRistorante)
REGALO(NomeInvitato, CodiceFesta, Regalo)
INVITATO(Nome, Indirizzo, Telefono)

Interrogazione 1

> Determinare, per ogni festa, il nome dell'invitato più generoso, ovvero dell'invitato che ha portato il maggior numero di regali.

Analizzando il testo dell'interrogazione si nota che è presente una richiesta di *massimo relativo* (per *ogni festa* l'invitato che ha portato il *maggior numero* di regali). Per determinare il numero di regali di ciascun invitato a ciascuna festa, si raggruppano le tuple della tabella REGALO per *NomeInvitato* e *CodiceFesta* e quindi si contano. Quindi individuare l'invitato più generoso diventa un confronto fra quanti regali ogni invitato ha portato ad una determinata festa e il numero di regali portati da ciascuno degli altri invitati. Questa condizione può essere espressa attraverso una interrogazione nidificata, come la seguente:

```
SELECT   R.CodiceFesta, R.NomeInvitato
FROM     Regalo AS R
GROUP BY R.CodiceFesta, R.NomeInvitato
HAVING   COUNT(*) >=  ALL (
             SELECT   COUNT(*)
             FROM     Regalo AS R0
             WHERE    R0.CodiceFesta = R.CodiceFesta
             GROUP BY R0.NomeInvitato )
```

Si noti che la condizione *R0.CodiceFesta = R.CodiceFesta* è necessaria per restringere il confronto tra gli invitati della stessa festa; infatti *R.CodiceFesta* fa riferimento al gruppo di tuple della interrogazione più esterna, mentre *R0.CodiceFesta* fa riferimento a quella più interna.

Interrogazione 2

Determinare il nome degli invitati che hanno partecipato alla festa più costosa.

Analizzando il testo dell'interrogazione si nota che è presente una richiesta di *massimo assoluto*. Per individuare la festa più costosa si confronta il costo di ciascuna festa con quella di tutte le altre feste attraverso una interrogazione nidificata con l'operatore >= ALL. Vengono poi selezionati gli invitati a tali feste attraverso l'operatore IN.

```
SELECT   DISTINCT NomeInvitato
FROM     Regalo
WHERE    CoficeFesta IN (
             SELECT   Codice
             FROM     Festa
             WHERE    Costo>= ALL (
                          SELECT   Costo
                          FROM     Festa ))
```

Interrogazione 3

Determinare il codice delle feste dove almeno un invitato ha portato tre regali.

Analizzando il testo dell'interrogazione si nota che la richiesta comporta la valutazione di una condizione che non è valutabile sulla singola tupla. È quindi necessario utilizzare la clausola HAVING, dopo aver raggruppato correttamente le tuple di interessa, cioè in base alla festa e al nome dell'invitato.

```
SELECT   DISTINCT R.CodiceFesta
FROM     Regalo AS R
GROUP BY R.NomeInvitato, R.CodiceFesta
HAVING   COUNT(*) >= 3
```

2.2.3 Autobus

Sia dato lo schema relazionale:
AUTISTA(CF, Nome, Cognome, Età, NumAutobus)
PERCORRE(NumAutobus, NomeStrada)
STRADA(NomeStrada, Lunghezza, Pedonale)

Si noti che: *Pedonale* è un attributo booleano che vale *true* se la strada è di tipo pedonale; *false* altrimenti.

Interrogazione 1

> Determinare le coppie di autisti che guidano lo stesso autobus.

Per ottenere le coppie di autisti è necessario eseguire un *prodotto cartesiano* della tabella AUTISTA con se stessa, eliminando le tuple che non rispettano il vincolo richiesto (*guidare lo stesso autobus*). Per poter distinguere le due copie della tabella è necessario utilizzare gli *alias*. Al fine di evitare la duplicazione delle coppie di autisti (che si presenterebbero solo in posizione invertita) si aggiunge la condizione sull'ordine lessicografico dei codici fiscali nel `WHERE`.

```
SELECT   A1.*, A2.*
FROM     Autista AS A1 JOIN Autista AS A2 ON A1.NumAutobus = A2.NumAutobus
WHERE    A1.CF < A2.CF
```

Interrogazione 2

> Determinare gli autobus (NumAutobus) che percorrono tutte le strade pedonali, ed eventualmente anche altre strade.

Affinché un autobus *percorra tutte* le strade pedonali, *non deve esistere* una strada pedonale *che l'autobus non percorre*. Si mostra di seguito come ricavare l'interrogazione a partire dalla considerazione appena fatta.

1. Si individuano tutte gli autobus...

   ```
   SELECT   P.NumAutobus
   FROM     Percorre AS P
   ```

2. ... per cui non esiste una strada pedonale...

```
SELECT   P.NumAutobus
FROM     Percorre AS P
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Strada AS S0
             WHERE    S0.Pedonale = true )
```

3. ... che l'autobus non percorre.

```
SELECT   P.NumAutobus
FROM     Percorre AS P
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Strada AS S0
             WHERE    S0.Pedonale = true
                      AND NOT EXISTS (
                          SELECT   *
                          FROM     Percorre AS P1,
                          WHERE    P.NumAutobus = P1.NumAutobus
                                   AND P1.NomeStrada = S0.Nome ))
```

Si noti che l'interrogazione più interna fa riferimento alle due che la contengono attraverso gli *alias* P e S0.

Una soluzione alternativa consiste nel contare per ogni autobus il numero di strade pedonali percorse, confrontando il risultato ottenuto con il numero globale di strade pedonali. L'interrogazione risultante è la seguente:

```
SELECT   NumAutobus
FROM     Percorre JOIN Strada ON NomeStrada = Nome
WHERE    Pedonale = true
GROUP BY NumAutobus
HAVING   COUNT(*) =  (
             SELECT   COUNT(*)
             FROM     Strada
             WHERE    Pedonale = true )
```

Interrogazione 3

> Conteggiare gli autobus che percorrono solo strade pedonali (non devono necessariamente percorrerle tutte).

Per poter contare gli autobus che percorrono solo strade pedonali, è necessario aver prima individuato l'elenco delle linee che rispettano questa condizione. A tale fine, partendo dall'elenco di tutte gli autobus, si eliminano quelli che *percorrono almeno una strada non pedonale*, attraverso l'operatore insiemistico di differenza (`EXCEPT`) Dato che non è possibile applicare l'operatore di aggregazione `COUNT` direttamente al risultato di un'interrogazione insiemistica, è necessario l'utilizzo di una *vista*.

```
CREATE VIEW StradeSoloPedonali (NumAutobus) AS
            SELECT   DISTINCT NumAutobus
            FROM     Percorre JOIN Strada ON NomeStrada = Nome
                     EXCEPT
            SELECT   DISTINCT NumAutobus
            FROM     Percorre JOIN Strada ON NomeStrada = Nome
            WHERE    Pedonale = false

SELECT   COUNT(*)
FROM     StradeSoloPedonali
```

2.2.4 Supermercati

Sia dato lo schema relazionale:

SUPERMERCATO(Codice, Nome, Telefono, Via, Città, CAP)
ACQUISTO(CodiceSupermercato, CodiceProdotto, CodiceTransazione)
PRODOTTO(Codice, Nome, Costo, CodiceScaffaleSupermercato)
TRANSAZIONE(Codice, Data, CodiceCliente, NomeCliente, CittàResidenzaCliente)

Interrogazione 1

> Determinare le coppie (codice prodotto, codice cliente) tali per cui il cliente ha acquistato il prodotto sempre nello stesso supermercato.

Per individuare i prodotti che un cliente ha acquistato sempre e solo in un determinato supermercato, è necessario individuare prima tutte le coppie prodotto-cliente, per poi eliminare quelle che non rispettano la condizione. Quindi, per ciascuna transazione contenente una coppia prodotto-cliente legata ad un supermercato, si controlla che non esista un'altra transazione riguardante la stessa coppia ma relativa ad un diverso supermercato.

```
SELECT   DISTINCT A.CodiceProdotto, T.CodiceCliente
FROM     Acquisto AS A JOIN Transazione AS T
          ON A.CodiceTransazione = T.Codice
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Acquisto AS A0 JOIN Transazione AS T0
                       ON A0.CodiceTransazione = T0.Codice
             WHERE    T.CodiceCliente = T0.CodiceCliente
                      AND A.CodiceProdotto = A0.CodiceProdotto
                      AND A.CodiceSupermercato <> A0.CodiceSupermercato )
```

Interrogazione 2

> Determinare il codice delle transazioni che contengono almeno tre prodotti disposti sullo stesso scaffale.

Per poter valutare il numero di prodotti nella stessa transazione e disposti sullo stesso scaffale, si fa uso della clausola `GROUP BY`. Il conteggio viene poi eseguito nell'`HAVING`, essendo relativo ai singoli gruppi, attraverso la funzione di aggregazione `COUNT`.

```
SELECT   DISTINCT CodiceTransazione
FROM     Acquisto JOIN Prodotto ON CodiceProdotto = Codice
GROUP BY CodiceTransazione, CodiceScaffaleSupermercato
HAVING   COUNT(*) >= 3
```

Interrogazione 3

> Determinare quanto il signor 'Andrea Rossi' ha speso nel supermercato 'Famila' di 'Crema'.

Si noti che per poter valutare tutte le condizioni, è necessario eseguire un join fra tutte le tabelle, per poi eseguire la somma sui costi.

```
SELECT   SUM(Costo)
FROM     Acquisto JOIN Prodotto
          ON CodiceProdotto = Prodotto.Codice
                JOIN Transazione
          ON CodiceTransazione = Transazione.Codice
                JOIN Supermercato
          ON CodiceSupermercato = Supermercato.Codice
WHERE    NomeCliente = 'Andrea Rossi'
         AND Supermercato.Nome = 'Famila'
         AND Supermercato.Città = 'Crema'
```

2.3 Esercizi con Soluzione

2.3.1 Ambulatorio

Sia dato lo schema relazionale:

MEDICO(Codice, Nome, Cognome, Specializzazione, Città, Telefono)
PAZIENTE(CodiceSSN, Nome, Cognome, DataNascita, Città, Telefono)
VISITA(CodiceMedico, CodicePaziente, Data, Diagnosi, CodiceMedicinale)
MEDICINALE(Codice, Nome, PrincipioAttivo, Prezzo)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare nome e cognome dei pazienti, nati dopo il 1980, che assumono il medicinale 'Aulin' per curare l'emicrania.

2. Determinare il codice sanitario dei pazienti che nel 2005 hanno speso più di 150 euro per curare l'anemia (si consideri la visita contestuale all'acquisto dei medicinali prescritti).

3. Determinare nome e cognome dei medici cardiologi che non hanno mai visitato pazienti della loro città per problemi di ipertensione.

Interrogazione 1

```
SELECT   DISTINCT Paziente.Nome, Paziente.Cognome
FROM     Paziente JOIN Visita ON CodiceSSN = CodicePaziente
                  JOIN Medicinale ON CodiceMedicinale = Codice
WHERE    DataNascita >= 01-01-1981
         AND Medicinale.Nome = 'Aulin'
         AND Diagnosi = 'emicrania'
```

Interrogazione 2

```
SELECT   CodicePaziente
FROM     Visita JOIN Medicinale ON CodiceMedicinale = Codice
WHERE    Diagnosi = 'anemia'
         AND Data BETWEEN 01-01-2005 AND 31-12-2005
GROUP BY CodicePaziente
HAVING   SUM(Prezzo) > 150
```

Interrogazione 3

```
SELECT   Nome, Cognome
FROM     Medico
WHERE    Specializzazione = 'cardiologia'
         AND Codice NOT IN (
             SELECT   Codice
             FROM     Paziente JOIN Visita ON CodiceSSN = CodicePaziente
                              JOIN Medico ON CodiceMedico = Codice
             WHERE    Paziente.Città = Medico.Città
                      AND Diagnosi = 'ipertensione' )
```

2.3.2 Biscotti

Sia dato lo schema relazionale:
BISCOTTO(Codice, Nome, Forma, NomeCreatore, NomeDittaProduttrice)
INGREDIENTE(Codice, Nome, Tipologia)
COMPOSIZIONE(CodiceBiscotto, CodiceIngrediente, Quantità)

Nota: L'attributo *Tipologia* e un attributo che può assumere solo i valori 'naturale' e 'artificiale'. L'attributo *Quantità* è espresso in percentuale.

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il codice dei biscotti di forma triangolare che sono prodotti da 'biSCOTTI&Figli' e che sono composti solo da ingredienti naturali.
2. Determinare il codice dei biscotti che sono prevalentemente composti da ingredienti artificiali (cioè tali per cui il numero di ingredienti artificiali è prevalente).
3. Determinare i nomi dei creatori dei biscotti di forma 'esagonale' che sono composti almeno al 70% da ingredienti di tipo 'naturale'.

Interrogazione 1

```
SELECT   Codice
FROM     Biscotto
WHERE    Forma = 'triangolo'
         AND NomeDittaProduttrice = 'biSCOTTI&Figli'
         EXCEPT
SELECT   CodiceBiscotto
FROM     Composizione JOIN Ingrediente ON CodiceIngrediente = Codice
WHERE    Tipologia <> 'naturale'
```

Soluzione Alternativa

```
SELECT   Codice
FROM     Biscotto
WHERE    Forma = 'triangolo'
         AND NomeDittaProduttrice = 'biSCOTTI&Figli'
         AND Codice NOT IN (
              SELECT   CodiceBiscotto
              FROM     Composizione, Ingrediente
              WHERE    CodiceIngrediente = Codice
                       AND Tipologia <> 'naturale' )
```

Interrogazione 2

```
CREATE VIEW Artificiale (Codice, NumArtificiali) AS
            SELECT   CodiceBiscotto, COUNT(*)
            FROM     Composizione JOIN Ingrediente
                      ON CodiceIngrediente = Codice
            WHERE    Tipologia = 'artificiale'
            GROUP BY CodiceBiscotto

CREATE VIEW Naturale (Codice, NumNaturali) AS
            SELECT   CodiceBiscotto, COUNT(*)
            FROM     Composizione JOIN Ingrediente
                      ON CodiceIngrediente = Codice
            WHERE    Tipologia = 'naturale'
            GROUP BY CodiceBiscotto

SELECT   A.Codice
FROM     Artificiale AS A JOIN Naturale AS N ON A.Codice = N.Codice
WHERE    A.NumArtificiali > N.NumNaturali
```

Interrogazione 3

```
SELECT   NomeCreatore
FROM     Biscotto JOIN Composizione ON Biscotto.Codice = CodiceBiscotto
                  JOIN Ingrediente ON CodiceIngrediente = Ingrediente.Codice
WHERE    Forma = 'esagonale'
         AND Tipologia = 'naturale'
GROUP BY Biscotto.Codice
HAVING   SUM(Quantità) >= 0.70
```

2.3.3 Conferenze

Sia dato lo schema relazionale:

PERSONA(CF, Nome, Cognome, Affiliazione, Indirizzo, CittàResidenza)
CONFERENZA(Nome, Data, Indirizzo, Città)
ISCRIZIONE(NomeConferenza, DataConferenza, CFPersona, Data, Importo)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare le conferenze (nome e data) che hanno un numero di iscrizioni superiore a 100.

2. Determinare le conferenze (nome e data) per le quali tutti gli iscritti appartengono alla stessa città.

3. Determinare le conferenze (nome e data) cui si sono iscritte persone che risiedono in almeno 20 città diverse.

Interrogazione 1

```
SELECT   NomeConferenza, DataConferenza
FROM     Iscrizione
GROUP BY NomeConferenza, DataConferenza
HAVING   COUNT(*) > 100
```

Interrogazione 2

```
SELECT   DISTINCT I.NomeConferenza, I.DataConferenza
FROM     Iscrizione AS I JOIN Persona AS P ON I.CFPersona = P.CF
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Iscrizione AS I0 JOIN Persona AS P0
                       ON I0.CFPersona = P0.CF
             WHERE    I.NomeConferenza = I0.NomeConferenza
                      AND I.DataConferenza = I0.DataConferenza
                      AND P.CittàResidenza <> P0.CittàResidenza )
```

Interrogazione 3

```
SELECT   NomeConferenza, DataConferenza
FROM     Iscrizione JOIN Persona ON CFPersona = CF
GROUP BY NomeConferenza, DataConferenza
HAVING   COUNT(DISTINCT Persona.CittàResidenza) >= 20
```

2.3.4 Donut

Sia dato lo schema relazionale:

DONUT(Codice, Nome, Crema, Glassa, Decorazione, DataPrimaProduzione)
CLIENTE(CF, Nome, Cognome, CittàResidenza, DataNascita)
ACQUISTO(CFCliente, CodiceDonut, DataAcquisto)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il codice dei donut con glassa al 'cioccolato' che sono stati prodotti per la prima volta nel 1999 e che sono stati acquistati da più di 2000 diversi clienti.

2. Determinare il nome e cognome dei clienti che hanno acquistato donut farciti di crema 'pasticciera' e decorati con 'zucchero a granelli' ma che non hanno mai acquistato donut con glassa al 'cioccolato'.

3. Determinare il nome, cognome e il codice fiscale dei clienti che hanno acquistato almeno tre differenti tipi di Donut.

Interrogazione 1

```
SELECT   Codice
FROM     Donut JOIN Acquisto ON Codice = CodiceDonut
WHERE    Glassa = 'cioccolato'
         AND DataPrimaProduzione BETWEEN 01-01-1999 AND 31-12-1999
GROUP BY Codice
HAVING   COUNT(DISTINCT CFCliente) > 2000
```

Interrogazione 2

```
SELECT   DISTINCT C.Nome, C.Cognome
FROM     Donut JOIN Acquisto ON Codice = CodiceDonut
               JOIN Cliente AS C ON CFCliente = CF
WHERE    Crema = 'pasticciera'
         AND Decorazione = 'zucchero a granelli'
         AND NOT EXISTS (
             SELECT   *
             FROM     Donut JOIN Acquisto ON Codice = CodiceDonut
             WHERE    CFCliente = C.CF
                      AND Glassa = 'cioccolato' )
```

Soluzione Alternativa

```
SELECT   DISTINCT Cliente.Nome, Cliente.Cognome
FROM     Donut, Acquisto, Cliente
WHERE    Codice = CodiceDonut
         AND CFCliente = CF
         AND Crema = 'pasticciera'
         AND Decorazione = 'zucchero a granelli'
         AND CF NOT IN (
             SELECT   CFCliente
             FROM     Acquisto, Donut
             WHERE    CodiceDonut = Codice
                      AND Glassa = 'cioccolato' )
```

Interrogazione 3

```
SELECT   DISTINCT C.Nome, C.Cognome, C.CF
FROM     Cliente JOIN Acquisto ON CF = CFCliente
GROUP BY C.Nome, C.Cognome, C.CF
HAVING   COUNT(DISTINCT CodiceDonut) >= 3
```

2.3.5 Missioni

Sia dato lo schema relazionale:

MISSIONE(Codice, Città, DataPartenza, Scopo, DurataPrevista)
AGENTE(Codice, Nome, Cognome, Specializzazione)
PARTECIPA(CodiceMissione, CodiceAgente, Ruolo)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il codice degli agenti specializzati in 'travestimenti' che hanno partecipato a missioni svolte nella città di Washington.

2. Determinare il codice delle missioni a cui hanno partecipato almeno due agenti specializzati in 'spionaggio' e due agenti con il ruolo di 'infiltrati' che non devono essere specializzati in 'spionaggio'.

3. Determinare le città delle missioni eseguite a scopo di spionaggio, dove partecipa almeno un agente infiltrato.

Interrogazione 1

```
SELECT   DISTINCT Agente.Codice
FROM     Agente JOIN Partecipa ON Agente.Codice = CodiceAgente
                JOIN Missione ON Missione.Codice = CodiceMissione
WHERE    Specializzazione = 'travestimento'
         AND Città = 'Washington'
```

Interrogazione 2

```
SELECT   DISTINCT CodiceMissione
FROM     Agente JOIN Partecipa ON Agente.Codice = CodiceAgente
WHERE    Specializzazione = 'spionaggio'
GROUP BY CodiceMissione
HAVING   COUNT(*) >= 2
         INTERSECT
SELECT   DISTINCT CodiceMissione
FROM     Agente JOIN Partecipa ON Agente.Codice = CodiceAgente
WHERE    Specializzazione <> 'spionaggio'
         AND Ruolo = 'infiltrato'
GROUP BY CodiceMissione
HAVING   COUNT(*) >= 2
```

Interrogazione 3

```
SELECT   DISTINCT Città
FROM     Missione JOIN Partecipa ON Codice = CodiceMissione
WHERE    Scopo = 'Spionaggio'
         AND Ruolo = 'Infiltrato'
```

2.3.6 Orologi

Sia dato lo schema relazionale:

PRODUTTORE(Nome, Nazione)
NEGOZIO(Codice, Città)
OROLOGIO(Codice, Tipo, PrezzoProd, AnnoProduzione, NomeProduttore)
FORNITURA(NomeProduttore, CodiceNegozio, CodiceOrologio, PrezzoVendita)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il nome dei produttori italiani che producono orologi di tipo 'cronografo'.

2. Determinare, per ciascun produttore, il numero di negozi servito in ogni città.

3. Determinare i negozi che, nella loro città, sono gli unici che vendono orologi prodotti dalla società 'Acme'.

Interrogazione 1

```
SELECT   Nome
FROM     Produttore JOIN Orologio ON Nome = NomeProduttore
WHERE    Nazione = 'Italia'
         AND Tipo = 'cronografo'
```

Interrogazione 2

```
SELECT   NomeProduttore, Città, COUNT(DISTINCT Codice)
FROM     Fornitura JOIN Negozio ON CodiceNegozio = Codice
GROUP BY NomeProduttore, Città
```

Interrogazione 3

```
SELECT   DISTINCT N.Codice
FROM     Fornitura AS F JOIN Negozio AS N ON F.CodiceNegozio = N.Codice
WHERE    F.NomeProduttore = 'Acme'
         AND NOT EXISTS (
             SELECT   *
             FROM     Fornitura AS F0 JOIN Negozio AS N0
                       ON F0.CodiceNegozio = N0.Codice
             WHERE    F.CodiceNegozio <> F0.CodiceNegozio
                      AND N.Città = N0.Città
                      AND F0.NomeProduttore = 'Acme' )
```

2.3.7 Rimborso

Sia dato lo schema relazionale:

FONDO(Codice, Importo, DataInizioErogazione, Scadenza, MatrAmministratore)
DIPENDENTE(Matricola, Nome, Cognome, Posizione)
PARTECIPA(MatrDipendente, CodiceFondo)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il nome e cognome dei dipendenti che non sono amministratori di fondi erogati a partire dal 2003.

2. Determinare i fondi (si richiede di restituire il codice del fondo e la scadenza) per i quali tutti i partecipanti sono anche amministratori di fondi.

3. Determinare il nome e il cognome degli amministratori del fondo più vecchio (cioè, quello con data di inizio erogazione più vecchia).

Interrogazione 1

```
SELECT   Nome, Cognome
FROM     Dipendente AS D
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Fondo
             WHERE    D.Matricola = MatrAmministratore
                      AND DataInizioErogazione >= 01-01-2003 )
```

Soluzione Alternativa

```
SELECT   Nome, Cognome
FROM     Dipendente
WHERE    Matricola NOT IN (
             SELECT   MatrAmministratore
             FROM     Fondo
             WHERE    DataInizioErogazione >= 01-01-2003 )
```

Interrogazione 2

```
SELECT   Codice, Scadenza
FROM     Fondo
WHERE    Codice NOT IN (
             SELECT   CodiceFondo
             FROM     Partecipa
             WHERE    MatrDipendente NOT IN (
                          SELECT   MatrAmministratore
                          FROM     Fondo))
```

Interrogazione 3

```
CREATE VIEW FondiVecchi (Codice, MatrAmministratore) AS
            SELECT   Codice
            FROM     Fondo
            WHERE    Codice NOT IN (
                         SELECT   F1.Codice
                         FROM     Fondo AS F1, Fondo AS F2
                         WHERE    F1.DataInizioErogazione
                                  >F1.DataInizioErogazione )


SELECT   DISTINCT Nome, Cognome
FROM     Dipendente JOIN FondiVecchi ON Matricola = MatrAmministratore
```

Soluzione Alternativa

```
SELECT   DISTINCT Nome, Cognome
FROM     Dipendente JOIN Fondo ON Matricola = MatrAmministratore
WHERE    DataInizioErogazione <=  ALL (
             SELECT   DataInizioErogazione
             FROM     Fondo )
```

2.3.8 Sentenze

Sia dato lo schema relazionale:

PERSONA(CF, Nome, Cognome, DataNascita, CittàNascita, CittàResidenza)
CONDANNA(CFPersona, DataCondanna, Tipo, CittàReato, AnnoReato, Durata)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il numero medio annuo di condanne per 'omicidio' che si sono verificati nella città di Milano.
2. Determinare il codice fiscale, nome e cognome di tutte le persone nate nel 1971 che non hanno mai subito una condanna per un reato compiuto nella città di nascita (o che non sono mai state condannate).
3. Determinare le città ove si compiono più reati.

Interrogazione 1

```
CREATE VIEW Omicidio (Anno, Numero) AS
            SELECT   AnnoReato, COUNT(*)
            FROM     Condanna
            WHERE    TipoReato = 'omicidio'
                     AND CittàReato = 'Milano'
            GROUP BY AnnoReato

SELECT   AVG(Numero)
FROM     Omicidio
```

Interrogazione 2

```
SELECT   CF, Nome, Cognome
FROM     Persona
WHERE    DataNascita BETWEEN 01-01-1971 AND 31-12-1971
         AND NOT EXISTS (
             SELECT   *
             FROM     Condanna
             WHERE    CFPersona = CF
                      AND CittàNascita = CittàReato )
```

Soluzione Alternativa

```
SELECT   CF, Nome, Cognome
FROM     Persona
WHERE    DataNascita BETWEEN 01-01-1971 AND 31-12-1971
         AND CF NOT IN (
             SELECT   CF
             FROM     Persona, Condanna
             WHERE    CFPersona = CF
                      AND CittàNascita = CittàReato )
```

Interrogazione 3

```
SELECT   CittàReato
FROM     Condanna
GROUP BY CittàReato
HAVING   COUNT(*) >=  ALL (
             SELECT   COUNT(*)
             FROM     Condanna
             GROUP BY CittàReato
             )
```

2.3.9 Ditta

Sia dato lo schema relazionale:

PERSONA(CF, Nome, Cognome, Stipendio)
LAVORA(CFPersona, PIvaDitta, Anzianità)
DIRIGE(CFPersona, PIvaDitta)
DITTA(PIva, Denominazione, NumeroDipendenti)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare nome e cognome dei dirigenti che percepiscono lo stipendio più alto.

2. Determinare la denominazione delle ditte in cui esiste almeno un lavoratore che percepisce uno stipendio maggiore di quello di almeno un dirigente della stessa ditta.

3. Determinare per ogni ditta la sua denominazione ed il totale degli stipendi pagati a lavoratori e dirigenti.

Interrogazione 1

```
SELECT   Nome, Cognome
FROM     Persona JOIN Dirige ON CF = CFPersona
WHERE    Stipendio =  (
             SELECT   MAX(Stipendio)
             FROM     Persona JOIN Dirige ON CF = CFPersona )
```

Interrogazione 2

```
SELECT   DISTINCT D.Denominazione
FROM     Persona AS P JOIN Lavora AS L ON P.CF = L.CFPersona
                      JOIN Ditta AS D ON L.PIva = D.PIva
WHERE    P.Stipendio >  ANY  (
             SELECT   P0.Stipendio
             FROM     Persona AS P0 JOIN Dirige AS DI0
                       ON P0.CF = DI0.CFPersona
             WHERE    L.PIva = DI0.PIva )
```

Interrogazione 3

```
CREATE VIEW SLav (PIva, Denominazione, Stipendio) AS
            SELECT   Ditta.PIva, Ditta.Denominazione, SUM(Persona.Stipendio)
            FROM     Persona JOIN Lavora ON Persona.CF = Lavora.CFPersona
                             JOIN Ditta ON Lavora.PIva = Ditta.PIva
            GROUP BY Ditta.PIva, Ditta.Denominazione

CREATE VIEW SDir (PIva, Denominazione, Stipendio) AS
            SELECT   Ditta.PIva, Ditta.Denominazione, SUM(Persona.Stipendio)
            FROM     Persona JOIN Dirige ON Persona.CF = Dirige.CFPersona
                             JOIN Ditta ON Dirige.PIva = Ditta.PIva
            GROUP BY Ditta.PIva, Ditta.Denominazione

SELECT   SLav.Denominazione, SLav.Stipendio+SDir.Stipendio
FROM     SLav JOIN SDir ON SLav.PIva = SDir.PIva
```

2.3.10 CD

Sia dato lo schema relazionale:

CD(Codice, Titolo, Genere)
NOLEGGIO(Data, CodiceCD, NomeCliente, Giorni)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare i clienti che hanno noleggiato CD di genere 'musicale', ma non li hanno mai tenuti per più di cinque giorni.

2. Determinare i clienti che in data 05-01-2000 hanno noleggiato il maggior numero di CD.

3. Determinare i generi di CD per i quali sono stati effettuati almeno 10 noleggi.

Interrogazione 1

```
SELECT   DISTINCT NomeCliente
FROM     CD JOIN Noleggio ON Codice = CodiceCD
WHERE    Genere = 'musicale'
         EXCEPT
SELECT   NomeCliente
FROM     CD JOIN Noleggio ON Codice = CodiceCD
WHERE    Genere = 'musicale'
         AND Giorni > 5
```

Soluzione Alternativa

```
SELECT   DISTINCT NomeCliente
FROM     CD JOIN Noleggio ON Codice = CodiceCD
WHERE    Genere = 'musicale'
         AND NomeCliente NOT IN (
             SELECT   NomeCliente
             FROM     CD JOIN Noleggio ON Codice = CodiceCD
             WHERE    Genere = 'musicale'
                      AND Giorni > 5 )
```

Interrogazione 2

```
SELECT   NomeCliente
FROM     Noleggio
WHERE    Data = 05-01-2000
GROUP BY NomeCliente
HAVING   COUNT(*) >=  ALL (
             SELECT   COUNT(*)
             FROM     Noleggio
             WHERE    Data = 05-01-2000
             GROUP BY NomeCliente )
```

Interrogazione 3

```
SELECT   Genere
FROM     CD JOIN Noleggio ON Codice = CodiceCD
GROUP BY Genere
HAVING   COUNT(*) >= 10
```

2.3.11 Mucca

Sia dato lo schema relazionale:

MUCCA(Codice, Nome, DataNascita, Razza, Peso)
VETERINARIO(CF, Nome, Ente)
VISITA(CFVeterinario, CodiceMucca, DataVisita, Esito)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare codice e nome delle mucche che hanno riportato esito 'positivo' in una visita effettuata dopo il 01-01-2001 da un veterinario della 'ASL'.

2. Determinare il nome delle mucche nate nell'anno 1999, che sono state visitate almeno una volta dalla loro nascita ma mai dopo il 30-09-2000.

3. Determinare, nell'anno 1980, il nome dei Veterinari che hanno visitato il minor numero di mucche.

Interrogazione 1

```
SELECT   DISTINCT Mucca.Codice, Mucca.Nome
FROM     Mucca JOIN Visita ON Codice = CodiceMucca
                JOIN Veterinario ON CFVeterinario = CF
WHERE    Ente = 'ASL'
         AND DataVisita > 01-01-2001
         AND Esito = 'positivo'
```

Interrogazione 2

```
SELECT   DISTINCT Nome
FROM     Mucca JOIN Visita ON Codice = CodiceMucca
WHERE    DataNascita > 01-01-1999
         EXCEPT
SELECT   Mucca
FROM     Mucca JOIN Visita ON Codice = CodiceMucca
WHERE    DataVisita > 30-09-2000
```

Soluzione Alternativa

```
SELECT   DISTINCT Nome
FROM     Mucca
WHERE    DataNascita > 01-01-1999
         AND Codice IN (
             SELECT   CodiceMucca
             FROM     Visita )
         AND Codice NOT IN (
             SELECT   CodiceMucca
             FROM     Visita
             WHERE    DataVisita > 30-09-2000 )
```

Interrogazione 3

```
CREATE VIEW NumMuccaVeterinario (CFVeterinario, Num) AS
            SELECT   CFVeterinario, COUNT(*)
            FROM     Visita
            WHERE    DataVisita BETWEEN 01-01-1980 AND 31-12-1980
            GROUP BY CFVeterinario

SELECT   Nome
FROM     Veterinario JOIN NumMuccaVeterinario ON CF = CFVeterinario
WHERE    Num <=  (
             SELECT   MIN(Num)
             FROM     NumMuccaVeterinario)
```

2.3.12 Arredamento

Sia dato lo schema relazionale:

MOBILE(Codice, Linea, Dimensione, Colore, Costo)
CLIENTE(CF, Nome, Cognome, CittàResidenza)
ORDINE(CFCliente, CodiceMobile, DataOrdine, DataConsegna, ModalitàTrasporto)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il codice dei mobile della linea 'bonde' che sono stati ordinati solo da clienti tutti residenti nella medesima città.
2. Determinare i clienti che nell'anno 2003 hanno effettuato almeno 2 ordini ma non più di 5.
3. Determinare il CF dei clienti che hanno fatto ordini nell'anno 2004 relativamente a mobili di colore rosso.

Interrogazione 1

```
SELECT   DISTINCT M.Codice
FROM     Mobile JOIN Ordine ON Codice = CodiceMobile
               JOIN Cliente AS C ON CFCliente = C.CF
WHERE    M.Linea = 'bonde'
         AND NOT EXISTS (
             SELECT   *
             FROM     Ordine JOIN Cliente AS C0
                       ON CFCliente = C0.CF
             WHERE    Mobile.Codice = CodiceMobile
                      AND C.CittàResidenza <> C0.CittàResidenza )
```

Interrogazione 2

```
SELECT   CFCliente
FROM     Ordine
WHERE    DataOrdine BETWEEN 01-01-2003 AND 31-12-2003
GROUP BY CFCliente
HAVING   COUNT(*) >= 2 AND COUNT(*) <= 5
```

Interrogazione 3

```
SELECT   DISTINCT CFCliente
FROM     Ordine JOIN Mobile ON CodiceMobile = Codice
WHERE    DataOrdine BETWEEN 01-01-2004 AND 31-12-2004
         AND Colore = 'rosso'
```

2.3.13 Gommista

Sia dato lo schema relazionale:
PNEUMATICO(Codice, Materiale, Dimensione, CodiceProduttore)
PRODUTTORE(Codice, Nome, Via, Città)
ACQUISTO(CFCliente, CodicePneumatico, DataVendita, Costo)
CLIENTE(CF, Nome, Telefono, Via, Città)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il nome dei produttori che nel 2003 hanno venduto pneumatici per un valore totale superiore o uguale a 2.000.000 euro.
2. Determinare il codice dei pneumatici che sono stati acquistati solo da clienti residenti nella stessa città del produttore del pneumatico.
3. Determinare, per ogni produttore, quanti pneumatici di materiale diverso producono.

Interrogazione 1

```
SELECT   Nome
FROM     Pneumatico JOIN Produttore ON CodiceProduttore = Produttore.Codice
                    JOIN Acquisto ON Pneumatico.Codice = CodicePneumatico
WHERE    DataVendita BETWEEN 01-01-2003 AND 31-12-2003
GROUP BY Produttore.Codice, Produttore.Nome
HAVING   SUM(Costo) >= 2000000
```

Interrogazione 2

Si noti che il join con *Acquisto* nell'interrogazione seguente è necessario per recuperare i pneumatici che sono stati acquistati almeno una volta.

```
SELECT   DISTINCT P.Codice
FROM     Pneumatico AS P JOIN Produttore AS PR
         ON P.CodiceProduttore = PR.Codice
                         JOIN Acquisto
         ON P.Codice = CodicePneumatico
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Acquisto JOIN Cliente
                      ON CFCliente = CF
             WHERE    P.Codice = CodicePneumatico
                      AND Città <> PR.Città )
```

Interrogazione 3

```
SELECT   CodiceProduttore, Nome, COUNT(DISTINCT Materiale)
FROM     Produttore JOIN Pneumatico ON Produttore.Codice=CodiceProduttore
GROUP BY CodiceProduttore, Nome
```

2.3.14 Calendari

Sia dato lo schema relazionale:

CALENDARIO(Codice, Prezzo, Tipo, AnnoCalendario, CodiceProduttore)
PRODUTTORE(Codice, Nome, Indirizzo)
FOTOGRAFIA(Numero, NomeFotografo, Anno, MeseCalendario, CodiceCalendario)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il nome del fotografo che ha almeno tre fotografie pubblicate in uno stesso calendario.
2. Determinare il codice del produttore che ha prodotto il maggior numero di calendari 'da muro' e che non ha mai prodotto calendari 'da tavolo'.
3. Determinare il prezzo medio dei calendari dove appaiono fotografie scattate da 'Herb Ritts'.

