

Dario Maio · Stefano Rizzi · Annalisa Franco

Esercizi di PROGETTAZIONE DI BASI DI DATI

SOCIETÀ EDITRICE
ESCOLAPPIO

Dario Maio · Stefano Rizzi · Annalisa Franco

**Esercizi di
PROGETTAZIONE
DI
BASI DI DATI**



ISBN 978-88-7488-113-0

© Copyright 2005, 1997
Società Editrice Esculapio s.r.l.
Via Terracini, 30 – 40131 Bologna
www.editrice-esculapio.com – info@editrice-esculapio.it

Ristampa 2013

Impaginazione: Giancarla Panigali e Carlotta Lenzi
Stampato da: Legodigit – Lavis (TN)
Printed in Italy

Le fotocopie per uso personale (cioè privato e individuale, con esclusione quindi di strumenti di uso collettivo) possono essere effettuate, nei limiti del 15% di ciascun volume, dietro pagamento alla S.I.A.E del compenso previsto dall'art. 68, commi 4 e 5, della legge 22 aprile 1941 n. 633. Tali fotocopie possono essere effettuate negli esercizi commerciali convenzionati S.I.A.E o con altre modalità indicate da S.I.A.E. Per le riproduzioni ad uso non personale (ad esempio: professionale, economico o commerciale, strumenti di studio collettivi, come dispense e simili) l'editore potrà concedere a pagamento l'autorizzazione a riprodurre un numero di pagine non superiore al 15% delle pagine del volume.

CLEARedi - Centro Licenze e Autorizzazioni per le Riproduzioni Editoriali Corso di Porta Romana, n. 108 - 20122 Milano

e-mail: autorizzazioni@clearedi.org - sito: <http://www.clearedi.org>.



Prefazione alla Seconda Edizione

Obiettivo del volume è offrire al lettore un efficace strumento per la verifica e la revisione della propria preparazione teorica e pratica sui fondamenti delle basi di dati relazionali, con particolare accento sulle problematiche di progettazione. La collezione di esercizi proposta nasce dalle esperienze didattiche condotte dagli autori, nell'ambito dei corsi di basi di dati e sistemi informativi, presso la Facoltà di Ingegneria e la Facoltà di Scienze dell'Università di Bologna. Nell'esposizione delle soluzioni è stato dato spazio alla discussione critica delle possibili alternative, motivando le scelte adottate e stimolando il lettore a dare risposte ragionate a problemi ricorrenti nella pratica. L'applicazione consapevole e non meramente meccanica delle tecniche è infatti un requisito essenziale per affrontare con successo le complesse fasi della progettazione di sistemi informativi. Spesso, al termine dell'esposizione della soluzione di un esercizio, un paragrafo *Suggerimenti* riporta modifiche di specifiche e spunti per ulteriori approfondimenti.

Gli esercizi della prima parte affrontano il problema della progettazione concettuale a partire da requisiti in linguaggio naturale, facendo ricorso al formalismo Entity/Relationship. La seconda parte riguarda la progettazione logica, ovvero la trasformazione di uno schema concettuale in schemi relazionali; ampio spazio è dedicato alle problematiche di normalizzazione. La terza parte raccoglie esercizi di formulazione di interrogazioni in linguaggio SQL e di espressioni dell'algebra relazionale. Nella quarta parte vengono affrontati i problemi della stima dei costi di esecuzione e della progettazione fisica (limitatamente alla scelta degli indici). Infine, gli esercizi della quinta parte riguardano i dispositivi di memoria secondaria e le organizzazioni dei dati.

La tipologia degli argomenti affrontati e il livello di approfondimento fanno sì che il testo rappresenti un valido complemento per un corso di basi di dati di primo livello.

Per affrontare gli esercizi e trarre pieno profitto dall'esame delle soluzioni proposte è necessario possedere buone conoscenze teoriche di base sugli argomenti trattati; ognuna delle diverse parti in cui si articola il volume inizia con un breve capitolo che riassume le specifiche ipotesi e notazioni adottate.

Le correzioni e i suggerimenti per una migliore impostazione sono stati molteplici; per questo dobbiamo ringraziare Matteo Golfarelli, Alessandra Lumini e Davide Maltoni. Infine, un ringraziamento speciale va indirizzato ai nostri studenti che hanno svolto pazientemente un intenso *debugging* di tutto il materiale incluso nella prima edizione del 1997.

In questa seconda edizione sono stati aggiunti complessivamente più di sessanta nuovi esercizi, con l'obiettivo di rendere ancor più ampia la casistica delle situazioni di progetto affrontate.

Bologna, maggio 2005

A. Franco, D. Maio, S. Rizzi

Indice

Progettazione Concettuale	1
Nota sul formalismo adottato.....	2
Esercizi.....	5
Soluzioni.....	21
Progettazione Logica	61
Nota sul formalismo adottato.....	62
Esercizi.....	63
Soluzioni.....	73
SQL e Algebra Relazionale	91
Nota sul formalismo adottato.....	92
Esercizi.....	93
Soluzioni.....	107
Stima dei Costi e Progettazione Fisica	129
Nota sul formalismo adottato.....	130
Esercizi.....	131
Soluzioni.....	143
Dispositivi e Organizzazioni dei Dati	167
Nota sul formalismo adottato.....	168
Esercizi.....	169
Soluzioni.....	179
Bibliografia di base	195

Parte Prima

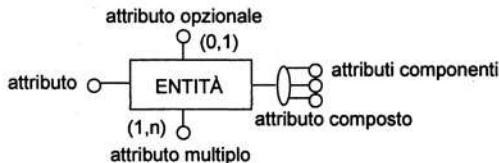
Progettazione Concettuale

Nota sul formalismo adottato

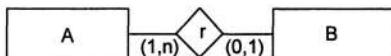
Per la progettazione concettuale abbiamo scelto il formalismo Entity/Relationship, di gran lunga il più diffuso sia negli ambienti universitari sia nelle organizzazioni aziendali. Inoltre, poiché in questo testo la fase di progettazione concettuale è mirata alla realizzazione di basi di dati relazionali, il formalismo E/R costituisce di fatto una scelta obbligata.

Esiste in letteratura una molteplicità di notazioni grafiche per la rappresentazione di associazioni e loro cardinalità, di gerarchie e loro coperture, ecc. Quella qui adottata può essere così riassunta:

- Ogni entità è rappresentata da un rettangolo che ne riporta il nome. Gli attributi di un'entità sono rappresentati tramite circoletti. Un attributo composto è rappresentato aggregando più circoletti in un ovale. Si intende che la cardinalità degli attributi sia (1,1); attributi multipli e/o opzionali verranno individuati indicandone esplicitamente la cardinalità.



- Un'associazione tra entità è rappresentata da un rombo che ne riporta il nome e eventuali attributi, indicati con circoletti bianchi. La cardinalità delle associazioni viene indicata in modo esplicito riportando cardinalità massima e minima per entrambi i versi di lettura dell'associazione. Ad esempio, lo schema in figura verrà così interpretato: un'istanza di A è in relazione r con una o più istanze di B; un'istanza di B è in relazione r con zero o una istanza di A.

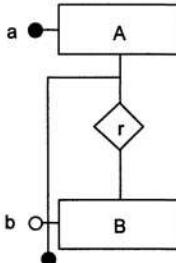


- Un attributo di un'entità che funge da identificatore è segnalato da un circoletto nero. Un identificatore composto è indicato raggruppando gli attributi componenti con un trattino. Più circoletti neri su una stessa entità indicano l'esistenza di più possibili identificatori.

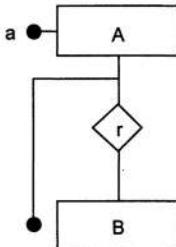


La presenza di un attributo di un'altra entità all'interno dell'identificatore di un'entità (identificatore *misto*) viene segnalata come mostrato in figura: l'identificatore di B è costituito dalla composizione degli attributi a e b. Si

ricorda che, affinché sia possibile utilizzare attributi di A nell'identificatore di B, occorre che B partecipi all'associazione r con cardinalità (1,1).

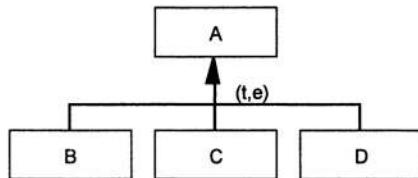


In alcuni casi si hanno identificatori esterni, a cui partecipano solo attributi di altre entità (l'identificatore di B è l'attributo a):



- Le gerarchie di specializzazione sono rappresentate come mostrato nella figura successiva. La copertura viene indicata in modo esplicito riportando una sigla con il seguente significato:

(t,e)	\Rightarrow	totale, esclusiva
(t,s)	\Rightarrow	totale, sovrapposta
(p,e)	\Rightarrow	parziale, esclusiva
(p,s)	\Rightarrow	parziale, sovrapposta



(le entità B, C, e D sono specializzazioni dell'entità A; la copertura è totale ed esclusiva).

Il processo completo di progettazione concettuale attraverso il filtraggio dei requisiti espressi in linguaggio naturale è documentato dall'esercizio 1.16. Per semplicità, i testi degli esercizi precedenti all'1.16 racchiudono in linea di massima requisiti già filtrati, così da evitare possibili ambiguità di interpretazione. Cionondimeno, al fine di rendere gli esercizi comunque interessanti e non banali, abbiamo talvolta omesso volontariamente alcune specifiche (tipicamente quelle

relative alla cardinalità delle associazioni), lasciando al buon senso il compito di desumerle.

□ Esercizio 1.1

Si consideri un sistema per la gestione di listini di vendita in valute diverse. Un listino è relativo a una zona di vendita, e comprende i prezzi di un insieme di prodotti. Un prodotto può comparire in più listini; in alcuni listini uno stesso prodotto può comparire più volte, quotato in valute diverse (ad esempio: lire, dollari e marchi). A ogni listino è associato uno sconto.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.2

Si vuole automatizzare il sistema di gestione degli animali in uno zoo. Ogni esemplare di animale ospitato è identificato dal suo genere (es. zebra) e da un codice unico all'interno del genere di appartenenza. Per ogni esemplare si memorizzano la data di arrivo nello zoo, il nome proprio, il sesso, il paese di provenienza e la data di nascita. Lo zoo è diviso in aree; in ogni area c'è un insieme di case, ognuna destinata a un diverso genere di animali. Ogni casa contiene un insieme di gabbie, ognuna contenente un solo esemplare. Ogni casa ha un solo addetto che pulisce ciascuna gabbia in un determinato giorno della settimana. Gli animali sono sottoposti periodicamente a controllo veterinario; in un controllo, un veterinario rileva il peso di un esemplare, diagnostica un'eventuale malattia e prescrive il tipo di dieta da seguire.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.3

È dato uno schema relazionale:

R1 (K1, A1)
R2 (K2, A2)
R3 (K1, K2, K3, A3)

Si disegni un possibile schema concettuale E/R corrispondente.

Esercizio 1.4

Si intende automatizzare la gestione degli orari delle lezioni in una facoltà universitaria. La facoltà prevede diversi corsi di laurea (ad esempio, per la facoltà di Ingegneria: corso di laurea in Ingegneria Elettronica, corso di laurea in Ingegneria Meccanica, ecc.), e un certo numero di insegnamenti. Ogni insegnamento è contraddistinto da un codice e un nome. La maggior parte degli insegnamenti appartiene al piano di studi di più corsi di laurea (ad esempio: l'insegnamento C15 Analisi Matematica I appartiene tanto a Ingegneria Elettronica quanto a Ingegneria Meccanica). Un dato insegnamento all'interno di un dato corso di laurea prevede un insieme di lezioni, ciascuna svolta in un certo orario e in una certa aula. Gli insegnamenti con molti studenti vengono suddivisi raggruppando alfabeticamente gli studenti; per un dato insegnamento all'interno di un dato corso di laurea, si possono pertanto avere più docenti (ad esempio, C15 Analisi Matematica I per Ingegneria Meccanica è suddiviso in due: A-K e L-Z, per gli studenti con iniziale del cognome compresa rispettivamente tra A e K e tra L e Z. Il docente è Rossi per gli A-K, Bianchi per gli L-Z). Le lezioni degli insegnamenti con pochi studenti vengono invece unificate per più corsi di laurea (ad esempio: le lezioni di Complementi di Matematica per i Meccanici e i Nucleari sono unificate, e vengono svolte dal Prof. Verdi).

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

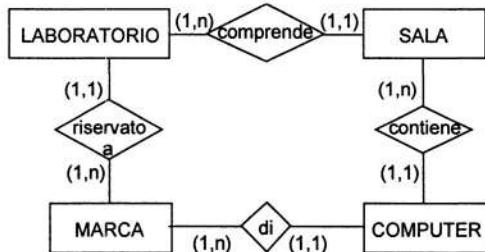
Esercizio 1.5

Un museo si compone di diverse sezioni, ciascuna comprendente un certo numero di sale. Ogni sezione ha un orario di visita ed è custodita giornalmente da un solo custode, nel rispetto di un turno settimanale che resta invariato per tutto l'anno; il turno definisce per ciascun custode la sezione di cui si deve occupare in ciascun giorno della settimana. Ciascuna sala comprende diverse opere d'arte; le opere d'arte si dividono in dipinti, sculture (a loro volta distinte in statue, bassorilievi e altorilievi), arazzi e ceramiche. Ogni opera ha un autore, ma di alcune non se ne conosce il nome. Ogni opera appartiene a un periodo storico (ad es. Rinascimento, Ottocento, ecc.); alcuni autori appartengono a un movimento artistico (ad es. Impressionismo, Puntoinismo).

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.6

Si consideri lo schema E/R in figura, in cui è presente un ciclo. Si stabilisca se il ciclo crea o meno una ridondanza; in caso affermativo, si determini quale delle 4 associazioni conviene cancellare al fine di eliminare la ridondanza.



□ Esercizio 1.7

Si vuole automatizzare il sistema di gestione di una palestra. Per ogni iscritto si vogliono memorizzare, oltre ai dati anagrafici, le date di inizio e di fine abbonamento, le presenze effettive in palestra (ogni presenza è descritta dalla data e dal numero di ore di permanenza; le presenze vengono cancellate allo scadere dell'abbonamento) e le schede di allenamento da lui attualmente utilizzate. Una scheda descrive la sequenza di esercizi da effettuare durante un allenamento; un iscritto può usare contemporaneamente fino a tre schede (a rotazione). Una scheda, compilata da un istruttore, contiene fino a un massimo di 20 esercizi; ciascun esercizio deve essere svolto in un certo numero di serie, ciascuna caratterizzata da un certo numero di ripetizioni ed, eventualmente, da un peso utilizzato (esempi: "distensioni panca piana": 3 serie di cui la prima da 12 ripetizioni con peso 10 Kg, la seconda da 10 con 12 Kg, la terza da 8 con 14 Kg; "panca addominali": 2 serie da 20 ripetizioni). Ogni esercizio riguarda un gruppo specifico di muscoli.

Si disegni il modello concettuale del dominio descritto utilizzando il formalismo E/R.

□ Esercizio 1.8

Si consideri il sistema delle prenotazioni di posti in un teatro. Un cliente può prenotare uno o più posti per una rappresentazione; può anche prenotare posti per più di una rappresentazione. Per ciascuna prenotazione fatta, interessa sapere se il biglietto è già stato pagato oppure no.

Si disegni lo schema E/R che modella la porzione di realtà descritta.

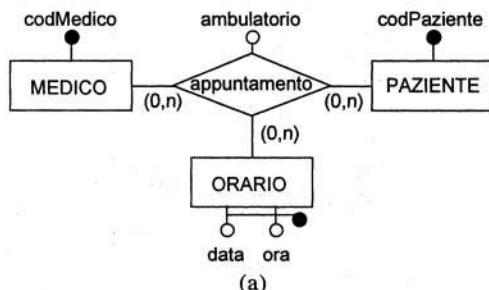
□ Esercizio 1.9

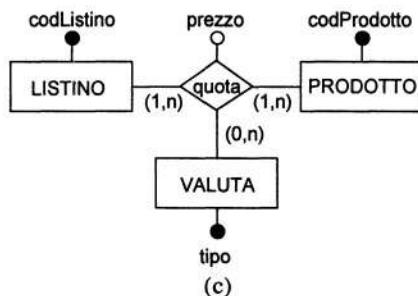
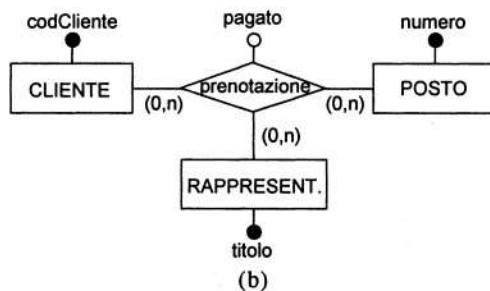
Si vuole automatizzare la gestione di un noleggio di compact disc e videocassetta. Per ogni compact disc si vuole memorizzare, oltre al titolo, all'autore e al genere musicale, anche il titolo e la durata di ogni canzone; esistono dischi doppi che possono essere noleggiati solo insieme. Non interessa memorizzare informazioni su autori di cui non sono presenti opere. Per ogni videocassetta si memorizzano il titolo, il regista, il genere, la durata e i due attori principali (se si tratta di un film). Si tenga presente che, sia per i compact disc sia per le videocassette, possono essere disponibili più copie dello stesso titolo. Per ogni copia interessa memorizzare la posizione nel magazzino (la codifica è unica per compact disc e videocassetta) e, in caso di noleggio, il numero di tessera del cliente (la tessera ha durata annuale) e la data di effettuazione del prestito. La durata di un prestito è di 3 giorni. Non interessa mantenere un archivio storico dei prestiti. A un cliente possono essere dati in prestito più titoli; un nuovo prestito viene rifiutato se risulta che il cliente non ha ancora restituito un prestito scaduto. Per ogni cliente si memorizzano nome, cognome, indirizzo e numero di documento. Ogni noleggio effettuato dà diritto a un "punto"; al raggiungimento di 20 punti, il cliente viene omaggiato di un noleggio.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.10

Si considerino i tre schemi E/R in figura, che utilizzano associazioni ternarie. Dalla conoscenza dei relativi domini applicativi si desumano, con l'aiuto del buon senso, le eventuali dipendenze funzionali non rappresentate negli schemi. Per ciascuno schema si disegni poi uno schema E/R alternativo che non utilizzi associazioni ternarie e che rappresenti al meglio le dipendenze funzionali individuate.





Si tenga presente che un medico può avere più appuntamenti con un paziente; un cliente può prenotare più posti per una rappresentazione; un prodotto può essere quotato in più listini in valute diverse.

□ Esercizio 1.11

Si vuole automatizzare la gestione delle stanze di un hotel. Ogni stanza è identificata dal suo numero, ed è descritta da una categoria (normale, lussuosa), dal numero di letti presenti (da uno a tre) e dal piano in cui è situata. Un cliente può richiedere di prenotare una stanza con un certo numero di letti e di una data categoria in un dato periodo; le stanze vengono assegnate anche senza prenotazione, a seconda della disponibilità. Quando un cliente inizia un soggiorno, viene aperto un conto su cui vengono registrati, oltre agli importi dovuti per l'uso della camera, eventuali extra (frigobar, uso della cassaforte, colazioni, ecc.). Per ogni cliente dell'hotel (inclusi gli occupanti dei letti supplementari per camere doppie e triple) si registrano i dati anagrafici. Alla partenza di un ospite, tutti i dati inerenti vengono cancellati.

Si disegni il modello concettuale del dominio descritto utilizzando il formalismo E/R.

Esercizio 1.12

Un insegnamento universitario è compreso nello statuto di uno o più corsi di laurea, e appartiene a uno specifico settore disciplinare (ad esempio, l'insegnamento Il appartiene al settore K1 sia per il corso di laurea C1 sia per C2). Un dato insegnamento per un dato corso di laurea (corso) si compone di uno o più moduli, ciascuno tenuto da un docente; tra i docenti che tengono i moduli di un corso, uno è il titolare del corso. Un docente può essere titolare al più di un corso. Ogni docente afferisce a un settore disciplinare, eventualmente diverso da quello del corso/i che tiene.

Si disegnino, utilizzando il formalismo E/R, due modelli concettuali del dominio descritto: uno di tipo *snapshot*, che “fotografi” la situazione della base di dati all’istante corrente) e uno in cui le informazioni relative alla titolarità dei corsi e alla docenza dei moduli siano storiche (uno stesso docente, in anni diversi, può essere titolare di corsi diversi e tenere moduli diversi).

Esercizio 1.13

Si consideri un sistema per la gestione di interventi su impianti. Un intervento è relativo a un impianto, a un impianto possono essere associati uno o più interventi. Un intervento è caratterizzato da una data e da un insieme di lavori effettuati ciascuno da una squadra con una macchina per un certo numero di ore. Una macchina può essere impiegata in più interventi da squadre diverse; per un intervento viene registrato il numero di ore di utilizzo, da parte di ciascuna squadra coinvolta, di ogni macchina impiegata. A seconda dell'impianto il costo orario di utilizzo di una macchina assume valori differenti.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

Esercizio 1.14

L'obiettivo è la gestione automatizzata del sistema di prenotazione di viaggi comitiva. L'agenzia organizza diverse gite, ciascuna per una specifica destinazione (per ciascuna destinazione non si effettua mai più di una partenza al giorno). Esistono varie tipologie di gite, ciascuna caratterizzata da un mezzo di trasporto utilizzato (aereo, nave, treno, pullman) e da un numero minimo di partecipanti. I clienti dell'agenzia sono privati e aziende.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

Esercizio 1.15

L'obiettivo è la gestione automatizzata delle richieste di prestazioni mediche. Il paziente esibisce una richiesta, compilata dal medico richiedente, per un insieme di prestazioni mediche. Le prestazioni sono amministrate da un tariffario, in cui il prezzo dipende dalla particolare convenzione a cui è soggetta la classe di utenza cui appartiene il paziente. Il calcolo del ticket complessivo deve tenere conto di eventuali fasce di esenzione previste dalla richiesta. L'erogazione di una prestazione può richiedere l'utilizzo di una particolare strumentazione medica. Si tenga presente che alcune prestazioni sono incompatibili tra loro. Due prestazioni incompatibili possono essere erogate allo stesso paziente solo se tra le due erogazioni intercorre almeno uno specifico intervallo di tempo. Alcune prestazioni, viceversa, richiedono che altre prestazioni siano eseguite prima (prestazioni propedeutiche), nel qual caso l'intervallo tra le erogazioni non deve superare un tempo massimo.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R. Si tracci poi lo schema di navigazione della query per il calcolo del ticket complessivo che il paziente deve corrispondere per una data richiesta.

Esercizio 1.16

Si consideri la seguente intervista, che si suppone rilasciata da un esperto del dominio applicativo:

La ARI è una associazione di ricercatori nel campo dell'informatica. Le conferenze ARI hanno lo scopo di richiamare da tutto il mondo esperti in un'area di ricerca scientifica.

Ciascuna conferenza è organizzata da un gruppo di lavoro ARI, ovvero un team di ricercatori (membri dell'ARI) che lavorano su un'area comune; quest'area diventa il tema della conferenza. Alla conferenza presiede un comitato di programma di cui fanno parte dieci associati, con ruoli diversi (presidente, segretario, revisori, ecc.).

Per pubblicizzare la conferenza viene spedito un dépliant (call for papers) a ricercatori di tutto il mondo; esso contiene, oltre alle date significative per la conferenza, un invito a sottomettere un articolo su uno dei temi pertinenti all'area di ricerca coperta dalla conferenza. Il sistema deve essere in grado di preparare automaticamente la lista per il mailing, includendovi solo i ricercatori che si occupano di temi pertinenti o comunque affini all'area di ricerca della conferenza.

Il programma stilato per la conferenza include articoli invitati e articoli sottomessi. Ciascun gruppo di lavoro ARI che si occupa della stessa area coperta

dalla conferenza viene invitato a scrivere un articolo; i lavori così ottenuti entrano automaticamente a far parte del programma. Gli articoli sottomessi, invece, sono soggetti a un processo di revisione a opera di membri del comitato di programma. Ciascun manoscritto viene spedito a tre revisori specializzati sul tema oggetto dell'articolo o su temi affini. Quando i revisori restituiscono il manoscritto allegano un giudizio; il presidente, sulla base dei tre giudizi, decide se accettare o meno l'articolo. Il sistema deve essere in grado di suggerire, per ogni articolo, tre nomi di revisori ritenuti idonei al compito; per ogni articolo deve poi spedire, all'autore principale, una lettera comunicante l'accettazione o il rifiuto del lavoro.

Il programma della conferenza, che deve essere stilato dal sistema raggruppando gli articoli sulla base del loro tema, viene spedito a tutti gli autori (compresi quelli il cui lavoro non è stato accettato). Al programma è allegato un modulo per l'iscrizione al convegno, che presenta tariffe differenziate per gli autori di lavori accettati e per i membri ARI. Sulla base dei moduli ricevuti, il sistema deve tenere la contabilità e stampare un cartellino di riconoscimento da consegnare a ciascun iscritto.

Si effettui una fase di filtraggio dei requisiti rivolta a eliminare dal testo eventuali sinonimie e omonimie di termini; a partire dai requisiti filtrati si disegni poi il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.17

Si vuole rappresentare la base dati utilizzata dalla motorizzazione. Esistono diversi produttori di auto, individuati da un nome unico (es. FIAT, FERRARI); ciascuno produce più modelli (es. UNO, TESTAROSSA), e per ogni modello esistono più esemplari individuati da un numero di serie unico per ciascun modello. I produttori possono vendere le auto solo ai rivenditori (individuati da un nome unico), i quali devono immatricolare le auto prima di venderle ai privati. Un'auto immatricolata ha un numero di matricola. Un privato può acquistare un'auto anche da un altro privato. Per ogni privato si memorizzano il nome e la data di nascita. Si vuole tenere traccia di tutti i passaggi di proprietà delle auto e della data in cui sono avvenuti.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.18

Un gruppo di 10 amici gioca a fantacalcio. Ogni giocatore possiede una fantasquadra, costituita da 25 calciatori di serie A. Ogni calciatore appartiene in realtà a una vera squadra di serie A; una fantasquadra può includere calciatori

appartenenti a squadre di serie A diverse, e un calciatore può appartenere a una sola fantasquadra. Ogni fantacampionato è composto da 18 giornate, ciascuna coincidente con una giornata del vero campionato; in ogni giornata vengono disputate 5 fantapartite tra coppie di fantasquadre. Per ogni fantapartita, i due giocatori titolari delle due fantasquadre coinvolte scelgono le formazioni da mandare in campo (11 calciatori tra i 25 a disposizione). Il risultato di ogni fantapartita è determinato dai voti assegnati ai calciatori dalla stampa a seguito delle vere partite disputate.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.19

I clienti del bagno “Graticola” possono prenotare in anticipo ombrelloni per l'estate. Ogni prenotazione può essere relativa a più ombrelloni ognuno dei quali può essere prenotato per un solo intervallo di tempo (all'interno di una prenotazione) specificato da data inizio e data fine. Per ogni ombrellone prenotato possono essere richiesti uno o più lettini. Ogni ombrellone appartiene a una zona (riva, centrale, bar, ...) dalla quale dipende la tariffa applicata.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.20

Si vuole realizzare un sistema di raccolta ed elaborazione dei dati relativi alle vendite del settore librario. Per la raccolta dati vengono utilizzate librerie distribuite sul territorio nazionale. In ogni libreria, un giorno prestabilito alla settimana, un rilevatore segna su un apposito modulo le informazioni sulle vendite di quel giorno della libreria. Per ogni rilevazione si vuole conoscere la data di effettuazione e il numero totale di copie vendute in quella data per ciascun libro o giornale. Inoltre, durante ogni rilevazione, il rilevatore intervista alcuni acquirenti scelti casualmente e ne registra sesso ed età oltre ai libri o giornali acquistati. Dei rilevatori si vuole conoscere Codice Fiscale, nome, indirizzo, telefono e fax. Ogni libro è caratterizzato da titolo, prezzo, genere e autori, per i quali si vogliono mantenere le informazioni sul nome, la nazionalità e il sesso. Un libro può essere stato stampato da più editori, caratterizzati dalla forma abbreviata della loro denominazione (es.: MON per Mondadori). Il prezzo di un libro dipende dall'edizione. Per i giornali si conosce il nome, la periodicità e la tiratura.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.21

Una casa d'aste memorizza informazioni relative alle opere d'arte e agli oggetti di antiquariato in vendita. Ogni oggetto d'antiquariato ha un nome, un codice identificativo, una datazione, una provenienza, un tipo di materiale, un prezzo di base, un ingombro e una foto. Ogni opera d'arte ha un nome, un codice, un autore, una datazione, una tipologia (pittura, scultura,...), un prezzo base, un ingombro e una foto. Le aste, che avvengono su internet, sono relative a un singolo oggetto o a un lotto di oggetti, hanno un prezzo base, un minimo rilancio, una data di inizio e una di fine asta. Ad ogni asta sono associati tutti i possibili rilanci degli utenti registrati e, se l'asta termina con una vendita, si registrano il prezzo, la tipologia di pagamento e la data di ricezione del pagamento. Allo scadere del tempo prestabilito un'asta viene chiusa e il migliore offerente ha 3 giorni di tempo per effettuare il pagamento che garantisce l'acquisto; in caso contrario l'oggetto può essere rimesso all'asta. Ogni cliente deve registrarsi (login, password, nome, cognome, e-mail) per effettuare delle offerte, mentre indirizzo e codice fiscale vengono registrati solo in conseguenza di un'effettiva vendita.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.22

Un Circolo Velico vuole realizzare il sistema informativo per la gestione delle anagrafiche dei soci, delle quote annuali e delle imbarcazioni. Tutti i soci del circolo hanno un tesserino in cui sono registrati il codice identificativo, i dati anagrafici (nome, cognome, data di nascita) e la data di annessione al circolo. Il tesserino viene rinnovato automaticamente ogni anno previo il pagamento della quota di iscrizione (della quale vengono registrati l'importo e l'avvenuto pagamento, senza necessità di storicizzare l'informazione). Nel sistema informativo vengono inoltre mantenuti i recapiti dei soci (indirizzo, telefono, ...) e dei relativi familiari (i quali però non possiedono il tesserino). I familiari vengono gestiti nell'anagrafica del sistema mediante un collegamento con il socio di riferimento, la loro quota di iscrizione viene quindi addebitata al referente. Ogni socio può possedere una o più imbarcazioni (ciascuna imbarcazione può avere più proprietari, tutti soci, per ciascuno dei quali viene registrata una quota). I dati registrati per ciascuna imbarcazione sono: matricola, marca e modello, nome e dimensioni (larghezza e lunghezza), alimentazione e potenza del motore se è una banca a motore. La quota annuale di ciascun socio prevede un importo fisso per ciascuno dei familiari e un importo variabile relativo al rimessaggio delle imbarcazioni di proprietà del socio. Per ciascuna imbarcazione, il costo di rimessaggio viene calcolato sulla base di un tariffario che stabilisce i prezzi a seconda delle dimensioni (il tariffario deve essere registrato nel sistema, ma non storicizzato).

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.23

Gli organizzatori della gara cicloturistica annuale Nove Colli vogliono creare un sistema informativo per la gestione della corsa. Per ogni edizione della Nove Colli si vogliono memorizzare la data di svolgimento, una descrizione dei percorsi (uno di 130 km e uno di 200 km) e il numero totale dei partecipanti. La descrizione dei percorsi è costituita dall'elenco dei punti di "controllo" e relative distanze (es. Cesena 20 Km, MercatoSaraceno 32 Km, Colle Barbotto 15 km,...). I corridori al momento dell'iscrizione forniscono le generalità (nome, cognome, data di nascita, indirizzo, codice fiscale), il codice del tesserino e il gruppo ciclistico al quale sono affiliati (entrambi obbligatori per partecipare); indicano inoltre la categoria di appartenenza (ad es. cicloamatore, elite, senior, ...) e il tipo di percorso prescelto (130 km o 200 km) e ricevono un numero di maglia (univoco per ogni edizione). La gestione della corsa prevede la registrazione dei tempi in tutti i punti di controllo e della posizione di classifica assoluta e di categoria di tutti gli iscritti al fine di poter stampare le classifiche singole e per gruppi sportivi di ciascuna gara e di ciascuna categoria. Viene inoltre stilata una particolare classifica per il migliore scalatore realizzata in base al tempo di scalata del "Bardotto".

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.24

La FederVolley vuole realizzare il sistema informativo per la gestione dell'XI campionato italiano assoluto di Beach Volley maschile. Il campionato è composto da 10 tappe (ciascuna corrispondente a un singolo torneo) che si svolgono in diverse località italiane. Ciascuna tappa è caratterizzata da una data inizio, data fine, nome località, numero progressivo (I, II, III,...) e montepremi. A ciascuna tappa si possono iscrivere fino a 48 coppie di giocatori, che entrano a far parte del torneo. Di ciascun giocatore sono memorizzati i dati anagrafici, il numero di tesserino FIPAV, il recapito, il punteggio totale acquisito nelle tappe precedenti, il montepremi vinto e la posizione in classifica. Ciascun giocatore può cambiare compagno nei diversi tornei, ma non durante le partite di uno stesso torneo. Di tutte le partite di ciascun torneo viene registrato un numero progressivo, il livello (quarti, semifinale..), la durata, le coppie sfidanti e il risultato (es. 21-14, 19-21, 15-13) dei set giocati (due o tre per ogni partita). A seguito di ogni torneo la posizione di ciascun giocatore viene aggiornata in base al punteggio e al montepremi acquisito.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.25

Il Club Amici del Borgo mantiene nel proprio database informazioni sulle sagre di paese e fornisce delle recensioni su tali sagre compilate dai collaboratori del Club (di cui sono schedati i dati anagrafici). Per ogni sagra vengono mantenute le informazioni relative al nome, alla data inizio e alla durata. Relativamente al comune ospitante si memorizzano nome, provincia, CAP, istruzioni per raggiungerlo; ciascun comune può ospitare più di una sagra, ma in diversi periodi dell'anno. Le recensioni forniscono una valutazione in base ad alcuni parametri (Musica, Animazione, Servizio, Rapporto Qualità/Prezzo, ecc): a ogni parametro è associata una scala di valori numerici. Sono memorizzati sia i voti e il commento dei singoli recensori che i punteggi finali (ottenuti dalla media dei voti dei diversi recensori).

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.26

La redazione multimediale NOTWeb (News On The Web) gestisce la pubblicazione di un giornale on-line. Le pagine, pubblicate quotidianamente, sono numerate e divise in sezioni (prima pagina, interni, esteri,). Di ogni pagina viene memorizzato il file in formato pdf e i singoli oggetti che lo compongono. Gli oggetti pubblicati si dividono in articoli, immagini, annunci e banner pubblicitari. Ciascun oggetto è caratterizzato da una data (giorno di pubblicazione) e da una posizione (pagina e coordinate interne alla pagina). Gli articoli hanno un titolo, un luogo (opzionale, in genere è la città alla quale si riferisce il servizio) e un autore. Le immagini hanno un titolo, un autore (se si tratta di una vignetta), una didascalia ed eventualmente un articolo collegato se si tratta di fotografie. Gli annunci hanno un titolo, un testo e un riferimento alfanumerico (tale codice corrisponde all'autore dell'annuncio di cui la redazione conserva i dati anagrafici). I banner pubblicitari hanno un testo formattato o un'immagine e una ditta pubblicitaria responsabile. Tutti gli articoli che fanno riferimento alla stessa notizia (anche pubblicati in date diverse) vengono collegati fra loro per facilitarne la ricerca. La presentazione dei dati su web permette la consultazione diretta delle informazioni del giorno e la ricerca negli archivi in base a data, testo libero, argomento della sezione e articoli correlati.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.27

Una ditta per l'imbottigliamento e la distribuzione di vini interagisce con clienti e con produttori. I produttori, caratterizzati da un nome, indirizzo e tipologia del vigneto, producono ogni anno una certa quantità di vino. Il vino, caratterizzato da una denominazione e un'annata, viene normalmente imbottigliato; in alcuni casi, la ditta produce anche vino in damigiane. I clienti acquisiscono le bottiglie o le damigiane per corrispondenza; ciascun ordine comprende un numero arbitrario di bottiglie oppure damigiane di vini differenti. Ogni ordine è associato ad un ammontare complessivo e a una modalità di pagamento. I clienti hanno un indirizzo e talvolta un numero di carta di credito. Ciascun ordine emesso da un cliente che non paga tramite carta di credito deve essere saldato entro 30 giorni tramite vaglia postale inviato assieme alla merce; se, trascorsi i 30 giorni, l'ordine non è saldato, viene inviato un sollecito, e contemporaneamente il cliente viene caratterizzato come "in mora". Se dopo altri 30 giorni il sollecito non è saldato, il nominativo del cliente è segnalato ad un'azienda di recupero dei crediti.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.28

La redazione della trasmissione "Il grande fratello" vuole realizzare un sistema informativo per la memorizzazione di tutti i dati e gli eventi relativi alla permanenza dei concorrenti nella casa. Si vogliono mantenere le informazioni personali relative ai concorrenti e quelle relative alla loro permanenza (data di ingresso, eventuale data di uscita). Devono essere modellate anche le attività settimanali: ogni settimana i concorrenti hanno a disposizione un budget che è costituito da un ammontare iniziale più o meno la percentuale scommessa durante la prova settimanale (nome, descrizione) che può avere esito positivo o negativo e determina la scelta di uno o più concorrenti migliori e peggiori. Per i migliori di ogni settimana si devono registrare le scelte relative ai premi (costo e tipo di premio scelto). Devono infine essere gestire le nomination: ogni settimana (oppure ogni 2) ciascun concorrente nomina due suoi compagni; i concorrenti (2 o più) che hanno avuto il maggior numero di voti sono candidati all'eliminazione stabilita in base ai voti (memorizzati in percentuale) dei telespettatori.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.29

La segreteria della facoltà vuole memorizzare informazioni sui Consigli di Facoltà secondo le seguenti specifiche. Un Consiglio di Facoltà (CdF) è descritto dalla

data (univoca) dall'ora e dalla sede. Ad un CdF vengono convocati, come componenti, i docenti della Facoltà, i rappresentanti dei ricercatori e degli studenti. Ognuno di questi componenti, descritto dagli usuali dati anagrafici, può quindi partecipare al CdF, o giustificare la propria assenza riportando il motivo dell'assenza, oppure risultare assente ingiustificato. Per ogni CdF viene nominato un segretario, scelto tra i professori ordinari. Per un CdF viene riportato l'Ordine del Giorno (OdG) costituito da una serie di punti, per ciascuno dei quali viene riportata una descrizione. Un certo punto dell'OdG può far riferimento a uno o più allegati; un allegato è descritto da un numero univoco e da un testo e a esso possono fare riferimento uno o più punti dell'OdG. Per ogni punto dell'OdG si possono indicare uno o più relatori tra i componenti presenti al CdF: durante un certo CdF un componente non può relazionare più di cinque punti dell'OdG.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.30

Si vuole progettare un sistema informativo per la gestione di informazioni di transazioni di compravendita e affitto immobili effettuate da una catena di agenzie immobiliari affiliate alla InfoCasa. Ciascuna agenzia è caratterizzata da un numero identificativo, il nome del titolare, l'indirizzo, da 1 a 3 recapiti telefonici, l'e-mail ed eventualmente l'URL della pagina web. Una transazione immobiliare è relativa alla vendita o all'affitto di un solo immobile a una o più persone, è caratterizzata da un ammontare totale e da una percentuale di anticipo, ed è gestita da una sola agenzia. Per gli affitti interessa anche il periodo di affitto. Gli immobili (con codice, indirizzo, città di ubicazione, metri quadrati e numero locali) oggetto delle transazioni sono di proprietà di una o più persone, in percentuali diverse. A seguito della vendita di un immobile il proprietario dell'immobile viene modificato, e i vecchi proprietari rimangono memorizzati nel DB solo come parte della transazione di vendita. Alcuni immobili sono di interesse storico, e di essi interessa l'anno di costruzione. Alcuni sono ristrutturati, e di essi interessa la data di ultima ristrutturazione.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

□ Esercizio 1.31

Si vuole modellare la base dati topografica di una città. L'archivio deve contenere informazioni sugli elementi topografici della città, come le strade e le piazze (nome, per entrambi gli elementi, e lunghezza solo per le strade). Di ogni elemento sono inoltre note le coordinate all'interno della mappa (es Piazza Dante settore A5, viale Carducci settori B1, B2, B3 ecc.) Una piazza è caratterizzata da una

forma e da un'estensione. Può contenere uno o più monumenti, di ciascuno dei quali è noto il personaggio (o il fatto) che rappresenta, la data di costruzione, l'artista che lo ha prodotto, il materiale di cui è fatto e la posizione. In una piazza può esserci un giardino pubblico, di cui si sa il nome e l'estensione in metri quadri. Di una strada si sa quali altre strade incrocia (un incrocio può coinvolgere due o più strade ed eventualmente una piazza) e in quali piazze (eventualmente) sfocia. Gli incroci possono essere controllati o meno da semafori. Le strade possono avere dei tratti a doppio senso di marcia, a senso unico, a circolazione limitata o a isola pedonale: in questo caso è necessario indicare gli incroci (o le piazze) di inizio e fine del tratto di strada e la tipologia di limitazione della circolazione. Sia in una strada che in una piazza possono esserci edifici pubblici, dei quali si conoscono il nome, la destinazione (es: municipio, piscina comunale, ...) e il numero civico.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

Esercizio 1.32

Si vuole realizzare un sistema informativo per gestire la vendita di biglietti per gli spettacoli di Broadway. Ogni spettacolo ha un codice univoco, una durata, un titolo, un autore e un regista. Le rappresentazioni di uno spettacolo si svolgono tutte nello stesso teatro, in più periodi di tempo disgiunti; ciascuno spettacolo viene rappresentato a orari fissati nei giorni di rappresentazione (gli orari possono variare nei diversi giorni della settimana o per i giorni festivi); ad es. lo spettacolo Cats si svolge nel periodo dal 01/01/2005 al 15/02/2005 il sabato alle 21:00 e la domenica alle 15:00 e alle 21:00, mentre nel periodo dal 10/03/2005 al 25/04/2005 si svolge il venerdì alle 20:30, il sabato alle 21:00 e la domenica alle 15:00. Ciascun teatro, identificato da un nome, ha un indirizzo, un numero di telefono e una pianta della sede in forma di insieme di settori, ognuno dei quali ha associata una matrice che rappresenta i posti del settore. Ad ogni settore è associato un costo (dipendente dallo spettacolo) e ogni posto (elemento della matrice associata al settore) può essere libero, prenotato o venduto. Per ogni rappresentazione è possibile prenotare un posto libero e, successivamente, acquistare i relativi biglietti. Il sistema deve mantenere uno storico di tutti gli spettacoli, le rappresentazioni e le vendite dei biglietti. Non è necessario mantenere un archivio dei clienti, che non vengono registrati per l'acquisto dei biglietti, ma lasciano semplicemente il nome per prenotare.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

Esercizio 1.33

Si vuole progettare un sistema informativo per la gestione del personale di un villaggio vacanze. Il personale del villaggio (di cui sono mantenuti i dati anagrafici e retributivi: nome, cognome, data assunzione, stipendio, ...) si divide in "temporaneo" e "fisso". I temporanei lavorano per il villaggio per periodi di tempo determinato (es. stagione estiva di 6 mesi), non godono di ferie, ma hanno un giorno di riposo settimanale e alcuni giorni di permesso (il cui numero dipende dalla tipologia e dalla durata del contratto). I "fissi" hanno invece un contratto a tempo indeterminato che prevede oltre al giorno di riposo anche 30 giorni di ferie all'anno. Inoltre il personale si distingue in base al ruolo svolto nel villaggio (animatore, cuoco, istruttore sportivo,...). Il sistema deve registrare il giorno di riposo, il numero di giorni di permesso utilizzati dal personale temporaneo e i periodi di ferie utilizzati dal personale fisso.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

Esercizio 1.34

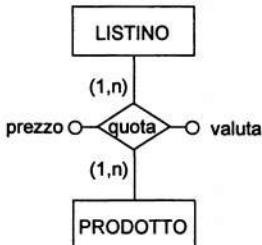
Una società multinazionale che produce abbigliamento sportivo ha sedi in diversi paesi. Le sedi sono identificate in modo univoco da un codice e sono caratterizzate da un indirizzo, da un numero di telefono e dal nominativo del responsabile. Le sedi sono suddivise in magazzini, centri di produzione e sedi di vendita al minuto. I magazzini, caratterizzati da un loro ulteriore codice e dal numero di addetti, riforniscono i negozi in franchising (dunque non di proprietà della società). I negozi in franchising si riforniscono unicamente dal magazzino più vicino. I centri di produzione sono caratterizzati dal numero di addetti e dalla capacità produttiva (in unità al giorno) relativa ai singoli prodotti. Le sedi di vendita al minuto sono caratterizzate da un fatturato medio e da un elenco dei prodotti disponibili a magazzino, con relative giacenze. I negozi e le sedi di vendita al minuto vendono i loro prodotti ai clienti (che sono identificati da un codice) senza alcun rapporto esclusivo. Per ogni transazione di vendita (sia al dettaglio che tra magazzini e negozi in franchising) devono essere registrate i prodotti venduti (e relative quantità), la data di vendita e il totale della transazione.

Si disegni il modello concettuale del dominio descritto, utilizzando il formalismo E/R.

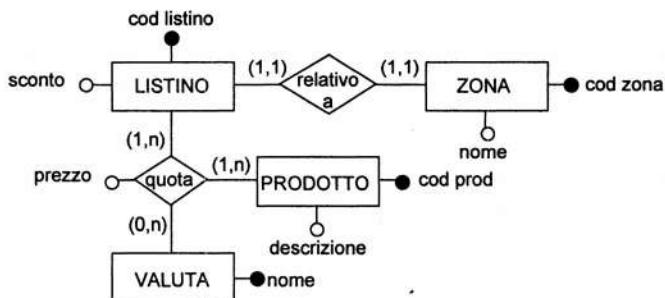
✓ Esercizio 1.1

Le specifiche permettono agevolmente di individuare due entità: listino e prodotto. Anche per il concetto di zona è conveniente modellare un'entità, così da poter rappresentare eventuali informazioni abbinate (nome del rappresentante, estensione, clienti, ecc.). Per quanto riguarda lo sconto, invece, in assenza di informazioni sull'esistenza di particolari tipologie di sconto definite a priori, si preferisce modellare un attributo di listino.

Il punto focale dell'esercizio è la rappresentazione della relazione esistente tra listini, prodotti e valute. Se ogni prodotto fosse quotato in ciascun listino in una sola valuta, la modellazione potrebbe utilizzare una associazione binaria con attributi prezzo e valuta:



Poiché invece un prodotto può essere quotato in uno stesso listino in valute diverse, si deve ricorrere a un'associazione ternaria; infatti, poiché un attributo di un'associazione non può essere utilizzato come identificatore, non ci sarebbe altrimenti modo di discriminare le diverse accoppiate listino-prodotto. Si passa quindi alla rappresentazione in figura, dove la valuta viene modellata come un'entità e il prezzo come attributo di un'associazione ternaria.

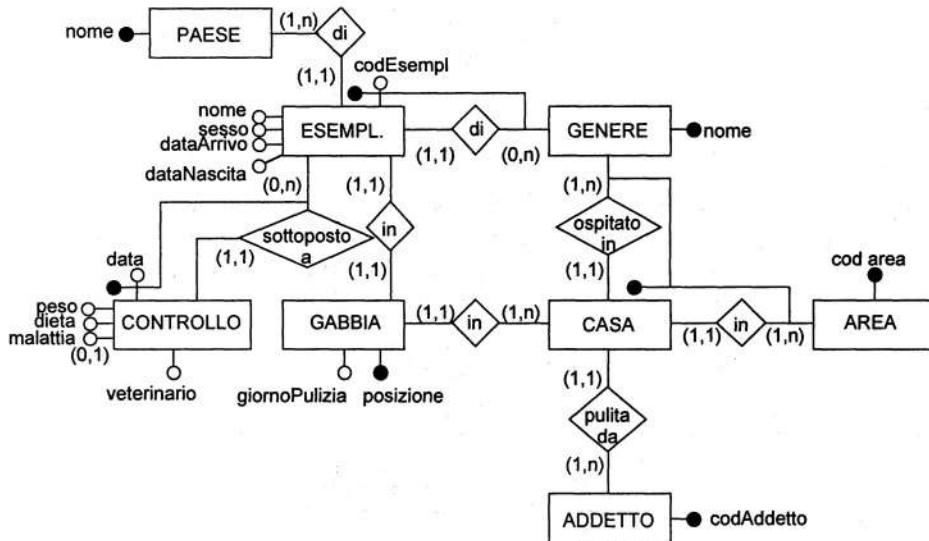


 Suggerimento:

Si modifichi lo schema considerando il caso in cui a ogni listino sia associato un insieme di sconti, ciascuno valido in un dato periodo temporale e calcolato come percentuale sul prezzo indipendentemente dalla valuta e dal prodotto.

