Dipartimento di Ingegneria Elettrica, Gestionale e Meccanica
Università di Udine

Dispense del corso di

Basi di Dati

Anno Accademico 2012-13

Compiti d'esame (svolti e non)

a cura di
Andrea Schaerf

12 giugno 2013

Introduzione

In questa prima parte vengono fornite informazioni generali sul corso. Successivamente vengono forniti i testi dei compiti, ed infine le soluzioni di molti di essi.

Programma del corso e libro di testo

Il programma del corso si articola nei seguenti argomenti.

1. Introduzione alle basi di dati
2. Il modello relazionale
   - tabelle e attributi
   - valori nulli
   - chiavi e vincoli
3. L'algebra relazionale
   - operatori dell'algebra
   - interrogazione in algebra relazionale
4. Il linguaggio SQL
   - interrogazione in SQL
     - interrogazioni di base
     - interrogazioni annidate
     - operatori di aggregazione
   - definizioni e aggiornamenti
   - utilizzo del DBMS PostgreSQL
5. Progettazione di basi di dati
   - il modello Entità-Relazione
   - progettazione concettuale
   - progettazione logica
   - normalizzazione: dipendenze funzionali, BCNF, 3NF, decomposizioni, proprietà delle decomposizioni

Libro di testo: P. Atzeni, S. Ceri, S. Paraboschi, R. Torlone, Basi di dati, Modelli e linguaggi di interrogazione, McGraw-Hill Italia. Parti del testo da saltare:

Ied (2002): Paragrafi 3.3, 4.5, 4.6, 4.7, 4.8, 5.2, 5.3, 5.3, 7.7, 8.5 e 8.6 e le appendici.

IIed (2006): Paragrafi 3.2, 3.3, 5.1.6, 5.2, 5.3, 5.4, 6.1, 6.2, 6.4, 8.7, 9.5, 9.6 e le appendici.

IIIed (2009): Paragrafi 3.2, 3.3, 5.1.6, 5.2, 5.3, 5.4, 6.1, 6.2, 6.4, 7.4, 8.7, 9.6, 10.1, 10.2 e le appendici A e B.

Modalità d'esame

L'esame di Basi di Dati si compone di 3 parti:

Compito scritto: Lo scritto si compone di un insieme di domande ed esercizi su tutti gli argomenti svolti in classe. La durata della prova scritta è normalmente di 2 ore.

Elaborato svolto a casa: Lo studente deve necessariamente svolgere almeno una (a scelta) tra le seguenti due attività:

Tesina: La tesina riguarda la progettazione di una base di dati e il suo sviluppo in PostgreSQL (o altro DBMS relazionale a scelta). Il lavoro deve essere svolto individualmente o in coppia. L'argomento del progetto viene proposto dallo studente e concordato con il docente. L'elaborato viene consegnato allo scritto e discusso all'orale.

Verifica del compito: Nel tempo che intercorre tra il compito scritto e la prova orale (tipicamente 3-4 giorni) lo studente è tenuto a verificare la sua soluzione di parte (comunicata in classe) del compito.

La soluzione a casa consiste della realizzazione della base di dati presentata sul testo e nell'esecuzione delle interrogazioni svolte dallo studente in classe. Nel caso la soluzione svolta in classe non sia corretta, questa deve essere modificata opportunamente fino ad ottenere il corretto funzionamento.

Nel caso lo studente non abbia svolto un esercizio nel compito, dovrà comunque portare la soluzione svolta a casa; tale soluzione può ovviamente essere qualsiasi.

Prova orale: La prova orale consiste in una discussione dell'elaborato svolto a casa e dello scritto ed un approfondimento di alcuni esercizi, nonché altre eventuali domande per verificare la correttezza della valutazione delle altre prove in base alla effettiva preparazione dimostrata dal candidato.

Le tre prove devono essere svolte nel medesimo appello. Se uno studente non supera la prova scritta, l'eventuale tesina non viene valutata e deve essere ripresentata (eventualmente modificato) nell'appello in cui lo studente si ripresenta per sostenere l'esame.

Per gli studenti del corso integrato, gli esami dei due moduli possono essere svolti anche in appelli separati. E' inoltre possibile svolgerli nello stesso appello anche se gli esami si tengono nella stessa data e nello stesso orario. Lo studente che intenda farli nello stesso appello è pregato di avvisare preventivamente i docenti. Il voto viene registrato quando sono stati superati entrambi i moduli.

Altro materiale didattico

All'indirizzo `http://www.diegm.uniud.it/schaerf/BasiDati/` sono disponibili:

- Testi delle esercitazioni
- Sorgenti SQL delle basi di dati necessarie per svolgere le esercitazioni
- Lucidi di alcune lezioni
- Copia elettronica in formato PDF della presente dispensa

Commenti generali agli esercizi e alle soluzioni proposte

- Non tutti gli esercizi sono svolti. Le soluzioni mancanti verranno aggiunte in una futura versione della presente dispensa.
- Quasi tutti gli esercizi proposti hanno più soluzioni alternative ugualmente valide. Le soluzioni proposte qui rappresentano quindi solo una tra le varie possibilità. In particolare, per le interrogazioni in SQL, lo studente è vivamente invitato a verificare la proprie soluzioni tramite un DBMS.
- In molti casi le specifiche proposte nel testo sono incomplete. In tale situazione lo studente è tenuto a completare le specifiche facendo delle ipotesi ragionevoli sul dominio di interesse, *scrivere tali specifiche sul compito*, e svolgere l'esercizio in base ad esse.

- Molte, ma non tutte, delle interrogazioni in SQL presentate in questa dispensa sono state verificate effettivamente su un DBMS da parte del docente. Non si esclude comunque che possano esserci degli errori. Gli studenti sono quindi vivamente invitati a verificate non solo le proprie soluzioni, ma anche quelle riportate in questa dispensa, e di segnalare eventuali errori al docente.

Testi

Compito del 12 gennaio 2000 (soluzione a pagina 83)

Esercizio 1 (punti 2) Calcolare il risultato della seguente espressione algebrica

$$(\pi_{BD}(R1 \bowtie_{A=C} R2)) \cup (\rho_{D \leftarrow E}\pi_{BE}(R2 \bowtie_{D=E} (\rho_{EF \leftarrow CD}R2) \bowtie_{F=A} R1))$$

nei seguenti due casi (evidenziando anche alcuni dei risultati intermedi)

1. $R1(A, B) = \{(a, 1)\}$, $R2(C, D) = \{(a, a)\}$.
2. $R1(A, B) = \{(a, 10), (b, 5), (c, 12), (d, 13)\}$, $R2(C, D) = \{(a, b), (a, c), (b, d), (c, c)\}$

Si consideri il seguente schema relazionale di basi di dati:

SQUADRA(Nome,Citta)
PARTITA(SquadraDiCasa,SquadraOspite,GoalCasa,GoalOspiti,Turno)

Esercizio 2 (punti 3) Formulare la seguente interrogazione in SQL: "Trovare il numero di vittorie dell'Udinese contro squadre di Roma".

Esercizio 3 (punti 3) Formulare la seguente interrogazione in SQL: "Trovare per ciascun turno la media dei goal fatti dalla squadra di casa escludendo le partite il cui la squadra ospite è di Milano, e non considerando i turni in cui ci sia almeno una partita in cui la squadra di casa ha segnato 5 o più goal".

Esercizio 4 (punti 1) Si ricavi il diagramma ER dallo schema relazionale precedente.

Esercizio 5 (punti 3) Si modifichi il diagramma ER dell'esercizio precedente aggiungendo, nel modo che si ritiene opportuno, i dati sulla classifica corrente del campionato e sugli stadi (con nome e capienza). Si consideri anche l'eventualità che una partita si giochi in uno stadio diverso da quello della squadra ("campo neutro"), e in questo caso se ne memorizzi anche il motivo (impraticabilità del campo, squalifica, ...).

Compito del 26 gennaio 2000 (soluzione a pagina 84)

Si consideri il seguente schema relazionale di basi di dati che memorizza l'orario delle lezioni della facoltà.

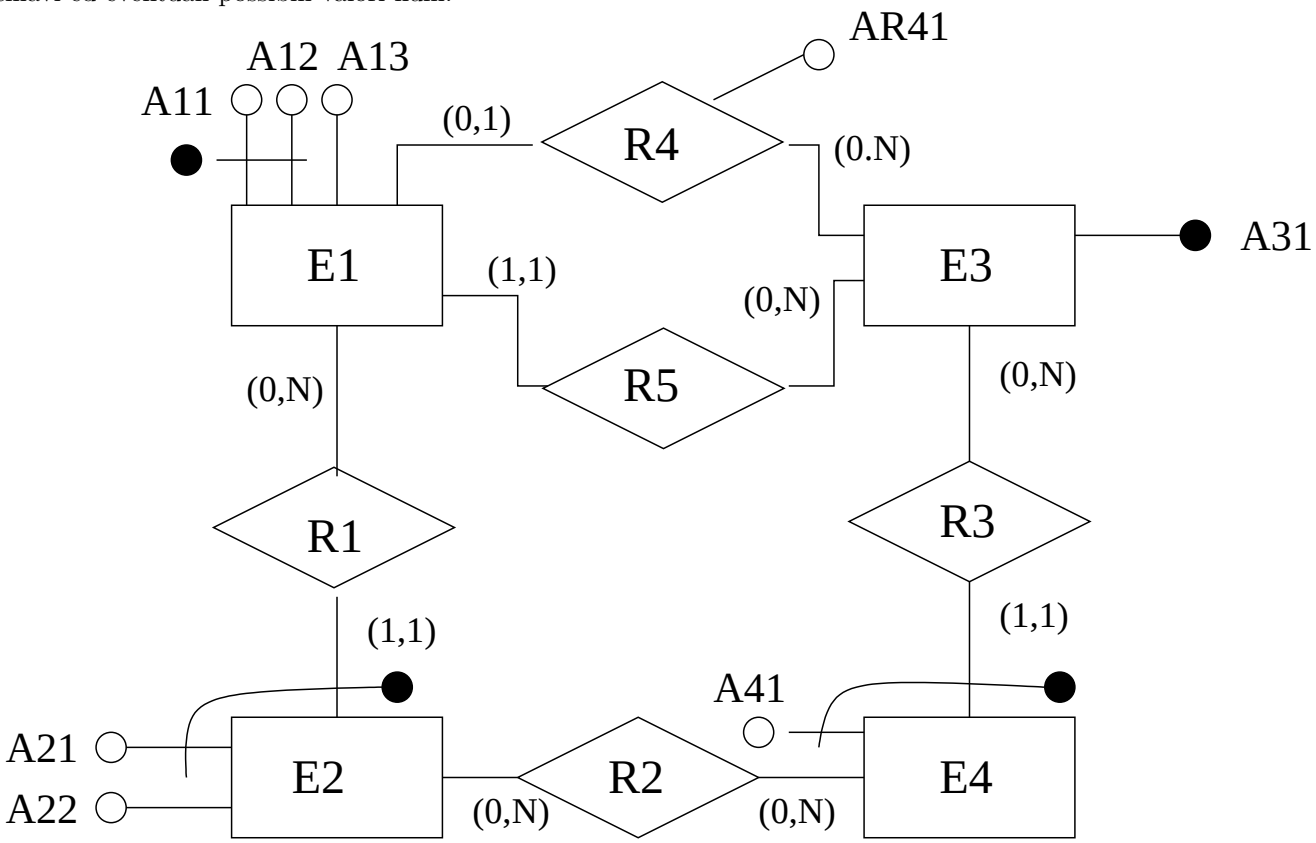
CORSO(Codice,Nome,Docente)
LEZIONE(CodCorso, CodPeriodo, Aula)
PERIODO(Codice,Giorno,OraInizio)

Esercizio 1 (punti 2) Formulare la seguente interrogazione in algebra relazionale: "Trovare i docenti che insegnano tre o più corsi"

Esercizio 2 (punti 3) Formulare la seguente interrogazione in SQL: "Trovare per ciascun docente che insegna esattamente due corsi il numero di lezioni che tiene tra Lunedì e Martedì in aula A".

Esercizio 3 (punti 2) Scrivere delle istruzioni in SQL che modifichino l'orario di inizio delle lezioni della mattina del Venerdì dalle 09:00 alle 09:15 e dalle 11:00 alle 11:15. Si assuma che l'ora sia memorizzata tramite una stringa di 5 caratteri.

Esercizio 4 (punti 3) Tradurre il seguente schema ER in uno schema relazionale. Si identifichino le chiavi ed eventuali possibili valori nulli.



Esercizio 5 (punti 2) Si descriva il concetto di decomposizione senza perdita (*lossless*). Si dia un esempio di decomposizione con perdita ed uno di decomposizione senza perdita.

Compito del 9 febbraio 2000 **(soluzione a pagina 85)**

Si consideri il seguente schema relazionale di basi di dati

PERSONA(CF, Nome, Cognome, Età)
CITTÀ(Nome, NumeroAbitanti)
HAABITATO(CFPersona, NomeCittà, Da_Anno, A_Anno)
HALAVORATO(CFPersona, NomeCittà, P_IVA_Ditta, Da_Anno, A_Anno)
DITTA(P_IVA, NumeroImpiegati, CapitaleSociale)
HAAVUTOSEDEIN(P_IVA_Ditta, NomeCittà, Da_Anno, A_Anno)

Esercizio 1 (punti 4) Progettare uno schema ER la cui traduzione dia luogo allo schema relazionale dato, nelle seguenti ipotesi:

- Ogni città presente nella base di dati ha almeno una ditta che vi ha avuto sede
- Ogni città presente nella base di dati ha almeno una persona che vi ha abitato o lavorato.
- Per ogni persona è specificata almeno una città un cui ha lavorato ed una in cui ha abitato.
- Per ogni ditta è specificata almeno una sede che ha avuto.

Esercizio 2 (punti 2) Scrivere delle istruzioni in SQL per creare le tabelle inclusi i vincoli di integrità referenziale, nelle ipotesi dell'esercizio precedente.

Esercizio 3 (punti 3) Formulare la seguente interrogazione in SQL: "Trovare le partite IVA delle ditte che hanno (o hanno avuto) sedi in tutte le città".

Esercizio 4 (punti 3) Formulare la seguente interrogazione in SQL: "Per ogni ditta trovare il numero degli impiegati che vi hanno lavorato, risiedendo nella stessa città dove aveva sede la ditta."

Compito del 2 giugno 2000 (soluzione a pagina 87)

Si consideri il seguente schema relazionale di basi di dati, in cui l'attributo Restituito ha un valore booleano

CD(Codice,Autore,Titolo,Durata)
AFFITTO(CD,Cliente,Data,Restituito)
CLIENTE(Codice,Nome,Città)

Esercizio 1 (punti 2) Formulare la seguente interrogazione in algebra relazionale: "Trovare il codice dei CD di durata più lunga"

Esercizio 2 (punti 2) Formulare la seguente interrogazione in SQL: "Trovare gli autori di cui almeno un CD è in affitto, mostrando il nome dell'autore e del cliente"

Esercizio 3 (punti 2) Formulare la seguente interrogazione in SQL: "Trovare per ciascun autore il numero di CD che sono in catalogo"

Esercizio 4 (punti 6) Progettare lo schema Entità-Relazione dell'applicazione descritta mediante le seguenti specifiche: Un circolo di tennis organizza un torneo di doppio tra i propri soci. Il torneo è ad eliminazione diretta: la coppia vincitrice di una partita passa al turno successivo. Di ogni socio si conoscono nome, cognome, indirizzo, livello di gioco e gli anni di appartenenza al circolo. Gli arbitri delle partite sono soci che non partecipano al torneo. Un socio può giocare in più squadre (coppie di giocatori). Il circolo organizza il calendario delle partite che comunque, per cause tecniche, possono essere giocate anche in data differente da quella prevista. Di ogni partita interessa conoscere: il turno a cui si riferisce (eliminatorie, sedicesimi, ottavi, quarti, semifinali, finali), la data in cui doveva essere giocata e quella in cui è stata effettivamente giocata, se diversa, le coppie che la hanno disputata, nonché il punteggio.

Compito del 17 luglio 2000 (soluzione a pagina 88)

Si consideri il seguente schema relazionale di basi di dati):

IMPIEGATO(Matricola, Cognome, Eta, Salario)
LAVORA(Matricola, Codice, PercentualeTempo)
DIPARTIMENTO(Codice, Nome, Budget, MatricolaManager)

Esercizio 1 (punti 2) Scrivere le istruzioni SQL necessarie per creare lo schema suddetto, specificando anche le chiavi e i vincoli di integrità referenziale che si reputano necessari. Si consideri anche il vincolo che ogni dipartimento abbia sempre un manager.

Esercizio 2 (punti 1) Scrivere l'istruzione SQL opportuna per inserire l'impiegato "Rossi", con matricola "112", età "25" e salario "35.000"

Esercizio 3 (punti 1) Scrivere l'istruzione SQL opportuna per aumentare il salario di tutti gli impiegati del 20%.

Esercizio 4 (punti 2) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale suddetto completo di tutti i vincoli finora menzionati.

Esercizio 5 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire la matricola degli impiegati che lavorano in uno o più dipartimenti e sono manager di un altro".

Esercizio 6 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire il nome dei dipartimenti il cui budget è inferiore alla somma dei salari degli impiegati che vi lavorano".

Esercizio 7 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire i cognomi di quelli tra gli impiegati più giovani che lavorano per la maggiore percentuale di tempo."

Compito del 6 settembre 2000 (soluzione a pagina 90)

Si consideri il seguente schema relazionale di basi di dati:

IMPIEGATO(Matricola, Cognome, Eta, Salario)
LAVORA(Matricola, Codice, PercentualeTempo)
DIPARTIMENTO(Codice, Nome, Budget, MatricolaManager, MatricolaViceManager)

Esercizio 1 (punti 2) Scrivere le istruzioni SQL opportune per inserire l'impiegato "Verdi", con matricola "112", età ignota e salario "35.000" che lavori per il dipartimento "Vendite" al 70% del suo tempo.

Esercizio 2 (punti 2) Scrivere l'istruzione SQL opportuna per inserire tutti gli impiegati di 25 anni o di età ignota nel dipartimento "Vendite" al 10% del loro tempo.

Esercizio 3 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire la matricola degli impiegati che lavorano in uno o più dipartimenti e sono vicemanager di un altro".

Esercizio 4 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire il nome dei dipartimenti il cui budget è inferiore alla somma dei salari degli impiegati che vi lavorano oltre il 50% del loro tempo".

Esercizio 5 (punti 2) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale considerato, completo dei vincoli di cardinalità che si reputano necessari. Si assuma che sia il manager che il suo vice siano degli impiegati e che ogni dipartimento abbia sempre un manager.

Esercizio 6 (punti 2) Si modifichi lo schema Entità-Relazione dell'esercizio precedente inserendo il concetto di ruolo con cui un impiegato lavora in un dipartimento. Ad esempio, "Verdi" può essere "Analista" nel dipartimento "Vendite" e "Programmatore" nel dipartimento "Acquisti". Si consideri anche la possibilità che un impiegato lavori per lo stesso dipartimento con due ruoli diversi.

Compito del 8 gennaio 2001 (soluzione a pagina 92)

Si consideri il seguente schema relazionale di basi di dati:

IMPIEGATO(Matricola, Cognome, Eta, Salario)
LAVORA(Matricola, Codice, PercentualeTempo)
DIPARTIMENTO(Codice, Nome, Budget, MatricolaManager)

Esercizio 1 (punti 3) Esprimere in SQL la seguente interrogazione: "Fornire la matricola degli impiegati che lavorano in due (o più) dipartimenti e sono manager di un altro".

Esercizio 2 (punti 2) Esprimere in SQL la seguente interrogazione: "Fornire il nome dei dipartimenti il cui budget è inferiore alla somma dei salari degli impiegati che vi lavorano il cui cognome inizia per R".

Esercizio 3 (punti 7) Progettare lo schema Entità-Relazione dell'applicazione descritta mediante le seguenti specifiche. Un club nautico ha necessità di progettare una base di dati per memorizzare e gestire le informazioni sulle sue imbarcazioni e sui suoi dipendenti. Tali informazioni riguardano:

- Imbarcazioni: ogni imbarcazione ha un numero di matricola ed è di un certo modello. Esistono vari modelli di imbarcazione che possono essere ospitati nel club nautico, ognuno di essi è identificato da un codice e sono noti per esso lunghezza, stazza e pescaggio.
- Tecnici che lavorano per il club nautico. Per ognuno di essi si vuole memorizzare codice fiscale, nome, cognome, indirizzo, telefono e salario. Inoltre, ogni tecnico è esperto di uno o più modelli di imbarcazione.
- Personale marittimo. Anche di questi si vogliono memorizzare le stesse informazioni personali descritte per i tecnici. Inoltre, il personale marittimo deve sottoporsi a periodiche verifiche mediche e per ogni componente si vuole memorizzare la data dell'ultima visita. Infine, il personale marittimo comprende i "capitani" delle imbarcazioni, ognuno di essi è abilitato a comandare uno o più modelli di imbarcazione.
- Test di abilitazione alla navigazione delle imbarcazioni. Ogni test ha un codice, un nome e un punteggio massimo. Per ogni test effettuato su di una certa imbarcazione è necessario memorizzare anche le informazioni riguardanti il tecnico che lo ha eseguito, la data in cui è stato effettuato, il tempo impiegato per effettuarlo ed il punteggio assegnato all'imbarcazione.

Compito del 19 marzo 2001 **(soluzione a pagina 93)**

Esercizio 1 (punti 4) Si consideri il seguenti schema di base di dati relazionale che rappresenta una realtà bancaria, e i relativi vincoli di integrità referenziale

Persone(<u>CodiceFiscale</u>,Cognome,Nome,DataDiNascita)
Dipendenti(<u>CodiceFiscale</u>,Filiale,Qualifica)
Qualifiche(<u>Codice</u>,Descrizione)
Filiali(<u>Codice</u>,Città,Direttore)
Agenzie (<u>Numero</u>,<u>Filiale</u>,Indirizzo,Reggente)
ContiCorrenti (<u>Numero</u>,Agenzia,Filiale)
Titolarita-CC (<u>Conto</u>,<u>Titolare</u>)

Dipendenti(CodiceFiscale) $\subseteq$ Persone(CodiceFiscale)
Dipendenti(Qualifica) $\subseteq$ Qualifiche(Codice)
Dipendenti(Filiale) $\subseteq$ Filiali(Codice)
Filiali(Direttore) $\subseteq$ Dipendenti(CodiceFiscale)
Agenzie(Filiale) $\subseteq$ Filiali(Codice)
Agenzie(Reggente) $\subseteq$ Dipendenti(CodiceFiscale)
ContiCorrenti(Agenzia,Filiale) $\subseteq$ Agenzie(Numero,Filiale)
Titolarita-CC(Conto) $\subseteq$ ContiCorrenti(Numero)
Titolarita-CC(Titolare) $\subseteq$ Persone(CodiceFiscale)

Si ricavi lo schema ER da cui tale schema relazionale proviene.

Esercizio 2 (punti 2) Con riferimento allo schema relazionale precedente, specificare la seguente interrogazione in algebra relazionale: "Per ogni agenzia, mostrare numero, codice della filiale, città e cognome del reggente."

Esercizio 3 (punti 2) Con riferimento allo schema relazionale precedente, specificare la seguente interrogazione in SQL: "Trovare i conti correnti che hanno due o più titolari, mostrandone numero, agenzia e filiale."

Esercizio 4 (punti 2) Con riferimento allo schema relazionale precedente, specificare la seguente interrogazione in SQL: "Trovare le agenzie presso le quali non esiste alcun conto corrente mostrandone numero e filiale."

Esercizio 5 (punti 2) Considerare le relazioni $R_1(\underline{A}, B, C)$ e $R_2(\underline{D}, E, F)$ aventi rispettivamente cardinalità $c1$ e $c2$. Assumere che sia definito un vincolo di integrità referenziale fra l'attributo C di $R1$ e la chiave D di $R2$. Indicare la cardinalità minima e massima di ciascuno dei seguenti join:

1. $R1 \bowtie_{A=D} R2$ .....
2. $R1 \bowtie_{C=D} R2$ .....
3. $R1 \bowtie_{A=F} R2$ .....

Compito del 26 marzo 2001 **(soluzione a pagina 94)**

Esercizio 1 (punti 2) Considerare uno schema di base di dati relazionale contenente le seguenti relazioni:

INSEGNAMENTI(Codice,Denominazione)
STUDENTI(Matricola,Cognome,Nome)
ESAMI(Studente,Corso,Data,Voto)

specificare la seguente interrogazione in algebra relazionale: "trovare denominazione, data e voto per gli esami superati da Bartolomeo Pestalozzi".

Esercizio 2 (punti 2) Con riferimento allo schema relazionale precedente, specificare la seguente interrogazione in SQL: "trovare la media dei voti riportati agli esami per ciascun insegnamento (indicando codice, denominazione e voto medio)"

Esercizio 3 (punti 2) Con riferimento allo schema relazionale precedente, specificare la seguente interrogazione in SQL: "trovare gli studenti (mostrando il numero di matricola) che hanno superato almeno due esami dopo il 1/1/2000."

Esercizio 4 (punti 5) Mostrare uno schema ER che modelli la seguente realtà (scegliendo liberamente per gli aspetti lasciati indefiniti o ambigui nelle specifiche):

- oggetto dell'interesse è una serie di festival cinematografici (più precisamente, si tratta delle varie edizioni, in anni diversi, dello stesso festival) ciascuno organizzato in rassegne, di cui una (per ciascun festival) destinata ai film in concorso e altre su temi diversi (identificate attraverso un nome); ogni film viene presentato in una sola rassegna di un solo festival;
- per ogni film ci possono essere diverse proiezioni, di cui e importante indicare sala, giorno, ora e incasso;
- per ogni festival, sono assegnati i seguenti premi: miglior film, miglior regia, miglior attore e miglior attrice (protagonista e non protagonista).

Progettare anche lo schema relazionale corrispondente allo schema concettuale definito.

Esercizio 5 (punti 1) Considerare una relazione $R(A, \underline{B, C}, D, E)$. Indicare quali delle seguenti proiezioni hanno certamente lo stesso numero di tuple di R:

- $\pi_{ABCD}(R)$ ☐ Sì ☐ No
- $\pi_{AC}(R)$ ☐ Sì ☐ No
- $\pi_{BC}(R)$ ☐ Sì ☐ No
- $\pi_{C}(R)$ ☐ Sì ☐ No
- $\pi_{CD}(R)$ ☐ Sì ☐ No

Compito del 2 luglio 2001 (soluzione a pagina 96)

Esercizio 1 (punti 5) Si deve progettare la base di dati di una compagnia aerea nazionale, per la quale sono di interesse:

- I voli, con codice, durata in minuti, aeroporto di partenza e aeroporto di arrivo. Alcuni voli prevedono tappe intermedie in aeroporti diversi da quelli di partenza e arrivo, e delle tappe intermedie di un volo interessa l'ordine con cui esse si susseguono (ad esempio, il volo 124 da Milano Linate a Palermo Punta Raisi prevede prima l'aeroporto di Bologna e poi quello di Napoli come tappe intermedie). Ogni volo (sia esso diretto, oppure con tappe intermedie) appartiene ad uno ed uno solo dei seguenti tipi:
  - giornaliero (di questi voli interessa l'orario di partenza),
  - settimanali (di questi voli interessa il giorno della settimana e l'orario di partenza),
  - mensile (di questi voli interessa il giorno del mese, l'orario di partenza, e anche le regioni sorvolate).
- Gli aeroporti, con codice, nome, categoria e città (a sua volta con nome, numero di abitanti e regione).

Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale.

Esercizio 3 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in SQL: "Trovare i voli che partono da un aeroporto situato a Roma".

Esercizio 4 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in SQL: "Trovare i voli che sorvolano il Veneto ma che partono da un aeroporto fuori dal Veneto."

Compito del 9 luglio 2001 (soluzione a pagina 97)

Esercizio 1 (punti 5) La base di dati di un'agenzia di pubbliche relazioni deve contenere le seguenti informazioni:

- un catalogo di clienti (codice fiscale, città, indirizzo, telefono), che possono essere aziende o persone fisiche;
- un insieme di informazioni su dei banchetti (codice, data, costo, numero partecipanti) organizzati nell'ambito di manifestazioni o in occasione di singoli avvenimenti (congressi, matrimoni, cresime, ecc.) per conto dei clienti; ogni banchetto si tiene in un ristorante e prevede un certo menù.
- gli invitati ai singoli banchetti (ogni invitato ha un codice all'interno del banchetto);

- un elenco di ristoranti con le loro caratteristiche (nome, località, numero posti). I ristoranti offrono un certo insieme di menù, ciascuno costituito da portate. Ogni menù ha un codice all'interno del ristorante che lo offre; delle portate interessa il nome e il tipo (pesce, pasta, ecc.).

Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale. Seguire l'indicazione di evitare valori nulli nella base di dati.

Esercizio 3 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in algebra relazionale: "Dato un ristorante, trovare tutti coloro che sono stati invitati ad un banchetto tenutosi in quel ristorante".

Esercizio 4 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in SQL: "Trovare i banchetti con un numero di partecipanti esattamente uguale al numero di posti del ristorante in cui si tiene".

Compito del 23 luglio 2001 (soluzione a pagina 98)

Esercizio 1 (punti 5) L'applicazione a cui si fa riferimento riguarda i concorsi pubblici, ed è descritta dai seguenti requisiti. Ogni concorso è identificato da un codice, prevede un certo numero di prove, un certo numero di vincitori, ed è indetto mediante un bando. Un bando è relativo a uno o più concorsi, ha una data di pubblicazione, un codice e una data di scadenza per la presentazione delle domande di ammissione. Per ogni concorso si nomina una commissione, formata da un certo numero di membri, uno dei quali è presidente, ed un altro è segretario. Dei membri della commissione interessa: codice fiscale, nome cognome, indirizzo e numeri di telefono. Al bando del concorso rispondono i candidati, dei quali interessa: codice fiscale, nome e cognome. Un concorso prevede una o più prove, ciascuna in una certa data, e con un certo punteggio massimo. Dei candidati che si presentano alle prove interessa anche l'indirizzo e il numero di telefono. Ogni candidato che effettua una prova totalizza un certo punteggio per quella prova. Alla conclusione del concorso viene stilata la graduatoria, nella quale ogni candidato che si è presentato a tutte le prove compare con il relativo punteggio totale, ottenuto come somma dei punteggi che ha totalizzato alle prove.

Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale. Seguire l'indicazione di evitare valori nulli nella base di dati.

Esercizio 3 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in SQL: "Trovare i presidenti dei concorsi il cui bando ha una data di pubblicazione posteriore al 22 maggio 2001".

Esercizio 4 (punti 2) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in SQL: "Dato un concorso, trovare codice fiscale, nome e cognome del secondo in graduatoria.".

Compito del 6 settembre 2001 (soluzione a pagina 99)

Esercizio 1 (punti 6) Per ciascuno dei seguenti schemi logici (in cui A* indica che l'attributo A ammette valori nulli), mostrare uno schema concettuale dal quale possa essere stato ottenuto, indicando anche cardinalità e identificatori.

Schema (a)

- Libri(Codice, Titolo, Genere, Autore) con vincolo di integrità referenziale fra Autore e la chiave della relazione Scrittori
- Edizioni(Libro, Editore, Collana*, Anno) con vincoli di integrità referenziale fra Libro e la chiave della relazione Libri e fra Editore e la chiave della relazione Editori
- Editori(Sigla, Nome, Città)
- Scrittori(Codice, Cognome, Nome)

Schema (b)

- Libri(Codice, Titolo, Genere*) con vincolo di integrità referenziale fra Genere e la chiave della relazione Generi
- Edizioni(Libro, Editore, Collana*, Anno) con vincoli di integrità referenziale fra Libro e la chiave della relazione Libri, fra Editore e la chiave della relazione Editori e fra Collana e la chiave della relazione Collane
- Autori(Libro, Scrittore) con vincoli di integrità referenziale fra Libro e la chiave della relazione Libri e fra Scrittore e la chiave della relazione Scrittori
- Collane(SiglaCollana, Nome)
- Generi(SiglaGenere, Nome)
- Editori e Scrittori come nello schema (a)

Esercizio 2 (punti 4) Con riferimento allo schema (a) nella domanda precedente, formulare sia in algebra relazionale che in SQL la seguente interrogazione: "Trovare i cognomi e i nomi degli autori di libri pubblicati da editori di Milano".

Esercizio 3 (punti 2) Con riferimento allo schema (a) nella domanda precedente, formulare in SQL la seguente interrogazione: "Trovare i codici degli scrittori che hanno pubblicato con un solo editore (mostrando anche la sigla dell'editore)".

Compito del 20 settembre 2001 **(soluzione a pagina 101)**

Esercizio 1 (punti 2) Sia dato lo schema di una base di dati relativa a studenti ed esami da essi superati:

- Studenti(Matricola, Cognome, Nome)
- Esami(Studente,Materia,Voto,Data) con vincolo di integrità referenziale fra l'attributo Studente di Esami e la chiave della relazione Studenti.

Considerare la seguente espressione in algebra relazionale:

$$\pi_{Matricola,Cognome,Nome}(Studenti \bowtie_{Matricola=Studente} \sigma_{Voto=30}(Esami))$$

Descriverla in linguaggio naturale e formulare la stessa interrogazione in SQL

Esercizio 2 (punti 2) Dato lo schema dell'esercizio precedente, formulare in SQL l'interrogazione che mostra, per ogni studente, matricola, cognome e media dei voti riportati negli esami superati.

Esercizio 3 (punti 4) Definire uno schema Entità-Relazione che descriva i dati di una applicazione relativa ai listini prezzi di un insieme di case automobilistiche. Sono di interesse:

- Le case produttrici, con nome (identificante) e indirizzo.
- I modelli (ad esempio la Punto o la Golf), con nome (identificante), anno di lancio e segmento di mercato (codificato con una lettera e con una breve descrizione: ad esempio, al segmento A corrisponde la descrizione utilitaria).

- Le versioni dei modelli, identificate attraverso il nome del modello e un nome specifico (ad esempio la Punto 75S). Per ogni versione sono rilevanti il prezzo, il motore, la cilindrata, la potenza, il numero di porte e la velocità massima. Ogni versione di modello ha almeno un motore.
- I motori (ad esempio il motore Fire 1000), identificati attraverso un codice e con le seguenti proprietà: cilindrata, numero cilindri e potenza. Possono esistere motori (attualmente) non utilizzati in alcun modello.

Esercizio 4 (punti 2) Ricavare lo schema relazionale corrispondente allo schema ER nell'esercizio precedente.

Esercizio 5 (punti 2) Indicare quali fra le seguenti affermazioni sono vere:

1. □ ogni relazione ha almeno una chiave
2. □ ogni relazione ha esattamente una chiave
3. □ ogni attributo appartiene al massimo ad una chiave
4. □ possono esistere attributi che non appartengono a nessuna chiave
5. □ una chiave può essere sottoinsieme di un'altra

Compito del 10 dicembre 2001 **(soluzione a pagina 102)**

Esercizio 1 (punti 6) Definire uno schema Entità-Relazione che descriva i dati di un'applicazione per gestire un'agenzia matrimoniale. Le informazioni di interesse riguardano:

I clienti: uomini e donne, di cui interessa il numero di tessera, il nome (inteso come nome e cognome), l'età, l'aspetto fisico (piacevole, interessante, bello, ...), l'insieme dei suoi interessi (es. cinema, arte, vela, ...), ed eventualmente la sua professione. I clienti si dividono (oltre che in uomini e donne) anche in clienti *regolari* e clienti *in prova*.

Le opzioni: un cliente regolare può dichiarare all'agenzia il suo interesse per uno o più clienti dell'altro sesso.

Gli incontri: L'agenzia programma incontri tra clienti regolari. Per ogni incontro interessano i due partecipanti, la data, il locale in cui si è svolto, e l'eventuale esito (positivo o negativo). Due clienti possono avere anche più incontri (in date diverse).

I locali: Ristoranti convenzionati con l'agenzia, di cui interessa il nome, il numero di tavoli, il prezzo tipico e le carte di credito accettate.

Esercizio 2 (punti 4) Ricavare lo schema relazionale corrispondente allo schema ER nell'esercizio precedente, cercando di minimizzare la presenza di valori nulli.

Esercizio 3 (punti 3) Scrivere le istruzioni SQL di creazione di due tabella risultanti dell'esercizio precedente. Si scelgano due tabelle legate tra loro da un vincolo di integrità referenziale. Si scrivano inoltre due istruzioni SQL di inserimento (una per ciascuna tabella) che rispettino il vincolo di integrità referenziale.

Esercizio 4 (punti 3) Esprimere in SQL la seguente interrogazione: "Trovare per tutti gli uomini `piacevoli` il numero di interessi."

Esercizio 5 (punti 4) Esprimere in SQL la seguente interrogazione: "Trovare l'importo medio speso da `Mario Rossi` negli incontri con donne di più di 40 anni, assumendo che paghi sempre lui e che spenda sempre il prezzo tipico del ristorante".

Esercizio 6 (punti 4) Considerare le relazioni $R_1(\underline{A}, B, C, D)$ e $R_2(\underline{E}, F)$ aventi rispettivamente cardinalità c_1 e c_2. Assumere che sia definito un vincolo di integrità referenziale fra l'attributo D di $R1$ e la chiave E di $R2$. Indicare la cardinalità minima e massima di ciascuno dei seguenti join:

1. $R1 \bowtie_{A=E} R2$ ....
2. $R1 \bowtie_{C=E} R2$ ....
3. $R1 \bowtie_{A=F} R2$ ....
4. $R1 \bowtie_{B=F} R2$ ....

Compito del 20 marzo 2002 (soluzione a pagina 105)

Esercizio 1 (punti 5) Considerare la seguente relazione, che contiene informazioni relative ad alcuni giocatori di calcio.

Cod	Cognome	Nome	CodRuolo	Ruolo	CodNaz	Nazione	DataNascita	Presenze
342	Rossi	Mario	A	Attaccante	I	Italia	11/02/1976	143
342	Rossi	Mario	C	Centrocampista	I	Italia	11/02/1976	143
522	Rossi	Luca	A	Attaccante	I	Italia	11/02/1976	45
213	Bruni	Piero	P	Portiere	I	Italia	20/01/1974	143
425	Santos	Joao	D	Difensore	BR	Brasile	21/03/1979	65
425	Santos	Joao	C	Centrocampista	BR	Brasile	21/03/1979	65

Individuare la chiave (o le chiavi) della relazione e le dipendenze funzionali definite su di essa (ignorando quelle che si ritiene siano *occasionali*) e spiegare perché essa non soddisfa la BCNF.

Decomporla in BCNF nel modo che si ritiene più opportuno.

Esercizio 2 (punti 2) Per lo schema risultante dall'esercizio 1, scrivere le istruzioni SQL per creare 2 tabelle a scelta, che abbiano almeno un vincolo referenziale tra loro.

Esercizio 3 (punti 4) Definire uno schema ER da cui sia ragionevole derivare lo schema relazionale risultante dall'esercizio 1.

Esercizio 4 (punti 3) Con riferimento allo schema relazionale risultante dall'esercizio 1, formulare la seguente interrogazione in **algebra relazionale**: "trovare nome e cognome dei giocatori che hanno due (o più) ruoli".

Esercizio 5 (punti 3) Con riferimento allo schema relazionale risultante dall'esercizio 1, formulare la seguente interrogazione in **SQL**: "trovare, per ciascun ruolo, il numero di giocatori nati dal 1979 in poi per i quali è indicato quel ruolo.".

Esercizio 6 (punti 5) Con riferimento allo schema relazionale risultante dall'esercizio 1, formulare la seguente interrogazione in **SQL**: "trovare il nome delle nazioni che forniscono solamente giocatori che hanno tutti lo stesso unico ruolo".

Esercizio 7 (punti 2) Si modifichi lo schema ER risultante dall'esercizio 3 considerando anche gli infortuni dei giocatori. Un infortunio è caratterizzato dal tipo (rottura menisco, contrazione muscolare, ...), la durata media in mesi per quel tipo di infortunio, il costo medio dell'operazione per quel tipo, la data di inizio, la data di fine e il costo effettivo dell'operazione.

Compito dell'8 aprile 2002 (soluzione a pagina 108)

Si vuole progettare una base di dati per gestire un torneo di scacchi. Le informazioni di interesse riguardano:

I giocatori: di cui interessano il nome, il cognome, il titolo (grande maestro, maestro internazionale, ...), e il numero di tornei vinti.

Le partite: che si dividono in *finite*, *in corso* e *in programma*. Delle partite finite interessano i giocatori (bianco e nero), e il risultato finale (vittoria di uno dei due o patta). Delle partite in programma interessano i giocatori senza informazione su chi giocherà con i pezzi bianchi e chi con i neri. Delle partite in corso interessa l'esatta posizione di tutti i pezzi sulla scacchiera e a chi spetta il turno. A questo scopo le caselle della scacchiera sono individuate da una lettera (da `a` ad h) ed un numero (da `1` ad 8).

Le aperture: A ciascuna partita finita ed ad alcune di quelle in corso è associata un'*apertura*. Dell'apertura interessa il nome (ad es. difesa siciliana, partita spagnola, gambetto di re), l'eventuale variante (ad es. variante di cambio, variante del dragone), e l'insieme dei partecipanti al torneo che sono soliti utilizzare tale apertura.

Esercizio 1 (punti 7) Si esegua la progettazione concettuale dei dati per un'applicazione che gestisca le informazioni suddette.

Esercizio 2 (punti 5) Si esegua la progettazione logica dei dati per un'applicazione che gestisca le informazioni dell'esercizio precedente.

Sia dato il seguente schema di una base di dati.

- Editori(Codice, Nome, Città)
- Autori(Codice, Cognome, Nome)
- Libri(Codice, Titolo, Genere, Editore, Prezzo)
- HaScritto(Libro, Autore, ProgressivoAutore)
- Librerie(Codice,Nome, Sede, NumeroDipendenti)
- Scorte(Libro, Libreria, Copie)

L'attributo ProgressivoAutore vale 1 per il primo autore del libro, 2 per il secondo autore e così via.

Esercizio 3 (punti 4) Con riferimento allo schema relazionale precedente, formulare la seguente interrogazione in **SQL**: "Trovare le copie disponibili, i titoli e il cognome del primo autore per ogni libro nella libreria `NonSoloLibri`".

Esercizio 4 (punti 4) Con riferimento allo schema relazionale precedente, formulare la seguente interrogazione in **SQL**: "Trovare titolo e codice dell'editore per ogni libro il cui prezzo sia maggiore di tutti i libri di `Fantascienza`".

Esercizio 5 (punti 4) Data la relazione $R(A, B, C, D, E)$ e le dipendenze funzionali $A \to B$, $BC \to D$ e $DE \to A$, determinare le chiavi di R a specificare se R è in 3NF o in BCNF, motivando la risposta.

Compito del 20 giugno 2002 **(soluzione a pagina 110)**

Si deve progettare la base di dati di un insieme di stabilimenti balneari che offrono servizi per l'attuale stagione estiva. Per la base di dati in questione sono di interesse:

Gli stabilimenti balneari, con codice identificativo, numero di cabine disponibili, clienti abbonati, città e regione in cui sono ubicati;

I clienti abbonati agli stabilimenti balneari, con codice fiscale e nome. Si noti che uno stesso cliente può essere abbonato a diversi stabilimenti balneari, e che ogni cliente appartiene ad uno ed uno solo dei seguenti tipi:

- Persona (di questi clienti interessa l'età e la città di residenza),

- Azienda (di questi clienti interessa il numero di dipendenti e le imbarcazioni di proprietà).

Le imbarcazioni, con codice, anno di immatricolazione, città di immatricolazione, e proprietario (ogni imbarcazione ha come proprietario una ed una sola azienda).

Esercizio 1 (punti 7) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare valori nulli nella base di dati.

Esercizio 3 (punti 3) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **algebra relazionale**: "Fornire il codice di tutti gli stabilimenti balneari con almeno 50 cabine.".

Esercizio 4 (punti 3) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Fornire il numero di immatricolazioni di imbarcazioni immatricolate dopo il 1999 che sono proprietà di aziende con più di 100 dipendenti."

Esercizio 5 (punti 4) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Fornire il codice fiscale di tutti i clienti abbonati ad uno stabilimento balneare che abbia almeno un cliente abbonato che è proprietario di una imbarcazione."

Esercizio 6 (punti 4) Descrivere brevemente le finalità della normalizzazione e il ruolo che essa può avere nella progettazione di basi di dati.

