Capitolo 2

Esercizio 2.1

Considerare le informazioni per la gestione dei prestiti di una biblioteca personale. Il proprietario presta libri ai suoi amici, che indica semplicemente attraverso i rispettivi nomi o soprannomi (così da evitare omonimie) e fa riferimento ai libri attraverso i titoli (non possiede 2 libri con lo stesso titolo). Quando presta un libro, prende nota della data prevista di restituzione. Definire uno schema di relazione per rappresentare queste informazioni, individuando opportuni domini per i vari attributi e mostrarne un'istanza in forma tabellare. Indicare la chiave (o le chiavi) della relazione.

Soluzione:

Queste informazioni possono essere rappresentate da una sola relazione contente i prestiti, perché non ci sono altre informazioni su amici e libri oltre ai nomi e ai titoli.
Un possibile schema è il seguente:

PRESTITO (Titolo, Nome, DataRestituzione)

Questi attributi denotano rispettivamente il titolo del libro, il nome o il soprannome dell'amico e la data di restituzione prevista del libro. La chiave è "Titolo" perché non possiede libri con lo stesso nome, quindi ogni libro è unico. Un amico invece può avere più libri e restituirli in date differenti.

Questo è un esempio in forma tabellare della relazione:

Titolo	**Nome**	**DataRestituzione**
Il signore degli anelli	Vittorio	12/12/2003
Timeline	Danilo	10/08/2003
L'ombra dello scorpione	Angelo	05/11/2003
Piccolo mondo antico	Valerio	15/04/2004

Esercizio 2.2

Rappresentare per mezzo di una o più relazioni le informazioni contenute nell'orario delle partenze di una stazione ferroviaria: numero, orario, destinazione finale, categoria, fermate intermedie, di tutti i treni in partenza.

Soluzione:

Ecco un possibile schema:

PARTENZE (Numero, Orario, Destinazione, Categoria)
FERMATE (Treno, Stazione, Orario)

La relazione PARTENZE rappresenta tutte le partenze della stazione; contiene il numero di treno che è la chiave, l'orario, la destinazione finale e la categoria.
Le fermate sono rappresentate dalla seconda relazione FERMATE, perché il numero di fermate cambia per ogni treno, rendendo impossibile la rappresentazione delle fermate in PARTENZE, che deve avere un numero fisso di attributi. La chiave di questa relazione è composta da due attributi, "Treno" e "Stazione", che indicano il numero di treno e le stazioni in cui si fermano. È necessario introdurre un vincolo di integrità referenziale tra "Treno" in FERMATE e "Numero" in PARTENZE.

Esercizio 2.3 Definire uno schema di base di dati per organizzare le informazioni di un'azienda che ha impiegati (ognuno con codice fiscale, cognome, nome e data di nascita) e filiali (con codice, sede e direttore, che è un impiegato). Ogni impiegato lavora presso una filiale. Indicare le chiavi e i vincoli di integrità referenziale dello schema. Mostrare un'istanza della base di dati e verificare che soddisfi i vincoli.

Soluzione

Gli schemi delle relazioni e le relative chiavi sono indicati nelle tabelle di figura 2.I.

Impiegati				
CF	Cognome	Nome	DataNascita	Filiale
RSS MRA 76E27 H501 Z	Rossi	Mario	27/05/1976	GT09
BRN GNN 90D03 F205 E	Bruni	Giovanni	03/04/1990	AB04
GLL BRN 64E04 F839 H	Gialli	Bruno	04/05/1964	GT09
NRE GNI 64L01 G273 Y	Neri	Gino	01/07/1964	AB04
RSS NNA 45R42 D969 X	Rossi	Anna	02/10/1945	PT67
RGI PNI 77M05 M082 B	Riga	Pino	05/08/1977	AB04

Filiali		
Codice	Sede	Direttore
AB04	Roma Tiburtina	NRE GNI 64L01 G273 Y
GT09	Roma Monteverde	RSS NNA 45R42 D969 X
PT67	Roma Eur	RSS MRA 76E27 H501 Z

Figura 2.I Una base di dati per l'esercizio 2.3

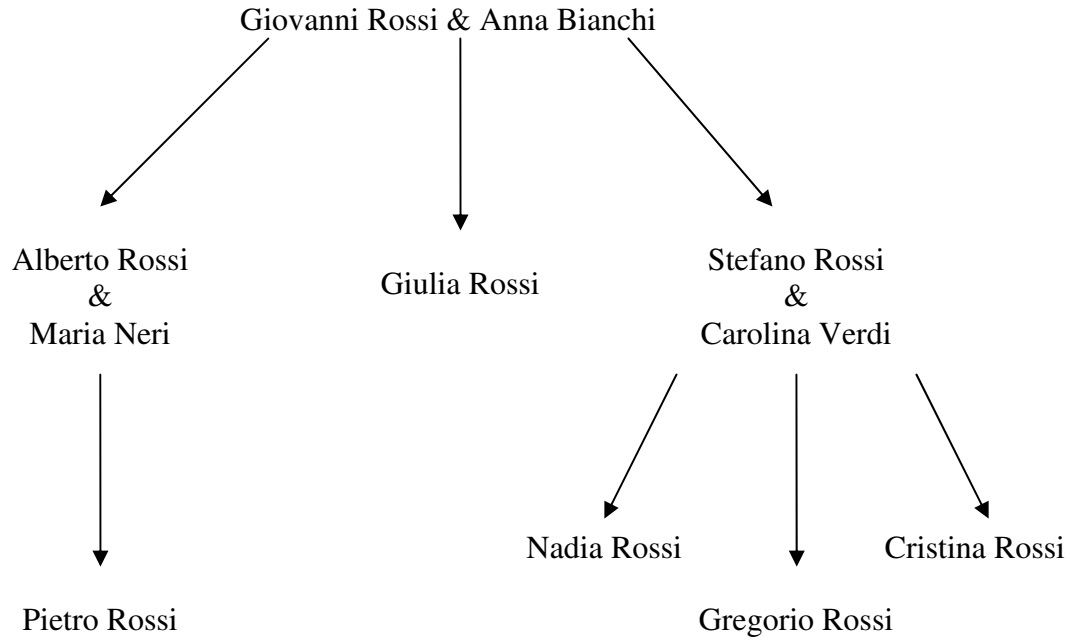
Vi è un vincolo di integrità referenziale fra Filiale della relazione IMPIEGATI e la chiave della relazione FILIALI e un vincoli di integritaà referenziale fra Direttore della relazione FILIALI e la chiave della relazione IMPIEGATI.

Esercizio 2.4

Un albero genealogico rappresenta, in forma grafica, la struttura di una famiglia (o più famiglie, quando è ben articolato). Mostrare come si possa rappresentare, in una base di dati relazionale, un albero genealogico, cominciando eventualmente da una struttura semplificata, in cui si rappresentano solo le discendenze in linea maschile (cioè i figli vengono rappresentati solo per i componenti di sesso maschile) oppure solo quelle in linea femminile.

Soluzione:

Un tipico albero genealogico può essere simile a questo:



Queste informazioni possono essere rappresentate nel database:

MATRIMONIO (Marito, Moglie)
PATERNITÀ (Padre, Figlio)

Questo schema implica che ogni persona abbia un unico nome.
La famiglia vista sopra diventa:

MATRIMONIO

Marito	**Moglie**
Giovanni Rossi	Anna Bianchi
Alberto Rossi	Maria Neri
Stefano Rossi	Carolina Verdi

PATERNITÀ

Padre	**Figlio**
Giovanni Rossi	Alberto Rossi
Giovanni Rossi	Giulia Rossi
Giovanni Rossi	Stefano Rossi
Alberto Rossi	Pietro Rossi
Stefano Rossi	Nadia Rossi
Stefano Rossi	Gregorio Rossi
Stefano Rossi	Cristina Rossi

Per rappresentare anche i rappresentanti femminili è necessario aggiungere un'altra relazione per MATERNITÀ.

Esercizio 2.5

Per ciascuno degli esercizi 2.1 – 2.4, valutare le eventuali esigenze di rappresentazioni di valori nulli, con i benefici e le difficoltà connesse.

Soluzione:

Nello schema dell'esercizio 2.1, i valori nulli possono essere ammessi sull'attributo DataRestituzione, perché è possibile prestare un libro senza aver fissato una precisa data di restituzione; sarebbe difficile accettare valori nulli sull'attributo "Nome", perché di solito è necessario sapere chi ha il libro. L'attributo "Libro" è la chiave e quindi non può avere valori nulli.

Nell'esercizio 2.2, oltre che alle chiavi, è difficile assegnare valori nulli a qualsiasi altro attributo, perché tutte le informazioni che contengono sono molto importanti per i viaggiatori.

Nell'esercizio 2.3, è interessante sottolineare che un valore nullo può essere ammesso nell'attributo "Direttore" nella relazione FILIALE, se il rispettivo valore "Codice" non ha riferimenti in "Filiale" nella relazione IMPIEGATO; questa situazione potrebbe significare, per esempio, che la filiale è appena stata creata e che al momento non ha impiegati. Naturalmente, se "Codice" ha un riferimento in "Filiale", il valore di "Direttore" deve essere presente.
Ovviamente possiamo immaginare valori nulli sugli attributi "Cognome", "Nome" o "DataDiNascita", ma è molto strano che queste informazioni non siano conosciute.

Nell'esercizio 2.4 si possono ammettere valori nulli sull'attributo "Moglie" in MATRIMONIO se consideriamo solo la linea maschile della famiglia.

Esercizio 2.6

Descrivere in linguaggio naturale le informazioni organizzate nella base dati in figura 2.23.

PAZIENTI

Cod	Cognome	Nome
A102	Necchi	Luca
B372	Rossini	Piero
B543	Missoni	Nadia
B444	Missoni	Luigi
S555	Rossetti	Gino

RICOVERI

Paziente	Inizio	Fine	Reparto
A102	2/05/94	9/05/94	A
A102	2/12/94	2/01/95	A
S555	5/10/94	3/12/94	B
B444	1/12/94	2/01/95	B
S555	5/10/94	1/11/94	A

MEDICI

Matr	Cognome	Nome	Reparto
203	Neri	Piero	A
574	Bisi	Mario	B
431	Bargio	Sergio	B
530	Belli	Nicola	C
405	Mizzi	Nicola	A
201	Monti	Mario	A

REPARTI

Cod	Nome	Primario
A	Chirurgia	203
B	Medicina	574
C	Pediatria	530

Soluzione:

Questo database è per un ospedale o per una clinica.
La relazione PAZIENTI contiene le informazioni riguardanti le persone che sono state ammesse almeno una volta. Le persone sono identificate da un codice.
La relazione RICOVERI contiene tutti i ricoveri fatti nell'ospedale. Per ogni ricovero abbiamo il paziente (identificato dal codice), la data di ammissione e di dimissione e il reparto in cui il paziente è stato ricoverato.
La relazione MEDICI contiene le informazioni dei dottori che lavorano per l'ospedale e fornisce il cognome, il nome e il reparto. Il reparto è indicato da un codice. Ogni medico è identificato da un numero di matricola.
La relazione REPARTI descrive i vari reparti dell'ospedale, mostrando per ognuno di essi il nome del reparto e il primario che ne è a capo (attraverso un riferimento alla relazione MEDICI). I reparti sono identificati con un codice (A,B,C)

Esercizio 2.7

Individuare le chiavi e i vincoli di integrità referenziale che sussistono nella base di dati di figura 2.23 e che è ragionevole assumere siano soddisfatti da tutte le basi di dati sullo stesso schema. Individuare anche gli attributi sui quali possa essere sensato ammettere valori nulli.

Soluzione:

Le chiavi sono:
- "Cod" per la relazione PAZIENTI
- "Paziente" e "Inizio" per la relazione RICOVERI
- "Matr" per la relazione MEDICI
- "Cod" per la relazione REPARTI

La scelta fatta sulla relazione RICOVERI presume che un paziente possa essere ricoverato solo una volta nello stesso giorno.
Se supponiamo che questa ipotesi non venga soddisfatta, e che un paziente possa essere ammesso due o più volte nello stesso giorno, la relazione non sarebbe corretta. Infatti due o più ricoveri nello stesso giorno e nello stesso reparto dovrebbero avere anche la stessa data di dimissione, e così sarebbe rappresentata nella stessa riga nella relazione.

I vincoli di integrità che esistono nel database sono tra l'attributo "Paziente" in RICOVERI e "Cod" in PAZIENTI, tra "Reparto" nella relazione RICOVERI e "Cod" nella relazione REPARTI, tra "Primario" in REPARTI e "Matr" nella relazione MEDICI e infine tra "Reparto" in MEDICI e "Cod" in REPARTI.

I valori nulli possono essere ammessi negli attributi "Cognome" e "Nome" nella relazione PAZIENTI, "Fine" nella relazione RICOVERI, "Cognome" e "Nome" nella relazione MEDICI e "Nome" nella relazione REPARTI. Tutti questi attributi non sono chiavi e non hanno nessun vincolo di integrità referenziale.

Esercizio 2.8

Definire uno schema di basi di dati che organizzi i dati necessari a generare la pagina dei programmi radiofonici di un quotidiano, con stazioni, ore e titoli dei programmi; per ogni stazione sono memorizzati, oltre al nome, anche la frequenza di trasmissione e la sede.

Soluzione:

Una possibile soluzione è:

STAZIONE(Nome, Frequenza, Sede)
PROGRAMMA(Titolo, Stazione, Orario)

Questo schema presume che lo stesso titolo di un programma non possa essere utilizzato da due stazioni differenti. Se questo dovesse accadere, il campo chiave per PROGRAMMA dovrebbe essere composto da Titolo e Stazione

Esercizio 2.9 Indicare quali tra le seguenti affermazioni sono vere in una definizione rigorosa del modello relazionale:

1. ogni relazione ha almeno una chiave
2. ogni relazione ha esattamente una chiave
3. ogni attributo appartiene al massimo ad una chiave
4. possono esistere attributi che non appartengono a nessuna chiave
5. una chiave può essere sottoinsieme di un'altra chiave
6. può esistere una chiave che coinvolge tutti gli attributi
7. può succedere che esistano più chiavi e che una di esse coinvolga tutti gli attributi
8. ogni relazione ha almeno una superchiave
9. ogni relazione ha esattamente una superchiave
10. può succedere che esistano più superchiavi e che una di esse coinvolga tutti gli attributi.

Soluzione

1.	ogni relazione ha almeno una chiave	**SÌ**
2.	ogni relazione ha esattamente una chiave	**NO**
3.	ogni attributo appartiene al massimo ad una chiave	**NO**
4.	possono esistere attributi che non appartengono a nessuna chiave	**SÌ**
5.	una chiave può essere sottoinsieme di un'altra chiave	**NO**
6.	può esistere una chiave che coinvolge tutti gli attributi	**SÌ**
7.	può succedere che esistano più chiavi e che una di esse coinvolga tutti gli attributi	**NO**
8.	ogni relazione ha almeno una superchiave	**SÌ**
9.	ogni relazione ha esattamente una superchiave	**NO**
10.	può succedere che esistano più superchiavi e che una di esse coinvolga tutti gli attributi.	**SÌ**

Esercizio 2.10 Considerare la base di dati relazionale in figura 2.24, relativa a impiegati, progetti e partecipazioni di impiegati a progetti. Indicare quali possano

IMPIEGATI

Matricola	Cognome	Nome	Età
101	Rossi	Mario	35
102	Rossi	Anna	42
103	Gialli	Mario	34
104	Neri	Gino	45

PROGETTI

ID	Titolo	Costo
A	Luna	70
B	Marte	60
C	Giove	90

PARTECIPAZIONE

Impiegato	Progetto
101	A
101	B
103	A
102	B

Figura 2.24 Una base di dati per l'esercizio 2.10

essere, per questa base di dati, ragionevoli chiavi primarie e vincoli di integrità referenziale. Giustificare brevemente la risposta, con riferimento alla realtà di interesse (cioè perché si può immaginare che tali vincoli sussistano) e all'istanza mostrata (verificando che sono soddisfatti).

Soluzione

Per per quanto riguarda la relazione IMPIEGATI l'attributo Matricola è una chiave primaria in quanto identifica univocamente un impiegato, non essendoci anche nell'esempio di riferimento duplicati su tale attributo. Allo stesso modo l'attributo ID costituisce una chiave per la relazione PROGETTI. Per quanto riguarda invece la relazione PARTECIPAZIONE la chiave è costituita dagli attributi Impiegato, Progetto.

Sono presenti vincoli di integrità referenziale fra l'attributo Impiegato di PARTECIPAZIONE e la relazione IMPIEGATI e fra l'attributo Progetto di PARTECIPAZIONE e la relazione PROGETTI.

Esercizio 2.11 Si supponga di voler rappresentare in una base di dati relazionale le informazioni relative al calendario d'esami di una facoltà universitaria, che vengono pubblicate con avvisi con la struttura mostrata in figura 2.25.

Calendario esami

Codice	Titolo	Prof	Appello	Data
1	Fisica	Neri	1	01/06/2006
			2	05/07/2006
			3	04/09/2006
			4	30/09/2006
2	Chimica	Rossi	1	06/06/2006
			2	05/07/2006
3	Algebra	Bruni		da definire

Figura 2.25 Un avviso con il calendario d'esami (esercizio 2.11)

Mostrare gli schemi delle relazioni da utilizzare (con attributi e vincoli di chiave e di integrità referenziale) e l'istanza corrispondente ai dati sopra mostrati.
Soluzione
Gli schemi delle relazioni e le relative chiavi sono indicati nella figura 2.II.

Corsi

Codice	Titolo	Prof
1	Fisica	Neri
2	Chimica	Rossi
3	Algebra	Bruni

Appelli

CodiceCorso	Appello	Data
1	1	01/06/2006
1	2	05/07/2006
1	3	04/09/2006
1	4	30/09/2006
2	1	06/06/2006
2	2	05/07/2006

Figura 2.II Una base di dati per l'esercizio 2.11

Vi è un vincolo di integrità referenziale fra CodiceCorso nella relazione APPELLI e la relazione CORSI.

Esercizio 2.12 Si considerino le seguenti relazioni utilizzate per tenere traccia degli studenti di un'università, dei loro esami superati e verbalizzati attraverso gli esoneri e dei loro esami superati e verbalizzati attraverso i comuni appelli:

- ESAMIESONERI (Studente, Materia, VotoEson1, VotoEson2, VotoFinale)
- ESAMIAPPELLI (Studente, Materia, Voto)
- STUDENTI (Matricola, Nome, Cognome).

Indicare i vincoli di intergità che è ragionevole pensare debbano essere soddisfatti da tutte le basi di dati definite su questo schema.

Soluzione

Possiamo individuare i seguenti vincoli di integrità:

1. $17 <$ VotoEson1 < 32
2. $17 <$VotoEson2 < 32
3. $17 <$ VotoFinale < 32
4. $17 <$ Voto < 32
5. VotoFinale = avg(VotoEson1, VotoEson2)
6. (Studente, Materia) chiave per la relazione ESAMIESONERI
7. (Studente, Materia) chiave per la relazione ESAMIAPPELLI
8. Matricola chiave per STUDENTI
9. vincolo di integrità referenziale fra Studente della relazione ESAMIESONERI e STUDENTI
10. vincolo di integrità referenziale fra Studente della relazione ESAMIAPPELLI e STUDENTI
11. una coppia (Studente, Materia) o compare nella tabella ESAMIESONERI o compare nella tabella ESAMIAPPELLI in quanto uno studente non può aver superato un esame in due modi diversi.

Esercizio 2.13 Supponendo di voler rappresentare una base di dati relazionale contenente le informazioni relative agli autori di una serie di libri raccolte secondo la struttura della figura 2.III, mostrare gli schemi delle relazioni da utilizzare (con attributi, vincoli di chiave e vincoli di integrità referenziale) e l'istanza corrispondente ai dati mostrati.

Libri e Autori

Codice	Titolo	Autore	Telefono	Data Pubblicazione
1	Leggende	Neri Aldo	02 345	04/05/2006
		Bianchi Ennio	02 487	04/05/2006
2	Miti	Gialli Enzo	06 343	03/03/2009
3	Fiabe	Neri Aldo	02 345	30/09/2008
		Verdi Lisa	08 467	30/09/2008
		Marroni Ada	09 445	30/09/2008
4	Racconti	Rossi Anna	03 888	06/06/2006
		Bianchi Ennio	02 487	06/06/2006

Figura 2.III Dati raccolti su un insieme di libri ed i loro autori, esercizio 2.13

Soluzione
Gli schemi delle relazioni e le relative chiavi sono indicati nella figura 2.IV.

Sono presenti vincoli di integrità referenziale fra gli attributi Nome, Cognome della relazione PUBBLICAZIONI e la relazione AUTORI e fra l'attributo Libro della relazione PUBBLICAZIONI e la relazione LIBRI.

Libri		
Codice	Titolo	DataPubblicazione
1	Leggende	04/05/2006
2	Miti	03/03/2009
3	Fiabe	30/09/2008
4	Racconti	06/06/2006

Autori		
Nome	Cognome	Telefono
Anna	Rossi	03 888
Aldo	Neri	02 345
Ennio	Bianchi	02 487
Enzo	Gialli	06 343
Ada	Marroni	09 445
Lisa	Verdi	08 467

Pubblicazione		
Nome	Cognome	Libro
Anna	Rossi	4
Aldo	Neri	1
Aldo	Neri	3
Ennio	Bianchi	1
Ennio	Bianchi	4
Enzo	Gialli	2
Ada	Marroni	3
Lisa	Verdi	3

Figura 2.IV Una base di dati per l'esercizio 2.13

Capitolo 3

Esercizio 3.1 Considerare una relazione $R(A, \underline{B}, \underline{C}, D, E)$. Indicare quali delle seguenti proiezioni hanno certamente lo stesso numero di ennuple di R:

1. $\pi_{ABCD}(R)$
2. $\pi_{AC}(R)$
3. $\pi_{BC}(R)$
4. $\pi_{C}(R)$
5. $\pi_{CD}(R)$.

Soluzione

1. $\pi_{ABCD}(R)$ **SÌ**
2. $\pi_{AC}(R)$ **NO**
3. $\pi_{BC}(R)$ **SÌ**
4. $\pi_{C}(R)$ **NO**
5. $\pi_{CD}(R)$ **NO**.

Esercizio 3.2 Considerare le relazioni $R_1(\underline{A}, B, C)$ e $R_2(\underline{D}, E, F)$ aventi rispettivamente cardinalità N_1 e N_2. Assumere che sia definito un vincolo di integrità referenziale fra l'attributo C di R_1 e la chiave D di R_2. Indicare la cardinalità di ciascuno dei seguenti join (specificare l'intervallo nel quale essa può variare):

1. $R_1 \bowtie_{A=D} R_2$
2. $R_1 \bowtie_{C=D} R_2$
3. $R_1 \bowtie_{A=F} R_2$
4. $R_1 \bowtie_{B=E} R_2$.

Soluzione

1. $R_1 \bowtie_{A=D} R_2$ **compresa fra** 0 **e il minimo fra** N_1 **e** N_2
2. $R_1 \bowtie_{C=D} R_2$ **esattamente** N_1
3. $R_1 \bowtie_{A=F} R_2$ **compresa fra** 0 **e** N_2
4. $R_1 \bowtie_{B=E} R_2$ **compresa fra** 0 **e** $N_1 \cdot N_2$**.**

Esercizio 3.3 Considerare le seguenti relazioni (tutte senza valori nulli):

- $R_1(\underline{A}, B, C)$, con vincolo di integrità referenziale fra C e R_2 e con cardinalità $\mathrm{N}_1 = 100$
- $R_2(\underline{D}, E, F)$, con vincolo di integrità referenziale fra F e R_3 e con cardinalità $\mathrm{N}_2 = 200$
- $R_3(\underline{G}, H, I)$, con cardinalità $\mathrm{N}_3 = 50$.

Indicare la cardinalità del risultato di ciascuna delle seguenti espressioni (specificando l'intervallo nel quale essa può variare)

1. $\pi_{AB}(R_1)$
2. $\pi_{E}(R_2)$
3. $\pi_{BC}(R_1)$
4. $\pi_{G}(R_3)$
5. $R_1 \bowtie_{A=D} R_2$
6. $R_1 \bowtie_{C=D} R_2$
7. $R_3 \bowtie_{I=A} R_1$
8. $(R_3 \bowtie_{I=A} R_1) \bowtie_{C=D} R_2$
9. $(R_3 \bowtie_{I=A} R_1) \bowtie_{C=E} R_2$.

Soluzione

1. $|\pi_{AB}(R_1)| = 100$
   la cardinalità dell'operazione è esattamente pari a 100 poiché la proiezione coinvolge la chiave della relazione.
2. $1 \leq |\pi_{E}(R_2)| \leq 200$
   la cardinalità dell'operazione è compresa tra 1 e 200. 1 come cardinalità minima in quanto non possono essere presenti valori nulli e non essendo coinvolta la chiave tutti i valori di E potrebbero essere uguali. 200 come cardinalità massima poiché tutti i valori di E potrebbero essere diversi ed al più saranno in numero tanti quanti la chiave.
3. $1 \leq |\pi_{BC}(R_1)| \leq 100$
   la cardinalità dell'operazione è compresa tra 1 e 100. 1 come cardinalità minima in quanto non possono essere presenti valori nulli e non essendo coinvolta la chiave tutti i valori della proiezione su B e C potrebbero essere uguali. 100 come cardinalità massima poiché tutti i valori della proiezione su B e C potrebbero essere diversi ed al più saranno in numero tanti quanti sono gli elementi della chiave.
4. $|\pi_{G}(R_3)| = 50$
   la cardinalità dell'operazione è esattamente 50 poiché G è chiave per R_3.
5. $0 \leq |R_1 \bowtie_{A=D} R_2| \leq 100$
   la cardinalità dell'operazione è compresa tra 0 e 100. 0 come cardinalità minima in quanto il join potrebbe essere vuoto. 100 come cardinalità massima poiché al più tutti i valori di A si combineranno al più con un valore di D poiché sia A che D sono delle chiavi per le due relazioni.

6. $|R_1 \bowtie_{C=D} R_2| = 100$
   la cardinalità dell'operazione è esattamente 100 poiché esiste un vincolo di integrità referenziale tra C e D e quindi ogni valore di C si combina con esattamente un valore di D.
7. $0 \leq |R_3 \bowtie_{I=A} R_1| \leq 50$
   la cardinalità dell'operazione è compresa tra 0 e 50 poiché al più ogni valore di I si combina con esattamente un valore di A.
8. $0 \leq |(R_3 \bowtie_{I=A} R_1) \bowtie_{C=D} R_2| \leq 50$
   riutilizzando il risultato ottenuto al punto precedente abbiamo che la cardinalità della primo join è compresa tra 0 e 50. La cardinalità del secondo join rimane invariata rispetto a quella ottenuta con il primo in quanto ogni valore di C si combina esattamente con un valore di D e sarà quindi compresa 0 e 50.
9. $0 \leq |(R_3 \bowtie_{I=A} R_1) \bowtie_{C=E} R_2| \leq 10000$
   riutilizzando il risultato ottenuto al punto precedente abbiamo che la cardinalità del primo join è compresa tra 0 e 50. La cardinalità del secondo join sarà compresa tra 0 e 10000 in quanto se i valori di C e D sono tutti diversi si avrà un join vuoto, mentre se sono tutti uguali si avrà il prodotto cartesiano delle tuple.

Esercizio 3.4 Date le relazioni R_1 (A,B,C),R_2 (E,F,G,H),R_3 (J,K), R_4 (L,M) aventi rispettivamente cardinalità N_1, N_2, N_3 e N_4 quali vincoli di chiave e di integrità referenziale vanno definiti (se possibile) affinché nei casi seguenti valgano le condizioni indicate?

1. $|R_1 \bowtie_{B=G} R_2| = N_1$
2. $|R_2 \bowtie_{G=B} R_1| = N_1$
3. $|\pi_J(R_3)| = N_3$
4. $|\pi_J(R_3)| < N_3$
5. $|\pi_L(R_4) \bowtie_{L=J} R_3| = N_4$
6. $|R_4 \bowtie_{M=K} R_3| = N_3$
7. $|R_1 \bowtie_{BC=GK} R_2| = N_2$
8. $|R_1 \bowtie_{BC=GH} R_2| = N_1$
9. $0 \leq |R_1 \bowtie_{A=F} R_2| \leq N_1 \cdot N_2$
10. $|R_1 \bowtie_{A=F} R_2| = N_1 \cdot N_2$.

Soluzione

1. $|R_1 \bowtie_{B=G} R_2| = N_1$
   B chiave, G chiave e vincolo di integrità referenziale tra B e G.
2. $|R_2 \bowtie_{G=B} R_1| = N_1$
   B chiave, G chiave e vincolo di integrità referenziale tra B e G.
3. $|\pi_J R_3| = N_3$
   J chiave.
4. $|\pi_J R_3| < N_3$
   Non è possibile imporre vincoli che garantiscano lo strettamente minore.
5. $|\pi_L(R_4) \bowtie_{L=J} (R_3)| = N_4$
   L chiave, J chiave e vincolo di integrità referenziale tra L e J.
6. $|R_4 \bowtie_{M=K} R_3| = N_3$
   K chiave, M chiave e vincolo di integrità referenziale tra K e M.
7. $|R_1 \bowtie_{BC=GH} R_2| = N_2$
   BC chiave, GH chiave e vincolo di integrità referenziale tra GH e BC.
8. $|R_1 \bowtie_{BC=GH} R_2| = N_1$
   BC chiave, GH chiave e vincolo di integrità referenziale tra BC e GH.
9. $0 \leq |R_1 \bowtie_{A=F} R_2| \leq N_1 \cdot N_2$
   Nessun vincolo, perché la cardinalità è sempre nell'intervallo.
10. $|R_1 \bowtie_{A=F} R_2| = N_1 \cdot N_2$
    Non è possibile imporre vincoli in quanto A e F dovrebbero essere non chiave e con un unico valore.

Esercizio 3.5 Con riferimento ai punti 1 e 2 dell'esercizio precedente, considerando i vincoli di integrità imposti in ogni punto spiegare le differenze che si avrebbero nei risultati delle operazioni nel caso di join destro e join sinistro e come cambia di conseguenza la cardinalità del risultato.
Soluzione

1. Nel caso di join destro il risultato rimane invariato in quanto tutte le tuple di R_1 già partecipano al risultato per il vincolo di integrità referenziale tra B e G. Se invece prendiamo in considerazione un join sinistro il risultato cambia in quanto in esso saranno presenti tutte le tuple di R_2 che prima non partecipavano al risultato. La cardinalità del risultato finale sarà dunque pari a N_2.
2. Nel caso di join sinistro la cardinalità del risultato sarà pari a N_2 in quanto partecipano al risultato anche le tuple di N_2 che altrimenti non partecipavano al risultato. Nel caso di join destro ci ritroviamo nella situazione precedente quindi, considerando la presenza del vincolo di integrità referenziale tra due attributi B e G chiavi la cardinalità sarà pari a N_1.

Esercizio 3.6

Considerare lo schema di base di dati contenente le relazioni:

```
Film(CodiceFilm, Titolo, Regista, Anno, CostoNoleggio)
Artisti(CodiceAttore,Cognome, Nome, Sesso, DataNascita,
        Nazionalità)
Interpretazioni(CodiceFilm,CodiceAttore, Personaggio)
```

1. Mostrare una base di dati su questo schema per la quale i join fra le varie relazioni siano tutti completi.
2. Supponendo che esistano due vincoli di integrità referenziale fra la relazione `Interpretazioni` e le altre due, discutere i possibili casi di join non completo.
3. Mostrare un prodotto cartesiano che coinvolga relazioni in questa base di dati.
4. Mostrare una base di dati per la quale uno (o più) dei join sia vuoto.

Soluzione:

FILM

CodiceFilm	Titolo	Regista	Anno	CostoNoleggio
145684	Armageddon	15434	1997	5000000
457343	La vita è bella	67532	1998	1000000
563822	Ronin	34573	1997	2000000

ARTISTI

CodiceAttore	Cognome	Nome	Sesso	DataNascita	Nazionalità
67532	Benigni	Roberto	M	14/03/1950	Italiana
12456	DeNiro	Robert	M	22/04/1951	Americana
45673	Braschi	Nicoletta	F	1/05/1954	Italiana
67777	Willis	Bruce	M	3/2/1959	Americana
12345	Tyler	Liv	F	18/02/1962	Americana

INTERPRETAZIONI

CodiceFilm	CodiceAttore	Personaggio
457343	67532	Guido
457343	45673	Dora
145684	67777	Harry
145684	12345	Grace
563822	12456	Sam

1. In questa instanza della base di dati, i due join (naturali) tra INTERPRETAZIONI e ARTISTI e tra INTERPRETAZIONI e FILM sono completi (in quanto non vi sono tuple che non partecipano al risultato)
2. Se inserissimo tutti i registi nella tabella ARTISTI, allora il join con INTERPRETAZIONI non sarebbe completo. Il vincolo di integrità referenziale in questo schema è posto su `CodiceAttore` e su `CodiceFilm`. Nella tabella INTERPRETAZIONI non ha senso una tupla in cui i valori `CodiceFilm` e `CodiceAttore` non abbiano corrispondenza nelle tabelle FILM e ARTISTI. È comunque possibile ammettere tuple della tabella ARTISTI senza corrispondenza in INTERPRETAZIONI (ad esempio, nel caso dell'introduzione dei registi tra gli artisti). Questa situazione causa join incompleti.

3. Un esempio di prodotto cartesiano tra ARTISTI e FILM sulla base di dati è qui riportato:

CodiceFilm	Titolo	Reg.	Anno	se di oleggio	Codice Attore	Cognome	Nome	Sesso	DataNascita	Nazionalità
145684	Armageddon	15434	1997	5000000	67532	Benigni	Roberto	M	14/03/50	Italiana
457343	La vita è bella	67532	1998	1000000	67532	Benigni	Roberto	M	14/03/50	Italiana
563822	Ronin	34573	1997	2000000	67532	Benigni	Roberto	M	14/03/50	Italiana
145684	Armageddon	15434	1997	5000000	12456	DeNiro	Robert	M	22/04/51	Americana
457343	La vita è bella	67532	1998	1000000	12456	DeNiro	Robert	M	22/04/51	Americana
563822	Ronin	34573	1997	2000000	12456	DeNiro	Robert	M	22/04/51	Americana
145684	Armageddon	15434	1997	5000000	45673	Braschi	Nicoletta	F	1/05/54	Italiana
457343	La vita è bella	67532	1998	1000000	45673	Braschi	Nicoletta	F	1/05/54	Italiana
563822	Ronin	34573	1997	2000000	45673	Braschi	Nicoletta	F	1/05/54	Italiana
145684	Armageddon	15434	1997	5000000	67777	Willis	Bruce	M	3/2/59	Americana
457343	La vita è bella	67532	1998	1000000	67777	Willis	Bruce	M	3/2/59	Americana
563822	Ronin	34573	1997	2000000	67777	Willis	Bruce	M	3/2/59	Americana
145684	Armageddon	15434	1997	5000000	12345	Tyler	Liv	F	18/02/62	Americana
457343	La vita è bella	67532	1998	1000000	12345	Tyler	Liv	F	18/02/62	Americana
563822	Ronin	34573	1997	2000000	12345	Tyler	Liv	F	18/02/62	Americana

4. Un esempio di Base di dati con join vuoti può essere il seguente: i valori nella tabella INTERPRETAZIONI non hanno corrispondenza nelle altre tabelle:

FILM

CodiceF	Titolo	Regista	Anno	CostoNoleggio
145684	Armageddon	15434	1997	5000000
457343	La vita e bella	67532	1998	1000000
563822	Ronin	34573	1997	2000000

ARTISTI

CodiceAttore	Cognome	Nome	Sesso	DataNascita	Nazionalità
67532	Benigni	Roberto	M	14/03/1950	Italiana
12456	DeNiro	Robert	M	22/04/1951	Americana
45673	Braschi	Nicoletta	F	1/05/1954	Italiana
67777	Willis	Bruce	M	3/2/1959	Americana
12345	Tyler	Liv	F	18/02/1962	Americana

INTERPRETAZIONI

CodiceFilm	CodiceAttore	Personaggio
478384	67500	Peter
467343	42223	Dora
185682	67754	Harry
945684	99845	John
963822	12000	Mark

Esercizio 3.7

Con riferimento allo schema nell'esercizio 3.6, formulare in algebra relazionale, in calcolo su domini, in calcolo su tuple e in Datalog le interrogazioni che trovano:

1. i titoli dei film nei quali Henry Fonda sia stato interprete;
2. i titoli dei film per i quali il regista sia stato anche interprete;
3. i titoli dei film in cui gli attori noti siano tutti dello stesso sesso.

Soluzione:

1.

Algebra Relazionale:

Π_{Titolo}(FILM $\rhd\lhd$ ($\sigma_{(Nome=\text{"Henry"}) \wedge (Cognome=\text{"Fonda"})}$ (ARTISTI) $\rhd\lhd$ INTERPRETAZIONI))

Calcolo dei Domini:

{ Titolo: t | FILM (CodiceFilm: fn, Titolo: t,Regista: d, Anno: y,CostoNoleggio: pc) $\wedge$
ARTISTI (CodiceAttore: an, Cognome:cogn , Nome: n, Sesso: s,
DataNascita:b, Nazionalità:naz) $\wedge$
INTERPRETAZIONI (CodiceFilm: fn, CodiceAttore: an, Personaggio: ch) $\wedge$
(cogn= "Fonda") $\wedge$ (n= "Henry") }

Calcolo delle Tuple:

{ F.titolo | F(FILM), A(ARTISTI), I(INTERPRETAZIONI) |
F.CodiceFilm = I.CodiceFilm $\wedge$ A.CodiceAttore =I. CodiceAttore $\wedge$
A.Cognome= "Fonda" $\wedge$ A.Nome= "Henry" }

Datalog:

FILMCONFONDA (Titolo: t) $\leftarrow$
FILM (CodiceFilm: fn, Titolo: t, Regista: d, Anno: y, CostoNoleggio: pc),
ARTISTI (CodiceAttore: an, Cognome: "Fonda", Nome: "Henry", Sesso: s,DataNascita: b,
Nazionalità : naz),
INTERPRETAZIONI (CodiceFilm : fn, CodiceAttore : an, Personaggio: ch)

2.

Algebra Relazionale:

Π_{Titolo} ($\sigma_{(Regista=CodiceAttore)}$(INTERPRETAZIONI$\rhd\lhd$ FILM))

Calcolo dei Domini:

{ Titolo: t | FILM (CodiceFilm : fn, Titolo: t, Regista: d, Anno: y, CostoNoleggio: pc) $\wedge$

INTERPRETAZIONI(CodiceFilm : fn, CodiceAttore:d, Personaggio: ch) }

Calcolo delle tuple:

{ F.Titolo | F(FILM), I(INTERPRETAZIONI) |
F. CodiceFilm =I. CodiceFilm $\wedge$ F.Regista=I.CodiceAttore}

Datalog:

FILMCONREGISTAARTISTA(Titolo: t) ←
FILM (CodiceFilm:fn, Titolo:t,Regista:d, Anno: y,CostoNoleggio: pc)
INTERPRETAZIONI (CodiceFilm : fn, CodiceAttore: d, Personaggio: ch)

3.

Algebra Relazionale:

Π_{Titolo} (FILM) -
Π_{Titolo} (FILM) $\triangleright\triangleleft \sigma_{Sex<>Sex1}$((ARTISTI $\triangleright\triangleleft$ INTERPRETAZIONI) $\triangleright\triangleleft$
$\rho_{Sex1\leftarrow Sex}(\Pi_{CodiceFilm,Sex}$ (ARTISTI $\triangleright\triangleleft$ INTERPRETAZIONI)))

Calcolo dei Domini:

{ Titolo: t | FILM (CodiceFilm: fn, Titolo:t, Regista:d, Anno: y, CostoNoleggio: pc) ∧

¬∃t1(∃d1(∃y1(∃pd1(FILM(CodiceFilm: fn,Titolo:t1,Regista:d1, Anno: y1, CostoNoleggio:pd1)
∧
ARTISTI (CodiceAttore: an1,Cognome: sur1,Nome: n1, Sesso: s1,
DataNascita: b1, Nazionalità :nat1) ∧
ARTISTI (CodiceAttore: an2,Cognome: sur2,Nome: n2, Sesso: s2,
DataNascita: b2, Nazionalità :nat2) ∧
INTERPRETAZIONI (CodiceFilm: fn, CodiceAttore: an1, Personaggio: ch1)
∧
INTERPRETAZIONI (CodiceFilm: fn, CodiceAttore: an2, Personaggio: ch2)
∧
(s1 ≠ s2))))) }

Calcolo delle Tuple:

{ F.Titolo | F(FILM) |
¬(∃F1(FILM)(∃A1(ARTISTI) (∃A2(ARTISTI) (∃I1(INTERPRETAZIONI)
(∃I2(INTERPRETAZIONI) ∧
A1.CodiceAttore=I1.CodiceAttore∧ F1.CodiceFilm=I1.CodiceFilm ∧
A2.CodiceAttore=I2. CodiceAttore ∧ I1.CodiceFilm =I2.CodiceFilm ∧
A1.Sesso≠A2.Sesso))))) }

Datalog:

SESSODIVERSO (CodiceFilm: fn) ←
FILM (CodiceFilm: fn, Titolo: t,Regista: d, Anno: y, CostoNoleggio: pc),
ARTISTI (CodiceAttore: an1, Cognome:sur1 , Nome: n1, Sesso: s1, DataNascita:b1,
Nazionalità :nat1),
ARTISTI (CodiceAttore: an2, Cognome:sur2 , Nome: n2, Sesso: s2, DataNascita:b2,
Nazionalità :nat2),
INTERPRETAZIONI (FilmNumber: fn, CodiceAttore: an1, Personaggio: ch1),
INTERPRETAZIONI (FilmNumber: fn, CodiceAttore: an2, Personaggio: ch2),
(s1 ≠ s2)

STESSOSESSO(Titolo: t) ←
FILM (CodiceFilm: fn, Titolo: t,Regista: d, Anno: y, CostoNoleggio: pc),
NOT SESSODIVERSO (CodiceFilm: fn)

Esercizio 3.8

Si consideri lo schema di base di dati che contiene le seguenti relazioni:

DEPUTATI (Codice, Cognome, Nome, Commissione, Provincia, Collegio)
COLLEGI (Provincia, Numero, Nome)
PROVINCE (Sigla, Nome, Regione)
REGIONI (Codice, Nome)
COMMISSIONI (Numero, Nome, Presidente)

Formulare in algebra relazionale, in calcolo dei domini e in calcolo delle tuple le seguenti interrogazioni:

1. Trovare nome e cognome dei presidenti di commissioni cui partecipa almeno un deputato eletto in una provincia della Sicilia;
2. Trovare nome e cognome dei deputati della commissione Bilancio;
3. Trovare nome, cognome e provincia di elezione dei deputati della commissione Bilancio;
4. Trovare nome, cognome, provincia e regione di elezione dei deputati della commissione Bilancio;
5. Trovare le regioni in cui vi sia un solo collegio, indicando nome e cognome del deputato ivi eletto;
6. Trovare i collegi di una stessa regione in cui siano stati eletti deputati con lo stesso nome proprio.

Soluzione:

1.

Algebra Relazionale:

$$\Pi_{Nom,\ Cogn}((\rho_{Nom,Cogn \leftarrow Nome,Cognome}(DEPUTATI)) \triangleright\triangleleft_{Presidente=Codice} (COMMISSIONI \triangleright\triangleleft_{Numero=Comm} (\rho_{Comm \leftarrow Commissione}(DEPUTATI \triangleright\triangleleft_{Provincia=Sigla} (PROVINCE \triangleright\triangleleft_{Regione=Codice} (\sigma_{Nome='Sicilia'}(REGIONI)))))))$$

Calcolo dei Domini:

$$\{ Cognome: surp, Nome: fnp \mid DEPUTATI(Codice: np, Cognome: surp, Nome: fnp, Commissione: cmp, Provincia: sigl, Collegio: cstp) \wedge COMMISSIONI(Numero: cm, Nome: nameC, Presidente: np) \wedge COLLEGI(Provincia: usl22, Numero: usl23, Nome: usl32) \wedge PROVINCE(Sigla: sigl, Nome: usl1, Regione: co) \wedge REGIONI(Codice:co, Nome: RegName) \wedge (RegName = \text{“Sicilia”}) \}$$

Calcolo delle Tuple:

$$\{ PRES.(Cognome, Nome) \mid PRES(DEPUTATI), D(Deputati), COM(COMMISSIONI), COL(COLLEGI), P(PROVINCE), R(REGIONI) \mid (PRES.Codice=COM.Presidente) \wedge (COM.Numero=D.Numero) \wedge (D.Provincia=P.Sigla) \wedge (P.Regione=R.Codice) \wedge (R.Nome = \text{“Sicilia”}) \}$$

2.

Algebra Relazionale:

$$\Pi_{\text{NomeC,Cognome}}((\rho_{\text{NomeC}\leftarrow\text{Nome}}(\text{DEPUTATI}))\triangleright\triangleleft_{\text{Commissione=Numero}}(\sigma_{\text{Nome="Bilancio"}}(\text{COMMISSIONE})))$$

Calcolo dei Domini:

$$\{\text{Nome: fn, Cognome: sur} \mid \text{DEPUTATI(Codice: n, Cognome: sur, Nome: fn, Commissione: cm, Provincia:co,Collegio: cst)} \wedge \text{COMMISSIONE(Numero: cm, Nome: nomeC, Presidente: np)} \wedge (\text{nomeC= "Bilancio"}) \}$$

Calcolo delle tuple:

$$\{\text{D.(Nome,Cognome)} \mid \text{D(DEPUTATI), C(COMMISSIONI)} \mid (\text{D.Commissione=C.Numero}) \wedge (\text{C.Nome= "Bilancio"}) \}$$

3.

Algebra Relazionale:

$$\Pi_{\text{NomeC,Cognome,nom1}}(((\rho_{\text{Nom1}\leftarrow\text{Nome}}(\text{PROVINCIA}))\triangleright\triangleleft_{\text{Sigla=Provincia}}((\rho_{\text{Nome1}\leftarrow\text{Nome}}(\text{DEPUTATI}))\triangleright\triangleleft_{\text{Commissione=Numero}}(\sigma_{\text{Nome="Bilancio"}}(\text{COMMISSIONE})))$$

Calcolo dei Domini:

$$\{\text{Nome: fn, Cognome: sur, Provincia: prv} \mid \text{DEPUTATI(Codice: n, Cognome: sur, Nome: fn, Commissione: cm, Provincia:co,Collegio: cst)} \wedge \text{PROVINCE(Sigla: usl1 ,Nome: prv ,Regione: co)} \wedge \text{COMMISSIONE(Numero: cm, Nome: nameC, Presidente: usl2)} \wedge (\text{nameC= "Bilancio"}) \}$$

Calcolo delle tuple:

$$\{\text{D.(Nome,Cognome), P.(Nome),} \mid \text{D(DEPUTATI), C(COMMISSIONI), P(PROVINCE)} \mid (\text{D.Commissione=C.Numero}) \wedge (\text{C.Nome= "Bilancio"}) \wedge (\text{D.Provincia= P.Sigla}) \}$$

4.

Algebra Relazionale:

$$\Pi_{\text{NomeC,Cognome,nom1,rege}}(((\rho_{\text{Rege}\leftarrow\text{Nome}}(\text{REGIONE}))\triangleright\triangleleft_{\text{Codice=Regione}}(\rho_{\text{Nom1}\leftarrow\text{Nome}}(\text{PROVINCIA}))\triangleright\triangleleft_{\text{Sigla=Provincia}}((\rho_{\text{Nome1}\leftarrow\text{Nome}}(\text{DEPUTATI}))\triangleright\triangleleft_{\text{Commissione=Numero}}(\sigma_{\text{Nome="Bilancio"}}(\text{COMMISSIONE})))$$

Calcolo dei Domini:

$$\{\text{Nome: fn, Cognome: sur, Provincia: prv, Regione:rege} \mid \text{DEPUTATI(Codice: n, Cognome: sur, Nome: fn, Commissione: cm, Provincia:co,Collegio: cst)} \wedge \text{PROVINCE(Sigla: usl1 ,Nome: prv ,Regione: co)} \wedge \text{REGIONI(Codice:co, Nome: rege)} \wedge \text{COMMISSIONE(Numero: cm, Nome: nameC, Presidente: usl2)} \wedge (\text{nameC= “Bilancio”}) \}$$

Calcolo delle tuple:

$$\{\text{D.(Nome,Cognome), P.(Nome), R.(Nome)} \mid \text{D(DEPUTATI), C(COMMISSIONI), R(REGIONI), P(PROVINCE)} \mid (\text{D.Commissione=C.Numero}) \wedge (\text{C.Nome= “Bilancio”}) \wedge (\text{D.Provincia= P.Sigla}) \wedge (\text{P.Regione= R.Codice}) \}$$

5. Algebra Relazionale:

$$\Pi_{\text{RegioneC,Nome, Cognome}} (\text{DEPUTATI} \triangleright\!\triangleleft_{\text{Provincia=ProvinciaC} \wedge \text{Collegio=NumeroC}} (\text{REGIONI} \triangleright\!\triangleleft_{\text{Codice=RegioneC}} ((\rho_{\text{SiglaC,NomeC,RegioneC,NumeroC, Nome1C} \leftarrow \text{Sigla,Nome,Regione,Numero, Nome1}} (\text{PROVINCE} \triangleright\!\triangleleft_{\text{Sigla=Provincia}} (\rho_{\text{Nome1} \leftarrow \text{Nome}} \text{COLLEGI})))) - \Pi_{\text{SiglaC,NomeC,RegioneC,NumeroC, Nome1C}} ((\text{PROVINCE} \triangleright\!\triangleleft_{\text{Sigla=Provincia}} (\rho_{\text{Nome1} \leftarrow \text{Nome}} \text{COLLEGI})) \triangleright\!\triangleleft_{\text{Regione=RegioneC}} (\rho_{\text{SiglaC,NomeC,RegioneC,NumeroC, Nome1C} \leftarrow \text{Sigla,Nome,Regione,Numero, Nome1}} (\text{PROVINCE} \triangleright\!\triangleleft_{\text{Sigla=Provincia}} (\rho_{\text{Nome1} \leftarrow \text{Nome}} \text{COLLEGI})))))$$

Calcolo dei Domini:

$$\{\text{Nome: fn, Cognome: sur, Regione:rege} \mid \text{DEPUTATI(Codice: n, Cognome: sur, Nome: fn, Commissione: cm, Provincia:prov,Collegio: coll)} \wedge \text{PROVINCE(Sigla: prov,Nome: prv ,Regione: co)} \wedge \text{REGIONI(Codice:co, Nome: rege)} \wedge \text{PROVINCE(Sigla: n1 ,Nome: nom13 ,Regione: co)} \wedge \text{COLLEGI(Provincia: n1 ,Numero: coll,Nome: usl31)} \wedge \neg(\exists\, \text{n1}(\exists\, \text{co PROVINCE(Sigla: n1 ,Nome: nom1 ,Regione: co)} \wedge \text{COLLEGI(Provincia: n1 ,Numero: usl11 ,Nome: usl21)} \wedge \text{PROVINCE(Sigla: n21 ,Nome: nom2 ,Regione: co)} \wedge \text{COLLEGI(Provincia: n2 ,Numero: usl12 ,Nome: usl22)} \wedge \text{usl11} <> \text{usl12}\}$$

Calcolo delle tuple:

$$\{D.(Nome,Cognome),R.(Nome) \mid D(DEPUTATI), C(COLLEGI), R(REGIONI), C1(COLLEGI), P1(PROVINCE), C2(COLLEGI), P2(PROVINCE) \mid (D.Collegio=C.Numero) \neg(\exists\ P1(PROVINCE)\ ((P2.Sigla=C2.Provincia) \wedge (P1.Sigla=C1.Provincia) \wedge (P1.Regione=P2.Regione) \wedge (C1.nome \neq C2.nome)\ )\ \}$$

6. Algebra Relazionale:

$$(\Pi_{NomeColl1} (\sigma_{NomeD1=NomeD2} (\rho_{NomeD2,NomeColl2,Regione2\leftarrow ,NomeD,Nome,Regione} (\Pi_{Regione,NomeD,Provincia,CollegioNome}(\rho_{NomeD,ProvinciaD\leftarrow Nome,Provincia}\ DEPUTATI\ ) \rhd\lhd_{(Provincia=ProvinciaD \wedge Collegio=Numero)} (COLLEGI \rhd\lhd_{Provincia= Sigla} (\rho_{NomeP\leftarrow Nome}\ PROVINCE))))) \rhd\lhd_{Regione2=Regione1} (\rho_{NomeD1,NomeColl1,Regione1\leftarrow ,NomeD,Nome,Regione} (\Pi_{Regione,NomeD,Provincia,CollegioNome}(\rho_{NomeD,ProvinciaD\leftarrow Nome,Provincia}\ DEPUTATI\ ) \rhd\lhd_{(Provincia=ProvinciaD \wedge Collegio=Numero)} (COLLEGI \rhd\lhd_{Provincia= Sigla} (\rho_{NomeP\leftarrow Nome}\ PROVINCE))))))))$$

Calcolo dei Domini:

$$\{Nome: nomecoll \mid DEPUTATI(Codice: n1, Cognome: sur1, Nome: fn, Commissione: cm1, Provincia:n1,Collegio: coll1) \wedge PROVINCE(Sigla: n1,Nome: prv1 ,Regione: rege) \wedge COLLEGI(Provincia: n1 ,Numero: coll1 ,Nome: nomecoll1) \wedge DEPUTATI(Codice: n1, Cognome: sur2, Nome: fn, Commissione: cm2, Provincia:n2,Collegio: coll2) \wedge PROVINCE(Sigla: n2,Nome: prv2 ,Regione: rege) \wedge COLLEGI(Provincia: n2 ,Numero: coll2,Nome: Nomecoll2)\}$$

Calcolo delle tuple:

$$\{C1.(Nome) \mid D1(DEPUTATI),P1(PROVINCE), C1(COLLEGI), D2(DEPUTATI),P2(PROVINCE), C2(COLLEGI) \mid (C1.Provincia=P1.Sigla) \wedge (C2.Provincia=P2.Sigla) \wedge (D1.Provincia= C1.Provincia) \wedge (D1.Collegio= C1.Numero) \wedge (D2.Provincia= C2.Provincia) \wedge (D2.Collegio= C2.Numero) \wedge (P2.Regione=P1.Regione) \wedge (D1.Nome=D2.Nome)\ \}$$

Esercizio 3.9

Mostrare come le interrogazioni nell'esercizio 3.8 possano trarre vantaggio, nella specifica, dalla definizione di viste.

Soluzione:

I punti 2, 3 e 4 dell'esercizio 3.3 sono molto simili. Per rendere più veloci le interrogazioni è utile definire una vista con le parti in comune, inserendo in una tabella COMBILANCIO tutte le informazioni necessarie:

$$
\begin{aligned}
&\text{COMBILANCIO} = (\\
&(\rho_{\text{Rege}\leftarrow\text{Nome}}(\text{ REGIONE}))\rhd\lhd_{\text{Codice=Regione}}\\
&(\rho_{\text{Nom1}\leftarrow\text{Nome}}(\text{ PROVINCIA}))\rhd\lhd_{\text{Sigla=Provincia}}\\
&((\rho_{\text{Nome1}\leftarrow\text{Nome}}(\text{DEPUTATI}))\rhd\lhd_{\text{Commissione=Numero}}\\
&(\sigma_{\text{Nome=“Bilancio”}}(\text{ COMMISSIONE})))
\end{aligned}
$$

A questo punto le interrogazioni dell'esercizio 3.3 si risolvono più semplicemente:

3.2 $\Pi_{\text{NomeC,Cognome}}$ (COMBILANCIO)

3.3 $\Pi_{\text{NomeC,Cognome,nom1}}$ (COMBILANCIO)

3.4 $\Pi_{\text{NomeC,Cognome,nom1,rege}}$ (COMBILANCIO)

Esercizio 3.10

Si consideri lo schema di base di dati sulle relazioni:

MATERIE (Codice, Facoltà, Denominazione, Professore)
STUDENTI (Matricola, Cognome, Nome, Facoltà)
PROFESSORI (Matricola, Cognome, Nome)
ESAMI (Studente, Materia, Voto, Data)
PIANIDISTUDIO (Studente, Materia, Anno)

Formulare, in algebra relazionale, in calcolo su domini, in calcolo su tuple e in Datalog le interrogazioni che producono:

1. gli studenti che hanno riportato in almeno un esame una votazione pari a 30, mostrando, per ciascuno di essi, nome e cognome e data della prima di tali occasioni;
2. per ogni insegnamento della facoltà di ingegneria, gli studenti che hanno superato l'esame nell'ultima seduta svolta;
3. gli studenti che hanno superato tutti gli esami previsti dal rispettivo piano di studio;
4. per ogni insegnamento della facoltà di lettere, lo studente (o gli studenti) che hanno superato l'esame con il voto più alto;
5. gli studenti che hanno in piano di studio solo gli insegnamenti della propria facoltà;
6. nome e cognome degli studenti che hanno sostenuto almeno un esame con un professore che ha il loro stesso nome proprio.

Soluzione:

1)

Algebra Relazionale:

$$\Pi_{Nome,Cognome,\ Data}$$
$$((\Pi_{Studente,Data}\ (\sigma_{Voto=\ '30'}(ESAMI)) -$$
$$\Pi_{Studente,Data}((\sigma_{Voto=\ '30'}(ESAMI) \rhd\lhd_{(Studente=Studente1)\wedge(Data>Data1)}$$
$$\rho_{Studente1,Corso1,Voto1,Data1\leftarrow\ Studente,Corso,Voto,Data}\sigma_{Voto=\ '30'}(ESAMI)))$$
$$\rhd\lhd_{Studente=Matricola}(STUDENTE)\ )$$

Calcolo dei Domini:

$$\{Nome: fn, Cognome: sur, Data: d \mid$$
$$STUDENTE(Matricola: n, Nome: fn, Cognome: sur, Facoltà: f) \wedge$$
$$ESAMI(studente: n, Corso: c, Voto: g, Data: d\ ) \wedge (g= \text{`30'}) \wedge$$
$$\neg(\exists c1\ (\exists\ d1\ (ESAMI\ (Studente: n, Corso: c1, Voto: g, Data: d1) \wedge (d1<d)))) \}$$

Calcolo delle Tuple:

$$\{\ S.(Nome,Cognome), E.Data \mid S\ (STUDENTE), E(ESAMI) \mid$$
$$(S.matricola=E.Studente) \wedge (E.voto= \text{`30'}) \wedge$$
$$\neg(\exists\ E1(ESAMI)\ ((E1.Studente=S.matricola) \wedge(E1.voto= \text{`30'}) \wedge (E1.Data < E.Data))) \}$$

Datalog:

OTHERA (Studente: n, Data: d) ←
ESAMI(Studente: n, Corso: c, Voto: ‘30’, Data: d),
ESAMI(Studente: n, Corso: c1, Voto: ‘30’, Data: d1),
(d > d1)

FIRSTA (Nome: fn, Cognome: sur, Data: d) ←
STUDENTE(Matricola: n, Nome: fn, Cognome: sur, Facoltà: f),
ESAMI(Studente: n, Corso: c, Voto: ‘30’, Data: d),
not OTHERA(Studente: n, Data: d)

2)

Algebra Relazionale:

$\sigma_{(Voto\geq 18)\wedge(Facoltà='Ingegneria')}$
$(\Pi_{Cognome,Nome,\ Materia,Voto,Data}$
$(STUDENTI \triangleright\triangleleft_{Studente=Matricola} (\Pi_{Studente,Materia,\ Voto,\ Data}\ (\sigma_{(D\geq Data)}$
$(ESAMI \triangleright\triangleleft_{Materia=M}\ (\rho_{\ S,M,V,D\leftarrow Studente,Materia,\ Voto,\ Data}(ESAMI)\ ))))))$

Calcolo Domini:

{Nome: fn, Cognome: sur, Materia: c, Voto: g, Data:d |
STUDENTE(Matricola: sn, Nome: fn, Cognome: sur, Facoltà: f) ∧
ESAMI(studente: sn, Materia: c, Voto: g, Data: d) ∧
ESAMI(studente: sn, Materia: c, Voto: v1, Data: d1) ∧
(f='Ingegneria') ∧ (d ≥ d1) }

Calcolo delle Tuple:

{ S.(Nome,Cognome, Materia,Voto, Data) |
S (STUDENTI), E(ESAMI),E1(ESAMI) |
(E.Materia=E1.Materia) ∧ (E.Data> E1.Data) ∧ (E.Studente=S.Matricola) ∧
(Facoltà=“Ingegneria”) ∧ (E.Voto≥ 18) }

Datalog:

BRAVISTUDENTI(Nome: fn, Cognome: sur, Materia: c, Voto: g , Data: d) ←
STUDENTI(Matricola: sn, Nome: fn, Cognome: sur, Facoltà: sf),
ESAMI(Studente: sn, Materia: c, Voto: g, Data: d),
ESAMI(Studente: sn, Materia: c, Voto: g1, Data: d1),
(d ≥ d1),(sf = 'Ingegneria'), (g ≥ 18)

3)

Algebra Relazionale:

$\Pi_{Studente}(PIANIDISTUDIO)-$
$(\Pi_{Studente}(\Pi_{Studente,Materia}(PIANIDISTUDIO) -\Pi_{Studente,\ Materia}(ESAMI)))$

Calcolo dei Domini:

$\{$ Studente: n $|$ PIANIDISTUDIO (Studente:n, Materia :c , Anno: y) $\wedge$
$(\forall c1(\forall y1(\exists g1,\exists d1(\neg($ PIANIDISTUDIO (Studente: n, Materia: c1, Anno: y1) $\vee$
(ESAMI(Studente: n, Materia: c1, Voto: g1, Data: d1)))))) $\}$

Calcolo delle Tuple:

$\{$ S.Studente $|$ P(PIANIDISTUDIO) $|$ $(\forall$P(PIANIDISTUDIO) $(\exists$E(ESAMI)
$(\neg$(S.Studente=P.Studente) $\vee$ ((P.Studente=E.Studente)$\wedge$(S1.Materia=E.Materia)))) $\}$

Datalog:

ESAMISTUDENTE(Studente: n, Materia: c) $\leftarrow$
ESAMI(Studente: n, Materia: c, Voto: g, Data:d)

ESAMIINCOMPLETISTUDENTE(Studente: n) $\leftarrow$
PIANIDISTUDIO (Studente:n, Materia :c , Anno: y),
NOT ESAMISTUDENTE (Studente: n, Materia : c)

TUTTIESAMI(Studente: n) $\leftarrow$ PIANIDISTUDIO (Studente:n, Materia :c , Anno: y)
NOT ESAMIINCOMPLETISTUDENTE(Studente: n)

4)

Algebra Relazionale :

$\Pi_{Studente,Materia}(ESAMI \triangleright\!\triangleleft_{Materia=Codice}\ \sigma_{Facoltà=\text{“Lettere”}}\ (MATERIE)) -$
$\Pi_{Studente,Materia}\sigma_{(Voto< Voto1)\wedge(Facoltà=\text{“Lettere”})}(MATERIE \triangleright\!\triangleleft_{Codicer=\ Materia}\ ESAMI$
$\triangleright\!\triangleleft_{\ Materia=\ Materia1}\ \rho_{Studente1,\ Materia1\ ,\ Voto1,\ Data1\leftarrow\ Studente,\ Materia\ ,\ Voto,\ Data}(ESAMI)\ )$

Calcolo dei Domini:

$\{$ Studente: sn, Materia: c $|$ ESAMI(Studente: sn, Materia : c, Voto: g, Data: d) $\wedge$
MATERIE(Codice: c, Facoltà: f, Denominazione: ct, Professore: t) $\wedge$ (f= “Lettere”) $\wedge$
$\neg(\exists$sn1($\exists$d1 ($\exists$ g1(ESAMI(Studente: sn1, Materia: c, Date: d1, Voto: g1) $\wedge$ (g1> g)))))
}

Calcolo delle Tuple:

$\{$ E.Studente, E.Materia $|$ E(ESAMI), M(MATERIE) $|$
(E. Materia =M.Codice) $\wedge$ (M.Facoltà= “Lettere”) $\wedge$
(($\exists$E1(ESAMI) $\neg$(E1. Materia =E. Materia) $\wedge$ (E1.Voto > E.Voto)))

Datalog:

STUDENTINONMIGLIORI(Studente: sn, Materia: c) ←
ESAMI(Studente: sn, Materia : c, Voto: g, Data: d),
MATERIE(Number: c, Facoltà : "Lettere", Denominazione: ct, Professore: t),
ESAMI(Studente: sn1, Materia : c, Voto: g1, Data: d1),
(g < g1)

STUDENTIMIGLIORI(Studente: sn, Materia : c) ←
ESAMI(Studente: sn, Materia : c, Voto: g, Data: d),
MATERIE(Codice: c, Facoltà: " Lettere ", Denominazione : ct, Professore : t),
NOT STUDENTINONMIGLIORI(Studente: sn, Materia : c)

5)

Algebra Relazionale:

$\Pi_{Studente}(PIANIDISTUDIO)-$
$\Pi_{Studente}(\sigma_{SfFacoltà\neq Facoltà)}(\rho_{SMatricola,SFacoltà\leftarrow Matricolar,Facoltà}(STUDENTI)$
$\triangleright\triangleleft_{SMatricols=Studente}$ PIANIDISTUDIO
$\triangleright\triangleleft_{Materia=Codice}$ MATERIE))

Calcolo dei Domini:

{ Studente: n | PIANIDISTUDIO (Studente:n, Materia:c , Anno: y)∧
¬(∃fn(∃sur,∃sf(∃f,(∃ct,∃t(
STUDENTI(Matricola: n, Nome: fn, Cognome: sur, Facoltà: sf) ∧
MATERIE (Codice: c, Facoltà: f, Denominazione: ct, Professore: t)∧(sf≠f)))))) }

Calcolo delle Tuple:

{ SP.Studente | SP(PIANIDISTUDIO) | ¬(∃S(STUDENTI) (∃C(MATERIE)
((SP.Materia=C.Codice)∧(S.Matricola=SP.Studente)∧(C.Facoltà≠S.Facoltà)))) }

Datalog:

DIFFERENTEFACOLTA(Studente: n) ←
PIANIDISTUDIO (Studente:n, Materia:c , Anno: y),
STUDENTS(Matricola: n, Nome: fn, Cognome: sur, Facoltà: sf) ,
MATERIE (Codice: c, Facoltà: f, Denominazione: ct, Professore: t),
(sf≠f)

STESSAFACOLTA(Studente: n) ←
PIANIDISTUDIO (Studente:n, Materia :c , Anno: y),
NOT DIFFERENTEFACOLTA(Studente: n)

6)

Algebra Relazionale:

$\Pi_{Nome,Cognome}$ ($\sigma_{Nome=NomeP}$(
(ESAMI$\triangleright\triangleleft_{Studente=\ Matricola}$ STUDENTI)$\triangleright\triangleleft_{Materia=Codice}$
($\rho_{NomeP,CognomeP\ \leftarrow Nome,Cognome}$(PROFESSORI $\triangleright\triangleleft_{Matricola=Professore}$ MATERIE))))

Calcolo dei Domini:

{ Nome: fn, Cognome: sur |
STUDENTI(Matricola: n, Nome: fn, Cognome: sur, Facoltà: sf) ∧
ESAMI(Studente: n, Materia: c, Voto: g, Data: d) ∧
MATERIE(Codice: c, Facoltà: f, Denominazione: ct, Professore: t)∧
PROFESSORI(Matricola: t, Nome:fn, Cognome: tfn) }

Calcolo delle Tuple:

{ S.(Nome,Cognome) |
S(STUDENTI), E (ESAMI), M(MATERIE), P(PROFESSORI) |
(S.Matricola=E.Studente)∧(E.Materia=M.Codice) ∧ (M.Professore=P.Matricola) ∧
(P.Nome=S.Nome) }

Datalog:

STESSONOME (Nome: fn, Cognome: sur) ←
STUDENTI(Matricola: n, Nome: fn, Cognome: sur, Facoltà: sf) ,
ESAMI(Studente: n, Materia: c, Voto: g, Data: d) ,
MATERIE(Codice: c, Facoltà: f, Denominazione: ct, Professore: t),
PROFESSORI(Matricola: t, Cognome:surf, Nome: fn)

Esercizio 3.11

Con riferimento al seguente schema di base di dati:

CITTÀ (Nome, Regione, Abitanti)
ATTRAVERSAMENTI (Città, Fiume)
FIUMI (Fiume, Lunghezza)

formulare, in algebra relazionale,in calcolo sui domini, in calcolo su tuple e in Datalog le seguenti interrogazioni:

1. Visualizza nome, regione e abitanti per le città che hanno più di 50000 abitanti e sono attraversate dal Po oppure dall'Adige;
2. Trovare le città che sono attraversate da (almeno) due fiumi, visualizzando il nome della città e quello del più lungo di tali fiumi.

Soluzione:

1) Algebra Relazionale:

$\Pi_{\text{Nome, Regione, Abitanti}} (\sigma_{(\text{Fiume="Po"})\vee(\text{Fiume="Adige"})}(\text{ATTRAVERSAMENTO}) \rhd\lhd_{\text{Citta=Nome}} \sigma_{\text{Abitanti}>50000}(\text{CITTA}))$

Calcolo dei Domini:

{ Nome: n, Regione: reg, Abitanti: p | CITTA (Name: n, Regione : reg, Abitanti : p) ∧
ATTRAVERSAMENTO (Città: n, Fiume: r)∧ (p > 50000) ∧
((r= “Po”) ∨ (r= “Adige”)) }

Calcolo delle Tuple:

{ C.(Nome,Regione, Abitanti) | C(CITTA), A (ATTRAVERSAMENTO) |
(C.Nome= A.Città) ∧ (C. Abitanti > 50000) ∧
((A.Fiume= “Po”) ∨ (A.Fiume= “Adige”)) }

Datalog:

POEADIGE (Città: c) ←
ATTRAVERSAMENTO (Città: c, Fiume: r),
(r = “Po”)

2) Algebra Relazionale:

$\Pi_{\text{Città,Fiume}} (\sigma_{\text{Fiume} \neq \text{Fiume 1}} (\text{ATTRAVERSAMENTO} \rhd\lhd_{\text{Città=Città1}} \rho_{\text{Città1,Fiume 1}\leftarrow\text{Città, Fiume}} (\text{ATTRAVERSAMENTO}))) –$

$\Pi_{\text{Città,Fiume}} (\sigma_{(\text{Fiume}\neq\text{Fiume1})\wedge(\text{Lughezza} < \text{Lunghezza1})}((\text{FIUMI} \rhd\lhd \text{ATTRAVERSAMENTO}) \rhd\lhd_{\text{Città=Città1}} \rho_{\text{Città1,Fiume1, Lunghezza1}\leftarrow \text{Città,Fiume, Lunghezza}}(\text{FIUMI} \rhd\lhd \text{ATTRAVERSAMENTO})))$

Calcolo dei Domini:

$$\{ \text{Città: n, Fiume: r} \mid \text{ATTRAVERSAMENTO (Città: n, Fiume: r)} \wedge \text{ATTRAVERSAMENTO (Città: n, Fiume: r1)} \wedge (r \neq r1) \wedge \text{FIUMI(Fiume: r, Lunghezza: l)} \wedge (\forall r2(\forall l2 \neg((\text{ATTRAVERSAMENTO (Città: n, Fiume: r2)} \wedge \text{FIUMI (Fiume: r2, Lunghezza: l2)}) \vee (l2 < l)))) \}$$

Calcolo delle Tuple:

$$\{C.(\text{Città}, \text{Fiume}) \mid C(\text{ATTRAVERSAMENTO}), C1(\text{ATTRAVERSAMENTO}), F(\text{FIUMI}) \mid (C.\text{Città} = C1.\text{Città}) \wedge (C.\text{Fiume} \neq C1.\text{Fiume}) \wedge (C.\text{Fiume} = F.\text{Fiume}) \wedge (\forall C2(\text{ATTRAVERSAMENTO})(\forall F2(\text{FIUMI})(C2.\text{Città} \neq C.\text{Città}) \vee (F2.\text{Fiume} = F.\text{Fiume}) \wedge (F2.\text{Lunghezza} < F.\text{Lunghezza}))))\}$$

Datalog:

FIUMECORTO (Città : c, Fiume : r) ←
ATTRAVERSAMENTO (Città : c, Fiume : r),
FIUMI (Fiume : r, Lunghezza : l),
ATTRAVERSAMENTO (Città : c, Fiume : r1),
FIUMI (Fiume : r1, Lunghezza : l1),
(l < l1)

FIUMELUNGO (Città : c, Fiume : r) ←
ATTRAVERSAMENTO (Città : c, Fiume : r),
ATTRAVERSAMENTO (Città : c, Fiume : r1),
(r ≠r1),
NOT FIUMECORTO (Città : c, Fiume : r)

Esercizio 3.12

Con riferimento al seguente schema di base di dati:

AFFLUENZA (Affluente, Fiume)

FIUMI (Fiume, Lunghezza)

formulare l'interrogazione in Datalog che trova tutti gli affluenti, diretti e indiretti dell'Adige.