Interrogazione 1

```
SELECT   DISTINCT NomeFotografo
FROM     Fotografia
GROUP BY NomeFotografo, CodiceCalendario
HAVING   COUNT(*) >= 3
```

Interrogazione 2

```
CREATE VIEW CalMuro (CodiceProduttore) AS
            SELECT   CodiceProduttore
            FROM     Calendario
            WHERE    Tipo = 'da muro'

CREATE VIEW CalTavolo (CodiceProduttore) AS
            SELECT   CodiceProduttore
            FROM     Calendario
            WHERE    Tipo = 'da tavolo'

SELECT   CodiceProduttore
FROM     CalMuro
WHERE    CodiceProduttore NOT IN (
             SELECT   *
             FROM     CalTavolo )
GROUP BY CodiceProduttore
HAVING   COUNT(*) >=  ALL (
             SELECT   COUNT(*)
             FROM     CalMuro
             WHERE    CodiceProduttore NOT IN (
                          SELECT   *
                          FROM     CalTavolo )
             GROUP BY CodiceProduttore )
```

Soluzione Alternativa

```
CREATE VIEW SoloMuro (Codice, Numero) AS
            SELECT   CodiceProduttore, COUNT(*)
            FROM     Calendario
            WHERE    Tipo = 'da muro'
                     AND CodiceProduttore NOT IN  (
                         SELECT   CodiceProduttore
                         FROM     Calendario
                         WHERE    Tipo = 'da tavolo' )
            GROUP BY C.CodiceProduttore

SELECT   Codice
FROM     SoloMuro
WHERE    Numero >=  ALL (
             SELECT   Numero
             FROM     SoloMuro )
```

Interrogazione 3

```
SELECT   AVG(Prezzo)
FROM     Calendario
WHERE    Codice IN (
             SELECT   CodiceCalendario
             FROM     Fotografia
             WHERE    NomeFotografo = 'Herb Ritts' )
```

2.3.15 Fattorino

Sia dato lo schema relazionale:

FATTORINO(Codice, NomeDitta, Nome, Cognome, OraInizioTurno, OraFineTurno)
CLIENTE(CF, Nome, Cognome, Città)
CONSEGNA(CodiceFattorino, NomeDittaFattorino, CFCliente, Data, Orario)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare nome e cognome dei fattorini che hanno effettuato consegne al di fuori del loro orario di lavoro.

2. Determinare il nome e cognome del fattorino della ditta 'Rossi & Bianchi' che ha effettuato il maggior numero di consegne.

3. Determinare il codice e il nome della ditta dei fattorini che hanno effettuato consegne solo a Milano.

Interrogazione 1

```
SELECT   DISTINCT Nome, Cognome
FROM     Fattorino JOIN Consegna ON Codice = CodiceFattorino
         AND  NomeDitta = NomeDittaFattorino
WHERE    Orario NOT BETWEEN OraInizioTurno AND OraFineTurno
```

Interrogazione 2

```
SELECT   DISTINCT Nome, Cognome
FROM     Fattorino JOIN Consegna ON Codice = CodiceFattorino
         AND  NomeDitta = NomeDittaFattorino
WHERE    NomeDitta = 'Rossi & Bianchi'
GROUP BY Nome, Cognome, Codice
HAVING   COUNT(*) >=  ALL (
             SELECT   COUNT(*)
             FROM     Consegna
             WHERE    NomeDitta = 'Rossi & Bianchi'
             GROUP BY CodiceFattorino )
```

Interrogazione 3

```
SELECT   DISTINCT NomeDittaFattorino, CodiceFattorino
FROM     Consegna AS C
WHERE    NOT EXISTS (
             SELECT   *
             FROM     Consegna AS C1 JOIN Cliente ON CFCliente = CF
             WHERE    Città <> 'Milano'
                      AND C.NomeDittaFattorino = C1.NomeDittaFattorino
                      AND C.CodiceFattorino = C1.CodiceFattorino )
```

2.3.16 Mondiali

Sia dato lo schema relazionale:

SQUADRA(Nazione, AnnoEdizione, CT, IDCapitano)
EDIZIONE(Anno, NazioneOspitante, NazioneVincitrice)
GIOCATORE(ID, Cognome, Nome)
CONVOCAZIONE(IDGiocatore, Nazione, AnnoEdizione, Ruolo, NumGoalSegnati)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare la nazione, l'anno e il CT delle squadre che hanno vinto i mondiali l'anno in cui li hanno ospitati.

2. Determinare, per ogni anno di edizione, la nazione che ha convocato più giocatori.

3. Determinare quali giocatori hanno partecipato ad almeno 3 edizioni dei mondiali (cioè in 3 anni diversi).

Interrogazione 1

```
SELECT   Nazione, AnnoEdizione, CT
FROM     Squadra JOIN Edizione ON AnnoEdizione = Anno
WHERE    Nazione = NazioneOspitante
         AND NazioneOspitante = NazioneVincitrice
```

Interrogazione 2

```
SELECT   AnnoEdizione, Nazione
FROM     Convocazione AS C0
GROUP BY AnnoEdizione, Nazione
HAVING   COUNT(IDGiocatore) >=  ALL (
             SELECT   COUNT(IDGiocatore)
             FROM     Convocazione
             WHERE    C0.AnnoEdizione = AnnoEdizione
             GROUP BY Nazione )
```

Interrogazione 3

```
SELECT   ID, Nome, Cognome
FROM     Giocatore JOIN Convocazione ON ID = IDGiocatore
GROUP BY ID, Nome, Cognome
HAVING   COUNT(DISTINCT AnnoEdizione) = 3
```

2.3.17 SportClub

Sia dato lo schema relazionale:

SOCIO(NumTessera, Classe, Nome, Cognome, AnnoNascita)
PRENOTAZIONE(Progressivo, NumTesseraSocio, Sport, Data, Ora, Durata)

Nota: La durata viene espressa in ore.

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il Nome e il Cognome dei Soci di classe 'Gold' che hanno una prenotazione con una durata maggiore di 2 ore.

2. Determinare, per ogni sport, il numero della tessera dei soci che hanno fatto la prenotazione di minor durata.

3. Determinare il numero della tessera e il nome e il cognome dei soci che non hanno prenotazioni né per giocare a tennis né per giocare a calcetto.

Interrogazione 1

```
SELECT   DISTINCT Nome, Cognome
FROM     Socio JOIN Prenotazione ON NumTessera = NumTesseraSocio
WHERE    Classe = 'Gold'
         AND Durata > 2
```

Interrogazione 2

```
SELECT   NumTesseraSocio
FROM     Prenotazione AS P
WHERE    Durata =  (
             SELECT   MIN(Durata)
             FROM     Prenotazione
             WHERE    P.Sport = Sport )
```

Interrogazione 3

```
SELECT   NumTessera, Nome, Cognome
FROM     Socio
WHERE    NumTessera NOT IN (
             SELECT   NumTesseraSocio
             FROM     Prenotazione
             WHERE    Sport = 'Tennis'
                      OR Sport = 'Calcetto' )
```

2.3.18 Yogurt

Sia dato lo schema relazionale:

YOGURT(Codice, NomeTipo, Confezione, Peso, PrezzoConsigliato)
INGREDIENTE(Codice, Nome, ValoreNutritivo, CostoAlKg, Tipologia)
COMPOSIZIONE(CodiceYogurt, CodiceIngrediente, Quantità)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il codice e il nome del tipo degli yogurt che contengono almeno un ingrediente di tipo 'frutta'.

2. Determinare il codice degli yogurt che, per la loro produzione, usano più di 300 grammi di ingredienti di tipo 'cereale'.

3. Determinare il codice e il nome del tipo di tutti gli yogurt che non contengono ingredienti di tipo 'latticino'.

Interrogazione 1

```
SELECT   Yogurt.Codice, NomeTipo
FROM     Yogurt JOIN Composizione ON Yogurt.Codice = CodiceYogurt
                JOIN Ingrediente ON Ingrediente.Codice = CodiceIngrediente
WHERE    Tipologia = 'frutta'
```

Interrogazione 2

```
SELECT   CodiceYogurt
FROM     Composizione JOIN Ingrediente ON CodiceIngrediente = Codice
WHERE    Tipologia = 'cereale'
GROUP BY CodiceYogurt
HAVING   SUM(Quantità) > 300
```

Interrogazione 3

```
SELECT   Codice, NomeTipo
FROM     Yogurt
WHERE    Codice NOT IN (
             SELECT   CodiceYogurt
             FROM     Composizione JOIN Ingrediente
                      ON Codice = CodiceIngrediente
             WHERE    Tipologia = 'latticino' )
```

2.3.19 Aereo

Sia dato lo schema relazionale:
VOLO(Codice, Data, NomeCompagnia, Partenza, Destinazione, DistanzaMiglia)
PRENOTAZIONE(CodiceVolo, Data, CFCliente)
CLIENTE(CF, Cognome, Nome)

Si richiede di scrivere in SQL le seguenti interrogazioni:

1. Determinare il nome delle compagnie aeree che hanno almeno un volo con destinazione 'Londra'.
2. Determinare l'anagrafica (CF, cognome e nome) dei clienti che hanno percorso più di 5000 miglia con la stessa compagnia aerea.
3. Determinare il cognome e il nome dei clienti che hanno prenotato solo viaggi di distanza inferiore alle 500 miglia.

Interrogazione 1

```
SELECT   DISTINCT NomeCompagnia
FROM     Volo
WHERE    Destinazione = 'Londra'
```

Interrogazione 2

```
SELECT   DISTINCT Cliente.*
FROM     Cliente JOIN Prenotazione ON CF = CFCliente
                 JOIN Volo ON Codice = CodiceVolo
GROUP BY CF, Cognome, Nome, NomeCompagnia
HAVING   SUM(DistanzaMiglia) > 5000
```

Interrogazione 3

```
SELECT   Cognome, Nome
FROM     Cliente JOIN Prenotazione ON CF = CFCliente
WHERE    CF NOT IN (
             SELECT   CFCliente
             FROM     Volo JOIN Prenotazione ON Codice = CodiceVolo
             WHERE    DistanzaMiglia >= 500 )
```

Capitolo 3

Progettazione di Basi di Dati

Questo Capitolo è dedicato alle fasi di progettazione concettuale e di progettazione logica di una base di dati.
Per la *progettazione concettuale*, si fa riferimento al modello *Entità-Relazione* (ER), mentre per la *progettazione logica*, si fa riferimento al modello *relazionale*.

3.1 Progettazione Concettuale

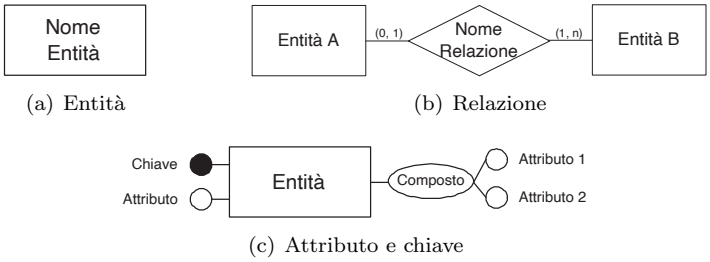
Il modello ER si basa sui concetti di *entità*, *relazione* ed *attributi* che vengono utilizzati per progettare una base di dati in accordo ai requisiti posti dal committente.

Entità

Rappresenta l'astrazione di diversi elementi, detti occorrenze, aventi caratteristiche comuni. Una entità rappresenta un elemento che può essere considerato indipendentemente dal resto del sistema. Ad esempio, i clienti di un'azienda possono rappresentare una entità per la progettazione della base di dati dell'azienda, così come i fornitori e i prodotti.
Graficamente è rappresentata da un rettangolo contenente il nome dell'entità, come in Figura 3.1(a).

Relazione

Rappresenta un'associazione fra due o più entità e può essere considerata come un legame funzionale fra le stesse. Ad esempio, l'entità prodotto potrebbe essere collegata all'entità cliente tramite la relazione vendita.
Graficamente è rappresentata da un rombo contenente il nome della relazione, collegato tramite delle linee alle entità coinvolte, come in Figura 3.1(b).
Per ciascun legame entità-relazione, deve essere specificata una *cardinalità minima* e una *cardinalità massima*, che rappresentano rispettivamente il numero minimo e massimo di



(a) Entità (b) Relazione

(c) Attributo e chiave

Figura 3.1: Esempi di entità, relazione, attributo (semplice e composto) e chiave

occorrenze con cui l'entità partecipa alla relazione. La cardinalità è rappresentata da una coppia *(cmin, cmax)*, dove *cmin* e *cmax* sono entrambi numeri interi naturali e $cmin \leq cmax$. Le cardinalità minime utilizzate in genere sono:

- 0 (opzionale): possono esserci delle occorrenze dell'entità che non partecipano alla relazione;
- 1 (obbligatoria): tutte le occorrenze dell'entità devono partecipare alla relazione.

Le cardinalità massime utilizzate in genere sono:

- 1 ciascuna occorrenza dell'entità non può partecipare più di una volta alla relazione;
- n ciascuna occorrenza dell'entità può partecipare un numero arbitrario di volte alla relazione.

Sulla base delle cardinalità massime con cui le entità coinvolte partecipano ad una relazione, le relazioni si distinguono in tre categorie:

- $1 : 1$ (uno a uno): ciascuna occorrenza di ciascuna entità può comparire al più una volta nella relazione.
- $1 : n$ (uno a molti): ciascuna occorrenza delle entità con cardinalità massima 1 compare al più una volta nella relazione mentre ciascuna occorrenza delle altre entità, può comparire più volte;
- $n : n$ (molti a molti): ciascuna occorrenza delle entità può comparire più volte nella relazione.

Attributo

Rappresenta una caratteristica elementare, definita nei requisiti, di un'entità o di una relazione. Esempi di attributi per l'entità cliente sono la partita iva, la ragione sociale e il tipo di società.
Graficamente è rappresentato da un cerchio vuoto affiancato al nome dell'attributo, collegato tramite una linea all'entità (o relazione) a cui si riferisce, come in Figura 3.1(c).

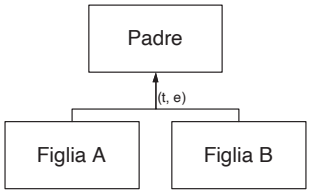


Figura 3.2: Esempio di generalizzazione totale ed esclusiva

Un attributo formato da più proprietà elementari logicamente correlate fra loro si dice composto e si rappresenta come in Figura 3.1(c). Gli attributi composti non possono essere modellati come entità perché i concetti che rappresentano non avrebbero legami con altre entità e non hanno sufficiente rilevanza nel sistema per essere considerati autonomi. Ad esempio, l'attributo indirizzo potrebbe essere composto dalla via, dal numero civico, dal CAP, dalla città e dalla provincia.
Così come per le relazioni, anche ciascun legame attributo-relazione è caratterizzato da una cardinalità, che rappresenta il numero minimo e massimo di occorrenze di un attributo associato ad una occorrenza della relazione. Se non indicata, la cardinalità dell'attributo è $(1, 1)$. Gli attributi con cardinalità massima maggiore di uno si dicono *multipli*, quelli con cardinalità minima zero *opzionali*.
Ogni entità deve avere un'attributo (o un insieme di attributi) che identificano univocamente ciascuna sua occorrenza. Tale attributo è detto *chiave* e graficamente è rappresentato da un cerchio pieno, come in Figura 3.1(c).
Esistono particolari entità, dette *deboli*, le cui occorrenze sono univocamente identificate dalla chiave di un'altra entità collegata ad essa tramite una relazione, insieme a zero, uno o più attributi dell'entità debole. Naturalmente, l'entità debole deve partecipare con cardinalità uno alla relazione che la lega all'entità da cui dipende.

3.1.1 Generalizzazione

Rappresenta un legame di ereditarietà fra un'entità, detta *padre*, e una o più entità, dette *figlie*. Le entità figlie rappresentano delle specializzazioni dell'entità padre e ne ereditano tutti gli attributi (compresa la chiave) e tutte le relazioni.
Graficamente una generalizzazione è rappresentata da una freccia uscente dalle figlie e diretta verso il padre, come in Figura 3.2.

Le generalizzazioni possono essere classificate in due diversi modi fra loro indipendenti. In base alla prima classificazione, una generalizzazione può essere:

- *totale*: tutte le occorrenze dell'entità padre fanno parte di almeno una delle figlie.
- *parziale*: esistono delle occorrenze dell'entità padre che non sono occorrenza di nessuna delle figlie.

In base alla seconda classificazione, una generalizzazione può essere:

- *esclusiva*: ogni occorrenza dell'entità padre può far parte di al più una delle figlie.
- *sovrapposta*: ogni occorrenza dell'entità padre può far parte di più figlie.

Si noti che il modello ER non offre, con i suoi costrutti, la possibilità di esprimere in modo completo i requisiti. Alcuni vincoli non sono infatti esprimibili direttamente nel modello. Per questo motivo, il modello ER è in genere accompagnato da una documentazione, dove si esprimono nel dettagli aspetti che appesantirebbero troppo il modello ER o che esso non consente di rappresentare. Ad esempio, data l'entità dipendente e l'entità dipartimento, legate da una relazione di afferenza, nel modello ER non è possibile esprimere il vincolo secondo cui i dipendenti di un dipartimento non possono avere stipendio superiore al responsabile dello stesso.

3.2 Progettazione Logica

Una volta conclusa la fase di progettazione concettuale, il modello ER deve essere tradotto nel corrispondente modello logico, nel nostro caso nel modello relazionale. Dato che il modello ER non può essere direttamente tradotto nel modello logico corrispondente, è necessaria una fase preliminare di ristrutturazione.
Questa fase prevede i seguenti passi:

- eliminazione degli attributi composti: le singole proprietà dell'attributo composto vengono collegate direttamente all'entità;
- eliminazione degli attributi multipli: la proprietà rappresentata dall'attributo multiplo diventa una nuova entità del modello, collegata per mezzo di una relazione all'entità cui era legato l'attributo multiplo;
- eliminazione delle generalizzazioni: esistono tre metodi
  1. *collasso verso l'alto* (accorpamento delle figlie nel padre): le entità figlie vengano rimosse e l'entità padre ne acquisisce attributi e relazioni, al padre va inoltre aggiunto un attributo (o più) selettore che discrimina, per ogni occorrenza, la figlia (o le figlie) cui apparteneva;
  2. *collasso verso il basso* (accorpamento del padre nelle figlie): l'entità padre viene rimossa e le figlie ne acquisiscono attributi e relazioni;
  3. *mantenimento delle entità*: tutte le entità vengono mantenute e si aggiunge, per ciascuna figlia, una relazione $(1:1)$ che la lega al padre; si osservi che in questo caso le entità figlie sono delle *entità deboli* le cui occorrenze sono quindi identificate dalla chiave del padre.

A questo punto è possibile effettuare la traduzione del modello ER in modello logico:

- traduzione delle entità: ogni entità diventa una tabella[1] con, come attributi, gli attributi dell'entità e, come chiave, la chiave dell'entità;

[1] Nel modello relazionale le relazioni vengono anche dette *tabelle*.

- traduzione standard delle relazioni: ogni relazione diventa una tabella con, come attributi, i suoi attributi e, come chiavi, le chiavi delle entità che collega. Esistono due casi particolari per le relazioni uno a molti e uno a uno:

  - relazioni $(1 : n)$: la relazione viene tradotta aggiungendo alla tabella della relazione collegata con cardinalità massima 1, la chiave della entità collegata con cardinalità massima n;

  - relazioni $(1 : 1)$: le due entità collegate e la relazione vengono tradotte in una sola tabella che ha come attributi tutti gli attributi delle entità di partenza.

Si noti che, in fase di traduzione di una relazione, le tabelle che traducono le entità coinvolte e/o la relazione stessa acquisiscono attributi già presenti in altre tabelle. In particolare, per tradurre una relazione, una tabella contiene la chiave di un'entità tradotta da un'altra tabella. Tali attributi sono detti *chiavi esterne* e sono legati da un vincolo detto di *integrità referenziale* al corrispondente attributo nell'altra tabella. In particolare, una chiave esterna può assumere solo uno fra i valori che assume l'attributo corrispondente nella tabella cui si fa riferimento.

3.3 Esercizi Svolti e Commentati

In questa sezione vengono presentati alcuni esercizi svolti. Nel seguito verrà utilizzata la seguente notazione:

- i nomi delle *entità* sono scritti in SmallCaps;

- i nomi delle *relazioni* sono scritti in **Grassetto**;

- i nomi degli *attributi* sono scritti in *Italico*;

- le *chiavi* sono *Sottolineate*.

3.3.1 La Fattoria

Testo del Problema

La fattoria YaYaHoo è specializzata nell'allevamento di animali. Di ciascun animale si conoscono: il codice identificativo, il peso espresso in chilogrammi, il sesso e la data di nascita. In particolare, gli animali si distinguono fra suini, bovini ed equini. Di questi ultimi si vuole tenere traccia della data di ultima ferratura, per poter programmare la prossima.

Ogni animale vive in una stalla. Di ciascuna stalla si conosce la posizione geografica e la superficie occupata, espressa in metri quadrati. Ogni stalla è equipaggiata con gli strumenti necessari alla sua manutenzione. In particolare, di ogni strumento si vuole tenere traccia della descrizione, del tipo di attrezzo, del suo prezzo di acquisto e della stalla dove è collocato. Ogni strumento è identificato da un codice, unico all'interno della stalla.

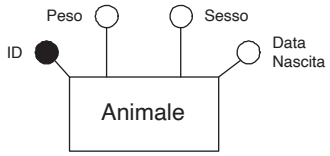
Nella fattoria lavorano diversi dipendenti, di cui si conosce l'anagrafica, che include il codice fiscale, nome, cognome, indirizzo (composto da via, CAP, città e provincia) e numero di telefono, e lo stipendio mensilmente percepito. I dipendenti si dividono in due categorie: gli addetti alla manutenzione e coloro che si occupano degli animali. I primi, per svolgere il loro compito, utilizzano gli attrezzi siti nelle varie stalle; per ogni utilizzo degli attrezzi si vuole tenere traccia della data di utilizzo. Degli altri, invece, si vuole mantenere l'elenco degli animali di cui si occupano. Si noti che ciascun dipendente può occuparsi, in momenti diversi della giornata, sia della manutenzione sia della cura degli animali, a seconda delle necessità del momento.

La fattoria YaYaHoo intrattiene rapporti commerciali con aziende esterne per l'acquisto e la vendita degli animali. Ciascuna di queste aziende è caratterizzata da una Partita IVA, un nome, un indirizzo (composto da via, CAP, città e provincia) e uno o più numeri telefonici. Ogni volta che un animale viene acquistato è necessario mantenere la data di acquisto e il prezzo pagato; allo stesso modo in caso di vendita si tiene traccia della data e del prezzo di vendita.

Analisi Passo Passo del Testo

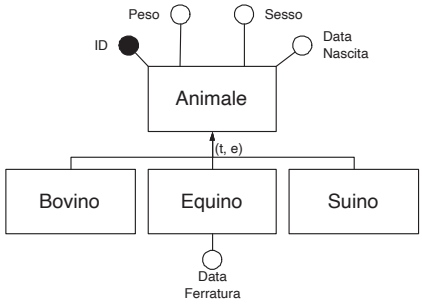
Di seguito si propone un'analisi dettagliata dei requisiti di partenza, utilizzando la notazione precedentemente introdotta.
Per ogni parte del testo analizzata, si riporta la porzione di modello ER corrispondente.

> La fattoria YaYaHoo è specializzata nell'allevamento di ANIMALI. Di ciascun animale si conoscono: il *codice identificativo*, il *peso* espresso in chilogrammi, il *sesso* e la *data di nascita*.



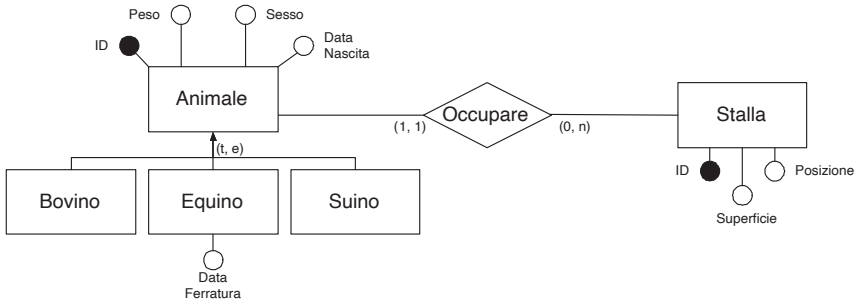
> In particolare, gli animali si distinguono fra SUINI, BOVINI ed EQUINI. Di questi ultimi si vuole tenere traccia della *data di ultima ferratura*, per poter programmare la prossima.

Dato che suini, bovini ed equini sono delle classi di animali e che quindi ne condividono le caratteristiche, si possono rappresentare come una gerarchia. In particolare, la gerarchia sarà totale ed esclusiva, dato che nei requisiti viene esplicitamente detto che la fattoria si occupa solo di questi tipi di animali e che non esiste alcun animale che faccia parte di due o più di tali classi.



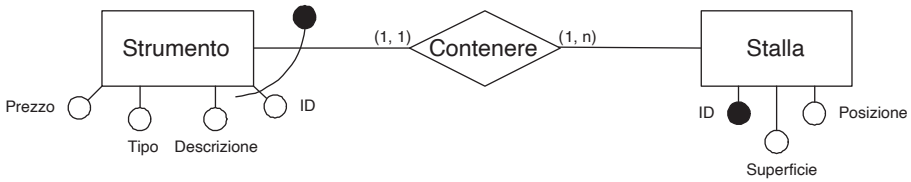
> Ogni animale **vive** in una STALLA. Di ciascuna stalla si conosce la *posizione geografica* e la *superficie occupata*, espressa in metri quadrati.

Dato che per l'entità stalla non viene indicato un attributo chiave, è stato aggiunto un attributo ID che funge da chiave.
La relazione Occupare collega le entità animale e stalla ed è di tipo uno a molti. Infatti ogni animale vive in una sola stalla, ma ogni stalla può ospitare più animali.



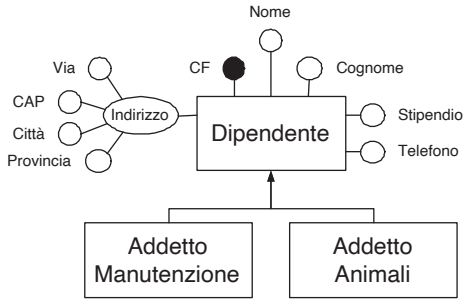
Ogni stalla è **equipaggiata** con gli STRUMENTI necessari alla sua manutenzione. In particolare, di ogni strumento si vuole tenere traccia della *descrizione*, del *tipo* di attrezzo, del suo *prezzo di acquisto* e della stalla dove è collocato. Ogni strumento è identificato da un *codice*, unico all'interno della stalla.

Dato che il codice di ciascuno strumento è *unico all'interno della stalla*, questo attributo da solo non può essere chiave per l'entità strumento. Proprio per questo, la chiave deve contenere il riferimento all'occorrenza dell'entità stalla cui l'occorrenza dello strumento fa riferimento. Entità di questo tipo si dicono *deboli* e si rappresentano come nella Figura seguente.



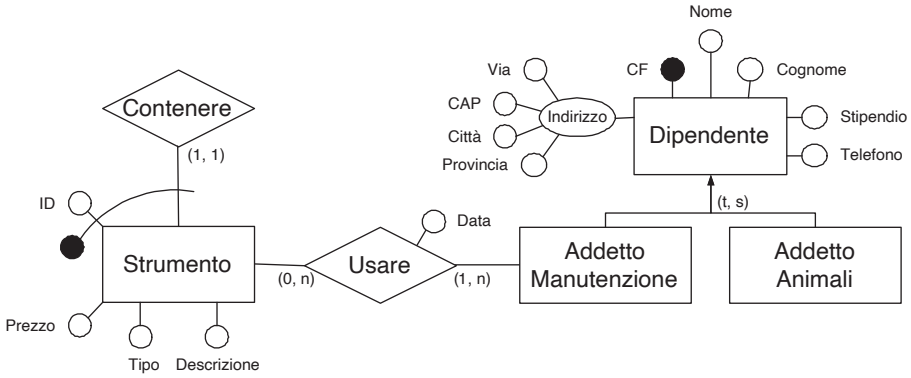
Nella fattoria lavorano diversi DIPENDENTI, di cui si conosce l'anagrafica, che include il *codice fiscale*, *nome*, *cognome*, *indirizzo* (composto da via, CAP, città e provincia) e *numero di telefono*, e lo *stipendio* mensilmente percepito. I dipendenti si dividono in due categorie: gli ADDETTI ALLA MANUTENZIONE e coloro che si OCCUPANO DEGLI ANIMALI.

Dato che l'attributo indirizzo si compone di via, CAP, città e provincia, questo verrà rappresentato tramite un *attributo composto*. Inoltre, l'entità dipendente è una generalizzazione delle altre due entità, quindi si presenta una gerarchia.



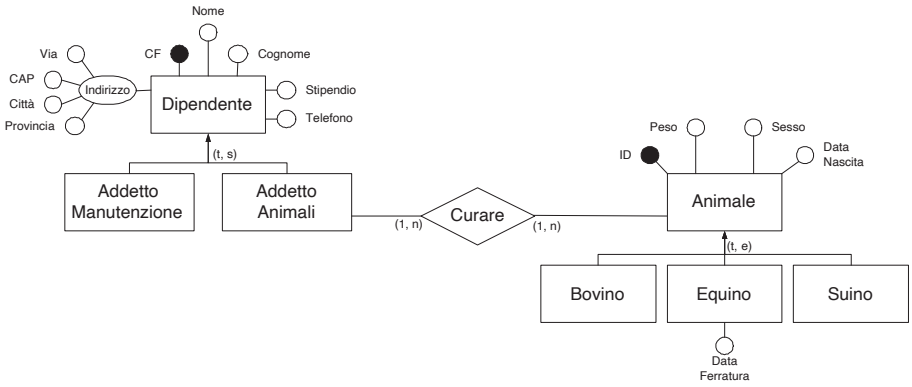
> I primi (addetti alla manutenzione), per svolgere il loro compito, **utilizzano** gli attrezzi siti nelle varie stalle; per ogni utilizzo degli attrezzi si vuole tenere traccia della *data* di utilizzo.

Dato che la data di utilizzo è relativa alla relazione Usare, tale attributo dovrà essere collegato alla relazione stessa nel modello ER.



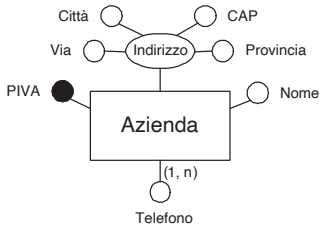
> Degli altri (coloro che si occupano degli animali), invece, si vuole mantenere l'elenco degli animali di cui si **occupano**. Si noti che ciascun dipendente può occuparsi, in momenti diversi della giornata, sia della manutenzione sia della cura degli animali, a seconda delle necessità del momento.

In questa parte del testo viene esplicitato che ciascun dipendente può svolgere entrambe le mansioni, quindi la gerarchia sopra introdotta è di tipo *totale* e *sovrapposto*.

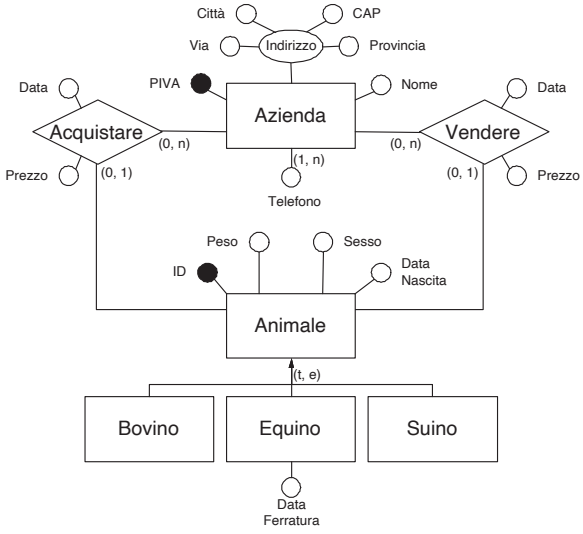


> La fattoria YaYaHoo intrattiene rapporti commerciali con aziende esterne per l'acquisto e la vendita degli animali. Ciascuna di queste aziende è caratterizzata da una *Partita IVA*, un *nome*, un *indirizzo* (composto da via, CAP, città e provincia) e uno o più *numeri telefonici*.

Anche per l'entità azienda, l'attributo indirizzo è composto. Dato poi che una stessa azienda può avere più di un numero telefonico, siamo in presenza anche di un *attributo multiplo*, cioè con cardinalità minima 1 e massima n.



Ogni volta che un animale viene **acquistato** è necessario mantenere la *data* di acquisto e il *prezzo* pagato; allo stesso modo in caso di **vendita** si tiene traccia della *data* e del *prezzo* di vendita.



Dato che acquisto e vendita hanno significati diversi, si hanno due relazioni diverse fra le stesse due entità. Si noti che entrambe le relazioni hanno dei loro attributi.
Il modello ER finale è rappresentato in Figura 3.3.

Ristrutturazione del Modello ER

Come primo passo, devono essere eliminati gli attributi composti. Nel caso specifico, il modello ER presenta l'attributo composto indirizzo per l'entità dipendente e l'attributo composto indirizzo per l'entità azienda. Tali attributi vengono eliminati aggiungendo gli attributi semplici via, CAP, città e provincia alle entità coinvolte.

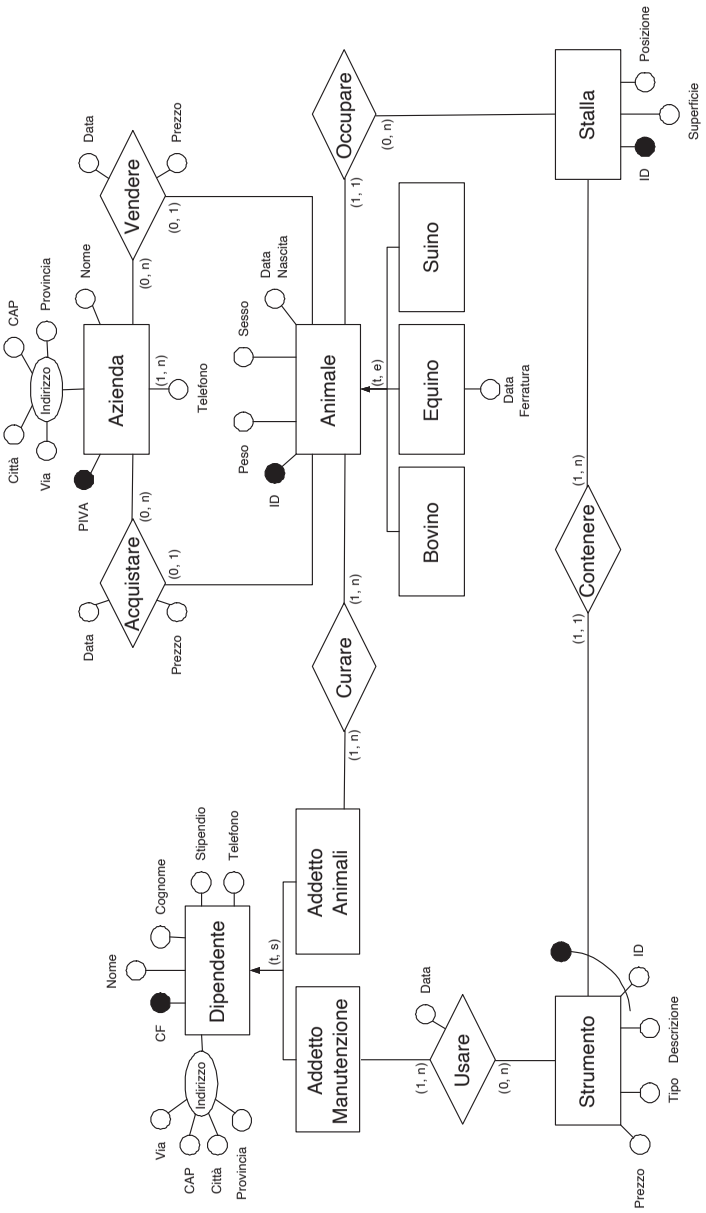
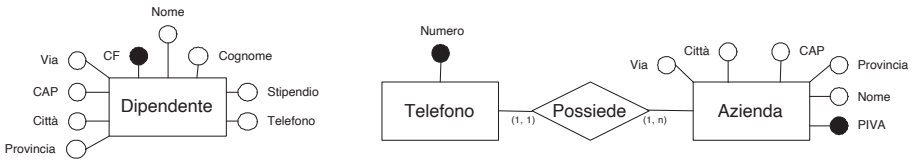
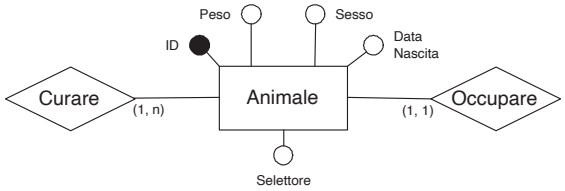


Figura 3.3: Modello ER esercizio Fattoria

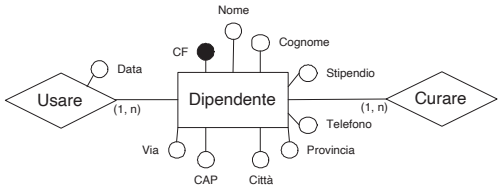
Così come gli attributi composti, anche gli attributi multipli devono essere eliminati. In particolare, l'attributo multiplo telefono dell'entità azienda si traduce in un'entità con lo stesso nome, dotata di un solo attributo, che funge anche da chiave. L'entità telefono è collegata all'entità azienda attraverso una relazione $1 : n$, dato che un numero di telefono può fare riferimento ad una sola azienda.



Procediamo a questo punto con la eliminazione della generalizzazione dell'entità animale, che è totale ed esclusiva. Dato che le entità figlie non partecipano a nessuna relazione, al contrario della relazione padre, risulta più conveniente un collasso verso l'alto. In particolare, l'entità animale acquisisce un attributo (detto selettore), e l'attributo opzionale data ferratura che era dell'entità equino che è stata rimossa.