✓ Esercizio 1.2

La scelta degli identificatori per le entità ESEMPLARE, CONTROLLO e CASA di questo schema merita un approfondimento. Per ESEMPLARE, le specifiche definiscono chiaramente l'identificatore come misto: infatti il codice che identifica un esemplare è unico soltanto all'interno del genere di appartenenza, quindi l'identificatore deve essere costruito abbinando a tale codice l'identificatore del genere di appartenenza. Per quanto riguarda CONTROLLO, un possibile identificatore misto nasce dall'abbinamento tra l'identificatore dell'esemplare e la data in cui il controllo è stato effettuato, ovviamente nell'ipotesi che uno stesso esemplare non subisca più controlli in uno stesso giorno (in caso contrario conviene introdurre un attributo codice controllo univoco). Infine, un possibile identificatore di CASA è definito osservando che, come stabilito dalle specifiche, ogni casa di un'area è destinata a un diverso genere di animali: la casa può quindi essere univocamente individuata da un identificatore esterno comprendente l'identificatore di AREA e quello di GENERE.



Si noti che lo schema presenta un ciclo tra le entità ESEMPLARE, GENERE, GABBIA e CASA. Questo ciclo è ridondante: infatti, eliminando l'associazione appartiene a non si ha perdita di informazione poiché è comunque possibile determinare il genere di qualunque esemplare percorrendo in sequenza le associazioni rinchiuso in, in, e ospitato in. L'associazione appartiene a non può però essere eliminata, essendo indispensabile per la definizione dell'identificatore misto di ESEMPLARE.

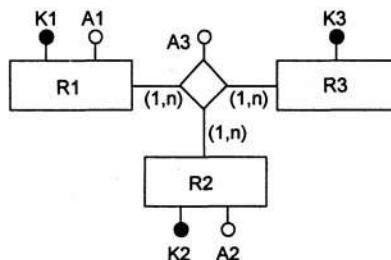
✓ Esercizio 1.3

Le relazioni R1 e R2 provengono ovviamente da due entità con identificatori, rispettivamente, K1 e K2. Per la relazione R3, la presenza della chiave composta suggerisce la provenienza da un'associazione ternaria tra R1, R2 e una terza entità R3 con identificatore K3 e senza altri attributi. A3 proviene necessariamente da un attributo dell'associazione; se infatti nello schema E/R fosse stato attributo di R3, lo schema relazionale risultante dal progetto logico sarebbe stato:

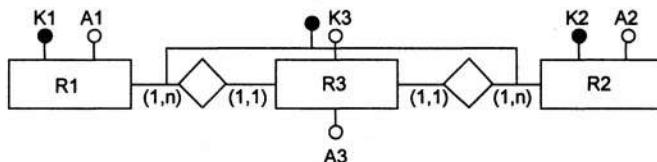
R1 (K1, A1)
R2 (K2, A2)
R3 (K3, A3)
T (K1, K2, K3)

Si noti che la cardinalità minima della ternaria dal lato R3 deve necessariamente essere 1: se infatti ci fosse opzionalità, lo schema relazionale dovrebbe includere 4 relazioni:

R1 (K1, A1)
R2 (K2, A2)
R3 (K3)
R3 (K1, K2, K3, A3)



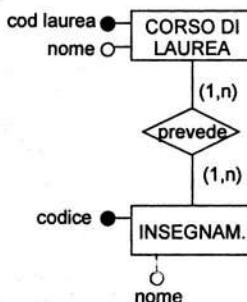
La seconda soluzione possibile utilizza due associazioni binarie equivalenti alla ternaria: in questo caso l'identificatore di R3 è misto e coinvolge, oltre all'attributo K3, gli identificatori di R1 e R2.



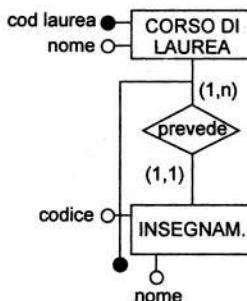
✓ Esercizio 1.4

Il nucleo di questo esercizio consiste nella definizione dell'entità INSEGNAMENTO. Per raggiungere la soluzione finale procediamo considerando in modo graduale i vari requisiti. Cominciamo rappresentando il fatto che “*Ogni insegnamento è*

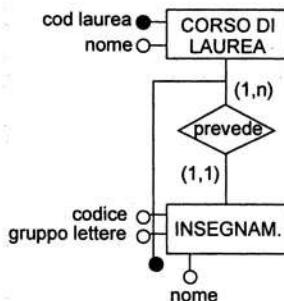
contraddistinto da un codice e un nome. La maggior parte degli insegnamenti appartiene al piano di studi di più corsi di laurea: Secondo la soluzione rappresentata nella figura seguente, le istanze dell'entità INSEGNAMENTO sono "Analisi I", "Analisi II", ecc.



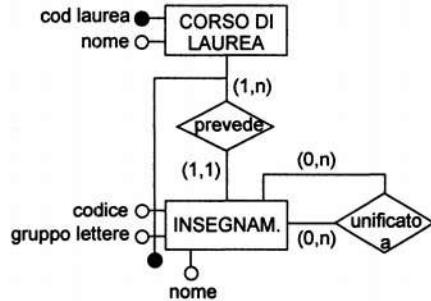
In questo schema, l'entità INSEGNAMENTO è troppo generica per poter essere utile; modificando l'identificatore come nella figura successiva, rendiamo più specifico il significato di INSEGNAMENTO: le sue istanze sono ora "Analisi I per Ing. Elettronica", "Analisi I per Ing. Meccanica", "Analisi II per Ing. Elettronica", ecc.



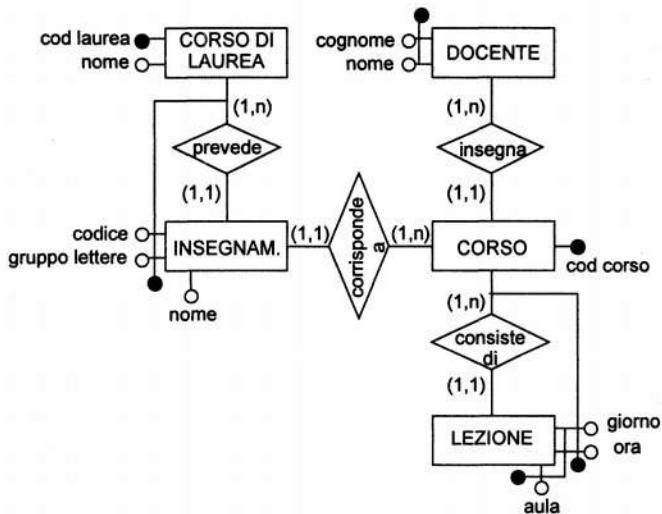
Occorre ora considerare che "*Gli insegnamenti con molti studenti vengono suddivisi raggruppando alfabeticamente gli studenti*". Si può tenere conto di questa specifica semplicemente aggiungendo nell'identificatore misto di INSEGNAMENTO un attributo **gruppo lettere** (che vale A-Z per i corsi non suddivisi): le istanze di INSEGNAMENTO sono ora "Analisi I per Ing. Elettronica A-K", "Analisi I per Ing. Elettronica L-Z", "Analisi I per Ing. Meccanica A-Z".



Per tenere conto dell'ultimo requisito, "Le lezioni degli insegnamenti con pochi studenti vengono unificate per più corsi di laurea", si può procedere in due modi. Il primo, raffigurato sotto, esprime l'unificazione tra insegnamenti tramite un'associazione unaria, e associa i docenti e gli orari delle lezioni direttamente all'entità INSEGNAMENTO; si ha quindi ridondanza, poiché docente e lezioni vengono replicati per ogni insegnamento unificato.



La soluzione proposta sotto, migliore in quanto non ridondante, prevede l'introduzione di una nuova entità CORSO che rappresenta con una istanza ogni insegnamento effettivamente tenuto da un docente durante un orario delle lezioni. Il legame tra INSEGNAMENTO e CORSO è sancito dall'associazione 1-N corrisponde, che modella l'unificazione tra corsi.



Per quanto riguarda il legame tra corsi e lezioni, si noti che esistono due dipendenze funzionali tra corsi, orari e aule. La prima,

giorno, ora, aula → corso

è modellata implicitamente dall'identificatore interno di LEZIONE e dalla cardinalità dell'associazione **consiste di**.

La seconda,

corso, giorno, ora → aula

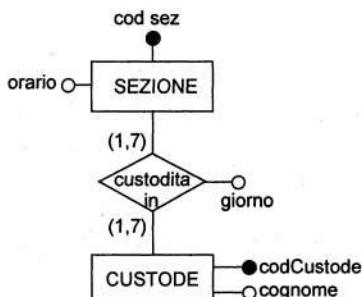
è stata modellata definendo per LEZIONE anche un identificatore misto.

☞ Suggerimento:

Modificare lo schema E/R per prevedere che un insegnamento possa esistere ma non essere ancora attivato in alcun corso di laurea.

✓ Esercizio 1.5

L'aspetto più rilevante di questo esercizio è la rappresentazione dei turni effettuati dai custodi per le varie sezioni. A tale proposito si consideri la porzione di schema nella figura seguente.



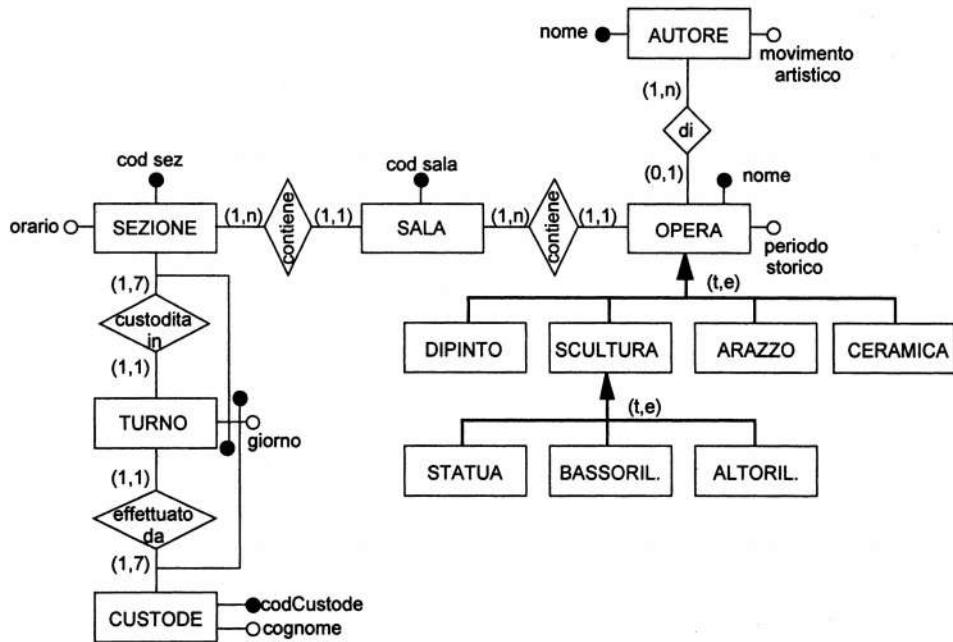
Poiché l'attributo giorno sull'associazione custodita da non può fungere da identificatore, questa soluzione può considerarsi corretta solo nel caso in cui, durante la settimana, un custode non possa custodire più di una volta la stessa sezione.

Questo vincolo non compare tra i requisiti dell'esercizio, per cui occorre mettere a punto una soluzione più generale. Ciò può essere fatto introducendo un'entità TURNO come nello schema in figura successiva. Il problema degli autori sconosciuti è risolto rendendo l'associazione di opzionale dal lato di OPERA. I due identificatori misti di TURNO sono utili per esprimere le due dipendenze funzionali esistenti:

sezione, giorno → custode
custode, giorno → sezione

☞ Suggerimento:

Si modelli il caso in cui, durante una stessa giornata, più custodi si possano avvicendare in una stessa sezione cosicché uno stesso custode possa custodire più sezioni.



✓ Esercizio 1.6

Il ciclo in figura è ridondante se la sistemazione dei computer nelle sale dei laboratori viene effettuata nel rispetto dei vincoli di assegnamento di ciascun laboratorio a una sola marca. Per confermare la ridondanza occorre dimostrare che è possibile mantenere intatto il contenuto informativo del database pur eliminando una delle associazioni che lo costituiscono. Consideriamo le diverse possibilità:

- Eliminazione di comprende. In assenza di comprende non sarebbe possibile, data una sala, risalire al laboratorio che la contiene. Infatti,

sala → marca

ma

marca → laboratorio

- Eliminazione di contiene. In assenza di contiene non sarebbe possibile, dato un computer, risalire alla sala che lo contiene. Infatti,

computer → marca

ma

marca → laboratorio
laboratorio → sala

- Eliminazione di riservato a. In assenza di riservato a è possibile determinare a quale marca è riservato un laboratorio solo nell'ipotesi che, nel database, tutti i computer di tutte le sale di ciascun laboratorio risultino essere effettivamente della stessa marca.
- Eliminazione di di. Questa è la scelta più corretta. Infatti, anche in assenza di di, la marca di un computer può essere determinata univocamente in virtù delle dipendenze funzionali esistenti:

computer → sala
 sala → laboratorio
 laboratorio → marca

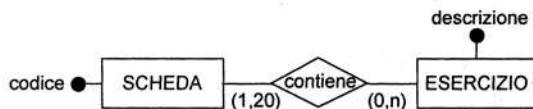
per cui

computer → marca

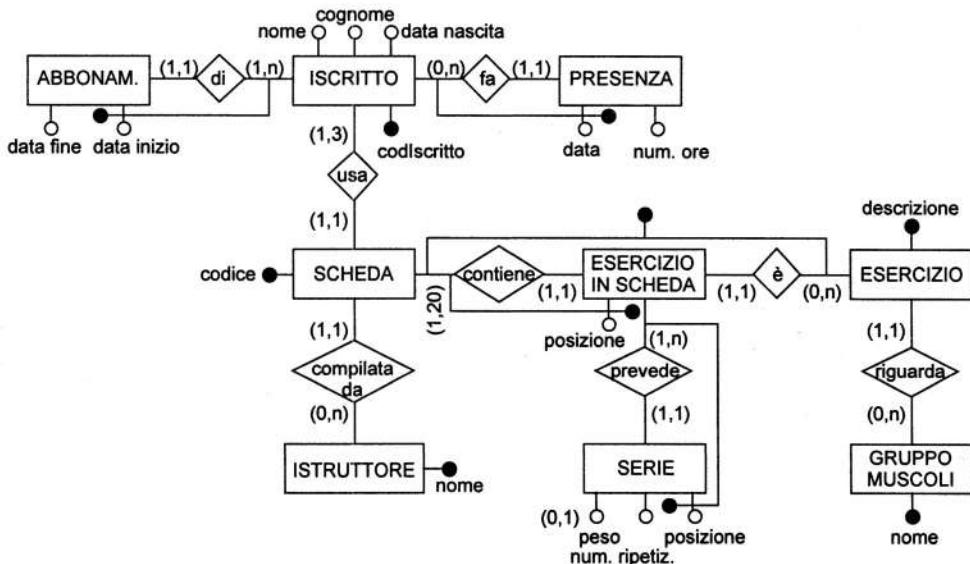
✓ Esercizio 1.7

Anche per questo esercizio sono necessari alcuni commenti sulla scelta degli identificatori. Per quanto riguarda gli iscritti, abbiamo ipotizzato che l'identificazione avvenga sulla base dei dati anagrafici; supponendo che gli abbonamenti vengano mantenuti in memoria anche dopo la loro scadenza, l'identificazione dell'abbonamento può avvenire abbinando la sua data di inizio all'identificatore dell'iscritto (un iscritto non ha più abbonamenti contemporanei). L'identificatore misto per PRESENZA è stato scelto assumendo che non interessa mantenere distinte eventuali presenze multiple di un iscritto nello stesso giorno.

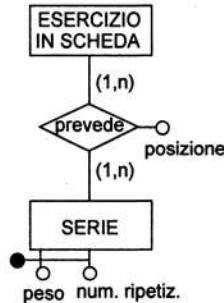
Per modellare il legame tra SCHEDA e ESERCIZIO, sarebbe stata sufficiente una semplice associazione binaria N-M qualora non fosse stata richiesta la rappresentazione delle informazioni sulle serie:



Poiché si deve invece rappresentare il numero di ripetizioni e l'eventuale peso, occorre aggiungere un'entità ESERCIZIO IN SCHEDA a cui collegare le informazioni sulle serie. Questa entità è ovviamente identificata assegnando una scheda e un esercizio (identificatore esterno); esiste però un identificatore alternativo (misto) costruito abbinando l'identificatore della scheda e la posizione all'interno della scheda.



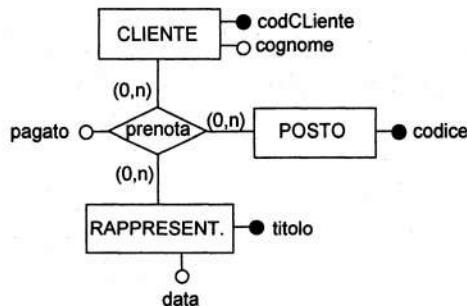
La soluzione scelta per la rappresentazione delle serie è analoga. In alternativa, se per ogni serie fosse definito un peso, potremmo intraprendere la soluzione in figura (un attributo può far parte di un identificatore solo se obbligatorio).



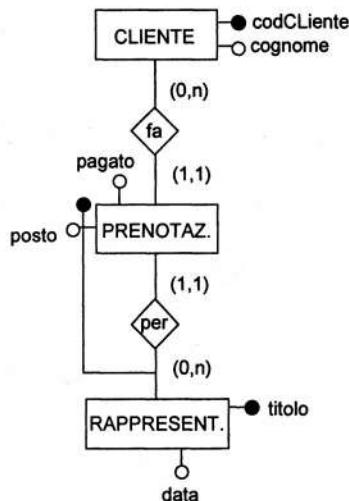
✓ Esercizio 1.8

Esaminiamo due possibili soluzioni a questo esercizio. Nella prima, rappresentata di seguito, la prenotazione è modellata tramite una associazione ternaria tra CLIENTE, RAPPRESENTAZIONE e POSTO, di cui pagato è un attributo. L'inconveniente di questa soluzione è che non viene in alcun modo rappresentata la dipendenza funzionale che afferma che, per una data rappresentazione, un dato posto può avere al più una prenotazione:

rappresentazione, posto → cliente



La seconda soluzione, rappresentata nella figura successiva, modella esplicitamente la prenotazione tramite un'entità ed è preferibile poiché consente, tramite la definizione dell'identificatore misto, di rappresentare la dipendenza funzionale sopra descritta.



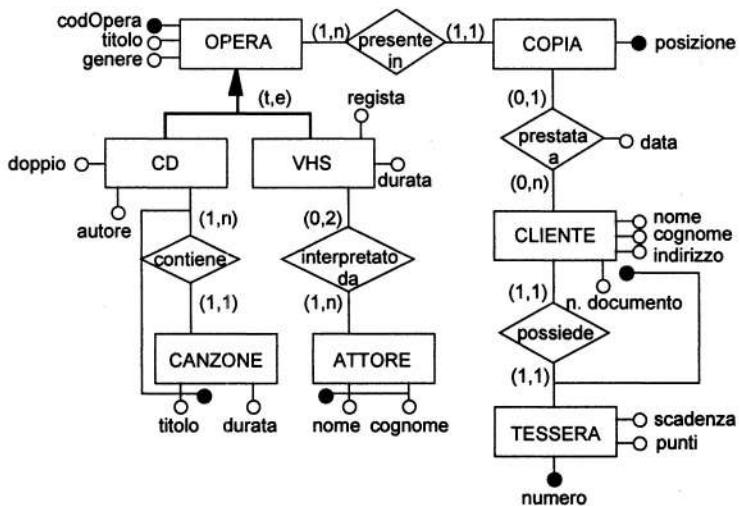
Suggerimento:

- Si modifichi lo schema E/R per tenere conto del fatto che il teatro è costituito da:
- una platea, suddivisa in 4 settori (A, B, C, D) di 200 posti ciascuno ripartiti in 10 file di 20 posti;
 - una galleria (settore E) di 100 posti ripartiti in 4 file di 25 posti.

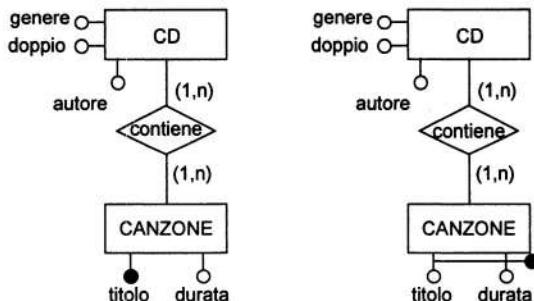
✓ Esercizio 1.9

Per risolvere questo esercizio occorre prestare attenzione alla distinzione tra opere (su compact disc e su videocassetta) da un lato, e loro copie dall'altro (il termine copia è qui inteso come *esemplare*). Entrambi i concetti devono essere modellati: per l'opera si rappresenta la descrizione generale (titolo, genere, autore o regista, ecc.);

per la copia si rappresentano la collocazione in magazzino e l'eventuale prestito. Utilizzando un'unica entità si avrebbe replicazione di dati, poiché la descrizione generale di ogni opera verrebbe ripetuta per ciascuna copia.



Nella definizione dell'identificatore misto per CANZONE, si è ipotizzato che un compact non contenga mai due canzoni con titolo identico, e sia invece possibile avere canzoni con lo stesso titolo, ed eventualmente durata diversa, su compact diversi. Vediamo due soluzioni alternative: nella prima si mette in evidenza il fatto che una stessa canzone può comparire in più compact, ma si nega l'esistenza di versioni diverse della stessa canzone o di canzoni diverse con lo stesso titolo (infatti il titolo determina la durata). Nella seconda si estende l'identificatore a comprendere anche la durata; sono quindi possibili durate diverse per lo stesso titolo, e uno stesso titolo può comparire più volte sullo stesso compact se con durate diverse.



Suggerimento:

Si modifichi lo schema E/R nel caso in cui sia necessario mantenere memorizzati i prestiti anche dopo la loro estinzione.

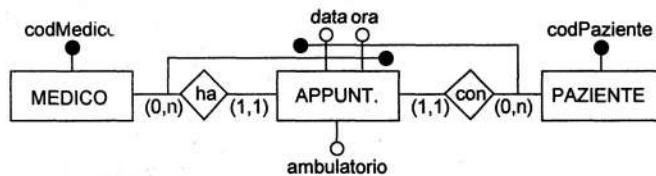
✓ Esercizio 1.10

a) Poiché né un medico né un paziente possono avere due appuntamenti contemporaneamente, nel primo schema esistono due dipendenze funzionali:

MEDICO, ORARIO → PAZIENTE

PAZIENTE, ORARIO → MEDICO

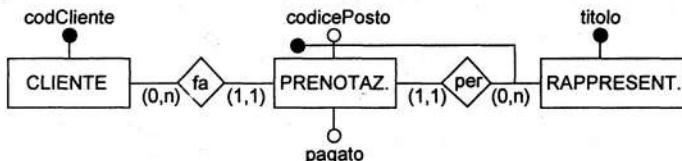
È quindi possibile rappresentare lo schema come mostrato nella figura successiva. Si noti che questa rappresentazione è più espressiva di quella che utilizza l'associazione ternaria: infatti, il doppio identificatore dell'entità APPUNTAMENTO permette di cogliere le due dipendenze funzionali esistenti.



b) Poiché uno stesso posto non può essere prenotato due volte per la stessa rappresentazione, nello schema esiste una dipendenza funzionale:

RAPPRESENTAZIONE, POSTO → CLIENTE

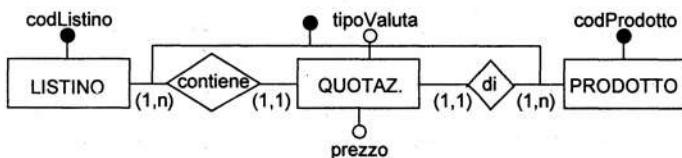
È quindi possibile rappresentare lo schema come mostrato in figura. Anche in questo caso la rappresentazione con associazioni binarie è più espressiva di quella che utilizza la ternaria: infatti, l'identificatore dell'entità PRENOTAZIONE permette di cogliere la dipendenza funzionale esistente.



c) In questo schema non esistono dipendenze funzionali oltre a quella, banale,

LISTINO, PRODOTTO, VALUTA → prezzo

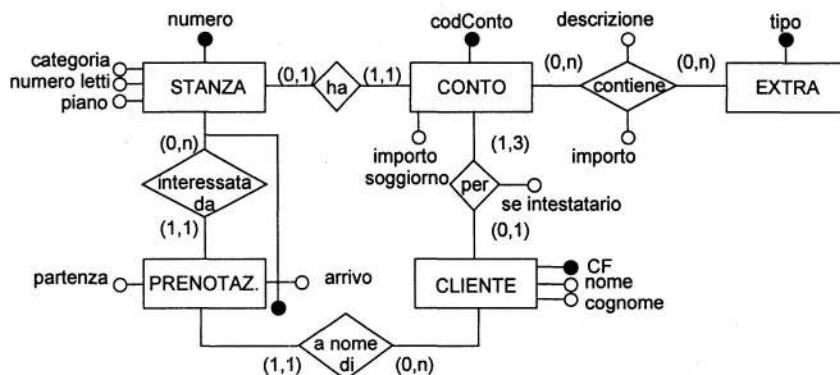
Lo schema mostrato in figura è quindi del tutto equivalente a quello che utilizza l'associazione ternaria.



✓ Esercizio 1.11

Un possibile schema E/R è mostrato in figura.

Ogni prenotazione è rappresentata da una istanza dell'entità PRENOTAZIONE, il cui identificatore è dato dalla coppia stanza-data di arrivo. Per quanto riguarda i soggiorni si mantiene solo, come richiesto dalle specifiche, uno *snapshot* dei conti; per questo motivo una stanza può avere, all'istante corrente, al più un conto. Ogni conto viene collegato ai clienti che alloggiano nella stanza, uno dei quali sarà indicato come l'intestatario del conto. Il calcolo totale del conto può essere effettuato sommando l'importo del soggiorno a quello degli extra.



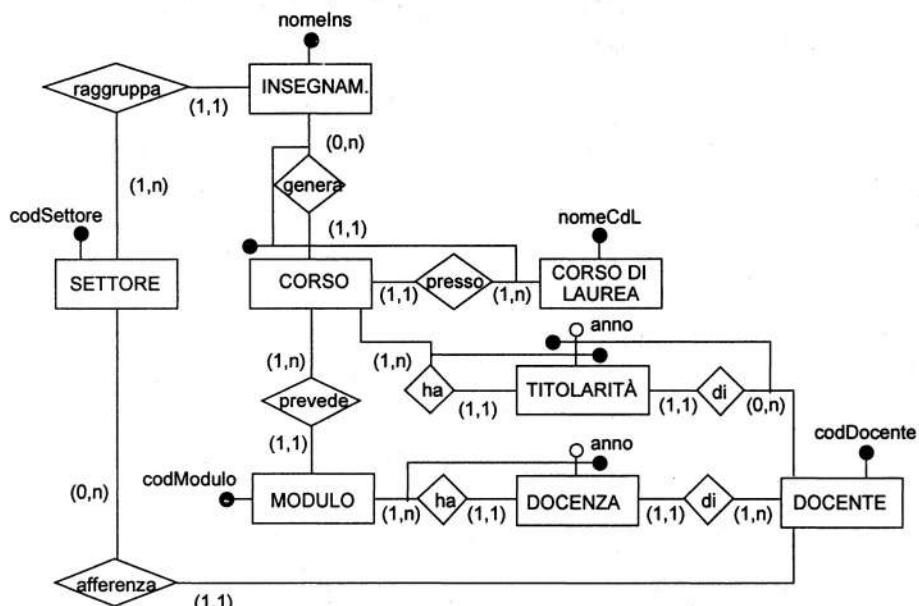
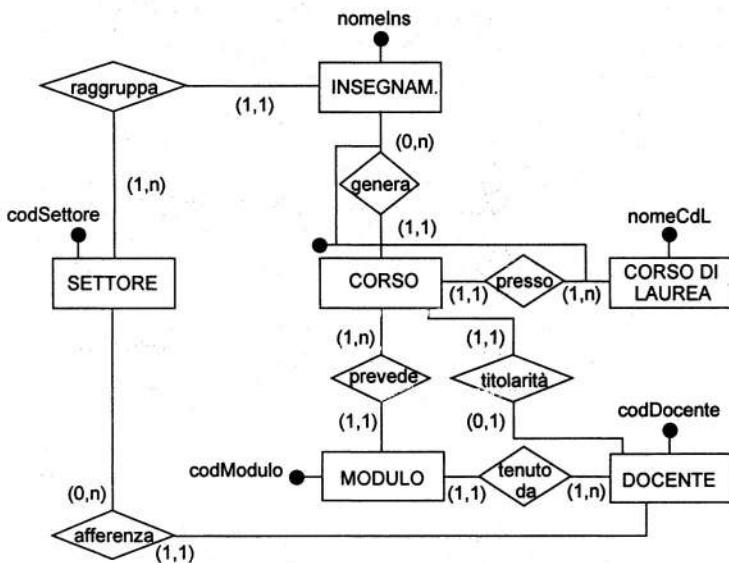
☞ Suggerimento:

Si modifichi lo schema così da rendere possibile intestare il conto a un soggetto non ospitato (ad esempio a una ditta o a un ente). Si consideri inoltre il caso in cui sia necessario mantenere memorizzate tutte le informazioni riguardanti un soggiorno anche dopo il termine dello stesso.

✓ Esercizio 1.12

La pagina successiva mostra lo schema snapshot e quello storico.

L'aspetto più significativo dello schema snapshot è la definizione dell'insegnamento. Vi sono infatti informazioni (i moduli previsti e il docente titolare) che non pertengono all'insegnamento ma alla coppia <insegnamento, corso di laurea>; risulta allora conveniente introdurre un'entità CORSO che, come appare evidente dalla scelta dell'identificatore, modella le istanze di insegnamenti legate ai diversi corsi di laurea.



Per trasformare lo schema snapshot in uno schema storico, si consideri l'associazione titolarità. Se esistesse un vincolo in base al quale lo stesso docente non potesse essere titolare di uno stesso corso in due anni diversi, la storizzizzazione potrebbe essere effettuata semplicemente ponendo a n le cardinalità massime dell'associazione in entrambi i versi; l'anno di validità di ciascuna coppia <docente,corso> sarebbe espresso da un attributo sull'associazione. Poiché questo vincolo non esiste, occorre trasformare l'associazione titolarità in una entità

TITOLARITÀ con un attributo *anno*. La nuova entità sarà identificata tramite la coppia <corso,anno>, poiché in un certo anno ogni corso ha un solo titolare (esiste cioè una dipendenza funzionale del tipo *corso,anno* → *docente*); d'altronde, è possibile utilizzare in alternativa l'identificatore <*docente,anno*>, poiché in un certo anno ogni docente è titolare di al più un corso (esiste cioè una dipendenza funzionale del tipo *docente,anno* → *corso*).

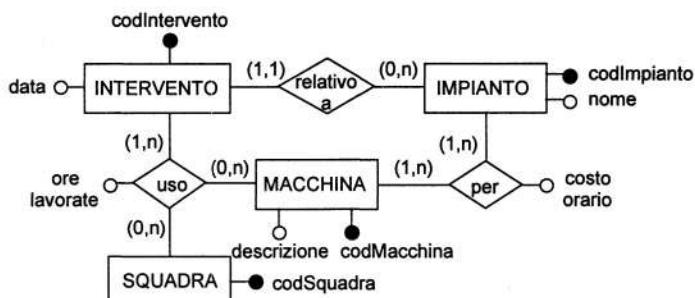
Per quanto riguarda la seconda associazione da storicizzare, *tenuto da*, valgono considerazioni analoghe; l'entità **DOCENZA** introdotta ha però un solo identificatore, <*modulo,anno*>, poiché un docente può tenere, nello stesso anno, più moduli contemporaneamente.

Si osservi che, qualora l'associazione *tenuto da* fosse stata di tipo *n:m* (un modulo può essere tenuto da più docenti), l'identificatore dell'entità **DOCENZA** sarebbe stato la tripla <*modulo,docente,anno*>.

La soluzione qui adottata per la storicizzazione è di tipo generale; ha però l'inconveniente di generare, a ogni nuovo anno, nuove istanze per le entità **TITOLARITÀ** e **DOCENZA**, indipendentemente dal fatto che la titolarità e la docenza siano effettivamente variate. Qualora sia richiesta la storicizzazione di un associazione che varia raramente, per evitare la proliferazione inutile di istanze conviene utilizzare come attributo della nuova entità introdotta non l'anno di validità, ma l'anno di inizio validità: in questo modo, infatti, una nuova istanza dell'entità viene creata solo all'atto di una variazione dell'associazione.

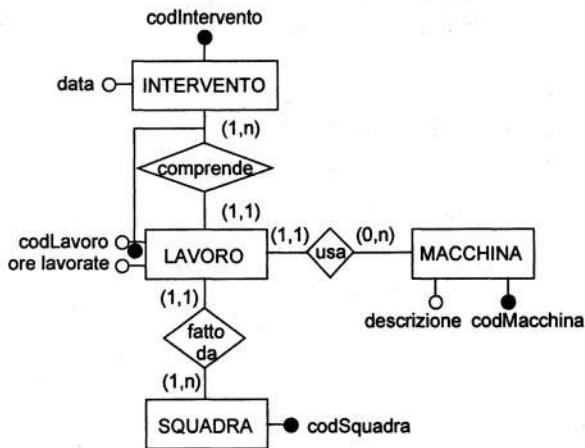
✓ Esercizio 1.13

Si noti come le frasi “*Un intervento è caratterizzato da (...) un insieme di lavori effettuati ciascuno da una squadra con una macchina per un certo numero di ore*” e “*per un intervento viene registrato il numero di ore di utilizzo, da parte di ciascuna squadra coinvolta, di ogni macchina impiegata*” definiscono in realtà un'unica specifica, modellata nello schema proposto tramite un'associazione ternaria tra **INTERVENTO**, **SQUADRA** e **MACCHINA** le cui istanze rappresentano di fatto i diversi lavori svolti.

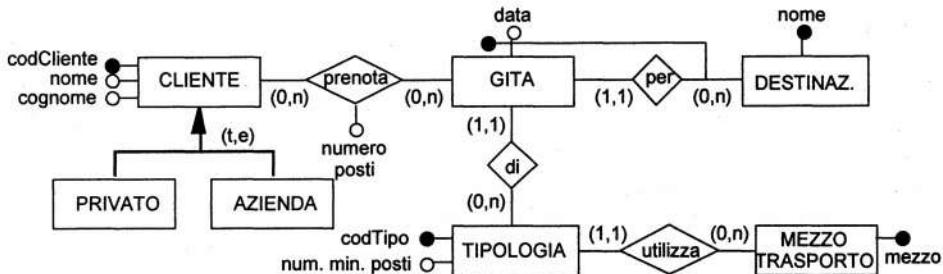


Volendo mettere in evidenza il concetto di lavoro si può optare per lo schema rappresentato nella figura successiva, dove un'istanza di **LAVORO** è identificata da un

codice unico nell'ambito di ogni intervento. In alternativa, un lavoro può anche essere identificato importando gli identificatori di INTERVENTO, SQUADRA e MACCHINA.



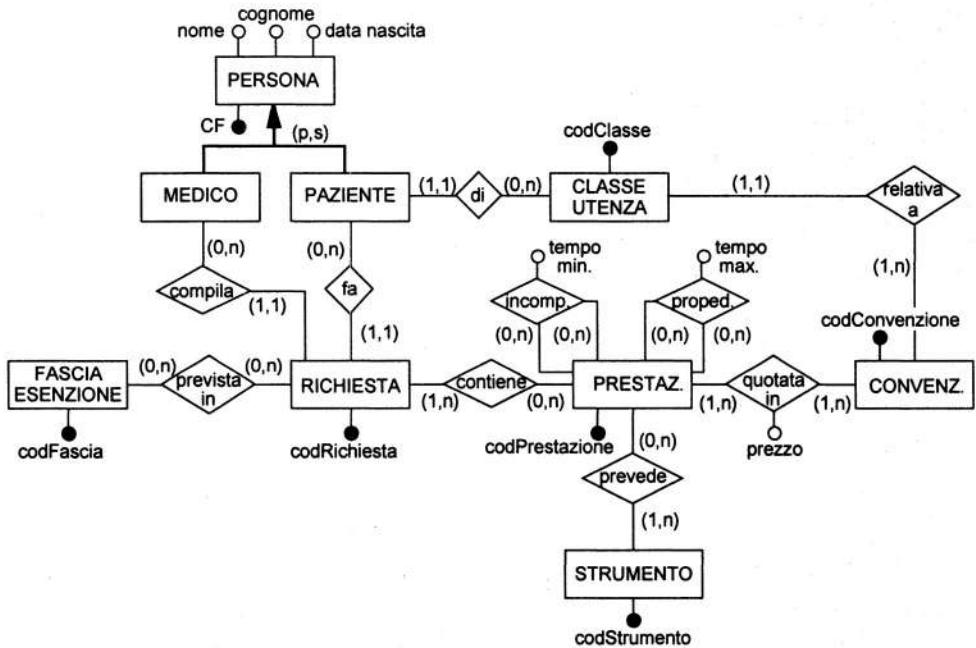
✓ Esercizio 1.14



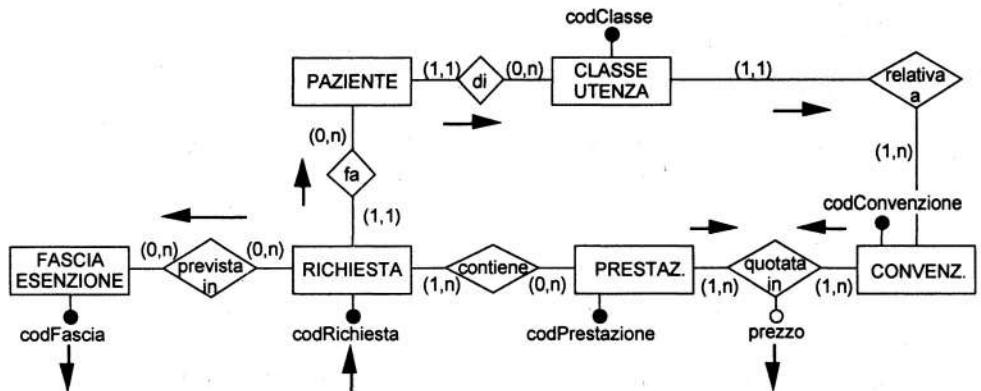
Lo schema proposto in figura non necessita di particolari spiegazioni; si richiama solamente l'attenzione sull'identificatore adottato per GITA, che riflette il vincolo secondo cui non vengono mai effettuate più gite per la stessa destinazione nello stesso giorno.

✓ Esercizio 1.15

Lo schema, nonostante la sua apparente complessità, risulta di facile costruzione. L'entità PERSONA nasce dalla generalizzazione di MEDICO e PAZIENTE. Per le prestazioni incompatibili e propedeutiche si utilizzano due associazioni unarie con attributo.



Lo schema di navigazione della query proposta è mostrato nella figura successiva; il ticket è determinato dalla convenzione della classe di utenza cui il paziente appartiene e dalle eventuali esenzioni previste nella richiesta.



Suggerimento:

Si estenda lo schema al fine di modellare anche le prenotazioni per l'erogazione delle prestazioni mediche contenute nelle richieste. Ogni prestazione verrà erogata, in un certo giorno e una certa ora, in una certa sala diagnostica.

✓ Esercizio 1.16

La fase di filtraggio dei requisiti ha l'obiettivo di individuare e correggere le eventuali sinonimie e omonimie presenti nel testo. Di regola, questa fase viene svolta con la collaborazione attiva degli esperti del dominio applicativo.

Si ha una sinonimia quando due termini diversi nel testo si riferiscono allo stesso concetto: ad esempio, il termine “*esperti*” (seconda riga, primo capoverso) è usato come sinonimo di “*ricercatori*”. Per evitare duplicazione inutile di entità, il termine più vago (in questo caso “*esperti*”) viene sostituito da quello più preciso.

Si ha una omonimia quando due termini uguali nel testo si riferiscono a concetti diversi: ad esempio, il concetto di “*tema pertinente all'area di ricerca*” (terza riga, terzo capoverso) è diverso dal concetto di “*tema della conferenza*” (seconda riga, secondo capoverso); al fine di evitare ambiguità, per uno dei due concetti viene scelto un termine differente.

capoverso riga termine correzione motivo

<i>capoverso</i>	<i>riga</i>	<i>termine</i>	<i>correzione</i>	<i>motivo</i>
1	2	esperti	ricercatori	sinonimia
2	2	membri dell'ARI	soci	più preciso
2	3	associati	soci	sinonimia
3	3	temi	argomenti	omonimia
3	3	area di ricerca	tema	sinonimia
3	5	temi	argomenti	sinonimia
3	5	area di ricerca	tema	sinonimia
4	2	area	tema	sinonimia
4	3	lavori	articoli	sinonimia
4	5	manoscritto	articolo	sinonimia
4	5/6	tema/i	argomento/i	sinonimia
4	6	manoscritto	articolo	sinonimia
4	9	lavoro	articolo	sinonimia
5	2	tema	argomento	sinonimia
5	2	lavoro	articolo	sinonimia
5	3	convegno	conferenza	sinonimia
5	4	membri ARI	soci	sinonimia

I requisiti filtrati, ottenuti effettuando le sostituzioni di termini indicate nella tabella precedente, sono esposti di seguito:

La ARI è una associazione di ricercatori. Le conferenze ARI hanno lo scopo di richiamare da tutto il mondo ricercatori esperti in un'area di ricerca.

Ciascuna conferenza è organizzata da un gruppo di lavoro, ovvero un team di ricercatori (soci) che lavorano su un'area di ricerca comune; quest'area diventa il tema della conferenza. Alla conferenza presiede un comitato di programma di cui fanno parte dieci soci, con ruoli diversi (presidente, segretario, revisori).

*Per pubblicizzare la conferenza viene spedito un **depliant** (**call for papers**) a ricercatori di tutto il mondo; esso contiene, oltre alle **date significative** per la conferenza, un invito a sottomettere un **articolo** su uno degli **argomenti** pertinenti al tema della conferenza. Il sistema deve essere in grado di preparare automaticamente la lista per il mailing, includendovi solo i ricercatori che si occupano di argomenti pertinenti o comunque affini al tema della conferenza.*

*Il **programma** stilato per la conferenza include **articoli invitati** e **articoli sottomessi**. Ciascun gruppo di lavoro che si occupa della stessa area del tema della conferenza viene invitato a scrivere un articolo; gli articoli così ottenuti entrano automaticamente a far parte del programma. Gli **articoli sottomessi**, invece, sono soggetti a un processo di revisione a opera di membri del comitato di programma. Ciascun articolo viene spedito a tre **revisori** specializzati sull'argomento dell'articolo o su argomenti affini. Quando i revisori restituiscono l'articolo allegano un **giudizio**; il presidente, sulla base dei tre giudizi, decide se accettare o meno l'articolo. Il sistema deve essere in grado di suggerire, per ogni articolo, tre nomi di revisori ritenuti idonei al compito; per ogni articolo deve poi spedire, all'autore principale, una **lettera** comunicante l'accettazione o il rifiuto dell'articolo.*

*Il **programma** della conferenza, che deve essere stilato dal sistema raggruppando gli articoli sulla base del loro argomento, viene spedito a tutti gli **autori** (compresi quelli il cui articolo non è stato accettato). Al programma è allegato un **modulo di iscrizione** alla conferenza, che presenta **tariffe** differenziate per gli autori di lavori accettati e per i soci. Sulla base dei moduli di iscrizione ricevuti, il sistema deve tenere la contabilità e stampare un **cartellino di riconoscimento** da consegnare a ciascun iscritto.*

I termini fondamentali che emergono dall'analisi delle specifiche sono evidenziati in grassetto nella descrizione dei requisiti filtrati. Al fine di identificare le entità da inserire nello schema E/R, isoliamo tra questi i termini che corrispondono a entità fisiche o a concetti del dominio applicativo:

<i>ricercatore</i>	<i>conferenza</i>
<i>area di ricerca</i>	<i>gruppo di lavoro</i>
<i>socio</i>	<i>comitato di programma</i>
<i>articolo</i>	<i>argomento</i>
<i>articolo invitato</i>	<i>articolo sottomesso</i>
<i>programma</i>	

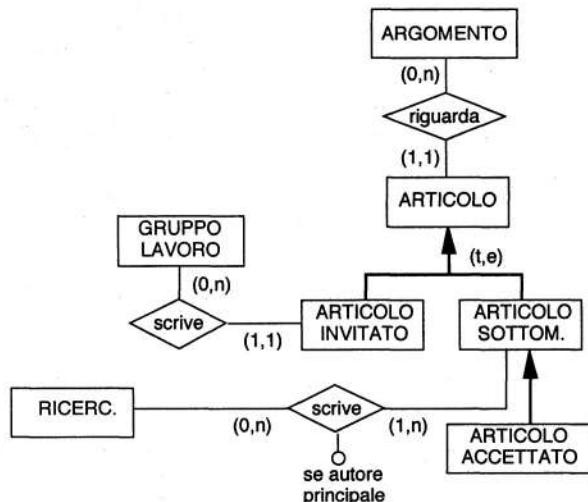
Si noti che non tutti i termini evidenziati dall'analisi delle specifiche sono stati candidati alla definizione di entità. Alcuni di essi, infatti, esprimono più propriamente associazioni (ad esempio, un'area di ricerca è *tema* di una conferenza, un ricercatore è *autore* di un articolo); altri, attributi di entità o associazioni (ad esempio, *date significative* della conferenza e *giudizio* della revisione); altri, infine, pur corrispondendo a oggetti fisici, non risultano di interesse ai fini della modellazione

dati ma piuttosto di un'eventuale modellazione dinamica o funzionale del dominio (ad esempio, *modulo di iscrizione*).