Soluzione:

TUTTIAFFLUENTI (Affluenti: t) ←
AFFLUENZA (Affluenti : t, Fiume: “Adige”)

TUTTIAFFLUENTI (Affluenti : t) ←
AFFLUENZA (Affluenti : t, Fiume:r),
TUTTIAFFLUENTI (Affluenti : r)

Esercizio 3.13

Si consideri lo schema relazionale composto dalle seguenti relazioni:

PROFESSORI (Codice, Cognome, Nome)
CORSI (Codice, Denominazione, Professore)
STUDENTI (Matricola, Cognome, Nome)
ESAMI (Studente, Corso, Data, Voto)

Formulare, con riferimento a tale schema, le espressioni dell'algebra, del calcolo relazionale su tuple e del Datalog, che producano:

1. Gli esami superati dallo studente Pico della Mirandola (supposto unico), con indicazione, per ciascuno, della denominazione del corso, del voto e del cognome del professore;
2. i professori che tengono due corsi (e non più di due), con indicazione di cognome e nome del professore e denominazione dei due corsi.

Soluzione:

1) Algebra Relazionale:

$$\Pi_{\text{Denominazione,Voto,CognomeP,NomeP}}(\sigma_{\text{Cognome= "della Mirandola"} \wedge \text{ Nome= "Pico"}} (\text{STUDENTI} \rhd\lhd_{\text{Matricola=Studente}} \text{ESAMI} \rhd\lhd_{\text{Corso=CCodice}} \rho_{\text{CCodice} \leftarrow \text{Codice}}(\text{CORSI}) \rhd\lhd_{\text{Professore=TCodice}} \rho_{\text{TCodice,T CognomeP,NomeP} \leftarrow \text{Codice, Cognome,Nome}} (\text{PROFESSORI})))$$

Calcolo delle Tuple:

$$\{ \text{C.Denominazione, E.Voto, P.(Cognome,Nome)} \mid \text{C(CORSI), E(ESAMI), P(PROFESSORI), S(STUDENTI)} \mid (\text{S.Matricola=E.Studente}) \wedge (\text{E.Corso=C.Codice}) \wedge (\text{C.Professore= P.Codice}) \wedge (\text{S.Cognome= "Della Mirandola"}) \wedge (\text{S.Nome= "Pico"})$$

Datalog:

```
ESAMIPICO( Denominazione: cn, Voto: g, Cognome: tsur, Nome: tname ) ←
    STUDENTI(Matricola: n, Cognome: "Della Mirandola", Nome: "Pico"),
    ESAMI ( Studente: n, Corso: c, Voto: g, Data: d),
    CORSI  (Codice: c, Denominazione:cn, Professore: t),
    PROFESSORI (Codice: t, Cognome: tsur, Nome: tname )
```

2) Algebra Relazionale:

$$\Pi_{\text{Cognome,Nome,Denominazione, Denominazione1}} (\Pi_{\text{Professore,Denominazione,Denominazione1}} (\text{CORSI} \rhd\lhd_{(\text{,Professore=,Professore1}) \wedge (\text{Codice} \neq (\text{Codice})} \rho_{\text{Codice1,Denominazione1,Professore1} \leftarrow \text{ Codice,Denominazione, Professore}} (\text{CORSI})) - \Pi_{\text{Professore,Denominazione, Denominazione1}} (\sigma_{\text{Codice} \neq \text{Codice2}} (\text{CORSI} \rhd\lhd_{(\text{Professore=Professore1}) \wedge (\text{Codice} \neq \text{Codice1})} \rho_{\text{Codice1, Denominazione1 ,Professore1} \leftarrow \text{ Codice, Denominazione , Professore}} (\text{CORSI}) \rhd\lhd_{(\text{Professore1=Professore2}) \wedge (\text{Codice 1} \neq \text{ Codice 2}) \wedge (\text{Codice} \neq \text{ Codice 2})} \rho_{\text{ Codice 2,Denominazione2,Professore2} \leftarrow \text{ Codice,Denominazione, Professore}} (\text{CORSI}))) \rhd\lhd_{\text{Professore=PCodice}} \rho_{\text{PCodice} \leftarrow \text{Codice}} (\text{PROFESSORI}))$$

Calcolo delle tuple:

{ P.(Nome,Cognome), C.Denominazione, C1. Denominazione |
P(PROFESSORI), C (CORSI), C1(CORSI) |
(P.Codice)=(C. Professore) ∧ (P. Codice)=(C1. Professore) ∧
(C. Codice . Codice) ∧
¬(∃ C2(CORSI) ((C2.Professore=P. Codice) ∧
(C2. Codice ≠ C. Codice) ∧(C2. Codice ≠ C1. Codice)∧(C. Codice ≠ C1. Codice)) }

Datalog:

PIUDIDUE (Professore : t) ←
CORSI (Codice : n, Denominazione : cn, Professore : t),
CORSI (Codice : n1, Denominazione : cn1, Professore : t),
CORSI (Codice : n1, Denominazione : cn1, Professore : t),
(n≠n1),(n≠n2),(n1≠n2)

ESATTAMENTEDUE (Nome : fn, Cognome: sn, Corso1 : cn1, Corso2 : cn2) ←
PROFESSORI (Codice : t, Nome: fn, Cognome: sn),
CORSI (Codice : n1, Denominazione : cn1, Professore : t),
CORSI (Codice : n2, Denominazione : cn2, Professore : t),
(n1≠n2),
NOT PIUDIDUE(Professore : t)

Esercizio 3.14

Considerare uno schema relazionale contenente le relazioni

R_1 (ABC), R_2 (DG), R_3(EF)

Formulare in calcolo relazionale su tuple e su domini l'interrogazione realizzata in algebra relazionale dalla seguente espressione:

$(R_3 \rhd\lhd_{G=E} R_2) \cup (\rho_{DG \leftarrow AC} (\Pi_{ACEF} (R_1 \rhd\lhd_{B=F} R_3)))$

Soluzione:

Questa espressione non è esprimibile in calcolo sulle tuple a causa dell'unione tra due diverse tabelle.
In calcolo sui domini l'espressione diventa:

$\{ D: d, G: g, E: e, F: f \mid R_3(E:e, F:f) \wedge ((R_2(D: d, G: g) \wedge (g=e)) \vee (R_1(A: d, B: b, C: g) \wedge (b=f))) \}$

Esercizio 3.15

Con riferimento allo schema dell'esercizio 3.14, formulare in algebra relazionale le interrogazioni realizzate in calcolo su domini dalle seguenti espressioni:

$\{ H: g, B: b \mid R_1(A: a, B: b, C: c) \wedge R_2(D: c, G: g) \}$
$\{ A: a, B: b \mid R_2(D:a , G: b) \wedge R_3 (E: a, F: b) \}$
$\{ A: a, B: b \mid R_1(A:a , B: b, C: c) \wedge \exists a' (R_1(A: a' , B: b, C: c) \wedge a \neq a') \}$
$\{ A: a, B: b \mid R_1(A:a , B: b, C: c) \wedge \forall a' (\neg R_1(A: a' , B: b, C: c) \vee a = a') \}$
$\{ A: a, B: b \mid R_1(A:a , B: b, C: c) \wedge \neg \exists a' (R_1(A: a' , B: b, C: c) \wedge a \neq a') \}$

Soluzione:

$\rho_{H \leftarrow G} (\Pi_{BG} (R_1 \rhd\lhd_{C=D} R_2))$
$\rho_{AB \leftarrow DG} (\Pi_{DG} (R_2 \rhd\lhd_{(D=E) \wedge (G=F)} R_3))$
$\Pi_{AB}(\sigma_{A \neq A1} (R_1 \rhd\lhd_{(B=B1) \wedge (C=C1)} \rho_{A1,B1,C1 \leftarrow ABC}(R_1)))$
$\Pi_{AB} (R_1) - \Pi_{AB}(\sigma_{A \neq A1} (R_1 \rhd\lhd_{(B=B1) \wedge (C=C1)} \rho_{A1,B1,C1 \leftarrow ABC}(R_1)))$
$\Pi_{AB} (R_1) - \Pi_{AB}(\sigma_{A \neq A1} (R_1 \rhd\lhd_{(B=B1) \wedge (C=C1)} \rho_{A1,B1,C1 \leftarrow ABC}(R_1)))$

Esercizio 3.16

Trasformare la seguente espressione dall'algebra, che fa riferimento allo schema

$$R_1(AB),\ R_2\ (CDE),\ R_3(FGH)$$

con l'obiettivo di ridurre le dimensioni dei risultati intermedi:

$\Pi_{ADH}\ (\ \sigma_{(B=C)\wedge(E=F)\wedge(A>20)\wedge(G=10)}\ ((\ R_1 \rhd\lhd R_3)\ \rhd\lhd\ R_2\ )$

Soluzione:

$\Pi_{ADH}\ (\ \sigma_{A>20}(R_1) \rhd\lhd_{B=C}\ \Pi_{CDH}\ (R_2 \rhd\lhd_{E=F}\ \Pi_{FH}\ (\sigma_{G=10}(R_3)))\ )$

Esercizio 3.17 Considerare la seguente base di dati relazionale:

- FARMACI (Codice, NomeFarmaco, PrincipioAttivo, Produttore, Prezzo)
- PRODUTTORI (CodProduttore, Nome, Nazione)
- SOSTANZE (ID, NomeSostanza, Categoria)

con vincoli di integrità referenziale fra Produttore e la relazione PRODUTTORI, fra PrincipioAttivo e la relazione SOSTANZE.
Formulare in algebra relazionale le seguenti interrogazioni:

1. l'interrogazione che fornisce, per i farmaci il cui principio attivo è nella categoria "sulfamidico," il nome del farmaco e quello del suo produttore;
2. l'interrogazione che fornisce, per i farmaci con produttore italiano, il nome del farmaco e quello della sostanza del suo principio attivo.

Soluzione

1. A = FARMACI $\bowtie_{PrincipioAttivo=ID}(\sigma_{Categoria='sulfamidico'}$ SOSTANZE);

   $\pi_{NomeFarmaco,Nome}$(PRODUTTORI $\bowtie_{CodProduttore=Produttore}$ A).

2. B = FARMACI $\bowtie_{Produttore=CodProduttore}(\sigma_{Nazione='Italia'}$ PRODUTTORI);

   $\pi_{NomeFarmaco,NomeSostanza}$(SOSTANZE $\bowtie_{ID=PrincipioAttivo}$ B).

Esercizio 3.18 Mostrare, con riferimento alla base di dati dell'esercizio 2.10, il risultato finale e quelli intermedi della seguente espressione dell'algebra relazionale:
$\pi_{Cognome}$(IMPIEG $\bowtie_{Matr=Imp}$ (PARTECIP $\bowtie_{Prog=Cod}$ $\sigma_{Costo>65}$(PROG)))
Soluzione

$\sigma_{Costo>65}$(PROG)

ID	Titolo	Costo
A	Luna	70
C	Giove	90

PARTECIP $\bowtie_{Prog=Cod}$ $\sigma_{Costo>65}$(PROG)

Imp	Prog	ID	Titolo	Costo
101	A	A	Luna	70
103	A	A	Luna	70

IMPIEG $\bowtie_{Matr=Imp}$ (PARTECIP $\bowtie_{Prog=Cod}$ $\sigma_{Costo>65}$(PROG))

Matr	Cognome	Nome	Età	Imp	Prog	ID	Titolo	Costo
101	Rossi	Mario	35	101	A	A	Luna	70
103	Gialli	Mario	34	103	A	A	Luna	70

$\pi_{Cognome}$(IMPIEG $\bowtie_{Matr=Imp}$ (PARTECIP $\bowtie_{Prog=Cod}$ $\sigma_{Costo>65}$(PROG)))

Cognome
Rossi
Gialli

Esercizio 3.19 Considerare la seguente base di dati relazionale:

- CLIENTI (Codice, Nome, Indirizzo, Città)
- NOLEGGI (Cliente, Auto, DataPrelievo, DataRestituzione) con vincolo di integrità referenziale fra lattributo Auto e la relazione AUTOVETTURE
- AUTOVETTURE (Targa, Modello, Colore, AnnoImmatricolazione, CostoGiornaliero) con vincolo di integrità referenziale fra lattributo Cliente e la relazione CLIENTI

formulare in algebra relazionale:

1. l'interrogazione che restituisce i dati dei clienti che hanno noleggiato almeno un'autovettura nell'anno 2006
2. l'interrogazione che restituisce i clienti che hanno noleggiato più di un'autovettura
3. l'interrogazione che restituisce i clienti che hanno noleggiato autovetture di un solo modello.

Soluzione

1. A = $\sigma_{(DataPrelievo \geq' 01/01/2006') \wedge (DataPrelievo \leq' 31/12/2006')}$NOLEGGI;

   CLIENTI $\bowtie_{Codice=Cliente}$ A

2. $\pi_{Cliente}$ ($\sigma_{Auto \neq Auto'}$ (NOLEGGI $\bowtie_{Cliente=Cliente'}$ ($\rho_{X' \leftarrow X}$ NOLEGGI)))

3. V = NOLEGGI $\bowtie_{Auto=Targa}$ AUTOVETTURE

   V_1 = V $\bowtie_{Cliente=Cliente'}$ ($\rho_{X' \leftarrow X}$ V)

   $\pi_{Cliente}$NOLEGGI - $\pi_{Cliente}$($\sigma_{Modello \neq Modello'}$($V_1$)

Esercizio 3.20 Considerando la seguente base di dati:

- FORNITORI (CF, Nome, Indirizzo, Città)
- PRODOTTI (CP, Nome, Marca, Modello)
- CATALOGO (CF, CP, Costo) con vincoli di integrità referenziale fra l'attributo CF e la relazione FORNITORI e fra l'attributo CP e la relazione PRODOTTI

formulare in algebra relazionale le seguenti interrogazioni:

1. trovare Nome, Marca e Modello dei prodotti acquistabili con meno di 2.000;
2. trovare i nomi dei fornitori che distribuiscono prodotti IBM (IBM è la marca di un prodotto);
3. trovare i codici di tutti i prodotti che sono forniti da almeno due fornitori;
4. trovare i nomi dei fornitori che distribuiscono tutti i prodotti presenti nel catalogo;
5. trovare i nomi dei fornitori che forniscono tutti i prodotti IBM presenti nel catalogo.

Soluzione

1. $\pi_{Nome,Marca,Modello}$ ($\sigma_{Costo<2.000}$PRODOTTI $\bowtie$ CATALOGO)

2. R = (FORNITORI $\bowtie$ CATALOGO) $\bowtie$ ($\pi_{CP,Marca}$PRODOTTI)

   $\pi_{Nome}(\sigma_{Marca='IBM'}$(R))

3. COPIACATALOGO = $\rho_{CF'\leftarrow CF}(\pi_{CP,CF}$CATALOGO)

   $\pi_{CP}(\sigma_{CF>CF'}$(CATALOGO $\bowtie$ COPIACATALOGO))

4. Formuliamo innanzitutto l'interrogazione R che trova i fornitori a cui manca almeno un prodotto del catalogo:

   R = π_{CF}((π_{CF}(FORNITORI) $\bowtie$ π_{CP}(CATALOGO)) - $\pi_{CF,CP}$(CATALOGO))

   π_{Nome}((π_{CF}(FORNITORI) - R) $\bowtie$ FORNITORI)

5. La soluzione di questa interrogazione è uguale a quella del punto precedente usando al posto della relazione CATALOGO la relazione CATALOGO IBM definita come segue:

   CATALOGOIBM = $\pi_{CP,CF}(\sigma_{Marca='IBM'}$(CATALOGO $\bowtie$ PRODOTTI))

Esercizio 3.21 Data la seguente istanza di basi di dati, mostrare i risultati intermedi e quelli finali delle interrogazioni definite nell'esercizio 3.19.

FORNITORI			
Nome	CodiceFornitore	Indirizzo	Città
Ladroni	001	Via Ostense	Roma
Risparmietti	002	Viale Marconi	Roma
Teloporto	010	Via Roma	Milano

CATALOGO		
CodiceFornitore	CodiceProdotto	Costo
001	0002	3.200
001	0003	2.200
002	0001	1.900
002	0002	2.500
002	0003	1.800
010	0001	2.200
010	0003	2.000

Prodotti			
CodiceProdotto	Nome	Marca	Modello
0001	Notebook	IBM	390
0002	Desktop	IBM	510
0003	Desktop	ACER	730

Soluzione

1. Trovare Nome, Marca e Modello dei prodotti acquistabili con meno di 2.000.

   Passo 1: PRODOTTI $\bowtie$ CATALOGO

CP	Nome	Marca	Modello	CF	Costo
0001	Notebook	IBM	390	002	1.900
0001	Notebook	IBM	390	010	2.200
0002	Desktop	IBM	510	002	2.500
0002	Desktop	IBM	510	001	3.200
0003	Desktop	ACER	730	001	2.200
0003	Desktop	ACER	730	010	2.000
0003	Desktop	ACER	730	002	1.800

   Passo 2: $\sigma_{Costo<2.000}$ (PRODOTTI $\bowtie$ CATALOGO)

CP	Nome	Marca	Modello	CF	Costo
0001	Notebook	IBM	390	002	1.900
0003	Desktop	ACER	730	002	1.800

   Passo 3: $\pi_{Nome,Marca,Modello}$ ($\sigma_{Costo<2.000}$ (PRODOTTI $\bowtie$ CATALOGO))

Nome	Marca	Modello
Notebook	IBM	390
Desktop	ACER	730

2. Trovare i nomi dei fornitori che distribuiscono prodotti IBM (IBM è la marca di un prodotto).

   Passo1: FORNITORI $\bowtie$ CATALOGO

Nome	Indirizzo	Città	CF	CP	Costo
Ladroni	Via Ostense	Roma	001	0003	2.200
Ladroni	Via Ostense	Roma	001	0002	3.200
Risparmietti	Viale Marconi	Roma	002	0001	1.900
Risparmietti	Viale Marconi	Roma	002	0002	2.500
Risparmietti	Viale Marconi	Roma	002	0003	1.800
Teloporto	Via Roma	Milano	010	0001	2.200
Teloporto	Via Roma	Milano	010	0003	2.000

   Passo 2: R = (FORNITORI $\bowtie$ CATALOGO) $\bowtie$ ($\pi_{CP,Marca}$PRODOTTI)

Nome	Indirizzo	Città	CF	Costo	CP	Marca
Ladroni	Via Ostense	Roma	001	2.200	0003	ACER
Risparmietti	Viale Marconi	Roma	002	1.900	0001	IBM
Risparmietti	Viale Marconi	Roma	002	2.500	0002	IBM
Teloporto	Via Roma	Milano	010	2.200	0001	IBM
Ladroni	Via Ostense	Roma	001	3.200	0002	IBM
Teloporto	Via Roma	Milano	010	2.000	0003	ACER
Risparmietti	Viale Marconi	Roma	002	1.800	0003	ACER

   Passo 3: $\sigma_{Marca='IBM'}$(R)

Nome	Indirizzo	Città	CF	Costo	CP	Marca
Risparmietti	Viale Marconi	Roma	002	1.900	0001	IBM
Teloporto	Via Roma	Milano	010	2.200	0001	IBM
Risparmietti	Viale Marconi	Roma	002	2.500	0002	IBM
Ladroni	Via Ostense	Roma	001	3.200	0002	IBM

   Passo 4: $\pi_{Nome}(\sigma_{Marca='IBM'}$(R)

Nome
Ladroni
Risparmietti
Teloporto

3. Trovare i nomi dei fornitori che distribuiscono prodotti IBM (IBM è la marca di un prodotto).

   Passo 1: COPIACATALOGO = $\rho_{CF'\leftarrow CF}(\pi_{CP,CF}$CATALOGO)

CF'	CP
001	0002
001	0003
002	0001
002	0002
002	0003
010	0001
010	0003

Passo 2: CATALOGO $\bowtie$ COPIACATALOGO

CF'	CP	CF
002	0001	002
010	0001	002
002	0001	010
010	0001	010
002	0002	002
001	0002	002
002	0002	001
001	0002	001
001	0003	001
010	0003	001
002	0003	001
001	0003	010
010	0003	010
002	0003	010
001	0003	002
010	0003	002
002	0003	002

Passo 3: $\sigma_{CF>CF'}$(CATALOGO $\bowtie$ COPIACATALOGO)

CF'	CP	CF
001	0002	002
002	0001	010
001	0003	010
002	0003	010
001	0003	002

Passo 4: $\pi_{CP}(\sigma_{CF>CF'}$(CATALOGO $\bowtie$ COPIACATALOGO)

CodiceProdotto
0001
0002
0003

4. Trovare i codici di tutti i prodotti che sono forniti da almeno due fornitori.

Passo 1: π_{CF}(FORNITORI) $\bowtie$ π_{CP}(CATALOGO)
Prodotto cartesiano tra CF di FORNITORI e CP di CATALOGO.

CF	CP
001	0001
001	0002
001	0003
002	0001
002	0002
002	0003
010	0001
010	0002
010	0003

Passo 2: (π_{CF}(FORNITORI) $\bowtie$ π_{CP}(CATALOGO)) - $\pi_{CF,CP}$(CATALOGO)
Attraverso questa interrogazione si ottiene una relazione che contiene i CF di Fornitori associati ai CP dei prodotti che non hanno nel catalogo, ossia i CF dei fornitori che non hanno almeno uno dei prodotti nel catalogo.

CF	CP
001	0001
001	0002
001	0003
002	0001
002	0002
002	0003
010	0001
010	0002
010	0003

-

CF	CP
001	0003
002	0001
002	0002
010	0001
001	0002
010	0003
002	0003

=

CF	CP
001	0001
010	0002

Passo 3:
R = π_{CF}(π_{CF}(FORNITORI) $\bowtie$ π_{CP}(CATALOGO)) - $\pi_{CF,CP}$(CATALOGO)

CF
001
010

Passo 4 (π_{CF}(FORNITORI) - R)

CF
001

Passo 5: π_{Nome}((π_{CF}(FORNITORI) - R) $\bowtie$ FORNITORI)

Nome
Risparmietti

5. Trovare i nomi dei fornitori che forniscono tutti i prodotti IBM presenti nel

catalogo.

Passo 1: CATALOGOIBM = $\pi_{CP,CF}(\sigma_{Marca='IBM'}(\text{CATALOGO} \bowtie \text{PRODOTTI}))$

CP	CF
0001	002
0001	010
0002	002
0002	001

Passo 2:

Nome
Risparmietti

Capitolo 4

Esercizio 4.1

Ordinare i seguenti domini in base al valore massimo rappresentable, supponendo che `integer` abbia una rappresentazione a 32 bit e `smallint` a 16 bit: `numeric(12,4)`, `decimal(10)`, `decimal(9)`, `integer`, `smallint`, `decimal(6,1)`.

Soluzione:

Dominio	Valore Massimo
1. `decimal(10)`	9999999999
2. `integer`	4294967296
3. `decimal(9)`	999999999
4. `numeric(12,4)`	99999999.9999
5. `decimal(6,1)`	99999.9
6. `smallint`	65536

Esercizio 4.2

Definire un attributo che permetta di rappresentare stringhe di lunghezza massima pari a 256 caratteri, su cui non sono ammessi valori nulli e con valore di default “sconosciuto”.

Soluzione:

```
Create domain STRING as character varying (256) default 'sconosciuto'
not null
```

Esercizio 4.3

Dare le definizioni SQL delle tre tabelle

```
FONDISTA(Nome, Nazione, Età)
GAREGGIA(NomeFondista, NomeGara, Piazzamento)
GARA(Nome, Luogo, Nazione, Lunghezza)
```

rappresentando in particolare i vincoli di foreign key della tabella `GAREGGIA`.

Soluzione:

```
Create Table FONDISTA
(
Nome     character(20)      primary key,
Nazione  character(30),
Età      smallint
)

Create table GARA
(
Nome     character(20)     primary key,
Luogo    character(20),
Nazione  character(20),
Lunghezza         integer
)

Create table GAREGGIA
(
NomeFondista      character(20)   references FONDISTA(Nome),
NomeGara          character(20),
Piazzamento       smallint,
primary key (NomeFondista, NomeGara),
foreign key (NomeGara)    references GARA(Nome)
)
```

Esercizio 4.4

Dare le definizioni SQL delle tabelle

```
AUTORE (Nome, Cognome, DataNascita, Nazionalità)
LIBRO (TitoloLibro, NomeAutore, CognomeAutore, Lingua)
```

Per il vincolo foreign key specificare una politica di `cascade` sulla cancellazione e di `set null` sulle modifiche.

Soluzione:

```
Create table AUTORE
(
Nome              character(20),
Cognome           character(20),
DataNascita       date,
Nazionalità       character(20),
primary key(Nome, Cognome)
)

Create table LIBRO
(
TitoloLibro       character(30)    primary key,
NomeAutore        character(20),
CognomeAutore     character(20),
Lingua            character(20),
foreign key (NomeAutore, CognomeAutore)
                  references AUTORE(Nome, Cognome)
                  on delete cascade
                  on update set NULL
)
```

Esercizio 4.5

Dato lo schema dell'esercizio precedente, spiegare cosa può capitare con l'esecuzione dei seguenti comandi di aggiornamento:

```
delete from AUTORE where Cognome = 'Rossi'

update LIBRO set NomeAutore= 'Umberto'
          where CognomeAutore = 'Eco'

insert into AUTORE(Nome,Cognome) values('Antonio','Bianchi')

update AUTORE set Nome = 'Italo' where Cognome = 'Calvino'
```

Soluzione:

1. Il comando cancella dalla tabella AUTORE tutte le tuple con Cognome = 'Rossi'. A causa della politica `cascade` anche le tuple di LIBRO con CognomeAutore = 'Rossi' verranno eliminate.
2. Il comando non è corretto: Nome e Cognome sono attributi della tabella AUTORE e non della tabella LIBRO
3. Il comando aggiunge una nuova tupla alla tabella AUTORE. Non ha alcun effetto sulla tabella LIBRO
4. Le tuple di AUTORE con Cognome = Calvino vengono aggiornate a Nome = Italo. A causa della politica `set null` gli attributi NomeAutore e CognomeAutore delle tuple di Libro con CognomeAutore = Calvino vengono posti a `NULL`.

Esercizio 4.6

Date le definizioni:

```
create domain Dominio as integer default 10
create table Tabella (Attributo Dominio default 5)
```

indicare cosa avviene in seguito ai comandi:

1. `alter table Tabella alter column Attributo drop default`
2. `alter domain Dominio drop default`
3. `drop domain Dominio`

Soluzione:

1. Il comando cancella il valore di default di Attributo. Il valore di default dopo il comando sarà quello impostato in Dominio, ossia 10.
2. Il comando cancella il valore di default di Dominio. Dopo l'operazione il valore di default sarà `NULL`
3. Il comando cancella l'intero dominio Domain. In Tabella il dominio di Attributo diventerà `integer`

Esercizio 4.7 Con riferimento ad una relazione PROFESSORI(CF, Nome, Età, Qualifica), scrivere le interrogazioni SQL che calcolano l'età media dei professori di ciascuna qualifica, nei due casi seguenti:

1. se l'età non è nota si usa per essa il valore nullo
2. se l'età non è nota si usa per essa il valore `0`.

Soluzione

1. Le funzioni aggregative escludono dalla valutazione le ennuple con valori nulli:

```
select Qualifica, avg(Eta) as EtaMedia
from Professori
group by Qualifica
```

2. È necessario escludere esplicitamente dal calcolo della media le ennuple con il valore che denota l'informazione incompleta:

```
select Qualifica, avg(Eta) as EtaMedia
from Professori
where Eta <> 0
group by Qualifica.
```

Esercizio 4.8 Spiegare perché in SQL è previsto (e necessario) un operatore di unione mentre in molte versioni non esistono gli operatori di intersezione e differenza.

Soluzione

In SQL è possibile esprimere l'intersezione attraverso dei join e la differenza attraverso l'uso dell'espressione "NOT IN" mentre non c'è modo di esprimere l'unione.

Esercizio 4.9 Considerare le relazioni IMPIEGATI (Matricola, Nome, Stipendio, Direttore) e DIPARTIMENTI (Codice, Direttore) e le due interrogazioni seguenti, specificare se e in quali casi esse possono produrre risultati diversi:

```
select avg(Stipendio)
from Impiegati
where Direttore in (select Direttore
                    from   Dipartimenti)

select avg(Stipendio)
from Impiegati I, Dipartimenti D
where I.Direttore = D.Direttore .
```

Soluzione
Le due interrogazioni dovrebbero calcolare lo stipendio medio di un direttore di dipartimento.
Nel caso in cui esistono impiegati direttori di più dipartimenti la seconda query produce un risultato scorretto in quanto conta più occorrenze di questi impiegati nel computo della media.
La prima query produce il risultato corretto.

Esercizio 4.10 Si consideri una base di dati sulle relazioni:

- $R_1(\underline{A}, B, C)$
- $R_2(\underline{D}, \underline{E}, F)$.

Facendo riferimento ad una versione dell'SQL che non prevede la differenza (parole chiave `EXCEPT` e `MINUS`) e che permette l'uso dei confronti nella nidificazione solo su singoli attributi (e quindi non ammette condizioni del tipo ... `(A,B) IN SELECT C,D FROM` ...), scrivere interrogazioni in SQL equivalenti alle seguenti espressioni dell'algebra relazionale:

1. $\pi_{BC}(\sigma_{C>10}(R_1))$
2. $\pi_{B}(R_1 \bowtie_{C=D} \sigma_{F=2}(R_2))$
3. $\pi_{AB}(R_1) - \pi_{AB}(R_1 \bowtie_{C=D} R_2))$.

Soluzione

1. ```
   select distinct B, C
   from R1
   where  C > 10;
   ```

2. ```
   select distinct B
   from R1 join R2
        on (C = D)
   where  F = 2;
   ```

3. ```
   select A, B
   from R1 left join R2
           on (C = D)
   where  C <> D.
   ```
   ```

Esercizio 4.11 Con riferimento alla base di dati nell'esercizio 4.10 scrivere espressioni dell'algebra relazionale equivalenti alle seguenti interrogazioni SQL:

1. ```
   select distinct A, B
   from R1, R2
   where C = D and E > 100;
   ```

2. ```
   select distinct A, B
   from R1 X1
   where not exixts(
       select *
       from R1 Y1, R2
       where Y1.C = D and X1.A = Y1.A and F>10).
   ```

Soluzione

1. $\pi_{AB}(R_1 \rtimes_{C=D} (\sigma_{F>2}(R_2)))$
2. $\pi_{AB}(R_1 - (R_1 \rtimes_{C=D} (\sigma_{F>10}(R_2)))$.

Esercizio 4.12 Con riferimento alla base di dati nell'esercizio 4.10, indicare, per ciascuna delle seguenti interrogazioni, se la parola chiave `DISTINCT` è necessaria:

1. l'interrogazione 1 nell'esercizio 4.11
2. l'interrogazione 2 nell'esercizio 4.11
3. ```
   select distinct A, B
   from R1, R2
   where B = D and C = E
   ```
4. ```
   select distinct B, C
   from R1, R2
   where B = D and C = E.
   ```

Soluzione

1. **SÌ** perché nel join effettuato tra le relazioni R_1 e R_2 sulla condizione C=D potrebbe accadere che una stessa tupla di R_1 si combini con più tuple di R_2 (essendo D solo uno degli attributi della chiave di R_2) generando dei duplicati nel risultato.
2. **NO** perché gli argomenti della select coinvolgono l'attributo A chiave della relazione R_1 e l'operazione di differenza effettuata non produce tuple aggiuntive nel risultato.
3. **NO** perché gli argomenti della select coinvolgono l'attributo A chiave della relazione R_1 e la condizione di join tra le tabelle R_1 e R_2 coinvolge entrambi gli attributi chiave DE della relazione R_2 quindi non potranno essere presenti nel risultato tuple uguali.
4. **SÌ** perché gli argomenti della select non coinvolgono una chiave per la relazione R_1 e sebbene ogni tupla di R_1 si possa combinare con al più una tupla di R_2 a causa delle condizioni di join non è garantito che non ci siano coppie di attributi sempre diversi tra loro su BC.

Esercizio 4.13 Con riferimento ad una base di dati sullo schema R_1(A,B,C), R_2(A,B,C),R_3(C,D,E) considerare l'espressione dell'algebra relazionale $\pi_{AE}((R_1 \cup R_2) \bowtie R_3)$ e scrivere un'espressione SQL ad essa equivalente senza utilizzare il join esplicito (cioè la parola chiave JOIN) né viste.

Soluzione

```
select distinct A, E
from R1, R3
where R1.C = R3.C
union
select A, E
from R2, R3
where R2.C = R3.C.
```

Esercizio 4.14

Dato il seguente schema:

```
AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart,
                  CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)
```

scrivere le interrogazioni SQL che permettono di determinare:

1. Le città con un aeroporto di cui non è noto il numero di piste;
2. Le nazioni da cui parte e arriva il volo con codice AZ274;
3. I tipi di aereo usati nei voli che partono da Torino;
4. I tipi di aereo e il corrispondente numero di passeggeri per i tipi di aereo usati nei voli che partono da Torino. Se la descrizione dell'aereo non è disponibile, visualizzare solamente il tipo;
5. Le città da cui partono voli internazionali;
6. Le città da cui partono voli diretti a Bologna, ordinate alfabeticamente;
7. Il numero di voli internazionali che partono il giovedì da Napoli;
8. Il numero di voli internazionali che partono ogni settimana da città italiane (farlo in due modi, facendo comparire o meno nel risultato gli aeroporti senza voli internazionali);
9. Le città francesi da cui partono più di venti voli alla settimana diretti in Italia;
10. Gli aeroporti italiani che hanno solo voli interni. Rappresentare questa interrogazione in quattro modi:
    a) con operatori insiemistici;
    b) con un interrogazione nidificata con l'operatore `not in`;
    c) con un interrogazione nidificata con l'operatore `not exists`;
    d) con l'outer join e l'operatore di conteggio
11. Esprimere l'interrogazione pure in algebra relazionale;
12. Le città che sono servite dall'aereo caratterizzato dal massimo numero di passeggeri;

Soluzione:

1.
```
select Città
from AEROPORTO
where NumPiste is NULL
```

2.
```
select A1.Nazione, A2.Nazione
from AEROPORTO as A1 join VOLO on A1.Città=CittàArr
join AEROPORTO as A2 on CittàPart=A2.Città
where IdVolo= 'AZ274'
```

3.
```
select TipoAereo
from VOLO
where CittàPart='Torino'
```

4.
```
select VOLO.TipoAereo, NumPasseggeri
```

```
        from VOLO left join AEREO
        on  VOLO.TipoAereo=aereo.TipoAereo
        where CittàPart= 'Torino'
```

5.
```
        select CittàPart
        from AEROPORTO as A1 join VOLO on CittàPart=A1.Città
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione <> A2.Nazione
```

6.
```
        select CittàPart
        from VOLO
        where CittàArr= 'Bologna'
        order by CittàPart
```

7.
```
        select count(*)
        from VOLO join AEROPORTO on CittàArr=Città
        where CittàPart = 'Napoli' and Nazione <> 'Italia' and
        GiornoSett= 'Giovedì'
```

8. a.
```
        select count(*), CittàPart
        from AEROPORTO as A1 join VOLO  on A1.Città=CittàPart
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione='Italia' and A2.Nazione <> ' Italia'
        group by CittàPart
```
   b.
```
        select count(CittàArr)
        from AEROPORTO as A1 join VOLO  on A1.Città=CittàPart
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione='Italia' and A2.Nazione <> ' Italia'
        group by CittàPart
```

9.
```
        select CittàPart
        from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione='Francia' and A2.Nazione= 'Italia'
        group by CittàPart
        Having count(*) >20
```

10. a.
```
        select CittàPart
        from VOLO join AEROPORTO on CittàPart=Città
        where Nazione = 'Italia'
        except
        select CittàPart
        from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
        join AEROPORTO as A2 on CittàArr=A2.Città
        where (A1.Nazione=' Italia ' and A2.Nazione<>' Italia ' )
```

b.
```
        select CittàPart
        from VOLO join AEROPORTO on CittàPart=Città
        where Nazione= 'Italia' and CittàPart not in
        ( select CittàPart
        from AEROPORTO as A1 join VOLO on
```

```
A1.Città=CittàPart join AEROPORTO as A2 on CittàArr=A2.Città
where  A1.Nazione='Italia' and
A2.Nazione<> 'Italia'  )
```

c.
```
select CittàPart
from VOLO join AEROPORTO as A1 on CittàPart=Città
where Nazione= 'Italia' and
not exists ( select * from VOLO join AEROPORTO as A2
on A2.Città=CittàArr
where A1.Città=CittàPart and
A2.Nazione<>'Italia'  )
```

d.
```
select CittàPart
from AEROPORTO as A1 join VOLO on
A1.Città=CittàPart left join AEROPORTO as A2 on
(CittàArr=A2.Città and A2.Nazione='Italia')
where A1.Nazione='Italia'
group by CittàPart
having count (district A2.Nazione)= 1 )
```

11
$$\Pi_{\text{Cittàpart}}\ \sigma_{\text{Nazione='Italia'}}(\text{AEROPORTO} \rhd\lhd_{\text{Città=CittàPart}} \text{VOLO})$$
$$-$$
$$\Pi_{\text{CittàPart}}\ \sigma_{\text{Nazione='Italia'}}\ (\text{AEROPORTO} \rhd\lhd_{\text{Città=CittàPart}} \text{VOLO} \rhd\lhd_{\text{CittàArr=CittàK}} \rho_{\text{CittàK,NazioneK,nK}\leftarrow\text{Città,Nazione,NumPiste}} (\sigma_{\text{Nazione}\neq\text{'Italia'}}(\text{AEROPORTO})))$$

12
```
select CittàPart
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri=      (select
                          max( NumPasseggeri )from AEREO )
union
select CittàArr
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri=      (select max( NumPasseggeri )
                           from AEREO)
```

Esercizio 4.15

Dato il seguente schema:

```
DISCO(NroSerie, TitoloAlbum,Anno,Prezzo)
CONTIENE(NroSerieDisco,CodiceReg,NroProg)
ESECUZIONE(CodiceReg,TitoloCanz,Anno)
AUTORE(Nome,TitoloCanzone)
CANTANTE(NomeCantante,CodiceReg)
```

formulare le interrogazioni SQL che permettono di determinare:

1. I cantautori (persone che hanno cantato e scritto la stessa canzone) il cui nome inizia per 'D';
2. I titoli dei dischi che contengono canzoni di cui non si conosce l'anno di registrazione;
3. I pezzi del disco con numero di serie 78574, ordinati per numero progressivo, con indicazione degli interpreti per i pezzi che hanno associato un cantante;
4. Gli autori e i cantanti puri, ovvero autori che non hanno mai registrato una canzone e cantanti che non hanno mai scritto una canzone;
5. I cantanti del disco che contiene il maggior numero di canzoni;
6. Gli autori solisti di "collezioni di successi" (dischi in cui tutte le canzoni sono di un solo cantante e in cui almeno tre registrazioni sono di anni precedenti la pubblicazione del disco);
7. I cantanti che non hanno mai registrato una canzone come solisti;
8. I cantanti che non hanno mai inciso un disco in cui comparissero come unici cantanti;
9. I cantanti che hanno sempre registrato canzoni come solisti.

Soluzione:

1.
```
select NomeCantante
from  CANTANTE join ESECUZIONE on
CANTANTE.CodiceReg=ESECUZIONE.CodiceReg
join AUTORE on ESECUZIONE.TitoloCanz=AUTORE.TitoloCanzone
where Nome=NomeCantante and Nome like 'd%'
```

2.
```
select TitoloAlbum
from DISCO join CONTIENE
        on DISCO.NroSerie=CONTIENE.NroSerieDisco
        join ESECUZIONE on
        CONTIENE.CodiceReg=ESECUZIONE.CodiceReg
where ESECUZIONE.anno is NULL
```

3.
```
select NroProg, TitoloCanz, NomeCantante
from (CONTIENE left join CANTANTE on
        CONTIENE.NroSerieDisco=CANTANTE.CodiceReg)
        join ESECUZIONE
        on CONTIENE.codiceReg= ESECUZIONE.CodiceReg
where NroSerieDisco=78574
order by NroProg
```

4.
```
select Nome
from AUTORE
where Nome not in
         ( select NomeCantante
           from CANTANTE )
union
select NomeCantante
from CANTANTE
where NomeCantante not in
      ( select Nome
                    from AUTORE )
```

5.
```
create view DiscoNumerato (NroSerieDisco,NumCanzoni)
as select  NroSerieDisco , count(*)
from CONTIENE
group by NumSerieDisco

select NomeCantante
from CANTANTE join CONTIENE on
                  CANTANTE.CodiceReg=CONTIENE.CodiceReg
                  join DiscoNumerato on

CONTIENE.NroSerieDisco=DiscoNumerato.NroSerieDisco
where NumCanzoni= ( select max (NumCanzoni)
                               from DiscoNumerato)
```

6.
```
select NroSerie
from DISCO
where NroSerie not in
( select NroSerieDisco
  from CONTIENE join CANTANTE as S1 on
  CONTIENE.CodiceReg=S1.CodiceReg
  join CANTANTE as S2 on
  CONTIENE.CodiceReg =S2.CodiceReg
  where   S1.NomeCantante<>S2.NomeCantante )
   and NroSerie in
 ( select NroSerieDisco
  from (CONTIENE join ESECUZIONE on
   CodiceReg= CodiceReg ) join DISCO on
  DISCO.NroSerie=CONTIENE.NroSerieDisco)
   where ESECUZIONE.Anno<DISCO.Anno
  group by NroSerieDisco
  having count(*) >=3 )
```

7.
```
select distinct NomeCantante
from CANTANTE
where NomeC not in
```

```
        ( select S1.NomeCantante
          from CANTANTE as S1
          where CodiceReg not in
         ( select CodiceReg
           from CANTANTE S2
           where S2.NomeCantante <> S1.NomeCantante ) )
```

8.
```
Create view CantantiUnDisco (NomeCantante) as
select NomeCantante
from CONTIENE join CANTANTE on
         CONTIENE.CodiceReg=CANTANTE.CodiceReg
where NroSerieDisco not in
     (  select NroSerieDisco
        from CONTIENE join CANTANTE as S1 on
                      CONTIENE.CodiceReg=S1.CodiceReg
                       join CANTANTE as S2 on
                      CodiceReg=S2.CodiceReg
        where S1.NomeCantante<>S2.NomeCantante  )

select NomeCantante
from CANTANTE
where NomeCantante not in (select NomeCantante
                           from CantantiUnDisco)
```

9.
```
select NomeCantante
from CANTANTE
where NomeCantante not in
( select S1.NomeCantante
from CANTANTE as S1 join ESECUZIONE on
CodiceReg=S1.CodiceReg join CANTANTE as S2 on
CodiceReg=S2.CodiceReg )
where S1.NomeCantante<> S2.NomeCantante )
```

Esercizio 4.16 Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table Studenti (
       matricola numeric not null primary key,
       cognome char(20) not null,
       nome char(20) not null,
       eta numeric not null
);
create table Esami (
       codiceCorso numeric not null,
       studente numeric not null
           references Studenti(matricola),
       data date not null,
       voto numeric not null,
       primary key (codiceCorso, studente, data)
).
```

Si supponga che vengano registrati anche gli esami non superati, con voti inferiori al 18.
Formulare in SQL:

1. l'interrogazione che trova gli studenti che non hanno superato esami;
2. l'interrogazione che trova gli studenti che hanno riportato in almeno un esame un voto più alto di Archimede Pitagorico;
3. l'interrogazione che trova i nomi degli studenti che hanno superato almeno due esami;
4. l'interrogazione che trova, per ogni studente, il numero di esami superati e la relativa media.

Soluzione

1.
```
select *
from Studenti
where matricola not in (
        select distinct studente
        from Esami);
```

2.
```
select *
from Studenti join Esami
    on matricola=studente
where voto > any (
        select voto
        from Esami join Studenti
                on studente = matricola
                where cognome = 'pitagorico'
                and nome = 'archimede');
```

3.
```
select distinct s.nome, s.cognome
from Studenti s,
     Esami e1 join Esami e2
        on (e1.studente = e2.studente)
where
e1.codiceCorso > e2.codiceCorso and
e1.voto >= 18 and
e2.voto >= 18 and
s.matricola = e1.studente;
```

4.
```
select s.nome, s.cognome, count(*), avg(voto)
from Studenti s, Esami e
where
s.matricola = e.studente
group by s.nome, s.cognome.
```

Esercizio 4.17 Considerare la seguente base di dati relazionale:

- NEGOZI (IDNegozio, Nome, Città)
- PRODOTTI (CodProdotto, NomeProdotto, Marca)
- LISTINO (Negozio, Prodotto, Prezzo) con vincoli di integrità referenziale fra Negozio e la relazione NEGOZI fra Prodotto e la relazione PRODOTTI.

Formulare in SQL:

1. l'interrogazione che fornisce nome e città dei negozi che vendono prodotti della marca XYZ
2. l'interrogazione che trova, per ciascun prodotto, la città in cui viene venduto al prezzo più basso
3. l'interrogazione che trova i prodotti che vengono venduti in una sola città.

Fare riferimento ad una versione dell'SQL che non prevede la differenza (parole chiave EXCEPT e MINUS) e che permette l'uso dei confronti nella nidificazione solo su singoli attributi.
Sono quindi sono ammesse condizioni del tipo

... A IN SELECT C FROM ...

ma non del tipo

... (A,B) IN SELECT C,D FROM ...

Soluzione

1.
```
select distinct Nome, Citta
from Negozi, Prodotti, Listino
where IDNegozio = Negozio
and Prodotto = CodProdotto
and Marca = 'XYZ';
```

2.
```
select distinct P.NomeProdotto, P.Citta
from Negozi, Listino, Prodotti P
where prezzo <= all (
        select Prezzo, NomeProdotto
        from Prodotti P2, Listino
        where P.NomeProdotto = P2.NomeProdotto
        );
```

3.
```
select distinct P1.NomeProdotto
from Prodotti P1
where NomeProdotto not in (
        select P1.NomeProdotto
        from Negozi N1, Negozi N2, Prodotti P2,
        Listino L1, Listino L2
        where N1.IdNegozio = L1.Negozio
        and N2.IdNegozio = L2.Negozio
```

```
and L1.Prodotto = L2.Prodotto
and L1.Prodotto = P2.CodProdotto
and N1.Citta <> N2.Citta).
```

Esercizio 4.18

Dare una sequenza di comandi di aggiornamento che modifichi l'attributo Stipendio della tabella Impiegato, aumentando del 10% gli stipendi sotto i 30 mila € diminuendo del 5% gli stipendi sopra i 30 mila € (gli stipendi di 30.000 € rimangono invariati).

Soluzione:

```
update Impiegato set Stipendio= Stipendio*0.5
where Stipendio < 30000

update Impiegato set  Stipedio= Stipendio*0.95
where Stipendio > 30000

update Impiegato set Stipendio= Stipendio*2.2
where Stipendio < 15000
```

Esercizio 4.19 Considerare la base di dati relazionale con il seguente schema:

- PRODOTTI (Codice, Nome, Categoria)
- VENDITE (CodiceProdotto, Data, Incasso) con vincolo di integrità referenziale fra l'attributo CodiceProdotto e la relazione PRODOTTI

e la sua seguente istanza:

PRODOTTI		
Codice	Nome	Categoria
101	A	Bevanda
102	B	Bevanda
103	C	Pasta
104	D	Biscotti

VENDITE		
CodiceProdotto	Data	Incasso
101	24/11/2008	2.000
101	25/11/2008	1.000
102	23/11/2008	2.500
102	24/11/2008	4.000
103	25/11/2008	1.320

mostrare il risultato delle tre seguenti interrogazioni:

1.
```
select Codice
from Prodotti
where not exists
    (select *
     from Vendite
     where CodiceProd=Codice);
```

2.
```
select Codice
from Prodotti
where not exists
    (select *
     from Vendite
     where Data = 2008-11-24
     and CodiceProd=Codice);
```

3.
```
select Codice
from Prodotti
where not exists
    (select *
     from Vendite
     where Data = 2008-11-24);
```

Soluzione

1.

CODICE
104

2.

CODICE
103
104

3.

CODICE
103

Esercizio 4.20 Con riferimento all'esercizio 4.19 mostrare le istruzioni SQL per creare e popolare l'istanza di basi di dati mostrata.
Soluzione

```
create table Prodotti (
 codice numeric not null primary key,
 nome char(20) not null,
 categoria char(20) not null
);

create table Vendite (
 codiceProd numeric not null
          references Prodotti(codice),
 data date not null,
 incasso numeric not null,
 primary key (codiceProd, data));

delete * from Prodotti;
insert into Prodotti values(101,A,Bevanda);
insert into Prodotti values(102,B,Bevanda);
insert into Prodotti values(103,C,Pasta);
insert into Prodotti values(104,D,Biscotti);

delete * from Vendite;
insert into Vendite values(101,2008-11-24,2000);
insert into Vendite values(101,2008-11-25,1000);
insert into Vendite values(102,2008-11-23,2500);
insert into Vendite values(102,2008-11-24,4000);
insert into Vendite values(103,2008-11-25,1320);
```

Esercizio 4.21 Si consideri una base di dati sulle relazioni:

- R_1 (A,B,C)
- R_2 (D,E, F)

Mostrare unistanza della base di dati che confermi il fatto che le due espressioni seguenti non sono equivalenti:

1. $\pi_{AB}\,(R_1 \bowtie_{B=D} \sigma_{F=2}(R_2))$
2. ```
   select A , B
   from R1, R2
   where B = D
   and C = E
   and F=2.
   ```

Soluzione

R_1		
A	B	C
1	1	1

R_2		
D	E	F
1	2	2

Risultati:

1.

R_1	
A	B
1	1

2.

R_1	
A	B
   ```

Esercizio 4.22 Con riferimento alla base di dati dell'esercizio 4.20, supponendo che le cardinalità delle due relazioni siano rispettivamente N_1 e N_2, indicare le cardinalità (minime e massime) dei risultati delle seguenti interrogazioni:

1.
```
select *
from R1, R2
where C = D
and E > 100;
```

2.
```
select *
from R1 X1
where not exists
    (select *
     from R1 Y1, R2
     where Y1.C = D
     and X1.A = Y1.A
     and F>10);
```

3.
```
select distinct A , B
from R1, R2
where B = D
and C = E.
```

Soluzione

1. Compreso fra 0 e $N_1 \times N_2$
2. Compreso fra 0 e N_1
3. Compreso fra 0 e N_1.

Esercizio 4.23 Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table Tariffa (
  tipoAuto numeric not null primary key,
  costoAlKm numeric not null
);
create table Automobile (
  targa numeric not null primary key,
  tipologia char(20) not null
        references Tariffa(TipoAuto),
  lunghezza char(20) not null
);
create table Transito (
  codice numeric not null primary key,
  auto numeric not null
        references Automobile(targa),
  orarioIngresso numeric not null,
  orarioUscita numeric not null,
  KmPercorsi numeric not null
).
```

Formulare in SQL:

1. l'interrogazione che restituisce, per ogni transito, i dati del veicolo, del transito e il costo del pedaggio (ottenuto moltiplicando il costo al Km per i Km percorsi);
2. l'interrogazione che restituisce tutti i dati delle automobili che sono transitate più di una volta sull'autostrada;
3. l'interrogazione che restituisce le auto che in ogni transito hanno percorso sempre lo stesso numero di Km;
4. l'interrogazione che restituisce i dati dei transiti (gli stessi richiesti nella prima domanda) per cui la velocità media è superiore ai 140Km/h (si assuma che una differenza tra i due orari produca il risultato espresso in ore).

Soluzione

1.
```
select A.targa, A.tipologia, A.lunghezza,
        T.kmPercorsi * TA.costoAlKm
  from Transito T join Automobile A
       on (T.auto = A.targa)
           join Tariffa TA
               on (TA.tipoAuto = A.tipologia);
```
2.
```
select A.targa, A.tipologia, A.lunghezza
  from Automobile A join Transito T
      on (T.auto = A.targa)
          join transito T2
               on (T1.auto = T2.auto)
  where T1.codice > T2.codice;
```

3.
```
select A.targa
from Automobile A join Transito T
    on (A.targa = T.auto)
        left join Transito T2
            on (T1.auto = T2.auto)
where T1.codice > T2.codice
exept
select A.targa
from Automobile A join Transito T
    on (A.targa = T.auto)
        join Transito T2
            on (T1.auto = T2.auto)
where T1.codice > T2.codice
and (T1.km <> T2.Km);
```
4.
```
select A.*, T.*, T.KmPercorsi * TA.costoKM
from Transito T join Automobile A
    on (T.auto = A.targa)
        join Tariffa TA
            on (TA.tipoAuto = A.Tipologia)
where (T.KmPercorsi / (T.uscita-T.ingresso)) > 140.
```

Esercizio 4.24 Dato uno schema di base di dati relazionale con le seguenti due relazioni:

- R (A,B,C)
- S (C,D,E)

scrivere l'interrogazione in algebra relazionale corrispondente alla seguente interrogazione SQL:

```
select R1.A, R2.C, E
from R R1, R R2, S
where R1.C = R2.A
and S.C = R2.C
and R1.B > 7.
```

Soluzione
$proj_{A,I,E}$ ($\sigma_{B>7}$(R $\bowtie_{C=G}$ (($\rho_{G,H,I \leftarrow A,B,C}$ R) $\bowtie_{I=C}$ S))).

Esercizio 4.25 Si supponga di avere un sistema di basi di dati non in grado di eseguire left join espliciti. Si traduca la query SQL seguente in un suo equivalente supportato da tale dbms.

```
select *
from Animali left join Padroni
     on (padrone = codiceFiscale and etaPadrone>40).
```

Soluzione

```
select *
from Animali, Padroni
where (padrone = codiceFiscale and etaPadrone>40)
or  padrone = null.
```

Esercizio 4.26 Si consideri una base di dati che gestisce dati relativi ai voli in partenza da un dato aeroporto (ad esempio Roma), con le seguenti relazioni:

- AEROPORTI(Codice,Città,Nome);
- AEREI(Codice,Nome,NumeroPosti);
- VOLI(Compagnia,Numero,Destinazione,OraPart,OraArr,Aereo) con vincoli di integrità referenziale fra Destinazione e la relazione AEROPORTI e fra Aereo e la relazione AEREI.

Formulare in SQL:

1. l'interrogazione che trova le città raggiungibili con un volo diretto che utilizzi un aereo con almeno 200 posti.
2. l'interrogazione che trova le città raggiungibili con voli diretti e, per ciascuna, mostra il numero di tali voli.

Soluzione

1.
```
select distinct Citta
from Aeroporti join Voli
     on Aeroporti.Codice = Voli.Destinazione
     join Aerei on Aerei.Codice = Voli.Aereo
where Aerei.NumeroPosti >= 200;
```

2.
```
select Citta, count(*)
from Aereoporti join Voli
on Aeroporti.Codice = Voli.Destinazione
group by Citta.
```

Capitolo 5

Esercizio 5.1

Definire sulla tabella `Impiegato` il vincolo che il dipartimento `Amministrazione` abbia meno di 100 dipendenti, con uno stipendio medio superiore ai 40 mila €.

Soluzione:

```
check (100 >= ( select count(*)
          from Impiegato
                 where Dipartimento='Amministrazione' )
       and 40000 <= ( select avg(Stipendio)
                      from Impiegato
                      where        Dipartimento='Amministrazione'))
```

Esercizio 5.2 Definire (con una opportuna notazione) su una relazione PAGHE (Matricola, StipLordo, Ritenute, StipNetto, OK) un vincolo che imponga che il valore di OK è:

- zero se StipNetto è pari alla differenza fra StipLordo e Ritenute
- uno altrimenti.

Soluzione

```
    (Verifica = 0 and (Netto = StipLordo-Tasse))
or
    (Verifica = 1 and (Netto <> StipLordo-Tasse)).
```

Esercizio 5.3

Definire a livello di schema il vincolo che il massimo degli stipendi degli impiegati di dipartimenti con sede a Firenze sia minore dello stipendio di tutti gli impiegati del dipartimento Direzione.

Soluzione:

```
create assertion ControlloSalari
  check ( not exists(    select *
              from Impiegato join Dipartimento on
              Impiegato.Dipartimento=Dipartimento.Nome
              where Dipartimento.Città='Firenze' and
              Stipendio > (  select min(Stipendio)
                        from Impiegato
                        where
     Dipartimento='Direzione')
         )        )
```

Esercizio 5.4 Indicare quali delle seguenti affermazioni sono vere.

1. Nei sistemi relazionali le viste possono essere utili al fine di rendere più semplice la scrittura delle interrogazioni.
2. Nei sistemi relazionali le viste possono essere utili al fine di rendere più efficienti le interrogazioni.
3. Nei sistemi relazionali le viste introducono ridondanze.

Soluzione

1. Nei sistemi relazionali le viste possono essere utili al fine di rendere più semplice la scrittura delle interrogazioni. **VERO**
2. Nei sistemi relazionali le viste possono essere utili al fine di rendere più efficienti le interrogazioni. **VERO**
3. Nei sistemi relazionali le viste introducono ridondanze. **FALSO**

Esercizio 5.5

Dato il seguente schema:

```
AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart,
          CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)
```

scrivere, facendo uso di una vista, l'interrogazione SQL che permette di determinare il massimo numero di passeggeri che possono arrivare in un aeroporto italiano dalla Francia di giovedì (se vi sono più voli, si devono sommare i passeggeri).

Soluzione:

```
create view Passeggeri(Numero)
  as select sum ( NumPasseggeri )
  from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
     join AEROPORTO as A2 on A2.Città=CittàArr
  capitolo5_viste        join              AEREO              on
  VOLO.TipoAereo=Aereo.TipoAereo
      where A1.Nazione='Francia' and A2.Nazione='Italia'
          and GiornoSett='Giovedì'
      group by A2.Città

  select max(Numero)
  from Passeggeri
```

Esercizio 5.6

Definire una vista che mostra per ogni dipartimento il valore medio degli stipendi superiori alla media del dipartimento

Soluzione:

```
create view SalariSopraMedia (Dipartimento,Stipendio) as
select Dipartimento, avg(Stipendio)
from I
where Stipendio > ( select avg(Stipendio)
                    from Impiegato as I )
                where I.Dipartimento=I.Dipartimento )
group by Dipartimento
```

Esercizio 5.7 Dato il seguente schema relazionale:

- DIPENDENTE (CodiceFiscale, Cognome, Nome)
- PROFESSORE (CodiceFiscale, Qualifica, Anzianità,Facoltà*) con vincolo di integrità referenziale fra l'attributo CodiceFiscale e la relazione DIPENDENTE e fra l'attributo Facoltà e la relazione FACOLTÀ
- FACOLTÀ (Codice, Nome, Indirizzo)
- CORSODISTUDIO (Codice, Nome, Facoltà, Presidente) con vincolo di integrità referenziale fra l'attributo Facoltà e la relazione FACOLTÀ a fra l'attributo Presidente e la relazione PROFESSORE
- COLLABORAZIONE (CorsoDiStudio, Facoltà, Professore, Tipo) con vincolo di integrità referenziale fra l'attributo CorsodiStudio, Facoltà e la relazione CORSODISTUDIO e fra l'attributoProfessore e la relazione PROFESSORE
- CORSO (Codice, Materia, Docente, Semestre) con vincolo di integrità referenziale fra l'attributo Materia e la relazione MATERIA e fra Docente e la relazione PROFESSORE
- MATERIA (Sigla, Nome)

formulare le interrogazioni in SQL:

1. mostrare i professori, con codice fiscale, cognome, cognome, qualifica, anzianità e nome della eventuale facoltà di afferenza (per i professori che non afferiscono ad alcuna facoltà dovrà comparire il valore nullo)
2. trovare cognome e qualifica dei professori che afferiscono alla stessa facoltà di un professore chiamato Mario Bruni di qualifica "ordinario"
3. trovare i codici delle facoltà cui non afferisce alcun professore con cognome Bruni e qualifica "ordinario".

Soluzione
Poiché tutte le interrogazioni richiedono (anche più volte), il join di DIPENDENTE e PROFESSORE, è utile la vista:

```
create view prof as
    select d.cf, cognome, nome, qualifica,
           anzianita, facolta
    from dipendente d join professore p on d.cf=p.cf.
```

Interrogazioni:

1. ```
   select cf, cognome, qualifica,
        anzianita facolta.nome as facolta
   from prof left join facolta
                  on facolta=codice;
   ```

2. ```
   select distinct p1.cognome, p1.qualifica
   from prof p1 join prof p2
               on p1.facolta=p2.facolta
   where p2.cognome='Bruni'
   and p2.nome='Mario'
   ```

```
  and p2.qualifica='Ordinario';
```

3.
```
  select codice as codicefacolta
  from facolta
  where codice not in (
          select facolta
          from  prof
          where cognome='Bruni'
          and qualifica='Ordinario').
```

Esercizio 5.8 Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni (è lo schema già visto nell'esercizio 4.16):

```
create table Studenti (
       matricola numeric not null primary key,
       cognome char(20) not null,
       nome char(20) not null,
       eta numeric not null
);
create table Esami (
       codiceCorso numeric not null,
       studente numeric not null
           references Studenti(matricola),
       data date not null,
       voto numeric not null,
       primary key (codiceCorso, studente, data)
).
```

Formulare in SQL l'interrogazione che trova lo studente con la media più alta.
Soluzione

```
create view StudMedia as
        select studente, avg(voto) as media
        from Esami
        group by Studente;

select studente
from StudMedia
where media >= all (
    select media
    from StudMedia ).
```

Esercizio 5.9 Considerare la seguente base di dati relazionale:

- VENDITE (NumeroScontrino, Data)
- CLIENTI (Codice, Cognome, Età)
- DETTAGLIVENDITE (NumeroScontrino, Riga, Prodotto, Importo, Cliente) con valori nulli ammessi sull'attributo Cliente e con vincoli di integrità referenziale fra NumeroScontrino e la relazione VENDITE e fra Cliente e la relazione CLIENTI;

formulare in SQL:

1. l'interrogazione che restituisce i prodotti acquistati in ciascuna data (che mostra cioè le coppie $< p, d >$ tali che il prodotto p è stato acquistato nella data d;
2. l'interrogazione che restituisce i prodotti che sono stati acquistati in due date diverse;
3. la vista VENDITECONTOTALE(NumeroScontrino, Totale), che riporta, per ogni scontrino l'importo totale (ottenuto come somma degli importi dei prodotti riportati sullo scontrino).

Soluzione
Definiamo innanzitutto una vista:

1.
```
create view VD(P, D) as
select Prodotto, Data
from Vendite V join DettagliVendite D
             on V.Numeroscontrino = D.NumeroScontrino;

select distinct P, D
from VD;
```

2.
```
select P
from VD as VD1
where not exists (
    select *
    from VD as VD2
    where VD1.P=VD2.P and VD1.D<>VD2.D);
```

3.
```
create view VenditeConTotale as
     select NumeroScontrino, sum(Importo) As Totale
     from DettagliVendite
     group by NumeroScontrino.
```

Esercizio 5.10 Considerare la seguente base di dati relazionale:

- PERSONE (FC, Cognome, Nome, Età)
- IMMOBILI (Codice, Via, NumeroCivico, Città, Valore)
- PROPRIETÀ (Persona, Immobile, Percentuale) con vincolo di di integrità referenziale fra l'attributo Immobile e la relazione PERSONE e fra l'attributo Immobile e la relazione IMMOBILI.

Nota: l'attributo Percentuale indica la percentuale di proprietà.
Definire in SQL:

- la vista definita per mezzo della seguente espressione dell'algebra relazionale: Vista = Immobili $\bowtie_{Codice=Immobile}$ Proprietà
- l'interrogazione che fornisce codici fiscali, nome e cognome delle persone che posseggono un solo immobile e lo posseggono a 100%
- l'interrogazione che fornisce, per ciascuna persona, il codice fiscale, il nome, il cognome e il valore complessivo degli immobili di sua proprietà (dove il valore è la somma dei valori ciascuno pesato con la percentuale di proprietà: se Tizio possiede un immobile di valore 150 al 100% e uno di valore 200 al 50%, allora il valore complessivo sarà (150 x 100)/100 + (200 x 50)/100 = 250).

Soluzione

1.
```
create view ProprImmobili(Codice, Via, NumeroCivico,
                          Citta, Valore, Persona,
                          Percentuale) as
    select Codice, Via, NumeroCivico, Citta, Valore,
           Persona, Percentuale
    from Immobili, Proprieta
    where Codice = Immobile);
```
2.
```
select CF, Cognome, Nome
from ProprImmobili, Persone
where Persona = CF
and Percentuale = "100%"
and Codice <> all (
    select Proprietario
    from ProprImmobili PI1, ProprImmobili PR2
    where PI1.Codice <> PI2.Codice
    and PI1.Proprietario = PI2.Proprietario)
```
3.
```
select CF, Cognome, Nome, sum(Valore * Percentuale / 100)
from Persone join Proprieta
        on (CF = Persona)
            join Immobili
                on (Codice = Immobile
group by (CF, Cognome, Nome).
```

Esercizio 5.11

Si supponga di avere le tabelle:

Magazzino(Prodotto, QtaDisp, Soglia, QtaRiordino)
OrdineInCorso(Prodotto, Qta)

Scrivere una procedura SQL che realizza il prelievo dal magazzino accettando 2 parametri, il prodotto `prod` e la quantità da prelevare `QtaPrelievo`. La Procedura deve verificare inizialmente che `QtaPrelievo` sia inferiore al valore di `QtaDisp` per il prodotto indicato. `QtaPrelievo` viene quindi sottratta al valore di `QtaDisp`. A questo punto la procedura verifica se per il prodotto `QtaDisp` risulta minore di `Soglia`, senza che in `OrdineInCorso` compaia già una tupla relativa al prodotto prelevato; se si, viene inserito un nuovo elemento nella tabella `OrdineInCorso`. Con i valori di `Prod` e del corrispondente attributo `QtaRiordino`.

Soluzione:

```
#include<stdlib.h>
main()
{
exec sql begin declare section;
     char Prodotto, prod;
     int QtaDisp, Soglia, QtaRiordino, Qta, Qta1, i;
exec sql end declare section;

exec sql declare MagazzinoCursore cursor for
     select  Prodotto, QtaDisp, Soglia, QtaRiordino
     from Magazzino;
exec sql open MagazzinoCursore;

prod = sceltaProdotto();
Qta= sceltaQta();
i=0;

do{
exec sql fetch MagazzinoCursore into
     :Prodotto, :QtaDisp, :Soglia, :QtaRiordino;
if (Prodotto == prod) i=1;
}while(Prodotto == prod || sqlca.sqlcode==0 )

if (i=1){
     printf("ERRORE - Prodotto non trovato");
     exit(1);
     }
if (QtaDisp < Qta){
     printf("ERRORE - Quantità non disponibile");
     exit(1);
     }
QtaDisp = QtaDisp - Qta;

if (QtaDisp < Soglia){
     Qtariordino = QtaRiordino + QtaDisp - Soglia;
     }


exec sql update magazzino
```

```
     set QtaDisp = :QtaDisp
     set QtaRiordino = :QtaRiordino
     where current of MagazzinoCursore;

exec sql declare OrdineCursore cursor for
     select  Prodotto, Qta
     from OrdineInCorso;
exec sql open OrdineCursore ;

i=0;
do{
exec sql fetch  OrdineCursore into
     :Prodotto, :Qta1;
if (Prodotto == prod) i=1;
}while(Prodotto == prod || sqlca.sqlcode==0 )

if (i=1){
     exec sql update OrdineInCorso
          set Qta = :QtaRiordino;
          where current of OrdineCursore;
     }
else exec sql insert into OrdineInCorso
     values(:prod,:QtaRiordino)";

}
```

Esercizio 5.12

Dato lo schema relazionale:

IMPIEGATO (Nome, Salario, DipNum)
DIPARTIMENTO (DipNum, NomeManager)

Definire le seguenti regole attive in Oracle e DB2:

1. una regola, che quando il dipartimento è cancellato, mette ad un valore di default (99) il valore di DipNum degli impiegati appartenenti a quel dipartimento;
2. una regola che cancella tutti gli impiegati appartenenti a un dipartimento quando quest'ultimo è cancellato;
3. una regola che, ogni qual volta il salario di un impiegato supera il salario del suo manager, pone tale salario uguale al salario del manager;
4. una regola che, ogni qual volta vengono modificati i salari, verifica che non vi siano dipartimenti in cui il salario medio cresce più del tre per cento, e in tal caso annulla la modifica.

Soluzione:

I 4 trigger hanno la stessa sintassi sia per Oracle che per DB2

1)
```
create trigger T1
after delete on DIPARTIMENTO
for each row
when (exists (select *
              from IMPIEGATO
              where DipNum=Old.DipNum))
udpade IMPIEGATO.DipNum = 99
```

2)
```
create trigger T2
after delete on DIPARTIMENTO
for each row
when (exist (select *
             from IMPIEGATO
             where DipNum=Old.DipNum))
delete from IMPIEGATO where DipNum=Old.DipNum
```

3)
```
create trigger T3
after update of Salario on IMPIEGATO
for each row
     declare x number;
     begin
          select Salary into x
          from IMPIEGATO join DIPARTIMENTO on
                         Nome = NomeManager
          Where DIPARTIMENTO.DipNum = New.DipNum
          if new.Salario > x then
               update IMPIEGATO set Salario = x
               where Nome = New.Nome
     end
```

4)
```
create trigger T4
after update of Salario on IMPIEGATO
for each row
     declare x number;
     declare y number;
     declare l number;
     begin
          select avg(salario), count(*) into x,l
          from IMPIEGATO
          where DipNum=new.DipNum;
          y=((x*l)-new.Salario+old.Salario)/l;
          if (x>(y*1.03)) then
          update IMPIEGATO set Salario=old.Salario
          where DipNum=new.DipNum;
     end
```

Esercizio 5.13

Riferendosi alla base di dati dell'esercizio precedente, definire in DB2 e in Oracle un trigger R_1 che, quando è cancellato un impiegato che svolge il ruolo di manager di un dipartimento, cancella quel dipartimento e tutti i suoi dipendenti. Definire inoltre un trigger R_2 che, ogni qual volta vengono modificati i salari, verifica la loro media, e se essa supera i 50.000 cancella tutti gli impiegati il cui salario è stato modificato e attualmente supera gli 80.000. Si consideri poi uno stato di base di dati con sei impiegati: Giovanna, Maria, Andrea, Giuseppe, Sandro e Carla, in cui:

- Giovanna è manager del dipartimento 1, in cui lavorano Giovanna, Maria e Giuseppe;
- Maria è Manager del dipartimento 2, in cui lavora Andrea.
- Giuseppe è manager del dipartimento 3, in cui lavorano Sandro e Carla.

Si assuma infine una transazione che cancella l'impiegata Giovanna e modifica i salari in modo tale che la loro media ecceda i 50.000 e il salario di Maria dopo le modifiche ecceda 80.000. Descrivere l'operato dei trigger.