Compito dell'11 luglio 2002 **(soluzione a pagina 111)**

Si deve progettare la base di dati di un insieme di gestori telefonici che offrono servizi di telecomunicazioni. Per la base di dati in questione sono di interesse:

I gestori con codice identificativo, nome, possibili tariffe offerte ai clienti, ed i clienti che hanno un contratto con i gestori, con la relativa tariffa stabilita per il contratto. Ogni gestore appartiene ad uno ed uno solo dei seguenti tipi:

- gestore di telefonia fissa (di questi gestori interessa il numero di dipendenti, e la città in cui ha sede la direzione, con la relativa regione),
- gestore di telefonia mobile (di questi gestori interessa l'anno di inizio delle attività).

Le tariffe offerte dai gestori, con codice interno del gestore, data in cui è stata attivata, prezzo base della telefonata per minuto prevista da quella tariffa (ad esempio la tariffa con codice A1 del gestore HappyTel è stata attivata il 20-01-2002, e prevede 0.10 € al minuto come costo di ogni telefonata).

I clienti dei gestori, con codice fiscale, nome, cognome, città e regione di residenza. Si noti che una stessa persona può essere cliente di più gestori telefonici, e può avere anche più contratti con lo stesso gestore, purché con tariffe diverse.

Esercizio 1 (punti 7) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** per la creazione di due delle tabelle dello schema prodotto per l'esercizio 2, scegliendo due tabelle legate tra loro da un vincolo di riferimento.

Esercizio 4 (punti 3) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **algebra relazionale**: "Calcolare le città e le relative regioni in cui sia ubicato almeno un gestore con 500 o più dipendenti.".

Esercizio 5 (punti 4) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Calcolare i gestori che offrono almeno una tariffa che non è utilizzata in alcun contratto che quel gestore ha con i clienti. "

Esercizio 6 (punti 4) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Calcolare per ogni cliente la tariffa più nuova che utilizza, cioè quella che è attiva da meno tempo. "

Compito del 9 settembre 2002 **(soluzione a pagina 113)**

Si deve progettare la base di dati della *Camera dei Deputati* di una certa nazione. Di ciascun deputato interessa il nome, il partito politico a cui appartiene, il collegio in cui è stato eletto e la regione di appartenenza del collegio. Per ciascuna regione interessa il suo nome (ad es. Toscana, Catalogna, Baviera) e la parte della nazione in cui è locata (ad es. Nord, Centro, Sud, Isole, Colonie). La base di dati inoltre memorizza ogni progetto di legge con il codice (un valore intero progressivo), il nome e i deputati proponenti. Se il progetto di legge è stato votato, si memorizza anche la data in cui è stato votato e l'esito della votazione (Sì oppure No). La base dati tiene inoltre traccia di come ogni deputato ha votato su ciascun progetto di legge (Sì, No, Astenuto).

Esercizio 1 (punti 7) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** per la creazione di due delle tabelle dello schema prodotto per l'esercizio 2, scegliendo due tabelle legate tra loro da un vincolo di riferimento.

Esercizio 4 (punti 3) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le regioni che non hanno deputati del partito PXP".

Esercizio 5 (punti 4) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Trovare la parte della nazione in cui è presente la regione che conta il maggior numero di deputati che hanno votato Sì alla legge PizzaPerTutti".

Esercizio 6 (punti 4) Sulla base dello schema relazionale prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Trovare i *franchi tiratori recidivi*, cioè i deputati che hanno votato No ad almeno due leggi che hanno proposto loro stessi."

Compito del 25 settembre 2002 **(soluzione a pagina 115)**

Si deve progettare la base di dati del torneo sociale di doppio del circolo *Gli amici del tennis*. Del torneo interessano i giocatori (con nome, cognome e numero di tessera del circolo), le coppie partecipanti, le partite disputate e quelle in calendario. Di ciascuna partita disputata interessano le coppie che giocano, la coppia vincitrice, il punteggio di ciascun set e l'arbitro (che è un socio che non partecipa al torneo). Nel caso la partita finisca per infortunio interessa sapere chi si è infortunato e il tipo di infortunio.

Ciascun giocatore può essere sostituito in al massimo una partita del torneo. In tal caso interessa sapere chi è il sostituto, che deve essere un socio che non gioca il torneo, e in quale partita ha giocato.

Esercizio 1 (punti 7) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema di base di dati:

PERSONA(CF,Nome, Cognome, AnnoNascita, Sesso)
CONIUGIO(Moglie,Marito,AnnoMatrimonio,LuogoMatrimonio)

con i vincoli:

CONIUGIO(Moglie) $\subseteq$ PERSONA(CF)
CONIUGIO(Marito) $\subseteq$ PERSONA(CF)

Esercizio 3 (punti 3) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome degli uomini più giovani della loro moglie".

Esercizio 4 (punti 3) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che si sono sposate entro i 20 anni di età".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le donne che si sono sposate più volte, fornendo nome, cognome, e cognome dell'ultimo marito"

Esercizio 6 (punti 3) Si consideri il seguente schema di tabella $R(\underline{A, B}, C, D, E)$. Si forniscano una istanza di R ed una decomposizione dei suoi attributi tali che R si decomponga *con perdita* su tali attributi.

Compito del 16 dicembre 2002 **(soluzione a pagina 118)**

Si vuole progettare una base di dati per la gestione di un insieme di partite di un *gioco di ruolo*. Il gioco è composto da un insieme di giocatori, un insieme di caselle su cui transitano i giocatori, ed un insieme di oggetti che i giocatori possono raccogliere dalle caselle che li forniscono. La possibilità di muoversi da una casella all'altra è denotata da una relazione di *adiacenza* tra caselle che viene definita esplicitamente per ciascuna partita (e non fissata a priori come in una scacchiera).

Ogni giocatore è caratterizzato da un nome, un coefficiente di forza ed uno di energia, la casella in cui si trova e gli oggetti che possiede. Ogni oggetto ha un nome ed un valore. Ogni casella ha un nome e da un eventuale oggetto in essa presente.

I giocatori, le caselle e gli oggetti utilizzati possono variare da partita a partita (così come la relazione di adiacenza), ma le loro caratteristiche (forza, valore, ...) sono le stesse in tutte le partite.

Per ogni partita (identificata da un codice numerico progressivo) interessa anche l'ordine di gioco dei giocatori e il giocatore correntemente di turno.

Esercizio 1 (punti 7) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema di base di dati:

PERSONA(CF, Nome, Cognome, DataNascita, LuogoNascita, Sesso)
CONIUGIO(Moglie,Marito,DataMatrimonio,LuogoMatrimonio)

con i vincoli:

CONIUGIO(Moglie) $\subseteq$ PERSONA(CF)
CONIUGIO(Marito) $\subseteq$ PERSONA(CF)

Esercizio 3 (punti 3) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome degli uomini non sposati".

Esercizio 4 (punti 3) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che si sono sposate nel luogo in cui sono nate".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli uomini che si sono sposati più volte, fornendo nome e cognome della prima moglie"

Esercizio 6 (punti 3) Si consideri il seguente schema di tabella $R(\underline{A, B}, C, D, E)$. Si forniscano una istanza di R ed una decomposizione dei suoi attributi tali che R si decomponga *senza perdita* su tali attributi.

Compito del 20 marzo 2003 **(soluzione a pagina 120)**

Si vuole progettare la base di dati di un'applicazione relativa ai finanziamenti ottenuti dai comuni italiani. Di ogni comune interessa il codice (identificativo), il nome, il numero di abitanti, la regione a cui appartiene, ed i finanziamenti ricevuti dalle varie istituzioni. In particolare, per ogni finanziamento, interessa la quota di soldi ricevuta, l'anno in cui il finanziamento è stato erogato, e l'istituzione che ha erogato il finanziamento stesso. Si noti che un comune, in uno stesso anno, può ricevere al massimo un finanziamento da ogni istituzione. Di ogni comune interessa anche sapere chi sono stati i vari sindaci (con codice fiscale, nome, cognome ed età) eletti nei vari anni in cui si sono tenute le elezioni. Di ogni istituzione che può erogare fondi ai comuni, interessa il codice (identificativo) ed il nome. Esistono due e solo due tipi di istituzioni: regioni e altri. Di ogni regione interessa il nome (identificativo), il comune capoluogo della regione, le regioni confinanti, e la superficie. Di ogni istituzione che non sia una regione interessa l'anno di fondazione, ed il presidente attuale (con codice fiscale, nome, cognome ed età).

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati:

PERSONA(CF, Nome, Cognome, DataNascita, LuogoNascita, Sesso)
CONIUGIO(Moglie,Marito,DataMatrimonio,LuogoMatrimonio)

con i vincoli:

CONIUGIO(Moglie) $\subseteq$ PERSONA(CF), CONIUGIO(Marito) $\subseteq$ PERSONA(CF)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome delle donne che si sono sposate con almeno due uomini diversi".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che sono nate a Udine e si sono sposate a Roma prima del 2000".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il codice fiscale degli uomini che si sono sposati due volte *consecutive* con la stessa moglie".

Esercizio 6 (punti 3) Si consideri la tabella relazionale R sotto a sinistra. Si calcoli il risultato dell'interrogazione SQL a destra.

R

A	B	C	D	E
2	1	3	4	5
2	1	2	5	7
2	1	6	6	8
2	1	7	8	0
2	1	3	3	2
3	1	3	4	5
3	1	2	5	7
3	1	7	8	1
7	1	7	8	2
8	1	7	7	3
8	1	7	0	8
9	1	7	7	0
9	1	8	0	8

```
select a, sum(e)
from r
where c > 2
group by a, b
having count (distinct c) > 1;
```

Compito del 3 aprile 2003 (soluzione a pagina 122)

Si vuole progettare la base di dati di un'applicazione relativa alla gestione giornaliera di un cinema multisala. Dei film interessa in titolo (identificativo) e il regista. In un giorno viene effettuata una sola proiezione per ciascuna sala, ed uno stesso film può essere anche proiettato in più sale[1]. Delle sale interessa il codice identificativo e il numero totale di posti. Le sale si dividono in grandi e piccoli. Nelle sale grandi ogni posto è identificato dal settore (centrale, laterale, o galleria), da un numero che indica il numero del posto e dalla fila. Nelle sale piccole non si ha la divisione in settori e i posti non hanno numero e fila. Il prezzo del biglietto dipende dalla sala e, nelle sale grandi, anche dal settore. Ogni biglietto venduto per una sala viene conteggiato, e per le sale grandi viene anche assegnato un posto di cui quindi viene registrata l'occupazione.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati:

FILM(Titolo, Regista, Anno, Genere)
HARECITATOIN(Attore,Film)

con i vincoli:

HARECITATOIN(Film) $\subseteq$ FILM(Titolo)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare gli attori che hanno recitato solo nei film diretti da loro stessi".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli attori che hanno recitato in tutti i film diretti da `Woody Allen`".

Esercizio 5 (punti 6) Esprimere la seguente interrogazione in **SQL**: "Trovare le coppie di attori che hanno recitato esattamente negli stessi film".

Esercizio 6 (punti 3) Si considerino le tabelle relazionali R1 e R2 sotto a sinistra, dove le caselle bianche indicano valori nulli. Si calcoli il risultato dell'interrogazione SQL a destra.

R1

A	B	C	D
x	6	6	4
y	2	5	7
z	9	8	1
t	6	7	6
w	4	7	8

R2

A	C	D
4	5	6
7	3	
8	6	7
2		5

```
select A, B
from R1
where D <=all (select C
               from R2
               where B < D)
```

[1] Nel testo originale erano considerate più proiezioni giornaliere, queste sono state eliminate perché introducevano alcune ambiguità nel dominio

Compito del 16 giugno 2003 (soluzione a pagina 124)

Si vuole progettare la base di dati di un'applicazione relativa ad un programma di concerti, secondo le seguenti specifiche.

- Ogni concerto ha un codice, un titolo e una descrizione, ed è composto da una sequenza (ordinata) di pezzi musicali.
- Ogni pezzo ha un codice, un titolo e un autore (con codice e nome); uno stesso pezzo può essere rappresentato in diversi concerti *o anche più volte in uno stesso concerto.*
- Ogni concerto è eseguito da un gruppo; ogni gruppo ha un nome, e un insieme di componenti.
- Ogni componente ha una matricola (univoca nell'ambito della base di dati), nome e cognome, può partecipare a più gruppi, e suona uno o più strumenti, gli stessi in ciascuno dei gruppi.
- Ogni concerto è tenuto più volte, in date diverse, ma sempre nella stessa sala.
- Ogni sala ha un codice, un nome e una capienza.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati:

GUIDATORI(Codice, Nome, Affidabilità, Età)
AUTOMOBILI(Codice, Nome, Colore)
PRENOTAZIONI(Guidatore,Automobile, Data)

con i vincoli:

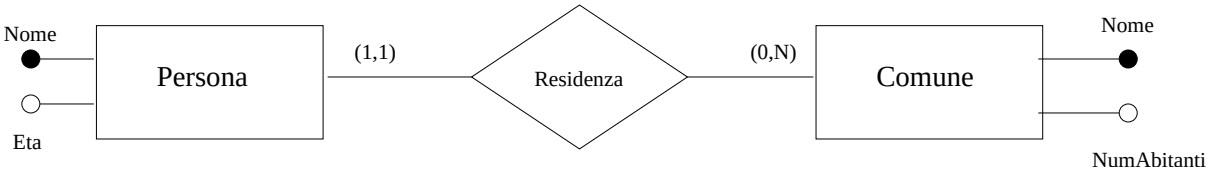
PRENOTAZIONI(Guidatore) $\subseteq$ GUIDATORI(Codice)
PRENOTAZIONI(Automobile) $\subseteq$ AUTOMOBILI(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il codice delle automobile che sono state prenotate solo da guidatori sotto i 21 anni".

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il livello di affidabilità minimo tale che nessun guidatore sotto i 21 anni possiede tale livello o uno superiore".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il numero totale di prenotazioni di automobili rosse o di colore ignoto da parte di guidatori tra i 20 e i 40 anni che non abbiano mai prenotato un'automobile verde".

Esercizio 6 (punti 4) Si consideri le schema ER seguente nel quale l'attributo NumAbitanti di Comune è ottenuto come numero dei legami con Persona.



Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità delle due entità sono 20.000 per Persona e 200 per Comune e che le operazioni più importanti sono:

Op1 inserimento di una nuova persona, con frequenza $f_1 = 100$/ora

Op2 lettura del numero di abitanti, con frequenza $f_2 = 10$/ora

e assumendo che una scrittura costi mediamente come tre letture.

Compito del 14 luglio 2003 (soluzione a pagina 126)

Si vuole progettare la base di dati di un'applicazione relativa all'archivio di un amministratore di condomini, secondo le seguenti specifiche.

- ogni condominio ha un nome (che lo identifica) e un indirizzo e comprende una o più scale, ognuna delle quali comprende un insieme di appartamenti;
- ad ogni scala sono associati:
  - un codice (es: scala `A`) che la identifica assieme al nome del condominio;
  - un valore, detto quota della scala, che rappresenta, in millesimi, la frazione delle spese del condominio che sono di competenza degli appartamenti compresi nella scala;
- ogni appartamento è identificato, nel rispettivo condominio, dalla scala e da un numero (l'interno). Ad ogni appartamento è associata una quota, espressa in millesimi, che indica la frazione delle spese (della scala) che sono di competenza dell'appartamento;
- ogni appartamento ha un proprietario per il quale sono di interesse il nome, il cognome, il codice fiscale e l'indirizzo al quale deve essere inviata la corrispondenza relativa all'appartamento;
- per la parte contabile, è necessario tenere traccia (includendo la data) delle spese sostenute dal condominio e dei pagamenti effettuati dai proprietari:
  - ogni spesa è associata ad un intero condominio, oppure ad una scala oppure ad un singolo appartamento;
  - ogni pagamento è relativo ad uno e un solo appartamento.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Esercizio 3 (punti 4) A partire dallo schema prodotto per l'esercizio 2, esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome dei proprietari di almeno due appartamenti in scale diverse di uno stesso condominio".

Esercizio 4 (punti 5) A partire dallo schema prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Calcolare le spese totali effettuate per ciascuna scala, non tenendo conto delle spese generali del condominio e dei singoli appartamenti"

Esercizio 5 (punti 5) A partire dallo schema prodotto per l'esercizio 2, esprimere la seguente interrogazione in **SQL**: "Trovare l'importo totale dei pagamenti per ciascuna scala del condominio `Le terrazze` che tenga conto sia delle spese relative alla scala che della quota relativa alle spese di condominio.".

Esercizio 6 (punti 4) A partire dallo schema prodotto per l'esercizio 2, realizzare la seguente modifica in **SQL**: "Cancellare di tutti i pagamenti effettuati nel 2001 relativi agli appartamenti di proprietà della persona avente codice fiscale `XXXXXX76A21F555A`".

Compito del 2 settembre 2003 (soluzione a pagina 127)

Si vuole realizzare una base di dati per la gestione di prenotazioni ferroviarie. Ogni treno è identificato da un tipo (IC, EC, reg, ...) e da un numero, ed è caratterizzato da una stazione di partenza, una stazione di arrivo, da un orario di partenza e da un orario di arrivo. Inoltre alcuni treni possono non viaggiare tutti i giorni (feriali, festivi, solo mercoledì e giovedì,...).

Ogni treno percorre un certo numero di tratte. Ogni tratta è caratterizzata dalla stazione di partenza, la stazione di arrivo, ha un chilometraggio ed una tariffa che dipende dalla classe e dai tipi di treno che la percorrono.

Ogni treno è costituito da più carrozze, contrassegnate da un numero. Ogni carrozza è di prima o di seconda classe. In ogni carrozza è presente un certo numero di compartimenti corrispondenti ai posti in un certo intervallo (ad es., nel primo compartimento ci sono i posti da 1 a 6 della carrozza, nel secondo quelli da 7 a 12, e così via).

I clienti che effettuano le prenotazioni sono caratterizzati da nome, cognome e numero di telefono. Ogni prenotazione è effettuata da un cliente per un posto su un certo treno, per una certa tratta, in una certa data, per una certa classe. Un cliente più prenotare più tratte e più posti e per ogni prenotazione si vogliono memorizzare il posto assegnato al cliente e l'importo dovuto da questo.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati relativo ad un campionato di calcio.

SQUADRA(Nome, Città, Sponsor,ColoriSociali,Allenatore)
GIOCATORE(NTessera, Squadra, Numero, Nome, Cognome, AnnoN, Ruolo)
PARTITA(IdPartita, Giornata, SqCasa, SqTrasf, GoalCasa, GoalTrasf)
GOL(IdPartita, Minuto, Marcatore, Autogol)

con i vincoli:

GIOCATORE(Squadra) $\subseteq$ SQUADRA(Nome)
PARTITA(SqCasa) $\subseteq$ SQUADRA(Nome)
PARTITA()SqTrasf $\subseteq$ SQUADRA(Nome)
GOL(IdPartita) $\subseteq$ PARTITA(IdPartita)
GOL(Giocatore) $\subseteq$ GIOCATORE(NTessera)

Nella relazione GOL l'attributo Marcatore memorizza il numero di tessera del giocatore che ha segnato il goal, mentre l'attributo AutoGol è un valore booleano che vale `True` se il goal è stato un autogol.

Esercizio 3 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le squadre che hanno subito almeno un autogol".

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i portieri delle squadre che hanno subito più gol in una sola partita"

Esercizio 5 (punti 6) Esprimere la seguente interrogazione in **SQL** utilizzando, se lo si ritiene utile, la definizione di viste per memorizzare risultati intermedi.: "Per ogni squadra determinare il capocannoniere (cioè il giocatore che ha segnato più gol)".

Compito del 16 settembre 2003 **(soluzione a pagina 130)**

L'applicazione da progettare riguarda le vaccinazioni. Ogni vaccino è identificato da un codice, ed ha associato un numero intero che rappresenta il livello di rischio della sua somministrazione. Per ogni vaccino è di interesse conoscere la sua importanza. Esistono infatti due e solo due tipi di vaccini: i vaccini semplici ed i vaccini composti. Ogni vaccino semplice viene somministrato per prevenire una ed una sola malattia. Di ogni vaccino semplice interessa l'anno in cui è stato introdotto nel sistema sanitario. Ogni vaccino composto viene somministrato per prevenire due o più malattie. Di ogni vaccino composto interessa il ticket che occorre pagare per la sua somministrazione. Di ogni malattia interessa il codice identificativo ed il tipo (contagiosa, ereditaria, ecc.). Si noti che non per tutte le malattie esiste un corrispondente vaccino semplice che la previene, ma se esiste, esso è unico. Al contrario, per una malattia può esistere un numero qualunque di vaccini composti che la prevengono. Una vaccinazione rappresenta la somministrazione di un vaccino ad una persona (una persona viene sottoposta al massimo ad una somministrazione di ogni vaccino). Di ogni vaccinazione interessa la data in cui è avvenuta. Di ogni persona interessa il nome, il cognome, e la data di nascita. Infine, di ogni malattia interessano le persone che l'hanno contratta.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** necessarie per creare due tabelle a scelta dell'applicazione legate tra loro da un vincolo di riferimento.

Si consideri il seguente schema di base di dati relativo ad un campionato di calcio.

SQUADRA(Nome, Città, Sponsor, ColoriSociali, Allenatore)
GIOCATORE(NTessera, Squadra, Numero, Nome, Cognome, DataNascita, Ruolo)
PARTITA(IdPartita, Giornata, SqCasa, SqTrasf, GoalCasa, GoalTrasf)
GOL(IdPartita, Minuto, Marcatore, Autogol)

con i vincoli:

GIOCATORE(Squadra) $\subseteq$ SQUADRA(Nome)
PARTITA(SqCasa) $\subseteq$ SQUADRA(Nome)
PARTITA(SqTrasf) $\subseteq$ SQUADRA(Nome)
GOL(IdPartita) $\subseteq$ PARTITA(IdPartita)
GOL(Marcatore) $\subseteq$ GIOCATORE(NTessera)

Nella relazione GOL l'attributo Marcatore memorizza il numero di tessera del giocatore che ha segnato il goal, mentre l'attributo AutoGol è un valore booleano che vale `True` se il goal è stato un autogol.

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il giocatore (nome e cognome) che ha segnato il gol (esclusi gli autogol) più avanti nella partita (cioè nel minuto più grande) di tutto il campionato".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i ruoli per i quali non esiste alcun giocatore che li ricopra in una squadra di `Milano`".

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni squadra il giocatore più giovane (con nome e cognome)"

Compito del 15 dicembre 2003

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle seguenti specifiche. Un'esposizione di arte contemporanea ha necessità di progettare una base di dati per memorizzare e gestire le informazioni sulle opere esposte, sugli autori e sui propri dipendenti. Tali informazioni riguardano:

- Le opere, ciascuna delle quali ha un numero di matricola, un anno di realizzazione, un autore, ed è esposta in una certa sala. Le opere sono quadri o sculture, per i primi si vogliono memorizzare altezza e larghezza e tecnica (olio, tempera, ...), per i secondi altezza, peso e materiale.
- Le sale, di cui si vogliono memorizzare il numero, la larghezza e la lunghezza ed il piano a cui si trovano. Dei piani si vuole conoscere il numero d ordine, la superficie complessiva e l'altezza dei soffitti.
- Gli autori di cui si vuole memorizzare codice fiscale, cognome, età, e uno o più numeri di telefono.
- I guardiani di cui si vogliono memorizzare gli stessi dati personali degli autori, nonché ora di inizio e di fine del turno. Inoltre ciascun guardiano è responsabile della sorveglianza di una o più sale.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relazionale:

FORNITORE(CodiceF, Nome, Città)
CLIENTE(CodiceC, Nome, Città)
PRODOTTO(CodiceP, Nome)
ACQUISTO(CodiceA, CodiceC, CodiceP, CodiceF, Importo, Anno)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi dei clienti di Milano che hanno acquistato prodotti dal fornitore `Rossi`".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi di tutti i clienti che risiedono in città per le quali non è presente alcun fornitore".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il massimo numero di acquisti effettuati da uno stesso cliente presso un unico fornitore durante il 2000, considerando solo il caso di gruppi di acquisti (cliente-fornitore) di importo complessivo superiore a 100.000".

Esercizio 6 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome della città i cui fornitori hanno il fatturato complessivo di importo massimo durante il 2002"

Compito del 19 marzo 2004

Esercizio 1 (punti 10) Si vuole progettare una base di dati per gestire le informazioni presenti su un libro di ricette di cucina. Ogni ricetta ha un codice, un titolo, un grado di difficoltà e un tempo di preparazione. Gli ingredienti hanno un codice, un nome, una quantità ed una unità di misura (litri, grammi, unità, ...).

Le ricette sono suddivise in una sequenza ordinata di passi ciascuno con la propria durata. Ciascun passo è rappresentato dalla lavorazione (cottura, impasto, assemblaggio, ...), dall'insieme dei componenti (ingredienti di base o semilavorati di passi precedenti) e dal nuovo semilavorato ottenuto come risultato. Alcuni passi possono anche avere una nota, che consiste in un riferimento ad un altro passo di un'altra ricetta, corredata da una descrizione della similitudine tra i passi.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relazionale:

SCRITTORE(Nome, AnnoNascita, AnnoMorte*)
LIBRO(Titolo, Autore, Genere, CasaEditrice, Anno, Copie)

con il vincolo:

LIBRO(Autore) $\subseteq$ SCRITTORE(Nome)

in cui l'attributo AnnoMorte ha valore nullo per gli scrittori viventi.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le case editrici che hanno pubblicato solo libri di fantascienza".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli scrittori viventi che hanno scritto almeno 5 libri di successo prima dei 30 anni (un libro è di successo se ha venduto almeno 10000 copie)".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli autori che non hanno mai scritto un libro con lo stesso titolo di un altro libro già pubblicato da un autore più giovane di loro".

Esercizio 6 (punti 4) Fornire il risultato della interrogazione qui sotto a destra in base allo schema e l'istanza generati dai comandi a sinistra.

```
create table R1
(
  a integer,
  b integer,
  c char(2),
  primary key(b,c)
);
insert into R1 values(10,2,'cc');
insert into R1 values(10,3,'aa');
insert into R1 values(13,2,'ab');
insert into R1 values(14,3,'bb');
```

```
select b, c
from R1 S
where a >=all (select a
               from R1
               where b < S.b + 2
                 and c > S.c);
```

Compito del 31 marzo 2004

Si vuole progettare una base di dati per gestire le informazioni riguardanti un *circolo scolastico*. Il circolo gestisce un insieme di scuole di diverso tipo: materne, elementari e medie. Di ciascuna scuola interessa il nome, l'indirizzo e il numero di telefono; per le scuole medie soltanto interessa anche l'anno della fondazione. Nelle scuole elementari e medie ciascuna classe ha un livello (prima, seconda, ...) ed una sezione (A, B, ...), ed i ragazzi iscritti in una classe sono tutti dello stesso livello. Nelle scuole materne invece le classi hanno solo un nome ed esistono sia classi omogenee che classi eterogenee, cioè formate da bambini di livelli diversi. Degli alunni iscritti alle scuole interessa (oltre al livello e alla classe) il nome, il cognome, la data di nascita, la zona di residenza, l'indirizzo ed il numero di telefono di almeno uno dei due genitori. Ciascuna classe ha due o tre insegnanti, dei quali interessa il nome, il cognome, il codice fiscale e i vari titoli di studio.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli. In mancanza di dati quantitativi preferire le scelte che evitano la presenza di valori nulli.

Esercizio 3 (punti 2) Scrivere il comando **SQL** necessario per creare una vista a propria scelta, avendo cura di sceglierne una che a proprio giudizio risulta utile per semplificare una o più interrogazione che si reputano di interesse.

Si consideri il seguente schema di base di dati relazionale:

> CD(<u>Autore,Titolo</u>,Durata)
> Affitto(<u>AutoreCD,TitoloCD,Cliente</u>,DataPrestito,Restituito)
> Cliente(<u>Nome</u>,Città)

con i vincoli:

> Affitto(AutoreCD,TitoloCD) $\subseteq$ CD(Autore,Titolo)
> Affitto(Cliente) $\subseteq$ Cliente(Nome)

in cui l'attributo Restituito ha un valore booleano.

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare gli autori tali che un loro CD non è mai stato affittato a clienti di Udine".

Esercizio 5 (punti 4) Esprimere l'interrogazione dell'esercizio precedente in **SQL**.

Esercizio 6 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni CD di durata maggiore di 50' trovare il numero di prestiti a clienti di Udine o di Pordenone."

Esercizio 7 (punti 2) Disegnare lo schema entità-relazione da cui lo schema relazionale può essere stato derivato.

Compito del 28 giugno 2004

Esercizio 1 (punti 10) Si vuole progettare una base di dati per gestire il menù di un ristorante. Il menù comprende una lista di piatti e ciascun piatto è caratterizzato da un nome, un prezzo, e dai suoi ingredienti (con le relative quantità). Ciascun piatto inoltre appartiene ad una categorie (antipasti, primi, secondi, ...). Il menù comprende inoltre i *pasti completi*, che hanno un nome ed un prezzo e sono insiemi di piatti della lista. I pasti completi possono anche contenere delle scelte tra due o più piatti diversi. Ad esempio, il pasto del camionista (11 €) è composto da: risotto, costata di maiale o stracotto, frutta o tiramisù). I vini (bianchi o rossi) sono caratterizzati dal nome, il produttore, l'anno di produzione, l'indirizzo del produttore e il prezzo della bottiglia. Per ciascun piatto, infine, possono esistere uno o più vini consigliati per accompagnarlo.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema relazionale di basi di dati:

GIOCATORE(Nome,Nazionalità, DataNascita)
PARTITA(Codice, Vincitore, Perdente)
SET(Partita, Numero, Punti1, Punti2)

con i vincoli:

PARTITA(Vincitore) $\subseteq$ GIOCATORE(Nome)
PARTITA(Perdente) $\subseteq$ GIOCATORE(Nome)
SET(Partita) $\subseteq$ PARTITA(Codice)

che memorizza i risultati di un insieme di partite di tennis, con i punteggi di tutti i set giocati. L'attributo `Punti1` memorizza i punti del giocatore che ha vinto la partita (che non necessariamente ha vinto quel set) e `Punti2` quelli del giocatore che ha perso.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi dei giocatori che non hanno mai battuto un connazionale".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi dei giocatori che non hanno mai affrontato un avversario più giovane"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome del giocatore che ha vinto complessivamente più set"

Esercizio 6 (punti 4) Si consideri la seguente relazione R.

R

X	Y	Z
1	1	1
1	1	2
2	1	1
2	1	3

Elencare le dipendenze funzionali che sono verificate da R ed identificarne le chiavi.

Compito del 8 luglio 2004

Esercizio 1 (punti 9) Un commercialista vuole disporre di una base di dati per gestire le informazioni riportate nei moduli per il pagamento delle imposte dei propri clienti. Ogni modulo contiene:

- il codice fiscale e i dati del contribuente: nome, cognome, data di nascita, luogo di nascita (comune e provincia), domicilio fiscale (comune, provincia e indirizzo); nella base di dati sono presenti le informazioni relative a tutti i comuni d'Italia, con le relative province, per garantire la correttezza delle informazioni al riguardo (si supponga per semplicità che interessino solo contribuenti nati e residenti in Italia)
- una lista di *pagamenti elementari*, ognuno dei quali contiene
  - il codice del tributo (che deve appartenere ad un insieme noto alla base di dati, con codice e descrizione)
  - un anno di riferimento
  - un importo
- l'importo totale dei pagamenti, pari alla somma dei pagamenti elementari
- la data del versamento e il codice della banca presso cui il versamento è stato effettuato; la banca deve essere nota alla base di dati, con codice, nome e indirizzo.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema relazionale di basi di dati:

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

con i vincoli:

FARMACI(Produttore) $\subseteq$ PRODUTTORI(Codice)
FARMACI(PrincipioAttivo) $\subseteq$ SOSTANZE(ID)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare per i farmaci il cui principio attivo è nella categoria `Antibiotico`, il nome del farmaco e quello del suo produttore."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare, fra i farmaci con lo stesso principio attivo, quello con costo minore, mostrando il nome del farmaco, quello del produttore e quello della sostanza del suo principio attivo."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i farmaci *esclusivi*, cioè quelli per i quali non esiste un altro farmaco, di produttore diverso, con lo stesso principio attivo. Mostrare il nome del farmaco e quello del produttore."

Esercizio 6 (punti 5) Considerare le seguenti relazioni (tutte senza valori nulli)

- R1(A,B,C), con vincolo di integrità referenziale fra C e R2(D) e con cardinalità N1 = 100
- R2(D,E, F), con vincolo di integrità referenziale fra F e R3(G) e con cardinalità N2 = 200
- R3(G,H, I), con cardinalità N3 = 50

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

1. $\pi_{AB}(R1)$

2. $\pi_E(R2)$
3. $\pi_{BC}(R1)$
4. $\pi_G(R3)$
5. $R1 \bowtie_{A=D} R2$
6. $R1 \bowtie_{C=D} R2$
7. $R3 \bowtie_{I=A} R1$

Compito del 3 settembre 2004

Esercizio 1 (punti 8) Si richiede di progettare una base di dati per la gestione di una compagnia aerea. I dati di interesse sono i seguenti.

- Ogni aereo ha un codice ed un modello. I modelli hanno portata, peso e numero di posti.
- Gli aerei vengono sottoposti a diversi controlli. Ciascun controllo ha un codice, un nome ed un punteggio massimo. Su ciascun aereo possono essere effettuati più controlli diversi in un giorno. Per ciascun controllo effettuato viene memorizzato anche il giorno in cui è stato effettuato ed il punteggio ottenuto.
- I piloti sono caratterizzati dal CF, il nome, l'indirizzo e i numeri di telefono. I piloti sono abilitati a guidare uno o più modelli di aereo.
- I piloti devono passare un controllo medico annuale, e per ciascun pilota si deve memorizzare la data dell'ultimo controllo.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema relazionale di basi di dati:

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

con i vincoli:

FARMACI(Produttore) $\subseteq$ PRODUTTORI(Codice)
FARMACI(PrincipioAttivo) $\subseteq$ SOSTANZE(ID)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il nome di produttori italiani che producono almeno due farmaci il cui principio attivo è nella categoria `Antibiotico`."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascun produttore il numero di farmaci prodotti che non contengono `Insulina`."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i produttori *monopolisti*, cioè quelli per i quali esiste un farmaco che ha una sostanza che non è presente in alcun farmaco di un altro produttore. Mostrare il nome del produttore, quello del farmaco e quello della sostanza."

Esercizio 6 (punti 2) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale precedente.

Esercizio 7 (punti 4) Si consideri il seguente schema di relazione riguardante la base di dati di una facoltà universitaria.

STUDENTI(CF, matricola, nome, cognome, indirizzo, telefono, data_nascita, comune_nascita, provincia_nascita, regione_nascita, corso_laurea, anno_corso)

Si identifichino *tutte* le chiavi della relazione, facendo (e riportando) le assunzioni che si ritengono ragionevoli sulle possibili istanze. Si discutano anche assunzioni alternative e come l'insieme delle chiavi si modifica in accordo con le assunzioni fatte.

Si identifichino infine le dipendenze funzionali e si discuta l'eventualità di normalizzare la relazione.

Compito del 14 settembre 2004

Si consideri il seguente schema di base di dati:

FILM(Titolo, Regista, Anno, Genere)
HARECITATOIN(Attore, Film, Direttore)
ARTISTA(Nome,Nazionalità,Età,Sesso)

con i vincoli:

HARECITATOIN(Film, Direttore) $\subseteq$ FILM(Titolo,Regista)
HARECITATOIN(Attore) $\subseteq$ ARTISTA(Nome)
FILM(Regista) $\subseteq$ ARTISTA(Nome)

Esercizio 1 (punti 3) Scrivere i comandi `SQL` necessari per eliminare tutti i dati riguardanti `Film` precedenti al `2000`.

Esercizio 2 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i registi che hanno diretto almeno un film entro il `2002`, ma nessuno nel `2003`".

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli attori che hanno recitato solo nei film diretti da connazionali".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli artisti `Spagnoli` che non hanno partecipato in alcuna forma a qualche film".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le coppie di attori di sesso opposto che hanno recitato almeno due film insieme".

Esercizio 6 (punti 3) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale precedente.

Esercizio 7 (punti 4) Si modifichi lo schema ER dell'esercizio precedente aggiungendo un codice identificativo sia per i film che per gli attori, ma non per i registi che restano identificati dal nome.

Esercizio 8 (punti 4) Data la relazione $R(A, B, C, D, E)$ e le dipendenze funzionali $B \to C$, $CD \to E$ e $EA \to B$, determinare le chiavi di R a specificare se R è in 3NF o in BCNF, motivando la risposta.

Compito del 9 dicembre 2004

Si vuole progettare la basi di dati di un'applicazione relativa ad una catena di garage, descritta dalle seguenti specifiche:

- Di ogni garage interessa il codice identificativo, il proprietario (con codice fiscale, nome, cognome, e indirizzo e la città), l'indirizzo e la città. Ogni posto auto di un garage ha un numero progressivo unico nell'ambito del garage, ed appartiene ad una ed una sola categoria. Ogni categoria di posto auto è caratterizzata da un codice identificativo e dalle dimensioni (area e altezza).
- I garage affittano i posti alle automobili per tutto il giorno. Di ogni giorno interessa conoscere quale automobile ha occupato quale posto auto dei vari garage della catena. Il prezzo giornaliero di un posto auto fissato da un garage dipende della categoria del posto auto. Alcuni posti auto sono speciali, nel senso che sono riservati ad uffici convenzionati, ed il loro utilizzo prevede il beneficio di uno sconto. Di ogni posto convenzionato interessa l'ammontare dello sconto per esso praticato, a quale piano si trovi nel garage corrispondente, e l'ufficio con il quale è in convenzione. Ovviamente, i posti convenzionati con un ufficio possono essere utilizzati solo da automobili i cui proprietari lavorano in quell'ufficio.
- Delle automobili interessa la targa, l'anno di immatricolazione, la marca, il modello, e il proprietario.
- Di ogni proprietario di automobile interessano il codice fiscale, il nome, il cognome, l'indirizzo, la città di residenza, e l'eventuale ufficio in cui lavora.
- Di ogni ufficio interessa il nome (unico), l'indirizzo e la città in cui è situato.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema di base di dati relativo ad un campionato di calcio.

SQUADRA(Nome, Città, Sponsor,ColoriSociali,Allenatore)
GIOCATORE(NTessera, Squadra, Numero, Nome, Cognome, DataNascita, Ruolo)
PARTITA(IdPartita, Giornata, SqCasa, SqTrasf, GoalCasa, GoalTrasf)
GOL(IdPartita, Minuto, Marcatore, Autogol)

con i vincoli:

GIOCATORE(Squadra) $\subseteq$ SQUADRA(Nome)
PARTITA(SqCasa) $\subseteq$ SQUADRA(Nome)
PARTITA()SqTrasf $\subseteq$ SQUADRA(Nome)
GOL(IdPartita) $\subseteq$ PARTITA(IdPartita)
GOL(Giocatore) $\subseteq$ GIOCATORE(NTessera)

Nella relazione GOL l'attributo Marcatore memorizza il numero di tessera del giocatore che ha segnato il goal, mentre l'attributo AutoGol è un valore booleano che vale `True` se il goal è stato un autogol.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i giocatori che hanno fatto un autogol in una partita con una squadra della stessa città."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni squadra il numero di gol medio fatti nelle partite vinte giocate in casa".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni partita nome e cognome del giocatore che ha segnato per primo."

Esercizio 6 (punti 3) Si consideri la tabella relazionale R sotto a sinistra. Si calcoli il risultato dell'interrogazione SQL a destra.

R

A	B	C	D	E
6	2	3	4	5
6	2	2	5	7
6	2	7	8	0
6	3	3	4	5
6	2	6	6	8
7	3	2	5	7
7	3	7	8	7
7	4	7	8	2
7	5	7	7	3
7	5	3	3	2
7	5	7	2	8
7	6	8	0	8
7	6	7	7	0

```
select B, max(E)
from R
where C >= 3
group by A, B
having sum(D) > 7
```

Compito del 16 marzo 2005

Si vuole progettare la basi di dati di un'applicazione relativa ad un piccolo aeroporto da turismo, descritta dalle seguenti specifiche:

- Ogni aeroplano ha una matricola, è di un tipo e viene posto in un hangar specifico. Ogni tipo di aeroplano ha un codice, una capacità ed un peso. Di ogni hangar interessa il codice, la capacità, la zona in cui è ubicato, ed il tempo necessario per raggiungerla.
- Il proprietario di un aereo può essere una persona (con nome e codice fiscale) oppure una società (con nome, indirizzo e telefono).
- Dei tecnici dell'aeroporto interessa il nome, il codice fiscale, il numero del tesserino di lavoro, e (se noto) l'indirizzo di posta elettronica. Ciascun tecnico dell'aeroporto lavora su uno o più tipi di aereo, ed effettua interventi di manutenzione. Di ogni intervento, interessa (oltre all'aereo ed al tecnico coinvolti) la data in cui è stato effettuato, il tempo necessario ed il tipo di lavoro svolto.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** per la creazione di due delle tabelle dello schema prodotto per l'esercizio 2 legate tra loro da un vincolo di riferimento. Si scelgano le tabelle in modo da poter inserire anche il vincolo che il peso di un aereo non sia inferiore ai 300 chili.

Si consideri il seguente schema di base di dati.

VOLO(Numero, CittàPartenza, CittàArrivo, OraPartenza, OraArrivo, Aereo, Prezzo)
AEREO(Codice, Nome, Velocità)
ABILITAZIONE(Aereo, Pilota)
PILOTA(Matricola, Nome, Stipendio)

con i vincoli:

VOLO(Aereo) $\subseteq$ AEREO(Codice)
ABILITAZIONE(Aereo) $\subseteq$ AEREO(Codice)
ABILITAZIONE(Pilota) $\subseteq$ PILOTA(Matricola)

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi dei piloti che non sono abilitati per alcun aereo con velocità superiore a 800 Km/h"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e stipendio dei piloti che non sono abilitati per alcun aereo e che guadagnano più della media dei piloti abilitati per qualche aereo"

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare gli orari di partenza dei viaggi tra `Roma` e `Trieste` che arrivano entro le 21 con al massimo una sosta (cioè composti da uno o due voli)".

Esercizio 7 (punti 3) Si consideri una base di dati composta dalle seguenti relazioni

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

scrivere interrogazioni in **SQL** equivalenti alle seguenti espressioni dell'algebra relazionale:

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

Compito del 31 marzo 2005

Un pediatra vuole progettare la base di dati per un'applicazione per la gestione dei suoi pazienti. Di ogni bambino interessa il numero di tessera sanitaria, il nome, il cognome, l'indirizzo, la data di nascita ed i numeri di telefono dei genitori. Ciascuna visita viene memorizzata con la data e la durata. Inoltre, per ogni paziente interessano anche tutte le malattie che ha avuto, con la loro durata, ed il livello di febbre per ciascun giorno di malattia e la visita in cui la malattia è stata diagnosticata. Inoltre, in alcune visite il medico registra anche l'altezza e il peso correnti del bambino verificando se sono nella norma.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema di base di dati.

VOLO(Numero, CittàPartenza, CittàArrivo, OraPartenza, OraArrivo, Aereo, Prezzo)
AEREO(Codice, Nome, Velocità)
ABILITAZIONE(Aereo, Pilota)
PILOTA(Matricola, Nome, Stipendio)

con i vincoli:

VOLO(Aereo) $\subseteq$ AEREO(Codice)
ABILITAZIONE(Aereo) $\subseteq$ AEREO(Codice)
ABILITAZIONE(Pilota) $\subseteq$ PILOTA(Matricola)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi degli aerei tali che tutti i piloti abilitati al volo per quegli aerei guadagnano più di 50000€."

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome dei piloti che guadagnano più del prezzo di qualsiasi volo tra `Roma` e `Milano` effettuato da un aereo per il quale sono abilitati".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni pilota con stipendio maggiore di 60000€ che sia abilitato per almeno tre aerei, la media delle velocità degli aerei per cui è abilitato. Si escludano i piloti abilitati solo per aerei che hanno velocità inferiore alla media della velocità di tutti gli aerei"

Esercizio 6 (punti 4) Si considerino la seguente tabella

TESI(MatricolaStudente, NomeStudente, NomeRelatore, Dipartimento,
FacoltàDocente, Materia, Argomento, NomeCorrelatore, TipoTesi)

e le seguenti assunzioni:

1. relatori e correlatori sono docenti, ed un docente ha un solo dipartimento ed una sola facoltà ma insegna più materie.
2. un docente può fare da correlatore per le tesi di un altro docente solo per una materia
3. un argomento di testi è specifico di una materia
4. una materia può essere insegnata da più docenti.