Il passo successivo consiste nell'identificazione delle associazioni e delle gerarchie di specializzazione, con l'obiettivo di individuare dipendenze e riferimenti tra classi che individuino proprietà strutturali del dominio. Per quanto riguarda gli articoli, si ha che:

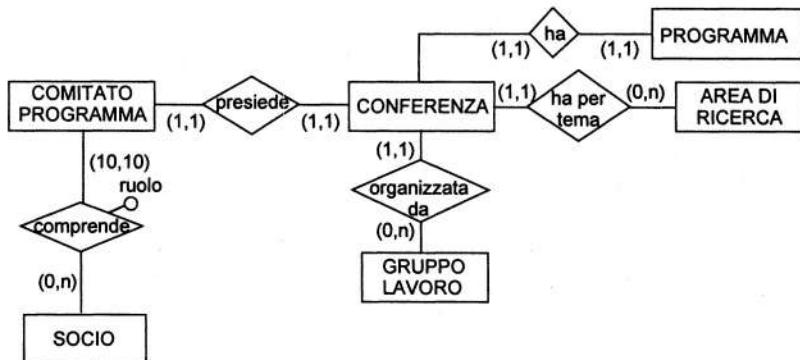
- un articolo riguarda un argomento*
- gli articoli si dividono in articoli sottomessi e articoli invitati*
- un articolo invitato è scritto da un gruppo di lavoro*
- un articolo sottomesso è scritto da ricercatori*
- un articolo sottomesso può essere accettato o meno*

Sulla base di queste informazioni è agevole disegnare lo schema concettuale seguente:



Per quanto riguarda più propriamente le conferenze e la loro organizzazione, dalle specifiche si desume che (si veda la figura successiva):

- una conferenza è organizzata da un gruppo di lavoro*
- una conferenza ha per tema un'area di ricerca*
- una conferenza è presieduta da un comitato di programma*
- un comitato di programma è costituito da 10 soci*



Per quanto riguarda infine i ricercatori:

un ricercatore si occupa di uno o più argomenti

un ricercatore può essere socio

un gruppo di lavoro è costituito da soci

un gruppo di lavoro lavora su un'area di ricerca



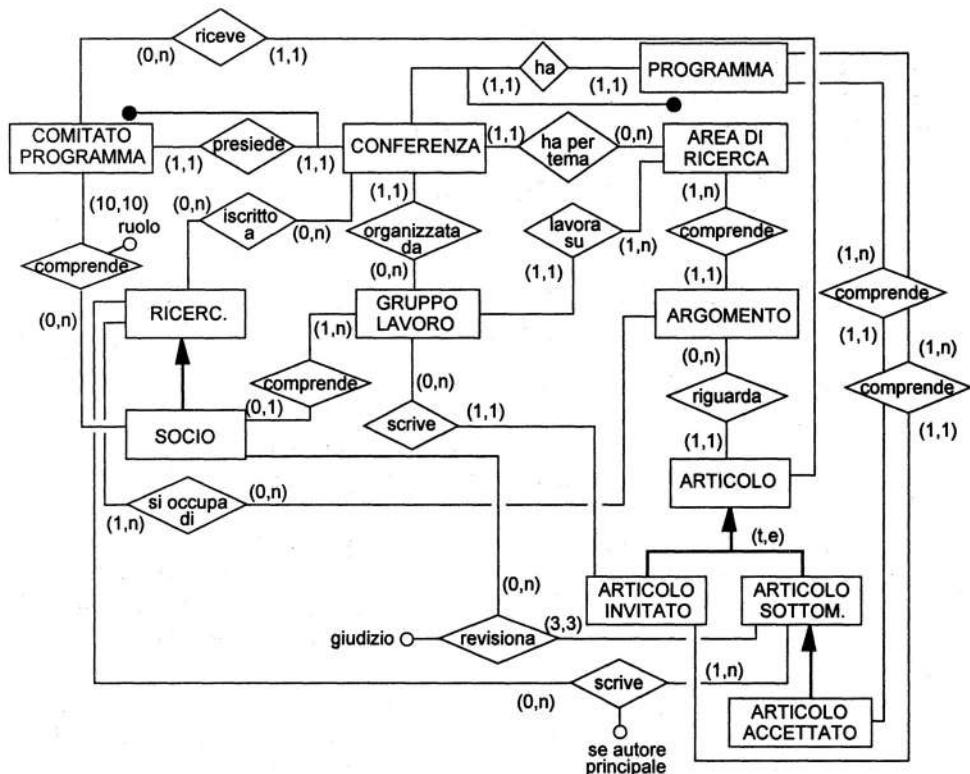
Riportiamo di seguito alcune considerazioni sulle scelte fatte:

1. Le molteplicità che non si possono evincere direttamente dalle specifiche sono definite facendo ricorso al buon senso: ad esempio, è evidente che un ricercatore può essere autore di più articoli.
2. L'accettazione di un articolo poteva essere rappresentata anche tramite un attributo dell'entità ARTICOLO SOTTOMESSO; la rappresentazione degli articoli accettati tramite un'entità risulta tuttavia preferibile, poiché permette di rappresentare nello schema il vincolo secondo cui il programma comprende solo articoli accettati.
3. Apprendiamo dalle specifiche che "*un ricercatore è esperto in un'area di ricerca*", ma anche che "*un ricercatore si occupa di argomenti*". Nello schema viene modellata solo la seconda associazione, di tipo più generale. È tuttavia accettabile anche uno schema in cui siano rappresentate entrambe le associazioni, purché siano attribuiti significati diversi alle locuzioni "*essere esperti*" e "*occuparsi di*".
4. Il tema della conferenza può sembrare rappresentabile come un attributo dell'entità CONFERENZA; poiché però il tema è in realtà una possibile area di ricerca, ed è stata modellata una entità corrispondente, l'uso di un attributo risulta non conveniente. Si noti che anche rappresentare i temi tramite una

sottoentità di AREA DI RICERCA è errato: infatti, il fatto di essere un tema non è una proprietà strutturale di un'area di ricerca, ma è legato all'esistenza di una specifica conferenza.

5. Il "ruolo" (presidente, revisore, ecc.) non è un attributo della entità SOCIO: infatti, uno stesso socio può ricoprire, nell'ambito di comitati di conferenze diverse, ruoli differenti. Una rappresentazione alternativa a quella adottata consiste nell'utilizzare, al posto della generica associazione comprende, associazioni diverse per ogni possibile ruolo. Si avrebbe così: "*un comitato ha come presidente un socio*", "*un comitato ha come revisori uno o più soci*", ecc.

Integrando i tre schemi proposti si ottiene il modello disegnato nella figura successiva.



Il collegamento tra i sottoschemi è stato effettuato considerando che:

1. Un articolo sottomesso è revisionato da tre revisori, che restituiscono ciascuno un giudizio. È quindi corretto rappresentare il giudizio come un attributo dell'associazione tra SOCIO e ARTICOLO SOTTOMESSO.
2. Un'area di ricerca comprende più argomenti.
3. Un articolo è inviato a una conferenza.

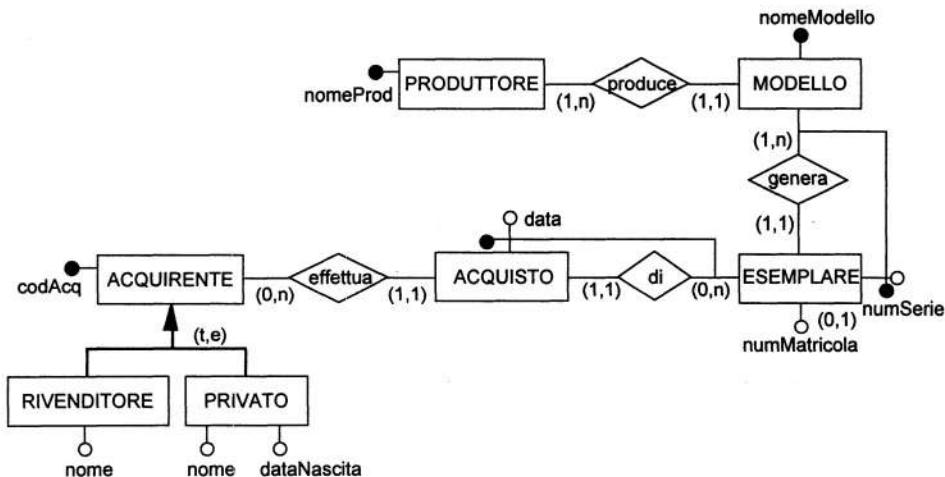
4. Un ricercatore può essere iscritto a diverse conferenze.
5. Un programma è costituito da articoli invitati e articoli sottomessi.

Per semplicità di disegno non sono stati indicati gli attributi delle entità ma solo quelli delle associazioni. Per ogni entità per la quale non sia stato esplicitamente indicato un identificatore esterno, si intende che l'identificatore sarà un appropriato codice (ad esempio, per l'entità CONFERENZA, codConferenza).

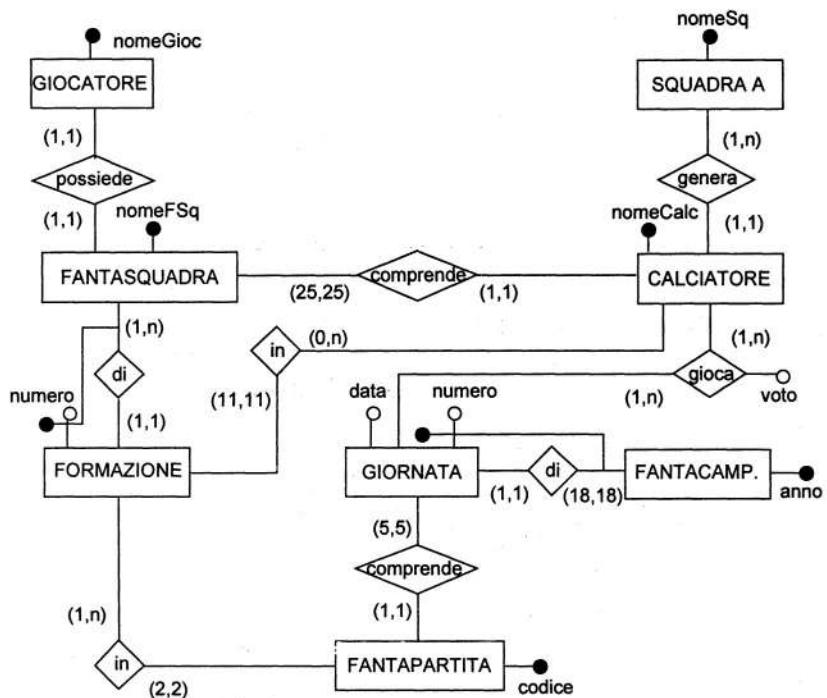
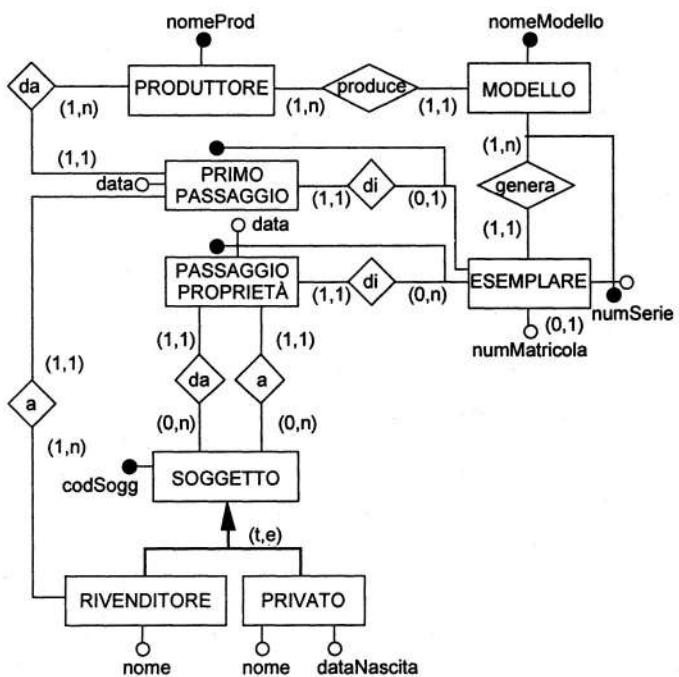
✓ Esercizio 1.17

Un possibile schema E/R è mostrato in figura.

Per la modellazione dei passaggi di proprietà è stata adottata una soluzione non ridondante in cui l'entità ACQUISTO rappresenta l'acquisizione di un'esemplare da parte di un rivenditore o di un privato. I vantaggi principali sono la semplicità e la possibilità di modellare anche eventuali cessioni di auto usate da un privato a un rivenditore. Per contro, non viene rappresentato il vincolo in base al quale un'esemplare acquisisce un numero di matricola nel momento in cui viene ceduto dal produttore al rivenditore. Inoltre, la ricostruzione di un passaggio di proprietà non è immediata ma necessita di un semplice algoritmo (il primo acquisto a_0 corrisponde a un passaggio di proprietà dal produttore dell'auto al rivenditore acquirente; ogni altro acquisto a_i corrisponde a un passaggio di proprietà dall'acquirente del precedente acquisto a_{i-1} all'acquirente dell'acquisto a_i).



La soluzione alternativa mostrata nelle figure seguente (in alto) rappresenta esplicitamente cedente e acquirente per ogni passaggio di proprietà, ma ha lo svantaggio di essere ridondante.



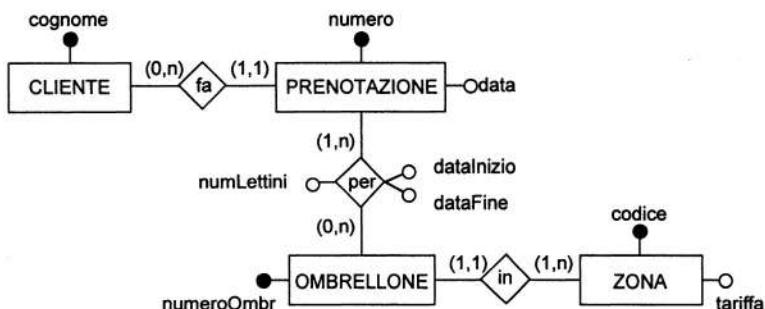
✓ Esercizio 1.18

Il problema principale dell'esercizio è costituito dalla modellazione delle formazioni delle fantasquadre. Si noti che, nella soluzione mostrata nella pagina precedente (in basso), il legame tra una fantapartita e le due fantasquadre che la disputano non è rappresentato direttamente bensì tramite le formazioni.

In linea di principio, sarebbe possibile eliminare l'associazione di tra le entità FORMAZIONE e FANTASQUADRA; infatti, è possibile risalire alla fantasquadra cui una formazione si riferisce tramite un qualunque giocatore della formazione. D'altronde, l'informazione su quali squadre hanno disputato una partita è troppo importante per essere affidata a un percorso costituito da tre associazioni.

✓ Esercizio 1.19

L'aspetto più interessante di questo esercizio è legato alla rappresentazione delle prenotazioni. Si noti come, con la soluzione presentata in figura, viene implicitamente modellato il vincolo in base al quale, all'interno di una data prenotazione, ogni ombrellone viene prenotato per un solo intervallo di tempo. L'attributo numLettini è posto sull'associazione per poiché assume un valore indipendente per ogni ombrellone.

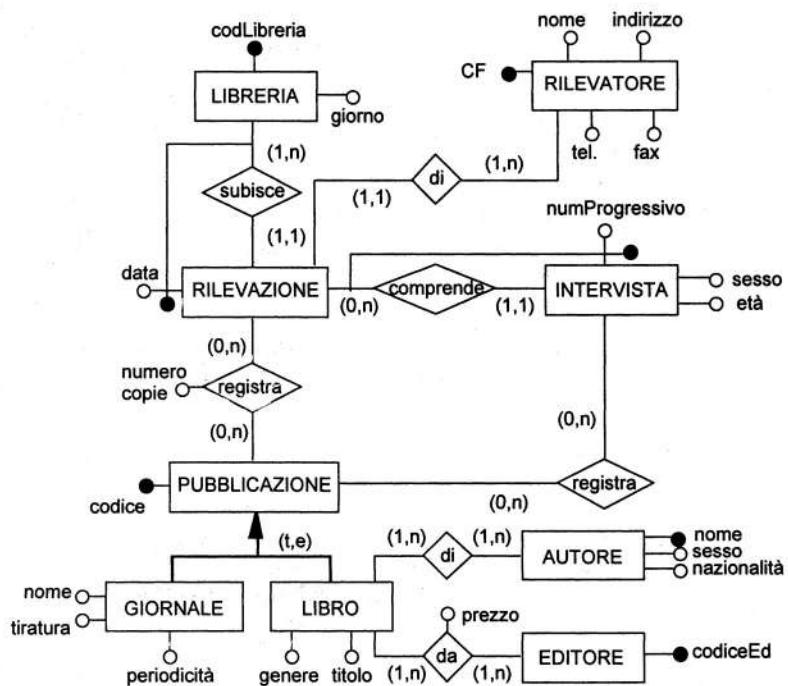


✓ Esercizio 1.20

La figura seguente mostra una possibile soluzione.

L'entità RILEVATORE è collegata a RILEVAZIONE invece che a LIBRERIA poiché non è detto che in ciascuna libreria presti servizio un solo rilevatore.

È conveniente rappresentare la gerarchia di generalizzazione che definisce PUBBLICAZIONE, così da poter utilizzare quest'ultima per le associazioni registra con RILEVAZIONE e INTERVISTA.

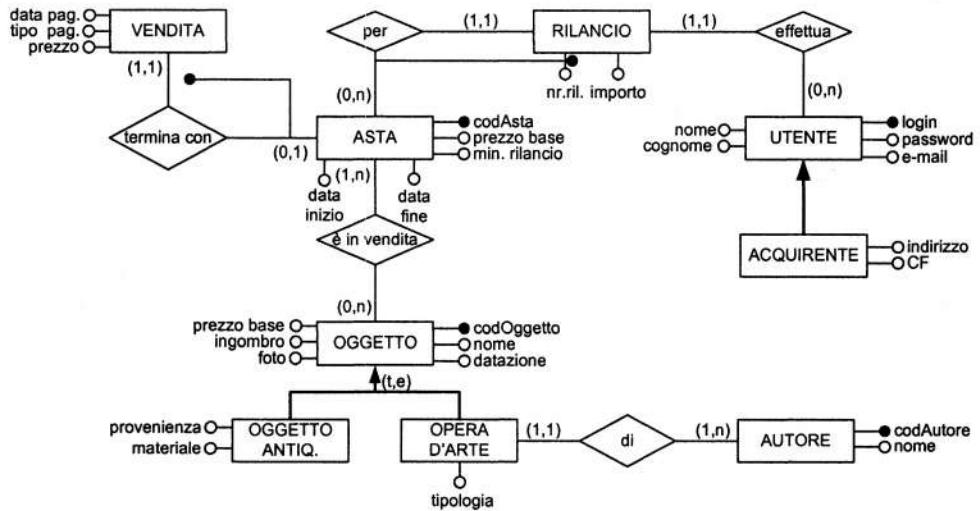


✓ Esercizio 1.21

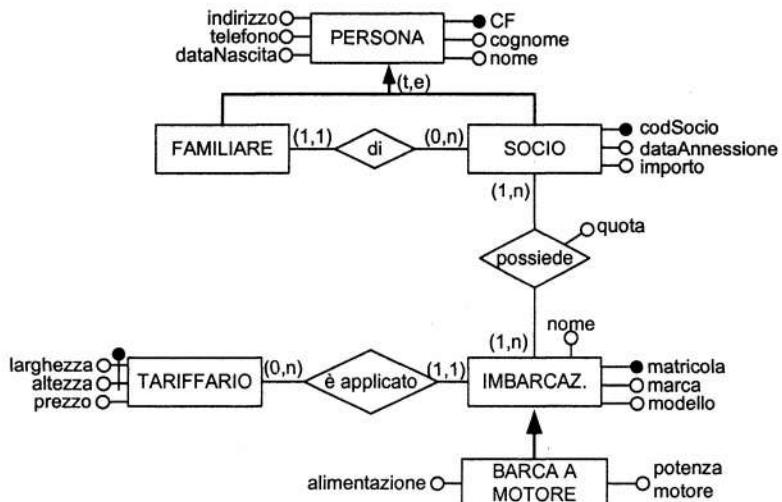
Un possibile schema E/R è mostrato nella figura seguente.

L'aspetto più significativo di questo esercizio è la rappresentazione dei rilanci. Poiché ciascun utente effettua normalmente più rilanci per una stessa asta, questi non possono essere modellati attraverso una semplice associazione N:M tra ASTA e UTENTE, ma si rende necessario introdurre una nuova entità RILANCIO alla quale collegare le informazioni relative alle diverse offerte. Per questa entità è possibile definire un identificatore misto ottenuto abbinando l'identificatore dell'asta e il numero del rilancio.

Nel caso in cui l'asta si concluda con una vendita, devono essere memorizzati i dati relativi al pagamento e all'acquirente; non deve invece essere introdotta un'associazione tra VENDITA e ACQUIRENTE in quanto è possibile derivare il nome dell'acquirente selezionando colui che ha effettuato il rilancio di importo maggiore e quindi l'informazione rappresentata attraverso l'associazione sarebbe ridondante.



✓ Esercizio 1.22



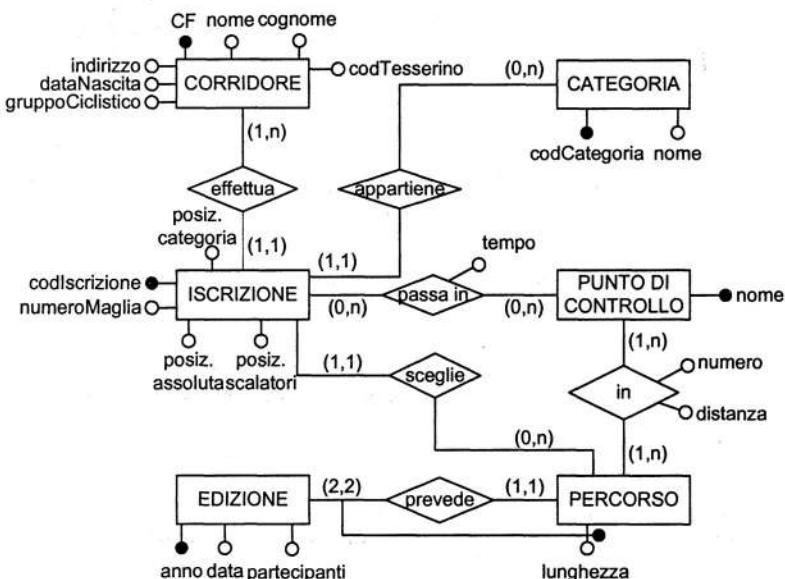
Lo schema proposto in figura non necessita di particolari commenti. Si richiama solamente l'attenzione sui dati necessari per il calcolo della quota annuale di ciascun socio: mentre non è necessario memorizzare la quota fissa che viene addebitata per ciascun familiare (uguale per tutti), è necessario invece rappresentare il tariffario che fissa il prezzo sulla base delle dimensioni delle imbarcazioni.

 Suggerimento:

Si estenda lo schema al fine di mantenere uno storico dei prezzi di rimessaggio delle imbarcazioni praticati dal circolo nel corso degli anni.

✓ Esercizio 1.23

Il punto centrale di questo esercizio consiste nella rappresentazione dei percorsi delle diverse edizioni della gara. In particolare bisogna prestare attenzione a gestire correttamente l'attributo *distanza*: questo non può essere associato all'entità PUNTO DI CONTROLLO in quanto indica la distanza dal punto di controllo successivo e dipende quindi dal percorso all'interno del quale è stato inserito. Per questo viene rappresentato come attributo nell'associazione che lega PUNTO DI CONTROLLO e PERCORSO. Per poter ricostruire la sequenza dei punti (e quindi attribuire un significato all'attributo *distanza*) è necessario inoltre indicare la posizione (numero) di ciascun punto di controllo all'interno della sequenza prevista per un determinato percorso.

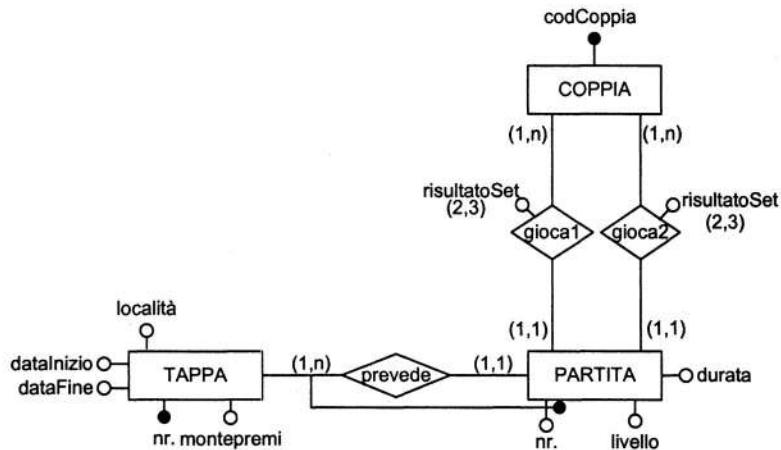


Poiché si vuole mantenere uno storico di tutte le edizioni della gara, è necessario scindere i dati del corridore che rimangono invariati nel tempo (nome, cognome, ecc...) da quelli legati alle singole iscrizioni alla gara (numero di maglia, posizione in classifica, ecc...). Anche la categoria di appartenenza del corridore deve essere legata a ISCRIZIONE, in quanto potrebbe cambiare nel tempo.

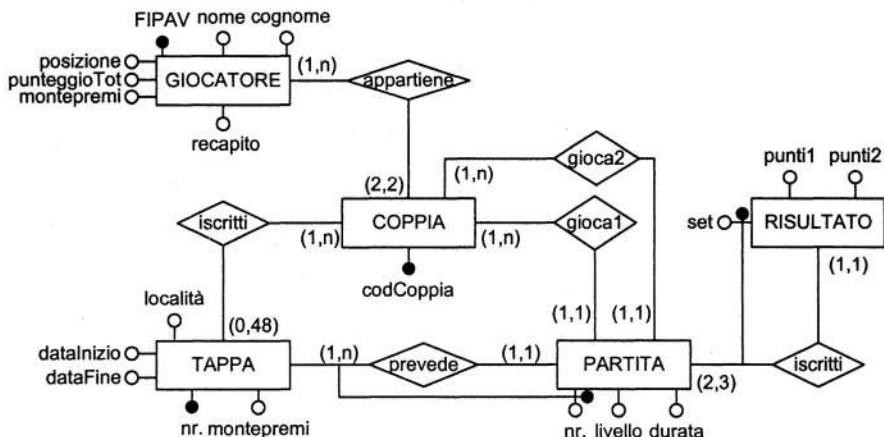
Non è necessario creare un'associazione tra ISCRIZIONE e EDIZIONE in quanto questa informazione può essere ricostruita attraverso il percorso scelto da ciascun corridore, che determina l'edizione alla quale il corridore stesso si è iscritto.

✓ Esercizio 1.24

La difficoltà principale di questo esercizio consiste nella rappresentazione delle partite e, in particolare, delle coppie che le hanno disputate e dei risultati che esse hanno ottenuto nei diversi set. A tale scopo si consideri la porzione di schema riportata nella figura seguente.



Questa soluzione non può essere considerata corretta in quanto non permette di ricostruire i risultati dei set giocati durante le diverse partite. Si ricorda infatti che, pur essendo **risultatoSet** un attributo multiplo, che permette quindi di indicare i risultati di più set, non esiste un ordinamento tra i valori e non è possibile risalire al set al quale si riferisce ciascun risultato. Una possibile soluzione al problema è riportata nello schema seguente.



In questa soluzione viene forzato un ordinamento dei risultati dei diversi set (attraverso l'attributo **set**) che permette di ricostruire correttamente l'informazione:

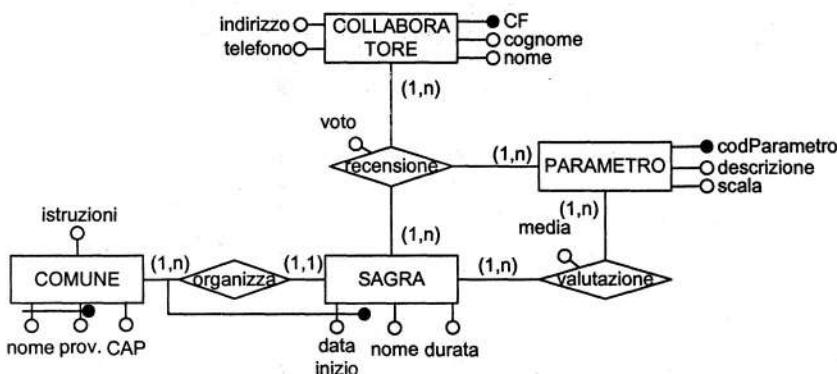
per ciascun set l'attributo punti1 (punti2) rappresenta il punteggio ottenuto dalla coppia che partecipa all'associazione gioca1 (gioca2) per la partita in esame.

L'associazione iscritti tra COPPIA e TAPPA è ridondante in quanto è possibile risalire alle tappe alle quali una determinata coppia è iscritta attraverso le partite giocate; può tuttavia essere inserita comunque nello schema per modellare il vincolo secondo il quale a una partita non possono iscriversi più di 48 coppie di giocatori.

✓ Esercizio 1.25

Una possibile soluzione è mostrata in figura.

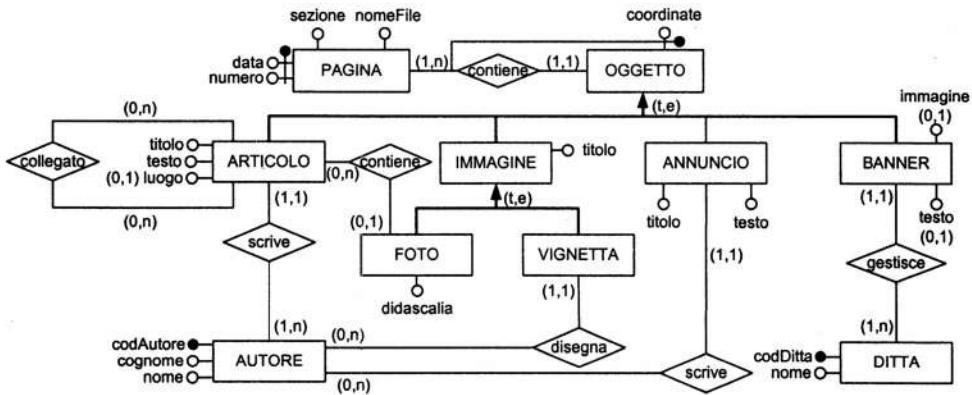
L'aspetto centrale di questo schema è la rappresentazione delle recensioni. Poiché ciascun parametro riceve per una stessa sagra valutazioni diverse da parte di collaboratori differenti, si deve ricorrere a un'associazione ternaria. In questo modo, ciascuna istanza dell'associazione recensisce rappresenta il voto attribuito da un determinato collaboratore a un parametro specifico, relativamente a una determinata sagra.



✓ Esercizio 1.26

Un possibile schema E/R è mostrato in figura.

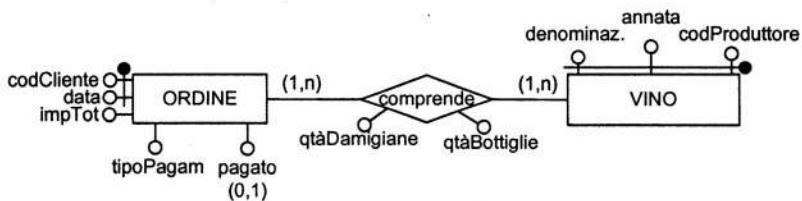
Per l'entità OGGETTO è stato definito un identificatore misto che nasce dall'abbinamento tra l'identificatore della pagina e le coordinate dell'oggetto, poiché ovviamente in una determinata data, una pagina del giornale ha una composizione ben precisa, nella quale ciascun oggetto occupa una determinata posizione; naturalmente in pagine diverse possono esistere oggetti con le stesse coordinate, rendendo inutilizzabile il solo attributo coordinate come identificatore nell'entità OGGETTO.



Un altro aspetto interessante è la rappresentazione del requisito secondo il quale “*Tutti gli articoli che fanno riferimento alla stessa notizia (anche pubblicati in date diverse) vengono collegati fra loro per facilitarne la ricerca*”. Il problema può essere risolto introducendo un’associazione unaria collegato che mette in relazione articoli diversi che trattano la stessa notizia.

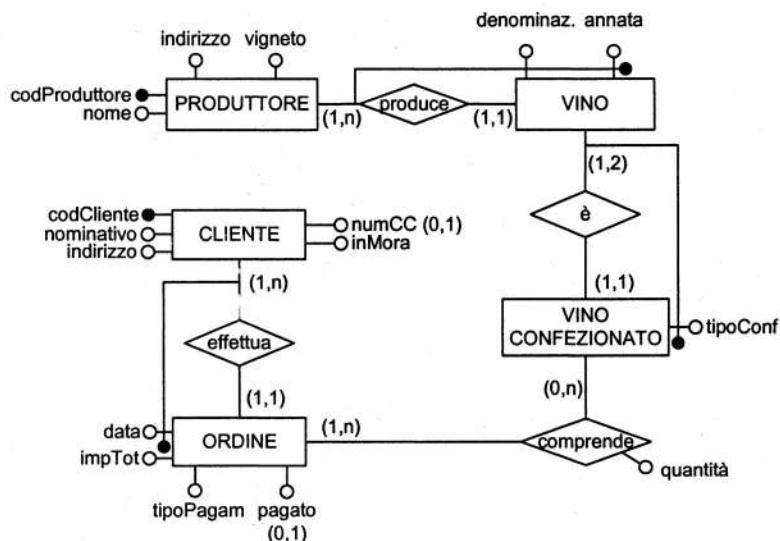
✓ Esercizio 1.27

L’aspetto fondamentale di questo esercizio consiste nella definizione del contenuto degli ordini. Il testo dell’esercizio specifica che “*ciascun ordine comprende un numero arbitrario di bottiglie oppure damigiane di vini differenti*”. Una possibile soluzione è mostrata nella figura successiva.



In questo modo, in ciascun ordine possono essere richieste un numero arbitrario di bottiglie e di damigiane dello stesso vino. Poiché, secondo quanto detto nel testo, la maggior parte del vino viene imbottigliata, per la maggior parte degli ordini l’attributo qtàDamigiane assumerà valore 0. Una soluzione alternativa è quella presentata nella figura successiva che riporta lo schema concettuale completo. In questa soluzione è stata introdotta una nuova entità VINO CONFEZIONATO che rappresenta un determinato tipo di vino confezionato in bottiglia o in damigiana. L’ordine sarà dunque composto da una determinata quantità di vino confezionato. Si noti che questa seconda soluzione risulta maggiormente flessibile nel caso in cui si rendesse necessario inserire nuovi tipi di confezioni (es. bottiglie con capacità diverse); nella prima soluzione, questo aggiornamento richiederebbe una ristrutturazione dello schema, mentre nella seconda

soluzione si gestirebbe attraverso un semplice inserimento nell'entità VINO CONFEZIONATO.



Quanto descritto in “*Ciascun ordine emesso da un cliente che non paga tramite carta di credito deve essere saldato entro 30 giorni tramite vaglia postale inviato assieme alla merce; se, trascorsi i 30 giorni, l’ordine non è saldato, viene inviato un sollecito, e contemporaneamente il cliente viene caratterizzato come “in mora”. Se dopo altri 30 giorni il sollecito non è saldato, il nominativo del cliente è segnalato ad un’azienda di recupero dei crediti*” descrive aspetti dinamici del problema che non trovano un riscontro nello schema E/R, se non per l’attributo **inMora** nell’entità **CLIENTE**.

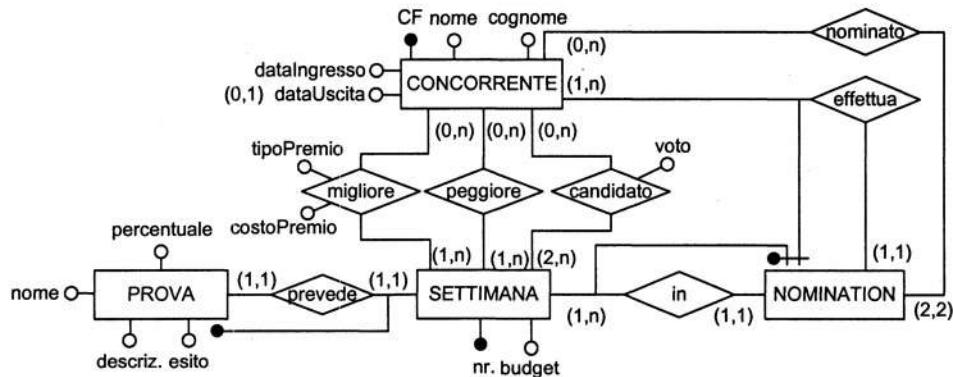
✓ Esercizio 1.28

Un possibile schema E/R è mostrato in figura.

Molti aspetti del problema sono legati al concetto di settimana, rendendo consigliabile l’introduzione di un’entità lo rappresenti.

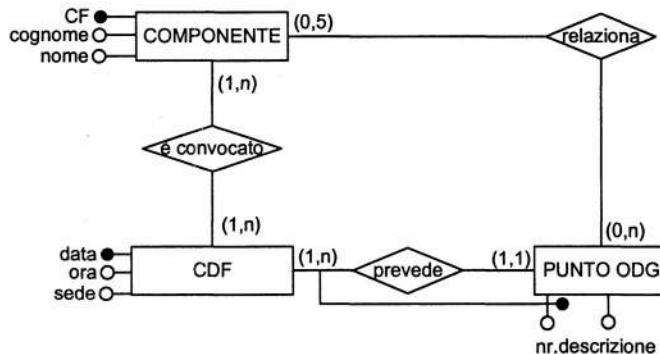
Si richiama l’attenzione alla rappresentazione delle nomination. Ciascuna nomination può essere identificata dalla settimana di riferimento e dal concorrente che l’ha effettuata. I due nominati sono rappresentati attraverso un’ulteriore associazione tra NOMINATION e CONCORRENTE.

I concetti di “migliore della settimana”, “peggiore della settimana” e “candidato all’eliminazione” sono stati descritti attraverso le relative associazioni tra le entità CONCORRENTE e SETTIMANA. Sarebbe stato invece un errore rappresentarli attraverso una gerarchia di generalizzazione definita su CONCORRENTE in quanto non descrivono una classificazione stabile dei concorrenti, ma una caratteristica dinamica legata alla settimana di riferimento.



✓ Esercizio 1.29

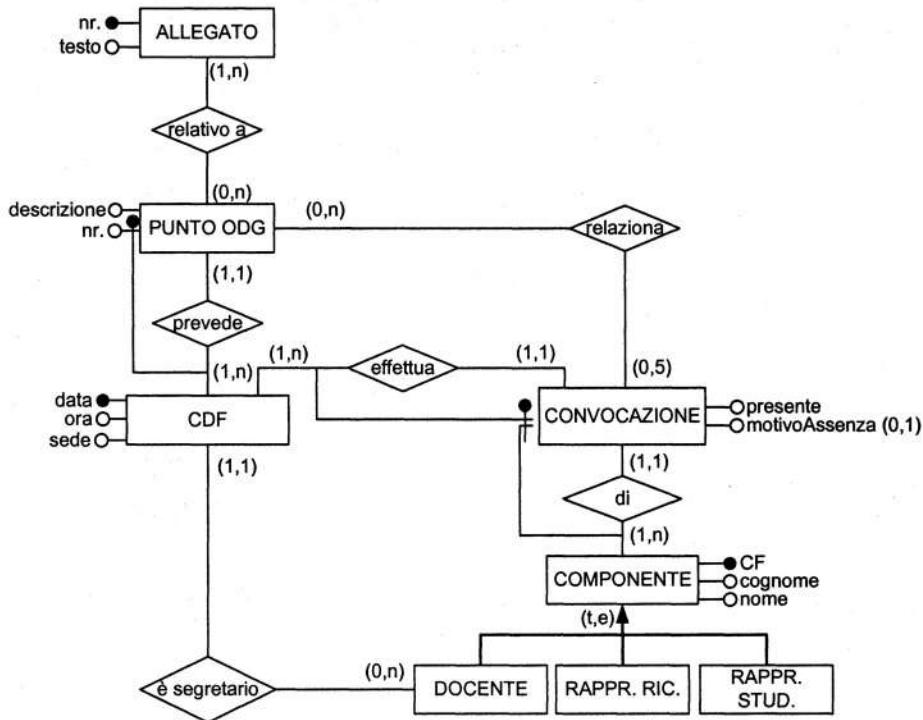
L'aspetto centrale di questo esercizio è la rappresentazione delle convocazioni ai consigli di facoltà. Nella soluzione riportata nella figura seguente, la convocazione viene rappresentata attraverso un'associazione tra le entità **COMPONENTE** e **CDF**. Il vincolo descritto dalla frase “durante un certo CdF un componente non può relazionare più di cinque punti dell’OdG” viene espresso attraverso la cardinalità massima (5) con la quale l’entità **CONCORRENTE** partecipa all’associazione relaziona.



La soluzione sopra rappresentata non è corretta, in quanto il vincolo relativo al numero di punti è valido all'interno di un determinato consiglio di facoltà, mentre lo schema indica che complessivamente, ciascun componente può relazionare al massimo su cinque punti. Una possibile soluzione è rappresentata nello schema seguente, nel quale è stata introdotta l'entità **CONVOCAZIONE**. In questa soluzione l'associazione relaziona lega **PUNTO ODG** e **CONVOCAZIONE** e il vincolo può essere correttamente rappresentato definendo opportunamente la cardinalità dell'associazione.

Gli attributi `presente` e `motivoAssenza` nell'entità **CONVOCAZIONE** rispondono al requisito “Ognuno di questi componenti ... può quindi partecipare al CdF, o

giustificare la propria assenza riportando il motivo dell'assenza, oppure risultare assente ingiustificato".

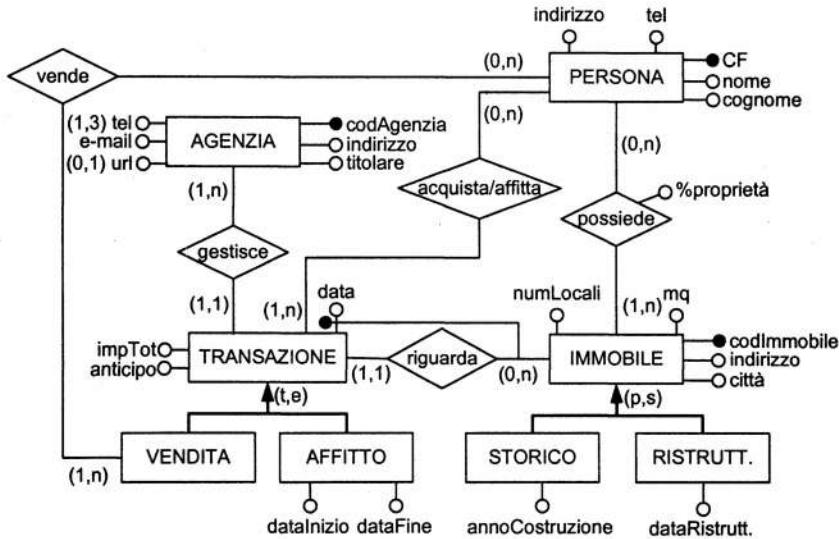


✓ Esercizio 1.30

Questo esercizio non presenta particolari difficoltà. Una possibile soluzione è riportata in figura.

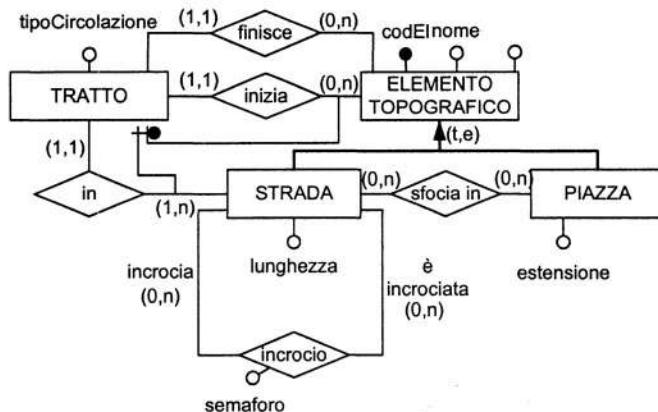
Si richiama l'attenzione sul requisito "*A seguito della vendita di un immobile il proprietario dell'immobile viene modificato, e i vecchi proprietari rimangono memorizzati nel DB solo come parte della transazione di vendita*". Nella soluzione proposta, l'associazione possiede permette di risalire ai proprietari attuali di ciascun immobile, mentre lo storico dei proprietari viene modellato attraverso l'associazione **vende** che lega le transazioni di vendita e le persone, indicando chi ha venduto l'immobile (che ne era naturalmente proprietario in quel momento).

Per **TRANSAZIONE**, supponendo che non vengano effettuate due transazioni per lo stesso immobile nello stesso giorno, è stato definito un identificatore misto che nasce dall'unione dell'identificatore dell'immobile e dalla data della transazione.



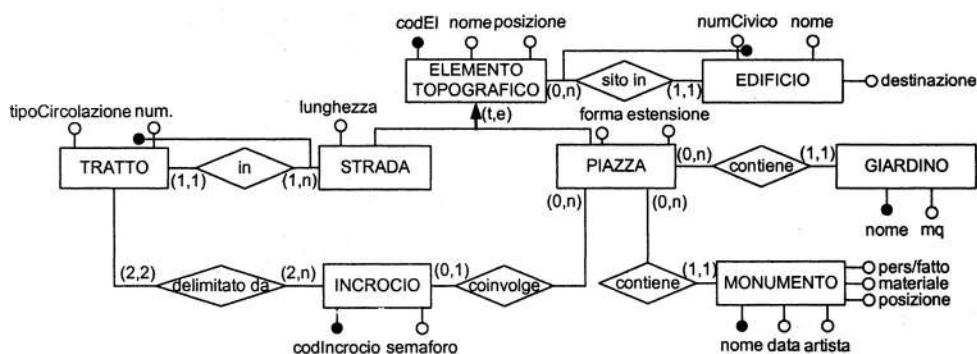
✓ Esercizio 1.31

La difficoltà maggiore in questo esercizio è legata alla rappresentazione delle strade, dei tratti che le compongono e degli incroci secondo quanto descritto nel testo: *“Di una strada si sa quali altre strade incrocia (un incrocio può coinvolgere due o più strade ed eventualmente una piazza) e in quali piazze (eventualmente) sfocia. Gli incroci possono essere controllati o meno da semafori. Le strade possono avere dei tratti a doppio senso di marcia, a senso unico, a circolazione limitata o a isola pedonale: in questo caso è necessario indicare gli incroci (o le piazze) di inizio e fine del tratto di strada e la tipologia di limitazione della circolazione”*.



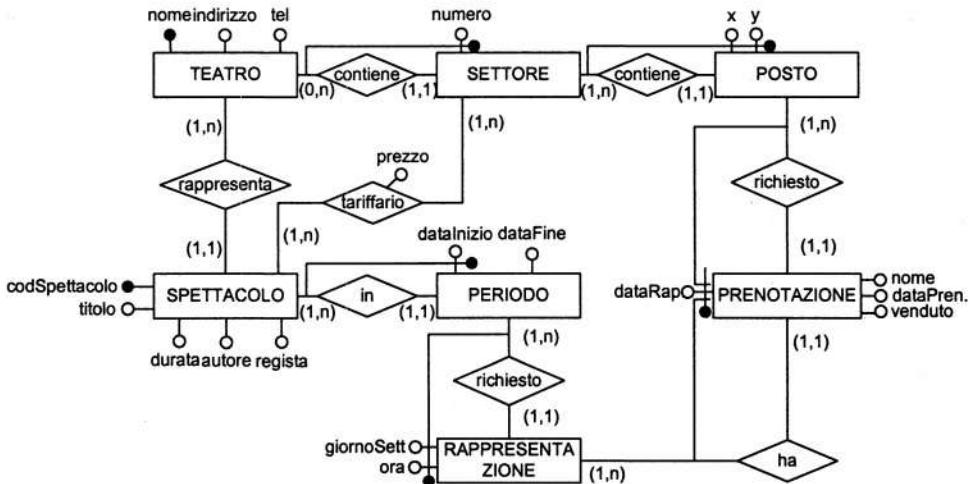
Una prima soluzione è rappresentata nella figura precedente, in cui gli incroci sono rappresentati attraverso l'associazione unaria *incrocio* che mette in relazione le diverse strade che costituiscono lo stesso incrocio. Bisogna però considerare che ciascun tratto di strada è delimitato da due incroci e la soluzione precedente non permette di definire un legame tra gli incroci di una strada e i tratti che la compongono.

Una soluzione corretta è rappresentata in figura. In questo schema gli incroci non sono più rappresentati attraverso un'associazione, bensì per mezzo di un'entità. Ciascun tratto di strada, identificato dalla strada stessa e dal numero del tratto, è legata ai due incroci che lo delimitano. Gli incroci non sono associati direttamente alle strade in quanto è possibile ricostruire l'informazione attraverso l'entità TRATTO.



✓ Esercizio 1.32

Il nucleo di questo esercizio consiste nella rappresentazione degli spettacoli, con i relativi periodi e orari di rappresentazione. Innanzitutto è stata introdotta un'entità PERIODO che modella i diversi periodi di rappresentazione degli spettacoli (es. Spettacolo Cats, dal 01/01/2005 al 15/02/2005). Per questa entità è possibile definire un identificatore misto composto dal codice dello spettacolo e dalla data di inizio del periodo. All'interno di ciascun periodo di rappresentazione bisogna poi dettagliare gli orari delle rappresentazioni (es. il sabato alle 21:00 e la domenica alle 15:00 e alle 21:00). Questo concetto è rappresentato dall'entità RAPPRESENTAZIONE, nella quale ciascuna rappresentazione può essere identificata attraverso un identificatore misto composto dall'identificatore del periodo e dal giorno e ora della rappresentazione; è necessario includere anche l'attributo *ora* nell'identificatore in quanto, nello stesso giorno della settimana, possono tenersi più rappresentazioni in orari diversi. Rimane da risolvere il problema delle prenotazioni dei posti. In ciascuna prenotazione viene riservato un posto per una rappresentazione in un giorno specifico (es. rappresentazione di domenica 09/01/2005 alle ore 21), rendendo necessaria l'introduzione di un attributo *dataRap* che rappresenta il giorno specifico di riferimento della prenotazione (nell'esempio 09/01/2005). L'identificatore misto per PRENOTAZIONE nasce dall'abbinamento della rappresentazione, della data di rappresentazione scelta e del posto riservato.