Soluzione:

La soluzione per il trigger R_1 è uguale sia per Oracle che per DB2.

```
create trigger R1
after delete on IMPIEGATO
for each row
when (Old.No in( select NomeManager
                 from DIPARTIMENTO )
begin
     delete from IMPIEGATO where DipNum = Old.DipNum;
     delete from DIPARTIMENTO where DipNum = Old.DipNum;
end
```

La soluzione per il trigger R_2 è diversa per i due DBMS.

```
(ORACLE)
create trigger R2Oracle
after update of Salario on IMPIEGATO
for each statement
when ( (select avg(Salario) from new_table) > 50000)
delete from IMPIEGATO
where Salario > 80000
and Nome in ( Select new_table.Nome
              from new_tableas n join old_table as o
                   on n.Nome = o.Nome
              where n.Salario <> o.Salario)

(DB2)
create trigger R2DB2
after update of Salario on IMPIEGATO
when ( (select avg(Salario) from new_table) > 50000)
delete from IMPIEGATO
where Salario > 80000
and Nome in ( Select new_table.Nome
```

```
        from new_tableas n join old_table as o
             on n.Nome = o.Nome
        where n.Salario <> o.Salario)
```

Assegnando il seguente stato iniziale alla base di dati:

IMPIEGATO

Nome	Salario	DipNum
Giovanna	60000	1
Maria	50000	1
Giuseppe	50000	1
Andrea	40000	2
Sandro	40000	3
Carla	40000	3

DIPARTIMENTO

DipNum	NomeManager
1	Giovanna
2	Maria
3	Giuseppe

Transazione SQL:

```
delete from IMPIEGATO where Nome = "Giovanna"
```
update IMPIEGATO set Salario = Salario * 1,2
update IMPIEGATO set Salario = 85000 where Nome = "Maria"

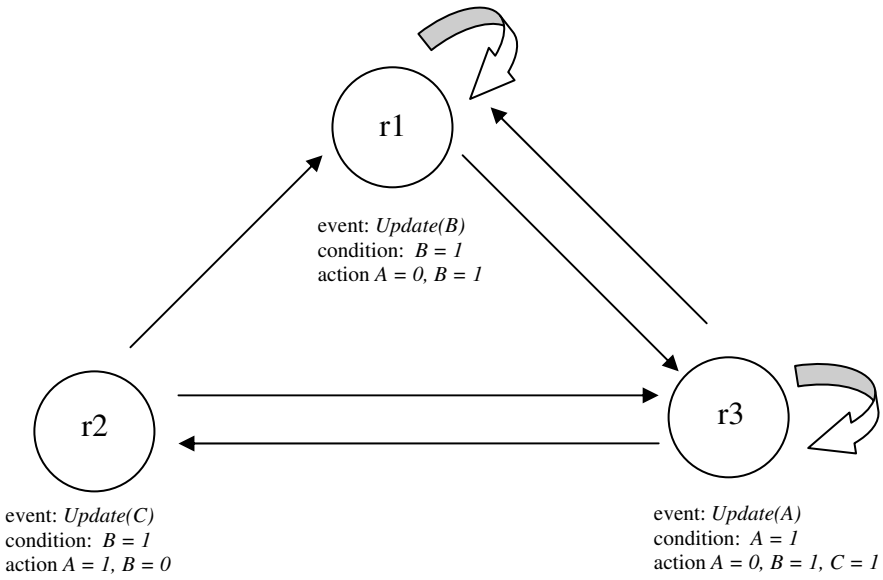
Il comportamento della base dati dell'esempio è molto semplice. Il trigger R1 reagisce alla cancellazione dell'impiegato Giovanna cancellando i dipartimento e tutti i suoi impiegati. Gli impiegati di Giovanna erano Maria e Giuseppe. La cancellazione di questi dipendenti causa nuovamente l'esecuzione del trigger R1, che cancella gli impiegati Andrea, Sandro e Carla e i loro Dipartimenti. Come si può vedere il trigger degenera e cancella tutti gli impiegati della base dati.
Le successive operazioni non modificano il DB.

Esercizio 5.14

Mostrare il grafo di attivazione delle seguenti regole, descritte in forma sintetica, e discutere quindi la loro terminazione:

- r1:
  - event: *Update(B)*
  - condition: *B = 1*
  - action *A = 0, B = 1*
- r2:
  - event: *Update(C)*
  - condition: *B = 1*
  - action *A = 1, B = 0*
- r3:
  - event: *Update(A)*
  - condition: *A = 1*
  - action *A = 0, B = 1, C = 1*

Soluzione



Analizziamo ogni singola regola attiva del database.
La regola r1 si attiva quando avvengono delle modifiche su B e se B=1 esegue le modifiche descritte. Queste modifiche fanno sospendere il trigger e partire il nuovo trigger sulla regola r3, che gestisce la risorsa A. Questa non fa nulla in quanto la condizione A = 1 non viene verificata. L'esecuzione torna a r1 che setta B = 1 ed entra in ciclo infinito.
La regola r2 si verifica quando si ha un aggiornamento della risorsa C. La condizione è che B sia uguale a 1. Se non è verificata il trigger non fa nulla. Se la condizione si verifica il trigger esegue

l'azione ed esegue la regola r3 che controlla la risorsa A. Se A = 1 la regola setta la risorsa A a 0, imposta B a 1, facendo partire il trigger sulla regola r1 che mette A a 0 (come già era) e torna a r3 che imposta C = 1 facendo rieseguire la regola r2 ed entra in un ciclo infinito.
La regola r3 si attiva sugli eventi sulla risorta A e cicla con la regola r2 come nell'esempio precedente.

Esercizio 5.15

Dato lo schema relazionale:

DOTTORANDO (Nome, Disciplina, Relatore)
PROFESSORE (Nome, Disciplina)
CORSO (Titolo, Professore)
ESAMI (NomeStud, TitoloCorso)

Descrivere i trigger che gestiscono i seguenti vincoli di integrità (*business rules*):

1. ogni dottorando deve lavorare nella stessa area del suo relatore;
2. ogni dottorando deve aver sostenuto almeno 3 corsi nell'area del suo relatore;
3. ogni dottorando deve aver sostenuto l'esame del corso di cui è responsabile il suo relatore.

Soluzione:

1)
```
create trigger T1
after update of Disciplina on DOTTORANDO
for each row
when Disciplina <> select Disciplina
     from PROFESSORE
     where PROFESSORE.Nome=new.Relatore
then signal SQLSTATE 70005 ( "Disciplina sbagliata" )
```

2)
```
create trigger T2
after update of Disciplina on DOTTORANDO
for each row
when 3 < ( select count(*)
     from ESAMI join CORSO on TitoloCorso = Titolo
          join PROFESSORE on Professore = PROFESSORE.Nome
          join DOTTORANDO on NomeStud = DOTTORANDO.Nome
          join PROFESSORE as P2 on Relatore = P2.Nome
     where PROFESSORE.Disciplina = P2.Disciplina
          and DOTTORANDO.Nome=new.Nome )
then signal SQLSTATE 70006 ("Almeno 3 corsi")
```

3)
```
create trigger T3
after update of Relatore on DOTTORANDO
for each row
when not exists ( select *
                  from ESAME join CORSO on TitoloCorso = Titolo
                  where NomeStud = new.Nome and
                      Professore = new.Relatore )
then signal SQLSTATE 70007 ("Esame mancante")
```

Esercizio 5.16

Tramite la definizione di una vista permettere all'utente "Carlo" di accedere al contenuto di Impiegato, escludendo l'attributo Stipendio.

Soluzione:

Ipotizzando la tabella Impiegato

```
   Impiegato(Codice, Nome, Cognome, Stipendio, Dipartimento)
create view ImpiegatoRistretto
(Codice,Nome,Cognome,Dipartimento) as
select  Codice,Nome,Cognome,Dipartimento
from Impiegato

grant select on ImpiegatoRistretto to Carlo
```

Esercizio 5.17

Descrivere l'effetto delle seguenti istruzioni: quali autorizzazioni sono presenti dopo ciascuna istruzione? (ciascuna linea è preceduta dal nome dall'utente che esegue il comando)

```
Stefano:  grant select on Table1 to Paolo,Riccardo
               with grant option
Paolo:         grant select on Table1 to Piero
Riccardo: grant select on Table1 to Piero with grant option
Stefano:       revoke select on Table1 from Paolo cascade
Piero:         grant select on Table1 to Paolo
Stefano:       revoke select on Table1 from Riccardo cascade
```

Soluzione:

1. Stefano concede a Paolo e a Riccardo l'autorizzazione di `select` e di concedere a loro volta l'autorizzazione
2. Paolo concede a Piero l'autorizzazione di `select`
3. Riccardo concede a Piero l'autorizzazione di `select` e di `grant`. Ora Piero ha 2 diverse autorizzazioni sulla tabella.
4. Stefano revoca l'autorizzazione data a Paolo. A causa dell'attributo `cascade` anche Piero perde le autorizzazioni concesse da Paolo ma continua ad avere quella concessa da Riccardo.
5. Ora Paolo può di nuovo accedere alla tabella grazie all'autorizzazione concessa da Piero
6. Stefano revoca l'autorizzazione di Riccardo e tramite `cascade` anche di Piero e di Paolo. Ora solo Stefano ha autorizzazioni sulla tabella.

Esercizio 5.18 Considerare i seguenti vincoli di integrità:

- a

```
CHECK ((Crediti = 0 AND Voto < 18) OR
(Crediti > 0 AND Voto >= 18))
```

- b

```
CHECK (Crediti > 0 AND Voto >= 18)
```

- c

```
CHECK (Crediti = 0 AND Voto < 18)
```

- d nessuna delle precedenti

e le seguenti specifiche:

1. sono registrati solo gli esami superati (con voto pari almeno a 18) e i crediti sono sempre positivi
2. il voto è pari almeno a 18 se e solo se i crediti sono maggiori di zero
3. se il voto è pari almeno a 18 i crediti sono positivi, se il voto è inferiore a 18 non c'è vincolo sui crediti
4. nessuna delle precedenti.

Abbinare vincoli e specifiche.
Soluzione

1. b
2. a
3. d
4. c

Esercizio 5.19 Indicare quali tra le seguenti affermazioni sono vere:

1. tra viste è possibile definire vincoli di integrità referenziale
2. possibile inserire record in viste utilizzando operazioni DDL
3. le possono essere ottenute esclusivamente come risultato di una valutazione di una query
4. l'utilizzo di viste può consentire il miglioramento delle prestazioni nell'esecuzione di una query
5. non è possibile definire una vista con record duplicati
6. la modifica dei dati a cui si riferisce la query che genera una vista implica la modifica del risultato della vista stessa.

Soluzione

1. Falso
2. Falso
3. Vero
4. Vero
5. Falso
6. Falso

Esercizio 5.20 Indicare quali tra le seguenti affermazioni sono vere:

1. sulle viste è possibile definire vincoli di dominio
2. le viste possono essere utilizzate per semplificare sintatticamente la scrittura di una query
3. la DDL "INSERT INTO nomeVista" permette l'inserimento di un record in una vista
4. la cancellazione di una vista implica la cancellazione nelle tabelle originarie di tutti i dati riportati
5. è necessario definire a priori la chiave primaria di una vista
6. l'esecuzione di una query su viste è sempre più efficiente dell'esecuzione della stessa query su tabelle poiché le viste non sono materializzate.

Soluzione

1. Falso
2. Vero
3. Falso
4. Falso
5. Falso
6. Falso

Esercizio 5.21 Nell'assunzione che queste due query siano sintatticamente e semanticamente corrette, dedurre lo schema della vista STUDENTI e della tabella PERSONA.

```
create view Studenti as
   select Nome, Cognome, 'studente', Eta
   from Persona
   where
       Mansione <> 'dipendente';

select *
from Studenti
  minus
select *
from Persone
where Mansione = 'Studente'.
```

Quale relazione esiste fra la cardinalità della vista STUDENTI e la cardinalità della relazione PERSONE?
Soluzione

- STUDENTI (Nome, Cognome, Mansione, Età)
- PERSONE (Nome, Cognome, Mansione, Età)

La cardinalità delle due relazioni risulta essere:
$|\text{PERSONE}| >= |\text{STUDENTI}|$;
vale l'uguale nell'ipotesi di assenza di dipendenti.

Esercizio 5.22 Sia dato il seguente schema relazionale:

- FILM (Titolo, Anno, Genere)
- ATTORE (Cognome, Nome, Nazionalità)
- PARTECIPAZIONE (CognomeAttore, NomeAttore, TitoloFilm, Compenso) con vincoli di integrità referenziale fra gli attributi CognomeAttore, NomeAttore e la relazione ATTORI e fra l'attributo TitoloFilm e la relazione FILM.

Definire in SQL, anche attraverso l'uso di viste:

1. l'interrogazione che trova gli attori che hanno partecipato ad almeno un film:
2. l'interrogazione che trova gli attori che hanno partecipato ad almeno cinque film;
3. l'interrogazione che trova i cognomi di tutti gli attori che hanno partecipato ad un film insieme a Sylvester Stallone.

Soluzione

1.
```
select Cognome, Nome
from Attore a
where exists (
    select *
    from Partecipazione
    where a.Cognome = NognomeAttore
    and a.Nome = NomeAttore);
```

2.
```
create view AttoriFilm as
    select Cognome, Nome, count(*) as NumeroFilm
    from Attore a join Partecipazione p
             on (a.Cognome = p.CongnomeAttore
                 and a.Nome = p.NomeAttore)
    group by Cognome, Nome;

select Cognome, Nome
from AttoriFilm
where NumeroFilm >=5;
```

3.
```
create view FilmStallone as
    select Titolo
    from Partecipazione
    where CognomeAttore = 'Stallone'
    and NomeAttore = 'Sylvester';

select Cognome
from Attori join Partecipazione
          on (Cognome = CognomeAttore
              and Nome = NomeAttore)
```

```
where Titolo not in (
       select Titolo
       from FilmStallone).
```

Esercizio 5.23 Dato il modello relazionale seguente:

- SPEDIZIONE (Pacco, Mittente, Destinatario, DataStimata, DataEffettiva)
- PACCO (IdPacco, TipoMerceologico, GradoFragilità, Peso)
- TIPOMERCEOLOGICO (IdTipoMerceologico, Descrizione, PolizzaAssicurativa)
- ASSICURAZIONE (IdAssicurazione, Nome, Indirizzo, Coefficiente)
- UTENTE (IdUtente, Cognome, Nome, Indirizzo)

scrivere in SQL:

1. l'interrogazione che trova tutti i pacchi spediti a Paolo Rossi
2. l'interrogazione che trova tutti i pacchi spediti da Paolo Rossi a Mario Bruni
3. l'interrogazione che trova i cognomi di tutti gli utenti che hanno spedito almeno due pacchi con la assicurazione SECUR.

Soluzione

1.
```
select Pacco.*
from Pacco, Spedizione, Utente
where Pacco.Destinatario = Utente.IdUtente
and Utente.Cognome = 'Rossi'
and Utente.Nome = 'Paolo';
```

2.
```
select Pacco.*
from Pacco, Spedizione, Utente u1, Utente u2
where Pacco.Destinatario = u1.IdUtente
and Pacco.Mittente = u2.IdUtente
and u2.Cognome = 'Rossi'
and u2.Nome = 'Paolo'
and u1.Cognome = 'Bruni'
and u1.Cognome = 'Mario';
```

3.
```
create view Pacchi_spediti_con_SECUR as
     select idUtente, count (*) as NumeroPacchi
     from Spedizione S, Pacco P, TipoMerceologico TM,
          Assicurazione A, Utente U
     where P.Mittente = U.IdUtente
     and   S.Pacco = P.IdPacco
     and   TM.IdTipoMerceologico = P.TipoMerceologico
     and   A.IdAssicurazione = TM.PolizzaAssicurativa
     and   Assicurazione.Nome = 'SECUR'
     group by IdUtente;

select Utente.Cognome
from Pacchi_spediti_con_SECUR v, Utente
where NumeroPacchi >= 2
and Utente.IdUtente = v.IdUtente.
```

Esercizio 5.24 Dato lo schema relazionale dell'esercizio 5.18 scrivere in SQL:

1. l'interrogazione che trova i cognomi di tutti gli utenti che hanno ricevuto esattamente due pacchi da Michele Argento
2. l'interrogazione che trova il pacco più fragile spedito prima di dicembre 2008
3. l'interrogazione che trova tutti i pacchi spediti con ritardo.

Soluzione

1. ```
   select cognome, count(*)
   from Spedizione, Pacco, Utente u1, Utente u2
   where Spedizione.Mittente = u1.IdUtente
   and Spedizione.Destinatario = u2.IdUtente
   and u1.Nome = 'Michele'
   and u1.Cognome = 'Argento'
   having count(*) = 2;
   ```

2. ```
   select Pacco.*
   from Spedizione S, Pacco P, TipoMerceologico TM
   where S.Pacco = Pacco.IdPacco
   and TM.IdTipoMerceologico = P.TipoMerceologico
   and DataEffettiva < '01-DEC-2008'
   and GradoFragilita <= all (
       select GradoFragilita
       from Spedizione S, Pacco P, TipoMerceologico TM
       where S.Pacco = P.IdPacco
       and TM.IdTipoMerceologico = P.TipoMerceologico
       and DataEffettiva < '01-DEC-2008'
       );
   ```

3. ```
   select Pacco.*
   from Pacco, Spedizione
   where Spedizione.pacco = Pacco.IdPacco and
   dataEffettiva > dataStimata.
   ```
   ```

Capitolo 6

Esercizio 6.1

Considerare lo schema ER in figura 6.36: lo schema rappresenta varie proprietà di uomini e donne.

- Correggere lo schema tenendo conto delle proprietà fondamentali delle generalizzazioni.
- Lo schema rappresenta solo le lavoratrici donne; modificare lo schema rappresentando ora tutti i lavoratori, uomini e donne.
- Tra le proprietà delle città, l'attributo `Regione` può essere visto anche come un attributo del concetto PROVINCIA. Ristrutturare lo schema in tal senso.

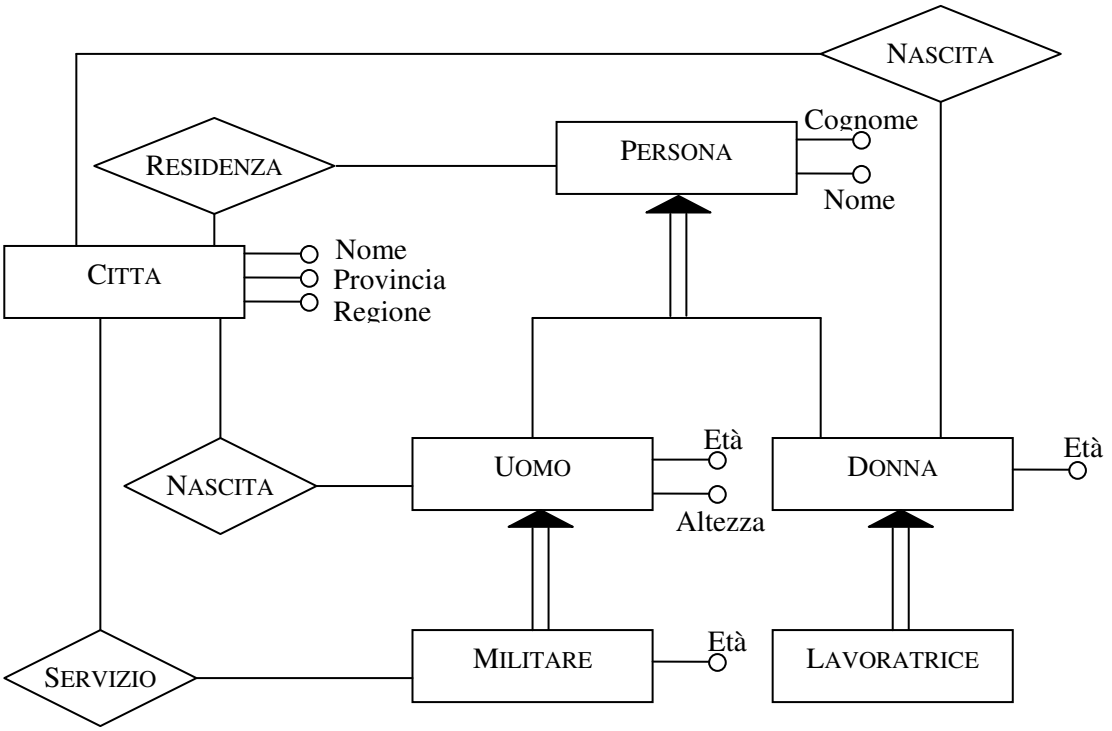
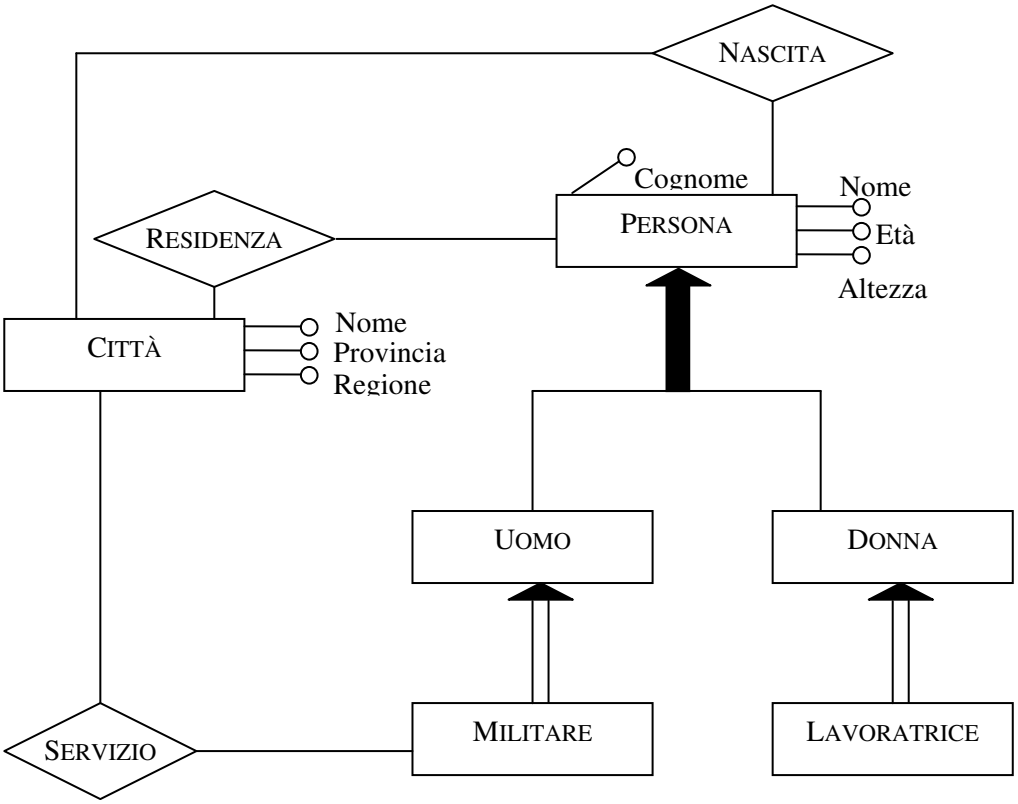


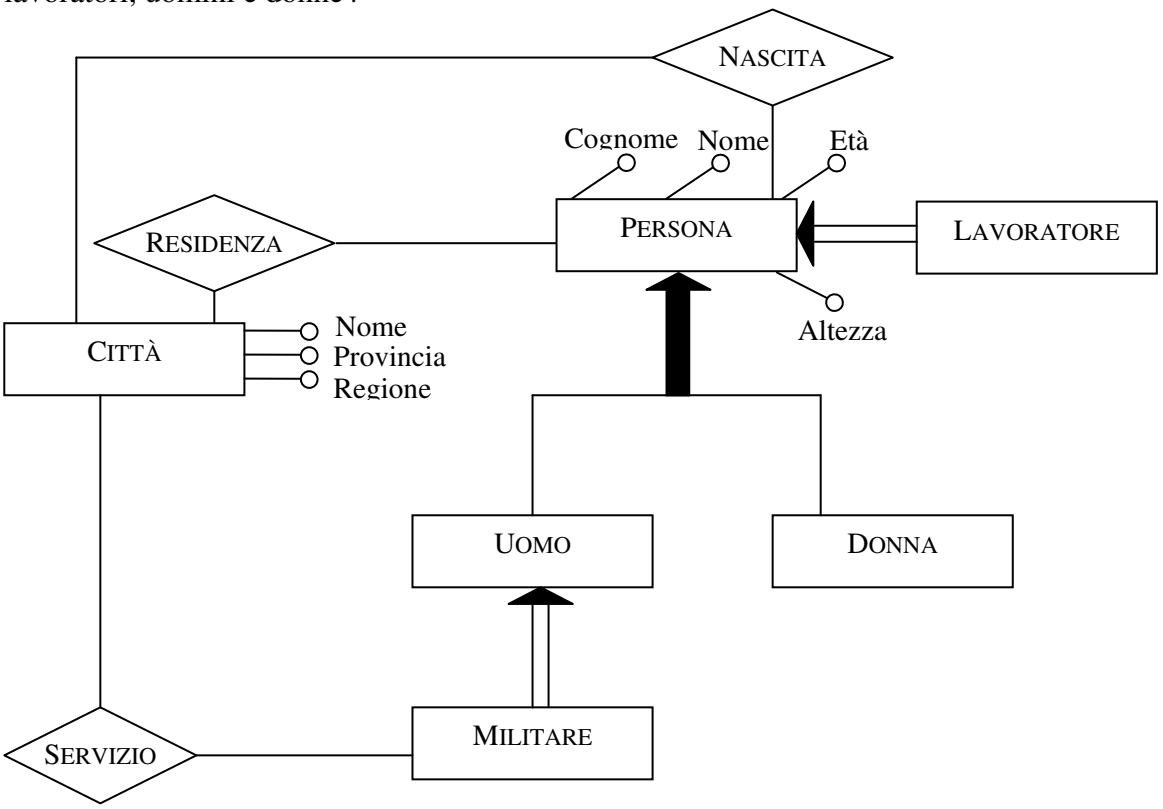
Figura 6.36 Schema E-R per l'esercizio 6.1

Soluzione:

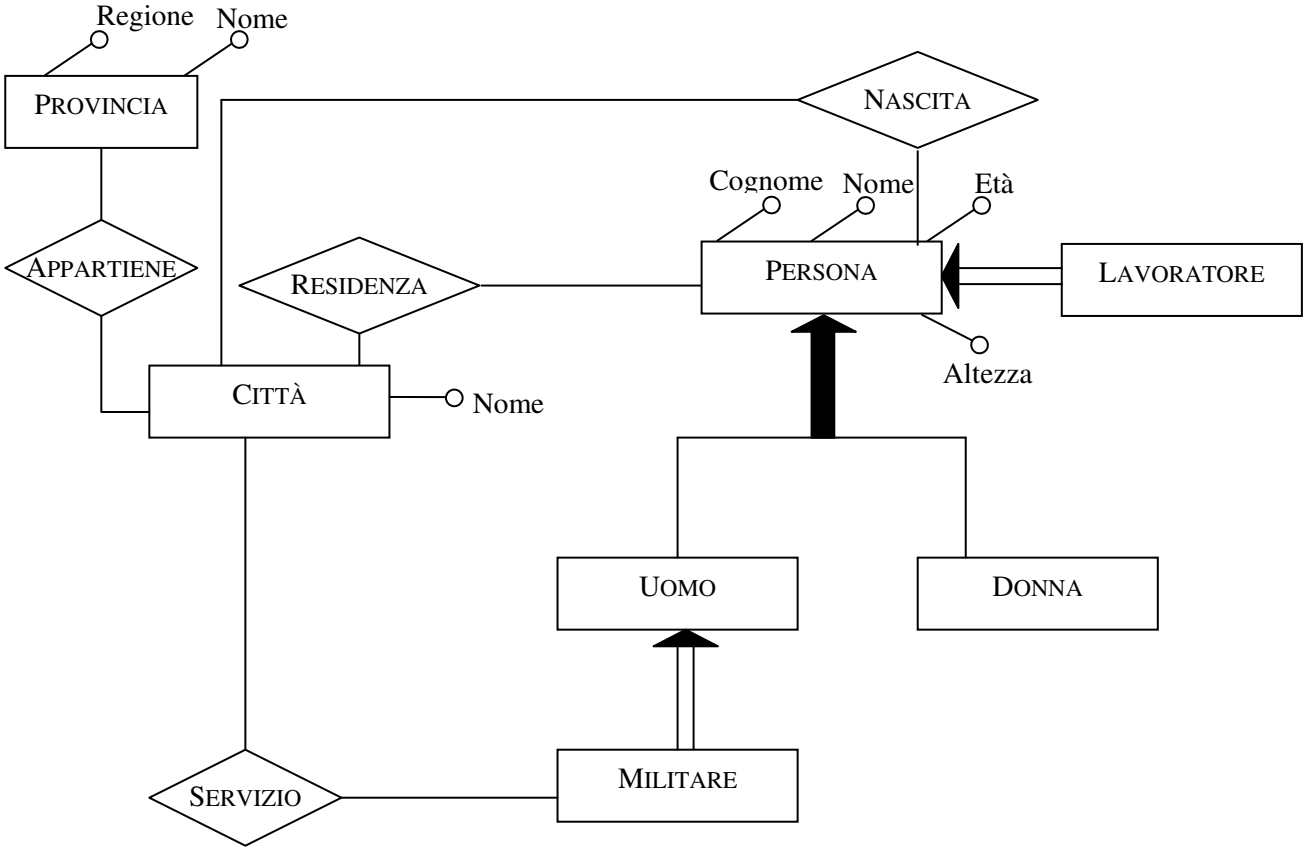
1) Correggere lo schema tenendo conto delle proprietà fondamentali delle generalizzazioni



2) Lo schema rappresenta solo le lavoratrici donne; modificare lo schema rappresentando ora tutti i lavoratori, uomini e donne .



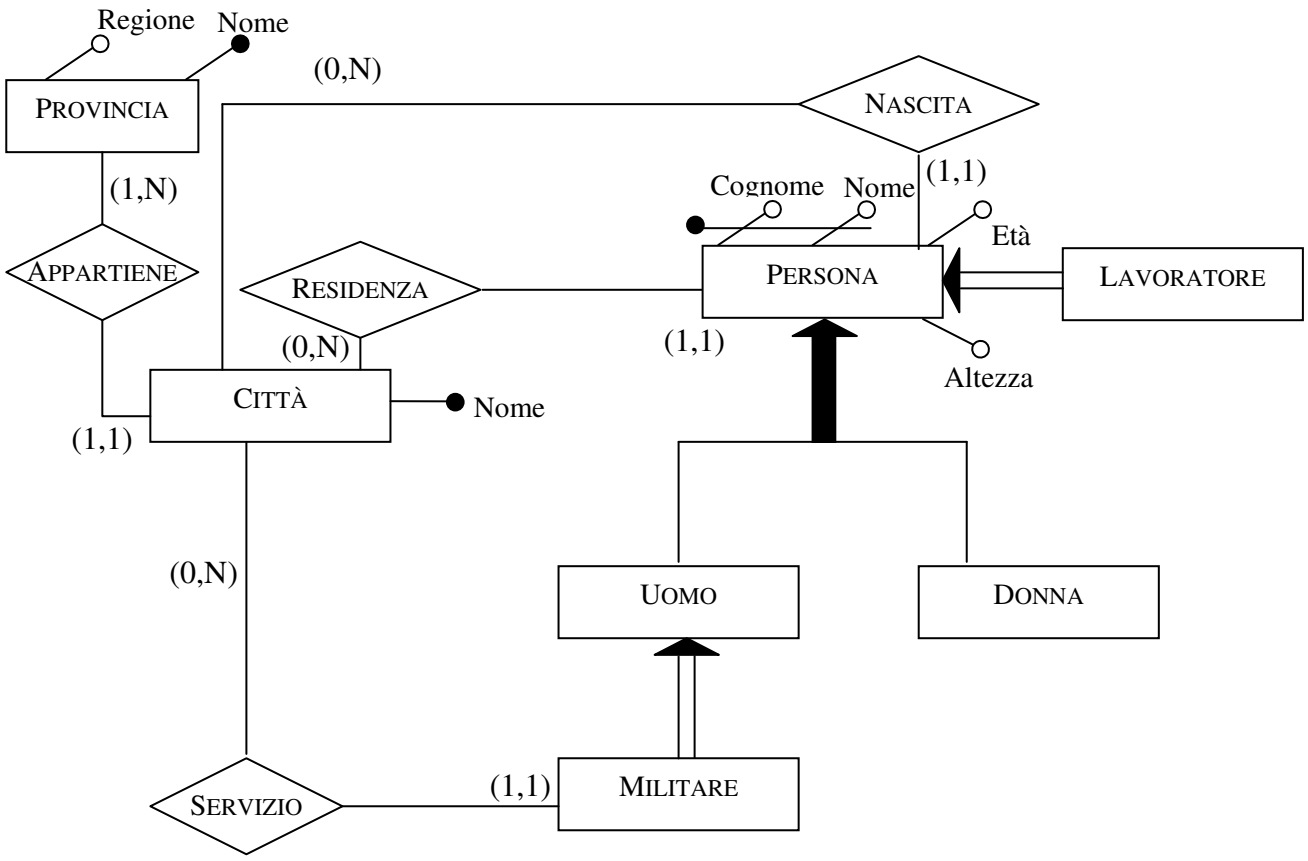
3) Tra le proprietà delle città, l'attributo `Regione` può essere visto anche come un attributo del concetto PROVINCIA. Ristrutturare lo schema in tal senso.



Esercizio 6.2

Aggiungere le cardinalità minime e massime allo schema prodotto nell'esercizio 6.1 e gli identificatori principali. Dire se esistono dei vincoli di integrità sullo schema che non possono essere espressi con il modello Entità-Relazione.

Soluzione:



I vincoli che non possono essere espressi nello schema Entità-Relazione sono:

- L'età degli uomini che svolgono il servizio militare deve essere superiore ai 18 anni.
- I lavoratori devono avere almeno 18 anni.
- L'altezza degli uomini che svolgono il servizio militare deve essere almeno uguale ad un minimo richiesto.

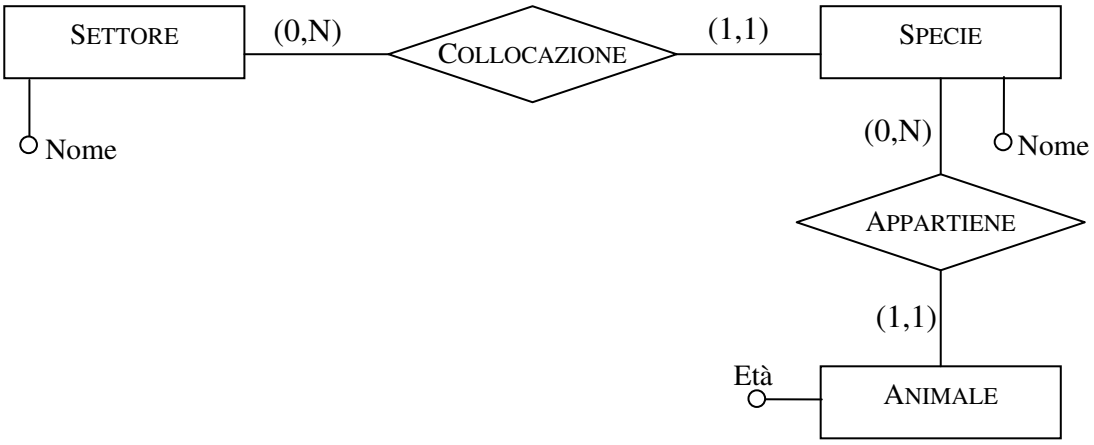
Esercizio 6.3

Rappresentare le seguenti realtà utilizzando i costrutti del modello Entità-Relazione e introducendo solo le informazioni specificate.

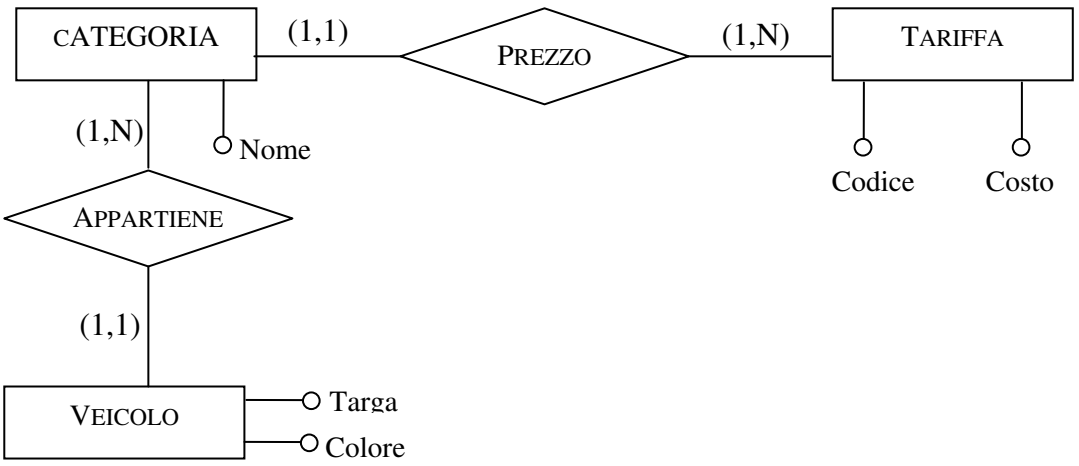
- In un giardino zoologico ci sono degli animali appartenenti a una specie e aventi una certa età; ogni specie è localizzata in un settore (avente un nome) dello zoo.
- Una agenzia di noleggio di autovetture ha un parco macchine ognuna delle quali ha una targa, un colore e fa parte di una categoria; per ogni categoria c'è una tariffa di noleggio.
- Una casa discografica produce dischi aventi un codice ed un titolo; ogni disco è inciso da uno o più cantanti, ognuno dei quali ha un nome, un indirizzo e, qualcuno, un nome d'arte.

Soluzione:

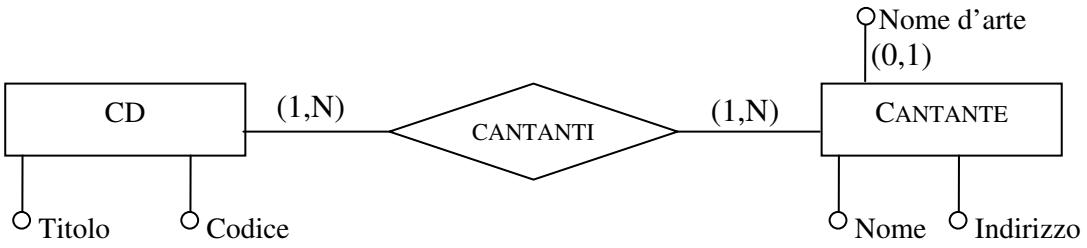
1) In un giardino zoologico ci sono degli animali appartenenti a una specie e aventi una certa età; ogni specie è localizzata in un settore (avente un nome) dello zoo.



2) Una agenzia di noleggio di autovetture ha un parco macchine ognuna delle quali ha una targa, un colore e fa parte di una categoria; per ogni categoria c'è una tariffa di noleggio.



3) Una casa discografica produce dischi aventi un codice ed un titolo; ogni disco è inciso da uno o più cantanti, ognuno dei quali ha un nome, un indirizzo e, qualcuno, un nome d'arte.

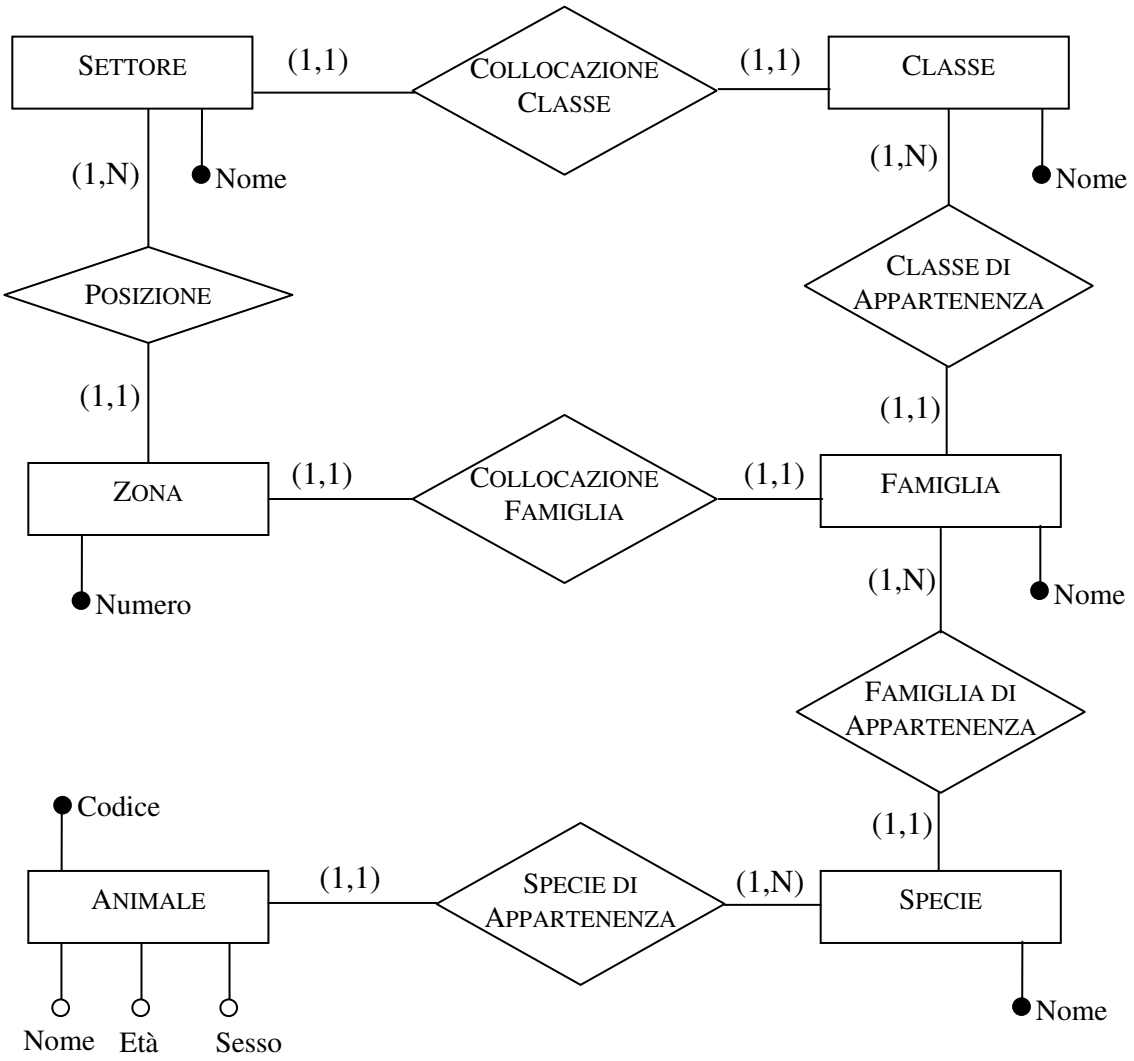


Esercizio 6.4

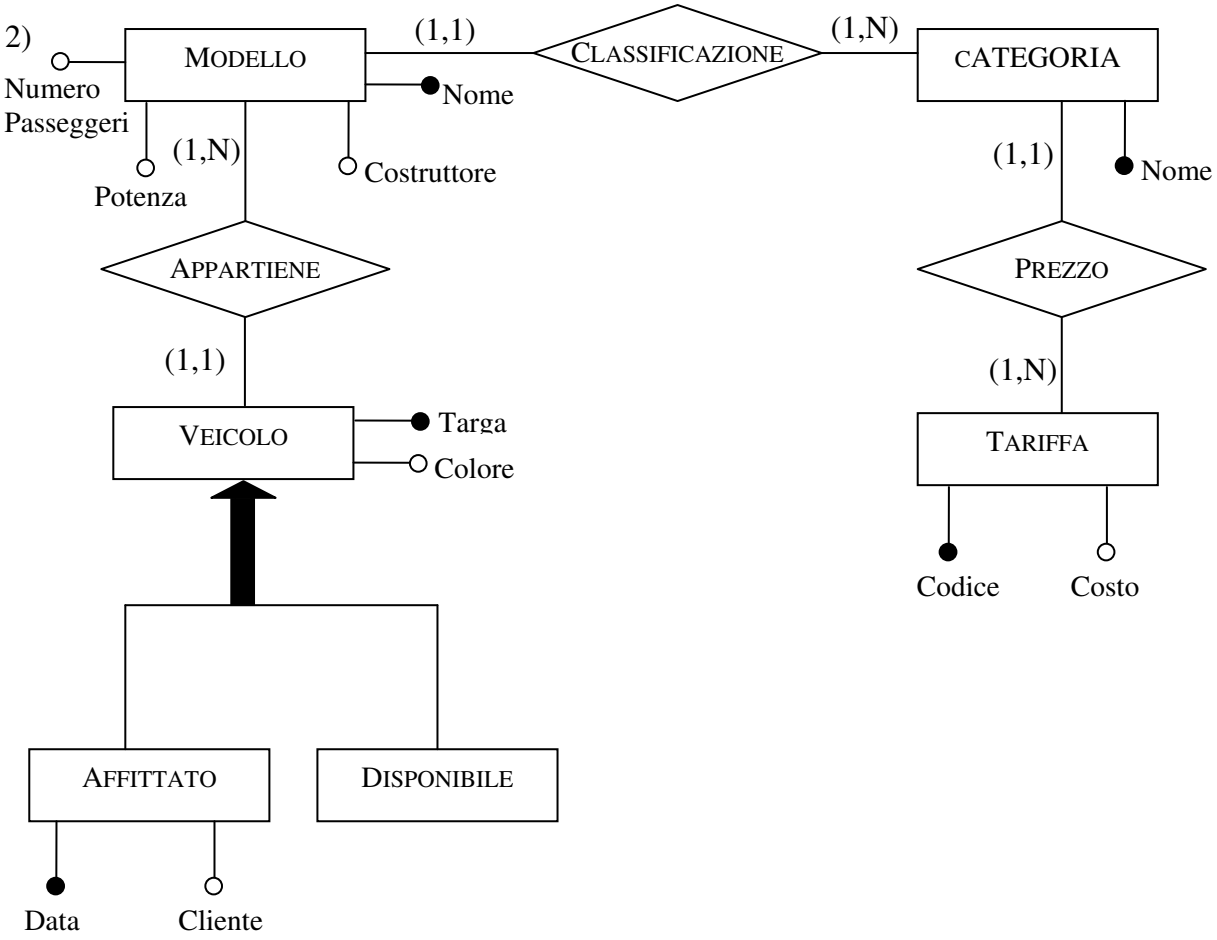
Completare i frammenti di schema prodotti nell'esercizio precedente con ulteriori informazioni, basandosi sulle proprie conoscenze o facendo delle ipotesi sulle rispettive realtà di interesse.

Soluzione:

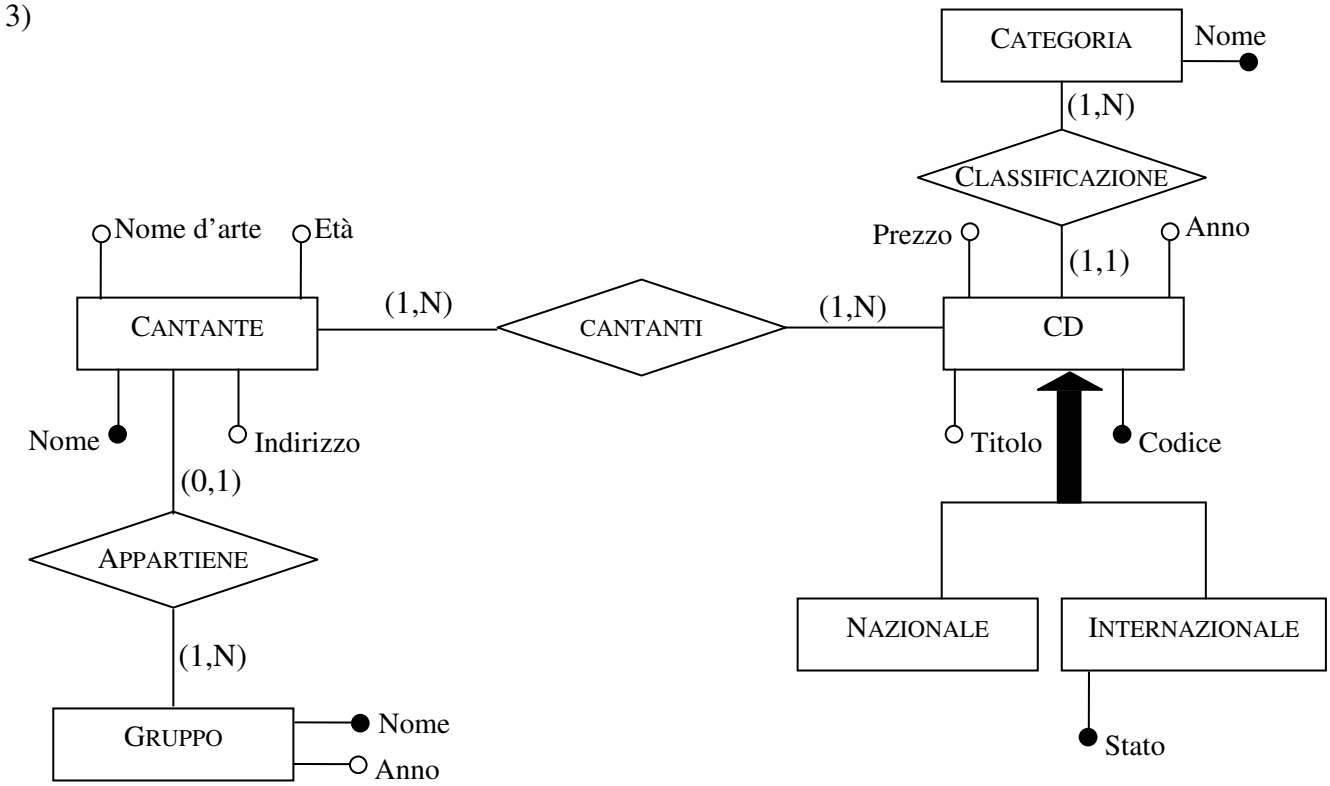
1)



2)


MODELLO
(1,1)
CLASSIFICAZIONE
(1,N)
CATEGORIA
Numero Passeggeri
Nome
Potenza
(1,N)
Costruttore
(1,1)
Nome
APPARTIENE
PREZZO
(1,1)
(1,N)
VEICOLO
Targa
Colore
TARIFFA
Codice
Costo
AFFITTATO
DISPONIBILE
Data
Cliente

3)


CATEGORIA
Nome
(1,N)
CLASSIFICAZIONE
Nome d'arte
Età
Prezzo
Anno
(1,1)
CANTANTE
(1,N)
CANTANTI
(1,N)
CD
Nome
Indirizzo
Titolo
Codice
(0,1)
APPARTIENE
(1,N)
NAZIONALE
INTERNAZIONALE
GRUPPO
Nome
Anno
Stato

Esercizio 6.5

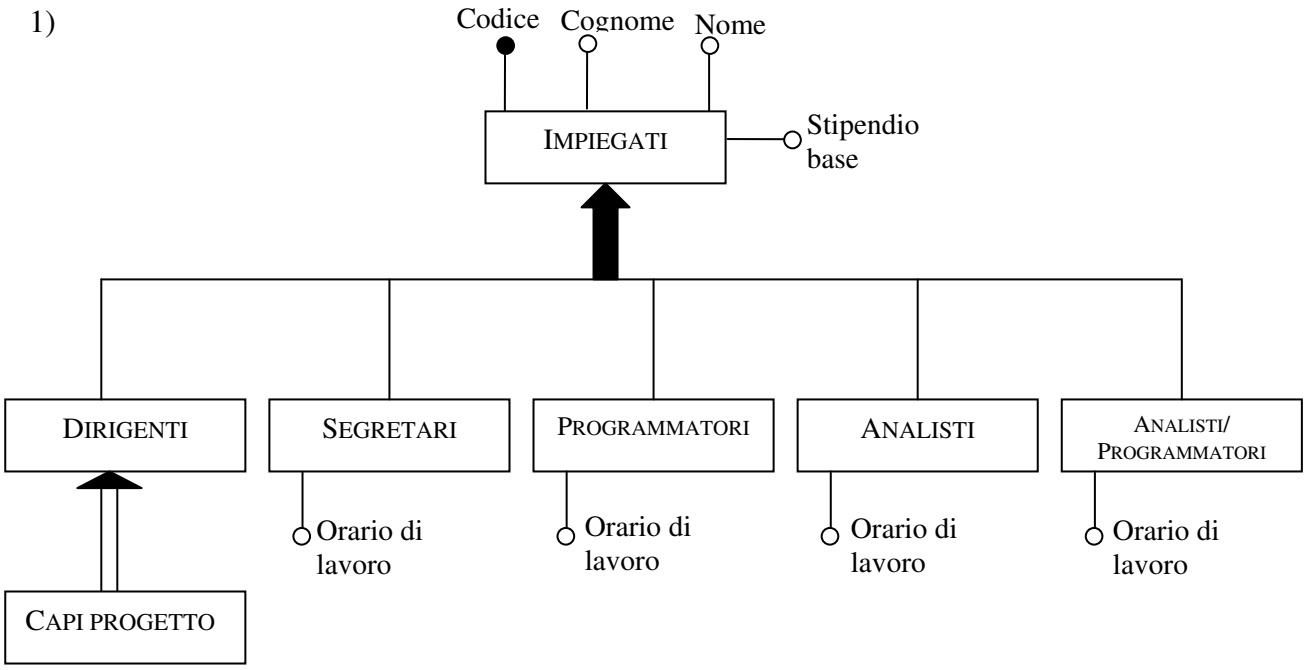
Rappresentare le seguenti classi di oggetti facendo uso, dove opportuno, del costrutto di generalizzazione del modello Entità-Relazione. Indicare nei vari casi, gli attributi delle varie entità e il tipo di generalizzazione, risolvendo i casi di sovrapposizione.

- Gli impiegati di una azienda si dividono in dirigenti, programmatori, analisti, capi progetto e segretari. Ci sono analisti che sono anche programmatori. I capi progetto devono essere dirigenti. Gli impiegati hanno un codice, un nome e un cognome. Ogni categoria di impiegato ha un proprio stipendio base. Ogni impiegato, tranne i dirigenti, ha un orario di lavoro.
- Una compagnia aerea offre voli che possiedono un numero che identifica la tratta (per esempio, Roma-Milano), una data (25 marzo 2001), un orario di partenza (ore 8:00) e uno di arrivo (ore 9:00), un aeroporto di partenza e uno di destinazione. Ci sono voli nazionali e internazionali. I voli internazionali possono avere uno o più scali. Dei voli passati è di interesse l'orario reale di partenza e di arrivo (per esempio, con riferimento al volo suddetto, ore 8:05 e 9:07), di quelli futuri è di interesse il numero di posti disponibili.
- Una casa automobilistica produce veicoli che possono essere automobili, motocicli, camion e trattori. I veicoli sono identificati da un numero di telaio e hanno un nome (per esempio, Punto), una cilindrata e un colore. Le automobili si suddividono in utilitarie (lunghezza sotto i due metri e mezzo) e familiari (lunghezza sopra i due metri e mezzo). Vengono anche classificate in base alla cilindrata: piccola (fino a 1200 cc), media (da 1200 cc a 2000cc) e grossa cilindrata(sopra i 2000 cc). I motocicli si suddividono in motorini (cilindrata sotto i 125 cc) e moto (cilindrata sopra i 125 cc). I camion hanno un peso e possono avere un rimorchio.

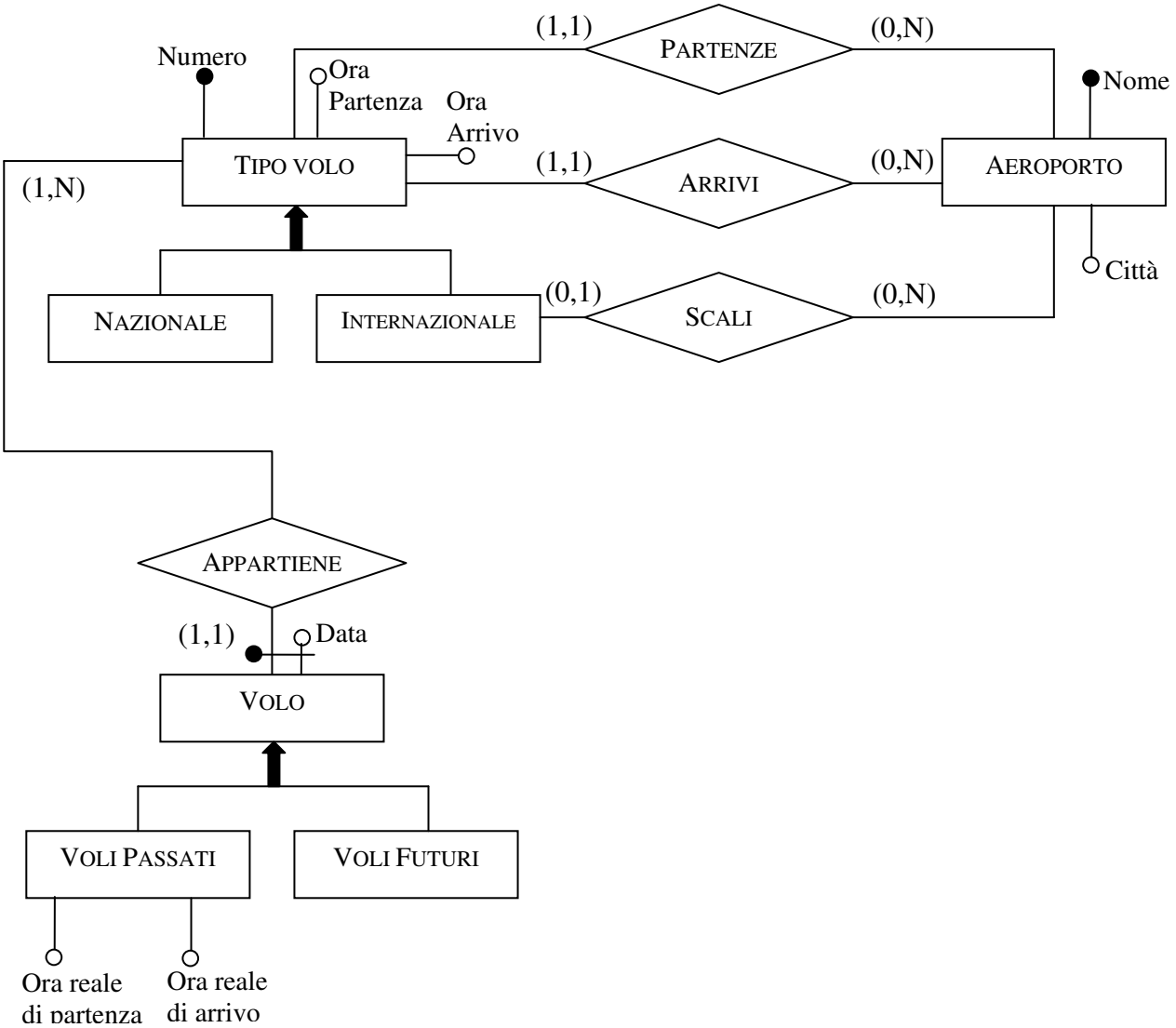
Soluzione:

Tutte le generalizzazioni sono esclusive

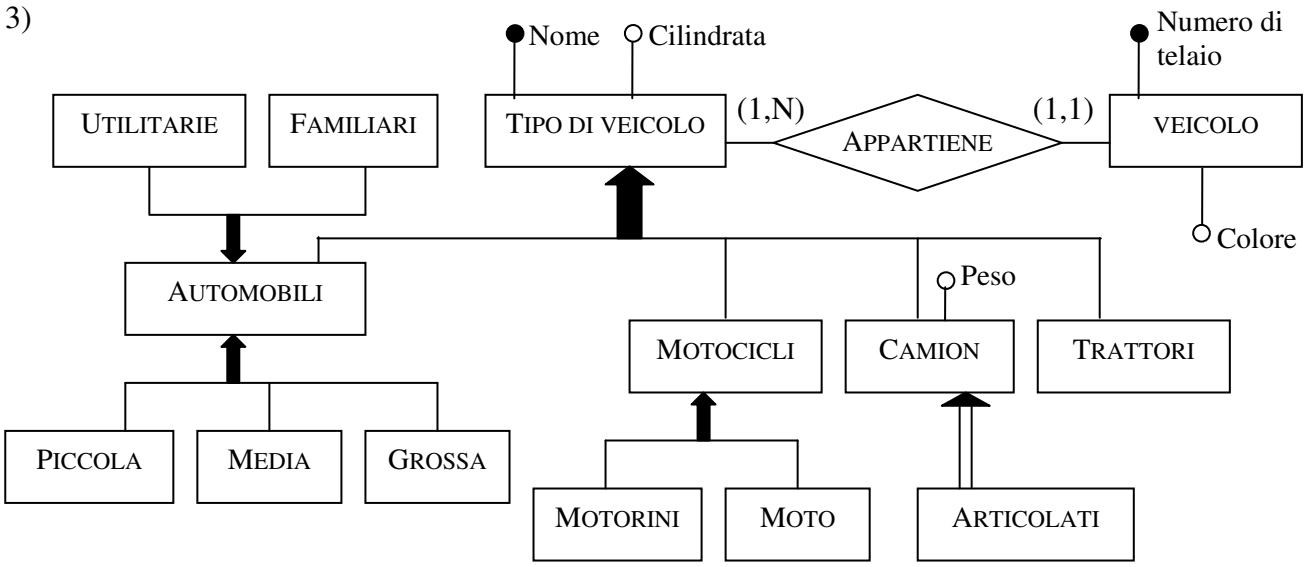
1)



2)


(1,1)
PARTENZE
(0,N)
Numero
Ora Partenza
Ora Arrivo
Nome
TIPO VOLO
(1,1)
ARRIVI
(0,N)
AEROPORTO
(1,N)
Città
NAZIONALE
INTERNAZIONALE
(0,1)
SCALI
(0,N)
APPARTIENE
(1,1)
Data
VOLO
VOLI PASSATI
VOLI FUTURI
Ora reale di partenza
Ora reale di arrivo

3)


Nome
Cilindrata
Numero di telaio
UTILITARIE
FAMILIARI
TIPO DI VEICOLO
(1,N)
APPARTIENE
(1,1)
VEICOLO
Colore
AUTOMOBILI
Peso
MOTOCICLI
CAMION
TRATTORI
PICCOLA
MEDIA
GROSSA
MOTORINI
MOTO
ARTICOLATI

Esercizio 6.6

Si consideri lo schema Entità-Relazione in figura 6.37. Descrivere le informazioni che esso rappresenta utilizzando il linguaggio naturale.

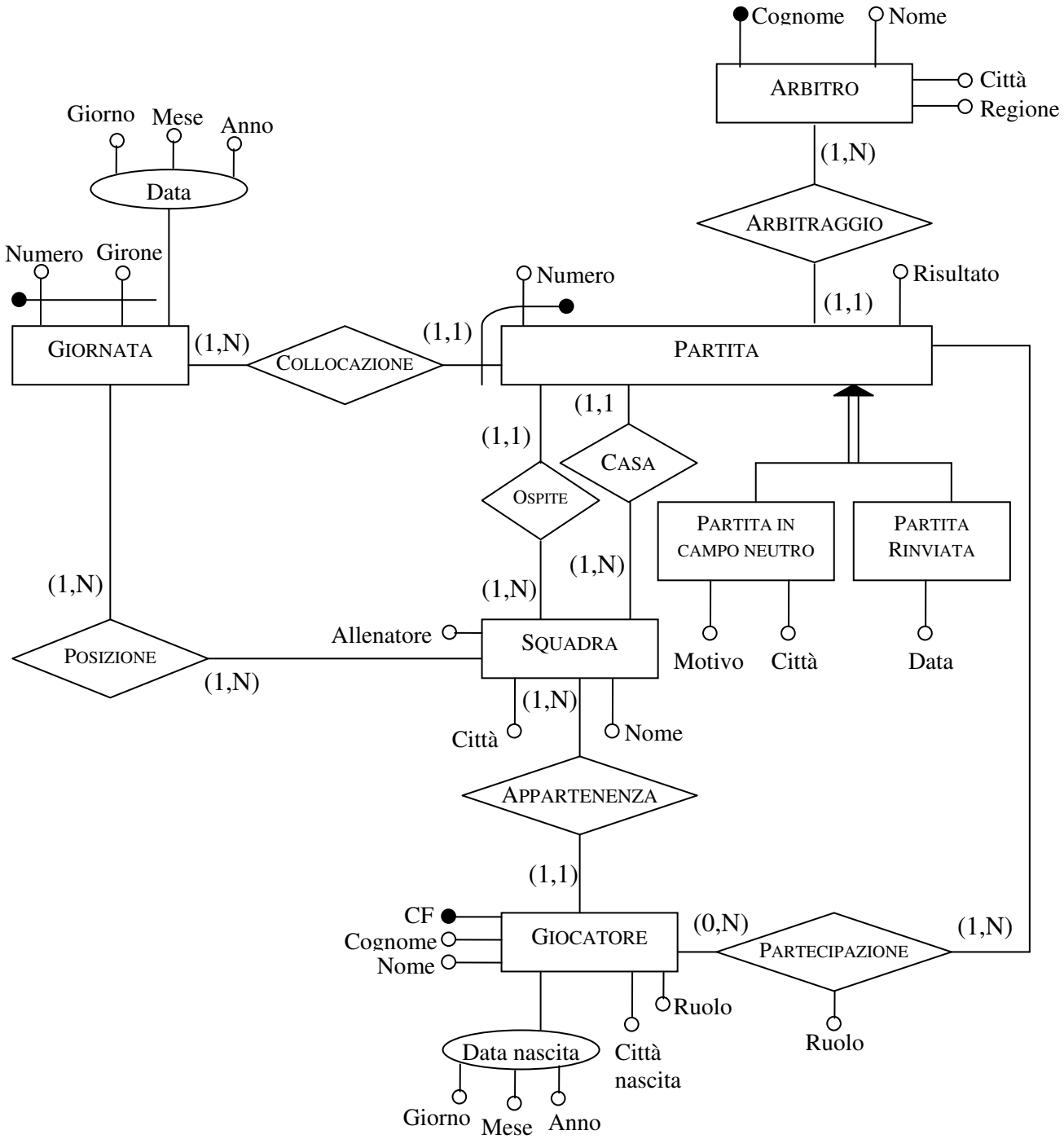


Figura 6.37 Schema E-R per l'esercizio 6.6

Soluzione:

Lo schema contiene le informazioni di un campionato (per esempio un campionato di calcio).
L'entità SQUADRA rappresenta tutte le squadre del campionato, indicando per ognuna di esse il nome, la città e il nome dell'allenatore. L'entità GIOCATORE rappresenta i giocatori delle squadre: ogni giocatore ha un contratto con una sola squadra e ogni squadra ha più giocatori.
I giocatori sono identificati dal loro Codice Fiscale (CF) e per ognuno di essi è indicato il nome, il cognome, il ruolo nella squadra, la città di nascita e la data di nascita.
Lo schema contiene anche informazioni sulle partite del campionato con l'entità PARTITA. Una partita è identificata con un numero (che deve essere differente per tutte le partite dello stesso giorno) e con un riferimento al giorno (attraverso la relazione COLLOCAZIONE e l'entità GIORNATA).
Le relazioni CASA e OSPITE rappresentano le due squadre che giocano la partita: per ogni partita è indicato il risultato e l'arbitro, con la relazione ARBITRAGGIO tra PARTITA e ARBITRO; questa entità rappresenta tutti gli arbitri del campionato e per ognuno di essi è indicato il Nome, il Cognome, la Città e la Regione. Un arbitro è rappresentato solo se ha arbitrato almeno una partita.
Una partita può essere giocata su campo neutrale o può essere rinviata ad un'altra data (ma questi due eventi non sono ammessi contemporaneamente nello schema).
La relazione Partecipazione rappresenta il fatto che un giocatore abbia giocato in una partita, la sua posizione (che può essere diversa dalla sua solita). Lo schema non esprime la condizione che i giocatori che giocano una partita devono avere un contratto con una delle due squadre.
L'entità GIORNATA rappresenta la giornata del campionato. Sono identificate con Numero e Girone.
La relazione Posizione dà il punteggio di ogni squadra in ogni giornata.

Esercizio 6.7

Tradurre in regole aziendali le seguenti proprietà sui dati lo schema di figura 6.37.

- Non ci possono essere più di 5 giocatori in una squadra che giocano nello stesso ruolo.
- Una squadra guadagna 3 punti se vince, 1 se pareggia e 0 se perde.
- Se una squadra gioca in casa una partita, allora è ospite nella partita successiva

Produrre quindi una documentazione completa per tale schema.

Soluzione:

RA1) In una squadra, il numero di giocatori con la stessa posizione **DEVE ESSERE** inferiore a cinque.
RA2) Il numero di punti guadagnato da una squadra in una partita **È OTTENUTO** sottraendo il punteggio della giornata della partita dal punteggio che aveva nella giornata precedente.
RA3) Il numero di punti guadagnato da una squadra che vince una partita **DEVE ESSERE** 3.
RA4) Il numero di punti guadagnato da una squadra che pareggia una partita **DEVE ESSERE** 1.
RA5) Il numero di punti guadagnato da una squadra che perde una partita **DEVE ESSERE** 0.
RA6) La prossima partita di una squadra **È OTTENUTA** ricercando, tra tutti gli incontri della prossima giornata, l'unico che coinvolge la squadra.
RA6) La prossima partita di una squadra che ha giocato come ospite **DEVE ESSERE** in casa.

Si osserva, incidentalmente, che l'ultima regola non può essere rispettata da tornei "all'italiana" con n (pari) squadre in cui ogni squadra incontra tutte le altre squadre in n-1 turni di campionato, a meno che non vi siano solo 2 squadre. Si invita il lettore a dimostrare questa impossibilità.

DIZIONARIO DEI DATI

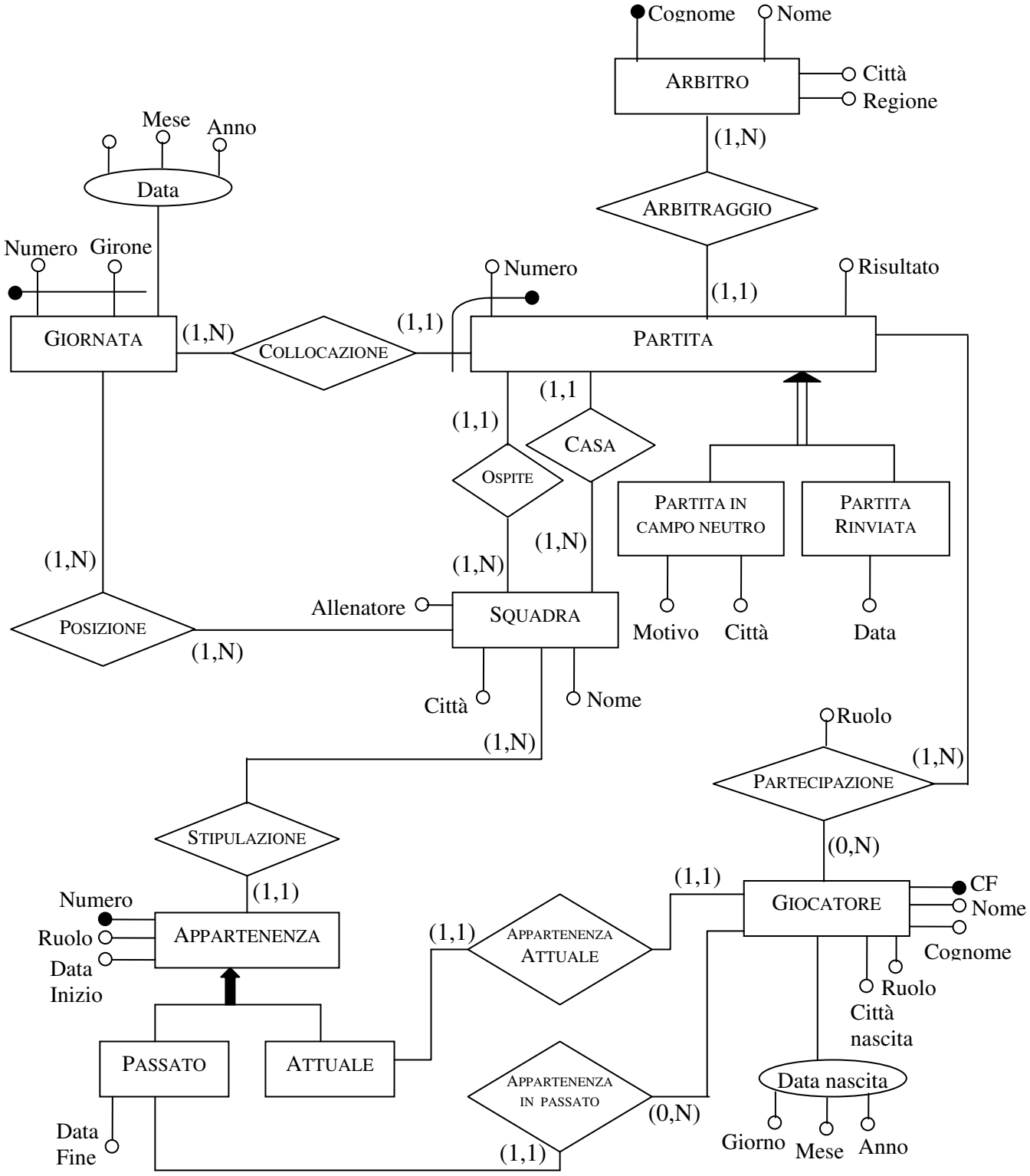
Entità	Descrizione	Attributi	Identificatore
Squadra	Una squadra che gioca nel campionato	Nome, Città, Allenatore	Nome
Giocatore	Giocatore che gioca in una squadra	Codice Fiscale, Cognome, Nome, Ruolo, Città di nascita, Data di nascita (Giorno, Mese, Anno)	Codice Fiscale
Partita	Una partita giocata durante il campionato	Numero, Risultato	Numero + Giornata (identificatore esterno)
Partita in campo neutro	Una partita giocata su un campo neutrale	Motivo, Città, Numero, Risultato	Numero + Giornata (identificatore esterno)
Partita rinviata	Una partita che è stata rinviata ad un'altra data	Data, Numero, Risultato	Numero + Giornata (identificatore esterno)
Giornata	Una giornata del campionato	Numero, Girone, Data (Giorno, Mese, Anno)	Numero, Girone
Arbitro	Un arbitro del campionato	Nome, Cognome, Città, Regione	Cognome
Arbitraggio	Associa una partita con il	Arbitro, Partita	

	rispettivo arbitro		
Collocazione	Associa una partita con la rispettiva giornata di campionato. È necessaria per identificare una partita	Partita, Giornata	
Casa	Associa una partita con una squadra: rappresenta la squadra che gioca la partita in casa	Partita, Squadra	
Ospite	Associa una partita con una squadra: rappresenta la squadra che gioca la partita fuori casa	Partita, Squadra	
Posizione	Associa una giornata con una squadra: rappresenta (dando il punteggio) la posizione della squadra dopo ogni giornata	Giornata, Squadra	Punteggio
Appartenenza	Associa una squadra con un giocatore: rappresenta il fatto che un giocatore gioca attualmente con una squadra	Squadra, Giocatore	
Partecipazione	Associa un giocatore con una partita: rappresenta il fatto che un giocatore ha giocato in una partita. Può aver giocato in una posizione diversa dalla sua abitale.	Partita, Giocatore	Posizione

Esercizio 6.8

Modificare lo schema Entità-Relazione in figura 6.37 in maniera da descrivere anche i rapporti passati tra giocatori e squadre con dati di inizio e fine del rapporto e il ruolo principale ricoperto da ogni giocatore in ogni squadra. È possibile che un giocatore abbia diversi rapporti con la stessa squadra in periodi diversi. Per i rapporti in corso si vuole conoscere la data di inizio.