Decomporre (rispettando le note proprietà delle decomposizioni) la tabella TESI allo scopo di portarla in BCNF, o in 3NF se la prima non fosse possibile. Identificare tutte le chiavi di tutte le tabelle ottenute.

Compito del 15 luglio 2005

Si consideri l'offerta alberghiera del comune di Roccaspaccatella, come descritta dal dépliant informativo mostrato nella pagina seguente.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale della base di dati che contenga tutte le informazioni sull'offerta alberghiera d Roccaspaccatella, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica della base di dati, producendo il relativo schema relazionale completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** necessarie per creare almeno due tabelle della base di dati e per inserire alcune delle tuple corrispondenti ai dati mostrati nel dépliant.

Si consideri il seguente schema di base di dati.

PERSONE(CF, Cognome, Nome, Età)
IMMOBILI(Codice, Via, NumeroCivico, Città, Valore)
PROPRIETÀ(Persona, Immobile, Percentuale)

con i vincoli:

PROPRIETÀ(Persona) $\subseteq$ PERSONE(CF)
PROPRIETÀ(Immobile) $\subseteq$ IMMOBILI(Codice)

in cui l'attributo Percentuale indica la percentuale di proprietà della persona.

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome delle persone che posseggono immobili in almeno due città."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare codice fiscale, nome e cognome delle persone che posseggono un solo immobile e lo posseggono al 100%".

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascuna persona, il codice fiscale, il nome, il cognome e il valore complessivo degli immobili di sua proprietà". Il valore è la somma dei valori ciascuno pesato con la percentuale di proprietà: se una persona possiede un immobile di valore 150 al 100% e uno di valore 200 al 50%, allora il valore complessivo sarà $(150 \times 100)/100 + (200 \times 50)/100 = 250$.

Comune di Roccaspaccatella
ALBERGHI

Roccaspaccatella di sopra (altitudine 1700m)

- Hotel Stella Alpina**** (50 stanze)
  Via della vallelunga 123
  Tel: 0499 123456 Fax: 0499 123457

	Bassa stagione	Media stagione	Alta stagione
Mezza pensione	50	70	90
Pensione completa	60	80	110
Bed and breakfast	30	50	N.D.
Week-end mezza pensione	90	120	N.D.

  Supplemento singola 20%
  Supplemento bambino (lettino aggiunto) 20%
  Sconto tripla 10%
  Sconto gruppi 10%

- Hotel Rododendro*** (65 stanze)
  Via della vallelunga 145
  Tel: 0499 125433

	Bassa stagione	Media stagione	Alta stagione
Bed and breakfast	30	50	70
Week-end bed and breakfast	50	70	130

  Sconto gruppi 15%
  Supplemento bambino (lettino aggiunto) 20%

- . . . (seguono altri alberghi)

Roccaspaccatella di sotto (altitudine 1400m)

- Hotel Valle*** (55 stanze)
  Via Roma 23
  Tel: 0499 123324 Fax: 0499 125432

	Bassa stagione	Media stagione	Alta stagione
Mezza pensione	45	60	N.D.
Pensione completa	60	80	110
Bed and breakfast	30	50	N.D.

  Supplemento bambino (lettino aggiunto) 15%

- . . . (seguono altri alberghi)

Roccaspaccatella di dentro (altitudine 1500m)

- . . . (vari alberghi)

Bassa stagione: 1.06 - 10.07 e 1.09 - 30.09
Media stagione: 11.07 - 31.07 e 22.08 - 31.08
Alta stagione: 1.08 - 21.08

Compito del 30 agosto 2005

Si vuole progettare la basi di dati di un'applicazione relativa alla gestione dei testi d'esame per un corso universitario, descritta dalle seguenti specifiche.

Ciascun compito d'esame è assegnato da un docente, per un corso, in una certa data. Esso consiste di una serie di domande (di tipo a risposta multipla). Si vuole tenere traccia del numero totale di domande proposto per ciascun compito. Ad una domanda corrisponde un punteggio; la domanda consiste del testo ed eventualmente di una o più figure, di cui è noto il nome del file. Una figura appartiene solo a una domanda. Ciascuna domanda è associata ad una serie di risposte proposte di cui qualcuna è corretta (anche più d'una) e le altre sono sbagliate. Una risposta consiste del testo proposto e può essere proposta come risposta a più domande; inoltre può essere corretta per una domanda ma sbagliata per un'altra.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica della base di dati, producendo il relativo schema relazionale completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere le istruzioni **SQL** necessarie per creare almeno due tabelle della base di dati e per inserire alcune tuple che si ritengono ragionevoli.

Si consideri il seguente schema di base di dati.

LIBRO(Codice, Titolo, CasaEditrice, Anno, Prezzo)
AUTORE(Nome, Libro)
UTENTE(Tessera, Nome, Indirizzo, Telefono)
PRESTITO(Libro, Utente, DataInizio, DataFine)

con i vincoli:

AUTORE(Libro) $\subseteq$ LIBRO(Codice)
PRESTITO(Libro) $\subseteq$ LIBRO(Codice)
PRESTITO(Utente) $\subseteq$ UTENTE(Tessera)

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il nome degli utenti che hanno preso in prestito almeno due libri diversi di Ugo Foscolo."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il numero totale di volte in cui ciascun autore è stato letto, ordinando il risultato in maniera decrescente per numero di volte".

Esercizio 6 (punti 6) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome e il telefono degli utenti che hanno preso in prestito *tutti* i libri di Alessandro Manzoni".

Compito del 20 settembre 2005

Esercizio 1 (punti 10) Si vuole progettare una base di dati per gestire le informazioni presenti su un libro di ricette di cucina. Ogni ricetta ha un numero progressivo, un titolo, un livello di difficoltà e un tempo totale di preparazione (somma delle durate dei suoi passi). Gli ingredienti hanno un nome, una quantità ed una unità di misura (litri, grammi, unità, ...).

Le ricette sono suddivise in una sequenza ordinata di passi ciascuno con la propria durata. Ciascun passo è rappresentato dalla lavorazione (cottura, impasto, assemblaggio, ...), dall'insieme degli ingredienti utilizzati. Alcuni passi hanno solo una funzione estetica, e possono anche essere saltati. Alcune ricette possono anche avere una nota, che consiste in un riferimento ad un'altra ricetta, corredata da una descrizione della similitudine tra le ricette.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli. In mancanza di dati quantitativi, si segua l'indicazione di evitare ridondanze e valori nulli.

Si consideri il seguente schema di base di dati relazionale:

SCRITTORE(Nome, AnnoNascita, AnnoMorte*)
LIBRO(Titolo, Autore, Genere, CasaEditrice, Anno, Copie)

con il vincolo:

LIBRO(Autore) $\subseteq$ SCRITTORE(Nome)

in cui l'attributo AnnoMorte ha valore nullo per gli scrittori viventi.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le case editrici che nell'anno 2000 hanno pubblicato solo libri gialli".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli scrittori non viventi che hanno scritto almeno 4 libri di successo prima dei 40 anni (un libro è di successo se ha venduto almeno 10000 copie)".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare gli scrittori non viventi che non hanno scritto libri nell'anno della loro morte".

Esercizio 6 (punti 4) Fornire il risultato della interrogazione qui sotto a destra in base allo schema e l'istanza generati dai comandi a sinistra.

```
create table R
(
  A integer,
  B integer,
  C char(2),
  primary key(B,C)
)

insert into R values(10,2,'cc');
insert into R values(18,3,'aa');
insert into R values(13,5,'ab');
insert into R values(14,7,'bb');
```

```
select B, C
from R as S
where A >=all (select A
               from R
               where B < S.B + 1
                 and C > S.C)
```

Compito del 12 dicembre 2005

Esercizio 1 (punti 8) Si vuole progettare una base di dati per gestire le informazioni di una libreria. Ciascun libro presente in libreria è identificato da un codice (ISBN, *International Standard Book Numbering*), ha un titolo, un numero di pagine e il numero di copie presenti in libreria.

Ciascun libro è scritto da uno o più autori, dei quali ci interessa il nome, il cognome, la data di nascita e la data della morte (qualora non sia un autore vivente). Il libro è poi edito da una casa editrice di cui ci interessa la ragione sociale (il suo nome), l'indirizzo e la partita IVA.

Una casa editrice ha uno o più magazzini dai quali la libreria può approvvigionarsi. Un magazzino è caratterizzato dal suo indirizzo (c'è un solo magazzino di una data casa editrice per ogni città) ed ha uno o più numeri di telefono al quale rivolgersi.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli. In mancanza di dati quantitativi, si segua l'indicazione di evitare ridondanze e valori nulli.

Si consideri il seguente schema di base di dati relazionale:

STAZIONE(Nome, Provincia)
TRENO(Codice, Categoria, Nome, StazioneOrigine, StazioneDestinazione)
FERMATE(Codice, Stazione, OrarioArrrivo, OrarioPartenza)

con i vincoli:

TRENO(StazioneOrigine) $\subseteq$ STAZIONE(Nome)
TRENO(StazioneDestinazione) $\subseteq$ STAZIONE(Nome)
FERMATE(Codice) $\subseteq$ TRENO(Codice)
FERMATE(Stazione) $\subseteq$ STAZIONE(Nome)

Esercizio 3 (punti 2) Scrivere le istruzioni SQL per creare la base di dati che corrisponde allo schema dato.

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il codice e il nome di tutti i treni che effettuano fermate nella stazione di `Tarvisio Boscoverde` che non siano della categoria `Regionale`."

Esercizio 5 (punti 3) Esprimere la seguente interrogazione in **SQL**: "Per ciascuna provincia, trovare il numero dei treni che hanno come stazione di origine del viaggio una località di tale provincia."

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare la destinazione finale di tutte le coincidenze del treno `ES929` presso la stazione di `Venezia Mestre`." Una coincidenza è, per semplicità, un qualunque treno che effettua una fermata presso la stazione considerata e riparte ad un ora successiva a quella della fermata del treno `ES929` presso la stessa stazione.

Esercizio 7 (punti 4) Dati i seguenti schemi e istanze di relazione, dire qual è il risultato dell'interrogazione riportata di lato.

R

A	B	C
alfa	1	d
beta	5	a
gamma	-1	b
delta	-3	a
epsilon	12	e
eta	-1	b
theta	-5	b

S

A	E	F
eta	y	1
alfa	z	8
beta		2
delta	k	12

```
select C, sum(B) as Somma
from R join S on R.A = S.A
where E is not null
group by C
having count(distinct B) < 2
```

Compito del 15 marzo 2006 (soluzione a pagina 132)

Si vuole progettare una base di dati per gestire le informazioni di interesse per un insieme di operatori nazionali di telefonia mobile. Di ogni operatore telefonico interessa il codice identificativo, il fatturato annuale e la località della sede legale. Di ogni località interessa il codice identificativo, il nome, la provincia e la regione. Di ogni utenza interessa l'operatore telefonico con cui l'utenza stessa ha stipulato il contratto, il numero telefonico dell'utenza stessa (identificatore), il nome dell'utente ed il costo per ogni secondo di conversazione previsto dal contratto. Di ogni utenza interessano anche le telefonate fatte dall'utenza stessa, e di ogni telefonata interessa l'utenza chiamata, la data, il costo e l'ora ed il minuto in cui è iniziata. Una stessa utenza non può iniziare più di una chiamata nello stesso minuto della stessa ora della stessa data. Di ogni telefonata interessa anche la cella che ha gestito l'inizio della telefonata, dove ogni cella è identificata da un numero unico nell'ambito della località in cui si trova. Ci sono due e solo

due tipi di telefonate: "sms" e "fonia". Per le telefonate di tipo "fonia" interessa la durata in secondi, mentre per le telefonate di tipo "sms" interessa il numero di parole di cui è formato il messaggio inviato. Per tutti gli operatori e per tutti i contratti, il costo di una telefonata di tipo "sms" è calcolato contando ogni parola inviata come un secondo di conversazione.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli. In mancanza di dati quantitativi, si segua l'indicazione che alle telefonate di tipo "sms" si accede separatamente rispetto alle telefonate di tipo "fonia".

Si consideri il seguente schema di base di dati relazionale:

LIBRO(Codice, Titolo, CasaEditrice, Anno, Prezzo)
AUTORE(Nome, Libro)
UTENTE(Tessera, Nome, Indirizzo, Telefono)
PRESTITO(Libro, Utente, DataInizio, DataFine*)

con i vincoli:

AUTORE(Libro) $\subseteq$ LIBRO(Codice)
PRESTITO(Libro) $\subseteq$ LIBRO(Codice)
PRESTITO(Utente) $\subseteq$ UTENTE(Tessera)

in cui l'attributo `DataFine` ha valore nullo per i prestiti correnti.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare gli autori che non hanno scritto libri che costano più di 20€"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il numero totale di volte in cui ciascun autore è stato *letto*." Un libro si considera letto dall'utente se è stato tenuto in prestito per almeno 3 giorni (ed è stato restituito).

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome e il telefono degli utenti che hanno preso in prestito *tutti* i libri di Alessandro Manzoni"

Esercizio 6 (punti 5) Si consideri la seguente relazione:

ESAMI(ID_Corso, NomeCorso, Docente, CorsoDiStudi, Data, Aula)

che rappresenta il calendario degli appelli d'esame di una sessione di una facoltà. Si assuma che:

- L'attributo `ID_Corso` identifica il corso
- Più corsi possono avere lo stesso nome, ma solo in corsi di studio diversi
- Ciascun corso ha un unico docente
- Ciascun corso può avere anche più appelli nella stessa sessione, ma non nella stessa data
- Ciascun appello esame si tiene in una sola data, ma anche in più aule
- Non si possono tenere esami diversi nella stessa aula nello stesso giorno

Identificare le dipendenze funzionali e le chiavi della relazione. Se possibile, decomporla in BCNF rispettando le due proprietà fondamentali delle decomposizioni.

Compito del 3 aprile 2006

Si vuole progettare un'applicazione relativa all'utilizzo di posti auto in un insieme di garage. Di ogni garage interessa il codice (identificativo), la superficie, il quartiere in cui è situato ed il proprietario. Ogni quartiere appartiene ad una città, e di ognuno di essi interessa il nome (unico nell'ambito della città), ed

il numero di abitanti. Ogni proprietario di garage è una persona, della quale interessa il codice fiscale, la data di nascita, la città di nascita, la città di residenza, e, se noto, anche il quartiere di residenza. Di ogni città interessa il nome (unico nell'ambito della regione), e la regione. Ogni garage ha un insieme di posti auto, ciascuno identificato da un numero unico nell'ambito del garage stesso. Rispetto alla dimensione, esistono due e solo due tipi di posti auto: posto grande e posto piccolo. Di ogni singolo posto grande di ogni garage interessa la superficie che esso occupa, mentre di ogni singolo posto piccolo di ogni garage interessa l'altezza a disposizione per quel posto. Rispetto all'uso per le auto, esistono due e solo due tipi di posti auto: posto mensile e posto giornaliero. Di ogni singolo posto mensile interessa il suo costo al mese, e di ogni singolo posto giornaliero interessa il costo giornaliero. Inoltre, di ogni singolo posto mensile di ogni garage interessa, in ogni mese, quale auto ha eventualmente utilizzato quel posto. Analogamente, di ogni singolo posto giornaliero di ogni garage interessa, in ogni giorno, quale auto ha eventualmente utilizzato quel posto. Di ogni auto interessa la targa (identificativa), il modello, ed il proprietario (una persona, della quale interessa il codice fiscale, la data e la città di nascita, la città di residenza, e, se noto, anche il quartiere di residenza).

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relazionale:

OPERE(Codice, Titolo, Librettista, Musicista, AnnoComposizione)
SPETTACOLI(Codice, Opera, Regista, Orchestra, Direttore)
CALENDARIO(Spettacolo, Data, Ora)
CANTANTI(Nome, Voce, Nazione)
CANTA(Cantante, Spettacolo, Ruolo)

con i vincoli:

SPETTACOLI(Opera) $\subseteq$ OPERE(Codice)
CALENDARIO(Spettacolo) $\subseteq$ SPETTACOLI(Codice)
CANTA(Cantante) $\subseteq$ CANTANTI(Nome)
CANTA(Spettacolo) $\subseteq$ SPETTACOLI(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i cantanti baritoni che interpretano solo il ruolo di Rigoletto"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i musicisti che sono stati messi in scena per due volte con opere diverse a distanza di meno di 5 giorni".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i cantanti italiani che non hanno mai cantato di pomeriggio (tra le 12 e le 19) un'opera composta prima del 1700"

Esercizio 6 (punti 4) Si consideri la seguente tabella:

LIBRI(Titolo, Autore, CasaEditrice, Anno, Prezzo)

Si identifichino tutti gli insiemi di attributi che potrebbero essere ragionevolmente delle chiavi in una qualche realtà applicativa. Per ciascuna chiave potenziale si formuli in linguaggio naturale l'ipotesi che la supporta.

Compito del 10 luglio 2006

Si vuole progettare una base dati per la gestione della vendita di piante, tenendo conto delle seguenti informazioni:

- Per ciascuna specie di piante sono noti sia il nome latino che il nome comune, ed un codice univoco attraverso cui la specie viene identificata. Per ciascuna specie è inoltre noto se sia tipicamente da

giardino o da appartamento e se sia una specie esotica o no. Le piante possono essere verdi oppure fiorite. Nel caso di specie di piante fiorite, sono note tutte le colorazioni in cui ciascuna specie è disponibile.

- I clienti sono identificati attraverso un codice cliente e sono costituiti da privati e da rivendite. Per ciascun privato sono noti il codice fiscale, il nome e l'indirizzo della persona, mentre per ogni rivendita sono noti la partita IVA, il nome e l'indirizzo della rivendita.
- I fornitori sono identificati attraverso un codice fornitore; per ciascun fornitore sono inoltre noti il nome, il codice fiscale e l'indirizzo. Il fornitore può fornire diverse specie di piante. Tuttavia le piante della stessa specie sono acquistate sempre da uno stesso fornitore.
- Si vuole tener traccia di tutti gli acquisti eseguiti da ciascun cliente. Un acquisto è relativo a una certa quantità di piante appartenenti ad una determinata specie.
- Il listino prezzi, in cui si vuole tener traccia dei prezzi assunti nel tempo da ciascuna specie di piante.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Dato lo schema relazionale:
PERSONA(CodiceFiscale, Nome, Cognome, Genere, Anni)
GENITORE(CodGenitore, CodFiglio)
ABITAZIONE(CodPersona, Via, NumCiv, CAP, Città)

In cui valgono i seguenti vincoli:
GENITORE(CodGenitore) $\subseteq$ PERSONA(CodiceFiscale)
GENITORE(CodFiglio) $\subseteq$ PERSONA(CodiceFiscale)
ABITAZIONE(CodPersona) $\subseteq$ PERSONA(CodiceFiscale)

Esercizio 3 (punti 3) Si tracci il diagramma Entità-Relazione da cui lo schema relazionale è stato derivato.

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare tutti i figli (e le figlie) che abitano in una città in cui non abita alcun genitore"

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare tutti i figli maschi che non abitano nella città in cui abitano tutti i genitori".

Esercizio 6 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Restituire le città in cui abitano i genitori i cui figli hanno un'età media superiore ai trent'anni"

Compito del 6 settembre 2006

Si vuole progettare una base dati per la gestione delle partite di calcio svolte nell'ambito del Campionato Mondiale 2006, tenendo conto delle seguenti informazioni:

- Le squadre che partecipano sono identificate univocamente dal nome della nazione di appartenenza (ad esempio Italia, Germania, Inghilterra, ...). Per ogni squadra è noto il nome dell'attuale allenatore. Inoltre per ogni squadra è noto il nome di un dirigente se disponibile.

- Il campionato è organizzato in turni di gioco. Ciascun turno è identificato univocamente dal nome del turno stesso, ("gironi", "ottavi", "quarti", "semifinali" e "finali"). La base di dati contiene l'elenco delle squadre che prendono parte a ciascun turno di gioco.
- Le partite sono identificate attraverso un numero d'ordine univoco all'interno di ciascun turno di gioco. Per ogni partita sono noti i nomi delle due squadre coinvolte, dove si gioca l'incontro ed a che ora.
- I giocatori sono identificati univocamente dal nome. Per ciascuno è inoltre nota la nazione per cui gioca, e con quale numero di maglia. Per ogni giocatore è ancora noto un recapito. Per ciascuno dei giocatori che ha segnato, si vuole memorizzare, per ogni partita giocata, il minuto di gioco in cui tale giocatore ha segnato un gol, e se questo è avvenuto su rigore.
- Infine la base di dati contiene l'informazione relativa a quale arbitro è stato assegnato a ciascuna partita. Per ogni arbitro è noto il nome, che lo identifica univocamente, un recapito, ed il numero complessivo di presenze al campionato.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relativo ad un campionato di calcio.

SQUADRA(Nome, Città, Sponsor, ColoriSociali, Allenatore)
GIOCATORE(NTessera, Squadra, Numero, Nome, Cognome, DataNascita, Ruolo)
PARTITA(IdPartita, Giornata, SqCasa, SqTrasf, GolCasa, GolTrasf)
GOL(IdPartita, Minuto, Marcatore, Autogol)

con i vincoli:

GIOCATORE(Squadra) $\subseteq$ SQUADRA(Nome)
PARTITA(SqCasa) $\subseteq$ SQUADRA(Nome)
PARTITA(SqTrasf) $\subseteq$ SQUADRA(Nome)
GOL(IdPartita) $\subseteq$ PARTITA(IdPartita)
GOL(Giocatore) $\subseteq$ GIOCATORE(NTessera)

Nella relazione GOL l'attributo Marcatore memorizza il numero di tessera del giocatore che ha segnato il gol, mentre l'attributo AutoGol è un valore booleano che vale `True` se il gol è stato un autogol, `False` altrimenti.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il numero di tessera del giocatore che ha segnato il più tardi in una partita"

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i portieri delle squadre che hanno disputato la partita in cui sono stati segnati più gol".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le squadre che non hanno mai perso nei derby (partita contro squadra della stessa città)"

Esercizio 6 (punti 4) Dati i seguenti schemi e istanze di relazione, dire qual è il risultato dell'interrogazione riportata di lato.

R

a	b	c
alfa	3	d
beta	2	a
gamma	-2	b
delta	-5	a
epsilon	13	e
eta	-3	b
theta	-7	b

S

a	e	f
eta	y	3
alfa	z	4
beta		3
delta	k	17

```
select C, sum(B) as Somma
from R join S on R.A = S.A
where E is not null
group by C
having count(distinct B) > 2
```

Compito del 22 settembre 2006

Si vuole rappresentare una base dati per la gestione dei prodotti disponibili in una farmacia tenendo conto delle seguenti informazioni:

- Ciascun prodotto è caratterizzato univocamente dal nome del prodotto stesso e dall'informazione relativa alla ditta fornitrice del prodotto. I prodotti presenti nella farmacia possono essere medicinali oppure prodotti di profumeria. Per ciascun prodotto è comunque noto l'elenco degli usi possibili del prodotto stesso (ad esempio malattie da raffreddamento, dolori alle ossa, oppure detergente per il viso o per il corpo). Della ditta fornitrice sono invece noti un recapito, il nome, utilizzato per identificare la ditta stessa, ed eventualmente il numero di telefono se disponibile.
- Nel caso dei medicinali, la base dati contiene l'informazione relativa al fatto che un medicinale sia "da banco" oppure se la vendita sia effettuabile solo se viene presentata una ricetta medica. Inoltre è nota la categoria farmacoterapeutica di appartenenza del medicinale (ad esempio antibiotico, oppure anti-infiammatorio) e se esistono interazioni tra quella categoria farmacoterapeutica ed altre categorie farmacoterapeutiche.
- I medicinali sono contenuti in cassetti, contenuti a loro volta in armadietti. Gli armadietti sono identificati da un codice numerico univoco per ciascuna categoria farmacoterapeutica ed i cassetti da un codice numerico univoco per ciascun armadietto.
- Nel caso infine di medicinali che richiedano la ricetta medica, si vuole tener traccia di ogni vendita effettuata per quel medicinale, indicando il giorno, la quantità ed il nome del medico che ha fatto la prescrizione.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Dato lo schema relazionale:
PERSONA(CodiceFiscale, Nome, Cognome, Genere, Anni)
GENITORE(CodGenitore, CodFiglio)
ABITAZIONE(CodPersona, Via, NumCiv, CAP, Città)

In cui valgono i seguenti vincoli:
GENITORE(CodGenitore) $\subseteq$ PERSONA(CodiceFiscale)
GENITORE(CodFiglio) $\subseteq$ PERSONA(CodiceFiscale)
ABITAZIONE(CodPersona) $\subseteq$ PERSONA(CodiceFiscale)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le persone che non hanno figli conviventi"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le coppie nonno/nonna che hanno almeno 4 nipoti maggiorenni".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le mamme non nonne che hanno avuto figli con uomini diversi".

Esercizio 6 (punti 4) Si consideri la seguente tabella e le successive assunzioni:

CORSI(NomeCorso, CorsoDiStudi, AnnoDiCorso, Crediti, Docente, Assistente, Dipartimento)

- Un docente può insegnare più corsi, ma un corso ha un solo docente
- Esistono corsi con lo stesso nome, ma solo in corsi di studio diversi
- Un docente afferisce ad un solo dipartimento, ma ha più assistenti (per corsi diversi, però)
- Corsi con lo stesso nome hanno lo stesso numero di crediti

Si determini se la tabella è in BCNF, ed in caso contrario produrre, se possibile, una decomposizione in BCNF che rispetti le due proprietà fondamentali delle decomposizioni.

Compito del 10 gennaio 2007

Si vuole rappresentare una base dati per la gestione di una catena di centri di noleggio di DVD, tenendo conto delle seguenti informazioni:

- Ogni centro è identificato attraverso un codice numerico; inoltre viene riportato l'indirizzo del centro ed il numero di telefono.
- La base di dati contiene le informazioni relative a tutte le persone impiegate presso la catena. Per ciascun impiegato sono noti il codice fiscale, il nome, il titolo di studio ed uno o più recapiti telefonici. Gli impiegati possono essere spostati da un centro all'altro a seconda delle esigenze; si vuole pertanto tenere traccia di tutti i *periodi di servizio* che un impiegato ha prestato presso un centro (con data di inizio e di fine) e della carica che ha rivestito in quel periodo (per esempio, cassiere o commesso). Ovviamente un impiegato può servire in più periodi successivi nello stesso centro.
- I film disponibili presso la catena sono identificati dal titolo e dal nome del regista; inoltre sono noti l'anno in cui il film è stato girato, l'elenco degli attori principali del film, il costo corrente di noleggio ed eventualmente i film disponibili presso la catena di cui il film in questione rappresenta la versione "remake".
- Per ogni film è nota la collocazione all'interno di ciascun centro. In particolare, sono noti il settore, la posizione all'interno del settore ed il numero di copie in cui il film è disponibile nel centro. Ciascun settore è identificato attraverso un codice numerico univoco all'interno del centro e dal codice del centro stesso.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Dato lo schema relazionale:
PERSONA(CodiceFiscale, Nome, Cognome, Genere, Anni)
GENITORE(CodGenitore, CodFiglio)
ABITAZIONE(CodPersona, Via, NumCiv, CAP, Città)

In cui valgono i seguenti vincoli:
GENITORE(CodGenitore) $\subseteq$ PERSONA(CodiceFiscale)
GENITORE(CodFiglio) $\subseteq$ PERSONA(CodiceFiscale)

Abitazione(CodPersona) $\subseteq$ Persona(CodiceFiscale)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i genitori che non hanno figli conviventi"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le terne di nomi ⟨nonno, nonna, nipote⟩".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare le mamme non nonne che hanno avuto figli con uomini diversi".

Esercizio 6 (punti 4) Si consideri la seguente tabella e le successive assunzioni:

Corsi(NomeCorso, CorsoDiStudi, AnnoDiCorso, Crediti, Docente, Assistente, Dipartimento)

- Un docente può insegnare più corsi, ma un corso ha un solo docente
- Esistono corsi con lo stesso nome, ma solo in corsi di studio diversi
- Un docente afferisce ad un solo dipartimento
- Un docente ha più assistenti, ma per corsi diversi (non per lo stesso corso)
- Corsi con lo stesso nome hanno lo stesso numero di crediti

Si determini se la tabella è in BCNF, ed in caso contrario produrre, se possibile, una decomposizione in BCNF che rispetti le due proprietà fondamentali delle decomposizioni.

Compito del 21 marzo 2007 **(soluzione a pagina 134)**

Si vuole rappresentare una base dati per la gestione dei corsi di una scuola di una qualche attività artistica. La base di dati deve contenere le seguenti informazioni:

- I corsi sono organizzati per livelli. Ciascun livello è identificato dal nome del livello stesso (ad esempio: Base, Intermedio, Avanzato, Agonistico); inoltre sono specificati per ciascun livello i concetti fondamentali che vengono spiegati e se viene richiesto di sostenere un esame finale.
- I corsi sono identificati dal nome del livello cui afferiscono e da un codice progressivo, necessario per distinguere corsi che fanno riferimento allo stesso livello. Per ciascun corso sono note la data di attivazione, il numero di iscritti e l'elenco dei giorni in cui è tenuto.
- Per gli insegnanti sono noti il nome, l'indirizzo, la nazione di provenienza, ed i corsi a cui sono stati assegnati. Si assuma che a ciascun corso sia assegnato un unico insegnante.
- Per gli allievi sono noti il nome, un recapito telefonico, il corso a cui sono iscritti, la data di iscrizione al corso, il sesso e l'eventuale *tutore* (un insegnante della scuola che segue e consiglia l'allievo). Gli allievi possono anche prenotare lezioni private individuali, qualora vogliano approfondire alcuni aspetti. Si vuole tener traccia di tutte le lezioni private eventualmente richieste da un allievo, in quale data e con quale insegnante.
- La scuola organizza poi un insieme di esibizioni. Ciascuna esibizione è identificata da un codice progressivo, e sono noti il giorno e l'ora in cui verrà tenuta e i docenti e gli allievi che vi partecipano. Gli allievi però possono partecipare solo alle esibizioni amatoriali, mentre le esibizioni professionali sono eseguite solo dai docenti. Per le esibizioni professionali è anche fissato il prezzo del biglietto d'ingresso.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, cercando di evitare per quanto possibile la

presenza di valori nulli.

Dato lo schema relazionale:

BEVITORE(Nome, Nazionalità, Età)
BIRRA(Nome, Produttore, Nazione)
BAR(Nome, Indirizzo)
FREQUENTA(Bevitore, Bar)
VENDE(Bar, Birra, Prezzo)
PIACE(Bevitore, Birra)

con i vincoli:

FREQUENTA(Bevitore) ⊆ BEVITORE(Nome)
PIACE(Bevitore) ⊆ BEVITORE(Nome)
FREQUENTA(Bar) ⊆ BAR(Nome)
VENDE(Bar) ⊆ BAR(Nome)
VENDE(Birra) ⊆ BIRRA(Nome)
PIACE(Birra) ⊆ BIRRA(Nome)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le coppie bevitore/birra tale che al bevitore piace la birra e frequenta un bar in cui la vendono per meno di 2 €".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascun bevitore il numero totale di birre diverse prodotte nella sua nazione e vendute in almeno un bar che frequenta."

Esercizio 5 (punti 6) Esprimere la seguente interrogazione in **SQL**: "Trovare i bevitori che frequentano solo bar in cui si vende almeno una birra che gli piace".

Esercizio 6 (punti 2) Spiegare brevemente il concetto di chiave e come le chiavi vengono specificate nella creazione di una tabella in SQL.

Compito del 20 aprile 2007 **(soluzione a pagina 136)**

Si vuole progettare una base dati per la gestione delle opere presenti in un museo. Le opere sono classificate in base al loro tipo. I tipi principali sono: *dipinto* e *scultura*. Per i dipinti interessa la tecnica con cui è stato fatto, le dimensioni e lo stile. Per le sculture interessa il materiale, il peso e lo stile. Per le opere di altro tipo interessano solo il tipo stesso e lo stile.

Le opere d'arte sono anche classificate come *permanenti* e *in prestito*. Per le opere permanenti interessa la data e il prezzo di acquisto, mentre per quelle prestate interessa il museo da cui provengono, la data del prestito e la data presunta di restituzione.

Per un'opera interessa anche la sua epoca, la cultura a cui si riferisce e la nazione in cui si è sviluppata quella cultura.

Il museo tiene traccia anche dell'artista che ha realizzato un'opera, se noto. Degli artisti interessa il nome (assunto univoco), la data di nascita, la data di morte (se non è vivente), la cultura, la nazione di nascita e l'epoca in cui è vissuto.

Il museo organizza delle esposizioni, con nome, date di inizio e di fine e le opere che vi sono esposte.

Infine, si tengono informazioni sui musei con cui il museo interagisce per scambi di opere. Di ciascun museo interessa in nome, la città e la nazione in cui è ubicato, il numero di telefono e l'indirizzo di posta elettronica.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema relazionale di basi di dati:

GIOCATORE(Nome,Nazionalità, DataNascita)
PARTITA(Codice, Vincitore, Perdente)
SETPARTITA(Partita, Numero, Punti1, Punti2)

con i vincoli:

$$\text{Partita(Vincitore)} \subseteq \text{Giocatore(Nome)}$$
$$\text{Partita(Perdente)} \subseteq \text{Giocatore(Nome)}$$
$$\text{SetPartita(Partita)} \subseteq \text{Partita(Codice)}$$

che memorizza i risultati di un insieme di partite di tennis, con i punteggi di tutti i set giocati. L'attributo `Punti1` memorizza i punti del giocatore che ha vinto la partita (che non necessariamente ha vinto quel set) e `Punti2` quelli del giocatore che ha perso.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi dei giocatori che non hanno mai battuto un avversario più vecchio".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi dei giocatori che non hanno mai vinto un set".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni giocatore il numero di set vinti in partite che ha vinto e il numero di set vinti nelle partite che ha perso" utilizzando, se lo si ritiene necessario, alcune viste.

Esercizio 6 (punti 3) Si discuta, anche tramite esempi in SQL, il concetto di *vista* illustrando i vantaggio derivanti dal suo utilizzo.

Compito del 26 giugno 2007

Esercizio 1 (punti 6) Si consideri la seguente relazione che rappresenta alcune informazioni sui prodotti di una falegnameria e i loro componenti. Vengono indicati: il tipo del componente (attributo Tipo), la quantità del componente necessaria per un certo prodotto (attributo Q), il prezzo unitario del componente di un certi prodotto (attributo PC), il fornitore del componente (attributo Fornitore) e il prezzo totale del singolo prodotto (attributo PT).

Prodotto	Componente	Tipo	Q	PC	Fornitore	PT
Libreria	Legno	Noce	50	10000	Forrest	400000
Libreria	Bulloni	B212	200	100	Bolt	400000
Libreria	Vetro	Cristal	3	5000	Clean	400000
Scaffale	Legno	Mogano	5	15000	Forrest	300000
Scaffale	Bulloni	B212	250	100	Bolt	300000
Scaffale	Bulloni	B412	150	300	Bolt	300000
Scrivania	Legno	Noce	10	8000	Wood	250000
Scrivania	Maniglie	H621	10	20000	Bolt	250000
Tavolo	Legno	Noce	4	10000	Forrest	200000

Individuare la dipendenze funzionali e le chiavi di questa relazione. Trovare, se esiste, una decomposizione in BCNF senza perdita e che rispetti le dipendenze.

Esercizio 2 (punti 4) Siano date le seguenti relazioni (con valori nulli)

R1

A	B	C
1	xz	4
3	yy	4
7	yz	2
6	yz	3
5	yz	3

R2

C	D	E
4	f	zz
4	g	yz
2	f	zz
5	h	xx
5	j	xz

R3

F	G	H
yz	34	2
yz	24	3
zq		4
yz	12	7
		8

Si calcoli il risultato della seguenti espressione algebrica, riportando anche i risultati intermedi.

$$(\pi_{B,C} R1 - (\pi_{B,C}(R1 \bowtie R2))) \bowtie_{F=B} R3$$

Esercizio 3 (punti 5) Si consideri il seguente schema relazionale di basi di dati:

GIOCATORE(Nome,Nazionalità, DataNascita)
PARTITA(Codice, Vincitore, Perdente, Data)
SETPARTITA(Partita, Numero, Punti1, Punti2)

con i vincoli:

PARTITA(Vincitore) $\subseteq$ GIOCATORE(Nome)
PARTITA(Perdente) $\subseteq$ GIOCATORE(Nome)
SETPARTITA(Partita) $\subseteq$ PARTITA(Codice)

che memorizza i risultati di un insieme di partite di tennis, con i punteggi di tutti i set giocati. L'attributo `Punti1` memorizza i punti del giocatore che ha vinto la partita (che non necessariamente ha vinto quel set) e `Punti2` quelli del giocatore che ha perso.

Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi dei giocatori che non hanno mai battuto un avversario più giovane e di nazionalità diversa dalla propria"

Esercizio 4 (punti 5) Dato lo schema dell'esercizio precedente, esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e le nazionalità dei giocatori che non hanno perso un set nel 2007."

Esercizio 5 (punti 6) Progettare uno schema ER la cui traduzione dia luogo allo schema relazionale dell'esercizio 3.

Compito del 10 settembre 2007

Si vuole progettare una base dati per la gestione dei voli giornalieri di un aeroporto. I dati di interesse riguardano i voli in partenza e in arrivo all'aeroporto, per i quali interessa il codice del volo, l'orario previsto di partenza/arrivo e (solo per le partenze) quello di imbarco. Inoltre, per i voli interessa tener traccia anche degli eventuali ritardi, memorizzando l'orario effettivo di partenza/arrivo, ma anche ciascun annuncio di ritardo con il nuovo orario previsto e l'orario in cui è stato dato l'annuncio (per un volo possono essere dati anche più annunci). Ogni volo è gestito da una compagnia aerea (con nome e sede) tramite un velivolo di sua proprietà. Dei velivoli interessa la compagnia proprietaria, il modello del velivolo, la data di costruzione del velivolo e il numero di posti del modello. Alcuni voli possono essere in *code sharing*, cioè altre compagnie, oltre alla compagnia che lo opera concretamente, danno a quel volo un loro codice.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli. In mancanza di dati quantitativi, si segua l'indicazione di evitare ridondanze e valori nulli.

Si consideri lo schema relazionale risultante dall'esercizio precedente.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare codice e compagnia dei voli che non sono in code-sharing, cioè né hanno altre compagnie che offrono quel volo né sono voli operati da un'altra compagnia".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascuna compagnia il numero totale di voli, operati concretamente, che sono in ritardo di almeno un'ora".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le compagnie che utilizzano al più 5 diversi modelli di aereo per i voli che operano".

Esercizio 6 (punti 4) Si definiscano le condizioni che rendono una qualsiasi tabella rettangolare contenente dei valori qualsiasi una relazione nel modello relazionale.

Compito del 21 settembre 2007

Si vuole creare una base dati per il mantenimento delle informazioni sui passeggeri e sui bagagli relativi ai voli aerei di una data compagnia.

- I voli sono caratterizzati da un numero del volo (codice IATA della compagnia aerea, costituito da tre lettere, seguito da un numero a 4 cifre), dall'aeroporto di partenza e da quello di destinazione.
- I passeggeri sono identificati dal loro cognome e dall'iniziale del nome. I passeggeri possono avere particolari restrizioni alimentari (ad es., vegetariani, allergie, ...). I passeggeri possono avere uno o più bagagli registrati.
- I bagagli sono caratterizzati da un codice univoco (codice IATA della compagnia aerea, seguito da un numero a 6 cifre), dal peso, dal tipo e da un fattore di fragilità del tipo di bagaglio.
- I passeggeri sono imbarcati sui voli in una determinata data (lo stesso volo, con lo stesso numero, può essere effettuato in date diverse per un determinato periodo temporale, es. un anno), hanno assegnato un posto a sedere (composto da numero della fila più una lettera).

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica della base di dati, producendo il relativo schema relazionale completo di vincoli.

Si consideri lo schema relazionale generato dalle seguenti istruzioni SQL.

```
create table Spettatore
(
  NumTessera integer,
  Nome varchar(20) not null,
  Cognome varchar(20) not null,
  Indirizzo varchar(40),
  Citta varchar(20),
  Telefono varchar(20),
  primary key (NumTessera)
);

create table Regista
(
  Nome varchar(20) not null,
  Cognome varchar(20) not null,
  DataNascita date not null,
  DataMorte date,
  Nazionalita varchar(30),
  primary key (Nome, Cognome)
);

create table Film
(
  Titolo varchar(30),
  DataProduzione date not null,
  NazioneProduzione varchar(30),
  NomeRegista varchar(20) not null,
  CognomeRegista varchar(20) not null,
```

```
  primary key (Titolo),
  foreign key (NomeRegista, CognomeRegista) references Regista(Nome, Cognome)
);

create table Visione
(
  TesseraSpettatore integer,
  TitoloFilm varchar(30),
  DataOra timestamp not null,
  primary key (TesseraSpettatore, TitoloFilm),
  foreign key (TesseraSpettatore) references Spettatore(NumTessera),
  foreign key (TitoloFilm) references Film(Titolo)
);
```

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare gli spettatori di Pordenone che non hanno mai visto un film del regista Kubrick" .

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare tutti gli spettatori dei film girati da registi stranieri."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il/i regista/i del film meno visto (ossia di quello che ha avuto il minimo numero di spettatori)."

Esercizio 6 (punti 4) Scrivere la definizione di 3NF e mostrare degli esempi di schemi di basi di dati che soddisfano e non soddisfano la 3NF.

Compito del 9 gennaio 2008

Si consideri il seguente schema di base di dati relativo ad un campionato di calcio.

> Squadra(Nome, Città, Sponsor, ColoriSociali, NomeAllenatore, CognomeAllenatore)
> Giocatore(NTessera, Squadra, Numero, Nome, Cognome, DataNascita, Ruolo)
> Partita(IdPartita, Giornata, SqCasa, SqTrasf, GoalCasa, GoalTrasf)
> Gol(IdPartita, Minuto, Tempo, Marcatore)

con i vincoli:

> Giocatore(Squadra) $\subseteq$ Squadra(Nome)
> Partita(SqCasa) $\subseteq$ Squadra(Nome)
> Partita(SqTrasf) $\subseteq$ Squadra(Nome)
> Gol(IdPartita) $\subseteq$ Partita(IdPartita)
> Gol(Marcatore) $\subseteq$ Giocatore(NTessera)

Esercizio 1 (punti 4) Specificare se, oltre alle chiavi primarie indicate, esistono altre chiavi per le relazioni, specificando anche sotto quali ipotesi queste sono chiavi.

Esercizio 2 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome degli attaccanti che non hanno mai segnato nel secondo tempo".

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome dei giocatori che hanno lo stesso cognome dell'allenatore di un'altra squadra della stessa città"

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome dei giocatori del `Siena` che hanno segnato in tutte le partite in cui la propria squadra ha vinto in casa"

Esercizio 5 (punti 5) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale precedente.

Esercizio 6 (punti 8) Si consideri lo schema di relazione relativo ai dati di una biblioteca costituito dai seguenti attributi

PRESTITI(NomeUt,IndUt,TelUt,TipoUt,NumL,GenereL,TitoloL,AutoreL,Durata,DataP,DataR)

corrispondenti, rispettivamente, al nome utente, indirizzo utente, telefono utente, tipo utente (es. studente, pensionato, ...), codice libro, genere libro (es. romanzo, manuale, ...), titolo libro, autore libro, durata massima del prestito, data prestito, data restituzione.

Si consideri il seguente insieme di dipendenze funzionali su tale relazione:

- $NomeUt \rightarrow IndUt, TelUt, TipoUt$
- $NumL \rightarrow TitoloL, AutoreL, GenereL$
- $TipoUt, GenereL \rightarrow Durata$
- $NumL, DataP \rightarrow NomeUt, DataR$
- $NumL, DataR \rightarrow NomeUt, DataP$

Si richiede di:

1. determinare le chiavi della relazione;
2. determinare se lo schema è in BCNF o in 3NF;
3. mostrare una decomposizione senza perdita dello schema in BCNF;

Compito del 20 marzo 2008 (soluzione a pagina 138)

Si vuole rappresentare una base dati per la gestione di un corso professionale, tenendo conto delle seguenti informazioni. Per ciascun partecipante, vogliamo memorizzare nome, cognome, data di nascita, se è sposato, e, nel caso lo sia, il numero di figli. Vogliamo, poi, memorizzare le città in cui risiedono e le città in cui sono nati, insieme al numero di abitanti. Per le città capoluogo di regione, vogliamo memorizzare anche la regione. Vogliamo poi sapere le lezioni che i partecipanti hanno frequentato, con i (o il) docenti che le hanno svolte (nome, cognome, e tipo di ente di provenienza), il corrispondente argomento ed il giorno in cui si sono svolte. Ad ogni lezione va associato un numero progressivo. Relativamente ai docenti provenienti dall'Università, si vuole memorizzare l'università da cui provengono (compresa la città in cui è ubicata) e la materia che hanno ivi in affidamento. Si vuole anche sapere in quale tipo di scuola i partecipanti hanno ottenuto l'ultimo titolo di studio e in quale città la scuola ha sede.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Si consideri il seguente schema di base di dati.