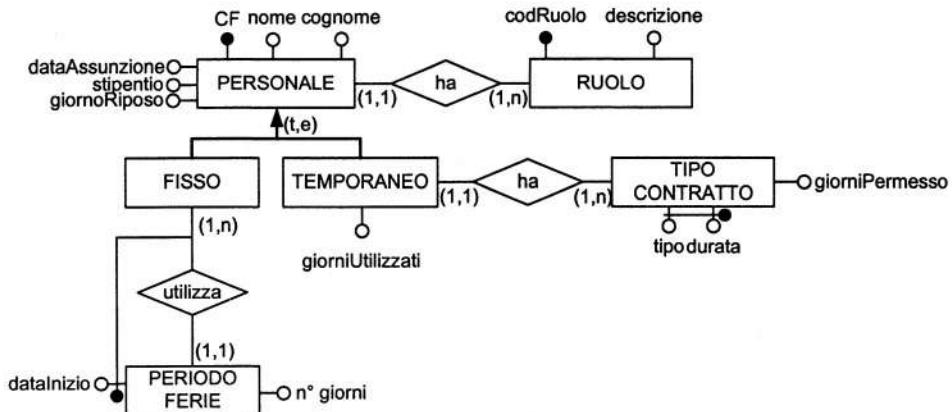


Anche per SETTORE e POSTO sono stati introdotti degli identificatori misti: ciascun settore ha un numero univoco all'interno di un teatro, e ciascun posto può essere identificato univocamente dalle sue coordinate all'interno di un settore.

Il vincolo “*le rappresentazioni di uno spettacolo si svolgono tutte nello stesso teatro*” è stato rappresentato introducendo l’associazione rappresenta tra SPETTACOLO e TEATRO. Infine il prezzo degli spettacoli varia a seconda del settore ed è stato inserito come attributo nell’associazione tra le due entità.

✓ Esercizio 1.33

Una possibile soluzione è mostrata in figura.

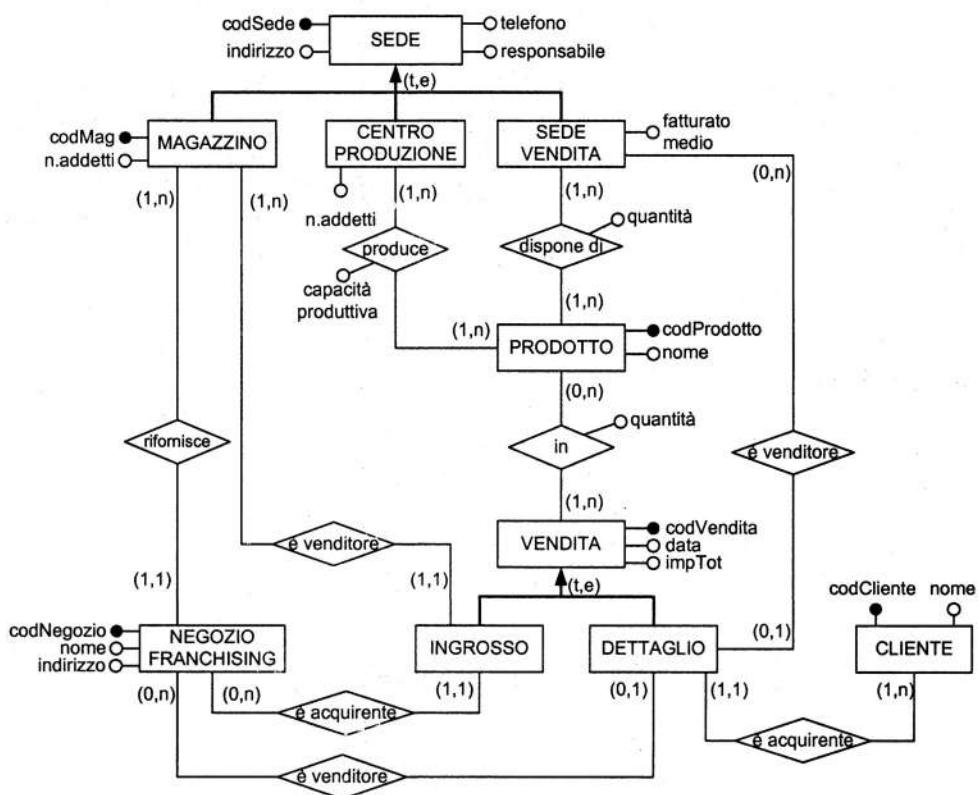


Per il personale fisso vengono memorizzati i singoli periodi di ferie utilizzati (ciascun periodo è identificato dalla persona e dalla data di inizio) mentre non è possibile rappresentare il vincolo secondo il quale ciascun dipendente può prendere un

massimo di 30 giorni di ferie all'anno. Per il personale temporaneo viene semplicemente mantenuto un attributo che indica il numero di giorni di permesso già utilizzati per l'anno in corso, mentre il numero di giorni a disposizione, che dipende solo dal tipo e dalla durata del contratto, è riportato nell'entità TIPO CONTRATTO.

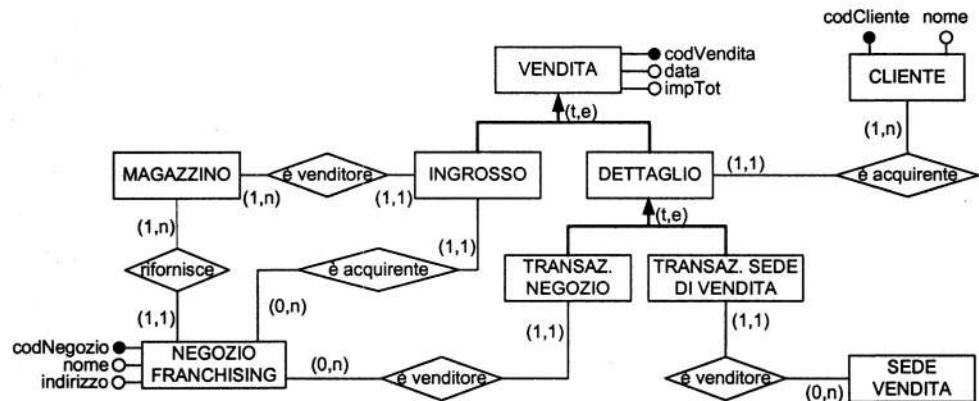
✓ Esercizio 1.34

La difficoltà maggiore di questo esercizio è la rappresentazione delle transazioni di vendita secondo quanto specificato nel testo: "Per ogni transazione di vendita (sia al dettaglio che tra magazzini e negozi in franchising) devono essere registrate i prodotti venduti (e relative quantità), la data di vendita e il totale della transazione". Una prima soluzione è riportata nella figura seguente.



In questo schema è stata introdotta una gerarchia di generalizzazione che divide le transazioni di vendita in transazioni all'ingrosso e al dettaglio. Per queste ultime l'acquirente è un cliente, ma il venditore può essere un negozio in franchising o una sede di vendita al minuto. Nella soluzione proposta questo concetto viene espresso introducendo due associazioni, entrambe opzionali, che legano le transazioni al dettaglio e le due entità NEGOZIO FRANCHISING e SEDE VENDITA. L'opzionalità di

queste associazioni non permette di rappresentare il vincolo secondo il quale, per ciascuna transazione, deve essere indicato esattamente un venditore. Una soluzione migliore è descritta nello schema seguente in cui le transazioni al dettaglio sono state ulteriormente suddivise riportando in due sotto-entità quelle effettuate da negozi in franchising e quelle eseguite da sedi di vendita al minuto. Questo permette di risolvere il problema precedente in quanto per ciascuna transazione viene individuato uno e un solo venditore.



Parte Seconda

Progettazione Logica

Nota sul formalismo adottato

Per gli schemi Entity/Relationship vengono adottate le stesse convenzioni riassunte all'inizio della parte prima.

Gli schemi relazionali sono così denotati:

NOME_RELAZIONE (nomeAttr1, nomeAttr2, ...)

Gli attributi facenti parte della chiave primaria vengono sottolineati. Qualora interessi evidenziare l'esistenza di un vincolo di integrità referenziale, si utilizzerà la seguente notazione:

R1 (chiavel, attr1)
R2 (chiave2, attr2, chiavel:R1)

a indicare che l'attributo chiavel in R2 è chiave importata (*foreign key*) da R1. Se la chiave importata è composta, si scriverà:

R1 (chiavel1, chiavel2, attr1)
R2 (chiave2, attr2, (chiavel1, chiavel2):R1)

Nell'ambito dello studio della normalizzazione, una dipendenza funzionale del tipo: "la combinazione di a_1, a_2, \dots, a_n determina b " verrà denotata con

$a_1, a_2, \dots, a_n \rightarrow b$

□ Esercizio 2.1

Sono dati i seguenti schemi relazionali, utilizzati per la gestione di un laboratorio:

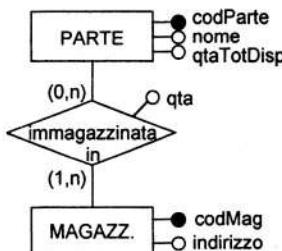
COMPUTER (codComputer, marca, modello, indirizzoFornitore)
INSTALLAZIONI (codComputer:COMPUTER, codSoftw, descrizSoftw,
dataInstallaz)

(esistono più computer della stessa marca e modello, differenziati solo dal codice; i computer della stessa marca - ad esempio IBM - hanno tutti lo stesso fornitore; uno stesso sw è in genere installato su più computer)

Si evidenzino le dipendenze funzionali presenti nei due schemi. Se uno o entrambi gli schemi risultano non normalizzati, si determini un insieme di schemi che siano in terza forma normale e risultino equivalenti, dal punto di vista informativo, agli schemi dati.

□ Esercizio 2.2

È dato lo schema E/R in figura.



dove qtaTotDisp è un attributo derivato che conta, per ciascuna parte, il numero di esemplari totali nei vari magazzini. I volumi di dati presunti sono:

entità PARTE: 600 istanze

entità MAGAZZINO: 20 istanze

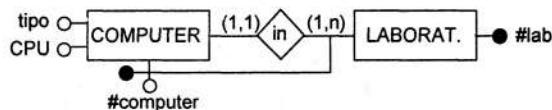
associazione IMMAGAZZINATA IN: 1200 istanze

Effettuare il progetto logico, valutando in particolare l'opportunità di memorizzare l'attributo derivato qtaTotDisp. A tale scopo, si consideri un carico di lavoro costituito dalle tre operazioni seguenti:

- q1: dato un codice di parte, determinazione della quantità totale disponibile per quella parte (10 volte al giorno)
- q2: dato un codice magazzino e un codice parte, carico nel magazzino di una data quantità di esemplari della parte (1 volta ogni 5 giorni)
- q3: dato un codice magazzino e un codice parte, scarico dal magazzino di una data quantità di esemplari della parte (1 volta al giorno)

□ Esercizio 2.3

Si discuta il progetto logico dello schema concettuale rappresentato in figura.



□ Esercizio 2.4

È dato il seguente schema relazionale, utilizzato per la gestione di un circo:

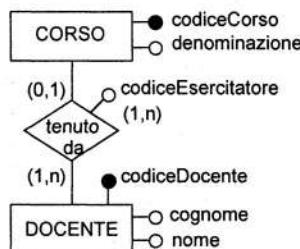
ZOO (codEsemplare, genere, gabbia, codAddetto, nomeAddetto,
giornoPuliziaGabbia, giornoSpettacolo, oraSpettacolo)

(ogni gabbia è accudita da un solo addetto; un addetto accudisce più gabbie; in una gabbia possono essere rinchiusi più esemplari; a uno spettacolo partecipano più esemplari)

Si evidenzino tutte le dipendenze funzionali presenti nello schema. Se lo schema risulta non normalizzato, si determini un insieme di schemi che siano in terza forma normale e risultino equivalenti, dal punto di vista informativo, allo schema dato.

□ Esercizio 2.5

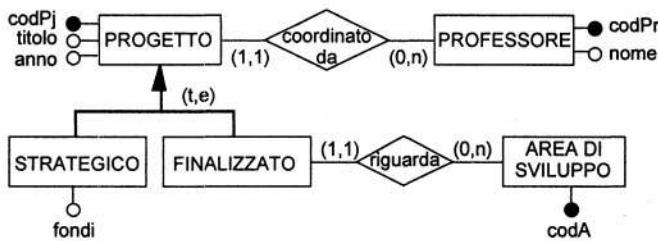
Si discuta il progetto logico relazionale dello schema concettuale rappresentato in figura.



□ Esercizio 2.6

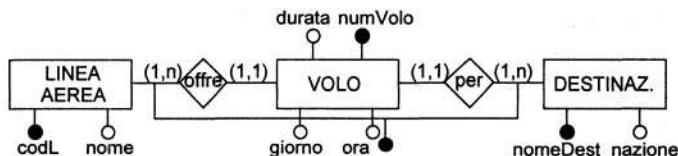
Discutere il progetto logico della porzione di schema E/R rappresentata in figura, considerando che:

- circa il 70% dei progetti è di tipo finalizzato;
- il carico di lavoro è composto al 40% da query che indirizzano insieme gli attributi titolo, anno, codA per i progetti finalizzati, al 40% da query che indirizzano insieme gli attributi titolo, anno, fondi per i progetti strategici, e al 20% da query che indirizzano insieme gli attributi titolo, anno, nome per tutti i progetti.



□ Esercizio 2.7

È dato lo schema E/R in figura.



Si effettui il progetto logico; delle seguenti query si disegni poi lo schema di navigazione e si scriva la formulazione SQL:

- a) data una destinazione ‘X’ e un giorno ‘Y’, si visualizzino i nomi delle linee aeree che effettuano voli per ‘X’ nel giorno ‘Y’;
- b) data una destinazione ‘X’, si calcoli la durata media di un volo per ‘X’.

□ Esercizio 2.8

Si consideri lo schema E/R proposto come soluzione dell’esercizio 1.9. Se ne effettui il progetto logico, considerando un volume dati e un carico di lavoro così costituiti:

compact disc: 5000
videocassette: 3000
clienti: 500

percentuale media di compact disc in prestito: 1%
percentuale media di videocassette in prestito: 0.5%

- a) ricerca dei clienti che hanno almeno un prestito scaduto (2 volte al giorno);
- b) nuovo prestito di uno o più titoli, previa verifica di disponibilità di una copia del titolo e di validità della tessera del cliente, previo controllo sul punteggio raggiunto dal cliente e sui prestiti scaduti del cliente (40 volte al giorno);
- c) restituzione di uno o più titoli (40 volte al giorno);
- d) dato un autore e il titolo di una canzone, ricerca del titolo del compact disc corrispondente (1 volta al giorno);
- e) dato un genere e un attore, ricerca di un film appartenente al genere e interpretato dall'attore (5 volte al giorno).

□ Esercizio 2.9

Si consideri il seguente schema relazionale:

```
PROVE_LAB (codStudente, nomeStudente, codCorso, nomeCorso,  
codTitolare, nomeTitolare, codEsaminatore,  
nomeEsaminatore, dataProva, voto)
```

(Vengono registrate anche eventuali prove non superate. Un corso ha un solo professore titolare, che non coincide necessariamente con il professore esaminatore). Si evidenzino tutte le dipendenze funzionali non banali presenti nello schema. Se lo schema risulta non normalizzato, si determini un insieme di schemi che siano in terza forma normale e risultino equivalenti, dal punto di vista informativo, allo schema dato.

Si disegni poi uno schema E/R da cui lo schema di database determinato potrebbe avere origine.

□ Esercizio 2.10

Si effettui il progetto logico dello schema E/R proposto come soluzione dell'esercizio 1.7.

□ Esercizio 2.11

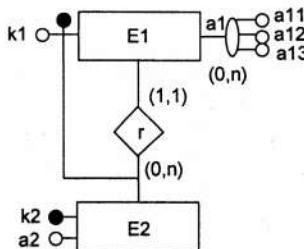
Si dica se la seguente relazione è normalizzata:

```
ESAMI (studente, corso, esaminatore, dataEsame, voto)
```

(si noti che un esaminatore può fare esami per un solo corso) e, in caso negativo, si determini un insieme di schemi normalizzati equivalenti.

□ Esercizio 2.12

Si effettui il progetto logico relazionale dello schema E/R in figura:



□ Esercizio 2.13

Dato lo schema $R(x, y, w, z, v)$ e le dipendenze funzionali

$$\begin{aligned}x &\rightarrow y \\y &\rightarrow w \\yz &\rightarrow vw\end{aligned}$$

si determini la chiave di R e si spieghi brevemente perché la decomposizione

$$\begin{aligned}R1 &\quad (x, y) \\R2 &\quad (y, w) \\R3 &\quad (y, z, w, v)\end{aligned}$$

non è in terza forma normale e perché non è corretta. Si fornisca quindi una decomposizione adeguata in schemi in terza forma normale e si disegni uno schema E/R equivalente.

□ Esercizio 2.14

Data la relazione $R(a, b, c, d, e, f, g, h, i)$ e le dipendenze funzionali

$$\begin{aligned}a &\rightarrow ei \\bd &\rightarrow h \\c &\rightarrow f\end{aligned}$$

normalizzare la relazione e disegnare uno schema E/R equivalente.

□ Esercizio 2.15

Dato il seguente schema relazionale:

CLASSICHE_CICLISMO (nomeClassica, cittàPartenza, cittàArrivo,
stagione, codCorridore, nomeCorridore, squadra,
posizione, direttoreSportivo,
tempoPrimoClassificato, distacco)

Sapendo che:

- ogni corridore corre, in una stagione, per una sola squadra
- ogni corridore può partecipare, nella stessa stagione, a più gare
- ogni squadra ha un solo direttore sportivo a stagione
- il risultato di ogni corridore in una gara è rappresentato tramite la posizione in classifica e il distacco dal primo classificato
- ogni classica ha un solo percorso che non varia di stagione in stagione
- ogni gara è corsa da più corridori ma ha associato un solo tempoPrimoClassificato

evidenziare tutte le dipendenze funzionali non banali presenti nella relazione e decomporre lo schema in terza forma normale.

□ Esercizio 2.16

È dato il seguente schema relazionale:

FATTURA (codProdotto, descrizioneProdotto, quantità, prezzo,
codCliente, nomeCliente, indirizzoCliente, data, codAgente,
nomeAgente, telAgente, prezzoTotale)

Sapendo che:

- un cliente è rifornito da un solo agente
- l'attributo prezzo rappresenta il costo totale relativo a un prodotto all'interno di una determinata fattura
- non possono essere emesse 2 fatture allo stesso cliente nello stesso giorno
- l'attributo prezzoTotale rappresenta l'importo totale della fattura

si evidenzino tutte le dipendenze funzionali non banali presenti nello schema. Se lo schema risulta non normalizzato, si determini un insieme di schemi che siano in terza forma normale e risultino equivalenti, dal punto di vista informativo, allo schema dato.

□ Esercizio 2.17

Data la seguente relazione:

VOLI (codVolo, codCompagnia, nomeCompagnia, giorno, ora, durataVolo, nomeDestinazione, nazioneDestinazione)

Sapendo che:

- ogni volo, identificato univocamente da un codice, è effettuato da una sola compagnia aerea
- una stessa compagnia aerea può offrire più voli per diverse destinazioni
- uno stesso volo può essere effettuato in orari diversi nel corso della settimana, ma la durata del volo non cambia

elencare tutte le dipendenze funzionali non banali presenti e decomporre lo schema in terza forma normale.

Esercizio 2.18

Si consideri il seguente schema:

COMPOSIZIONE (squadra, sede, indirizzoSede, cittàSede, stadio, indirizzoStadio, nomeAllenatore, tesseroAllenatore, nomeGiocatore, codGiocatore)

Sapendo che ciascun giocatore gioca per una sola squadra, che un allenatore può allenare una sola squadra, che una squadra gioca (in casa) in un solo stadio e ha una sola sede, elencare le dipendenze funzionali presenti nello schema e ricondurre lo schema in terza forma normale.

Esercizio 2.19

Dato il seguente schema relazionale:

BUDGET (codProgetto, nomeProgetto, sede, indirizzoSede, codRicercatore, nomeRicercatore, responsabileProgetto, capoProgetto, anno, budget)

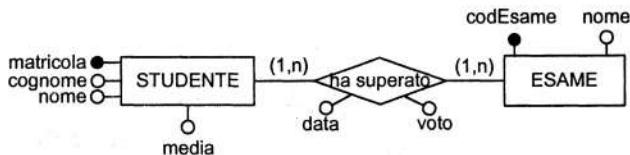
Sapendo che:

- ogni progetto può essere portato avanti su diverse sedi
- in ciascuna sede è nominato un responsabile per ciascun progetto
- ogni progetto ha un solo capo progetto
- ogni ricercatore lavora in una sede, ma può partecipare a più progetti, purché sulla stessa sede
- ogni progetto ha un budget annuale
- ogni sede ha un solo indirizzo

evidenziare tutte le dipendenze funzionali non banali presenti nello schema e decomporre lo schema in terza forma normale.

□ Esercizio 2.20

È dato lo schema E/R in figura



dove *media* è un attributo derivato che rappresenta, per ciascuno studente, la media dei voti ottenuti negli esami superati. I volumi di dati presunti sono:

entità STUDENTE: 1000 istanze

entità ESAME: 30 istanze

associazione HA SUPERATO: 8000 istanze

Effettuare il progetto logico, valutando in particolare l'opportunità di memorizzare l'attributo derivato *media*. A tale scopo, si consideri un carico di lavoro costituito dalle tre operazioni seguenti:

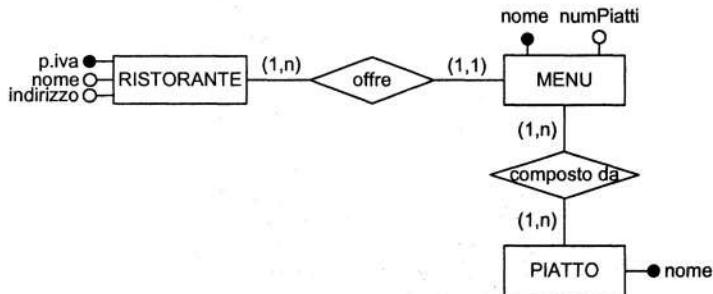
q1: iscrizione di un nuovo studente (20 volte al mese)

q2: data la matricola di uno studente, calcolo della sua media (100 volte al mese)

q3: data la matricola di uno studente e il codice di un esame, registrazione del voto ottenuto dallo studente in quel determinato esame (300 volte al mese).

□ Esercizio 2.21

È dato lo schema E/R in figura



dove *numPiatti* è un attributo derivato che rappresenta, per ciascun menù, il numero di piatti che lo compongono. I volumi di dati presunti sono:

entità RISTORANTE: 80 istanze

entità MENU: 320 istanze

entità PIATTO: 96 istanze

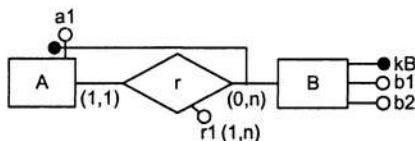
associazione COMPOSTO DA: 1920 istanze

Effettuare il progetto logico, valutando in particolare l'opportunità di memorizzare l'attributo derivato numPiatti. A tale scopo, si consideri un carico di lavoro costituito dalle tre operazioni seguenti:

- q1: inserimento di un nuovo piatto in un menù già esistente (1 volta al giorno)
- q2: ricerca dei menù offerti da un ristorante (100 volte al giorno)
- q3: dato il nome di un menù, visualizzazione del numero di piatti che lo compongono (10 volte al giorno).

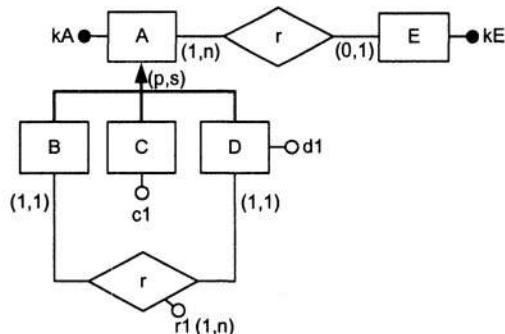
□ Esercizio 2.22

Si discuta il progetto logico dello schema concettuale rappresentato in figura.



□ Esercizio 2.23

Si discuta il progetto logico dello schema concettuale rappresentato in figura, giustificando la scelta per la traduzione della gerarchia.



✓ Esercizio 2.1

Sulla base dei requisiti assegnati è possibile dedurre le seguenti dipendenze funzionali non banali:

- nella relazione COMPUTER esiste una dipendenza transitiva:

marca → indirizzoFornitore

- nella relazione INSTALLAZIONI esiste una dipendenza parziale:

codSoftw → descrizSoftw

La relazione COMPUTER risulta essere in prima e seconda forma normale ma non in terza; la relazione INSTALLAZIONI in prima ma non in seconda (e, di conseguenza, nemmeno in terza). Per normalizzare COMPUTER occorre “spezzare” la dipendenza transitiva su due relazioni:

COMPUTER (codComputer, marca, modello)
FORNITORI (marca, indirizzoFornitore)

Per normalizzare INSTALLAZIONI si procede analogamente, “spezzando” la dipendenza parziale:

INSTALLAZIONI (codComputer, codSoftw, dataInstallaz)
SOFTWARE (codSoftw, descrizSoftw)

☞ Suggerimento:

Si risolva l'esercizio nel caso in cui computer della stessa marca possono essere forniti da più fornitori.

✓ Esercizio 2.2

Lo schema relazionale risultante dal progetto logico è il seguente:

PARTI (codParte, nome, qtaTotDisp)
MAGAZZINI (codMag, indirizzo)
IN (codParte:PARTI, codMag:MAGAZZINI, qta)

Al fine di valutare l'opportunità di mantenere l'attributo derivato `qtaTotDisp` occorre stimare il numero medio di accessi in lettura e scrittura legati al carico di lavoro.

- q1: In presenza di `qtaTotDisp`, ogni esecuzione di q1 provoca un solo accesso in lettura alla relazione `PARTI` (lettura della parte). In assenza di `qtaTotDisp`, verranno invece effettuati tanti accessi in lettura a `IN` quanti sono i magazzini in cui quella parte è immagazzinata. Dal volume dei dati si deduce che una parte è mediamente immagazzinata in $1200/600=2$ magazzini.
- q2: In assenza di `qtaTotDisp`, ogni esecuzione di q2 comporta soltanto l'aggiornamento di una tupla di `IN`, quindi un accesso in lettura e uno in scrittura. In presenza di `qtaTotDisp`, dovrà anche essere aggiornata una tupla in `PARTI`, per cui si avrà un ulteriore accesso in lettura e uno in scrittura.
- q3: Analogamente per q2.

Riepilogando, su base di cinque giorni si ha:

	frequenza	con <code>qtaTotDisp</code>	senza <code>qtaTotDisp</code>
q1	50x	1R	2R
q2	1x	2R+2W	1R+1W
q3	5x	2R+2W	1R+1W
		62R+12W	
		106R+6W	

Dal confronto tra i risultati ottenuti appare evidente la convenienza di mantenere l'attributo `qtaTotDisp` nella relazione `PARTI`.

☞ Suggerimento:

Determinare quale relazione ci deve essere tra le frequenze delle tre query affinché sia conveniente eliminare l'attributo `qtaTotDisp`.

✓ Esercizio 2.3

L'identificatore misto per l'entità `COMPUTER` si risolve importando l'identificatore dell'entità `LABORATORIO`:

`COMPUTER (#computer, #lab, CPU, tipo)`

Poiché l'entità `LABORATORIO` non ha attributi e ogni laboratorio contiene almeno un computer, non occorre realizzare una relazione `LABORATORIO`. Si noti che, se la cardinalità minima dell'associazione dal lato `LABORATORIO` fosse stata 0, la relazione `LABORATORIO` sarebbe stata necessaria per rappresentare i laboratori non contenenti alcun computer.

✓ Esercizio 2.4

Dall'analisi dei requisiti riportati si deducono le seguenti dipendenze funzionali non banali:

```
codEsemplare → genere  
codEsemplare → gabbia  
giornoSpettacolo → oraSpettacolo  
gabbia → giornoPuliziaGabbia  
gabbia → codAddetto  
codAddetto → nomeAddetto
```

Le prime tre dipendenze elencate sono di tipo parziale, mentre le rimanenti sono di tipo transitivo; la relazione ZOO non è quindi né in seconda né in terza forma normale. Spezzando le dipendenze parziali si ottengono tre relazioni in seconda forma normale:

```
SPETTACOLI (giornoSpettacolo, oraSpettacolo)  
PARTECIPAZIONI (codEsemplare, giornoSpettacolo)  
ZOO (codEsemplare, genere, gabbia, codAddetto, nomeAddetto,  
      giornoPuliziaGabbia)
```

SPETTACOLI e PARTECIPAZIONI sono normalizzate, mentre ZOO è in seconda ma non in terza forma normale. Per ottenere la terza forma normale occorre:

a) spezzare le due dipendenze legate a gabbia:

```
SPETTACOLI (giornoSpettacolo, oraSpettacolo)  
PARTECIPAZIONI (codEsemplare, giornoSpettacolo)  
ZOO (codEsemplare, genere, gabbia)  
GABBIE (gabbia, codAddetto, nomeAddetto, giornoPuliziaGabbia)
```

b) e quella legata a codAddetto:

```
SPETTACOLI (giornoSpettacolo, oraSpettacolo)  
PARTECIPAZIONI (codEsemplare, giornoSpettacolo)  
ZOO (codEsemplare, genere, gabbia)  
GABBIE (gabbia, codAddetto, giornoPuliziaGabbia)  
ADDETTI (codAddetto, nomeAddetto)
```

✓ Esercizio 2.5

Poiché l'attributo codiceEsercitatore è multiplo, è impossibile progettare l'associazione con due sole relazioni. Si introduce quindi la relazione ESERCITATORI, che rappresenta i collegamenti tra corsi ed esercitatori; la chiave è data dall'accoppiamento tra la chiave della relazione CORSI e l'attributo codiceEsercitatore.

```
CORSI (codiceCorso, denominazione, codiceDocente:DOCENTI)
DOCENTI (codiceDocente, cognome, nome)
ESERCITATORI (codiceCorso:CORSI, codiceEsercitatore)
```

Qualora si voglia evitare la presenza di valori nulli per l'attributo `codiceDocente` in `CORSI`, si ricorrerà a una soluzione con quattro relazioni:

```
CORSI (codiceCorso, denominazione)
TENUTI_DA (codiceCorso:CORSI, codiceDocente:DOCENTI)
DOCENTI (codiceDocente, cognome, nome)
ESERCITATORI (codiceCorso:CORSI, codiceEsercitatore)
```

Si noti che la soluzione

```
CORSI (codiceCorso, denominazione)
DOCENTI (codiceDocente, cognome, nome)
ESERCITATORI (codiceCorso:CORSI, codiceDocente:DOCENTI,
codiceEsercitatore)
```

non è normalizzata, poiché `ESERCITATORI` contiene la dipendenza funzionale parziale

`codiceCorso` → `codiceDocente`

✓ Esercizio 2.6

Il punto focale dell'esercizio è la progettazione della gerarchia. Data la copertura totale ed esclusiva, sono possibili tre soluzioni: con una relazione (collasso sulla superentità), con due relazioni (collasso sulle sottoentità) e tre relazioni (trasformazione tramite associazioni).

La preponderanza (80%) di query che indirizzano congiuntamente attributi della superentità e delle sottoentità ci spinge ad abbandonare la soluzione con tre relazioni, utilizzando la quale si dovrebbe accedere per ogni query a due relazioni. D'altra parte, la scarsità di query estese all'insieme di tutti i progetti (20%) rende preferibile la soluzione con due relazioni, vantaggiosa per le query eseguite separatamente su progetti dell'uno o dell'altro tipo.

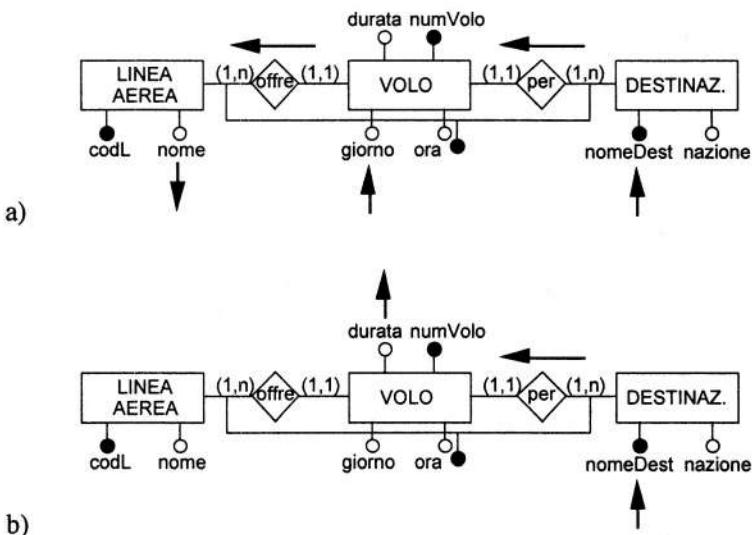
Lo schema logico risultante sarà quindi:

```
PROG_STRAT (codPi, titolo, anno, codPr:PROFESSORI, fondi)
PROG_FIN (codPi, titolo, anno, codPr:PROFESSORI, codA:AREE)
PROFESSORI (codPr, nome)
AREE (codA)
```

Si noti che la relazione `AREE` è stata prevista, nonostante l'assenza di attributi oltre alla chiave, poiché esistono aree non associate ad alcun progetto finalizzato.

✓ Esercizio 2.7

Gli schemi di navigazione delle due query sono i seguenti:



Il progetto logico viene effettuato importando sulla relazione VOLI le chiavi delle relazioni LINEE e DESTINAZIONI:

```
LINEE (codL, nome)
DESTINAZIONI (nomeDest, nazione)
VOLI (numVolo, codL:LINEE, nomeDest:DESTINAZIONI, giorno, ora,
durata)
```

Come chiave primaria di VOLI è stato scelto l'identificatore interno numVolo .

Le formulazioni SQL delle due query sono:

a)

```
SELECT DISTINCT LINEE.nome
FROM LINEE, VOLI
WHERE LINEE.codL = VOLI.codL
AND VOLI.nomeDest = 'X'
AND VOLI.giorno = 'Y'
```

b)

```
SELECT AVG(durata)
FROM VOLI
WHERE nomeDest = 'X'
```

Suggerimento:

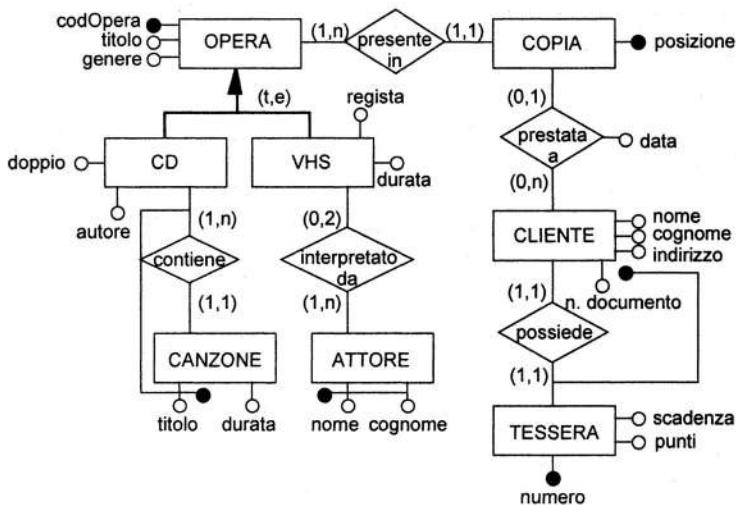
Scrivere la query SQL che, per ogni destinazione per cui la linea 'X' effettua almeno due voli, restituisce la durata media dei voli effettuati da tutte le linee per quella destinazione.

✓ Esercizio 2.8

Riproponiamo, per comodità del lettore, lo schema E/R di partenza.

Cominciamo il progetto logico esaminando l'associazione 1-1 tra CLIENTE e TESSERA. Data la molteplicità, risulta possibile effettuare la traduzione tramite un'unica relazione. D'altronde, l'accesso ai clienti per le operazioni di prestito e restituzione avverrà unicamente sulla base del numero di tessera; i dati di interesse per la funzionalità (b) sono la scadenza della tessera e i punti raggiunti, mentre i dati personali del cliente verranno presumibilmente utilizzati solo per operazioni sporadiche di spedizione di posta. Può quindi convenire modellare l'associazione possiede e le entità CLIENTE e TESSERA tramite due relazioni distinte:

```
TESSERE (numTessera, scadenzaTess, puntiTess)
CLIENTI (numTessera:TESSERE, nome, cognome, indirizzo, numDoc)
```



Da notare le possibili scelte di gestione conseguenti a questa soluzione:

- Alla scadenza di una tessera, i dati sulla tessera e sul cliente possono essere cancellati. Se il cliente rinnova la tessera, i suoi dati dovranno essere reinseriti.
- Alla scadenza di una tessera, i dati relativi vengono conservati (la tessera è comunque riconoscibile come scaduta grazie all'attributo scadenzaTess). I dati sul cliente restano disponibili, ad esempio per l'invio di materiale

- informativo. Se il cliente rinnova la tessera, gli viene attribuito lo stesso numero di tessera avuto in precedenza, e l'attributo scadenzaTess viene aggiornato.
- Per evitare di ingrossare l'archivio con dati inutili relativi a tessere scadute, è possibile disaccoppiare logicamente le chiavi di CLIENTI e TESSERE. In questo caso, a tessera scaduta, diventa possibile cancellare i soli dati sulla tessera.

Per quanto riguarda l'associazione 1-n prestata a, è possibile una traduzione con due relazioni. Data però la forte incidenza dell'opzionalità risultante dal carico di lavoro, si preferisce le traduzione con tre relazioni (la terza relazione è CLIENTI):

```
COPIE (posizione)
PRESTITI (posizioneCopia:COPIE, numTessera:TESSERE, data)
```

Veniamo al progetto della gerarchia di specializzazione. Si presume che la maggior parte degli accessi verrà effettuata separatamente per compact e videocassette; conviene quindi collassare la superentità sulle sottoentità:

```
CD (codOpera, titolo, genere, doppio, autore)
CANZONI (titoloCanz, codOpera:CD, durata)
VHS (codOpera, titolo, genere, regista, durata, primoAttore,
      secondoAttore)
```

L'associazione contiene viene rappresentata dall'identificatore misto di CANZONI, determinato importando la chiave di CD su CANZONI. Poiché l'entità ATTORE non ha altri attributi oltre all'identificatore, i due attori principali si inseriscono direttamente nella relazione VHS (primoAttore e secondoAttore sono campi stringa definiti dalla concatenazione di nome e cognome).

Resta da modellare l'associazione presente in. Trattandosi di un'associazione 1-n, si potrà importare la chiave di OPERA su COPIE; poiché OPERA è stata eliminata, occorrerà aggiungere un qualificatore tipoOpera che codifichi la natura dell'opera (compact o videocassetta):

```
COPIE (posizione, codOpera: $\pi_{codOpera}(CD) \cup \pi_{codOpera}(VHS)$ , tipoOpera)
```

✓ Esercizio 2.9

Dall'analisi dei requisiti si deducono le seguenti dipendenze funzionali non banali:

```
codStudente → nomeStudente
codCorso → nomeCorso
codCorso → codTitolare
codTitolare → nomeTitolare
codEsaminatore → nomeEsaminatore
```

La prime tre dipendenze elencate sono di tipo parziale, mentre le rimanenti sono di tipo transitivo; la relazione PROVE_LAB non è quindi né in seconda né in terza forma normale. Spezzando le dipendenze parziali si ottengono due relazioni in seconda (ma non in terza) forma normale:

PROVE_LAB (codStudente, codCorso, codEsaminatore,
 nomeEsaminatore, dataProva, voto)
 STUDENTI (codStudente, nomeStudente)
 CORSI (codCorso, nomeCorso, codTitolare, nomeTitolare)

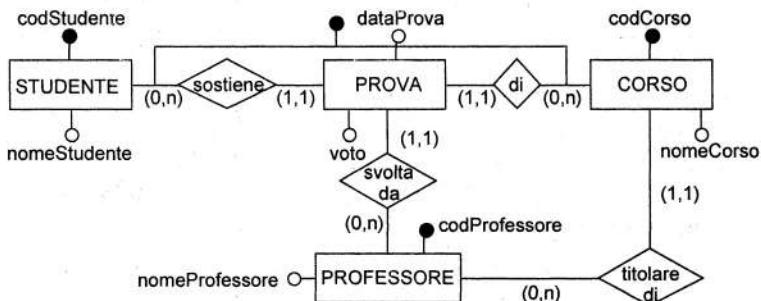
Per ottenere la terza forma normale occorre spezzare le due dipendenze transitive:

PROVE_LAB (codStudente, codCorso, codEsaminatore, dataProva,
 voto)
 PROFESSORI (codProfessore, nomeProfessore)
 STUDENTI (codStudente, nomeStudente)
 CORSI (codCorso, nomeCorso, codTitolare)

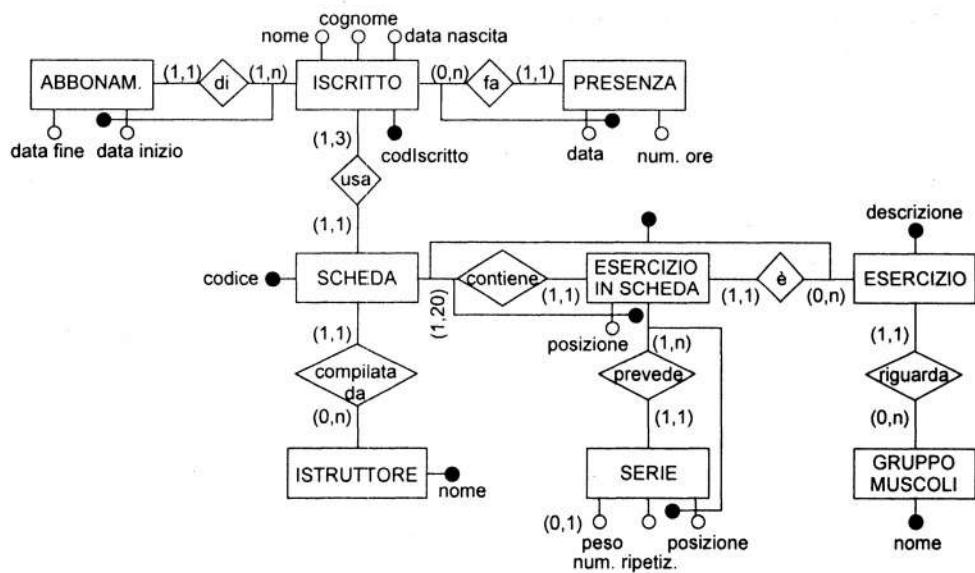
Uno schema E/R corrispondente allo schema di database dato è mostrato nella figura successiva. Si noti che il concetto di prova non può essere correttamente rappresentato tramite un'associazione ternaria tra STUDENTE, CORSO e una terza entità TEMPO, poiché risulterebbe impossibile modellare l'associazione svolta da con PROFESSORE. Ricorrendo invece a una associazione quaternaria tra STUDENTE, CORSO, TEMPO e PROFESSORE si perderebbe la rappresentazione della dipendenza funzionale

codStudente, codCorso, dataProva → codEsaminatore

che deriva dalla relazione originaria.



✓ Esercizio 2.10



In assenza di informazioni sul carico di lavoro, il progetto logico può essere effettuato seguendo criteri di massima.

```

ISCRITTI (codIscritto, nome, cognome, dataNascita)
ABBONAM (codIscritto:ISCRITTI, dataInizio, dataFine)
PRESENZE (codIscritto:ISCRITTI, data, numOre)
SCHEDE (codScheda, nomeIstruttore:ISTRUTORI,
        codIscritto:ISCRITTI)
ISTRUTORI (nomeIstruttore)
ESERCIZI_IN_SCHEDE (codScheda:SCHEDA,
                     descrizioneEsercizio:ESERCIZI, posizione)
SERIE (codScheda:ESERCIZI_IN_SCHEDE,
       posizioneInScheda:ESERCIZI_IN_SCHEDE,
       posizioneInEsercizio, peso, numeroRipetizioni)
ESERCIZI (descrizioneEsercizio, nomeGruppo:GRUPPI_MUSCOLI)
GRUPPI_MUSCOLI (nomeGruppo)

```

Come chiave primaria della relazione **ESERCIZI_IN_SCHEDE** è stato scelto uno dei due identificatori proposti dallo schema E/R. Le entità **ISTRUTORI** e **GRUPPO_MUSCOLI** sono state mantenute, nonostante l'assenza di attributi descrittivi, in virtù dell'opzionalità nella loro partecipazione alle associazioni **compilata da** e **riguarda**, rispettivamente.

✓ Esercizio 2.11

Il requisito secondo cui un esaminatore fa esami per un solo corso si traduce nella dipendenza funzionale

esaminatore → corso

La relazione è in terza forma normale poiché corso è un attributo primo (cioè è parte di una chiave), ma non è in forma normale di Boyce-Codd poiché esaminatore non è superchiave. Una possibile soluzione è costituita dagli schemi

ESAMI (studente, esaminatore, dataEsame, voto)
CORSI (esaminatore, corso)

Le due relazioni ottenute sono normalizzate; risulta però impossibile verificare la dipendenza funzionale

studente,corso → esaminatore

su una qualsiasi di esse. In altri termini, l'inserimento di due esami sostenuti dallo stesso studente per lo stesso corso con esaminatori diversi non può essere evitato a meno di fare riferimento contemporaneamente a entrambe le relazioni.

✓ Esercizio 2.12

La chiave di E1 è costituita dall'unione di k1 e della chiave di E2; per la memorizzazione dell'attributo composto, opzionale e multiplo a1 si introduce una relazione A1 la cui chiave è l'unione della chiave di E1 e dei tre attributi semplici che costituiscono a1:

E1 (k1, k2:E2)
A1 (k1:E1, k2:E1, a11, a12, a13)
E2 (k2, a2)

✓ Esercizio 2.13

La chiave di R è l'accoppiata x,z. Infatti, considerando le dipendenze funzionali assegnate, è vero che

xz → ywv

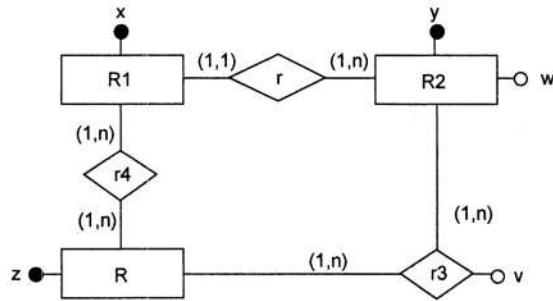
La decomposizione descritta nel testo non è in seconda (e quindi nemmeno in terza) forma normale poiché lo schema R3 contiene la dipendenza funzionale parziale

y → w

Non è corretta poiché non esprime il vincolo di integrità legato alla chiave di R. La decomposizione corretta è quindi la seguente:

- R1 (x, y)
 R2 (y, w)
 R3 (y, z, v)
 R4 (x, z)

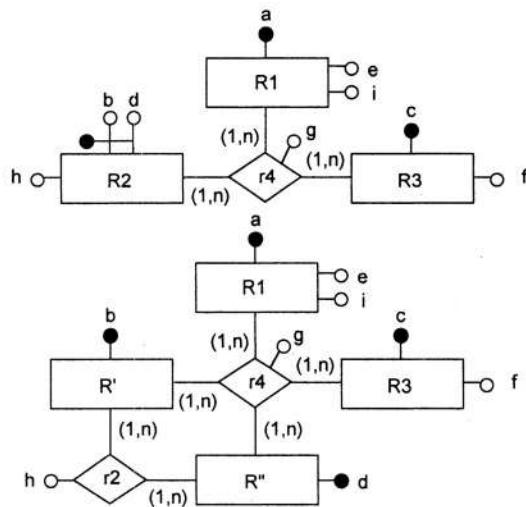
Uno schema E/R da cui lo schema relazionale indicato potrebbe avere origine è mostrato in figura.



✓ Esercizio 2.14

Lo schema relazionale normalizzato e due schemi E/R equivalenti sono riportati di seguito.