Soluzione:



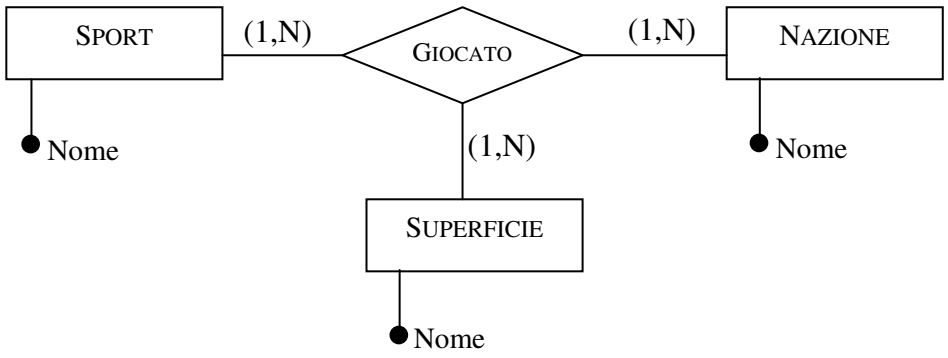
Esercizio 6.9

In ciascuno dei seguenti casi, si fa riferimento a due o più entità definite in uno schema Entità-Relazione e a un concetto che le coinvolge. Specificare i relativi frammenti di schema, definendo i costrutti (una o più relazioni e, se necessario, ulteriori entità con il relativo identificatore) necessari a rappresentare il concetto, mantenendo le entità indicate e introducendo solo gli attributi richiesti esplicitamente.

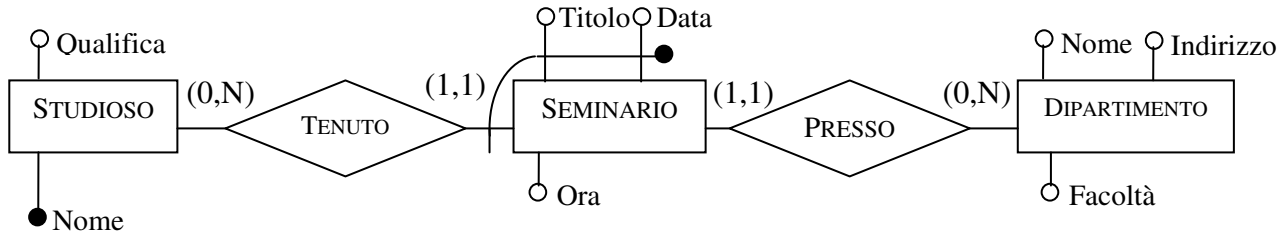
- Entità: Sport, nazione e superficie. Concetto: il fatto che uno sport si pratichi in una certa nazione su una certa superficie (ad esempio, il tennis si gioca sull'erba in Inghilterra e in Australia, sulla terra rossa in Italia e in Francia, sul sintetico in USA, Italia e Francia; il calcio sull'erba in Italia, sul sintetico e sull'erba in USA, sull'erba in Inghilterra).
- Entità: studioso e dipartimento. Concetto: il fatto che lo studioso abbia tenuto seminari presso il dipartimento. Per ogni seminario è necessario rappresentare data, ora e titolo, con il vincolo che uno studioso non possa tenere più seminari nello stesso giorno.
- Entità: professionista e azienda. Concetto: il fatto che il professionista abbia svolto consulenze per l'azienda. È necessario rappresentare il numero di consulenze effettuate dal professionista per ciascuna azienda, con il relativo costo totale.

Soluzione:

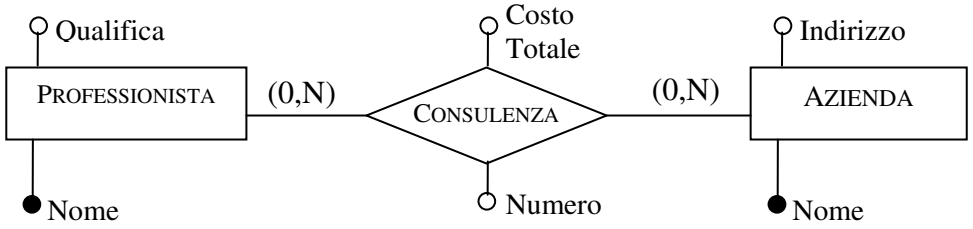
1)



2)



3)



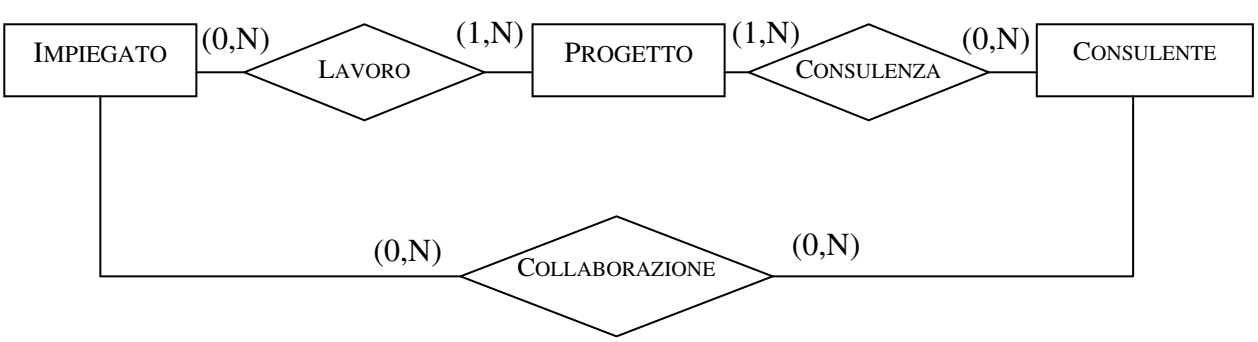
Esercizio 6.10

Si consideri una relazione ternaria che coinvolge le seguenti entità: IMPIEGATO, PROGETTO e CONSULENTE. Indicare in quali dei seguenti casi (e, in caso affermativo, come) è opportuno sostituire a tale relazione due (o tre) relazioni binarie.

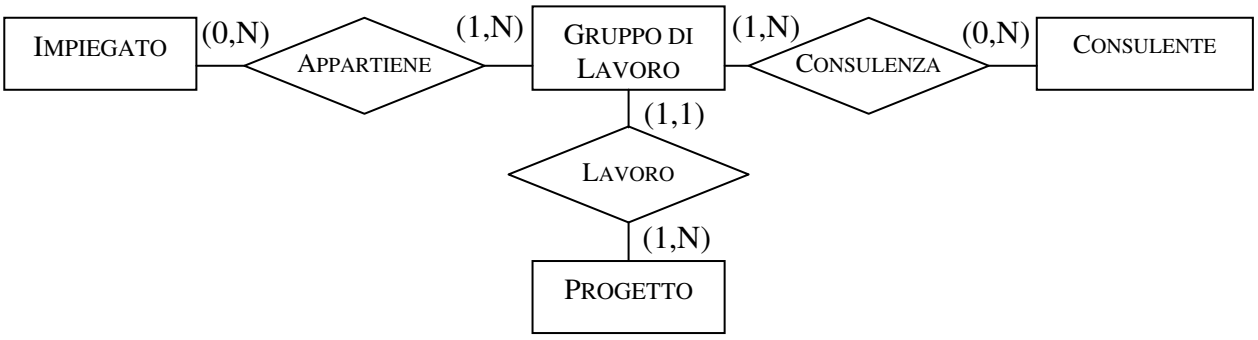
1. Ogni impiegato è coinvolto in zero o più progetti e interagisce con zero o più consulenti. Ogni consulente è coinvolto in zero o più progetti e interagisce con zero o più impiegati. Ogni progetto coinvolge uno o più impiegati e uno o più consulenti (che possono non interagire tra di loro). Un impiegato e un consulente collaborano nell'ambito di un progetto se e solo se essi collaborano fra loro e sono entrambi coinvolti nello stesso progetto.
2. Ogni impiegato è coinvolto in zero o più progetti, in ciascuno dei quali interagisce con uno o più consulenti (che possono essere diversi da progetto a progetto e che possono in generale essere un sottoinsieme dei consulenti coinvolti nel progetto). Ogni consulente è coinvolto in zero o più progetti, in ciascuno dei quali interagisce con uno o più impiegati (che possono essere diversi da progetto a progetto e che possono in generale essere un sottoinsieme degli impiegati coinvolti nel progetto). Ogni progetto coinvolge una o più coppie impiegato-consulente.
3. Ogni impiegato è coinvolto in zero o più progetti. Ogni consulente è coinvolto in zero o più progetti. Ogni progetto coinvolge uno o più impiegati e uno o più consulenti. Un impiegato e un consulente interagiscono se e solo se esiste almeno un progetto in cui siano entrambi coinvolti.

Soluzione:

1)



2)



3)

Esercizio 6.11 Modificare lo schema in figura 6.38 (decomponendo la relation-

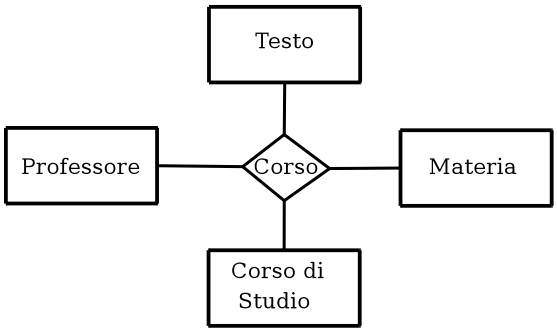


Figura 6.38 Figura di riferimento per l'esercizio 6.11

ship ed aggiungendo ulteriori entità, se necessario; indicare le cardinalità delle relationship e eventuali necessità di identificatori esterni) tenendo conto delle seguenti specifiche:

- per ogni materia possono esistere più corsi, tenuti dallo stesso professore o da professori diversi;
- ogni corso è relativo ad una e una sola materia;
- ogni professore tiene zero o più corsi;
- ogni corso ha uno e un solo professore ed è offerto ad uno e un solo corso di studio;
- per ogni corso di studio esiste al più un corso di una data materia;
- tutti i corsi di una data materia hanno lo stesso libro di testo (uno e uno solo).

Soluzione
La soluzione dell'esercizio è riportata nella figura 6.I.

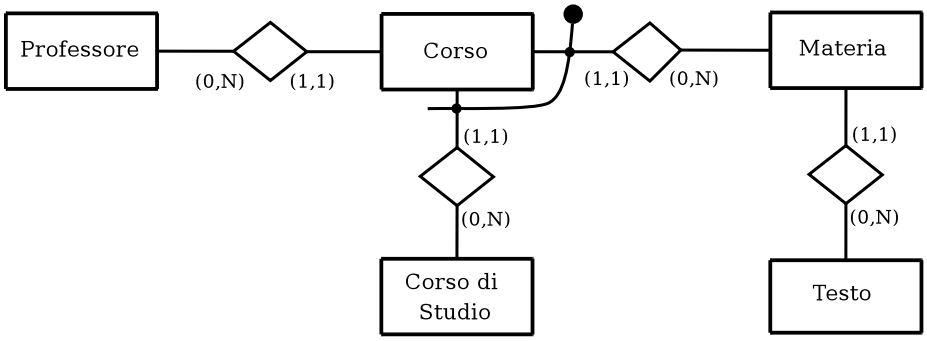


Figura 6.I Figura soluzione dell'esercizio 6.11

Esercizio 6.12 Considerare ancora lo schema in figura 6.38 e modificarlo (decomponendo la relationship ed aggiungendo ulteriori entità, se necessario; indicare le cardinalità delle relationship e eventuali necessità di identificatori esterni) sulla base delle seguenti specifiche:

- per ogni materia possono esistere più corsi, tenuti dallo stesso professore o da professori diversi;
- ogni corso è relativo ad una ed una sola materia;
- ogni professore tiene zero o più corsi;
- ogni corso ha uno o più professori ed è offerto ad uno ed un solo corso di studio;
- per ogni corso di studio esiste al più un corso di una data materia;
- ogni corso ha uno ed un solo libro di testo; i corsi di una data materia non hanno necessariamente lo stesso libro di testo.

Soluzione
La soluzione dell'esercizio è riportata nella figura 6.II.

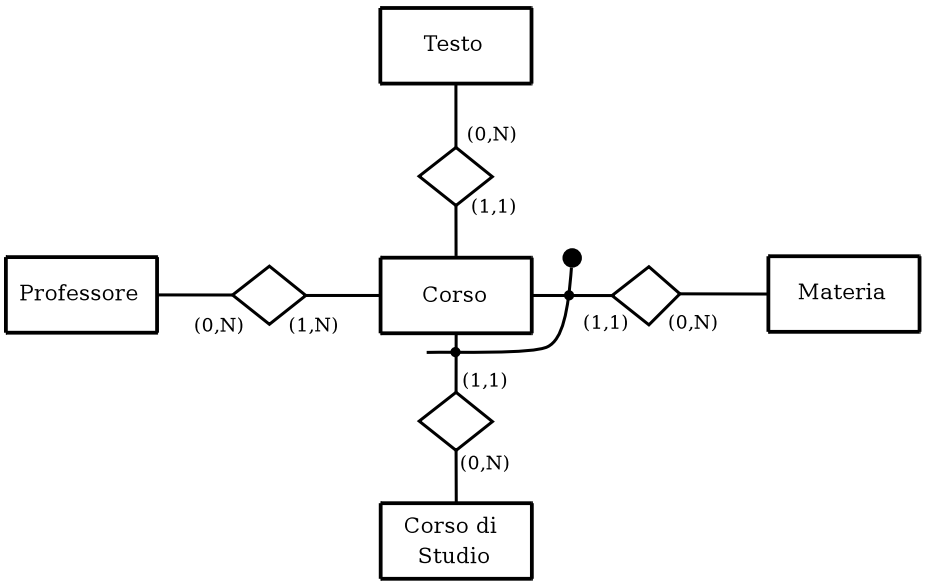


Figura 6.II Figura soluzione dell'esercizio 6.12

Esercizio 6.13 Considerare gli schemi della figura 6.39 e le seguenti specifiche.

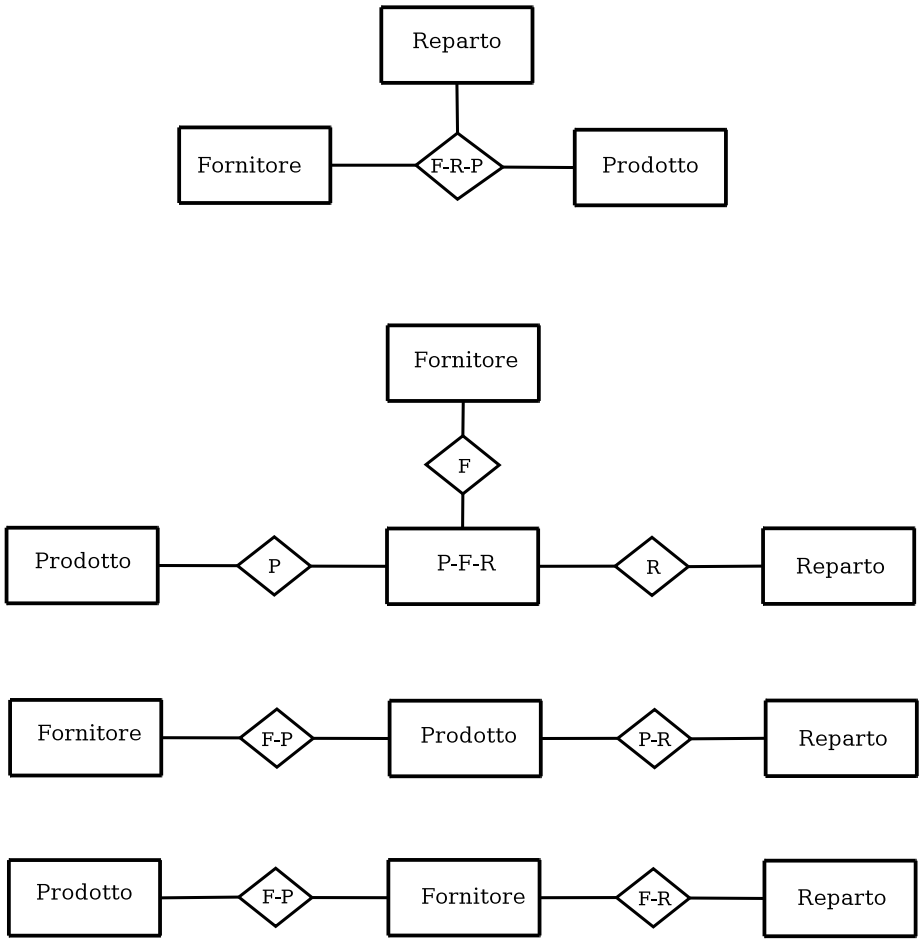


Figura 6.39 Figura di riferimento per l'esercizio 6.13.

Individuare, per ciascuna specifica, lo schema che meglio la descrive, precisando le cardinalità delle relationship e gli eventuali identificatori esterni delle entità, che potrebbero includere anche attributi.

1. Interessano le singole forniture di prodotti ai reparti, avvenute in date specifiche; per ogni data, c'è al più una fornitura di un certo prodotto ad un certo reparto, con un solo fornitore (però in date diverse ci potrebbero essere altre forniture, di altri fornitori).
2. Interesano le singole forniture, avvenute in date specifiche; per ogni data, c'è al più una fornitura di un certo fornitore ad un certo reparto, con un insieme di prodotti (specifico per quella data, e quindi potenzialmente diverso in altre date).

3. Ogni reparto utilizza un certo insieme di prodotti, ognuno dei quali ha uno ed un solo fornitore e può essere utilizzato da più reparti.
4. Ogni reparto ha un insieme di fornitori e utilizza un insieme di prodotti; in generale, un fornitore potrebbe fornire alcuni prodotti ad un reparto e altri prodotti ad altri reparto; un prodotto può essere fornito da più fornitori e utilizzato da diversi reparti.
5. Ogni fornitore dispone di un insieme di prodotti e può rifornire zero o più reparti; ogni reparto ha un insieme di fornitori e da ciascuno di essi può ricevere tutti i prodotti di cui esso dispone; ogni prodotto ha un solo fornitore

Soluzione
La soluzione dell'esercizio è riportata in figura 6.III.

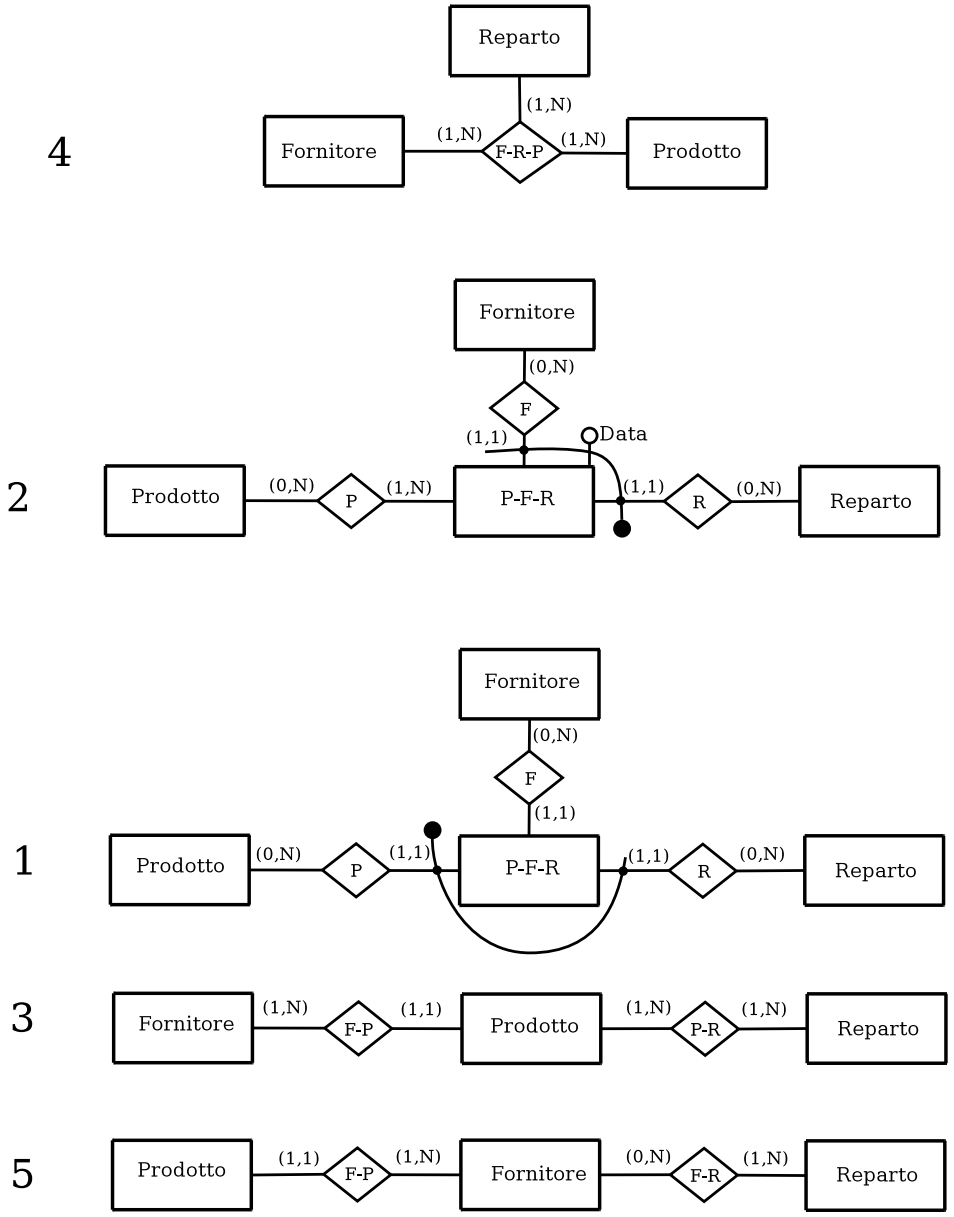


Figura 6.III Figura di riferimento per l'esercizio 6.13

Esercizio 6.14 Rappresentare lo schema Entità relazione in Figura 6.37 con un diagramma delle classi UML.
Soluzione La soluzione di questo esercizio è rappresentata in figura 6.IV.

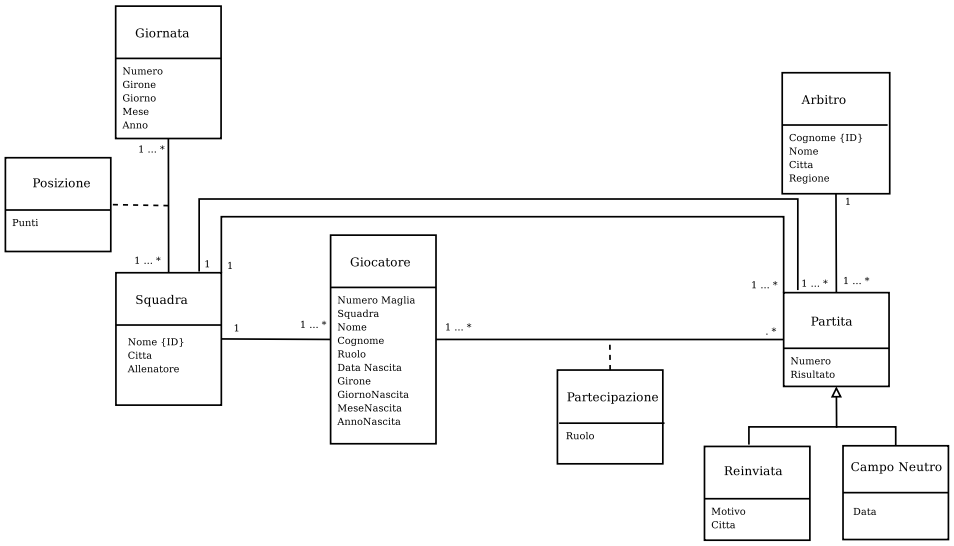


Figura 6.IV Soluzione dell'esercizio 6.14

Esercizio 6.15 Si consideri una relationship ternaria sulle entità impiegato, progetto, consulente. Indicare (con una giustificazione sintetica) in quali dei seguenti casi è opportuno sostituire a tale relationship due (o tre) relationship binarie. In caso affermativo, indicare le entità coinvolte e le relative cardinalità.

1. Ogni impiegato è coinvolto in zero o più progetti. Ogni progetto coinvolge uno o più impiegati e uno e un solo consulente. Un impiegato e un consulente interagiscono se e solo se esiste almeno un progetto in cui siano entrambi coinvolti.
2. Ogni impiegato è coinvolto in zero o più progetti e interagisce con zero o più consulenti. Ogni consulente è coinvolto in zero o più progetti e interagisce con zero o più impiegati. Ogni progetto coinvolge uno o più impiegati e uno o più consulenti (che possono interagire fra loro). Ha senso dire che un impiegato e un consulente collaborano nell'ambito di un progetto se e solo se essi collaborano fra loro e sono entrambi coinvolti nel progetto.
3. Ogni impiegato è coinvolto in zero o più progetti. Ogni consulente è coinvolto in zero o più progetti. Ogni progetto coinvolge uno o più impiegati e uno o più consulenti. Un impiegato e un consulente interagiscono se e solo se esiste almeno un progetto in cui siano entrambi coinvolti, nel qual caso interagiscono in tutti i progetti in cui sono entrambi coinvolti.

Soluzione
Per la soluzione di questo esercizio fare riferimento alla figura 6.V per il primo punto ed alla figura 6.VI per il secondo punto.

Figura 6.V Soluzione dell'esercizio 6.15 punto 1

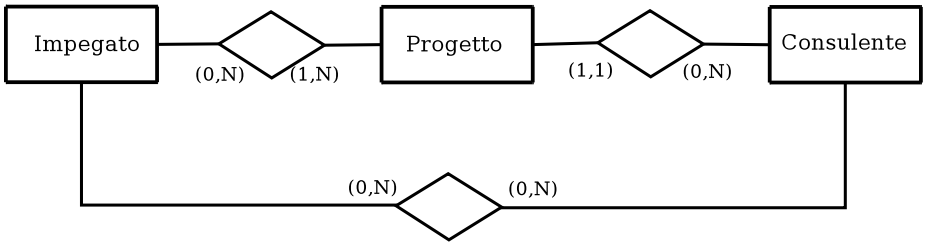


Figura 6.VI Soluzione dell'esercizio 6.15 punto 2

1. Si deve partizionare la relationship ternaria poiché c'è un cambio di cardinalità rispetto al caso n-ario, come si vede in figura 6.V.
2. Si deve partizionare la relationship ternaria poiché occorrono relazioni specifiche sulle varie entità coinvolte, come si vede in figura 6.VI.
3. Non si devono effettuare cambiamenti schema originale.

Esercizio 6.16 Determinare quali delle seguenti affermazioni sul modello ER sono vere e quali sono false:

1. il modello ER è una rappresentazione semplificata del modello relazionale
2. il potere espressivo del modello ER è minore di quello del modello relazionale
3. esistono delle strutture del modello ER che non sono direttamente rappresentabili nel modello relazionale
4. la struttura di una base di dati determinata nella progettazione logica può influire sulle prestazioni
5. nel modello ER non possono esservi cicli di relationship.

Soluzione

1. Falso
2. Falso
3. Vero
4. Vero
5. Falso.

Esercizio 6.17 Determinare quali delle seguenti affermazioni sul modello ER sono vere e quali false:

1. la progettazione di basi di dati attraverso il modello ER dà luogo a schemi relazionali normalizzati
2. una generalizzazione tra due entità può essere sempre rappresentata come una relationship tra le due
3. una relationship tra due entità può essere sempre rappresentata come una generalizzazione opportuna tra le due
4. l'indicazione dei ruoli su una relationship ha influenza sulla cardinalità della stessa
5. non può esistere un identificatore esterno su un'entità collegata ad una relationship con cardinalità diversa da (1,1).

Soluzione

1. Vero
2. Vero
3. Falso
4. Falso
5. Vero.

Esercizio 6.18 Rappresentare attraverso uno schema ER la realtà di interesse seguente.
Un capo di bestiame è identificato univocamente da un numero all'interno di una specifica fattoria. Una fattoria è identificata univocamente da una sigla all'interno di uno specifico allevamento. Ogni allevamento ha un codice univoco ed un nome.
Soluzione
La soluzione di questo esercizio è rappresentata in figura 6.VII.

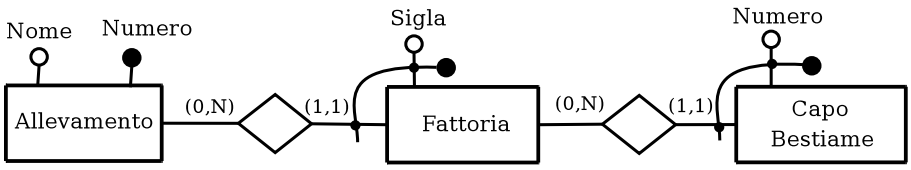


Figura 6.VII Figura di riferimento per l'esercizio 6.18

Esercizio 6.19 Utilizzare il formalismo ER per modellare le seguenti realtà di interesse:

- ogni uomo è padre di altri uomini e figlio di due uomini
- ogni uomo ha molti amici, ciascuno conosciuto in un paese diverso
- ogni uomo è un erede di altri uomini

Soluzione
La soluzione di questo esercizio è rappresentata in figura 6.VIII.

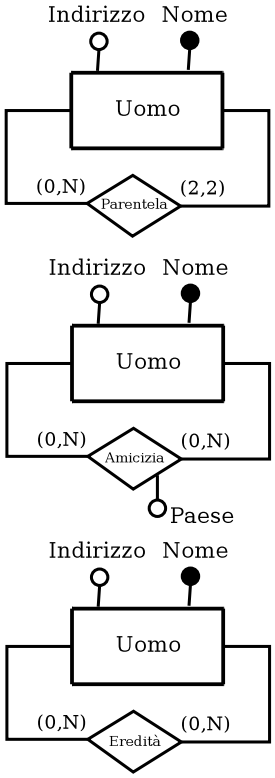


Figura 6.VIII Figura di riferimento per l'esercizio 6.19

Esercizio 6.20 Utilizzare il formalismo ER per modellare una realtà di interesse in cui:

- ogni dipendente è un uomo
- alcuni uomini sono dipendenti
- alcuni dipendenti sono professori
- tutti i professori sono laureati
- alcuni uomini sono laureati.

Soluzione
La soluzione di questo esercizio è rappresentata in figura 6.IX.

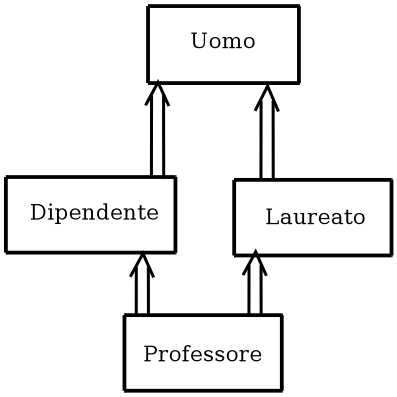
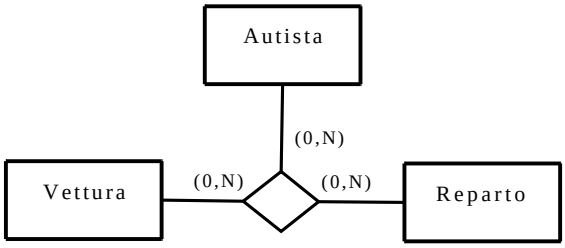


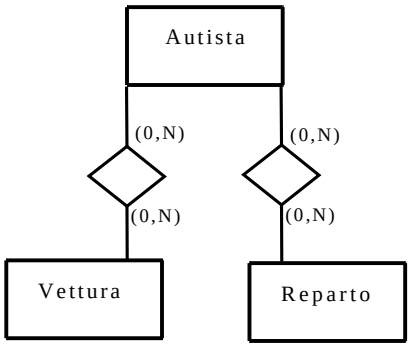
Figura 6.IX Figura di riferimento per l'esercizio 6.20

Esercizio 6.21 Si consideri il primo schema E-R mostrato in figura 6.X, in cui ciascuna istanza (A,V,R) della relationship indica che l'autista A può guidare la vettura V per un'attività di reparto R. Descrivere le condizioni che rendono preferibile il mantenimento della relationship ternaria oppure la sua sostituzione con due relationship come indicato negli altri due schemi della figura 6.X.

1)



2)



3)

Figura 6.X Figura di riferimento per l'esercizio 6.21

Soluzione
La soluzione del primo schema è preferibile alle seconde due quando si vuole tener traccia di quali autisti siano assegnati a quali reparti e con quali autovetture. Non è di interesse, ad esempio, l'assegnazione di un autista ad un reparto se a questo non è stata assegnata ancora una vettura.

La soluzione del secondo schema modella una realtà in cui un autista è assegnato ed un insieme di reparti ed ha assegnate a sua volta un insieme di autovetture. Non è possibile modellare il fatto che un autista utilizza per uno specifico reparto solo un sottoinsieme delle vetture assegnategli.

La terza soluzione modella una realtà in cui ad ogni reparto sono assegnate delle autovetture ed un insieme di autisti. Non è possibile viene modellato il fatto che un autista potrebbe utilizzare unicamente un sottoinsieme delle vetture a disposizione di un reparto.

Essendo i tre schemi diversi tra loro, la scelta su quale utilizzare dipende dalla realtà che si vuole modellare.

Esercizio 6.22 Descrivere lo schema Entità Relazione riportato in figura 6.XI. I nomi delle relazioni sono i seguenti:

- R_1 Assegnamento
- R_2 PostoPrenotazione
- R_3 PostoUtente
- R_4 Composizione
- R_5 FarePrenotazione
- R_6 FareUtilizzo
- R_7 Frequenza
- R_8 Controllo
- R_9 Autorizzazione

Soluzione
La base di dati è relativa alla gestione delle prenotazioni dei posti di lavoro di un laboratorio da parte di studenti.
Ogni studenti è caratterizzato dalla propria matricola, dal nome, cognome, data e luogo di nascita, resisdenza, recapito telefonico.
Gli studenti frequentano alcuni laboratori didattici. I laboratori contengono un insieme di posti di lavoro ed un'insieme di risorse. Ad ogni posto di lavoro si assegnano delle risorse. Alcune risorse possono essere assegnate liberamente agli studenti che frequentano i laboratori, previa autorizzazione. Uno studente può utilizzare un posto di lavoro solamente se effetua una prenotazione. Si tiene traccia di tutte le prenotazioni e di tutte le volte che uno studente utilizza un posto di lavoro. Ogni laboratorio ha un solo responsabile, il quale si occupa solamente di un laboratiorio.

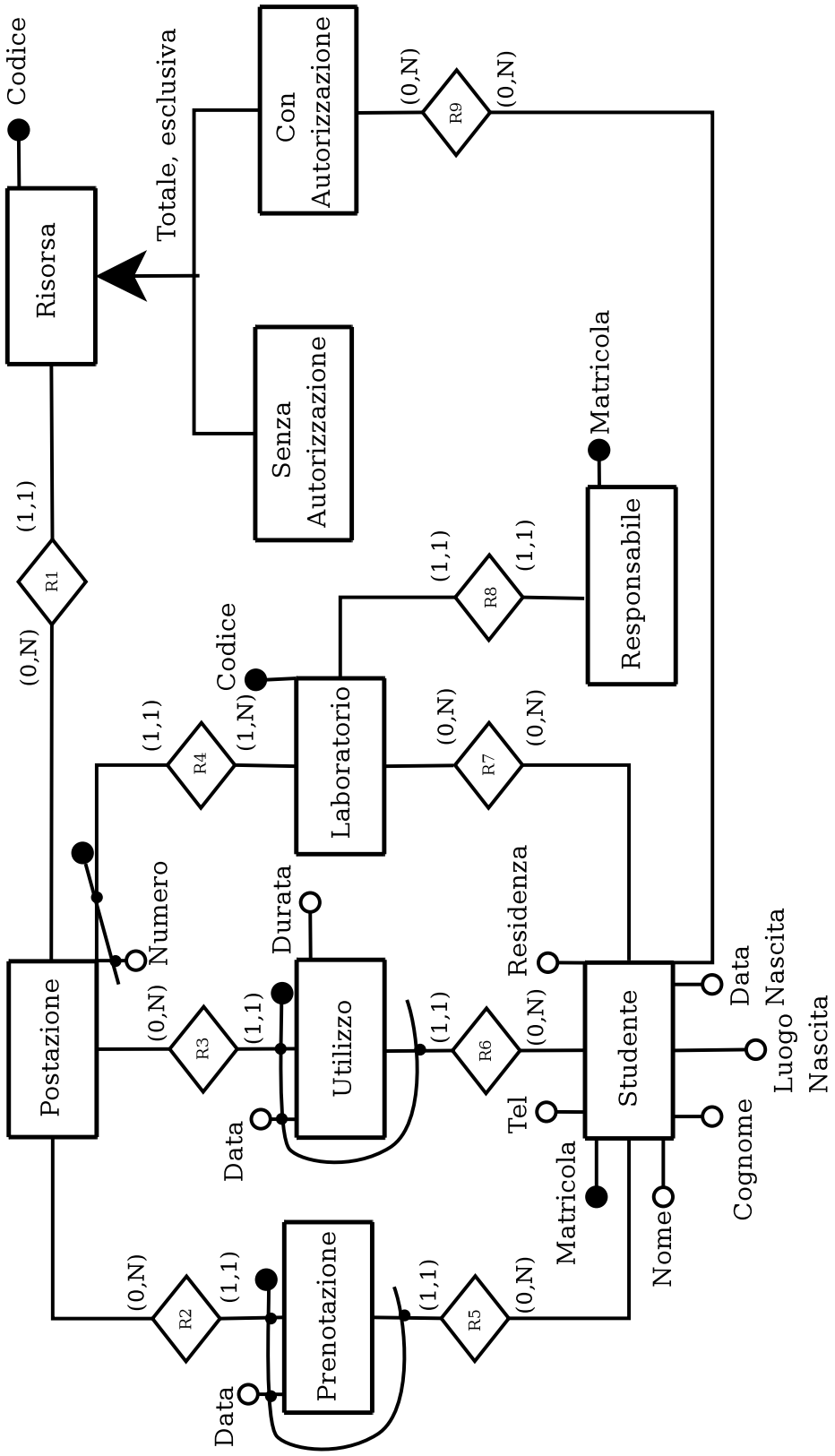


Figura 6.XI Figura soluzione dell'esercizio 6.22.

Capitolo 7

Esercizio 7.1

Si desidera automatizzare il sistema di prestiti di una biblioteca.
Le specifiche del sistema, acquisite attraverso un'intervista con il bibliotecario, sono quelle riportate in figura 7.32. Analizzare tali specifiche, filtrare le ambiguità presenti e poi raggrupparle in modo omogeneo. Prestare particolare attenzione alla differenza esistente tra il concetto di *libro* e di *copia* di libro. Individuare i collegamenti esistenti tra i vari gruppi di specifiche così ottenuti.

Biblioteche
I lettori che frequentano la biblioteca hanno una tessera su cui è scritto il nome e l'indirizzo ed effettuano richieste di prestito per i libri che sono catalogati nella biblioteca. I libri hanno un titolo, una lista di autori e possono esistere in diverse copie. Tutti i libri contenuti nella biblioteca sono identificati da un codice. A seguito di una richiesta viene dapprima consultato l'archivio dei libri disponibili (cioè non in prestito). Se il libro è disponibile, si procede alla ricerca del volume negli scaffali; il testo viene poi classificato come in prestito. Acquisito il volume, viene consegnato al lettore, che procede alla consultazione. Terminata la consultazione, il libro viene restituito, reinserito in biblioteca e nuovamente classificato come disponibile. Per un prestito si tiene nota degli orari e delle date di acquisizione e di riconsegna.

Figura 7.32 Specifiche per l'esercizio 7.1

Soluzione:

Termine	Descrizione	Sinonimo	Collegamenti
Lettore	Una persona che prende in prestito libri dalla biblioteca	Utente	Copia, Prestito
Libro	Tipo di libro presente in biblioteca. La biblioteca ha una o più copie di uno stesso libro.		Copia
Copia	Ogni copia di un libro presente in biblioteca. Può essere prestato a un lettore.	Libro, Testo, Volume	Libro, Lettore, Prestito
Prestito	Un prestito fatto a un lettore: ogni prestito si riferisce ad una copia di un libro.		Lettore, Copia

FRASI RELATIVE AI LETTORI:

I lettori che frequentano la biblioteca hanno una tessera su cui è scritto il nome e l'indirizzo ed effettuano richieste di prestito per i libri che sono catalogati nella biblioteca.

FRASI RELATIVE AI LIBRI:

I libri hanno un titolo, una lista di autori e possono esistere in diverse copie.

FRASI RELATIVE ALLE COPIE:

Tutti i libri contenuti nella biblioteca sono identificati da un codice.
A seguito di una richiesta viene dapprima consultato l'archivio dei libri disponibili (cioè non in prestito).
Se il libro è disponibile, si procede alla ricerca del volume negli scaffali;

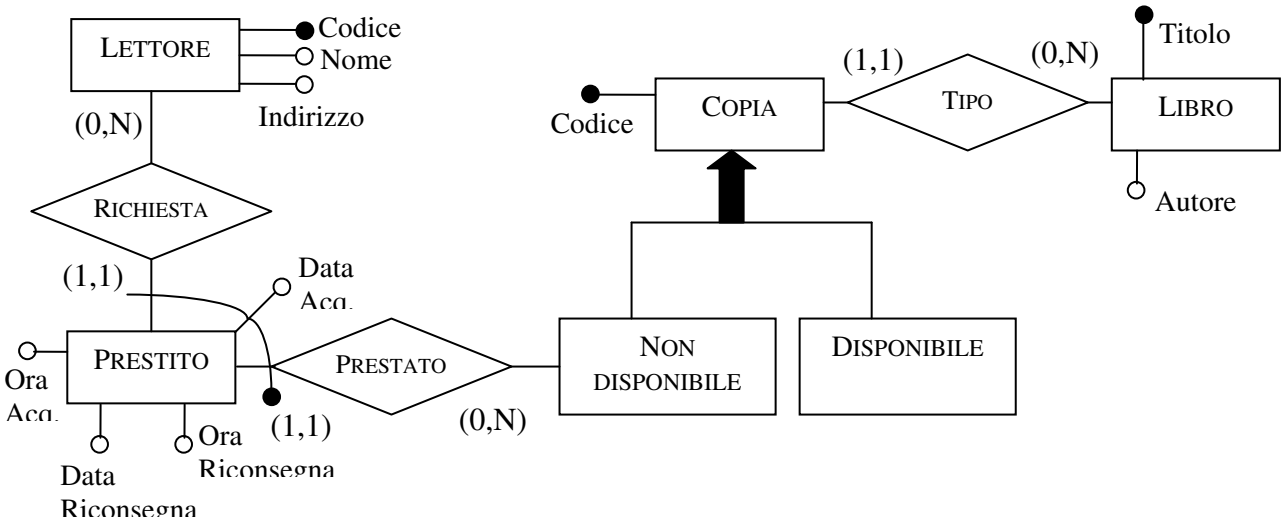
FRASI RELATIVE AI PRESTITI:

Acquisito il volume, viene consegnato al lettore, che procede alla consultazione.
il testo viene poi classificato come in prestito.
Per un prestito si tiene nota degli orari e delle date di acquisizione e di riconsegna.

Esercizio 7.2

Rappresentare le specifiche dell'esercizio precedente (dopo la fase di riorganizzazione) con uno schema del modello Entità-Relazione.

Soluzione:



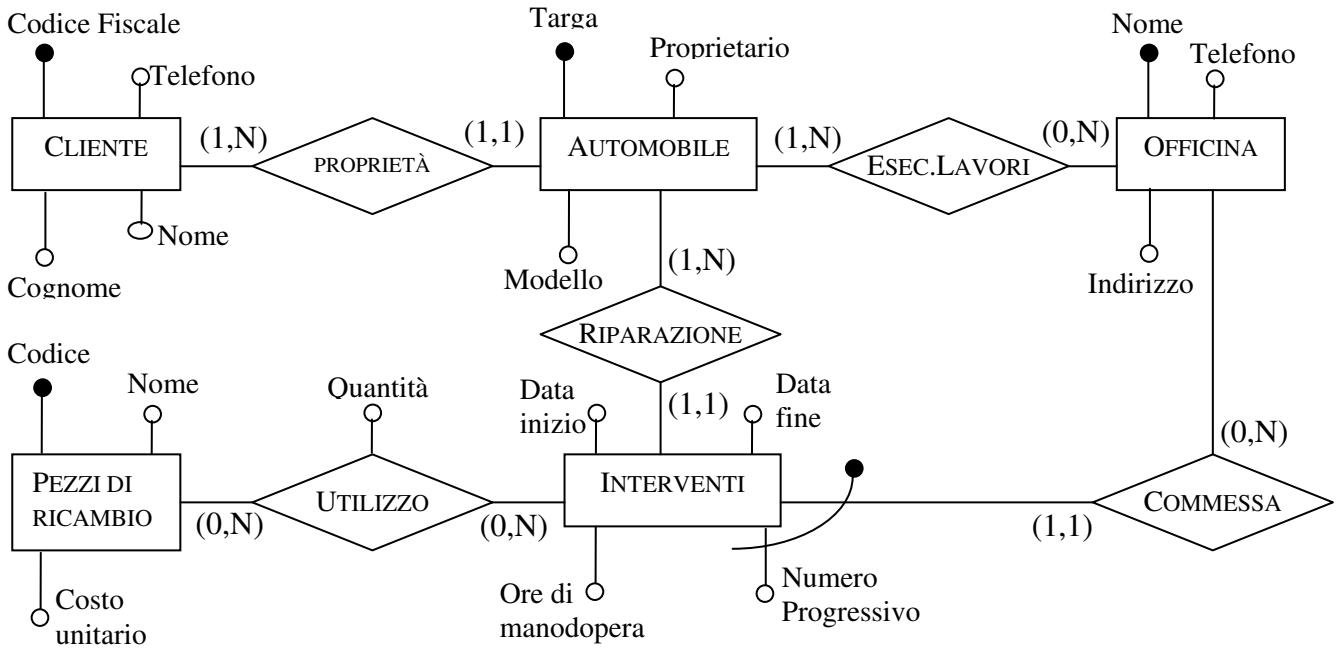
Esercizio 7.3

Definire uno schema Entità-Relazione che descriva i dati di una applicazione relativa a una catena di officine. Sono di interesse le seguenti informazioni.

- Le officine, con nome (identificante), indirizzo e telefono.
- Le automobili, con targa (identificante) e modello (una stringa di caratteri senza ulteriore struttura) e proprietario.
- I clienti (proprietari di automobili), con codice fiscale, cognome, nome e telefono. Ogni cliente può essere proprietario di più automobili.
- Gli "interventi" di manutenzione, ognuno effettuato presso un'officina e con numero progressivo (unico nell'ambito della rispettiva officina), date di inizio e di fine, pezzi di ricambio utilizzati (con le rispettive quantità) e numero di ore di manodopera.
- I pezzi di ricambio, con codice, nome e costo unitario.

Indicare le cardinalità delle relazioni e (almeno) un identificatore per ciascuna entità.

Soluzione:



Esercizio 7.4

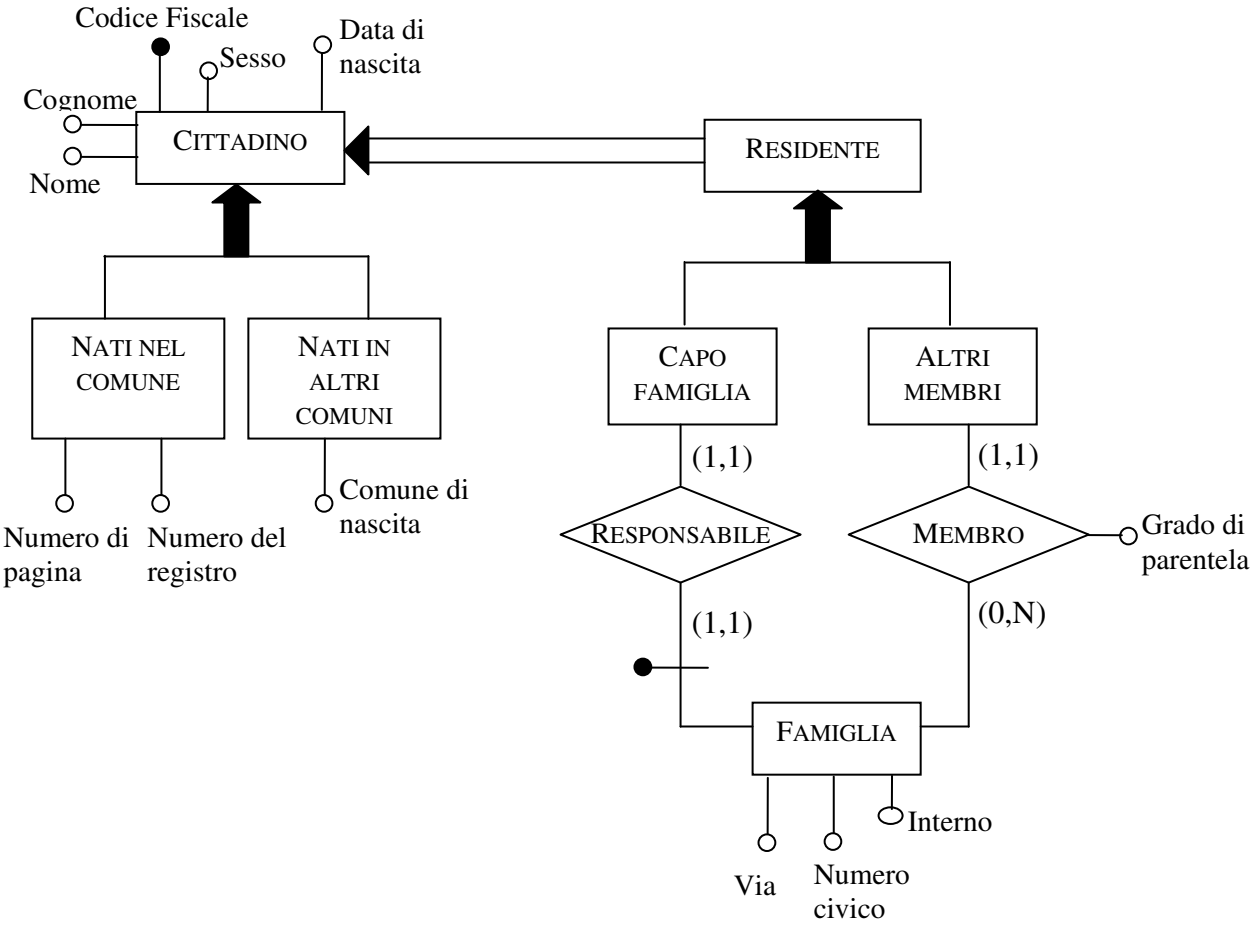
Definire uno schema E-R che descriva i dati di una applicazione relativa all'anagrafe del comune di Chissadove, con cittadini e famiglie. Vanno memorizzate:

- Informazioni sui cittadini nati nel comune e su quelli residenti in esso; ogni cittadino è identificato dal codice fiscale e ha cognome, nome, sesso e data di nascita; inoltre:
  - Per i nati nel comune, sono registrati anche gli estremi di registrazione (numero del registro e pagina)
  - Per i nati in altri comuni, è registrato il comune di nascita

- Informazioni sulle famiglie residenti, ognuna delle quali ha uno e un solo capofamiglia e zero o più membri, per ognuno dei quali è indicato (con la sigla) il grado di parentela (coniuge, figlio, genitore o altro); ogni cittadino residente appartiene ad una e una sola famiglia; tutti i membri di una famiglia hanno lo stesso domicilio (via, numero civico, interno)

Cercare di procedere secondo la strategia inside-out. Al termine, verificare le qualità dello schema ottenuto.

Soluzione:



Usando la strategia inside-out , la creazione di questo schema parte con l'entità CITTADINO; attorno ad esso, possiamo aggiungere le due specializzazioni e le specializzazioni di RESIDENTE (questa specializzazione rappresenta i residenti, ovunque siano nati).
Infine, possiamo aggiungere l'entità FAMIGLIA con le due relazioni Responsabile e Membro.
Questo schema è corretto e completo perché rappresenta tutte le specifiche con i costrutti corretti.

Esercizio 7.5

Analizzare le specifiche relative a partite di un campionato di calcio riportate in figura 7.33 e costruite un glossario dei termini ad esse relativo.

Campionato di calcio
Per ogni partita, descrivere il girone e la giornata in cui si è svolta, il numero progressivo nella giornata (es. prima partita, seconda partita, ecc), la data, con giorno , mese e anno, le squadre coinvolte nella partita, con nome, città della squadra e allenatore, e infine per ciascuna squadra se ha giocato in casa. Si vogliono conoscere i giocatori che giocano in ogni squadra con i loro nomi e cognomi, la loro data di nascita e il loro ruolo principale. Si vuole conoscere, per ogni partita, i giocatori che hanno giocato, i ruoli di ogni giocatore (i ruoli dei giocatori possono cambiare di partita in partita) e nome, cognome, città e regione di nascita dell'arbitro della partita. Distinguere le partite giocate regolarmente da quelle rinviate. Per quelle rinviate, rappresentare la data in cui si sono effettivamente giocate. Distinguere anche le partite giocate in una città diversa da quella della squadra ospitante; per queste si vuole rappresentare la città in cui si svolgono, nonché il motivo della variazione di sede. Dei giocatori interessa anche la città di nascita.

Figura 7.33 Specifiche per l'esercizio 7.5

Soluzione:

Glossario dei termini

Termine	Descrizione	Sinonimo	Collegamento
Partita	Una partita giocata nel torneo; può essere rinviata o giocata in campo neutrale		Giocatore, Squadra, Giornata, Arbitro
Giornata	In una giornata si giocano molte partite. Ogni giornata ha la sua data (giorno, mese e anno)		Partita, Squadra
Squadra	Una squadra che gioca nel campionato		Giocatore, Partita, Giornata
Giocatore	Un giocatore che gioca in una squadra; è importante conoscere in quali partite ha giocato ed in quali posizioni		Squadra, Partita
Arbitro	Un arbitro che arbitra una partita del campionato		Partita

Esercizio 7.6

Dopo aver riorganizzato in gruppi omogenei le specifiche dell'esercizio precedente, rappresentarle con il modello Entità-Relazione, procedendo in maniera top-down per livelli di astrazione successiva a partire da uno schema scheletro iniziale. Si osservi che lo schema in figura 6.28 rappresenta una possibile soluzione di questo esercizio.

Soluzione:

FRASI RELATIVE ALLA PARTITA E ALLA GIORNATA

Per ogni partita, descrivere il girone e la giornata in cui si è svolta, il numero progressivo nella giornata (es. prima partita, seconda partita, ecc), la data, con giorno, mese e anno.
Distinguere le partite giocate regolarmente da quelle rinviate. Per quelle rinviate, rappresentare la data in cui si sono effettivamente giocate
Distinguere anche le partite giocate in una città diversa da quella della squadra ospitante; per queste si vuole rappresentare la città in cui si svolgono, nonché il motivo della variazione di sede

FRASI RELATIVE ALL'ARBITRO

Si vuole conoscere, per ogni partita, nome, cognome, città e regione di nascita dell'arbitro della partita

FRASI RELATIVE ALLE SQUADRE

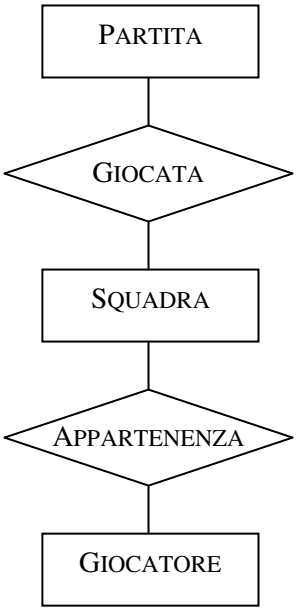
Per ogni partita, descrivere le squadre coinvolte nella partita, con nome, città della squadra e allenatore, e infine per ciascuna squadra se ha giocato in casa.
Memorizziamo, per ogni giornata, quanti punti ha ogni squadra.

FRASI RELATIVE AI GIOCATORI

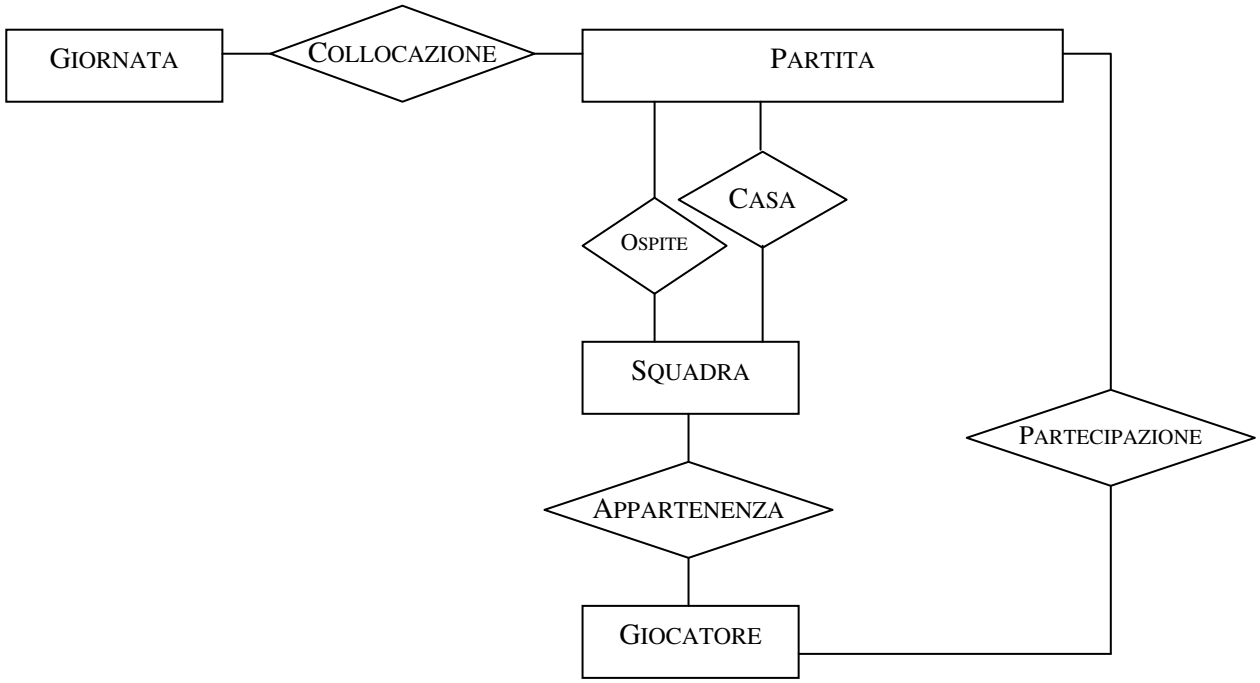
Si vogliono conoscere i giocatori che giocano in ogni squadra con i loro nomi e cognomi, la loro data di nascita e il loro ruolo principale. Si vogliono conoscere, per ogni partita, i giocatori che hanno giocato, i ruoli di ogni giocatore (i ruoli dei giocatori possono cambiare di partita in partita).
Per ogni giocatore siamo interessati alla città di nascita.

I seguenti schemi rappresentano i passi per la costruzione dello schema finale, usando la strategia top-down.

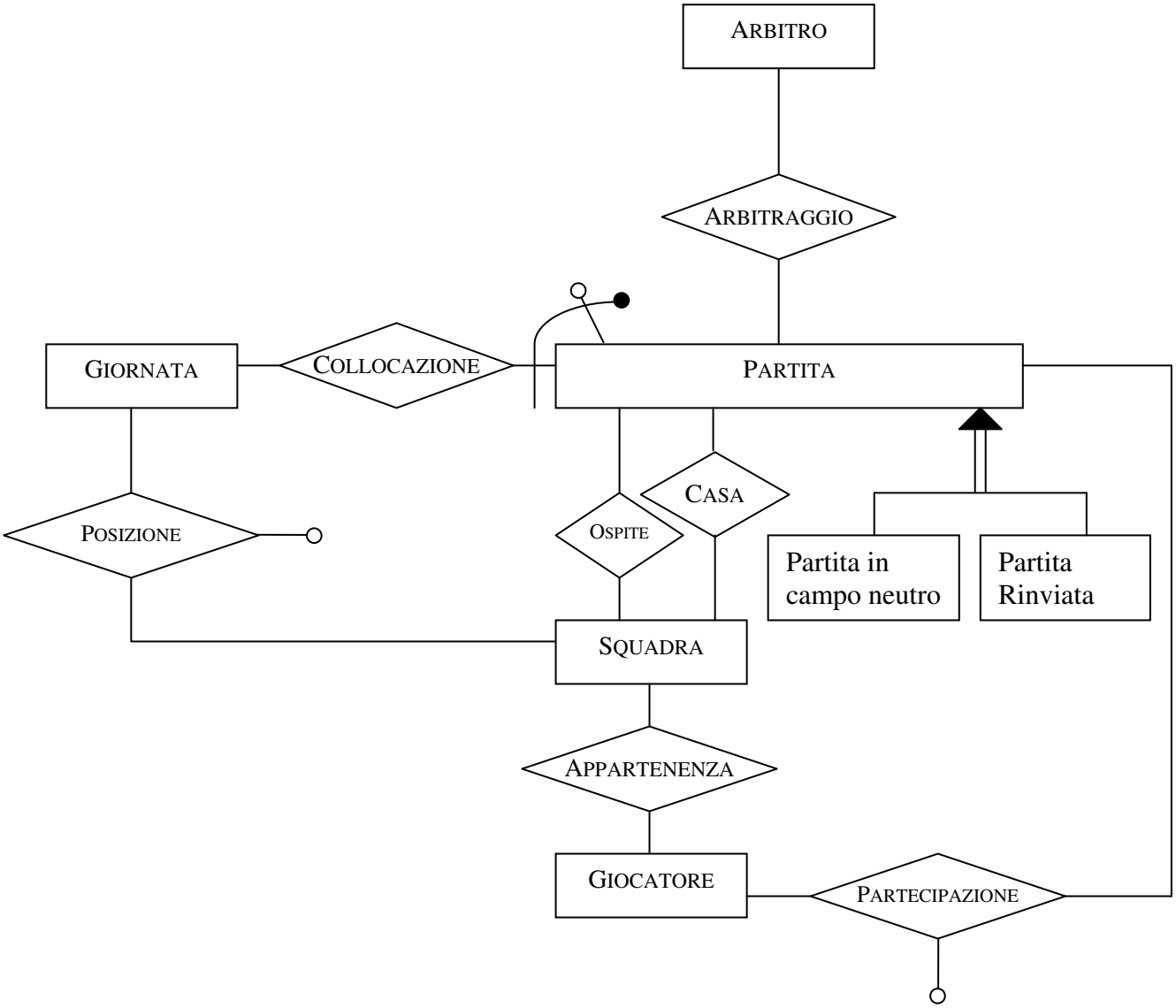
1) Schema Skeleton



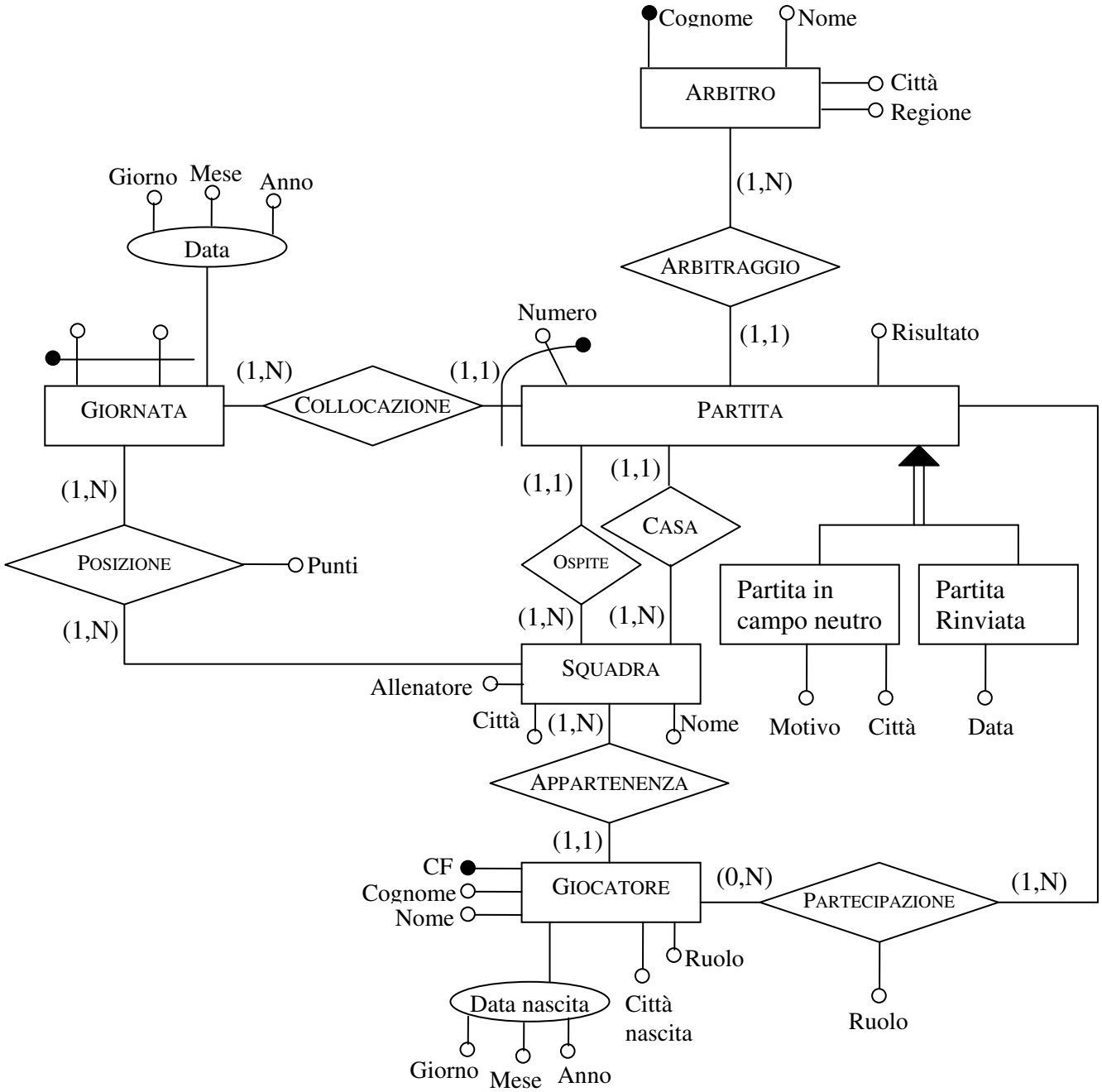
2)



3)


ARBITRO
ARBITRAGGIO
GIORNATA
COLLOCAZIONE
PARTITA
POSIZIONE
CASA
OSPITE
Partita in campo neutro
Partita Rinviata
SQUADRA
APPARTENENZA
GIOCATORE
PARTECIPAZIONE

4)


Cognome
Nome
ARBITRO
Città
Regione
(1,N)
ARBITRAGGIO
(1,1)
Risultato
Giorno
Mese
Anno
Data
Numero
(1,N)
(1,1)
GIORNATA
COLLOCAZIONE
PARTITA
(1,N)
(1,1)
(1,1)
CASA
OSPITE
POSIZIONE
Punti
(1,N)
(1,N)
(1,N)
Partita in campo neutro
Partita Rinviata
SQUADRA
Allenatore
Città
(1,N)
Nome
Motivo
Città
Data
APPARTENENZA
(1,1)
CF
Cognome
Nome
GIOCATORE
(0,N)
PARTECIPAZIONE
(1,N)
Ruolo
Data nascita
Città nascita
Ruolo
Giorno
Mese
Anno

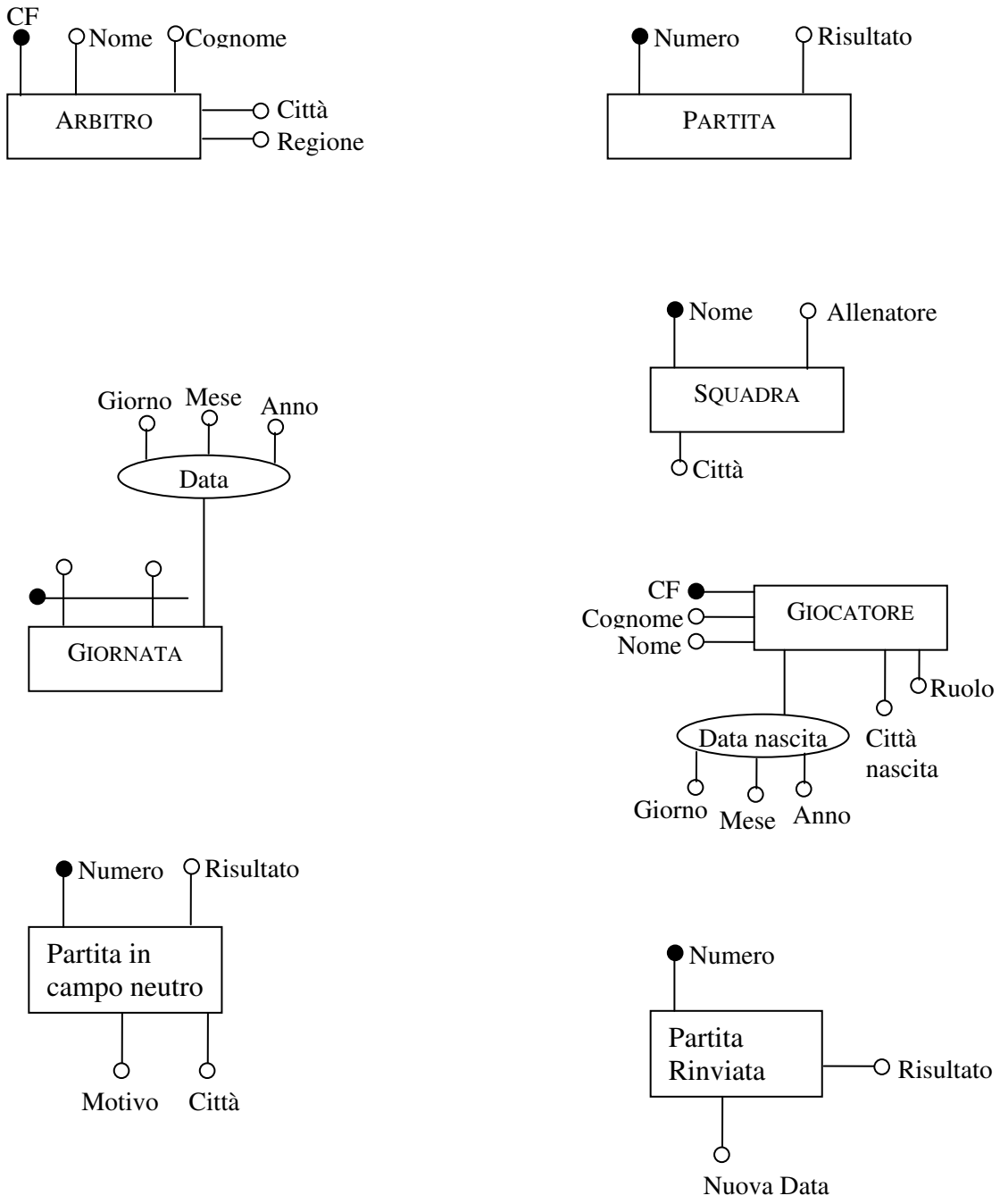
Esercizio 7.7

Provare a rappresentare di nuovo le specifiche dell'esercizio 7.5 con uno schema Entità-Relazione, procedendo però in maniera bottom-up: costruire frammenti di schema separati che descrivono le varie componenti omogenee delle specifiche e poi procedere per integrazione dei vari schemi. Confrontare il risultato con lo schema ottenuto nell'esercizio 7.6.