PERSONE(<u>Nome</u>, Età)
CANI(<u>Nome</u>, Età)
CURA(<u>Persona, Cane</u>)

con i vincoli:

CURA(Persona) $\subseteq$ PERSONE(Nome)
CURA(Cane) $\subseteq$ CANI(Nome)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome di ogni persona che cura tutti i cani di 5 anni."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni numero intero che corrisponde all'età di almeno una persona minorenne, restituire l'età media dei cani curati dalle persone di quella età"

Esercizio 5 (punti 4) Fornire il risultato della interrogazione qui sotto a destra in base allo schema e l'istanza generati dai comandi a sinistra.

```
create table R
(
  A integer,
  B integer,
  C integer,
  D integer,
  primary key(A,B)
)
insert into R(A,B,D) values(1,1,4); insert into R values(2,5,3,8);
insert into R values(1,3,4,5); insert into R values(3,2,7,8);
insert into R values(1,5,3,5); insert into R values(4,1,5,4);
insert into R values(2,1,7,7); insert into R values(4,2,5,6);
insert into R(A,B,C) values(4,5,6);
```

```
select A, B
from R as R1
where D <= 5
  and C <=any (select B
               from R
               where B <= R1.D - 2)
```

Esercizio 6 (punti 5) Si consideri la seguente relazione che memorizza i contratti attraverso i quali il fornitore si impegna a fornire una quantità Qta di un dato pezzo ad un progetto per un dipartimento.

CONTRATTI(Codice, Fornitore, Progetto, Dipartimento, Pezzo, Qta, Valore)

sotto le seguenti condizioni aggiuntive:

- Un progetto acquista un pezzo usando un unico contratto.
- Un dipartimento acquista al più un oggetto da un fornitore.
- Un progetto si rivolge ad un unico fornitore.

Si individuino le dipendenze funzionali (oltre a quelle dalla chiave Codice) e si effettui una decomposizione in BCNF senza perdita.

Compito del 9 aprile 2008 **(soluzione a pagina 140)**

Si richiede di progettare la base di dati di un'applicazione relativa alle macchine che erogano bibite. Di ogni macchina erogatrice interessano il codice (unico per la ditta produttrice), la ditta produttrice, il tipo, e le bibite che la macchina eroga attualmente, con il relativo prezzo praticato, e la data in cui si è iniziato a praticare tale prezzo. Di ogni macchina interessano anche i prezzi eventualmente praticati precedentemente per le bibite attualmente erogate (con il relativo periodo specificato con data di inizio e data di fine), e le bibite che la macchina ha erogato e che non eroga più, sempre con i relativi prezzi nei vari periodi. Per ogni macchina erogatrice e per ogni bibita, interessa poi avere informazioni su ogni prelevamento effettuato dai clienti, con data e orario (espresso in ora, minuti e secondi) in cui è stato effettuato. Ogni prelevamento riguarda una sola bibita, e si assume che le macchine erogatrici consentano al massimo un prelevamento al secondo. Di ogni macchina erogatrice interessa anche in quale quartiere di quale città è ubicata. Di ogni quartiere interessano il codice (unico all'interno della città) ed il livello sociale (numero intero positivo). Ogni città è identificata da un codice, e di ogni città interessano la nazione ed il numero di abitanti. Di ogni bibita interessano il codice, il prezzo standard praticato al bar, e la ditta produttrice. Di ogni ditta produttrice di macchine erogatrici interessano la ragione sociale (identificativo), il fatturato, il numero di dipendenti, e la nazione in cui è situata la sede. Di ogni ditta produttrice di bibite interessano la ragione sociale (identificativo), il fatturato, l'anno di fondazione, e la città in cui è situata la sede.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Si consideri il seguente schema di base di dati.

VIDEO(Codice,Regista,Anno)
ASSOCIAZIONE(Video,Brano)
BRANOMUSICALE(Codice,Titolo,Artista,Anno)

con i vincoli:

ASSOCIAZIONE(Video) $\subseteq$ VIDEO(Codice)
ASSOCIAZIONE(Brano) $\subseteq$ BRANOMUSICALE(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il codice del video e l'autore del brano tali che il video è uscito 10 o più anni dopo la pubblicazione del brano al quale è associato."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il codice dei brani per i quali tutti i video associati hanno come regista l'artista del brano."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni artista restituire il numero dei suoi brani musicali ed il numero dei "suoi" video (ovvero dei video associati ad un brano musicale di cui egli è l'artista)"

Esercizio 6 (punti 5) Si consideri la seguente relazione che memorizza gli spettacoli eseguiti dagli animali di uno zoo.

ZOO(CodAnimale, GenereAnimale, Gabbia, CodAddetto, NomeAddetto, GiornoPulizia, GiornoSpettacolo, OraSpettacolo)
sotto le seguenti condizioni:

- Ogni gabbia è pulita da un solo addetto
- Ogni gabbia è pulita sempre nello stesso giorno
- Un addetto pulisce più gabbie
- In ogni gabbia possono esserci più esemplari
- Ad ogni spettacolo partecipano diversi esemplari
- C'è un solo spettacolo al giorno

Si individuino le dipendenze funzionali e le chiavi e si effettui una decomposizione in BCNF senza perdita che preservi le dipendenze.

Compito del 26 giugno 2008

Si richiede di progettare la base di dati relativa alla gestione di un insieme di *prove di selezione*. Ciascuna prova è identificata dalla data in cui è stata effettuata ed è composta da un insieme di domande. Le domande, che possono anche ripetersi nelle varie prove, hanno un codice, un testo ed un insieme di risposte. Le domanda possono essere a risposta singola (una ed una sola risposta giusta) o a risposta multipla (numero di risposte giuste qualsiasi). Le risposte alle domande hanno un numero progressivo riferito alla domanda, un testo e possono essere giuste o sbagliate per quella domanda.

A ciascuna prova si iscrivono dei candidati, dei quali interessa il nome e il codice fiscale. Un candidato può iscriversi a più prove. Di un candidato iscritto ad una prova, nel caso che si presenti per sostenerla, si vuole memorizzare ma anche a quali domande ha risposto esattamente, e il loro numero totale.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per la creazione di due tabelle riguardanti le domande e le risposte.

Esercizio 4 (punti 3) Si scrivano i comandi SQL per modificare il risultato del candidato `Mario Rossi` per la prova del 20 giugno 2008, mettendo giuste tutte le sue risposte.
Si consideri lo schema di base di dati risultante dall'esercizio 2.

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il testo delle domande a risposta multipla che non sono state utilizzate in prove del 2008."

Esercizio 6 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il testo delle domande che, pur essendo state utilizzate nelle prove, mai nessuno vi ha risposto correttamente."

Esercizio 7 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni prova il numero di domande che hanno zero risposte esatte e che sono state indovinate da almeno due candidati"

Compito del 11 settembre 2008

Una rivista periodica di fumetti vuole memorizzare informazioni relative a tutte le storie che ha pubblicato e ai relativi personaggi. Di una storia interessa il titolo, che la identifica, e le informazioni relative alle puntate in cui è stata divisa: per ogni puntata interessa il numero di pagine, il numero d'ordine all'interno della storia e il numero della rivista su cui è stata pubblicata (con la sua data di pubblicazione). I personaggi si dividono in principali e secondari. Per tutti i personaggi interessa il nome, che li identifica. Per i personaggi secondari interessa ricordare le storie in cui sono apparsi, mentre per quelli principali si vogliono memorizzare precisamente anche tutte le puntate di apparizione. Se due personaggi sono parenti, se ne memorizza la relazione di parentela (con anche il grado di parentela).

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Si consideri il seguente schema di base di dati.

ATTORI(Codice, Nome, AnnoNascita)
RECITAZIONE(Attore, Film)
FILM(Codice, Titolo, AnnoProduzione, Regista)
PROIEZIONI(Film, Sala, Incasso, Data)
SALA(Codice, Posti, Nome, Città)

con i vincoli:

RECITAZIONE(Attore) $\subseteq$ ATTORI(Codice)
RECITAZIONE(Film) $\subseteq$ FILM(Codice)
PROIEZIONI(Film) $\subseteq$ FILM(Codice)
PROIEZIONI(Sala) $\subseteq$ SALA(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Per ogni film in cui appaiono due attori nati lo stesso anno trovare il codice del film e i nomi dei due attori"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome delle sale di Udine in cui *Il gattopardo* di Visconti è stato proiettato una sola volta."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni film in cui appaiono solo attori nati prima del 1970 restituire il regista del film ed il numero di attori che vi hanno recitato"

Esercizio 6 (punti 4) Si consideri la relazione R(A,B,C,D,E) con le dipendenze funzionali $AD \to B$, $CB \to A$, $DE \to A$, $A \to E$.

- Trovare tutte le chiavi
- Determinare se lo schema e in 3NF o in BCNF

Compito del 25 settembre 2008

Una ditta che gestisce un motore di ricerca sul web vuole creare una base di dati per tenere traccia della struttura delle pagine indicizzate e delle sessioni che gli utenti hanno con il motore di ricerca. Per quanto riguarda la struttura delle pagine, interessa conoscere, per ogni pagina, l'URL, il titolo, l'insieme delle pagine puntate dai riferimenti (*link*) che appaiono nella pagina, l'insieme dei termini che vi appaiono e, per ogni termine, il numero di volte in cui il termine appare nella pagina.

In ciascuna sessione l'utente presenta un termine di ricerca, il sistema segnala tutte le pagine correlate, e l'utente legge alcune di queste pagine. In dettaglio, per ogni sessione, identificata da un codice, interessa il giorno in cui si è svolta e il termine utilizzato per la ricerca. La sessione può essere eseguita da un utente anonimo, sul quale il sistema non ha informazioni, oppure da un utente registrato, del quale il sistema conosce *username*, *password*, indirizzo *email*, e l'elenco di tutti gli indirizzi IP dai quali l'utente si è collegato nel passato. Per le sessioni con utenti registrati interessa memorizzare anche l'ora di inizio e di fine, l'utente, e la liste delle pagine effettivamente lette dall'utente.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Si consideri il seguente schema di base di dati.

> Studenti(<u>Matricola</u>, Nome, CorsoStudi, DataNascita, Facoltà)
> Corsi(<u>Sigla</u>, Crediti, Docente)
> Iscrizioni(<u>Studente, Corso</u>)
> Docenti(<u>Id</u>, Nome, Dipartimento)

con i vincoli:

> Iscrizioni(Studente) $\subseteq$ Studenti(Matricola)
> Iscrizioni(Corso) $\subseteq$ Corsi(Sigla)
> Corsi(Docente) $\subseteq$ Docenti(Id)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il nome e la data di nascita dello studente più anziano (o degli studenti più anziani) tra quelli iscritti ad Ingegneria Meccanica".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni coppia di corsi che hanno almeno dieci studenti in comune, trovare la sigla e il numero di crediti dei due corsi e il numero di studenti in comune"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare la sigla dei corsi che sono frequentati da tutti gli studenti della Facoltà di Ingegneria"

Esercizio 6 (punti 4) Si consideri il seguente schema relazionale che rappresenta informazioni relative agli spettacoli programmati per una stagione in un insieme di teatri:

SPETTACOLI(Compagnia, Regista, Titolo, Data, Teatro)

1. Si esprimano, se possibile, i seguenti vincoli come dipendenze funzionali:
   (a) due spettacoli contemporanei hanno il regista diverso;
   (b) in ogni teatro c'è al più uno spettacolo al giorno;
   (c) se due spettacoli hanno il regista diverso anche le compagnie sono diverse.
2. Trovare tutte le chiavi della relazione.

Compito del 22 dicembre 2008

Si richiede di progettare la base di dati di un'applicazione per la gestione dei progetti gestiti da un'azienda. Di ogni progetto interessa il nome (identificativo), la durata, la tipologia, i gruppi di lavoro che vi partecipano (almeno uno), e i liberi professionisti che revisionano il progetto (almeno uno). Ogni gruppo di lavoro partecipa ad un solo progetto. Di ogni gruppo di lavoro interessa il codice, unico nell'ambito del progetto a cui partecipa, le risorse finanziare assegnate al gruppo per la partecipazione al progetto, e le persone che sono membri del gruppo (almeno una), ciascuna con il relativo impegno in termini di ore settimanali. Interessa inoltre sapere qual è il membro del gruppo che lo dirige. Ciascuna persona può essere membro di più gruppi di lavoro, ma può dirigerne al massimo uno (di cui deve essere membro). Di ogni persona di interesse per l'applicazione (dipendente o libero professionista) è rilevante il codice fiscale, il nome e la data di nascita. Di ogni dipendente interessa lo stipendio annuo ed il dipartimento in cui lavora, con l'anno in cui ha iniziato a lavorare nel dipartimento. Di ogni libero professionista interessa la partita IVA ed il numero di telefono, se disponibile. Di ogni dipartimento interessa il nome (identificativo), il fatturato e l'anno di fondazione. Interessa infine sapere quali progetti sono di interesse per quali dipartimenti, con il relativo grado di interesse (intero positivo).

Esercizio 1 (punti 8) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando per quanto possibile la presenza di valori nulli.

Si consideri il seguente schema di base di dati.

STUDENTI(<u>Matricola</u>, Nome, Cognome, AnnoDiCorso)
DOCENTI(<u>Id</u>, Nome, Cognome, Dipartimento)
CORSI(<u>Sigla</u>, Nome, Crediti, DocenteTitolare)
ESAMI(<u>Studente, Corso</u>, Docente, Data, Voto)

con i vincoli:

ESAMI(Studente) $\subseteq$ STUDENTI(Matricola)
ESAMI(Corso) $\subseteq$ CORSI(Sigla)
ESAMI(Docente) $\subseteq$ DOCENTI(Id)
CORSI(DocenteTitolare) $\subseteq$ DOCENTI(Id)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare la matricola degli studenti che non hanno mai fatto un esame con il titolare del corso".

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome, cognome e matricola degli studenti che hanno fatto l'esame del corso di `Analisi Matematica` tenuto come titolare dal prof `Rossi` e inoltre hanno una media almeno del 24".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare la matricola degli studenti che hanno fatto tutti gli esami dei corsi tenuti come titolare da `Bianchi`"

Esercizio 6 (punti 4) Si consideri il seguente schema relazionale che rappresenta informazioni relative a dei film, in cui per ciascun film vengono memorizzati il regista e la sua nazionalità, l'anno di produzione, gli attori che hanno lavorato al film e la loro nazionalità.

FILM(Titolo, Regista, NazRegista, AnnoProduzione, Attore, NazAttore)

Supponendo che ciascun film abbia un solo regista e che non si presentino casi di omonimia (stesso titolo) tra i film, si individuino le dipendenze funzionali e le chiavi della relazione.

Compito del 3 febbraio 2009

Una federazione sportiva vuole realizzare la base di dati per la gestione del suo campionato annuale. Il campionato è composto da 10 tappe (ciascuna corrispondente a un singolo torneo) che si svolgono in diverse località. Ciascuna tappa è caratterizzata da data inizio, data fine, nome località, numero progressivo (1, 2, ..., 10) 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 tessera federale, 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 interno al torneo, il livello (gironi, quarti, semifinali, ...), la durata, le coppie sfidanti e il risultato di ciascun set. Si gioca *al meglio dei 3 set* e per ogni set si arriva fino a 10 punti (ad es. 10-8, 7-10, 10-3).

Esercizio 1 (punti 8) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati.

PERSONE(<u>CF</u>, Nome, Cognome, Indirizzo, Comune)

MULTE(<u>Id</u>, Persona, Articolo, Data, Importo)

con il vincolo:

MULTE(Persona) $\subseteq$ PERSONE(CF)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare per ogni persona che abbia preso almeno una multa, il nome, il cognome e l'importo massimo pagato per una singola multa".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone per cui il numero di multe ricevute nel 2008 è superiore al numero di multe ricevute nel 2007".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che hanno preso almeno una multa per violazione dell'Art. 44 e tutte le loro multe hanno un ammontare complessivo compreso tra 300 € e 500 €."

Esercizio 6 (punti 5) Si consideri il seguente schema relazionale

CONTICORRENTI(CodCliente, CognomeCliente, NomeCliente, NumeroCC, TipoCC, DataApertura, CodFiliale, NomeFiliale, CodAgenzia, IndirizzoAgenzia, CAB)

che contiene dati relativi a conti correnti presso una banca, con le seguenti proprietà:

- Ogni conto corrente è identificato da NumeroCC, ha un TipoCC, una DataApertura, uno o più titolari (ognuno con un CodCliente, CognomeCliente e NomeCliente) ed è aperto presso una agenzia di una filiale.
- Per un cliente che abbia più conti, CodCliente, CognomeCliente e NomeCliente hanno gli stessi valori, e CodCliente determina gli altri due valori (ma non viceversa).
- CodFiliale e NomeFiliale identificano entrambi univocamente una filiale. Il valore di CodAgenzia è unico solo nell'ambito di ciascuna filiale: CodFiliale e CodAgenzia individuano l'agenzia, mentre CodAgenzia non è sufficiente.
- Il CAB (Codice di Avviamento Bancario) determina univocamente CodFiliale e CodAgenzia e viceversa.

Si individuino le dipendenze funzionali e le chiavi della relazione. Si verifichi se lo schema dato è in BCNF e, in caso contrario, lo si decomponga in BCNF senza perdita.

Compito del 19 febbraio 2009

Si vuole creare una base dati per il mantenimento delle informazioni sulle merci trasportate da una compagnia di spedizioni.

- I lotti trasportati sono identificati da un codice e sono caratterizzati dal volume di carico, espresso in metri cubi, e dal peso.
- I lotti sono trasportati da un magazzino di partenza ad uno di destinazione in una determinata data.
- È possibile che alcuni lotti facciano tappa in uno o più magazzini durante il trasporto. Delle eventuali tappe interessa anche l'ordine in cui sono state eseguite.
- I magazzini si trovano presso una determinata località e sono identificati da un codice interno alla località. Solo alcuni magazzini particolari possono essere utilizzati per le tappe intermedie. Soli per questi ultimi, si vuole memorizzare anche la capienza totale.
- I lotti sono trasportati per conto di un cliente , del quale ci interessano i classici dati anagrafici e i numeri di telefono.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati (dove `DataMorte` ha valore `NULL` per i registi viventi).

SPETTATORE(NumTessera, Nome, Cognome, Indirizzo, Citta, Telefono)
REGISTA(Nome, Cognome, DataNascita, DataMorte, Nazionalita)
FILM(Titolo, DataProduzione, NazioneProduzione, NomeRegista, CognomeRegista, Durata)
VISIONE(Spettatore, Film, Data, Orario)

con i vincoli:

FILM(NomeRegista, CognomeRegista) $\subseteq$ REGISTA(Nome,Cognome)
VISIONE(Spettatore) $\subseteq$ SPETTATORE(NumTessera)
VISIONE(Film) $\subseteq$ FILM(Titolo)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare tutti i registi italiani i cui film non sono stati visti da alcuno spettatore di `Udine`".

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Per ciascun regista deceduto contare il numero di volte che ha visto un suo film nei suoi ultimi 100 giorni di vita."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome degli spettatori che hanno visto tutti i film di `Stanley Kubrick` prima che morisse."

Esercizio 6 (punti 4) Spiegare il concetto di *chiave* nel modello relazionale. Illustrare, con esempi, i modi utilizzati per specificare le chiavi in **SQL**.

Compito del 26 giugno 2009

Si vuole progettare una base di dati per la gestione del calendario settimanale delle lezioni dei vari corsi di studi offerti da una facoltà universitaria in uno specifico periodo didattico; ad esempio, le lezioni dei corsi di studi offerti dalla Facoltà di Ingegneria relativamente al primo semestre dell'A.A. 2008-09.

Di ogni corso di studi (ad esempio, Laurea Triennale in Ingegneria Elettronica) vogliamo memorizzare il nome, il numero totale di studenti iscritti e il docente che ricopre il ruolo di presidente del consiglio di corso di studi. Di ogni insegnamento (ad esempio Basi di Dati), vogliamo memorizzare il nome; il nome, il cognome e la matricola del docente; il numero di studenti iscritti e il corso di studi cui appartiene. Insegnamenti con lo stesso nome possono appartenere solo a corsi di studi diversi. Si assuma che le lezioni di un dato corso di studi possano essere tenute in edifici diversi, ma che ogni edificio venga utilizzato da un unico corso di studi. Di ogni edificio vogliamo memorizzare il nome, che lo identifica univocamente, l'indirizzo, un recapito telefonico e il numero di aule disponibili (in ogni edificio è presente almeno un'aula). Di ogni aula vogliamo conoscere il nome, il piano, l'edificio in cui si trova e il numero di posti disponibili. Non possono esservi aule con lo stesso nome in uno stesso edificio. Di ogni lezione vogliamo memorizzare l'insegnamento, l'aula, l'edificio, il giorno e la fascia oraria (prima: 8:30–10:30, seconda: 10:30–12:30, ..., quinta:16:30–18:30).

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati (dove `DataMorte` ha valore `NULL` per i registi viventi).

SPETTATORE(<u>NumTessera</u>, Nome, Cognome, Indirizzo, Telefono)
REGISTA(<u>Nome, Cognome</u>, DataNascita, DataMorte, Nazionalità)
FILM(<u>Titolo</u>, DataProduzione, NazioneProduzione, NomeRegista, CognomeRegista, Durata)
VISIONE(<u>Spettatore</u>, Film, <u>Data, Orario</u>)

con i vincoli:
FILM(NomeRegista, CognomeRegista) $\subseteq$ REGISTA(Nome,Cognome)
VISIONE(Spettatore) $\subseteq$ SPETTATORE(NumTessera)
VISIONE(Film) $\subseteq$ FILM(Titolo)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome degli spettatori che hanno visto due film diversi dello stesso regista vivente nello stesso giorno".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome, cognome, numero di tessera e indirizzo degli spettatori che non hanno mai visto uno stesso film due volte nello stesso giorno."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il numero di tessera degli spettatori che hanno visto tutti i film di `Woody Allen`."

Esercizio 6 (punti 4) Disegnare lo schema Entità-Relazione da cui è stato ricavato lo schema relazionale precedente.

Compito del 13 luglio 2009

Esercizio 1 (punti 8) Si vuole progettare una base di dati per la gestione delle opere rappresentate negli anni in un certo teatro (ad esempio al fine di produrre un documento come quello schematizzato nel riquadro).

Stagione 1966-67, direttore artistico Mario De Rossi

- **Così è (se vi pare)** (1917) di Luigi Pirandello (1867-1936) — dal 24/10/1966 al 5/11/1967
- **L'opera da tre soldi** (1928) di Bertolt Brecht (1898-1956) — dal 20/11/1966 al 6/12/1966
- ...

Stagione 1967-68, direttore artistico Luigi De Bianchi

- **Enrico IV** (1921) di Luigi Pirandello (1867-1936) — dal 6/10/1967 al 4/11/1967
- **Morte di un commesso viaggiatore** (1959) di Arthur Miller (1915-2005) — dal 7/11/1967 al 9/12/1967
- **Così è (se vi pare)** (1917) di Luigi Pirandello (1867-1936) — dal 5/01/1968 al 7/02/1968
- ...

Stagione 1968-69, direttore artistico Mario De Rossi ...
Stagione ...

In particolare, sono rilevanti:

- le stagioni (ad esempio, 1966-67), con il relativo direttore artistico;
- le opere rappresentate in ciascuna stagione, con il relativo periodo (ad esempio dal 24/10/1966 al 5/11/1967);
- per ogni opera, l'autore (uno e uno solo, con data di nascita ed eventualmente di morte) e l'anno della prima rappresentazione.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 5) Produrre un nuovo schema Entità-Relazione modificando quello dell'esercizio 1 in base alle seguenti specifiche aggiuntive:

- Per ogni opera rappresentata in una certa stagione, interessano il regista (uno e uno solo) e gli attori principali (uno o più).
- Per ogni opera interessano città e nazione della prima rappresentazione.
- Per attori, registi e direttori artistici, interessano, come per gli autori, anno di nascita ed eventualmente di morte. Una persona può aver svolto, nello stesso momento o in momenti diversi, anche ruoli diversi (ad esempio, attore e regista)

Si consideri il seguente schema di base di dati relazionale.

CLIENTI(Id, Nome, Cognome, NumNoleggi)
FILM(Titolo, Regista, Genere, Durata)
CASSETTE(Id, Film)
DVD(Id, Film)
NOLEGGI(IdCassettaODvd, Cliente, DataPrestito,DataRestituzione)

con i vincoli:

CASSETTA(Film) $\subseteq$ FILM(Titolo)
DVD(Film) $\subseteq$ FILM(Titolo)
NOLEGGI(Cliente) $\subseteq$ CLIENTI(Id)
NOLEGGI(IdCassettaODvd) $\subseteq$ CASSETTE(Id) $\cup$ DVD(Id)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il titolo dei film per i quali esiste un'unica copia il cassetta (e nessuna in DVD)".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il titolo dei cortometraggi (cioè film di durata inferiore ai 30 minuti) di cui sono stati noleggiati più volte i DVD che le videocassette".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare il titolo e il genere dei film le cui cassette sono state prese in prestito più di venti volte nel mese di Gennaio 2009 e non sono presenti in formato DVD."

Esercizio 6 (punti 4) Disegnare lo schema Entità-Relazione da cui è stato ricavato lo schema relazionale precedente.

Compito del 7 settembre 2009

Esercizio 1 (punti 10) Si vuole progettare una base di dati per una ditta che gestisce corsi di aggiornamento professionali di informatica. Ogni corso ha un codice, un titolo, varie parole chiave, è insegnato da uno o più docenti caratterizzati da nome, affiliazione, settore disciplinare (con sigla e denominazione) e curriculum vitae.

Ciascun corso ha vari materiali, che possono essere brochure pubblicitarie, lucidi o dispense. Le dispense e le brochure sono documenti Microsoft Word, mentre i lucidi sono resi con Microsoft PowerPoint oppure in formato Adobe PDF. Ogni materiale è caratterizzato da una data di produzione, un indirizzo Web e una dimensione. Un corso può essere la ripetizione di un precedente corso e in tal caso può riusare alcuni dei materiali, senza necessariamente averne più copie.

I partecipanti al corso hanno nome, recapiti telefonici e e-mail; si dividono in dipendenti delle ditte e partecipanti individuali. Per i primi, è la ditta, caratterizzata da un codice fiscale e una partita IVA, a pagare; i pagamenti vengono effettuati in una certa data e relativamente a un certo numero di partecipanti ad un corso. Invece i partecipanti individuali pagano direttamente le loro quote di iscrizione.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relazionale.

ASILONIDO(Nome,NroPosti,Indirizzo,Città,Provincia)
BAMBINO(CF,Nome,DataNascita,Punteggio)
DOMANDA(Asilo,Bambino,Data,NroPreferenza)
ISCRIZIONE(Asilo,Bambino,Data,RettaMensile)

con i vincoli:

Domanda(Asilo) $\subseteq$ AsiloNido(Nome)
Domanda(Bambino) $\subseteq$ Bambino(CF)
Iscrizione(Asilo) $\subseteq$ AsiloNido(Nome)
Iscrizione(Bambino) $\subseteq$ Bambino(CF)

L'attributo NroPreferenza della relazione Domanda rappresenta l'ordine di preferenza: vale 1 per l'asilo preferito dal bambino, 2 per l'eventuale secondo, 3 per la sua terza scelta e così via.

Un bambino può aver fatto domanda, ma essere non iscritto; è anche possibile che un bambino sia iscritto ad un asilo senza aver fatto domanda.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il codice fiscale e il nome dei bambini che né hanno fatto domanda né si sono iscritti ad asili della provincia di Udine."

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e indirizzo degli asili che non presentano alcun iscritto che l'abbia designato come prima scelta."

Esercizio 5 (punti 5) Costruire in **SQL** una vista PrimiDaAssegnare(CodFisc, NomeAsilo) che estrae il codice fiscale dei bambini con il valore di punteggio più elevato tra tutti quelli non ancora iscritti, associandolo al nome dell'asilo che rappresenta la loro prima scelta.

Esercizio 6 (punti 3) Spiegare brevemente il concetto di chiave e illustrare come le varie chiavi possono essere specificate nella creazione di una tabella in SQL.

Compito del 17 settembre 2009

Esercizio 1 (punti 6) Si vuole progettare una base di dati per un insieme di musei. Ogni museo ha un nome, si trova in una città (della quale interessa anche la nazione, con il nome a la relativa sigla) e ha una serie di sale, ognuna delle quali ha un nome e una dimensione. I musei espongono opere d'arte, per ognuna delle quali interessano l'autore (con codice, cognome, nome, data di nascita ed eventualmente di morte), l'anno di esecuzione e la sala nella quale viene esposta (che si assume fissa).

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Esercizio 3 (punti 5) Estendere lo schema concettuale ottenuto in risposta all'esercizio 1, per tenere conto delle seguenti specifiche aggiuntive:

- È di interesse rappresentare possibili itinerari di visita. Ogni itinerario è relativo ad un solo museo, ha un codice identificativo (unico nell'ambito del museo) ed è costituito da una lista ordinata di sale da visitare e, per ogni sala, da una lista ordinata di opere da vedere.
- Si vogliono rappresentare anche visite guidate che hanno ciascuna un nome, un orario di inizio, una durata e si basano su un certo itinerario. Le visite guidate si ripetono nei vari giorni della settimana con un numero massimo di partecipanti diverso.

Esercizio 4 (punti 4) Si consideri una base di dati il cui schema è composto dalle relazioni R1(A,B,C) e R2(D,E,F). Supponendo che le cardinalità delle due relazioni siano rispettivamente N_1 e N_2, indicare le cardinalità (minime e massime) dei risultati delle seguenti interrogazioni:

```
select *
from R1 join R2 on C = D
```

```
where E > 100

select *
from R1 as X1
where not exists (select *
                  from R1 as Y1 join R2 on Y1.C = D
                  where X1.A = Y1.A and F > 10)

select distinct A, B
from R1 join R2 on B = D
where C = E
```

Si consideri il seguente schema di base di dati relazionale.

PRODOTTI(Codice, Nome, Categoria)
VENDITE(Prodotto, Data, Incasso)

con il vincolo:
VENDITE(Prodotto) $\subseteq$ PRODOTTI(Codice)

e la sua istanza seguente:

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

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

Esercizio 5 (punti 4) Mostrare il risultato delle tre seguenti interrogazioni:

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

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

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

Esercizio 6 (punti 4) Esprimere in **algebra relazionale** la prima delle tre interrogazioni dell'esercizio 5.

Esercizio 7 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni categoria e per ogni data l'incasso complessivo, cioè la somma degli incassi registrati per quella data dai prodotti di quella categoria, mostrando categoria, data e incasso complessivo."

Compito del 18 dicembre 2009 (soluzione a pagina 142)

Esercizio 1 (punti 10) Un'amministrazione comunale deve realizzare una base di dati relativa all'organizzazione di una fiera. A questo riguardo è necessario gestire i dati relativi agli stand espositivi, che hanno un codice e una dimensione. Gli stand sono di due tipi: all'aperto e al chiuso. Gli stand al chiuso sono posizionati in un padiglione, mentre quelli all'aperto sono in una zona comune della fiera. I padiglioni hanno un nome, una dimensione globale e un orario di apertura. Infine, gli stand al chiuso hanno un numero progressivo che li identifica all'interno del loro padiglione.

Per i soggetti espositori, i dati di interesse sono la ragione sociale, l'indirizzo e i prodotti che espongono (con nome e categoria). Interessa inoltre memorizzare anche lo stand in cui ciascun espositore è posizionato per ciascun giorno della fiera, con i relativi costi di locazione da versare al comune. Gli stand al chiuso vengono affittati per tutta la fiera, mentre per quelli all'aperto è possibile che un espositore sia presente solo in alcuni giorni della fiera ed anche che sia assegnato a stand diversi in giorni diversi.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Si consideri il seguente schema di base di dati relazionale.

IMPIEGATI(CF, Nome, Cognome, Genere, Stipendio, Reparto)
REPARTI(Codice, Budget, Direttore)

con il vincolo:

IMPEGATI(Reparto) $\subseteq$ REPARTI(Codice)
REPARTI(Direttore) $\subseteq$ IMPEGATI(CF)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome, cognome e codice fiscale degli impiegati che dirigono un reparto in cui non lavorano"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il codice e il budget dei reparti in cui lavorano tutti impiegati dello stesso genere"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni reparto con più di cinque impiegati totali il numero delle sue impiegate che guadagnano più di 30.000 €"

Esercizio 6 (punti 4) Si consideri il seguente schema di base di dati relazionale relativa a dei terreni di una provincia

TERRENI(IdTerreno, Comune, NumTerreno, Dimensione, PrezzoTotale, Proprietario)
con le seguenti assunzioni:

- l'attributo IdTerreno identifica un terreno
- l'attributo NumTerreno è un numero progressivo assegnato al terreno da ciascun comune della provincia indipendentemente (quindi può ripetersi in comuni diversi)
- l'attributo Dimensione rappresenta i metri quadri del terreno, il prezzo al metro quadro è unico nel comune
- un terreno può avere più proprietari, ma una persona può possedere al massimo un terreno

Scrivere le dipendenze funzionali e identificare le chiavi.

Compito del 4 febbraio 2010

Esercizio 1 (punti 7) Si vuole progettare una base di dati per la gestione degli articoli pubblicati su una rivista scientifica, secondo le seguenti specifiche.

- Gli articoli hanno un titolo, un eventuale sottotitolo, uno o più autori e delle parole chiave.

- Gli autori hanno nome, cognome, email e affiliazione (l'istituzione per la quale lavorano).
- Per ogni istituzione sono di interesse il nome, l'indirizzo, la città, la nazione e la sigla della nazione (ITA, GER, USA, ...).
- La rivista viene pubblicata un certo numero di volte in un anno. Le pubblicazioni di un anno vengono raccolte in un volume (a cui viene dato un titolo complessivo). Ogni pubblicazione ha un numero, unico nel rispettivo volume, una data di pubblicazione e una serie di articoli, per ognuno dei quali viene registrata la pagina di inizio e quella di fine.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 3) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Esercizio 3 (punti 6) Estendere lo schema concettuale ottenuto in risposta all'esercizio 1, per tenere conto delle seguenti specifiche aggiuntive. Mostrare *separatamente* le due estensioni.

- si vogliono gestire più riviste, ognuna con un codice identificante, un nome e uno o più curatori, che possono essere anche autori di articoli (e per i quali interessano le stesse informazioni degli autori)
- gli autori possono cambiare affiliazione e indirizzo di posta elettronica nel tempo e quindi possono avere affiliazione diversa e indirizzo diverso per articoli diversi;

Si consideri il seguente schema di base di dati relazionale.

DETTAGLIO(Ricevuta, Riga, Prodotto, Quantità, Importo)
RICEVUTE(Numero, Data, Cliente)
CLIENTI(Nome, Indirizzo, CAP, Comune)

con i vincoli:
DETTAGLIO(Ricevuta) $\subseteq$ (Numero)
RICEVUTE(Cliente) $\subseteq$ CLIENTI(Nome)

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i prodotti che in tutto il 2009 sono stati acquistati una e una sola volta".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni prodotto la quantità complessiva venduta ai clienti di `Udine` per ciascuna data, escludendo le vendite di importo inferiore a 100 €".

Esercizio 6 (punti 6) Decomporre (senza perdita e preservando le dipendenze) in BCNF la seguente relazione che contiene informazioni relative ad alcuni film, con il regista e gli attori che vi hanno partecipato.

CodFilm	Titolo	Anno	CodRegista	Regista	CodAttore	Attore
1	Roma Città Aperta	1945	101	R. Rossellini	201	Anna Magnani
2	Viaggio in Italia	1954	101	R. Rossellini	203	Ingrid Bergman
3	Casablanca	1942	102	M. Curtiz	203	Ingrid Bergman
3	Casablanca	1942	102	M. Curtiz	202	Humphrey Bogart

Compito del 22 febbraio 2010

Esercizio 1 (punti 10) Si vuole progettare una base di dati per la gestione di un archivio, secondo le seguenti specifiche. Un documento è caratterizzato da autore, titolo, eventuale descrizione, data di creazione, tipo ed estensione (pdf, rtf, txt, zip, ...). Per ciascun tipo di documento è memorizzato il suo strumento per la lettura/scrittura del documento (ad es. Acrobat Reader). Alcuni documenti possono contenere altri documenti, ognuno dei quali è registrato a sua volta nel sistema. Il documento

componente potrebbe essere un allegato, oppure essere un capitolo, un grafico, una immagine o una tabella del documento composto.

Il sistema deve tenere traccia degli utenti, cioè di coloro che possono accedere ai documenti registrati nel sistema, caratterizzati da nome utente, nome, cognome, dipartimento e ruolo.

Il sistema, inoltre, tiene traccia dei permessi assegnati a ciascun utente registrato, quali la lettura e/o la scrittura (specifici per ciascun documento) e la creazione di nuovi documenti.

Infine, il sistema deve tenere traccia dello stato corrente degli accessi del documento, in modo tale che un documento che è acceduto in scrittura, può essere soltanto acceduto in lettura. Un documento può essere acceduto in scrittura solo se nessun altro lo ha acceduto in scrittura. Un documento può essere acceduto in lettura anche da più utenti contemporaneamente.

Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 5) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli, evitando, per quanto possibile, la presenza di valori nulli.

Si consideri il seguente schema di base di dati relazionale.

AEROPORTI(Città, Nazione)
VOLI(Codice, Aereo, CittàPartenza, OraPartenza, CittàArrivo, OraArrivo, Compagnia)
AEREI(Tipo, Capienza, NumeriMotori)
COMPAGNIE(Codice, Nome, NazioneSede)
COMPAGNIEBANDITE(Compagnia, Nazione)

con i vincoli:

VOLI(CittàPartenza) $\subseteq$ AEROPORTI(Città)
VOLI(CittàArrivo) $\subseteq$ AEROPORTI(Città)
VOLI(Aereo) $\subseteq$ AEREI(Tipo)
VOLI(Compagnia) $\subseteq$ COMPAGNIE(Codice)
COMPAGNIEBANDITE(Compagnia) $\subseteq$ COMPAGNIE(Codice)

La relazione COMPAGNIEBANDITE mantiene l'informazione su quali compagnie non possono effettuare voli su aeroporti di una certa nazione.

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare codice e nome delle compagnie i cui voli in arrivo o in partenza da aeroporti francesi non possono essere autorizzati (perché la compagnia è bandita in Francia)".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il numero totale di voli che arrivano in ciascuna nazione, escludendo quelli che sono di compagnie bandite e quelli operati da aerei bimotori".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni tipo di aereo il numero di voli *internazionali* che opera, escludendo quelli di compagnie che hanno sede in una delle due nazioni collegate dal volo".

Esercizio 6 (punti 3) Spiegare brevemente il concetto di chiave e illustrare *tutti* i modi con cui le chiavi possono venir specificate nella creazione di una tabella in SQL.

Compito del 22 giugno 2010

Si vuole progettare una base dati per la gestione di un *video noleggio*. I film disponibili presso la struttura sono identificati dal titolo e dal nome del regista; inoltre sono noti l'anno in cui il film è stato girato e gli attori principali del film. Per ciascuna copia di film posseduta dal video noleggio è memorizzato il supporto (DVD, BluRay, VHS, ...), la data di acquisizione ed un intero progressivo che è diverso per ciascuna copia dello stesso film, indipendentemente dal supporto (gli stessi numeri di copia invece possono ripetersi per film diversi). È necessario memorizzare anche il costo giornaliero di noleggio, che è diverso

per i vari film e per i diversi supporti. È richiesto inoltre di memorizzare i clienti registrati presso la struttura, di cui interessa il numero di tessera, il nome e il cognome, il credito corrente, e la password per il prelievo dei film. Infine, bisogna memorizzare tutti i noleggi delle copie dei film effettuati dai clienti, con la data di prelievo e, se il film è già stato restituito, la data di restituzione e l'importo addebitato per il noleggio.

Esercizio 1 (punti 8) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Dato lo schema relazionale:
PERSONA(CF, Nome, Cognome, Genere, Età)
GENITORE(CodGen, CodFig)
ABITAZIONE(CodPersona, Via, NumCiv, CAP, Città)

In cui valgono i seguenti vincoli:
GENITORE(CodGen) $\subseteq$ PERSONA(CF)
GENITORE(CodFig) $\subseteq$ PERSONA(CF)
ABITAZIONE(CodPersona) $\subseteq$ PERSONA(CF)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i genitori i cui figli sono tutti conviventi con loro"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le persone di almeno 18 anni i cui genitori sono cugini tra loro (sono cugini due individui che hanno almeno un nonno/a in comune)".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e i cognomi delle coppie di persone che vivono in città diverse ed hanno avuto almeno 3 figli insieme".

Esercizio 6 (punti 5) Si consideri la seguente tabella e le successive assunzioni:

ESAME(MatrStudente, NomeStudente, CognomeStudente, DataEsame, VotoEsame,
NomeMateria, CodiceMateria, MatrDocente, NomeDocente, Domanda, VotoDomanda)

- All'esame di uno studente partecipano più docenti e ognuno fa allo studente una sua domanda.
- Un docente può fare esami per più materie.
- Uno studente può ripetere più volte l'esame di una materia, ma non nello stesso giorno.

Si scrivano le dipendenze funzionali e le chiavi della tabella. Si determini se la tabella è in BCNF, ed in caso contrario produrre, se possibile, una decomposizione in BCNF che rispetti le due proprietà fondamentali delle decomposizioni.

Compito del 13 luglio 2010

Si vuole progettare una base dati per la gestione del collaudo di uno strumento meccanico. Un collaudo si compone di una serie di prove con diverse configurazioni dei valori dei parametri dello strumento. Per la stessa configurazione, cioè per la stessa assegnazione di valori ai parametri, possono essere effettuate più prove distinte, che possono anche dare risultati diversi.

I parametri dello strumento sono di due tipi: qualitativi e quantitativi. Per i parametri qualitativi è memorizzato l'insieme dei valori possibili mentre per quelli quantitativi sono memorizzati i valori minimo e massimo. Per entrambi è presente un valore di *default*.

Per ciascuna prova interessa, oltre alla configurazione dei parametri con cui è stata ottenuta, il tempo di risposta dello strumento ed un valore di qualità del risultato, oltre che l'orario e la data di esecuzione. Le configurazioni sono caratterizzate da un valore progressivo.

Infine, le prove possono essere inserite in specifici esperimenti, caratterizzati da un nome ed un parametro di interesse. Una prova può anche far parte di più esperimenti.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per creare due delle tabella della base di dati dell'esercizio 2. Si scelgano due tabelle che abbiamo un vincolo di riferimento tra loro. Si includano anche i comandi per l'inserimento dei valori di due tuple, una per ciascuna tabella, legate dal vincolo di riferimento.

Si consideri il seguente schema di base di dati.

PERSONE(Nome, Cognome, Età)
SOCIETÀ(Nome, AnnoFondazione, CapitaleSociale, Sede)
PROPRIETÀ(NomePersona, CognomePersona, Società, PercentualePossesso)

con i vincoli:

PROPRIETÀ(NomePersona, CognomePersona) $\subseteq$ PERSONE(Nome, Cognome)
PROPRIETÀ(Società) $\subseteq$ SOCIETÀ(Nome)

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome delle persone che posseggono ad Udine al massimo una società".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che sono proprietarie almeno al 20% di tutte le società fondate nel 2009".

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e i cognomi delle coppie di persone che sono comproprietarie di almeno 5 società di `Udine`".