- R1 (a, e, i)
 R2 (b, d, h)
 R3 (c, f)
 R4 (a, b, c, d, g)



✓ Esercizio 2.15

Dall'analisi dei requisiti si deducono le seguenti dipendenze funzionali non banali:

```
codCorridore → nomeCorridore  
codCorridore, stagione → squadra  
squadra, stagione → direttoreSportivo  
codCorridore, nomeClassica, stagione → posizione  
codCorridore, nomeClassica, stagione → distacco  
nomeClassica → cittàPartenza  
nomeClassica → cittàArrivo  
nomeClassica, stagione → tempoPrimoClassificato
```

Nella relazione sono presenti sei dipendenze di tipo parziale; la relazione CLASSICHE_CICLISMO non è né in seconda né in terza forma normale. Spezzando le dipendenze parziali si ottengono sei relazioni che, non essendo presenti dipendenze transitive, sono in terza forma normale:

```
CORRIDORE (codCorridore, nomeCorridore)  
CORRIDORE_SQUADRA (codCorridore, stagione, squadra)  
DIRETTORE_SPORTIVO (squadra, stagione, direttoreSportivo)  
CLASSIFICA (nomeClassica, stagione, codCorridore, posizione,  
distacco)  
CLASSICA (nomeClassica, cittàPartenza, cittàArrivo)  
GARA (nomeClassica, stagione, tempoPrimoClassificato)
```

✓ Esercizio 2.16

Le dipendenze funzionali non banali presenti nello schema sono:

```
codProdotto → descrizioneProdotto  
codCliente → nomeCliente  
codCliente → indirizzoCliente  
codCliente → codAgente  
codAgente → nomeAgente  
codAgente → telAgente  
codCliente, data → prezzoTotale  
codCliente, data, codProdotto → quantità  
codCliente, data, codProdotto → prezzo
```

Nello schema sono presenti sia dipendenze di tipo parziale che di tipo transitivo; la relazione FATTURA non è quindi né in seconda né in terza forma normale. Spezzando le dipendenze parziali si ottengono quattro relazioni in seconda forma normale:

```
PRODOTTO (codProdotto, descrizioneProdotto)  
CLIENTE (codCliente, nomeCliente, indirizzoCliente, codAgente,  
nomeAgente, telAgente)  
FATTURA (codCliente, data, prezzoTotale)  
DETTAGLIO_FATTURA (codCliente, data, codProdotto, quantità, prezzo)
```

Per ottenere uno schema in terza forma normale occorre spezzare la dipendenza transitiva:

PRODOTTO (codProdotto, descrizioneProdotto)
CLIENTE (codCliente, nomeCliente, indirizzoCliente, codAgente)
AGENTE (codAgente, nomeAgente, telAgente)
FATTURA (codCliente, data, prezzoTotale)
DETTAGLIO_FATTURA (codCliente, data, codProdotto, quantità, prezzo)

✓ Esercizio 2.17

Le dipendenze funzionali non banali presenti nello schema sono:

codVolo → durataVolo
codVolo → codCompagnia
codVolo → nomeDestinazione
nomeDestinazione → nazioneDestinazione
codCompagnia → nomeCompagnia

Nello schema sono presenti sia dipendenze di tipo parziale che di tipo transitivo; la relazione VOLI non è quindi né in seconda né in terza forma normale. Spezzando sia le dipendenze parziali che quelle transitive si ottengono quattro relazioni in terza forma normale:

VOLO (codVolo, durataVolo, codCompagnia, nomeDestinazione)
COMPAGNIA (codCompagnia, nomeCompagnia)
DESTINAZIONE (nomeDestinazione, nazioneDestinazione)
ORARIO (codVolo, giorno, ora)

✓ Esercizio 2.18

Dall'analisi dei requisiti si deducono le seguenti dipendenze funzionali non banali:

squadra → sede
squadra → stadio
sede → indirizzoSede
sede → cittàSede
codGiocatore → nomeGiocatore
codGiocatore → squadra
tesserinoAllenatore → nomeAllenatore
tesserinoAllenatore → squadra

In questo schema la chiave della relazione è semplice e non possono ovviamente esistere dipendenze di tipo parziale; la relazione COMPOSIZIONE è quindi in seconda forma normale ma non in terza a causa delle dipendenze transitive sopra evidenziate. Spezzando le dipendenze si ottengono tre relazioni in terza forma normale:

SQUADRA (squadra, sede, stadio)
SEDE (sede, indirizzoSede, cittàSede)
GIOCATORE (codGiocatore, nomeGiocatore, squadra)

✓ Esercizio 2.19

Dall'analisi dei requisiti si deducono le seguenti dipendenze funzionali non banali:

$\text{codProgetto} \rightarrow \text{nameProgetto}$
 $\text{codProgetto} \rightarrow \text{capoProgetto}$
 $\text{sede} \rightarrow \text{indirizzoSede}$
 $\text{codRicercatore} \rightarrow \text{sede}$
 $\text{codRicercatore} \rightarrow \text{nameRicercatore}$
 $\text{codProgetto}, \text{sede} \rightarrow \text{responsabileProgetto}$
 $\text{codProgetto}, \text{anno} \rightarrow \text{budget}$

Lo schema non è né in seconda né in terza forma normale in quanto esistono sia dipendenze parziali che transitive. Spezzando le dipendenze esistenti si ottiene il seguente schema in terza forma normale:

PROGETTO (codProgetto, nomeProgetto, capoProgetto)
SEDE (sede, indirizzoSede)
RICERCATORE (codRicercatore, nomeRicercatore, sede)
RESPONSABILE(codProgetto, sede, responsabileProgetto)
PARTECIPAZIONE (codProgetto, codRicercatore)
BUDGET (codProgetto, anno, budget)

✓ Esercizio 2.20

Al fine di valutare l'opportunità di mantenere l'attributo derivato media occorre stimare il numero medio di accessi in lettura e scrittura legati al carico di lavoro.

- q1: Indipendentemente dalla presenza di media, ogni esecuzione di q1 provoca un solo accesso in scrittura alla relazione STUDENTE.
- q2: In presenza di media, ogni esecuzione di q2 comporta un accesso in lettura alla relazione STUDENTE, mentre in assenza di media verranno effettuati tanti accessi in lettura a HA SUPERATO quanti sono gli esami che lo studente ha superato. Dal volume dei dati si deduce che uno studente ha superato mediamente $8000/1000=8$ esami.
- q3: In assenza di media è necessario soltanto l'inserimento di una tupla in IN, quindi un accesso in scrittura. In presenza di media, dovrà anche essere aggiornata una tupla in STUDENTE, per cui si avrà un ulteriore accesso in lettura e uno in scrittura a STUDENTE ai quali si devono aggiungere 8 accessi in lettura a HA SUPERATO per il ricalcolo della media.

Riepilogando, su base mensile si ha:

	frequenza	con media	senza media
q1	20x	1W	1W
q2	100x	1R	8R
q3	300x	9R+2W	1W
		2800R+620W	
		800R+320W	

Dal confronto tra i risultati ottenuti appare evidente la convenienza di non mantenere l'attributo `media` nella relazione STUDENTE.

✓ Esercizio 2.21

Al fine di valutare l'opportunità di mantenere l'attributo derivato `numPiatti` occorre stimare il numero medio di accessi in lettura e scrittura legati al carico di lavoro.

- q1: In presenza di `numPiatti`, ogni esecuzione di q1 comporta un accesso in scrittura alla relazione PIATTO, un accesso in scrittura a COMPOSTO DA e un accesso in lettura e uno in scrittura a MENU per l'aggiornamento del numero di piatti. In assenza di `numPiatti` invece è necessario un solo accesso in scrittura a PIATTO e uno, sempre in scrittura, all'associazione COMPOSTO DA.
- q2: Indipendentemente dalla presenza di `numPiatti`, ogni esecuzione di q2 comporta tanti accessi in lettura a MENU quanti sono in media i menù offerti da un ristorante. Dal volume dei dati si deduce che un ristorante offre in media $320/80=4$ menù.
- q3: In presenza di `numPiatti` è necessario soltanto un accesso in lettura all'entità MENU. In assenza di `numPiatti` saranno necessari tanti accessi in lettura all'associazione COMPOSTO DA quanti sono mediamente i piatti presenti in un menù. Dal volume dei dati si deduce che un menù è composto in media da $1920/320=6$ piatti.

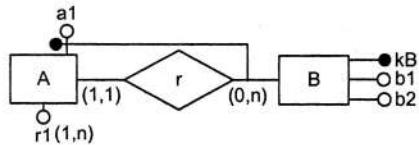
Riepilogando, su base giornaliera si ha:

	frequenza	con numPiatti	senza numPiatti
q1	1x	1R+3W	2W
q2	100x	4R	4R
q3	10x	1R	6R
			411R+3W
			460R+2W

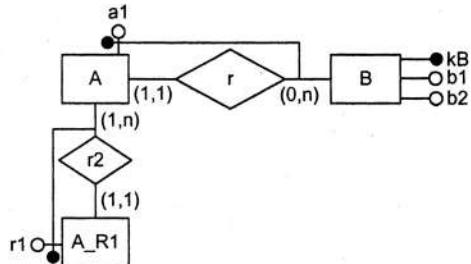
Dal confronto tra i risultati ottenuti risulta più conveniente mantenere l'attributo `numPiatti` nella relazione MENU.

✓ Esercizio 2.22

L'aspetto più problematico di questo schema è legato alla presenza dell'attributo multiplo `r1`. Osservando che l'associazione `r` è di tipo 1-n e in particolare che ciascuna istanza di A è associata a una e una sola istanza di B, `r1` può essere considerato come attributo dell'entità A. Riportiamo lo schema E/R risultante per maggiore chiarezza.



L'attributo multiplo può ora essere tradotto introducendo una nuova entità, che chiameremo A_R1 , che conterrà gli n valori di $r1$ associati a ciascuna istanza di A . La soluzione è mostrata nello schema seguente:



Ora restano da tradurre le due associazioni r e $r2$. Trattandosi in entrambi i casi di associazioni 1-n, si possono tradurre importando la chiave di B in A e quella di A in A_R1 rispettivamente.

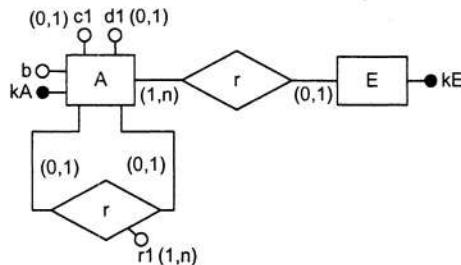
Lo schema relazionale risultante dal progetto logico è il seguente:

```

B (kB, b1, b2)
A (kB:B, a1)
A_R1 ((kB,a1):A, r1)
  
```

✓ Esercizio 2.23

Affrontiamo innanzitutto la progettazione della gerarchia. Data la copertura parziale, la traduzione tramite collasso sulle sottoentità non risulta praticabile in quanto comporterebbe una perdita d'informazione. Inoltre, anche in presenza di copertura totale questa soluzione sarebbe comunque sconsigliabile a causa della sovrapposizione tra le istanze delle tre sottoentità che causerebbe una ridondanza dei dati memorizzati. La soluzione più ragionevole è quella con una sola relazione (collasso sulla superentità). Lo schema risultante dopo l'eliminazione della gerarchia è riportato nella figura seguente.



Essendo la copertura di tipo parziale e sovrapposto sarebbe concettualmente necessario introdurre tre selettori, ciascuno dei quali indica l'appartenenza o meno a una delle tre sottoentità B, C e D; poiché le ultime due entità presentano attributi specifici (c_1 e d_1 rispettivamente), possiamo utilizzare questi come selettori e aggiungerne un altro per l'entità B (attributo b). L'associazione r che legava le entità B e D diventa ora un'associazione unaria sull'entità A con partecipazione opzionale.

Passiamo ora all'analisi dell'attributo multiplo $r1$. L'associazione r può essere tradotta con una sola relazione (A). All'entità A associamo anche l'attributo multiplo $r1$ che può poi essere tradotto introducendo una nuova entità che chiameremo A_R1 .

Lo schema relazionale risultante dal progetto logico è il seguente:

E (kE, kA:A)
A (kA, b, c1, d1, kA1:A)
A_R1 (kA:A, r1)

Si noti che la soluzione proposta causa una ridondanza dei dati memorizzati nella relazione A. Ciascuna coppia (k_A, k_{A1}) , che modella l'associazione ad anello, unitamente agli attributi b , c_1 e d_1 , viene infatti ripetuta due volte: una quando la chiave della tupla è k_A e l'altra quando chiave è k_{A1} . Una possibile soluzione è quella di attribuire un ruolo all'attributo all'interno dell'entità, rendendo così non significativa la permutazione dei valori. In questo caso però, non essendovi una semantica associata allo schema, la soluzione prospettata non è percorribile, rendendo necessaria l'introduzione di un vincolo relativo ai valori ammissibili per le tuple di A: $k_A \cdot k_{A1}$. Una delle due tuple violerà certamente il vincolo e non potrà essere inserita nella relazione, evitando così inutili ripetizioni.

Parte Terza

SQL e Algebra Relazionale

Nota sul formalismo adottato

Esistono numerose varianti del linguaggio SQL, alcune standardizzate (ad esempio SQL2, T-SQL), altre legate a specifiche realizzazioni su DBMS commerciali. Poiché nell'ambito degli esercizi proposti non risultano necessari costrutti "speciali", come quelli per la formulazione di outer join e di interrogazioni temporali, né si ritiene conveniente legarsi ai "dialetti" proposti dai diversi DBMS, nel seguito verrà utilizzato esclusivamente l'SQL standard.

Per quanto concerne l'algebra relazionale, si ricorda brevemente la notazione universalmente adottata per i quattro operatori relazionali utilizzati negli esercizi:

$\pi_{\text{attr}}(R)$	(proiezione della relazione R sull'attributo attr)
$\sigma_{\text{pred}}(R)$	(selezione della relazione R in base al predicato pred)
$R_1 \bowtie R_2$	(join naturale tra le relazioni R1 e R2)
$R_1 : R_2$	(divisione tra le relazioni R1 e R2)

Infine, per la definizione degli schemi relazionali si adottano le stesse convenzioni su chiavi primarie e importate presentate nella Parte Seconda.

□ Esercizio 3.1

Sono dati i seguenti schemi relazionali:

```
IMPIEGATI (codImp, nomeImp, qualifica)
PROGETTI (codProg, nomeProg)
COLLABORA (codImp:IMPIEGATI, codProg:PROGETTI, mesiUomo)
```

Si scriva l'espressione di algebra relazionale equivalente alla seguente query SQL:

```
SELECT codImp
FROM COLLABORA
GROUP BY codImp
HAVING COUNT(*) =
( SELECT COUNT(*)
  FROM PROGETTI )
```

□ Esercizio 3.2

Sono date le relazioni:

```
COMPUTER (codComputer, marca:FORNITURE, modello)
INSTALLAZIONI (codComputer:COMPUTER, codSoftware, dataInstallaz)
FORNITURE (marca, nomeFornitore)
```

(una marca è fornita da un solo fornitore, un fornitore può fornire più marche)

Si scriva la query SQL che determina, per ogni marca fornita dal fornitore "Rossi", il numero di pacchetti software distinti installati durante l'anno 1994.

□ Esercizio 3.3

Sono dati i seguenti schemi relazionali:

```
IMPIEGATI (codImp, nomeImp, qualifica)
PROGETTI (codProg, nomeProg)
COLLABORA (codImp:IMPIEGATI, codProg:PROGETTI, mesiUomo)
```

Si scriva la query SQL che restituisce i nomi e le qualifiche degli impiegati che non collaborano ad alcun progetto.

□ Esercizio 3.4

Dati gli schemi

PROFESSORI (prof, città, dipartimento)
CORSI (corso, prof:PROFESSORI)
STUDENTI (studente, corso:CORSI)

esprimere in algebra relazionale la seguente interrogazione: *in quali città vivono i professori dei corsi seguiti dallo studente Rossi?*

□ Esercizio 3.5

Sono date le seguenti relazioni, che descrivono un insieme di farmaci, i principi attivi in essi contenuti, e le relazioni di incompatibilità tra questi ultimi:

FARMACI (codF, nome, prezzo)
PR_ATTIVI (codP, descr)
CONTIENE (codF:FARMACI, codP:PR_ATTIVI, qtà)
INCOMP (codP1:PR_ATTIVI, codP2:PR_ATTIVI, grado)

Per ogni incompatibilità tra due principi attivi, nella relazione INCOMP esiste solo una tupla, per la quale si ha codP1 < codP2.

- Scrivere la query SQL che restituisce, per ogni principio attivo, il minimo tra i prezzi dei farmaci che contengono di quel principio attivo una quantità pari almeno a 0.1 mg.
- Scrivere la query SQL che aumenta del 5% i prezzi di tutti i farmaci contenenti il principio attivo chiamato "salicilato".
- Scrivere, in algebra relazionale, un'espressione che denoti le coppie di descrizioni di principi attivi che risultano incompatibili con grado di incompatibilità superiore a 0.5.

□ Esercizio 3.6

Sono dati i seguenti schemi relazionali:

INCASSI (film, anno, incasso)
CAST (film:INCASSI, attore)

Si scriva la query SQL che restituisce, per ogni attore che ha partecipato nei vari anni al cast di almeno 2 film, il massimo incasso registrato da tutti i film a cui ha partecipato nell'anno 1995.

□ Esercizio 3.7

Sono date le seguenti relazioni:

```
FORNITORI (codFor, ragioneSociale, livello, sede)
PARTI (codParte, nomeParte, colore, peso, sede)
PROGETTI (codProg, nomeProg, sede)
FORPARPRO (codFor:FORNITORI, codParte:PARTI, codProg:PROGETTI,
quant)
```

Nella relazione FORPARPRO una generica tupla memorizza la quantità quant della parte codParte usata nel progetto codProg e fornita dal fornitore codFor.

Si scriva in SQL la query che restituisce tutte le coppie di sedi tali che un fornitore della prima ha rifornito un progetto della seconda.

□ Esercizio 3.8

Sono date le relazioni:

```
COMPUTER (codComputer, marca, modello)
INSTALLAZIONI (codComputer:COMPUTER, codSoftware:SOFTWARE,
dataInstall)
SOFTWARE (codSoftware, nome, tipo)
```

Si scriva la query SQL che restituisce i nomi dei software di tipo “Word Processor” che contano almeno 5 installazioni posteriori all’1/1/95 su computer di marca “Apple”.

□ Esercizio 3.9

R	A	B	C
a1	b1	c1	
a2	b1	c2	
a3	b2	c2	
a4	b3	c4	
a5	b3	c4	

S	D	B	E
d2	b1	e1	
d3	b1	e2	
d1	b2	e1	
d5	b3	e4	
d6	b4	e3	
d7	b2	e4	
d8	b4	e1	
d9	b5	e3	
d4	b5	e2	

Con riferimento alle relazioni R e S riportate sopra, si disegnino le relazioni risultanti dalle seguenti operazioni di algebra relazionale:

- a) $\pi_{A, B, E}(\sigma_{D > 'd5'}(R \bowtie S))$
 b) $\pi_{B, E}(S) : \pi_B(\sigma_{C < 'c3'}(R))$

□ Esercizio 3.10

Sono dati gli schemi relazionali

R1 (k1, a1, b1)
 R2 (k2, a2, b2)
 R3 (k3, a3, k1:R1)

Scrivere una query SQL equivalente alla seguente espressione relazionale:

$$\sigma_{b2 < 10}(\pi_{k1, k3, a1, a3}(R3 \bowtie (\sigma_{b1 > 100}(R1) \bowtie R2)))$$

□ Esercizio 3.11

È dato il seguente schema di database, relativo a un noleggio di videocassette:

CLIENTI (codCliente, nome, indirizzo, età)
 VHS (codCopia, codTitolo)
 TITOLI (codTitolo, titolo, genere, regista, durata)
 PRESTITI (codCliente:CLIENTI, codCopia:VHS, data)

- a) Si scriva la query SQL che restituisce nome e indirizzo dei clienti che hanno almeno tre prestiti in data anteriore al 7/6/1996.
- b) Si scriva la query SQL che cancella tutti i prestiti anteriori al 7/6/1996 per i soli film di genere “comico”.
- c) Si scriva la query SQL che restituisce, per ogni regista che ha girato almeno tre film di genere “avventuroso”, la durata media di tutti i film in catalogo (di qualunque genere).

□ Esercizio 3.12

Con riferimento allo schema relazionale assegnato nell'esercizio 3.11, si ottimizzi la seguente espressione algebrica:

$$\pi_{\text{nome}, \text{data}}(\sigma_{\text{data} > '1/1/95' \wedge \text{età} < 20}(\text{CLIENTI} \bowtie \text{PRESTITI}))$$

□ Esercizio 3.13

Sono dati i seguenti schemi relazionali che descrivono una base di dati relativa a un istituto bancario:

```
CLIENTI (codCli, nomeCli)
CONTI (numConto, saldo)
POSSIEDE (numConto:CONTI, codCli:CLIENTI)
TRANSAZIONI (noTrans, numConto:CONTI, data, tipo, ammontare)
```

- Si scriva la query SQL che mostra le transazioni dell'ultimo mese per i clienti che hanno saldo totale (somma dei saldi dei singoli conti) negativo.
- Si scrivano le query SQL necessarie a prelevare £ 1.000.000 dal conto n° 8456, supponendo che il saldo iniziale del conto sia sufficientemente elevato da permettere l'operazione.

□ Esercizio 3.14

Sono dati i seguenti schemi relazionali che descrivono una base di dati relativa a un ricettario:

```
RICETTE (nomeRicetta, descrizione, tempo, porzioni, istruzioni)
INGREDIENTI (nomeIngrediente, prezzoUnitario, calorie)
INGR_RICETTA (nomeRicetta:RICETTE, nomeIngrediente:INGREDIENTI,
                quantità, commento)
```

- Si scriva la query SQL che mostra i nomi delle ricette ordinate per costo unitario (relativo a una singola porzione).
- Si scriva la query SQL che mostra le descrizioni delle ricette contenenti l'ingrediente più calorico.

□ Esercizio 3.15

Date le due relazioni:

```
R1 (a, b)
R2 (a:R1, c)
```

si scriva una query SQL che restituisca le tuple di R2 che non soddisfano il vincolo di integrità referenziale sulla chiave importata R2.a.

□ Esercizio 3.16

Sono dati i seguenti schemi relazionali che descrivono una base di dati relativa a un istituto bancario:

```
CLIENTI (codCli, nomeCli)
CONTI (numConto, saldo)
POSSIEDE (numConto:CONTI, codCli:CLIENTI)
TRANSAZIONI (noTrans, numConto:CONTI, data, tipo, ammontare)
```

- a) Si scriva la query SQL che mostra, per i clienti che hanno un unico conto corrente, l'ammontare *totale* delle transazioni relative all'ultimo anno.
- b) Si scrivano la query SQL e l'espressione in algebra relazionale che selezionano i nomi dei clienti che hanno effettuato almeno una transazione superiore a £ 100.000.000.

□ Esercizio 3.17

Dati i seguenti schemi relazionali che descrivono una base di dati relativa a un elenco telefonico:

PREFISSI (località, prefisso)
 NUMERI (località:PREFISSI, numero, nomeAbbonato, indirizzo)

- a) Si scriva la query SQL per verificare che l'inserimento del nuovo numero telefonico 321473 relativo alla località di Cesena non violi il vincolo in base al quale non possono esistere due numeri telefonici uguali (prefisso+numero). Si ricorda infatti che più località possono avere lo stesso prefisso.
- b) Si scriva la query SQL che permette di inserire il nuovo numero associandolo all'abbonato "Rossi Paolo", "Via Lunga, 3".

□ Esercizio 3.18

Sono dati i seguenti schemi relazionali che descrivono una base di dati relativa a un aeroporto:

AEROPORTO (codice, nome, città)
 VOLO (noVolo, partDa:AEROPORTO, arrA:AEROPORTO, oraPart, oraArr, tipoAereo)

- a) Per volare dal Kennedy di New York (codice "JFK") al Marconi di Bologna (codice "BLQ") i passeggeri devono suddividere il viaggio in 2 parti. Si elenchino i nomi degli aeroporti in cui è possibile fare scalo.
- b) Sempre in relazione al problema precedente si elenchino le caratteristiche dei 2 voli necessari a connettere il Kennedy al Marconi, relativamente ai soli viaggi per i quali il tempo di attesa nell'aeroporto di scalo è inferiore alle 3 ore.

□ Esercizio 3.19

Data la relazione

R (a, b, c, d)

e la dipendenza funzionale:

a → c

si scrivano le relazioni normalizzate equivalenti a R e le query SQL necessarie per trasferire i dati da R nelle nuove relazioni.

□ Esercizio 3.20

Dati i seguenti schemi relazionali:

PERSONA (id, nome)
PADRE (idPadre:PERSONA, idFiglio:PERSONA)

- Si scriva la query SQL per elencare tutti i nomi dei nonni.
- Si scriva la query SQL che mostra, per ogni nonno, il numero dei suoi nipoti.

□ Esercizio 3.21

Dati i seguenti schemi relazionali:

CATEGORIA (idCategoria, nome)
PRODOTTO (idProdotto, nome, fornitore, idCategoria:CATEGORIA,
prezzo)
CLIENTE (idCliente, nome, indirizzo, città, nazione)
ORDINE (idOrdine, idCliente:CLIENTE, data)
DETTAGLIO_ORDINE (idOrdine, idProdotto:PRODOTTO, quantità)

- Si scriva la query SQL che mostra i nomi dei prodotti di categoria ‘bevande’ che non sono stati ordinati nel 2004.
- Si scriva la query SQL che mostra l’importo totale dell’ultimo ordine fatto dal cliente avente codice ‘1234’.
- Si scriva la query SQL che mostra le coppie di clienti della stessa città.
- Si scriva la query SQL che visualizza il nome dei prodotti che non sono mai stati ordinati da clienti canadesi o francesi.
- Si scriva la query SQL che visualizza il nome del cliente che ha fatto il maggior numero di ordini.
- Si scriva la query SQL per visualizzare gli ordini relativi a prodotti di una sola categoria.
- Selezionare, per ciascuna categoria, il nome e la quantità totale venduta del prodotto più venduto.
- Selezionare il nome dei prodotti che sono stati ordinati in quantità superiore alla media in almeno 20 ordini.

□ Esercizio 3.22

Dati i seguenti schemi relazionali:

IMPIEGATO (idImpiegato, cognome, nome, posizione, indirizzo,
città, superiore:IMPIEGATO)

```
ORDINE (idOrdine, idCliente:CLIENTE, idImpiegato:IMPIEGATO,  
        data)  
CLIENTE (idCliente, nome, indirizzo, città, nazione)
```

- a) Selezionare gli impiegati che occupano la posizione di 'direttore commerciale' e hanno almeno tre rappresentanti alle loro dipendenze.
- b) Selezionare i rappresentanti che hanno ricevuto ordini solo da clienti della loro città.
- c) Visualizzare, per ogni anno, i dati dell'impiegato che ha ricevuto il minor numero di ordini.
- d) Selezionare, per ogni impiegato, il numero di clienti distinti da cui ha ricevuto ordini.
- e) Selezionare il codice e il nome degli impiegati che non hanno ricevuto ordini dopo il 20/05/2002.
- f) Per ogni impiegato avente posizione di 'Direttore', selezionare i nomi dei rappresentanti alle sue dipendenze che hanno avuto un numero di ordini maggiore di 70, visualizzando il risultato in ordine decrescente in base al numero di ordini.

□ Esercizio 3.23

Dati i seguenti schemi relazionali:

```
CORRIERE (idCorriere, nomeCorriere, telefono)  
ORDINE (idOrdine, idCliente:CLIENTE, idCorriere:CORRIERE,  
        dataOrdine, dataSpedizione)  
CLIENTE (idCliente, cognome, nome, indirizzo, città, nazione)
```

- a) Selezionare per ogni cliente il numero di ordini non ancora consegnati.
- b) Selezionare i clienti che hanno ricevuto spedizioni da un solo corriere.
- c) Selezionare il nome dei corrieri che nell'anno 1998 hanno consegnato ordini a tutti i clienti.

□ Esercizio 3.24

Dati i seguenti schemi relazionali:

```
IMPIEGATO (idImpiegato, cognome, nome, posizione, indirizzo,  
           stipendio, superiore:IMPIEGATO)  
ORDINE (idOrdine, idCliente:CLIENTE, idImpiegato:IMPIEGATO,  
        data)  
PRODOTTO (idProdotto, nomeProdotto)  
DETTOGLIO_ORDINE (idOrdine:ORDINE, idProdotto, prezzo, quantità)  
CLIENTE (idCliente, cognome, nome, indirizzo)
```

- a) Si scriva l'espressione di algebra relazionale che seleziona il nome e cognome dei superiori che guadagnano più di 30000 € l'anno.

- b) Si scriva l'espressione di algebra relazionale che seleziona i dati dei clienti dell'impiegato con `idImpiegato=5`. Se possibile, ottimizzare l'espressione applicando le regole di equivalenza per le espressioni di algebra relazionale.
- c) Si scriva la query SQL che seleziona, per ogni cliente, il numero di ordini effettuati in ciascun anno.

□ Esercizio 3.25

Dati i seguenti schemi relazionali:

```
IMPIEGATO (idImpiegato, cognome, nome, superiore:IMPIEGATO)
ORDINE (idOrdine, idCliente:CLIENTE, idImpiegato:IMPIEGATO,
         data, idCorriere:CORRIERE)
CORSIERE (idCorriere, nomeCorriere, telefono)
CLIENTE (idCliente, cognome, nome, indirizzo)
```

- a) Si scriva l'espressione di algebra relazionale equivalente alla seguente query SQL:

```
SELECT idCorriere
FROM ORDINE
GROUP BY idCorriere
HAVING COUNT(DISTINCT idCliente) = (SELECT COUNT(*)
                                         FROM CLIENTE)
```

- b) Si scriva la query SQL che seleziona, per ogni anno, il corriere che ha consegnato il maggior numero di ordini.
- c) Si scriva la query SQL che seleziona il nome e il cognome degli impiegati che non hanno ricevuto più di cinque ordini da uno stesso cliente.

□ Esercizio 3.26

Dati i seguenti schemi relazionali:

```
CLIENTE (idCliente, cognome, nome, indirizzo)
PRODOTTO (idProdotto, nome, idCategoria:CATEGORIA, fornitore,
           prezzo)
ORDINE (idOrdine, idCliente:CLIENTE, data)
DETALGIO_ORDINE (idOrdine:ORDINE, idProdotto:PRODOTTO,
                  quantità)
CATEGORIA (idCategoria, nome)
```

- a) Si scriva la query SQL che seleziona il cognome e il nome dei clienti che non hanno effettuato ordini né nel 1997, né nel 1998.
- b) Si scriva la query SQL che seleziona, per ogni prodotto, la data dell'ultimo ordine in cui è comparso.
- c) Si scriva la query SQL che seleziona i prodotti che hanno un prezzo superiore alla media di quelli della stessa categoria.

Esercizio 3.27

Dati i seguenti schemi relazionali:

```

CLIENTE (idCliente, cognome, nome, indirizzo)
PRODOTTO (idProdotto, nome, idCategoria:CATEGORIA, fornitore)
ORDINE (idOrdine, idCliente:CLIENTE, data)
DETTAGLIO_ORDINE (idOrdine:ORDINE, idProdotto:PRODOTTO,
                  quantità, prezzo)
CATEGORIA (idCategoria, nome)

```

- a) Selezionare per ogni cliente l'ordine di importo massimo.
- b) Si scriva la query SQL che seleziona l'anno in cui è stato ricevuto il maggior numero di ordini.
- c) Si scriva la query SQL che seleziona, per ogni anno, il numero di prodotti che sono stati richiesti in più di 10 ordini.
- d) Si scriva la query SQL che seleziona, per ogni anno, il numero di clienti che non hanno effettuato ordini.

Esercizio 3.28

Dati i seguenti schemi relazionali:

```

PRODOTTO (idProdotto, nomeProdotto)
ORDINE (idOrdine, idCliente:CLIENTE, dataOrdine,
         idImpiegato:IMPIEGATO, idCorriere:CORRIERE)
DETTAGLIO_ORDINE (idOrdine:ORDINE, idProdotto, quantità, prezzo)
CORRIERE (idCorriere, nome, indirizzo)
CLIENTE (idCliente, cognome, nome)
IMPIEGATO (idImpiegato, cognome, nome, indirizzo)

```

- a) Si scrivano l'espressione di algebra relazionale e la query SQL che selezionano gli impiegati che non hanno ricevuto ordini.
- b) Si scriva la query SQL che seleziona, per ogni corriere, la data e l'importo dell'ultimo ordine consegnato.
- c) Si scriva la query SQL che seleziona, per ogni mese dell'anno 1998, il numero di ordini di importo superiore a 10000 €, ordinando il risultato per mese.

Esercizio 3.29

Dati i seguenti schemi relazionali:

```

PRODOTTO (idProdotto, nomeProdotto)
CLIENTE (idCliente, nomeCliente, nazione)
ORDINE (idOrdine, idCliente:CLIENTE, dataOrdine)
DETTAGLIO_ORDINE (idOrdine:ORDINE, idProdotto:PRODOTTO,
                  quantità, prezzo)

```

- a) Si scriva la query SQL che seleziona i prodotti che non sono stati ordinati da clienti italiani dopo il 10/01/2004.
- b) Si scriva la query SQL che seleziona, per ogni anno, il numero di prodotti che sono stati richiesti in meno di 1000 ordini di importo superiore a 8000 €.
- c) Si scriva la query SQL che visualizza il codice e il nome dei clienti che nel 1998 hanno richiesto almeno un prodotto ordinato dal cliente 'Bon app' nello stesso anno.

□ Esercizio 3.30

Dati i seguenti schemi relazionali:

```
PRODOTTO(idProdotto, nomeProdotto)
CLIENTE(idCliente, nomeCliente, nazione)
COPPIERE(idCorriere, nomeCorriere)
ORDINE (idOrdine, idCliente:CLIENTE, dataOrdine,
        idCorriere:COPPIERE)
DETALGO_ORDINE (idOrdine:ORDINE, idProdotto:PRODOTTO,
                 quantità, prezzo)
```

- a) Si scriva la query SQL che seleziona i corrieri che hanno consegnato ordini a tutti i clienti italiani.
- b) Si scriva la query SQL che seleziona, per ogni corriere, il numero di ordini consegnati in ogni anno.
- c) Si scriva la query SQL che seleziona il codice degli ordini relativi al prodotto più venduto.

□ Esercizio 3.31

Dati i seguenti schemi relazionali:

```
TITOLO (nome, settore)
CHIUSURA (nome:TITOLO, data, valore, diffPerc)
TRANSAZIONE (nome:TITOLO, data, numProg, valore)
```

- a) Scrivere l'espressione di algebra relazionale e la query SQL che selezionano le date in cui nessun titolo ha avuto chiusura con $diffPerc < 2\%$;
- b) Scrivere la query SQL che visualizza, per ciascun settore, i titoli che hanno chiuso con una $diffPerc < 2\%$ almeno 5 volte dall'inizio dell'anno 2004;
- c) Scrivere la query SQL che seleziona, per ciascun anno, il settore che ha avuto il volume di transazioni (somma del valore delle transazioni) minore.

□ Esercizio 3.32

Dati i seguenti schemi relazionali:

ARTISTA (idArtista, nomeArtista, località, regione)
CANZONE (idCanzone, nomeCanzone, durata, anno, vendite)
SCRIVE (idArtista:ARTISTA, idCanzone:CANZONE)

- Scrivere la query SQL che visualizza gli artisti che hanno scritto almeno una canzone che ha avuto un volume di vendite superiore a 200;
- Scrivere la query SQL che seleziona, per ogni artista, il numero di canzoni scritte per ciascun anno;
- Scrivere la query SQL che visualizza l'elenco delle canzoni scritte da un solo artista.

□ Esercizio 3.33

Dati i seguenti schemi relazionali:

CLIENTE (idCliente, nominativo, indirizzo, città, nazione)
OGGETTO (idOggetto, nome, datazione, provenienza, materiale)
ASTA (idAsta, idOggetto:OGGETTO, dataInizio, dataFine,
prezzoBase, rilancioMinimo)
RILANCIO (idCliente:CLIENTE, idAsta:ASTA, dataOra, importo)

- Scrivere l'espressione di algebra relazionale e la query SQL che visualizzano gli oggetti che non sono mai stati messi all'asta.
- Scrivere la query SQL che seleziona, per ogni cliente, il numero di rilanci effettuati per ogni asta del 2003 alla quale il cliente ha partecipato.
- Scrivere la query SQL che visualizza, per ogni oggetto, il rilancio di importo massimo e codice e nominativo del cliente che lo ha effettuato.

□ Esercizio 3.34

Dati i seguenti schemi relazionali:

CATEGORIA (idCategoria, descrizione)
CORRIDORE (idCorridore, cognome, nome, codiceTesserino,
gruppoCiclistico, idCategoria:CATEGORIA)
EDIZIONE (anno, dataSvolgimento, aperturaIscrizioni)
ISCRIZIONE(anno:EDIZIONE, idCorridore:CORRIDORE, tipoPercorso)

- Scrivere l'espressione di algebra relazionale e la query SQL che visualizzano l'elenco dei corridori che non hanno partecipato all'edizione del 1998.
- Scrivere la query SQL che visualizza, per ogni edizione, il numero di corridori partecipanti, raggruppati per tipoPercorso.
- Scrivere la query SQL che seleziona l'elenco dei corridori di categoria 'senior' che hanno partecipato ad almeno due edizioni della gara.

□ Esercizio 3.35

Dati i seguenti schemi relazionali:

```
SOCIO (tesserino, cognome, nome, indirizzo)
IMBARCAZIONE (matricola, nome, marca, modello, tipo)
POSSIENDE (tesserino:SOCIO, matricola:IMBARCAZIONE,
            percProprietà)
```

- a) Scrivere l'espressione di algebra relazionale e la query SQL che visualizzano l'elenco dei soci che non possiedono imbarcazioni di tipo 'Barca a motore'.
- b) Scrivere la query SQL che visualizza il numero di imbarcazioni possedute da ciascun socio per una percentuale superiore al 50%.
- c) Scrivere la query SQL che seleziona l'elenco dei soci che possiedono almeno due imbarcazioni di tipo 'Barca a motore'.
- d) Scrivere la query SQL che visualizza il numero di imbarcazioni di proprietà di un solo socio.
- e) Scrivere la query SQL che seleziona, per ciascuna imbarcazione di tipo 'Barca a motore', il numero di soci proprietari.

✓ Esercizio 3.1

La query seleziona i codici degli impiegati che partecipano a un numero di progetti pari al numero totale di progetti, ovvero che partecipano a tutti i progetti. Ciò corrisponde, in algebra relazionale, all'effettuazione di una divisione tra COLLABORA e PROGETTI:

$$\pi_{\text{codImp}, \text{codProg}}(\text{COLLABORA}) : \pi_{\text{codProg}}(\text{PROGETTI})$$

La proiezione di COLLABORA è necessaria per eliminare l'attributo mesiUomo; infatti, l'espressione

$$\pi_{\text{codImp}}(\text{COLLABORA} : \pi_{\text{codProg}}(\text{PROGETTI}))$$

restituisce solo gli impiegati che partecipano a tutti i progetti per un identico numero di mesi-uomo.

✓ Esercizio 3.2

Si riportano due formulazioni equivalenti:

```
SELECT COMPUTER.marca, COUNT(DISTINCT codSoftware)
FROM INSTALLAZIONI, FORNITURE, COMPUTER
WHERE FORNITURE.marca = COMPUTER.marca
AND COMPUTER.codComputer = INSTALLAZIONI.codComputer
AND dataInstallazione BETWEEN '1/1/1994' AND '31/12/1994'
AND nomeFornitore = 'Rossi'
GROUP BY COMPUTER.marca
```

```
SELECT marca, COUNT(DISTINCT codSoftware)
FROM INSTALLAZIONI, COMPUTER
WHERE COMPUTER.codComputer = INSTALLAZIONI.codComputer
AND dataInstallazione BETWEEN '1/1/1994' AND '31/12/1994'
AND marca IN(SELECT marca
              FROM FORNITURE
              WHERE nomeFornitore = 'Rossi')
GROUP BY marca
```

✓ Esercizio 3.3

```
SELECT nomeImp, qualifica
FROM IMPIEGATI
WHERE NOT EXISTS (SELECT *
                   FROM COLLABORA
                   WHERE COLLABORA.codImp = IMPIEGATI.codImp)
```

✓ Esercizio 3.4

Riportiamo due soluzioni equivalenti:

$$\begin{aligned} &\pi_{\text{città}}(\sigma_{\text{studente}='Rossi'}(\text{PROFESSORI} \bowtie \text{CORSI} \bowtie \text{STUDENTI})) \\ &\pi_{\text{città}}(\text{PROFESSORI} \bowtie \pi_{\text{prof}}(\text{CORSI} \bowtie \pi_{\text{corso}}(\sigma_{\text{studente}='Rossi'}(\text{STUDENTI})))) \end{aligned}$$

✓ Esercizio 3.5

a)

```
SELECT codP, MIN(prezzo)
FROM FARMACI, CONTIENE
WHERE FARMACI.codF = CONTIENE.codF
AND qtà >= 0.1
GROUP BY codP
```

b)

```
UPDATE FARMACI
SET prezzo = prezzo/100*105
WHERE codF IN (SELECT codF
                FROM CONTIENE
                WHERE codP = (SELECT codP
                               FROM PR_ATTIVI
                               WHERE descr = 'salicilato'))
```

c)

$$\begin{aligned} &\pi_{\text{descr}, \text{descr}}(\text{PR_ATTIVI} \bowtie_{\text{codP}=\text{codP1}} (\sigma_{\text{grado}>0.5}(\text{INCOMP})) \\ &\quad \bowtie_{\text{codP2}=\text{codP}\text{PR_ATTIVI}}) \end{aligned}$$

✓ Esercizio 3.6

```
SELECT attore, MAX(incasso)
FROM INCASSI, CAST
WHERE INCASSI.film = CAST.film
AND anno = 1995
AND attore IN (SELECT attore
```

```
FROM CAST
GROUP BY attore
HAVING COUNT(*) >= 2
GROUP BY attore
```

Si osservi che la query

```
SELECT attore, MAX(incasso)
FROM INCASSI, CAST
WHERE INCASSI.film = CAST.film
AND anno = 1995
GROUP BY attore
HAVING COUNT(*) >= 2
```

non risolve il problema in modo esatto, poiché seleziona gli attori che hanno partecipato al cast di almeno 2 film durante il solo 1995.

✓ Esercizio 3.7

Riportiamo due formulazioni equivalenti:

```
SELECT DISTINCT FORNITORI.sede, PROGETTI.sede
FROM FORNITORI, PROGETTI, FORPARPRO
WHERE FORNITORI.codFor = FORPARPRO.codFor
AND PROGETTI.codProg = FORPARPRO.codProg
```

```
SELECT DISTINCT FORNITORI.sede, PROGETTI.sede
FROM FORNITORI, PROGETTI
WHERE EXISTS(SELECT * FROM FORPARPRO
            WHERE FORPARPRO.codProg = PROGETTI.codProg
            AND FORNITORI.codFor = FORPARPRO.codFor)
```

✓ Esercizio 3.8

Riportiamo due formulazioni equivalenti:

```
SELECT nome
FROM SOFTWARE, INSTALLAZIONI, COMPUTER
WHERE SOFTWARE.codSoftware = INSTALLAZIONI.codSoftware
AND COMPUTER.codComputer = INSTALLAZIONI.codComputer
AND tipo = 'Word Processor'
AND dataInstall >= '1/1/1995'
AND marca = 'Apple'
GROUP BY INSTALLAZIONI.codSoftware, nome
HAVING COUNT(*) >= 5
```

```
SELECT nome
FROM SOFTWARE
WHERE tipo = 'Word Processor'
AND 5 <= (SELECT COUNT(*) FROM INSTALLAZIONI, COMPUTER
```

```

WHERE SOFTWARE.codSoftware = INSTALLAZIONI.codSoftware
AND COMPUTER.codComputer = INSTALLAZIONI.codComputer
AND dataInstall >= '1/1/1995'
AND marca = 'Apple')

```

✓ Esercizio 3.9

- a) Il join naturale tra R e S viene eseguito sull'attributo comune B. Per la proprietà distributiva della selezione rispetto al join si ha:

$$\sigma_{D>'d5'}(R \bowtie S) = R \bowtie (\sigma_{D>'d5'}(S))$$

Sia

$$T = \sigma_{D>'d5'}(S)$$

Si ha:

T	D	B	E
	d6	b4	e3
	d7	b2	e4
	d8	b4	e1
	d9	b5	e3

L'unico matching tra T e R sull'attributo B avviene per il valore b2; la relazione richiesta è quindi:

A	B	E
a3	b2	e4

- b) Valutiamo separatamente i due termini della divisione:

$$\begin{aligned}
T1 &= \pi_{B,E}(S) \\
T2 &= \pi_B(\sigma_{C<'c3'}(R))
\end{aligned}$$

T1	B	E	T2	B
	b1	e1		b1
	b1	e2		b2
	b2	e1		
	b2	e4		
	b3	e4		
	b4	e1		
	b4	e3		
	b5	e2		
	b5	e3		

Il risultato della divisione è una relazione, con un unico attributo E, contenente i valori di E che in T1 compaiono associati a tutti i valori di B in T2:

E
e1

✓ Esercizio 3.10

```
SELECT k1,k3,a1,a3,k2,a2,b2
FROM R1,R2,R3
WHERE R1.k1 = R3.k1
AND R1.a1 > R2.a2
AND R1.b1 > 100
AND R2.b2 < 10
```

✓ Esercizio 3.11

a)

```
SELECT nome,indirizzo
FROM CLIENTI
WHERE codCliente IN (SELECT codCliente
                      FROM PRESTITI
                      WHERE data < '7/6/1996'
                      GROUP BY codCliente
                      HAVING COUNT(*) >= 3)
```

b)

```
DELETE
FROM PRESTITI
WHERE data < '7/6/1996'
AND codCopia IN (SELECT codCopia
                  FROM VHS,TITOLI
                  WHERE VHS.codTitolo = TITOLI.codTitolo
                  AND genere = 'comico')
```

c)

```
SELECT regista,AVG(durata)
FROM TITOLI
WHERE regista IN (SELECT regista
                   FROM TITOLI
                   WHERE genere = 'avventuroso'
                   GROUP BY regista
                   HAVING COUNT(*) >= 3)
GROUP BY regista
```

✓ Esercizio 3.12

```
 $\pi_{\text{nome}, \text{data}}(\pi_{\text{nome}, \text{codCliente}}(\sigma_{\text{età} < 20}(\text{CLIENTI})) \bowtie \pi_{\text{codCliente}, \text{data}}(\sigma_{\text{data} > '1/1/95'}(\text{PRESTITI})))$ 
```

✓ Esercizio 3.13

- a) Si suppone che, per la gestione delle date, sia disponibile una funzione TODAY() che restituiscia la data odierna; si suppone inoltre che la differenza tra due valori di tipo data corrisponda al numero di giorni intercorrenti.

```
SELECT codCli, data, tipo, ammontare
FROM TRANSAZIONI, POSSIEDE
WHERE TRANSAZIONI.numConto = POSSIEDE.numConto
AND TODAY() - data <= 30
AND codCli IN (SELECT codcli
                FROM POSSIEDE, CONTI
                WHERE POSSIEDE.numConto = CONTI.numConto
                GROUP BY codcli
                HAVING SUM(saldo) < 0)
```

- b) L'effettuazione di un prelievo implica due operazioni sulla base di dati: l'inserimento della nuova transazione nella relazione TRANSAZIONI e l'aggiornamento del saldo del conto nella relazione CONTI:

```
INSERT INTO TRANSAZIONI (numConto, data, tipo, ammontare)
VALUES (8456, TODAY(), 'PRELIEVO', 1000000)
```

Si noti che è stato omesso il valore per la chiave noTrans. Si suppone infatti che i valori di chiave per la tabella TRANSAZIONI siano numeri progressivi generati automaticamente dal DBMS.

```
UPDATE CONTI
SET saldo = saldo - 1000000
WHERE numConto = 8456
```

✓ Esercizio 3.14

a)

```
SELECT nomeRicetta, SUM(prezzoUnitario * quantità) / porzioni
FROM INGR_RICETTA, INGREDIENTI, RICETTE
WHERE INGR_RICETTA.nomeIngrediente = INGREDIENTI.nomeIngrediente
AND INGR_RICETTA.nomeRicetta = RICETTE.nomeRicetta
GROUP BY nomeRicetta, porzioni
ORDER BY 2
```

Come argomento dell'operatore GROUP BY è stato aggiunto l'attributo porzioni poiché esso compare nella select-list di una query raggruppata. Equivalentemente, si può riformulare la query come segue:

```
SELECT nomeRicetta, SUM(prezzoUnitario*quantità/porzioni)
FROM INGR_RICETTA, INGREDIENTI, RICETTE
WHERE INGR_RICETTA.nomeIngrediente = INGREDIENTI.nomeIngrediente
AND INGR_RICETTA.nomeRicetta = RICETTE.nomeRicetta
GROUP BY nomeRicetta
ORDER BY 2
```

b)

```
SELECT DISTINCT descrizione
FROM INGR_RICETTA, RICETTE
WHERE INGR_RICETTA.nomeRicetta = RICETTE.nomeRicetta
AND nomeIngrediente IN (SELECT nomeIngrediente
                         FROM INGREDIENTI
                         WHERE calorie = (SELECT MAX(calorie)
                                           FROM INGREDIENTI))
```

(si utilizza l'operatore IN poiché si potrebbero avere più ingredienti con le stesse calorie).