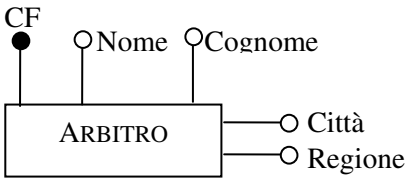
Soluzione:

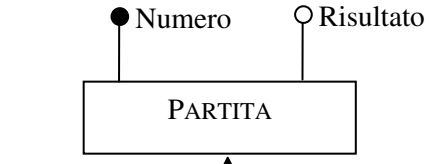
I passi seguenti rappresentano lo schema finale utilizzando la strategia bottom-up.
Gli schemi così ottenuti sono simili a quelli dell'esercizio precedente.

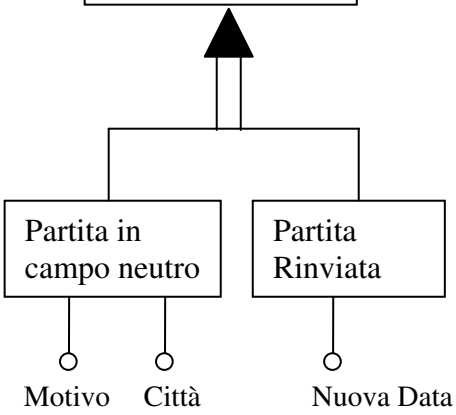
1)

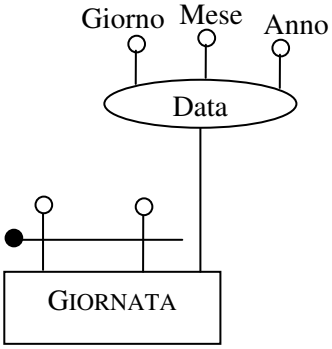


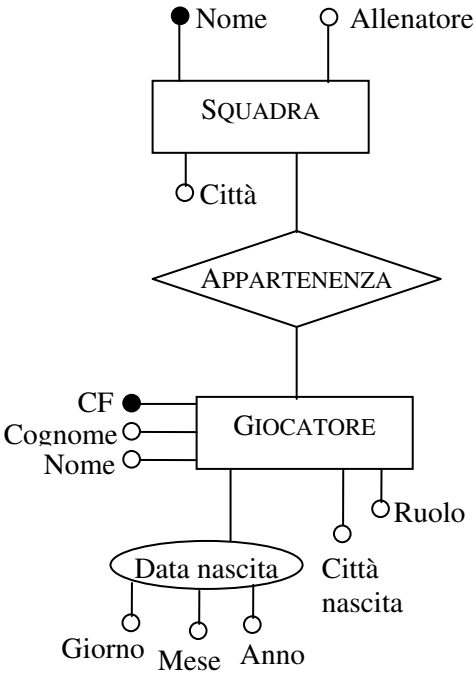
2)



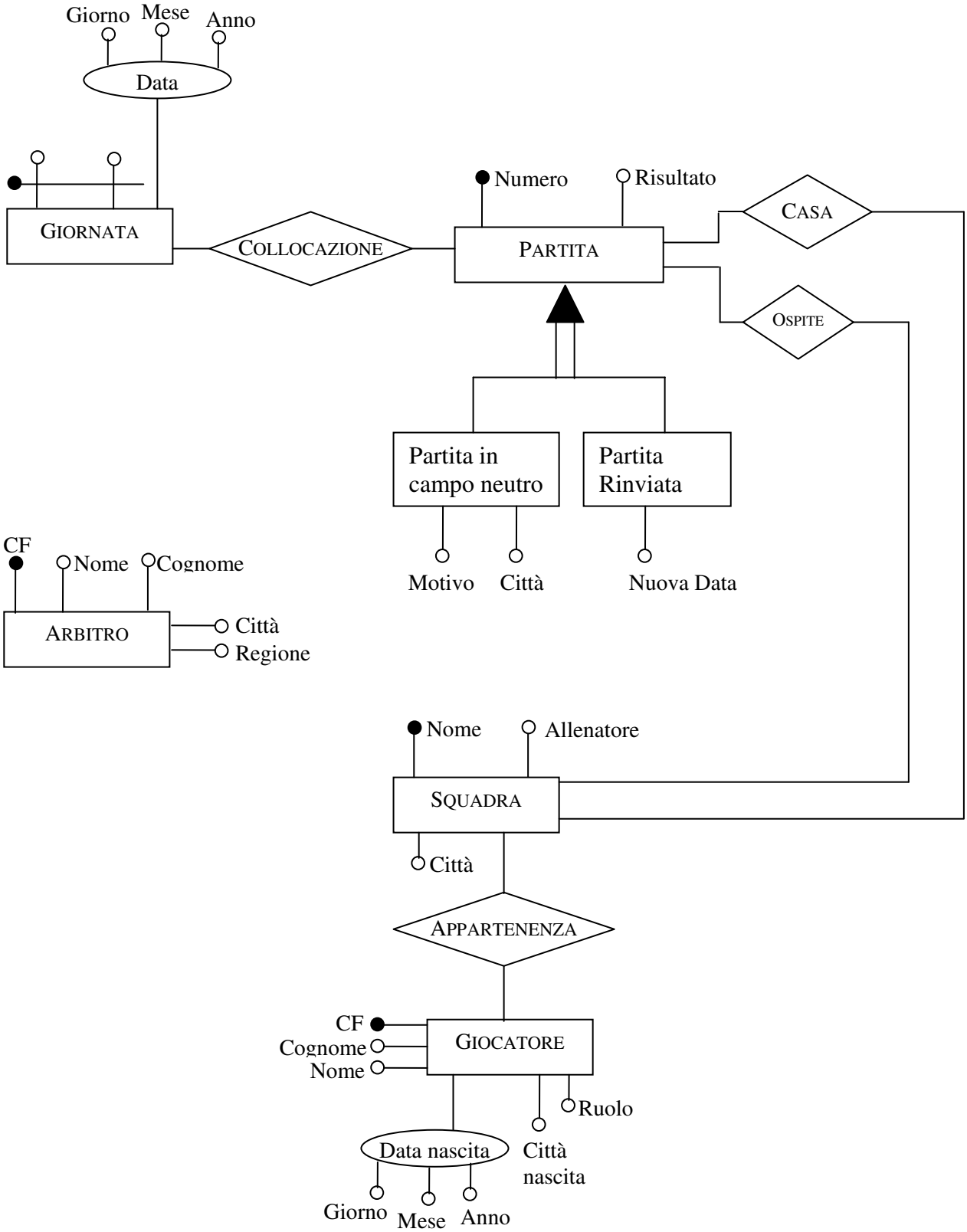




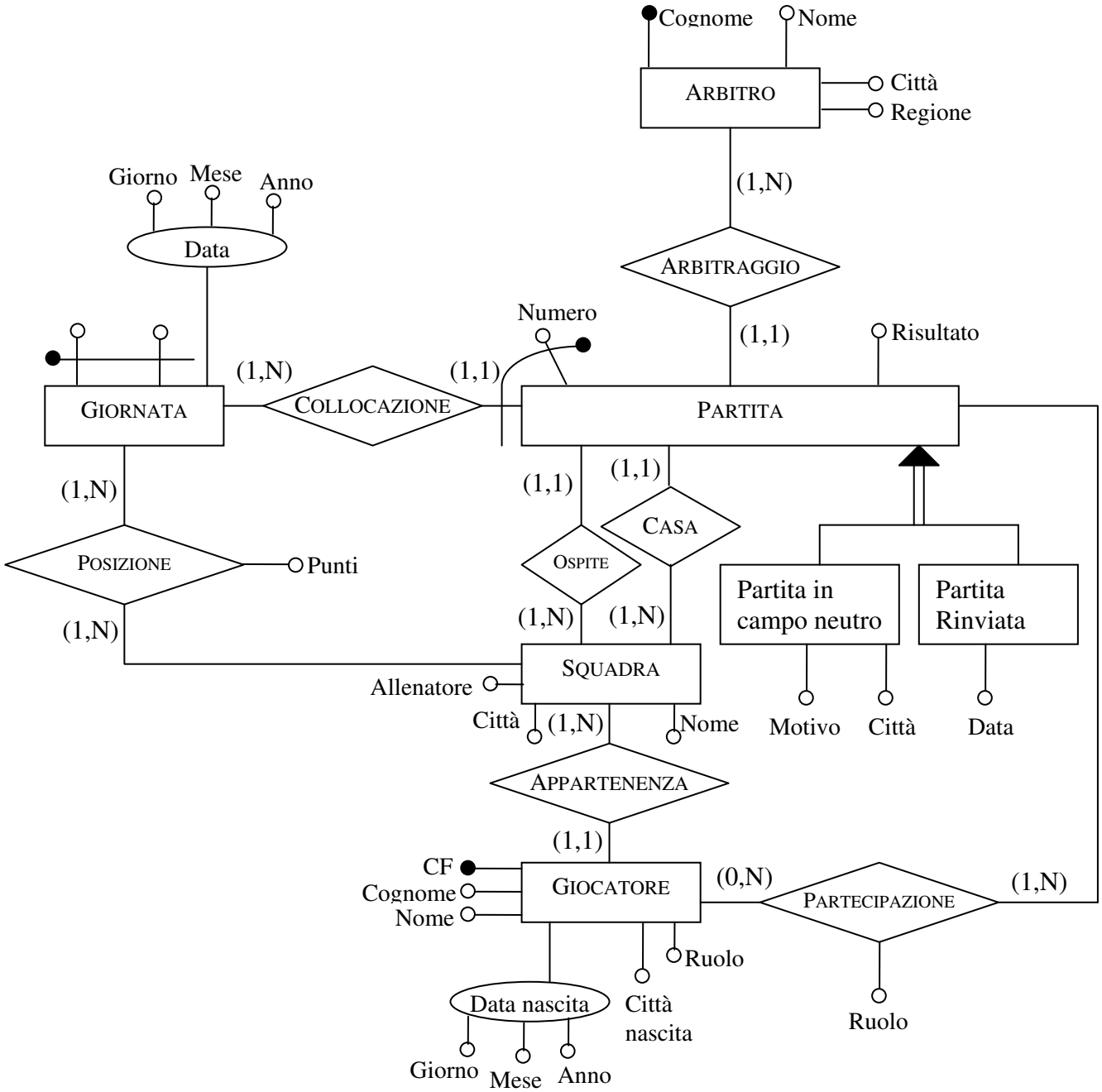




3)


Giorno
Mese
Anno
Data
GIORNATA
COLLOCAZIONE
Numero
Risultato
PARTITA
CASA
OSPITE
Partita in campo neutro
Partita Rinviata
Motivo
Città
Nuova Data
CF
Nome
Cognome
ARBITRO
Città
Regione
Nome
Allenatore
SQUADRA
Città
APPARTENENZA
CF
Cognome
Nome
GIOCATORE
Ruolo
Data nascita
Città nascita
Giorno
Mese
Anno

4)


Cognome
Nome
ARBITRO
Città
Regione
(1,N)
ARBITRAGGIO
(1,1)
Risultato
Giorno
Mese
Anno
Data
Numero
(1,N)
(1,1)
GIORNATA
COLLOCAZIONE
PARTITA
(1,N)
(1,1)
(1,1)
CASA
OSPITE
POSIZIONE
Punti
(1,N)
(1,N)
(1,N)
Partita in campo neutro
Partita Rinviata
SQUADRA
Allenatore
Città
(1,N)
Nome
Motivo
Città
Data
APPARTENENZA
(1,1)
CF
Cognome
Nome
GIOCATORE
(0,N)
PARTECIPAZIONE
(1,N)
Ruolo
Data nascita
Città nascita
Ruolo
Giorno
Mese
Anno

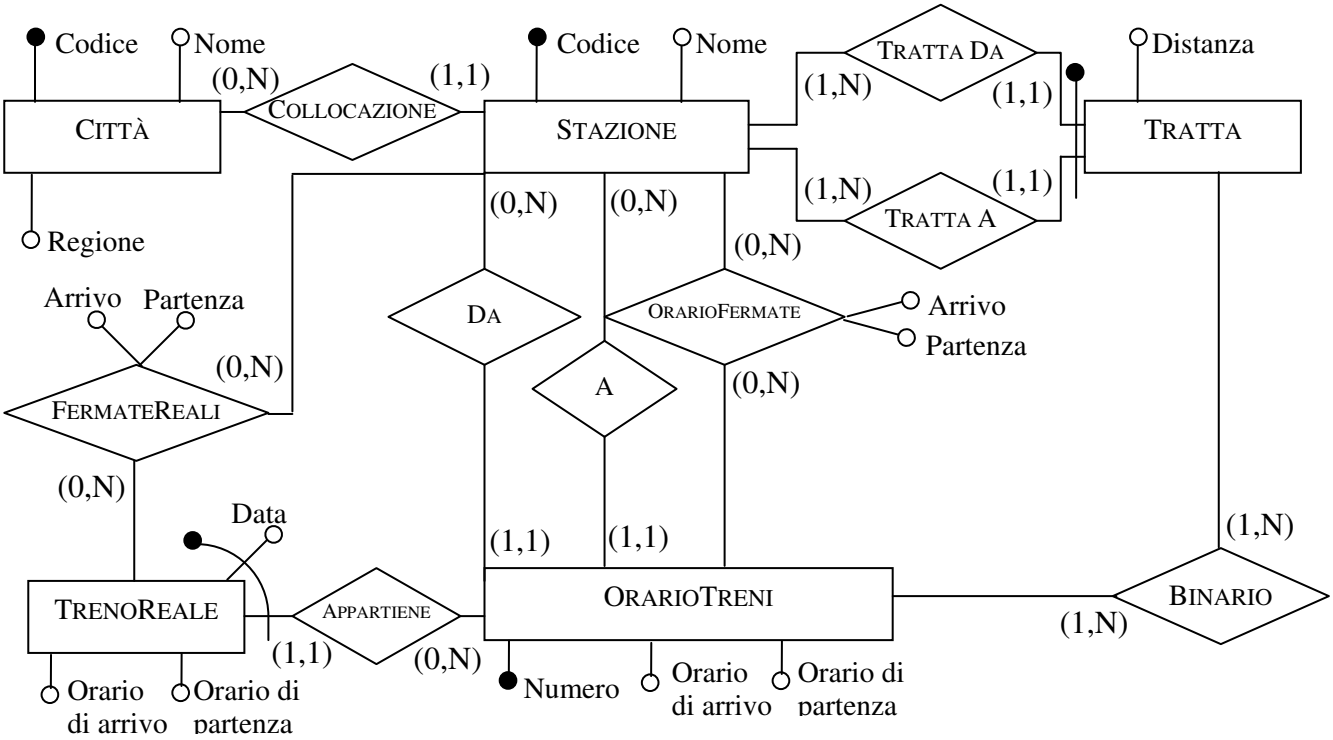
Esercizio 7.8

Si vuole effettuare una operazione di re*verse-engeneering*, ovvero si vuole ricostruire, a partire da una base di dati relazionale, una sua rappresentazione concettuale con il modello Entità-Relazione. La base di dati è relativa a una applicazione su treni e stazioni ferroviarie ed è composta dalle seguenti relazioni:

- STAZIONE(Codice, Nome, Città), con il vincolo di integrità referenziale fra l'attributo Città e la relazione CITTÀ;
- CITTÀ(Codice, Nome, Regione);
- TRATTA(Da, A,Distanza) con i vincoli di integrità referenziale tra l'attributo Da e la relazione STAZIONE e tra l'attributo A e la relazione STAZIONE; questa relazione contiene tutte e sole le coppie di stazioni connesse da una linea in modo diretto (cioè senza stazioni intermedie);
- ORARIOTRENI(Numero, Da, A, OrarioDiPartenza, OrarioDiArrivo) con vincoli di integrità referenziale tra l'attributo Da e la relazione STAZIONE e tra l'attributo A e la relazione STAZIONE;
- TRATTETRENO(NumeroTreno, Da, A) con vincoli di integrità referenziale tra l'attributo NumeroTreno e la relazione ORARIOTRENI e tra gli attributi Da e A e la relazione TRATTA;
- ORARIOFERMATE(NumeroTreno, Stazione, Arrivo, Partenza) con il vincolo di integrità referenziale tra l'attributo numero treno e la relazione OrarioTreni e tra l'attributo Stazione e la relazione STAZIONE;
- TRENOREALE(Numero, Data, OrarioDiPartenza, OrarioDiArrivo) con il vincolo di integrità referenziale tra l'attributo Numerto e la relazione ORARIOTRENI;
- FERMATEREALI(NumeroTreno, Data, Stazione, Arrivo, Partenza) con il vincolo di integrità referenziale tra gli attributi NumeroTreno e Stazione e la relazione ORARIOFERMATE.

Segnalare eventuali ridondanze. In particolare, qualora si tratti di relazioni derivate.

Soluzione:



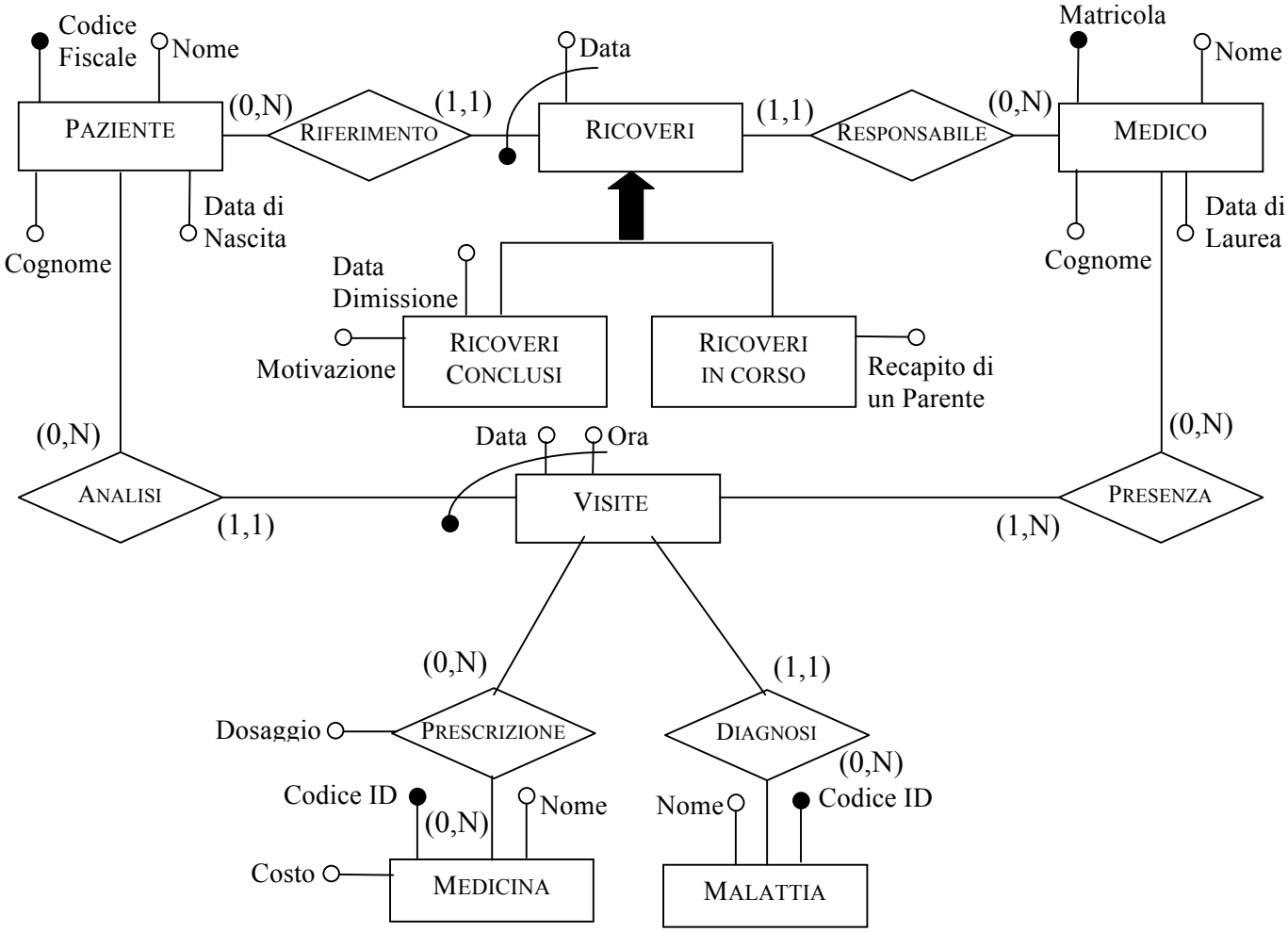
La relazione BINARIO contiene informazioni ridondanti, perché il binario può essere ottenuto dalle relazioni Da, A, OrarioFermate. Comunque questa informazione potrebbe essere un po' più difficile, e le informazioni sui binari potrebbero essere richieste spesso nel database; così la ridondanza è utile per migliorare le prestazioni, inoltre il concetto di tratta è importante in questo contesto, e così è corretto rappresentarlo nello schema E-R.

Esercizio 7.9

Definire uno schema Entità-Relazione che descriva i dati di una applicazione relativa ad un reparto ospedaliero. Sono di interesse le seguenti informazioni.

- I pazienti, con codice fiscale, nome, cognome e data di nascita.
- I ricoveri dei pazienti, ognuno con data di inizio (identificante nell'ambito dei ricoveri di ciascun paziente) e medico curante; inoltre, per i ricoveri conclusi, la data di conclusione e la motivazione (dimissione, trasferimento, etc.), e, per i ricoveri in corso, il recapito di un parente (che si può assumere sia semplicemente una stringa)
- I medici, con numero di matricola, cognome, nome e data di laurea.
- Le visite, con la data, l'ora, i medici visitanti, le medicine prescritte (con le relative quantità) e le malattie diagnosticate; ogni visita è identificata dal paziente coinvolto, dalla data e dall'ora.
- Per ogni medicina sono rilevanti un codice identificativo, un nome e un costo.
- Per ogni malattia sono rilevanti un codice identificativo e un nome.

Soluzione:



Esercizio 7.10

Definire uno schema Entità-Relazione che descriva i dati di una applicazione relativa all'archivio di un amministratore di condomini, secondo le seguenti specifiche (semplificate rispetto a molte realtà).

- Ogni condominio ha un nome (che lo identifica) e un indirizzo e comprende una o più *scale*, ognuna delle quali comprende un insieme di appartamenti.
- Se il condominio comprende più scale, ad ogni scala sono associati:

– Un codice (es: scala "A") che la identifica insieme al nome del condominio;
– Un valore, detto *quota della scala*, che rappresenta, in millesimi, la frazione delle spese del condominio che sono complessivamente di competenza degli appartamenti compresi nella scala.

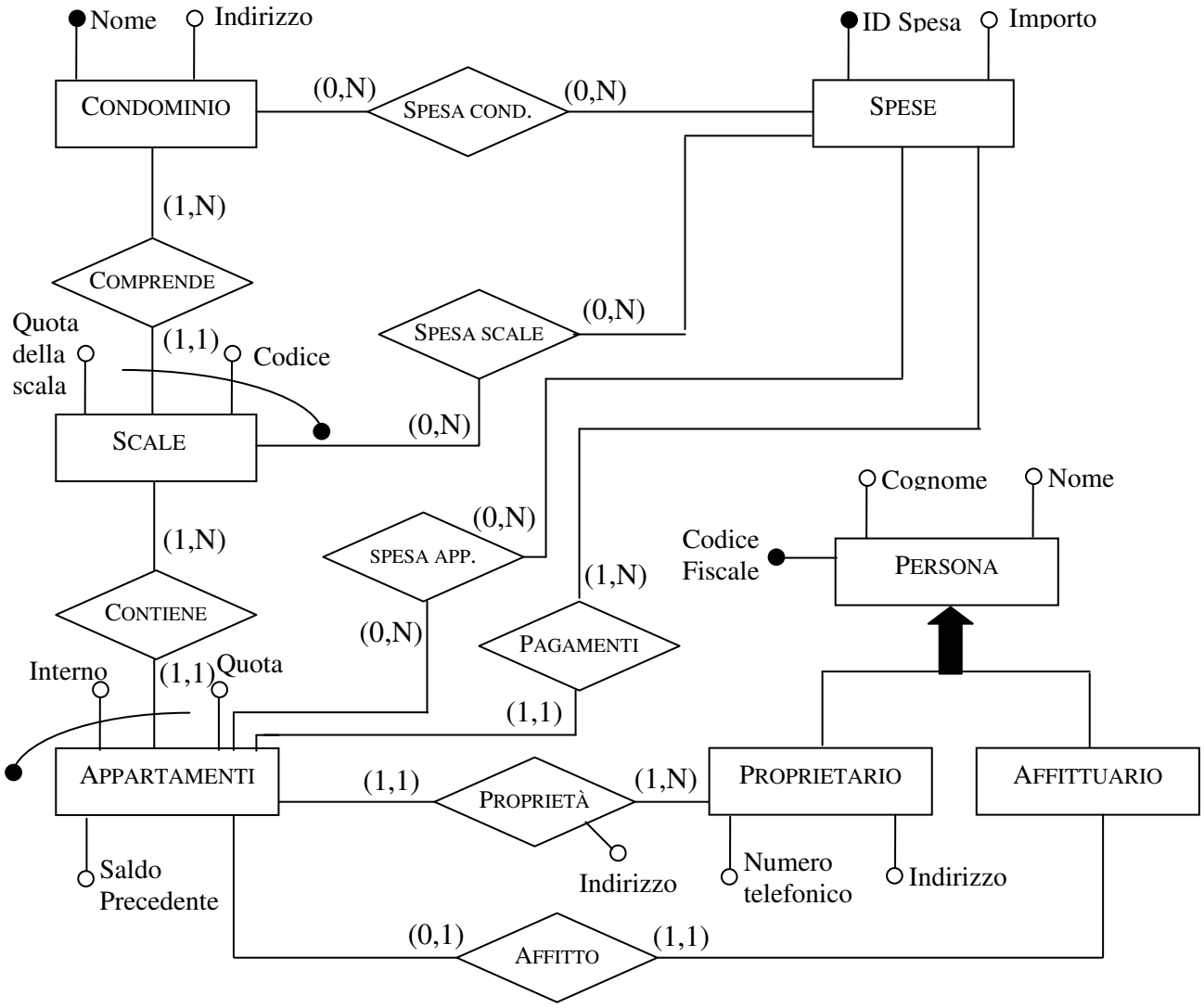
- Ogni appartamento è identificato, nel rispettivo condominio, dalla scala (se esiste) e da un numero (l'*interno*). Ad ogni appartamento è associata una quota (ancora espressa in millesimi), che indica la frazione della spese (della scala) che sono di competenza dell'appartamento.
- Ogni appartamento ha un proprietario per il quale sono di interesse il nome, il cognome, il codice fiscale e l'indirizzo al quale deve essere inviata la corrispondenza relativa all'appartamento. Ogni persona ha un solo codice fiscale, ma potendo essere proprietario di più appartamenti, potrebbe anche avere indirizzi diversi per appartamenti diversi. Di solito, anche chi è proprietario di molti appartamenti ha comunque solo uno o pochi indirizzi. In molti casi, l'indirizzo del proprietario coincide con quello del condominio.
- Per la parte contabile, è necessario tenere traccia delle spese sostenute dal condominio e dei pagamenti effettuati dai proprietari.

– Ogni spesa è associata ad un intero condominio, oppure ad una scala o ad un singolo appartamento.
– Ogni pagamento è relativo ad uno e un solo appartamento.

Nella base di dati vengono mantenuti pagamenti e spese relativi all'esercizio finanziario in corso (di durata annuale) mentre gli esercizi precedenti vengono sintetizzati attraverso un singolo valore (il *saldo precedente*) per ciascun appartamento che indica il debito o il credito del proprietario. In ogni istante esiste un *saldo corrente* per ciascun appartamento, definito come somma algebrica del saldo precedente e dei pagamenti (positivi) e delle spese addebitate (negative).

Se e quando lo si ritiene opportuno, introdurre codici identificativi sintetici.

Soluzione:


Nome
Indirizzo
CONDOMINIO
(0,N)
SPESA COND.
(0,N)
ID Spesa
Importo
SPESE
(1,N)
COMPRENDE
(1,1)
Quota della scala
Codice
SCALE
SPESA SCALE
(0,N)
(0,N)
(1,N)
CONTIENE
(1,1)
Interno
Quota
SPESA APP.
(0,N)
(0,N)
(1,N)
PAGAMENTI
(1,1)
Cognome
Nome
Codice Fiscale
PERSONA
APPARTAMENTI
(1,1)
PROPRIETÀ
(1,N)
PROPRIETARIO
AFFITTUARIO
Saldo Precedente
Indirizzo
Numero telefonico
Indirizzo
(0,1)
AFFITTO
(1,1)

Esercizio 7.11 In figura 7.34 è mostrata una schematizzazione dei programmi di una stagione teatrale nei diversi teatri di una città. Con riferimento ad essa:

1. definire uno schema concettuale (nel modello ER) che descriva la realtà di interesse; limitarsi agli aspetti che vengono espressamente mostrati, introducendo tutt'al più, dove lo si ritenga necessario, opportuni codici identificativi; mostrare le cardinalità delle relationship e gli identificatori delle entità;
2. progettare lo schema logico relazionale corrispondente allo schema concettuale definito al punto precedente, mostrando i nomi delle relazioni, quelli degli attributi e i vincoli di chiave e di integrità referenziale;
3. mostrare un'istanza della base di dati progettata al punto precedente, utilizzando i dati nell'esempio (o anche parte di essi, purché si riescano a mostrare gli aspetti significativi).

Osservazione: le risposte ai punti 2 e 3 richiedono lo studio del capitolo successivo, ma sono utili per verificare la correttezza della risposta al punto 1 e quindi la domanda viene proposta qui.

Soluzione

Si è scelto di usare molti codici identificativi: di per sé non sarebbe necessario ma può essere comodo.

La soluzione del punto 1 è mostrata in figura 7.I.

*N*ote:

- si poteva definire OperaInCartellone come una relazione ternaria con l'ipotesi che un'opera possa essere in cartellone una volta sola; in caso contrario deve essere una entità e l'identificatore deve includere l'attributo DataIniziale;
- si suppone che ogni opera abbia un solo autore; altrimenti la relationship fra Autore e Opera deve essere molti-a-molti;
- ssando meno identificatori, le entità CategoriaSpettacolo e TipoPosto avrebbero potuto essere omesse ed essere sostituite da attributi di Prezzo entrando a far parte dell'identificatore.

La stagione Teatrale in città

Teatro Comunale Via Roma, 25 Tel: 6555432

Prezzi:	Prime	Sab e Dom	Feriale
Platea	85	70	40
Palchi	70	50	30
Loggione	30	25	15

Riduzioni:

- studenti 20%
- CRAL 10%

Spettacoli

- **Così è (se vi pare) (1917)**
  L.Pirandello (1867-1936)
  dal 05.10.2005 al 21.11.2005
- **L'opera da tre soldi (1928)**
  B.Brecht (1967-1836)
  dal 25.11.2005 al 17.12.2005
- ...

Teatro Cittadino Piazza del municipio, 32 Tel: 6535455

Prezzi:	Prime	Sabato sera	Domenica	Altri
Platea	90	70	60	50
Galleria	60	40	50	30

Riduzioni:

- studenti 20%
- insegnanti 20%
- gruppi 10%

Spettacoli:

- **Enrico IV (1921)**
  L.Pirandello (1867-1936)
  dal 6.10.2005 al 5.11.2005
- **Uno sguardo dal ponte (1955)**
  Arthur Miller (1915-2005)
  dal 7.11.2005 al 9.12.2005
- **Così è (se vi pare) (1917)**
  L.Pirandello (1867-1936)
  dal 5.01.2006 al 7.02.2006
- seguono altri spettacoli
  ...

Teatro Nuovo ...
...

Figura 7.34 Le informazioni da modellare per l'esercizio 7.11.

Lo schema relazionale corrispondente allo schema concettuale di figura 7.I, soluzione del punto 2:

- TEATRI (Codice, Nome, Indirizzo, Telefono)
- TIPIRIDUZIONE(Codice, Descrizione)
- RIDUZIONI (Teatro,TipoRiduzione, Percentuale) con vincoli di integrità referenziale fra Teatro e la relazione TEATRI e fra TipoRiduzione e la relazione TIPIRIDUZIONE
- TIPIPOSTO (Codice, Descrizione)
- CATEGORIESPETTACOLO (Codice, Descrizione)
- PREZZI (Teatro,Categoriaspettacolo, TipoPosto, Importo) con vincoli di integrità referenziale fra l'attributo Teatro e la relazione TEATRI, fra l'attributo CategoriaSpettacolo e la relazione CATEGORIESPETTACOLI, fra l'attributo TipoPosto e la relazione TIPIPOSTO
- OPERETEATRALI (Codice, Titolo, Anno, Autore)
- AUTORI (Codice, Cognome, Nome, DataNascita, DataMorte)
- OPEREINCARTELLONE (Opera, Teatro, DataIniziale, DataFinale con vincoli di integrità referenziale fra l'attributo Opera e la relazione OPERETEATRALI e fra la l'attributo Teatro e la relazione TEATRI.

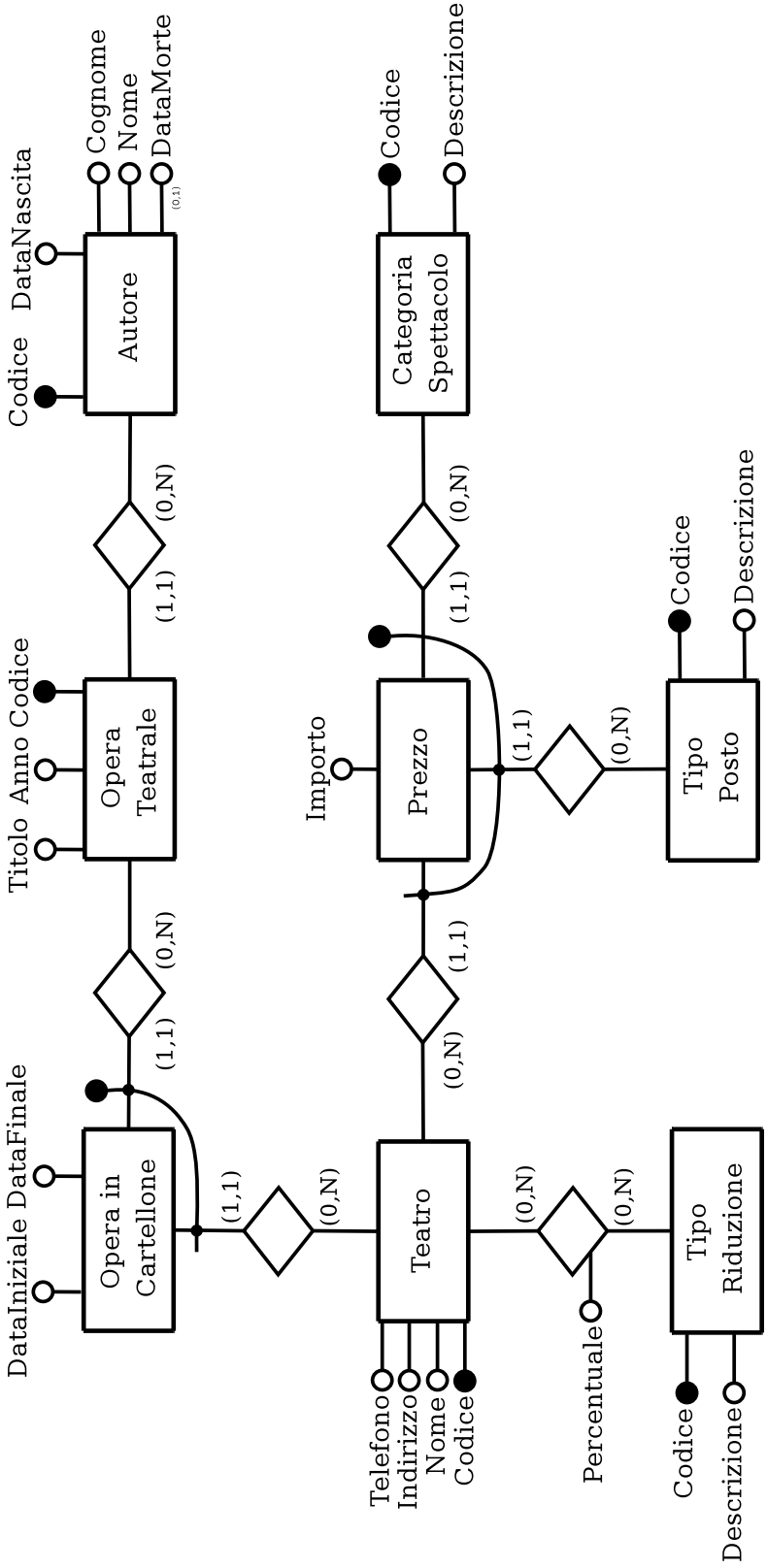


Figura 7.I Figura soluzione dell'esercizio 7.11.

Teatri			
Codice	Nome	Indirizzo	Telefono
TT001	Teatro Comunale	Via ...	...
TT002	Teatro Cittadino	...	...
...	...	...	...

Riduzioni		
Teatro	TipoRiduzione	Percentuale
TT001	S	20
TT001	C	10
TT002	S	20
...	...	...

TipiRiduzione	
Codice	Descrizione
S	Studenti
C	CRAL
I	Insegnanti
G	Gruppi

TipiPosto	
Codice	Descrizione
Pl	Platea
Pa	Palchi
L	Loggione
...	...

CategorieSpettacoli	
Codice	Descrizione
P	Prime
SD	Sab e DomL
F	Feriale
SS	Sabato sera
...	...

Prezzi			
Teatro	CategoriaSpettacolo	TipoPosto	Importo
T001	Pl	P	85
T001	Pa	P	70
...	...	...	...
T001	Pl	SD	70
T002	Pl	P	90
...	...	...	...

OpereTeatrali			
Codice	Titolo	Anno	Autore
Op01	Così è (se vi pare)	1917	LB
Op02	L'opera da tre soldi	1928	BB
Op03	Enrico IV	1921	LP
...	...	...	...

Autori				
Codice	Cognome	Nome	DataNascita	DataMorte
LP	Pirandello	Luigi	1967	1936
BB	Brecht	Bertold	1898	1956
...	...	...	...	...

OpereInCartellone			
Opera	Teatro	DataIniziale	DataFinale
Op01	TT01	5.10.2005	21.11.2005
Op02	TT01	15.11.2005	17.12.2005
...	...	...	...
Op01	T002	5.01.2006	7.02.2006
...	...	...	...

Figura 7.II Esempio di istanza per la base di dati relazionale soluzione dell'esercizio 7.11

Esercizio 7.12 Mostrare lo schema concettuale di una base di dati per tornei di calcio, secondo le seguenti specifiche:

- i vari tornei hanno codice e nome
- ogni torneo è composto da un certo numero di squadre e da una classifica che assegna ad ogni squadra un punteggio
- le squadre partecipano ad un solo torneo e hanno un nome e una rosa di giocatori di cui registriamo il numero di maglia, il nome e la data di nascita.
- le partite si svolgono tra due squadre dello stesso torneo, in una certa data, in un certo stadio e hanno un risultato finale
- si vogliono registrare i giocatori che giocano nelle varie partite e il ruolo ricoperto (che è lo stesso in una partita ma può variare in partite diverse).

Indicare gli eventuali vincoli di integrità che non è possibile rappresentare nello schema.

Soluzione

La soluzione viene mostrata nella figura 7.III

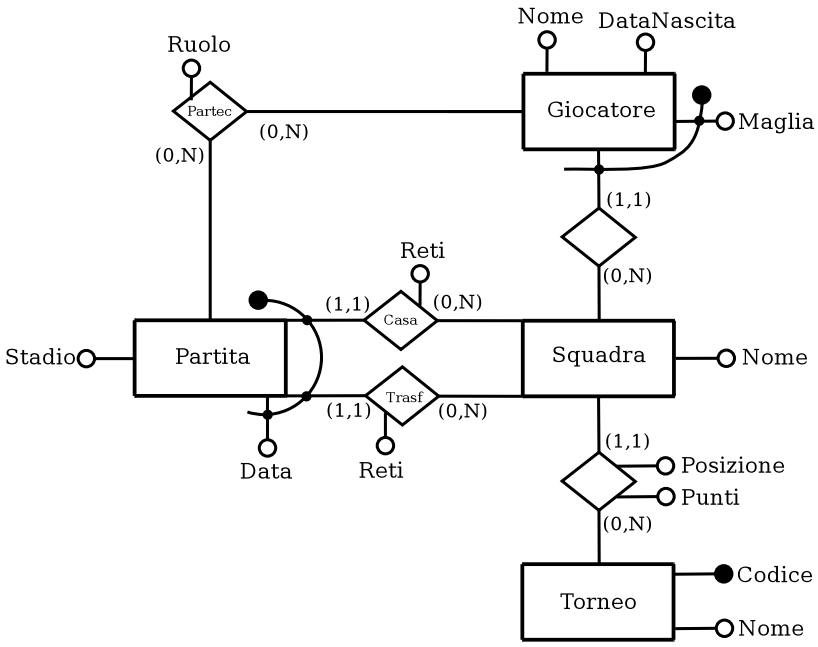


Figura 7.III Figura di riferimento per la soluzione dell'esercizio 7.12

Esercizio 7.13 Estendere lo schema concettuale ottenuto in risposta alla domanda precedente, per tenere conto delle seguenti specifiche aggiuntive:

1. è di interesse rappresentare l'evoluzione temporale della classifica (una squadra può avere due punti un certo giorno e quattro in un altro).
2. i tornei si ripetono negli anni e ogni squadra partecipa ad un torneo all'anno, con giocatori eventualmente diversi.

Mostrare separatamente i due frammenti di schema necessari per rappresentare le modifiche.

Soluzione

Per la soluzione di questo esercizio fare riferimento alla figura 7.IV per il primo punto, alla figura 7.V per il secondo punto.

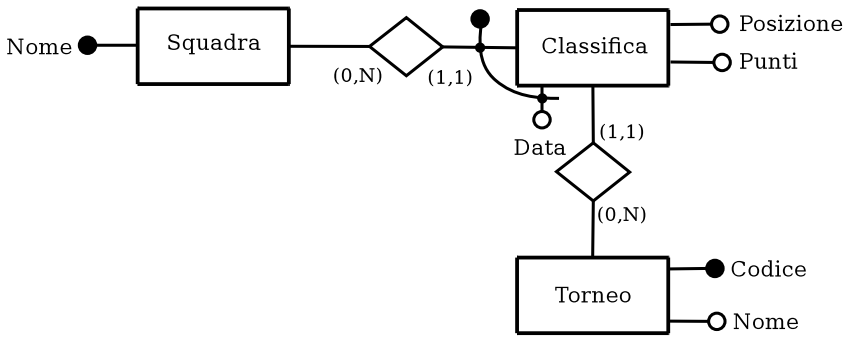


Figura 7.IV Figura di riferimento per la soluzione dell'esercizio 7.13, prima parte.

Ogni occorrenza di "Classifica" corrisponde ad una "riga" della classifica, relativa ad una squadra in un certo momento; le occorrenze di Classifica di una squadra sono tutte relative allo stesso Torneo.

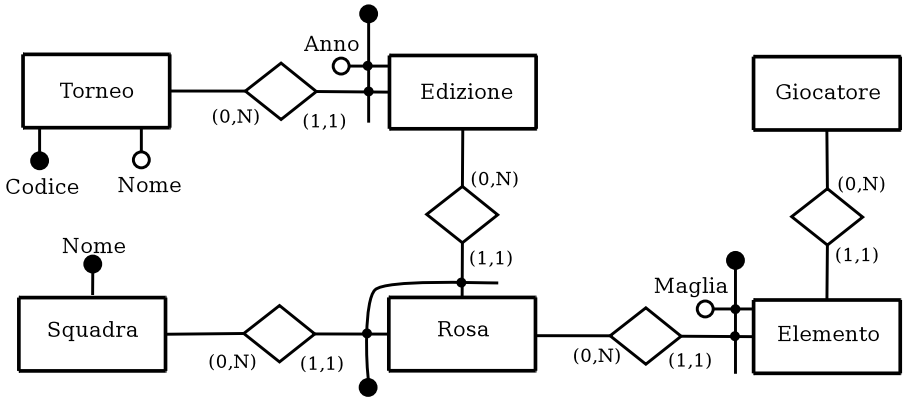


Figura 7.V Figura di riferimento per la soluzione dell'esercizio 7.13, seconda parte.

Vincolo: un giocatore (attraverso “Elemento”) può partecipare al più ad una rosa all’anno.

Esercizio 7.14 Si consideri la seguente schematizzazione di alcune prenotazioni aeree:

```
Prenotazione N. 1270
   Passeggeri
   Mario  Rossi   (Cod.1230)   Tel.   06/45531123
   Lucia  Neri    (Cod.1231)   Tel.   06/64352134
   Piero  Rossi   (Cod.1232)
   Itinerario
      Da    A      Data       Ora   NumeroVolo  Aeromobile Classe
   1. FCO  LHR  11/03/2008   07:50    AZ024        A321       V
   2. LHR  MAN  11/03/2008   11:30    BA233        M80X       F
   3. LHR  FCO  18/03/2008   11:50    AZ175        A320       C

Prenotazione N.1343
   Passeggeri
   Giulio  Rossi   (Cod.1343)   Tel.   06/45521123
   Itinerario
      Da    A      Data       Ora   NumeroVolo  Aeromobile Classe
   1. FCO  LHR  12/04/2008   08:20    AZ024        A321       G
   2. LHR  FCO  21/04/2008   13:50    AZ175        A320       C

Prenotazione N.1777
   Passeggeri
   Mario  Rossi   (Cod.1230)   Tel.   06/45521123
   Itinerario
      Da    A      Data       Ora   NumeroVolo  Aeromobile Classe
   1. FCO  LHR  12/04/2008   08:20    AZ024        A321       G
   2. LHR  FCO  21/04/2008   13:50    AZ175        A320       C
```

Si tenga conto a riguardo delle seguenti precisazioni:

- per ogni passeggero esistono codice (identificativo), cognome, nome e numero di telefono che è opzionale ed è lo stesso in tutte le prenotazioni;
- le colonne Da e A contengono codici di aeroporti, per i quali sono memorizzati anche il nome e la città (ad esempio, a "FCO" sono associati "Fiumicino" come nome e "Roma" come città);
- il numero del volo (ad esempio "AZ024") è costituito dal codice della compagnia (per la quale interessa anche il nome; ad esempio "AZ" è il codice della compagnia il cui nome è "Alitalia") e da un intero;
- un volo con un certo NumeroVolo ha sempre gli stessi aeroporti di partenza e di arrivo (Da e A) e lo stesso tipo di aeromobile (colonna Aeromobile), ma può avere orario diverso in date diverse; per il tipo di aeromobile al codice (mostrato nella scheda, ad esempio "A321") è associato un nome (nell'esempio potrebbe essere "airbus 321");
- la colonna Classe contiene un codice (della "classe di prenotazione") che, come si vede dai dati, è specificatamente associato a volo e prenotazione; per ogni valore di tale codice è memorizzata una descrizione.

Con riferimento alla corrispondente realtà definire uno schema concettuale che la descriva limitandosi agli aspetti che vengono citati e mostrando sia le cardinalità delle relationship sia gli identificatori delle entità.

Soluzione
La soluzione è mostrata in figura 7.VI.

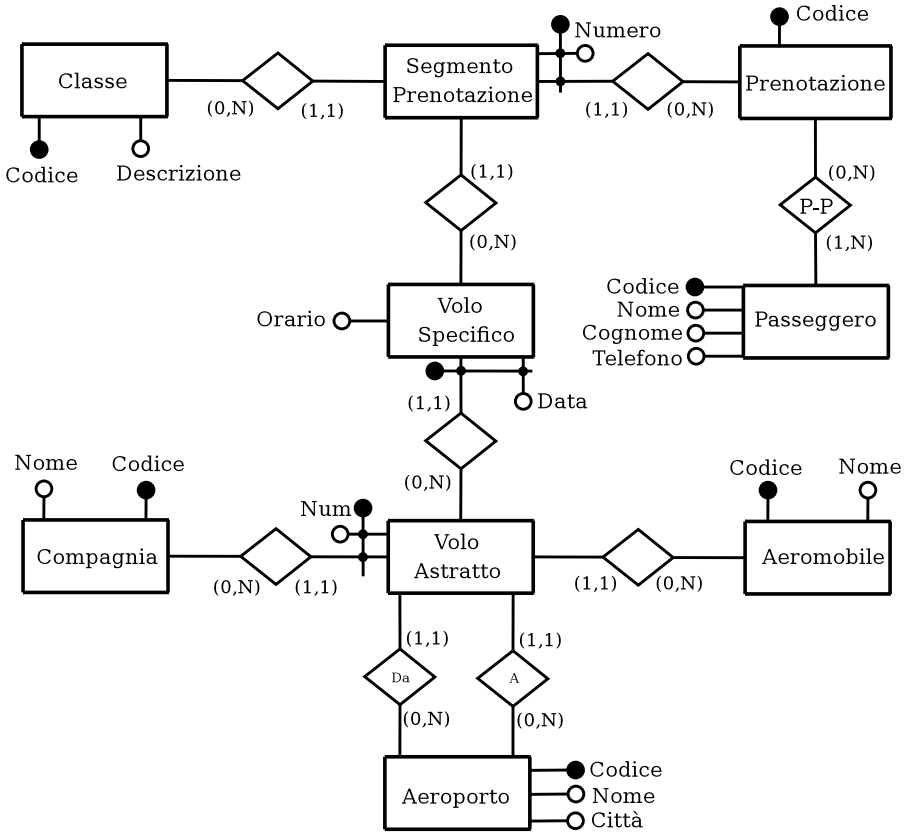


Figura 7.VI Soluzione dell'esercizio 7.14

Esercizio 7.15 Mostrare lo schema concettuale per una base di dati per un programma di concerti, secondo le specifiche seguenti:

- ogni concerto ha un codice, un titolo e una descrizione ed è composto da una sequenza (ordinata) di pezzi musicali;
- ogni pezzo ha un codice, un titolo e un autore (con codice e nome); uno stesso pezzo può essere rappresentato in diversi concerti;
- ogni concerto è eseguito da un'orchestra: ogni orchestra ha un nome, un direttore (del quale interessano solo nome e cognome) e un insieme di orchestrali;
- ogni orchestrale ha una matricola (univoca nell'ambito della base di dati), nome e cognome, può partecipare a più orchestre, in ciascuna delle quali suona uno e un solo strumento, ma in orchestre diverse può suonare strumenti diversi;
- ogni concerto è tenuto più volte, in giorni diversi, ma sempre nella stessa sala;
- ogni scala ha un codice, un nome e una capienza.

Soluzione
La soluzione di questo esercizio è riportata nella figura 7.VII.

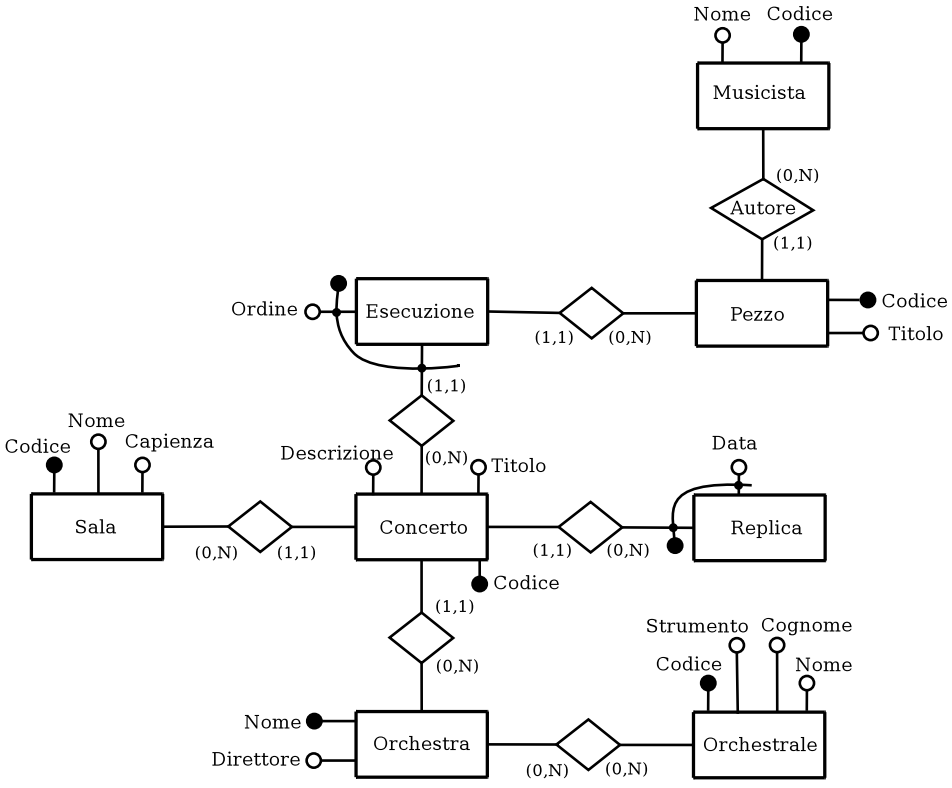


Figura 7.VII Soluzione dell'esercizio 7.15

Esercizio 7.16 In figura 7.35 è mostrata una schematizzazione del catalogo dei viaggi di studio all'estero proposti da un operatore del settore.
Con riferimento ad essa definire uno schema concettuale (nel modello ER) che descriva la realtà di interesse. Limitarsi agli aspetti che vengono espressamente mostrati, introducendo tutt'al più, ove lo si ritenga necessario, opportuni codici identificativi; mostrare le cardinalità delle relazioni e degli idenfificatori delle entità.

Soluzione

La soluzione di questo esercizio è riportata nella figura 7.VIII.

Cambridge *Aeroporto*: Heathrow *Esame*: Pet

King's College — 101 King's Street — Tel: +44 123 6667777
15 ore di lezione a settimana

Periodo	Prezzo
1/7-15/07/2006	1500
15/7-29/7/2006	1700
29/7-13/8/2006	1650

Sconto seconda quindicina 10%
Sconto gruppi 15%

Queen's College — 1021 Queen's Road — Tel: +44 123 7665433
20 ore di lezione a settimana

Periodo	Prezzo
5/7-19/07/2006	1400
19/7-2/8/2006	1600
...	...

Sconto seconda quindicina 10%
Sconto fratelli 8%

seguono altri college

Oxford *Aeroporto*: Heathrow *Esame*: Trinity

Prince College — 1021 St.John's Road — Tel: +44 125 6765443
18 ore di lezione alla settimana

Periodo	Prezzo
4/7-18/07/2006	1200
...	...

seguono altri college

Stirling *Aeroporto*: Edimburgo *Esame*: Trinity seguono altri college

INFORMAZIONI GENERALI, PER TUTTE LE LOCALITÀ

Esami:
Pet: 30 euro
Trinity: 35 Euro

Voli:

	Heatrow	Edimburgo	Dublino
Roma	450	600	500
Milano	400	550	430
Palermo	550	700	650

Figura 7.35 Le informazioni da modellare per l'esercizio 7.16.

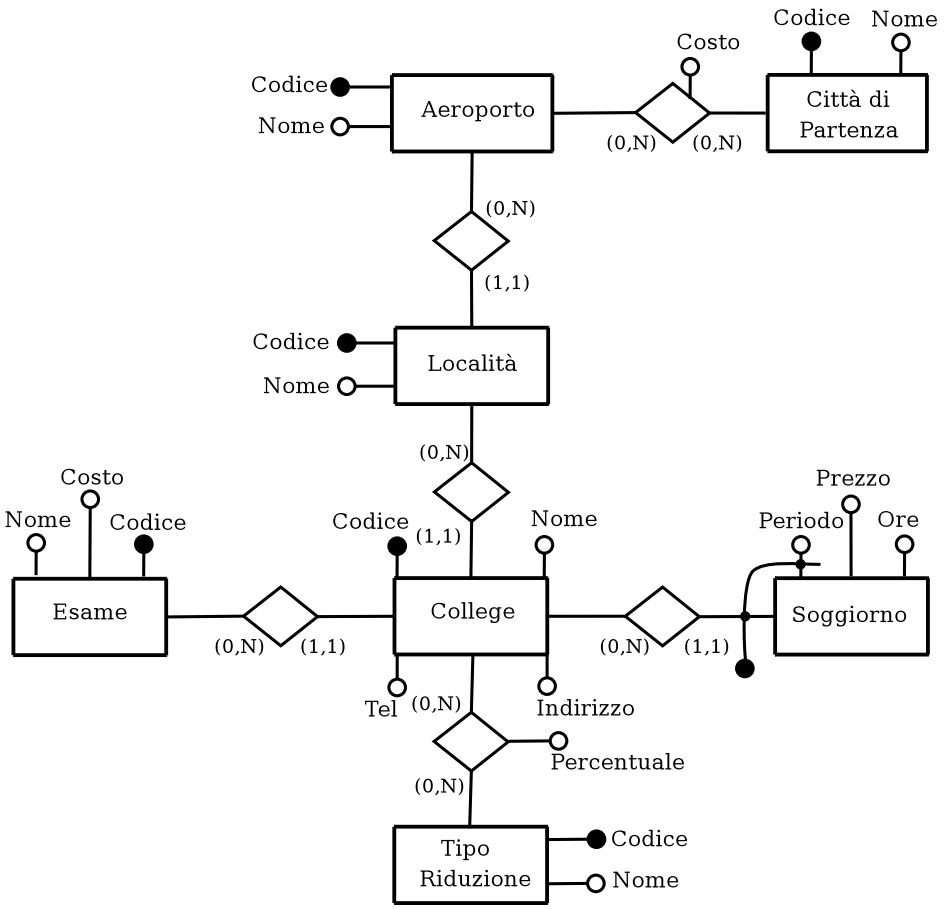


Figura 7.VIII Soluzione dell'esercizio 7.16, prima parte

Esercizio 7.17 Definire uno schema E-R che descriva i dati di una applicazione relativa alla gestione ed evasione degli ordini da parte di una azienda, secondo le specifiche elencate di seguito.

- L'azienda riceve gli ordini emessi dai clienti (ognuno dei quali ha numero di partita IVA, che identifica ragione sociale, indirizzo e percentuale di sconto). Ogni ordine ha un numero (attribuito dal cliente), indica il nome di un referente interno del cliente (che può essere lo stesso per tutti gli ordini) e richiede uno o più prodotti, per ciascuno dei quali indica una quantità e una sede di destinazione (in quanto ciascun cliente può, anche nell'ambito di uno stesso ordine, richiedere che i vari prodotti siano consegnati in sedi diverse; ad esempio: "tre calcolatori X386, due stampanti Z322 a via Roma 103 e due calcolatori X343 e una stampante Z320 a Corso Garibaldi 12"). Ad ogni ordine viene assegnato, dall'azienda, all'atto della ricezione, un numero progressivo identificante. Ogni sede di destinazione viene rappresentata da un codice e un indirizzo e non ha correlazione formale con il cliente.
- Gli ordini vengono evasi attraverso consegne, ognuna delle quali è relativa ad un unico cliente e un'unica sede di destinazione, ma può riferirsi a più ordini. Ogni ordine, a sua volta, è soddisfatto attraverso una o più consegne. Per ogni consegna sono rilevanti la data, l'ora e il numero di bolla di accompagnamento. L'azienda ha vari mezzi di trasporto (identificati ognuno da un codice e senza ulteriori proprietà di interesse), ognuno dei quali effettua al più un giro di consegne al giorno, per il quale è di interesse l'ora di uscita dal magazzino.
- Ogni prodotto ha un codice identificante, un nome e un prezzo unitario. I prodotti si dividono in due categorie: inventariabili (per i quali ciascun esemplare ha un numero di matricola di cui si deve tenere traccia nell'ambito della consegna) e di consumo (per i quali è sufficiente far riferimento alle quantità).
- Quando un ordine è stato completamente evaso, viene emessa la fattura, che ha un numero progressivo, una data e un importo.

Indicare le cardinalità delle relazioni, (almeno) un identificatore per ciascuna entità e i vincoli non esprimibili per mezzo dello schema. Indicare se è necessario formulare delle ipotesi aggiuntive alle specifiche descritte, senza contraddirle. Limitare le relationship ridondanti che secondo le specifiche potrebbero essere presenti: ad esempio, è evidente che i prodotti sono associati agli ordini e alle consegne (compaiono su vari documenti, ordini, bolle e fatture) ma si richiede di rappresentare tutti i concetti di interesse senza ripeterli. Specificatamente si può pensare di associare i prodotti agli ordini solo inizialmente, per poi associarli alle consegne che, essendo comunque legate agli ordini stessi, permettono di ricostruire l'informazione originaria; per le stesse ragioni, è inopportuno associare i prodotti alle fatture (anche se sulla fattura sono elencati, ma è possibile ricostruire l'elenco per altra via).

Soluzione

La soluzione di questo esercizio è riportata nella figura 7.IX.

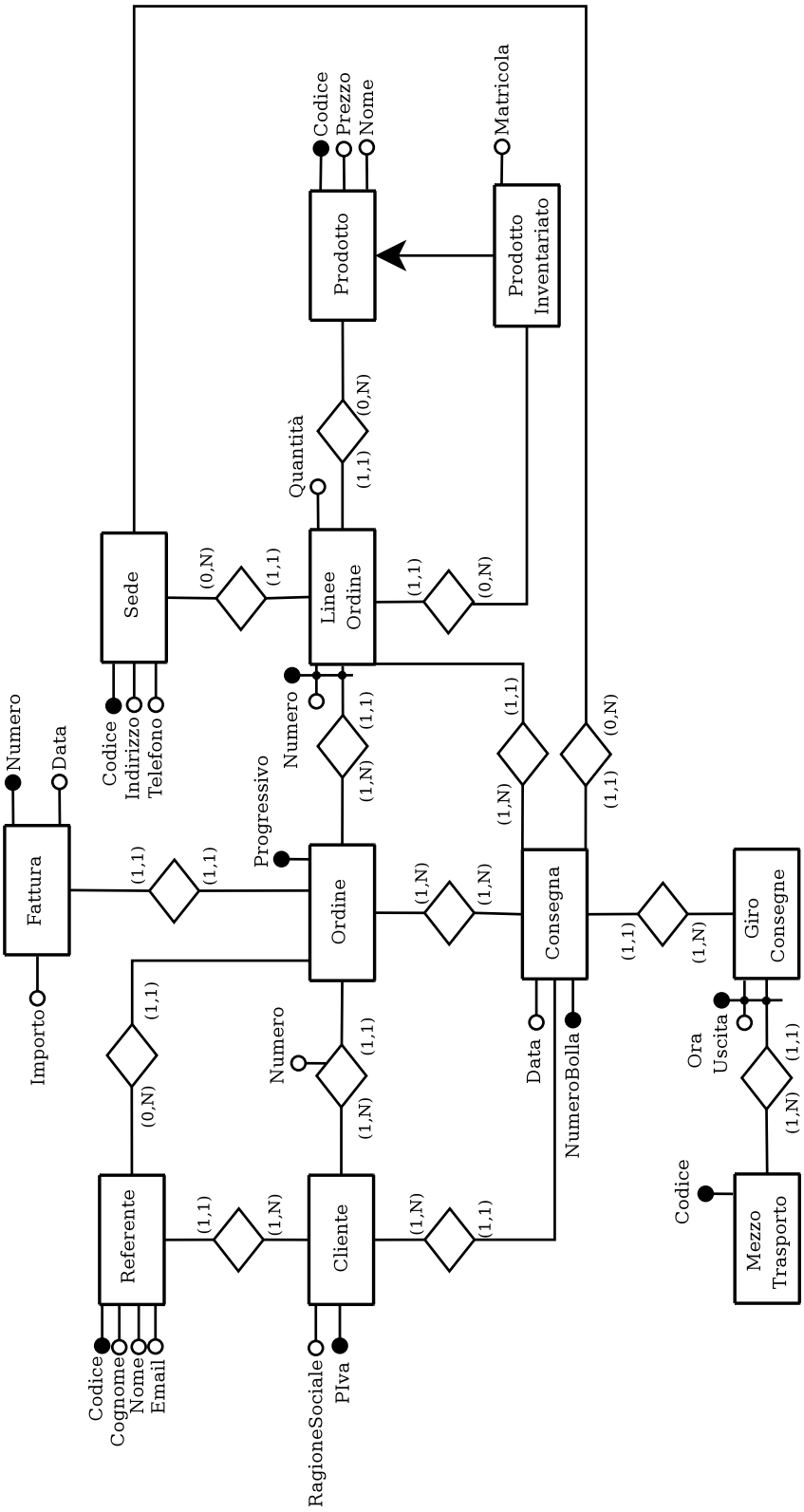


Figura 7.IX Soluzione dell'esercizio 7.17

Esercizio 7.18 Definire uno schema Entity-Relationship che descriva informazioni relative a sale cinematografiche di una città, secondo le seguenti specifiche:

- ogni cinema ha un nome che lo identifica univocamente, un indirizzo e un numero di telefono. Un cinema è organizzato in più sale, ognuna delle quali ha un codice che la distingue (nell'ambito del cinema) e un numero fissato di posti;
- per ogni sala interessa la programmazione di una sola giornata (quella odierna, senza traccia di quelle passate e future) che consiste in un elenco di proiezioni di film (eventualmente anche diversi), ognuna delle quali ha un orario di inizio;
- per ogni film si registrano il titolo, il genere (codice e nome descrittivo), la nazionalità (una semplice stringa) e il regista (con codice identificativo, nome, cognome e anno di nascita).

Soluzione
La soluzione dell'esercizio è riportata nella figura 7.X.

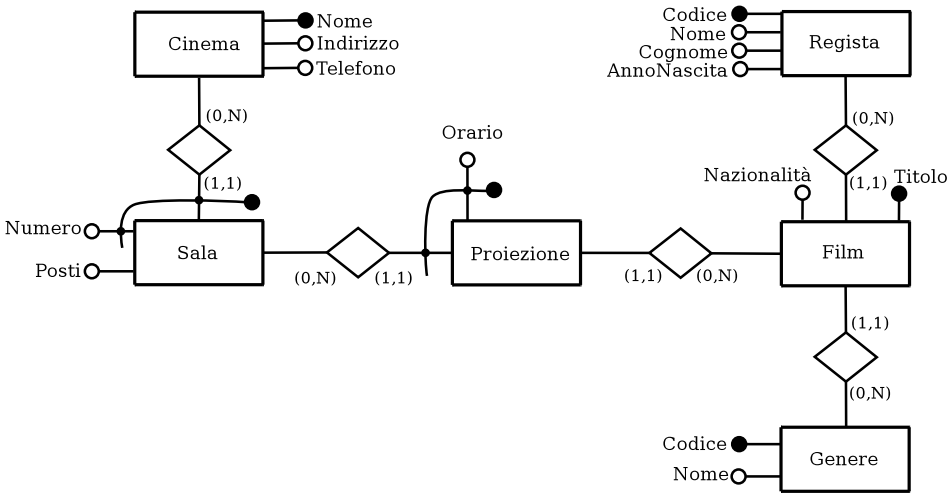


Figura 7.X Figura soluzione dell'esercizio 7.18.

Esercizio 7.19 Estendere lo schema concettuale proposto in risposta alla domanda precedente per rappresentare anche le seguenti specifiche:

- i posti di ciascuna sala sono numerati;
- per ogni proiezione è possibile effettuare prenotazioni, ognuna delle quali ha un codice identificativo, un nominativo e un insieme di posti.

Soluzione
La soluzione dell'esercizio è riportata in figura 7.XI.

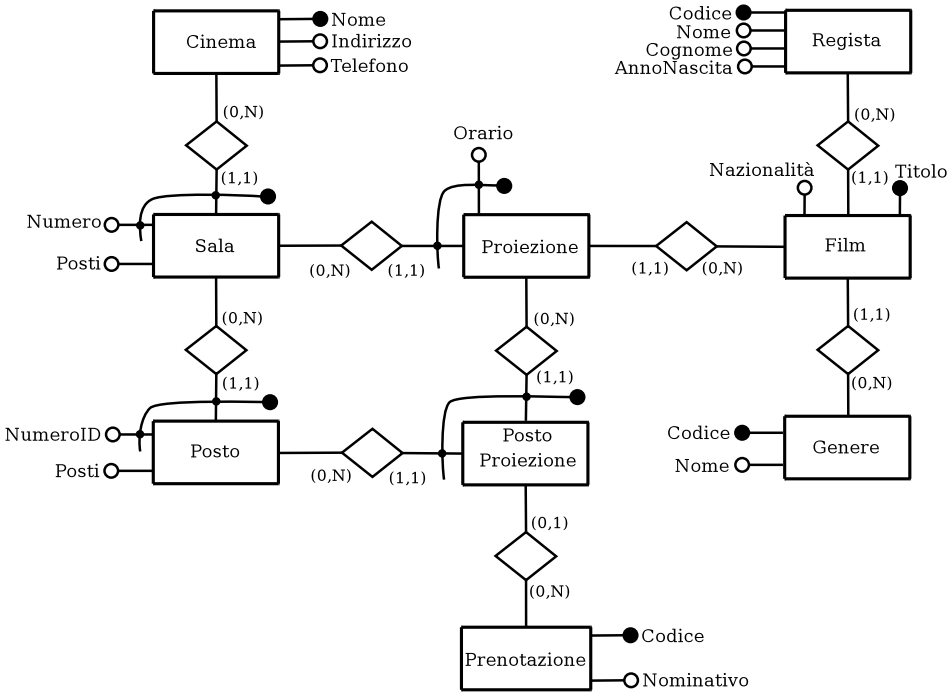


Figura 7.XI Figura soluzione dell'esercizio 7.19.

Esercizio 7.20 Mostrare lo schema concettuale di una base di dati per la gestione di articoli di una rivista scientifica secondo le seguenti specifiche:

- Gli articoli hanno un titolo, un sottotitolo, uno o più autori e un testo (una stringa molto grande, ma comunque gestibile);
- gli autori hanno nome, cognome, email ed affiliazione (l'istituzione per la quale lavorano);
- per ogni istituzione (degli autori) sono d'interesse il nome, l'indirizzo e la nazione;
- la rivista viene pubblicata un certo numero di volte in un anno. Le pubblicazioni di un anno vengono raccolte in un volume (a cui viene dato un titolo complessivo). Ogni pubblicazione ha un numero, unico nel rispettivo volume, una data di pubblicazione e una serie di articoli, per ognuno dei quali viene registrata la pagina di inizio e quella di fine.