Si elimina poi anche la generalizzazione sull'entità dipendente, che è totale e sovrapposta. Anche se l'entità padre non partecipa a nessuna relazione, al contrario delle relazioni figlie, risulta più conveniente un collasso verso l'alto, perché adottando un collasso verso il basso si darebbe luogo a ridondanza. In particolare, l'entità dipendente acquisisce due diversi selettori, per indicare se un dipendente si occupa della manutenzione, degli animali o di entrambe le cose.



Si passa a questo punto alla traduzione delle entità nel modello ER. È interessante notare come, per tradurre correttamente l'entità debole strumento, nel modello logico la tabella corrispondente ha un attributo aggiuntivo, che fa parte della chiave: *IDStalla*, che identifica la stalla in cui si trova lo strumento.

Relazione	**Attributi**
AZIENDA	PIVA, Nome, Via, Cap, Città, Provincia
ANIMALE	ID, Peso, Sesso, DataNascita, DataFerratura, Selettore
STALLA	ID, Superficie, Posizione
STRUMENTO	IDStalla, ID, Descrizione, Tipo, Prezzo
DIPENDENTE	CF, Nome, Cognome, Stipendio, Telefono, Via, Cap, Città, Provincia, CuraAnimali, Manutentore
TELEFONO	Numero

Naturalmente, per completare la traduzione, è necessario tradurre anche le relazioni. In particolare, le relazioni $1 : n$ occupare e contenere sono tradotte aggiungendo una chiave esterna alle entità animale (*IDStalla*) e strumento (*PIVAAzienda*) rispettivamente. Per le altre relazioni, invece, si sono aggiunte delle nuove tabelle.

Relazione	**Attributi**
AZIENDA	PIVA, Nome, Via, Cap, Città, Provincia
ANIMALE	ID, Peso, Sesso, DataNascita, DataFerratura, Selettore, IDStalla
STALLA	ID, Superficie, Posizione
STRUMENTO	IDStalla, ID, Descrizione, Tipo, Prezzo
DIPENDENTE	CF, Nome, Cognome, Stipendio, Telefono, Via, Cap, Città, Provincia, CuraAnimali, Manutentore
TELEFONO	Numero, PIVAAzienda
ACQUISTARE	IDAnimale, PIVAAzienda, Data, Prezzo
VENDERE	IDAnimale, PIVAAzienda, Data, Prezzo
USARE	IDStalla, IDStrumento, CFDipendente, Data
CURARE	IDAnimale, CFDipendente

Di seguito sono riportati i vincoli di integrità referenziale fra le tabelle.

Chiave Esterna	**Referenzia**
TELEFONO.PIVAAzienda	AZIENDA.PIVA
ANIMALE.IDStalla	STALLA.ID
ACQUISTARE.IDAnimale	ANIMALE.ID
ACQUISTARE.PIVAAzienda	AZIENDA.PIVA
VENDERE.IDAnimale	ANIMALE.ID
VENDERE.PIVAAzienda	AZIENDA.PIVA
STRUMENTO.IDStalla	STALLA.ID
USARE.{IDStalla, IDStrumento}	STRUMENTO.{IDStalla, ID}
USARE.CFDipendente	DIPENDENTE.CF
CURARE.IDAnimale	ANIMALE.ID
CURARE.CFDipendente	DIPENDENTE.CF

3.3.2 Centro di Addestramento Cinofilo

Testo del Problema

I dipendenti del centro di addestramento cinofilo FIDO sono identificati dalla matricola ed hanno associato un nome, cognome, indirizzo, stipendio e uno o più numeri di telefono. Di ogni cane, identificato da un codice, si vogliono mantenere le informazioni relative a: razza, genitrice (nel caso questa faccia parte dei cani presenti all'interno del centro), data di nascita ed eventuali malattie contratte. Per le malattie è necessario sapere il nome scientifico della malattia ed una sua descrizione, la descrizione dei sintomi che la caratterizzano, quando la malattia stessa è stata contratta, il livello di gravità e la terapia a cui il cane è stato sottoposto. Di ciascuna terapia si vuole mantenere traccia del nome della terapia, la data di inizio e fine della sua applicazione e l'insieme di medicinali usati. Di ciascun medicinale si conoscono il nome ed il principio attivo. Si noti che un cane può aver contratto più malattie e può essere stato sottoposto a più terapie nell'ambito della stessa malattia. Inoltre una stessa terapia può essere applicata a diversi cani per diverse malattie. Ad ogni cane è associato un addestratore, dipendente del centro, che addestra solamente quel cane. Gli addestratori possono collaborare con altri centri di addestramento. In questo caso, per motivi di reperibilità, si deve mantenere l'informazione relativa al nome ed indirizzo dei centri con cui gli addestratori collaborano nonché il numero di telefono. Un cane, in un dato giorno e ad una data ora, può prendere parte ad una lezione di addestramento. Ad ogni lezione deve essere presente un medico veterinario che può assistere a più lezioni in parallelo. Ogni veterinario, anch'esso dipendente del centro, da la propria disponibilità solamente per alcuni giorni alla settimana.

Analisi del Testo

Procediamo a questo punto con l'analisi del testo.

> I DIPENDENTI del centro di addestramento cinofilo FIDO sono identificati dalla *matricola* ed hanno associato un *nome*, *cognome*, *indirizzo*, *stipendio* e uno o più *numeri di telefono*. Di ogni CANE, identificato da un *codice*, si vogliono mantenere le informazioni relative a: *razza*, **genitrice** (nel caso questa faccia parte dei cani presenti all'interno del centro), *data di nascita* ed eventuali MALATTIE **contratte**. Per le malattie è necessario sapere il *nome scientifico* della malattia ed una sua *descrizione*, la descrizione dei *sintomi* che la caratterizzano, *quando* la malattia stessa è stata contratta, il *livello di gravità* e la TERAPIA a cui il cane è stato **sottoposto**. Di ciascuna terapia si vuole mantenere traccia del *nome* della terapia, la *data di inizio* e *fine* della sua applicazione e l'insieme di MEDICINALI **usati**. Di ciascun medicinale si conoscono il *nome* ed il *principio attivo*. Si noti che un cane può aver contratto più malattie e può essere stato sottoposto a più terapie nell'ambito della stessa malattia. Inoltre una stessa terapia può essere applicata a diversi cani per diverse malattie. Ad ogni cane è associato un ADDESTRATORE, dipendente del centro, che **addestra** solamente quel cane. Gli addestratori possono **collaborare** con altri CENTRI DI ADDESTRAMENTO. In questo caso, per motivi di reperibilità, si deve mantenere l'informazione relativa al *nome* ed *indirizzo* dei centri con cui gli addestratori collaborano nonché il *numero di telefono*. Un cane, in un dato *giorno* e ad una data *ora*, può **prendere parte** ad una LEZIONE di addestramento. Ad ogni lezione deve essere presente un medico VETERINARIO che può **assistere** a più lezioni in parallelo. Ogni veterinario, anch'esso dipendente del centro, da la propria *disponibilità* solamente per alcuni giorni alla settimana.

Si noti, come prima cosa, che l'entità dipendente è una generalizzazione per addestratore e veterinario. Si ha quindi una gerarchia di tipo parziale, dato che non si sa se ci sono altri tipi di dipendenti nel centro, ed esclusiva.
La relazione genitrice è una *autorelazione*, che collega l'entità cane a sé stessa per rappresentare il rapporto genitrice-figlio. La relazione sottoposto è invece ternaria, in quanto collega le tre entità cane, malattia e terapia.
Il modello ER ha poi due attributi multipli: il numero di telefono del dipendente e i giorni di disponibilità dei veterinari.
Come ultima cosa, si può notare che alcune entità non hanno una loro chiave, quindi sarà necessario aggiungere un attributo che ne identifichi univocamente le occorrenze.

Modello ER

In Figura 3.4 è presentato il modello ER ottenuto dall'analisi dei requisiti.

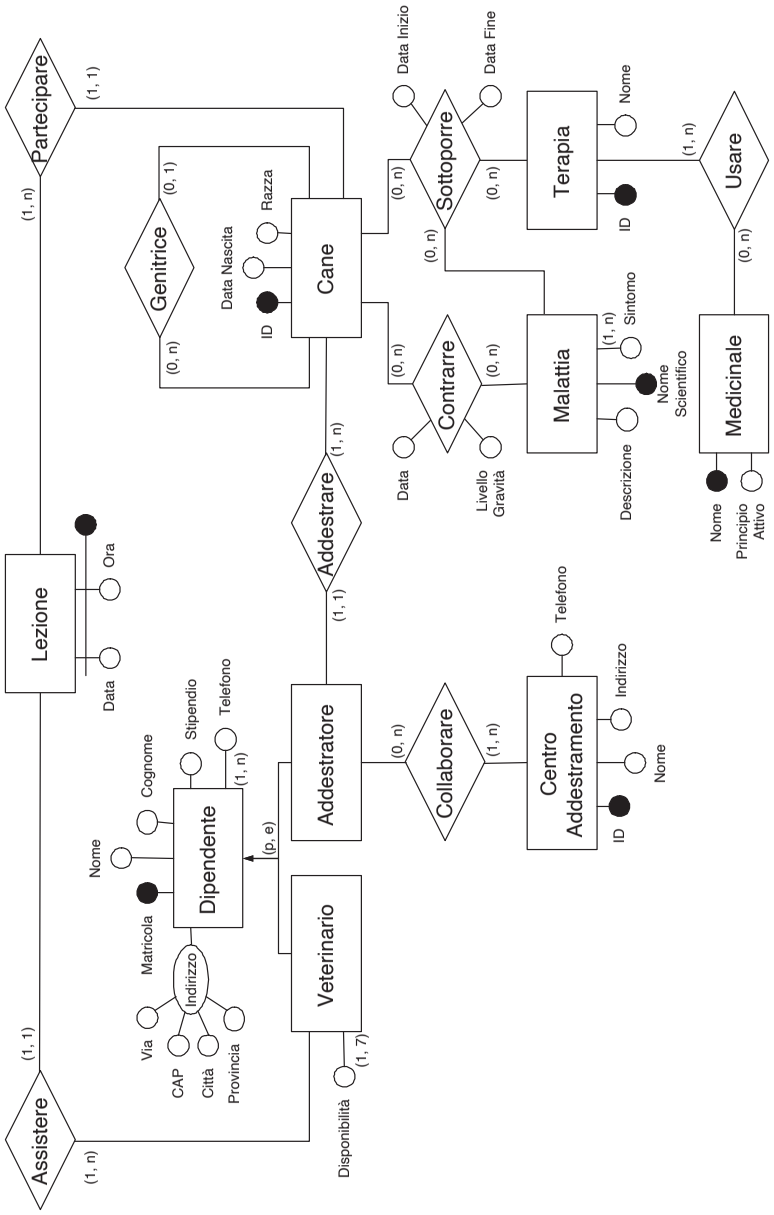
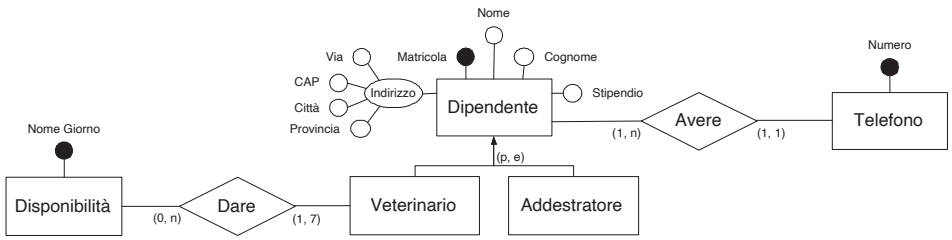


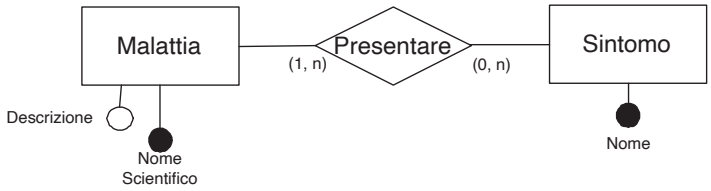
Figura 3.4: Modello ER esercizio Centro di Addestramento Cinofilo

Ristrutturazione

La ristrutturazione procede come mostrato nell'esercizio precedente. Qui, in particolare, ci si sofferma sulla ristrutturazione degli attributi multipli.
L'attributo *Telefono* viene ristrutturato attraverso una entità TELEFONO, avente come chiave *Numero*, e collegata all'entità DIPENDENTE attraverso una relazione $1 : n$, dato che un numero di telefono può appartenere ad una sola utenza.
L'attributo *Disponibilità* dà invece luogo ad una relazione $n : n$, **Dare**.



In modo analogo, l'attributo *Sintomo* dà luogo alla relazione $n : n$ **Presentare**. Si noti però che, anche in questi casi, nel modello logico compare una tabella per tradurre l'entità DISPONIBILITÀ e la relazione **Dare**, e una tabella per l'entità SINTOMO e la relazione **Presentare**. Questa soluzione è ammessa poiché le due entità hanno un solo attributo, che funge anche da chiave, e che fa parte della chiave della tabella risultante.



Modello Logico

La gerarchia che rappresenta i due tipi di dipendenti è stata ristrutturata con un collasso verso l'alto.

Relazione	Attributi
DIPENDENTE	Matricola, Nome, Cognome, Indirizzo, Stipendio, Selettore
TELEFONO	Numero, MatricolaDipendente
DISPONIBILITÀ	MatricolaDipendente, Giorno
CENTROADDESTRAMENTO	ID, Nome, Indirizzo, Telefono
CANE	ID, Razza, DataNascita, IDGenitrice, MatricolaAddestratore, GiornoLezione, OraLezione
LEZIONE	Giorno, Ora, MatricolaVeterinario
MALATTIA	NomeScientifico, Descrizione
TERAPIA	ID, Nome
MEDICINALE	Nome, PrincipioAttivo
SINTOMO	ID, Descrizione
COLLABORARE	MatricolaDipendente, IDCentro
CONTRARRE	IDCane, NomeMalattia, Data, LivelloGravità
SOTTOPORRE	IDCane, IDTerapia, NomeMalattia, DataInizio, DataFine
USARE	NomeMedicinale, IDTerapia
RIGUARDARE	IDSintomo, NomeMalattia

Chiave Esterna	Referenzia
TELEFONO.MatricolaDipendente	DIPENDENTE.Matricola
DISPONIBILITÀ.MatricolaDipendente	DIPENDENTE.Matricola
CANE.IDGenitrice	CANE.ID
CANE.MatricolaAddestratore	DIPENDENTE.Matricola
CANE.{GiornoLezione, OraLezione}	LEZIONE.{Giorno, Ora}
LEZIONE.MatricolaVeterinario	DIPENDENTE.Matricola
COLLABORARE.MatricolaDipendente	DIPENDENTE.Matricola
COLLABORARE.IDCentro	CENTRO.ID
CONTRARRE.IDCane	CANE.ID
CONTRARRE.NomeMalattia	MALATTIA.NomeScientifico
SOTTOPORRE.IDCane	CANE.ID
SOTTOPORRE.IDTerapia	TERAPIA.ID
SOTTOPORRE.NomeMalattia	MALATTIA.NomeScientifico
USARE.NomeMalattia	MALATTIA.NomeScientifico
USARE.IDTerapia	TERAPIA.ID
RIGUARDARE.IDSintomo	SINTOMO.ID
RIGUARDARE.NomeMalattia	MALATTIA.NomeScientifico

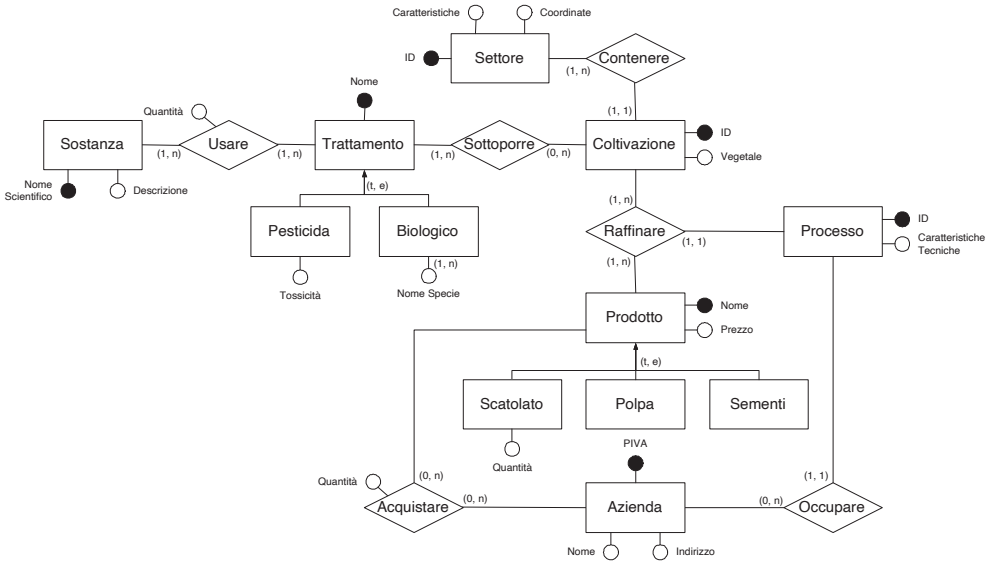
3.4 Esercizi con Soluzione

3.4.1 Azienda Agricola

Le coltivazioni trattate dall'azienda agricola Grano d'Oro sono caratterizzate da un codice e dal particolare vegetale in esse coltivato. Ogni coltivazione è contenuta in un settore che può essere in grado di ospitare coltivazioni diverse. I settori sono caratterizzati da un codice, dalle coordinate e da una descrizione delle caratteristiche ambientali (es., esposizione al sole). Ogni coltivazione viene sottoposta a svariati trattamenti per i quali si conosce il nome e la lista di sostanze usate durante il trattamento. Di queste sostanze è importante tenere traccia del nome scientifico, della descrizione delle loro proprietà e della quantità impiegata. In particolare, i trattamenti si distinguono in pesticidi, per i quali si vuole tenere traccia del grado di tossicità, e biologici, per i quali si vuole tenere traccia delle specie animali o vegetali coinvolte. Tramite un processo di raffinamento, ogni coltivazione viene sfruttata per produrre uno o più prodotti. I processi di raffinamento hanno associato un codice che li identifica e le caratteristiche tecniche. Processi diversi possono essere applicati ad una stessa coltivazione per produrre un medesimo prodotto. Inoltre, applicando lo stesso processo alla stessa coltivazione si ottiene sempre lo stesso prodotto. I prodotti sono invece caratterizzati da un nome e da un prezzo all'ingrosso e possono essere destinati all'inscatolamento diretto (in tal caso è necessario mantenere l'informazione relativa alla quantità grezza necessaria a produrre una singola unità di vendita), alla produzione di polpa vegetale o alla produzione di sementi.
L'azienda agricola in questione è in rapporti commerciali, per scopi diversi, con altre aziende caratterizzate da una partita iva, nome ed indirizzo. Alcune di queste aziende sono acquirenti di alcuni dei prodotti e per queste si vuole, per ciascun prodotto acquistato, tenere traccia delle quantità acquistate. Altre aziende sono invece responsabili di alcuni dei processi produttivi di cui sopra; aziende diverse sono responsabili di processi diversi.

Soluzione

Le gerarchie di generalizzazione delle entità TRATTAMENTO e PRODOTTO sono entrambe totali ed esclusive. Infatti vengono indicate le due uniche categorie di trattamenti e le sole tipologie di prodotti. Inoltre, le categorie rappresentate dalle due generalizzazioni non possono essere sovrapposte.



Le gerarchie di generalizzazione delle entità TRATTAMENTO e PRODOTTO sono state ristrutturate con un collasso verso l'alto.

Relazione	**Attributi**
COLTIVAZIONE	ID, Vegetale, IDSettore
SETTORE	ID, Coordinate, Caratteristiche
PROCESSO	ID, CaratteristicheTecniche, PIVAAzienda
AZIENDA	PIVA, Nome, Indirizzo
PRODOTTO	Nome, Prezzo, Quantità, Selettore
TRATTAMENTO	Nome, Tossicità, Selettore
SPECIE	NomeTrattamento, NomeSpecie
SOSTANZA	NomeScientifico, Descrizione
ACQUISTARE	NomeProdotto, PIVAAzienda, Quantità
SOTTOPORRE	IDColtivazione, NomeTrattamento
RAFFINARE	IDColtivazione, IDProcesso, NomeProdotto
USARE	NomeSostanza, NomeTrattamento, Quantità

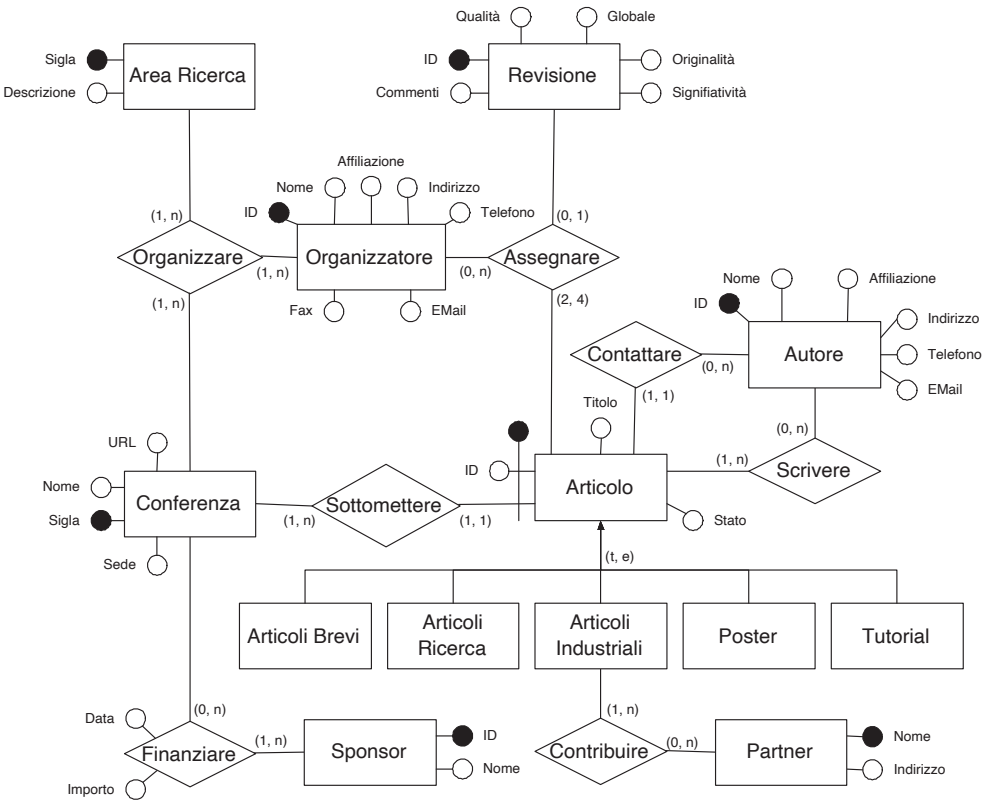
Chiave Esterna	Referenzia
COLTIVAZIONE.IDSettore	SETTORE.ID
PROCESSO.PIVAAzienda	AZIENDA.PIVA
ACQUISTARE.NomeProdotto	PRODOTTO.Nome
ACQUISTARE.PIVAAzienda	AZIENDA.PIVA
SPECIE.NomeTrattamento	TRATTAMENTO.Nome
SOTTOPORRE.IDColtivazione	COLTIVAZIONE.ID
SOTTOPORRE.NomeTrattamento	TRATTAMENTO.Nome
RAFFINARE.IDColtivazione	COLTIVAZIONE.ID
RAFFINARE.NomeProdotto	PRODOTTO.Nome
RAFFINARE.IDProcesso	PROCESSO.ID
USARE.NomeSostanza	SOSTANZA.Nome
USARE.NomeTrattamento	TRATTAMENTO.Nome

3.4.2 Organizzazione Conferenza

Ogni conferenza organizzata dalla società Ci Pensiamo Noi è identificata da una sigla ed ha associato un nome, la sede in cui avrà luogo, l'URL della home page e l'eventuale insieme di sponsor che finanziano la conferenza. Di ciascun sponsor sono noti il nome, la data in cui è stato erogato il finanziamento e l'importo. Ad ogni conferenza viene assegnato un insieme di organizzatori, che formano il comitato di programma e per i quali si conosce il nome, l'affiliazione, l'indirizzo, il telefono, il fax e l'e-mail. Ciascun organizzatore può indicare, per ognuna delle conferenza di cui fa parte come membro del comitato di programma, uno o più aree di ricerca denotate da una sigla e da una descrizione. Gli articoli sottomessi ad una conferenza (un articolo non può essere sottomesso a più di una conferenza) sono caratterizzati da un numero progressivo nell'ambito della conferenza, un titolo e uno o più autori per i quali si devono mantenere le informazioni relative al nome, affiliazione, indirizzo, telefono, ed e-mail. Si osservi che, fra gli autori, uno di essi deve essere indicato come "autore di contatto" il quale riceverà tutte le comunicazioni inerenti l'articolo sottomesso. Gli articoli sono inoltre suddivisi in diverse categorie: tutorial, articoli brevi, poster, articoli industriali ed articoli di ricerca. Per gli articoli industriali devono essere memorizzate le informazioni (nome ed indirizzo) relative a tutti i partner che hanno contribuito alla ricerca presentata nell'articolo stesso. In base all'area di ricerca specificata, gli articoli vengono assegnati ai membri del comitato di programma (il numero di revisori per articoli varia da un minimo di due, ad un massimo di quattro) i quali dovranno preparare una revisione. La revisione preparata da un revisore per un determinato articolo include un punteggio sulla originalità, significatività e qualità del lavoro proposto ed un punteggio globale, nonché dei commenti che verranno inviati all'autore di contatto. In base ai giudizi raccolti, lo stato di ogni articolo potrà essere uguale ad "accettato" oppure "rifiutato".

Soluzione

La gerarchia di generalizzazione dell'entità ARTICOLO è totale ed esclusiva, poiché i tipi di articolo sono ben definiti e tra loro distinti.



La gerarchia di generalizzazione dell'entità ARTICOLO è stata ristrutturata con un collasso verso l'alto.
Dato che l'entità REVISIONE partecipa con cardinalità massima 1 alla relazione **Assegnare**, non è necessario che il suo attributo *ID* faccia parte della chiave.

Relazione	Attributi
CONFERENZA	Sigla, Nome, Sede, URL
SPONSOR	ID, Nome
ORGANIZZATORE	ID, Nome, Affiliazione, Indirizzo, EMail, Telefono, Fax
AREARICERCA	Sigla, Descrizione
ARTICOLO	ID,SiglaConferenza, Titolo, Stato, Selettore, IDAutoreContatto
AUTORE	ID, Nome, Affiliazione, Indirizzo, Telefono, EMail
PARTNER	Nome, Indirizzo
REVISIONE	ID, Originalità, Significatività, Qualità, Globale, Commenti
FINANZIARE	IDSponsor, SiglaConferenza, Data, Importo
ORGANIZZARE	IDOrganizzatore, SiglaArea, SiglaConferenza
SCRIVERE	IDAutore, IDArticolo, SiglaConferenza
CONTRIBUIRE	NomePartner, IDArticolo, SiglaConferenza
ASSEGNARE	IDOrganizzatore, IDArticolo, SiglaConferenza, IDRevisione

Chiave Esterna	Referenzia
ARTICOLO.SiglaConferenza	CONFERENZA.Sigla
ARTICOLO.IDAutoreContatto	AUTORE.ID
FINANZIARE.IDSponsor	SPONSOR.ID
FINANZIARE.SiglaConferenza	CONFERENZA.Sigla
ORGANIZZARE.IDOrganizzatore	ORGANIZZATORE.ID
ORGANIZZARE.SiglaArea	AREARICERCA.Sigla
ORGANIZZARE.SiglaConferenza	CONFERENZA.Sigla
SCRIVERE.IDAutore	AUTORE.ID
SCRIVERE.{IDArticolo, SiglaConferenza}	ARTICOLO.{ID, SiglaConferenza}
CONTRIBUIRE.NomePartner	PARTNER.Nome
CONTRIBUIRE.{IDArticolo, SiglaConferenza}	ARTICOLO.{ID, SiglaConferenza}
ASSEGNARE.IDOrganizzatore	ORGANIZZATORE.ID
ASSEGNARE.{IDArticolo, SiglaConferenza}	ARTICOLO.{ID, SiglaConferenza}
ASSEGNARE.IDRevisione	REVISIONE.ID

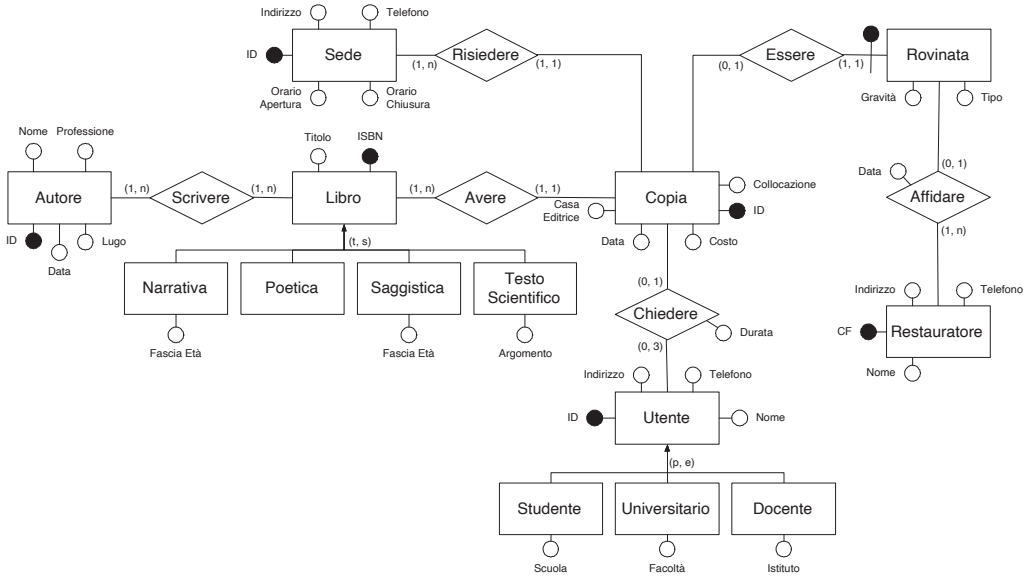
3.4.3 Biblioteca

La biblioteca della città Free-Read ha diverse sedi di quartiere per le quali si vuole sapere l'indirizzo, il numero di telefono, l'orario di apertura e l'orario di chiusura. Per i libri di proprietà della biblioteca si è interessati a gestire le informazioni relative a: titolo, autori, tipo e ISBN (identificativo univoco). Per gli autori dei libri si vuole sapere il nome, data e luogo di nascita e la professione. Si noti che un libro può avere più autori e, viceversa, un autore può aver scritto più libri. I possibili tipi di libri sono narrativa, saggistica, poetica e testi scientifici (si ricordi che un libro può appartenere anche a più tipi). Per i libri di narrativa e saggistica è necessario gestire l'informazione relativa alla fascia di età dei lettori a cui tali libri si rivolgono. Per i testi scientifici, si è invece interessati alla informazione circa l'argomento (es., informatica, matematica e così via). Di ciascun libro possono essere a disposizione più copie contrassegnate da un proprio codice. Per ogni copia si vuole tenere traccia delle informazioni relative alla collocazione, alla casa editrice, alla data di pubblicazione, al costo e a quale sede risulta in carico. Alcune copie possono essere rovinate; per queste si vuole mantenere anche l'informazione relativa al tipo di difetto e alla gravità del difetto stesso. Le copie rovinate possono essere sottoposte a restauro. Il restauro viene

eseguito da uno ed un solo restauratore di cui si conosce il nome, l'indirizzo ed il recapito telefonico. Delle copie in restauro si vuole anche mantenere traccia della data in cui le copie sono entrate in restauro.
Gli utenti della biblioteca hanno associato un nome, un codice identificativo, l'indirizzo di residenza ed il recapito telefonico. Gli utenti sono inoltre classificati in: studenti non universitari, di cui si vuole sapere il nome della scuola presso cui sono iscritti; studenti universitari, di cui si vuole sapere la facoltà presso cui sono iscritti; docenti, di cui si vuole sapere il nome della scuola e/o università presso la quale insegnano; ed altri utenti. Gli utenti possono chiedere in prestito al massimo tre copie di libri. La richiesta può essere fatta presso una qualunque delle sedi della biblioteca. Si vuole tenere traccia dello stato corrente dei prestiti di ciascun utente, mantenendo informazioni sulle copie chieste in prestito, sulla sede in cui le ha chieste e sulla durata del prestito.

Soluzione

La gerarchia di generalizzazione dell'entità Libro è totale e sovrapposta. Infatti, ad esempio, un saggio potrebbe essere anche un testo scientifico. La gerarchia di generalizzazione dell'entità Utente è parziale ed esclusiva perché i requisiti esplicitano che esistono anche altri tipi di utenti.



Le gerarchie di generalizzazione delle entità Libro e Utente sono state ristrutturate con un collasso verso l'alto. Per la gerarchia che rappresenta i vari tipi di libri nell'entità Libro, risultante dalla ristrutturazione, sono stati aggiunti i selettori *SelNarrativa*, *SelPoetica*, *SelSaggistica* e *SelScientifica* per poter gestire la sovrapposizione.
L'entità Rovinata acquisisce come chiave l'ID del libro, infatti logicamente rappresenta lo stato attuale di un libro (rovinato o integro). La rappresentazione adottata nel modello ER

garantisce di rappresentare il vincolo per cui solo i libri rovinati possono essere affidati ad un restauratore.

Relazione	Attributi
AUTORE	ID, Nome, Data, Luogo, Professione
LIBRO	ISBN, Titolo, Fascia, Argomento, SelNarrativa, SelPoetica, SelSaggistica, SelScientifica
COPIA	ID, Collocazione, Data, CasaEditrice, Costo, ISBNLibro, IDSede, IDUtente, Durata
ROVINATA	IDCopia, Gravità, Tipo, CFRestauratore, Data
RESTAURATORE	CF, Nome, Indirizzo, Telefono
SEDE	ID, Indirizzo, OrarioApertura, OrarioChiusura, Telefono
UTENTE	ID, Nome, Indirizzo, Telefono, Scuola, Facoltà, Istituto, Selettore
SCRIVERE	IDAutore, ISBNLibro

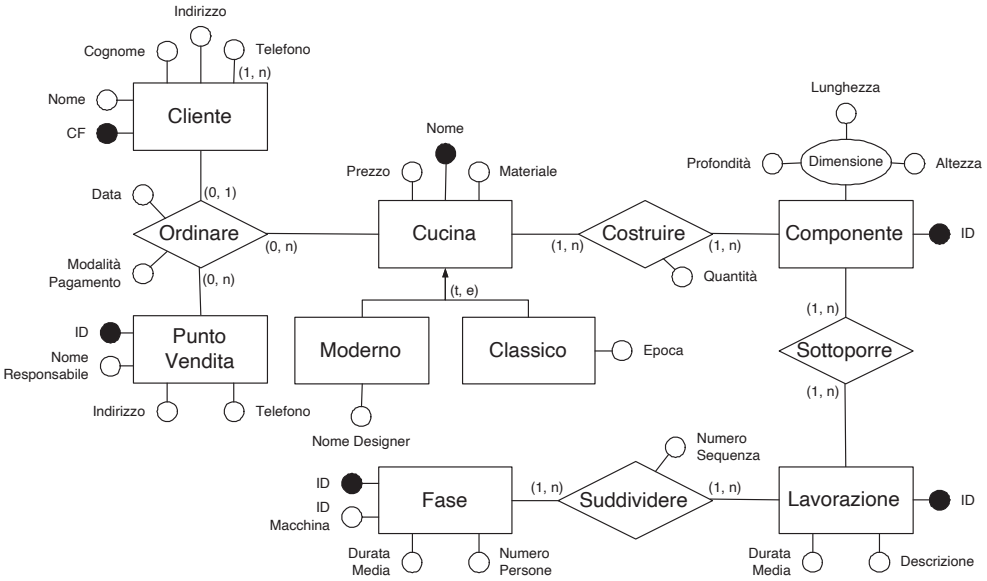
Chiave Esterna	Referenzia
COPIA.ISBNLibro	LIBRO.ISBN
COPIA.IDSede	SEDE.ID
COPIA.ID	ROVINATA.IDCopia
ROVINATA.CFRestauratore	RESTAURATORE.CF
SCRIVERE.IDAutore	AUTORE.ID
SCRIVERE.ISBNLibro	LIBRO.ISBN

3.4.4 Centro Cucine

Le cucine che l'azienda Scavo-line vende sono caratterizzate da un nome, prezzo di base e il tipo di materiale di cui sono composte. La gamma di cucine dell'azienda in esame è articolata su due aree di gusto: moderno contemporaneo e classico. Delle cucine appartenenti alla prima area si conosce il nome del designer che le ha disegnate. Delle cucine appartenenti alla seconda area si conosce invece l'epoca a cui si ispirano. Le cucine sono costituite da uno o più componenti ed uno stesso componente può far parte di diverse cucine. Per ciascun componente è necessario memorizzare in quale quantità è impiegato (es., 3 piani di lavoro per la cucina c1, 2 piani di lavoro per la cucina c2 e così via). Dei componenti è noto il codice e la dimensione (altezza, lunghezza e profondità). Ciascun componente viene sottoposto a delle lavorazioni di cui si conosce la durata media ed una breve descrizione. Ogni lavorazione si suddivide in più fasi e ciascuna fase può far parte di più lavorazioni. Per le fasi si deve tenere traccia del numero di sequenza, che varia a seconda della lavorazione di cui fa parte, del codice della macchina usata durante la fase, durata media e numero di persone necessarie per monitorare la fase.
La struttura distributiva dell'azienda è composta da punti vendita per i quali si deve tenere traccia del nome del responsabile, indirizzo e numero di telefono. Per i clienti si è interessati a gestire informazioni anagrafiche quali nome, cognome, codice fiscale, indirizzo ed i numeri di telefono (ogni cliente ha almeno un numero di telefono). Ogni cliente effettua un ordine per una data cucina in un dato punto vendita. Dell'ordine si conosce la data e la modalità di pagamento. Si suppone che un cliente può fare al più un unico ordine.