✓ Esercizio 3.15

Il vincolo di integrità referenziale non è soddisfatto per le tuple di R2 in cui il valore di a non fa parte dell'insieme dei valori assunti da a stesso in R1. La query richiesta è di tipo correlato:

```
SELECT *
FROM R2
WHERE NOT EXISTS (SELECT *
                   FROM R1
                   WHERE R1.a = R2.a)
```

✓ Esercizio 3.16

a) La query interna restituisce i codici dei clienti che possiedono un solo conto corrente:

```
SELECT codCli, SUM(ammontare)
FROM POSSIEDE, TRANSAZIONI
WHERE POSSIEDE.numConto = TRANSAZIONI.numConto
AND TODAY() - data <=365
AND codCli IN (SELECT codCli
                FROM POSSIEDE
                GROUP BY codcli
                HAVING COUNT(*)=1)
GROUP BY codcli
```

b)

```
SELECT DISTINCT nomeCli
FROM CLIENTI, POSSIEDE, TRANSAZIONI
WHERE CLIENTI.codCli = POSSIEDE.codCli
AND POSSIEDE.numConto = TRANSAZIONI.numConto
AND ammontare > 100000000
```

Questa interrogazione è equivalente alla seguente espressione di algebra relazionale:

$$\pi_{\text{nomeCli}}(\text{CLIENTI} \bowtie \text{POSSIEDE} \bowtie \sigma_{\text{ammontare} > 100000000}(\text{TRANSAZIONI}))$$

✓ Esercizio 3.17

- a) Si riportano due formulazioni equivalenti della query che restituisce l'eventuale numero in conflitto con il numero da inserire:

```
SELECT NUMERI.*
FROM PREFISSI, NUMERI
WHERE PREFISSI.località = NUMERI.località
AND numero = '321473'
AND prefisso = (SELECT prefisso
                 FROM PREFISSI
                 WHERE località = 'Cesena')

SELECT *
FROM NUMERI
WHERE numero = '321473'
AND località IN (SELECT P1.località
                  FROM PREFISSI P1, PREFISSI P2
                  WHERE P1.prefisso = P2.prefisso
                  AND P2.località = 'Cesena')
```

b)

```
INSERT INTO NUMERI
VALUES ('Cesena', '321473', 'Rossi Paolo', 'Via Lunga, 3')
```

✓ Esercizio 3.18

- a) La query interna restituisce i codici degli aeroporti nei quali arriva almeno un volo dal Marconi e dai quali parte almeno un volo per il Kennedy:

```

SELECT nome
FROM AEROPORTO
WHERE codice IN (SELECT DISTINCT A.arrA
                  FROM VOLO A, VOLO B
                  WHERE A.partDa = 'JFK'
                  AND A.arrA = B.partDa
                  AND B.arrA = 'BLQ')

```

b)

```

SELECT B.* , A.*
FROM VOLO A, VOLO B
WHERE A.partDa = 'JFK'
AND A.arrA = B.partenzaDa
AND B.arrA = 'BLQ'
AND ((B.oraPart - A.oraArr)<3 AND (B.oraPart - A.oraArr)>0)
      OR (B.oraPart- A.oraArr)<21)

```

✓ Esercizio 3.19

Le relazioni normalizzate equivalenti a R sono le seguenti:

R1 (a, c)
R2 (a, b, d)

Le query per il trasferimento dei dati sono:

```

INSERT INTO R1
    SELECT DISTINCT a,c FROM R

```

```

INSERT INTO R2
    SELECT DISTINCT a,b,d FROM R

```

✓ Esercizio 3.20

- a) Si riportano due formulazioni equivalenti che restituiscono i nomi dei padri i cui figli (almeno uno) sono a loro volta padri:

```

SELECT nome
FROM PERSONA
WHERE id IN (SELECT idPadre
              FROM PADRE P1
              WHERE EXISTS (SELECT *
                            FROM PADRE P2
                            WHERE P1.idFiglio = P2.idPadre)
            )

```

```
SELECT nome
FROM PERSONA
WHERE id IN (SELECT P1.idPadre
              FROM PADRE P1, PADRE P2
              WHERE P1.idFiglio = P2.idPadre)
```

- b) Si noti che i "non nonni" sono esclusi in quanto l'inner join elimina le tuple che non soddisfano il predicato di join:

```
SELECT nome, COUNT(P2.idFiglio)
FROM PERSONA, PADRE P1, PADRE P2
WHERE P1.idFiglio = P2.idPadre
AND id = P1.idPadre
GROUP BY id, nome
```

✓ Esercizio 3.21

a)

```
SELECT idProdotto, P.nome
FROM PRODOTTO P, CATEGORIA C
WHERE P.idCategoria = C.idCategoria
AND C.nome = 'bevande'
AND idProdotto NOT IN (SELECT idProdotto
                        FROM ORDINE O, DETTAGLIO_ORDINE D
                        WHERE O.idOrdine = D.idOrdine
                        AND YEAR(data) = '2004')
```

b)

```
SELECT O.idOrdine, SUM(prezzo*quantità)
FROM ORDINE O, DETTAGLIO_ORDINE D, PRODOTTO P
WHERE O.idOrdine = D.idOrdine
AND D.idProdotto = P.idProdotto
AND O.idCliente = '1234'
AND data = (SELECT MAX(data)
            FROM ORDINE
            WHERE idCliente = '1234')
GROUP BY O.idOrdine
```

c)

```
SELECT C1.città, C1.idCliente, C1.nome, C2.idCliente, C2.nome
FROM CLIENTE C1, CLIENTE C2
WHERE C1.città = C2.città
AND C1.idCliente < C2.idCliente
```

d)

```
SELECT idProdotto, nome
FROM PRODOTTO P
WHERE NOT EXISTS (SELECT *
                  FROM CLIENTE C, ORDINE O, DETTAGLIO_ORDINE D
                  WHERE C.idCliente = O.idCliente
                  AND O.idOrdine = D.idOrdine
                  AND P.idProdotto = D.idProdotto
                  AND C.nazione IN ('Canada', 'Francia')
                 )
```

e)

```
SELECT C.idCliente, nome
FROM CLIENTE C, ORDINE O
WHERE C.idCliente = O.idCliente
GROUP BY C.idCliente, nome
HAVING COUNT(idOrdine) >= ALL (SELECT COUNT(idOrdine)
                                    FROM ORDINE
                                    GROUP BY idCliente)
```

f)

```
SELECT O.idOrdine, data
FROM ORDINE O, DETTAGLIO_ORDINE D, PRODOTTO P
WHERE O.idOrdine = D.idOrdine
AND P.idProdotto = D.idProdotto
GROUP BY O.idOrdine, data
HAVING COUNT(DISTINCT idCategoria) = 1
```

g)

```
SELECT C.nome, P.nome, SUM(quantità)
FROM PRODOTTO P, CATEGORIA C, DETTAGLIO_ORDINE D
WHERE P.idCategoria = C.idCategoria
AND P.idProdotto = D.idProdotto
GROUP BY C.idCategoria, C.nome, P.idProdotto, P.nome
HAVING SUM(quantità) >
ALL (SELECT SUM(quantità)
      FROM PRODOTTO P1, DETTAGLIO_ORDINE D1
      WHERE P1.idProdotto = D1.idProdotto
      AND P1.idCategoria = C.idCategoria
      GROUP BY D1.idProdotto)
```

h)

```
SELECT idProdotto, nome
FROM PRODOTTO
WHERE idProdotto IN (SELECT idProdotto
                      FROM DETTAGLIO_ORDINE
                      WHERE quantità > (SELECT AVG(quantità)
                                         FROM DETTAGLIO_ORDINE)
                      GROUP BY idProdotto
                      HAVING COUNT(idOrdine) >= 20)
```

✓ Esercizio 3.22

a)

```
SELECT I1.idImpiegato, I1.cognome, I1.nome
FROM IMPIEGATO I1, IMPIEGATO I2
WHERE I1.idImpiegato = I2.superiore
AND I1.posizione = 'Direttore commerciale'
AND I2.posizione = 'Rappresentante'
GROUP BY I1.idImpiegato, I1.cognome, I1.nome
HAVING COUNT(I2.idImpiegato) >= 3
```

b)

```
SELECT idImpiegato, cognome, nome
FROM IMPIEGATO I
WHERE posizione = 'Rappresentante'
AND NOT EXISTS (SELECT *
                  FROM CLIENTE C, ORDINE O
                  WHERE C.idCliente = O.idCliente
                  AND O.idImpiegato = I.idImpiegato
                  AND C.città <> I.città)
```

c)

```
SELECT YEAR(data), I.idImpiegato, cognome, nome
FROM IMPIEGATO I, ORDINE O
WHERE I.idImpiegato = O.idImpiegato
GROUP BY YEAR(data), I.idImpiegato, cognome, nome
HAVING COUNT(idOrdine) <= ALL (SELECT COUNT(O1.idOrdine)
                                    FROM ORDINE O1
                                    WHERE YEAR(O1.data) = YEAR(O.data)
                                    GROUP BY O1.idImpiegato)
```

d)

```
SELECT I.idImpiegato, cognome, nome, COUNT(DISTINCT idCliente)
FROM IMPIEGATO I, ORDINE O
WHERE I.idImpiegato = O.idImpiegato
GROUP BY I.idImpiegato, cognome, nome
```

e)

```
SELECT idImpiegato, cognome, nome
FROM IMPIEGATO
WHERE idImpiegato NOT IN (SELECT idImpiegato
                           FROM ORDINE
                           WHERE data > '20/05/2002')
```

f)

```
SELECT I1.idImpiegato, I2.idImpiegato, COUNT(idOrdine)
FROM IMPIEGATO I1, IMPIEGATO I2, ORDINE O
WHERE I1.posizione = 'Direttore'
AND I2.posizione = 'Rappresentante'
AND I2.superiore = I1.idImpiegato
AND O.idImpiegato = I2.idImpiegato
AND I2.idImpiegato in (SELECT idImpiegato
                        FROM ORDINE
                        GROUP BY idImpiegato
                        HAVING COUNT(idOrdine)>70)
GROUP BY I1.idimpiegato, I2.idimpiegato
ORDER BY 3 DESC
```

✓ Esercizio 3.23

a)

```
SELECT C.idCliente, cognome, nome, COUNT(idOrdine)
FROM CLIENTE C, ORDINE O
WHERE C.idCliente = O.idCliente
AND dataSpedizione IS NULL
GROUP BY C.idCliente, cognome, nome
```

b)

```
SELECT C.idCliente, cognome, nome
FROM CLIENTE C, ORDINE O
WHERE C.idCliente = O.idCliente
GROUP BY C.idCliente, cognome, nome
HAVING COUNT(DISTINCT idCorriere) = 1
```

c) Riportiamo due formulazioni alternative:

```
SELECT idCorriere, nomeCorriere
FROM CORRIERE C
WHERE NOT EXISTS
  (SELECT *
   FROM CLIENTE CL
   WHERE NOT EXISTS(SELECT *
                    FROM ORDINE O
                    WHERE YEAR(dataSpedizione) = '1998'
                      AND O.idCorriere = C.idCorriere
                      AND O.idCliente = CL.idCliente)
  )
SELECT C.idCorriere, nomeCorriere
FROM CORRIERE C, ORDINE O
WHERE C.idCorriere = O.idCorriere
AND YEAR(dataSpedizione) = '1998'
GROUP BY C.idCorriere, nomeCorriere
HAVING COUNT(DISTINCT idCliente) = (SELECT COUNT(*)
                                      FROM CLIENTE)
```

✓ Esercizio 3.24

- a) $\pi_{\text{nome}, \text{cognome}}(\sigma_{\text{stipendio} > 30000}(\text{IMPIEGATO}) \bowtie_{\text{superiore} = \text{idImpiegato}} \pi_{\text{superiore}}(\text{IMPIEGATO}))$
- b) Una possibile soluzione è data da:

$\pi_{\text{idCliente}, \text{nome}, \text{cognome}}(\sigma_{\text{idImpiegato} = 5}(\text{CLIENTE}) \bowtie_{\text{ORDINE}))$

L'espressione precedente può essere ottimizzata come segue:

$\pi_{\text{idCliente}, \text{nome}, \text{cognome}}(\text{CLIENTE} \bowtie \pi_{\text{idCliente}}(\sigma_{\text{idImpiegato} = 5}(\text{ORDINE})))$

c)

```
SELECT idCliente, YEAR(data), COUNT(idOrdine)
FROM ORDINE
GROUP BY idCliente, YEAR(data)
```

✓ Esercizio 3.25

- a) $\pi_{\text{idCorriere}, \text{idCliente}}(\text{ORDINE}) : \pi_{\text{idCliente}}(\text{CLIENTE})$

b)

```
SELECT YEAR(data), idCorriere
FROM ORDINE O
GROUP BY YEAR(data), idCorriere
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
                           FROM ORDINE O1
                           WHERE YEAR(O.data) = YEAR(O1.data)
                           GROUP BY O1.idCorriere)
```

c)

```
SELECT idImpiegato, cognome, nome
FROM IMPIEGATO
WHERE idImpiegato NOT IN (SELECT idImpiegato
                           FROM ORDINE
                           GROUP BY idImpiegato, idCliente
                           HAVING COUNT(idOrdine) > 5)
```

✓ Esercizio 3.26

a)

```
SELECT idCliente, cognome, nome
FROM CLIENTE
WHERE idCliente NOT IN (SELECT idCliente
                          FROM ORDINE
                          WHERE YEAR(data) = '1997'
                          OR YEAR(data) = '1998')
```

b)

```
SELECT P.idProdotto, nome, MAX(data)
FROM PRODOTTO P, ORDINE O, DETTAGLIO_ORDINE D
WHERE D.idOrdine = O.idOrdine
AND D.idProdotto = P.idProdotto
GROUP BY P.idProdotto, nome
```

c)

```
SELECT idProdotto, nome
FROM PRODOTTO P
WHERE prezzo > (SELECT AVG(prezzo)
                  FROM PRODOTTO P1
                  WHERE P.idCategoria = P1.idCategoria)
```

✓ Esercizio 3.27

a)

```
SELECT idCliente, SUM(prezzo*quantità)
FROM ORDINE O, DETTAGLIO_ORDINE D
WHERE O.idOrdine = D.idOrdine
GROUP BY O.idCliente, O.idOrdine
HAVING SUM(prezzo*quantità) >=
ALL (SELECT SUM(prezzo*quantità)
      FROM ORDINE O1, DETTAGLIO_ORDINE D1
      WHERE O1.idOrdine = D1.idOrdine
      AND O1.idCliente = O.idCliente
      GROUP BY O1.idOrdine)
```

b)

```
SELECT YEAR(data), COUNT(idOrdine)
FROM ORDINE
GROUP BY YEAR(data)
HAVING COUNT(idOrdine) >= ALL (SELECT COUNT(idOrdine)
                                    FROM ORDINE
                                    GROUP BY YEAR(data))
```

c)

```
SELECT YEAR(data), COUNT(P.idProdotto)
FROM PRODOTTO P, ORDINE O, DETTAGLIO_ORDINE D
WHERE P.idProdotto = D.idProdotto
AND D.idOrdine = O.idOrdine
AND P.idProdotto IN
  (SELECT P1.idProdotto
   FROM PRODOTTO P1, ORDINE O1, DETTAGLIO_ORDINE D1
   WHERE P1.idProdotto = D1.idProdotto
   AND D1.idOrdine = O1.idOrdine
   AND YEAR(O1.data) = YEAR(O.data)
   GROUP BY P1.idProdotto
   HAVING COUNT(O1.idOrdine) > 10)
GROUP BY YEAR(data)
```

d)

```
SELECT YEAR(data), (SELECT COUNT(*) FROM CLIENTE)
                     - COUNT(DISTINCT idCliente)
FROM ORDINE
GROUP BY YEAR(data)
```

✓ Esercizio 3.28

a) $\pi_{idImpiegato, cognome, nome}(\text{IMPIEGATO})$
 $\bowtie (\pi_{idImpiegato}(\text{IMPIEGATO}) - \pi_{idImpiegato}(\text{ORDINE}))$

```
SELECT idImpiegato, cognome, nome
FROM IMPIEGATO I
WHERE NOT EXISTS (SELECT *
                   FROM ORDINE O
                   WHERE O.idImpiegato = I.idImpiegato)
```

b)

```
SELECT C.idCorriere, nome, dataOrdine, SUM(prezzo*quantità)
FROM CORRIERE C, ORDINE O, DETTAGLIO_ORDINE D
WHERE C.idCorriere = O.idCorriere
AND O.idOrdine = D.idOrdine
AND dataOrdine = (SELECT MAX(dataOrdine)
                  FROM ORDINE O1
                  WHERE O1.idCorriere = O.idCorriere)
GROUP BY C.idCorriere, nome, dataOrdine
```

c)

```
SELECT MONTH(dataOrdine), COUNT(idOrdine)
FROM ORDINE
WHERE idOrdine IN (SELECT O.idOrdine
                     FROM ORDINE O, DETTAGLIO_ORDINE D
                     WHERE O.idOrdine = D.idOrdine
                     AND YEAR(dataOrdine) = '1998'
                     GROUP BY O.idOrdine
                     HAVING SUM(prezzo*quantità)>10000)
GROUP BY MONTH(dataOrdine)
ORDER BY 1
```

✓ Esercizio 3.29

a)

```
SELECT idProdotto, nomeProdotto
FROM PRODOTTO P
WHERE NOT EXISTS (SELECT *
                   FROM ORDINE O, DETTAGLIO_ORDINE D, CLIENTE C
                   WHERE O.idOrdine = D.idOrdine
                   AND C.idCliente = O.idCliente
                   AND P.idProdotto = D.idProdotto
                   AND dataOrdine > '10/01/2004'
                   AND C.nazione = 'Italia')
```

b)

```
SELECT YEAR(O.dataOrdine), COUNT(DISTINCT D.idProdotto)
FROM DETTAGLIO_ORDINE D, ORDINE O
WHERE O.idOrdine = D.idOrdine
AND D.idProdotto IN
    (SELECT D1.idProdotto
     FROM DETTAGLIO_ORDINE D1, ORDINE O1
     WHERE O1.idOrdine = D1.idOrdine
     AND O1.idOrdine IN
         (SELECT O2.idOrdine
          FROM ORDINE O2, DETTAGLIO_ORDINE D2
          WHERE O2.idOrdine = D2.idOrdine
          GROUP BY O2.idOrdine
          HAVING SUM(prezzo*quantità) >= 8000)
     AND YEAR(O1.dataOrdine) = YEAR(O.dataOrdine)
     GROUP BY D1.idProdotto
     HAVING COUNT(O1.idOrdine) < 1000)
GROUP BY YEAR(O.dataOrdine)
```

c)

```
SELECT DISTINCT C.idCliente, nomeCliente
FROM CLIENTE C, ORDINE O, DETTAGLIO_ORDINE D
WHERE C.idCliente = O.idCliente
AND D.idOrdine = O.idOrdine
AND YEAR(dataOrdine) = '1998'
AND EXISTS (SELECT *
    FROM ORDINE O1, CLIENTE C1, DETTAGLIO_ORDINE D1
    WHERE O1.idCliente = C1.idCliente
    AND D1.idOrdine = O1.idOrdine
    AND D1.idProdotto = D1.idProdotto
    AND C1.nomeCliente = 'Bon App'
    AND YEAR(O1.dataOrdine) = '1998')
```

✓ Esercizio 3.30

a)

```
SELECT idCorriere, nomeCorriere
FROM CORRIERE CO
WHERE NOT EXISTS
(SELECT *
    FROM CLIENTE C
    WHERE nazione = 'Italia'
    AND NOT EXISTS (SELECT *
        FROM ORDINE O
        WHERE O.idCorriere = CO.idCorriere
        AND O.idCliente = C.idCliente)
    )
```

b)

```
SELECT C.idCorriere, nomeCorriere, YEAR(dataOrdine),
COUNT(idOrdine)
FROM CORRIERE C, ORDINE O
WHERE C.idCorriere = O.idCorriere
GROUP BY C.idCorriere, nomeCorriere, YEAR(dataOrdine)
```

c)

```
SELECT DISTINCT O.idOrdine
FROM ORDINE O, DETTAGLIO_ORDINE D
WHERE O.idOrdine = D.idOrdine
AND idProdotto IN
(SELECT idProdotto
    FROM DETTAGLIO_ORDINE
    GROUP BY idProdotto
    HAVING SUM(quantità) >= ALL (SELECT SUM(quantità)
        FROM DETTAGLIO_ORDINE
        GROUP BY idProdotto)
    )
```

✓ Esercizio 3.31

a) $\pi_{\text{data}}(\text{CHIUSURA}) - \pi_{\text{data}}(\sigma_{\text{diffPerc} < 2}(\text{CHIUSURA}))$

```
SELECT DISTINCT data
FROM CHIUSURA
WHERE data NOT IN (SELECT data
                     FROM CHIUSURA
                     WHERE diffPerc < 2)
```

b)

```
SELECT settore, T.nome
FROM CHIUSURA C, TITOLO T
WHERE T.nome = C.nome
AND diffPerc < 2
AND data > '01/01/2004'
GROUP BY settore, T.nome
HAVING COUNT(data) >= 5
```

c)

```
SELECT YEAR(data), settore
FROM TRANSAZIONE T, TITOLO TI
WHERE T.nome = TI.nome
GROUP BY YEAR(data), settore
HAVING SUM(valore) <= ALL(SELECT SUM(valore)
                           FROM TRANSAZIONE T1, TITOLO TI1
                           WHERE T1.nome = TI1.nome
                           AND YEAR(T1.data) = YEAR(T.data)
                           GROUP BY settore)
```

✓ Esercizio 3.32

a)

```
SELECT DISTINCT A.idArtista, nomeArtista
FROM ARTISTA A, CANZONE C, SCRIVE S
WHERE A.idArtista = S.idArtista
AND S.idCanzone = C.idCanzone
AND vendite > 200
```

b)

```
SELECT A.idArtista, anno, COUNT(C.idCanzone) AS numeroCanzoni
FROM ARTISTA A, CANZONE C, SCRIVE S
WHERE A.idArtista = S.idArtista
AND S.idCanzone = C.idCanzone
GROUP BY A.idArtista, anno
```

c)

```
SELECT C.idCanzone, nomeCanzone
FROM CANZONE C, SCRIVE S
WHERE C.idCanzone = S.idCanzone
GROUP BY C.idCanzone, nomeCanzone
HAVING COUNT(idArtista) = 1
```

✓ Esercizio 3.33

a) OGGETTO▷◁($\pi_{idOggetto}(OGGETTO) - \pi_{idOggetto}(ASTA)$)

```
SELECT *
FROM OGGETTO
WHERE idOggetto NOT IN (SELECT idOggetto
                           FROM ASTA)
```

b)

```
SELECT idCliente, A.idAsta, COUNT(*)
FROM ASTA A, RILANCIO R
WHERE A.idAsta = R.idAsta
AND YEAR(dataInizio) = '2003'
GROUP BY idCliente, A.idAsta
```

c)

```
SELECT idOggetto, C.idCliente, nominativo, importo
FROM ASTA A, RILANCIO R, CLIENTE C
WHERE A.idAsta = R.idAsta
AND R.idCliente = C.idCliente
WHERE importo = (SELECT MAX(importo)
                  FROM RILANCIO R1, ASTA A1
                  WHERE R1.idAsta = A1.idAsta
                  AND A1.idOggetto = A.idOggetto)
```

✓ Esercizio 3.34

a) CORRIDORE▷◁($\pi_{idCorridore}(CORRIDORE) - \pi_{idCorridore}(\sigma_{anno='1998'}(ISCRIZIONE))$)

```
SELECT *
FROM CORRIDORE
WHERE idCorridore NOT IN (SELECT idCorridore
                           FROM ISCRIZIONE
                           WHERE anno = '1998')
```

b)

```
SELECT anno, tipoPercorso, COUNT(idCorridore)
FROM ISCRIZIONE
GROUP BY anno, tipoPercorso
```

c) Riportiamo due formulazioni equivalenti:

```
SELECT C.idCorridore, cognome, nome
FROM CORRIDORE C, CATEGORIA CA, ISCRIZIONE I
WHERE C.idCategoria = CA.idCategoria
AND C.idCorridore = I.idCorridore
AND descrizione = 'senior'
GROUP BY C.idCorridore, cognome, nome
HAVING COUNT(anno) >= 2
```

```
SELECT idCorridore, cognome, nome
FROM CORRIDORE C, CATEGORIA CA
WHERE C.idCategoria = CA.idCategoria
AND descrizione = 'senior'
AND idCorridore IN (SELECT idCorridore
                     FROM ISCRIZIONE
                     GROUP BY idCorridore
                     HAVING COUNT(anno) >= 2)
```

✓ Esercizio 3.35

a) SOCIO▷◁($\pi_{\text{tesserino}}(\text{SOCIO})$ -
 $\pi_{\text{tesserino}}(\sigma_{\text{tipo}='Barca a motore'}(\text{IMBARCAZIONE})▷◁\text{POSSIEDE}))$

```
SELECT *
FROM SOCIO S
WHERE NOT EXISTS (SELECT *
                   FROM IMBARCAZIONE I, POSSIEDE P
                   WHERE I.matricola = P.matricola
                   AND P.tesserino = S.tesserino
                   AND tipo = 'Barca a motore')
```

b)

```
SELECT S.tesserino, cognome, nome, COUNT(matricola)
FROM SOCIO S, POSSIEDE P
WHERE S.tesserino = P.tesserino
AND percProprietà > 50
GROUP BY S.tesserino, cognome, nome
```

c)

```
SELECT *
FROM SOCIO
WHERE tesserino IN (SELECT tesserino
                     FROM POSSIEDE P, IMBARCAZIONE I
                     WHERE P.matricola = I.matricola
                     AND tipo = 'Barca a motore'
                     GROUP BY tesserino
                     HAVING COUNT(matricola) >= 2)
```

d)

```
SELECT COUNT(matricola)
FROM IMBARCAZIONE
WHERE matricola IN (SELECT matricola
                     FROM POSSIEDE
                     GROUP BY matricola
                     HAVING COUNT(tesserino) = 1)
```

e)

```
SELECT I.matricola, nome, COUNT(tesserino)
FROM IMBARCAZIONE I, POSSIEDE P
WHERE I.matricola = P.matricola
AND tipo = 'Barca a motore'
GROUP BY I.matricola, nome
```

Parte Quarta

Stima dei Costi e Progettazione Fisica

Nota sul formalismo adottato

Per i parametri del modello dei costi si adotta la seguente notazione:

NP	numero di pagine della relazione
NT	numero di tuple della relazione
ET	numero di tuple residue
NK _{attr}	numero di valori distinti per l'attributo attr
NL _{attr}	numero di foglie per l'indice costruito sull'attributo attr
f(p)	fattore di selettività del predicato p
min(attr)	valore minimo per l'attributo attr
max(attr)	valore massimo per l'attributo attr
len(attr)	numero di byte occupati dall'attributo attr
D	dimensione di una pagina
u	fattore di riempimento delle pagine

Quando sarà necessario utilizzare contemporaneamente più relazioni, i relativi parametri saranno distinti tramite un pedice.

Si ricorda che, per un predicato che seleziona un singolo valore di un attributo, si ha:

$$f(\text{attr}=\text{val}) = 1/NK_{\text{attr}}$$

In assenza di informazioni specifiche, per prediciati di range assumeremo

$$f(\text{attr} > \text{val}) = f(\text{attr} < \text{val}) = f(\text{attr} \geq \text{val}) = f(\text{attr} \leq \text{val}) \approx 0.5$$

Elenchiamo brevemente le ipotesi di lavoro sottese, a meno di esplicite indicazioni contrarie, da tutti gli esercizi:

- gli attributi delle relazioni sono indipendenti tra loro;
- i valori degli attributi sono uniformemente distribuiti sulle tuple;
- gli indici sono di tipo B⁺-tree;
- l'accesso alle relazioni avviene o per scansione sequenziale o utilizzando un solo indice per volta;
- gli indici unclustered utilizzano TID-list ordinate;
- ove si reperiscono tuple con diversi valori di chiave tramite un indice unclustered, le liste di TID relative a tali valori non vengono globalmente ordinate.

Inoltre, per la maggior parte degli esercizi, nel calcolo dei costi di esecuzione verranno trascurati il numero di livelli degli indici e il costo di elaborazione delle tuple residue in memoria centrale.

□ Esercizio 4.1

È definita una vista in SQL mediante lo statement:

```
CREATE VIEW PERSONE_RICCHE
AS SELECT codFisc, cognome, nome, professione
        FROM PERSONE
       WHERE reddito > 100.000.000
```

essendo PERSONE la relazione:

```
PERSONE (codFisc, cognome, nome, reddito, indirizzo, dataNasc,
          professione)
```

Si valutino le possibili strategie per la risoluzione della seguente query in funzione dei parametri del modello di costi adottato:

```
SELECT codFisc
      FROM PERSONE_RICCHE
     WHERE professione IN ('DENTISTA', 'AVVOCATO')
```

nell'ipotesi che sulla relazione PERSONE siano costruiti un indice clustered sull'attributo reddito e un indice unclustered su professione.

□ Esercizio 4.2

È data la query:

```
SELECT *
      FROM COMPUTER
     WHERE marca = 'Apple'                                (p1)
       AND modello IN ('IIIfx', 'Quadra')                (p2)
       AND disco < 300                                     (p3)
```

sulla relazione:

```
COMPUTER (codComputer, marca, modello, disco)
```

su cui sono costruiti due indici: uno clustered su modello (con NL_{modello}=8 foglie) e uno unclustered su marca (con NL_{marca}=7 foglie).

Si determini il miglior piano d'accesso per la risoluzione della query, tenendo conto dei seguenti dati:

NP = 50, NT = 1000

$NK_{modello} = 30$, $NK_{marca} = 5$, $disco \in [100..500]$

Si stimi inoltre il numero di tuple residue della query.

□ Esercizio 4.3

È data la query:

```
SELECT C.modello, I.codSoftware  
FROM COMPUTER C, INSTALLAZIONI I  
WHERE C.marca IN ('Olivetti', 'IBM')  
AND C.codComputer = I.codComputer  
AND I.dataInstall > '1/1/1995'
```

(p1)
(p2)
(p3)

sulle relazioni:

```
COMPUTER (codComputer, marca, modello)  
INSTALLAZIONI (codComputer:COMPUTER, codSoftware, dataInstall)
```

su cui sono costruiti i seguenti indici:

su COMPUTER:

indice clustered su codComputer ($NL_{codComputer}=100$),
indice unclustered su marca ($NL_{marca}=70$);

su INSTALLAZIONI:

indice clustered su codSoftware ($NL_{codSoftware}=150$),
indice unclustered su codComputer ($NL_{codComputer}=200$).

Si determini il miglior piano d'accesso per la risoluzione della query (per l'esecuzione del join si ipotizzi l'utilizzo degli algoritmi nested-loop e sort-merge), tenendo conto dei seguenti dati:

$NPC = 150$, $NT_C = 10000$
 $NK_{marca} = 6$

$NP_I = 750$, $NT_I = 50000$
 $NK_{codSoftware} = 20$

□ Esercizio 4.4

È data la query:

```
UPDATE ANIMALI  
SET razione = 20  
WHERE genere = 'leone'  
AND razione < 20  
AND gabbia BETWEEN 1 AND 100
```

(p1)
(p2)
(p3)

sulla relazione:

```
ANIMALI (codEsemplare, genere, gabbia, età, razione)
```

su cui sono costruiti tre indici: uno clustered su genere (con $N_{L_{genere}}=30$ foglie), uno unclustered su razione (con $N_{L_{razione}}=30$ foglie) e uno unclustered su gabbia (con $N_{L_{gabbia}}=32$ foglie).

Si determini il miglior piano d'accesso per la risoluzione della query, tenendo conto dei seguenti dati:

$NP = 250, NT = 5000$
 $NK_{genere} = 50, NK_{gabbia} = 300, razione \in [0..50]$

□ Esercizio 4.5

È data la relazione:

R (a, b, c, d)

(tutti gli attributi hanno valori interi e ciascuno occupa 4 byte) con

$NT = 100000$
 $NK_a = 100000, NK_b = 5000, NK_c = 10, NK_d = 1000$
 $\min(a) = 1, \max(a) = 100000$
 $\min(b) = 1, \max(b) = 5000$
 $\min(c) = 1, \max(c) = 10$
 $\min(d) = 1, \max(d) = 1000$
 $\text{len}(TID) = 4 \text{ byte}, D=4 \text{ Kbyte}, u=0.69$

Si effettui il progetto fisico, limitandosi alla scelta degli indici senza considerare criteri di ordinamento, sulla base di un carico di lavoro così costituito:

- Query q1, frequenza 100:

```
SELECT *
FROM R
WHERE a BETWEEN 11 AND 30
AND c = 10          (p1)
AND d = 10          (p2)
```

- Query q2, frequenza 20:

```
UPDATE R
SET a = a + 10
WHERE b < 6
AND (c = 1 OR a <= 50000)      (p3)
AND (c = 1 OR a <= 50000)      (p4)
```

- Query q3, frequenza 300:

```
SELECT *
FROM R
WHERE d IN (10, 100, 500)        (p5)
```

Esiste un vincolo in base al quale le foglie degli indici creati non devono occupare più di 500 pagine.

□ Esercizio 4.6

Si calcoli il costo di esecuzione della query

```
SELECT nome  
FROM PERSONE  
WHERE età BETWEEN 30 AND 39
```

(p1)

nei seguenti casi:

1. nessun indice presente;
2. indice clustered su età;
3. indice unclustered con TID-list ordinate su età;
4. indice unclustered con TID-list disordinate su età;

assumendo

NP = 1000, NT = 50000

min(età) = 1, max(età) = 100 (età di tipo intero)

NL_{età} = 70

□ Esercizio 4.7

È data la query:

```
SELECT codLibro, titolo, genere, anno  
FROM BIBLIOTECA  
WHERE scaffale between 1 and 400  
AND anno < 1995  
ORDER BY genere
```

(p1)

(p2)

sulla relazione:

BIBLIOTECA (codLibro, titolo, genere, scaffale, anno)

su cui sono costruiti due indici, entrambi unclustered: uno su scaffale (con NL_{scaffale}=24 foglie) e uno su genere (con NL_{genere}=22 foglie).

Si determini il miglior piano d'accesso per la risoluzione della query, tenendo conto dei seguenti dati:

NP = 235, NT = 8000

NK_{genere} = 5, NK_{scaffale} = 500, anno ∈ [1970..1996]

len(codLibro+titolo+genere+anno) = 50 byte, D = 2 Kbyte

□ Esercizio 4.8

È data la query:

```
SELECT titolo, anno  
FROM QUADRI  
WHERE autore = 'Van Gogh'  
AND tecnica = 'olio su tela' (p1)  
(p2)
```

sulla relazione:

```
QUADRI (codQuadro, titolo, autore, tecnica, anno)
```

su cui sono costruiti due indici: uno clustered su tecnica (con $NL_{tecnica}=8$ foglie) e uno unclustered su autore (con $NL_{autore}=8$ foglie).

Utilizzando i dati seguenti si verifichi come, tenendo conto del costo di elaborazione delle tuple residue in memoria centrale, il piano di accesso migliore non coincida con il piano di accesso migliore ottenuto considerando i soli costi di I/O.

$NP = 75, NT = 5000$
 $NK_{tecnica} = 10, NK_{autore} = 100$
 $\alpha = 0.1$

□ Esercizio 4.9

È data la relazione:

```
CONFERENZE (codConf, titolo, anno, città, tema)
```

(conferenze con lo stesso titolo vengono ripetute in anni successivi) con

$NP = 100, NT = 2000$
 $NK_{codConf} = 2000, NK_{titolo} = 200, NK_{anno} = 10, NK_{città} = 500, NK_{tema} = 100$
 $NL_{codConf} = 20, NL_{titolo} = 10, NL_{anno} = 6, NL_{città} = 10, NL_{tema} = 8$

Se ne effettui il progetto fisico, limitandosi alla scelta degli indici secondari, sulla base di un carico di lavoro così costituito:

- Query q1, frequenza 50:

```
SELECT titolo  
FROM CONFERENZE  
WHERE anno = X  
AND tema = Y (p1)  
(p2)
```

- Query q2, frequenza 10:

```
SELECT anno, titolo  
FROM CONFERENZE
```

```
WHERE città = X  
AND tema = Y (p4)
```

(p3)

- Query q3, frequenza 100:

```
SELECT anno, città  
FROM CONFERENZE  
WHERE titolo in (X1,X2,X3)  
AND anno > Y (p6)  
AND tema = Z (p7)
```

(p5)

- Query q4, frequenza 30:

```
SELECT anno  
FROM CONFERENZE  
WHERE titolo = X  
AND città = Y
```

(p8)

(p9)

□ Esercizio 4.10

Sono date le relazioni

```
COMPUTER (codComputer, marca, modello)  
INSTALLAZIONI (codComputer:COMPUTER, codSoftware, dataInstall)
```

e la query

```
SELECT modello, codSoftware  
FROM COMPUTER, INSTALLAZIONI  
WHERE marca IN ('Olivetti', 'IBM')  
AND COMPUTER.codComputer = INSTALLAZIONI.codComputer (p1)  
AND dataInstall > '1/1/1995' (p2) (p3)
```

effettuata con frequenza w=50.

Supponendo di dover effettuare il progetto fisico di entrambe le relazioni, si mostri come la query di join assegnata può essere scomposta in due sottoquery su singola relazione; si calcoli poi la frequenza di ciascuna delle due sottoquery determinate. Per i calcoli si assuma:

NT_{COMPUTER} = 1000
NK_{marca} = 10

NT_{INSTALLAZIONI} = 5000
NK_{codSoftware} = 20

□ Esercizio 4.11

Sono dati lo schema

```
IMPIEGATI (codFiscale, nome, cognome, dataNascita, indirizzo,  
stipendio)
```

e la query

```

SELECT nome, cognome, stipendio
FROM IMPIEGATI
WHERE stipendio BETWEEN 1500000 AND 2000000
ORDER BY stipendio

```

(p1)

La relazione è ordinata sull'attributo codFiscale e comprende 3000 tuple memorizzate in 150 pagine. Sull'attributo stipendio è presente un indice di altezza 2 con 9 foglie. Lo stipendio minimo è 500.000 e quello massimo 5.000.000; tutti gli stipendi sono multipli di 100.000. La lunghezza totale dei campi nome, cognome, stipendio è di 50 byte, e le pagine di disco hanno una capacità di 2 Kbyte.

Determinare il miglior piano di accesso per l'esecuzione della query.

□ Esercizio 4.12

È data la relazione:

FILM (titolo, regista, genere, anno)

soggetta a un carico di lavoro costituito da 3 query q1, q2, q3 con frequenze rispettivamente 100, 50, 10. Effettuare il progetto fisico della relazione, limitatamente alla scelta degli indici senza considerare criteri di ordinamento, supponendo che le matrici dei costi siano così costituite:

$$\underline{C} = \begin{bmatrix} 200 & 150 & 100 & 200 \\ 10 & 200 & 200 & 200 \\ 150 & 250 & 250 & 250 \end{bmatrix}, \underline{C_M} = \begin{bmatrix} 200 \\ 200 \\ 250 \end{bmatrix}, \underline{U} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 \end{bmatrix}$$

(\underline{C} contiene i costi di accesso, \underline{U} i costi di modifica degli indici, $\underline{C_M}$ i costi della scansione sequenziale). Non esistono vincoli sull'occupazione di memoria da parte degli indici.

□ Esercizio 4.13

È data la query:

```

SELECT cod, nome
FROM GIOCATTOLO
WHERE tipo = 'Bambola'
AND numPezzi > 290

```

(p1)
(p2)

sulla relazione:

GIOCATTOLO (cod, nome, tipo, numPezzi)

su cui sono costruiti due indici: uno clustered su tipo (con $NL_{tipo} = 40$ e $h_{tipo} = 3$) e uno unclustered su numPezzi (con $NL_{numPezzi} = 30$ e $h_{numPezzi} = 3$).

Si calcolino i fattori di selettività dei due predicati e si determini il miglior piano d'accesso per la risoluzione della query tenendo conto dei seguenti dati:

NP = 260, NT = 5000

NK_{tipo} = 50, NK_{numPezzi} = 100, numPezzi ∈ [70..320]

□ Esercizio 4.14

È data la query (con parametro)

```
SELECT *
FROM TESI
WHERE argomento = X
```

(p1)

sulla relazione:

TESI (cod, titolo, autore, anno, argomento)

su cui è costruito un indice parziale unclustered su argomento relativo a 1/20 degli argomenti presenti.

Sapendo che in base al carico di lavoro stimato X appartiene all'insieme di valori indicizzati con probabilità dell'80%, si calcoli il costo atteso della query noti i seguenti dati:

NT=100000

NK_{argomento} = 2000 (si suppongano equamente distribuiti)

len(argomento) = 10 byte

len(cod+titolo+autore+anno+argomento) = 50 byte

len(TID) = 4 byte, D = 1 Kbyte, u = 0.69

□ Esercizio 4.15

È data la query:

```
SELECT codHotel, categoria, città
FROM HOTEL
```

WHERE città = 'Roma'

(p1)

AND numCamere > 20

(p2)

sulla relazione:

HOTEL (codHotel, categoria, città, numCamere)

su cui sono costruiti due indici, uno clustered su città (con NL_{città} = 36) e uno unclustered su numCamere (con NL_{numCamere} = 29).

Tenendo conto dei seguenti dati:

NP = 260, NT = 20000

$NK_{\text{città}} = 1000$, $NK_{\text{numCamere}} = 50$, $\text{numCamere} \in [10..100]$

si calcolino i fattori di selettività dei due predicati e si determini il miglior piano d'accesso per la risoluzione della query.

□ Esercizio 4.16

Si calcoli il costo di esecuzione della query:

```
SELECT codCliente, nome  
FROM CLIENTE  
WHERE nazionalità = 'italiana'  
AND età < 20
```

(p1)
(p2)

sulla relazione:

CLIENTE (codCliente, nome, nazionalità, età)

su cui sono costruiti due indici: uno clustered su nazionalità, con $NL_{\text{nazionalità}} = 30$, e uno unclustered su età, con $NL_{\text{età}} = 20$.

Si determini il miglior piano d'accesso per la risoluzione della query, tenendo conto dei seguenti dati:

$NP = 250$, $NT = 5000$
 $NK_{\text{nazionalità}} = 50$, $\text{età} \in [13..63]$

□ Esercizio 4.17

È data la query:

```
SELECT prodotto, AVG(quantità)  
FROM DETTAGLIO_ORDINE  
WHERE prezzoUnitario > 10  
AND idOrdine <= 1500  
GROUP BY prodotto
```

(p1)
(p2)

sulla relazione:

DETTAGLIO_ORDINE (idOrdine, prodotto, prezzoUnitario, quantità)

sulla quale sono costruiti due indici entrambi unclustered: uno su idOrdine (con $NL_{\text{idOrdine}} = 30$) e uno su prodotto (con $NL_{\text{prodotto}} = 145$).

Si determini il miglior piano d'accesso per la risoluzione della query tenendo conto dei seguenti dati:

$NP = 250$, $NT = 20000$
 $NK_{\text{idOrdine}} = 4800$, $NK_{\text{prodotto}} = 20$
 $\text{prezzoUnitario} \in [5..50]$, $\text{idOrdine} \in [1..4800]$, $\text{quantità} \in [1..100]$

$\text{len}(\text{prodotto}) = 20 \text{ byte}$, $\text{len}(\text{quantità}) = 4 \text{ byte}$
 $D = 4 \text{ Kbyte}$

Si ipotizzi di utilizzare l'algoritmo Sort-Merge a $Z=3$ vie per l'eventuale ordinamento del risultato.

□ Esercizio 4.18

È data la query:

```
SELECT corso, AVG(voto)
FROM ESAME
WHERE data >= '1/1/2000'                                (p1)
AND docente = 123                                       (p2)
GROUP BY corso
```

sulla relazione:

```
ESAME (matricola, corso, voto, data, docente)
```

sulla quale sono costruiti due indici entrambi unclustered: uno su matricola (con $\text{NL}_{\text{matricola}} = 170$) e uno su corso (con $\text{NL}_{\text{corso}} = 140$).

Si determini il miglior piano di accesso per la risoluzione della query tenendo conto dei seguenti dati:

$NP = 2200$, $NT = 25000$
 $NK_{\text{matricola}} = 4800$, $NK_{\text{corso}} = 20$, $NK_{\text{docente}} = 15$
 $\text{data} \in [1/1/1998, 1/1/2002]$, $\text{voto} \in [18..30] \cup \{33\}$
 $\text{len}(\text{corso}) = 50 \text{ byte}$, $\text{len}(\text{voto}) = 4 \text{ byte}$
 $D = 1 \text{ Kbyte}$

Si ipotizzi di utilizzare l'algoritmo Sort-Merge a $Z=3$ vie per l'eventuale ordinamento del risultato.

□ Esercizio 4.19

Si calcoli il costo di esecuzione della query:

```
SELECT corsoDiLaurea, AVG(media), MIN(media), MAX(media)
FROM STUDENTE
WHERE annoIscrizione BETWEEN 1981 AND 2000          (p1)
AND corsoDiLaurea <= 25                               (p2)
GROUP BY corsoDiLaurea
ORDER BY 2
```

sulla relazione:

```
STUDENTE (matricola, nome, cognome, annoIscrizione, media,  
corsoDiLaurea)
```

nell'ipotesi che:

NP = 200, NT = 10000

NK_{corsoDiLaurea} = 50

corsoDiLaurea ∈ [1..50], annoIscrizione ∈ [1951..2000]

len(corsoDiLaurea) = 4 byte, len(media) = 8 byte

D = 1 Kbyte

Per l'ordinamento si utilizzi l'algoritmo sort-merge a Z=3 vie.

□ Esercizio 4.20

È data la query

```
SELECT idOrdine, SUM(quantità)  
FROM DETTAGLIO_ORDINE  
WHERE idProdotto IN ('BEVG1', 'BEVG2', 'BEVG3') (p1)  
GROUP BY idOrdine
```

sulla relazione:

```
DETTAGLIO_ORDINE (idOrdine, idProdotto, quantità,  
prezzoUnitario)
```

su cui sono costruiti due indici, uno clustered su idOrdine e uno unclustered su idProdotto.

Si determini il miglior piano di accesso per la risoluzione della query tenendo conto dei seguenti dati, nell'ipotesi di utilizzare per l'eventuale ordinamento l'algoritmo di sort-merge a Z vie (Z=3):

NT = 100000

NK_{idOrdine} = 5000, NK_{idProdotto} = 1000

len(TID) = 4 byte, len(idOrdine) = 8 byte, len(idProdotto) = 8 byte

len(quantità) = 8 byte, len(prezzoUnitario) = 8 byte

D = 1 Kbyte, u = 0.69

Si stimi inoltre il numero di tuple residue della query.

□ Esercizio 4.21

È data la query

```
SELECT idFattura, SUM(quantità)
FROM DETTAGLIO_FATTURA
WHERE idProdotto BETWEEN 601 AND 610          (p1)
AND taglia = 'L'                                (p2)
GROUP BY idFattura
HAVING SUM(quantità) > 30
```

sulla relazione:

DETTAGLIO_FATTURA (*idFattura, numRiga, idProdotto, colore,*
taglia, quantità, prezzoUnitario)
su cui sono costruiti due indici, uno clustered su *idFattura* ($n_{idFattura} = 3$) e uno
unclustered su *idProdotto* ($n_{idProdotto} = 3$).

Si determini il miglior piano di accesso per la risoluzione della query tenendo conto
dei seguenti dati, nell'ipotesi di utilizzare per l'eventuale ordinamento l'algoritmo di
sort-merge a Z vie ($Z = 3$):

$$NT = 250000$$

$$NK_{idFattura} = 20000, NK_{idProdotto} = 1000, NK_{taglia} = 5, NK_{colore} = 15$$

$$\text{len}(idFattura) = 4 \text{ byte}, \text{len}(numRiga) = 4 \text{ byte}, \text{len}(idProdotto) = 4 \text{ byte}$$

$$\text{len}(colore) = 4 \text{ byte}, \text{len}(taglia) = 4 \text{ byte}, \text{len}(quantità) = 4 \text{ byte}$$

$$\text{len}(prezzoUnitario) = 8 \text{ byte}, \text{len}(TID) = 4 \text{ byte}$$

$$D = 1 \text{ Kbyte}, u = 0.69$$

✓ Esercizio 4.1

La query sulla vista PERSONE_RICCHE viene tradotta dall'ottimizzatore in una query equivalente sulla relazione PERSONE:

```
SELECT codFisc
FROM PERSONE
WHERE reddito > 100.000.000
AND professione IN ('DENTISTA', 'AVVOCATO')(p1)(p2)
```

Sono quindi possibili tre piani d'accesso:

1. *Scansione sequenziale.* Costo di accesso:
 $C_a = NP$
2. *Accesso tramite indice clustered su reddito.* Costo di accesso:
 $C_a = (h_{reddito} - 1) + [f(p1) \times NL_{reddito}] + [f(p1) \times NP]$
 (essendo $h_{reddito}$ il numero di livelli dell'indice su reddito)
3. *Accesso tramite indice unclustered su professione.* Costo di accesso:
 $C_a = (h_{professione} - 1) + 2 \times [f(professione=val) \times NL_{professione}] +$
 $+ 2 \times \Phi(f(professione=val)) \times NT, NP$

Si noti che, al punto 3, il termine $(h_{professione} - 1)$ esprime il caso migliore, quello in cui le foglie dell'indice relative ai due valori di *professione* sono raggiungibili nell'indice attraverso un unico percorso. Nel caso peggiore, il costo risulta invece essere $(h_{professione} - 1) + (h_{professione} - 2)$, poiché l'unico nodo dell'albero che non deve essere riletto è la radice.