Soluzione
Lo schema E-R risultato è mostrato in figura 7.XII

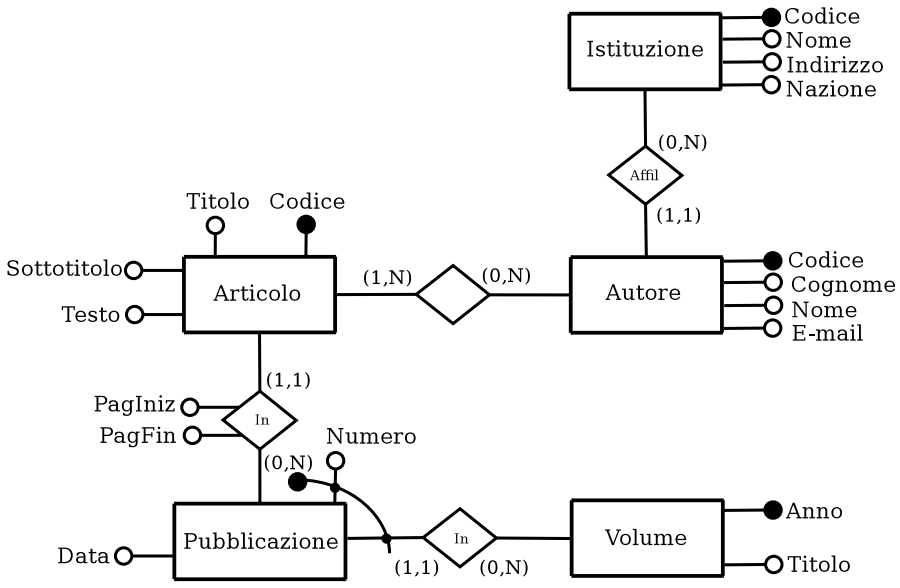


Figura 7.XII Figura di riferimento per la soluzione dell'esercizio 7.20

Esercizio 7.21 Estendere lo schema concettuale ottenuto in risposta alla domanda precedente, per rappresentare l'attività di selezione degli articoli, sulla base delle seguenti specifiche aggiuntive:

- La rivista riceve proposte, per le quali sono di interesse le stesse informazioni registrate per gli articoli, oltre che la data di presentazione.
- Ogni proposta viene revisionata da due o più esperti (per i quali sono di interesse le stesse informazioni degli autori; si noti che gli autori possono essere esperti e viceversa, ma ovviamente un esperto non può revisionare una propria proposta), che assegnano alla proposta un punteggio tra 0 e 10 e forniscono un commento (un semplice testo).
- Le proposte che ricevono un punteggio medio superiore a 7 diventano articoli da pubblicare e hanno a quel punto una data di accettazione.

Indicare gli eventuali vincoli di integrità che non è possibile rappresentare nello schema.

Soluzione

Lo schema risultato è mostrato nella figura 7.XIII.

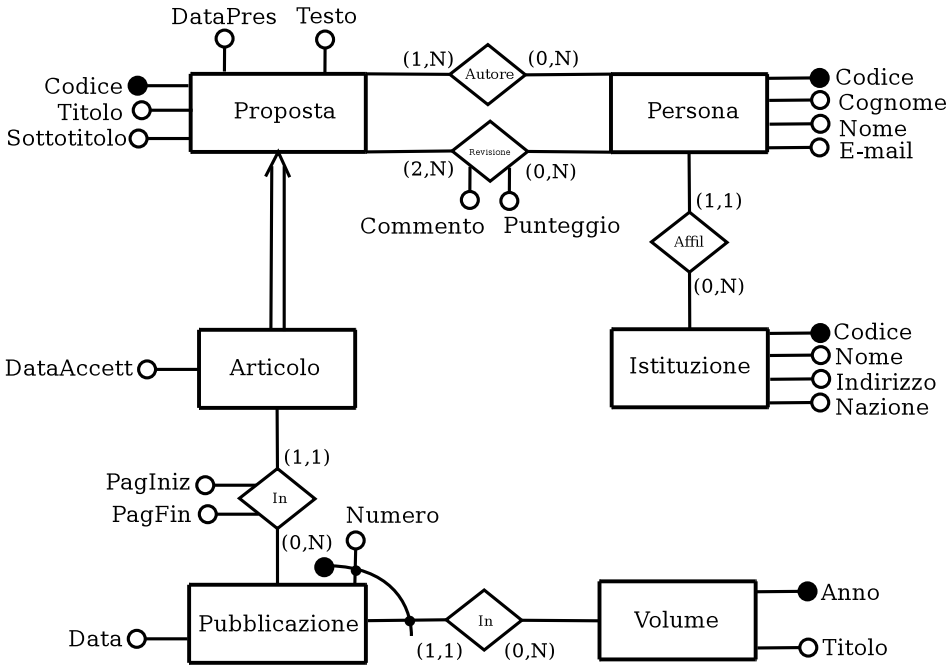


Figura 7.XIII Figura soluzione dell'esercizio 7.21.

Non sono esprimibili i seguenti due vincoli:

- una persona non può essere autore e revisore dello stesso articolo
- solo le proposte con punteggio medio superiore a 7 diventano articoli.

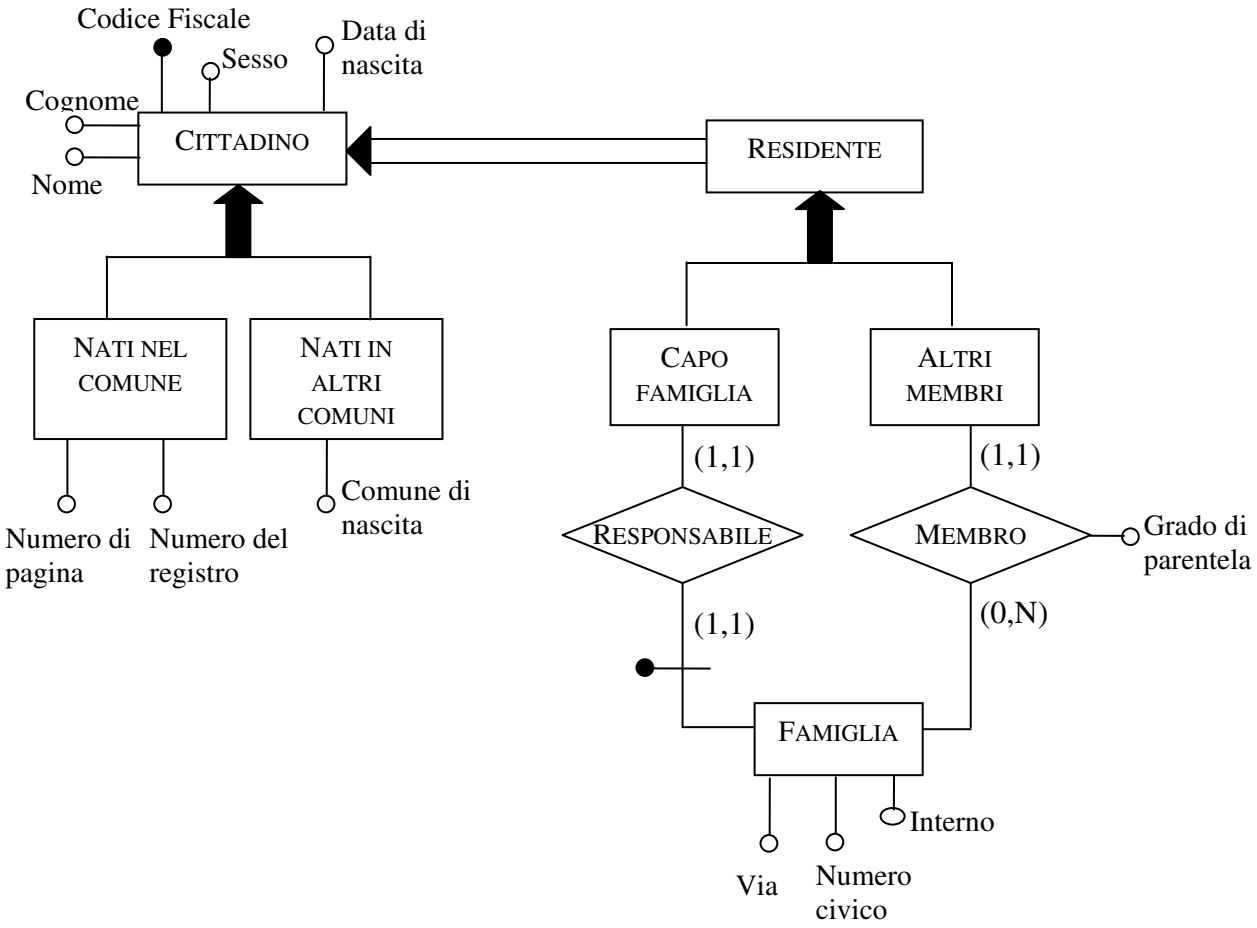
Capitolo 8

Esercizio 8.1

Si consideri lo schema Entità-Relazione ottenuto come soluzione dell'esercizio 7.4. Fare delle ipotesi sul volume dei dati e sulle operazioni possibili su questi dati e, sulla base di queste ipotesi, effettuare le necessarie ristrutturazioni dello schema. Effettuare poi la traduzione verso il modello relazionale.

Soluzione:

Questo è lo schema prodotto nell'esercizio 7.4



Le seguenti tavole contengono ipotesi sui volumi e sulle operazioni:

Volumi:

Concetto	Tipo	Volume
Cittadino	E	1.100.000
Nati nel comune	E	1.000.000
Nati in altri comuni	E	100.000
Residente	E	1.000.000
Capo famiglia	E	250.000
Altri membri	E	750.000
Famiglia	E	250.000
Responsabile	R	250.000
Membro	R	750.000

Operazioni:

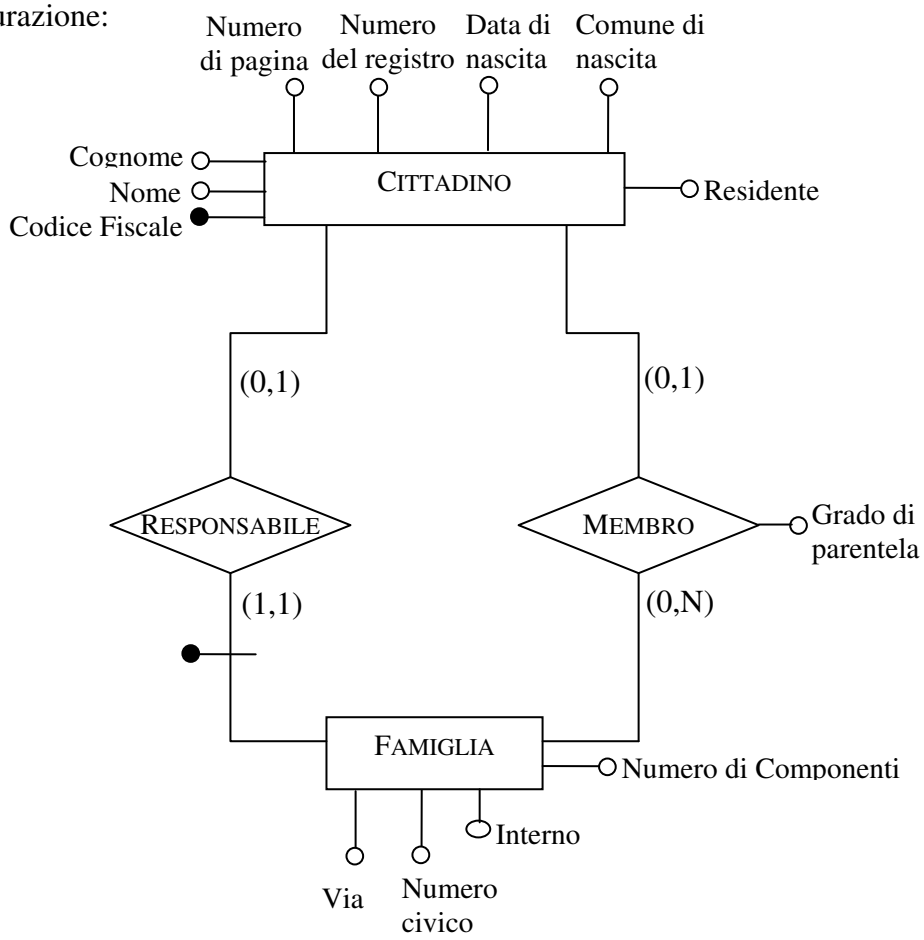
Operazione	Descrizione	Frequenza	Tipo
1	Aggiungere un nuovo cittadino nato nel comune	100 al giorno	I
2	Aggiungere un nuovo cittadino residente nel comune ma nato in un altro comune	20 al giorno	I
3	Aggiungere una nuova famiglia	20 al giorno	I
4	Cancellare un cittadino	100 al giorno	I
5	Cancellare una famiglia	5 al giorno	I
6	Visualizzare il numero di cittadini residenti nel comune	1 al giorno	B
7	Visualizzare un numero di residenti uomini e donne	1 al giorno	B

Potrebbe essere utile per aggiungere un attributo ridondante "Numero di Componenti" all'entità FAMIGLIA. Senza questo attributo, l'operazione 6 ha bisogno di 1.000.000 di accessi in lettura all'entità RESIDENTE ogni giorno. Con questo attributo ridondante, l'operazione 6 ha bisogno di soli 250.000 accessi in lettura all'entità FAMIGLIA.
Comunque, la presenza di questo attributo cambia il costo delle operazioni 1, 2 e 4; infatti queste 3 operazioni hanno ora bisogno, oltre agli accessi che già avevano, anche di un accesso in lettura a CAPO FAMIGLIA (o ad ALTRI MEMBRI), un accesso a RESPONSABILE (o a MEMBRO), un accesso in lettura ed uno in scrittura all'entità FAMIGLIA (per aggiornare l'attributo "Numero di componenti").

Supponendo che un accesso in scrittura abbia il costo di 2 accessi in lettura, il costo totale è (1+1+1+2)*90 + (1+1+1+2)*20 + (1+1+1+2)*100 = 1.050
La frequenza dell'operazione 1 è 90 perché non tutti i cittadini nati nel comune sono residenti, ma solo il 90%.

Così, il vantaggio dell'attributo ridondante è 750.000 – 1.050 = 748.950 accessi al giorno.

Ristrutturazione:



Traduzioni:

CITTADINO(<u>Codice Fiscale</u>, Cognome, Nome, Numero di pagina, Numero del registro, Data di nascita, Comune di nascita, Residente)

FAMIGLIA(<u>Capo Famiglia</u>, Via, Numero civico, Interno, Numero di Componenti) con vincolo di integrità referenziale tra **Capo Famiglia** e la relazione CITTADINO.

MEMBRO(<u>Cittadino</u>, <u>Famiglia</u>, Grado di parentela)) con vincolo di integrità referenziale tra **Cittadino** e la relazione CITTADINO e tra **Famiglia** e la relazione FAMIGLIA.

Esercizio 8.2

Tradurre lo schema Entità-Relazione che abbiamo più volte incontrato sul personale di un'azienda (riportato per comodità in figura 8.36) in uno schema del modello relazionale.

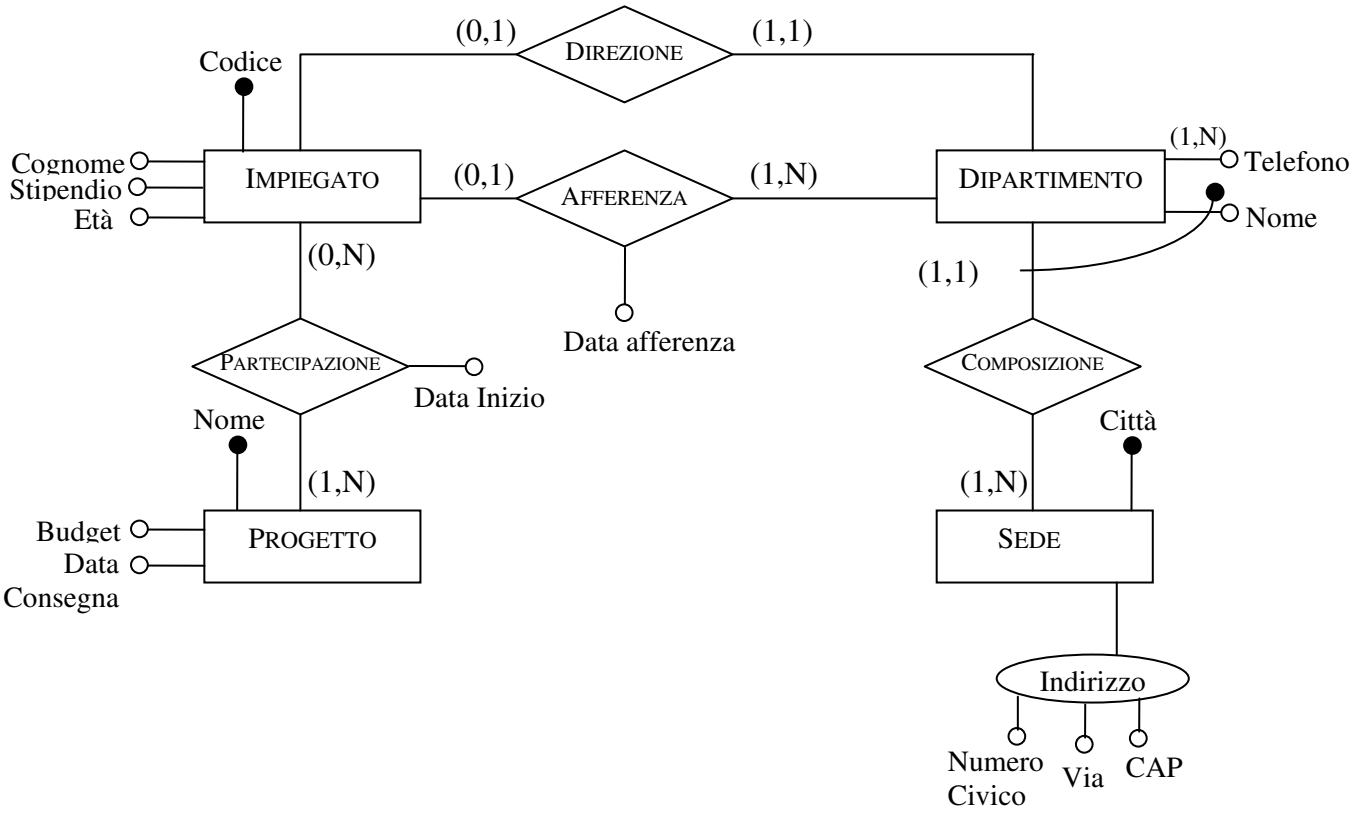


Figura 8.36 Uno schema E-R sul personale di un'azienda

Soluzione:

IMPIEGATO(Codice, Cognome, Stipendio, Età, Dipartimento, Sede, Data afferenza), con vincolo di integrità referenziale tra **Dipartimento** e la relazione DIPARTIMENTO, e tra **Sede** e la relazione SEDE.

DIPARTIMENTO(Nome, Sede) con vincolo di integrità referenziale tra **Sede** e la relazione SEDE.

TELEFONO(Dipartimento, Numero), con vincolo di integrità referenziale tra **Dipartimento** e la relazione DIPARTIMENTO.

SEDE(Città, CAP, Via, Numero Civico)

PROGETTO(Nome, Budget, Data Consegna)

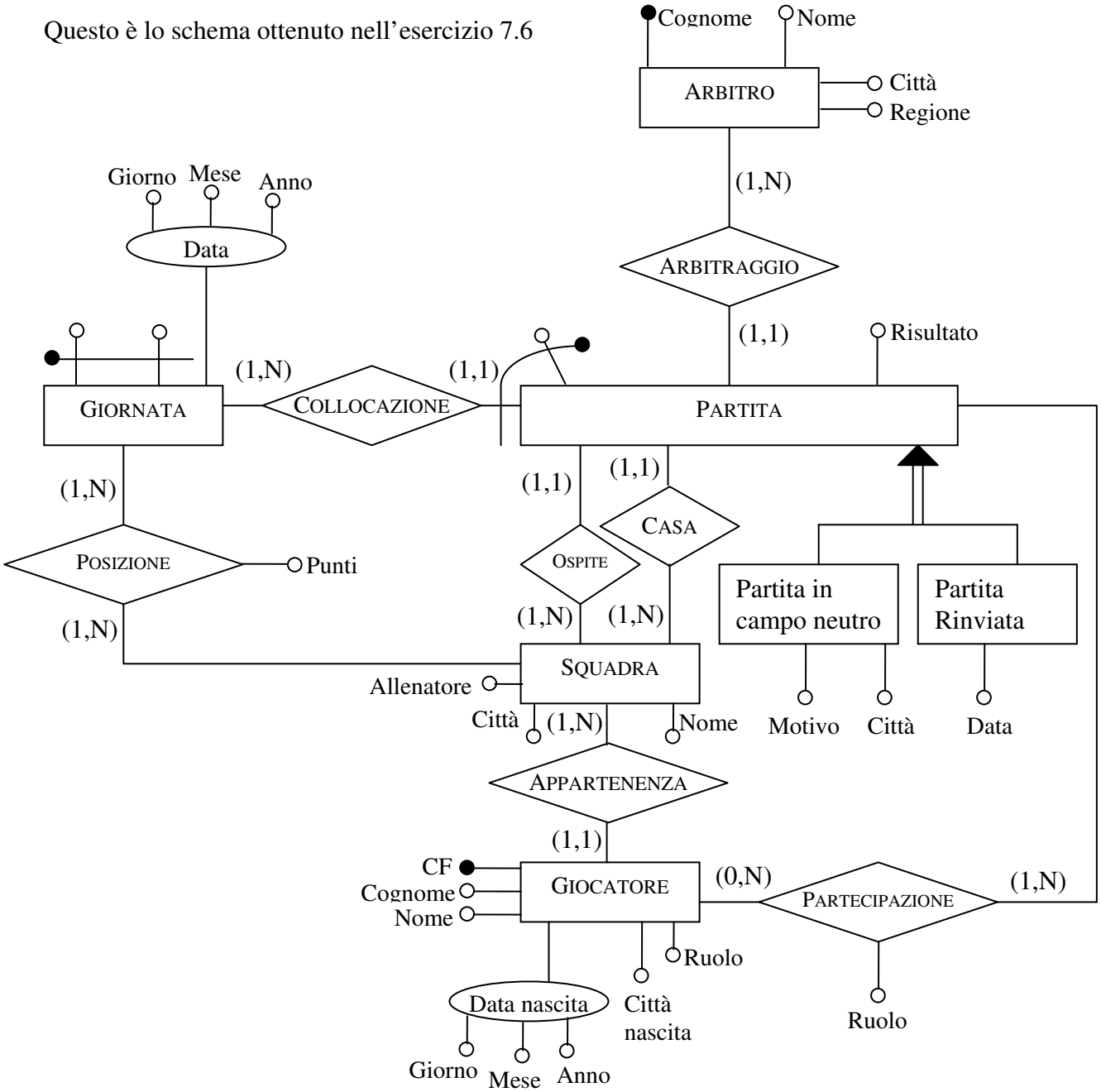
PARTECIPAZIONE(Impiegato, Progetto, Data Inizio) con vincolo di integrità referenziale tra **Impiegato** e la relazione IMPIEGATO e tra **Progetto** e la relazione PROGETTO.

Esercizio 8.3

Tradurre lo schema Entità-Relazione ottenuto nell'esercizio 7.6 in uno schema del modello relazionale.

Soluzione:

Questo è lo schema ottenuto nell'esercizio 7.6



Traduzioni:

ARBITRO(Cognome, Nome, Città, Regione)

GIORNATA(Numero, Serie, Giorno, Mese, Anno)

SQUADRA(Nome, Città, Allenatore)

GIOCATORE(Codice Fiscale, Cognome, Nome, Ruolo, Città di Nascita, Squadra) con vincolo di integrità referenziale tra **Squadra** e la relazione SQUADRA.

PARTITA(Numero, DNumero, DSerie, Risultato, Arbitro, Casa, Ospite) con vincoli di integrità referenziale tra **DNumero** e **DSerie** e la relazione GIORNATA, tra **Arbitro** e ARBITRO e tra **Casa** e **Ospite** con la relazione SQUADRA.

PARTITA IN CAMPO NEUTRO(Partita, Numero, Serie, Motivo, Città) con vincoli di integrità referenziale tra **Partita**, **Numero** e **Serie** con la relazione PARTITA.

PARTITA RINVIATA(Partita, Numero, Serie, Data) con vincoli di integrità referenziale tra **Partita**, **Numero** e **Serie** con la relazione PARTITA.

POSIZIONE(Squadra, Numero, Serie, Punteggio) con vincoli di integrità referenziale tra **Squadra** e la relazione SQUADRA e tra **Numero** e **Serie** e la relazione GIORNATA.

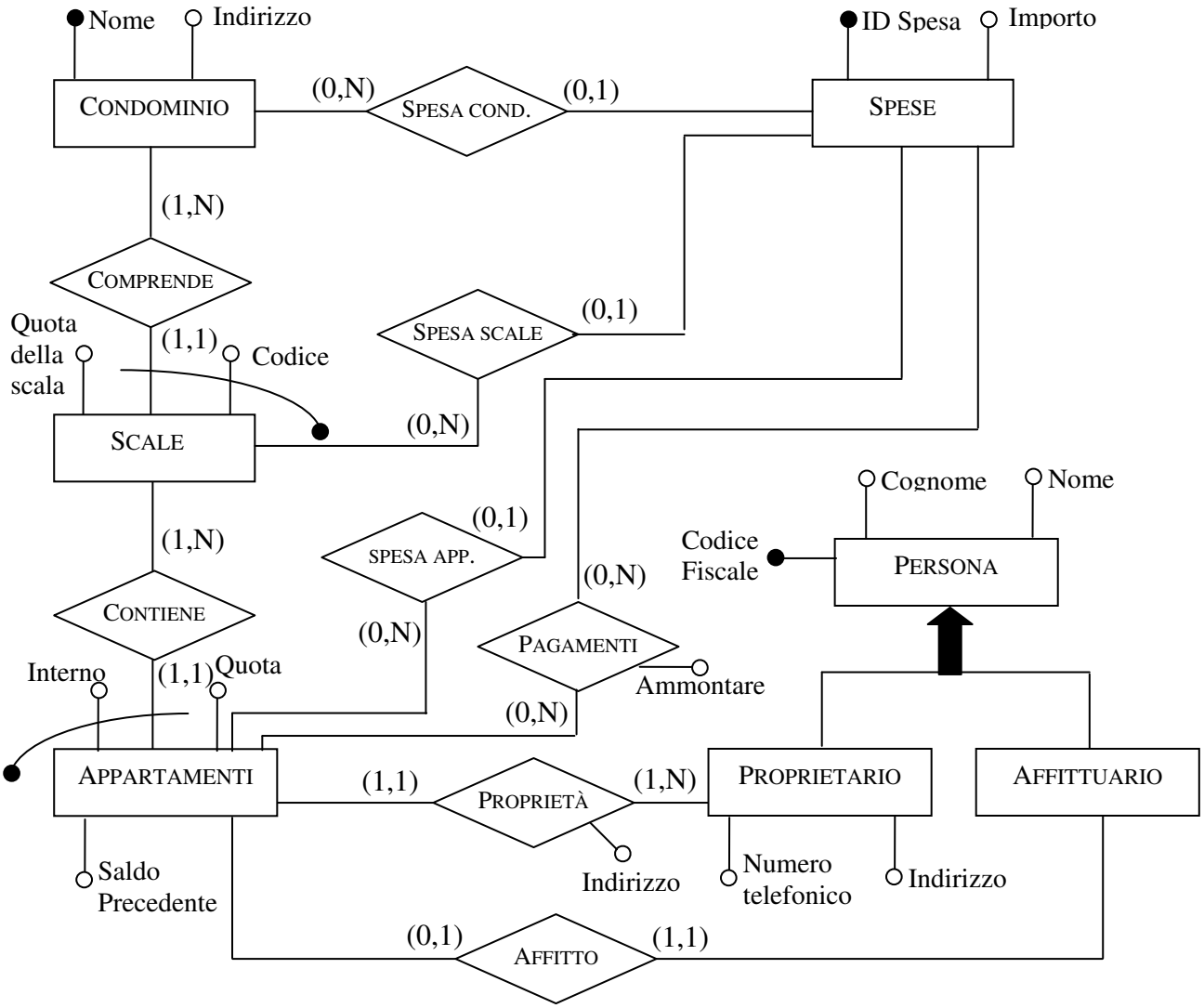
PARTECIPAZIONE(Giocatore, Partita, Numero, Serie, Ruolo) con vincoli di integrità referenziale tra **Giocatore** e la relazione GIOCATORE e tra **Partita**, **Numero**, **Serie** e la relazione PARTITA.

Esercizio 8.4

Definire uno schema logico relazionale corrispondente allo schema E-R ottenuto nell'esercizio 7.10. Per la fase di ristrutturazione, indicare le possibili alternative e sceglierne poi una, facendo assunzioni sui parametri quantitativi. Come riferimento per i parametri principali, assumere che la base di dati riguardi cento condomini, mediamente con cinque scale ciascuno, e che ogni scala abbia mediamente venti appartamenti e che le registrazioni principali siano la registrazione di una spesa (cinquanta all'anno per condominio più dieci per scala e cinque per appartamento) e di un pagamento (dieci all'anno per appartamento); annualmente viene stilato il bilancio di ciascun condominio, con il totale degli accrediti e degli addebiti per ciascun appartamento e quindi il calcolo del nuovo saldo (la stampa di ciascun bilancio deve essere organizzata per scale e ordinata).

Soluzione:

Questo è lo schema ottenuto nell'esercizio 7.10



Supponendo di avere 100 condomini i volumi del database sono:

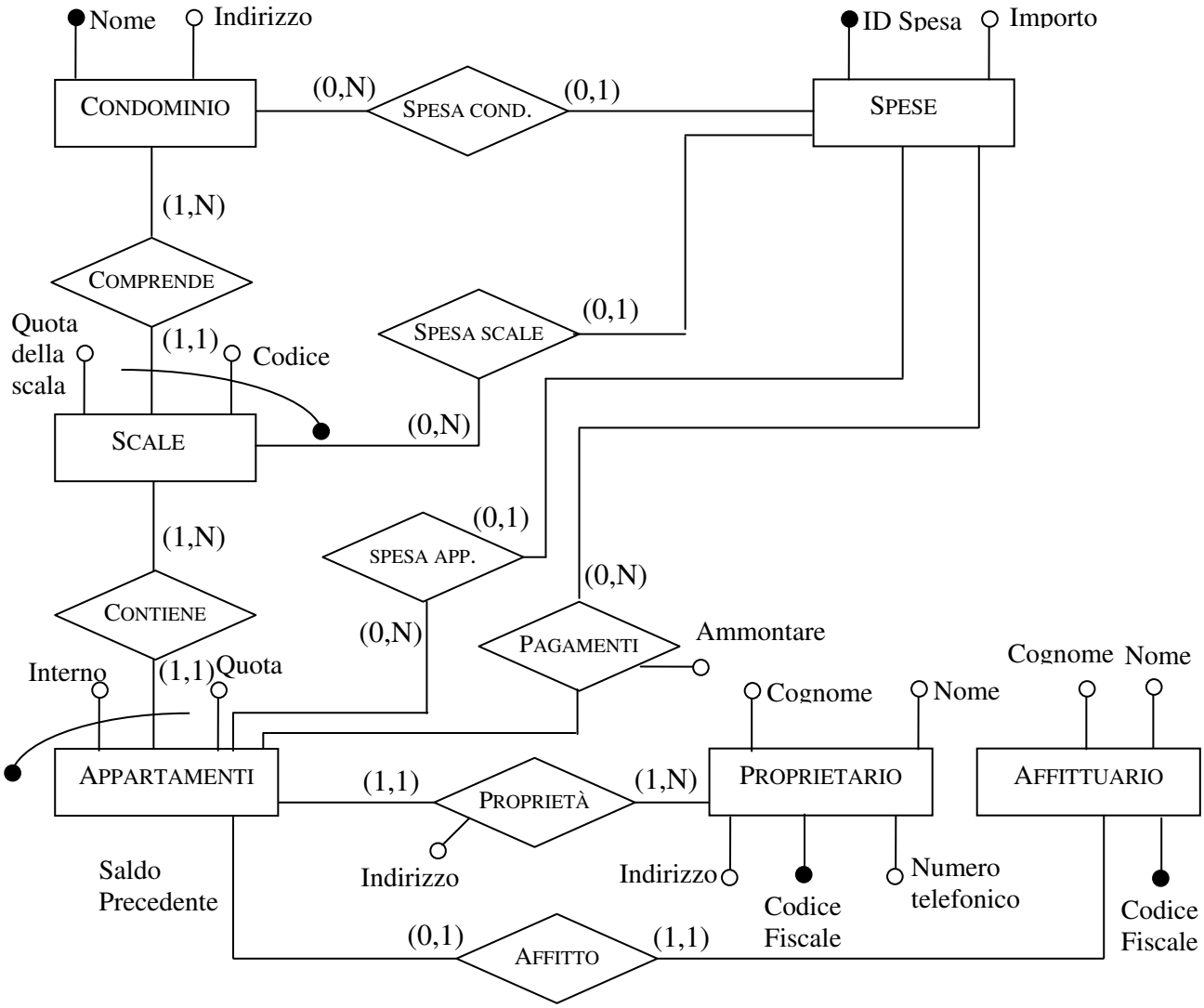
Concetto	**Tipo**	**Volume**
Condominio	E	100
Scale	E	500
Appartamento	E	10.000
Spese	E	60.000
Pagamento	E	100.000
Persona	E	10.000
Proprietario	E	8000
Comprende	R	500
Contiene	R	10.000
Spesa cond	R	5.000
Spesa scale	R	5.000
Spesa App	R	50.000
Pagamenti	R	100.000

Viste le tavole dei volumi e le operazioni effettuate, possiamo procedere nella ristrutturazione dello schema.

Eliminazione delle gerarchie:

Nello schema è presente una sola gerarchia ed è relativa alle persone ed è totale ed esclusiva, in quanto una persona se è proprietaria di un appartamento non ne è contemporaneamente anche affittuario, quindi si decide di lasciare due entità distinte: l'entità PROPRIETARIO e AFFITTUARIO.

Il nuovo schema sarà il seguente:



Traduzione nel modello relazionale:

CONDOMINIO(Nome, Indirizzo)
SPESE(ID Spesa, Importo)
PROPRIETARIO(Codice Fiscale, Cognome, Nome, Indirizzo, Numero Telefonico)
AFFITTUARIO(Codice Fiscale, Cognome, Nome)

SCALE(Codice, Nome, Quote della scala)
APPARTAMENTI(Interno, Codice, Nome, Quota, CF Proprietario, CF Affittuario)

SPESA COND(ID Spesa, Nome)
SPESA SCALE(ID Spesa, Nome, Codice)
SPESA APP(ID Spesa, Nome, Codice, Interno)
PAGAMENTI(ID Spesa, Nome, Codice, Interno, Ammontare)

Esercizio 8.5

Tradurre lo schema Entità-Relazione in figura 8.37 in uno schema di basi di dati relazionale. Per ciascuna relazione (dello schema relazionale) si indichi la chiave (che si può supporre unica) e, per ciascun attributo, si specifichi se sono ammessi valori nulli (supponendo che gli attributi dello schema E-R non ammettano valori nulli).

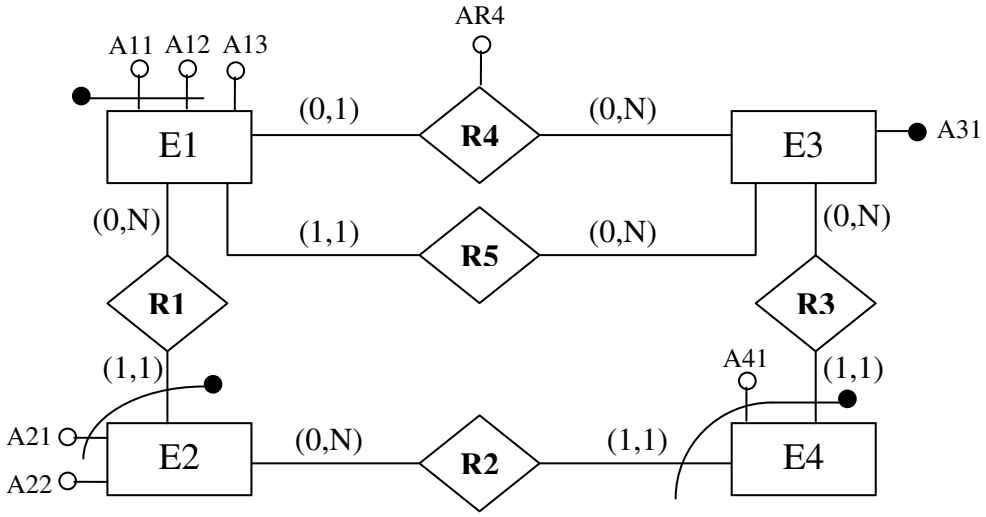


Figura 8.37 Uno schema E-R da tradurre

Soluzione:

Per prima cosa, traduciamo ciascuna entità con una relazione. La traduzione delle entità dotate di identificatore interno è immediata.
E1(A11, A13, A13)
E3(A31)

Traduciamo ora le entità con le identificazioni esterne. Otteniamo le seguenti relazioni:
E2(A21, A11, A12, A22)
E4(A41, A31, A21, A11, A12)

Passiamo ora alla traduzione delle associazioni. Le associazioni R1, R2 e R3 sono già state tradotte come conseguenza dell'identificazione esterna di E2 ed E4.

- Per tradurre R4, introduciamo con opportune ridenominazioni gli attributi che identificano E3 tra quelli di E1, nonché l'attributo AR4 proprio di R4; in pratica, introduciamo A31R4 e AR4. Data la natura della relazione (0,N), per questi attributi sono ammessi valori nulli.
- Per tradurre R5, analogamente al caso precedente, introduciamo A31R5 in E1. In questo caso non sono ammessi valori nulli.

Lo schema relazionale ottenuto è il seguente:
E1(A11, A13, A13, A31R4*, AR4*, A31R5)
E2(A31)
E2(A21, A11, A12, A22)
E4(A41, A31, A21, A11, A12)

Esercizio 8.6

Sia dato il seguente schema Entità-Relazione in figura 8.38. Ristrutturare lo schema, eliminando le gerarchie, supponendo che le operazioni più significative siano le seguenti, ciascuna eseguita 10 volte al giorno:

Operazione 1: Accesso agli attributi A_{21}, A_{22}, A_{11}, A_{12}, A_{13} dell'entità E_2;
Operazione 2: Accesso agli attributi A_{41}, A_{42}, A_{31}, A_{11}, A_{12}, A_{13} dell'entità E_4
Operazione 3 Accesso agli attributi A_{51},A_{52}, A_{31} , A_{11}, A_{12}, A_{13} dell'entità E_5;

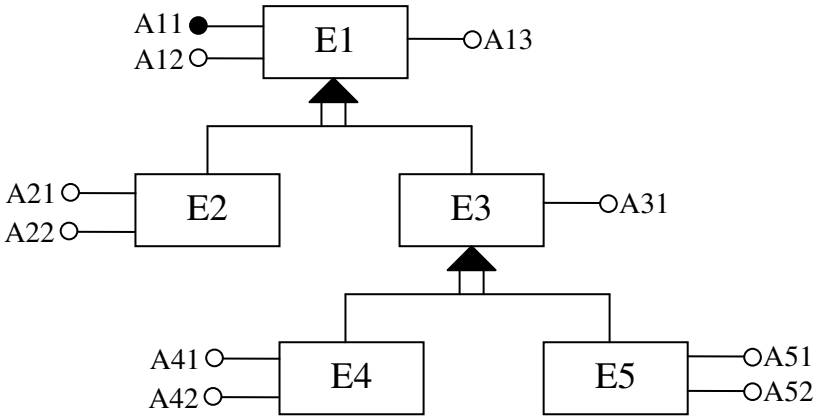


Figura 8.38 Uno schema E-R con generalizzazioni

Soluzione:

Lo schema precedente può essere ristrutturato in vari modi, dipendenti dal volume delle tabelle e dalla complessità che si vuole ottenere.

Tenendo conto delle possibili sovrapposizioni tra le popolazioni E2, E4, E5, la soluzione migliore consiste nel ristrutturare la gerarchia in un'unica entità. Questa soluzione è la più semplice in assoluto ed ha il pregio di avere solamente 30 accessi giornalieri. Di contro, l'entità ha la presenza di valori nulli e il conseguente spreco di memoria.

E(<u>A11</u>, A12, A13, A21*, A22*, A31*, A41*, A42*, A51*, A52*)

Esercizio 8.7

Si consideri lo schema concettuale di Figura 8.39, che descrive i dati di conti correnti bancari. Si osservi che un cliente può essere titolare di più conti correnti e che uno stesso conto corrente può essere intestato a diversi clienti. Si supponga che su questi dati, sono definite le seguenti operazioni principali:

Operazione 1: Apri un conto corrente ad un cliente.
Operazione 2: Leggi il saldo totale di un cliente.
Operazione 3: Leggi il saldo di un conto.
Operazione 4: Ritira i soldi da un conto con una transazione allo sportello.
Operazione 5: Deposita i soldi in un conto con una transazione allo sportello.
Operazione 6: Mostra le ultime 10 transazioni di un conto.
Operazione 7: Registra transazione esterna per un conto.
Operazione 8: Prepara rapporto mensile dei conti.
Operazione 9: Trova il numero dei conti posseduti da un cliente.
Operazione10: Mostra le transazioni degli ultimi 3 mesi dei conti delle società con saldo negativo.

Si supponga infine che, in fase operativa, i dati di carico per questa applicazione bancaria siano quelli riportati in figura 8.40.
Effettuare la fase di progettazione logica sullo schema E-R tenendo conto dei dati forniti. Nella fase di ristrutturazione si tenga conto del fatto che sullo schema esistono due ridondanze: Gli attributi **Saldo Totale** e **Numero di Conti** dell'entità CLIENTE. Essi possono infatti essere derivati dall'associazione TITOLARITÀ e dall'entità CONTO.

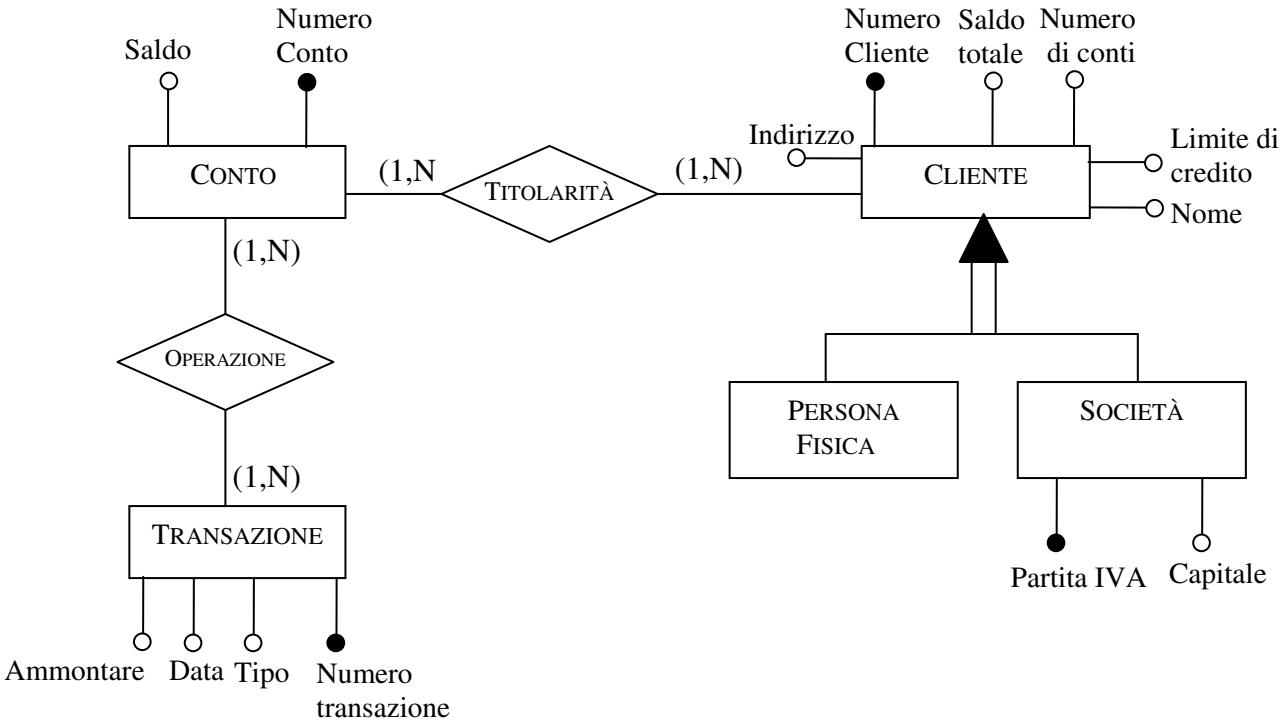


Figura 8.40 Uno schema E-R da tradurre

Tavola dei volumi

Concetto	Tipo	Volume
Cliente	E	15000
Conto	E	20000
Transazione	E	600000
Persona Fisica	E	14000
Società	E	1000
Titolarità	R	30000
Operazione	R	800000

Tavola delle operazioni

Operazione	Tipo	Frequenza
Op. 1	I	100/giorno
Op. 2	I	2000/giorno
Op. 3	I	1000/giorno
Op. 4	I	2000/giorno
Op. 5	I	1000/giorno
Op. 6	I	200/giorno
Op. 7	B	1500/giorno
Op. 8	B	1/mese
Op. 9	B	75/giorno
Op. 10	I	20/giorno

Tavola 8.40 Tavole dei volumi e delle operazioni per lo schema in figura 8.39

Soluzione:

Analisi delle ridondanze:
Nello schema esistono 2 dati ridondanti: **Saldo totale** e **Numero di Conti**.

Saldo totale:
Ipotizzando che l'attributo saldo totale sia di tipo float (32 bit e quindi 4 bytes per ogni occorrenza), l'utilizzo di questo dato richiederebbe 4*15.000 bytes, con un utilizzo di memoria pari a 60 KB.
Le operazioni coinvolte con questo dato sono la 2, la 4, la 5, la 7 e la 8.
Si procede analizzando il costo per ognuna di queste operazioni, non conteggiando l'operazione 8 che viene svolta in batch una sola volta al mese.

- Con dato ridondante
  Per l'operazione 2 abbiamo: 1x2.000 accessi in lettura = 2.000 accessi al giorno
  Per l'operazione 4 abbiamo: 1x2.000 accessi in lettura (leggo il saldo totale) + 2x2.000 accessi in scrittura (scrivo il nuovo saldo) = 6.000 accessi al giorno
  Per l'operazione 5 abbiamo: 1x1.000 accessi in lettura (leggo il saldo totale) + 2x1.000 accessi in scrittura (scrivo il nuovo saldo) = 3.000 accessi al giorno
  Per l'operazione 7 abbiamo: 1x1.500 accessi in lettura (leggo il saldo totale) + 2x1.500 accessi in scrittura (scrivo il nuovo saldo) = 4.500 accessi al giorno

- Senza dato ridondante
  Per l'operazione 2 abbiamo: ipotizzando che la query di ricerca dei conti di un cliente abbia un costo pari a 20 accessi in lettura, moltiplicato per 2.000 operazioni al giorno, ottengo 40.000 accessi al giorno.
  Per l'operazione 4 abbiamo: Nessun accesso aggiuntivo
  Per l'operazione 5 abbiamo: Nessun accesso aggiuntivo
  Per l'operazione 7 abbiamo: Nessun accesso aggiuntivo

In conclusione, il dato ridondante ho 15.000 accessi, mentre senza il dato ridondante ho 40.000 accessi al giorno. Il dato ridondante mi fa risparmiare 25.000 accessi a fronte di 60 KB di memoria.

Numero di conti:
Per quanto riguarda il numero di conti posseduto da un cliente, l'utilizzo di memoria col dato ridondante è di 1 byte per cliente, che equivale a 15 KB di memoria.
Le operazioni coinvolte sono la 1 e la 9.
Anche senza svolgere i conti, si può notare che l'utilizzo del dato è di sole 75 volte al giorno e in modalità batch con l'operazione 9.
Il conseguente miglioramento di efficienza sarà nell'ordine di un migliaio di accessi in meno al giorno. Sarà quindi a discrezione del progettista l'utilizzo o meno del dato.
In questo caso ipotizziamo quindi di non ritenerlo necessario.

Eliminazione delle gerarchie:

Nello schema è presente una sola gerarchia relativa all'entità CLIENTE, che viene distinto in PERSONA FISICA o SOCIETÀ. L'entità SOCIETÀ ha gli attributi Partita IVA e Capitale che la distinguono. L'unica operazione che fa una distinzione sul tipo di cliente è la numero 10.
Visto lo scarso numero di operazioni e il poco spazio necessario per accorpare le due entità, si decide di accorpare gli attributi Partita IVA e Capitale in Cliente. Sarà l'attributo Partita IVA ad identificare un cliente come società.

Scelta degli identificatori principali:

Gli identificatori sono Numero transazione per l'entità TRANSAZIONE, Numero Conto per l'entità CONTO.
Per quanto riguarda l'entità CLIENTE, l'identificatore è l'attributo Numero cliente; l'attributo Partita Iva identifica le società e se presente deve essere univoco.

Schema relazionale

TRANSAZIONE(Numero transazione, Tipo, Data, Ammontare)
CONTO(Numero Conto, Saldo)
CLIENTE(Numero cliente, Saldo Totale, Limite di credito, Nome, Indirizzo, Partita IVA*, Capitale*)
OPERAZIONE(Numero conto, Numero transazione)
TITOLARITÀ(Numero conto, Numero cliente)

Esercizio 8.8 Si consideri lo schema concettuale della figura 8.41, nel quale l'attributo Saldo di una occorrenza di CONTOCORRENTE è ottenuto come somma dei valori dell'attributo Importo per le occorrenze di OPERAZIONE ad essa correlate tramite la relationship MOVIMENTO.

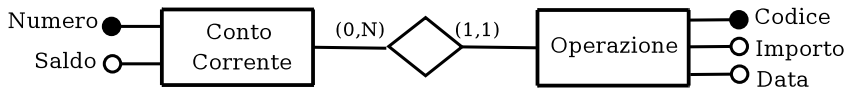


Figura 8.41 Schema per l'esercizio 8.8

Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità delle due entità sono L_{CC} = 2.000 e L_{OP} =20.000 e che le operazioni più importanti sono:

- OP$_1$ scrittura di un movimento, con frequenza f_1 = 10
- OP$_2$ lettura del saldo con frequenza f_2 = 1000.

Soluzione
Consideriamo sia gli accessi ad occorrenze di entità sia ad occorrenze di relationship e contando doppio il costo degli accessi in scrittura. Il costo complessivo è sempre pari a $c_1 \times f_1 + c_2 \times f_2$, dove c_1 e c_2 sono i costi delle singole esecuzioni delle operazioni.
In presenza di ridondanza:

- Op_1: l'operazione di scrittura di un movimento ha costo c_1 pari a 7 (una lettura e tre scritture)
- Op_2: l'operazione di lettura del saldo ha un costo c_2 pari a 1

e quindi si rileva un costo complessivo pari a $1 \times f_2 + 7 \times f_1 = 1.000 + 70 = 1070$.
In assenza di ridondanza:

- Op_1: l'operazione di scrittura di un movimento ha un costo c_1 pari a 4 (due scritture)
- Op_2: l'operazione di lettura del saldo ha un costo c_2 pari al doppio del numero medio di movimenti per conto corrente e cioè $2 \times L_{OP}/L_{CC}$

e quindi il costo complessivo è pari a $2 \times L_{OP}/L_{CC} \times f_2 + 4 \times f_1 = 2 \times 10 \times 1.000 + 4 \times 10 = 20.040$.

Esercizio 8.9 Lo schema concettuale della figura 8.42 rappresenta un insieme di viaggi e un insieme di partecipanti a questi viaggi. Ogni viaggio ha diversi partecipanti e la stessa persona può partecipare a più viaggi. Nello schema l'attributo incasso è ridondante perché può essere ottenuto moltiplicando il costo del viaggio per il numero di partecipanti (cioè il prodotto del valore dell'attributo Costo di ogni occorrenza dell'entità viaggio per il numero di occorrenze dell'entità Partecipante a cui è correlato tramite la relazione V-P).

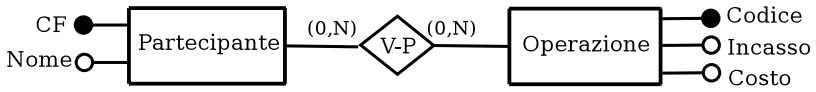


Figura 8.42 Schema per l'esercizio 8.9

Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità dei concetti in gioco sono NViaggio=20.000, NV-P=300.000 e NPartecipante = 100.000 e che le operazioni più importanti sono:

- Op1 calcolo dell'incasso di un viaggio, con frequenza $f_1 = 10$ al mese
- Op2 inserimento di un partecipante al viaggio, con frequenza $f_2 = 5$ al giorno

Assumere che il costo di una lettura e quello di una scrittura siano uguali e che un mese sia di 20 giorni lavorativi.

Soluzione

Senza Ridondanza

Op1: mediamente c_1 = 1 + NV-P / NViaggio = 16

- 1 per entità Viaggio
- 1 per ciascuna occorrenza di V-P cui essa partecipa (e sono NV-P /NViaggio = 15)

Op2: c_2 = 2 (un inserimento di Partecipante e uno di V-P)

In totale (in un mese) $f_1 \times c_1 + f_2 \times c_2 = 10 \times 16 + 5 \times 20 \times 2 = 360$

Con Ridondanza

Op1: c_1 = 1 (l'incasso è nell'entità Viaggio)

Op2: c_2 = 4 (i due inserimenti come sopra più lettura del vecchio valore di Incasso e scrittura del nuovo)

In totale (in un mese) $f_1 \times c_1 + f_2 \times c_2 = 10 \times 1 + 5 \times 20 \times 4 = 410$

Esercizio 8.10 Considerare un frammento di schema E-R contenente le entità E_0 (con attributi $A_{0,1}$, identificante, e $A_{0,2}$), E_1 (con attributo $A_{1,1}$), E_2 (con attributo $A_{2,1}$), E_3 (con attributo $A_{3,1}$), E_4 (con attributo $A_{4,1}$) e due generalizzazioni, la prima totale con genitore E_0 e figlie E_1 ed E_2 e la seconda parziale con genitore E_1 e figlie E_3 ed E_4. Supporre paragonabili fra loro le dimensioni degli attributi. Indicare, per ciascuno dei casi seguenti, considerati separatamente, la scelta (o le scelte, qualora ve ne siano diverse paragonabili) che si ritiene preferibile per l'eliminazione delle generalizzazioni nella progettazione logica:

1. le operazioni nettamente più frequenti sono due, che accedono rispettivamente a tutte le occorrenze di E_1 (con stampa dei valori di $A_{0,1}$, $A_{0,2}$ e $A_{1,1}$) e a tutte le occorrenze di E_2 (con stampa dei valori di $A_{0,1}$, $A_{0,2}$ e $A_{2,1}$);
2. le operazioni nettamente più frequenti sono due, che accedono rispettivamente a tutte le occorrenze di E_1 (con stampa dei valori di $A_{0,1}$, $A_{1,1}$ e, se esiste, $A_{3,1}$) e a tutte le occorrenze di E_2 (con stampa dei valori di $A_{0,1}$ e $A_{2,1}$);
3. l'operazione nettamente più frequente prevede l'accesso a tutte le occorrenze di E_0 (con stampa dei valori di $A_{0,1}$, $A_{0,2}$);
4. l'operazione nettamente più frequente prevede l'accesso a occorrenze (tutte o alcune) di E_0 (con stampa dei valori di tutti gli attributi, inclusi quelli di tutte le altre entità, ove applicabili).

Soluzione
Le soluzioni di questo esercizio sono riportate rispettivamente nelle figure: 8.I, 8.II, 8.III e 8.IV.

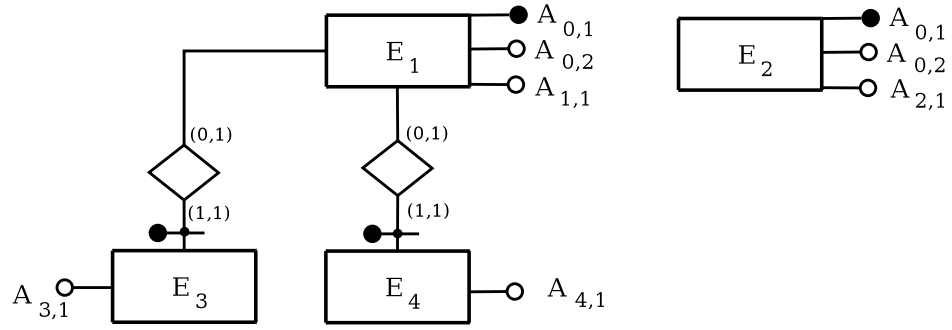


Figura 8.I Soluzione dell'esercizio 8.10, punto 1.

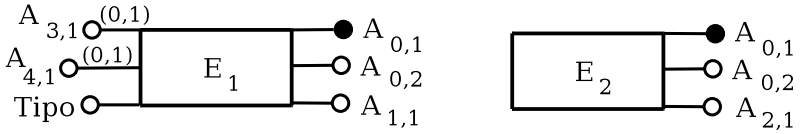


Figura 8.II Soluzione dell'esercizio 8.10, punto 2.

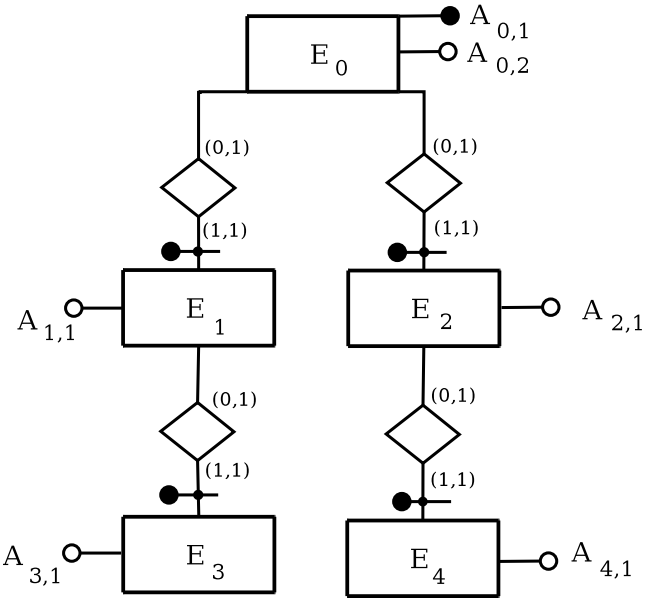


Figura 8.III Soluzione dell'esercizio 8.10 punto 3.

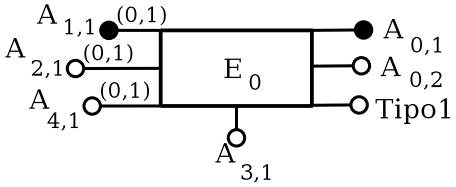


Figura 8.IV Soluzione dell'esercizio 8.10 punto 4.

Esercizio 8.11 Mostrare uno schema E-R che descriva una realtà di interesse corrispondente a quella rappresentata da uno schema relazionale composto dalle seguenti relazioni:

- CICLISTA (Codice, Cognome, Nome, Squadra)
- COMPETIZIONE (Codice, Nome, Organizzatore, KmTotali)
- TAPPA (Numero, Competizione, Partenza, Arrivo, KM) con vincolo di integrità referenziale fra Competizione e COMPETIZIONE
- CLASSIFICATAPPA (NumeroTappa, Competizione, Ciclista, Posizione, Distacco)con vincoli di integrità referenziale fra gli attributi NumeroTappa, Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA
- CLASSIFICAGENERALE (NumeroTappa, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa, Competizione e la relazione TAPPA e fra Ciclista e la relazione CICLISTA.

Indicare quali sono i vincoli non espressi nello schema modellato.
Soluzione
La soluzione di questo esercizio è riportata in figura 8.V.

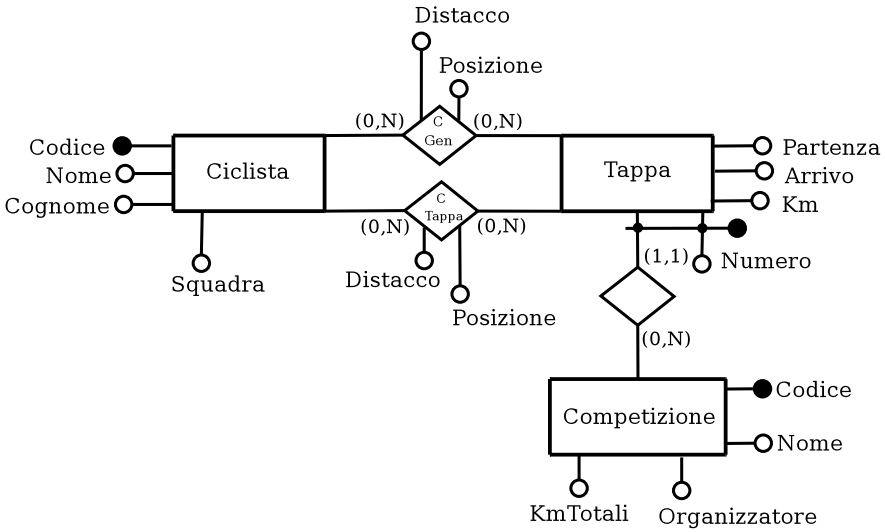


Figura 8.V Soluzione dell'esercizio 8.11.

Vincoli non espressi:

- se un ciclista compare nella classifica generale di una tappa deve comparire anche nella classifica della tappa stessa;
- se un ciclista compare nella classifica generale di una tappa deve comparire nella classifica generale e di tappa di tutte le tappe precedenti;
- i Km totali della competizione devono essere pari alla somma dei chilometri di ogni tappa della competizione.

Esercizio 8.12 Per ciascuno dei seguenti schemi logici (in cui A* indica che l'attributo A ammette valori nulli), mostrare uno schema concettuale dal quale possa essere stato ottenuto (indicando anche cardinalità e identificatori).
Schema (a):

- LIBRI (Codice, Titolo, Genere*, Autore) con vincolo di integrità referenziale fra Autore e la relazione SCRITTORI
- EDIZIONI (Libro, Editore, Collana*, Anno) con vincoli di integrità referenziale fra Libro e la relazione LIBRI e fra Editore e la relazione EDITORI
- EDITORI (Nome, Città)
- SCRITTORI (Codice, Cognome, Nome).

Schema (b):

- LIBRI (Codice, Titolo, Genere*) con vincolo di integrità referenziale fra Genere e la relazione GENERI
- EDIZIONI (Libro, Editore, Collana*, Anno) con vincoli di integrità referenziale fra Libro e la relazione LIBRI, fra Editore e la relazione EDITORI e fra Collana e la relazione COLLANE
- AUTORI (Libro, Scrittore) con vincoli di integrità referenziale fra Libro e la relazione LIBRI e fra Scrittore e la relazione SCRITTORI
- COLLANE (SiglaCollana, Nome)
- GENERI (SiglaGenere, Nome)
- EDITORI e SCRITTORI come nello schema (a).

Soluzione
Lo schema (a) è riportato nella figura 8.VI, lo schema (b) è riportato nella figura 8.VII.

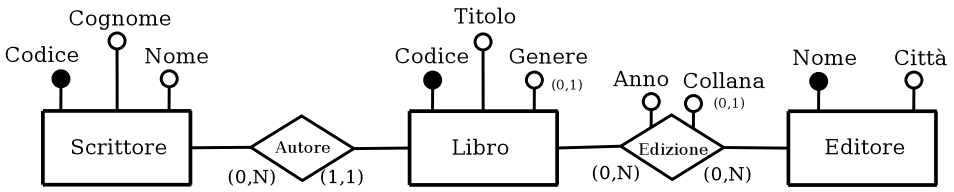


Figura 8.VI Soluzione dell'esercizio 8.12 - Schema (a)

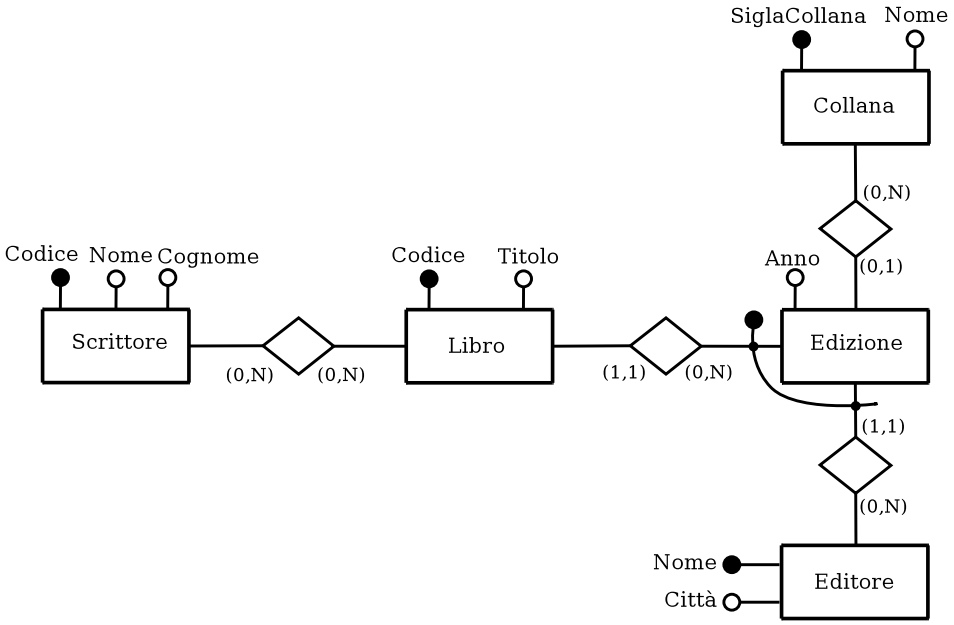


Figura 8.VII Soluzione dell'esercizio 8.12 - Schema (b)

Esercizio 8.13 Per ciascuno dei seguenti schemi logici (in cui A* indica che l'attributo A ammette valori nulli), mostrare uno schema concettuale dal quale possa essere stato ottenuto (indicando anche cardinalità e identificatori).
Schema (a):

- CASECOSTRUTTRICI (Codice, Nome, Nazione*)
- MODELLI (Casa, Nome, Categoria*) con vincolo di integrità referenziale fra Casa e la relazione CASECOSTRUTTRICI
- AUTOMOBILI (Targa, Casa, Modello, Anno, Proprietario) con vincoli di integrità referenziale fra gli attributi Casa e Modello e la relazione MODELLI e fra Proprietario e la relazione PERSONE
- PERSONE (CodiceFiscale, Cognome, Nome).

Schema (b):

- MODELLI e PERSONE come nello Schema(a)
- CASECOSTRUTTRICI (Codice, Nome, Nazione*) con vincolo di integrità referenziale fra Nazione e la relazione NAZIONI
- VERSIONI (Casa, Modello, CodiceVersione, Cilindrata) con vincolo di integrità referenziale fra gli attributi Casa e Modello e la relazione MODELLI
- AUTOMOBILI(Targa, Casa, Modello, Versione, Anno) con vincolo di integrità referenziale fra gli attributi Casa, Modello, Versione e la relazione VERSIONI
- ACQUISTO (Auto, Data, Acquirente)con vincoli di integrità referenziale fra Auto e la relazione AUTOMOBILI e fra Acquirente e la relazione PERSONE
- NAZIONI (SiglaNazione, Nome).

Soluzione
Lo schema (a) è riportato nella figura 8.VIII, lo schema (b) è riportato nella figura 8.IX.

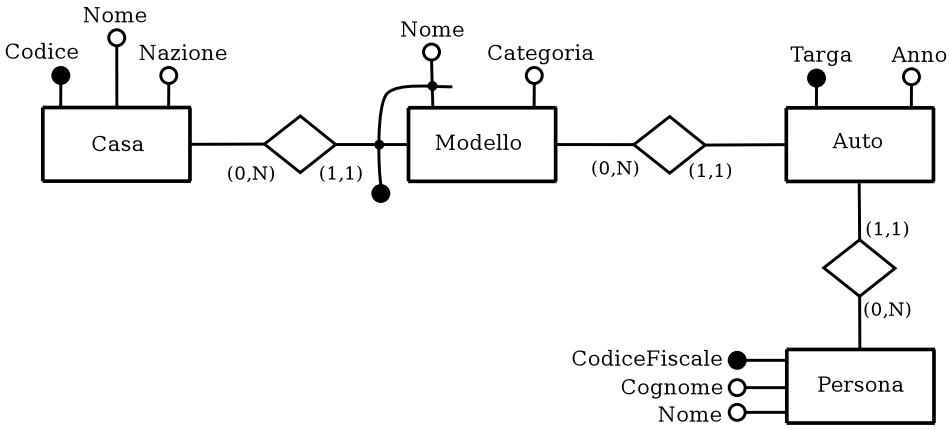


Figura 8.VIII Soluzione dell'esercizio 8.13 - Schema (a)

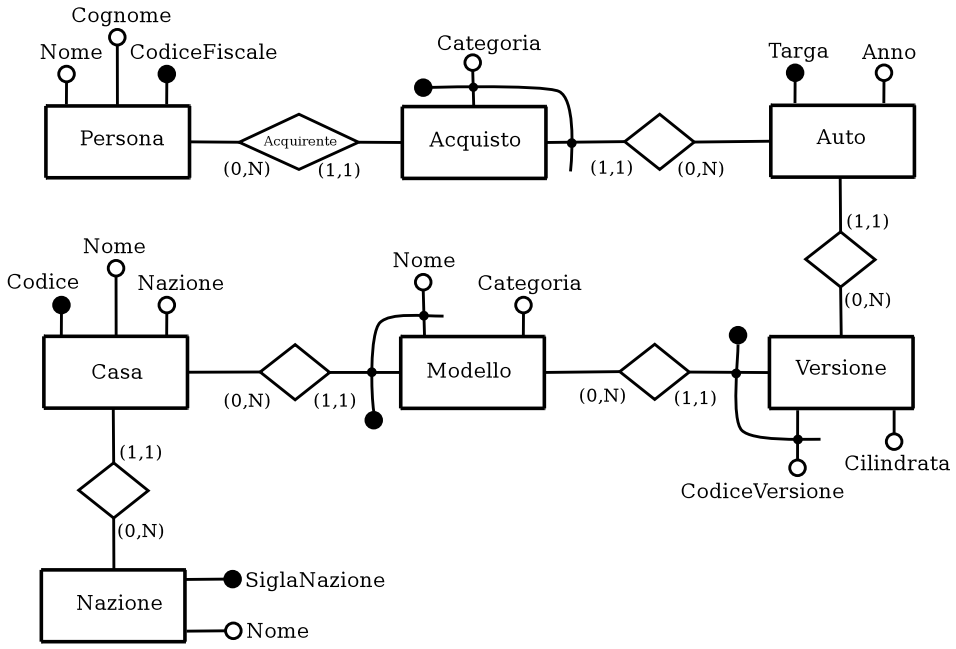


Figura 8.IX Soluzione dell'esercizio 8.13, Schema (b)

Esercizio 8.14 Progettare lo schema logico relazionale corrispondente allo schema concettuale definito nell'esercizio 7.14 mostrando i nomi degli attributi, i vincoli di chiave e di integrità referenziale.
Soluzione

- PASSEGGERI (Codice, Cognome, Nome, Telefono)
- PRENOTAZIONI (Codice) con vincoli di integrità referenziale fra l'attributo CodPass e al relazione PASSEGGERI e fra l'attributo CodPrenot e la relazione PRENOTAZIONI
- PRENOTAZIONIPASSEGGERI (CodPass, CodPrenot)
- SEGMENTIPRENOTAZIONI (CodicePrenot, Numero, CodComp, NumVolo, Data, CodClasse) con vincolo di integrità referenziale fra CodPrenot e la relazione PRENOTAZIONI
- AEROPORTI (Codice, Nome, Città)
- COMPAGNIE (Codice, Nome)
- VOLISPECIFICI (CodComp, NumVolo, Data, Orario) con vincolo di integrità referenziale fra gli attributi CodComp, NumVolo e la relazione VOLOASTRATTO
- VOLIASTRATTI (CodComp, NumVolo, Da, A, CodAeromobile) con vincoli di integrità referenziale fra l'attributo CodComp e la relazione COMPAGNIE fra l'attributo A e la relazione AEROPORTI fra l'attributo Da e la relazione AEROPORTI e fra l'attributo CodAeromobile e la relazione AEROMOBILI
- AEROMOBILI (Codice, Descrizione)
- CLASSE (Codice, Descrizione).

Esercizio 8.15 Mostrare uno schema logico che possa essere ottenuto dallo schema E-R in figura 8.43.

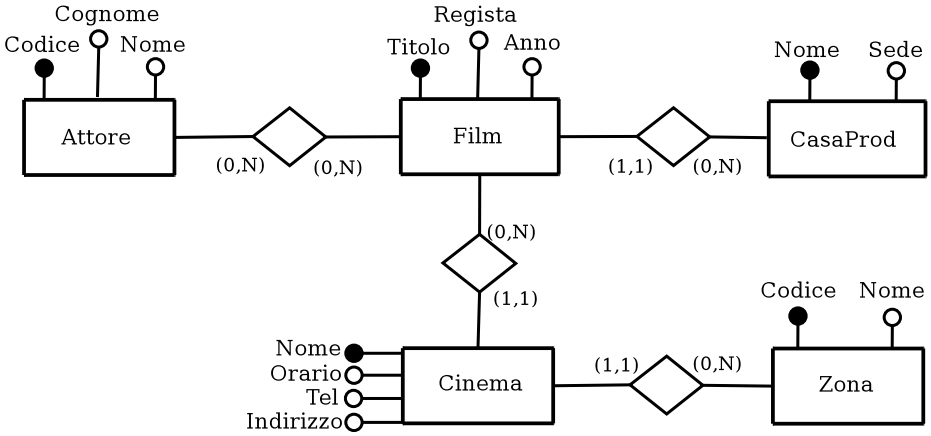


Figura 8.43 Schema per l'esercizio 8.15.