Soluzione

La gerarchia di generalizzazione dell'entità CUCINA è totale ed esclusiva, come esplicitato nei requisiti.



La gerarchia di generalizzazione dell'entità CUCINA è stata ristrutturata con un collasso verso l'alto.

Relazione	**Attributi**
CUCINA	Nome, Prezzo, Materiale, NomeDesigner, Epoca, Selettore
COMPONENTE	ID, Altezza, Lunghezza, Profondità
LAVORAZIONE	ID, DurataMedia, Descrizione
FASE	ID, IDMacchina, DurataMedia, NumeroPersone
PUNTOVENDITA	ID, NomeResponsabile, Indirizzo, Telefono
CLIENTE	CF, Nome, Cognome, Indirizzo
TELEFONO	Numero, CFCliente
COSTITUIRE	NomeCucina, IDComponente, Quantità
SOTTOPORRE	IDComponente, IDLavorazione
SUDDIVIDERE	IDFase, IDLavorazione, NumeroSequenza
ORDINARE	CFCliente, NomeCucina, IDPuntoVendita, Data, ModalitàPagamento

Chiave Esterna	Referenzia
TELEFONO.CFCliente	CLIENTE.CF
COSTITUIRE.NomeCucina	CUCINA.Nome
COSTITUIRE.IDComponente	COMPONENTE.ID
SOTTOPORRE.IDComponente	COMPONENTE.ID
SOTTOPORRE.IDLavorazione	LAVORAZIONE.ID
SUDDIVIDERE.IDFase	FASE.ID
SUDDIVIDERE.IDLavorazione	LAVORAZIONE.ID
ORDINARE.CFCliente	CLIENTE.CF
ORDINARE.NomeCucina	CUCINA.Nome
ORDINARE.IDPuntoVendita	PUNTOVENDITA.ID

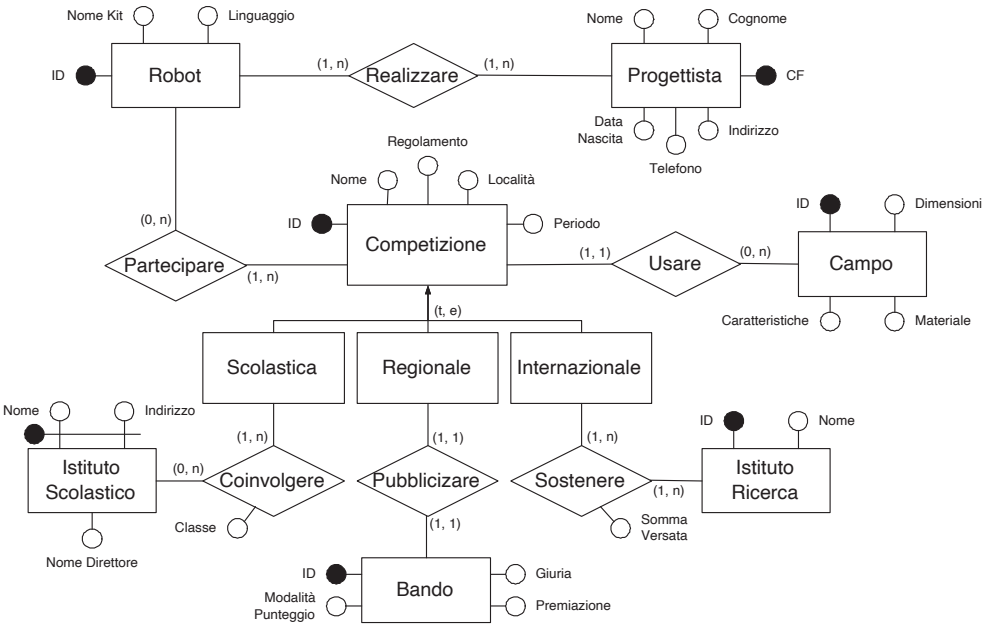
3.4.5 CyberGame

Il centro CyberGame (CG) organizza delle competizioni tra robot. Ogni competizione ha un preciso nome e regolamento e si svolge in una data località ed in un dato periodo. Per ogni competizione viene inoltre fornita una descrizione del campo da gioco che include informazioni relative alle dimensioni, caratteristiche del campo ed il materiale con cui è realizzato. Si noti che uno stesso campo da gioco può essere utilizzato all'interno di diverse competizioni. Le competizioni si suddividono in tre categorie: scolastiche, regionali e internazionali. Le gare scolastiche coinvolgono uno o più istituti scolastici di cui è noto il nome, l'indirizzo, il nome del direttore. Ogni istituto può partecipare a ogni competizione con al più una classe, ma può partecipare a più competizioni, anche con classi diverse. Le gare regionali sono pubblicizzate tramite un apposito bando che include informazioni circa le modalità di assegnazione dei punteggi di merito, la composizione della giuria e la premiazione. Per ultimo, le gare internazionali sono sostenute da istituti di ricerca di cui si conosce il nome e l'eventuale somma versata come sponsorizzazione della competizione. Un medesimo istituto può sponsorizzare più competizioni e per ognuna di esse l'ammontare del contributo versato può essere diverso.

Per ogni robot utilizzato nelle competizioni si conosce il nome del kit robotico usato per la sua costruzione, il linguaggio con cui è stato realizzato il programma che lo controlla, la competizione a cui ha partecipato ed il progettista o i progettisti che lo hanno realizzato. In particolare, per ogni progettista sono note le usuali informazioni anagrafiche (nome, cognome e data di nascita), l'indirizzo e un numero di telefono.

Soluzione

La gerarchia di generalizzazione dell'entità COMPETIZIONE è totale ed esclusiva, come esplicitato nei requisiti.



La gerarchia di generalizzazione dell'entità COMPETIZIONE è stata ristrutturata mantenendo tutte le entità.
Nella traduzione al modello logico, le due entità REGIONALE e BANDO sono state accorpate in un'unica tabella REGIONALE, dato che legate da una relazione 1 : 1.

Relazione	**Attributi**
COMPETIZIONE	ID, Nome, Regolamento, Località, Periodo, IDCampo
CAMPO	ID, Dimensioni, Materiale, Caratteristiche
SCOLASTICA	ID
REGIONALE	ID, IDBando, ModalitàPunteggi, Giuria, Premiazione
INTERNAZIONALE	ID
ISTITUTOSCOLASTICO	Nome, Indirizzo, NomeDirettore
ISTITUTORICERCA	ID, Nome
ROBOT	ID, NomeKit, Linguaggio
PROGETTISTA	CF, Nome, Cognome, DataNascita, Indirizzo, Telefono
COINVOLGERE	IDScolastica, NomeIstitutoScolastico, IndirizzoIstitutoScolastico, Classe
SOSTENERE	IDInternazionale, IDIstitutoRicerca, SommaVersata
PARTECIPARE	IDRobot, IDCompetizione
REALIZZARE	IDRobot, CFProgettista

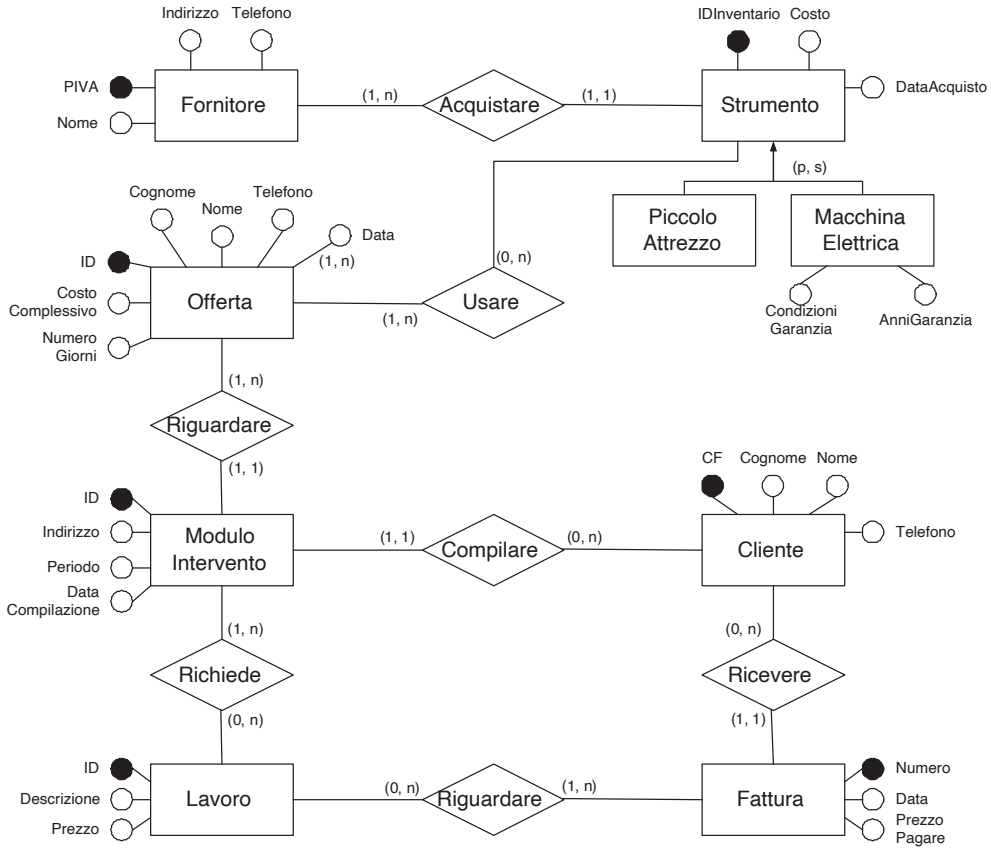
Chiave Esterna	Referenzia
COMPETIZIONE.IDCampo	CAMPO.ID
SCOLASTICA.ID	COMPETIZIONE.ID
REGIONALE.ID	COMPETIZIONE.ID
INTERNAZIONALE.ID	COMPETIZIONE.ID
COINVOLGERE.IDScolastica	SCOLASTICA.ID
COINVOLGERE.{NomeIstitutoScolastico, IndirizzoIstitutoScolastico}	ISTITUTOSCOLASTICO.{Nome, Indirizzo}
SOSTENERE.IDInternazionale	INTERNAZIONALE.ID
SOSTENERE.IDIstitutoRicerca	ISTITUTORICERCA.ID
PARTECIPARE.IDRobot	ROBOT.ID
PARTECIPARE.IDCompetizione	COMPETIZIONE.ID
REALIZZARE.IDRobot	ROBOT.ID
REALIZZARE.CFProgettista	PROGETTISTA.CF

3.4.6 Erba del Vicino

L'azienda L'Erba del Vicino è sempre più Corta (EVC), specializzata nella cura di giardini e terrazze, dispone dei più moderni strumenti per la cura di piante e fiori. Fra questi strumenti in particolare si riscontrano due grosse categorie: piccoli attrezzi (es., cesoie) e macchine elettriche. Gli strumenti sono tutti inventariati per cui per ognuno di essi si conosce il codice di inventario, la data di acquisto, il costo, il fornitore e, per i macchinari elettrici, le condizioni ed il numero di anni di garanzia. Dei fornitori, caratterizzati dalla partita IVA, sono noti il nome, l'indirizzo ed il numero di telefono della sede principale.
I clienti dell'azienda EVC possono prenotare un suo intervento compilando un "modulo di intervento" sul quale sono riportate le seguenti informazioni: nome, cognome e recapito telefonico del cliente, l'indirizzo del giardino o terrazzo su cui eseguire i lavori, l'elenco dei lavori richiesti, il periodo in cui effettuare tali lavori e per finire la data di compilazione del modulo. Per ciascuno dei lavori che EVC offre sono note una descrizione ed il prezzo unitario. Si noti che uno stesso cliente può compilare più moduli. A fronte di uno o più moduli di intervento compilati da uno stesso cliente, l'azienda EVC redige una offerta nella quale viene specificato il costo complessivo dei lavori, gli strumenti richiesti, il numero di giorni necessari ad eseguire i lavori, le date di intervento ed il nome, cognome e numero di telefono della persona che ha compilato l'offerta. Dopo che i lavori sono stati eseguiti, il cliente riceve una fattura la quale riporta la data di compilazione, l'elenco dei lavori ed il prezzo finale da pagare.

Soluzione

La gerarchia di generalizzazione dell'entità STRUMENTO è parziale e sovrapposta, dato che un piccolo attrezzo può essere elettrico e che possono esistere anche grandi attrezzi che non lo sono.



La gerarchia di generalizzazione dell'entità STRUMENTO è stata ristrutturata con un collasso verso l'alto. Per garantire la proprietà di sovrapposizione presente nella gerarchia stessa, all'entità STRUMENTO sono stati aggiunti gli attributi *SelPiccoloAttrezzo* e *SelMacchinaElettrica*.

Relazione	Attributi
STRUMENTO	IDInventario, DataAcquisto, Costo, CondizioniGaranzia, AnniGaranzia, SelPiccoloAttrezzo, SelMacchinaElettrica, PIVAFornitore
FORNITORE	PIVA, Nome, Indirizzo, Telefono
CLIENTE	CF, Nome, Cognome, Telefono
MODULOINTERVENTO	ID, DataCompilazione, Periodo, Indirizzo, CFCliente, IDOfferta
OFFERTA	ID, CostoComplessivo, NumeroGiorni, Nome, Cognome, Telefono
DATAOFFERTA	Data, IDOfferta
LAVORO	ID, Descrizione, Prezzo
FATTURA	Numero, Data, PrezzoPagare, CFCliente
USARE	IDStrumento, IDOfferta
RICHIEDERE	IDModuloIntervento, IDLavoro
RIGUARDARE	NumeroFattura, IDLavoro

Chiave Esterna	Referenzia
STRUMENTO.PIVAFornitore	FORNITORE.PIVA
MODULOINTERVENTO.CFCliente	CLIENTE.CF
MODULOINTERVENTO.IDOfferta	OFFERTA.ID
DATAOFFERTA.IDOfferta	OFFERTA.ID
FATTURA.CFCliente	CLIENTE.CF
USARE.IDStrumento	STRUMENTO.ID
USARE.IDOfferta	OFFERTA.ID
RICHIEDERE.IDModuloIntervento	MODULOINTERVENTO.ID
RICHIEDERE.IDLavoro	LAVORO.ID
RIGUARDARE.NumeroFattura	FATTURA.Numero
RIGUARDARE.IDLavoro	LAVORO.ID

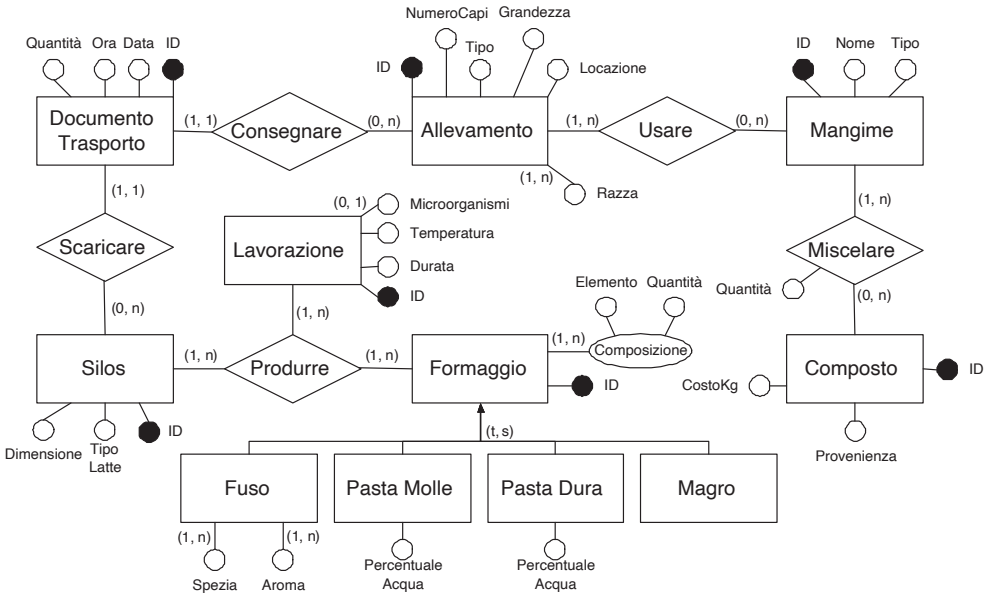
3.4.7 Fermento Vivo

L'azienda FermentoVivo (FV), specializzata nella produzione di formaggi, dispone di diversi allevamenti di bestiame dislocati nella regione Lombardia. Ciascun allevamento dispone di un certo numero di capi di bestiame (mucche, pecore e capre). Gli allevamenti possono occuparsi di una sola o di più razze diverse; gli allevamenti possono essere di vario tipo (es., biologici, aperti e così via). Oltre a queste informazioni, è inoltre nota la grandezza dell'allevamento e la sua locazione. Gli allevamenti fanno uso di mangimi, che l'azienda stessa prepara, caratterizzati da un nome e dal tipo. Ciascun mangime è ottenuto miscelando più composti di cui si conosce la provenienza, il costo al chilo e la quantità di composto necessaria per ottenere una tonnellata di mangime. Gli allevamenti consegnano il latte prodotto all'azienda FV il quale viene raccolto in appositi silos. Ogni silos è caratterizzato dalla dimensione e dal tipo di latte che può contenere. Di ogni consegna si tiene traccia, tramite il relativo documento di trasporto, della data e ora in cui viene effettuata, della quantità consegnata, dell'allevamento che ha prodotto il latte e del silos all'interno del quale viene scaricato il latte.
FV produce diverse tipologie di formaggi: fusi, a pasta molle, a pasta dura e magri. Ogni tipologia di formaggio è identificata da un codice e ha associata una composizione (la composizione è un insieme di coppie elemento e quantità presente in 100 grammi di parte edibile).

Per la tipologia di formaggi fusi sono inoltre riportate le spezie e gli aromi che possono essere utilizzati mentre per i formaggi a pasta molle e dura è nota la percentuale di acqua. Le diverse tipologie di formaggi sono ottenute applicando delle lavorazioni al latte di un certo tipo (prelevato quindi da determinati silos). Ogni lavorazione è caratterizzata dalla temperatura di lavorazione del latte, gli eventuali microrganismi della fermentazione che devono essere aggiunti e dalla durata. Si noti che: (1) una stessa lavorazione può essere utilizzata per produrre più tipologie di formaggio e vice versa; (2) diversi tipi di latte (raccolti quindi in diversi silos) possono essere usati per la produzione di una stessa tipologia di formaggio; (3) tipi di latte diverso sono soggetti a diverse lavorazioni.

Soluzione

La gerarchia di generalizzazione dell'entità FORMAGGIO è totale e sovrapposta, dato che i formaggi fusi, a pasta molle e a pasta dura possono essere anche magri.



La gerarchia di generalizzazione dell'entità FORMAGGIO è stata ristrutturata con un collasso verso l'alto. Per garantire la proprietà di sovrapposizione presente nella gerarchia stessa, all'entità FORMAGGIO sono stati aggiunti gli attributi *SelFuso*, *SelPastaMolle*, *SelPastaDura* e *SelMagro*.

Relazione	Attributi
ALLEVAMENTO	ID, NumeroCapi, Tipo, Grandezza, Locazione
RAZZA	IDAllevamento, Razza
MANGIME	ID, Nome, Tipo
COMPOSTO	ID, Provenienza, CostoKg
DOCUMENTOTRASPORTO	ID, Data, Ora, Quantità, IDAllevamento, IDSilos
SILOS	ID, Dimensione, TipoLatte
FORMAGGIO	ID, PercentualeAcqua, SelFuso, SelPastaMolle, SelPastaDura, SelMagro
SPEZIA	IDFormaggio, Spezia
AROMA	IDFormaggio, Aroma
COMPOSIZIONE	IDFormaggio, Elemento, Quantità
LAVORAZIONE	ID, Temperatura, Microorganismi, Durata
MISCELARE	IDComposto, IDMangime, Quantità
USARE	IDAllevamento, IDMangime
PRODURRE	IDSilos, IDLavorazione, IDFormaggio

Chiave Esterna	Referenzia
RAZZA.IDAllevamento	ALLEVAMENTO.ID
DOCUMENTOTRASPORTO.IDAllevamento	ALLEVAMENTO.ID
DOCUMENTOTRASPORTO.IDSilos	SILOS.ID
SPEZIA.IDFormaggio	FORMAGGIO.ID
AROMA.IDFormaggio	FORMAGGIO.ID
COMPOSIZIONE.IDFormaggio	FORMAGGIO.ID
MISCELARE.IDComposto	COMPOSTO.ID
MISCELARE.IDMangime	MANGIME.ID
USARE.IDAllevamento	ALLEVAMENTO.ID
USARE.IDMangime	MANGIME.ID
PRODURRE.IDSilos	SILOS.ID
PRODURRE.IDLavorazione	LAVORAZIONE.ID
PRODURRE.IDFormaggio	FORMAGGIO.ID

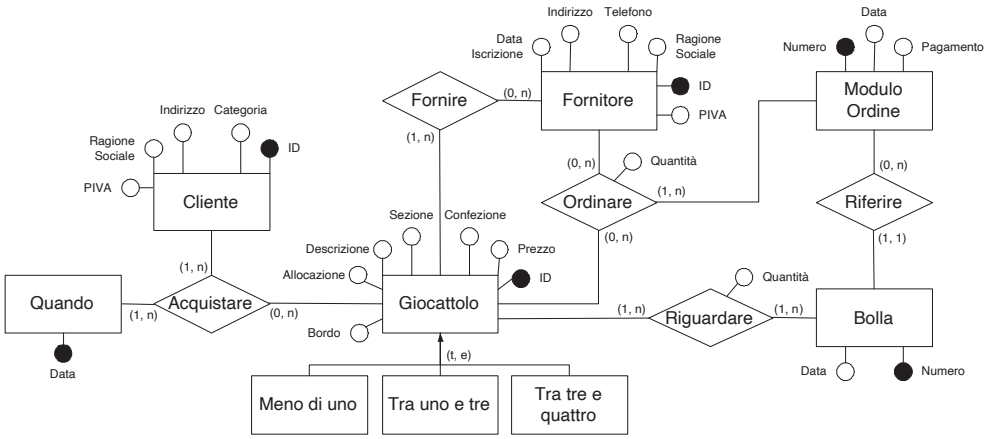
3.4.8 Giocattoli

I giocattoli che l'azienda Gioca Bimbo distribuisce sono univocamente identificati da un codice e sono caratterizzati da informazioni relative al prezzo, al tipo di allocazione (cassetto, vano o altro) nella quale vengono posti, una descrizione, la forma del bordo, la sezione e la modalità di confezionamento. I giocattoli possono essere suddivisi in tre diverse tipologie: (1) giocattoli adatti ai bambini di età inferiore ad un anno; (2) giocattoli adatti ai bambini di età compresa tra uno e tre anni; (3) giocattoli adatti ai bambini di età compresa tra i tre e i quattro anni. L'azienda si avvale di fornitori i quali possono fornire diversi giocattoli e, viceversa, gli stessi giocattoli possono essere forniti da più fornitori. Per i fornitori si è interessati a gestire il seguente insieme di informazioni: codice fornitore, ragione sociale, partita IVA, indirizzo della sede legale, numero telefonico e data iscrizione. Gli ordini ai fornitori sono espressi utilizzando un opportuno modulo d'ordine nel quale si specificano le informazioni relative al numero d'ordine, la data di emissione ordine ed i termini di pagamento. Ciascun modulo d'ordine fa riferimento ad un fornitore ed a uno o più giocattoli di cui si conosce la quantità ordinata. I giocattoli gestiti dall'azienda possono essere distribuiti a diverse categorie di clienti quali supermercati, catene di negozi o singoli negozi, centri commerciali e così via. Questi clienti, caratterizzati da un codice, possono acquistare più

giocattoli; dell'acquisto si deve tenere traccia della data. Per i clienti si è anche interessati a mantenere informazioni circa la ragione sociale, l'indirizzo e la partita IVA. L'ingresso nell'azienda di giocattoli è documentato da una bolla di accompagnamento nella quale viene riportato il numero e la data. Ciascuna bolla fa riferimento ad un ben preciso modulo d'ordine; la bolla fa anche riferimento ad un certo insieme di giocattoli di cui si conosce la quantità effettivamente consegnata. Non è detto che il quantitativo di giocattoli consegnati corrisponda a quello specificato nel corrispondente ordine fornitore. Per tale ragione, lo stesso modulo d'ordine può fare riferimento a più bolle.

Soluzione

La gerarchia di generalizzazione dell'entità GIOCATTOLO è totale ed esclusiva, dato che si suppone che un giocattolo sia adatto ad una sola fascia di età ed è specificato dal testo che l'azienda si occupa solo di quei tipi di giocattolo.
Per poter permettere ad un cliente di acquistare più di una volta lo stesso giocattolo, è stata aggiunta l'entità QUANDO. In questo modo, nella traduzione al modello logico, la data dell'acquisto può far parte della chiave della tabella ACQUISTARE. L'entità QUANDO, invece, non compare nel modello logico perché avrebbe un solo attributo, già parte della chiave della tabella ACQUISTARE.



La gerarchia di generalizzazione dell'entità GIOCATTOLO è stata ristrutturata con un collasso verso l'alto.
Si noti che *IDFornitore* non è parte della chiave per ORDINARE perché ciascun modulo d'ordine fa riferimento ad un solo fornitore.

Relazione	Attributi
GIOCATTOLO	ID, Prezzo, Allocazione , Descrizione, Bordo, Sezione, Confezione, Selettore
FORNITORE	ID, RagioneSociale, PIVA, Indirizzo, Telefono, DataIscrizione
MODULOORDINE	Numero, Data, Pagamento
CLIENTE	ID, Categoria, RagioneSociale, PIVA, Indirizzo
BOLLA	Numero, Data, NumeroModuloOrdine
FORNIRE	IDGiocattolo, IDFornitore
ORDINARE	IDGiocattolo, NumeroModuloOrdine, IDFornitore, Quantità
ACQUISTARE	IDCliente, IDGiocattolo, Data
RIGUARDARE	IDGiocattolo, NumeroBolla, Quantità

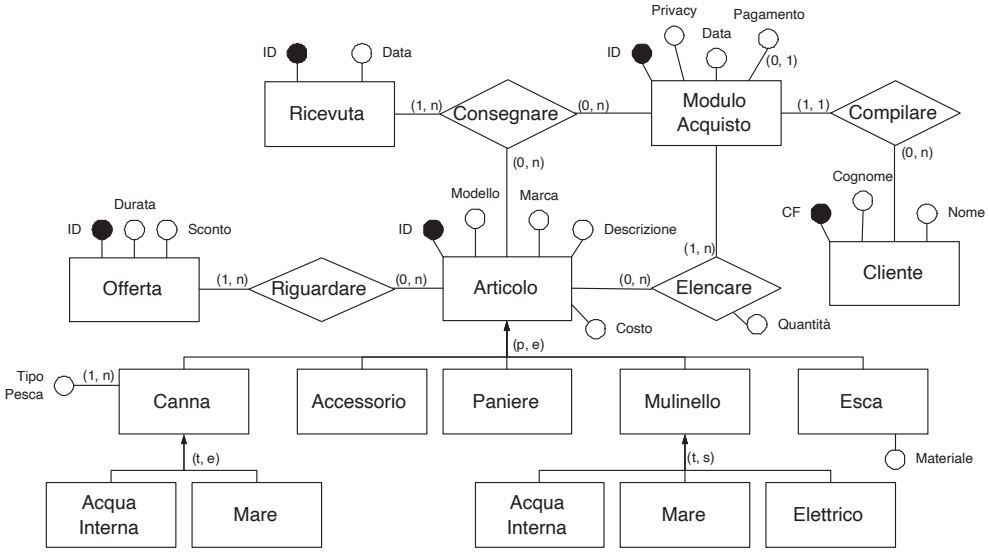
Chiave Esterna	Referenzia
BOLLA.NumeroModuloOrdine	MODULOORDINE.Numero
FORNIRE.IDGiocattolo	GIOCATTOLO.ID
FORNIRE.IDFornitore	FORNITORE.ID
ORDINARE.IDGiocattolo	GIOCATTOLO.ID
ORDINARE.NumeroModuloOrdine	MODULOORDINE.Numero
ORDINARE.IDFornitore	FORNITORE.ID
ACQUISTARE.IDCliente	CLIENTE.ID
ACQUISTARE.IDGiocattolo	GIOCATTOLO.ID
RIGUARDARE.IDGiocattolo	GIOCATTOLO.ID
RIGUARDARE.NumeroBolla	BOLLA.Numero

3.4.9 Articoli da Pesca

Lenza è un negozio specializzato in articoli sportivi per la pesca in mare. Gli articoli venduti sono suddivisi in cinque categorie principali: canne, mulinelli, accessori, panieri ed esche. Le canne si suddividono a loro volta in canne per acque interne e per mare, mentre i mulinelli possono essere per acque interne, per mare ed elettrici. Di ogni articolo viene indicato il modello, la marca, il costo e una breve descrizione. Per le canne viene inoltre specificato il tipo (o tipi) di pesca per cui sono più adatte, e per le esche viene indicato il tipo di materiale di cui sono composte. Il negozio, periodicamente, propone delle offerte che riguardano alcuni fra gli articoli in vendita. Le offerte hanno una durata limitata e sono caratterizzate dalla percentuale di sconto da applicare al prezzo di listino per i prodotti che l'offerta coinvolge. I clienti di Lenza, caratterizzati da nome, cognome e CF, possono effettuare acquisti sia recandosi direttamente nel negozio sia tramite il portale Web. In entrambi i casi il cliente deve compilare un modulo di acquisto, sul quale indicare la data di compilazione, i propri dati anagrafici, se dare o negare il consenso al trattamento dei dati personali e, nel caso in cui il modulo sia una form web, anche la modalità di pagamento. Nel modulo devono anche essere elencati gli articoli che si intende acquistare e, per ciascuno, la quantità. In seguito alla compilazione dei moduli di acquisto, Lenza provvede alla consegna della merce ai clienti. Per ciascuna consegna, viene rilasciata una ricevuta, completa di numero e data, con l'elenco degli articoli consegnati ed il riferimento ai moduli di acquisto cui fa seguito. Si noti che: (1) ciascun cliente può compilare più di un modulo di acquisto, (2) gli articoli elencati in un dato modulo di acquisto possono essere consegnati in diversi momenti (e quindi risultare su ricevute diverse) e (3) ciascuna consegna può riguardare articoli elencati in più di un modulo di acquisto.

Soluzione

La gerarchia di generalizzazione dell'entità ARTICOLO è parziale ed esclusiva dato che le categorie considerate sono separate fra loro, ma elencano solo i principali tipi di prodotti trattati. La gerarchia di generalizzazione dell'entità CANNE è invece totale ed esclusiva. La gerarchia di generalizzazione dell'entità MULINELLI è totale e sovrapposta, poiché sia i mulinelli per acque interne sia per acque esterne possono essere elettrici.



La gerarchia di generalizzazione dell'entità ARTICOLO è stata ristrutturata preservando tutte le entità coinvolte. Le altre gerarchie sono state ristrutturate con un collasso verso l'alto. In particolare, per garantire la proprietà di sovrapposizione presente nella gerarchia, all'entità MULINELLO sono stati aggiunti gli attributi *SelAcquaInterna*, *SelMare* e *SelElettrico*.
L'attributo *Pagamento* dell'entità MODULOACQUISTO, essendo opzionale, può assumere il valore *NULL*.

Relazione	**Attributi**
ARTICOLO	ID, Modello, Marca, Costo, Descrizione
CANNA	ID, Selettore
MULINELLO	ID, SelAcquaInterna, SelMare, SelElettrico
ACCESSORIO	ID
PANIERE	ID
ESCA	ID, Materiale
PESCA	IDCanna, TipoPesca
OFFERTA	ID, Durata, Sconto
CLIENTE	CF, Nome, Cognome
MODULOACQUISTO	ID, Data, Privacy, Pagamento, CFCliente
RICEVUTA	ID, Data
RIGUARDARE	IDArticolo, IDOfferta
ELENCARE	IDArticolo, IDModuloAcquisto, Quantità
CONSEGNARE	IDArticolo, IDModuloAcquisto, IDRicevuta

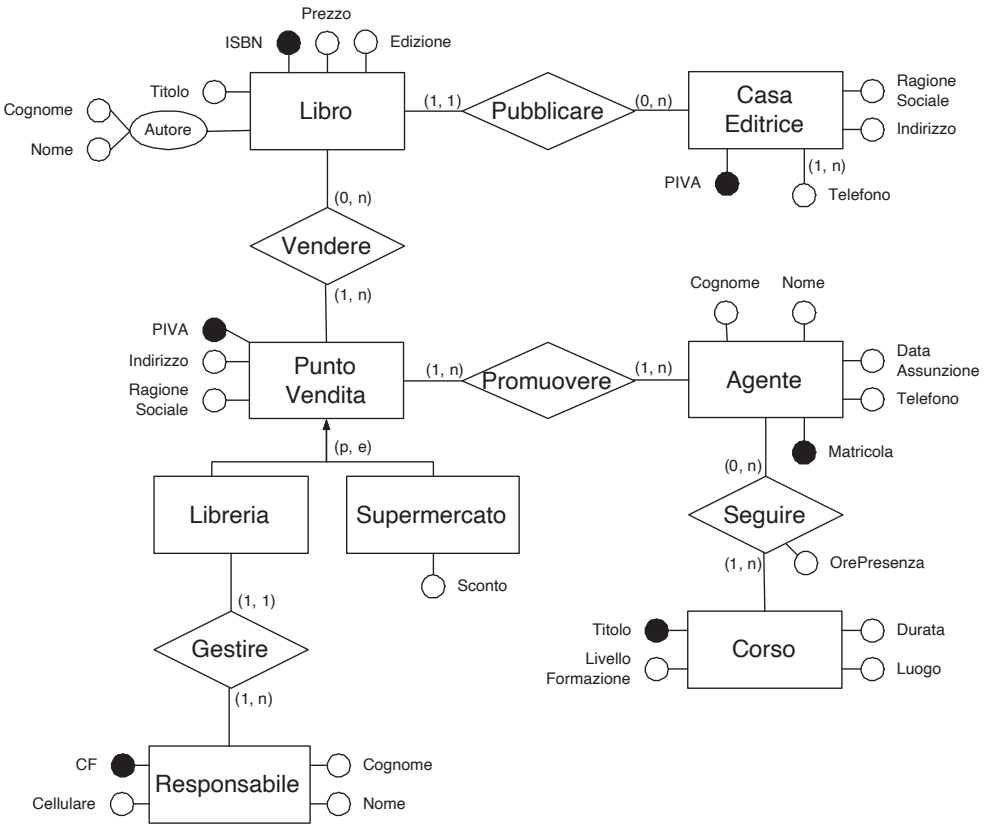
Chiave Esterna	Referenzia
CANNA.ID	ARTICOLO.ID
MULINELLO.ID	ARTICOLO.ID
ACCESSORIO.ID	ARTICOLO.ID
PANIERE.ID	ARTICOLO.ID
ESCA.ID	ARTICOLO.ID
PESCA.IDCanna	CANNA.ID
MODULOACQUISTO.CFCliente	CLIENTE.CF
RIGUARDARE.IDArticolo	ARTICOLO.ID
RIGUARDARE.IDOfferta	OFFERTA.ID
ELENCARE.IDArticolo	ARTICOLO.ID
ELENCARE.IDModuloAcquisto	MODULOACQUISTO.ID
CONSEGNARE.IDArticolo	ARTICOLO.ID
CONSEGNARE.IDModuloAcquisto	MODULOACQUISTO.ID
CONSEGNARE.IDRicevuta	RICEVUTA.ID

3.4.10 Libreria

La ditta LeggimiBene (LB) si occupa della distribuzione di libri, di ciascuno dei quali si conosce: il codice ISBN, unico per ciascun libro, il titolo, l'autore (cognome e nome), l'edizione e il prezzo di copertina. Ciascun libro è pubblicato da una sola casa editrice, della quale si vuole tenere traccia di: partita Iva, ragione sociale, indirizzo e numeri di telefono (uno o più).
I libri vengono distribuiti a dei punti vendita dislocati su tutto il territorio nazionale. Per ciascun punto vendita si conoscono partita Iva, ragione sociale e indirizzo. Tra i punti vendita, i più importanti sono librerie e supermercati, di questi ultimi si vuole tenere traccia della massima percentuale di sconto applicabile ai prezzi di copertina. Delle librerie, invece, si conosce il responsabile, caratterizzato da codice fiscale, cognome, nome e numero di telefono cellulare. Si noti che ciascuna libreria può avere un solo responsabile, mentre uno stesso responsabile può dirigere più librerie. La LB distribuisce a ciascun punto vendita, secondo un contratto non di esclusiva (quindi lo stesso libro può essere venduto in più punti vendita), un insieme di libri.
Per promuovere la sua attività nei punti vendita, la LB si avvale di un gruppo agenti per i quali si deve tenere traccia di: matricola, cognome, nome, data di assunzione, numero di telefono e insieme di punti vendita di cui si occupa. Periodicamente, la LB organizza dei corsi di aggiornamento per i propri agenti, al fine di migliorare la loro attività di promozione. Ogni corso è caratterizzato da titolo, durata (espressa in ore), luogo e livello di formazione. Per ogni agente si vuole tener traccia dei corsi seguiti, e per ciascun corso, l'indicazione delle ore di presenza.