Compito del 3 settembre 2010

Si vuole progettare una base dati per la gestione di un convegno internazionale. Il convegno riceve gli articoli proposti per la presentazione dagli autori (ogni articolo può avere anche più autori). Ogni autore è caratterizzato da email, nome, cognome, affiliazione e nazione. Inoltre il convegno ha un comitato di programma, formato da esperti per i quali sono disponibili le stesse informazioni degli autori. I membri del comitato di programma possono anche essere autori di articoli. Ciascun articolo viene valutato da almeno tre membri del comitato. La valutazione consiste in un voto ed un commento per ciascuna voce (originalità, rilevanza, presentazione, ...) del modulo di valutazione ed un voto complessivo. Alcuni articoli vengono accettati per la presentazione, mentre altri sono respinti. Gli articoli accettati vengono presentati al convegno da uno dei sui autori. I partecipanti alla conferenza possono essere autori (anche se non presentano un articolo), membri del comitato di programma, oppure semplici ricercatori interessati al convegno. Per tutti i partecipanti interessano le informazioni sopra elencate per gli autori.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale, producendo il relativo schema Entità-Relazione, per l'applicazione descritta dalle suddette specifiche.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione dell'esercizio 1, producendo il relativo schema relazionale completo di vincoli.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per creare due delle tabelle della base di dati dell'esercizio 2. Si scelgano due tabelle che abbiamo un vincolo di riferimento tra loro. Si includano anche i comandi per l'inserimento dei valori di due tuple, una per ciascuna tabella, legate dal vincolo di riferimento.

Si consideri il seguente schema di base di dati.

PERSONE(Nome, Cognome, Indirizzo)
CAVALLI(Nome, Età, Razza, Sesso)
PROPRIETÀ(NomePersona, CognomePersona, Cavallo, PercentualePossesso)

con i vincoli:

PROPRIETÀ(NomePersona, CognomePersona) $\subseteq$ PERSONE(Nome, Cognome)
PROPRIETÀ(Cavallo) $\subseteq$ CAVALLI(Nome)

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome delle persone che posseggono uno ed un solo cavallo".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome delle persone che sono proprietarie almeno al 50% di tutti i cavalli maschi di razza `Andalusa`".

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e i cognomi delle coppie di persone che sono comproprietarie di almeno 4 cavalli di razza `Camargue`".

Compito del 15 settembre 2010

Un medico di base vuole progettare la base di dati per un'applicazione per la gestione dei suoi pazienti. Di ogni paziente interessa il numero di tessera sanitaria, il nome, il cognome, l'indirizzo, la data di nascita, i suoi numeri di telefono, ed eventualmente l-indirizzo email. Ciascuna visita viene memorizzata con la data e la durata. Inoltre, per ogni paziente interessano anche tutte le malattie che ha avuto, con la loro durata, ed il livello di febbre per ciascun giorno di malattia e la visita in cui la malattia è stata diagnosticata. Inoltre, in alcune visite il medico registra anche la pressione ed il numero di battiti del paziente verificando se sono nella norma.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per creare due delle tabelle della base di dati dell'esercizio 2. Si scelgano due tabelle che abbiamo un vincolo di riferimento tra loro. Si includano anche i comandi per l'inserimento dei valori di due tuple, una per ciascuna tabella, legate dal vincolo di riferimento.

Si consideri il seguente schema relazionale di basi di dati:

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

con i vincoli:

FARMACI(Produttore) $\subseteq$ PRODUTTORI(Codice)
FARMACI(PrincipioAttivo) $\subseteq$ SOSTANZE(ID)

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il nome di produttori italiani che producono almeno due farmaci il cui principio attivo è nella categoria `Analgesico` e che costano meno di 3€."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascun produttore il numero di farmaci prodotti che non contengono `Insulina` (mostrando anche codice e nome del produttore)."

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i produttori per i quali esiste un farmaco che ha una sostanza che non è presente in alcun farmaco di un altro produttore. Mostrare il nome del produttore, quello del farmaco e quello della sostanza."

Esercizio 7 (punti 2) Fornire uno schema Entità-Relazione da cui possa essere stato ricavato lo schema relazionale precedente.

Compito del 31 gennaio 2011

Si vuole progettare un'applicazione per la gestione di aeromobili. Di ogni compagnia aerea interessa il codice fiscale internazionale (identificativo per tutte le aziende a livello internazionale), l'anno di fondazione, il numero di dipendenti, e la nazione in cui è registrata come azienda. Di ogni aeromobile interessa la compagnia aerea che ne è proprietaria, il codice (unico nell'ambito della compagnia aerea che ne è proprietaria), il modello, l'anno di costruzione e il numero di miglia percorse. Un aeromobile si trova sempre in una delle seguenti tre condizioni: in volo, in sosta, in manutenzione. Di un aeromobile in volo interessa l'aeroporto di partenza e l'aeroporto di arrivo del volo. Di un aeromobile in sosta interessa l'aeroporto in cui esso sosta. Di un aeromobile in manutenzione interessa l'aeroporto in cui avviene la manutenzione, con l'indicazione della data in cui si prevede di terminare la manutenzione, se nota. Di ogni nazione interessa il nome (identificativo) ed il numero di abitanti. Di ogni aeroporto interessa il codice (identificativo) e la città (identificata da un codice) e la nazione in cui si trova. Di ogni modello di aeromobile interessa l'azienda che lo ha costruito, il codice (unico nell'ambito dell'azienda che lo ha costruito), il numero di posti previsti per tale modello e l'anno di ideazione del modello stesso. Infine, di ogni azienda interessa il codice fiscale internazionale (identificativo per tutte le aziende a livello internazionale), l'anno di fondazione, il numero di dipendenti, e la nazione in cui è registrata come azienda.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli. Non disponendo di informazioni quantitative sulle operazioni, seguire l'indicazione di evitare valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

SQUADRA(Nome, Città, Allenatore)
PARTITA(Giornata, SqCasa, SqTrasf, GolCasa, GolTrasf)

con i vincoli:

PARTITA(SqCasa) $\subseteq$ SQUADRA(Nome)
PARTITA(SqTrasf) $\subseteq$ SQUADRA(Nome)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le città dove la Sampdoria non ha mai perso in trasferta."

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Per ogni squadra s che ha giocato almeno 10 partite in casa, calcolare la media dei goal segnati da s nei *derby* (cioè partite tra squadre della stessa città) in cui s era la squadra di casa."

Esercizio 5 (punti 5) Si consideri la tabella CORSI(Bambino, Livello, Età) che descrive i corsi seguiti da un insieme di bambini con il livello di difficoltà del corso e l'età a cui li hanno seguiti.

Si descriva che cosa calcola l'interrogazione a sinistra e si calcoli il risultato ottenuto con la popolazione della tabella a destra.

```
select Bambino, Livello
from Corsi as S1
where Livello >=all (select Livello
                     from Corsi
                     where Bambino = S1.Bambino
                       and Età < 10)
```

Bambino	Livello	Età
Mario	1	8
Mario	2	9
Mario	4	12
Mario	5	14
Giulia	1	6
Giulia	3	7
Giulia	6	9
Paola	2	7
Paola	4	8
Paola	5	10

Esercizio 6 (punti 3) Spiegare brevemente il concetto di chiave e come le chiavi vengono specificate nella creazione di una tabella in SQL.

Compito del 21 febbraio 2011

Si intende progettare la base di dati di un'applicazione relativa alle edicole per la vendita di giornali. Di ogni edicola interessa il comune in cui essa è registrata, il codice, che è unico nell'ambito del comune, l'anno di inizio attività (non sempre, però, questa informazione è disponibile) e i contratti che l'edicola ha con i distributori per l'approvvigionamento dei quotidiani. Ogni contratto riguarda un'edicola, un quotidiano ed un distributore, ed e caratterizzato dal costo mensile a carico dell'edicola. Inoltre, per ogni edicola, interessa conoscere le varie persone che sono state proprietarie dell'edicola nei diversi anni, tenendo conto del fatto che in ogni anno un'edicola ha al massimo un proprietario. Di ogni persona interessa il codice fiscale, l'anno di nascita, il comune di nascita, ed il comune di residenza. Di ogni distributore di quotidiani interessa la partita IVA (identificativo), il fatturato ed il comune in cui è situata la direzione. Di ogni quotidiano interessa il nome (identificativo), e l'anno in cui ne è iniziata la pubblicazione. Di ogni comune interessa la provincia di appartenenza, il nome (unico nell'ambito della provincia di appartenenza), ed il numero di abitanti.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

NEGOZIO(Codice, Proprietario, Città)
VENDITA(Negozio, Merce, Data, Incasso)

con il vincolo:

$$\text{VENDITA}(\text{Negozio}) \subseteq \text{NEGOZIO}(\text{Codice})$$

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le merci che non sono state mai vendute a Udine"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le merci che nel 2008 e nel 2009 sono state vendute esclusivamente da negozi di un'unica città."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare la media degli incassi relativi alle vendite nei vari giorni e nelle varie città, ma solo per quei giorni e per quelle città le cui vendite hanno dato luogo ad un totale di incassi maggiore di 10.000 €."

Esercizio 6 (punti 4) Data la relazione $R(A, B, C, D, E, F)$ e le dipendenze funzionali $A \to D$, $C \to B$, $AC \to E$ e $B \to F$, determinare le chiavi di R e decomporre R in BCNF senza perdita e, se possibile, preservando le dipendenze. In caso non sia possibile preservare tutte le dipendenze, specificare quali non sono state preservate.

Compito del 23 giugno 2011

Si vuole progettare la base di dati di un'applicazione relativa ad una compagnia di consegne a domicilio.

La compagnia possiede un insieme di veicoli di cui interessa il codice, la targa, la capienza in volume e il peso che può sostenere. La compagnia ha un insieme di clienti, con nome, cognome, indirizzo e codice fiscale. Ciascuna consegna è caratterizzata dalla data di consegna, il volume, il peso complessivo, il numero di colli e il cliente a cui consegnare.

Le consegne partono tutte da un unico deposito della compagnia.

La compagnia inoltre tiene traccia delle consegne effettuate da ciascun veicolo in ciascun giorno, memorizzando anche il loro ordine di esecuzione.

Infine, la compagnia memorizza tutte le distanze (sia in km che in ore di percorrenza media) sia tra ciascun cliente e il deposito che tra ciascuna coppia di clienti.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

NEGOZIO(Codice, Proprietario, Città)
VENDITA(Negozio, Merce, Data, Incasso)

con il vincolo:

VENDITA(Negozio) $\subseteq$ NEGOZIO(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le merci che nel 2010 sono state vendute solo ad Udine"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le merci che nel 2008 hai registrato il massimo incasso totale a Trieste"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare per ciascun proprietario che possegga negozi in città diverse la somma degli incassi complessivi dei suoi negozi."

Esercizio 6 (punti 4) Data la relazione $R(A, B, C, D, E, F, G)$ e le dipendenze funzionali $D \to A$, $DF \to B$, $E \to CG$, $F \to E$ ed $E \to CG$, determinare le chiavi di R e decomporre R in BCNF senza perdita e, se possibile, preservando le dipendenze. In caso non sia possibile preservare tutte le dipendenze, specificare quali non sono state preservate.

Compito dell'11 luglio 2011

Si vuole progettare la base di dati di un'applicazione relativa ad un ospedale. Un ospedale ha un nome ed un insieme di reparti. Ciascun reparto ha un nome ed un insieme di stanze. Inoltre, per alcuni reparti (pediatria, geriatria, ...) è fissata un'età minima oppure un'età massima dei pazienti che possono essere ospitati nelle sue stanze. Le stanze hanno un numero progressivo interno al reparto e possono essere singole (con pagamento extra) o condivise. Le stanze condivise hanno una capienza in termini di letti presenti.

I pazienti hanno un nome, un genere (maschio o femmina), un'età, una data di ricovero ed una di dimissione, ed una o più patologie da trattare. I pazienti vengono assegnati alle stanze e lo stesso paziente può anche essere assegnato ad una stanza per alcuni giorni e in un'altra per altri giorni della sua degenza.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per creare due delle tabelle della base di dati dell'esercizio 2. Si scelgano due tabelle che abbiamo un vincolo di riferimento tra loro. Si includano anche i comandi per l'inserimento dei valori di due tuple, una per ciascuna tabella, legate dal vincolo di riferimento.

Si consideri il seguente schema relazionale di basi di dati:

CONVEGNI(Codice, Città, Anno)
PERSONE(CF, Nome, Cognome, Affiliazioni, Nazione)
LOCALITÀ(Città, Nazione)
PARTECIPAZIONI(Persona, Convegno, Ruolo)

con i vincoli:

CONVEGNI(Città) $\subseteq$ LOCALITÀ(Città)
PERSONE(Nazione) $\subseteq$ LOCALITÀ(Nazione)
PARTECIPAZIONI(Persona) $\subseteq$ PERSONE(CF)
PARTECIPAZIONI(Convegno) $\subseteq$ CONVEGNI(Codice)

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e cognome delle persone che non sono mai andate ad un convegno all'estero."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Per ogni convegno svoltosi nel 2009 non in Italia, trovare nome e cognome di tutti i *relatori invitati* (cioè partecipanti con questo ruolo) in ordine alfabetico."

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Per ogni persona (con nome, cognome e CF), trovare a quanti convegni ha partecipato nei vari anni, ma solo per quegli anni nei quali ha partecipato ad almeno 3 convegni nella sua nazione."

Compito del 6 settembre 2011

Si vuole progettare una base di dati contenente le informazioni relative alle piscine gestite dal comune di Udine, tenendo conto delle seguenti informazioni:

- Le piscine sono identificate univocamente attraverso il nome. Per ciascuna sono inoltre noti l'indirizzo, un numero di telefono e l'insegnante che è il responsabile.
- Presso le piscine sono organizzati dei corsi; lo stesso tipo di corso può essere svolto presso piscine diverse, eventualmente con modalità differenti. Ciascun corso è pertanto identificato dal nome

dell'attività svolta (Acquagym, Nuoto Sincronizzato, Corso per Gestanti, ...) e dal nome della piscina presso cui tale corso si svolge. Per ciascun corso, svolto presso una certa piscina, è noto il costo, il numero minimo e massimo di partecipanti, in quali giorni della settimana si svolge e in quali orari. Si ipotizzi che presso ciascuna piscina ogni corso sia svolto una sola volta al giorno, ma anche più volte durante la settimana.

- Il corpo insegnante lavora a rotazione presso le varie piscine. Per ciascun insegnante è noto il codice fiscale, che lo identifica, un nome, il numero di cellulare, se disponibile, e l'elenco delle qualifiche dell'insegnante (ad esempio istruttore di sub, istruttore di aerobica, ecc). All'interno della base dati si vuole tener traccia di tutti gli intervalli di tempo in cui un insegnante ha lavorato presso ciascuna piscina. Non si escluda che lo stesso insegnante possa aver lavorato presso una stessa piscina più volte.

- Le piscine possono essere frequentate o da persone che sono iscritte ai corsi, o secondo la modalità ad "ingresso singolo" per svolgere nuoto libero (si noti che sono registrate per l'ingresso singolo solo le persone che non hanno mai frequentato corsi). Tutte le persone che accedono alle piscine sono identificate attraverso il loro codice fiscale ed inoltre sono noti il nome, un indirizzo ed un numero di telefono.

- Le persone che sono iscritte ai corsi devono presentare un certificato medico. Pertanto, nel caso la persona sia iscritta ad un corso, la base di dati contiene l'informazione del medico che ha redatto il certificato, la data in cui la persona ha presentato il certificato, l'età della persona, e l'elenco dei corsi a cui è iscritta. Per le persone che hanno fatto solo ingressi sono noti solo la data in cui è stato effettuato l'ultimo ingresso e presso quale piscina.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Si scrivano i comandi SQL per creare due delle tabelle della base di dati dell'esercizio 2. Si scelgano due tabelle che abbiamo un vincolo di riferimento tra loro. Si includano anche i comandi per l'inserimento dei valori di due tuple, una per ciascuna tabella, legate dal vincolo di riferimento.

Si consideri il seguente schema relazionale di basi di dati:

FILM(Titolo, Regista, Anno, Nazionalità , Genere, Durata)
SALE(Cinema, Numero, Città, NumPosti)
PROIEZIONI(Film, Regista, Cinema, NumSala, Città, DataProiezione, OraProiezione, Incasso)

con i vincoli:

PROIEZIONI(Film, Regista) $\subseteq$ FILM(Titolo, Regista)
PROIEZIONI(Cinema, NumSala, Città) $\subseteq$ SALE(Cinema, Numero, Città)

Esercizio 4 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare titolo e regista dei film che non sono mai stati proiettati a Udine"

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare, per ogni film di fantascienza, il titolo, il regista, l'anno e l'incasso totale delle sue proiezioni nel 2010."

Esercizio 6 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i registi tali che nessun film da loro diretto è stato proiettato nello stesso giorno in due sale diverse della stessa città ."

Compito del 14 settembre 2011

Si vuole progettare una base di dati contenente le informazioni relative ai software installati su un insieme di computer di un'azienda.

Il sistema deve tener traccia dei pacchetti *disponibili* (che lo sono indifferentemente per tutti i computer dell'azienda). Per ciascun pacchetto disponibile è noto il nome ed una descrizione. Per ciascun pacchetto esistono diverse *versioni* disponibili, che sono identificate da una coppia di numeri: *major release* e *minor release* (ad esempio ver. 3.5). Oltre al numero, per ciascuna versione interessa la sua dimensione su disco e la data di rilascio. Più versioni possono essere disponibili per lo stesso pacchetto, ma non con la stessa data di rilascio.

Inoltre, tra i pacchetti esiste una relazione di dipendenza: se il pacchetto A dipende dal pacchetto B significa che A necessita della presenza di B. Inoltre, alcune specifiche versioni di un pacchetto possono avere delle dipendenze ulteriori verso altri pacchetti.

Le versioni disponibili dei pacchetti possono essere *installate* nei computer. Per ciascuna installazione di una versione di un pacchetto su un computer interessa la data in cui è stata effettuata ed eventuali note. Per ogni pacchetto può essere installata una sola versione su un computer.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

FILM(Titolo, Regista, Anno, Nazionalità , Genere, Durata)
SALE(Cinema, Numero, Città, NumPosti)
PROIEZIONI(Film, Regista, Cinema, NumSala, Città, DataProiezione, OraProiezione, Incasso)

con i vincoli:

PROIEZIONI(Film, Regista) $\subseteq$ FILM(Titolo, Regista)
PROIEZIONI(Cinema, NumSala, Città) $\subseteq$ SALE(Cinema, Numero, Città)

Esercizio 3 (punti 5) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare titolo e regista dei film più lunghi di tutti gli altri dello stesso genere".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il titolo, il regista e l'anno di ogni commedia straniera che sia stata proiettata almeno una volta ad Udine e a Trieste nel 2010."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare per ogni sala il numero di film francesi che sono vi sono stati proiettati almeno 5 volte nel 2009."

Esercizio 6 (punti 3) Ricavare il diagramma ER da cui è stato ottenuto lo schema relazionale precedente.

Compito del 30 gennaio 2012

Si vuole progettare una base di dati per la gestione delle informazioni su un campionato di calcio. Di ciascuna quadra interessano l'anno di fondazione, il nome del presidente, la data di nascita del presidente, il nome dell'allenatore e i colori sociali. Di ciascun giocatore della rosa interessa il nome, la data di nascita, il ruolo, il numero di maglia e la nazionalità.

Inoltre per ciascun giocatore è necessario memorizzare tutta la sua carriera pregressa. In particolare, per ogni annata sportiva interessa: il campionato, la squadra e la serie in cui ha militato; il numero di presenze e i gol fatti.

La formula del campionato è quella del doppio girone all'italiana, per cui ciascuna squadra incontra ciascun'altra squadra due volte (una in casa e una in trasferta). Di ogni partita giocata interessa il risultato, il numero di spettatori e la data.

Per ciascuna partita inoltre è necessario memorizzare i giocatori che sono scesi in campo per ciascuna squadra e le sostituzioni (chi è entrato al posto di chi). Si memorizzi anche chi ha segnato e gli eventuali cartellini gialli e rossi distribuiti ai giocatori nella partita. Per tutti questi eventi è necessario memorizzare anche il minuto e il tempo (primo o secondo) in cui è avvenuto.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

CICLISTI(Nome, Nazione, Età)
ISCRIZIONI(Ciclista, Gara, NazioneGara, Piazzamento)
GARE(Nome, Nazione, Lunghezza)

con i vincoli:

ISCRIZIONI(Ciclista) $\subseteq$ CICLISTI(Nome)
ISCRIZIONI(Gara, NazioneGara) $\subseteq$ GARE(Nome, Nazione)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare il nome dei ciclisti che provengono da una nazione in cui non si svolge alcuna gara".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome ed età dei ciclisti che hanno gareggiato solo nella loro nazione"

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi delle coppie di ciclisti connazionali tali che esiste almeno una gara tale che il primo ha partecipato e il secondo non ha partecipato"

Esercizio 6 (punti 3) Si definiscano brevemente i concetti di BCNF e 3NF e si mostrino esempi di schemi che sono o non sono in BCNF e/o 3NF.

Compito del 41 febbraio 2012

Si vuole progettare la base di dati per memorizzare i dati di un insieme di siti web.

- Un sito web è caratterizzato da un indirizzo web (URL) e da un nome.
- Un sito ha una pagina principale (anche detta homepage), il cui indirizzo corrisponde a quello del sito stesso. Oltre alla homepage, il sito può contenere altre pagine. Di ciascuna pagina si vuole conoscere il titolo e l'indirizzo.
- Di ogni pagina si vuole conoscere, inoltre, il nome del file fisico, la dimensione e la data di ultima modifica.
- Le pagine possono essere statiche (le pagine HTML), o dinamiche, tra le quali interessa il linguaggio con cui sono scritte (JSP, PHP, ASP, ...).
- Il sito ha una struttura gerarchica: ogni pagina si trova all'interno di una cartella che può contenere a sua volta altre cartelle.
- Ogni cartella è caratterizzata da un nome e da un indirizzo relativo, che indica la posizione della cartella nel disco a partire dalla cartella principale.

- Le pagine web possono contenere dei collegamenti ad altre pagine web e a file di altro tipo (immagini, file eseguibili, cartelle compresse e così via), memorizzati anch'essi all'interno delle cartelle del sito (con nome e tipo).

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Si consideri il seguente schema relazionale di basi di dati:

AGENZIE(Codice, Nome, Città)
VETTURE(Targa, Marca, Modello, AnnoImmatricolazione, Categoria)
CATEGORIE(Codice, Costo, QuotaAssicurativa)
NOLEGGI(AgRitiro, AgRiconsegna, TargaVettura, Cliente, DataRitiro, DataConsegna, Importo)
CLIENTI(CF, Nome, Cognome, Città)

con i vincoli:

VETTURE(Categoria) $\subseteq$ CATEGORIE(Codice)
NOLEGGI(AgRitiro) $\subseteq$ AGENZIE(Codice),
NOLEGGI(AgRiconsegna) $\subseteq$ AGENZIE(Codice)
NOLEGGI(TargaVettura) $\subseteq$ VETTURE(Targa),
NOLEGGI(Cliente) $\subseteq$ CLIENTI(CF)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i clienti che non hanno richiesto noleggi per cui la riconsegna della vettura è avvenuta in un'agenzia della loro città"

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare il nome dell'agenzia che, nel maggio 2010, ha noleggiato il maggior numero di vetture (si intenda che l'agenzia che realizza il noleggio corrisponda con quella presso la quale la vettura viene ritirata)."

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare nome, cognome e codice fiscale dei clienti che hanno noleggiato una vettura presso tutte le agenzie della città di Milano "

Esercizio 6 (punti 3) Definire il concetto di decomposizione senza perdita (*lossless*) ed enunciare le condizioni sufficienti affinché una decomposizione sia senza perdita. Mostrare un esempio originale di decomposizione con perdita.

Compito del 15 giugno 2012

Si vuole progettare la base di dati per memorizzare i dati relativi ad una dispensa di esercizi d'esame svolti per un insieme di corsi universitari. Ogni compito d'esame ha una data, il corso a cui si riferisce, i suoi crediti, un orario e una durata, ed è composto da un insieme di esercizi. Ogni esercizio ha un numero progressivo interno del compito, un punteggio e un tipo. Un esercizio comprende una traccia ed una soluzione. Entrambe sono dati di testo (stringhe) con possibile aggiunta di figure.

Per ogni figura inserita in un testo, interessa il nome del file, il suo formato e la riga in cui è inserita nel testo. La stessa figura potrebbe anche comparire in più testi. Gli esercizi possono anche contenere dei riferimenti ai libri del corso. Ciascun riferimento è caratterizzato dall'esercizio che lo contiene, il libro e la pagina a cui si riferisce. Per ogni esercizio ci può essere al massimo un riferimento ad uno specifico libro (ma più riferimenti in generale). Il libro è caratterizzato dal titolo, l'autore, l'editore, l'edizione e il corrispondente l'anno dell'edizione.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli, seguendo l'indicazione di evitare, quando possibile, valori nulli nella base di dati.

Esercizio 3 (punti 2) Scrivere la definizione in **SQL** di due tabelle legate tra loro da un vincolo di riferimento, possibilmente che leghi tra loro coppie di attributi in ciascuna tabella. Si scrivano anche le istruzione per l'inserimento di due tuple legate tra loro.

Si consideri il seguente schema relazionale di basi di dati:

SAGRE(Codice, Tema, Organizzatore, Data, Quartiere)
PERSONE(CF, Nome, Cognome, Quartiere)

con il vincolo:

SAGRE(Organizzatore) $\subseteq$ PERSONE(CF)

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i quartieri presenti nella base di dati in cui non si sono tenute sagre dal 2010 in poi".

Esercizio 5 (punti 5) Si chiama *nostrana* una sagra organizzata in un quartiere da una persona che vive nel quartiere stesso. Esprimere la seguente interrogazione in **SQL**: "Per ogni quartiere in cui si sono tenute almeno 10 sagre dal 2005, calcolare quante sono state complessivamente in tale quartiere le sagre nostrane".

Esercizio 6 (punti 5) Si discutano brevemente i tipi di *vincoli* del modello relazionale. Si consideri il seguente vincolo: "dal 2009 in poi, nel quartiere `Rizzi` non si possono tenere sagre nel mese di maggio"; si enunci a quale tipo appartiene e si illustri come si esprime in **SQL**.

Compito del 10 luglio 2012

Si vuole progettare la base di dati per memorizzare i dati relativi ai gruppi musicali. Di ogni gruppo interessa il codice, il nome, il genere musicale (rock, blues, ecc.) che lo caratterizza, i musicisti che vi hanno militato nei vari anni ed i concerti che ha tenuto. Di ogni musicista interessa il codice fiscale, il nome, la data e la città di nascita. Ogni musicista di un gruppo è o un cantante, o un batterista, o un chitarrista o un tastierista. Di ogni cantante interessa la città di residenza, di ogni batterista interessa il peso, di ogni chitarrista interessa le chitarre che ha utilizzato nei vari concerti (anche più chitarre in uno stesso concerto) e di ogni tastierista interessa il padre, ma solo se tastierista anch'egli. Di ogni città interessa la regione e il nome (unico nella regione). Di ogni città che è capoluogo di regione interessa anche il numero di abitanti. Di ogni concerto interessa la data, il luogo (ad es., palasport, teatro dell'opera) e la città in cui si è svolto, ed il numero di spettatori. Per i concerti svolti nei capoluoghi di regione interessa anche l'ammontare di denaro incassato. Di ogni chitarra interessa il codice, il modello e l'anno di fabbricazione.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema relazionale di basi di dati:

PAZIENTI(CF, Nome, DataRicovero, MedicoCurante)
MEDICI(CF, Nome)
MALATTIE(Codice, Denominazione, Terapia)
SOFFRE(Paziente, Malattia, DataInizio)
SPECIALISTI(Medico, Malattia)

con i vincoli:

Pazienti(Medico) $\subseteq$ Medici(CF)
Soffre(Paziente) $\subseteq$ Pazienti(CF)
Soffre(Malattia) $\subseteq$ Malattie(Codice)
Specialisti(Medico) $\subseteq$ Medici(CF)
Specialisti(Malattia) $\subseteq$ Malattie(Codice)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare nome e codice fiscale dei pazienti che sono stati ricoverati senza che alla data del ricovero soffrissero di alcuna malattia.".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e codice fiscale dei pazienti il cui medico curante è specialista di tutte le malattie di cui il paziente stesso soffre".

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e codice fiscale dei medici che sono specialista di almeno 4 malattie e che hanno in cura almeno 10 pazienti".

Esercizio 6 (punti 4) Si consideri la relazione R(A,B,C,D,E) con le dipendenze funzionali $A \to B$, $B \to DA$, $CE \to BD$. Si richiede di: (i) trovare tutte le chiavi; (ii) determinare se lo schema è in 3NF o in BCNF; (iii) trovare, se esiste, una decomposizione in BCNF senza perdita che preservi le dipendenze.

Compito del 3 settembre 2012

Esercizio 1 (punti 8) Si consideri il seguenti schema di base di dati relazionale, che rappresenta una realtà accademica, e i relativi vincoli di riferimento.

Dipendenti(CodiceFiscale, Cognome, Nome)
Professori(CodiceFiscale, Qualifica, Anzianità, Facoltà*)
Facoltà(Codice, Nome, Indirizzo)
CorsiDiStudio(Codice, Nome, Facoltà, Presidente)
Collaborazioni(CorsoDiStudio, Facoltà, Professore, Tipo)
Corsi(Codice, Docente, Semestre, Materia, CFU)
Materie(Sigla, Nome, SSD)
Studenti(Matricola, Nome, Cognome)
Esami(Studente, Corso, Data, Voto)

Professori(CodiceFiscale) $\subseteq$ Dipendenti(CodiceFiscale)
Professori(Facoltà) $\subseteq$ Facoltà(Codice)
CorsiDiStudio(Facoltà) $\subseteq$ Facoltà(Codice)
CorsiDiStudio(Presidente) $\subseteq$ Professori(CodiceFiscale)
Collaborazioni(CorsoDiStudio, Facoltà) $\subseteq$ CorsiDiStudio(Codice, Facoltà)
Collaborazioni(Professore) $\subseteq$ Professori(CodiceFiscale)
Corsi(Materia) $\subseteq$ Materie(Sigla)
Corsi(Docente) $\subseteq$ Professori(CodiceFiscale)
Esami(Studente) $\subseteq$ Studenti(Matricola)
Esami(Corso) $\subseteq$ Corso(Codice)

Si ricavi un possibile schema ER da cui tale schema relazionale è stato derivato.

Esercizio 2 (punti 3) Scrivere i comandi per la creazione delle tabelle Dipendenti e Professori e per l'eliminazione da esse delle tuple relative al prof. Mario Rossi (assumendo che questo non violi alcun vincolo di riferimento rispetto ad altre tabelle).

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare le matricole delle coppie di studenti per i quali uno dei due ha riportato un voto più alto in tutti gli esami superati da entrambi".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i corsi per il quale la media dei voti riportati è massima, mostrando il codice del corso e la media in questione".

Esercizio 5 (punti 4) Si consideri la base di dati relazionale composta dalle tabelle $R_1(A, \underline{B, C})$ e $R_2(\underline{D}, E, F)$ aventi rispettivamente cardinalità c_1 e c_2 (non è noto se c_1 sia maggiore, minore o uguale a c_2). Assumere che sia definito il vincolo $R_2(E, F) \subseteq R_1(B, C)$. Indicare la cardinalità minima e massima del risultato di ciascuna delle seguenti espressioni algebriche:

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

Esercizio 6 (punti 4) Si consideri la base di dati dell'esercizio 5. Scrivere l'interrogazione **SQL** corrispondente alla seguente espressione algebrica.

$$\pi_{ABB'}(\sigma_{B \geq B'}(R_1 \bowtie_{A=A'} \rho_{A'B'C' \leftarrow ABC}(R_1)))$$

Esercizio 7 (punti 3) Descrivere il concetto di *valore nullo*, i motivi che hanno portato alla sua introduzione e la sua gestione nelle basi di dati relazionali.

Compito del 26 gennaio 2013

Si vuole automatizzare il sistema di gestione degli animali in un bioparco. Il bioparco memorizza delle informazioni su ogni genere di animale quali il nome scientifico, una breve descrizione, i cibi che mangia e la loro modalità di cottura. Ogni esemplare di animale ospitato è caratterizzato da un codice, che è unico all'interno del suo genere di appartenenza. Per ogni esemplare si memorizzano inoltre la data di arrivo, 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 ad un determinato genere di animali. Ogni casa contiene un insieme di gabbie, ognuna contenente uno o più esemplari. Ogni casa ha un addetto che pulisce ciascuna gabbia in un determinato giorno della settimana. Di ogni addetto si vuole tenere memoria del codice fiscale, nome, cognome, indirizzo, data di nascita. Gli animali sono sottoposti periodicamente a controllo veterinario; in ogni controllo, contraddistinto da una data, un veterinario rileva il peso degli esemplari, diagnostica un'eventuale malattia e prescrive il tipo di dieta da seguire. E' necessario memorizzare anche i dati anagrafici del veterinario.

Esercizio 1 (punti 9) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati relativo ad una agenzia che affitta appartamenti.

Cliente (<u>CodiceFiscale</u>, Cognome, Nome, Residenza)
Appartamento (<u>Codice</u>, Indirizzo, NumEdificio, Locali, Metratura, Piano)
Palazzo (<u>Indirizzo, NumEdificio</u>, NumPiani, NomeAmministratore, CognomeAmministratore)
Affitto (<u>Cliente, Appartamento, DataInizio</u>, DataFine, Prezzo)

con i vincoli:
Appartamento(Indirizzo, NumEdificio) $\subseteq$ Palazzo(Indirizzo, NumEdificio)
Affitto(Cliente) $\subseteq$ Cliente(CodiceFiscale)
Affitto(Appartamento) $\subseteq$ Appartamento(Codice)

Esercizio 3 (punti 2) Scrivere in SQL i comandi per la creazione delle tabelle APPARTAMENTO e PALAZZO e per l'inserimento di due tuple (una per tabella), legate tra loro da un vincolo di riferimento.

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare i nomi e i cognomi dei clienti che hanno effettuato almeno due affitti di appartamenti con più di 2 locali nell'anno 2005".

Esercizio 5 (punti 5) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e i cognomi degli amministratori che hanno in gestione almeno 5 appartamenti più grandi di 100mq, ognuno dei quali è stato affittato almeno una volta."

Esercizio 6 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare i nomi e i cognomi dei clienti che hanno effettuato il maggior numero di affitti di appartamenti di 3 locali nell'anno 2004.".

Esercizio 7 (punti 2) Definire brevemente in concetto di *chiave* nel modello relazionale.

Compito del 27 febbraio 2013 **(soluzione a pagina 144)**

Si desidera automatizzare la gestione di un ospedale. La base di dati dell'applicazione dovrà memorizzare informazioni relative al ricovero dei pazienti nei reparti ospedalieri, ai trattamenti cui sono sottoposti i pazienti e alle date di accettazione e di dimissione. Di ogni paziente, vengono registrati il nome, l'indirizzo, il sesso, il numero di carta di identità, il numero della tessera sanitaria, il reparto ove è ricoverato e il letto occupato (reparto e letto possono cambiare durante il periodo di degenza). Di ogni reparto, vengono memorizzati il nome, la localizzazione, il nome del primario responsabile, il nome degli (eventuali) altri medici presenti, il numero delle stanze (ognuna con un numero interno al reparto) e il numero di letti presenti per stanza (numerati da 1 alla capienza della stanza). Si vuole inoltre tener traccia delle date di ricovero, di (eventuale) trasferimento da un reparto all'altro e di dimissione dei pazienti. Ogni paziente può essere sottoposto a più trattamenti durante il periodo di degenza ospedaliera. Di ogni trattamento, vengono conservate informazioni relative al nome, alla durata media e alle possibili reazioni del paziente.

Esercizio 1 (punti 10) Effettuare la progettazione concettuale dell'applicazione, producendo il relativo schema Entità-Relazione.

Esercizio 2 (punti 4) Effettuare la progettazione logica dell'applicazione, producendo il relativo schema relazionale, completo di vincoli.

Si consideri il seguente schema di base di dati relativo alle tesi di laurea, in cui l'attributo Correlatore ha valore nullo per le tesi che sono state seguite dal solo relatore.

TESI(Studente, Titolo, Data, Voto, Relatore, Correlatore*)
DOCENTI(Matricola, Cognome, Nome, Dipartimento)
STUDENTI(Matricola, Cognome, Nome)
DIPARTIMENTI(Sigla, Nome, Area)

con i vincoli:

TESI(Studente) $\subseteq$ STUDENTI(Matricola)
TESI(Relatore) $\subseteq$ DOCENTI(Matricola)
TESI(Correlatore) $\subseteq$ DOCENTI(Matricola)
DOCENTI(Dipartimento) $\subseteq$ DIPARTIMENTI(Sigla)

Esercizio 3 (punti 4) Esprimere la seguente interrogazione in **algebra relazionale**: "Trovare le aree in cui non sono stati mai dati voti di laurea inferiori al 78".

Esercizio 4 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome, cognome e matricola degli studenti che hanno avuto nel 2011 sia il relatore che, se esiste, il correlatore di un dipartimento di area ingegneristica."

Esercizio 5 (punti 4) Esprimere la seguente interrogazione in **SQL**: "Trovare nome e cognome dei docenti tali che gli studenti di cui sono sono stati relatori *unici* (cioè senza correlatore) hanno preso tutti più di 100 alla tesi.".

Esercizio 6 (punti 4) Si descriva brevemente cosa sono e a cosa servono le *decomposizioni*, se enuncino le loro due proprietà fondamentali e si mostrino esempi di decomposizione che godono e non godono di tali proprietà.

Soluzioni

Soluzione del compito del 12 gennaio 2000

Esercizio 1

1. $\{(1, a)\}$
2. $\{(10, b), (10, c), (5, d), (12, c), (13, b)\}$

I passaggi sono lasciati per esercizio allo studente.

Esercizio 2

La difficoltà di questo esercizio risiede nella necessità di considerare entrambe le combinazioni tra squadra di casa e squadra ospite. Il problema è stato risolto usando un `or` nella clausola `where`. Alternativamente, si poteva usare una `union` tra interrogazioni distinte.

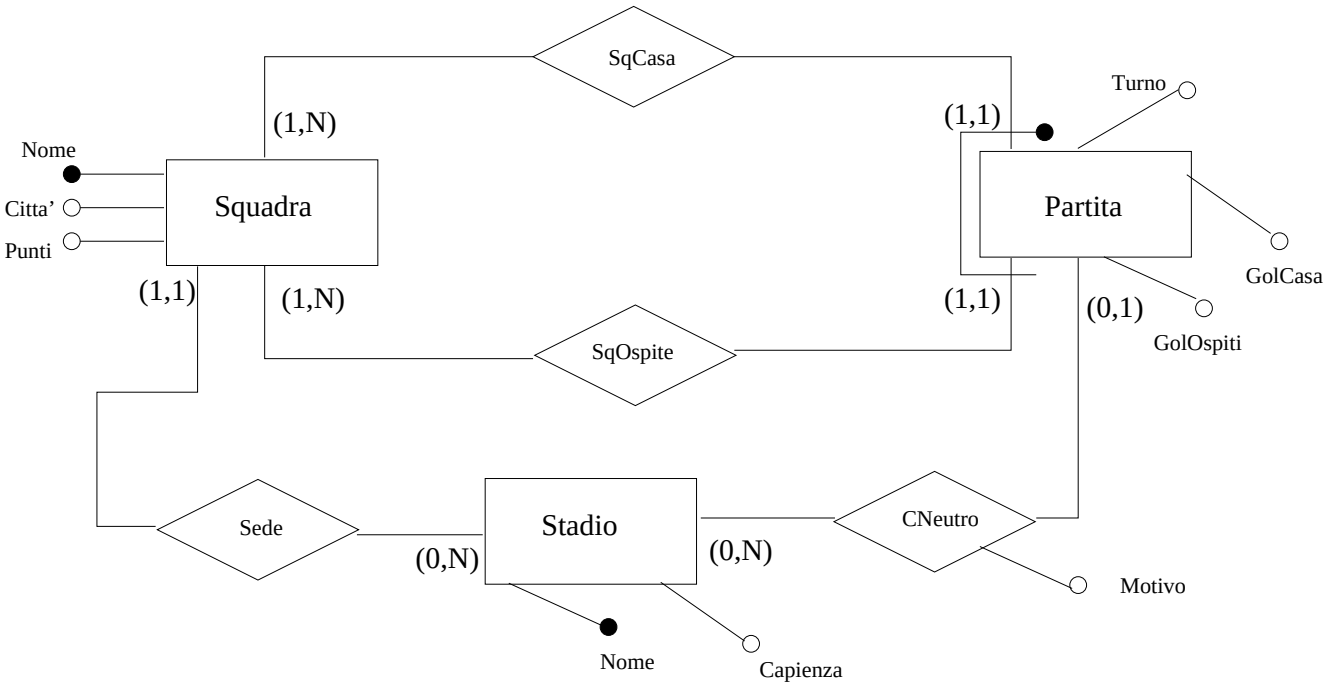
```
select count(*)
from Partita
     join Squadra as S1 on SquadraDiCasa = s1.Nome
     join Squadra as S2 on SquadraOspite = s2.Nome
where (SquadraDiCasa = 'udinese' and s2.Citta = 'Roma'
        and GoalCasa > GoalOspiti)
    or (SquadraOspite = 'udinese' and s1.Citta = 'Roma'
        and GoalOspiti > GoalCasa)
```

Esercizio 3

```
select Turno, avg(GoalCasa)
from Partita join Squadra on SquadraOspite = Nome
where Citta <> 'Milano'
group by Turno
having max(GoalCasa) < 5
```

Esercizio 4

Esercizio 5



Soluzione del compito del 26 gennaio 2000

Esercizio 1

Si assumano le seguenti abbreviazioni: D per Docente, C per Corso, N per Nome.

$$\pi_D(\textsc{Corso} \bowtie_{D=D1 \wedge C \neq C1} \rho_{C1,N1,D1 \leftarrow C,N,D}(\textsc{Corso}) \bowtie_{D=D2 \wedge C \neq C2 \wedge C1 \neq C2} \rho_{C2,N2,D2 \leftarrow C,N,D}(\textsc{Corso}))$$

Esercizio 2

```
select Docente, count(*)
from Corso
     join Lezione on Corso.Codice = CodCorso
     join Periodo on CodPeriodo = Periodo.Codice
where (Giorno = 'Lunedi' or Giorno = 'Martedi')
  and Aula = 'A'
  and Docente in (select Docente
                  from Corso
                  group by Docente
                  having count(*) = 2)
group by Docente
```

Esercizio 3

```
update periodo
   set OraInizio = '09:15'
 where OraInizio = '09:00'
   and giorno = 'Venerdi'

update periodo
   set OraInizio = '11:15'
 where OraInizio = '11:00'
```

```
   and giorno = 'Venerdi'
```

Esercizio 4

E1(A11,A12,A13,A31R5,A31R4*,AR4*)
E2(A21,A11,A12,A22)
E3(A31)
E4(A41,A31)
R2(A21,A11,A12,A41,A31)

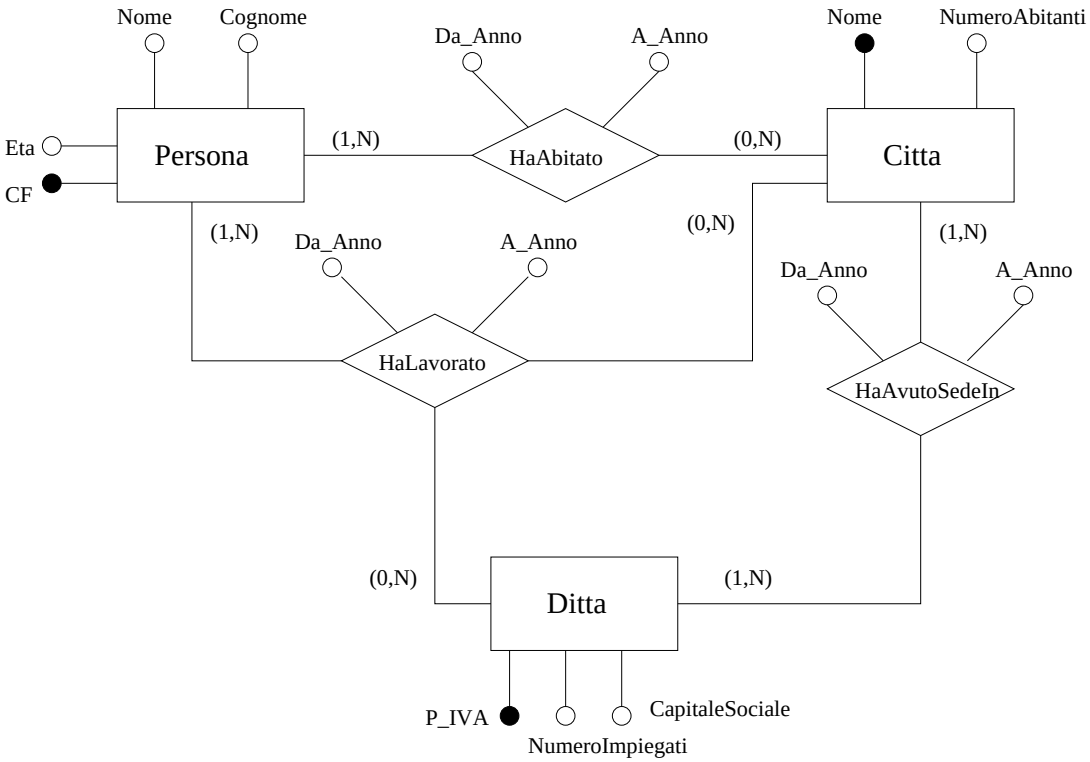
La tabella corrispondente all'entità E3 non è ridondante e quindi non può essere eliminata. Infatti in nessuna relazione che coinvolge E3 vi è cardinalità minima 1, e quindi nessuna relazione contiene sicuramente E3 stessa. Gli attributi AR4 e A31R4 della tabella E1 possono avere valori nulli.