✓ Esercizio 4.2

Calcoliamo i fattori di selettività dei tre predicati:

$$f(p1) = \frac{1}{NK_{marca}} = 0.20$$

$$f(p2) = \frac{2}{NK_{modello}} = \frac{1}{15}$$

$$f(p3) = \frac{val - min(disco)}{max(disco) - min(disco)} = 0.50$$

Il numero di tuple residue è quindi:

$$ET = f(p1) f(p2) f(p3) NT = 6.67$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.* Costo di accesso:
 $C_a = NP = 50$
2. *Accesso tramite indice clustered* su modello. Costo di accesso:
 $C_a = 2 \times [f(\text{modello=val}) \times NL_{\text{modello}}] + 2 \times [f(\text{modello=val}) \times NP] = 6$
3. *Accesso tramite indice unclustered* su marca. Costo di accesso:
 $C_a = [f(p1) \times NL_{\text{marca}}] + \Phi(f(p1) \times NT, NP) \approx 51$

Il piano preferibile è quindi quello che utilizza l'indice clustered.

☞ *Suggerimento:*

Si valuti il miglior piano di accesso nell'ipotesi che la relazione sia ordinata sull'attributo marca invece di modello.

✓ Esercizio 4.3

Calcoliamo i fattori di selettività dei predici locali:

$$f(p1) = \frac{2}{NK_{\text{marca}}} = \frac{1}{3}$$
$$f(p3) \approx 0.5$$

Si ha inoltre:

$$f(\text{marca=val}) = \frac{1}{NK_{\text{marca}}} = \frac{1}{6}$$
$$f(\text{codComputer=val}) = \frac{1}{10000}$$

(il calcolo di $f(\text{codComputer=val})$ vale con riferimento alla relazione COMPUTER poiché codComputer è chiave unica; per INSTALLAZIONI vale nell'ipotesi che per ogni computer esista almeno una installazione).

Il join con l'algoritmo nested-loop può essere eseguito usando come relazione esterna COMPUTER o INSTALLAZIONI:

1. COMPUTER → INSTALLAZIONI

Accesso a COMPUTER:

- a) *Scansione sequenziale.* Costo di accesso:
 $C_a(C) = NP_C = 150$
- b) *Accesso tramite indice unclustered* su marca. Costo di accesso:
 $C_a(C) = 2 \times [f(\text{marca=val}) \times NL_{\text{marca}}] + 2 \times \Phi(f(\text{marca=val}) \times NTC, NP_C) \approx 324$

- c) L'indice su codComputer non viene considerato: infatti la query non comprende predicati su codComputer (tranne il predicato di join), né è richiesto in uscita un ordinamento su codComputer. D'altronde, anche in presenza di un predicato ORDER BY codComputer, non converrebbe usare per l'accesso l'indice su codComputer: infatti, essendo la relazione ordinata proprio su codComputer, si pagherebbe un costo pari a:
 $C_a(C) = NL_{codComputer} + NPC$
sicuramente superiore a quello della scansione sequenziale.

Accesso a INSTALLAZIONI:

- d) *Scansione sequenziale.* Costo di accesso:
 $C_a(I) = NP_I = 750$
- e) *Accesso tramite indice unclustered* su codComputer. Costo di accesso:
 $C_a(I) = [f(codComputer=val) \times NL_{codComputer}] +$
 $+ \Phi(f(codComputer=val)) \times NT_I, NP_I \approx 6$

Utilizzando COMPUTER come relazione esterna, il piano preferibile risulta essere quello che utilizza la scansione sequenziale per l'accesso a COMPUTER e l'indice su codComputer per l'accesso a INSTALLAZIONI, con un costo totale pari a:

$$C_{tot} = C_a(C) + ET_C \times C_a(I) \approx 20148$$

essendo ET_C il numero di tuple residue della relazione COMPUTER:

$$ET_C = f(p1) \times NT_C \approx 3333$$

2. INSTALLAZIONI → COMPUTER

Accesso a INSTALLAZIONI:

- a) *Scansione sequenziale.* Costo di accesso:
 $C_a(I) = NP_I = 750$
- b) L'indice su codSoftware non viene considerato (vedi punto 1.c)
- Accesso a COMPUTER:
- c) *Scansione sequenziale.* Costo di accesso:
 $C_a(C) = NPC = 150$
- d) *Accesso tramite indice unclustered* su marca. Costo di accesso:
 $C_a(C) = 2 \times [f(marca=val) \times NL_{marca}] + 2 \times \Phi(f(marca=val)) \times NT_C, NP_C \approx 324$
- e) *Accesso tramite indice clustered* su codComputer. Costo di accesso:
 $C_a(C) = [f(codComputer=val) \times NL_{codComputer}] +$
 $+ [f(codComputer=val) \times NPC] = 2$

Utilizzando INSTALLAZIONI come relazione esterna, il piano preferibile risulta essere quello che utilizza la scansione sequenziale per l'accesso a INSTALLAZIONI e l'indice su codComputer per l'accesso a COMPUTER, con un costo totale pari a:

$$C_{tot} = C_a(I) + ET_I \times C_a(C) \approx 50750$$

essendo ET_I il numero di tuple residue della relazione INSTALLAZIONI:

$$ET_I = f(p3) \times NT_I = 25000$$

Il piano preferibile per l'algoritmo nested-loop è quindi quello che utilizza COMPUTER come relazione esterna (1.a - 1.e).

Se per effettuare il join viene impiegato l'algoritmo sort-merge, il costo è dato dalla somma di tre termini che esprimono, rispettivamente, i costi per ordinare la prima e la seconda relazione sull'attributo di join (ad esempio utilizzando l'algoritmo di sort-merge a Z=3 vie) e il costo per effettuare il merge delle relazioni così ordinate:

$$\begin{aligned}C_{\text{sort}}(C) &= 0 \\C_{\text{sort}}(I) &= 2 \times N_{P1} \times [\log_2(N_{P1})] \approx 9039 \\C_{\text{merge}} &= N_{PC} + N_{P1} = 900 \\C_{\text{tot}} &\approx 9939\end{aligned}$$

Appare evidente che, per l'esecuzione della query assegnata, l'algoritmo sort-merge risulta preferibile al nested-loop.

☞ **Suggerimento:**

In presenza di predicati locali, è possibile operare una variante dell'algoritmo sort-merge effettuando preventivamente la proiezione e la restrizione delle relazioni, memorizzando i risultati in relazioni temporanee che poi vengono ordinate sull'attributo di join. Si valuti il costo di esecuzione utilizzando questa variante.

Se entrambi gli attributi di join hanno un indice è possibile anche utilizzare l'algoritmo *simple tid*, che costruisce dapprima una lista delle coppie di TID delle tuple che partecipano al join e quindi accede alle relazioni. Se sono anche disponibili indici su attributi locali, si costruisce per ciascuna relazione una lista dei TID relativi alle tuple che li soddisfano; l'accesso alle relazioni viene effettuato per ciascuna coppia di TID di join, solo se entrambi i TID componenti sono presenti rispettivamente nelle liste di TID relative ai predicati locali. Nell'ipotesi che le liste siano in memoria centrale, si valuti il costo di esecuzione nel caso peggiore.

✓ **Esercizio 4.4**

Calcoliamo i fattori di selettività dei tre predicati:

$$\begin{aligned}f(p1) &= \frac{1}{N_{K_{\text{genera}}}} = 0.02 \\f(p2) &= \frac{\text{val} - \min(\text{razione})}{\max(\text{razione}) - \min(\text{razione})} = 0.4 \\f(p3) &= \frac{100}{N_{K_{\text{gabbia}}}} = \frac{1}{3}\end{aligned}$$

Il numero di pagine selezionate dal predicato sull'attributo clustered, p1, è

$$EP = f(p1) NP = 5$$

Il numero di tuple da modificare è

$$ET = f(p1) f(p2) f(p3) NT = 13.33$$

Il costo di uno statement di aggiornamento è la somma di tre contributi, che esprimono rispettivamente il costo di accesso ai dati, di modifica delle pagine dati e di modifica degli indici:

$$C_{tot} = C_a + C_w + C_m$$

Sono possibili tre piani d'accesso (l'attributo razione è oggetto della modifica, per cui si assume che l'indice su di esso non possa essere utilizzato):

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 250$$

Costo di modifica dati:

$$C_w = \Phi(ET, EP) = 4.72$$

Costo di modifica indice su razione:

$$C_m = C_{del} + C_{ins} = 2 \times ET + 2 = 28$$

Costo totale:

$$C_{tot} \approx 283$$

2. *Accesso tramite indice clustered su genere.*

Costo di accesso:

$$C_a = [f(p1) \times NL_{genere}] + [f(p1) \times NP] = 6$$

Costo di modifica dati:

$$C_w = 4.72$$

Costo di modifica indice su razione:

$$C_m = 28$$

Costo totale:

$$C_{tot} \approx 39$$

3. *Accesso tramite indice unclustered su gabbia.*

Costo di accesso:

$$C_a = [f(p3) \times NL_{gabbia}] + 100 \times \Phi(f(gabbia=val) \times NT, NP) = 1626.39$$

Costo di modifica dati:

$$C_w = 100 \times \Phi(ET/100, EP) = 14.30$$

Costo di modifica indice su razione:

$$C_m = 28$$

Costo totale:

$$C_{tot} \approx 1669$$

Il piano preferibile è quindi quello che utilizza l'indice clustered.

✓ Esercizio 4.5

Per risolvere l'esercizio occorre calcolare i costi di esecuzione di ciascuna query per ogni possibile piano d'accesso; tutti gli indici verranno considerati unclustered. Per la stima del numero di foglie di ciascun indice si usa la formula

$$NL = \left\lceil \frac{NK \times \text{len(chiave)} + NT \times \text{len(TID)}}{D \times u} \right\rceil$$

da cui:

$$NL_a = 284, NL_b = 149, NL_c = 142, NL_d = 143$$

Il numero di pagine in cui è contenuta la relazione è invece:

$$NP = \left\lceil \frac{\text{len(tupla)} \times NT}{D} \right\rceil = 391$$

I fattori di selettività sono calcolabili come segue:

$$\begin{aligned} f(p1) &= \frac{20}{NK_a} = \frac{1}{5000} \\ f(p2) &= \frac{1}{NK_c} = \frac{1}{10} \\ f(p3) &= \frac{5}{NK_b} = \frac{1}{1000} \\ f(p4) &= \frac{1}{NK_c} + \frac{50000}{NK_a} - \frac{1}{NK_c} \times \frac{50000}{NK_a} = \frac{11}{20} \\ f(p5) &= \frac{3}{NK_d} = \frac{3}{1000} \end{aligned}$$

1. Query q1:

a) Scansione sequenziale:

$$C_a = NP = 391$$

b) Accesso tramite indice su a:

$$C_a = [f(p1) \times NL_a] + 20 \times \Phi(f(a=val) \times NT, NP) \approx 21$$

c) Accesso tramite indice su c:

$$C_a = [f(p2) \times NL_c] + \Phi(f(p2) \times NT, NP) \approx 406$$

2. Query q2:

(EP = NP perché non si considera l'esistenza di un indice clustered;

$$ET = f(p3) \cdot f(p4) \cdot NT = 55$$

a) Scansione sequenziale:

$$C_a = NP = 391$$

$$C_w = \Phi(ET, EP) \approx 51$$

$$C_m = 2 \times ET + 2 \times EP = 220$$

b) Accesso tramite indice su b:

$$C_a = [f(p3) \times NL_b] + 5 \times \Phi(f(b=val) \times NT, NP) \approx 99$$

$$C_w = 5 \times \Phi(ET/5, EP) \approx 54$$

$$C_m = 2 \times ET + 2 \times EP = 220$$

L'indice su c non viene considerato poiché il predicato p4 contiene l'operatore OR.

3. Query q3:

a) Scansione sequenziale:

$$C_a = NP = 391$$

b) Accesso tramite indice su d:

$$C_a = 3 \times [f(d=val) \times NL_d] + 3 \times \Phi(f(d=val) \times NT, NP) \approx 268$$

Per determinare l'insieme ottimale di indici si può procedere come segue.

$$\underline{C} = \begin{bmatrix} 21 & 391 & 391 & 391 \\ 442 & 153 & 442 & 442 \\ 391 & 391 & 391 & 268 \end{bmatrix}, \underline{C}_M = \begin{bmatrix} 391 \\ 442 \\ 391 \end{bmatrix}, \underline{U} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 220 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \underline{W} = \begin{bmatrix} 100 \\ 20 \\ 300 \end{bmatrix},$$

$$\underline{M} = [284 \ 149 \ 142 \ 143]$$

Si denotano con I_1, I_2, I_3 e I_4 , rispettivamente, gli indici sugli attributi a, b, c e d. \underline{C}_M contiene i costi di riferimento, ovvero quelli relativi alla scansione sequenziale. \underline{C} è la matrice dei costi di accesso e modifica dei dati; l'elemento C_{ij} riporta il costo di esecuzione della query q_i in presenza del solo indice I_j (ovvero il minimo tra il costo che si ha utilizzando l'indice e il costo di riferimento). \underline{U} è la matrice dei costi di modifica degli indici; l'elemento U_{ij} riporta il costo di aggiornamento dell'indice I_j dovuto alla query q_i . \underline{W} è il vettore delle frequenze delle query; \underline{M} è un vettore il cui elemento j-esimo riporta l'occupazione di memoria dell'indice j.

I costi globali sono determinati dalla matrice

$$\underline{G} = \underline{C} + \underline{U} = \begin{bmatrix} 21 & 391 & 391 & 391 \\ 662 & 153 & 442 & 442 \\ 391 & 391 & 391 & 268 \end{bmatrix}$$

Poiché l'indice I_3 risulta essere dominato da I_2 e I_4 (la differenza nell'occupazione di memoria è trascurabile), I_3 viene eliminato dall'insieme degli indici candidati.

Occorre ora determinare l'insieme BSET di indici cui compete un costo globale minimo:

$$C_G(BSET) = \sum_{i=1}^3 w_i \times \left(\min\{C_{ij} \mid I_j \in BSET\} + \sum_{I_j \in BSET} U_{ij} \right)$$

$$\begin{aligned}C_G(\{I_1\}) &= 132640 \\C_G(\{I_2\}) &= 159460 \\C_G(\{I_4\}) &= 128340\end{aligned}$$

L'algoritmo utilizzato a tale scopo è un euristico di tipo ADD che considera progressivamente i diversi indici in ordine di costo globale crescente. Il vettore \underline{Y} memorizza quali indici, tra quelli in BSET, vengono utilizzati per le diverse query (il valore 0 indica l'utilizzo della scansione sequenziale). La variabile M(BSET) memorizza l'occupazione totale di memoria da parte degli indici in BSET.

Passo 1:

$$BSET = \{I_4\}$$

$$\underline{Y} = \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix}$$

$$C_G(BSET) = 128340$$

$$M(BSET) = 143$$

Passo 2:

$$BSET = \{I_4, I_1\}$$

$$\underline{Y} = \begin{bmatrix} 1 \\ 0 \\ 4 \end{bmatrix}$$

$$C_G(BSET) = 95740$$

$$M(BSET) = 427$$

Passo 3:

$$BSET = \{I_4, I_1, I_2\}$$

$$\underline{Y} = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}$$

$$C_G(BSET) = 89960$$

$$M(BSET) = 576$$

Considerando il vincolo sulla memoria totale occupabile dagli indici (500 Kbyte), l'insieme ottimale di indici per la relazione data risulta essere $BSET = \{I_1, I_4\}$. Verranno quindi previsti indici sugli attributi a e d .

Si tenga presente che, in realtà, l'indice sulla chiave primaria a è di fatto obbligatorio per il mantenimento dell'integrità referenziale della relazione.

✓ Esercizio 4.6

Il fattore di selettività del predicato p1 è

$$f(p1) = 0.1$$

1. In caso di assenza di indice, l'accesso avviene necessariamente per scansione sequenziale:
 $C_a = NP = 1000$
2. Se l'indice esiste ed è clustered, utilizzarlo costa
 $C_a = [f(p1) \times NL_{età}] + [f(p1) \times NP] = 107$
3. Se l'indice è unclustered con TID-list ordinate, utilizzarlo costa
 $C_a = [f(p1) \times NL_{età}] + 10 \times \Phi(f(età=val) \times NT, NP) \approx 3943$
4. Se l'indice è unclustered con TID-list disordinate, utilizzarlo costa
 $C_a = [f(p1) \times NL_{età}] + [f(p1) \times NT] = 5007$

✓ Esercizio 4.7

Calcoliamo i fattori di selettività dei due predicati:

$$f(p1) = \frac{400}{NK_{scaffale}} = \frac{4}{5}$$
$$f(p2) = \frac{25}{NK_{anno}} = \frac{25}{27}$$

Il numero di tuple residue è quindi:

$$ET = f(p1) f(p2) NT = 5925.93$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a Z=3 vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NPR = \left\lceil \frac{ET \times \text{len(select-list)}}{D} \right\rceil = 145$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.*
Costo di accesso:
 $C_a = NP = 235$
Costo di ordinamento delle tuple:
 $C_s = 2 \times NPR \times [\log_2(NPR)] = 1450$
Costo totale:
 $C_{tot} = 1685$
2. *Accesso tramite indice unclustered su scaffale.*
Costo di accesso:
 $C_a = [f(p1) \times NL_{scaffale}] + 400 \times \Phi(f(scaffale=val) \times NT, NP) = 6219.75$
Costo di ordinamento delle tuple:
 $C_s = 1450$

Costo totale:
 $C_{tot} \approx 7670$

3. *Accesso tramite indice unclustered su genere.*

Costo di accesso:

$$C_a = NL_{genere} + 5 \times \Phi(f(genere=val) \times NT, NP) = 1195.72$$

Costo di ordinamento delle tuple:

$$C_s = 0$$

Costo totale:
 $C_{tot} \approx 1196$

Il piano preferibile è quindi quello che utilizza l'indice unclustered su genere.

☞ Suggerimento:

Si valuti il miglior piano di accesso nell'ipotesi che la relazione sia ordinata sull'attributo scaffale.

✓ Esercizio 4.8

Calcoliamo i fattori di selettività dei prediciati:

$$f(p1) = \frac{1}{NK_{autore}} = 0.01$$

$$f(p2) = \frac{1}{NK_{tecnica}} = 0.1$$

Le tuple da elaborare in memoria centrale sono quelle recuperate accedendo al disco, a cui devono essere applicati i prediciati residui. Il numero di tuple da elaborare quando si accede alla relazione tramite l'indice su un attributo su cui esiste un predicato p è quindi calcolabile come:

$$ET_p = NT f(p)$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.* Costi di accesso e di elaborazione tuple residue:
 $C_a = NP = 75$
 $C_{el} = \alpha NT = 500$
(nel caso di scansione sequenziale vengono recuperate tutte le tuple)
2. *Accesso tramite indice clustered su tecnica.* Costi di accesso e di elaborazione tuple residue:
 $C_a = [f(p2) \times NL_{tecnica}] + [f(p2) \times NP] = 9$
 $C_{el} = \alpha ET_{p2} = 50$
3. *Accesso tramite indice unclustered su autore.* Costi di accesso e di elaborazione tuple residue:

$$C_a = [f(p1) \times NL_{autore}] + \Phi(f(p1) \times NT, NP) = 37.67$$

$$C_{el} = \alpha ET_{p1} = 5$$

Basando la valutazione solo sui costi di accesso a disco, il piano preferibile sembra essere quello che utilizza l'indice clustered. Tenendo invece conto anche del costo di elaborazione delle tuple residue, il piano di accesso migliore risulta essere quello che utilizza l'indice unclustered.

✓ Esercizio 4.9

I fattori di selettività dei predici sono:

$$f(p1) = \frac{1}{NK_{anno}} = \frac{1}{10}$$

$$f(p2) = f(p4) = f(p7) = \frac{1}{NK_{tema}} = \frac{1}{100}$$

$$f(p3) = f(p9) = \frac{1}{NK_{città}} = \frac{1}{500}$$

$$f(p5) = \frac{3}{NK_{titolo}} = \frac{3}{200}$$

$$f(p6) \approx 0.5$$

$$f(p8) = \frac{1}{NK_{titolo}} = \frac{1}{200}$$

1. Query q1:

- a) *Scansione sequenziale:*
 $C_a = NP = 100$

- b) *Accesso tramite indice su anno:*
 $C_a = [f(p1) \times NL_{anno}] + \Phi(f(p1) \times NT, NP) \approx 88$
- c) *Accesso tramite indice su tema:*
 $C_a = [f(p2) \times NL_{tema}] + \Phi(f(p2) \times NT, NP) \approx 19$

2. Query q2:

- a) *Scansione sequenziale:*
 $C_a = NP = 100$

- b) *Accesso tramite indice su città:*
 $C_a = [f(p3) \times NL_{città}] + \Phi(f(p3) \times NT, NP) \approx 5$
- c) *Accesso tramite indice su tema:*
 $C_a = [f(p4) \times NL_{tema}] + \Phi(f(p4) \times NT, NP) \approx 19$

3. Query q3:

- a) *Scansione sequenziale:*
 $C_a = NP = 100$

- b) *Accesso tramite indice su titolo:*
 $C_a = 3 \times [f(titolo=val) \times NL_{titolo}] + 3 \times \Phi(f(titolo=val) \times NT, NP) \approx 32$

- c) Accesso tramite indice su anno:
 $C_a = [f(p6) \times NL_{anno}] + f(p6) \times NK_{anno} \times \Phi(f(anno=val) \times NT, NP) \approx 436$
- d) Accesso tramite indice su tema:
 $C_a = [f(p7) \times NL_{tema}] + \Phi(f(p7) \times NT, NP) \approx 19$
4. Query q4:
- a) Scansione sequenziale:
 $C_a = NP = 100$
- b) Accesso tramite indice su titolo:
 $C_a = [f(p7) \times NL_{titolo}] + \Phi(f(p7) \times NT, NP) \approx 11$
- c) Accesso tramite indice su città:
 $C_a = [f(p8) \times NL_{città}] + \Phi(f(p8) \times NT, NP) \approx 5$

Determiniamo l'insieme ottimale di indici:

$$\underline{C} = \begin{bmatrix} 100 & 100 & 88 & 100 & 19 \\ 100 & 100 & 100 & 5 & 19 \\ 100 & 32 & 100 & 100 & 19 \\ 100 & 11 & 100 & 5 & 100 \end{bmatrix}, \underline{C}_M = \begin{bmatrix} 100 \\ 100 \\ 100 \\ 100 \end{bmatrix}, \underline{U} = [0], \underline{W} = \begin{bmatrix} 50 \\ 10 \\ 100 \\ 30 \end{bmatrix}$$

Si denotano con I_1, I_2, I_3, I_4 e I_5 rispettivamente, gli indici sugli attributi codConf, titolo, anno, città e tema. \underline{C} è la matrice dei costi di accesso ai dati; \underline{C}_M contiene i costi di riferimento, ovvero quelli relativi alla scansione sequenziale. La matrice dei costi di modifica degli indici, \underline{U} , è nulla poiché il carico di lavoro non comprende operazioni di modifica né di cancellazione. \underline{W} è il vettore delle frequenze delle query; non consideriamo l'occupazione di memoria degli indici poiché non è stato assegnato alcun vincolo in proposito.

Poiché l'indice I_1 risulta essere strettamente dominato dagli altri indici, viene eliminato dall'insieme degli indici candidati. Vediamo i costi globali dei rimanenti indici:

$$\begin{aligned} C_G(\{I_2\}) &= 9530 \\ C_G(\{I_3\}) &= 18400 \\ C_G(\{I_4\}) &= 15200 \\ C_G(\{I_5\}) &= 6040 \end{aligned}$$

L'algoritmo utilizzato è lo stesso dell'esercizio 4.5.

Passo 1:
 $BSET = \{I_5\}$

$$V = \begin{bmatrix} 5 \\ 5 \\ 5 \\ 0 \end{bmatrix}$$

$$C_G(BSET) = 6040$$

Passo 2:

$$BSET = \{I_5 I_2\}$$

$$V = \begin{bmatrix} 5 \\ 5 \\ 5 \\ 2 \end{bmatrix}$$

$$C_G(BSET) = 3370$$

Passo 3:

$$BSET = \{I_5 I_2 I_4\}$$

$$V = \begin{bmatrix} 5 \\ 4 \\ 5 \\ 4 \end{bmatrix}$$

$$C_G(BSET) = 3050$$

L'indice I_2 non viene più utilizzato, quindi viene eliminato da $BSET$.

Passo 4:

$$BSET = \{I_5 I_4 I_3\}$$

$$V = \begin{bmatrix} 5 \\ 4 \\ 5 \\ 4 \end{bmatrix}$$

$$C_G(BSET) = 3050$$

L'indice I_3 non viene utilizzato per nessuna query, quindi viene eliminato da $BSET$.

L'insieme ottimale di indici per la relazione data risulta essere $BSET = \{I_4 I_5\}$. Verranno quindi previsti indici sugli attributi `città` e `tema`.

✓ Esercizio 4.10

Assumendo di utilizzare per l'esecuzione l'algoritmo nested-loop con relazione esterna `COMPUTER` e relazione interna `INSTALLAZIONI`, la query di join può essere scomposta in due sottoquery. La prima, sulla relazione `COMPUTER`, ha frequenza $w_1 = w = 50$; restituisce, oltre all'attributo `modello`, l'attributo di join e contiene, tra i predicati della query originale, solo il predicato locale su `COMPUTER` p_1 .

q1:

```
SELECT modello, codComputer
FROM COMPUTER
WHERE marca IN ('Olivetti', 'IBM')
```

Questa sottoquery seleziona un numero di tuple pari a:

$$ET_1 = f(p1) \cdot NT_{COMPUTER} = \frac{2}{NK_{marca}} \cdot NT_{COMPUTER} = 200$$

La seconda sottoquery, q2, è sulla relazione INSTALLAZIONI e contiene il predicato locale su INSTALLAZIONI p3 accanto a un predicato parametrico che assumerà un valore definito, a ogni esecuzione di q2, in corrispondenza di ciascuna delle ET₁ tuple selezionate da q1:

q2:

```
SELECT codSoftware
FROM INSTALLAZIONI
WHERE dataInstall > '1/1/1995'
AND codComputer = <valore>
```

Questa sottoquery verrà effettuata per ogni tupla restituita da q1, quindi con frequenza w₂ = ET₁ · w = 10000.

✓ Esercizio 4.11

Calcoliamo i fattori di selettività del predicato su stipendio:

$$f(p1) = \frac{6}{NK_{stipendio}} = \frac{3}{23}$$

Il numero di tuple residue è quindi:

$$ET = f(p1) NT = 391.30$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a Z=3 vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NPR = \left\lceil \frac{ET \times \text{len(select-list)}}{D} \right\rceil = 10$$

Sono possibili due piani d'accesso:

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 150$$

Costo di ordinamento delle tuple:

$$C_s = 2 \times NPR \times [\log_2(NPR)] = 60$$

Costo totale:

$$C_{tot} = 210$$

2. Accesso tramite indice unclustered su stipendio.

Costo di accesso:

$$C_a = h-1 + [f(p1) \times NL_{stipendio}] + 6 \times \Phi(f(stipendio=val) \times NT, NP) = 320.19$$

Costo di ordinamento delle tuple:

$$C_s = 0$$

Costo totale:

$$C_{tot} \approx 320$$

Il piano preferibile è quindi quello che utilizza la scansione sequenziale.

✓ Esercizio 4.12

Determiniamo l'insieme ottimale di indici. Denotiamo con I_1 , I_2 , I_3 e I_4 , rispettivamente, gli indici sugli attributi titolo, regista, genere e anno; sia \underline{W} il vettore delle frequenze delle query:

$$\underline{W} = \begin{bmatrix} 100 \\ 50 \\ 10 \end{bmatrix}$$

I costi globali sono determinati dalla matrice

$$\underline{G} = \underline{C} + \underline{U} = \begin{bmatrix} 200 & 150 & 100 & 200 \\ 10 & 200 & 200 & 200 \\ 150 & 250 & 350 & 250 \end{bmatrix}$$

Gli indici I_2 e I_4 , dominati dagli altri indici, vengono eliminati dall'insieme degli indici candidati. Vediamo i costi globali dei rimanenti indici:

$$C_G(\{I_1\}) = 22000$$

$$C_G(\{I_3\}) = 23500$$

L'algoritmo utilizzato è lo stesso dell'esercizio 4.5.

Passo 1:

$$BSET = \{I_1\}$$

$$\underline{Y} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$C_G(BSET) = 22000$$

Passo 2:

$$BSET = \{I_1, I_3\}$$

$$\underline{Y} = \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix}$$

$$C_G(BSET) = 13000$$

L'insieme ottimale di indici per la relazione data risulta essere $BSET = \{I_1, I_3\}$. Verranno quindi previsti indici sugli attributi **titolo** e **genere**.

✓ Esercizio 4.13

I fattori di selettività sono calcolabili come segue:

$$f(p1) = \frac{1}{NK_{tipo}} = \frac{1}{50}$$

$$f(p2) = \frac{\max(\text{numPezzi}) - \text{val}}{\max(\text{numPezzi}) - \min(\text{numPezzi})} = \frac{3}{25}$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.* Costo di accesso: $C_a = NP = 260$
2. *Accesso tramite indice clustered su tipo.*
Costo di accesso: $C_a = (h_{\text{tipo}} - 1) + [f(p1) \times NL_{\text{tipo}}] + [f(p1) \times NP] = 9$
3. *Accesso tramite indice unclustered su numPezzi.* Costo di accesso:
 $C_a = (h_{\text{numPezzi}} - 1) + [f(p2) \times NL_{\text{numPezzi}}] + 12 \times \Phi(f(\text{numPezzi}) = \text{val}) \times NT, NP \approx 553$

Il piano preferibile è quindi quello che utilizza l'indice clustered su **tipo**.

✓ Esercizio 4.14

Calcoliamo il numero di pagine occupate dalla relazione:

$$NP = \left\lceil \frac{NT \times \text{len}(\text{cod} + \text{titolo} + \text{autore} + \text{anno} + \text{argomento})}{D \times u} \right\rceil = 7077$$

Il numero di chiavi di **argomento** indicizzate è pari a:

$$NK_{IX\text{argomento}} = \frac{1}{20} \times NK_{\text{argomento}} = 100$$

Il numero di foglie occupate dall'indice risulta dunque:

$$NL_{argomento} = \left\lceil \frac{NK_{IXargomento} \times \text{len(argomento)} + \frac{1}{20} \times NT \times \text{len(TID)}}{D \times u} \right\rceil = 30$$

La selettività del predicato, limitatamente all'insieme di valori di argomento indicizzati, è pari a:

$$f(p1) = \frac{1}{NK_{IXargomento}} = \frac{1}{100}$$

Essendo presente un indice parziale (costruito solo su una parte delle chiavi presenti nella relazione), sono possibili due piani di accesso:

1. *Scansione sequenziale.* Costo di accesso:

$$C_a = NP = 7077$$

2. *Accesso tramite indice unclustered su argomento nell'80% dei casi e tramite scansione sequenziale nel restante 20%.* Costo di accesso:

$$C_a = [f(p1) \times NL_{argomento}] + \Phi(f(\text{argomento}=val) \times NT, NP) \approx 51$$

In questo secondo caso il costo totale per l'esecuzione della query è dato dalla somma pesata (in base alla probabilità che il valore di argomento appartenga all'insieme di valori indicizzati) dei due costi di accesso (scansione sequenziale e accesso tramite indice):

$$C_{tot} = 0.8 \times 51 + 0.2 \times 7077 \approx 1456$$

Risulta ovviamente più conveniente sfruttare l'indice, che nella maggior parte dei casi (80%) permette di ridurre drasticamente il numero di pagine lette rispetto alla scansione sequenziale.

✓ Esercizio 4.15

I fattori di selettività dei prediciati risultano:

$$f(p1) = \frac{1}{NK_{città}} = \frac{1}{1000}$$

$$f(p2) = \frac{\max(\text{numCamere}) - \text{val}}{\max(\text{numCamere}) - \min(\text{numCamere})} = \frac{8}{9}$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.* Costo di accesso:

$$C_a = NP = 260$$

2. Accesso tramite indice clustered su città. Costo di accesso:
 $C_a = [f(p1) \times NL_{città}] + [f(p1) \times NP] = 2$
3. Accesso tramite indice unclustered su numCamere. Costo di accesso:
 $C_a = [f(p2) \times NL_{numCamere}] + 44.44 \times \Phi(f(numCamere=val) \times NT, NP) \approx 9107$

Il piano preferibile è quindi quello che utilizza l'indice clustered su città.

✓ Esercizio 4.16

Calcoliamo la selettività dei due predicati:

$$f(p1) = \frac{1}{NK_{nazionalità}} = \frac{1}{50}$$

$$f(p2) = \frac{\max(\text{età}) - val}{\max(\text{età}) - \min(\text{età})} = \frac{7}{50}$$

Sono possibili tre piani d'accesso:

1. Scansione sequenziale. Costo di accesso:
 $C_a = NP = 250$
2. Accesso tramite indice clustered su nazionalità. Costo di accesso:
 $C_a = [f(p1) \times NL_{nazionalità}] + [f(p1) \times NP] = 6$
3. Accesso tramite indice unclustered su età. Costo di accesso:
 $C_a = [f(p2) \times NL_{età}] + 7 \times \Phi(f(\text{età}=val) \times NT, NP) \approx 581$

Il piano preferibile è quindi quello che utilizza l'indice clustered su nazionalità.

✓ Esercizio 4.17

I fattori di selettività dei predici sono:

$$f(p1) = \frac{\max(\text{prezzoUnitario}) - val}{\max(\text{prezzoUnitario}) - \min(\text{prezzoUnitario})} = \frac{8}{9}$$

$$f(p2) = \frac{1500}{\max(idOrdine)} = \frac{5}{16}$$

Il numero di tuple residue prima del GROUP BY risulta:

$$ET = f(p1) \times f(p2) \times NT = 5555.55$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a Z=3 vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NPR = \left\lceil \frac{ET \times \text{len(select-list)}}{D} \right\rceil = 33$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 250$$

Costo di ordinamento delle tuple:

$$C_s = 2 \times DPR \times \lceil \log_2(DPR) \rceil = 264$$

Costo totale:

$$C_{tot} = 514$$

2. *Accesso tramite indice unclustered su idOrdine.*

Costo di accesso:

$$C_a = [f(p2) \times NL_{idOrdine}] + 1500 \times \Phi(f(idOrdine=val) \times NT, NP) \approx 6220$$

Costo di ordinamento delle tuple:

$$C_s = 264$$

Costo totale:

$$C_{tot} \approx 6484$$

3. *Accesso tramite indice unclustered su prodotto.*

Costo di accesso:

$$C_a = NL_{prodotto} + 20 \times \Phi(f(prodotto=val) \times NT, NP) \approx 5054$$

Costo di ordinamento delle tuple:

$$C_s = 0$$

Costo totale:

$$C_{tot} \approx 5054$$

Il piano preferibile è quindi quello che utilizza la scansione sequenziale.

✓ Esercizio 4.18

I fattori di selettività dei predicati sono calcolabili come segue:

$$f(p1) = \frac{1}{2}$$

$$f(p2) = \frac{1}{NK_{docente}} = \frac{1}{15}$$

Il numero di tuple attese prima del GROUP BY è quindi:

$$ET = f(p1) \times f(p2) \times NT = 833.33$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a Z=3 vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NPR = \left\lceil \frac{ET \times \text{len(select-list)}}{D} \right\rceil = 44$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 2200$$

Costo di ordinamento delle tuple:

$$C_s = 2 \times NPR \times \lceil \log_2(NPR) \rceil = 352$$

Costo totale:

$$C_{tot} = 2552$$

2. *Accesso tramite indice unclustered su matricola.*

Non è ragionevole utilizzare questo piano d'accesso in quanto non sono presenti predicati sull'attributo matricola; questo rende necessaria la lettura di tutte le foglie dell'indice, oltre a tutte le pagine della relazione, risultando in un costo d'accesso ai dati certamente superiore rispetto a quello ottenuto tramite scansione sequenziale.

3. *Accesso tramite indice unclustered su corso.*

Anche in questo caso non sono presenti predicati sull'attributo prodotto, ma potrebbe risultare vantaggioso utilizzare questo indice in quanto permette di risparmiare il costo di ordinamento delle tuple. Valutiamo dunque il costo di esecuzione della query.

Costo di accesso:

$$C_a = NL_{corso} + 20 \times \Phi(f(corso=val) \times NT, NP) \approx 19215$$

Costo di ordinamento delle tuple:

$$C_s = 0$$

Costo totale:

$$C_{tot} \approx 19215$$

Il piano preferibile è quindi quello che utilizza la scansione sequenziale.

✓ Esercizio 4.19

Calcoliamo il fattore di selettività dei due predicati:

$$f(p1) = \frac{2}{5}$$

$$f(p2) = \frac{25}{NK_{corsoDiLaurea}} = \frac{1}{2}$$

L'accesso può avvenire solo tramite scansione sequenziale, con costo $C_a = NP = 200$. Il GROUP BY avviene ordinando una tabella temporanea i cui record sono

lunghi $\text{len}(\text{corsoDiLaurea}) + \text{len}(\text{media}) = 12$ byte. Il numero di tuple attese per la tabella temporanea è $ET = f(p1) \times f(p2) \times NT = 2000$, e il numero di pagine

$$NP_R = \left\lceil \frac{ET \times 12}{D} \right\rceil = 24$$

Il costo di ordinamento per il raggruppamento delle tuple risulta:

$$C_s = 2 \times NP_R \times [\log_2(NP_R)] = 144$$

L'ORDER BY avviene ordinando una tabella temporanea i cui record sono lunghi $\text{len}(\text{corsoDiLaurea}) + \text{len}(\text{media}) \times 3 = 28$. Il numero di tuple attese per la tabella temporanea è $ET = NK_{\text{corsoDiLaurea}} \times f(p2) = 25$ e il numero di pagine

$$NP_R = \left\lceil \frac{ET \times 28}{D} \right\rceil = 1$$

Il costo di ordinamento risulta quindi essere trascurabile (si ordina in memoria).

Il costo totale è $C_{\text{tot}} = 344$.

✓ Esercizio 4.20

Calcoliamo innanzitutto il numero di pagine occupate dalla relazione:

$$NP = \left\lceil \frac{NT \times \text{len}(\text{idOrdine} + \text{idProdotto} + \text{quantità} + \text{prezzoUnitario})}{D \times u} \right\rceil = 4529$$

Il fattore di selettività del predicato è

$$f(p1) = \frac{3}{NK_{\text{idProdotto}}} = \frac{3}{1000}$$

Il numero di tuple residue prima del raggruppamento è quindi

$$ET = f(p1) \times NT = 300$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a $Z=3$ vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NP_R = \left\lceil \frac{ET \times \text{len}(\text{select-list})}{D} \right\rceil = 5$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 4529$$

Costo di ordinamento delle tuple:

$$C_s = 0 \text{ (la relazione è ordinata su idOrdine)}$$

Costo totale:

$$C_{tot} = 4529$$

2. *Accesso tramite indice clustered su idOrdine.*

Non è ragionevole utilizzare questo piano d'accesso. Il file dati, infatti, è già ordinato su idOrdine (attributo sul quale è richiesto il raggruppamento) e l'assenza di predici su tale attributo rende certamente più conveniente la scansione sequenziale.

3. *Accesso tramite indice unclustered su idProdotto.*

Calcoliamo il numero di foglie occupate dall'indice:

$$NL_{idProdotto} = \left\lceil \frac{NK_{idProdotto} \times \text{len}(idProdotto) + NT \times \text{len}(TID)}{D \times u} \right\rceil = 578$$

$$\begin{aligned} C_a &= 3 \times [f(idProdotto=val) \times NL_{idProdotto}] + \\ &+ 3 \times \Phi(f(idProdotto=val) \times NT, NP) \approx 300 \end{aligned}$$

Costo di ordinamento delle tuple:

$$C_s = 2 \times NPR \times [\log_2(NPR)] = 20$$

Costo totale:

$$C_{tot} \approx 320$$

Il piano preferibile è quindi quello che utilizza l'indice unclustered su idProdotto. Il numero massimo di tuple residue al termine della query è dato da:

$$ET_{finali} = \min(ET, NK_{idOrdine}) = 300$$

✓ **Esercizio 4.21**

Il numero di pagine occupate dalla relazione risulta:

$$NP = \left\lceil \frac{NT \times \text{len(tupla)}}{D \times u} \right\rceil = 11323$$

I fattori di selettività dei predici sono calcolabili come segue:

$$f(p1) = \frac{10}{NK_{idProdotto}} = \frac{1}{100}$$

$$f(p2) = \frac{1}{NK_{taglia}} = \frac{1}{5}$$

Il numero di tuple residue è quindi:

$$ET = f(p1) \times f(p2) \times NT = 500$$

Per quanto riguarda l'ordinamento del risultato, si suppone di utilizzare l'algoritmo di sort-merge a Z=3 vie. Il numero di pagine occupate dalla relazione da ordinare è:

$$NP_R = \left\lceil \frac{ET \times \text{len(select-list)}}{D} \right\rceil = 4$$

Sono possibili tre piani d'accesso:

1. *Scansione sequenziale.*

Costo di accesso:

$$C_a = NP = 11323$$

Costo di ordinamento delle tuple:

$$C_s = 0$$

Costo totale:

$$C_{tot} = 11323$$

2. *Accesso tramite indice clustered su idFattura.*

Non è ragionevole utilizzare questo piano d'accesso. Il file dati, infatti, è già ordinato su idFattura (attributo sul quale è richiesto il raggruppamento) e l'assenza di predicati su tale attributo rende certamente più conveniente la scansione sequenziale.

3. *Accesso tramite indice unclustered su idProdotto.*

Il numero di foglie occupate dall'indice risulta:

$$NL_{idProdotto} = \left\lceil \frac{NK_{idProdotto} \times \text{len}(idProdotto) + NT \times \text{len}(TID)}{D \times u} \right\rceil = 1421$$

Il costo d'accesso è dato da:

$$\begin{aligned} C_a &= (h_{idProdotto} - 1) + [f(p1) \times NL_{idProdotto}] + \\ &\quad + 10 \times \Phi(f(idProdotto=val) \times NT, NP) \approx 2490 \end{aligned}$$

Costo di ordinamento delle tuple:

$$C_s = 2 \times NP_R \times [\log_2(NP_R)] = 16$$

Costo totale:

$$C_{tot} \approx 2506$$

Il piano preferibile è allora quello che utilizza l'indice unclustered su idProdotto.

Parte Quinta

Dispositivi e Organizzazioni dei Dati

Nota sul formalismo adottato

Questo capitolo affronta alcuni esercizi relativi alla stima dei costi di accesso ai dispositivi di memoria secondaria, alle politiche di allocazione dei dati e alla gestione di organizzazioni di file. Data l'eterogeneità degli argomenti affrontati, ogni singolo esercizio illustra la notazione adottata.

□ Esercizio 5.1

È dato un disco con le seguenti caratteristiche:

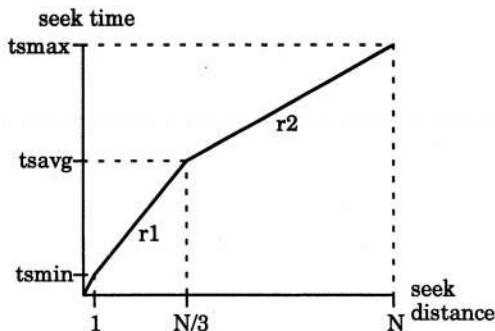
- bytes/sector = 512
- sectors/track = 88
- tracks/cylinder = 14
- rotation time = 11.1 msec
- average seek time t_s = 11.0 msec

Si stimi il tempo medio di accesso a un blocco di dimensione D = 4 KB.

□ Esercizio 5.2

È dato un disco con le seguenti caratteristiche:

- numero cilindri N = 1944
- block transfer time t_b = 1 msec
- rotational latency t_r = 5.55 msec
- min. seek time t_{smin} = 2.0 msec
- avg. seek time t_{savg} = 11.0 msec
- max seek time t_{smax} = 22.0 msec



Si assuma che il tempo di posizionamento (seek time) in funzione della distanza intertraccia (seek distance) d'abbia andamento lineare a tratti come in figura. Sia il braccio inizialmente posizionato sul cilindro 50 e sia la politica di gestione del disco di tipo SSTF (Shortest Seek Time First): è servita per prima la richiesta che

comporta il minimo spostamento del braccio). Supponendo che in coda di attesa vi siano quattro richieste di lettura di un blocco, rispettivamente nei cilindri 60, 200, 1000, 20, si calcoli il tempo totale di I/O speso per l'accesso ai quattro blocchi. Si consideri nulla la probabilità che un blocco richiesto sia già presente nella cache.

□ Esercizio 5.3

Un file ad accesso diretto, contenente $N = 2^{20}$ record memorizzati ordinatamente per valori di chiave primaria in $NP = 2^{15}$ blocchi, è allocato su disco nei cilindri compresi tra Cyl_{min} e Cyl_{max} . Si assuma il seguente modello per il calcolo del tempo $t_{I/O}$ di accesso a un blocco:

- seek time

$$t_s(d) = \begin{cases} 2 + 0.03 \times (d - 1), & \text{se } 0 < d \leq 201 \\ 7 + 0.005 \times (d - 1), & \text{se } d \geq 201 \end{cases}$$

dove d è la seek distance;

- block transfer time : $t_b = 0.95$ ms ;
- rotational latency : $t_r = 5.55$ ms.

Con riferimento ai due possibili casi di allocazione:

- a) $Cyl_{min} = 20$, $Cyl_{max} = 200$
- b) $Cyl_{min} = 40$, $Cyl_{max} = 700$

si calcoli il tempo di I/O $t_{avgI/O}$ speso in media per eseguire la ricerca dicotomica con successo di un record, nell'ipotesi di avere per ogni lettura di blocco una distanza di seek d pari a 1/3 dei cilindri utilizzati.

□ Esercizio 5.4

Per l'accesso a un insieme di $NP = 50$ blocchi di disco sia $HR=0.8$ l'Hit Ratio della cache di disco. Si assuma $t_{I/O} = 22$ msec il tempo medio di accesso a disco e $t_C = 0.01 \times t_{I/O}$ il tempo medio di accesso alla cache; si consideri inoltre nullo il tempo necessario a riconoscere se un blocco è già presente nella cache.

Calcolare l'effettivo tempo medio per l'accesso agli NP blocchi nei seguenti casi:

- a) per ogni blocco richiesto non già presente nella cache non si deve mai operare rimpiazzamento;
- b) per ogni blocco richiesto che non è già presente nella cache si deve operare un rimpiazzamento con riscrittura del blocco rimpiazzato.

□ Esercizio 5.5

Un disco A è caratterizzato da un datatransfer rate $DRA = 1.6 \text{ MB/sec}$, da un tempo di rotazione di 13.9 msec e da un tempo medio di posizionamento pari a 12.6 msec.

- Un disco B presenta una velocità di rotazione che è due volte quella del disco A. Quali sono gli effetti sul tempo medio di accesso a un blocco di 4 KB?
- Un disco C presenta una densità di area magnetica che è due volte quella del disco A. Quali sono gli effetti sul tempo medio di accesso a un blocco di 4 KB?

□ Esercizio 5.6

Sono dati due insiemi di interi A di cardinalità M , e B di cardinalità N , memorizzati rispettivamente in un array in memoria principale e in un file su disco; il file è ordinato in senso crescente, l'array è disordinato. Sia $N=8192$ il numero di record nel file, memorizzati in blocchi di 128 record ciascuno, e sia $M=512$ il numero degli elementi dell'array.

Si dispone di tre algoritmi Alg1, Alg2 e Alg3 per visualizzare gli interi facenti parte dell'insieme $A \cap B$, che operano rispettivamente secondo i criteri approssimativamente descritti:

- scandisce sequenzialmente il file e per ogni record ricerca se esiste già l'elemento nell'array;
- scandisce sequenzialmente l'array e per ogni elemento applica una ricerca dicotomica sul file.
- ordina dapprima l'array in memoria, quindi scandisce parallelamente l'array e il file.

Supponendo che la complessità in tempo sia determinata principalmente dall'I/O su disco, si determini per ognuno dei tre algoritmi il caso peggiore e si calcoli il numero di accessi a blocchi di disco.

□ Esercizio 5.7

Un file system è strutturato in modo che la richiesta di un blocco di un file possa essere soddisfatta indifferentemente su uno di due dischi mantenuti costantemente allineati.