Soluzione

La gerarchia di generalizzazione dell'entità Punto Vendita è parziale ed esclusiva, dato che librerie e supermercati sono solo le tipologie più importanti.



La gerarchia di generalizzazione dell'entità PUNTOVENDITA è stata ristrutturata con un collasso verso l'alto.

Relazione	**Attributi**
LIBRO	ISBN, Titolo, Edizione, Prezzo, CognomeAutore, NomeAutore, PIVACasaEditrice
CASAEDITRICE	PIVA, RagioneSociale, Indirizzo
TELEFONO	Numero, PIVACasaEditrice
PUNTOVENDITA	PIVA, RagioneSociale, Indirizzo, Sconto, Selettore, CFResponsabile
RESPONSABILE	CF, Cognome, Nome, Cellulare
AGENTE	Matricola, Cognome, Nome, DataAssunzione, Telefono
CORSO	Titolo, Durata, Luogo, LivelloFormazione
VENDERE	ISBNLibro, PIVAPuntoVendita
PROMUOVERE	PIVAPuntoVendita, MatricolaAgente
SEGUIRE	MatricolaAgente, TitoloCorso, OrePresenza

Chiave Esterna	Referenzia
LIBRO.PIVACasaEditrice	CASAEDITRICE.PIVA
TELEFONO.PIVACasaEditrice	CASAEDITRICE.PIVA
PUNTOVENDITA.CFResponsabile	RESPONSABILE.CF
VENDERE.ISBNLibro	LIBRO.ISBN
VENDERE.PIVAPuntoVendita	PUNTOVENDITA.PIVA
PROMUOVERE.PIVAPuntoVendita	PUNTOVENDITA.PIVA
PROMUOVERE.MatricolaAgente	AGENTE.Matricola
SEGUIRE.MatricolaAgente	AGENTE.Matricola
SEGUIRE.TitoloCorso	CORSO.Titolo

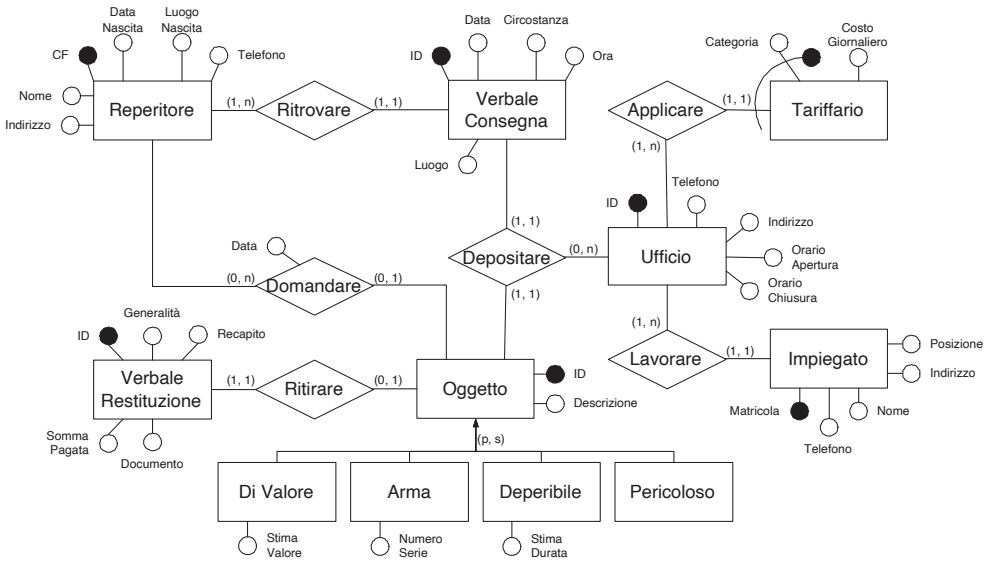
3.4.11 Oggetti Smarriti

Gli oggetti smarriti, rinvenuti sui mezzi di trasporto della azienda UrbanTravel (UT), sono raccolti presso uno degli uffici appositi situati nella città ove la UT svolge la sua attività. Di ciascun ufficio si conosce l'indirizzo, il numero di telefono, l'orario di apertura e chiusura e la lista degli impiegati che lavorano presso l'ufficio. Gli impiegati hanno associato un nome, indirizzo, numero di telefono, matricola e posizione. Ogni oggetto ritrovato e depositato presso uno degli uffici è sempre accompagnato da un verbale di consegna. Per permettere una rapida ricerca degli oggetti consegnati agli uffici, gli oggetti (di cui si mantiene una breve descrizione) sono suddivisi per gruppi merceologici quali: oggetti di valore, caratterizzati da una stima del valore della cosa ritrovata; armi, caratterizzati dal numero di serie; oggetti deperibili, caratterizzati da una stima della durata della cosa ritrovata; sostanze pericolose e altro. Sul verbale di consegna deve essere indicata la data e le circostanze del ritrovamento, il luogo e l'ora e le informazioni relative al reperitore di un oggetto. In particolare, per il reperitore, si vogliono mantenere nome, data e luogo di nascita, indirizzo e recapito telefonico. Trascorso un anno dal ritrovamento senza che alcuno si sia presentato a richiedere la restituzione dell'oggetto ritrovato, il reperitore può presentare domanda di ritiro per uno specifico oggetto. Poiché le domande di ritiro devono essere formulate entro tre mesi dalla data predetta, è necessario tenere traccia della data di presentazione della domanda.

Per gli oggetti restituiti (al legittimo proprietario o al reperitore) si deve preparare un verbale di restituzione nel quale si devono annotare le generalità, il recapito e gli estremi del documento di identificazione della persona a cui è stato consegnato l'oggetto. La persona che ritira l'oggetto deve pagare una somma, riportata sul verbale, a titolo di rimborso spese per la custodia. Ogni ufficio ha quindi associato un tariffario (che può variare da ufficio ad ufficio), caratterizzato da categoria merceologica e costo giornaliero di custodia per ciascuna categoria.

Soluzione

La gerarchia di generalizzazione dell'entità Oggetto è parziale e sovrapposta, infatti il testo indica che esistono anche altri tipi di oggetto. Inoltre un oggetto di valore potrebbe essere anche un'arma o un oggetto pericoloso.



La gerarchia di generalizzazione sull'entità OGGETTO è stata ristrutturata con un collasso verso l'alto. Per garantire la proprietà di sovrapposizione presente nella gerarchia stessa, all'entità OGGETTO sono stati aggiunti gli attributi *SelDiValore*, *SelArma*, *SelDeperibile* e *SelPericoloso*.
La relazione **Depositare** è stata tradotta con una tabella che ha come chiave solamente *IDOggetto*, dato che OGGETTO partecipa alla relazione con cardinalità $(1, 1)$.

Relazione	**Attributi**
UFFICIO	ID, Indirizzo, Telefono, OrarioApertura, OrarioChiusura
IMPIEGATO	Matricola, Nome, Indirizzo, Telefono, Posizione, IDUfficio
OGGETTO	ID, Descrizione, StimaValore, NumeroSerie, StimaDurata, SelDiValore, SelArma, SelDeperibile, SelPericoloso
VERBALECONSEGNA	ID, Data, Circostanza, Luogo, Ora, CFReperitore
REPERITORE	CF, Nome, DataNascita, LuogoNascita, Telefono, Indirizzo
VERBALERESTITUZIONE	ID, Generalità, Recapito, Documento, SommaPagata, IDOggetto
TARIFFARIO	IDUfficio, Categoria, CostoGiornaliero
DEPOSITARE	IDOggetto, IDVerbaleConsegna, IDUfficio
DOMANDARE	IDOggetto, CFReperitore, Data

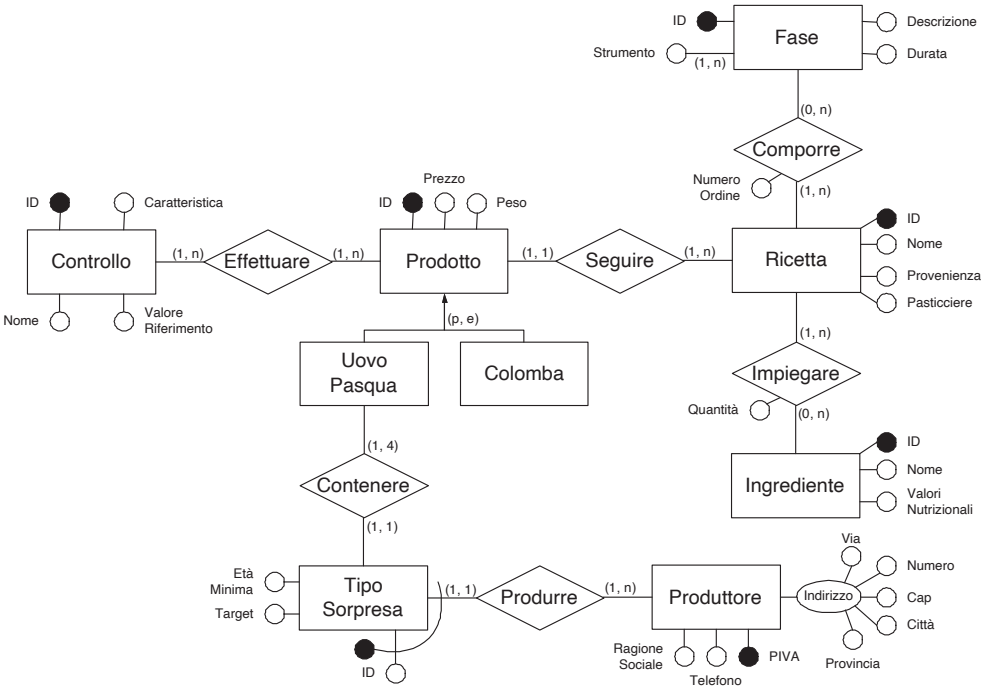
Chiave Esterna	**Referenzia**
IMPIEGATO.IDUfficio	UFFICIO.ID
VERBALECONSEGNA.CFReperitore	REPERITORE.CF
VERBALERESTITUZIONE.IDOggetto	OGGETTO.ID
TARIFFARIO.IDUfficio	UFFICIO.ID
DEPOSITARE.IDOggetto	OGGETTO.ID
DEPOSITARE.IDVerbaleConsegna	VERBALECONSEGNA.ID
DEPOSITARE.IDUfficio	UFFICIO.ID
DOMANDARE.IDOggetto	OGGETTO.ID
DOMANDARE.CFReperitore	REPERITORE.CF

3.4.12 Pasqua

La società Dolce Pasqua (DP) S.r.l. produce principalmente, ma non solo, uova di Pasqua e colombe in modo artigianale. Di ciascun prodotto si conosce, oltre al codice identificativo, il prezzo di vendita ed il peso, espresso in grammi. Per la realizzazione dei suoi prodotti, la DP segue delle ricette regionali classiche. Ovviamente, ciascun prodotto è realizzato seguendo una sola ricetta. Ciascuna ricetta è caratterizzata da un nome, dall'indicazione geografica di provenienza e dal nome del maestro pasticciere che l'ha trascritta. In ogni ricetta, si trovano le indicazioni degli ingredienti da impiegare. Per ciascun ingrediente, oltre al nome e ai valori nutritivi ogni 100 grammi, si conosce la quantità da impiegare per la realizzazione di ciascuna ricetta. Le ricette sono poi caratterizzate dalle fasi da seguire nella loro realizzazione. Per ciascuna fase, che può essere comune a più ricette, si conoscono la durata, una breve descrizione e l'elenco degli strumenti che richiede. Si tiene inoltre traccia, per ciascuna fase, del suo numero d'ordine nella sequenza delle fasi che compongono ciascuna ricetta di cui fa parte. I prodotti della DP vengono sottoposti a severissimi controlli di qualità. Di ciascun controllo si conosce il nome, la caratteristica valutata ed il valore di riferimento. Naturalmente, ogni prodotto viene sottoposto a più test, così come lo stesso test viene praticato su più prodotti. Le uova di Pasqua, come da tradizione, contengono una sorpresa. In particolare, ciascun tipo di sorpresa è abbinato ad un solo tipo di uovo, mentre lo stesso tipo di uovo può contenere fino a quattro tipi diversi di sorprese. Di ciascun tipo di sorpresa si conosce l'età minima consigliata ed il target di riferimento: bimbo, bimba o entrambi. Ciascuna tipo di sorpresa è identificato da un codice, unico per ciascuna società esterna produttrice. Di queste società esterne si conosce la partita Iva, ragione sociale, numero di telefono ed indirizzo (via, numero civico, CAP, città e provincia).

Soluzione

La gerarchia di generalizzazione dell'entità PRODOTTO è parziale ed esclusiva, dato che le uova e le colombe sono i principali, e quindi non gli unici, prodotti della DP.



La gerarchia di generalizzazione dell'entità PRODOTTO è stata ristrutturata con un collasso verso l'alto. Si noti che la relazione **Contenere**, legata all'entità UOVOPASQUA con cardinalità $(1, 4)$, viene tradotta come se la cardinalità fosse $(1, n)$, dato che il modello logico non tiene traccia di questi vincoli.

Relazione	**Attributi**
PRODOTTO	ID, Prezzo, Peso, Selettore, IDRicetta
RICETTA	ID, Nome, Provenienza, Pasticciere
INGREDIENTE	ID, Nome, ValoriNutrizionali
FASE	ID, Descrizione, Durata
STRUMENTO	Nome, IDFase
CONTROLLO	ID, Nome, Caratteristica, ValoreRiferimento
TIPOSORPRESA	ID, PIVAProduttore, EtàMinima, Target, IDProdotto
PRODUTTORE	PIVA, RagioneSociale, Telefono, Via, Numero, Cap, Città, Provincia
IMPIEGARE	IDRicetta, IDIngrediente, Quantità
COMPORRE	IDRicetta, IDFase, NumeroOrdine
EFFETTUARE	IDProdotto, IDControlo

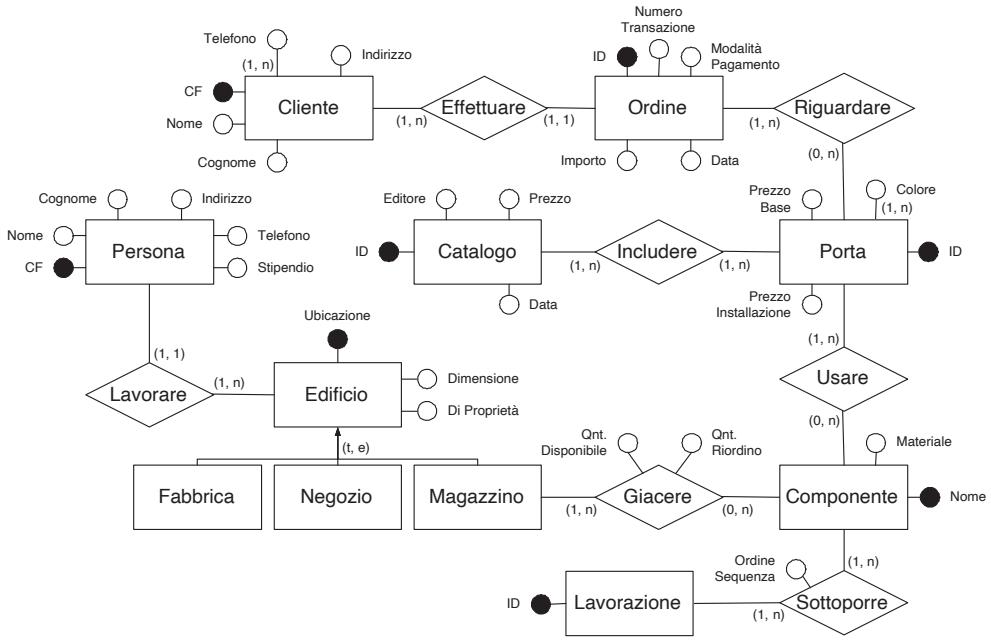
Chiave Esterna	Referenzia
PRODOTTO.IDRicetta	RICETTA.ID
STRUMENTO.IDFase	FASE.ID
TIPOSORPRESA.PIVAProduttore	PRODUTTORE.PIVA
TIPOSORPRESA.IDProdotto	PRODOTTO.ID
IMPIEGARE.IDRicetta	RICETTA.ID
IMPIEGARE.IDIngrediente	INGREDIENTE.ID
COMPORRE.IDRicetta	RICETTA.ID
COMPORRE.IDFase	FASE.ID
EFFETTUARE.IDProdotto	PRODOTTO.ID
EFFETTUARE.IDControlo	CONTROLLO.ID

3.4.13 Porte Sicure

L'azienda Porte Sicure (PS), specializzata nella fabbricazione ed installazione di porte blindate, dispone di un catalogo di prodotti caratterizzato dal nome dell'editore, la data di pubblicazione ed un prezzo. Ciascun catalogo contiene un elenco di porte blindate ognuna delle quali è identificata da un codice, può essere disponibile in più colori ed ha associato un prezzo base ed un prezzo di installazione. L'azienda PS dispone di diversi edifici che si suddividono in magazzini, fabbriche e negozi. Di ogni edificio si conosce l'ubicazione, la dimensione e se è di proprietà dell'azienda oppure se è in affitto. Per ciascun edificio sono anche note le persone (identificate dal codice fiscale e di cui si conosce il nome, cognome, indirizzo, telefono e stipendio) che ci lavorano; una persona può lavorare in un solo edificio. Si deve inoltre tenere traccia delle giacenze di magazzino e pertanto ad ogni magazzino è associata una lista di componenti e per ognuno di essi si conosce la quantità disponibile e la quantità di riordino. Per i componenti le informazioni disponibili sono il nome, il tipo di materiale di cui sono composti, le lavorazioni a cui devono essere sottoposti (associata a questa informazione è importante mantenere traccia anche dell'ordine in cui queste lavorazioni vanno effettuate; ordine che può cambiare da componente a componente) e, infine, le porte blindate di cui fanno parte (si noti che uno stesso componente può essere presente in più magazzini e la quantità disponibile può essere diversa).
Per i clienti dell'azienda si è interessati a gestire informazioni anagrafiche quali nome, cognome, codice fiscale, indirizzo ed i numeri di telefono (ogni cliente ha almeno un numero di telefono). Ogni cliente effettua un ordine per una o più porte blindate. Dell'ordine si conosce la data, la modalità di pagamento e l'importo totale. Si noti che gli ordini fatti dai clienti devono essere raggruppati per transazione, si deve cioè fare in modo che sia possibile sapere quali porte blindate sono state ordinate insieme.

Soluzione

La gerarchia di generalizzazione dell'entità EDIFICIO è totale ed esclusiva, come esplicitato nei requisiti.



La gerarchia di generalizzazione dell'entità EDIFICIO è stata ristrutturata con un collasso verso l'alto.

Relazione	**Attributi**
CATALOGO	ID, Data, Prezzo, Editore
PORTA	ID, PrezzoBase, PrezzoInstallazione
EDIFICIO	Ubicazione, Dimensione, DiProprietà, Selettore
PERSONA	CF, Nome, Cognome, Indirizzo, Telefono, Stipendio, UbicazioneEdificio
COMPONENTE	Nome, Materiale
CLIENTE	CF, Nome, Cognome, Indirizzo
TELEFONOCLIENTE	Telefono, CFCliente
ORDINE	ID, Data, ModalitàPagamento, Importo, NumeroTransazione, CFCliente
INCLUDERE	IDCatalogo, IDPorta
GIACERE	UbicazioneEdificio, IDComponente, QntDisponibile, QntRiordino
USARE	IDComponente, IDPorta
EFFETTUARE	CFCliente, IDOrdine
RIGUARDARE	IDPorta, IDOrdine
AVERE	Colore, IDPorta
SOTTOPORRE	IDComponente, IDLavorazione, OrdineSequenza

Chiave Esterna	Referenzia
ORDINE.CFCliente	CLIENTE.CF
RIGUARDARE.IDPorta	PORTA.ID
RIGUARDARE.IDOrdine	ORDINE.ID
USARE.IDComponente	COMPONENTE.ID
USARE.IDPorta	PORTA.ID
GIACERE.UbicazioneEdificio	EDIFICIO.Ubicazione
GIACERE.IDComponente	COMPONENTE.ID
PERSONA.UbicazioneEdificio	EDIFICIO.Ubicazione
INCLUDERE.IDCatalogo	CATALOGO.ID
INCLUDERE.IDPorta	PORTA.ID
AVERE.IDPorta	PORTA.ID
TELEFONOCLIENTE.CFCliente	CLIENTE.CF
SOTTOPORRE.IDComponente	COMPONENTE.ID

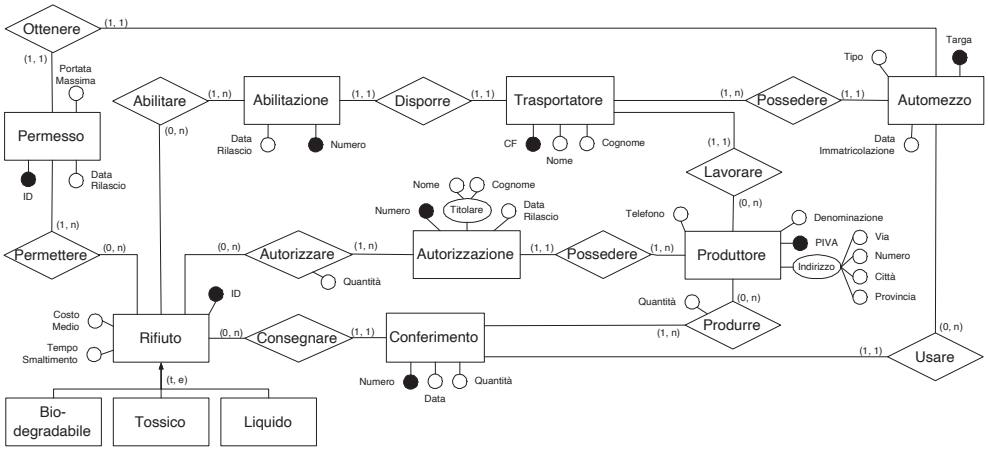
3.4.14 Rifiuti

I rifiuti speciali, gestiti nel depuratore della società Puliamo il Mondo, sono suddivisi in diverse tipologie: rifiuti biodegradabili, tossici e liquidi. Per ogni categoria di rifiuti si conosce il costo medio ed il tempo medio di smaltimento. I rifiuti sono prodotti da produttori autorizzati e convenzionati con il depuratore. Per ogni produttore si vuole tenere traccia della partita Iva, denominazione, indirizzo (via, numero civico, città e provincia), telefono e dei dati relativi alle autorizzazioni alla produzione dei rifiuti speciali da esso possedute e che autorizzano quindi un produttore a produrre uno o più tipi di rifiuto. Tali autorizzazioni sono caratterizzate da numero di autorizzazione alla produzione, (numero univoco su tutto il territorio nazionale), data di rilascio e riferimenti al titolare dell'autorizzazione (nome e cognome del dirigente dell'azienda produttrice a cui l'autorizzazione è stata concessa). Per ogni tipo di rifiuto coinvolto in una data autorizzazione viene specificata una quantità massima annuale di produzione. Si noti che un produttore può avere una o più autorizzazioni alla produzione. Il trasporto dei rifiuti avviene tramite autobotti o altri automezzi speciali condotti da trasportatori autorizzati (o loro dipendenti). Per i trasportatori che effettuano i conferimenti si conosce il CF, nome, cognome e produttore presso cui lavora. Ogni trasportatore deve poi disporre di una abilitazione al trasporto per uno o più tipi di rifiuti. Per ogni abilitazione al trasporto si conosce il numero di abilitazione al trasporto, (numero univoco su tutto il territorio nazionale), data di rilascio ed elenco delle tipologie dei rifiuti speciali per i quali si abilita al trasporto. Ogni trasportatore possiede inoltre uno o più automezzi (identificati dalla targa e a cui sono associate ulteriori informazioni quali tipo di automezzo e data immatricolazione) ciascuno dei quali deve a sua volta disporre di un permesso al trasporto di uno o più tipi di rifiuto. Per ogni permesso al trasporto sono previsti quindi i seguenti dati: data di rilascio, portata massima dell'automezzo, elenco delle tipologie dei rifiuti speciali per i quali si permette il trasporto.

Si vogliono infine rilevare tutte le informazioni relative ai singoli conferimenti. Ogni conferimento viene caratterizzato da un numero progressivo, da una data, da un tipo di rifiuto conferito (unico per ciascun conferimento), dalla quantità del rifiuto (espressa in Kg), dal trasportatore che ha effettuato il trasporto e dal relativo automezzo e infine dal o dai produttori che hanno prodotto il carico. Nel caso di carico prodotto da più produttori, si dovranno conoscere i pesi relativi al prodotto di ogni singolo produttore.

Soluzione

La gerarchia di generalizzazione dell'entità RIFIUTI è totale ed esclusiva, come esplicitato nei requisiti.



La gerarchia di generalizzazione dell'entità RIFIUTI è stata ristrutturata con un collasso verso l'alto.
Le entità TRASPORTATORE e ABILITAZIONE sono state accorpate in un'unica tabella, dato che legate da una relazione 1 : 1. Lo stesso è avvenuto per le entità AUTOMEZZO e PERMESSO.

Relazione	**Attributi**
RIFIUTO	ID, CostoMedio, TempoSmaltimento, Selettore
PRODUTTORE	PIVA, Denominazione, Telefono, Via, Numero, Città, Provincia
AUTORIZZAZIONE	Numero, DataRilascio, NomeTitolare, CognomeTitolare, PIVAProduttore
TRASPORTATORE	CF, Nome, Cognome, PIVAProduttore, NumeroAbilitazione, DataRilascioAbilitazione
AUTOMEZZO	Targa, Tipo, DataImmatricolazione, CFTrasportatore, IDPermesso, DataRilascioPermesso, PortataMassima
CONFERIMENTO	Numero, Data, Quantità, IDRifiuto, TargaAutomezzo
AUTORIZZARE	IDRifiuto, NumeroAutorizzazione, Quantità
ABILITARE	IDRifiuto, CFTrasportatore
PERMETTERE	IDRifiuto, TargaAutomezzo
PRODURRE	NumeroConferimento, PIVAProduttore, Quantità

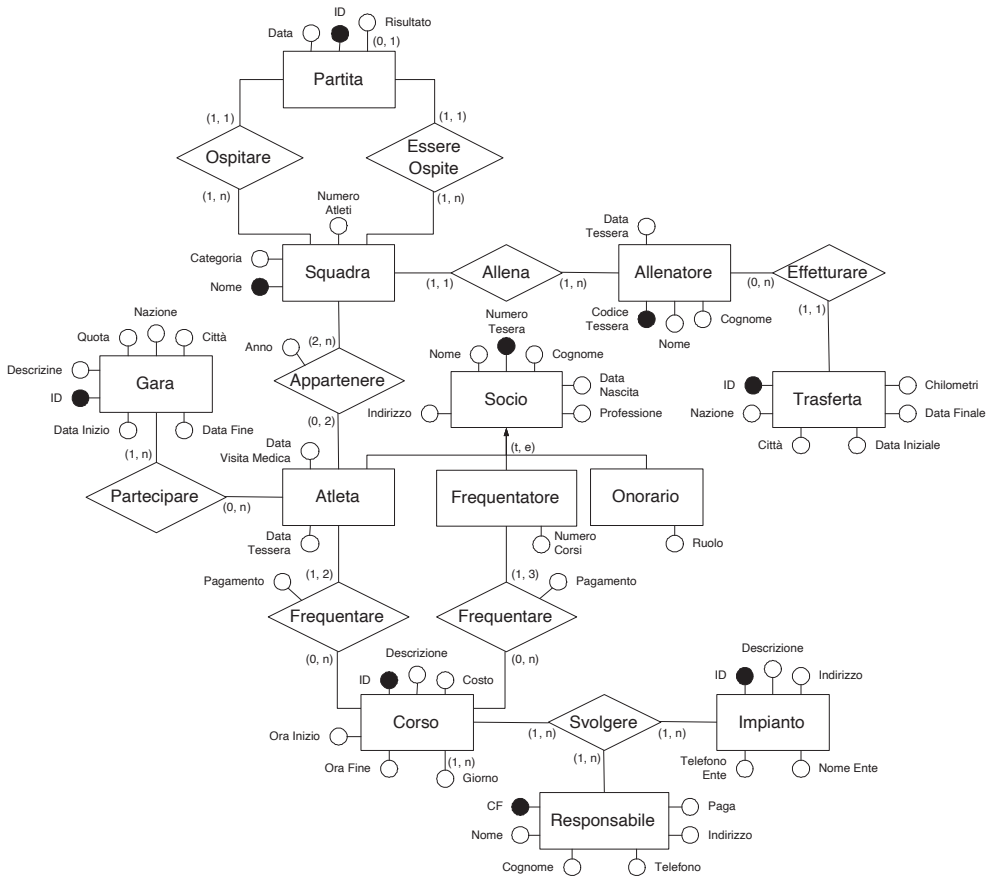
Chiave Esterna	Referenzia
AUTORIZZAZIONE.PIVAProduttore	PRODUTTORE.PIVA
TRASPORTATORE.PIVAProduttore	PRODUTTORE.PIVA
AUTOMEZZO.CFTrasportatore	TRASPORTATORE.CF
CONFERIMENTO.IDRifiuto	RIFIUTO.ID
CONFERIMENTO.TargaAutomezzo	AUTOMEZZO.Targa
AUTORIZZARE.IDRifiuto	RIFIUTO.ID
AUTORIZZARE.NumeroAutorizzazione	AUTORIZZAZIONE.Numero
ABILITARE.IDRifiuto	RIFIUTO.ID
ABILITARE.CFTrasportatore	TRASPORTATORE.CF
PERMETTERE.IDRifiuto	RIFIUTO.ID
PERMETTERE.TargaAutomezzo	AUTOMEZZO.Targa
PRODURRE.NumeroConferimento	CONFERIMENTO.Numero
PRODURRE.PIVAProduttore	PRODUTTORE.PIVA

3.4.15 Associazione Sportiva

I soci che si iscrivono alla società sportiva Tutti in Forma sono caratterizzati da: il numero della tessera di iscrizione, il nome, il cognome, l'indirizzo, la data di nascita e la professione. I soci si suddividono in atleti, frequentatori ed onorari. Per gli atleti è importante tener traccia della data dell'ultima visita medica e della data della tessera federativa. Per i frequentatori è necessario tener traccia del numero di corsi frequentati negli ultimi cinque anni. I soci onorari sono caratterizzati dal ruolo che svolgono all'interno della società. Gli atleti ed i soci frequentanti possono frequentare i corsi (al massimo tre per i frequentanti, due per gli atleti). È importante mantenere l'informazione relativa al pagamento relativo di ogni corso frequentato. I corsi sono caratterizzati dal codice corso, una breve descrizione, il costo, i giorni della settimana in cui sono programmati, gli impianti che li ospitano e le relative ore di inizio e di fine in cui sono tenuti. Gli impianti a disposizione della società sono rappresentati da: codice, descrizione, indirizzo, telefono e nome dell'ente che li ha in gestione. Ciascun corso (svolto in un determinato impianto) ha un responsabile esterno. Per quest'ultimo si conosce il codice fiscale, il nome, il cognome, il telefono, l'indirizzo e la paga oraria. Gli atleti possono svolgere uno sport individuale o essere inseriti in una o più squadre, ricordando per ciascuna l'anno di ingresso. In entrambi i casi devono essere iscritti ai relativi corsi. Ogni squadra è caratterizzata da un nome, categoria, numero di atleti titolari, allenatore e partite giocate (con i risultati) e non ancora giocate; per le partite si conosce la data e la squadra avversaria. Per gli allenatori sono noti la data della tessera federativa, il relativo codice, il nome e il cognome. Gli allenatori possono effettuare delle trasferte che sono caratterizzate da: codice, nazione, città, data iniziale, data finale e numero chilometri percorsi. Per gli atleti che svolgono sport individuali si conoscono le gare a cui partecipano. Le gare hanno associato una descrizione, quota di partecipazione per l'atleta, città, nazione, data di inizio e data di fine.

Soluzione

La gerarchia di generalizzazione dell'entità SOCIO è totale ed esclusiva, dato che non esistono altre categorie di soci e quelle indicate sono ben distinte fra loro.
La relazione **Frequentare** è rappresentata due volte nel modello ER, sia per ATLETA sia per FREQUENTATORE, al fine di esplicitare i differenti vincoli sulla cardinalità.



La gerarchia di generalizzazione dell'entità SOCIO è stata ristrutturata con un collasso verso il basso.
Si noti che l'entità PARTITA viene utilizzata sia per rappresentare le partite già giocate sia le partite non ancora giocate, grazie all'attributo opzionale *Risultato*, che per le partite ancora da giocare assume il valore *NULL*.
In fase di traduzione al modello logico, le cardinalità massime pari a 2 e 3 vengono equiparate al valore n, dato che il modello logico non tiene traccia dei vincoli sulle cardinalità.

Relazione	Attributi
ATLETA	NumTessera, Nome, Cognome, Indirizzo, DataNascita, Professione, DataVisitaMedica, DataTessera
FREQUENTATORE	NumTessera, Nome, Cognome, Indirizzo, DataNascita, Professione, NumeroCorsi
ONORARIO	NumTessera, Nome, Cognome, Indirizzo, DataNascita, Professione, Ruolo
CORSO	ID, Costo, Descrizione, OraInizio, OraFine
GIORNICORSO	Giorno, IDCorso
IMPIANTO	ID, Descrizione, Indirizzo, TelefonoEnte, NomeEnte
RESPONSABILE	CF, Nome, Cognome, Indirizzo, Telefono, Paga
SQUADRA	Nome, Categoria, NumeroAtleti, CodiceTesseraAllenatore
ALLENATORE	CodiceTessera, Datatessera, Nome, Cognome
TRASFERTA	ID, Nazione, Città, DataInizio, DataFine, Chilometri, CodiceTesseraAllenatore
GARA	ID, Quota, Descrizione, Nazione, Città, DataInizio, DataFine
FREQATLETA	NumTAtleta, IDCorso, Pagamento
FREQFREQUENTATORE	NumTFreq, IDCorso, Pagamento
SVOLGERE	IDCorso, IDImpianto, CFResponsabile
APPARTENERE	NumTesseraAtleta, NomeSquadra, Anno
PARTITA	IDPartita, NomeSquadraOspitante, NomeSquadraOspitata, Data, Risultato
PARTECIPA	NumTesseraAtleta, IDGara

Chiave Esterna	Referenzia
FREQATLETA.NumTAtleta	ATLETA.NumTessera
FREQATLETA.IDCorso	CORSO.ID
FREQFREQUENTATORE.NumTFreq	FREQUENTATORE.NumTessera
FREQFREQUENTATORE.IDCorso	CORSO.ID
SVOLGERE.IDCorso	CORSO.ID
SVOLGERE.IDImpianto	IMPIANTO.ID
SVOLGERE.CFResponsabile	RESPONSABILE.CF
APPARTENERE.NumTesseraAtleta	ATLETA.NumTessera
APPARTENERE.NomeSquadra	SQUADRA.Nome
PARTITA.NomeSquadraOspitante	SQUADRA.Nome
PARTITA.NomeSquadraOspitata	SQUADRA.Nome
SQUADRA.CodiceTesseraAllenatore	ALLENATORE.CodiceTessera
TRASFERTA.CodiceTesseraAllenatore	ALLENATORE.CodiceTessera
PARTECIPA.NumTesseraAtleta	ATLETA.NumTessera
PARTECIPA.IDGara	GARA.ID
GIORNICORSO.IDCorso	CORSO.ID

3.4.16 Sposi

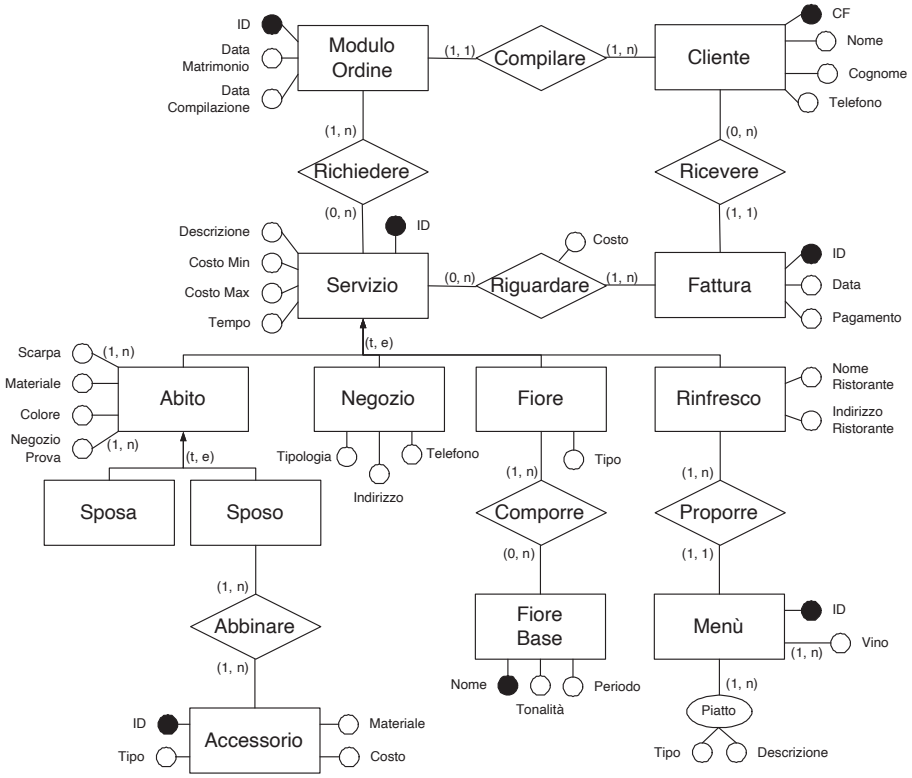
La società SposiFelici (SF), specializzata nella organizzazione di matrimoni, offre ai propri clienti una vasta gamma di servizi per ognuno dei quali si conosce una breve descrizione, il costo minimo e massimo di organizzazione del servizio e una indicazione di quanto tempo prima del matrimonio il servizio stesso deve essere ordinato. I servizi offerti possono essere suddivisi in quattro tipologie: abiti, negozi per lista nozze, fiori e rinfresco. La sezione abiti è a sua volta suddivisa in abiti per la sposa ed abiti per lo sposo e per ogni abito si specifica il tipo di materiale usato, il colore, i vari modelli di scarpe che si possono abbinare, l'elenco dei negozi dove recarsi per le prove dell'abito e, solo nel caso degli abiti da sposo, l'elenco

degli accessori abbinabili; per ogni accessorio si conosce il tipo (es, cravatta, gemelli), il materiale ed il costo. Per quanto concerne la sezione negozi per lista nozze viene fornito l'indirizzo del negozio, il numero di telefono e la tipologia del negozio stesso. La sezione fiori include il tipo di composizione (es., fiori per la chiesa, per il municipio, bouquet) e la lista di fiori che fanno parte della composizione. I fiori possono essere disponibili in diverse tonalità e solo in determinati periodi dell'anno. Si noti che uno stesso fiore può far parte di diverse composizioni. Per i rinfreschi vengono proposte diverse combinazioni per ognuna delle quali si conosce il menù (un menù si compone di una descrizione dei piatti forniti, con l'indicazione se si tratta di un antipasto, primo, secondo o dolce, e di una lista di vini), il nome e indirizzo del ristorante.