Esercizio 5

Vedere sul testo del corso.

Soluzione del compito del 9 febbraio 2000

Esercizio 1



Esercizio 2

```
create table Persona
( CF char(16) primary key,
  Nome varchar(20),
  Cognome varchar(20),
  Eta integer
)

create table Ditta
```

```
( P_IVA char(11) primary key,
  NumeroImpiegati integer,
  CapitaleSociale integer
)

create table Citta
( Nome varchar(20) primary key,
  NumeroAbitanti integer
)

create table HaAbitato
( CFPersona char(16) references Persona(CF),
  NomeCitta varchar(20) references Citta(Nome),
  Da_Anno integer,
  A_Anno integer,
  primary key (CFPersona, NomeCitta)
)

create table HaAvutoSedeIn
( P_IVA_Ditta char(11) references Ditta(P_IVA),
  NomeCitta varchar(20) references Citta(Nome),
  Da_Anno integer,
  A_Anno integer,
  primary key (P_IVA_Ditta, NomeCitta)
)

create table HaLavorato
( CFPersona char(16) references Persona(CF),
  P_IVA_Ditta char(11) references Ditta(P_IVA),
  NomeCitta varchar(20) references Citta(Nome),
  Da_Anno integer,
  A_Anno integer,
  primary key (CFPersona, P_IVA_Ditta, NomeCitta)
)
```

Esercizio 3

Per questo esercizio forniamo due soluzioni alternative. La prima fa uso di una doppia negazione, la seconda invece utilizza gli operatori di aggregazione (`count` nel caso specifico).

```
/* soluzione 1: con doppia negazione, senza operatori di aggregazione */
select P_IVA
from Ditta
where not exists (select Nome
                  from Citta
                  where Nome not in (select nomeCitta
                                     from HaAvutoSedeIn
                                     where P_IVA = P_IVA_Ditta))

/* soluzione 2: utilizzando count, group by e having */

select P_IVA_Ditta
from HaAvutoSedeIn
group by P_IVA_Ditta
having count(distinct NomeCitta) = (select count(*)
                                    from Citta)
```

Esercizio 4

```
select HaLavorato.P_IVA_Ditta, count(distinct HaLavorato.CFPersona)
```

```
from HaLavorato
     join HaAvutoSedeIn on HaLavorato.P_IVA_Ditta = HaAvutoSedeIn.P_IVA_Ditta
     join HaAbitato on HaAbitato.CFPersona = HaLavorato.CFPersona
where HaAbitato.CFPersona = HaLavorato.CFPersona
  and ((HaAbitato.Da_Anno <= HaAvutoSedeIn.A_Anno
        and HaAbitato.A_Anno >= HaAvutoSedeIn.A_Anno)
     or
      (HaAvutoSedeIn.Da_Anno <= HaAbitato.A_Anno
       and HaAvutoSedeIn.A_Anno >= HaAbitato.A_Anno))
group by HaLavorato.P_IVA_Ditta
```

Soluzione del compito del 2 giugno 2000

Esercizio 1

Si abbrevia ciascun attributo della relazione CD con la sua iniziale:

$$\pi_C(CD) - \pi_C(CD \bowtie_{D<D1} (\rho_{C1,A1,T1,D1 \leftarrow C,A,T,D} CD))$$

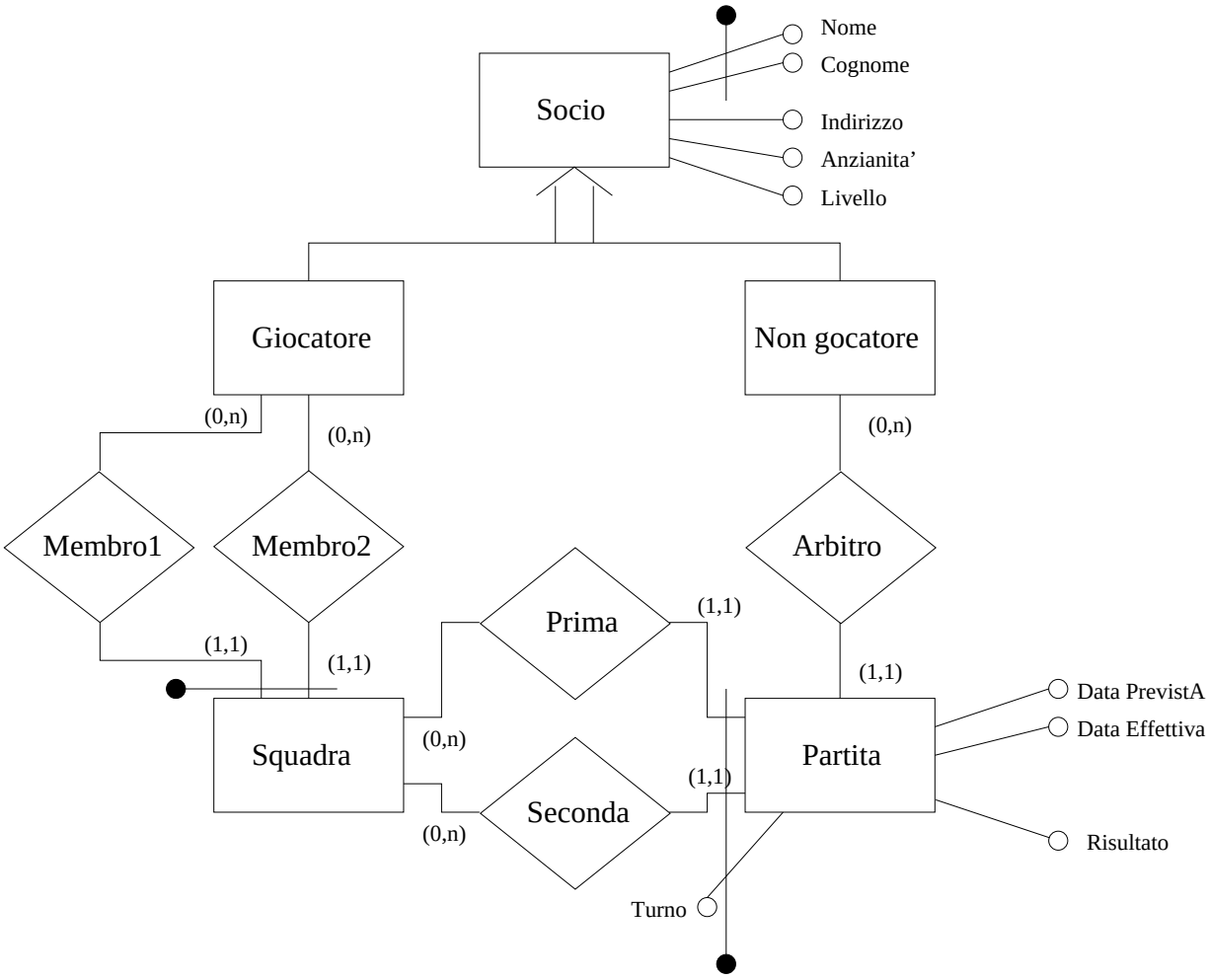
Esercizio 2

```
select distinct Autore, Nome
from CD
     join Affitto on CD.Codice = Affitto.CD
     join Cliente on Affitto.Cliente = Cliente.Codice
where not Restituito
```

Esercizio 3

```
select Autore, count(*)
from CD
group by Autore
```

Esercizio 4



Soluzione del compito del 17 luglio 2000

Esercizio 1

```
create table Impiegato
( Matricola integer primary key,
  Cognome varchar(20),
  Eta integer,
  Salario number
)

create table Dipartimento
( Codice integer primary key,
  Nome varchar(10),
  Budget number,
  MatricolaManager integer not null references Impiegato(Matricola),
)

create table Lavora
( Matricola integer references Impiegato(Matricola),
  Codice integer references Dipartimento(Codice),
  PercentualeTempo integer,
  primary key (Matricola,Codice)
```

)

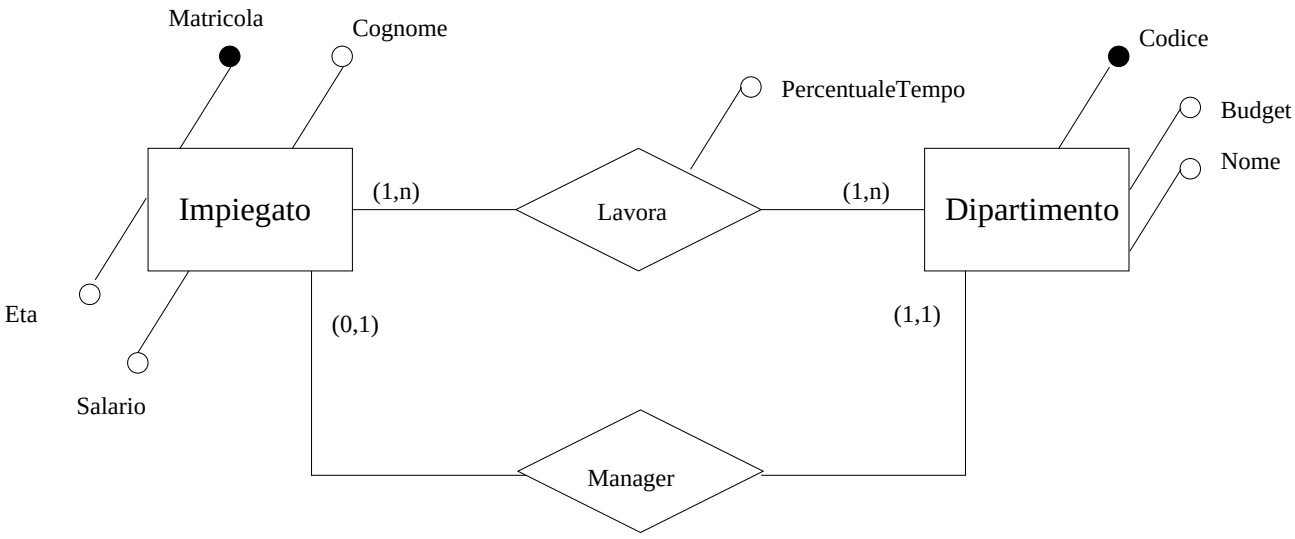
Esercizio 2

```
insert into Impiegato values (112,'Rossi',25,35000)
```

Esercizio 3

```
update Impiegato set salario = 1.2 * salario
```

Esercizio 4



Esercizio 5

```
select Matricola
from Dipartimento join Lavora on MatricolaManager = Matricola
where Lavora.Codice != Dipartimento.Codice
```

Esercizio 6

```
select Nome
from Dipartimento as D
where Budget < (select Sum(Salario)
                from Impiegato join Lavora on Impiegato.Matricola = Lavora.Matricola
                where Lavora.Codice = D.Codice)
```

Esercizio 7

```
select Cognome
from Impiegato join Lavora on Impiegato.Matricola = Lavora.Matricola
where eta <=all (select Eta
                 from Impiegato)
  and PercentualeTempo >=all (select PercentualeTempo
                              from Lavora)
```

Soluzione del compito del 6 settembre 2000

Esercizio 1

```
insert into Impiegato(Matricola,Cognome,Salario)
            values (112,'Verdi',35000)

insert into Lavora
       select 112, Codice, 70
       from Dipartimento
       where Nome = 'Vendite'
```

Esercizio 2

```
insert into Lavora
       select Matricola, Codice, 10
       from Impiegato join Dipartimento on true
       where Nome = 'Vendite'
         and (Eta = 25 or Eta is null)
```

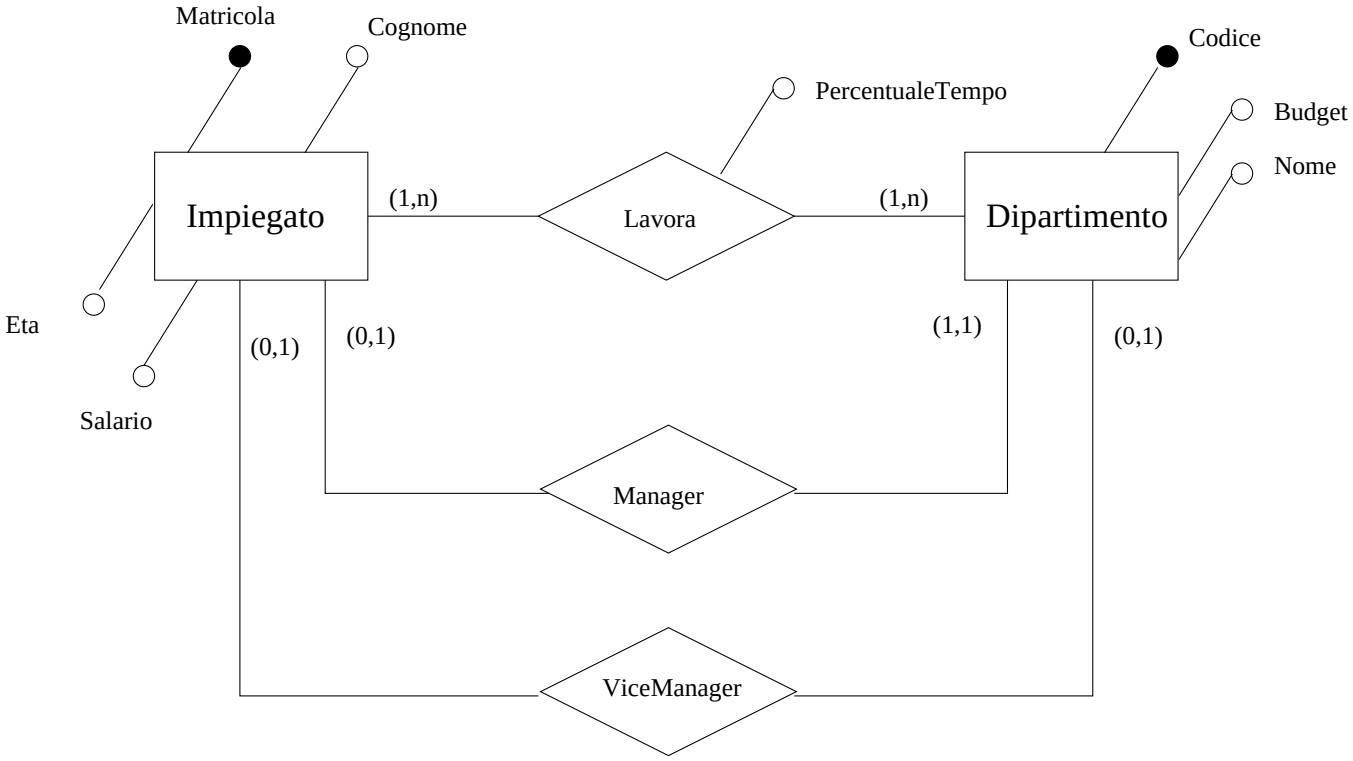
Esercizio 3

```
select distinct Matricola
from Lavora join Dipartimento on MatricolaViceManager = Matricola
  and Lavora.Codice != Dipartimento.Codice
```

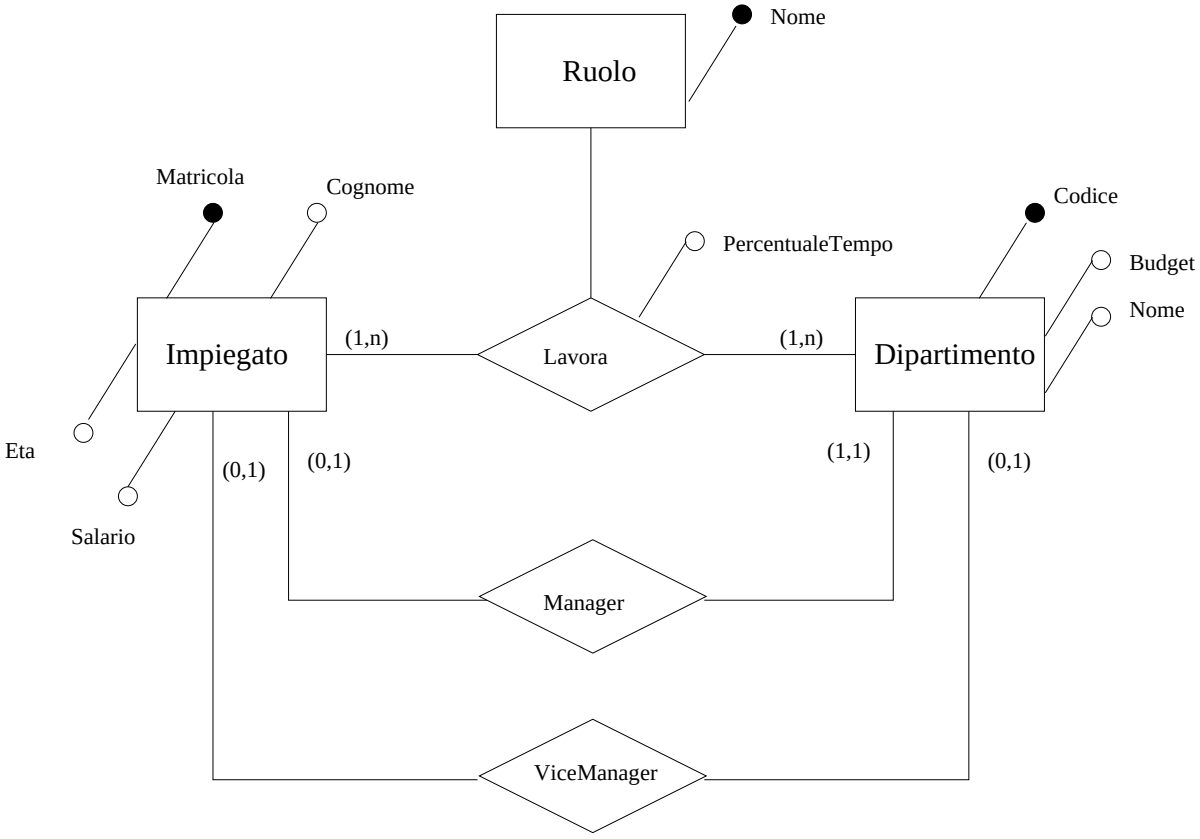
Esercizio 4

```
elect Nome
from Dipartimento as D
where Budget < (select Sum(Salario)
                from Impiegato join Lavora on Impiegato.Matricola = Lavora.Matricola
                where Lavora.Codice = D.Codice
                  and PercentualeTempo > 50)
```

Esercizio 5



Esercizio 6



Soluzione del compito del 8 gennaio 2001

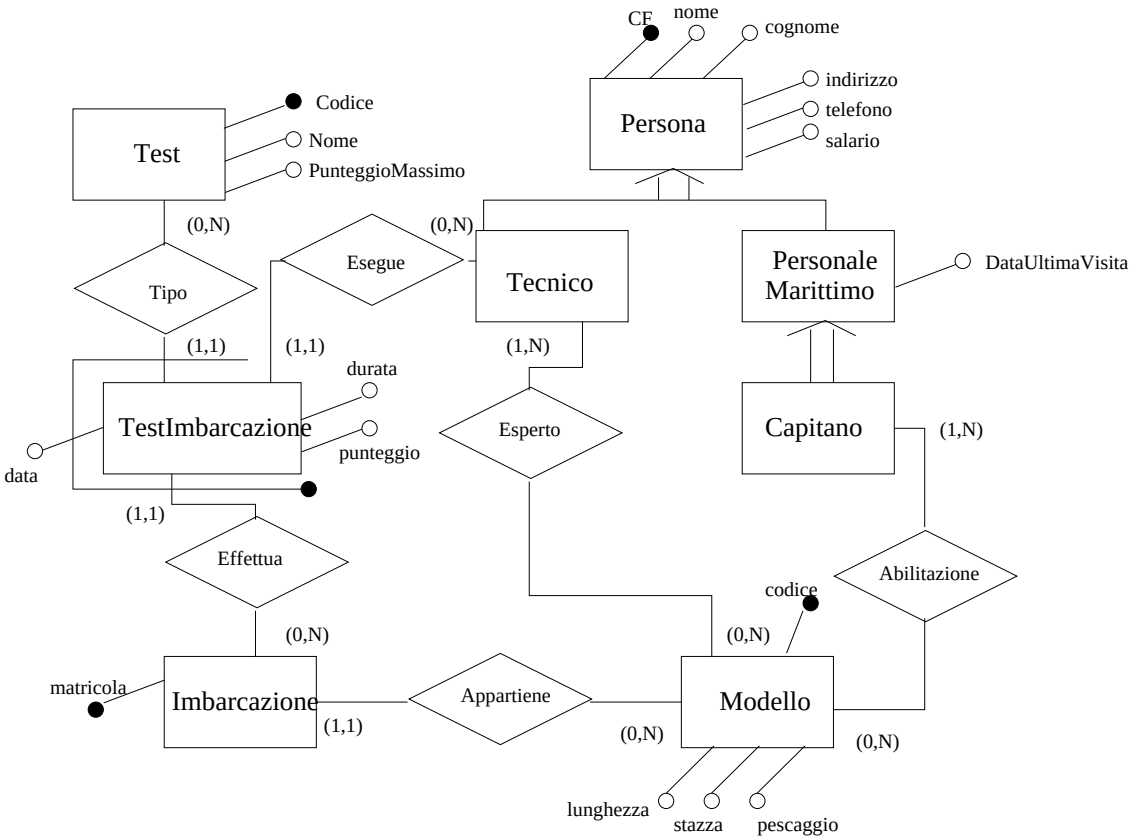
Esercizio 1

```
select distinct L1.Matricola
from Lavora as L1
     join Lavora as L2 on L1.Matricola = L2.Matricola
     join Dipartimento on MatricolaManager = L1.Matricola
where L1.Codice != L2.Codice
  and L1.Codice != Dipartimento.Codice
  and L2.Codice != Dipartimento.Codice
```

Esercizio 2

```
select Nome
from Dipartimento as D
where Budget < (select Sum(Salario)
                from Impiegato join Lavora on Impiegato.Matricola = Lavora.Matricola
                where Lavora.Codice = D.Codice
                 and Cognome like 'R%')
```

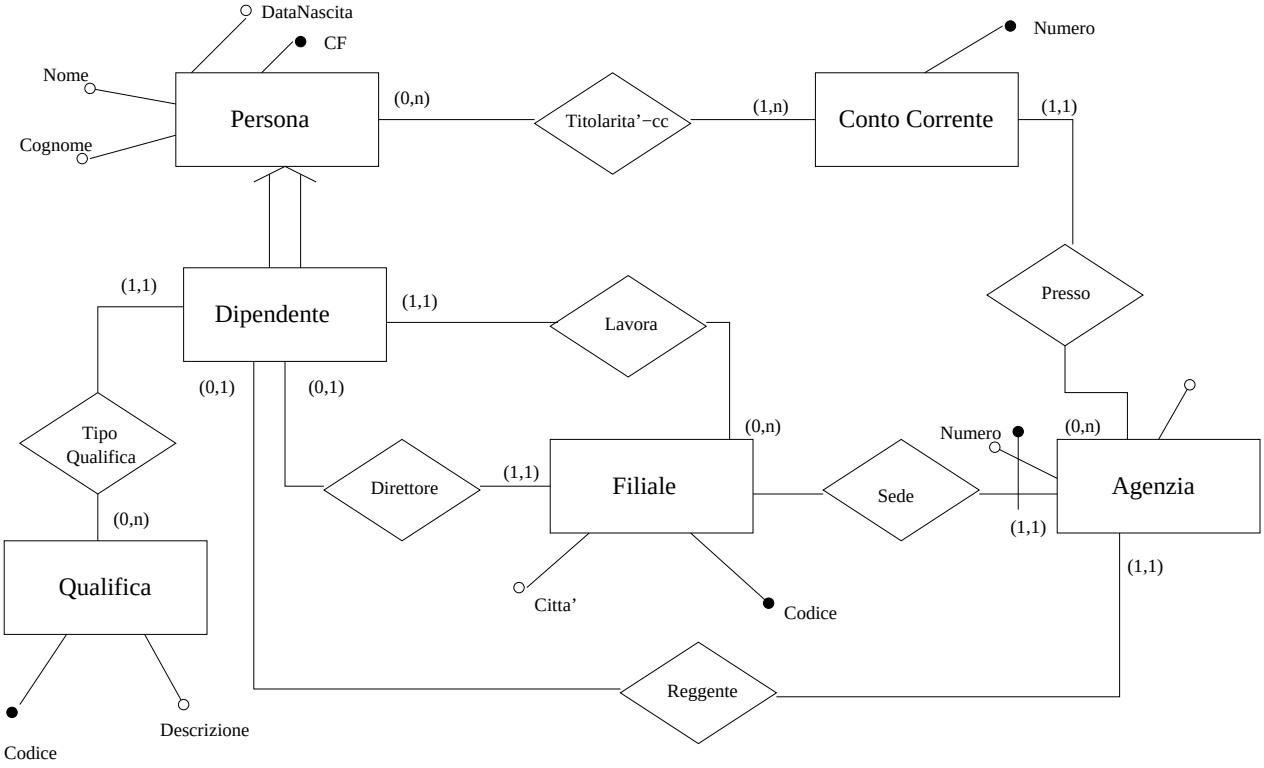
Esercizio 3



Nota: Si è assunto che un'imbarcazione non possa fare due test dello stesso tipo nella stessa data.

Soluzione del compito del 19 marzo 2001

Esercizio 1



Esercizio 2

$\pi_{numero,codice,citta,cognome}(agenzie \bowtie_{filiale=codice} filiali \bowtie_{reggente=codicefiscale} persone)$

Esercizio 3

Soluzione 1

```
select *
from ContiCorrenti
where Numero in (select Conto
                 from Titolarita-CC
                 group by Conto
                 having Conto(*) >= 2)
```

Soluzione 2

```
select Numero, Agenzia, Filiale
from ContiCorrenti,
     join Titolarita-CC as T1 on Numero = T1.Conto
     join Titolarita-CC as T2 on Numero = T2.Conto
where T1.Titolare <> T2.Titolare
```

Soluzione 3

```
select Numero, Agenzia, Filiale
from ContiCorrenti join Titolarita-CC on Numero = Conto
group by Numero, Agenzia, Filiale
having count(*) >= 2
```

Esercizio 4

```
select Numero, Filiale
from Agenzie
where (Numero, Filiale) not in (select Agenzia, Filiale
                                from ContiCorrenti)
```

Esercizio 5

1. $R1 \bowtie_{A=D} R2$ $\quad 0 \text{ — } min(c_1, c_2)$
2. $R1 \bowtie_{C=D} R2$ $\quad c_1 - c_1$
3. $R1 \bowtie_{A=F} R2$ $\quad 0 \text{ — } c_2$

Soluzione del compito del 26 marzo 2001

Esercizio 1

$$\pi_{Denominazione,Data,Voto}(Insegnamenti \bowtie_{Codice=Corso} Esami \bowtie_{Studente=Matricola} \sigma_{Cognome='Pestalozzi' \wedge Nome='Bartolomeo'}(Studenti))$$

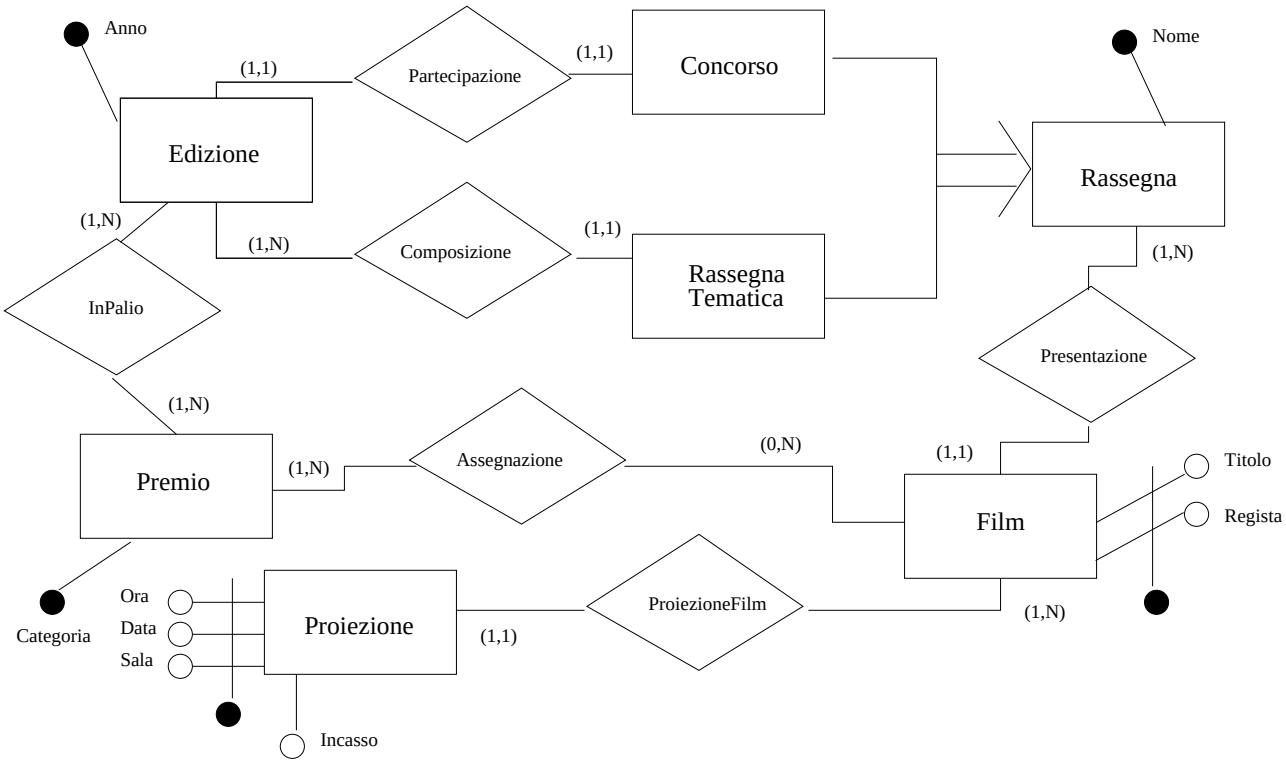
Esercizio 2

```
select Codice, Denominazione, avg(Voto)
from Insegnamenti join Esami on Codice = Corso
group by Codice, Denominazione
```

Esercizio 3

```
select Studente
from Esami
where Data > '01-Jan-2000'
group by Studente
having count(*) >= 2
```

Esercizio 4

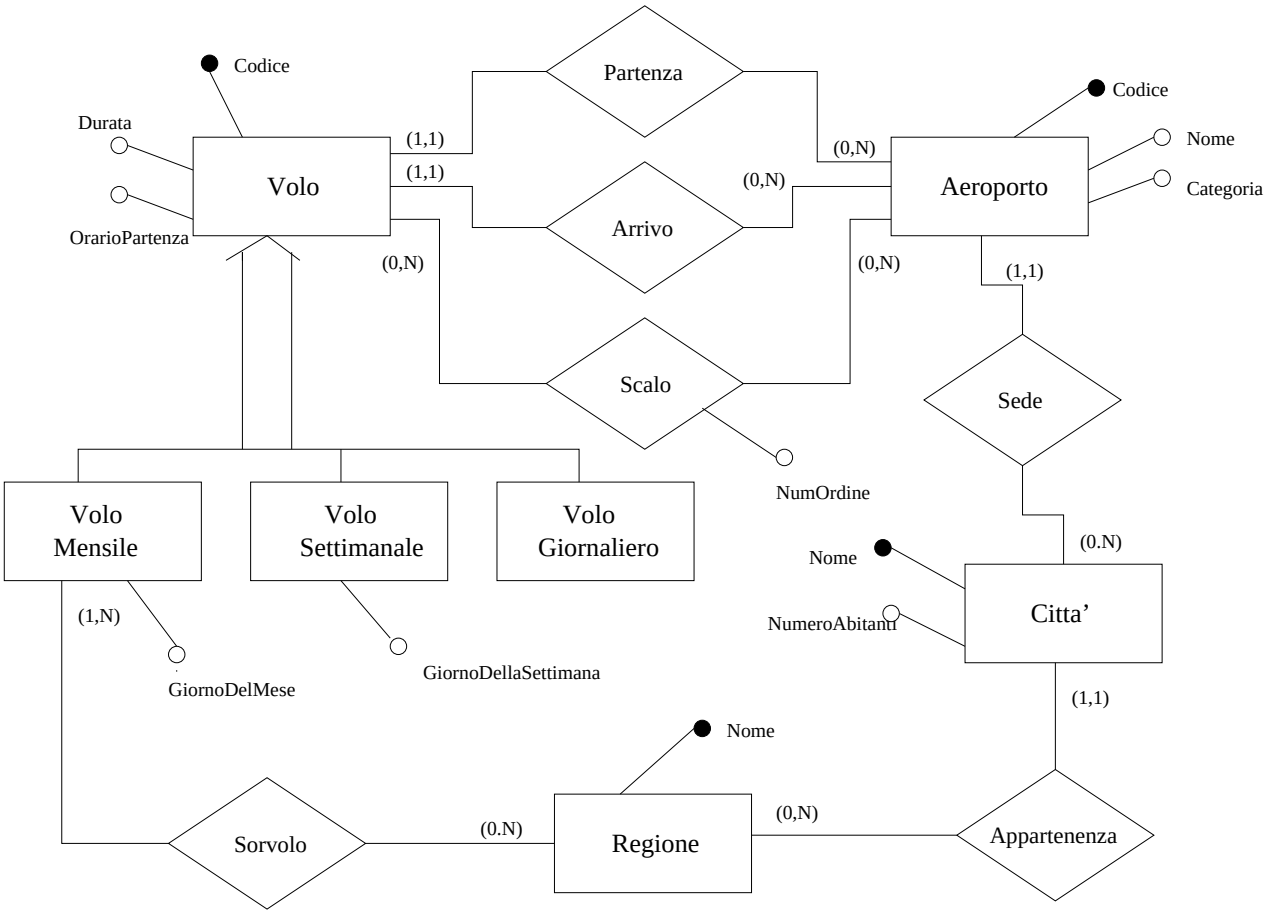


Esercizio 5

- $\pi_{ABCD}(R)$ Sì
- $\pi_{AC}(R)$ No
- $\pi_{BC}(R)$ Sì
- $\pi_{C}(R)$ No
- $\pi_{CD}(R)$ No

Soluzione del compito del 2 luglio 2001

Esercizio 1



Esercizio 2

Lo schema viene ristrutturato sostituendo alla gerarchia due relazioni tra le entità Volo e VoloMensile e tra le entitàVolo e VoloSettimanale. L'entità VoloGiornaliero viene accorpata all'entità Volo. Infine si aggiunge l'attributo Tipo all'entità Volo (lo schema è omesso).

La traduzione nel modello relazionale porta al seguente schema:

VOLO(Codice, Durata, OrarioPartenza, AeroportoPartenza, AeroportoArrivo, Tipo)
AEROPORTO(Codice, Nome, Categoria, Città)
CITTÀ(Nome, NumberoAbitanti, Regione)
VOLOSETTIMANALE(CodiceVolo, GiornoSettimana)
VOLOMENSILE(CodiceVolo, GiornoMese)
SCALO(Volo,Aeroporto,NumOrdine)
SORVOLO(Volo,Regione)

Assumendo che le regioni inserite nella base di dati siano solo quelle che riguardano e voli e le città, non è necessario inserire la relazione corrispondente all'entità Regione in quanto questa può essere ottenuta per unione delle proiezioni sull'attributo Regione delle tabelle CITTÀ e `Sorvolo`.

Esercizio 3

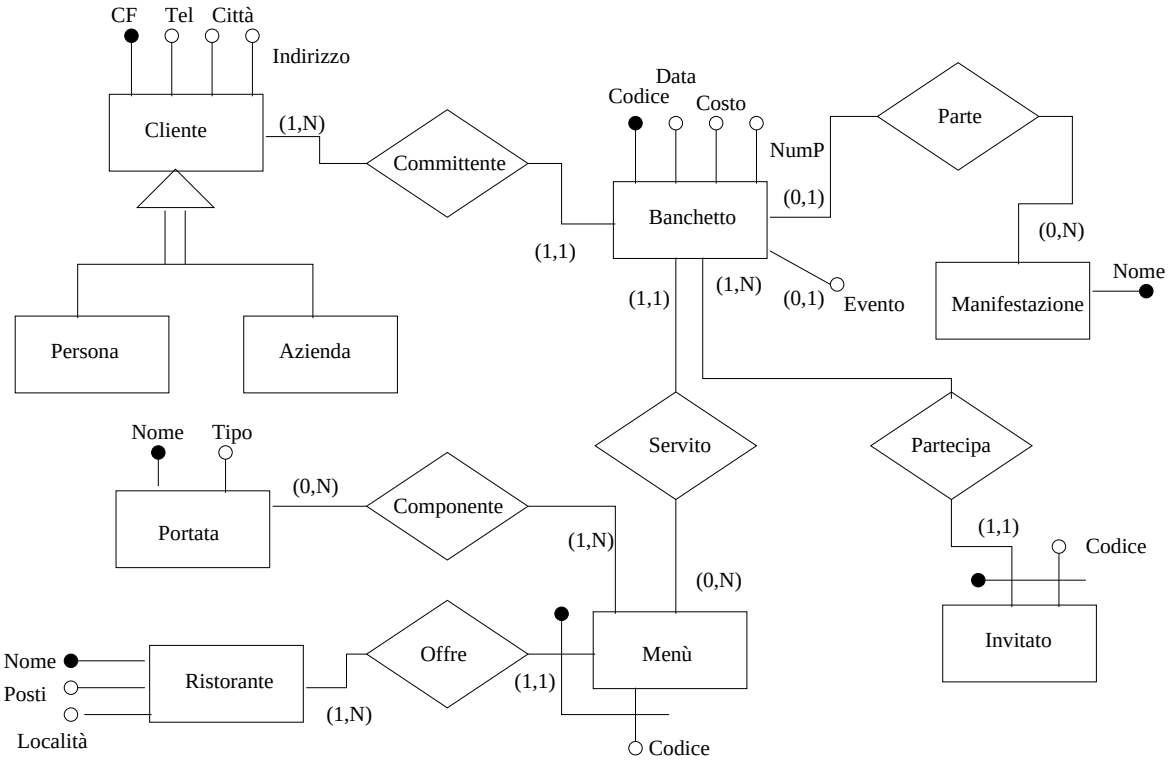
```
select Volo.Codice
from Volo join Aeroporto on AeroportoPartenza = Aeroporto.Codice
where Citta = 'Roma'
```

Esercizio 4

```
select Volo.Codice
from Volo
     join Aeroporto on Volo.AeroportoPartenza = Aeroporto.Codice
     join Citta on Aeroporto.Citta = Citta.Nome
     join Sorvolo on Volo.Codice = Sorvolo.Volo
where Citta.Regione != 'Veneto'
  and Sorvolo.Regione = 'Veneto'
```

Soluzione del compito del 9 luglio 2001

Esercizio 1



Si noti che una relazione diretta tra le entità Banchetto e Ristorante è concettualmente corretta, ma comunque ridondante in quanto il ristorante in cui si tiene un banchetto è identificato attraverso il suo menù.

Si noti che si è assunto, per semplicità, che il nome del ristorante sia sufficiente per identificarlo.

Esercizio 2

L'unica gerarchia presente viene eliminata accorpando le entità figlie nell'entità genitore, su cui viene aggiunto l'attributo Tipo. Altre ristrutturazioni dello schema ER non sono necessarie, quindi lo schema relazionale risultante è il seguente:

CLIENTE(<u>CF</u>, Telefono, Città, Indirizzo, Tipo)
BANCHETTO(<u>Codice</u>, Data, Costo, NumeroPartecipanti, Cliente,
Menù, Ristorante, Manifestazione*, Evento*)
MANIFESTAZIONE(<u>Nome</u>)
RISTORANTE(<u>Nome</u>, Località, Posti)
MENÙ(<u>Codice, Ristorante</u>)

PORTATA(Nome, Tipo)
INVITATO(Codice, Banchetto)
COMPOSIZIONEMENÙ(Menù, Ristorante, Portata)

Esercizio 3

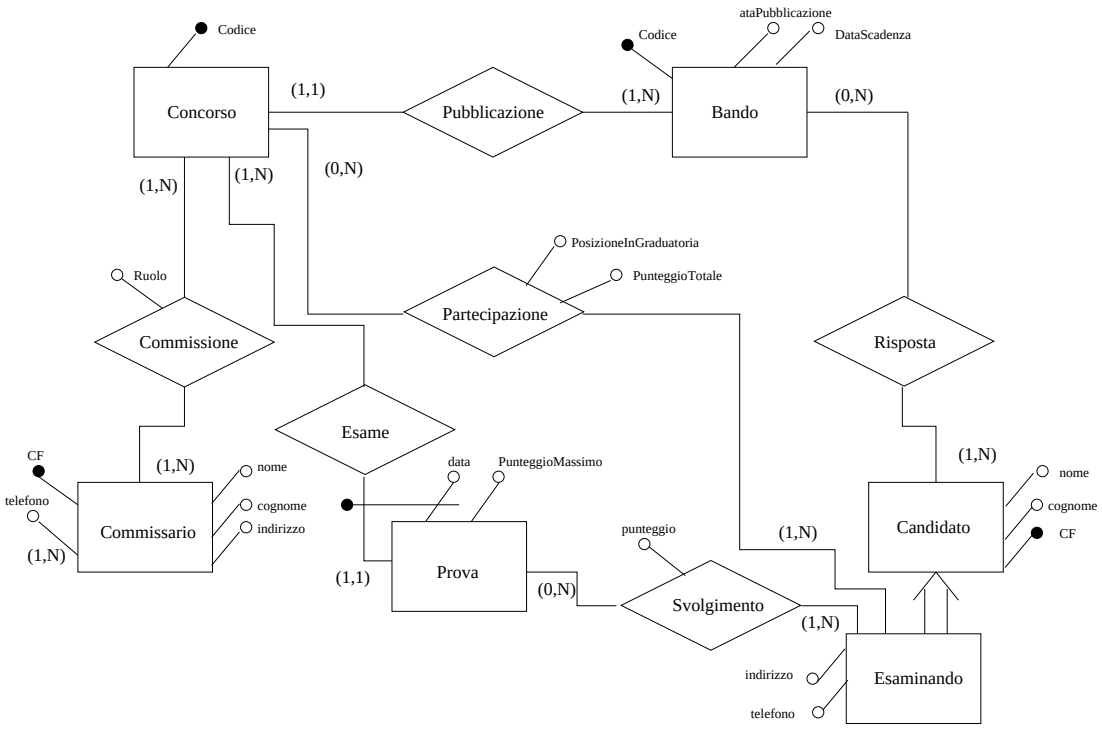
```
select Invitato.Codice, Invitato.Banchetto
from Invitato join Banchetto on Invitato.Banchetto = Banchetto.Codice
where Ristorante = 'DaMario'
```

Esercizio 4

```
select Banchetto.Codice
from Banchetto join Ristorante on Banchetto.Ristorante = Ristorante.Nome
where Banchetto.NumeroPartecipanti = Posti
```

Soluzione del compito del 23 luglio 2001

Esercizio 1



Si noti che:

- Si è distinto tra il candidato che risponde al bando e l'esaminando che si presenta alle prove.
- La relazione Risposta è ridondante in quanto si può risalire ai bandi a cui un candidati ha risposto dai concorsi a cui partecipa che siano pubblicati in quel bando
- Gli attributi PosizioneGraduatoria e PunteggioTotale sono ridondanti in quanto possono essere generati generata attraverso i punteggi ottenuti dai candidati nelle varie prove.

Esercizio 2

Per eliminare la gerarchia abbiamo deciso di accorpare le entità Esaminando e Candidato, introducendo quindi dei valori nulli negli attributi Indirizzo e Telefono.