Soluzione

- ATTORI (Codice, Cognome, Nome)
- FILM (Titolo, Regista, Anno)
- ATTORIFILM (Attore, Film) con vincoli di integrità referenziale fra l'attributo Attore e la relazione ATTORI e fra l'attributo Film e la relazione FILM
- CASEPRODUTTRICI (Nome, Sede)
- ZONE (Codice, Nome)
- CINEMA (Nome, Orario, Tel, Indirizzo, Zona, Film) con vincoli di integrità referenziale fra l'attributo Film e la relazione FILM e fra l'attributo Zona e la relazione ZONE.

Esercizio 8.16 Tradurre lo schema E-R della figura 8.44 nel corrispondente schema relazionale.

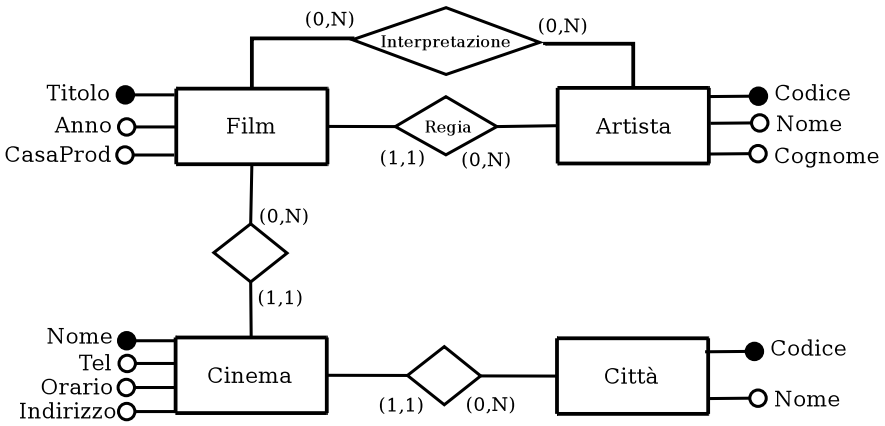


Figura 8.44 Schema per l'esercizio 8.16.

Soluzione

- ARTISTI (Codice, Nome, Cognome)
- FILM (Titolo, Anno, CasaProduttrice, Artista) con vincolo di integrità referenziale fra l'attributo Artista e la relazione ARTISTI
- INTERPRETAZIONI (Film, Artista)
- con vincoli di integrità referenziale fra l'attributo Artista e la relazione ARTISTI e fra l'attributo Film e la relazione FILM
- CITTÀ (Codice, Nome)
- CINEMA (Nome, Tel, Orario, Indirizzo, Città, Film) con vincoli di integrità referenziale fra l'attributo Film e la relazione FILM e fra l'attributo Città e la relazione CITTÀ.

Esercizio 8.17 Mostrare uno schema E-R che descriva una realtà di interesse corrispondente a quella rappresentata da uno schema relazionale composto dalle seguenti relazioni:

- UTENZE (Prefisso, Numero, CodiceCentrale, Titolare, Indirizzo, DataAttivazione) con vincoli di integrità referenziale fra gli attributi Prefisso e CodiceCentrale e la relazione CENTRALI e fra Utente e la relazione UTENTI
- CENTRALI (Distretto, Codice, Indirizzo, Capacità)
- UTENTI (CodiceFiscale, Cognome, Nome)
- DISTRETTI (Prefisso, Nome, Provincia), con vincolo di integrità referenziale fra l'attributo Provincia e la relazione PROVINCE
- PROVINCE (Sigla, Nome, Capoluogo)
- BOLLETTE (Prefisso, Numero, DataEmissione, Importo), con vincolo di integrità referenziale fra Prefisso, Numero e la relazione UTENZE
- PAGAMENTI (Codice, Prefisso, Numero, DataPagamento, Modalità, ImportoPagato, con vincolo di integrità referenziale fra Prefisso, Numero e la relazione UTENZE.

Soluzione
La soluzione di questo esercizio è illustrata in figura 8.X.

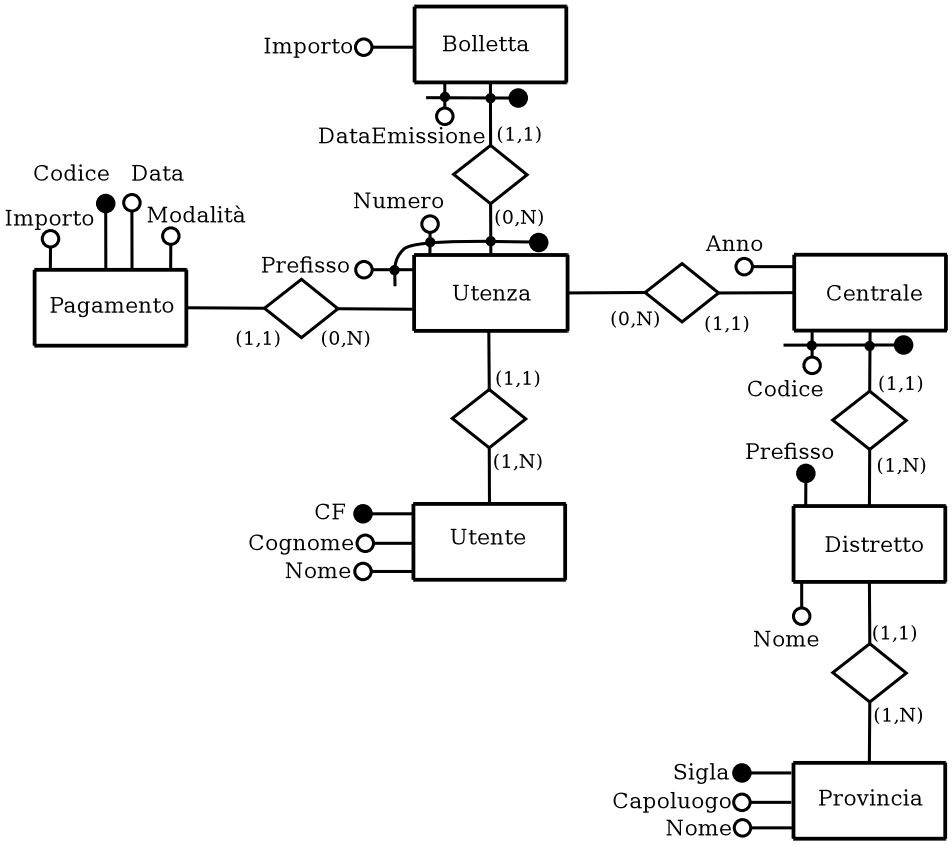


Figura 8.X Soluzione per l'esercizio 8.17

Esercizio 8.18 Modificare lo schema prodotto come soluzione all'esercizio precedente supponendo che, oltre alle utenze domestiche, siano descritte anche le utenze radiomobili, ognuna delle quali deve essere associata ad un'utenza domestica e quindi avere lo stesso titolare. Le bollette sono emesse con riferimento alle singole utenze. Le utenze radiomobili sono associate a pseudo-distretti, che corrispondono alle aree con lo stesso prefisso.

Soluzione

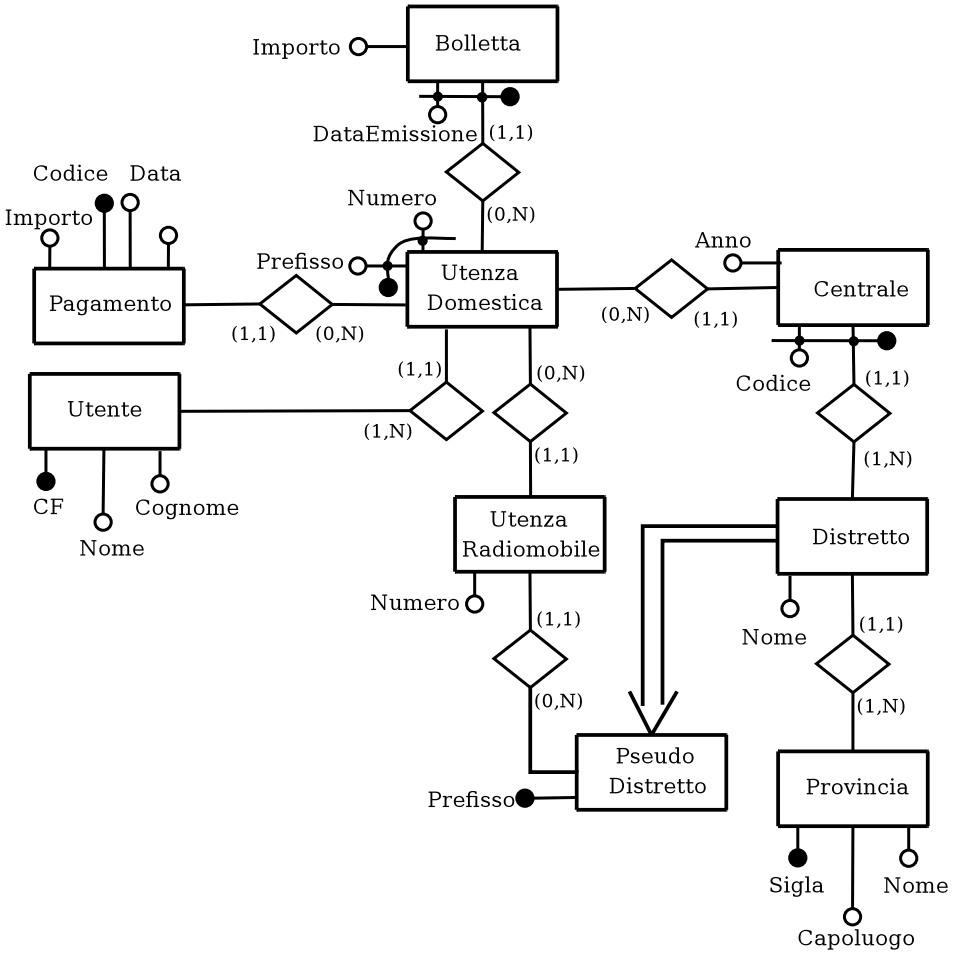


Figura 8.XI Soluzione per l'esercizio 8.18.

Esercizio 8.19 Mostrare gli schemi relazionali che si ottengono traducendo i tre schemi E-R della figura 6.X dell'esercizio 6.21 confrontando il risultato ottenuto con le considerazioni fornite a soluzione dell'esercizio 6.21.

Soluzione

- AUTISTA (ID)
- REPARTO (ID)
- VETTURA (ID)
- VETTAUTREP (IDAut, IDVett, IDRep)

- AUTISTA (ID)
- REPARTO (ID)
- VETTURA (ID)
- VETTAUT (IDAut, IDVett)
- AUTREP (IDAut, IDRep)

- AUTISTA (ID)
- REPARTO (ID)
- VETTURA (ID)
- AUTREP (IDAut, IDRep)
- VETTREP (IDVett, IDRep)

Esercizio 8.20 Dato lo schema relazionale seguente:

- ANIMALI (Id, Specie, Razza, NomeLatino)
- BESTIAME (Id, Specie, Razza, Allevamento)
- ALLEVAMENTO (Id, Nome, Indirizzo)

determinare un modello ER da cui potrebbe essere stato ottenuto facendo opportuno uso della generalizzazione.

Soluzione

La soluzione di questo esercizio è mostrata in figura 8.XII.

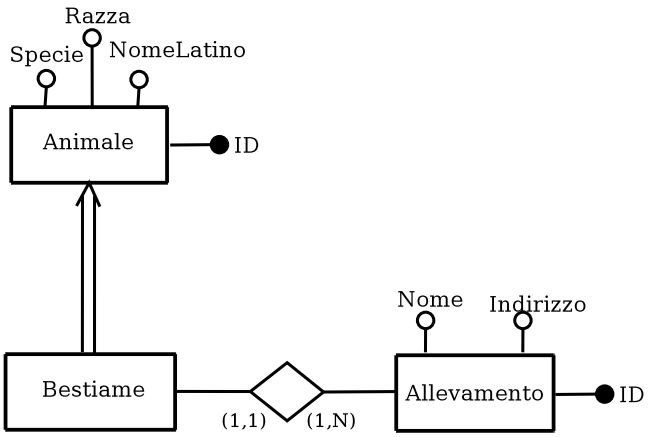


Figura 8.XII Figura di riferimento per l'esercizio 8.20

Esercizio 8.21 Mostrare lo schema relazionale che si ottiene dallo ER rappresentato in figura 8.XIII.

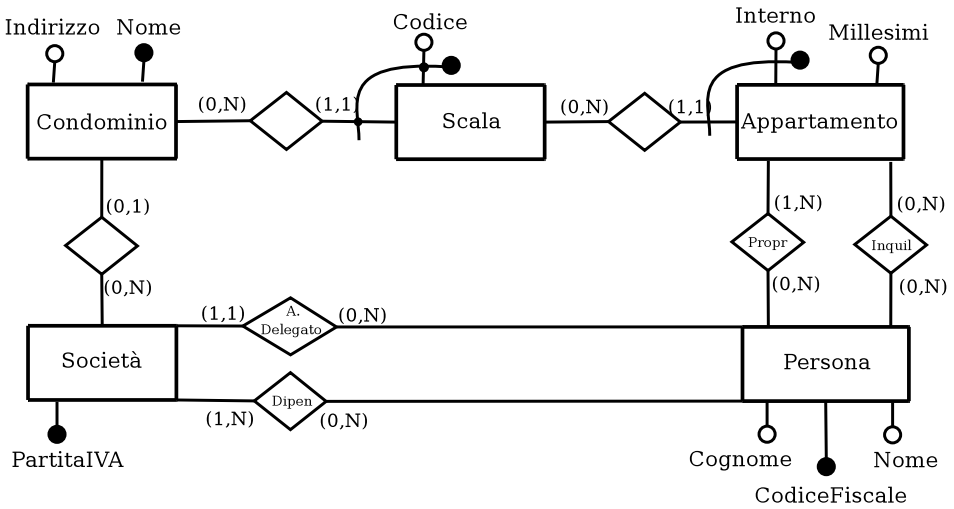


Figura 8.XIII Figura di riferimento per l'esercizio 8.20

Soluzione

- CONDOMINI (Nome, Indirizzo, PartitaIvaSocietà) con vincolo di integrità referenziale fra l'attributo PartitaIvaSocietà e la relazione SOCIETÀ
- SCALA (Codice, Condominio) con vincolo di integrità referenziale fra l'attributo Condominio e la relazione CONDOMINIO
- APPARTAMENTO (Interno, CodiceScala, Condominio, Millesimi) con vincolo di integrità referenziale fra gli attibuti CodiceScala, Condominio e la relazione CONDOMINI
- PROPRIETÀ (InternoAppartamento, CodiceScala, Condominio, CodiceFiscalePersona) con vincoli di integrità referenziae fra gli attributi InternoAppartamento, CodiceScala, Condominio e la relazione APPARTAMENTO e fra l'attributo CodiceFiscalePersona e la relazione PERSONA
- INQUILINO (InternoAppartamento, codiceScala, CodiceFiscalePersona) con vincoli di integrità referenziale fra gli attributi InternoAppartamento, CodiceScala, Condominio e la relazione APPARTAMENTO e fra l'attributo CodiceFiscalePersona e la relazione PERSONA
- PERSONA (CodiceFiscale, Cognome, Nome)
- SOCIETÀ (PartitaIva, CodiceFiscaleDelegato) con vincolo di integrità referenziale fra l'attributo CodiceFiscaleDelegato e la relazione PERSONA
- DIPENDENTI (PartitaIvaSocietà, CodiceFiscalePersona) con vincoli di integrità referenziale fra l'attributo PartitaIvaSocietà e la relazione SOCIETÀ e fra l'attributo CodiceFiscalePersona e la relazione PERSONA.

Esercizio 8.22 Tradurre lo schema Entità-Relazione ottenuto nell'Esercizio 7.16 in uno schema nel modello relazionale.

Soluzione

Lo schema relazionale corrispondente allo schema logico soluzione dell'esercizio 7.16:

- LOCALITÀ (Codice, Nome, Aeroporto) con vincolo di integrità referenziale fra l'attributo Aeroporto e la relazione AEROPORTI
- AEROPORTI (Codice, Nome)
- CITTÀDIPARTENZA (Codice, Nome)
- VOLI (CodicePartenza, CodiceAeroporto, Costo) con vincoli di integrità referenziale fra l'attributo CodicePartenza e la relazione CITTÀDIPARTENZA e fra l'attributo CodiceAeroporto e la relazione AEROPORTI
- COLLEGE (Codice, Località, Nome, Esame, Indirizzo, Tel) con vincoli di integrità referenziale fra l'attributo Località e la relazione LOCALITÀ e fra l'attributo Esame e la relazione ESAMI
- ESAMI (Codice, Nome, Costo)
- SOGGIORNI (CodiceCollege, Periodo, Costo, Ore) con vincolo di integrità referenziale fra l'attributo CodiceCollege e la relazione COLLEGE
- TIPIDIRIDUZIONE (Codice, Descrizione)
- RIDUZIONI (CodiceTipo, CodiceCollege, Percentuale) con vincoli di integrità referenziale fra l'attributo CodiceTipo e la relazione RIDUZIONI e fra l'attributo CodiceCollege e la relazione COLLEGE.

Esercizio 8.23 Mostrare un'istanza della base di dati definita come soluzione dell'esercizio 8.22.

Soluzione

La soluzione di questo esercizio è mostrata nella figura seguente.

Soggiorni			
CodiceCollege	Periodo	Costo	Ore
CC001	5/7-19/7/2006	1400	20
CC001	19/7-2/8/2006	1600	20
CC002	4/7-18/7/2006	1200	18
...	...	...	...

Località		
Codice	Nome	Aeroporto
LC001	Stirling	A1
LC002	Cambridge	A2
LC003	Oxford	A2
...	...	...

Aeroporti	
Codice	Nome
A1	Edimburgo
A2	Heathrow
...	...

CittàDiPartenza	
Codice	Nome
CP1	Roma
CP2	Milano
CP3	Palermo
...	...

Voli		
CittàPartenza	CodiceAeroporto	Costo
CP1	A2	450
CP1	A1	600
CP3	A2	550
CP3	A1	700
...	...	...

College					
Codice	Località	Nome	Esame	Indirizzo	Tel
C1	LC002	King's	E1	101, King's Street	1236667777
C2	LC002	Queen's	E1	1021, Queen's Road	1237665433
C3	LC003	Prince	E2	1021 St.Johns Road	1256765443
...	...	...	...	...	...

Esami		
Codice	Nome	Costo
E1	Pet	30
E2	Trinity	35
...	...	...

TipiRiduzione	
Codice	Descrizione
R1	Gruppi
R2	Seconda Quindicina
...	...

Riduzioni		
CodiceTipo	CodiceCollege	Percentuale
R1	C1	10
R2	C1	15
...	...	...
R1	C2	10
...	...	...

Figura 8.XIV Esempio di istanza per la base di dati relazionale soluzione dell'esercizio 7.16

Esercizio 8.24 Modificare lo schema concettuale ottenuto in risposta all'esercizio 8.11, assumendo che:

- ciascuna competizione si ripeta ogni anno, con lo stesso organizzatore ma diverso numero di Km totali;
- per ogni località di partenza e arrivo interessi memorizzare la provincia;
- ogni squadra abbia una sigla, un nome e un capitano (che è un ciclista).

Soluzione
La soluzione della prima parte di questo esercizio è riportata in figura 8.XV.

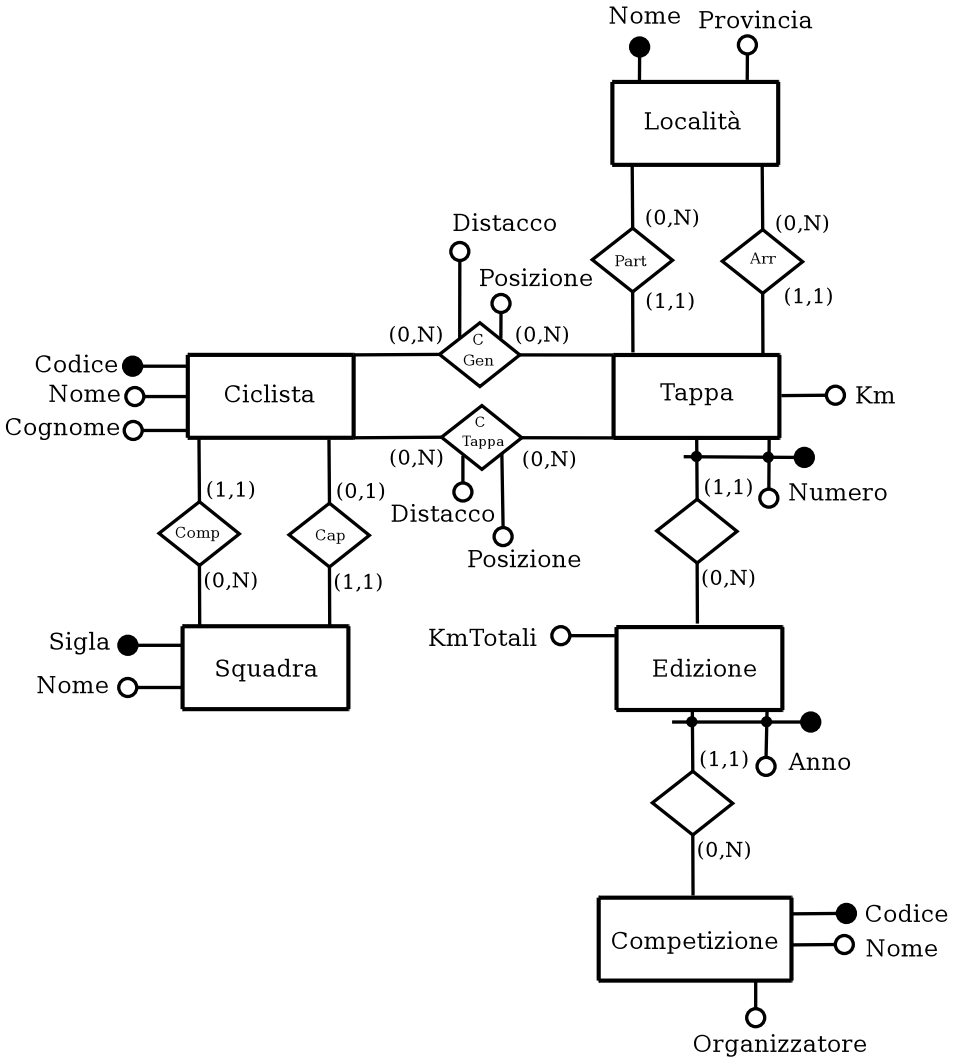


Figura 8.XV Soluzione dell'esercizio 8.24.

Esercizio 8.25 Mostrare lo schema relazionale che si ottiene dallo schema concettuale modificato ottenuto evidenziando le differenze rispetto allo schema relazionale dell'esercizio 8.22.

Soluzione

Lo schema relazionale derivante dallo schema concettuale soluzione dell'esercizio 8.24 e mostrato in figura 8.XV:

- SQUADRE (Sigla,Nome, Capitano) con vincolo di integrità referenziale fra l'attributo Capitano e la relazione CICLISTI
- CICLISTI (Codice, Cognome, Nome, Squadra) con vincolo di integrità referenziale fra l'attributo Squadra e la relazione SQUADRE
- LOCALITÀ (Nome, Provincia)
- COMPETIZIONI (Codice, Nome, Organizzatore)
- EDIZIONI (AnnoEdizione, Competizione, KmTotali) con vincolo di integrità referenziale fra l'attributo Competizione e la relazione COMPETIZIONI
- TAPPE (NumeroTappa. AnnoEdizione, Competizione, LocPartenza, LocArrivo) con vincoli di integrità referenziale fra gli attributi Competizione, AnnoEdizione e la relazione EDIZIONI, fra l'attributo LocPartenza e la relazione LOCALITÀ e fra l'attributo LocArrivo e la relazione LOCALITÀ
- CLASSIFICATAPPA (NumeroTappa, AnnoEdizione, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa, Competizione, AnnoEdizione e la relazione TAPPE e fra Ciclista e la relazione CICLISTA
- CLASSIFICAGENERALE (NumeroTappa, AnnoEdizione, Competizione, Ciclista, Posizione, Distacco) con vincoli di integrità referenziale fra gli attributi NumeroTappa, Competizione, AnnoEdizione e la relazione TAPPE e fra Ciclista e la relazione CICLISTA.

Capitolo 9

Esercizio 9.1

Considerare lo relazione in figura 9.20 e individuare le proprietà della corrispondente applicazione. Individuare inoltre eventuali ridondanze e anomalie nella relazione.

Docente	Dipartimento	Facoltà	Preside	Corso
Verdi	Matematica	Ingegneria	Neri	Analisi
Verdi	Matematica	Ingegneria	Neri	Geometria
Rossi	Fisica	Ingegneria	Neri	Analisi
Rossi	Fisica	Scienze	Bruni	Analisi
Bruni	Fisica	Scienze	Bruni	Fisica

Figura 9.20 Relazione per l'esercizio 9.1

Soluzione:

Una chiave per questa relazione è **Dipartimento, Facoltà, Corso**; una dipendenza funzionale che non riguarda la chiave **Facoltà** $\rightarrow$ **Preside**: questa dipendenza funzionale introduce una ridondanza nella relazione, perché per ogni corso nella stessa Facoltà, il Preside deve essere ripetuto.
La relazione ha un'anomalia di aggiornamento, perché se cambiamo il preside di una facoltà dobbiamo aggiornare tutte le tuple che contengono questa informazione, e non solamente una tupla.
La relazione contiene anche un'anomalia di cancellazione, perché se cancelliamo il preside di una facoltà, perdiamo anche tutte le informazioni sui docenti di quel dipartimento.

Esercizio 9.2

Individuare la chiave e le dipendenze funzionali della relazione considerata nell'Esercizio 9.1 e individuare poi una decomposizione in forma normale di Boyce e Codd.

Soluzione:

Una chiave per questa relazione è **Dipartimento, Facoltà, Corso.**
Anche gli attributi **Docente, Facoltà, Corso** sembrano formare una chiave in questa relazione, ma generalmente parlando questo non è corretto perché lo stesso docente può insegnare lo stesso corso in differenti dipartimenti di una Facoltà.

Decomposizione:

Docente	Dipartimento	Facoltà	Corso
Verdi	Matematica	Ingegneria	Analisi
Verdi	Matematica	Ingegneria	Geometria
Rossi	Fisica	Ingegneria	Analisi
Rossi	Fisica	Scienze	Analisi
Bruni	Fisica	Scienze	Fisica

Facoltà	Preside
Ingegneria	Neri
Scienze	Bruni

Questa decomposizione è corretta perché, con un join tra le due relazioni, otteniamo tutte le tuple della relazione originaria.
Inoltre, la decomposizione risolve il problema delle anomalie, perché è in forma normale di Boyce-Codd.

Esercizio 9.3

Si consideri la relazione riportata in figura 9.20 che rappresenta alcune informazioni sui prodotti di una falegnameria e i relativi componenti. Vengono indicati: il tipo del componente di un prodotto (attributo **Tipo**), la quantità del componente necessaria per un certo prodotto (attributo **Q**), il prezzo unitario del componente di un certo prodotto (attributo **PC**), il fornitore del componente (attributo **Fornitore**) e il prezzo totale del singolo prodotto (attributo **PT**). Individuare le dipendenze funzionali e la chiave di questa relazione.

Prodotto	Componente	Tipo	Q	PC	Fornitore	PT
Libreria	Legno	Noce	50	10.000	Forrest	400.000
Libreria	Bulloni	B212	200	100	Bolt	400.000
Libreria	Vetro	Cristal	3	5.000	Clean	400.000
Scaffale	Legno	Mogano	5	15.000	Forrest	300.000
Scaffale	Bulloni	B212	250	100	Bolt	300.000
Scaffale	Bulloni	B412	150	300	Bolt	300.000
Scrivania	Legno	Noce	10	8.000	Wood	250.000
Scrivania	Maniglie	H621	10	20.000	Bolt	250.000
Tavolo	Legno	Noce	4	10.000	Forrest	200.000

Figura 9.21 Una relazione contenente dati di una falegnameria

Soluzione:

Supponendo che un Tipo si riferisca solamente ad un componente, una chiave per la relazione è **Prodotto, Tipo;** così tutti gli attributi che contengono **Prodotto, Tipo** sono superchiavi per la relazione.
Gli attributi **Q** e **PC** sembrano un'altra chiave, ma potrebbe non essere vero in tutte le istanze di questo database.
Un'altra chiave apparente è **Tipo, PT**

Le dipendenze funzionali sono:

- **Prodotto → PT**
- **Prodotto, Tipo → PC, Q, Fornitore**
- **Tipo → Componente**

Esercizio 9.4

Con riferimento alla relazione in Figura 9.21 si considerino le seguenti operazioni di aggiornamento:

- Inserimento di un nuovo prodotto;
- Cancellazione di un prodotto;
- Aggiunta di una componente a un prodotto;
- Modifica del prezzo di un prodotto.

Discutere i tipi di anomalia che possono essere causati da tali operazioni.

Soluzione:

1) L'inserimento di un nuovo prodotto richiede l'aggiunta di una tupla per ogni tipo di componente. Il prezzo del componente, che è in funzione del prodotto, deve essere ripetuto in ogni tupla.. Anche il prezzo di un componente può essere ridondante perché lo stesso tipo di componente, con lo stesso fornitore è usato per altri prodotti, il prezzo del componente è già presente nella relazione. Questa è un'anomalia di inserimento.
2) La cancellazione di un prodotto implica che tutte le tuple che si riferiscono al prodotto devono essere cancellate; così se un prodotto ha più di un componente, la cancellazione di un prodotto implica la cancellazione di molte tuple; inoltre questa operazione cancella informazioni sui fornitori di componenti: se non ci sono altre tuple che si riferiscono a quei fornitori, le informazioni su di loro andranno perse. Questa è un'anomalia di cancellazione.
3) L'aggiunta di un nuovo componente implica l'aggiunta di una nuova tupla nella relazione. Questa è un'altra anomalia di aggiornamento perché, come per il punto 1, il prezzo totale e (eventualmente) il prezzo del componente devono essere ripetuti.
4) La modifica del prezzo di un prodotto produce un'anomalia di aggiornamento, perché l'aggiornamento di un attributo implica l'aggiornamento di più tuple nella relazione (una tupla per ogni tipo di componente dello stesso prodotto).

Esercizio 9.5

Si consideri sempre la relazione in figura 9.21. Descrivere le ridondanze presenti e individuare una decomposizione della relazione che non presenti tali ridondanze. Fornire infine l'istanza dello schema così ottenuto, corrispondente all'istanza originale. Verificare poi che sia possibile ricostruire l'istanza originale a partire da tale istanza.

Soluzione:

Le ridondanze presenti nella relazione sono riferite alle dipendenze funzionali. Gli attributi ridondanti sono:

- **PT**: che è ripetuto in ogni tupla che si riferisce allo stesso prodotto.
- **PC**: che è ripetuto in ogni tupla che ha lo stesso valore in **Tipo** e **Fornitore**.
- **Componente**: che è ripetuto in ogni tupla che ha lo stesso **Tipo**.

Una possibile decomposizione è:

R1

Prodotto	Tipo	Q	Fornitore
Libreria	Noce	50	Forrest
Libreria	B212	200	Bolt
Libreria	Cristal	3	Clean
Scaffale	Mogano	5	Forrest
Scaffale	B212	250	Bolt
Scaffale	B412	150	Bolt
Scrivania	Noce	10	Wood
Scrivania	H621	10	Bolt
Tavolo	Noce	4	Forrest

R2

Prodotto	PT
Libreria	400.000
Scaffale	300.000
Scrivania	250.000
Tavolo	200.000

R3

Tipo	Componente
Noce	Legno
B212	Bulloni
B412	Bulloni
Cristal	Vetro
Mogano	Legno
H621	Maniglie

R4

Fornitore	Tipo	PC
Forrest	Noce	10.000
Bolt	B212	100
Clean	Cristal	5.000
Forrest	Mogano	15.000
Bolt	B412	300
Wood	Noce	8.000
Bolt	H621	20.000

La relazione R1 ha la chiave originaria della relazione, ma non contiene ridondanze. Le relazioni R2, R3 e R4 hanno le chiavi sul lato sinistro delle dipendenze funzionali (vedi Esercizio 9.3).
Facendo il join su queste chiavi si possono ricostruire esattamente le informazioni dello schema originario.
Tutte le dipendenze sono preservate nella decomposizione, perché ognuna di loro è rappresentata con una relazione differente.

Esercizio 9.6

Decomporre la relazione in Figura 9.21 con l'algoritmo di sintesi di schemi in terza forma nomale illustrato nel Paragrafo 9.6.3

Soluzione

L'insieme di dipendenze mostrato nella risposta all'esercizio 9.3 è una copertura ridotta:

- **Prodotto → PT**
- **Prodotto, Tipo → PC, Q, Fornitore**
- **Tipo → Componente**

L'algoritmo di sintesi genera quindi tre relazioni
- **R1(Prodotto,PT)**
- **R2(Prodotto, Tipo, PC, Q, Fornitore)**
- **R3(Tipo, Componente)**

Poiché R2 contiene la chiave della relazione originaria (Prodotto, Tipo), no serve aggiungere ulteriori relazioni.

Esercizio 9.7 Considerare uno schema di relazione R (E, N, L, C, S, D, M, P, A, con le dipendenze $E \rightarrow NS$, $NL \rightarrow EMD$, $EN \rightarrow LCD$, $C \rightarrow S$, $D \rightarrow M$, $M \rightarrow D$ $EPD \rightarrow AE$ ed infine $NLCP \rightarrow A$. Calcolare una copertura ridotta per tale insieme e decomporre la relazione in terza forma normale.

Soluzione

I passi per calcolare la copertura ridotta di una relazione sono i seguenti:

1. sostituzione l'insieme di dipendenze funzionali con un insieme equivalente che ha i secondi membri costituiti da un singolo attributo;
2. per ogni dipendenza verifica dell'esistenza di attributi eliminabili dal primo membro.
3. eliminazione delle dipendenze ridondanti

Per questo esericizio il primo passo porta all'individuazione delle seguenti dipendenze funzionali:

1. $E \rightarrow N$
2. $E \rightarrow S$
3. $NL \rightarrow E$
4. $NL \rightarrow M$
5. $NL \rightarrow D$
6. $EN \rightarrow L$
7. $EN \rightarrow C$
8. $EN \rightarrow D$
9. $C \rightarrow S$
10. $D \rightarrow M$
11. $M \rightarrow D$
12. $EPD \rightarrow A$
13. $NLCP \rightarrow A$

Il secondo passo porta alle seguenti dipendenze funzionali:

1. $E \rightarrow N$
2. $E \rightarrow S$
3. $NL \rightarrow E$
4. $NL \rightarrow M$
5. $NL \rightarrow D$
6. $E \rightarrow L$
7. $E \rightarrow C$
8. $E \rightarrow D$
9. $C \rightarrow S$
10. $D \rightarrow M$
11. $M \rightarrow D$
12. $EP \rightarrow A$
13. $NLCP \rightarrow A$

Per il terzo passo esaminamo in dettaglio i passaggi con riferimento a due delle dipendenze funzionali.

Vediamo se la dipendenza $E \rightarrow N$ implicata dalle altre calcolando la chiusura iniziando dall'attributo E.
$N \notin X'_F = \{E, S, L, C, D, M\}$
quindi la dipendenza non è implicata dalle altre e non può essere eliminata.

Esaminiamo la dipendenza $E \rightarrow S$:
$S \in X^+{}_F = \{E, S, N, L, C, D, M\}$
quindi la dipendenza è implicata dalle altre e può essere eliminata.

Al termine delle iterazioni si ottiene il seguente risultato:

1. $E \rightarrow N$
2. $NL \rightarrow E$
3. $NL \rightarrow D$
4. $E \rightarrow L$
5. $E \rightarrow C$
6. $C \rightarrow S$
7. $D \rightarrow M$
8. $M \rightarrow D$
9. $NLCP \rightarrow A$

Esercizio 9.8

Si consideri lo schema della relazione in figura 9.22: La chiave di questa relazione è costituita dagli attributi Titolo e Copia, e su di essa è definita la dipendenza **Titolo → Autore Genere**. Verificare se lo schema è o meno in terza forma normale e, in caso negativo, decomporlo opportunamente.

Titolo	Autore	Genere	Copia	Scaffale
Decamerone	Boccaccio	Novelle	1	A75
Divina Commedia	Dante	Poema	1	A90
Divina Commedia	Dante	Poema	2	A90
I Malavoglia	Verga	Romanzo	1	A90
I Malavoglia	Verga	Romanzo	2	A75
I Promessi Sposi	Manzoni	Romanzo	1	B10
Adelchi	Manzoni	Tragedia	1	B20

Figura 9.22 Relazione per l'Esercizio 9.8

Soluzione:

La relazione non è in terza forma normale perché il lato destro della dipendenza funzionale **Titolo → Autore Genere** non è parte della chiave.
Una possibile decomposizione è:

R1

Titolo	Copia	Scaffale
Decamerone	1	A75
Divina Commedia	1	A90
Divina Commedia	2	A90
I Malavoglia	1	A90
I Malavoglia	2	A75
I Promessi Sposi	1	B10
Adelchi	1	B20

R2

Titolo	Autore	Genere
Decamerone	Boccaccio	Novelle
Divina Commedia	Dante	Poema
I Malavoglia	Verga	Romanzo
I Promessi Sposi	Manzoni	Romanzo
Adelchi	Manzoni	Tragedia

La relazione è in forma normale di Boyce-Codd, perché la chiave per R2 è **Titolo**, che è anche il lato sinistro della dipendenza funzionale.

Esercizio 9.9 Si consideri la relazione in figura 9.23 in cui CM e CD sono, rispettivamente, abbreviazioni di CodiceMateria e CodiceDocente e l'attributo CS assume valori di tipo stringa che indicano in qualche modo il corso di studio o i corsi di studio cui un corso è destinato. Individuare la chiave (o le chiavi) e

CM	Materia	CS	Sem.	CD	NomeDoc	Dipartimento
I01	Analisi I	Inf	I	NR1	Neri	Matematica
I01	Analisi I	El	I	NR2	Neri	Matematica
I02	Analisi II	El-Inf	I	NR1	Neri	Matematica
I04	Fisica I	El	II	BN1	Bianchi	Fisica
I04	Fisica I	Mec	I	BR1	Bruni	Meccanica
I04	Fisica I	Inf	I	BR1	Bruni	Meccanica
I05	Fisica II	El	II	BR1	Bruni	Meccanica
I06	Chimica	Tutti	I	RS1	Rossi	Fisica

Figura 9.23 Relazione per l'esercizio 9.9

le dipendenze funzionali definite su di essa (ignorando quelle che si ritiene siano eventualmente "occasionali") e spiegare perché essa non soddisfa la BCNF. Decomporla in BCNF nel modo che si ritiene più opportuno.

Soluzione

Dipendenze:

- CM →Materia
- Materia →CM
- CM CS →Sem. CD
- CD → NomeDoc Dipartimento

Chiavi:

- CM, CS
- Materia, CS

Decomposizione:

- MATERIE (CM, Materia)
- CORSI (CM, CS, Sem, Docente)
- DOCENTI (CD, NomeDoc, Dipartimento)

Esercizio 9.10 Considerare la relazione in figura 9.24, che contiene informazioni relative ai ristoranti di una città, da riportare in una guida turistica.

Cod	Nome	Indirizzo	CT	Tipo	CC	Carta	CZ	Zona
342	Da Piero	V. Larga 32	R	Region.	V	VISA	C	Centro
342	Da Piero	V. Larga 32	R	Region.	A	AmEx	C	Centro
421	Buono	Vic. Corto 1	R	Region.	A	AmEx	C	Centro
425	Paris	V. Lunga 4	I	Internaz.	D	Diners	N	Nord
425	Paris	V. Lunga 4	I	Internaz.	A	AmEx	N	Nord
655	Canton	V. Breve 2	C	Cinese	V	VISA	O	Ovest

Figura 9.24 Relazione per l'esercizio 9.10

Si noti che CT, CC e CZ sono, rispettivamente, abbreviazioni di CodiceTipo, CodiceCarta e CodiceZona.
Individuare la chiave (o le chiavi) della relazione e le dipendenze funzionali definite su di essa (ignorando quelle che si ritiene siano eventualmente "occasionali") e spiegare perché essa non soddisfa la BCNF.
Decomporla in BCNF nel modo che si ritiene più opportuno.
Soluzione
Dipendenze:

- Cod $\rightarrow$ Nome, Indirizzo, Tipo, Zona
- Nome $\rightarrow$ Cod
- CT $\rightarrow$ Tipo
- Tipo $\rightarrow$ CT
- CC $\rightarrow$ Carta
- Carta $\rightarrow$CC
- Zona $\rightarrow$ CZ
- CZ $\rightarrow$ Zona

Chiavi:

- Cod, CC
- Cod, Carta
- Nome, CC
- Nome, Carta

Decomposizione:

- RISTORANTI (Cod, Nome, Indirizzo, CT, Zona)
- TIPICUCINA (CT, Tipo)
- CARTEDICREDITO (CC, Carta)
- ZONE (CZ, Zona)
- CONVENZIONE (Cod, CC)

Esercizio 9.11

Si consideri lo schema Entità-Relazione in figura 9.25. Sui dati descritti da questo schema valgono le seguenti proprietà:

- Un giocatore può giocare per una sola squadra (o per nessuna);
- Un allenatore può allenare una sola squadra (o nessuna);
- Una squadra ha un solo allenatore, diversi giocatori e appartiene a un'unica città.

Verificare se lo schema soddisfa la forma normale di Boyce-Codd e, in caso negativo, ristrutturarlo in un nuovo schema in maniera che soddisfi tale forma normale.

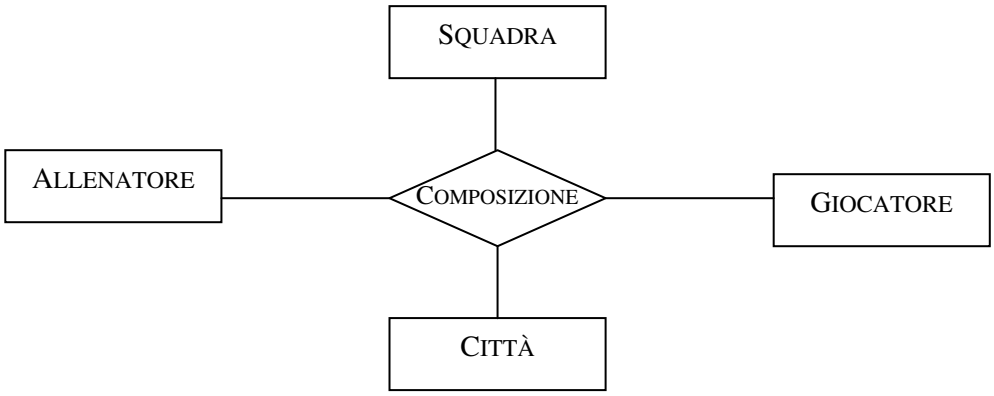


Figura 9.25 Uno schema da sottoporre alla verifica di normalizzazione

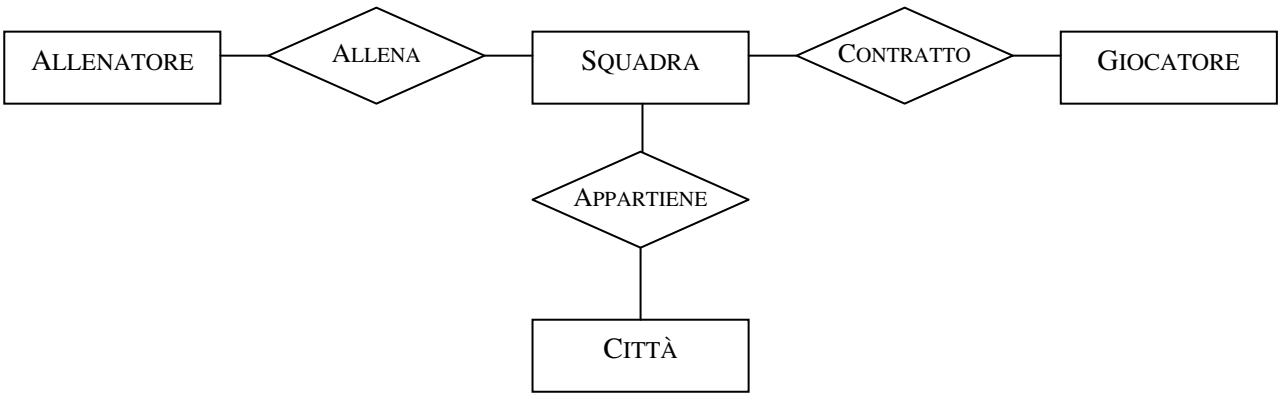
Soluzione:

Le dipendenze funzionali presenti nello schema sono:

- **Giocatore → Squadra**.
- **Allenatore → Squadra**
- **Squadra → Città**

La chiave per la relazione Composizione è Giocatore e così lo schema non è in forma normale di Boyce-Codd.

Una possibile ristrutturazione è:



In questo nuovo schema ci sono solamente relazioni binarie, e quindi rispetta la forma normale di Boyce-Codd.

Esercizio 9.12

Consideriamo la relazione in figura 9.26 e le sue seguenti possibile decomposizioni:

- **Reparto, Cognome** in una relazione e **Cognome, Nome, Indirizzo** nell'altra;
- **Reparto, Cognome, Nome** in una relazione e **Nome, Indirizzo** nell'altra;
- **Reparto, Cognome, Nome** in una relazione e **Cognome, Nome, Indirizzo** nell'altra;

Individuare, con riferimento sia all'istanza specifica sia all'insieme delle istanze sullo stesso schema (con le proprietà naturalmente associate), quali di tali decomposizioni sono senza perdita.

Reparto	Cognome	Nome	Indirizzo
Vendite	Rossi	Mario	Via Po 20
Acquisti	Rossi	Mario	Via Po 20
Bilancio	Neri	Luca	Via Taro 12
Personale	Rossi	Luigi	Via Taro 12

Figura 9.26 Relazione per l'Esercizio 9.12

Soluzione:

La chiave di questa relazione è **Reparto**. Assumiamo che le persone siano identificate dal Cognome e dal Nome.
La relazione ha una dipendenza funzionale: **Cognome, Nome → Indirizzo**

1) Questa soluzione non è corretta in generale; il join tra le due relazioni produce informazioni spurie. Infatti l'attributo **Cognome** non identifica una persona, e il join associerà a un Reparto tutte le persone con lo stesso cognome. Con questa istanza otterremo:

Reparto	Cognome
Vendite	Rossi
Acquisti	Rossi
Bilancio	Neri
Personale	Rossi

Cognome	Nome	Indirizzo
Rossi	Mario	Via Po 20
Neri	Luca	Via Taro 12
Rossi	Luigi	Via Taro 12

Reparto	Cognome	Nome	Indirizzo
Vendite	Rossi	Mario	Via Po 20
Vendite	Rossi	Luigi	Via Taro 12
Acquisti	Rossi	Mario	Via Po 20
Acquisti	Rossi	Luigi	Via Taro 12
Bilancio	Neri	Luca	Via Taro 12
Personale	Rossi	Luigi	Via Taro 12
Personale	Rossi	Mario	Via Po 20

2) Questa decomposizione è corretta in questa particolare istanza del database, perché non ci sono due persone con lo stesso nome e così il join tra le due relazioni da ancora la relazione originaria, ma generalmente parlando il **Nome** non identifica una persona e il loro join può dare una relazione con delle informazioni spurie.

3) Questa decomposizione è sempre corretta perché entrambi gli attributi **Cognome** e **Nome** sono presenti nelle relazioni, e la seconda relazione ha **Cognome** e **Nome** come chiave.
Questa decomposizione produce un database in forma normale di Boyce-Codd.

Reparto	Cognome	Nome
Vendite	Rossi	Mario
Acquisti	Rossi	Mario
Bilancio	Neri	Luca
Personale	Rossi	Luigi

Cognome	Nome	Indirizzo
Rossi	Mario	Via Po 20
Neri	Luca	Via taro 12
Rossi	Luigi	Via taro 12

Reparto	Cognome	Nome	Indirizzo
Vendite	Rossi	Mario	Via Po 20
Acquisti	Rossi	Mario	Via Po 20
Bilancio	Neri	Luca	Via taro 12
Personale	Rossi	Luigi	Via taro 12

Esercizio 9.13

Consideriamo nuovamente la relazione in figura 9.26. Individuare quali delle seguenti decomposizioni conservano le sue dipendenze:

- Una relazione sugli attributi **Reparto, Cognome** e **Nome** e l'altra sugli attributi **Cognome** e **Indirizzo**.
- Una relazione su **Reparto, Cognome** e **Nome** e l'altra su **Cognome, Nome** e **Indirizzo.**
- Una relazione su **Reparto** e **Indirizzo** e l'altra su **Reparto, Cognome** e **Nome**.

Soluzione:

1) Questa decomposizione non conserva la dipendenza **Cognome, Nome** $\rightarrow$ **Indirizzo,** perché gli attributi coinvolti sono suddivisi tra le due relazioni. Così, se abbiamo bisogno, per esempio, di cambiare l'indirizzo riferito al reparto "Vendite", possiamo fare questa operazione soltanto cambiando tutte le tuple della seconda relazione che hanno come Cognome "Rossi"; ma questo non è corretto perché ci sono due persone con lo stesso cognome e solo una è riferita al reparto "Vendite". Anche questa decomposizione non è corretta.

2) Questa decomposizione è corretta, perché la seconda relazione contiene tutti gli attributi della dipendenza funzionale.

3) Questa decomposizione è sbagliata, perché come nel punto 1, gli attributi della dipendenza funzionale sono divisi in due relazioni; in questo caso è possibile aggiungere tuple nella prima relazione, associando ai reparti un indirizzo che non si riferisce alla persona corretta (mentre è impossibile nella relazione originaria).

Capitolo 10

Esercizio 10.1

Realizzare una procedura in un linguaggio di programmazione di alto livello che tramite SQL Embedded elimina dalla tabella DIPARTIMENTO l'elemento che ha il nome che viene fornito come parametro alla procedura.

Soluzione:

Soluzione in C:

```
#include<stdlib.h>
main()
{
char Nome1;
char Nome2;
begin declare section;
         char Nome;
exec sql end declare section;

exec sql declare DipCursore for
         select Nome
         from Dipartimento;
exec sql open DipCursore;

do {
exec sql fetch DipCursore into
         :Nome1;
if (Nome1 == Nome2)
         exec sql delete from Dipartimento
         where nome = :Nome1;
}while (sqlca.sqlcode==0)
}
```

Esercizio 10.2

Realizzare un programma in un linguaggio di programmazione di alto livello che tramite SQL Embedded costruisce una videata in cui si presentano le caratteristiche di ogni dipartimento seguito dall'elenco degli impiegati che lavorano nel dipartimento, ordinati per cognome

Soluzione:

```
#include<stdlib.h>
main()
{

exec sql begin declare section;
        char Nome[20], Città[20], CognomeImpiegato[20],
                NomeImpiegato[20];
        int NumeroDip;
exec sql end declare section;

exec sql declare DipCursore cursor for
        select Nome, Citta, NumDip
        from DIPARTIMENTO;

exec sql declare ImpCursore cursor for
        select CognomeImpiegato, NomeImpiegato
        from Impiegato join Dipartimento on
        Impiegato.Dipartimento=Dipartimento.:Nome
        order by CognomeImpiegato;
exec sql open DipCursore;
do
{
exec sql fetch DipCursore into
        :Nome, :Città, :NumeroDip;
printf("Dipartimento: ", Nome, " situato in: ",Città,
        " Numero Dipendenti: ",NumeroDip." "Dipendenti:");
        exec sql open ImpCursore;
        do
        {
        exec sql fetch ImpCursore into
                :CognomeImpiegato, :NomeImpiegato;
        printf( CognomeImpiegato," ", NomeImpiegato);
        }while(sqlca.sqlcode==0);
        exec sql close cursor ImpCursore;
}while(sqlca.sqlcode==0);
exec sql close cursor DipCursore;
}
```

Esercizio 10.3

Realizzare l'esercizio precedente usando ADO

Soluzione:

```
#include<stdlib.h>
main()
{

conn ADODB.Connection;
impiegati ADODB.Recordset;
dipartimenti ADODB.Recordset;
comandoSQL ADODB.Command;
buffer string;

conn = new ADODB.connection;
conn.open("Server","Utente","Password");

dipartimenti = New ADODB.Recordset;
buffer="select Nome, Citta, NumDip from DIPARTIMENTO";
comandoSQL.CommandText = buffer;
dipartimenti.Open ComandoSQL(conn);

do{
        printf("Dipartimento: ", dipartimenti!Nome, " situato in:
        ",dipartimenti.Città," Numero Dipendenti:
        ",dipartimenti.NumeroDip." "Dipendenti:");

do
{
        impiegati = New ADODB.Recordset;
        buffer = "select CognomeImpiegato, NomeImpiegato
                 from Impiegato join Dipartimento on
                 Impiegato.Dipartimento=Dipartimento.",
                 dipartimenti!Nome,
                 "order by CognomeImpiegato";
        comandoSQL.CommandText = buffer;
        impiegati.Open ComandoSQL(conn);

        printf( impiegati.CognomeImpiegato," ",
                 impiegati.NomeImpiegato);
        impiegati.movenext;
        }while(impiegati.EOF);
dipartimenti.movenext;
}while(dipartimenti.EOF);
}
```

Esercizio 10.4

Realizzare un programma java che scandisce gli impiegati ordinati per cognome e inserisce ogni impiegato che si trova in una posizione che è un multiplo di 10 in una tabella IMPIEGATIESTRATTI

Soluzione:

```
import java.sql.*;
public class ImpiegatiEstratti {
public static void main(String[] arg){

connection conn = null;
try{
        // carica driver e inizializza connessione
        Class.forName("sun.jdbc.odbc.jbdcObdcDriver");
        conn = DriverManager.GettConnection("jdbc:odbc:impiegati)
        }
try{
        Statement interrogazione = conn.createStatement();
        ResultSet risultato = interrogazione.executeQuery(
                "select *
                 from IMPIEGATI"




        Statement interrogazione1 = conn.createStatement();
        ResultSet risultato1 = interrogazione1.executeQuery(
                "create table IMPIEGATIESTRATTI
                        (
                        NomeImpiegato char(20),
                        CognomeImpiegato char(20),
                        dipartimento char(20),
                        stipedio integer,
                        primary key(NomeImpiegato,CognomeImpiegato)
                        )
        i=0;
        While(risultato.next()) {

        if(i=10)
        {
        string NomeImpiegato = risultato.getString("NomeImpiegato");
        string CognomeImpiegato =
        risultato.getString("CognomeImpiegato");
        string Dipartimento= risultato.getString("Dipartimento");
        int Stipendio = risultato.getString("Stipendio");
```

```
        Statement interrogazione2 = conn.createStatement();
        ResultSet risultato2 = interrogazione2.executeQuery(
                "insert into IMPIEGATIESTRATTI
                        values(" NomeImpiegato","
                        CognomeImpiegato","
                        dipartimento","
                        stipedio",
                        )";
        i=0;
}}}}
```

Esercizio 10.5

Realizzare un programma che accede al contenuto di una tabella

```
Capitolo(Numero, Titolo, Lunghezza)
```

che descrive i capitoli di un libro, con il numero e la dimensione delle pagine. Il programma quindi popola una tabella

```
Indice(Numero, Titolo, NumPagine)
```

in cui si presenta il numero di pagina nel quale inizia il capitolo, supponendo che il capitolo 1 inizia sulla prima pagina e che i capitoli devono iniziare su pagine dispari (eventualmente introducendo una pagina bianca alla fine del capitolo)

Soluzione:

```
#include<stdlib.h>
main()
{
exec sql begin declare section;
        char Titolo[50];
        int Numero, Lunghezza;
exec sql end declare section;

exec sql declare CapCursore cursor for
        select Numero, Titolo, Lunghezza
        from Capitolo;
exec sql open CapCursore;
exec sql create table Indice
        (
        Numero integer primary key,
        Titolo char(50),
        NumPagine integer
        );
do{
exec sql fetch CapCursore into
        :Numero, :Titolo, :Lunghezza;
if (Lunghezza%2 != 0)
        Lunghezza = Lunghezza + 1; // Pagina Bianca
exec sql insert into Indice
        values(:Numero,:Titolo,:Lunghezza)";
}while(sqlca.sqlcode == 0)

exec sql close cursor CapCursore;
}
```

Esercizio 10.6 Con riferimento al seguente schema relazionale:

- IMPIEGATI (CodiceFiscale, Cognome, Nome, DataNascita, Dipartimento, Stipendio) con vincolo di integrità referenziale fra l'attributo Dipartimento e la relazione DIPARTIMENTI
- DIPARTIMENTI (Codice, Nome, Sede)
- PROGETTI (Sigla, Titolo, Valore)
- PARTECIPAZIONE (Impiegato, Progetto, Data) con vincoli di integrità referenziale fra l'attributo Progetto e la relazione PROGETTI e fra l'attributo Impiegato e la relazione IMPIEGATI

scrivere un metodo Java con JDBC (o un frammento di programma in SQL immerso in un linguaggio o pseudolinguaggio di programmazione) che inserisca un impiegato con tutti i dati (letti da input o passati come parametri), verificando l'esistenza del dipartimento, con rifiuto dell'operazione in caso negativo. Assumere, per semplicità, che il sistema non supporti i vincoli di riferimento.
Soluzione

```
static void inserimento(Connection con, String cf,
            String cognome, String nome,
            String dataNascita, String dipartimento,
            int stipendio){
    try {
      // Verifica codice dipartimento
      PreparedStatement pquery =
          con.prepareStatement(
                      "select * " +
                      "from Dipartimenti " +
                      "where Codice = ?");
      pquery.setString(1,dipartimento);
      ResultSet result = pquery.executeQuery();
      if (result.next()){
          pquery.close();
          // Il dipartimento esiste
          PreparedStatement pupdate =
            con.prepareStatement(
              "insert into Impiegati" +
              "(CodiceFiscale,Cognome,Nome," +
              "DataNascita,Dipartimento,Stipendio)" +
              "values (?,?,?,?,?,?)");
          pupdate.setString(1,cf);
          pupdate.setString(2,cognome);
          pupdate.setString(3,nome);
          pupdate.setString(4,dataNascita);
          pupdate.setString(5,dipartimento);
          pupdate.setInt(6,stipendio);
          pupdate.executeUpdate();
```

```
            pupdate.close();
        }
        else {
            pquery.close();
            System.out.println(
                      "Non esiste dipartimento " +
                       dipartimento);
        }
    }
    catch (SQLException e){
        System.out.println("Errore");
        System.out.println(e.getErrorCode() + " " +
                 e.getSQLState() +  e.getMessage() );
    }
}
```

Esercizio 10.7 Dato lo schema relazionale seguente:

- IMPIEGATI (Codice, Dati, Telefono) con vincolo di integrità referenziale fra l'attributo Dati e la relazione DATIIMPIEGATO
- DATIIMPIEGATI (CodiceDati, Cognome, Nome)

definire il metodo Java (o un frammento di programma in SQL immerso in un linguaggio o pseudolinguaggio di programmazione) getImpiegatoPerNome(String cognome, String nome) che, dato il nome di un impiegato, restituisce un Oggetto Impiegato in cui i campi (Codice, Nome, Cognome, Telefono) sono opportunamente valorizzati.

Soluzione

Si presuppone che nell'azienda un impiegato sia univocamente identificato dal suo nome e dal suo cognome.

```
public Impiegato getImpiegato(String cognome,
 String nome) throws SQLException {

    Impiegato i = null;
    PreparedStatement s = this.con.prepareStatement(
          "setect * " +
          "from Impiegati, DatiImpiegati" +
          "where Dati = CodiceDati" +
          "and cognome = ?" +
          "and nome = ?");
    s.setString(1, cognome);
    s.setString(2, nome);
    ResultSet rs = s.executeQuery();
    if (rs.hasNext()){
        i = new Impiegato (rs.getInt("Codice"),
                           rs.getString("Nome"),
                           rs.getString("Cognome"),
                           rs.getString("Telefono"));
    }
    s.close();
    return i;
}
```

Esercizio 10.8 Si consideri il seguente schema relazionale (con gli evidenti vincoli di integrità referenziale):

- VENDITE (CodArticolo, CodNegozio, CFCliente, Data, Quantità) con vincoli di integrità referenziale fra l'attributo CodArticolo e la relazione ARTICOLI, fra l'attributo CodNegozio e la relazione NEGOZI, fra l'attributo CFCliente e la relazione CLIENTI
- ARTICOLI (CodArticolo, Descrizione, CodMarca, CodCategoria, Prezzo) con vincoli di integrità referenziale fra l'attributo CodMarca e la relazione MARCHE e fra l'attributo CodCategoria e la relazione CATEGORIE
- CLIENTI (CFCliente, Cognome, Nome, Età)
- NEGOZI (CodNegozio, Nome, Indirizzo, Città, Provincia, Regione)
- MAECHE (CodMarca, Nome, CodNazione, Nazione)
- CATEGORIE (CodCategoria, Descrizione).

Scrivere un metodo Java con JDBC che inserisca un nuovo negozio con codice, nome, indirizzo e città (letti da input o passati come parametri), prelevando provincia e regione da altre della stessa tabella (nell'ipotesi che, fissata la città, provincia e regione siano univocamente determinate) e segnalando come errore (o eccezione) il caso in cui i dati sulla città non siano disponibili.
Soluzione

```
public void insertNegozio(String codice, String nome,
 String indirizzo, String citta)
 throws SQLException {

  /* query che preleva regione e provincia
     a cui appartiene una data citta */
  PreparedStatement getProvReg = con.prepareStatement(
      "select Provincia, Regione" +
      "from Negozi" +
      "where citta = ?");
  getProvReg.setString(citta);
  ResultSet rs = getProvReg.executeQuery();
  if (rs.size() == 0)
      throw new NoDataFoundException();
  String prov = rs.getString("Provincia");
  String reg = rs.getString("Regione");
  getProvReg.close();

   /* query di inserimento */
  PreparedStatement ins = con.PrepareStatement(
      "insert into negozi values (?,?,?,?,?,?);");

  ins.setString(1, codice);
  ins.setString(2, nome);
```

```
  ins.setString(3, indirizzo);
  ins.setString(5, prov);
  ins.setString(6, reg);

  ins.executeUpdate();
  ins.close();
}
```

Esercizio 10.9 Con riferimento allo relazionale seguente:

- FARMACI (Codice, NomeFarmaco, PrincipioAttivo, Produttore, Prezzo) con vincolo di integrità referenziale fra Produttore e la relazione PRODUTTORI e fra PrincipioAttivo e la relazione SOSTANZE
- PRODUTTORI (CodProduttore, Nome, Nazione)
- SOSTANZE (ID, NomeSostanza, Categoria)

scrivere un metodo Java con JDBC che (supponendo già disponibile una connessione, passata come parametro) stampi un prospetto con tutti i farmaci, organizzati per produttore:

CodProduttore Nome Nazione
CodiceFarmaco NomeFarmaco Prezzo Sostanza
CodiceFarmaco NomeFarmaco Prezzo Sostanza
...
CodProduttore Nome Nazione
...

Soluzione

```
public void prontProspetto(Connection con)
 throws SQLException{

    /* query di estrazione dati produttore */
    String q1 = "select CodProduttore" +
                " Nome, Nazione" +
                "from Produttori";

    /* query di estrazione dati per i farmaci */
    String q2 = "select CodiceFarmaco," +
                " NomeFarmaco, Prezzo, Sostanza" +
                "from Farmaci join Produttore" +
                "    on PrincipioAttivo = ID" +
                "where CodProduttore = ?";

    PreparedStatement p = con.prepareStatement(q1);
    ResultSet rs, rs1;
    PreparedStatement p2 = con.prepareStatement(q2);
    rs.con.executeQuery();

    while(rs.hasNext()) {
        p1.setString(rs.getString("CodProduttore");
        rs1=p1.executeQuery();
        System.out.println(
               rs.getString("CodProduttore") +
               " " + rs.getString("Nome") +
               " " + rs.getString("Nazione"));
        while(rs1.hasNext()) {
```

```
            System.out.Println(
               rs1.getString("CodiceFarmaco") +
               " " + rs1.getString("NomeFarmaco") +
               " " + rs1.getString("Prezzo") +
               " " + rs1.getString("Sostanza"));
        }
    }
    p.close();
    p2.close()
}
```

Esercizio 10.10 Si consideri una base di dati che contiene informazioni sugli impiegati, i progetti e le sedi di una azienda, con le partecipazioni degli impiegati ai progetti e le sedi di svolgimento dei progetti stessi; essa contiene le seguenti relazioni:

- IMPIEGATI (Matricola,Cognome,Nome,Progetto), con vincolo di integrità referenziale fra Progetto e la relazione PROGETTI
- PROGETTI (Codice,Titolo)
- SEDI (Nome,Città,Indirizzo)
- SVOLGIMENTO (Progetto,Sede), con vincoli di integrità referenziale fra Progetto e la relazione PROGETTI fra Sede e la relazione SEDI.

Formulare in Java-JDBC (o con SQL immerso in un linguaggio o pseudolinguaggio di programmazione), una classe (o un frammento di programma) che stampa tutti i progetti (con codice e titolo) e, per ciascuno di essi, gli impiegati coinvolti (mostrando matricola e cognome); in sostanza, va prodotto un prospetto del tipo seguente:

CodProgetto TitoloProgetto
MatricolaImpiegato CognomeImpiegato
MatricolaImpiegato CognomeImpiegato
...
CodProgetto TitoloProgetto
MatricolaImpiegato CognomeImpiegato
...

Soluzione

```
public void printProspetto(Connection con)
 throws SQLException {

    String codice = null;
    ResultSet rs1, rs2;

    /* query di estrazione dei progetti */
    String q1 = "select Codice, Titolo," +
                "from Progetto";

    /* query di estrazione degli impiegati */
    String q2 = "select Matricola, Cognome" +
                "from Impiegato join Progetto" +
                "    on (Progetto = Codice)" +
                "where codice = ?";

    PreparedStatement s1 = con.prepareStatement(q1);
    PreparedStatement s2 = con.prepareStatement(q2);

    rs1 = s1.executeQuery();
```

```
    while(rs1.hasNext()) {
        codice = rs1.getString("Codice");
        System.out.println( codice +
                 " " + getString("Titolo"));
        s2.setString(codice);
        rs2. = s2.executeQuery();
        while(rs2.hasNext()) {
             System.out.println(
                 rs2.getString("Matricola" +
                 " " + rs2.getString("Cognome"));
        }
    }
    s1.close();
    s2.close();
}
```

Esercizio 10.11 Estendere la risposta al quesito precedente mostrando anche, per ciascun progetto, la lista delle sedi di svolgimento, costruendo quindi un prospetto come il seguente:

CodProgetto TitoloProgetto
  MatricolaImpiegato CognomeImpiegato
  MatricolaImpiegato CognomeImpiegato
  ...
  NomeSede Città
  NomeSede Città
  ...
CodProgetto TitoloProgetto
  ...

Soluzione

```
public void printProspetto(Connection con)
 throws SQLException {

    String codice = null;
    ResultSet rs1, rs2, rs3;

    /* query di estrazione dei progetti */
    String q1 = "select Codice, Titolo," +
                "from Progetto";

    /* query di estrazione degli impiegati */
    String q2 = "select Matricola, Cognome" +
                "from Impiegato join Progetto" +
                "    on (Progetto = Codice)" +
                "where codice = ?";

    /* query di estrazione delle sedi */
    String q3 = "S.Nome as NomeSede," +
                "S.Citta as Citta," +
                "from Progetto P join Svolgimento Sv" +
                "    on (P.Progetto = S.Progetto)" +
                "        join Sedi S " +
                "            on (S.Nome = Sv.Sede)" +
                "where Codice = ?";


    PreparedStatement s1 = con.prepareStatement(q1);
    PreparedStatement s2 = con.prepareStatement(q2);
    PreparedStatement s3 = con.prepareStatement(q3);

    rs1 = s1.executeQuery();
```

```
    while(rs1.hasNext()) {
       codice = rs1.getString("Codice");
       System.out.println( codice +
              " " + getString("Titolo"));
       s2.setString(codice);
       rs2. = s2.executeQuery();
       while(rs2.hasNext()) {
           System.out.println(
              rs2.getString("Matricola" +
              " " + rs2.getString("Cognome"));
       }

       s3.setString(codice);
       rs3 = s3.executeQuery();
       while(rs3.hasNext()) {
           System.out.println(
           rs3.getString("NomeSede") + " " +
                 rs3.getString("Citta"));

       }
    }
    s1.close();
    s2.close();
    s3.close();
}
```

Esercizio 10.12 Si ha uno schema di tabella IMPIEGATO (Nome, Indirizzo, Capo) in cui l'attributo Capo rappresenta il nome del superiore dell'impiegato, descritto a sua volta nella tabella. Definire il metodo Java (o un frammento di programma in SQL immerso in un linguaggio o pseudolinguaggio di programmazione) setCapo(Impiegato i1, Impiegato i2) che memorizza la relazione tra capo e sottoposto esistente tra due impiegati rifiutando l'inserimento se la relazione o la sua inversa già esistono. Supporre che gli oggetti Impiegato abbiano una variabile di istanza per ogni campo della corrispondente colonna nello schema relazionale con opportuna corrispondenza di tipi.

Soluzione

```
public boolean setCapo(Impiegato i, Impiegato capo)
 throws SQLException {

   /* query di verifica */
   String qV = "select * " +
               "from Impiegato" +
               "where nome = ?" +
               "and capo = ?" +
               "or" +
               "nome = ?" +
               "and capo = ?";

   /* query di aggiornamento */
   String upd = "update Impiegati" +
                "   set capo = ?" +
                "   where nome = ?";

   PreparedStatement p1 = con.prepareStatement(qV);

   p1.setString(1, i.getNome());
   p1.setString(2, capo.getNome());
   p1.setString(3, capo.getNome());
   p1.setString(4, 1.getNome());
   ResultSet rs = p1.executeQuery();
   if(rs.size() > 0) {
       p1.close();
       return false;
   }

   PreparedStatement p2 = con.prepareStatement(upd);
   p2.setString(1, capo.getName());
   p2.setString(2, i.getName());
   p2.executeUpdate();

   p1.close();
```

```
    p2.close();
    return true;
}
```

Esercizio 10.13 Con riferimento allo schema relazionale dell'esercizio 10.10, si scriva un programma Java che realizza quanto richiesto usando JPA. Si mostrino le annotazioni alle classi Java necessarie per creare la mappatura relazionale degli oggetti Java. Tale mappatura deve contenere anche le annotazioni necessarie per la gestione automatica della cancellazione delle istanze persistenti della classe SVOLGIMENTO collegate tramite vincoli di integrità referenziale alle istanze delle classi PROGETTO e SEDE. Si realizzino le funzioni di estrazione dei dati mediante interrogazioni JPQL.

Soluzione
Vedere il codice nell'allegato file `Ex10.13.zip`

Esercizio 10.14 Si consideri un'applicazione che gestisce l'anagrafica il personale di un'azienda. Ogni membro del personale caratterizzato da Nome, Cognome, e data di nascita. Gli impiegati dell'azienda sono inoltre cartterizzati da una matricola e un salario. I collaboratori esterni sono caratterizzati da un numero di partita IVA e dal costo orario. La classificazione del personale in impiegati e collaboratori è totale ed esclusiva. Si scriva un programma Java che consenta di gestire oggetti delle tre classi descritte e si mostri la mappatura relazionale degli oggetti secondo i tre schemi: *Single table per class hierarchy*, *Table per concrete class* e *Joined tables*. Si includa nel programma Java un metodo che stampa tutti gli oggetti dell'anagrafica in ordine alfabetico di cognome, precisando oltre agli attributi persistenti anche il tipo di ciascun oggetto.

Soluzione
Vedere il codice nell'allegato file `Ex10.14.zip`

Esercizio 10.15 Effettuare una comparazione sull'efficienza dei due approcci seguenti per ottenere tutti i corsi in cui un determinato professore insegna. Il metalinguaggio utilizzato é una ipotetica codifica in cui si può immergere l'SQL.