I clienti della società SF possono prenotare uno o più servizi compilando un modulo ordine sul quale sono riportate le seguenti informazioni: nome, cognome e recapito telefonico del cliente, l'elenco dei servizi richiesti, la data del matrimonio e per finire la data di compilazione del modulo. Si noti che uno stesso cliente può ordinare più servizi in momenti diversi. Dopo che il matrimonio ha avuto luogo, il cliente riceve una fattura la quale riporta la data di compilazione, l'elenco dei servizi effettivamente erogati con il costo finale per ciascuno di essi e la modalità di pagamento.

Soluzione

La gerarchia di generalizzazione dell'entità SERVIZIO è totale ed esclusiva poiché contempla tutti i tipi di servizio, che sono fra loro ben distinti. La gerarchia di generalizzazione dell'entità ABITO è per la stessa ragione totale ed esclusiva.
L'attributo multiplo e composto *Piatto* poteva anche essere rappresentato attraverso un'entità legata all'entità MENÙ attraverso una relazione $1 : n$.



La gerarchia di generalizzazione dell'entità Servizio è stata ristrutturata preservando tutte le entità. La gerarchia di generalizzazione dell'entità Abito è stata ristrutturata con un collasso verso l'alto.

Relazione	Attributi
SERVIZIO	ID, Descrizione, CostoMin, CostoMax, Tempo
ABITO	ID, Materiale, Colore, Selettore
NEGOZIO	ID, Telefono, Indirizzo, Tipologia
FIORE	ID, Tipo
RINFRESCO	ID, NomeRistorante, IndirizzoRistorante
SCARPA	IDScarpa, IDAbito
NEGOZIOPROVA	IDNegozio, IDAbito
ACCESSORIO	ID, Tipo, Materiale, Costo
FIOREBASE	Nome, Tonalità, Periodo
MENU	ID, IDRinfresco
PIATTO	ID, Tipo, Descrizione
APPARTENERE	IDPiatto, IDMenu
VINO	IDVino, IDMenu
CLIENTE	CF, Nome, Cognome, Telefono
MODULOORDINE	ID, DataMatrimonio, DataCompilazione, CFCliente
FATTURA	ID, Data, Pagamento, CFCliente
ABBINARE	IDAbito, IDAccessorio
COMPORRE	NomeFioreBase, IDFiore
RIGUARDARE	IDFattura, IDServizio, Costo
RICHIEDERE	IDServizio, IDModuloOrdine

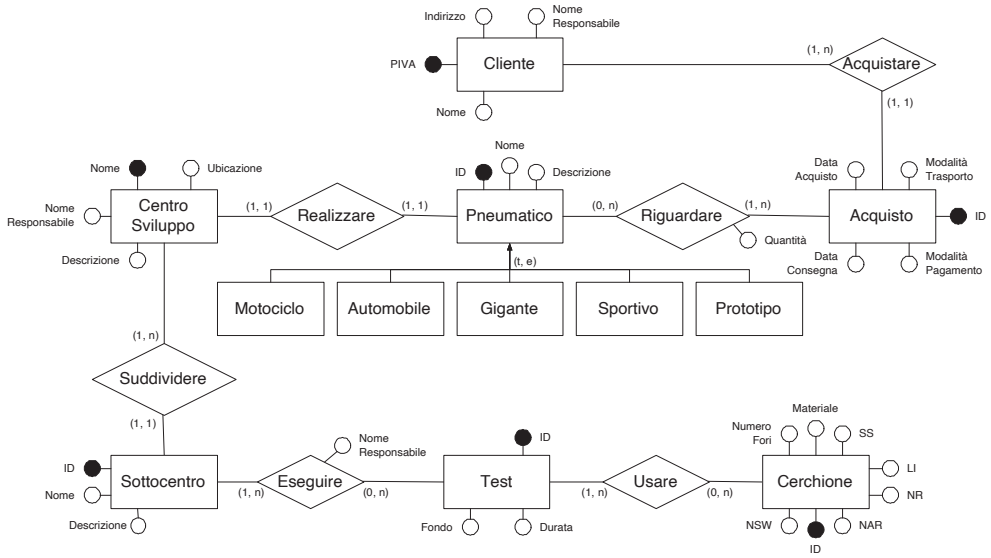
Chiave Esterna	Referenzia
ABITO.ID	SERVIZIO.ID
NEGOZIO.ID	SERVIZIO.ID
FIORE.ID	SERVIZIO.ID
RINFRESCO.ID	SERVIZIO.ID
SCARPA.IDAbito	ABITO.ID
NEGOZIOPROVA.IDAbito	ABITO.ID
MENU.IDRinfresco	RINFRESCO.ID
APPARTENERE.IDPiatto	PIATTO.ID
APPARTENERE.IDMenu	MENU.ID
VINO.IDMenu	MENU.ID
MODULOORDINE.CFCliente	CLIENTE.CF
FATTURA.CFCliente	CLIENTE.CF
ABBINARE.IDAbito	ABITO.ID
ABBINARE.IDAccessorio	ACCESSORIO.ID
COMPORRE.NomeFioreBase	FIOREBASE.Nome
COMPORRE.IDFiore	FIORE.ID
RIGUARDARE.IDFattura	FATTURA.ID
RIGUARDARE.IDServizio	SERVIZIO.ID
RICHIEDERE.IDServizio	SERVIZIO.ID
RICHIEDERE.IDModuloOrdine	MODULOORDINE.ID

3.4.17 Produzione Pneumatici

L'azienda ViaggiaSicuro (VS), specializzata nella produzione di pneumatici, ha diverse linee di sviluppo pneumatici adatti per vari settori: motociclo, automobile, gigante (per veicoli che superano le 3,5 tonnellate), impieghi sportivi e prototipi. Ogni linea di sviluppo è caratterizzata da un codice, il nome della linea e una descrizione. Ogni linea viene realizzata nell'ambito di un ben specifico centro di sviluppo per cui sono noti il nome del centro, l'ubicazione, il nome del responsabile del centro stesso e una descrizione. I centri di sviluppo si suddividono in uno o più sotto-centri ognuno dei quali è responsabile dell'esecuzione di diversi tipi di test a cui sono sottoposti i pneumatici. I sotto-centri hanno un codice, un nome, una descrizione. I test eseguiti sui pneumatici si differenziano per il tipo, il modello (o modelli) di cerchione usato, la durata e il tipo di fondo stradale su cui deve essere eseguito il test. I cerchioni si differenziano per il nominal section width (nsw), il nominal aspect ratio (nar), il nominal ring (nr), il load index (li), il speed symbol (ss), il numero di fori e per il materiale di cui sono composti. I clienti di VS sono le case automobilistiche per cui sono note le informazioni relative al nome, PIVA, l'indirizzo e il nome del responsabile della casa automobilistica a cui vanno intestate le fatture. Un singolo acquisto è caratterizzato dalla data di acquisto, la data prevista di consegna, la modalità di pagamento, la modalità di trasporto e può riguardare uno o più linee di pneumatici: per ogni linea deve essere indicato il numero di pneumatici ordinati.

Soluzione

La gerarchia di generalizzazione dell'entità PNEUMATICO è totale ed esclusiva, come esplicitato nei requisiti.



La gerarchia di generalizzazione dell'entità PNEUMATICO è stata ristrutturata con un collasso verso l'alto.

Relazione	**Attributi**
CLIENTE	PIVA, Nome, Indirizzo, NomeResponsabile
ACQUISTO	ID, ModalitàTrasporto, ModalitàPagamento, DataConsegna, DataAcquisto, PIVACliente
PNEUMATICO	ID, Nome, Descrizione, Selettore, NomeCentro
CENTROSVILUPPO	Nome, Ubicazione, Descrizione, NomeResponsabile
SOTTOCENTRO	ID, Nome, Descrizione, NomeCentro
TEST	ID, Fondo, Durata
CERCHIONE	ID, Nsw, Nar, Nr, Li, Ss, Materiale, NumeroFori
RIGUARDARE	IDPneumatico, IDAcquisto, Quantità
ESEGUIRE	IDTest, IDSottoCentro, NomeResponsabile
USARE	IDTest, IDCerchione

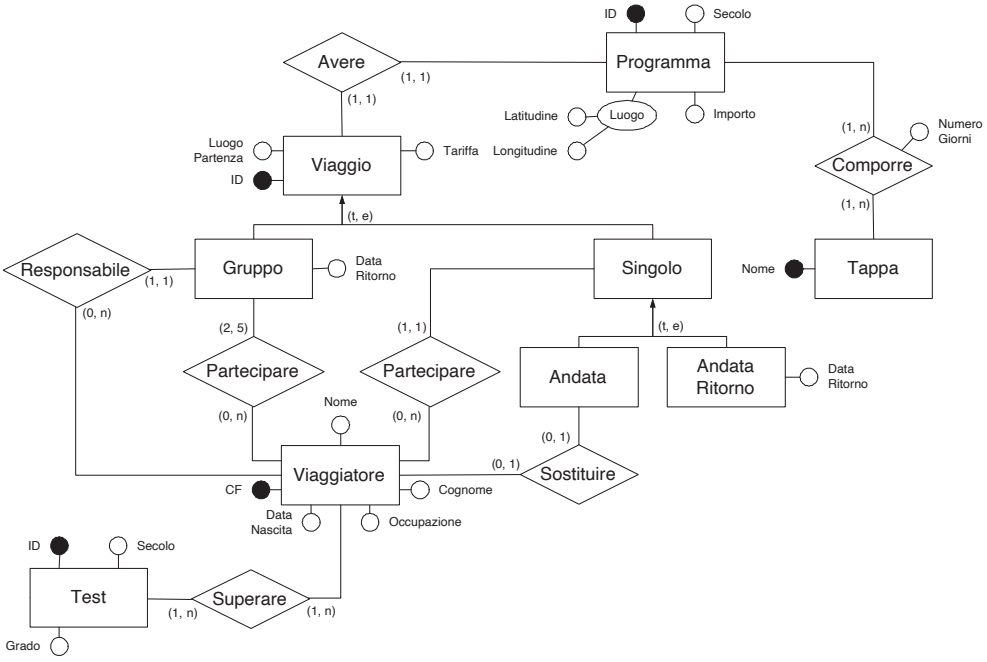
Chiave Esterna	**Referenzia**
ACQUISTO.PIVACliente	CLIENTE.PIVA
PNEUMATICO.NomeCentro	CENTRO.Nome
SOTTOCENTRO.NomeCentro	CENTRO.Nome
RIGUARDARE.IDPneumatico	PNEUMATICO.ID
RIGUARDARE.IDAcquisto	ACQUISTO.ID
ESEGUIRE.IDTest	TEST.ID
ESEGUIRE.IDSottoCentro	SOTTOCCENTRO.ID
USARE.IDTest	TEST.ID
USARE.IDCerchione	CERCHIONE.ID

3.4.18 Viaggio nel Tempo

I viaggi nel tempo della società Torno Ieri, caratterizzati da un codice, un luogo di partenza ed una tariffa, si suddividono in viaggi per singoli o gruppi (di non più di cinque persone). Di ciascun viaggio si deve mantenere traccia dei partecipanti; se il viaggio è di gruppo, una persona del gruppo funge da responsabile. I viaggiatori hanno associato il codice fiscale, nome, cognome, data di nascita e occupazione. Ogni viaggiatore deve mostrare di aver superato uno o più test di attitudine. I test hanno diversi gradi di difficoltà ed autorizzano viaggi non anteriori ad un dato secolo (es., il test base T1 permette di viaggiare solo nel XX secolo, ma già il test T2 permette viaggi sino al XV secolo). I viaggi per gruppi sono tutti viaggi di andata e ritorno mentre i viaggi per singoli possono essere anche di sola andata; solo per i viaggi di andata e ritorno si deve tenere traccia della data prevista per il rientro. Viaggi di sola andata possono essere interrotti quando è possibile trovare qualcuno con il biglietto di ritorno. Per i viaggi che vengono interrotti si deve mantenere il riferimento tra il viaggio ed il viaggiatore che possiede il biglietto di ritorno. Si noti che un viaggiatore può sostituire un altro viaggiatore in un viaggio di sola andata una ed una sola volta.
I programmi di viaggio (uno per ogni viaggio) specificano il secolo prescelto per il corrispondente viaggio, il luogo espresso in coordinate spaziali (latitudine, longitudine) e l'importo. I programmi di viaggio sono composti da una o più tappe intermedie di cui si conosce il nome ed il numero di giorni che si trascorreranno nel luogo che corrisponde alla tappa stessa. Si osservi che una tappa può far parte di più programmi di viaggio e che l'informazione relativa al numero di giorni varia a seconda del programma di viaggio di cui la tappa fa parte.

Soluzione

Le gerarchie di generalizzazione delle entità VIAGGIO e SINGOLO sono totali ed esclusive, come esplicitato nei requisiti.
La relazione **Partecipare** è rappresentata due volte nel modello ER, sia per GRUPPO sia per SINGOLO, al fine di esplicitare i differenti vincoli sulla cardinalità.



Le gerarchie di generalizzazione delle entità VIAGGIO e SINGOLO sono state ristrutturata con un collasso verso l'alto. Proprio per questo nell'entità VIAGGIO è stato aggiunto un *selettore* che può assumere i seguenti valori: *Gruppo*, *SingoloAndata*, *SingoloAndataRitorno*.
Le entità VIAGGIO e PROGRAMMA sono state accorpate in un'unica tabella, dato che legate da una relazione 1 : 1.

Relazione	**Attributi**
VIAGGIATORE	CF, Nome, Cognome, DataNascita, Occupazione
TEST	ID, Grado, Secolo
VIAGGIO	ID, LuogoPartenza, Tariffa, Selettore, DataRitorno, Secolo, Latitudine, Longitudine, Importo, CFResponsabile, CFSostituto
COMPORRE	IDViaggio, NomeTappa, Giorni
PARTECIPARE	IDViaggio, IDViaggiatore
SUPERARE	CFViaggiatore, IDTest

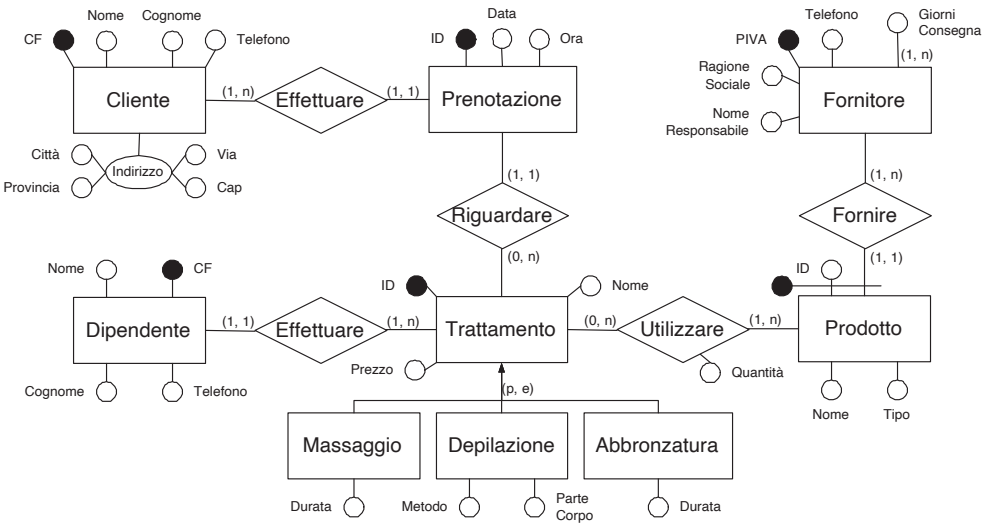
Chiave Esterna	Referenzia
SUPERARE.IDViaggiatore	VIAGGIATORE.ID
SUPERARE.IDTest	TEST.ID
COMPORRE.IDViaggio	VIAGGIO.ID
PARTECIPARE.IDViaggio	VIAGGIO.ID
PARTECIPARE.CFViaggiatore	VIAGGIATORE.CF
VIAGGIO.CFResponsabile	VIAGGIATORE.CF
VIAGGIO.CFSostituto	VIAGGIATORE.CF

3.4.19 Estetico

Il centro estetico TuttiBelli (TB) offre una serie di trattamenti di bellezza, caratterizzati da un codice identificativo, un nome e un prezzo. Fra i trattamenti offerti, si possono distinguere tre principali categorie: i massaggi, la depilazione e l'abbronzatura. In particolare, per i massaggi si conosce la durata (espressa in ore); per la depilazione si tiene traccia del metodo utilizzato e della parte del corpo interessata; infine, per l'abbronzatura si mantiene la durata (espressa in minuti). I clienti del centro estetico, di cui si conoscono codice fiscale, nome, cognome, numero di telefono e indirizzo (composto da via, CAP, città e provincia), possono prenotare il trattamento, specificando la data e l'ora dell'appuntamento. Ciascun cliente può prenotare più trattamenti. Il centro estetico ha un insieme di dipendenti, ciascuno specializzato in un trattamento. Dei dipendenti si vuole tenere traccia del codice fiscale, nome, cognome e numero di telefono. Uno stesso trattamento può essere effettuato da più dipendenti. Ciascun trattamento può anche utilizzare dei prodotti specifici per aumentarne l'efficacia. Di questi prodotti si conoscono il codice (unico rispetto al fornitore), il nome e il tipo, oltre che la quantità impiegata per ciascun trattamento in cui è utilizzato. Ogni fornitore, che può fornire più prodotti, è caratterizzato da una ragione sociale, dal nome del responsabile di zona, dal numero di telefono dello stesso e dai giorni settimanali di consegna.

Soluzione

La gerarchia di generalizzazione dell'entità TRATTAMENTO è parziale ed esclusiva, dato che il centro potrebbe offrire anche altri trattamenti.



La gerarchia di generalizzazione dell'entità TRATTAMENTO è stata ristrutturata con un collasso verso l'alto.
Si noti che l'entità PRENOTAZIONE è necessaria per legare le entità TRATTAMENTO e CLIENTE. Mettendo infatti una relazione **Prenotare** con attributi *Data* e *Ora* fra le due entità, ad un cliente non sarebbe consentito di prenotare più di una volta lo stesso trattamento.

Relazione	**Attributi**
TRATTAMENTO	ID, Nome, Prezzo, Durata, MetodoDepilazione, ParteCorpo, Selettore
DIPENDENTE	CF, Nome, Cognome, Telefono, IDTrattamento
CLIENTE	CF, Nome, Cognome, Telefono, Via, Cap, Città, Provincia
PRENOTAZIONE	ID, Data, Ora, CFCliente, IDTrattamento
PRODOTTO	ID, PIVAFornitore, Nome, Tipo
FORNITORE	PIVA, RagioneSociale, NomeResponsabile, Telefono
GIORNICONSEGNA	PIVAFornitore, Giorno
UTILIZZARE	IDTrattamento, IDProdotto, PIVAFornitore, Quantità

Chiave Esterna	**Referenzia**
DIPENDENTE.IDTrattamento	TRATTAMENTO.ID
PRENOTAZIONE.CFCliente	CLIENTE.CF
PRENOTAZIONE.IDTrattamento	TRATTAMENTO.ID
PRODOTTO.PIVAFornitore	FORNITORE.PIVA
GIORNICONSEGNA.PIVAFornitore	FORNITORE.PIVA
UTILIZZARE.IDTrattamento	TRATTAMENTO.ID
UTILIZZARE.{IDProdotto, PIVAFornitore}	PRODOTTO.{ID, PIVAFornitore}

Capitolo 4

Gestione delle Transazioni

Questo Capitolo è dedicato ad argomenti avanzati di basi di dati. In particolare, ci si concentra sulla risoluzione di guasti e sui vari metodi per garantire la serializzabilità degli schedule.

4.1 Ripresa a Caldo

La ripresa a caldo ha lo scopo di ripristinare la base di dati nel caso di guasti di sistema, cioè quando il contenuto della memoria principale viene perso, ma non quello della memoria secondaria.
Per effettuare la ripresa a caldo si considera il log compilato dal DBMS nel suo normale funzionamento fino al guasto.
Il log può contenere i seguenti tipi di record:

- $B(T)$: begin transaction
- $C(T)$: commit transaction
- $A(T)$: abort transaction (fa il rollback della transazione)
- $CK(T_1, \ldots, T_n)$: checkpoint con transazioni attive $T_1, \ldots, T_n$
- $I(T, O, A)$: inserimento dell'oggetto O con valore A da parte di T
- $D(T, O, B)$: eliminazione dell'oggetto O, che aveva valore B, da parte di T
- $U(T, O, B, A)$: modifica del valore all'oggetto O, che passa da B ad A, da parte di T
- $DUMP$: backup su disco della base di dati, chiude quindi tutte le operazioni pendenti.

La procedura per eseguire una ripresa a caldo è la seguente:

1. determinare l'insieme delle transazioni da disfare (UNDO) e da rifare (REDO)
   (a) percorrere il log all'indietro fino al record di checkpoint più recente:
   UNDO = insieme delle transazioni nel checkpoint
   REDO = insieme vuoto
   (b) percorrere il log in avanti (dal checkpoint) e aggiornare gli insiemi UNDO e REDO:
   per ogni record di begin transaction $B(T)$, aggiungere T a UNDO
   per ogni record di commit transaction $C(T)$, spostare T da UNDO a REDO
2. ripristinare la situazione a prima del guasto
   (a) percorrere all'indietro il log fino alla più vecchia operazione di una transazione da disfare o da rifare:
   per ciascuna operazione di transazione nell'UNDO, disfare l'operazione
   (b) dal punto in cui si è arrivati, percorrere in avanti il log:
   per ciascuna operazione delle transazioni nel REDO, rifare l'operazione

Si noti che le azioni di UNDO e REDO godono della *proprietà di idempotenza*, cioè anche se ripetute sulla stessa transazione, non cambia il risultato.

4.1.1 Esercizio Svolto e Commentato

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T1), I(T1,O1,A1), B(T2), I(T2,O2,A2), U(T2,O3,B3,A3), B(T3),
D(T3,O4,B4), B(T4), I(T4,O5,A5), A(T1), B(T5), I(T5,O6,A6),
CK(...), C(T2), U(T5,O7,B7,A7), D(T5,O8,B8), A(T5),
U(T3,O9,B9,A9), C(T3), GUASTO

Si richiede di:

1. scrivere, in corrispondenza di ogni record di CheckPoint, le transazioni attive;
2. illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

1. Come prima cosa, si individuano le transazioni attive al CheckPoint, cioè quelle transazioni già iniziate e per cui non è ancora stato eseguito un commit o un abort.
   In questo caso, le transazioni attive al CheckPoint sono T2, T3, T4 e T5.
   Il log da considerare per la ripresa a caldo è quindi il seguente:

B(T1), I(T1,O1,A1), B(T2), I(T2,O2,A2), U(T2,O3,B3,A3), B(T3), D(T3,O4,B4), B(T4), I(T4,O5,A5), A(T1), B(T5), I(T5,O6,A6), CK(T2,T3,T4,T5), C(T2), U(T5,O7,B7,A7), D(T5,O8,B8), A(T5), U(T3,O9,B9,A9), C(T3), GUASTO

2. Si determinano poi gli insiemi di UNDO e REDO

RECORD	UNDO	REDO
CK(T2,T3,T4,T5)	T2, T3, T4, T5	
C(T2)	T3, T4, T5	T2
C(T3)	T4, T5	T2, T3

Si disfano le operazioni delle transazioni nell'UNDO {T4, T5}
Si noti che si usa sempre e solo il valore B nel record.

RECORD	AZIONE
D(T5,O8,B8)	`insert` O8:=B8
U(T5,O7,B7,A7)	O7:=B7
I(T5,O6,A6)	`delete` O6
I(T4,O5,A5)	`delete` O5

Si rifanno le operazioni delle transazioni nel REDO {T2, T3}
Si noti che si usa sempre e solo il valore A nel record.

RECORD	AZIONE
I(T2,O2,A2)	`insert` O2:=A2
U(T2,O3,B3,A3)	O3:=A3
D(T3,O4,B4)	`delete` O4
U(T3,O9,B9,A9)	O9:=A9

4.2 Serializzabilità di Schedule

Uno schedule è lista ordinata di azioni (ad esempio: lettura, scrittura, modifica e così via) compiute da una o più transazioni. Uno schedule si dice *seriale* se, per ogni transazione t_i, tutte le sue operazioni sono eseguite consecutivamente.
Uno schedule si dice *serializzabile* se è uno schedule non seriale che produce lo stesso risultato di uno schedule seriale. Esistono varie nozioni di serializzabilità; nel seguito si tratteranno gli schedule *view-serializzabili*, *conflict-serializzabili* e gli schedule generati da scheduler *two Phase Locking*.

Due schedule si dicono *view-equivalenti* se hanno le stesse *relazioni legge-da* e le stesse *scritture finali*.

Una operazione $r_i(x)$ legge da $w_j(x)$ (con $i \neq j$) se la lettura segue la scrittura e non c'è nessuna altra scrittura sulla stessa risorsa x fra le due operazioni. L'operazione $w_i(x)$ è una scrittura finale su x se nello schedule non ci sono scritture successive per la stessa risorsa. Uno schedule si dice *view-serializzabile* (*VSR*) se è view-equivalente ad uno schedule seriale.

Due schedule si dicono *conflict-equivalenti* se contengono le stesse operazioni e se le coppie di operazioni in conflitto compaiono nello stesso ordine.
Due operazioni sono in conflitto fra loro se:

- fanno parte di due diverse transazioni;
- riguardano la stessa risorsa;
- almeno una delle due è una scrittura.

Uno schedule si dice *conflict-serializzabile* (*CSR*) se è conflict-equivalente ad uno schedule seriale. Per valutare questa proprietà si ricorre al grafo dei conflitti. Tale grafo ha un nodo per ciascuna transazione dello schedule ed un arco che connette t_i con t_j se e solo se nello schedule esiste una coppia di operazioni o_i (in t_i) e o_j (in t_j) fra loro in conflitto e o_i precede o_j nello schedule. Uno schedule è conflict-serializzabile se il suo grafo dei conflitti è aciclico.

Si dice *lock* una variabile che descrive lo stato di una risorsa rispetto a richieste di utilizzo che la riguardano. Se si ricorre al locking per la gestione della concorrenza, per accedere una risorsa una transazione deve chiederne il lock e, finché non le viene concesso, non può operarvi. La transazione rilascia il lock (unlock) quando la risorsa non le serve più.
Uno scheduler che usa il *locking a due fasi* impone che una transazione che ha rilasciato un lock, non può più acquisirne altri. Gli schedule generati da uno scheduler *2PL* (two Phase Locking) sono serializzabili.
Esistono due versioni del *2PL* a secondo del tipo di lock che viene utilizzato:

- a *2 stati*:
  in questa versione il *lock* può avere solo due valori (*locked* e *unlocked* e quindi quando una risorsa è bloccata da una transazione non risulta più accessibile da nessun'altra);
- a *3 stati*:
  in questa versione il *lock* può avere tre valori (*read locked*, *write locked* e *unlocked*). Più transazioni possono condividere la stessa risorsa se tutti i lock su di essa sono in lettura; ma una risorsa bloccata con un lock in scrittura non può essere acceduta da nessun'altra transazione fino a quando la stessa non viene rilasciata.

Per vedere se lo schedule può essere stato generato da uno scheduler basato su *2PL*, assumiamo che:

- una transazione richiede il lock subito prima dell'operazione relativa;
- se una transazione t_i ha bisogno del lock su una risorsa r ora in possesso di un'altra transazione t_j e t_j non ha più bisogno di accedere ad r, t_j acquisisce (se possibile) i lock per tutte le risorse che deve ancora accedere e quindi rilascia r, iniziando la sua fase calante.

Lo schedule è quindi uno schedule che può essere stato prodotto da uno scheduler *2PL* se riusciamo ad eseguire tutte le operazioni.
Lo schedule non è invece uno schedule che può essere stato prodotto da uno scheduler *2PL* se ci troviamo in uno stato per cui una transazione t_i ha bisogno di una risorsa r ora in possesso di un'altra transazione t_j e questo lock non può essere rilasciato perché:

- t_j deve ancora accedere ad r
- t_j non può acquisire i lock che le mancano per completare le sue operazioni.

Si noti che valgono le seguenti implicazioni:

$$2PL \Rightarrow CSR \Rightarrow VSR$$

4.2.1 Esercizio Svolto e Commentato

Sia dato lo schedule:

$r_1(x)\ r_1(t)\ r_2(z)\ w_3(x)\ w_1(x)\ r_1(y)\ w_3(t)\ w_2(x)\ w_1(y)$
Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia *VSR* o *CSR*, renderlo tale attraverso la rimozione di una più operazioni (nel quale caso indicare quali)
4. è stato generato da uno scheduler basato su *2PL* base (2 stati)

Soluzione

1. Come primo passo si verifica se lo schedule dato è o meno view serializzabile. Se infatti non lo è, si può subito concludere che lo schedule non è nemmeno *CSR* e *2PL*.

   Per la view equivalenza è necessario per primo individuare le relazioni legge-da. Si scorre allora lo schedule, analizzando ciascuna operazione di lettura e individuando, se esiste, la più recente operazione di scrittura sulla stessa risorsa che la precede.

Lettura	Legge-da
$r_1(y)$	nessuno
$r_2(z)$	nessuno
$r_1(t)$	nessuno
$r_1(x)$	nessuno

Si passa poi alla individuazione delle scritture finali. Si scorre quindi nuovamente lo schedule, partendo dal fondo, e individuando, per ciascuna risorsa, l'ultima operazione di scrittura nello schedule che la riguarda.

Risorsa	Scrittura finale	Altre scritture
x	$w_2(x)$	$w_1(x)$, $w_3(x)$
y	$w_1(y)$	nessuna
z	nessuna	nessuna
t	$w_3(t)$	nessuna

Sulla base delle scritture finali sopra individuate, è possibile dedurre i seguenti vincoli:

- La transazione 2 deve seguire le transazioni 1 e 3 dato che tutte scrivono su x e t_2 è l'ultima transazione ad eseguire una scrittura su x.

Sulla base delle relazioni legge-da sopra individuate e delle scritture finali, è possibile dedurre che:

- La transazione 1 deve precedere la transazione 3. Infatti, se si avesse la transazione 3 e poi la transazione 1, si avrebbe una leggi-da ($r_1(x)$ legge da $w_3(x)$) non presente nello schedule originale.

Lo schedule seriale con cui confrontare quello dato è quindi il seguente (t_1, t_3, t_2):

$$r_1(x), r_1(t), w_1(x), r_1(y), w_1(y), w_3(x), w_3(t), r_2(z), w_2(x)$$

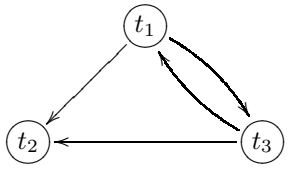
che ha le stesse scritture finali e le stesse relazioni legge-da dello schedule di partenza. Quindi i due schedule sono equivalenti.

Si conclude che lo schedule dato *VSR*.

2. Dato che lo schedule è *VSR*, è possibile che sia anche *CSR*. Per verificarlo, è necessario individuare tutti i conflitti che contiene:

   - $r_1(x), w_3(x)$
   - $r_1(x), w_2(x)$
   - $r_1(t), w_3(t)$
   - $w_3(x), w_1(x)$
   - $w_3(x), w_2(x)$
   - $w_1(x), w_2(x)$

   Si disegna poi il grafo dei conflitti, con un arco per ciascun conflitto. Il grafo dei conflitti riportato di seguito presenta un ciclo fra t_1 e t_3.



Si conclude che lo schedule dato *non* è *CSR*.

3. Per rendere lo schedule conflict-serializzabile è sufficiente eliminare quelle operazioni che contribuiscono alla formazione del ciclo. È quindi possibile rendere lo schedule conflict-serializzabile eliminando una qualsiasi delle seguenti operazioni

   - $w_3(x)$
   - $w_1(x)$
   - $r_1(x)$ e $r_1(t)$
   - $r_1(x)$ e $w_3(t)$

4. Dato che lo schedule non è *CSR*, non può essere stato generato nemmeno da uno scheduler *2PL*. Inoltre, guardando lo schedule, è subito possibile concludere che non è *2PL*, dato che t_1 esegue operazioni sia prima sia dopo t_3 e le due transazioni operano sulle stesse risorse.
   Verifichiamo comunque passo passo che lo schedule non è *2PL*.
   Le transazioni che costituiscono lo schedule sono

   $t_1 = r_1(x)\ r_1(t)\ w_1(x)\ r_1(y)\ w_1(y)$
   $t_2 = r_2(z)\ w_2(x)$
   $t_3 = w_3(x)\ w_3(t)$

   Valutando le singole operazioni che compongono lo schedule, e tenendo traccia dei lock di cui necessitano, si ottiene:

Operazione	Lock	x	y	z	t
$r_1(x)$	lock(t_1, x): OK	t_1			
$r_1(t)$	lock(t_1, t): OK	t_1			t_1
$r_2(z)$	lock(t_2, z): OK	t_1		t_2	t_1
$w_3(x)$	lock(t_1, y): OK	t_1	t_1	t_2	t_1
	unlock(t_1, x): NO	t_1	t_1	t_2	t_1

   Visto che non è possibile eseguire l'unlock da parte della transizione t_1 sulla risorsa x (infatti t_1 ha nel suo scheduler $w_1(x)$, ancora da eseguire), questo schedule non può essere 2PL.

4.3 Timestamp

Il metodo dei timestamp viene usato nel controllo della concorrenza per garantire la serializzabilità di uno schedule.
Esistono due tecniche per la gesione dei timestamp:

- *monoversione*: esiste una sola copia di ciascuna risorsa
- *multiversione*: ogni operazione di scrittura consentita, genera una nuova copia della risorsa.

Algoritmo Monoversione Ciascuna risorsa x ha associati due valori:

- $RTM(x)$: rappresenta il timestamp della transazione che ha effettuato la lettura più recente;
- $WTM(x)$: rappresenta il timestamp della transazione che ha effettuato la scrittura più recente

Una operazione viene accordata o rifiutata in base alle seguenti regole:

- `read`(x, ts): rifiutata se $ts < WTM(x)$, altrimenti viene accettata e si aggiorna $RTM(x) = max(RTM(x), ts)$
- `write`(x, ts): rifiutata se $ts < RTM(x)$ o $ts < WTM(x)$, altrimenti viene accettata e si aggiorna $WTM(x) = ts$

Algoritmo Multiversione Ciascuna versione x_i della risorsa x ha associati due valori:

- $RTM(x_i)$: rappresenta il timestamp della transazione che ha effettuato la lettura più recente sulla versione i-esima
- $WTM(x_i)$: rappresenta il timestamp della transazione che ha creato la copia i-esima

Una operazione viene accordata o rifiutata in base alle seguenti regole:

- `read`(x, ts): sempre accettata
  viene letta la versione x_i il cui WTM è pari a
  $max(WTM(x_j) : WTM(x_j) \leq ts)$, si aggiorna $RTM(x_i) = max(RTM(x_i), ts)$
- `write`(x, ts): si individua la versione x_i il cui WTM è pari a $max(WTM(x_j) : WTM(x_j) \leq ts)$; la richiesta è rifiutata se $ts < RTM(x_i)$, altrimenti si crea una nuova versione x_{i+1} con $WTM(x_{i+1}) = RTM(x_{i+1}) = ts$

Si noti che i due algoritmi qui presentati non sono gli unici adottabili per i timestamp.