Mancando i dati quantitativi non ci è possibile fare un'analisi accurata delle ridondanze. Abbiamo scelto di tradurre fedelmente lo schema ER mantenendo le ridondanze, rendendo più semplice la soluzione dell'esercizio 4. Lo schema relazionale risultante è il seguente.

CONCORSO(Codice, Bando)
BANDO(Codice, DataPubblicazione, DataScadenza)
CANDIDATO(CF, Cognome, Nome, Indirizzo*, Telefono*)
COMMISSARIO(CF, Cognome, Nome)
TELEFONOCOMMISSARIO(Commissario, NumeroTelefono)
PROVA(Concorso, Data, PunteggioMassimo)
SVOLGIMENTO(DataProva, ConcorsoProva, Candidato, Punteggio)
PARTECIPAZIONE(Concorso, Candidato, PunteggioTotale, PosizioneInGraduatoria)
RISPOSTA(Bando, Candidato)
COMMISSIONE(Concorso, Commissario, Ruolo)

Esercizio 3

```
select CF, Nome, Cognome
from Commissario
     join Commissione on Commissario.CF = Commissione.Commissario
     join Concorso on Commissione.Concorso = Concorso.Codice
     join Bando on Concorso.Bando = Bando.Codice
where Commissione.Ruolo = 'Presidente'
  and DataPubblicazione > '22-may-2001'
```

Esercizio 4

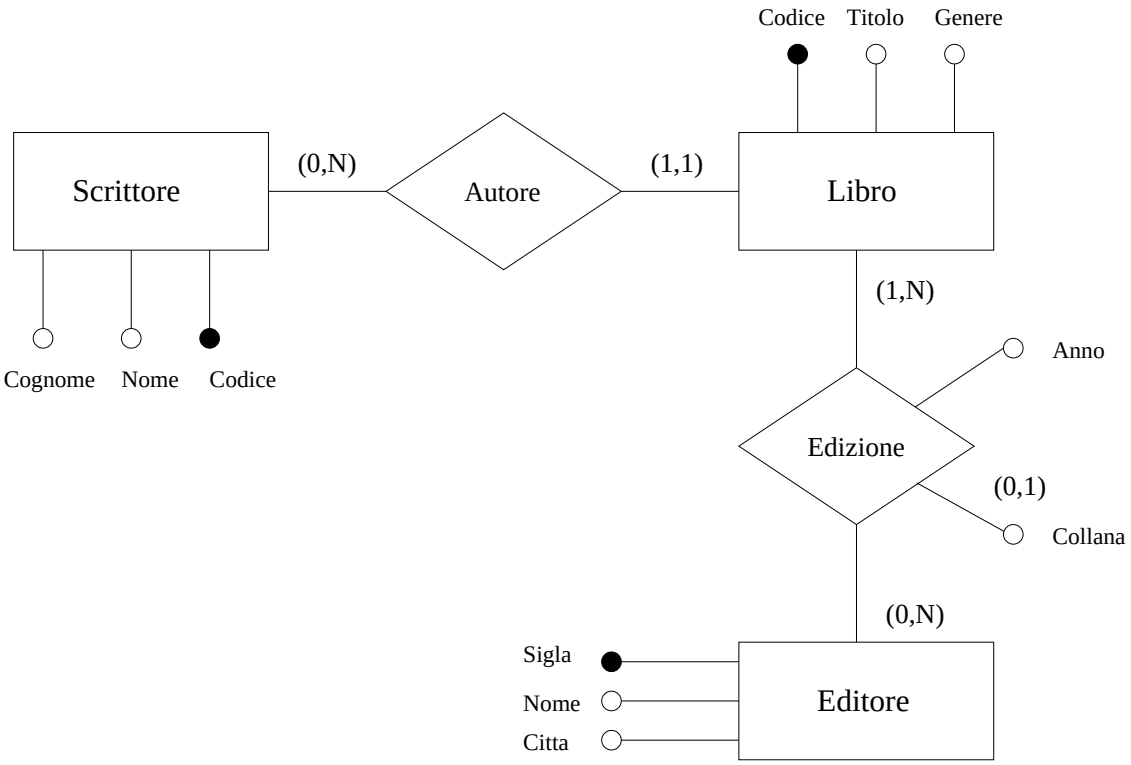
Assumiamo che il concorso cercato abbia codice `X01`.

```
select CF, Nome, Cognome
from Candidato join Partecipazione on CF = Candidato
where PosizioneInGraduatoria = 2
  and Concorso = 'X01'
```

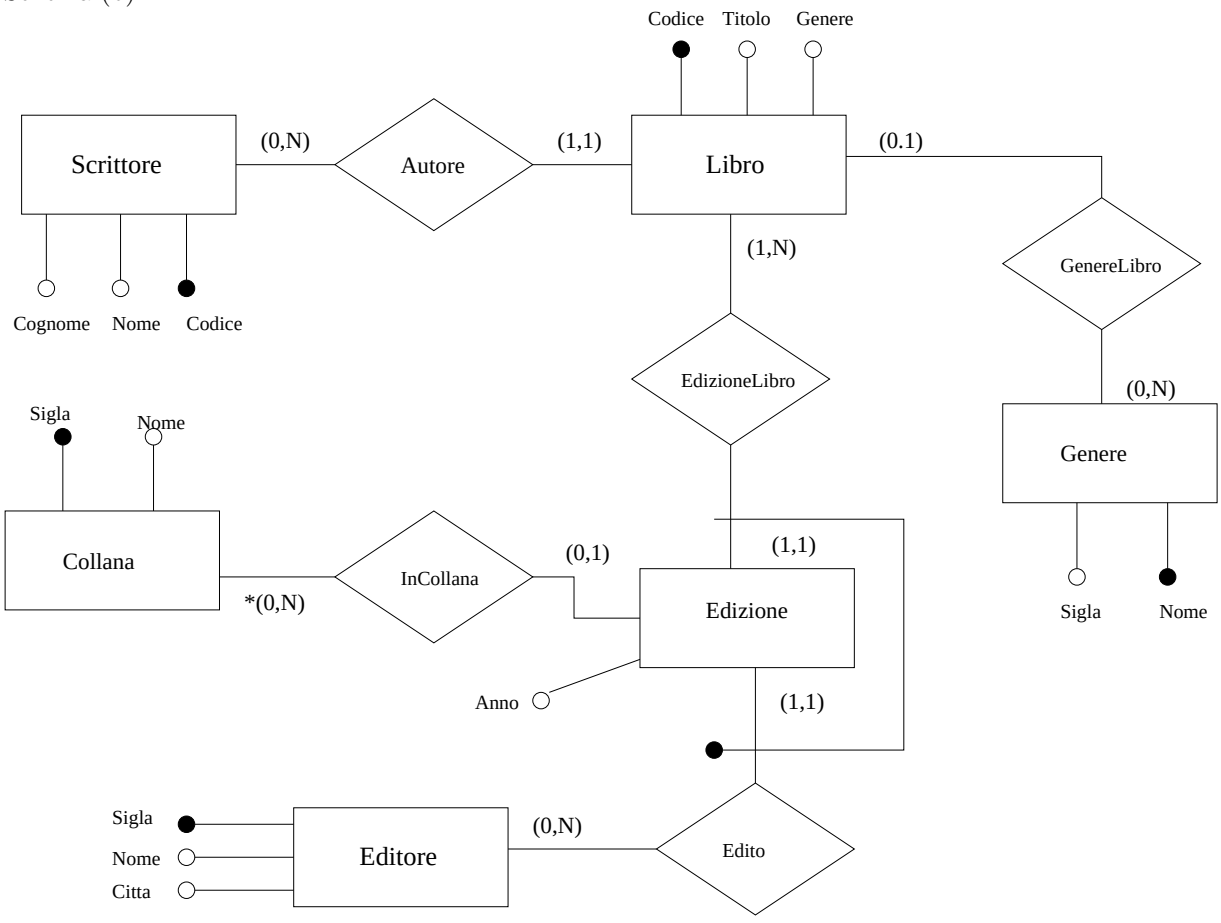
Soluzione del compito del 6 settembre 2001

Esercizio 1

Schema (a)


Codice
Titolo
Genere
Scrittore
(0,N)
Autore
(1,1)
Libro
(1,N)
Cognome
Nome
Codice
Anno
Edizione
(0,1)
Collana
(0,N)
Sigla
Nome
Citta
Editore

Schema (b)


Codice
Titolo
Genere
Scrittore
(0,N)
Autore
(1,1)
Libro
(0.1)
(1,N)
GenereLibro
Cognome
Nome
Codice
EdizioneLibro
(0,N)
Sigla
Nome
Genere
Collana
InCollana
(0,1)
(1,1)
*(0,N)
Edizione
Sigla
Nome
Anno
(1,1)
Sigla
Nome
Citta
Editore
(0,N)
Edito

Esercizio 2

$\pi_{Cognome,Nome}(\rho_{CS \to Codice} Scrittori \bowtie_{CS=Autore} Libri \bowtie_{Codice=Libro} Edizioni \bowtie_{Editore=Sigla} \sigma_{Citta='Milano'} Editori)$

```
select distinct Cognome, Nome
from Scrittori
     join Libri on Scrittori.Codice = Autore
     join Edizioni on Libri.Codice = Libro
     join Editori on Editore = Sigla
where Citta = 'Milano'
```

Esercizio 3

```
select distinct Autore, Editore
from Libri as L1
     join Edizioni on Codice = Libro
     join Editori as E1 on Editore = Sigla
where not exists (select *
                  from Libri
                       join Edizioni on Codice = Libro
                       join Editori on Editore = Sigla
                  where Autore = L1.Autore
                    and Sigla != E1.Sigla)
```

Soluzione del compito del 20 settembre 2001

Esercizio 1

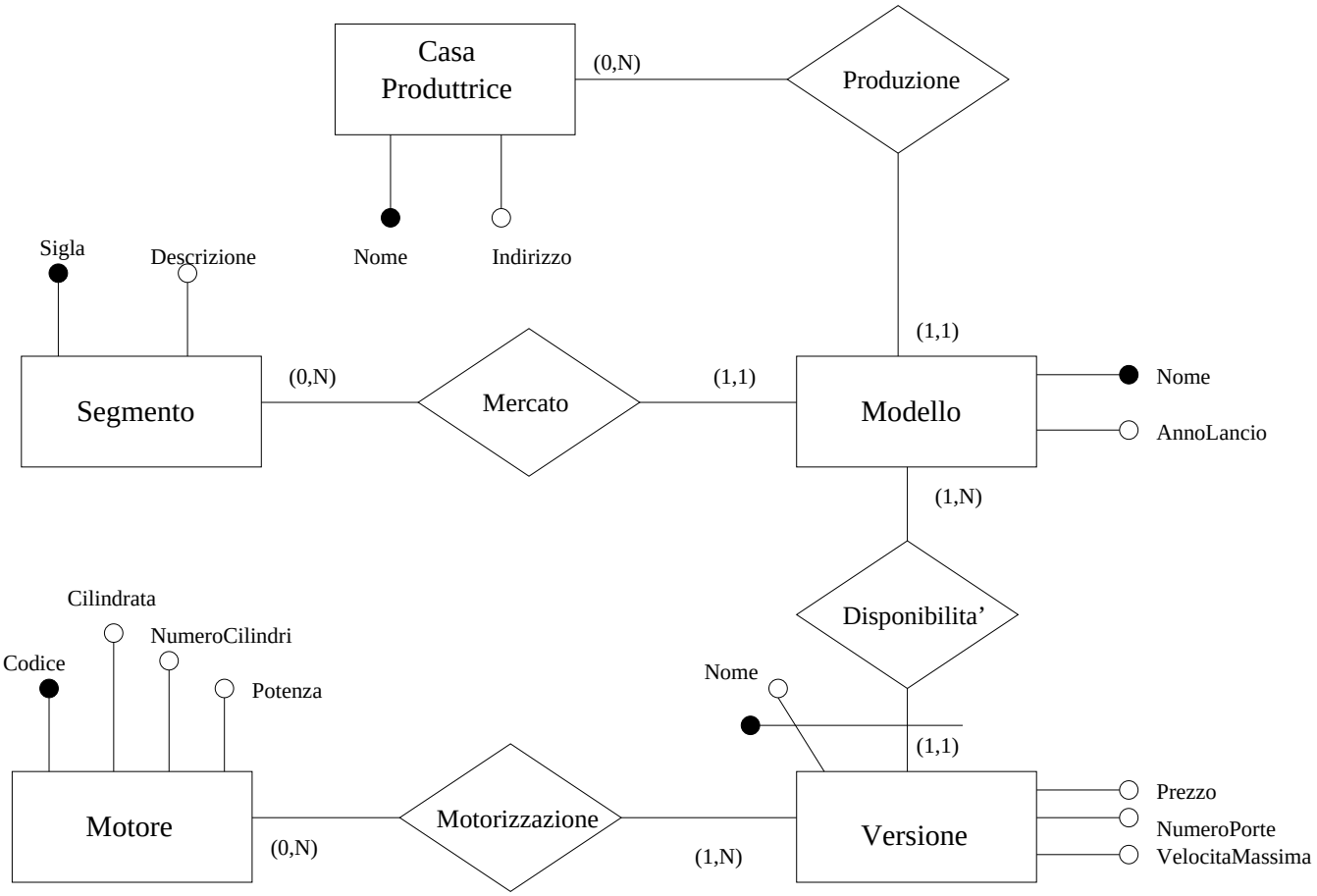
"trovare matricola, cognome e nome degli studenti che hanno preso almeno un trenta"

```
select distinct Matricola, Cognome, Nome
from Studenti join Esami on Matricola = Studente
where Voto = 30
```

Esercizio 2

```
select Matricola, Cognome, avg(Voto)
from Studenti join Esami on Matricola = Studente
group by Matricola, Cognome
```

Esercizio 3



Esercizio 4

CasaProduttrice(Nome, Indirizzo)
Modello(Nome, AnnoLancio, CasaProduttrice, Segmento)
Segmento(Sigla, Descrizione)
Versione(NomeVersione, NomeModello, Prezzo, NumeroPorte, VelocitàMassima)
Motore(Codice, Cilindrata, NumeroCilindri, Potenza)
Motorizzazione(Versione, Modello, Motore)

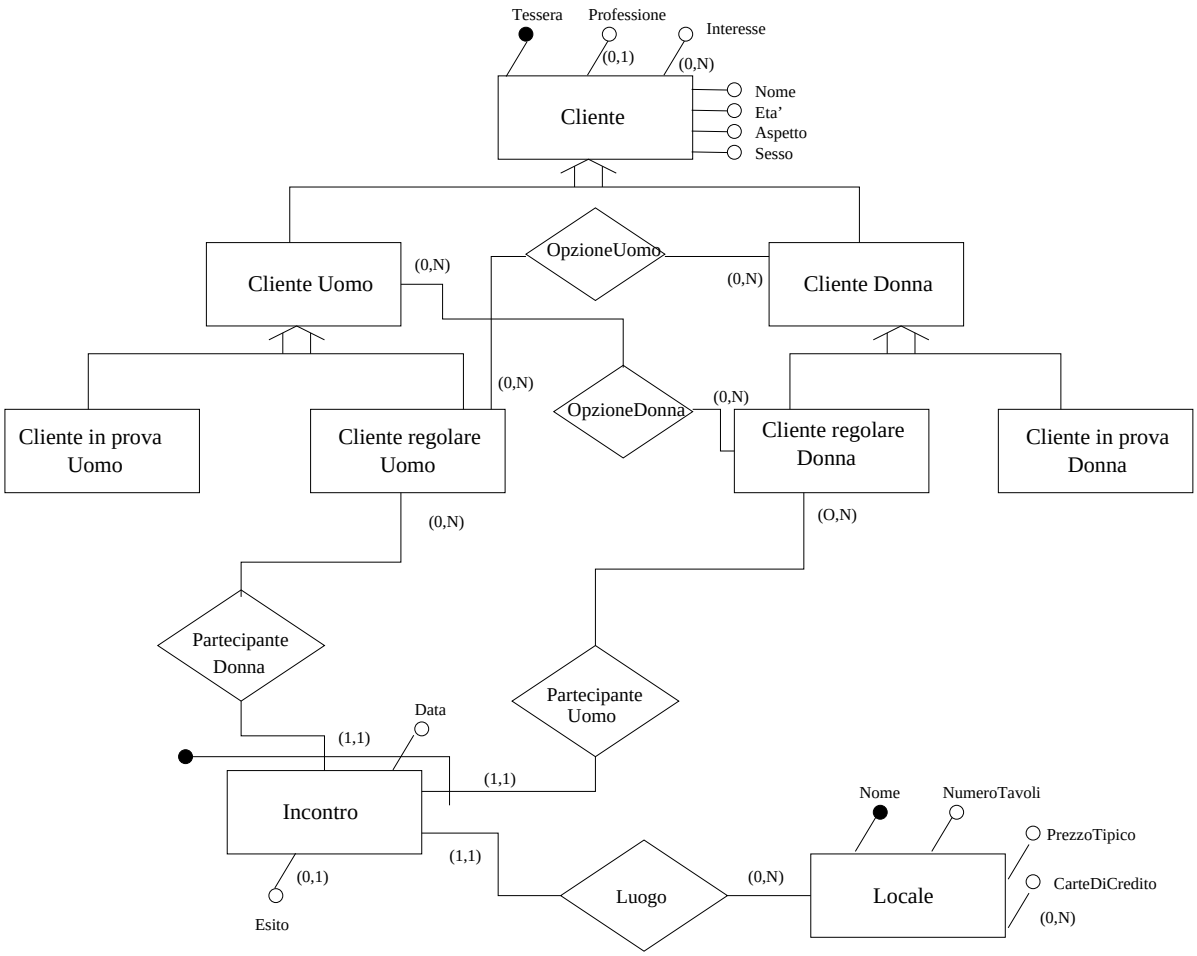
Esercizio 5

Vere: 1,4. False: 2,3,5.

Soluzione del compito del 10 dicembre 2001

Esercizio 1

Per poter gestire con dei vincoli di riferimento il fatto che un cliente possa opzionare e incontrare solo clienti dell'altro sesso è opportuno creare una gerarchia di clienti in base al sesso. Per ragioni analoghe è corretto creare in cascata un'ulteriore gerarchia in base all'iscrizione (regolare o in prova).



Esercizio 2

La gerarchia viene eliminata accorpando i clienti in prova nell'entità Cliente, mentre i clienti regolari (uomini o donne) vengono mantenuti come entità distinte.

Gli attributi multivalore Interesse e CarteDiCredito vengono trasformati in entità.

CLIENTI(Tessera, Nome, Età, Aspetto, Sesso, Professione*)
CLIENTIUOMINI(Tessera)
CLIENTIDONNE(Tessera)
CLIENTIUOMINIREGOLARI(Tessera)
CLIENTIDONNEREGOLARI(Tessera)
INTERESSI(Cliente,Interesse)
INCONTRI(PartecipanteDonna, PartecipanteUomo, Data, Locale, Esito*)
LOCALI(Nome, NumeroTavoli, PrezzoTipico)
CARTEACCETTATE(Locale, CartaDiCredito)
OPZIONIDONNE(Opzionante, Opzionato)
OPZIONIUOMINI(Opzionante, Opzionato)

con i vincoli:

CLIENTIUOMINI(Tessera) $\subseteq$ CLIENTI(Tessera)
CLIENTIDONNE(Tessera) $\subseteq$ CLIENTI(Tessera)
CLIENTIUOMINIREGOLARI(Tessera) $\subseteq$ CLIENTIUOMINI(Tessera)
CLIENTIDONNEREGOLARI(Tessera) $\subseteq$ CLIENTIDONNE(Tessera)
INTERESSI(Cliente) $\subseteq$ CLIENTI(Tessera)
INCONTRI(PartecipanteDonna) $\subseteq$ CLIENTIDONNEREGOLARI(Tessera)
INCONTRI(PartecipanteUomo) $\subseteq$ CLIENTIUOMINIREGOLARI(Tessera)
INCONTRI(Locale) $\subseteq$ LOCALI(Nome)

CarteAccettate(Locale) ⊆ Locali(Nome)
OpzioniDonne(Opzionante) ⊆ ClientiDonneRegolari(Tessera)
OpzioniDonne(Opzionato) ⊆ ClientiUomini(Tessera)
OpzioniUomini(Opzionante) ⊆ ClientiUominiRegolari(Tessera)
OpzioniUomini(Opzionata) ⊆ ClientiDonne(ClientiDonne)

Esercizio 3

```
create table Clienti
(
  Tessera char(5) primary key,
  Nome varchar(30) not null,
  Eta integer,
  Aspetto varchar(20),
  Sesso char(1),
  Professione varchar(30)
)

create table Interessi
(
  Cliente char(5) references Clienti(Tessera),
  Interesse varchar(30),
  primary key (Cliente, Interesse)
)

insert into Clienti values ('M0001', 'Mario Rossi', 23,
                            'Piacevole', 'M', 'Architetto')

insert into Interessi values ('M0001', 'Giardinaggio')
```

Esercizio 4

```
select Tessera, Nome, count(*)
from Clienti join Interessi on Tessera = Cliente
where Sesso = 'M'
  and Aspetto = 'piacevole'
group by Tessera, Nome
```

Esercizio 5

```
select avg(PrezzoTipico)
from Locali
     join Incontri on Locali.Nome = Incontri.Locale
     join Cliente as Uomo on Incontri.PartecipanteUomo = Uomo.Tessera
     join Cliente as Donna on Incontri.PartecipanteDonna = Donna.Tessera
where Uomo.Nome = 'Mario Rossi'
  and Donna.Eta > 40
```

Esercizio 6

1. $R1 \bowtie_{A=E} R2$ — 0 — $min(c_1, c_2)$
2. $R1 \bowtie_{C=E} R2$ — 0 — c_1
3. $R1 \bowtie_{A=F} R2$ — 0 — c_2
4. $R1 \bowtie_{B=F} R2$ — 0 — $c_1 \cdot c_2$

Soluzione del compito del 20 marzo 2002

Esercizio 1

Chiavi:

- Cod, CodRuolo
- Cod, Ruolo

Altre dipendenze:

- CodRuolo → Ruolo, Ruolo → CodRuolo
- CodNaz → Nazione, Nazione → CodNaz
- Cod → Cognome, Nome, CodNaz, Nazione, DataNascita, Presenze

Tabelle:

- Gioca(<u>Giocatore</u>, <u>Ruolo</u>)
- Ruoli(<u>Codice</u>, Descrizione)
- Nazioni(<u>Codice</u>, Nome)
- Giocatori(<u>Codice</u>, Cognome, Nome, Nazionalità, DataNascita, Presenze)

Vincoli:

- Gioca(Ruolo) ⊆ Ruoli(Codice)
- Gioca(Giocatore) ⊆ Giocatori(Codice)
- Giocatori(Nazionalità) ⊆ Nazioni(Codice)

Esercizio 2

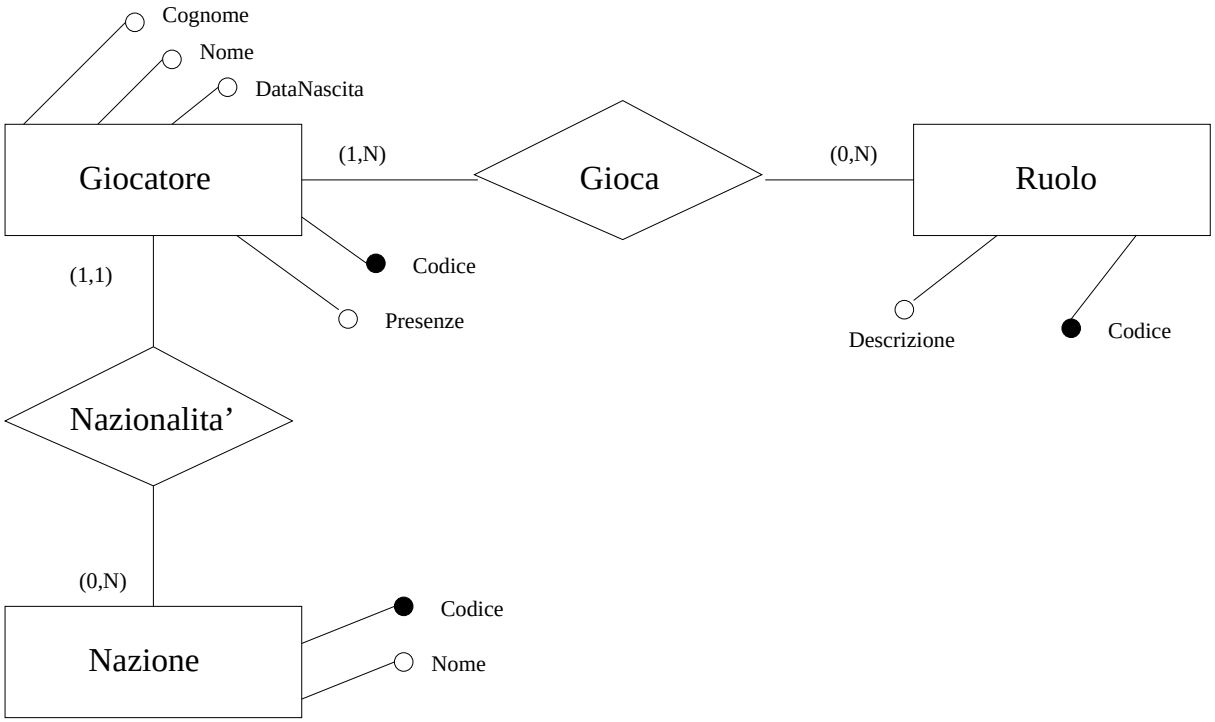
```
create table Ruoli
(
  Codice char(1) primary key,
  Descrizione varchar(20) unique
)

create table Nazioni
(
  Codice varchar(3) primary key,
  Nome varchar(20) unique
)

create table Giocatori
(
  Codice char(3) primary key,
  Cognome varchar(20),
  Nome varchar(20),
  Nazionalita varchar(3) references Nazioni(Codice),
  DataNascita date,
  Presenze integer
)

create table Gioca
(
  Giocatore char(3) references Giocatori(Codice),
  Ruolo char(1) references Ruoli(Codice),
  primary key(Giocatore, Ruolo)
)
```

Esercizio 3



Esercizio 4

$$\pi_{Nome,Cognome}(Giocatori \bowtie_{Cod=Gio} Gioca \bowtie_{Cod=Gio1 \wedge Ruolo \neq Ruolo1} (\rho_{Gio,Ruolo \leftarrow Gio1,Ruolo1} Gioca))$$

Esercizio 5

```
select Descrizione, count(*)
from Ruoli
     join Gioca on Ruoli.Codice = Ruolo
     join Giocatori on Giocatore = Giocatori.Codice
where DataNascita >= '1-jan-1979'
group by Descrizione
```

Esercizio 6

Per questo esercizio forniamo due soluzioni (con e senza operatori di aggregazione).

Soluzione 1

```
select Nazioni.Nome
from Giocatori
     join Nazioni on Nazioni.Codice = Nazionalita
     join Gioca on Giocatori.Codice = Giocatore
group by Nazioni.Nome
having count (distinct Ruolo) = 1
```

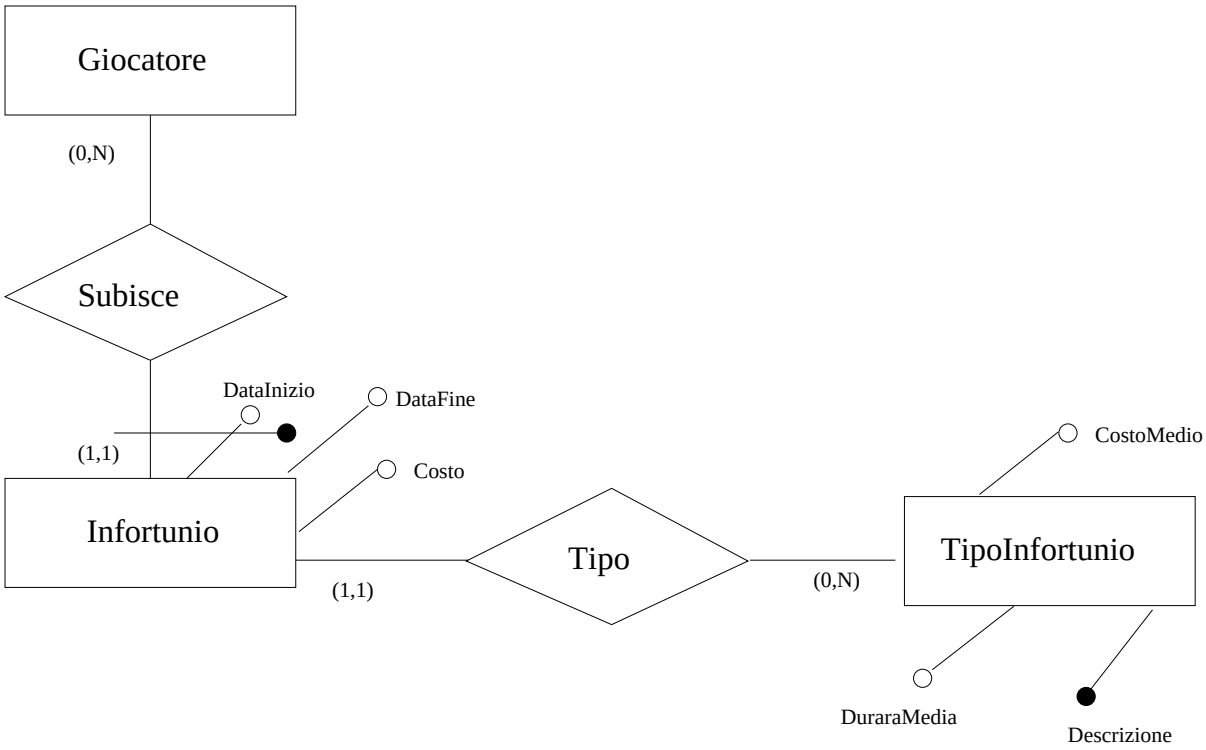
Soluzione 2

```
select Nazioni.nome
from Nazioni
where not exists (select *
```

```
from Giocatori as G1
     join Gioca as Gioca1 on G1.Codice = Gioca1.Giocatore
     join Giocatori as G2 on  G1.Nazionalita = G2.Nazionalita
     join Gioca as Gioca2 on G2.Codice = Gioca2.Giocatore
where G2.Nazionalita = Nazioni.Codice
  and Gioca1.Ruolo != Gioca2.Ruolo)
```

Nota: `g1` e `g2` possono essere indifferentemente lo stesso giocatore o giocatori diversi.

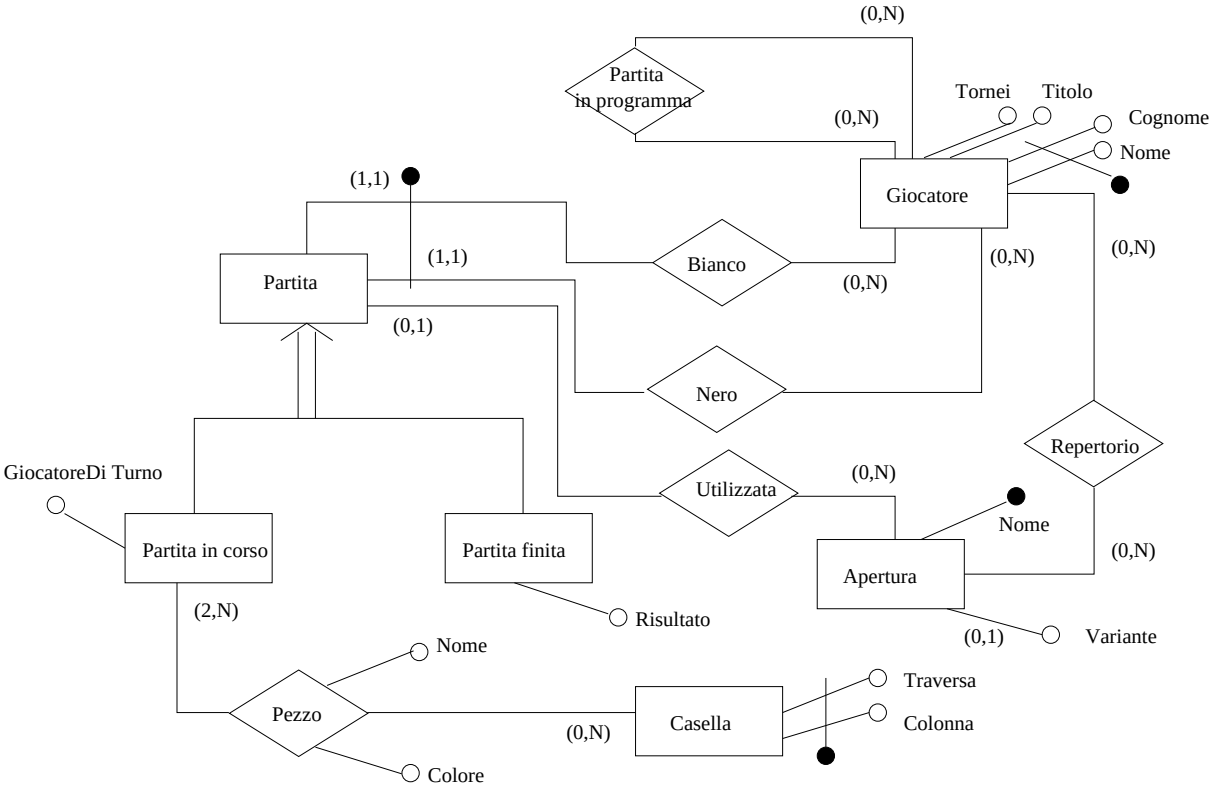
Esercizio 7



Nota: L'entità Infortunio non è una relazione perché un giocatore può avere due volte lo stesso tipo di infortunio.

Soluzione del compito dell'8 aprile 2002

Esercizio 1



Si è assunto che la formula del torneo sia tale che due giocatori non possano affrontarsi due volte con gli stessi colori.

Esercizio 2

Tabelle:

- Giocatori(<u>Cognome</u>, Nome, Titolo, TorneiVinti)
- PartiteInProgramma(<u>PrimoGiocatore</u>, <u>SecondoGiocatore</u>)
- Partite(<u>Bianco</u>, <u>Nero</u>, Risultato*, GiocatoreDiTurno*, Apertura*)
- Aperture(<u>Nome</u>, Variante*)
- Caselle(<u>Traversa</u>, <u>Colonna</u>)
- Repertorio(<u>Giocatore</u>, <u>Apertura</u>)
- Pezzi(<u>GiocatoreBianco</u>, <u>GiocatoreNero</u>, <u>Traversa</u>, <u>Colonna</u>, Pezzo, ColorePezzo)

Vincoli:

- PartiteInProgramma(PrimoGiocatore) $\subseteq$ Giocatori(Cognome)
- PartiteInProgramma(SecondoGiocatore) $\subseteq$ Giocatori(Cognome)
- Partite(Bianco) $\subseteq$ Giocatori(Cognome)
- Partite(Nero) $\subseteq$ Giocatori(Cognome)
- Partite(Apertura) $\subseteq$ Aperture(Nome)
- Repertorio(Giocatore) $\subseteq$ Giocatori(Cognome)
- Repertorio(Apertura) $\subseteq$ Aperture(Nome)
- Pezzi(GiocatoreBianco) $\subseteq$ Partite(Bianco)
- Pezzi(GiocatoreNero) $\subseteq$ Partite(Nero)

- Pezzi(Traversa,Colonna) $\subseteq$ Caselle

Commenti:

- La gerarchia viene eliminata accorpando le entità figlie nell'entità genitore.
- Per semplicità abbiamo assunto che l'attributo `Cognome` sia chiave della relazione `Giocatore`.
- Gli attributi con * possono avere valori nulli.
- L'attributo `Risultato` assume i valori `'Bianco'`, `'Nero'` o `'Patta'`
- L'attributo `GiocatoreDiTurno` assume i valori `'Bianco'` o `'Nero'`

Esercizio 3

```
select Copie, Titolo, Cognome
from Librerie
     join Scorte on Librerie.Codice = Scorte.Libreria
     join Libri on Scorte.Libro = Libri.Codice
     join HaScritto on Libri.Codice = HaScritto.Libro
     join Autori on HaScritto.Autore = Autori.Codice
where ProgressivoAutore = 1
  and Librerie.Nome = 'NonSoloLibri'
```

Esercizio 4

```
select Titolo, Editore
from Libri
where Prezzo >all (select Prezzo
                   from Libri
                   where Genere = 'Fantascienza')
```

Esercizio 5

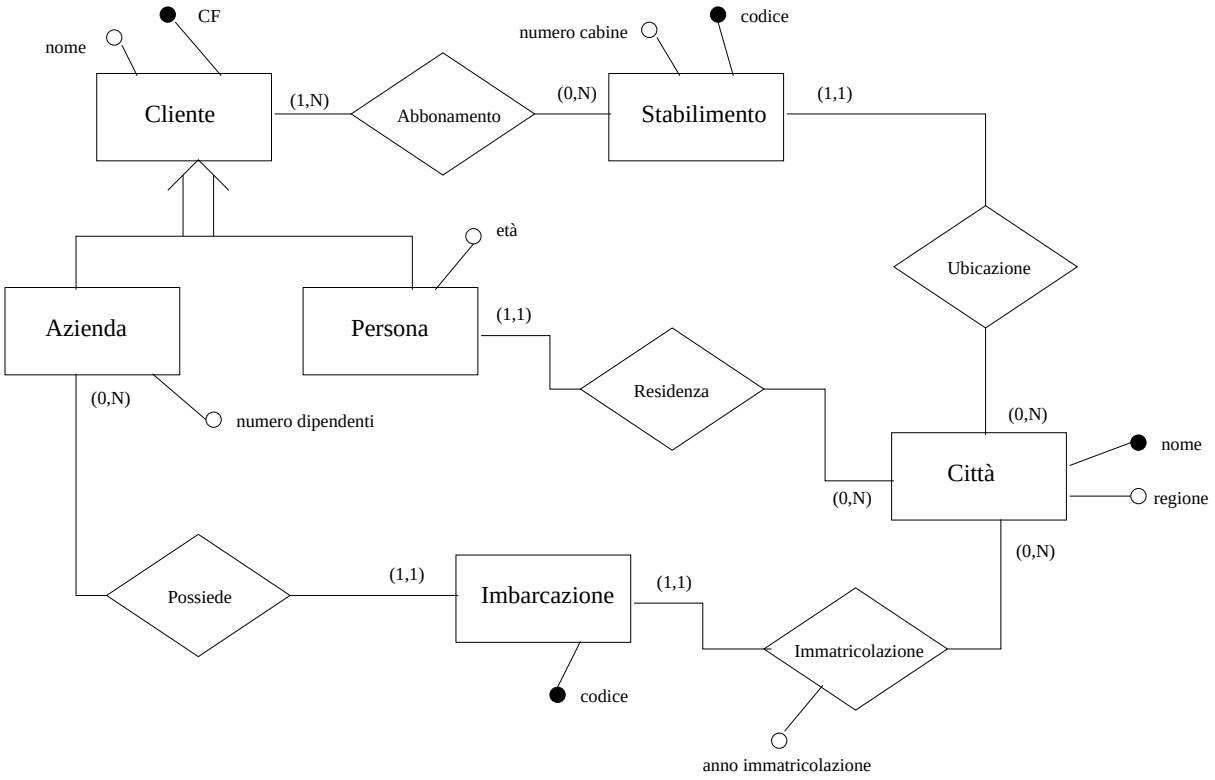
Chiavi: ACE, BCE, CDE

Forma Normale:

- R non è in BCNF perché ci sono dipendenze (ad esempio $A \to B$) in cui l'attributo a sinistra non è chiave
- R è in 3NF perchè in tutte le dipendenze gli attributi a destra sono membri di chiavi

Soluzione del compito del 20 giugno 2002

Esercizio 1



Esercizio 2

Tabelle:

- Clienti(<u>CF</u>,Nome)
- Persone(<u>CF</u>, Età, Città)
- Aziende(<u>CF</u>, NumeroDipendenti)
- Stabilimenti(<u>Codice</u>, NumeroCabine, Città)
- Città(<u>Nome</u>, Regione)
- Imbarcazioni(<u>Codice</u>, Azienda, CittàImmatricolazione, AnnoImmatricolazione)
- Abbonamento(<u>Cliente,Stabilimento</u>)

Vincoli:

- Persone(CF) $\subseteq$ Clienti(CF)
- Aziende(CF) $\subseteq$ Clienti(CF)
- Stabilimenti(Città) $\subseteq$ Città(Nome)
- Persone(Città) $\subseteq$ Città(Nome)
- Imbarcazioni(CittàImmatricolazione) $\subseteq$ Città(Nome)
- Imbarcazioni(Azienda) $\subseteq$ Aziende(CF)
- Abbonamento(Cliente) $\subseteq$ Clienti(CF)
- Abbonamento(Stabilimento) $\subseteq$ Stabilimenti(Codice)

Commenti:

- L'unica generalizzazione presente viene sostituita da due relazioni uno a uno che legano l'entità padre Cliente alle entità figlie (Persona e Azienda).

Esercizio 3

$\pi_{codice}\sigma_{NumeroCabine \geq 50}Stabilimenti$

Esercizio 4

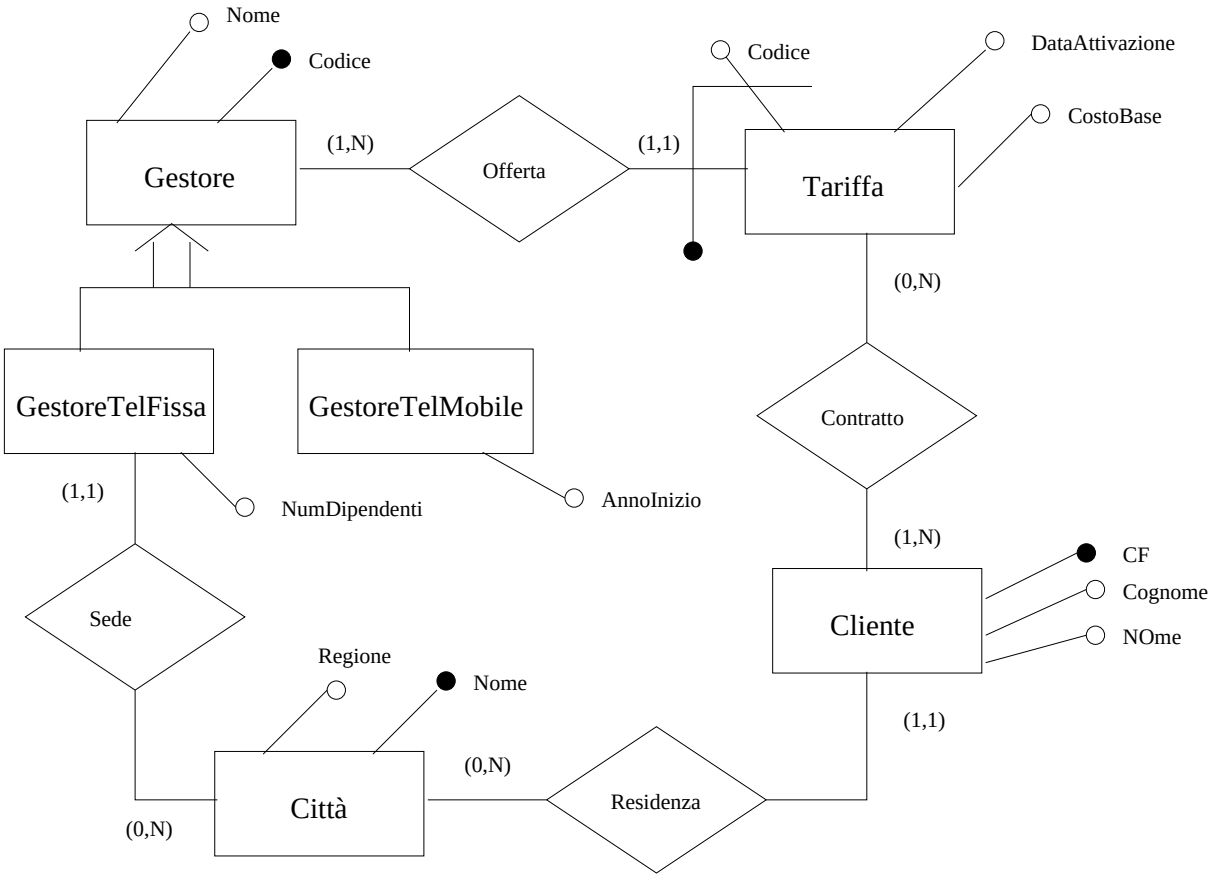
```
select count(*)
from Imbarcazioni join Aziende on Aziende.CF = Imbarcazioni.Azienda
where NumeroDipendenti > 100
  and AnnoImmatricolazione > 1999
```

Esercizio 5

```
select distinct A1.Cliente, A1.Stabilimento
from Abbonamenti as A1
     join Abbonamenti as A2 on A1.Stabilimento = A2.Stabilimento
     join Imbarcazioni on A2.Cliente = Imbarcazioni.Azienda
```

Soluzione del compito dell'11 luglio 2002

Esercizio 1



Esercizio 2

Tabelle:

- GESTORE(<u>Codice</u>,Nome)
- TARIFFA(<u>Codice,Gestore</u>,DataAttivazione,CostoBase)

- Citta(Nome,Regione)
- Cliente(CF,Nome,Cognome,Citta)
- GestoreTelMobile(Codice,AnnoInizio)
- GestoreTelFissa(Codice,NumeroDipendenti, Citta)
- Contratto(Gestore,Tariffa,Cliente)

Vincoli:

- Tariffa(Gestore) $\subseteq$ Gestore(Codice)
- Cliente(Citta) $\subseteq$ Citta(Nome)
- GestoreTelMobile(Codice) $\subseteq$ Gestore(Codice)
- GestoreTelFissa(Codice) $\subseteq$ Gestore(Codice)
- GestoreTelFissa(Citta) $\subseteq$ Citta(Nome)
- Contratto(Gestore,Tariffa) $\subseteq$ Tariffa(Codice,Gestore)
- Contratto(Cliente) $\subseteq$ Cliente(CF)

Commenti:

- L'unica generalizzazione presente viene sostituita da due relazioni uno a uno che legano l'entità padre Gestore alle entità figlie (GestoreTelFissa e GestoreTelMobile).