In corrispondenza della richiesta di accesso a un blocco viene scelto il disco per il quale il tempo di posizionamento è più breve; cioè l'indirizzo della traccia richiesta viene comparato con l'indirizzo relativo all'ultima richiesta in coda al primo e al secondo disco, e viene scelto il disco per il quale risulta minimo lo spostamento del braccio.

Si assumano indipendenti le richieste di accesso e uniforme la distribuzione degli accessi sulle tracce; è inoltre nota la distribuzione cumulativa di probabilità $F(t_{\text{seek}})$ del tempo di seek per ognuna delle due code di servizio:

$$P(T_{\text{seek}} \leq 0 \text{ msec}) = 0.004$$

$$P(T_{\text{seek}} \leq 3 \text{ msec}) = 0.036$$

$$P(T_{\text{seek}} \leq 7 \text{ msec}) = 0.2$$

$$P(T_{\text{seek}} \leq 13 \text{ msec}) = 1$$

$$t_{\text{savg}} = 11.644 \text{ msec.}$$

Si determini il valor medio del tempo di seek con riferimento al complesso dei due dischi.

□ Esercizio 5.8

Nel corso dell'elaborazione di un'applicazione vengono inviate due richieste di accesso a file su due differenti dischi e l'esecuzione viene sospesa fino a completamento delle due operazioni di I/O che sono effettuate in parallelo.

Assumendo che il tempo di risposta di I/O su un singolo disco segua una distribuzione esponenziale con valor medio pari a r msec, si determini il tempo medio di risposta per il soddisfacimento di entrambe le richieste.

□ Esercizio 5.9

È dato un file memorizzato su disco in NP blocchi; ogni blocco è accessibile in modo diretto specificandone l'indirizzo logico ind (da 1 a NP).

Si prenda in esame il seguente frammento di codice:

```
if odd(NP)
then npassi:=NP div 2
else npassi:=NP div 2 - 1;
for p:=1 to npassi do
begin
  ind:=p+1;
  for j:=1 to npassi-p+1 do
  begin
    scambia_blocco(ind,ind+1);
    ind:=ind+2
  end
end;
```

dove $scambia_blocco(j,k)$ è una procedura che scambia il contenuto del blocco di indirizzo logico j con il contenuto del blocco di indirizzo logico k .

Si determini l'effetto prodotto dal codice sul file e si valuti, in funzione di NP , il numero di operazioni di lettura e scrittura su disco.

Si assuma per ipotesi di disporre di due buffer in memoria centrale, ciascuno atto a ospitare un singolo blocco, il primo I/O_BUFF da adibire alle operazioni di lettura e scrittura, il secondo AUX_BUFF da utilizzare come appoggio per lo scambio.

□ Esercizio 5.10

Un jukebox di dischi ottici ha $N_{drive} = 4$ drive, ciascuno con doppia testina, e $N_{disk} = 100$ dischi ottici doppia faccia. Esiste un solo braccio per la rimozione e montaggio dei dischi. Sono note inoltre le seguenti caratteristiche del dispositivo:

$t_{exc} = 6$ sec : tempo medio per sostituire su un drive un disco con un altro;
 $t_{I/O} = 0.5$ sec : tempo medio di accesso a un file che si trova su un disco montato su un drive.

Si calcoli il tempo t_{search} speso per la ricerca di un file nelle seguenti situazioni:

- 1) singolo utente e caso peggiore;
- 2) singolo utente e caso medio nell'ipotesi che ogni disco abbia la stessa probabilità di ospitare il file richiesto;
- 3) $U=6$ utenti e caso peggiore.

□ Esercizio 5.11

Si consideri un nastro magnetico con fattore di bloccaggio $BF=1$; sia $K=\{k_1, k_2, \dots, k_n\}$ l'insieme dei valori di chiave dei record da memorizzare e si supponga nota, per ogni valore di chiave k_i , la probabilità $p(k_i)$ di dover soddisfare una richiesta di reperimento del record associato.

Assumendo che $p(k_1) \geq p(k_2) \geq \dots \geq p(k_n)$ si determini l'allocazione dei record sul nastro che minimizza globalmente i costi di accesso (a ogni ricerca il nastro viene riavvolto). A questo scopo si assume $C_{I/O}(j) = j$ il costo di accesso al record memorizzato nel blocco j sul nastro.

Si calcoli poi, scelta l'allocazione ottima, il valor medio $\overline{C_{I/O}}$ del costo di accesso a un record, rispettivamente con riferimento alle seguenti distribuzioni di probabilità:

caso 1)

$$p(k_i) = \begin{cases} \frac{1}{2^i}, & \text{per } i = 1, \dots, n-1 \\ \frac{1}{2^{n-1}}, & \text{per } i = n \end{cases}$$

caso 2)

$$p(k_i) = \frac{i^{-1}}{\sum_{g=1}^n g^{-1}}, \text{ per } i=1, \dots, n$$

□ Esercizio 5.12

Si consideri un file ad accesso sequenziale; sia $K=\{k_1, k_2, \dots, k_n\}$ l'insieme dei valori di chiave dei record memorizzati nel file e si supponga n un multiplo di 5. Un blocco del file ospita per ipotesi un solo record.

L'80% delle richieste di accesso interessa il 20% dei record, e ciascuno di essi con probabilità

$$p(k_i) = \frac{0.8}{0.2 \times n}, \quad \text{per } i = 1, \dots, 0.2 \times n;$$

questi record sono allocati casualmente nei primi $0.2 \times n$ blocchi del file.

Il restante 20% delle richieste interessa l'80% dei record, e ciascuno di essi con probabilità

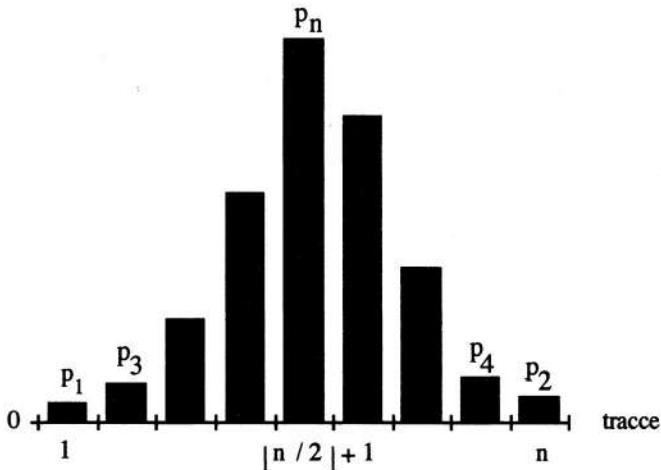
$$p(k_i) = \frac{0.2}{0.8 \times n}, \quad \text{per } i = 0.2 \times n + 1, \dots, n,$$

e questi record sono allocati casualmente nei blocchi da $0.2 \times n + 1$ a n .

Si assuma $C_{I/O}(j) = j$ il costo di accesso al record memorizzato nel blocco j del file. Si determini il valor medio $\overline{C_{I/O}}$ del costo di accesso a un record.

□ Esercizio 5.13

Si consideri il problema dell'allocazione ottima di n file su un disco che ha n tracce, nell'ipotesi che ogni file occupi un intera traccia. Sono note le probabilità di accesso ai file: $p_1 \leq p_2 \leq \dots \leq p_n$. Assumendo indipendenza fra richieste di accesso a file si argomenti a livello qualitativo che l'organizzazione "a canne d'organo" di figura rappresenta l'allocazione ottima (la figura si riferisce al caso di n dispari per motivi di chiarezza espositiva).



□ Esercizio 5.14

Calcolare l'occupazione di memoria di un secondary B+-tree con i seguenti dati:

- dimensione di un nodo: $D = 1KB$
- lunghezza puntatore: $\text{len}(p) = 4 \text{ byte}$
- lunghezza chiave: $\text{len}(k) = 10 \text{ byte}$
- numero record: $\text{NR} = 1000000$
- numero chiavi: $\text{NK} = 1000$
- utilizzazione: $u = \ln 2$

□ Esercizio 5.15

Calcolare l'occupazione di memoria delle foglie di un secondary B+-tree a **PID** (Page IDentifier) con i seguenti dati:

- dimensione di un nodo: $D = 1KB$
- lunghezza PID: $\text{len}(p) = 4 \text{ byte}$
- lunghezza chiave: $\text{len}(k) = 10 \text{ byte}$
- numero record: $\text{NR} = 1000000$
- numero chiavi: $\text{NK} = 1000$
- numero pagine dati: $\text{NP} = 50000$
- utilizzazione: $u = \ln 2$

sotto l'ipotesi che ogni pagina dati contenga $kdv = 5$ valori di chiave distinti.

□ Esercizio 5.16

È dato un file F con NK valori distinti di chiave primaria e un primary clustered index di tipo B⁺-tree con h livelli. Si esamini il seguente frammento di pseudocodice.

```
trovato:=false;
for i:=1 to M do
begin
    .....
    leggi(chiave);
    ricerca_su_indice(chiave,trovato,s);
    if not trovato
    then begin
        crea_record(puntr);
        { s puntatore a blocco indice per inserimento
          p puntatore a record dati }
        inserisci_in_indice(chiave,p,s)
    end;
    .....
end;
```

Ognuno degli M passi del ciclo comporta la lettura da terminale di un valore di chiave, la ricerca del valore nell'indice, e, se non è già presente, la scrittura sul file F di un nuovo record e l'inserimento nell'indice di un nuovo valore di chiave.

Si valuti la complessità nel caso peggiore, espressa in numero di chiamate alla procedura `inserisci_in_indice`; si indichi, per questo caso, quante operazioni di accesso a blocchi dati e a blocchi indice vengono effettuate nell'ipotesi che l'inserimento di un nuovo record nel file interessi un blocco dati già esistente, senza necessità di riordinamento, e che ogni inserimento di chiave nell'indice non provochi mai splitting.

□ Esercizio 5.17

È dato un B-tree di ordine g=9 con NK=999999 chiavi. Vengono effettuate M=10 operazioni di cancellazione di chiave e ogni cancellazione avviene in una foglia lasciando un numero di chiavi residuo maggiore o uguale a g.

Si stimi, nel caso peggiore, il costo complessivo delle M cancellazioni, in termini di numero di nodi letti e scritti, assumendo che la radice del B-tree resti disponibile in memoria principale dopo la prima cancellazione.

□ Esercizio 5.18

È dato un file con NR record memorizzati in NP pagine. Sul j-mo campo è costruito un secondary unclustered B⁺-tree a PID; siano rispettivamente NK_j il numero di valori distinti del campo j e DK_j la cardinalità del dominio del campo j.

Si calcoli la probabilità di aggiornare l'indice (ovvero aggiungere un nuovo PID) inserendo un nuovo record nel file, sapendo che il record viene inserito in una delle NP pagine già esistenti.

□ Esercizio 5.19

È dato un secondary unclustered index organizzato come un B⁺-tree che usa un posting file separato per la memorizzazione delle liste di TID:

dimensione di un nodo:	D = 1KB
lunghezza puntatore alla testa lista tid:	len(p) = 4 byte
lunghezza chiave:	len(k) = 20 byte
numero record file dati:	NR = 1000000
numero pagine file dati:	NP = 40000
numero valori di chiave distinti:	NK = 1000
utilizzazione:	u = ln 2
ordine:	g=21

Si ha una range query che porta a reperire EK=10 valori di chiave nel file dati; utilizzando per la ricerca l'indice suddetto si stimi il numero di accessi a pagine, assumendo pari a C_{post} il costo pagato per accedere a una lista di tid associata a un valore di chiave.

□ Esercizio 5.20

È dato un file con NR record memorizzati in NP pagine. Sul campo j, a valori ripetuti, è costruito un indice parziale unclustered strutturato come un B⁺-tree a TID di altezza h con NL foglie. Sia NK_j (multiplo di 3) il numero di valori distinti del campo j e sia {k₁, k₂, ..., k_{NK_j}} l'insieme di tali valori per i quali è nota inoltre la ripetitività:

$$p(k_i) = \begin{cases} \frac{0.8}{\frac{1}{3}NK_j}, & \text{per } i = 1, \dots, \frac{1}{3}NK_j \\ \frac{0.2}{\frac{2}{3}NK_j}, & \text{per } i = \frac{1}{3}NK_j, \dots, NK_j \end{cases}$$

L'indice è parziale nel senso che riporta nelle foglie solo i valori del campo j che cadono nel sottoinsieme {k_{NK_j/3+1}, ..., k_{NK_j}}.

Sia q la probabilità che una interrogazione con predicato di selezione (j = valore) interessi un valore del sottoinsieme {k₁, k₂, ..., k_{NK_j/3}} e (1-q) la probabilità che interessi un valore del sottoinsieme {k_{NK_j/3+1}, ..., k_{NK_j}}. Si calcoli il numero medio di

accessi a pagine (dati e indice) necessario per fornire risposta a una generica interrogazione del tipo suddetto.

□ Esercizio 5.21

È dato un file con NR record memorizzati in NP pagine. Sull'attributo A è disponibile un secondary unclustered index strutturato come un B⁺-tree di altezza h con NL foglie.

Il numero di valori distinti di A è pari a NK_A e per ogni possibile valore A_i, i=1,2, ..., NK_A, si conosce il numero n_i di record che presentano quel valore A_i nel file.

È dato inoltre un insieme Q={q₁,q₂,...,q_m} di interrogazioni con predicato di selezione (A = valore) che vengono risolte utilizzando l'indice suddetto come via di accesso.

Si stimi il valor medio del costo globale di accesso a pagine dati e indice, necessario per risolvere tutte le m interrogazioni, sapendo che la probabilità p_i che una generica interrogazione q_j interessi il valore A_i è pari a n_i/NK.

✓ Esercizio 5.1

Il tempo medio di accesso a un blocco può essere stimato come la somma del tempo medio di posizionamento, del tempo medio di latenza rotazionale e del tempo di traferimento del blocco:

$$t_{I/O} = t_s + t_r + t_b$$

Dai dati del problema si ricava:

$$t_s = 11.0 \text{ msec}, \quad t_r = \frac{1}{2} \times 11.1 \text{ msec} = 5.55 \text{ msec}, \quad t_b = \frac{D}{DR}$$

dove DR (Data Rate) è la velocità di trasferimento che può essere stimata, in assenza di altre informazioni, come:

$$DR = \frac{(\text{bytes/sector}) \times (\text{sectors/track})}{\text{rotation time}} = \frac{512 \times 88}{11.1} \approx 3.871 \text{ MB/sec}$$

Pertanto,

$$t_b = \frac{4 \text{ KB}}{3.871 \text{ MB/sec}} \approx 1 \text{ msec}, \quad t_{I/O} \approx 17.55 \text{ msec}$$

✓ Esercizio 5.2

Data la posizione iniziale del braccio, la politica SSTF soddisfa in sequenza le richieste nei cilindri 60, 20, 200, 1000 a cui corrispondono rispettivamente distanze di seek pari a 10, 40, 180, 800.

Per le prime tre richieste si fa riferimento alla retta r1 di equazione:

$$\frac{t_{\text{seek}} - t_{s\min}}{d - 1} = \frac{t_{\text{savg}} - t_{s\min}}{\frac{N}{3} - 1} \Rightarrow t_{\text{seek}} = \frac{9}{647} \times d + \frac{1285}{647}$$

mentre per l'ultima richiesta occorre fare riferimento alla retta r2 di equazione:

$$\frac{t_{\text{seek}} - t_{\text{savg}}}{d - \frac{N}{3}} = \frac{t_{\text{smax}} - t_{\text{savg}}}{N - \frac{N}{3}} \Rightarrow t_{\text{seek}} = \frac{11}{1296} \times d + \frac{7128}{1296}$$

Si ottiene pertanto :

$d = 10,$	$t_{\text{seek}}(1) = 2.125 \text{ msec}$
$d = 40,$	$t_{\text{seek}}(2) = 2.542 \text{ msec}$
$d = 180,$	$t_{\text{seek}}(3) = 4.490 \text{ msec}$
$d = 800,$	$t_{\text{seek}}(4) = 12.290 \text{ msec}$

Il tempo totale di I/O speso per l'accesso ai quattro blocchi richiesti vale:

$$(t_{\text{seek}}(1) + t_{\text{seek}}(2) + t_{\text{seek}}(3) + t_{\text{seek}}(4) + 4 \times t_b + 4 \times t_f) = 47.647 \text{ msec}$$

✓ Esercizio 5.3

Ricordando che la ricerca dicotomica con successo comporta in media $\lceil \log_2 NP \rceil$ accessi, si ha:

$$t_{\text{avgI/O}} = \lceil \log_2 NP \rceil \times (t_s(d) + t_b + t_f)$$

Nel caso (a) si ha

$$d = \frac{200 - 20 + 1}{3}$$

mentre nel caso (b)

$$d = \frac{700 - 40 + 1}{3}$$

Si ricava immediatamente:

caso a)	$t_{\text{avgI/O}} \approx 154.05 \text{ msec}$
caso b)	$t_{\text{avgI/O}} \approx 218.94 \text{ msec}$

✓ Esercizio 5.4

Si ricorda che l'Hit Ratio è definita come il rapporto fra il numero di riferimenti soddisfatti dalla cache e il numero totale di riferimenti.

caso a)

Il tempo effettivo medio di accesso a un blocco di disco è:

$$t_E = HR \times t_C + (1-HR) \times t_{I/O} = 4.576 \text{ msec}$$

Il tempo speso per accedere agli NP blocchi è:

$$NP \times t_E = 228.8 \text{ msec}$$

caso b)

$$t_E = HR \times t_C + (1-HR) \times 2 \times t_{I/O}$$

Il tempo speso per accedere agli NP blocchi è:

$$NP \times t_E = 448.8 \text{ msec}$$

✓ Esercizio 5.5

- a) Si ricorda che la velocità di trasferimento è inversamente proporzionale al tempo di rotazione secondo la formula:

$$DR = \frac{\text{(bytes/sector)} \times \text{(sectors/track)}}{\text{rotation time}}$$

Se B ruota due volte più velocemente di A si deduce immediatamente che

$$DR_B = 2 \times DR_A = 3.2 \text{ MB/sec}$$

dunque il tempo di trasferimento di un blocco di 4 KB è pari a

$$t_b(B) = \frac{1}{2} t_b(A) \approx 1.22 \text{ msec}$$

Il tempo di posizionamento non è influenzato dalla velocità di rotazione, mentre raddoppiando la velocità di rotazione si dimezza il tempo di latenza rotazionale. Pertanto si ha:

$$\begin{aligned} t_{I/O}(A) &\approx 12.6 + 6.95 + 2.44 = 21.99 \text{ msec} \\ t_{I/O}(B) &\approx 12.6 + 3.475 + 1.22 = 17.295 \text{ msec} \end{aligned}$$

- b) Se C presenta una densità magnetica superficiale doppia di quella di A, ciò significa che la densità lineare è $\sqrt{2}$ volte quella di A, dunque:

$$t_b(C) = \frac{t_b(A)}{\sqrt{2}} \approx 1.725$$

Il tempo di latenza rotazionale non varia e si può assumere che anche il tempo medio di posizionamento resti immutato, dunque:

$$\begin{aligned} t_{I/O}(A) &\approx 12.6 + 6.95 + 2.44 = 21.99 \text{ msec} \\ t_{I/O}(C) &\approx 12.6 + 6.95 + 1.725 = 21.27 \text{ msec} \end{aligned}$$

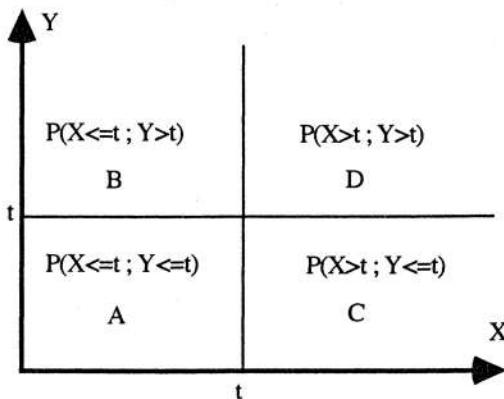
✓ Esercizio 5.6

Il file contiene $NP = 8192/128 = 64$ blocchi:

- Alg1) Nel caso peggiore accede a $NP = 64$ blocchi.
- Alg2) Per ogni elemento dell'array opera una ricerca dicotomica su file, pertanto, nel caso peggiore legge $M \times (\lfloor \log_2 NP \rfloor + 1) = 512 \times 7 = 3584$ blocchi.
- Alg3) Nel caso peggiore accede a $NP = 64$ blocchi.

✓ Esercizio 5.7

Si considerano due variabili aleatorie X e Y che rappresentano rispettivamente il tempo di seek per il primo e per il secondo disco, entrambe con la stessa distribuzione di probabilità e considerate per ipotesi indipendenti. La variabile aleatoria che interessa è $Z = \min(X, Y)$ e per questa si deriva la distribuzione cumulativa, applicando il ragionamento seguente: poiché X e Y assumono valori positivi, si osservi lo spazio degli eventi per la funzione di densità congiunta tracciato in figura.



L'evento $Z \leq t$ si verifica ogni volta che è vera l'espressione:

$$(X \leq t \text{ and } Y \leq t) \text{ or } (X \leq t \text{ and } Y > t) \text{ or } (X > t \text{ and } Y \leq t)$$

quindi si può scrivere che

$$P(Z \leq t) = P(A) + P(B) + P(C) = 1 - P(D)$$

Dall'ipotesi di indipendenza si ha:

$$\begin{aligned} P(D) &= P(X > t ; Y > t) = P(X > t) \times P(Y > t) = \\ &= (1 - P(X \leq t)) \times (1 - P(Y \leq t)) = (1 - F(t)) \times (1 - F(t)) = [1 - F(t)]^2 \end{aligned}$$

da cui

$$P(Z \leq t) = 1 - [1 - F(t)]^2$$

La distribuzione cumulativa relativa al complesso dei due dischi diventa dunque:

$$P(T_{\text{seek}} \leq 0 \text{ msec}) = 0.008$$

$$P(T_{\text{seek}} \leq 3 \text{ msec}) = 0.071$$

$$P(T_{\text{seek}} \leq 7 \text{ msec}) = 0.360$$

$$P(T_{\text{seek}} \leq 13 \text{ msec}) = 1$$

e il nuovo tempo medio di posizionamento è:

$$t_{\text{savg}} = (1 - 0.360) \times 13 + (0.360 - 0.071) \times 7 + (0.071 - 0.008) \times 3 = 10.532 \text{ msec}$$

✓ Esercizio 5.8

Si ricorda che la distribuzione esponenziale è descritta dalla funzione di densità $f(t) = \lambda e^{-\lambda t}$ con valor medio pari a $1/\lambda$. È immediato ricavare che la distribuzione cumulativa per un singolo disco è: $F(t) = 1 - e^{-t/r}$; poiché le due operazioni procedono in parallelo, il tempo per completarle entrambe è pari al massimo dei tempi richiesti singolarmente alle due unità disco.

Si considerino due variabili aleatorie X e Y indipendenti, caratterizzate dalla stessa distribuzione $F(t)$, e la variabile aleatoria $Z = \max(X, Y)$. Con riferimento alla figura dell'esercizio precedente si ha:

$$P(Z \leq t) = P(A) = P(X \leq t; Y \leq t)$$

e per l'ipotesi di indipendenza:

$$P(Z \leq t) = P(X \leq t) \times P(Y \leq t) = [F(t)]^2 = 1 - 2xe^{-t/r} + e^{-2xt/r}.$$

Il valor medio del tempo di risposta si calcola dunque come:

$$\int_0^\infty t \times d(P(Z \leq t)) = \int_0^\infty t \times \left(\frac{2}{r} \times e^{-t/r} - \frac{2}{r} \times e^{-2t/r} \right) dt = \frac{3}{2} r$$

✓ Esercizio 5.9

L'insieme dei blocchi può essere pensato come partizionato in due sottoinsiemi: il sottoinsieme dei blocchi di indirizzo dispari e il sottoinsieme dei blocchi di indirizzo pari; l'esecuzione del codice produce come effetto di compattare all'inizio tutti i blocchi che avevano indirizzo dispari e in fondo quelli che avevano indirizzo pari, conservando l'ordine relativo degli indirizzi all'interno di ogni sottoinsieme. Ad esempio:

prima dell'esecuzione:

1	2	3	4	5	6	7	indirizzo
a	b	c	d	e	f	g	contenuto

dopo l'esecuzione:

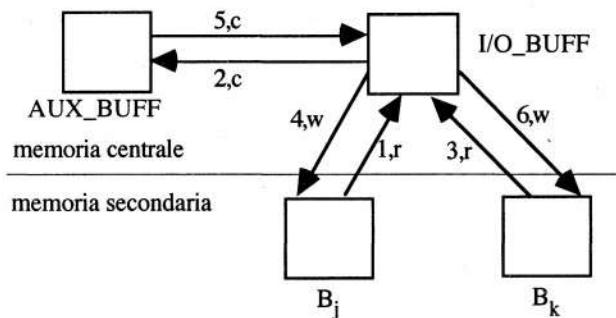
1	2	3	4	5	6	7	indirizzo
a	c	e	g	b	d	f	contenuto

Si deriva facilmente che il numero di chiamate alla procedura `scambia_blocco` è esprimibile, in funzione di NP, come:

$$NC_{\text{pari}} = \sum_{i=1}^{\frac{NP}{2}-1} i = \frac{NP^2 - 2 \times NP}{8}, \text{ se } NP \text{ è pari}$$

$$NC_{\text{dispari}} = \sum_{i=1}^{\frac{NP-1}{2}} i = \frac{NP^2 - 1}{8}, \text{ se } NP \text{ è dispari}$$

Si consideri ora la figura seguente, che illustra l'ordinamento temporale delle operazioni compiute per lo scambio di due generici blocchi B_j e B_k ; a fianco del numero d'ordine dell'operazione viene indicato il tipo (r = lettura di un blocco da disco, w = scrittura di un blocco su disco, c = copia di un blocco in memoria principale).



Considerando che ogni scambio comporta 2 letture da disco e 2 scritture su disco, segue che il numero totale di operazioni è:

per NP pari: $2 \times NC_{\text{pari}}$ letture ; $2 \times NC_{\text{pari}}$ scritture;
 per NP dispari: $2 \times NC_{\text{dispari}}$ letture ; $2 \times NC_{\text{dispari}}$ scritture.

✓ Esercizio 5.10

- 1) Con un singolo utente il caso peggiore si presenta quando il file richiesto si trova su un disco non montato su un drive, pertanto:

$$t_{\text{search}} = t_{\text{exc}} + t_{I/O} = 6.5 \text{ sec}$$

- 2) Con un singolo utente il caso medio si può ricavare osservando che, per le ipotesi fatte, $N_{\text{drive}}/N_{\text{disk}}$ rappresenta la probabilità che il file richiesto sia su un disco già montato sul drive, dunque:

$$t_{\text{search}} = \frac{(t_{\text{exc}} + t_{I/O}) \times (N_{\text{disk}} - N_{\text{drive}}) + t_{I/O} \times N_{\text{drive}}}{N_{\text{disk}}} = 6.26 \text{ sec}$$

- 3) Con U utenti il caso peggiore si verifica quando il file richiesto da ogni utente si trova su un disco non montato su un drive e inoltre la richiesta dell'utente da considerare viene servita per ultima, pertanto nell'ipotesi che l'accesso ai dischi e il trasferimento avvengano in parallelo si ha:

$$t_{\text{search}} = U \times \left(t_{\text{exc}} + \frac{t_{I/O}}{N_{\text{drive}}} \right) = 36.75 \text{ sec}$$

✓ Esercizio 5.11

Un nastro magnetico è ad accesso sequenziale, risulta dunque intuitivo che la scelta ottima consiste nel posizionare, a partire dall'inizio del nastro, i record in ordine non crescente rispetto alla probabilità di essere richiesti. Si può dare una dimostrazione formale di questa affermazione: sia infatti A una possibile generica allocazione e si indichi con B_j^A il valore della chiave del record in posizione j . Il valor medio del costo di accesso a un record è:

$$\overline{C}_{I/O}^A = \sum_{j=1}^n (C_{I/O}(j) \times p(B_j^A)) = \sum_{j=1}^n (j \times p(B_j^A))$$

Si osservi ora che ogni allocazione è ottenibile da un'altra semplicemente scambiando di posizione due record contigui; si consideri pertanto la convenienza di scambiare i record in posizione m e $m+1$ generando una nuova allocazione A' :

$$\overline{C}_{I/O}^{A'} \leq \overline{C}_{I/O}^A$$

se e solo se:

$$m \times p(B_m^{A'}) + (m+1) \times p(B_{m+1}^{A'}) \leq m \times p(B_m^A) + (m+1) \times p(B_{m+1}^A)$$

cioè se e solo se:

$$m \times p(B_{m+1}^A) + (m+1) \times p(B_m^A) \leq m \times p(B_m^A) + (m+1) \times p(B_{m+1}^A)$$

e quindi se e solo se:

$$p(B_m^A) \leq p(B_{m+1}^A)$$

Si deduce dunque che la sequenza ottima è tale per cui $B_j^{A\text{ott}} = k_j$ e in corrispondenza si ha un costo pari a:

$$\overline{C_{I/O}}^{\text{Aott}} = \sum_{j=1}^n \left(j \times p(k_j) \right)$$

caso 1)

$$\overline{C_{I/O}}^{\text{Aott}} = \sum_{j=1}^{n-1} \left(j \times \frac{1}{2^j} \right) + n \times \frac{1}{2^{n-1}}$$

Si faccia riferimento a quanto mostrato nella tabella seguente; si può dimostrare per induzione che:

$$\overline{C_{I/O}}^{\text{Aott}} = 2 - \frac{1}{2^{n-1}}$$

n	$\overline{C_{I/O}}^{\text{Aott}}$
1	$1 = 2 - 1/1$
2	$1/2 + 2/2 = 3/2 = 2 - 1/2$
3	$1/2 + 2/4 + 3/4 = 7/4 = 2 - 1/4$
4	$1/2 + 2/4 + 3/8 + 4/8 = 15/8 = 2 - 1/8$
...
n	$= 2 - 1/2^{(n-1)}$

È sufficiente osservare che:

$$\overline{C_{I/O}}_{n+1}^{\text{Aott}} = \sum_{j=1}^n \left(j \times \frac{1}{2^j} \right) + n \times \frac{1}{2^n} = \overline{C_{I/O}}_n^{\text{Aott}} + \frac{1}{2^n}$$

e quindi se è vero che

$$\overline{C_{I/O}}_n^{\text{Aott}} = 2 - \frac{1}{2^{n-1}}$$

allora si verifica che

$$\overline{C_{I/O}}^{\text{Aott}} = 2 - \frac{1}{2^n}$$

caso 2)

$$\overline{C_{I/O}}^{\text{Aott}} = \sum_{j=1}^n j \times \frac{1}{j} \times \frac{1}{\sum_{g=1}^n g^{-1}} = \sum_{j=1}^n \frac{1}{H_n} = \frac{n}{H_n} \approx \frac{n}{\log_e n}$$

dove H_n (numero armonico) è approssimabile con $\log_e n + \gamma + O(n^{-1})$.

✓ Esercizio 5.12

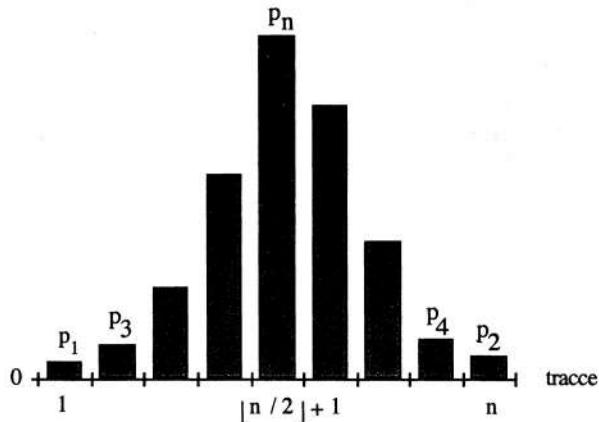
Indicando con B_j il valore della chiave del record in posizione j nell'allocazione sopracitata che rispetta la cosiddetta "legge 80-20", il valor medio del costo di accesso a un record è:

$$\overline{C_{I/O}} = \sum_{j=1}^n (C_{I/O}(j) \times p(B_j)) = \sum_{j=1}^n (j \times p(B_j))$$

che può essere riscritta come:

$$\overline{C_{I/O}} = \sum_{j=1}^{0.2 \times n} \left(j \times \frac{0.8}{0.2 \times n} \right) + \sum_{j=0.2 \times n+1}^n \left(j \times \frac{0.2}{0.8 \times n} \right) = 0.2 \times n + 0.5$$

✓ Esercizio 5.13



Indicando con $d(i,j)$ la distanza di seek per due accessi consecutivi rispettivamente al file in traccia i e al file in traccia j , si ha che il valor medio della distanza di seek è:

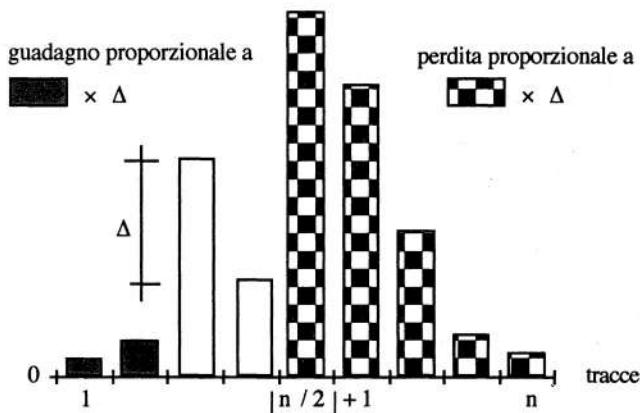
$$\sum_{1 \leq i, j \leq n} p_i \times p_j \times d(i, j)$$

Poichè $d(i,j)$ è una funzione monotona crescente l'allocazione ottima è quella che minimizza globalmente gli spostamenti del braccio, a fronte di richieste d'accesso consecutive.

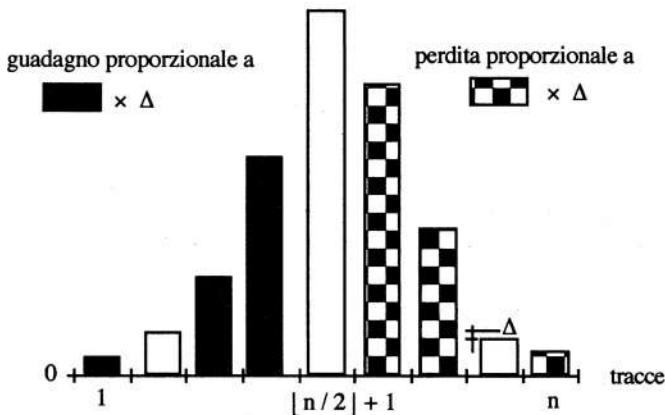
Con riferimento alla figura precedente si possono considerare gli effetti prodotti da uno scambio di posizione di due file e dimostrare qualitativamente che non apportano benefici rispetto all'organizzazione "a canna d'organo".

Si prendono in considerazione i seguenti casi: a) scambio di due file adiacenti, b) scambio di due file non adiacenti e simmetrici rispetto al centro, c) scambio di due file in posizioni arbitrarie.

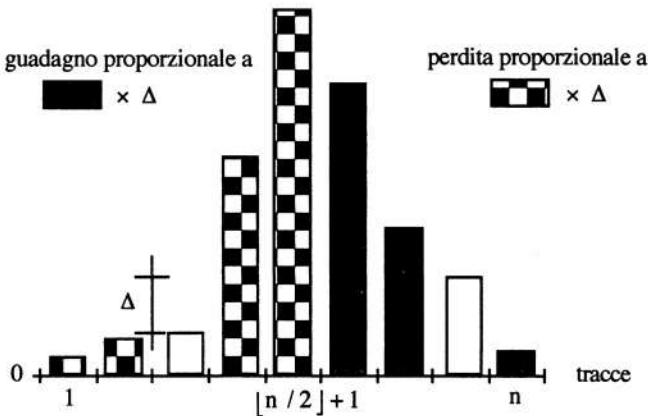
- a) Un esempio di scambio di due file adiacenti è riportato nella figura seguente. Concentrando l'attenzione su accessi consecutivi che hanno la traccia i o la traccia $i+1$ come punto di arrivo o di partenza, si evince chiaramente che la perdita è maggiore del guadagno, e pertanto lo scambio non è conveniente.



- b) Un esempio di scambio di due file non adiacenti simmetrici rispetto al centro è riportato nella figura seguente. Valgono analoghe considerazioni circa la non convenienza dello scambio.



- c) Un esempio di scambio di due file in posizioni arbitrarie è riportato nella figura seguente. Valgono analoghe considerazioni circa la non convenienza dello scambio.



✓ Esercizio 5.14

Trascurando per semplicità la presenza del puntatore alla foglia successiva, si determina il numero delle foglie come:

$$NL = \left\lceil \frac{NK \times \text{len}(k) + NR \times \text{len}(p)}{D \times u} \right\rceil = 5650$$

Assumendo che la lunghezza di un puntatore a un nodo intermedio richieda $\text{len}(q) = \text{len}(p)$ byte, è possibile derivare l'ordine g del B⁺-tree, utilizzando la relazione:

$$2g \times \text{len}(k) + (2g+1) \times \text{len}(q) \leq D$$

da cui:

$$g = \left\lceil \frac{D - \text{len}(q)}{2(\text{len}(k) + \text{len}(q))} \right\rceil = 36$$

Per calcolare l'altezza dell'albero occorre considerare il fatto che, essendo $NK/NL \approx 5.6$, solo una foglia ogni cinque sarà indirizzata al livello $h-1$. Sostituendo NK al posto di NL nella formula per calcolare l'altezza dell'albero si ottiene

$$1 + \lceil \log_2 g + 1 \rceil NK \leq h \leq 2 + \lfloor \log_{g+1} \frac{NK}{2} \rfloor$$

e quindi $h_{\min} = h_{\max} = 3$.

A questo punto, per il calcolo dell'occupazione di memoria si può procedere, in via approssimata, supponendo che al livello intermedio si abbiano $g+1=37$ puntatori. In questo caso si hanno:

5650	blocchi a livello foglie
$\lfloor 1000/37 \rfloor = 27$	blocchi a livello 2
1	radice

Dunque l'occupazione in blocchi del B^+ -tree è pari a 5678 KB.

✓ Esercizio 5.15

Trascurando per semplicità la presenza del puntatore alla foglia successiva, e osservando che ogni pagina dati viene indirizzata in media da kdv puntatori, si determina il numero delle foglie come:

$$NL = \left\lceil \frac{NK \times \text{len}(k) + NP \times kdv \times \text{len}(p)}{D \times u} \right\rceil = 1423$$

✓ Esercizio 5.16

Il caso peggiore si verifica quando si hanno M inserimenti nel B^+ -tree causando M chiamate alla procedura `inserisci_in_indice`. Per ogni inserimento, nell'ipotesi di assenza di splitting, si leggono h nodi, si riscrive una foglia, si legge un blocco dati e si riscrive; dunque il numero totale di operazioni di accesso a blocchi dati e indice è pari a:

$$M \times (h+1+2) = 3 \times M + M \times h$$

✓ Esercizio 5.17

Per la prima cancellazione si leggono h nodi e si riscrive un nodo, per ogni altra cancellazione si leggono $h-1$ nodi e si riscrive un nodo. Dunque si operano $M \times (h - 1) + 1$ letture e h scritture.

Si ricorda che se il B-tree contiene NK chiavi vale la seguente relazione:

$$\lceil \log_{g+1}(NK+1) \rceil \leq h \leq 1 + \lceil \log_{g+1} \frac{NK+1}{2} \rceil$$

Con i dati del problema si deriva che $5 \leq h \leq 6$. Per il caso peggiore si fa riferimento all'altezza massima $h=6$ e quindi in totale si leggono 51 nodi e si riscrivono 10 nodi.

✓ Esercizio 5.18

Due sono le situazioni in cui, a seguito dell'inserimento di un nuovo record nel file dati, è necessario aggiungere un nuovo PID nell'indice:

- il valore del campo j del nuovo record non è già presente nel file e quindi nell'indice stesso;
- il valore del campo j del nuovo record, pur già presente nel file, non è presente nella pagina dati dove il record viene inserito.

Con ipotesi di uniformità e supponendo che ogni pagina contenga mediamente un ugual numero di record pari a NR/NP , la probabilità di aggiornare l'indice a causa di un inserimento di un nuovo record può essere calcolata come:

$$\left(1 - \frac{NK_j}{DK_j}\right) + \frac{NK_j}{DK_j} \times \left(1 - \frac{1}{NK_j}\right)^{NR/NP}$$

✓ Esercizio 5.19

Il posting file memorizza le liste di TID associate a singoli valori di chiave e le foglie del B^+ -tree contengono solo valori di chiave ciascuno seguito dal puntatore alla relativa lista. Pertanto il numero di foglie è pari a:

$$NL = \left\lceil \frac{NK \times \text{len}(k) + NR \times \text{len}(p)}{D \times u} \right\rceil$$

da cui $NL = 34$.

Ora il reperimento dei record nel file dati corrispondenti agli EK valori di chiave comporta i seguenti costi:

- $C_I = h - 1 + \lceil \frac{EK}{NK} NL \rceil = 2$ per l'accesso all'indice
essendo l'altezza massima pari a

$$h \leq 2 + \lceil \log_{g+1} \frac{NL}{2} \rceil = 2$$

- $C_P = EK \times C_{post} = 10 \times C_{post}$ per l'accesso alle EK liste del posting file ;
- $C_D = EK \times \Phi(NR/NK, NP) = 10 * 988 = 9880$ per l'accesso ai dati
assumendo che a ogni valore di chiave siano associati $ER = NR/NK$ record e
utilizzando la formula di Cardenas per stimare il numero medio di pagine dati che
contengono almeno uno degli ER record; si ricorda infatti che:

$$\Phi(ER, NP) = NP \times \left(1 - (1 - 1/NP)^{ER} \right)$$

✓ Esercizio 5.20

L'indice può essere utilizzato per risolvere interrogazioni che riguardano valori di j nell'intervallo $\{k_{NK_j/3+1}, \dots, k_{NK_j}\}$ mentre un'interrogazione che riguarda un valore nell'intervallo $\{k_1, k_2, \dots, k_{NK_j/3}\}$ comporta la scansione dell'intero file. Dai dati del problema si ricava che a ogni valore nell'insieme $\{k_{NK_j/3+1}, \dots, k_{NK_j}\}$ sono associati

$$\left\lceil \frac{0.2 \times NR}{\frac{2}{3} NK_j} \right\rceil$$

record nel file dati e può essere usata la formula di Cardenas per stimare il numero medio di pagine dati che contengono almeno uno di questi record; inoltre assumendo che al numero delle foglie NL dell'indice ogni singolo valore contribuisca con una frazione pari a

$$\left\lceil \frac{NL}{\frac{2}{3} NK_j} \right\rceil$$

si ottiene la seguente espressione per il numero medio di accessi a pagine dati e indice:

$$q \times NP + (1-q) \times \left\{ \Phi \left(\left\lceil \frac{0.2 \times NR}{\frac{2}{3} \times NK_j} \right\rceil, NP \right) + h - 1 + \left\lceil \frac{NL}{\frac{2}{3} \times NK_j} \right\rceil \right\}$$

✓ Esercizio 5.21

La generica interrogazione q_j viene risolta via indice. Per l'accesso all'indice, poiché è sensato assumere che il singolo valore A_i comporti un numero di foglie pari a $\left\lceil \frac{n_i}{NR} NL \right\rceil$, quando q_j interessa il valore A_i si ha un costo pari a

$$C_I = h - 1 + \left\lceil \frac{n_i}{NR} NL \right\rceil$$

Per quanto riguarda l'accesso ai dati si può ricorrere alla formula di Cardenas stimando per la generica interrogazione, quando richiede record con valore A_i dell'attributo A, un costo pari a

$$C_D = \Phi(n_i, NP)$$

Pertanto il costo medio globale per l'insieme Q di interrogazioni è pari a:

$$m \times \left(\sum_{i=1}^{NK_A} \left(p_i \times \left(h - 1 + \left\lceil \frac{n_i}{NR} NL \right\rceil + \Phi(n_i, NP) \right) \right) \right)$$

Bibliografia di base

- A. Albano. *Basi di dati - Strutture e algoritmi*. Addison-Wesley Masson, 1992.
- A. Albano. *Costruire sistemi per basi di dati*. Addison-Wesley Longman, 2001.
- P. Atzeni, S. Ceri, P. Fraternali, S. Paraboschi, R. Torlone. *Basi di dati: architetture e linee di evoluzione*. McGraw-Hill Italia, 2003.
- P. Atzeni, S. Ceri, S. Paraboschi, R. Torlone. *Basi di dati: modelli e linguaggi di interrogazione*. McGraw-Hill Italia, 2002.
- P. Atzeni, V. De Antonellis. *Relational database theory*. Benjamin Cummings, 1993.
- E. Baralis, A. Belussi, G. Psaila. *Basi di Dati: temi d'esame svolti*. Esculapio, 2000.
- L. Baresi, C. Francalanci, F.A. Schreiber, L. Tanca. *Progettazione integrata di dati e funzioni*. Esculapio, 2003.
- C. Batini, S. Ceri, S. Navathe. *Conceptual database design: an Entity-Relationship approach*. Benjamin Cummings, 1991.
- D. Beneventano, S. Bergamaschi, M. Vincini. *Progetto di basi di dati relazionali: lezioni ed esercizi*. Pitagora Ed., 2000.
- L. Cabibbo, R. Torlone, C. Batini. *Basi di dati: progetti ed esercizi svolti*. Pitagora Ed., 1995.
- S. Cannan, G. Otten. *Il manuale SQL*. McGraw-Hill, 1994.
- E.F. Codd. *The relational model for database management*. Addison-Wesley, 1990.
- P. Ciaccia, D. Maio. *Lezioni di basi di dati*. Esculapio, 1997.
- R.A. Elmasri, S.B. Navathe. *Sistemi di basi di dati - Fondamenti*. Addison Wesley, 2004.
- C. Francalanci, F.A. Schreiber, L. Tanca. *Progetto di dati e funzioni*. Esculapio, 1995.
- G. Gardarin, P. Valduriez. *Relational databases and knowledge bases*. Addison-Wesley, 1989.
- J. Gray, A. Reuter. *Transaction processing: concepts and techniques*. Morgan Kaufmann, 1993.
- G. James, W. Paul. *SQL: The complete reference*. McGraw-Hill, 2002.
- L.A. Maciaszek. *Database design and implementation*. Prentice Hall, 1990.
- H. Mannila, K.-J. Räihä. *The design of relational databases*. Addison-Wesley, 1992.
- A.Y. Page. *Relational databases: concepts, selection and implementation*. Sigma Press, 1990.
- R. Ramakrishnan. *Database management systems*. McGraw-Hill, 1998.

- B.J. Salzberg. *File structures: an analytical approach*. Prentice Hall, 1988.
- D. E. Shasha. *Database tuning: a principled approach*. Prentice Hall, 1992.
- J. Ullman. *Basi di dati e basi di conoscenza*. Jackson, 1992.
- R.F. Van der Lans. *Introduzione a SQL*. Addison-Wesley, 2001.

Il volume contiene una raccolta di esercizi che ha l'obiettivo di introdurre il lettore all'impiego di strumenti metodologici per la progettazione e realizzazione di basi di dati relazionali. I temi trattati sono: progettazione concettuale con schemi E/R, progettazione logica, linguaggio SQL e algebra relazionale, stima dei costi di esecuzione e progettazione fisica, dispositivi e organizzazione dei dati.

Gli esercizi sono ampiamente commentati, spesso arricchendo la discussione con proposte di soluzioni alternative e suggerimenti per ulteriori sviluppi. La tipologia degli argomenti affrontati e il livello di approfondimento fanno sì che il volume rappresenti un valido complemento per un corso di basi di dati di primo livello.

La seconda edizione aggiunge più di sessanta esercizi, rendendo ancor più ampia la casistica delle situazioni di progetto affrontate.

DARIO MAIO è Professore Ordinario presso il Dipartimento di Elettronica, Informatica e Sistemistica dell'Università di Bologna e docente di Basi di Dati presso il Corso di Laurea in Scienze dell'Informazione a Cesena.

STEFANO RIZZI è Professore Associato presso il Dipartimento di Elettronica, Informatica e Sistemistica dell'Università di Bologna e docente di Ingegneria del Software e Sistemi Informativi Avanzati presso il Corso di laurea in Scienze dell'Informazione a Cesena.

ANNALISA FRANCO è dottore di ricerca presso il Dipartimento di Elettronica, Informatica e Sistemistica dell'Università di Bologna e tutor di Basi di Dati presso il Corso di Laurea in Scienze dell'informazione a Cesena.

ISBN 88-7488-113-4



9 788874 881130

Euro 19,00



SOCIETÀ EDITRICE
ESCOLAPIO

www.editrice-esculapio.it