4.3.1 Esercizio Svolto e Commentato

Si consideri un controllo di concorrenza monoversione e multiversione basato sui timestamp e un oggetto x con timestamp RTM(x)=3 e WTM(x)=1.
Si considerino le seguenti operazioni:

`read`(x,5), `read`(x,10), `write`(x, 7), `write`(x,12), `read`(x,8), `read`(x,11), `read`(x,15), `write`(x,20), `read`(x,22), `read`(x,18), `read`(x,4), `write`(x,6)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno;
2. l'eventuale transazione uccisa;
3. nel caso monoversione: i nuovi valori di RTM(x) e WTM(x);
4. nel caso multiversione:
   - se si tratta di una operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);
   - se si tratta di una operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione Monoversione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		3	1	
`read`(x,5)	OK	5	=1	–
`read`(x,10)	OK	10	=1	–
`write`(x, 7)	NO	=10	=1	t_7
`write`(x,12)	OK	=10	12	–
`read`(x,8)	NO	=10	=12	t_8
`read`(x,11)	NO	=10	=12	t_{11}
`read`(x,15)	OK	15	=12	–
`write`(x,20)	OK	=15	20	–
`read`(x,22)	OK	22	=20	–
`read`(x,18)	NO	=22	=20	t_{18}
`read`(x,4)	NO	=22	=20	t_4
`write`(x,6)	NO	=22	=20	t_6

La transazione t_7 viene uccisa poiché è già avvenuta una lettura a timestamp successivo `read`(x,10).
Le transazioni t_8 e t_{11} vengono uccise poiché è già avvenuta una scrittura a timestamp successivo `write`(x,12).
Le transazioni t_{18}, t_4 e t_6 vengono uccise poiché è già avvenuta una lettura con timestamp successivo `read`(x,22).

Soluzione Multiversione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	3	1	
`read`(x,5)	OK	x_1	5	=1	–
`read`(x,10)	OK	x_1	10	=1	–
`write`(x, 7)	NO	–	–	–	t_7
`write`(x,12)	OK	x_2	12	12	–
`read`(x,8)	OK	x_1	=10	=1	–
`read`(x,11)	OK	x_1	11	=1	–
`read`(x,15)	OK	x_2	=15	=12	–
`write`(x,20)	OK	x_3	20	20	–
`read`(x,22)	OK	x_3	22	=20	–
`read`(x,18)	OK	x_2	18	=12	–
`read`(x,4)	OK	x_1	=11	=1	–
`write`(x,6)	NO	–	–	–	t_6

Le transazioni t_7 e t_6 vengono uccise poiché non è possibile creare una nuova versione di x dato che x_1, creata al tempo 1 è già stata letta al tempo 10, successivo sia a 6 che a 7.
Si noti che le transazioni t_8, t_{11}, t_{18} e t_4 sono qui accettate perché si tratta di letture.

4.4 Esercizi con Soluzione

4.4.1 Ripresa a Caldo

4.4.1.1 Esercizio 1

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T1), B(T2), U(T2,O1,B1,A1), I(T1,O2,A2), U(T1,O3,B3,A3),
C(T1), B(T3), U(T3,O4,B4,A4), B(T4), I(T2,O5,A5), U(T3,O6,B6,A6),
D(T4,O7,B7), A(T2), B(T5), CK(...), B(T6), I(T5,O8,A8), C(T3),
I(T6,O9,A9), C(T5), B(T7), D(T7,O10,B10), U(T7,O11,B11,A11),
C(T4), GUASTO

Si richiede di:

1. scrivere, in corrispondenza di ogni record di CheckPoint, le transazioni attive;
2. illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

1. Le transazioni attive al CheckPoint sono T3, T4 e T5.
   Il log da considerare per la ripresa a caldo è quindi il seguente:

   DUMP, B(T1), B(T2), U(T2,O1,B1,A1), I(T1,O2,A2), U(T1,O3,B3,A3),
   C(T1), B(T3), U(T3,O4,B4,A4), B(T4), I(T2,O5,A5), U(T3,O6,B6,A6),
   D(T4,O7,B7), A(T2), B(T5), CK(T3,T4,T5), B(T6), I(T5,O8,A8), C(T3),
   I(T6,O9,A9), C(T5), B(T7), D(T7,O10,B10), U(T7,O11,B11,A11),
   C(T4), GUASTO

2. - Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T3, T4, T5)	T3, T4, T5	
B(T6)	T3, T4, T5, T6	
C(T3)	T4, T5, T6	T3
C(T5)	T4, T6	T3, T5
B(T7)	T4, T6, T7	T3, T5
C(T4)	T6, T7	T4, T3, T5

   - Si ripristina la situazione prima del guasto:

     UNDO {T6, T7}

RECORD	AZIONE
U(T7,O11,B11,A11)	O11:=B11
D(T7,O10,B10)	`insert` O10 := B10
I(T6,O9,A9)	`delete` O9

REDO {T4, T3, T5}

RECORD	AZIONE
U(T3,O4,B4,A4)	O4:=A4
U(T3,O6,B6,A6)	O6:=A6
D(T4,O7,B7)	`delete` O7
I(T5,O8,A8)	`insert` O8:=A8

4.4.1.2 Esercizio 2

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

B(T1), B(T2), B(T3), I(T1,O1,A1), D(T2,O2,B2), B(T4), U(T4,O3,B3,A3), U(T1,O4,B4,A4), C(T2), CK(T1,T3,T4), B(T5), B(T6), B(T7), A(T4), U(T7,O6,B6,A6), U(T6,O3,B7,A7), B(T8), A(T7), GUASTO

Si richiede di illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

- Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T1,T3,T4)	T1, T3, T4	
B(T5)	T1, T3, T4, T5	
B(T6)	T1, T3, T4, T5, T6	
B(T7)	T1, T3, T4, T5, T6, T7	
B(T8)	T1, T3, T4, T5, T6, T7, T8	

- Si ripristina la situazione prima del guasto:
  UNDO {T1, T3, T4, T5, T6, T7, T8}

RECORD	AZIONE
U(T6,O3,B7,A7)	O3:=B7
U(T7,O6,B6,A6)	O6:=B6
U(T1,O4,B4,A4)	O4:=B4
U(T4,O3,B3,A3)	O3:=B3
I(T1,O1,A1)	`delete` O1

REDO { }

4.4.1.3 Esercizio 3

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T1), B(T2), D(T1, O1, B1), B(T3), U(T2, O2, B2, A2), B(T4),
D(T4, O3, B3), U(T1, O4, B4, A4), C(T3), CK(...), B(T5),
U(T5, O5, B5, A5), B(T6), B(T7), CK(...), C(T1), C(T4), D(T7, O6, B6),
U(T6, O3, B7, A7), A(T7), GUASTO

Si richiede di:

1. scrivere, in corrispondenza di ogni record di CheckPoint, le transazioni attive;
2. illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

1. Le transazioni attive al primo CheckPoint sono T1, T2 e T4, mentre quelle attive al secondo sono T1, T2, T4, T5, T6 e T7.
   Il log da considerare per la ripresa a caldo è quindi il seguente:

   DUMP, B(T1), B(T2), D(T1, O1, B1), B(T3), U(T2, O2, B2, A2), B(T4),
   D(T4, O3, B3), U(T1, O4, B4, A4), C(T3), CK(T1, T2, T4), B(T5),
   U(T5, O5, B5, A5), B(T6), B(T7), CK(T1, T2, T4, T5, T6, T7), C(T1), C(T4),
   D(T7, O6, B6), U(T6, O3, B7, A7), A(T7), GUASTO

2. - Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T1, T2, T4, T5, T6, T7)	T1, T2, T4, T5, T6, T7	
C(T1)	T2, T4, T5, T6, T7	T1
C(T4)	T2, T5, T6, T7	T1, T4

   - Si ripristina la situazione prima del guasto:
     UNDO {T2, T5, T6, T7}

RECORD	AZIONE
U(T6, O3, B7, A7)	O3:=B7
D(T7, O6, B6)	`insert` O6:=B6
U(T5, O5, B5, A5)	O5:=B5
U(T2, O2, B2, A2)	O2:=B2

REDO {T1, T4}

RECORD	AZIONE
D(T1, O1, B1)	`delete` O1
D(T4, O3, B3)	`delete` O3
U(T1, O4, B4, A4)	O4:=A4

4.4.1.4 Esercizio 4

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T1), B(T2), U(T2,O1,B1,A1), I(T1,O2,A2), B(T3), C(T1), B(T4),
A(T2), CK(T3,T4), U(T3,O2,B3,A3), U(T4,O3,B4,A4), B(T5), C(T4),
U(T3,O3,B5,A5), U(T5,O4,B6,A6), D(T3,05,B7), A(T3), C(T5),
I(T2,O6,A8) GUASTO

Si richiede di illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

- Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T3, T4)	T3, T4	
B(T5)	T3, T4, T5	
C(T4)	T3, T5	T4
C(T5)	T3	T4, T5

- Si ripristina la situazione prima del guasto:
  UNDO {T3}

RECORD	AZIONE
D(T3,05,B7)	`insert` O5:=B7
U(T3,O3,B5,A5)	O3:=B5
U(T3,O2,B3,A3)	O2:=B3

REDO {T4, T5}

RECORD	AZIONE
U(T4,O3,B4,A4)	O3:=A4
U(T5,O4,B6,A6)	O4:=A6

4.4.1.5 Esercizio 5

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T1), B(T2), I(T1,O1,A1), U(T2,O2,B2,A2), B(T3), U(T1,O3,B3,A3), U(T3,O4,B4,A4), C(T1), B(T4), D(T4,O5,B5), U(T3,O6,B6,A6), I(T2,O7,A7), A(T2), B(T5), CK(...), B(T6), I(T5,O8,A8), C(T3), I(T6,O9,A9), C(T5), B(T7), U(T7,O10,B10,A10), C(T4), D(T7,O11,B11), GUASTO

Si richiede di:

1. scrivere, in corrispondenza di ogni record di CheckPoint, le transazioni attive;
2. illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

1. Le transazioni attive al CheckPoint sono T3, T4 e T5.
   Il log da considerare per la ripresa a caldo è quindi il seguente:

   DUMP, B(T1), B(T2), I(T1,O1,A1), U(T2,O2,B2,A2), B(T3), U(T1,O3,B3,A3), U(T3,O4,B4,A4), C(T1), B(T4), D(T4,O5,B5), U(T3,O6,B6,A6), I(T2,O7,A7), A(T2), B(T5), CK(T3,T4,T5), B(T6), I(T5,O8,A8), C(T3), I(T6,O9,A9), C(T5), B(T7), U(T7,O10,B10,A10), C(T4), D(T7,O11,B11), GUASTO

2. - Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T3, T4, T5)	T3, T4, T5	
B(T6)	T3, T4, T5, T6	
C(T3)	T4, T5, T6	T3
C(T5)	T4, T6	T3, T5
B(T7)	T4, T6, T7	T3, T5
C(T4)	T6, T7	T3, T4, T5

   - Si ripristina la situazione prima del guasto:
     UNDO {T6, T7}

RECORD	AZIONE
D(T7, O11, B11)	`insert` O11:=B11
U(T7, O10, B10, A10)	O10:=B10
I(T6, O9, A9)	`delete` O9

REDO {T3, T4, T5}

RECORD	AZIONE
U(T3, O4, B4, A4)	O4:=A4
D(T4, O5, B5)	`delete` O5
U(T3, O6, B6, A6)	O6:=A6
I(T5, O8, B8)	`insert` O8:=A8

4.4.1.6 Esercizio 6

Si consideri il log di un sistema di gestione di basi di dati che contiene la seguente sequenza di record.

DUMP, B(T0), B(T1), B(T2), U(T0,O1,B0,A0), I(T0,O6,A1), I(T1,O2,A2), U(T2,O2,B3,A3), U(T0,O3,B4,A4), B(T3), I(T3,O5,A10), A(T0), B(T4), B(T5), I(T4,O4,A7), C(T2), CK(...), U(T4,O2,B8,A8), C(T4), U(T5,O2,B6,A6), D(T5,O4,B9), U(T1,O2,B11,A11), A(T3), D(T1,O3,B5), C(T1), GUASTO

Si richiede di:

1. scrivere, in corrispondenza di ogni record di CheckPoint, le transazioni attive;
2. illustrare dettagliatamente i passi da compiere per effettuare la ripresa a caldo.

Soluzione

1. Le transazioni attive al CheckPoint sono T1, T3, T4 e T5.
   Il log da considerare per la ripresa a caldo è quindi il seguente:

   DUMP, B(T0), B(T1), B(T2), U(T0,O1,B0,A0), I(T0,O6,A1), I(T1,O2,A2), U(T2,O2,B3,A3), U(T0,O3,B4,A4), B(T3), I(T3,O5,A10), A(T0), B(T4), B(T5), I(T4,O4,A7), C(T2), CK(T1, T3, T4, T5), U(T4,O2,B8,A8), C(T4), U(T5,O2,B6,A6), D(T5,O4,B9), U(T1,O2,B11,A11), A(T3), D(T1,O3,B5), C(T1), GUASTO

2. - Si determinano gli insiemi di UNDO e di REDO

RECORD	UNDO	REDO
CK(T1, T3, T4, T5)	T1, T3, T4, T5	
C(T4)	T1, T3, T5	T4
C(T1)	T3, T5	T1, T4

   - Si ripristina la situazione prima del guasto:
     UNDO {T3, T5}

Record	Azione
D(T5,O4,B9)	`insert` O4:=B9
U(T5,O2,B6,A6)	O2:=B6
I(T3,O5,A10)	`delete` O5

REDO {T1, T4}

Record	Azione
I(T1,O2,A2)	`insert` O2:=A2
I(T4,O4,A7)	`insert` O4:=A7
U(T4,O2,B8,A8)	O2:=A8
U(T1,O2,B11,A11)	O2:=A11
D(T1,O3,B5)	`delete` O3

4.4.2 Serializzabilità

4.4.2.1 Esercizio 1

Dato lo schedule:

$r_1(x)\ w_1(y)\ w_2(x)\ r_3(y)\ w_3(z)\ r_4(z)\ r_4(t)\ w_2(t)\ r_2(t)\ w_3(t)$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una più operazioni (nel quale caso indicare quali)
4. è stato generato da uno scheduler basato su 2PL base (3 stati)

Rispondere a ciascun punto indipendetemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_2(t)$	$w_2(t)$
$r_4(t)$	nessuno
$r_4(z)$	$w_3(z)$
$r_3(y)$	$w_1(y)$
$r_1(x)$	nessuno

Scritture finali

Risorsa	Scrittura finale	Altre scritture
x	$w_2(x)$	nessuna
y	$w_1(y)$	nessuna
z	$w_3(z)$	nessuna
t	$w_3(t)$	$w_2(t)$

Vincoli sulle scritture finali

- La transazione 3 deve seguire la transazione 2 dato che entrambe scrivono su t e t_3 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- La transazione 4 deve seguire la transazione 3 per la lettura su z
- La transazione 3 deve seguire la transazione 1 per la lettura su y

Nello schedule seriale di riferimento, la transazione t_4 dovrebbe essere eseguita dopo la transazione t_3, per il primo vincolo di lettura.
Se così fosse, però, anche $r_4(t)$ sarebbe eseguita dopo $w_3(t)$, introducendo una relazione legge-da su $r_4(t)$ che non è presente nel nostro schedule.

Si conclude che lo schedule dato *non* è VSR

2. Possiamo inoltre concludere immediatamente che lo schedule *non* è neppure conflict-equivalente perché gli schedule conflict equivalenti sono un sottoinsieme di quelli view-equivalenti per cui *non è possibile* avere schedule che *non sono view-equivalenti ma che sono conflict-equivalenti*.

3. Per ridurre il nostro schedule ad uno schedule serializzabile dovremmo togliere *almeno* una delle operazioni che impedivano l'esistenza di uno schedule seriale equivalente.

   Per individuare quali sono queste operazioni, e poiché CSR $\subset$ VSR, è conveniente operare sul grafo dei conflitti.

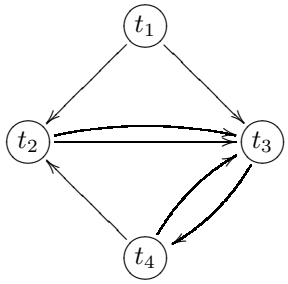
   I conflitti presenti nello schedule dato sono i seguenti

   - $r_1(x), w_2(x)$
   - $w_1(y), r_3(y)$
   - $w_3(z), r_4(z)$
   - $r_4(t), w_2(t)$
   - $r_4(t), w_3(t)$
   - $r_2(t), w_3(t)$
   - $w_2(t), w_3(t)$

Il grafo dei conflitti riportato di seguito presenta due cicli:

(a) t_3, t_4

(b) t_2, t_3, t_4.



L'unico arco che, se rimosso, toglie entrambi i cicli è quello comune ai due: $t_3 \longrightarrow t_4$, che corrisponde al conflitto: $w_3(z), r_4(z)$.

Dovremmo quindi togliere una delle due operazioni: $w_3(z)$ oppure $r_4(z)$, per rendere lo schedule CSR (e quindi anche VSR).

4. Le transazioni da considerare sono

$t_1 = r_1(x)\ w_1(y)$
$t_2 = w_2(x)\ w_2(t)\ r_2(t)$
$t_3 = r_3(y)\ w_3(z)\ w_3(t)$
$t_4 = r_4(z)\ r_4(t)$

Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t
$r_1(x)$	r_lock(t_1, x): OK	r_lock(t_1)			
$w_1(y)$	w_lock(t_1, y): OK unlock(t_1, x): OK unlock(t_1, y): OK	r_lock(t_1)	w_lock(t_1) w_lock(t_1)		
$w_2(x)$	w_lock(t_2, x): OK	w_lock(t_2)			
$r_3(y)$	r_lock(t_3, y): OK	w_lock(t_2)	r_lock(t_3)		
$w_3(z)$	w_lock(t_3, z): OK	w_lock(t_2)	r_lock(t_3)	w_lock(t_3)	
$r_4(z)$	w_lock(t_3, t): OK unlock(t_3, z): OK r_lock(t_4, z): OK	w_lock(t_2) w_lock(t_2) w_lock(t_2)	r_lock(t_3) r_lock(t_3) r_lock(t_3)	w_lock(t_3) r_lock(t_4)	w_lock(t_3) w_lock(t_3) w_lock(t_3)
$r_4(t)$	r_lock(t_4, t): NO				

Visto che non è possibile eseguire il lock da parte della transizione t_4 sulla risorsa t (infatti t_3 ha nel suo scheduler $w_3(t)$, ancora da eseguire), questo schedule non può essere 2PL.

4.4.2.2 Esercizio 2

Dato lo schedule:

$$w_1(x)\ r_1(x)\ w_1(t)\ w_2(x)\ w_2(t)\ r_3(x)\ r_3(t)\ w_4(y)\ r_2(z)\ r_1(y)\ r_3(t)\ r_4(x)\ w_4(x)\ w_3(t)$$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una più operazioni (nel quale caso indicare quali)
4. è stato generato da uno scheduler basato su 2PL base (2 stati)

Rispondere a ciascun punto indipendetemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_4(x)$	$w_2(x)$
$r_3(t)$	$w_2(t)$
$r_1(y)$	$w_4(y)$
$r_2(z)$	nessuno
$r_3(t)$	$w_2(t)$
$r_3(x)$	$w_2(x)$
$r_1(x)$	$w_1(x)$

Scritture finali

Risorsa	Scrittura finale	Altre scritture
x	$w_4(x)$	$w_1(x)$, $w_2(x)$
y	$w_4(y)$	nessuna
z	nessuna	nessuna
t	$w_3(y)$	$w_1(t)$, $w_2(t)$

Vincoli sulle scritture finali

- La transazione 4 deve seguire le transazioni 1 e 2 dato che tutte scrivono su x e t_4 esegue l'ultima operazione di scrittura
- La transazione 3 deve seguire le transazione 1 e 2 dato che tutte scrivono su t e t_3 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- La transazione 4 deve seguire la transazione 2 per la lettura su x
- La transazione 3 deve seguire la transazione 2 per la lettura su t
- La transazione 1 deve seguire la transazione 4 per la lettura su y

Ci fermiamo qui con l'analisi dei vincoli dati dalle relazioni legge-da perché l'ultimo vincolo è in contrasto con il primo dettato dalle scritture finali: la transazione t_4 non può precedere e seguire la transazione t_1 in uno schedule seriale

Si conclude che lo schedule *non* è VSR

2. Possiamo inoltre concludere immediatamente che lo schedule *non* è neppure conflict-equivalente perché gli schedule conflict equivalenti sono un sottoinsieme di quelli view-equivalenti per cui *non è possibile* avere schedule che *non sono view-equivalenti ma che sono conflict-equivalenti*.

3. Per ridurre il nostro schedule ad uno schedule serializzabile dovremmo togliere *almeno* una delle operazioni che impedivano l'esistenza di uno schedule seriale equivalente.

   Per individuare quali sono queste operazioni, e poiché CSR $\subset$ VSR, è conveniente operare sul grafo dei conflitti.

   I conflitti presenti nello schedule dato sono i seguenti

   - $w_1(x), w_2(x)$
   - $w_1(x), r_3(x)$
   - $w_1(x), r_4(x)$
   - $w_1(x), w_4(x)$
   - $r_1(x), w_2(x)$
   - $r_1(x), w_4(x)$
   - $w_1(t), w_2(t)$
   - $w_1(t), r_3(t)$
   - $w_1(t), r_3(t)$
   - $w_1(t), w_3(t)$
   - $w_2(x), r_3(x)$
   - $w_2(x), r_4(x)$

- $w_2(x), w_4(x)$
- $w_2(t), r_3(t)$
- $w_2(t), r_3(t)$
- $w_2(t), w_3(t)$
- $r_3(x), w_4(x)$
- $w_4(y), r_1(y)$

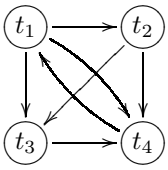
Il grafo dei conflitti riportato di seguito presenta quattro cicli:

(a) t_1, t_4

(b) t_1, t_2, t_4

(c) t_1, t_3, t_4

(d) t_1, t_2, t_3, t_4..



L'unico arco che, se rimosso, toglie tutti i cicli è quello comune ai quettro cicli: $t_4 \longrightarrow t_1$, che corrisponde al conflitto: $w_4(y), r_1(y)$.

Dovremmo quindi togliere una delle due operazioni: $w_4(y)$ oppure $r_1(y)$, per rendere lo schedule CSR (e quindi anche VSR).

4. Le transazioni da considerare sono

$t_1 = w_1(x)\ r_1(x)\ w_1(t)\ r_1(y)$
$t_2 = w_2(x)\ w_2(t)\ r_2(z)$
$t_3 = r_3(x)\ r_3(t)\ r_3(t)\ w_3(t)$
$t_4 = w_4(y)\ r_4(y)\ w_4(x)$

Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t
$w_1(x)$	lock(t_1, x): OK	t_1			
$r_1(x)$		t_1			
$w_1(t)$	lock(t_1, t): OK	t_1			t_1
$w_2(x)$	lock(t_1, y): OK	t_1	t_1		t_1
	unlock(t_1, x): OK		t_1		t_1
	lock(t_2, x): OK	t_2	t_1		t_1
$w_2(t)$	unlock(t_1, t): OK	t_2	t_1		
	lock(t_2, x): OK	t_2	t_1		t_2
$r_3(x)$	lock(t_2, z): OK	t_2	t_1	t_2	t_2
	unlock(t_2, x): OK		t_1	t_2	t_2
	lock(t_3, x): OK	t_3	t_1	t_2	t_2
$r_3(t)$	unlock(t_2, t): OK	t_3	t_1	t_2	
	lock(t_3, t): OK	t_3	t_1	t_2	t_3
$w_4(y)$	unlock(t_1, y): NO				

Visto che non è possibile eseguire l'unlock da parte della transizione t_1 sulla risorsa y (infatti t_1 ha nel suo scheduler $r_1(y)$, ancora da eseguire), questo schedule non può essere 2PL.

4.4.2.3 Esercizio 3

Dato lo schedule:

$$r_1(x)\ r_3(x)\ w_3(x)\ r_2(x)\ r_2(v)\ r_1(t)\ w_1(r)\ r_3(r)\ w_1(y)\ w_2(z)\ w_3(t)\ w_3(v)\ w_1(t)\ w_3(t)$$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una più operazioni (nel quale caso indicare quali)
4. è stato generato da uno scheduler basato su 2PL base (2 stati)

Rispondere a ciascun punto indipendetemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_3(r)$	$w_1(r)$
$r_1(t)$	nessuno
$r_2(v)$	nessuno
$r_2(x)$	$w_3(x)$
$r_3(x)$	nessuno
$r_1(x)$	nessuno

Scritture finali

Risorsa	Scrittura finale	Altre scritture
t	$w_3(t)$	$w_1(t)$
v	$w_3(v)$	nessuna
z	$w_2(z)$	nessuna
y	$w_1(y)$	nessuna
r	$w_1(r)$	nessuna
x	$w_3(x)$	nessuna

Vincoli sulle scritture finali

- La transazione 3 deve seguire la transazione 1 dato che entrambe scrivono su x e t_3 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- La transazione 2 deve seguire la transazione 3 per la lettura su x
- La transazione 3 deve seguire la transazione 1 per la lettura su r

Lo schedule seriale con cui confrontare quello dato è quindi il seguente

$r_1(x)\ r_1(t)\ w_1(r)\ w_1(y)\ w_1(t)\ r_3(x)\ w_3(x)\ r_3(r)\ w_3(t)\ w_3(v)\ w_3(t)\ r_2(x)\ r_2(v)\ w_2(z)$

che ha le stesse scritture finali e ha tutte le relazioni legge-da dello schedule di partenza, con l'aggiunta però della relazione $r_2(v)$ legge-da $w_3(v)$

Si conclude che lo schedule *non* è VSR

2. Possiamo inoltre concludere immediatamente che lo schedule dato *non* è neppure conflict-equivalente perché gli schedule conflict equivalenti sono un sottoinsieme di quelli view-equivalenti per cui *non è possibile* avere schedule che *non sono view-equivalenti ma che sono conflict-equivalenti*.
3. Per ridurre il nostro schedule ad uno schedule serializzabile dovremmo togliere *almeno* una delle operazioni che impedivano l'esistenza di uno schedule seriale equivalente.

Per individuare quali sono queste operazioni, e poiché CSR $\subset$ VSR, è conveniente operare sul grafo dei conflitti.

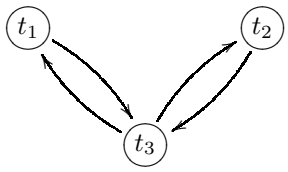
I conflitti presenti nello schedule sono i seguenti

- $r_1(x), w_3(x)$
- $w_3(x), r_2(x)$
- $r_2(v), w_3(v)$
- $r_1(t), w_3(t)$
- $r_1(t), w_3(t)$
- $w_1(r), r_3(r)$
- $w_3(t), w_1(t)$
- $w_1(t), w_3(t)$

Il grafo dei conflitti riportato di seguito presenta due cicli:

(a) t_1, t_3

(b) t_2, t_3



Gli archi da rimuovere per eliminare i due cicli sono $t_3 \longrightarrow t_1$ e $t_2 \longrightarrow t_3$, si scelgono questi archi per eliminare poche operazioni. Corrisponodono infatti ai conflitti $w_3(t), w_1(t)$ e $r_2(v), w_3(v)$.

Dovremmo quindi togliere una delle due operazioni: $w_3(t)$ oppure $w_1(t)$, e una fra le due operazioni $r_2(v)$ e $w_3(v)$ per rendere lo schedule CSR (e quindi anche VSR).

4. Le transazioni da considerare sono

$t_1 = r_1(x)\ r_1(t)\ w_1(r)\ w_1(y)\ w_1(t)$
$t_2 = r_2(x)\ r_2(r)\ w_2(y)$
$t_3 = r_3(x)\ w_3(x)\ r_3(r)\ w_3(t)\ w_3(r)\ w_3(t)$

Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t	r	v
$r_1(x)$	lock(t_1, x): OK	t_1					
$r_3(x)$	lock(t_1, t): OK	t_1			t_1		
	lock(t_1, r): OK	t_1			t_1	t_1	
	lock(t_1, y): OK	t_1	t_1				
	unlock(t_1, x): OK		t_1		t_1	t_1	
	lock(t_3, x): OK	t_3	t_1		t_1	t_1	
$w_3(x)$		t_3	t_1		t_1	t_1	
$r_2(x)$	unlock(t_1, r): NO	t_3	t_1		t_1	t_1	

Visto che non è possibile eseguire l'unlock da parte della transizione t_1 sulla risorsa r (infatti t_1 ha nel suo scheduler $w_1(w)$, ancora da eseguire), questo schedule non può essere 2PL.

4.4.2.4 Esercizio 4

Dato lo schedule:

$$r_1(x)\ w_1(x)\ w_1(t)\ w_2(x)\ w_2(t)\ r_3(x)\ r_3(t)\ r_4(y)\ r_2(z)\ r_1(y)\ r_4(x)\ w_4(x)\ w_3(t)$$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una o più operazioni (nel quale caso indicare quali).
4. è stato generato da uno scheduler basato su 2PL base (a tre stati).

Rispondere a ciascun punto indipendentemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_1(x)$	nessuno
$r_3(x)$	$w_2(x)$
$r_3(t)$	$w_2(t)$
$r_4(y)$	nessuno
$r_2(z)$	nessuno
$r_1(y)$	nessuno
$r_4(x)$	$w_2(x)$

Scritture finali

Dato	Scrittura finale	Altre scritture
x	$w_4(x)$	$w_2(x)$, $w_1(x)$
y	nessuna	nessuna
z	nessuna	nessuna
t	$w_3(t)$	$w_2(t)$, $w_1(t)$

Vincoli sulle scritture finali

- La transazione 4 deve seguire le transazioni 1 e 2 dato che tutte scrivono su x e t_4 esegue l'ultima operazione di scrittura
- La transazione 3 deve seguire le transazioni 1 e 2 dato che tutte scrivono su t e t_3 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- La transazione 3 deve seguire la transazione 2 per le letture su x e t
- La transazione 4 deve seguire la transazione 2 per la lettura su x
- Si osserva che la transazione 1 deve precedere le transazioni 2 e 4. Infatti, se si avesse la transazioni 2 oppure la transazione 4 e poi la transazione 1, si avrebbe una leggi-da ($r_1(x)$ legge da $w_2(x)$ o da $w_4(x)$) non presente nello schedule originale.

Lo schedule seriale con cui confrontare quello dato è quindi il seguente

$r_1(x)\ w_1(x)\ w_1(t)\ r_1(y)\ w_2(x)\ w_2(t)\ r_2(z)\ r_3(x)\ r_3(t)\ w_3(t)\ r_4(y)\ r_4(x)\ w_4(x)$

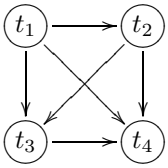
che ha le stesse scritture finali e le stesse relazioni legge-da dello schedule di partenza

Si conclude che lo schedule *è VSR*

2. I conflitti presenti nello schedule sono i seguenti:
   - $r_1(x), w_2(x)$
   - $r_1(x), w_4(x)$
   - $w_1(x), w_2(x)$
   - $w_1(x), r_3(x)$
   - $w_1(x), r_4(x)$
   - $w_1(x), w_4(x)$
   - $w_1(t), w_2(t)$
   - $w_1(t), r_3(t)$
   - $w_1(t), w_3(t)$

- $w_2(x), r_3(x)$
- $w_2(x), r_4(x)$
- $w_2(x), w_4(x)$
- $w_2(t), r_3(t)$
- $r_3(x), w_4(x)$

Il grafo dei conflitti riportato di seguito è aciclico.



Si conclude che lo schedule è CSR.

3. Lo schedule è già sia CSR che VSR.

4. Le transazioni da considerare sono

$t_1 = r_1(x)\ w_1(x)\ w_1(t)\ r_1(y)$
$t_2 = w_2(x)\ w_2(t)\ r_2(z)$
$t_3 = r_3(x)\ r_3(t)\ w_3(t)$
$t_4 = r_4(y)\ r_4(x)\ w_4(x)$

Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t
$r_1(x)$	r_lock(t_1, x): OK	r_lock(t_1)			
$w_1(x)$	w_lock(t_1, x): OK	w_lock(t_1)			
$w_1(t)$	w_lock(t_1, t): OK	w_lock(t_1)			w_lock(t_1)
$w_2(x)$	r_lock(t_1, y): OK	w_lock(t_1)	r_lock(t_1)		w_lock(t_1)
	unlock(t_1, x): OK		r_lock(t_1)		w_lock(t_1)
	w_lock(t_2, x): OK	w_lock(t_2)	r_lock(t_1)		w_lock(t_1)
$w_2(t)$	unlock(t_1, t): OK	w_lock(t_2)	r_lock(t_1)		
	w_lock(t_2, t): OK	w_lock(t_2)	r_lock(t_1)		w_lock(t_2)
$r_3(x)$	r_lock(t_2, z): OK	w_lock(t_2)	r_lock(t_1)	r_lock(t_2)	w_lock(t_2)
	unlock(t_2, x): OK		r_lock(t_1)	r_lock(t_2)	w_lock(t_2)
	r_lock(t_3, x): OK	r_lock(t_3)	r_lock(t_1)	r_lock(t_2)	w_lock(t_2)
$r_3(t)$	unlock(t_2, t): OK	r_lock(t_3)	r_lock(t_1)	r_lock(t_2)	
	r_lock(t_3, t): OK	r_lock(t_3)	r_lock(t_1)	r_lock(t_2)	r_lock(t_3)
$r_4(y)$	r_lock(t_4, y): OK	r_lock(t_3)	r_lock(t_1, t_4)	r_lock(t_2)	r_lock(t_3)
$r_2(z)$	OK	r_lock(t_3)	r_lock(t_1, t_4)	r_lock(t_2)	r_lock(t_3)
$r_1(y)$	OK	r_lock(t_3)	r_lock(t_1, t_4)	r_lock(t_2=	r_lock(t_3)
$r_4(x)$	r_lock(t_4, x): OK	r_lock(t_3, t_4)	r_lock(t_1, t_4)	r_lock(t_2)	r_lock(t_3)
$w_4(x)$	unlock(t_3, x): OK	r_lock(t_4)	r_lock(t_1, t_4)	r_lock(t_2)	r_lock(t_3)
	w_lock(t_4, x): OK	w_lock(t_4)	r_lock(t_1, t_4)	r_lock(t_2)	r_lock(t_3)
$w_3(t)$	w_lock(t_3, t): OK	w_lock(t_4)	r_lock(t_1, t_4)	r_lock(t_2)	w_lock(t_3)

Lo schedule può essere stato prodotto da uno scheduler 2PL con lock a tre stati.

4.4.2.5 Esercizio 5

Dato lo schedule:

$r_1(x)\ r_1(y)\ r_2(z)\ r_3(x)\ w_1(t)\ w_2(z)\ r_2(x)\ r_2(t)\ w_3(y)\ w_2(t)$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una o più operazioni (nel quale caso indicare quali).
4. è stato generato da uno scheduler basato su 2PL base (a due stati).

Rispondere a ciascun punto indipendentemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_1(x)$	nessuno
$r_1(y)$	nessuno
$r_2(z)$	nessuno
$r_3(x)$	nessuno
$r_2(x)$	nessuno
$r_2(t)$	$w_1(t)$

Scritture finali

Dato	Scrittura finale	Altre scritture
x	nessuna	nessuna
y	$w_3(y)$	nessuna
z	$w_2(z)$	nessuna
t	$w_2(t)$	$w_1(t)$

Vincoli sulle scritture finali

- La transazione 2 deve seguire la transazione 1 dato che entrambe scrivono su t e t_2 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- La transazione 2 deve seguire la transazione 1 per la lettura su t
- Si osserva che la transazione 1 deve precedere la transazione 3. Infatti, se si avesse la transazione 3 e poi la transazione 1, si avrebbe una leggi-da ($r_1(y)$ legge da $w_3(y)$) non presente nello schedule originale.

Lo schedule seriale con cui confrontare quello dato è quindi il seguente

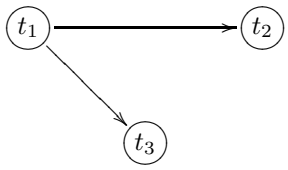
$r_1(x)\ r_1(y)\ w_1(t)\ r_2(z)\ w_2(z)\ r_2(x)\ r_2(t)\ w_2(t)\ r_3(x)\ w_3(y)$

che ha le stesse scritture finali e le stesse relazioni legge-da dello schedule di partenza
Si conclude che lo schedule dato *è VSR*

2. I conflitti presenti nello schedule sono i seguenti:

- $r_1(y), w_3(y)$
- $w_1(t), r_2(t)$
- $w_1(t), w_2(t)$

Il grafo dei conflitti riportato di seguito è aciclico.