1.
```
execute query1() into cursore1;

do
    loop while (not cursore1%empty);
    fetch cursore1 into Professore;
    execute query2(Professore) into cursore2;

    do
      loop while (not %cursore2%empty);
      fetch cursore2 into corso;
      print(Professore.Nome, Professore.Cognome,
            Corso);

    end loop;

end loop;


query1:

select * from Professori;

query2:

select *
from Corsi
where Professore = ?;
```

2.
```
select P.Cognome, P.Nome, Corso
from Professori P, Corsi C
where C.Professore = P.MatricolaProfessore;
```

Soluzione
L'approccio 2 è sicuramente più efficiente. La query è predicibile a priori ed interamente interpretata ed ottimizzata dal dbms, in grado di garantire prestazioni migliori di un qualsiasi compilatore di linguaggio di programmazione. Il programma 1 fornisce implicitamente una implementazione di un join mediante l'annidamento di due cicli. Tale join è sicuramente ottimizzato in maniera più avanzata attraverso la dichiarazione esplicita di voler effettuare una tale operazione a livello database. Inoltre, al punto 2 viene eseguita una sola query, mentre il punto 1 comporta l'esecuzione di una query di indagine per ogni professore.

Capitolo 11

Esercizio 11.1

Definire le strutture dati necessarie per la gestione dei buffer; implementare in un qualunque linguaggio le funzioni `fix`, `use` e `unfix`. Si suppongano disponibili le funzioni di file system descritte nel paragrafo 11.1.3.

Soluzione:

```
typedef struct page {
      int *address;
      int valid;
      int modified;
      int in_use;
      int file_id;
      int block_num;
}

typedef struct file_open {
      int file_id;
      char *file_name;
      int size;
      int blocks[]; // each element refers to an element in page table
}

typedef page_table *page;
typedef file_table *file_open;

page_table pt=new page_table[N];
file_table ft=new file_table[M];


int fix(char *file_name, int block) {
      int i=0;
      int found=0;
      while (i<0 && !found)
        if (strcmp(file_name,ft[i].file_name)!=0) i++; else found=1;

      if (found && ((ft[i].blocks[block])!=-1) &&
           pt[ft[i].blocks[block]].valid )
                      // -1 means that the block is
                      // not loaded
       return (pt[ft[i].blocks[block]]).address;

      if (!found) { // file is open
           int id=open(file_name);
           int position=get_file_position() // find a free position in ft
           ft[position].file_id=id;
           strcpy(file_name, ft[position].file_name);
           ft[position].size=get_size(id);
           for(int k=0; k<ft[position].size; k++)
                ft[position].blocks[k]=-1;
      }

      // block is read

      int free=get_free_page(); // find a free page in pt
```

```
      read(ft[i].file_id, block, free.address);
      pt[free].valid=1;
      pt[free].modified=0;
      pt[free].file_id=ft[i].file_id
      pt[free].block_num=block;
      return pt[free].address;
}

void use (int *page) {
      int i=0;
      int found=0;
      while (i<N && !found)
      if (pt[i].address!=page) i++ else found=1;
            pt[i].in_use=1;
}

void unfix(int *page) {
      int i=0;
      int found=0;
      while (i<N && !found)
            if (pt[i].address!=page) i++ else found=1;
      pt[i].in_use=0;
      if (pt[i].modified) {
            pt[i].valid=0;
            int j=0;
            found=0;
            while (j<N && !found)
                  if (ft[j].file_id!=pt[i].file_id) j++ else found=1;
            ft[j].block[pt[i].block_number]=-1;
      }
}
```

Esercizio 11.2

Si consideri una base di dati gestita tramite hashing, il cui campo chiave contenga i seguenti nominativi:
Green, Lovano, Osby, Peterson, Pullen Scofield, Allen, Haden, Sheep, Harris, MacCann, Mann, Brown, Hutcherson, Newmann, Ponty, Cobbham, Coleman, Mingus, Lloyd, Tyner, Fortune, Coltrane.

1. Proporre un algoritmo di hashing con B=8 e F=4.
2. Supponendo B=40 e F=1, qual è la probabilità di conflitto? E con B=20 e F=2?
3. Con F=5 e B=7, quanto vale approssimativamente la lunghezza media della catena di overflow?

Soluzione:

1) Una semplice funzione di hashing per i nomi dati è:
   - Per ogni carattere del nome, considerare il corrispondente numero in ordine alfabetico (a = 1, b = 2 …)
   - Sommare tutti i numeri ottenuti e fare il "modulo B" della divisione

In questo caso otterremo per ogni nome un numero compreso tra 0 e B-1.

Esempio:

Hash(Green)=(7+18+5+5+14) mod 8=1
Hash(Lovano)=(12+15+22+1+13+15) mod 8=6
Hash(Osby)=(15+19+2+25) mod 8=5
Hash(Peterson)=(16+5++20+5+18+19+15+13) mod 8=7

2) Con B = 40 e F = 1 la probabilità di conflitto è:

$$p = 1 - T\,(1/B)\,(1 - 1/B)^{T-1} = 1 - 23\,(1/40)\,(39/40)^{22} = 0{,}6706$$

Con B = 20 e F = 2

$$p = 1 - \sum_{i=1}^{F} \binom{T}{i} (1/B)^i (1-1/B)^{T-1} = 1 - 23(1/20)(19/20)^{22} - \binom{23}{2}(1/20)^2(19/20)^{21} = 0{,}4311$$

Nota che questa è la probabilità di avere due o più collisioni nello stesso blocco, perché ogni blocco contiene 2 tuple e 1 collisione è ammessa.

3) La lunghezza della catena di overflow può essere calcolata come la somma pesata delle probabilità di collisione.

$$l = \sum_{i=F+1}^{T} i \binom{T}{i} (1/B)^i (1-1/B)^{T-1} = 0{,}647$$

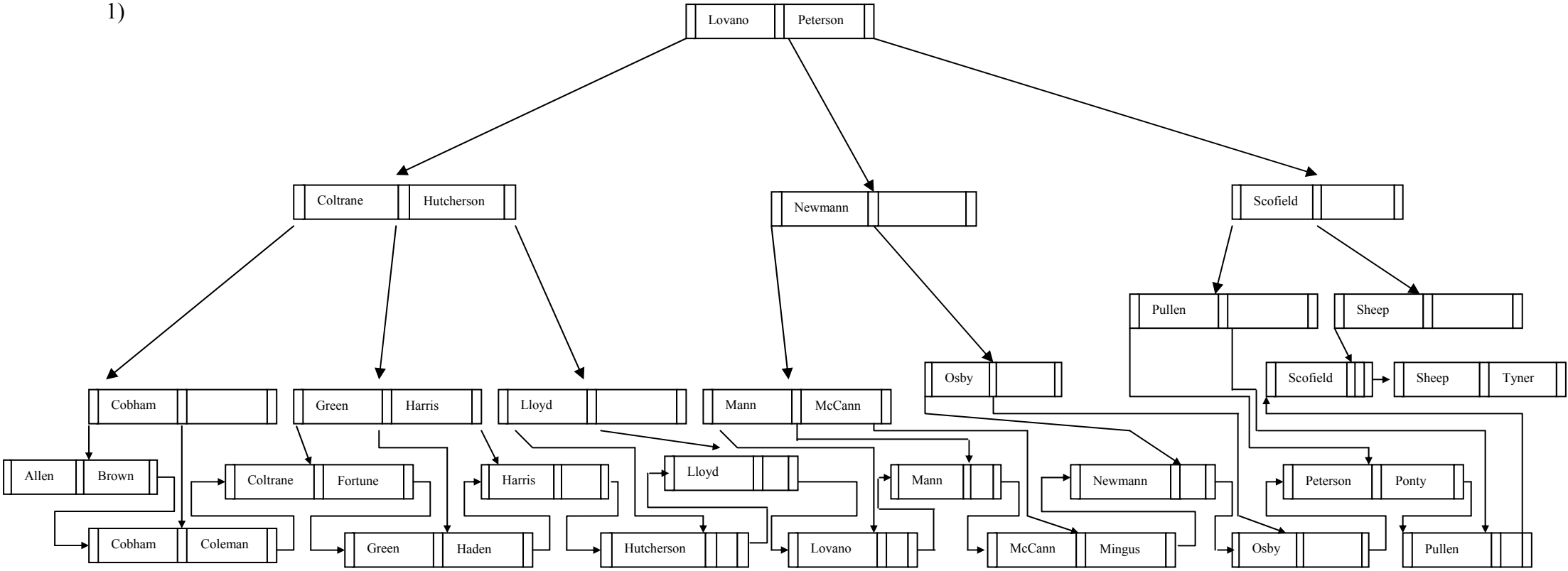
Esercizio 11.3

Si consideri una base di dati gestita tramite alberi B+, il cui campo chiave contenga i dati elencati nel precedente esercizio.

1. Descrivere una struttura ad albero B+ bilanciato, con F=2, che contenga i dati citati
2. Introdurre un dato che provochi lo split di un nodo al livello foglia, e mostrare cosa accade al livello foglia e al livello superiore.
3. Introdurre un dato che provochi il merge di un nodo al livello foglia, e mostrare cosa accade al livello foglia e al livello superiore.
4. Indicare una sequenza di inserimenti che provochi, ricorsivamente, lo split della radice e l'allungamento dell'albero.
5. Descrivere una struttura ad albero B, con F=3, che contenga i dati citati.

Soluzione:

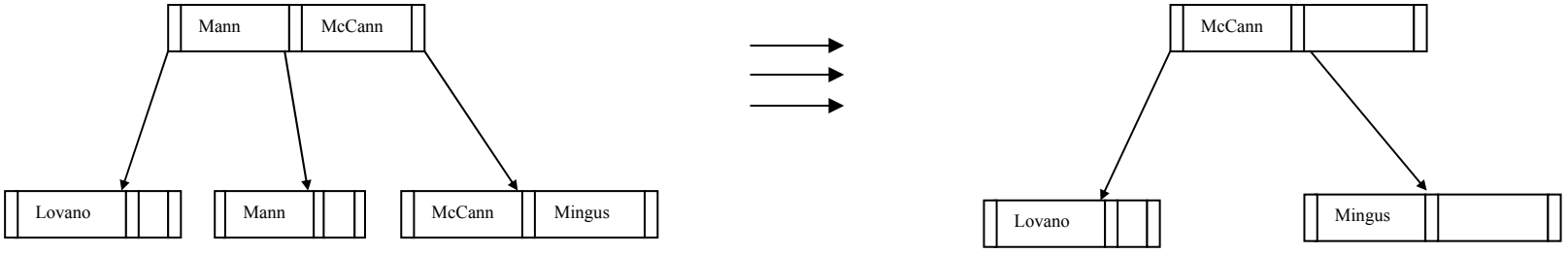
1)



L’albero B+Tree contiene tutti i campi chiave. Ha 16 nodi al livello foglia.

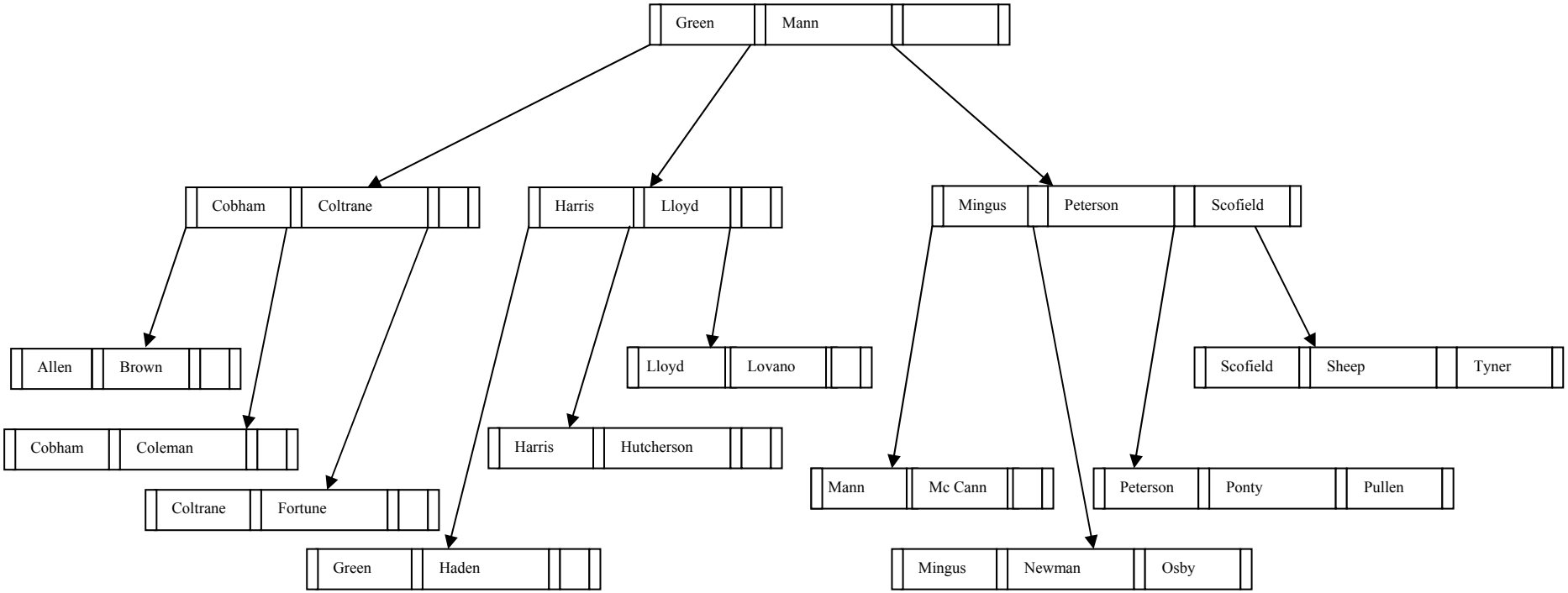
2) L’introduzione del valore “Brooke”, causa uno split a livello foglia, come illustrato nella seguente figura.

3) La cancellazione del valore “Mann” causa un merge al livello foglia.



4) L’inserimento di “Brooke”, “Addams” e “Gibson” causa uno slit alla root.

5)


Green
Mann
Cobham
Coltrane
Harris
Lloyd
Mingus
Peterson
Scofield
Allen
Brown
Lloyd
Lovano
Scofield
Sheep
Tyner
Cobham
Coleman
Harris
Hutcherson
Mann
Mc Cann
Peterson
Ponty
Pullen
Coltrane
Fortune
Green
Haden
Mingus
Newman
Osby

Esercizio 11.4

Si consideri una base di dati costituita dalle seguenti relazioni:

PRODUZIONE(NumeroSerie, TipoParte, Modello, Quan, Macchina)
PRELIEVO(NumeroSerie, Lotto)
ORDINE(Lotto, Cliente, Ammontare)
COMMISSIONE(Lotto. Venditore, Ammontare)

Si assumano inoltre i seguenti profili:

CARD(PRODUZIONE) = 200.000	SIZE(PRODUZIONE)=41
CARD(PRELIEVO)=50.000	SIZE(PRELIEVO)=15
CARD(ORDINE)=10.000	SIZE(ORDINE)=45
CARD(COMMISSIONE)=5.000	SIZE(COMMISSIONE)=35

SIZE(NumeroSerie)=10	VAL(NumeroSerie)=200.000
SIZE(TipoParte)=1	VAL(TipoParte)=4
SIZE(Modello)=10	VAL(Modello)=400
SIZE(Quan)=10	VAL(Quan)=100
SIZE(Macchina)=10	VAL(Macchina)=50
SIZE(Lotto)=5	VAL(Lotto)=10.000
SIZE(Cliente)=30	VAL(Cliente)=400
SIZE(Ammontare)=10	VAL(Ammontare)=5.000
SIZE(Venditore)=20	VAL(Venditore)=25

Descrivere l'ottimizzazione algebrica e il calcolo dei profili dei risultati intermedi relativi alle seguenti interrogazioni, che vanno inizialmente espresse in SQL e poi tradotte in algebra relazionale:

1. Determinare la quantità disponibile del prodotto 77Y6878.
2. Determinare le macchine utilizzate per la produzione dei pezzi venduti al cliente Rossi.
3. Determinare i clienti che hanno comprato dal rivenditore Bianchi un box modello 3478.

Per le ultime interrogazioni, che richiedono join fra tre o quattro tabelle, indicare gli ordinamenti tra join che sembrano più convenienti sulla base delle dimensioni degli operandi. Descrivere poi, prevedendo solo due alternative a scelta per l'esecuzione dei join, l'albero delle alternative relativo alla seconda interrogazione.

Soluzione:

1) SQL:

```
select Quan
from Produzione
where NumeroSerie="77Y6878"
```

Algebra Relazionale:

$$\Pi_{Quan}(\sigma_{NumeroSerie=\text{"77Y6878"}}(Produzione))$$

Questa query non ha bisogno di nessuna ottimizzazione algebrica. Se indichiamo con T la tabella dei risultati, il profilo è:

CARD(T)=1 SIZE(Quan)=10
SIZE(T)=10 VAL(Quan)=1

2) SQL:

```
select Macchina
from Produzione join Prelievo on
Produzione.NumeroSerie=Prelievo.NumeroSerie
join Ordine on Prelievo.Lotto=Ordine.Lotto
where Cliente= "Rossi"
```

Algebra relazionale:

$$\Pi_{Macchina}(\sigma_{Cliente=\text{"Rossi'}}(Produzione \bowtie_{NumeroSerie=p} \rho_{o,p \leftarrow NumeroSerie,\ Lotto}(Prelievo) \bowtie_{o=Lotto} Ordine))$$

Ottimizzazione algebrica: Push delle proiezioni e delle selezioni

$$\Pi_{Macchina}(\Pi_{Pr}(\Pi_{NumeroSerie}(\sigma_{Cliente=\text{"Rossi'}}(Ordine)) \bowtie_{NumeroSerie=Or} \rho_{Or,Pr \leftarrow NumeroSerie,\ Lotto}(Prelievo)) \bowtie_{Pr=Lotto} (\Pi_{Lotto,\ Macchina}(Produzione)))$$

Poniamo: $T_1 = \Pi_{NumeroSerie}(\sigma_{Cliente=\text{"Rossi'}}(Ordine))$

Abbiamo:

$CARD(T_1) = (1 / VAL(Cliente)) * CARD(Ordine) = (1 / 400) * 10.000 = 25$

$SIZE(T_1) = 5$

La scelta più conveniente in ordine di join è $(Ordine \bowtie Prelievo) \bowtie Produzione$

Poniamo: $T_2 = \Pi_{Pr}(T_1 \bowtie OrderDetail)$

$CARD(T_2) = CARD(T_1) * CARD(Prelievo) * (1 / VAL(Lotto)) = 25 * 50.000 * (1 / 10.000) = 125$

$SIZE(T_2) = 10$

Notare che la proiezione può essere eseguita con lo scan, in questo modo non sono necessari altri risultati intermedi.

Poniamo: $T_3 = \Pi_{\text{NumeroSerie, Macchina}}(\text{Produzione})$

$\text{CARD}(T_3) = 200.000$

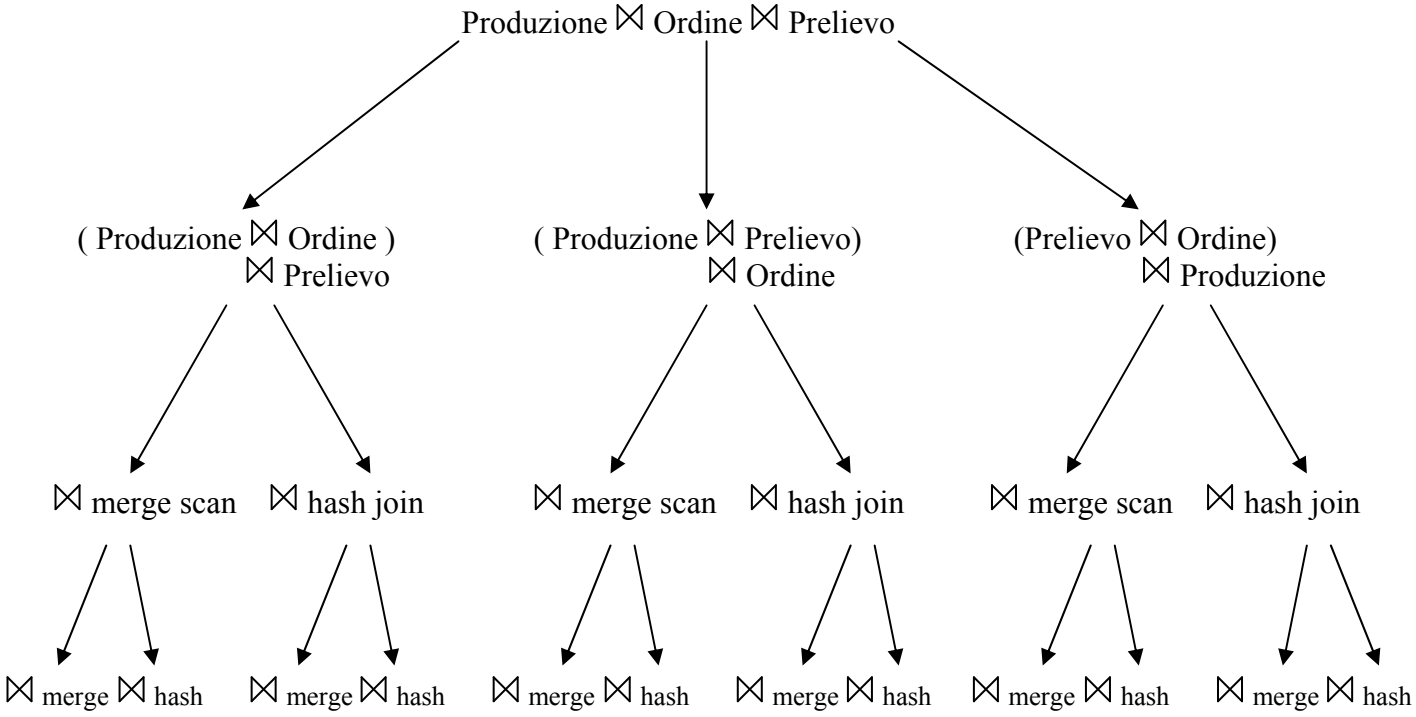
$\text{SIZE}(T_3) = 20$

Infine, poniamo $T_4 = \Pi_{\text{Macchina}}(T_2 \bowtie T_3)$

$\text{CARD}(T_4) = \textit{Min}(\text{CARD}(T_2) * \text{CARD}(T_3) * (1 / \text{VAL}(\text{NUMEROSERIE})), \text{VAL}(\text{Macchina})) = 50$

$\text{SIZE}(T_4) = 10$

Albero delle decisioni:



La prima scelta riguarda l'ordine dei join, la seconda scelta indica il tipo del primo join scelto, l'ultima scelta indica il tipo dell'ultimo join scelto.

3) SQL:

```
select Cliente
from Ordine join Prelievo on
Ordine.Lotto=Prelievo.Lotto
join Commissione on
Ordine.Lotto=Commissione.Lotto
where Venditore='Bianchi' and NumeroSerie='3478'
```

Algebra relazionale:

$$\Pi_{\text{Cliente}} (\sigma_{\text{Venditore='Bianchi'} \wedge \text{NumeroSerie='3478'}} (\text{Ordine} \bowtie_{\text{Lotto=Or}} \rho_{\text{Or}\leftarrow\text{Lotto}} (\text{Prelievo}) \bowtie_{\text{Lotto=Or2}} \rho_{\text{Or2}\leftarrow\text{Lotto}} (\text{Commissione})))$$

Ottimizzazione algebrica:

$$\Pi_{\text{Client}} (\Pi_{\text{OrdNumber}} (\sigma_{\text{ProdNumber='3478'}} (\text{OrderDetail}) \bowtie_{\text{OrderNumber=Or1}} (\Pi_{\text{Or1}} (\sigma_{\text{Seller='White'}} (\rho_{\text{Or1}\leftarrow\text{OrderNumber}} (\text{Commission}))) \bowtie_{\text{OrderNumber=Or2}} (\Pi_{\text{Or2,Client}} (\rho_{\text{Or2}\leftarrow\text{OrderNumber}} (\text{Order})))$$

Poniamo $T_1 = \sigma_{\text{NumberoSerie='3478'}} (\text{Prelievo})$

$\text{CARD}(T_1) = (1 / \text{VAL}(\text{NumeroSerie})) * \text{CARD}(\text{Prelievo}) = (1/200.000) * 50.000 = 0,25$

$\text{SIZE}(T_1) = 15$

Poniamo $T_2 = (\Pi_{\text{Lotto}} (\sigma_{\text{Venditore='Bianchi'}} (\text{Commissione}))$

$\text{CARD}(T_2) = 200$

$\text{SIZE}(T_2) = 5$

Poniamo $T_3 = \Pi_{\text{Lotto}} (T_1 \bowtie T_2)$

$\text{CARD}(T_3) = 0,25 * 200 * (1/10.000) = 0,005$

$\text{SIZE}(T_3) = 5$

Poniamo $T_4 = \Pi_{\text{Lotto, Cliente}} (\text{Ordine})$

$\text{CARD}(T_4) = 10.000$

$\text{SIZE}(T_4) = 35$

Poniamo $T_5 = \Pi_{\text{Client}} (T_3 \bowtie T_4)$

$\text{CARD}(T_5) = 0,005 * 10.000 * (1/10.000) = 0,005$

$\text{SIZE}(T_5) = 30$

Notare che il valore di $\text{CARD}(T_i)$ può essere anche minore di 1: CARD è solo un valore statistico.

Esercizio 11.5

Elencare le condizioni (dimensioni delle tabelle, presenza di indici o di organizzazioni sequenziali o a hash) che rendono più o meno conveniente la realizzazione di join con i metodi nested loop, merge scan e hash-based; per alcune di queste condizioni, proporre delle formule di costo che tengano conto essenzialmente del numero di operazioni di ingresso/uscita come funzione dei costi medi delle operazioni di accesso coinvolte (scansioni, ordinamenti, accessi via indici).

Soluzione:

Dimensione delle tabelle: se siamo in una situazione in cui una delle tabelle è molto grande rispetto all'altra, può essere una buona soluzione l'utilizzo di un join con il metodo nested loop, usando la tabella maggiore come esterna e quella più piccola come interna.. Se le tabelle hanno la stessa dimensione il join con il metodo nested loop è conveniente in presenza di indici o hashing su una delle tabelle. Negli altri casi è meglio scegliere un merge-scan

Hashing: la presenza di una funzione di hash può suggerire l'utilizzo di un join hash-based o nested loop. La scelta dipende dalla dimensione di una tabella e dal numero di partizioni prodotte dalla funzione di hash.

Organizzazione sequenziale: naturalmente, se le due tabelle hanno un'organizzazione sequenziale sugli attributi del join, il merge-scan è la scelta migliore. Se solo una tabella ha un'organizzazione sequenziale il merge scan può essere ancora una buona scelta se non ci sono particolari condizioni (indici o hashing).

Indici:La presenza di indici in generale suggerisce un nested loop. Comunque, se la tabella con gli indici è molto piccola, o se l'indice è sparso, può essere meglio non utilizzare l'indice e fare uno scan completo.

Il costo di un join di tipo nested loop senza indici può essere espresso come:

C_{NL}=scan(T_1) + T_1 (scan (T_2))

Dove T_1 e T_2 sono il numero di tuple delle due tabelle, e scan(T) è il costo medio di un'operazione di scansione.

Se c'è un indice sulla tabella 2:

C_{NL}=scan(T_1) + T_1 (index (T_2))

Un merge scan di un'organizzazione non sequenziale ha un costo:

C_{MS}= order(T_1) + order(T_2) + scan(T_1) + scan(T_2)

Esercizio 11.6

```
select distinct C, L
from R1, R2, R3
where R1.C = R2.D and R2.F = R3.G and R1.B = R3.L and R3.H > 10
```

Mostrare un possibile piano di esecuzione (in termini di operatori di algebra relazionale e loro realizzazioni; prestate attenzione anche alla `DISTINCT`, in quanto le relazioni degli operatori non producono necessariamente insiemi, ma liste di tuple), giustificando brevemente le scelte più significative, con riferimento alle seguenti informazioni sulla base di dati:

- la relazione R1(ABC) ha 100.000 tuple, una struttura heap e un indice secondario su C;
- la relazione R2(DEF) ha 30.000 tuple, una struttura heap e un indice secondario sulla chiave D;
- la relazione R3(GHL) ha 10.000 tuple, una struttura heap e un indice secondario sulla chiave G.

Soluzione:

Un possibile piano di esecuzione per l'interrogazione in oggetto è basato essenzialmente sulla struttura fisica delle relazioni e sulla loro cardinalità.
La parte dell'interrogazione da analizzare è quindi quella della condizione, che viene riportata di seguito per comodità.

```
where R1.C = R2.D and R2.F = R3.G and R1.B = R3.L and R3.H > 10
```

Si effettuano i seguenti passaggi:

1. MERGE JOIN di tipo **scan** sulle relazioni R1 ed R2, in quanto R1.C e R2.D hanno un indice secondario.
2. MERGE JOIN di tipo nested loop sulle relazioni R2 e R3, ponendo R2.F = R3.G, mantenendo R2 esterno e R3 interno.
3. Selezione, tramite una scansione, degli attributi R1.B = R3.L e R3.H > 10

Esercizio 11.7

```
select distinct A, L
from T1, T2, T3
where T1.C = T2.D and T2.E = T3.F and T1.B = 3
```

Mostrare un possibile piano di esecuzione (in termini di operatori dell'algebra relazionale e loro realizzazioni), giustificando brevemente le scelte più significative, con riferimento alle seguenti informazioni sulla base dati:

- la relazione T1(ABC) ha 800.000 tuple e 100.000 valori diversi per l'attributo B, distribuiti uniformemente; ha una struttura heap e un indice secondario sulla chiave A
- la relazione T2(DE) ha 500.000 tuple; è definito un vincolo di riferimento fra l'attributo E e la chiave F della relazione T3: ha una struttura heap e un indice secondario sulla chiave D
- la relazione T3(FL) ha 1.000 tuple e ha una struttura hash sulla chiave F.

Soluzione:

Un possibile piano di esecuzione per l'interrogazione in oggetto è basato essenzialmente sulla struttura fisica delle relazioni e sulla loro cardinalità.
La parte dell'interrogazione da analizzare è quindi quella della condizione, che viene riportata di seguito per comodità

```
where T1.C = T2.D and T2.E = T3.F and T1.B = 3
```

Si effettuano i seguenti passaggi

1. Selezione, tramite una scansione, dell'attributo `T1.B = 3`. Questa soluzione ha un costo molto elevato, 800.000 accessi in memoria, ma permette di estrarre un numero molto basso di record, in quanto possiede 100.000 valori diversi e mediamente ci saranno 8 record per ogni occorrenza.
2. JOIN di tipo nested loop sulle relazioni T1 e T2, ponendo T1.C = T2.D, tenendo il risultato del passo precedente come tabella interna e T2 come esterna.
3. JOIN di tipo nested loop usando l'indice hash su T3.F

Esercizio 11.8 Calcolare il fattore di blocco e il numero di blocchi occupati da una relazione con $T = 1000000)$ di tuple di lunghezza fissa pari a $L = 200$ byte in un sistema con blocchi di dimensione pari a $B = 2$ kilobyte.

Soluzione

Dimensioni tabella (D_T):
$D_T = T * L$
$D_T = 1000000 * 200 = 200000000$ byte $= 190.73$ MB

Numero blocchi:
$N_B = D_T/B$
$N_B = 200000000/2048 = 97.66$ blocchi

Fattore di blocco:
$F_B = B/L$
$F_B = 2048/200 = 10.24$ record / blocco

Esercizio 11.9 Si considerino:

- un sistema con blocchi di $B = 1000$ byte e indirizzi ai blocchi di $p = 2$ byte; il sistema (in effetti un po' obsoleto) prevede indici, primari o secondari, a un solo livello e solo sul campo chiave;
- una relazione con $N = 1000000$ tuple, ciascuna di $l = 100$ byte, di cui $k = 8$ byte per la chiave.

Calcolare:

1. il numero dei blocchi necessari per un indice primario sul campo chiave;
2. il numero dei blocchi necessari per un indice secondario;
3. il numero di accessi a memoria secondaria necessari sequenziale sulla chiave;
4. il numero di accessi a memoria secondaria necessari utilizzi un indice primario;
5. il numero di accessi a memoria secondaria necessari utilizzi un indice secondario.

Calcolare il numero di accessi sia nel caso peggiore sia medio, nell'ipotesi che le ricerche abbiano successo in casi (cioè otto volte su dieci il record cercato esiste).

Soluzione

1. Dimensione record indice primario:
   | valore chiave | valore ind memoria | $KP = K + P$
   numero di blocchi (e non di record poiché la struttura è ordinata) a cui deve puntare l'indice primario (sparso):

   $NB = (K * N)/B = 8 * 1000000/1000 = 8000$ blocchi
   dimensione indice = $(K + P) * NB = (8 + 2) * 8000 = 80000$byte
2. Dimensione record indice secondario
   | valore chiave | valore ind memoria | $KP = K + P$
   numero di record (e non di blocchi perch la struttura non è ordinata) a cui deve puntare l'indice secondario (denso) pari a l
   dimensione indice $= l * (p + k) = 1000000 * 10 = 10000000$ byte
3. Supponendo l'assenza di buffer, il numero di accessi alla memoria secondaria necessari per un FULLSCAN della tabella sono pari, nel caso peggiore, al numero dei blocchi della tabella stessa e quindi:
   Accessi =$l * n/b = 100 * 1000000/1000 = 100000$
4. Utilizzando l'indice primario (separato dal file) si fa riferimento ad una struttura ordinata ad un solo livello
   numero blocchi indice = dimensione_indice$/B = 80000/1000 = 80blocchi$
   80 accessi nel caso peggiore.
   É necessario, infine, un ulteriore accesso per recuperare il record puntato.
5. Utilizzando l'indice secondario ad un solo livello la complessità è data dal numero di blocchi dell'indice. Si tiene inoltre conto dell'opportuna probabilità di esistenza del record.
   numero blocchi indice = dimensione_indice $/B = 10000000/1000 = 10000$blocchi

Essendo la struttura ordinata, la probabilità di visitare tutto l'indice è data, nel caso peggiore, dalla probabilità di presenza della chiave in ultima posizione (s).
Si ha accessi_medi $= 10000 * s = 8000$accessi
E' necessario, infine, un ulteriore accesso per recuperare il record puntato.

Esercizio 11.10 Si considerino un sistema con blocchi di dimensione $B = 1000$ byte e puntatori ai blocchi di $P = 2$byte e una relazione R(A,B,C,D,E) di cardinalità pari circa a $N = 1000000$, con tuple di $L = 50$byte e campo chiave A di $K = 5$ byte. Valutare i pro e i contro (in termini di numero di accessi a memoria secondaria e trascurando le problematiche relative alla concorrenza) relativamente alla presenza di un indice secondario sulla chiave A e di un altro, pure secondario, su B, in presenza del seguente carico applicativo:

1. inserimento di una nuova tupla (con verifica del soddisfacimento del vincolo di chiave), con frequenza $f_1 = 500$ volte al minuto;
2. ricerca di una tupla sulla base del valore della chiave A con frequenza $f_2 = 500$ volte al minuto;
3. ricerca di tuple sulla base del valore di B con frequenza $f_3 = 100$.

Soluzione
Si suppone la disponibilità di buffer che permettano di mantenere stabilmente in memoria due livelli per ciascun indice e considerando che la relazione possa essere memorizzata in forma contigua (assumendo un rapporto 100:1 fra tempo di posizionamento della testina e tempo di lettura).

Si suppone inoltre $L_B = 3$.
È necessario calcolare il costo delle quattro operazioni, in presenza e assenza degli indici. Notazioni:

N_T numero di blocchi della relazione T; è pari circa a $N/(B/L)$ = 50.000
$cont$ riduzione di costo dovuta alla contiguità; è pari a 100.
seq costo della scansione sequenziale $1 + (NT1)/cont$; è pari a circa 500.
$prof_A$ il fattore di blocco dell' indice è 1.000/7, circa 140 e quindi, con un riempimento di circa il 60-70%, il fan-out è circa 100 e quindi i livelli necessari sono 3.
$prof_B$ il fattore di blocco dellindice è 1.000/5, circa 200 e quindi, con un riempimento di circa il 60-70%, il fan-out è circa 135 e quindi i livelli necessari sono ancora 3.

Op_1 È influenzata da entrambi gli indici anche se in modo diverso. Vediamo i diversi comportamenti.

Nessun indice: è necessaria la scansione sequenziale per verificare il vincolo di chiave. Costo pari a $seq = 500$; nessun costo per la manutenzione degli indici.

Indice su A: ed accesso tramite lindice. Costo pari alla profondità dellindice su A, più uno (per accedere al blocco del file) meno due (grazie ai buffer); costo pari a 1.

Indice su B: scansione sequenziale per la verifica della chiave e accesso allindice per laggiornamento.
Indici sia su A sia su B: accesso ai due indici.

Op_2 È influenzata dal solo indice su A accessi diretti o sequenziali.

Op_3 È influenzata dal solo indice su B. E' sufficiente aggiungere al risultato precedente il fattore relativo alla molteplicità.

Esercizio 11.11

Alcuni DBMS permettono una tecnica di memorizzazione chiamata "co-clustering" o "clustering eterogeneo" in cui un file contiene record di due o più relazioni e tali record sono raggruppati (eventualmente, ma non necessariamente ordinati) secondo i valori di opportuni campi dell'una e dell'altra relazione. Ad esempio, date due relazioni

- Ordini(CodiceOrdine, Cliente, Data, Importo)
- LineeOrdine(CodiceOrdine, Linea, prodotto, Quantità, Importo)

Questa tecnica (con riferimento agli attributi CodiceOrdine delle due relazioni) permetterebbe una memorizzazione contigua di ciascun ordine con le rispettive "linee d'ordine", cioè dei prodotti ordinati (ciascun ordine fa riferimento a più prodotti, ognuno su una "linea").
Con riferimento all'esempio, indicare quali delle seguenti operazioni possono trarre vantaggio dall'uso di questa opportunità e quali ne possono essere penalizzate (spiegare la risposta anche in termini quantitativi, individuando valori opportuni per i principali parametri di interesse; supporre che siano utilizzati indici su CodiceOrdine, in tutti i casi, due per la memorizzazione tradizionale e uno nel caso di utilizzo del cluster eterogeneo):

1. stampa dei dettagli (cioè delle linee d'ordine) di tutti gli ordini (ordinati per codice)
2. stampa dei dettagli di un ordine
3. stampa delle informazioni sintetiche (codice, cliente, data, totale) di tutti gli ordini.

Soluzione:

1. Per quanto riguarda la prima operazioni, non ci sono dubbi che una struttura co-cluster porti notevoli benefici, in quanto è sufficiente posizionarsi sul primo record della relazione e scorrerne tutto il contenuto. Quindi, la prima operazione è VANTAGGIOSA.
2. La seconda operazione è più delicata, in quanto se si ha a disposizione una struttura hash o ad albero sulla relazione ordini è vantaggiosa, mentre se si ha una struttura sequenziale o disordinata è svantaggiosa. Quindi, in questo caso DIPENDE DALLA STRUTTURA DELLA RELAZIONE ORDINI.
3. La terza operazione è sicuramente svantaggiosa, perché in ogni caso devo accedere a tutte le tuple del co-cluster nonostante abbia bisogno dei soli dati della relazione ordini. Quindi, la terza operazione è SVANTAGGIOSA.

Esercizio 11.12 Si consideri una relazione IMPIEGATO (Matricola, Cognome, Nome, Data- Nascita) con un numero di ennuple pari a Nabbastanza stabile nel tempo (pur con molti inserimenti ed eliminazioni) e una dimensione di ciascuna ennupla (a lunghezza fissa) pari a L byte, di cui K per la chiave Matricola e C per il campo Cognome.
Supporre di avere a disposizione un DBMS che permetta strutture fisiche disordinate (heap), ordinate (con indice primario sparso) e hash e che preveda la possibilità di definire indici secondari e operi su un sistema operativo che utilizza blocchi di dimensione B e con puntatori ai blocchi di P caratteri.
Indicare quale possa essere lorganizzazione fisica preferita nel caso in cui le operazioni principali siano le seguenti:

1. ricerca sul cognome (o una sua sottostringa iniziale, abbastanza selettiva, si supponga che mediamente una sottostringa identifichi $S = 10$ ennuple) con frequenza f_1;
2. ricerca sul numero di matricola, con frequenza f_2;
3. ricerca sulla base di un intervallo della data di nascita (poco selettivo, restituisce circa il 5% delle ennuple), con frequenza f_3 molto minore di f_1 e f_2;

assumendo $N = 10000000$ ennuple, $L = 100$ byte, $K = 5$ byte, $C = 20$ byte, $B = 1000$ byte, $P = 4$ byte, $f_1 = 100$ volte al minuto, $f_2 = 2000$ volte al minuto, $f_3 = 1$ volte al minuto. Individuare le alternative più sensate sulla base di ragionamenti qualitativi e poi valutarle quantitativamente, ignorando i benefici derivanti dai buffer e dalla contiguità di memorizzazione.

Soluzione

Dividiamo la risposta in tre parti:
Scelta qualitativa delle strutture fisiche più indicate:

Le strutture fisiche debbono privilegiare le operazioni 1 e 2 che sono più frequenti della 3 (che per giunta è poco selettiva).

1. Per l'operazione 1 la struttura hash su cognome non va bene, perché le ricerche non sono esatte (cioè sono spesso fatte con parte del cognome). Poiché nel testo non è chiarito se oltre ad essere abbastanza stabile la dimensione sono limitati anche gli inserimenti e le eliminazioni, non si può dire se sia preferibile una struttura ordinata su cognome (con un indice sparso) oppure una disordinata (con indice denso su cognome); nel prosieguo, supponiamo che il grado di dinamicità sia limitato (o che sia possibile avere tempi morti per la riorganizzazione) e quindi sia possibile una struttura ordinata (che darebbe grandi benefici per le ricerche su sottostringa). In ogni caso, nell'interesse dell'operazione 2, sarebbe utile un indice secondario sulla matricola.
2. La struttura che meglio privilegerebbe l'operazione 2 è quella hash su matricola: la struttura hash è la più effciente per l'accesso diretto puntuale (cioè con un valore di chiave completo), a patto che la dimensione della relazione

sia abbastanza stabile; in questo caso entrambe le condizioni sono soddisfatte. In questo contesto, per supportare adeguatamente anche l'operazione 1, si potrebbe utilizzare un indice secondario sul cognome.

Valutazione analitica:
Ignoriamo gli arrotondamenti. Sono necessari $K = 5$ byte per la chiave e $P = 4$ byte per i puntatori ai blocchi.
In tutti i casi abbiamo: fattore di blocco del file (numero di record per blocco): $F_f = B/L = 10$
fattore di blocco dell'indice: $F_i = B/(K + P) = 111$
Esaminiamo le due alternative sopra illustrate
Costo unitario delle tre operazioni:

1. accesso per cognome tramite l'indice secondario; l'indice ha N record (poiché denso) e quindi l'albero ha N/F_i foglie e profondità pari a $log_{F_i}(N/Fi)$; un'operazione ha costo pari alla profondità aumentata di S (l'indice è secondario, quindi gli accessi a cognomi consecutivi portano a blocchi diversi);
2. accesso hash sulla matricola: costo costante, circa 1.3 accessi a blocchi in media;
3. accesso sequenziale: costo pari al numero di blocchi del file: $(N * 1.5)/F_f$ (il fattore 1.5 è motivato dagli spazi liberi necessari per la struttura hash).

costo totale:
$f_2 * 1.3 + f_1 * (S + log_{F_i}(N/F_i)) + f_3 * (N * 1.5)/F_f$

con $f_1 = 100$, $f_2 = 2000$, $f_3 = 1$

$2000*1.3+100*(10+log_{111}(10000000/111))+1*(10000000*1.5)/10 = 78800$

Costo unitario delle tre operazioni:

1. accesso per cognome tramite l'indice primario; l'indice ha N/F_f record (poiché sparso) e quindi l'albero ha $(N/F_f)/F_i$ foglie e profondità pari a $log_{F_i}((N/F_f)/F_i)$; un'operazione ha costo pari alla profondità aumentata di 1 (in questo caso cognomi consecutivi portano a record adiacenti);
2. accesso per matricola tramite l'indice secondario; come sopra, ma con un solo blocco da accedere invece di S, visto che le matricole sono "esatte": un'operazione ha costo pari $1 + log_{F_i}(N/F_i)$
3. accesso sequenziale come sopra (ma senza il fattore 1.5)

Costo totale:

$$f_1 * (1 + log_{F_i}((N/F_f)/F_i)) + f_2 * (1 + log_{F_i}(N/F_i)) + f_3 * N/F_f$$

$$f_1 = 100, f_2 = 2000, f_3 = 1$$

$100*(1+log_{111}((10000000/10)/111))+(2000*(1+log_{111}(10000000/111))+10000000/111 = 6107$

Esercizio 11.13 Illustrare brevemente vantaggi e svantaggi (relativamente a vari tipi di operazioni) delle seguenti strutture fisiche, definite con riferimento ad uno stesso attributo A non chiave:

1. indice primario (sparso) multilivello statico;
2. indice secondario multilivello statico;
3. B+ tree secondario;
4. hash.

Soluzione

1. Indice primario (sparso) multilivello statico: va bene per ricerche puntuali e su intervallo, ma soffre in caso di aggiornamenti.
2. Indice secondario multilivello statico: va bene per ricerche puntuali e su intervallo (un po' meno del precedente, perché l'indice è più grande e soprattutto perché il file non è ordinato, il che è rilevante perchè il campo non è chiave e sono considerate ricerche su intervalli), ma soffre in caso di aggiornamenti.
3. B+ tree secondario: come i precedenti, un po' meno effciente ma senza svantaggi particolari in presenza di aggiornamenti.
4. Hash: molto effciente per accessi puntuali, ma non per intervalli; degenera se le dimensioni variano signifcativamente.

Esercizio 11.14 Indicare, in ciascuno dei quattro casi dell'esercizio 11.13(sia in forma simbolica sia in forma numerica), il costo di operazioni di ricerca basate su A ("trovare i record con un certo valore per l'attributo A"), con riferimento ad una relazione R contenente $L = 100000000$ ennuple di $R = 25$ byte ciascuna, di cui $a = 12$ per l'attributo A. Si considerino mediamente $m = 3$ record con lo stesso valore su A; supporre che i blocchi abbiano dimensione $B = 2$KB, approssimabile come $B = 2000$e che i puntatori ai blocchi abbiano lunghezza $p = 4$.

Soluzione

1. $\text{LIV}_1 = 4$, dove LIV_1 è la profondità dell'indice, pari a $log_F L/(B/R) = 3$, dove F è il fattore di blocco dell'indice, pari a $B/(a + p)$;
2. $\text{LIV}_2 + m = 7$, dove LIV_2 è la profondità dell'indice, pari a $log_F L = 4$; si noti che il termine m è necessario qui perchè il file è disordinato, mentre non è necessario nel caso precedente, perché i record sono tutti nello stesso blocco (salvo poche eccezioni).
3. $\text{LIV}_3 + m = 8$, dove LIV_3 è la profondità dell'indice, pari a $log_{F'} L = 5$ dove $F' = 2/3F$ (supponendo un riempimento medio del 66%).
4. poco più di 1.

Capitolo 12

Esercizio 12.1

Descrivere la ripresa a caldo, indicando la costituzione progressiva degli insiemi di UNDO e REDO e le azioni di recovery, a fronte del seguente log:

DUMP, B(T_1), B(T_2), B(T_3), I(T_1, O_1, A_1), D(T_2, O_2, B_2), B(T_4), U(T_4, O_3, B_3, A_3), U(T_1, O_4, B_4, A_4), C(T_2), CK(T_1, T_3, T_4), B(T_5), B(T_6), U(T_5, O_5, B_5, A_5), A(T_3), CK(T_1, T_4, T_5, T_6), B(T_7), A(T_4), U(T_7, O_6, B_6, A_6), U(T_6, O_3, B_7, A_7), B(T_8), A(T_7), guasto

Soluzione:

1) Per prima cosa bisogna percorrere il log a ritroso fino al più recente record di check-point: CK(T1,T4,T5,T6)

   Si costruiscono gli insiemi di UNDO e di REDO:

   UNDO= { T_1, T_4, T_5, T_6 } REDO={ }

2) Il log viene percorso in avanti, aggiornando i due insiemi:

B(T_7)	UNDO= { T_1, T_4, T_5, T_6, T_7 }	REDO={ }
A(T_4)	UNDO= { T_1, T_4, T_5, T_6, T_7 }	REDO={ }
B(T_8)	UNDO= { T_1, T_4, T_5, T_6, T_7, T_8 }	REDO={ }
A(T_7)	UNDO= { T_1, T_4, T_5, T_6, T_7 }	REDO={ }

3) Il log viene ripercorso ancora a ritroso, fino all'operazione I(T1,O1,A1), eseguendo le seguenti operazioni:

   O_3=B_7
   O_6=B_6
   O_5=B_5
   O_4=B_4
   O_3=B_3
   Delete O_1

4) Il log viene ripercorso in avanti per rieseguire le operazioni di REDO, ma essendo vuoto questo insieme, nessuna operazione verrà eseguita.

Esercizio 12.2

Si supponga che nella situazione precedente si verifichi un guasto di dispositivo che coinvolge gli oggetti O1, O2, O3; descrivere la ripresa a freddo.

Soluzione:

La ripresa a freddo è articolata in tre fasi successive.

1. Il log viene percorso a ritroso fino al primo record DUMP e si ricopia selettivamente la parte deteriorata della base dati.

2. Si ripercorre in avanti il log, applicando relativamente alla parte deteriorata della base di dati sia le azioni sulla base di dati sia le azioni di commit o abort e riportandosi così nella situazione precedente al guasto.

   Insert $O_1=A_1$
   Delete O_2
   $O_3=A_3$
   Commit (T_2)
   Abort (T_4)
   $O_3=A_7$

3. Si svolge una ripresa a caldo.

Esercizio 12.3

Il check-point, nei vari DBMS, viene realizzato in due modi diversi:

1. in alcuni sistemi si prende nota delle transazioni attive e si rifiutano (momentaneamente) nuovi commit
2. in altri si inibisce l'avvio di nuove transazioni e si attende invece la conclusione (commit o abort) delle transazioni attive

Spiegare, intuitivamente, le differenze che ne conseguono sulla gestione delle riprese a caldo.

Soluzione:

Nel primo caso la ripresa a caldo è articolata in quattro fasi ben precise:
1) Si accede al check point e si costruiscono gli insiemi di UNDO e di REDO.
2) Si percorre il log in avanti aggiornando i due insiemi
3) Il log viene ripercorso all'indietro disfacendo le operazioni contenute nell'insieme di UNDO.
4) Si ripercorre il log in avanti effettuando le operazioni di REDO.

Nel secondo caso il check point non conterrà nessuna transazione, in quanto tutte le transazioni si sono concluse o con un commit o con un abort, quindi non ci sono transazioni attive.

Le operazioni di ripresa a caldo saranno sicuramente molto più semplici e veloci se si utilizza ,la seconda tecnica, in quanto è sufficiente rieseguire tutte le operazioni che seguono il record di check point. Di contro bisogna dire che nel secondo caso la base di dati viene fermata ogni volta che si deve eseguire un check point, con un conseguente degrado delle prestazioni.

Dovendo decidere quale delle due soluzioni adottare, la prima sarà normalmente preferibile, in quanto è opportuno avere delle buone prestazioni per la maggior parte del tempo piuttosto che al verificarsi di eventi che richiedono una ripresa a caldo, considerati rari.

Esercizio 12.4

Indicare se i seguenti schedule possono produrre anomalie; i simboli *ci* e *ai* indicano l'esito (commit o abort) della transazione.

1. r1(x), w1(x), r2(x), w2(y), a1, c2
2. r1(x), w1(x), r2(y), w2(y), a1, c2
3. r1(x), r2(x), r2(y), w2(y), r1(z), a1, c2
4. r1(x), r2(x), w2(x), w1(x), c1, c2
5. r1(x), r2(x), w2(x), r1(y) , c1, c2
6. r1(x), w1(x), r2(x), w2(x) , c1, c2

Soluzione:

1) r1(x), w1(x), r2(x), w2(y), a1, c2

L'operazione r2(x) legge il valore scritto da w1(x), ma la transazione 1 termina con un abort. Questo è un caso di lettura sporca (Dirty Read)

2) r1(x), w1(x), r2(y), w2(y), a1, c2

Questo schedule non produce anomalie, perché le due transazioni fanno riferimento a oggetti differenti.

3) r1(x), r2(x), r2(y), w2(y), r1(z), a1, c2

Questo schedule non produce anomalie, perché la transazione 1 che termina in abort non effettua operazioni di scrittura.

4) r1(x), r2(x), w2(x), w1(x), c1, c2

Questo schedule ha una perdita di aggiornamento(lost update), in quanto gli effetti della transazione 2 vengono persi.

5) r1(x), r2(x), w2(x), r1(y) , c1, c2

Questo schedule non produce anomalie.

6) r1(x), w1(x), r2(x), w2(x) , c1, c2

Questo schedule non produce anomalie

Esercizio 12.5

Indicare se i seguenti schedule sono VSR.

1. r1(x), r2(y), w1(y), r2(x) w2,(x)
2. r1(x), r2(y), w1(x), w1(y), r2(x) w2,(x)
3. r1(x), r1(y), r2(y), w2(z), w1(z), w3(z), w3(x)
4. r1(y), r1(y), w2(z), w1(z), w3(z), w3(x), w1(x)

Soluzione:

Gli archi presentati indicano solo la relazione LEGGE - DA

1) r1(x), r2(y), w1(y), r2(x) w2(x)

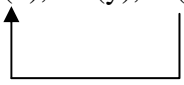
   Questo schedule non è VSR, perché i due schedule seriali:

   S1: r1(x), w1(y), r2(y), r2(x), w2(x) e

   S2: r2(y), r2(x), w2(x), r1(x), w1(y)

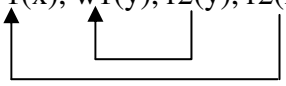
   non sono view-equivalenti con lo schedule dato. Hanno entrambi una differente relazione LEGGE - DA

2) r1(x), r2(y), w1(x), w1(y), r2(x) w2,(x)

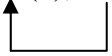


   Questo schedule non è VSR perché gli schedule

   S1: r1(x), w1(x), w1(y), r2(y), r2(x), w2(x)



   S2: r2(y), r2(x), w2(x), r1(x), w1(x),w1(y)



   hanno entrambi una differente relazione LEGGE – DA

3) r1(x), r1(y), r2(y), w2(z), w1(z), w3(z), w3(x)

   Questo schedule è VSR e view-equivalente allo schedule seriale, in quanto sono caratterizzati dalle stesse scritture finali e dalle stesse relazioni LEGGI –DA.

   S: r2(y), w2(z), r1(x), r1(y), w1(z), w3(z), w3(x)

4) r1(y), r1(y), w2(z), w1(z), w3(z), w3(x), w1(x)

Si noti che la transazione 1 ha due scritture, una su Z ed un'altra su X, ma anche la transazione 3 ha due scritture, una su Z ed un'altra su X. Nello schedule di partenza, le scritture finali su X e Z sono originate rispettivamente dalle transazioni 1 e 3. Nessuno schedule seriale potrà esibire le stesse scritture finali. Questo schedule non è quindi VSR.

Esercizio 12.6

Classificare i seguenti schedule (come: NonSR, VSR, CSR); nel caso uno schedule sia VSR oppure CSR, indicare tutti gli schedule seriali e esso equivalenti.

1. r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)
2. r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)
3. r1(x), r2(x), w2(x), r2(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)
4. r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)
5. r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)
6. r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)
7. r2(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

Soluzione:

1) r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)

   In questo schedule non ci sono conflitti, quindi è sia VSR che CSR, ed è conflict-equivalente a:

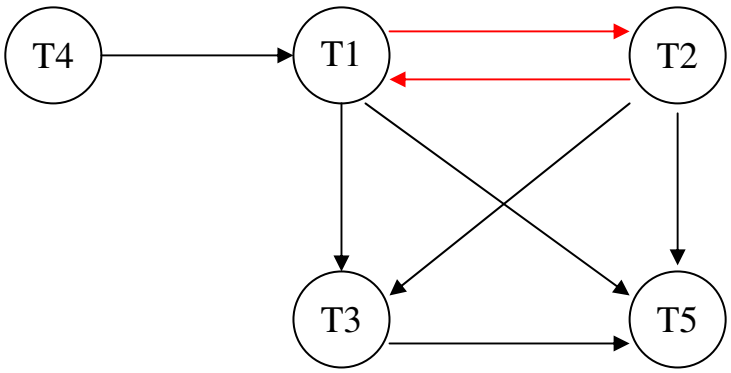
   S: r1(x), w1(x), r1(y), w1(y), r2(z), r2(x), w2(x), w2(z)

2) r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)

   Questo schedule è NonSR. In uno schedule seriale view-equivalente a questo schedule la transazione 1 dovrebbe seguire la transazione 3 a causa delle SCRITTURE FINALI su Y, ma dovrebbe anche precedere la transazione 3 a causa della relazione LEGGE – DA su X.

3) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)

   Per classificare questo schedule si deve realizzare un grafo dei conflitti.
   Consideriamo le operazioni relative a ogni risorsa separatamente:

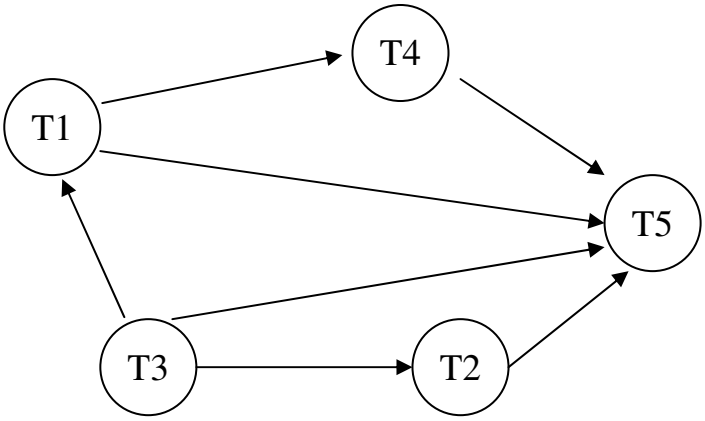


X: r1, r2, w2, r3, w1, w3, w5
Y: w3, w1, w5
Z: r4, w1, r5

   Questo schedule non è CSR perchè il suo grafo dei confliti è cliclico.
   Questo schedule è anche NonSR, perché, considerando la risorsa X, la transazione 1 dovrebbe precedere la transazione 2, e la transazione 2 dovrebbe precedere la transazione 1 (entrambe le transazioni leggono X prima di qualsiasi altra operazione).

4) r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)

Per classificare questo schedule si deve realizzare un grafo dei conflitti.
Consideriamo le operazioni relative a ogni risorsa separatamente:



T: w1, r4, r5
X: r1, w4, w5
Y: r3, w1
Z: r2, r3, w2, w5

Questo schedule è sia CSR che VSR.

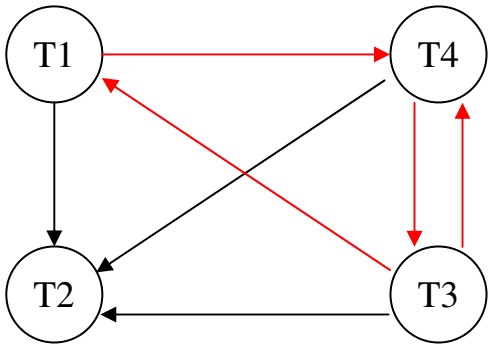
Gli schedule seriali equivalenti sono:
S1: r3(y), r3(z), r1(x), w1(y), w1(t), r2(z), w2(z), w4(x), r4(t), w5(x), w5(z), r5(t)
S2: r3(y), r3(z), r2(z), w2(z), r1(x), w1(y), w1(t), w4(x), r4(t), w5(x), w5(z), r5(t)

5) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)

Lo schedule è NonSR ; le transazioni 1 e 2 leggono e scrivono X. Leggono X prima di ogni altra operazione, e così nessuna sequenza di schedule con queste transazioni potrà verificare la relazione LEGGE - DA

6) r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)

Per classificare questo schedule si deve realizzare un grafico dei conflitti.
Consideriamo le operazioni relative a ogni risorsa separatamente:

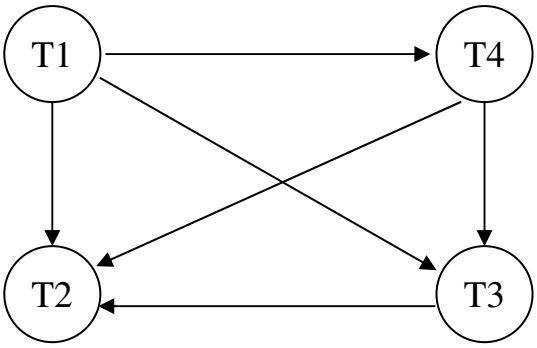


T: r1, w2
X: r1, r4, r3, w4
Y: w4, w3, w1
Z: r3, r4, w2

Questo schedule non è CSR perchè il suo grafo dei confliti è ciclico.
Questo schedule è anche NonSR; la transazione 1 ha la scrittura finale su Y, perciò dovrebbe seguire la transazione 4. La transazione 4 scrive su X, e dovrebbe seguire la transazione 1.

7) r1(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

Per classificare questo schedule si deve realizzare un grafo dei conflitti.
Consideriamo le operazioni relative a ogni risorsa separatamente:



T: w1, w2
X: r1, r4, w4
Y: r1, w3
Z: r4, w4, w3, w2

Visto che il grafo è aciclico è CSR, quindi anche VSR.

Lo schedule seriale equivalente è:

S: r1(x), r1(y), w1(t), r4(x), w4(x), r4(z), w4(z), w3(y), w3(z), w2(z), w2(t)

Esercizio 12.7

Se gli schedule dell'esercizio precedente si presentassero a uno scheduler che usa il locking a due fasi, quali transazioni verrebbero messe in attesa? Si noti che, una volta posta in attesa una transazione, le sue successive azioni non vanno più considerate.

Soluzione:

1) r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)

   Nessuna transazione in attesa

2) r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)

   Le transazioni 3 e 1 e poi 2 vengono messe in attesa, producendo una situazione di deadlock; infatti, l'azione r2(x) deve aspettare l'oggetto x, messo in lock dalla transazione 1, e la transazione 1 è in attesa su w1(y) dell'oggetto y, messo in lock dalla transazione 2.

3) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)

   Le transazioni 2, 1, 3 e 5 sono messe in attesa a causa del lock in lettura su x. Si crea una situazione di deadlock dovuta alla richiesta di lock in scrittura su x da parte delle transazioni 1 e 2, entrambe con un lock in lettura sulla stessa risorsa x.

4) r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)

   Le transazioni 1, 4, 5 e 2 sono in attesa. La transazione 1 deve aspettare per y (allocato da t3), le transazioni 4 e 5 devono aspettare per x (allocato da t1) e la transazione 2 deve aspettare per z (allocato da t3).

5) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)

   Le transazioni 2, 1 e 5 sono in attesa. Devono aspettare per x (allocata da t1, t2 e t3)

6) r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)

   Le transazioni 2, 3 e 4 vengono messe in attesa. t2 e t4 devono aspettare per z (allocata da t3) e t3 deve aspettare per y (allocata da t1)

7) r1(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

   Assumendo che t1 rilasci il lock esclusivo su t al termine delle sue operazioni, le transazioni 3 e 4 sono in attesa. Devono aspettare per x e y, allocate da t1.

Esercizio 12.8

Definire le strutture dati necessarie per la gestione del locking supponendo un modello non gerarchico e con read ripetibili; implementare in un linguaggio di programmazione a scelta le funzioni lock_r, lock_w e unlock. Si supponga disponibile un tipo di dato astratto "coda" con le opportune funzioni per inserire un elemento in coda ed estrarre il primo elemento da una coda.

Soluzione:

Le transazioni sono identificate con un numero, gli oggetti con una stringa.

```
typedef struct resource {
     char *identifier;
     int free; // 1 is free, 0 is busy
     int r_locked; // 1 or more locked, 0 free
     int w_locked; // 1 locked
     queue waiting_transaction;
}

typedef struct queue_element {
     int transaction;
     int type // 0 read, 1 write
}

typedef *resource locktable;

locktable lt=new locktable [N]; // N is the number of resources

int lock_r (int transaction, char *resource) {
     int i=0;
     int end=0;
     while ((i<N) && (!end))
       if (strcmp(resource, lt[i])!=0) i++; else end=1;
     if (!end) return -1; // resource not found
     if (lt[i].free)
     { lt[i].free=0;
       lt[i].r_locked=1;
       return transaction; }
       if (lt[i].r_locked)
       { lt[i].r-locked++;
         return transaction; }
     // the resource is w_locked

     queue_element q=new queue_element;
     q->transaction=transaction;
     q->type=0;
     in_queue(q,lt[i].queue);
     return 0; }

int lock_w (int transaction, char *resource) {
     int i=0;
     int end=0;
```

```
     while ((i<N) && (!end))
       if (strcmp(resource, lt[i])!=0) i++; else end=1;
     if (!end) return -1; // resource not found
     if (lt[i].free)
          { lt[i].free=0:
            lt[i].w_locked=1;
            return transaction;
          }
     queue_element q=new queue_element;
     q->transaction=transaction;
     q->type=1;
     in_queue(q,lt[i].queue);
     return 0;
}

int unlock (int transaction, char *resource) {
     int i=0;
     int end=0;
     while ((i<N) && (!end))
       if (strcmp(resource, lt[i])!=0) i++; else end=1;
     if (!end) return -1; // resource not found

     if (lt[i].r_locked)
          { lt[i].r_locked--;
          if (lt[i].r_locked) return 0;
          if (is_empty(lt[i].queue))
               { lt[i].free=1;
               return 0;
          }
          queue_element q=out_queue(lt[i].queue);
          lt[i].w_locked=1;

          return q.transaction;
          }
     if (is_empty(lt[i].queue))
          { lt[i].free=1;
            return 0;
          }
     queue_element q=out_queue(lt[i].queue);
     if (!(q.type)) // read transaction in queue
          { lt[i].r_locked=1;
            lt[i].w_locked=0; }
          // now unlock extracts all the read-transaction from
          //the queue;
     while (!(first_element(lt[i].queue)).type))
          { out_queue (lt[i].queue);
            r_locked++;
          }
     return q.transaction; }
```

Esercizio 12.9

Facendo riferimento all'esercizio precedente, aggiungere un meccanismo di timeout; si suppongano disponibili le funzioni di sistema per impostare il timeout, per verificare (a controllo di programma) se il tempo fissato è trascorso, e per estrarre uno specifico elemento da una coda.

Soluzione:

La struttura queue_element deve essere modificata nel seguente modo:

```
typedef struct queue_element {
      int transaction;
      int type;
      int time;
}
```

Il campo time rappresenta l'istante in cui la transazione viene messa in attesa.
Le funzioni lock_r e lock_w devono riempire anche questo campo prima di mettere le transazioni in coda, usando l'istruzione:

```
q.time=get_system_time();
```

Il meccanismo di timeout è realizzato da una nuova funzione, check_time, che è chiamata periodicamente dal sistema.

```
void check_time() {
      int now=get_system_time();
      for (int i=0:; i<N; i++)
      if (!is_empty(lt[i].queue))
      { int l=queue_length(lt[i].queue);
      for (int j=0; j<l; j++)
            { queue_element q=get_queue_element(lt[i].queue,j);
                  if ((q.time+MAX_TIME)<now)
                  remove_from_queue(lt[i].queue,j);
            }
      }
}
```

Esercizio 12.10

Se gli schedule descritti nell'esercizio 2.6 si presentassero a uno scheduler basato su timestamp, quali transazioni verrebbero abortite?

Soluzione:

1) r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
write(x,1)	Ok	WTM(x)=1
read(z,1)	Ok	RTM(z)=1
read(y,1)	Ok	RTM(y)=1
write(y,1)	Ok	WTM(y)=1
read(x,2)	Ok	RTM(x)=2
write(z,2)	Ok	WTM(z)=2

Non viene abortita nessuna transazione.

2) r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
write(x,1)	Ok	WTM(x)=1
write(x,3)	Ok	WTM(x)=3
read(y,2)	Ok	RTM(y)=2
read(y,3)	Ok	RTM(y)=3
write(y,3)	Ok	WTM(y)=3
write(y,1)	t_1 aborted	
read(x,2)	t_2 aborted	

3) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
read(x,2)	Ok	RTM(x)=2
write(x,2)	Ok	WTM(x)=2
read(x,3)	Ok	RTM(x)=3
read(z,4)	Ok	RTM(z)=4
write(x,1)	t_1 aborted	
write(y,3)	Ok	WTM(y)=3
write(x,5)	Ok	WTM(x)=5
write(y,5)	Ok	WTM(y)=5
read(z,5)	Ok	RTM(z)=5

4) r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
read(y,3)	Ok	RTM(y)=3
write(y,1)	t_1 aborted	
write(x,4)	Ok	WTM(x)=4
write(x,5)	Ok	WTM(x)=5
read(z,2)	Ok	RTM(z)=2
read(z,3)	Ok	RTM(z)=3
write(z,2)	t_2 aborted	
write(z,5)	Ok	WTM(z)=5
read(t,4)	Ok	RTM(t)=4
read(t,5)	Ok	RTM(t)=5

5) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
read(x,2)	Ok	RTM(x)=2
write(x,2)	Ok	WTM(x)=2
read(x,3)	Ok	RTM(x)=3
read(z,4)	Ok	RTM(z)=4
write(x,1)	t_1 aborted	
read(y,3)	Ok	RTM(y)=3
read(x,3)	Ok	RTM(x)=3
write(x,5)	Ok	WTM(x)=5
read(y,5)	Ok	RTM(y)=5
read(z,5)	Ok	RTM(z)=5

6) r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
read(t,1)	Ok	RTM(t)=1
read(z,3)	Ok	RTM(z)=3
read(z,4)	Ok	RTM(z)=4
write(z,2)	t_2 aborted	
read(x,4)	Ok	RTM(x)=4
read(x,3)	Ok	RTM(x)=4
write(x,4)	Ok	WTM(x)=4
write(y,4)	Ok	WTM(y)=4
write(y,3)	t_3 aborted	
write(y,1)	t_1 aborted	

7) r1(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

Operazione	**Risposta**	**Nuovo Valore**
read(x,1)	Ok	RTM(x)=1
read(x,4)	Ok	RTM(x)=4
write(x,4)	Ok	WTM(x)=4
read(y,1)	Ok	RTM(y)=1
read(z,4)	Ok	RTM(z)=4
write(z,4)	Ok	WTM(z)=4
write(y,3)	Ok	WTM(y)=3
write(z,3)	t_3 aborted	
write(t,1)	Ok	WTM(t)=1
write(z,2)	t_2 aborted	

Esercizio 12.11

Si consideri un oggetto X sul quale opera un controller di concorrenza basato su timestamp, con WTM(X) = 5, RTM(X) = 7. Indicare le azioni dello scheduler a fronte del seguente input nel caso mono-versione e in quello multi-versione:
r(x,8), r(x,17), w(x,16), w(x,18), w(x,23), w(x,29), r(x,20), r(x,30), r(x,25)

Soluzione:

Mono-versione:

Operazione	**Risposta**	**Nuovo Valore**
		WTM(x)=5
		RTM(x)=7
read(x,8)	Ok	RTM(x)=8
read(x,17)	Ok	RTM(x)=17
read(x, 16)	Ok	RTM(x)=17
write(x,18)	Ok	WTM(x)=18
write(x,23)	Ok	WTM(x)=23
write(x,29)	Ok	WTM(x)=29
read(x,20)	t_{20} aborted	
read(x,30)	Ok	RTM(x)=30
read(x,25)	t_{25} aborted	

Multi-versione:

Operazione	**Risposta**	**Nuovo Valore**
		$WTM_1(x)$=5
		RTM(x)=7
read(x,8)	Ok	RTM(x)=8
read(x,17)	Ok	RTM(x)=17
read(x, 16)	Ok	RTM(x)=17
write(x,18)	Ok	$WTM_2(x)$=18
write(x,23)	Ok	$WTM_3(x)$=23
write(x,29)	Ok	$WTM_4(x)$=29
read(x,20)	Ok	RTM(x)=20 reads from x_2
read(x,30)	Ok	RTM(x)=30 reads from x_4
read(x,25)	Ok	RTM(x)=30 reads from x_3

Esercizio 12.12

Spiegare perché il livello più alto di isolamento previsto nello standard SQL:1999 ("`serializable`") può avere effetti molto più pesanti anche solo del livello immediatamente inferiore ("`repeatable read`").

Soluzione:

Il livello `serializable` può avere effetti molto più pesanti rispetto al livello immediatamente inferiore, in quanto applica i lock di predicato che, in assenza di indici, possono bloccare gli accessi ad intere relazioni.