Si conclude che lo schedule è CSR.

3. Lo schedule è già sia CSR che VSR.

4. Le transazioni da considerare sono

   $t_1 = r_1(x)\ r_1(y)\ w_1(t)$
   $t_2 = r_2(z)\ w_2(z)\ r_2(x)\ r_2(t)\ w_2(t)$
   $t_3 = r_3(x)\ w_3(y)$

   Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t
$r_1(x)$	lock(t_1, x): OK	t_1			
$r_1(y)$	lock(t_1, y): OK	t_1	t_1		
$r_2(z)$	lock(t_2, z): OK	t_1	t_1	t_2	
$r_3(x)$	lock(t_1, t): OK	t_1	t_1	t_2	t_1
	unlock(t_1, x): OK		t_1	t_2	t_1
	lock(t_3, x): OK	t_3	t_1	t_2	t_1
$w_1(t)$	OK	t_3	t_1	t_2	t_1
$w_2(z)$	OK	t_3	t_1	t_2	t_1
$r_2(x)$	unlock(t_1, y): OK	t_3		t_2	t_1
	lock(t_3, y): OK	t_3	t_3	t_2	t_1
	unlock(t_3, x): OK		t_3	t_2	t_1
	lock(t_2, x): OK	t_2	t_3	t_2	t_1
$r_2(t)$	unlock(t_1, t): OK	t_2	t_3	t_2	
	lock(t_2, t): OK	t_2	t_3	t_2	t_2
$w_3(y)$	OK	t_2	t_3	t_2	t_2
$w_2(t)$	OK	t_2	t_3	t_2	t_2

   Lo schedule può essere stato prodotto da uno scheduler 2PL con lock a due stati.

4.4.2.6 Esercizio 6

Dato lo schedule:

$w_1(x)\ r_1(y)\ w_2(y)\ w_3(t)\ w_3(z)\ w_1(t)$

Determinare se:

1. è view-serializzabile
2. è conflict-serializzabile
3. è possibile, nel caso lo schedule non sia VSR o CSR, renderlo tale attraverso la rimozione di una o più operazioni (nel quale caso indicare quali).
4. è stato generato da uno scheduler basato su 2PL base (a due stati).

Rispondere a ciascun punto indipendentemente dai precedenti.

Soluzione

1. Relazioni legge-da

Lettura	Legge-da
$r_1(y)$	nessuno

Scritture finali

Dato	Scrittura finale	Altre scritture
x	$w_1(x)$	nessuna
y	$w_2(y)$	nessuna
z	$w_3(z)$	nessuna
t	$w_1(t)$	$w_3(t)$

Vincoli sulle scritture finali

- La transazione 1 deve seguire la transazione 3 dato che entrambe scrivono su t e t_1 esegue l'ultima operazione di scrittura

Vincoli sulle relazioni leggi-da

- Si osserva che la transazione 1 deve precedere la transazione 2. Infatti, se si avesse la transazione 2 e poi la transazione 1, si avrebbe una leggi-da ($r_1(y)$ legge da $w_2(y)$) non presente nello schedule originale.

Lo schedule seriale con cui confrontare quello dato è quindi il seguente

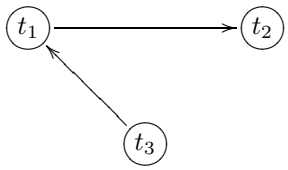
$w_3(t)\ w_3(z)\ w_1(x)\ r_1(y)\ w_1(t)\ w_2(y)$

che ha le stesse scritture finali e le stesse relazioni legge-da dello schedule di partenza
Si conclude che lo schedule dato *è VSR*

2. I conflitti presenti nello schedule dato sono i seguenti:

   - $r_1(y), w_2(y)$
   - $w_3(t), w_1(t)$

   Il grafo dei conflitti riportato di seguito è aciclico.



   Si conclude che lo schedule è CSR.

3. Lo schedule è già sia CSR che VSR.

4. Le transazioni da considerare sono

   $t_1 = w_1(x)\ r_1(y)\ w_1(t)$
   $t_2 = w_2(y)$
   $t_3 = w_3(t)\ w_3(z)$

   Valutando le singole operazioni si ottiene:

Operazione	Lock	x	y	z	t
$w_1(x)$	lock(t_1, x): OK	t_1			
$r_1(y)$	lock(t_1, y): OK	t_1	t_1		
$w_2(y)$	lock(t_1, t): OK	t_1	t_1		t_1
	unlock(t_1, y): OK	t_1			t_1
	lock(t_2, y): OK	t_1	t_2		t_1
$w_3(t)$	unlock(t_1, t): NO				

   Visto che non è possibile eseguire l'unlock da parte della transizione t_1 sulla risorsa t (infatti t_1 ha nel suo schedule $w_1(t)$, ancora da eseguire), questo schedule non può essere 2PL.

4.4.3 Timestamp Monoversione

4.4.3.1 Esercizio 1

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=4 e WTM(x)=2.
Si considerino le seguenti operazioni:

`read`(x,3), `write`(x,6), `write`(x,9), `read`(x,8), `read`(x,10), `write`(x,13), `write`(x,17), `read`(x,14), `write`(x,15), `read`(x,16), `read`(x,18)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno
2. l'eventuale transazione uccisa
3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		4	2	
`read`(x,3)	OK	=4	=2	–
`write`(x,6)	OK	=4	6	–
`write`(x,9)	OK	=4	9	–
`read`(x,8)	NO	=4	=9	t_8
`read`(x,10)	OK	10	=9	–
`write`(x,13)	OK	=10	13	–
`write`(x,17)	OK	=10	17	–
`read`(x,14)	NO	=10	=17	t_{14}
`write`(x,15)	NO	=10	=17	t_{15}
`read`(x,16)	NO	=10	=17	t_{16}
`read`(x,18)	OK	18	=17	–

4.4.3.2 Esercizio 2

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=3 e WTM(x)=2.
Si considerino le seguenti operazioni:

`read`(x,7), `read`(x,8), `read`(x, 5), `write`(x,6), `write`(x,9), `read`(x,12), `read`(x,11), `write`(x,15), `write`(x,14), `read`(x,13), `read`(x,16)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno

2. l'eventuale transazione uccisa

3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		3	2	
`read`(x,7)	OK	7	=2	–
`read`(x,8)	OK	8	=2	–
`read`(x, 5)	OK	=8	=2	–
`write`(x,6)	NO	=8	=2	t_6
`write`(x,9)	OK	=8	9	–
`read`(x,12)	OK	12	=9	–
`read`(x,11)	OK	=12	=9	–
`write`(x,15)	OK	=12	15	–
`write`(x,14)	NO	=12	=15	t_{14}
`read`(x,13)	NO	=12	=15	t_{13}
`read`(x,16)	OK	16	=15	–

4.4.3.3 Esercizio 3

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=4 e WTM(x)=2.
Si considerino le seguenti operazioni:

`read`(x,5), `read`(x,14), `write`(x,13), `write`(x,15), `write`(x,20), `write`(x,26),
`read`(x,17), `read`(x,27), `read`(x,22), `read`(x,30), `write`(x,28)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno;

2. l'eventuale transazione uccisa;

3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		4	2	
read(x,5)	OK	5	=2	–
read(x,14)	OK	14	=2	–
write(x,13)	NO	=14	=2	t_{13}
write(x,15)	OK	=14	15	–
write(x,20)	OK	=14	20	–
write(x,26)	OK	=14	26	–
read(x,17)	NO	=14	=26	t_{17}
read(x,27)	OK	27	=26	–
read(x,22)	NO	=27	=26	t_{22}
read(x,30)	OK	30	=26	–
write(x,28)	NO	=30	=26	t_{28}

4.4.3.4 Esercizio 4

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=3 e WTM(x)=2.
Si considerino le seguenti operazioni:

read(x,5), read(x,1), read(x,4), write(x,6), read(x,10), write(x,8),
write(x,12), read(x,11), read(x,14), read(x,13), write(x,18)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno;
2. l'eventuale transazione uccisa;
3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		3	2	
read(x,5)	OK	5	=2	–
read(x,1)	NO	=5	=2	t_1
read(x,4)	OK	=5	=2	–
write(x,6)	OK	=5	6	–
read(x,10)	OK	10	=6	–
write(x,8)	NO	=10	=6	t_8
write(x,12)	OK	=10	12	–
read(x,11)	NO	=10	=12	t_{11}
read(x,14)	OK	14	=12	–
read(x,13)	OK	=14	=12	–
write(x,18)	OK	=14	18	–

4.4.3.5 Esercizio 5

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=3 e WTM(x)=2.
Si considerino le seguenti operazioni:

`read`(x,4), `write`(x,7), `read`(x,6), `write`(x,10), `read`(x,12), `write`(x,11), `read`(x,9)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno
2. l'eventuale transazione uccisa
3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		3	2	
`read`(x,4)	OK	4	=2	–
`write`(x,7)	OK	=4	7	–
`read`(x,6)	NO	=4	=7	t_6
`write`(x,10)	OK	=4	10	–
`read`(x,12)	OK	12	=10	–
`write`(x,11)	NO	=12	=10	t_{11}
`read`(x,9)	NO	=12	=10	t_9

4.4.3.6 Esercizio 6

Si consideri un controllo di concorrenza monoversione basato sui timestamp e un oggetto x con timestamp RTM(x)=1 e WTM(x)=2.
Si considerino le seguenti operazioni:

`read`(x, 2), `write`(x, 5), `write`(x, 10), `read`(x, 8), `read`(x, 4), `write`(x, 6),
`read`(x,12), `write`(x, 14), `write`(x, 13), `write`(x, 11)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno
2. l'eventuale transazione uccisa
3. i nuovi valori di RTM(x) e WTM(x).

Soluzione

Richiesta	Risposta	RTM	WTM	Trans. uccisa
		1	2	
`read`(x, 2)	OK	2	=2	–
`write`(x, 5)	OK	=2	5	–
`write`(x, 10)	OK	=2	10	–
`read`(x, 8)	NO	=2	=10	t_8
`read`(x, 4)	NO	=2	=10	t_4
`write`(x, 6)	NO	=2	=10	t_6
`read`(x,12)	OK	12	= 10	–
`write`(x, 14)	OK	=12	14	–
`write`(x, 13)	NO	=12	=14	t_{13}
`write`(x, 11)	NO	=12	=14	t_{11}

4.4.4 Timestamp Multiversione

4.4.4.1 Esercizio 1

Si consideri un controllo di concorrenza *multiversione* basato su timestamp e un oggetto x con timestamp RTM(x_1)=15 e WTM(x_1)=15.
Si considerino le seguenti operazioni:

`read`(x, 19), `write`(x, 16), `read`(x, 21), `read`(x, 22), `read`(x, 20),
`write`(x, 25), `write`(x,18), `read`(x, 30), `write`(x, 32)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;

2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);

3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	15	15	–
`read`(x, 19)	OK	x_1	19	=15	–
`write`(x, 16)	NO	–	–	–	t_{16}
`read`(x, 21)	OK	x_1	21	=15	–
`read`(x, 22)	OK	x_1	22	=15	–
`read`(x, 20)	OK	x_1	=22	=15	–
`write`(x, 25)	OK	x_3	25	25	–
`write`(x,18)	NO	–	–	–	t_{18}
`read`(x, 30)	OK	x_3	30	=25	–
`write`(x, 32)	OK	x_4	32	32	–

4.4.4.2 Esercizio 2

Si consideri un controllo di concorrenza multiversione basato sui timestamp e un oggetto x con timestamp RTM(x_1)=10 e WTM(x_1)=8.
Si considerino le seguenti operazioni:

`read`(x,12), `write`(x,12), `write`(x,15), `write`(x,9), `read`(x,14),
`read`(x,17), `write`(x,16), `write`(x,18)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;

2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);

3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	10	8	
`read`(x,12)	OK	x_1	12	=8	–
`write`(x,12)	OK	x_2	12	12	–
`write`(x,15)	OK	x_3	15	15	–
`write`(x,9)	NO	–	–	–	t_9
`read`(x,14)	OK	x_2	14	=12	–
`read`(x,17)	OK	x_3	17	=15	–
`write`(x,16)	NO	–	–	–	t_{16}
`write`(x,18)	Ok	x_4	18	18	–

4.4.4.3 Esercizio 3

Si consideri un controllo di concorrenza multiversione basato sui timestamp e un oggetto x con timestamp RTM(x_1)=7 e WTM(x_1)=5.
Si considerino le seguenti operazioni:

`read`(x,6), `write`(x,6),`write`(x,8), `read`(x,7),
`read`(x,15), `write`(x,16), `read`(x,10), `write`(x,14)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;
2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);
3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	7	5	
`read`(x,6)	OK	x_1	=7	=5	–
`write`(x,6)	NO	–	–	–	t_6
`write`(x,8)	OK	x_2	8	8	–
`read`(x,7)	OK	x_1	=7	=5	–
`read`(x,15)	OK	x_2	15	=8	–
`write`(x,16)	OK	x_3	16	16	–
`read`(x,10)	OK	x_2	=15	=8	–
`write`(x,14)	NO	–	–	–	t_{14}

4.4.4.4 Esercizio 4

Si consideri un controllo di concorrenza *multiversione* basato su timestamp e un oggetto x con timestamp RTM(x_1)=3 e WTM(x_1)=2.
Si considerino le seguenti operazioni:

`read`(x,5), `read`(x,1), `read`(x,4), `write`(x,6), `read`(x,10), `write`(x,8),
`write`(x,12), `read`(x,11), `read`(x,14), `read`(x,13), `write`(x,18)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;
2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);

3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	3	2	
`read`(x,5)	OK	x_1	5	=2	–
`read`(x,3)	OK	x_1	=5	=2	–
`read`(x,4)	OK	x_1	=5	=2	–
`write`(x,6)	OK	x_2	6	6	–
`read`(x,10)	OK	x_2	10	=6	–
`write`(x,8)	NO	–	–	–	t_8
`write`(x,12)	OK	x_3	12	12	–
`read`(x,11)	OK	x_2	11	=6	–
`read`(x,14)	OK	x_3	14	=12	–
`read`(x,13)	OK	x_3	=14	=12	–
`write`(x,18)	OK	x_4	18	18	–

4.4.4.5 Esercizio 5

Si consideri un controllo di concorrenza *multiversione* basato su timestamp e un oggetto x con timestamp RTM(x_1)=3 e WTM(x_1)=2.
Si considerino le seguenti operazioni:

`read`(x,4), `write`(x,7), `read`(x,6), `write`(x,10), `read`(x,12), `write`(x,11), `read`(x,9)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;

2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);

3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	3	2	
`read`(x,4)	OK	x_1	4	=2	–
`write`(x,7)	OK	x_2	7	7	–
`read`(x,6)	OK	x_1	6	=2	–
`write`(x,10)	OK	x_3	10	10	–
`read`(x,12)	OK	x_3	12	=10	–
`write`(x,11)	NO	–	–	–	t_{11}
`read`(x,9)	OK	x_2	9	=7	–

4.4.4.6 Esercizio 6

Si consideri un controllo di concorrenza *multiversione* basato su timestamp e un oggetto x con timestamp RTM(x_1)=1 e WTM(x_1)=2.
Si considerino le seguenti operazioni:

`read`(x, 2), `write`(x, 5), `write`(x, 10), `read`(x, 8), `read`(x, 4), `write`(x, 6), `read`(x,12), `write`(x, 14), `write`(x, 13), `write`(x, 11)

Indicare, per ogni operazione

1. se l'operazione viene accordata o meno e, se non viene accordata, la transazione uccisa;
2. nel caso di operazione di lettura, il numero della versione x_k letta e il nuovo valore di RTM(x_k);
3. nel caso di operazione di scrittura, l'eventuale numero della versione creata e il corrispondente valore di WTM(x_k).

Soluzione

Richiesta	**Risposta**	x_k	**RTM**(x_k)	**WTM**(x_k)	**Trans. uccisa**
		x_1	1	2	
`read`(x, 2)	OK	x_1	2	=2	–
`write`(x, 5)	OK	x_2	5	5	–
`write`(x, 10)	OK	x_3	10	10	–
`read`(x, 8)	OK	x_2	8	=5	–
`read`(x, 4)	OK	x_1	4	=2	–
`write`(x, 6)	NO	–	–	–	t_6
`read`(x,12)	OK	x_3	12	=10	–
`write`(x, 14)	OK	x_4	14	14	–
`write`(x, 13)	OK	x_5	13	13	–
`write`(x, 11)	NO	–	–	–	t_{11}

Capitolo 5

Supporto alle Decisioni

Questo Capitolo è dedicato ad argomenti avanzati di basi di dati. In particolare, ci si concentra sulle operazioni di aggregazione specifiche per i data wharehouse e sul data mining.

5.1 Aggregazioni ROLAP

Per la gestione dei dati in un data wharehouse, lo standard SQL mette a disposizione due operatori di aggregazione aggiuntivi, al fine di migliorare il supporto alle decisioni.
I due operatori in questione utilizzano il valore polimorfo ALL per indicare l'insieme di tutti i valori che un attributo può assumere.
I due operatori sono:

- `with data cube`: fornisce tutte le possibili aggregazioni rispetto agli attributi argomento della clausola;
- `with roll up`: fornisce le aggregazioni ottenute considerando, nell'ordine, gli attributi da sinistra a destra rispetto a come sono elencati nell'argomento della clausola.

5.1.1 Esercizio Svolto e Commentato

Si consideri la seguente tabella, risultato di una interrogazione SQL. Si richiede di mostrare il risultato ottenuto applicando l'opzione `with cube` (gli attributi su cui si applica tale opzione sono Bevanda e Marca) e l'opzione `with roll-up` (gli attributi su cui si applica tale opzione sono Bevanda e Marca).

Bevanda	Marca	QuantitàTotale
Tè	Vera	100
Tè	San Benedetto	250
Acqua	San Benedetto	300

Soluzione

Risultato con l'opzione `with cube`

Bevanda	Marca	QuantitàTotale
Tè	Vera	100
Tè	San Benedetto	250
Tè	ALL	350
Acqua	San Benedetto	300
Acqua	ALL	300
ALL	Vera	100
ALL	San Benedetto	550
ALL	ALL	650

Come si può notare dalla tabella risultante, il valore polimorfo ALL compare in tutte le possibili combinazioni con i valori assunti dagli attributi Bevanda e Marca. Infatti, l'operatore `with cube` computa tutte le possibili aggregazioni.

Risultato con l'opzione `with roll-up`

Bevanda	Marca	QuantitàTotale
Tè	Vera	100
Tè	San Benedetto	250
Acqua	San Benedetto	300
Tè	ALL	350
Acqua	ALL	300
ALL	ALL	650

Come si può notare dalla tabella risultante, il valore polimorfo ALL compare solo alla destra di valori specifici per gli altri attributi. In particolare, si nota che la tabella ottenuta presenta il valore ALL solo nel triangolo inferiore destro.

5.2 Data Mining

Tra i metodi impiegati per l'estrazione di informazioni da un data wharehouse, le *regole di associazione* sono fra le più usate.
Le regole di associazione hanno lo scopo di individuare regolarità nei dati. Sono composte da due parti: *premessa*, o corpo della regola; e *conseguenza*, o testa della regola.
Ogni regola di associazione è caratterizzata in genere da due valori:

- *supporto*: è la probabilità che in una transazione siano presenti sia la premessa che la conseguenza della regola ed indica l'importanza della stessa; si calcola come il numero delle transazioni in cui compaiono testa e corpo della regola in rapporto con il totale delle transazioni

- *confidenza*: è la probabilità che, in una transazione dove c'è la premessa di una regola, sia presente anche la conseguenza e indica la correttezza della regola; si calcola come il numero delle transazioni in cui compaiono testa e corpo della regola in rapporto con il numero di transazioni dove compare la premessa

Sia la premessa sia la conseguenza di una regola di associazione possono essere rappresentate da un singolo valore o da un insieme. In generale, nell'analisi delle regole di associazione, si ammette la presenza di insiemi di valori solo nella premessa o solo nella conseguenza, poiché hanno un diverso significato. Infatti, considerando il punto di vista dell'analisi market basket, nel caso la premessa sia un insieme e la conseguenza un solo elemento, si analizza cosa promuovere per incrementare le vendite della conseguenza. Al contrario, avendo un solo elemento nella premessa e un insieme nella conseguenza, si analizza quali prodotti sarebbe opportuno abbinare alla vendita dell'elemento premessa.
Assumiamo, negli esercizi che seguono, di considerare il primo dei due tipi di analisi.

5.2.1 Esercizio Svolto e Commentato

Si consideri la seguente tabella.

NumTransazione	Data	Telefonia	Prezzo
1	4/7/06	Cellulare	50
1	4/7/06	SIM	50
1	4/7/06	Auricolare	10
2	5/7/06	Cellulare	150
2	5/7/06	SIM	50
3	10/7/06	Cellulare	150
3	10/7/06	Auricolare	10
4	12/7/06	SIM	50

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 20%.

5.2.1.1 Soluzione

Corpo	Testa	Supporto	Confidenza
Cellulare	SIM	2/4=0.50	2/3=0.66
Cellulare	Auricolare	2/4=0.50	2/3=0.66
SIM	Cellulare	2/4=0.50	2/3=0.66
SIM	Auricolare	1/4=0.25	1/3=0.33
Auricolare	Cellulare	2/4=0.50	2/2=1.00
Auricolare	SIM	1/4=0.25	1/20.50
{Cellulare, SIM}	Auricolare	1/4=0.25	1/2=0.50
{Cellulare, Auricolare}	SIM	1/4=0.25	1/2=0.50
{SIM, Auricolare}	Cellulare	1/4=0.25	1/1=1.00

Per compilare la tabella sopra indicata, si individuano tutte le coppie di valori che compaiono insieme in una transazione. Si indicano i due elementi della coppia come premessa e conseguenza di due regole, che sono l'una l'inverso dell'altra. Si procede poi con le terne di

valori, definendo tre regole per ciascuna terna, alternando i valori nella testa e nel corpo delle regole. Lo stesso si fa con insiemi di quattro (cinque e così via) valori, che nello specifico esempio non esistono poiché la transazione con più prodotti ne ha tre.
Si passa poi al calcolo dei valori di supporto e confidenza per ciascuna regola, secondo definizioni riportate nelle sezione precedente. Si noti che le regole con premessa e conseguenza invertite hanno lo stesso supporto, ma diversa confidenza. Nel calcolo del supporto, è importante notare che tutte le regole hanno lo stesso denominatore, mente il numeratore può cambiare. Per quanto riguarda invece la confidenza, regole con uguale premessa hanno uguale denominatore, regole con uguale conseguenza hanno uguale numeratore.
Come ultimo passo si rimuovono quelle regole i cui valori di supporto e confidenza non rispettano i vincoli del problema.

5.3 Esercizi con Soluzione

5.3.1 Aggregazioni ROLAP

5.3.1.1 Esercizio 1

Si consideri la seguente tabella, risultato di una interrogazione SQL. Si richiede di mostrare il risultato ottenuto applicando l'opzione `with cube` (gli attributi su cui si applica tale opzione sono Marca, Forma e Confezione) e l'opzione `with roll-up` (gli attributi su cui si applica tale opzione sono Marca, Forma e Confezione).

Marca	Forma	Confezione	Sum(Quantità)
Barilla	Fusilli	500	100
Barilla	Penne	500	120
Agnesi	Fusilli	500	150
Voiello	Spaghetti	500	90

Soluzione Risultato con l'opzione `with cube`

Marca	Forma	Confezione	Sum(Quantità)
Barilla	Fusilli	500	100
Barilla	Penne	500	120
Barilla	Fusilli	ALL	100
Barilla	Penne	ALL	120
Barilla	ALL	500	220
Barilla	ALL	ALL	220
Agnesi	Fusilli	500	150
Agnesi	Fusilli	ALL	150
Agnesi	ALL	500	150
Agnesi	ALL	ALL	150
Voiello	Spaghetti	500	90
Voiello	Spaghetti	ALL	90
Voiello	ALL	500	90
Voiello	ALL	ALL	90
ALL	Fusilli	500	250
ALL	Penne	500	120
ALL	Spaghetti	500	90
ALL	ALL	500	460
ALL	Fusilli	ALL	250
ALL	Penne	ALL	120
ALL	Spaghetti	ALL	90
ALL	ALL	ALL	460

Risultato con l'opzione `with roll-up`

Marca	Forma	Confezione	Sum(Quantità)
Barilla	Fusilli	500	100
Barilla	Penne	500	120
Agnesi	Fusilli	500	150
Voiello	Spaghetti	500	90
Barilla	Fusilli	ALL	100
Barilla	Penne	ALL	120
Agnesi	Fusilli	ALL	150
Voiello	Spaghetti	ALL	90
Barilla	ALL	ALL	220
Agnesi	ALL	ALL	150
Voiello	ALL	ALL	90
ALL	ALL	ALL	460

5.3.1.2 Esercizio 2

Si consideri la seguente tabella, risultato di una interrogazione SQL. Si richiede di mostrare il risultato ottenuto applicando l'opzione `with cube` (gli attributi su cui si applica tale opzione sono Animale e Marca) e l'opzione `with roll-up` (gli attributi su cui si applica tale opzione sono Animale e Marca).

Animale	Marca	QuantitàTotale
Cane	Trudi	100
Cane	Winnie	50
Delfino	Trudi	150

Soluzione Risultato con l'opzione `with cube`

Animale	Marca	Quantità Totale
Cane	Trudi	100
Cane	Winnie	50
Cane	ALL	150
Delfino	Trudi	150
Delfino	ALL	150
ALL	Trudi	250
ALL	Winnie	50
ALL	ALL	300

Risultato con l'opzione `with roll-up`

Animale	Marca	Quantità Totale
Cane	Trudi	100
Cane	Winnie	50
Delfino	Trudi	150
Cane	ALL	150
Delfino	ALL	150
ALL	ALL	300

5.3.1.3 Esercizio 3

Si consideri la seguente tabella, risultato di una interrogazione SQL. Si richiede di mostrare il risultato ottenuto applicando l'opzione `with cube` (gli attributi su cui si applica tale opzione sono Tratto, Colore e Spessore) e l'opzione `with roll-up` (gli attributi su cui si applica tale opzione sono Tratto, Colore e Spessore).

Tratto	Colore	Spessore	Sum(Quantità)
Stilografica	Blu	Fine	100
A Sfera	Rossa	Fine	50
A Sfera	Blu	Fine	200

Soluzione Risultato con l'opzione `with cube`

Tratto	Colore	Spessore	Sum(Quantità)
Stilografica	Blu	Fine	100
Stilografica	Blu	ALL	100
Stilografica	ALL	Fine	100
Stilografica	ALL	ALL	100
A Sfera	Rossa	Fine	50
A Sfera	Blu	Fine	200
A Sfera	Rossa	ALL	50
A Sfera	Blu	ALL	200
A Sfera	ALL	Fine	250
A Sfera	ALL	ALL	250
ALL	Blu	Fine	300
ALL	Rossa	Fine	50
ALL	Blu	ALL	300
ALL	Rossa	ALL	50
ALL	ALL	Fine	350
ALL	ALL	ALL	350

Risultato con l'opzione `with roll-up`

Tratto	Colore	Spessore	Sum(Quantità)
Stilografica	Blu	Fine	100
A Sfera	Rossa	Fine	50
A Sfera	Blu	Fine	200
Stilografica	Blu	ALL	100
A Sfera	Rossa	ALL	50
A Sfera	Blu	ALL	200
Stilografica	ALL	ALL	100
A Sfera	ALL	ALL	250
ALL	ALL	ALL	350

5.3.1.4 Esercizio 4

Si consideri la seguente tabella, risultato di una interrogazione SQL. Si richiede di mostrare il risultato ottenuto applicando l'opzione `with cube` (gli attributi su cui si applica tale opzione sono Marca, Tipo e Confezione) e l'opzione `with roll-up` (gli attributi su cui si applica tale opzione sono Marca, Tipo e Confezione).

Marca	Tipo	Confezione	Sum(Quantità)
Parmalat	Parzialmente Scremato	1 lt	100
Parmalat	Intero	1 lt	50
Granarolo	Intero	1 lt	70
Granarolo	Intero	0.5 lt	120

Soluzione Risultato con l'opzione `with cube`

Marca	Tipo	Confezione	Sum(Quantità)
Parmalat	Parzialmente Scremato	1 lt	100
Parmalat	Intero	1 lt	50
Parmalat	Parzialmente Scremato	ALL	100
Parmalat	Intero	ALL	50
Parmalat	ALL	1 lt	150
Parmalat	ALL	ALL	150
Granarolo	Intero	1 lt	70
Granarolo	Intero	0.5 lt	120
Granarolo	Intero	ALL	190
Granarolo	ALL	1 lt	70
Granarolo	ALL	0.5 lt	120
Granarolo	ALL	ALL	190
ALL	Parzialmente Scremato	1 lt	100
ALL	Intero	1 lt	120
ALL	Intero	0.5 lt	120
ALL	Parzialmente Scremato	ALL	100
ALL	Intero	ALL	240
ALL	ALL	1 lt	220
ALL	ALL	0.5 lt	120
ALL	ALL	ALL	340

Risultato con l'opzione `with roll-up`

Marca	Tipo	Confezione	Sum(Quantità)
Parmalat	Parzialmente Scremato	1 lt	100
Parmalat	Intero	1 lt	50
Granarolo	Intero	1 lt	70
Granarolo	Intero	0.5 lt	120
Parmalat	Parzialmente Scremato	ALL	100
Parmalat	Intero	ALL	50
Granarolo	Intero	ALL	190
Parmalat	ALL	ALL	150
Granarolo	ALL	ALL	190
ALL	ALL	ALL	340

5.3.2 Data Mining

5.3.2.1 Esercizio 1

Si consideri la seguente tabella.

NumTransazione	Data	Oggetto	Prezzo
1	4/2/05	occhiali	20
1	4/2/05	crema	2
1	4/2/05	pantaloncini	10
2	5/3/05	matita	1
2	5/3/05	gomma	2
2	5/3/05	quaderno	3
3	9/3/05	gomma	2
3	9/3/05	quaderno	3
3	9/3/05	matita	1
4	3/4/05	gomma	2
4	3/4/05	crema	2
5	5/7/05	matita	1
5	5/7/05	gomma	2
5	5/7/05	penna	3

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 20%.

Soluzione

Corpo	Testa	Supporto	Confidenza
occhiali	crema	0.20	1.00
occhiali	pantaloncini	0.20	1.00
pantaloncini	occhiali	0.20	1.00
pantaloncini	crema	0.20	1.00
crema	occhiali	0.2	0.50
crema	pantaloncini	0.20	0.50
crema	gomma	0.20	0.50
matita	gomma	0.60	1.00
matita	quaderno	0.40	0.66
matita	penna	0.20	0.33
gomma	matita	0.60	0.75
gomma	quaderno	0.40	0.50
gomma	crema	0.20	0.25
gomma	penna	0.20	0.25
quaderno	matita	0.40	1.00
quaderno	gomma	0.40	1.00
penna	gomma	0.20	1.00
penna	matita	0.20	1.00
{occhiali, crema}	pantaloncini	0.20	1.00
{occhiali, pantaloncini}	crema	0.20	1.00
{pantaloncini, crema}	occhiali	0.20	1.00
{matita, gomma}	quaderno	0.40	0.66
{quaderno, gomma}	matita	0.40	1.00
{quaderno, matita}	gomma	0.40	1.00
{matita, gomma}	penna	0.20	0.33
{penna, gomma}	matita	0.20	1.00
{matita, penna}	gomma	0.20	1.00

5.3.2.2 Esercizio 2

Si consideri la seguente tabella.

NumTransazione	Data	Oggetto	Prezzo
1	10/2/06	arance	10
1	10/2/06	mele	7
1	10/2/06	pere	3
2	15/2/06	pere	4
2	15/2/06	mele	6
3	16/2/06	arance	4
3	16/2/06	pere	2
4	18/2/06	mele	2
4	18/2/06	pere	2
4	18/2/06	banane	6

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 25%.

Soluzione

Corpo	Testa	Supporto	Confidenza
arance	mele	0.25	0.50
arance	pere	0.50	1.00
mele	arance	0.25	0.33
mele	pere	0.75	1.00
mele	banane	0.25	0.33
pere	arance	0.50	0.50
pere	mele	0.75	0.75
pere	banane	0.25	0.25
banane	pere	0.25	1.00
banane	mele	0.25	1.00
{arance, mele}	pere	0.25	1.00
{arance, pere}	mele	0.25	0.50
{mele, pere}	arance	0.25	0.33
{mele, pere}	banane	0.25	0.33
{banane, mele}	pere	0.25	1.00
{banane, pere}	mele	0.25	1.00

5.3.2.3 Esercizio 3

Si consideri la seguente tabella.

NumTransazione	Data	Oggetto	Prezzo
1	3/3/06	girasoli	3
1	3/3/06	rose	5
2	8/3/06	rose	5
2	8/3/06	orchidee	7
2	8/3/06	geranei	2
3	10/3/06	orchidee	7
3	10/3/06	rose	5
4	12/3/06	girasoli	3
4	12/3/06	geranei	2
4	12/3/06	orchidee	7
5	13/3/06	geranei	2
5	13/3/06	girasoli	3

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 20%.

Soluzione

Corpo	Testa	Supporto	Confidenza
girasoli	rose	0.20	0.33
girasoli	geranei	0.40	0.66
girasoli	orchidee	0.20	0.33
rose	girasoli	0.20	0.33
rose	orchidee	0.40	0.66
rose	geranei	0.20	0.33
orchidee	rose	0.40	0.66
orchidee	geranei	0.40	0.66
orchidee	girasoli	0.20	0.33
geranei	rose	0.20	0.33
geranei	orchidee	0.40	0.66
geranei	girasoli	0.40	0.66
{rose, orchidee}	geranei	0.20	0.50
{rose, geranei}	orchidee	0.20	1.00
{orchidee, geranei}	rose	0.20	0.50
{girasoli, geranei}	orchidee	0.20	0.50
{girasoli, orchidee}	geranei	0.20	1.00
{geranei, orchidee}	girasoli	0.20	0.50

5.3.2.4 Esercizio 4

Si consideri la seguente tabella.

NumTransazione	Data	Gelato	Prezzo
1	4/5/06	Fior di Latte	1.50
1	4/5/06	Cioccolato	1.50
2	5/5/06	Cioccolato	2.00
3	13/5/06	Cioccolato	2.30
3	13/5/06	Fior di Latte	2.30
3	13/5/06	Bacio	2.30

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 30%.

Soluzione

Corpo	Testa	Supporto	Confidenza
Fior di Latte	Cioccolato	0.66	1.00
Fior di Latte	Bacio	0.33	0.50
Cioccolato	Fior di Latte	0.66	0.66
Cioccolato	Bacio	0.33	0.33
Bacio	Fior di Latte	0.33	1.00
Bacio	Cioccolato	0.33	1.00
{Fior di Latte, Cioccolato}	Bacio	0.33	0.50
{Fior di Latte, Bacio}	Cioccolato	0.33	1.00
{Bacio, Cioccolato}	Fior di Latte	0.33	1.00

5.3.2.5 Esercizio 5

Si consideri la seguente tabella.

NumTransazione	Data	Colazione	Prezzo
1	3/9/06	Latte	1.20
1	3/9/06	Biscotti	2.40
2	5/9/06	Caffè	2.00
2	5/9/06	Latte	1.20
2	5/9/06	Biscotti	2.40
3	7/9/06	Caffè	2.00
3	7/9/06	Latte	2.40
4	8/9/06	Biscotti	2.40
4	8/9/06	Latte	1.20
4	8/9/06	Tè	1.10

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 25%.

Soluzione

Corpo	Testa	Supporto	Confidenza
Latte	Biscotti	0.75	0.75
Latte	Caffè	0.50	0.50
Latte	Tè	0.25	0.25
Biscotti	Latte	0.75	1.00
Biscotti	Caffè	0.25	0.33
Biscotti	Tè	0.25	0.33
Caffè	Latte	0.50	1.00
Caffè	Biscotti	0.25	0.50
Tè	Latte	0.25	1.00
Tè	Biscotti	0.25	1.00
{Caffè, Latte}	Biscotti	0.25	0.50
{Latte, Biscotti}	Caffè	0.25	0.33
{Biscotti, Caffè}	Latte	0.25	1.00
{Biscotti, Tè}	Latte	0.25	1.00
{Biscotti, Latte}	Tè	0.25	0.33
{Latte, Tè}	Biscotti	0.25	1.00

5.3.2.6 Esercizio 6

Si consideri la seguente tabella.

NumTransazione	Data	Dolci	Prezzo
1	10/09/05	Macine	2.50
1	10/09/05	Dietorelle	1.70
1	10/09/05	Raggianti	3.00
2	12/09/05	Raggianti	3.00
2	12/09/05	Dietorelle	1.70
3	13/09/05	Raggianti	3.00
4	15/09/05	Tegolino	4.00

Si richiede di individuare le regole di associazione con supporto e confidenza maggiori al 25%.

Soluzione

Corpo	Testa	Supporto	Confidenza
Macine	Dietorelle	0.25	1.00
Macine	Raggianti	0.25	1.00
Dietorelle	Macine	0.25	0.50
Dietorelle	Raggianti	0.50	1.00
Raggianti	Macine	0.25	0.33
Raggianti	Dietorelle	0.50	0.66
{Macine, Dietorelle}	Raggianti	0.25	1.00
{Macine, Raggianti}	Dietorelle	0.25	1.00
{Dietorelle, Raggianti}	Macine	0.25	0.50