Esercizio 3

```
create table Gestori
(
  codice char(3) primary key,
  nome varchar(20)
)

create table Tariffe
(
   codice char(2),
   gestore char(3) references Gestori(codice),
   DataAttivazione date,
   prezzo real,
   primary key(codice, gestore)
)
```

Esercizio 4

$\pi_{Nome,Regione}(Citta \bowtie_{Nome=Citta} \sigma_{NumeroDipendenti \geq 500}(GestoreTelFissa))$

Esercizio 5

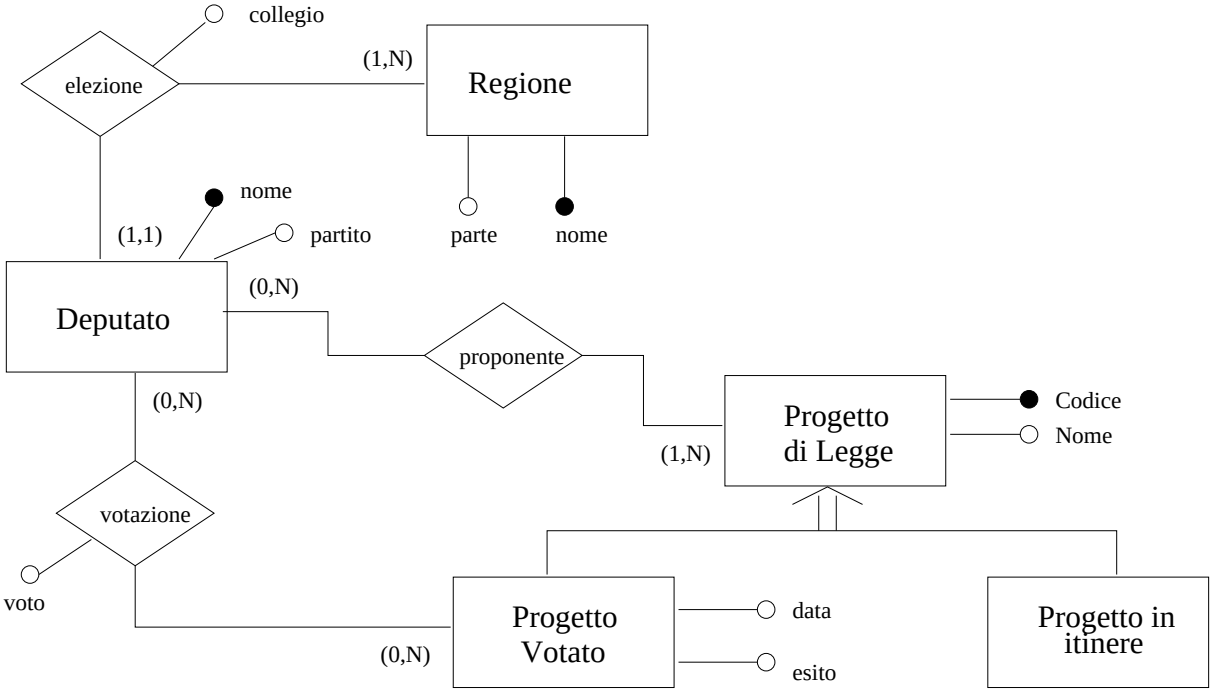
```
select distinct Gestori.Nome, Gestori.Codice
from Gestori join Tariffe on Gestori.Codice = Tariffe.Gestore
where (Tariffe.Codice, Tariffe.Gestore) not in (select Tariffa, Gestore
                                                from Contratti)
```

Esercizio 6

```
select Nome, Cognome, CF, Codice, Tariffe.Gestore
from Clienti
     join Contratti on CF = Cliente
     join Tariffe on Codice = Tariffa and  Tariffe.Gestore = Contratti.Gestore
where DataAttivazione >=all (select DataAttivazione
                             from Contratti join Tariffe on Codice = Tariffa
                                                        and Tariffe.Gestore = Contratti.Gestore
                             where CF = Cliente)
```

Soluzione del compito del 9 settembre 2002

Esercizio 1



Esercizio 2

Tabelle:

- DEPUTATI(Nome,Partito,Collegio,Regione)
- REGIONI(Nome,Parte)
- PROGETTIDILEGGE(Codice,Nome)
- PROGETTIVOTATO(Codice,Data,Esito)
- PROPONENTE(Deputato,Progetto)
- VOTAZIONE(Deputato, Progetto,Voto)

Vincoli:

- DEPUTATI(Collegio) $\subseteq$ REGIONI(Nome)
- PROGETTIVOTATO(Codice) $\subseteq$ PROGETTIDILEGGE(Codice)
- PROPONENTE(Deputato) $\subseteq$ DEPUTATI(Nome)
- PROPONENTE(Progetto) $\subseteq$ PROGETTIDILEGGE(Codice)
- VOTAZIONE(Deputato) $\subseteq$ DEPUTATI(Nome)
- VOTAZIONE(Progetto) $\subseteq$ PROGETTIVOTATO(Codice)

Commenti:

- La gerarchia viene eliminata accorpando l'entità ProgettoInItinere nell'entità ProgettoDiLegge e creando una relazione tra ProgettoDiLegge e ProgettoVotato.

Esercizio 3

```
create table ProgettiDiLegge
(
  Codice integer primary key,
  Nome varchar(100)
)
```

```
create table Proponente
(
  Deputato varchar(40) references Deputati(nome),
  Progetto integer references ProgettiDiLegge(codice),
  primary key (Deputato,Progetto)
)
```

Esercizio 4

$\pi_{Nome}(Regioni) - \pi_{Nome}(Regioni \bowtie_{Nome=Regione} \sigma_{Partito='PXP'} \rho_{NomeDep \leftarrow Nome}(Deputati))$

Esercizio 5

La soluzione di questa interrogazione è abbastanza complessa e quindi la risolviamo in due passi. Scriviamo inizialmente un'interrogazione che restituisce per ciascuna regione il numero di deputati che hanno votato sì alla legge in questione.

```
select Regioni.Nome, count(*)
from Regioni
     join Deputati on Regioni.Nome = Deputati.Regione
     join Votazione on Deputati.Nome = Votazione.Deputato
     join ProgettiDiLegge on Votazione.Progetto = ProgettiDiLegge.Codice
where ProgettiDiLegge.Nome = 'Pizza per tutti'
  and Votazione.Voto = 'SI'
group by Regioni.Nome
```

Ora scriviamo l'interrogazione che restituisce la parte del paese in cui la regione (le regioni) per cui il numero di deputati à massimo. Questo si ottiene ripetendo l'interrogazione stessa, quasi inalterata, all'interno delle clausola `having`.

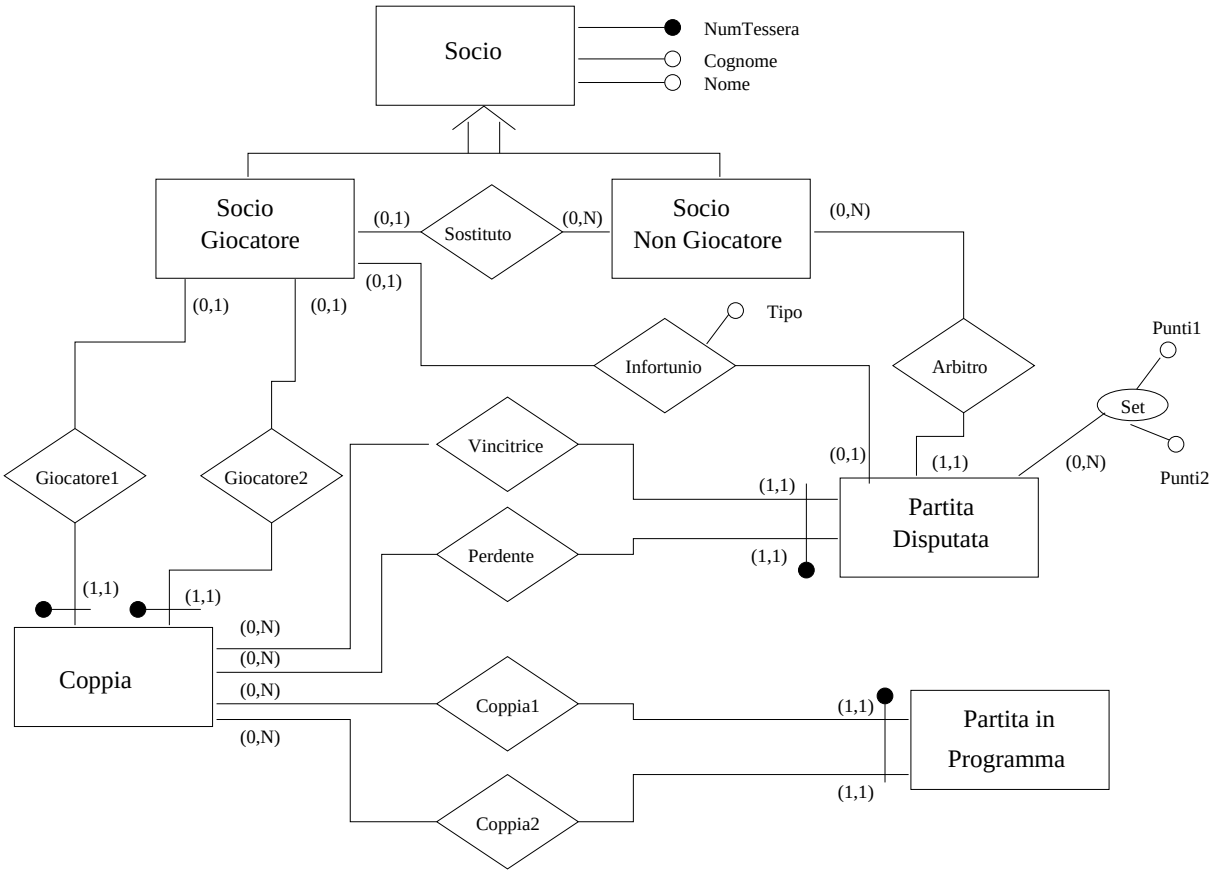
```
select Parte
from Regioni
     join Deputati on Regioni.Nome = Deputati.Regione
     join Votazione on Deputati.Nome = Votazione.Deputato
     join ProgettiDiLegge on Votazione.Progetto = ProgettiDiLegge.Codice
where ProgettiDiLegge.Nome = 'Pizza per tutti'
  and votazione.voto = 'SI'
group by Regioni.Nome
having count(*) >=all (select count(*)
                       from Regioni
                            join Deputati on Regioni.Nome = Deputati.Regione
                            join Votazione on Deputati.Nome = Votazione.Deputato
                            join ProgettiDiLegge on Votazione.Progetto = ProgettiDiLegge.Codice
                       where ProgettiDiLegge.Nome = 'Pizza per tutti'
                         and Votazione.Voto = 'SI'
                       group by Regioni.Nome)
```

Esercizio 6

```
select distinct V1.Deputato
from Votazione as V1
     join Votazione as V2 on V1.Deputato = V2.Deputato
     join Proponente as P1 on V1.Deputato = P1.Deputato and V1.Progetto = P1.Progetto
     join Proponente as P2 on V2.Deputato = P2.Deputato and V2.Progetto = P2.Progetto
where V1.Progetto != V2.Progetto
  and V1.Voto = 'NO'
  and V2.Voto = 'NO'
```

Soluzione del compito del 25 settembre 2002

Esercizio 1



Si noti che è stata sviluppata una soluzione in cui non è presente una gerarchia per le partite, contrariamente a quanto le specifiche sembrerebbero suggerire. Questo perché nella soluzione proposta le entità PartitaDisputata e PartitaInProgramma hanno identificatori diversi, e non esiste un identificatore comune da assegnare ad una ipotetica entità `Partita`.

Una scelta alternativa, ugualmente valida, consiste nell'introdurre la gerarchia e avere le relazioni che partono da Coppia che insistono sull'entità padre. In tal caso, l'entità PartitaDisputata dovrebbe avere un attributo che memorizzi chi ha vinto tra la coppia 1 e la coppia 2.

Si noti inoltre che si è supposto che un giocatore possa partecipare solo ad una coppia, e che conseguentemente sia il giocatore 1 da solo che il giocatore 2 da solo identifichino una coppia.

Per i risultati dei set, la ristrutturazione ci porta a creare una entità debole identificata dal numero del set e dalla relazione con partita.

Esercizio 2

Tabelle:

- Socio(NumTessera,Nome, Cognome)
- SocioGiocatore(NumTessera)
- SocioNonGiocatore(NumTessera)
- Sostituzione(Giocatore, Sostituto)
- Infortunio(Giocatore, GiocVincente1Partita, GiocPerdente1Partita, TipoInfortunio)
- Coppia(Giocatore1,Giocatore2)
- PartitaDisputata(Giocatore1Vincente,Giocatore1Perdente, Arbitro)
- PartitaInProgramma(Coppia1Giocatore1, Coppia2Giocatore1)
- RisultatoSet(Giocatore1Vincente,Giocatore1Perdente, NumSet, PunteggioVincente, PunteggioPerdente)

Vincoli:

- SocioGiocatore(NumTessera) ⊆ Socio(NumTessera)
- SocioNonGiocatore(NumTessera) ⊆ Socio(NumTessera)
- Sostituzione(Giocatore) ⊆ SocioGiocatore(NumTessera)
- Sostituzione(Sostituto) ⊆ SocioNonGiocatore(NumTessera)
- Infortunio(Giocatore) ⊆ SocioGiocatore(NumTessera)
- Infortunio(GiocatoreVincente1Partita, GiocatorePerdente1Partita) ⊆ PartitaDisputata(Giocatore1Vincente, Giocatore1Perdente)
- Coppia(Giocatore1) ⊆ SocioGiocatore(NumTessera)
- Coppia(Giocatore2) ⊆ SocioGiocatore(NumTessera)
- PartitaDisputata(Giocatore1Vincente) ⊆ Coppia(Giocatore1)
- PartitaDisputata(Giocatore1Perdente) ⊆ Coppia(Giocatore1)
- PartitaDisputata(Arbitro) ⊆ SocioNonGiocatore(NumTessera)
- PartitaInProgramma(Coppia1Giocatore1) ⊆ Coppia(Giocatore1)
- PartitaInProgramma(Coppia2Giocatore1) ⊆ Coppia(Giocatore1)
- RisultatoSet(Giocatore1Vincente) ⊆ Coppia(Giocatore1)
- RisultatoSet(Giocatore1Perdente) ⊆ Coppia(Giocatore1)

Commenti:

- L'unica gerarchia presente viene sostituita con due relazioni tra l'entità padre Socio e le due entità figlie SocioGiocatore e SocioNonGiocatore.
- Come identificatore primario di una coppia viene scelto il giocatore 1 (in alternativa al giocatore 2).
- L'attributo (composto) multivalore viene trasformato in un'entità Set. Questa entità non viene tradotta in quanto le sue informazioni sono ridondanti rispetto alla relazione `RisultatoSet` ottenuta traducendo la relazione tra PartitaDisputata e Set.

Esercizio 3

$$\pi_{Nome,Cognome}\sigma_{AnnoNascita>A}(Persona \bowtie_{CF=Marito} Coniugio \bowtie_{Moglie=CF1} \rho_{CF1,N,C,A,S\leftarrow CF,Nome,Cognome,AnnoNascito,Sesso}(Persona))$$

Esercizio 4

La difficoltà di questo esercizio risiede nel fatto che una persona deve comparire nella risposta sia che sia moglie che marito nella relazione `coniugio`. Siamo quindi in presenza di un legame più complesso di un semplice equi-join tra le relazioni `persona` e `coniugio`. La soluzione più semplice consiste nel usare come condizione di join un `or` tra le due condizioni di equi-join (`CF = Moglie` e `CF = Marito`).

```
select Nome, Cognome
from Persona jon Coniugio on (CF = Moglie or CF = Marito)
where AnnoNascita + 20 >= AnnoMatrimonio
```

Esercizio 5

Si cercano le donne che si sono sposate almeno due volte, e si impone che la data del matrimonio di cui si seleziona il cognome del marito sia maggiore di tutte le date dei matrimoni della stessa donna.

```
select distinct Mo.Nome, Mo.Cognome, Ma.Cognome,
from Persona as Mo
     join Coniugio as C1 on Mo.CF = C1.Moglie
     join Persona as Ma on Ma.CF = C1.Marito
     join Coniugio as C2 on C1.Moglie = C2.Moglie
where C1.Marito != C2.Marito
  and C1.DataMatrimonio >=all (select DataMatrimonio
                               from Coniugio
                               where moglie = Mo.CF)
```

Esercizio 6

Si devono scegliere un'istanza ed una decomposizione tali che gli attributi in comune non formino una chiave in entrambe le relazioni risultanti.

Si consideri ad esempio la seguente istanza, che è costruita appositamente affinché D non sia chiave in $\pi_{ABCD}(R)$ e non sia chiave in π_{DE}.

A	B	C	D	E
x	y	1	2	3
x	z	1	2	4

Decomponendo sugli insiemi di attributi $ABCD$ e DE si ottengono le relazioni

A	B	C	D
x	y	1	2
x	z	1	2

e

D	E
2	3
2	4

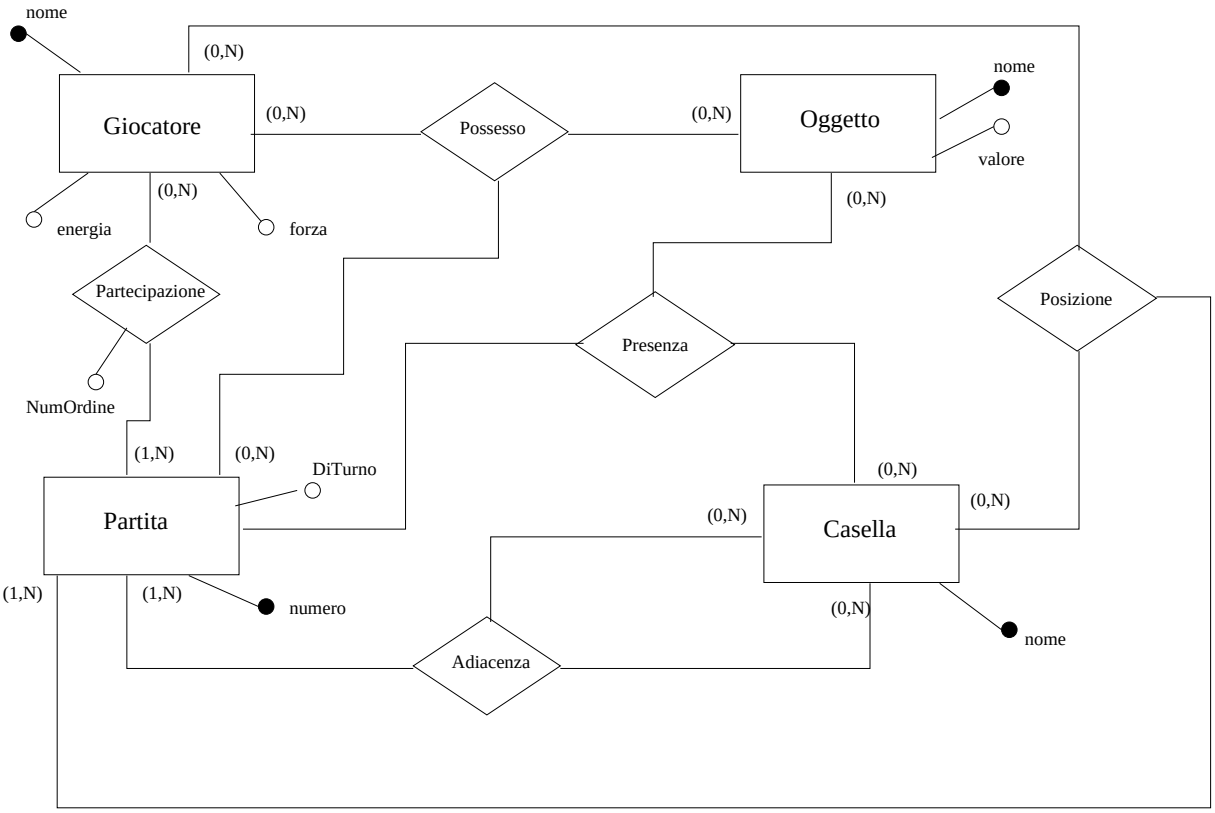
Facendo il join naturale di queste due relazione si ottiene la seguente relazione

A	B	C	D	E
x	y	1	2	3
x	z	1	2	4
x	y	1	2	4
x	z	1	2	3

che è diversa da R, quindi l'istanza e la decomposizione presentate rispondono all'esercizio.

Soluzione del compito del 16 dicembre 2002

Esercizio 1



Si noti che le relazioni Possesso, Presenza e Adiacenza sono tutte relazioni a tre legate anche all'entità Partita. Infatti è specificato esplicitamente che queste relazioni possono variare da partita a partita.

La relazione Partecipazione potrebbe essere accorpata nella relazione Posizione, visto che tutti e soli i giocatori che hanno una posizione assegnata in una partita partecipano ad quella partita. Si è preferito mantenere distinte le due relazioni concettualmente, l'accorpamento verrà peró fatto i fase di progettazione logica (cfr. esercizio 2).

Esercizio 2

Si noti che nella relaziona ternaria Presenza vi è una proprietà che non si evince dalle cardinalità: per ogni coppia `Partita`/Casella vi è un unico oggetto. Ne consegue che nella tabella `Presenza` la chiave primaria sarà composta solo da questi due attributi invece che da tutti e tre. Similarmente, per la relazione ternaria `Posizione`, la coppia `Partita`/`Giocatore` è identificativa. La relazione Partecipazione viene accorpata nella tabella `Posizione` (che ha la stessa chiave).

Tabelle:

- Giocatori(Nome,Forza,Energia)
- Caselle(Nome)
- Oggetti(Nome,Valore)
- Partite(NumOrdine,GiocatoreDiTurno)
- Posizione(Partita,Giocatore,Casella,NumOrdine)
- Adiacenza(Partita,Casella1,Casella2)
- Possesso(Partita,Giocatore,Oggetto)
- Presenza(Partita,Casella,Oggetto)

Vincoli:

- Posizione(Partita) $\subseteq$ Partite(NumOrdine)

- Posizione(Giocatore) ⊆ Giocatori(Nome)
- PosizioneCasella() ⊆ Caselle(Nome)
- Adiacenza(Partita) ⊆ Partite(NumOrdine)
- Adiacenza(Casella1) ⊆ Caselle(Nome)
- Adiacenza(Casella2) ⊆ Caselle(Nome)
- Possesso(Partita) ⊆ Partite(NumOrdine)
- Possesso(Giocatore) ⊆ Giocatori(Nome)
- Possesso(Oggetto) ⊆ Oggetti(Nome)
- Presenza(Partita) ⊆ Partite(NumOrdine)
- Presenza(Casella) ⊆ Caselle(Nome)
- Presenza(Oggetto) ⊆ Oggetti(Nome)

Commenti:

- Come annunciato, le relazioni concettuali Partecipazione e Posizione sono state accorpate nella singola relazione `Posizione`.
- Le relazioni `Presenza` e `Posizione` non hanno come chiave l'unione delle chiavi delle relazioni derivanti dalle tre entità coinvolte, ma solo un sottoinsieme di queste. Infatti un'analisi precisa della realtà da modellare ci porta a constatare che un giocatore ha un'unica posizione in una partita ed una casella può aver al massimo un oggetto presente in essa in una partita.

Esercizio 3

$\pi_{Nome,Cognome}\sigma_{Sesso='m'}(Persona) - \pi_{Nome,Cognome}(Persona \bowtie_{CF=Marito} Coniugio)$

Esercizio 4

Si veda il commento all'esercizio 4 del compito del 25 settembre 2002 (pagina 116).

```
select Nome, Cognome
from Persona join Coniugio on (CF = Moglie or CF = Marito)
where LuogoNascita = LuogoMatromonio
```

Esercizio 5

Questa interrogazione è sostanzialmente identica all'esercizio 5 del compito del 25 settembre 2002 (pagina 116).

Esercizio 6

Si devono scegliere un'istanza ed una decomposizione tali che gli attributi in comune formino una chiave in almeno una delle relazioni risultanti.

Si consideri ad esempio la seguente istanza, che è costruita appositamente affinché sia rispettata la dipendenza $D \to E$ e che quindi D sia chiave in π_{DE}.

A	B	C	D	E
x	y	1	2	3
x	z	2	2	3
t	g	3	3	5
v	f	4	3	5
r	r	6	3	5

Decomponendo sugli insiemi di attributi $ABCD$ e DE si ottengono le relazioni

A	B	C	D
x	y	1	2
x	z	2	2
t	g	3	3
v	f	4	3
r	r	6	3

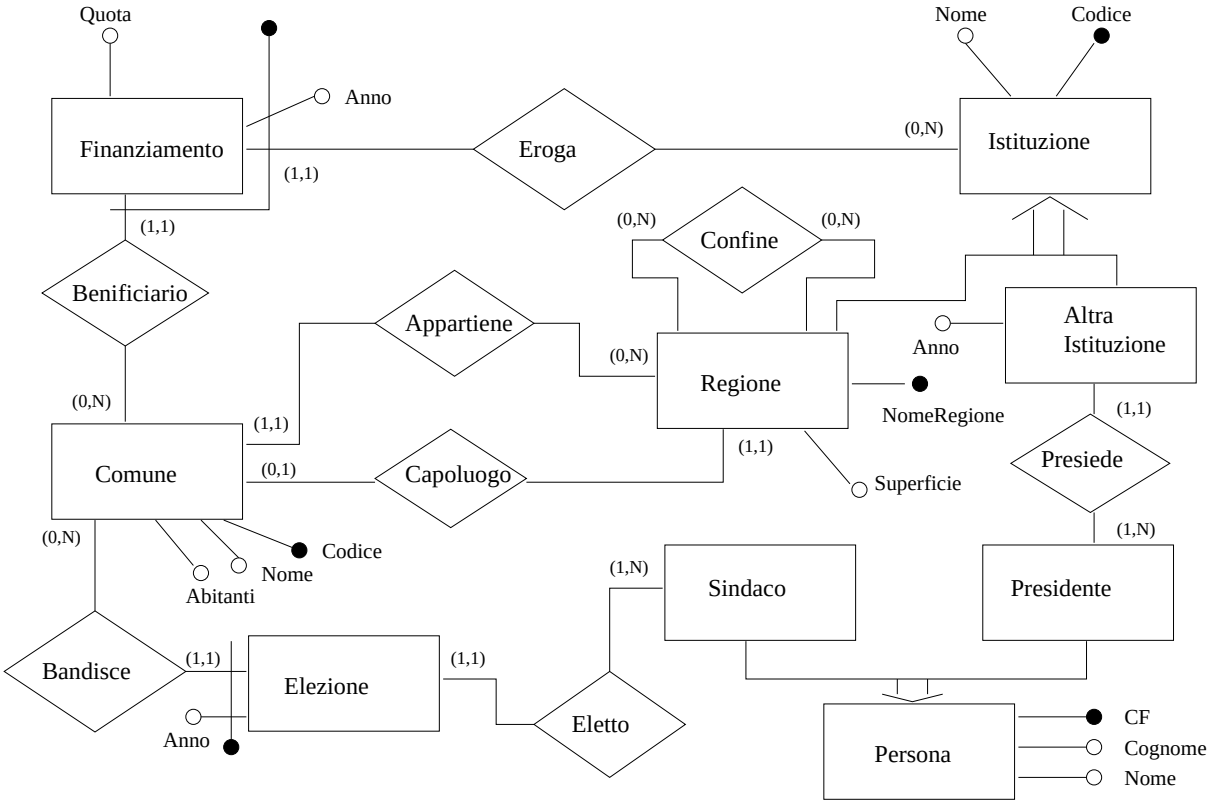
e

D	E
2	3
3	5

Facendo il join naturale di queste due relazione si ottiene la relazione R di partenza.

Soluzione del compito del 20 marzo 2003

Esercizio 1



Esercizio 2

Eliminazione delle gerarchie

La gerarchia con entità padre Persone viene eliminata accorpando le entità figlie nel padre.

La gerarchia con entità padre Istituzione viene sostituita da due relazioni.

Scelta degli identificatori primari

L'entità Regione ha due identificatori, uno interno `NomeRegione` ed uno esterno Istituzione. Si sceglie l'identificatore interno, quello esterno rimane come attributo non parte della chiave primaria.

Traduzione

COMUNE(Codice, Nome, Abitanti, Regione)
FINANZIAMENTO(Istituzione,Comune,Anno,Quota)
ISTITUZIONE(Codice,Nome)
REGIONE(Nome, Superficie, Capolougo, Istituzione)
ALTRAISTITUZIONE(Codice,Anno,Presidente)
CONFINE(Regione1,Regione2)
PERSONA(CF, Nome, Cognome, Età)
ELEZIONE(Comune, Anno, Sindaco)

Vincoli:

COMUNE(Regione) $\subseteq$ REGIONE(Nome)
FINANZIAMENTO(Istituzione) $\subseteq$ ISTITUZIONE(Codice)
FINANZIAMENTO(Comune) $\subseteq$ COMUNE(Codice)
REGIONE(Istituzione) $\subseteq$ ISTITUZIONE(Codice)
REGIONE(Capolougo) $\subseteq$ COMUNE(Codice)
ALTRAISTITUZIONE(Codice) $\subseteq$ ISTITUZIONE(Codice)
ALTRAISTITUZIONE(Presidente) $\subseteq$ PERSONA(CF)
CONFINE(Regione1) $\subseteq$ REGIONE(Nome)
CONFINE(Regione2) $\subseteq$ REGIONE(Nome)
ELEZIONE(Comune) $\subseteq$ COMUNE(Codice)
ELEZIONE(Sindaco) $\subseteq$ PERSONA(CF)

Esercizio 3

$\pi_{nome,cognome}\sigma_{marito \neq ma}(Persona \bowtie_{CF=moglie} Coniugio \bowtie_{CF=mo} \rho_{mo,ma,d,l \leftarrow moglie,marito,data,luogo}(Coniugio))$

Esercizio 4

Si veda il commento all'esercizio 4 del compito del 25 settembre 2002 (pagina 116).

Esercizio 5

Si cercano due matrimoni tra le stesse persone tali che non esista un matrimonio con data compresa tra le due del marito con un'altra moglie. Per far questo, si fissa un ordine tra i due matrimoni (`C1` prima di `C2`).

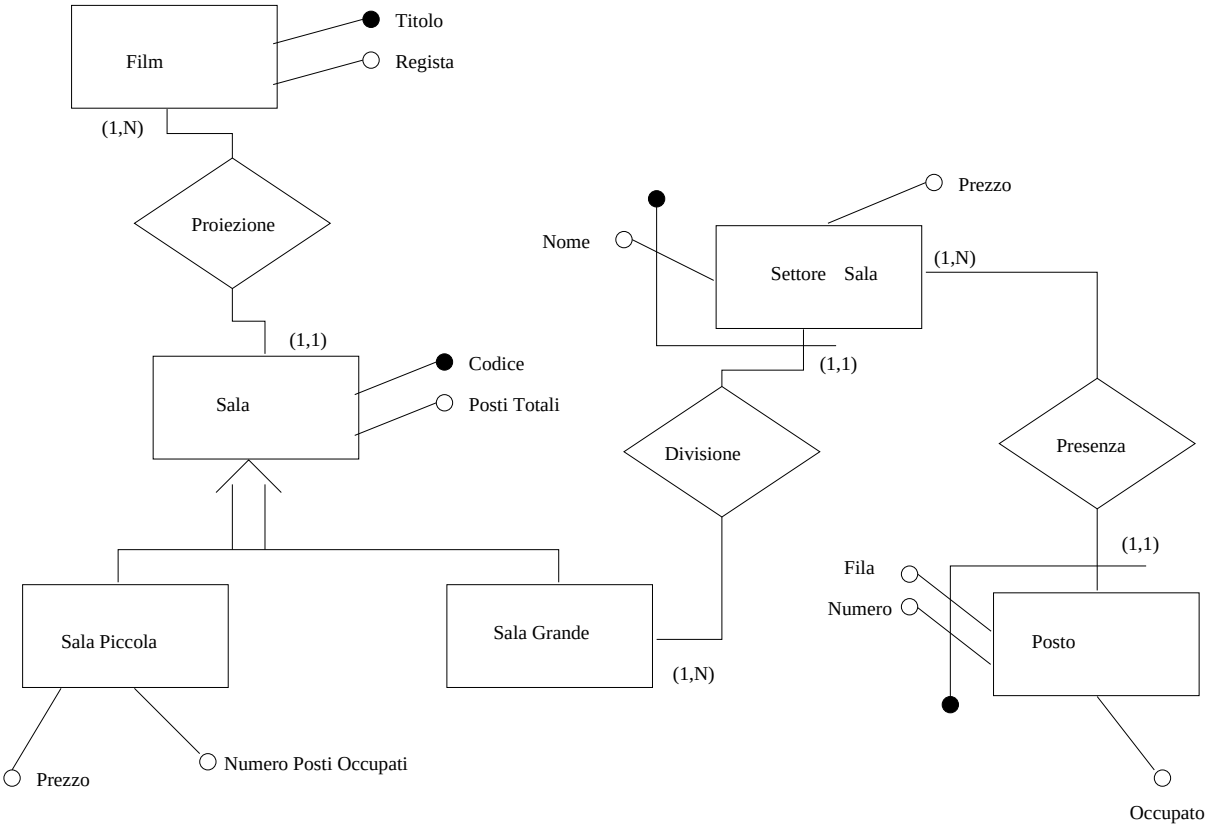
```
select distinct C1.Marito
from Coniugio as C1 join Coniugio as C2 on C1.Marito = C2.Marito and C1.Moglie = C2.Moglie
where  C1.DataMatrimonio < C2.DataMatrimonio
  and not exists (select DataMatrimonio
                  from Coniugio
                  where Marito = C1.Marito
                    and Moglie != C1.Moglie
                    and DataMatrimonio > C1.DataMatrimonio
                    and DataMatrimonio < C2.DataMatrimonio)
```

Esercizio 6

A	sum(E)
2	15
3	6
9	8

Soluzione del compito del 3 aprile 2003

Esercizio 1



Si noti che:

- La registrazione dei biglietti venduti avviene in modo diverso per le sale grandi e quelle piccole. Per quelle piccole, non essendoci l'assegnazione del posto, è sufficiente memorizzare il numero totale di biglietti venduti. Per le sale grandi, si memorizza per ogni posto se è occupato oppure no. Questo viene fatto attraverso l'attributo booleano Occupato dell'entità Posto.
- Il prezzo del biglietto viene memorizzato attraverso l'attributo Prezzo delle entità SalaPiccola e `SettoreSala`. A tale proposito si noti che l'entità SettoreSala rappresenta settori di specifiche sale (ad es. laterali sala 01, centrali sala 03) e non settori generici (ad es. laterali, galleria).

Esercizio 2

Eliminazione delle gerarchie

Per eliminare la gerarchia, si sceglie di eliminare l'entità `Sala` e distribuire la relazione `Proiezione` sulla entità figlie. Questa soluzione permette di non introdurre valori nulli.

Traduzione

FILM(Titolo,Regista)
SALAPICCOLA(Codice,PostiTotali,PostiOccupati,Prezzo, Film)
SALAGRANDE(Codice,PostiTotali, Film)
SETTORESALAGRANDE(Sala,Nome,Prezzo)
POSTO(Sala,Settore,Fila,Numero,Occupato)

Vincoli:

SALAPICCOLA(Film) $\subseteq$ FILM(Titolo)
SALAGRANDE(Film) $\subseteq$ FILM(Titolo)
SETTORESALAGRANDE(Sala) $\subseteq$ SALAGRANDE(Codice)
POSTO(Sala,Settore) $\subseteq$ SETTORESALAGRANDE(Sala,Nome)

Esercizio 3

$\pi_{Attore}(HaRecitatoIn) - \pi_{Attore}(HaRecitatoIn \bowtie_{Film=Titolo \wedge Attore \neq Regista} Film)$

Esercizio 4

Per scrivere l'interrogazione correttamente in SQL è necessario riformularla in italiano nel seguente modo:"Trovare gli attore tale che *non* esiste un film di Woody Allen in cui *non* hanno recitato". Così formulata, l'interrogazione viene tradotta con due `select` annidate entrambe legate da un `not` (corrispondente al *non* in italiano) all'interrogazione esterna.

Si noti inoltre che è necessario legare l'attore nella `select` più interna con l'attore della `select` più esterna (due livelli sopra). Questo è perfettamente lecito in basi alle regole di visibilità di SQL.

```
select distinct Attore
from HaRecitatoIn as HRI1
where not exists (select *
                  from Film
                  where Regista = 'Woody Allen'
                  and Titolo not in (select Film
                                     from HaRecitatoIn
                                     where Attore = HRI1.Attore))
```

Esercizio 5

Si cercano due attori che hanno fatto un film insieme, e tali che non esiste in film in cui il primo ha recitato e il secondo no, o viceversa.

Si noti la condizione `h1.attore < h2.attore` che si sostituisce alla condizione necessaria `h1.attore != h2.attore` ed inoltre evita che compaiano le stesse coppie invertite di posto.

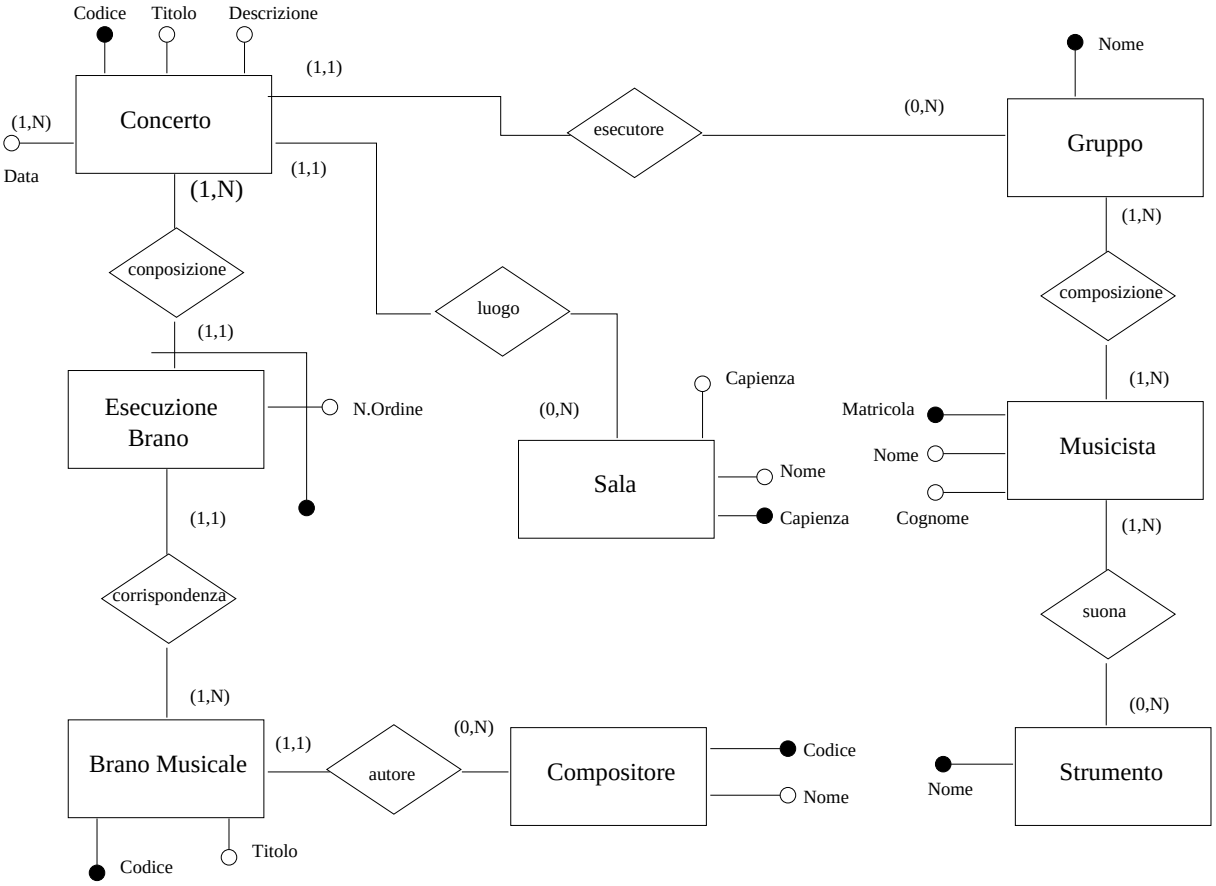
```
select distinct H1.Attore,H2.Attore
from HaRecitatoIn as H1 join HaRecitatoIn as H2 on H1.Film = H2.Film
where H1.Attore < H2.Attore
  and not exists (select *
                  from HaRecitatoIn as H3
                  where H1.Attore = H3.Attore
                  and H2.Attore not in (select Attore
                                        from HaRecitatoIn
                                        where Film = H3.Film))
  and not exists (select *
                  from HaRecitatoIn as H4
                  where H2.Attore = H4.Attore
                  and H1.Attore not in (select Attore
                                        from HaRecitatoIn
                                        where Film = H4.Film))
```

Esercizio 6

A	B
x	6
z	9
t	6

Soluzione del compito del 16 giugno 2003

Esercizio 1



Si noti che la possibilità che un brano venga eseguito più volte in un concerto, ci impone di inserire l'entità EsecuzioneBrano tra l'entità Concerto e l'entità BranoMusicale. Al contrario, se ogni brano fosse stato eseguito una sola volta, `EsecuzioneBrano` avrebbe dovuto essere una semplice relazione.

Esercizio 2

Ristrutturazione

L'unico passo necessario è l'eliminazione dell'attributo multivalore Data dell'entità Concerto. Questo viene fatto sostituendolo l'attributo con un'entità.

Traduzione

SALA(Codice,Nome,Capienza)
GRUPPO(Nome)
CONCERTO(Codice,Titolo,Descrizione,Gruppo,Sala)
REPLICA(Concerto,Data)
COMPOSITORE(Codice,Nome)
BRANOMUSICALE(Codice,Titolo,Autore)
ESECUZIONEBRANO(Concerto,Brano,NumOrdine)
MUSICISTA(Matricola,Nome,Cognome)
STRUMENTO(Nome)
COMPOSIZIONE(Gruppo,Componente)
SUONA(Musicista,Strumento)

Vincoli:

CONCERTO(Gruppo) $\subseteq$ GRUPPO(Nome)
CONCERTO(Sala) $\subseteq$ SALA(Codice)
REPLICA(Concerto) $\subseteq$ CONCERTO(Codice)

BranoMusicale(Autore) ⊆ Compositore(Codice)
EsecuzioneBrano(Concerto) ⊆ Concerto(Codice)
EsecuzioneBrano(Brano) ⊆ BranoMusicale(Codice)
Composizione(Gruppo) ⊆ Gruppo(Nome)
Composizione(Componente) ⊆ Musicista(Matricola)
Suona(Musicista) ⊆ Strumento(Nome)
Suona(Strumento) ⊆ Strumento(Nome)

Esercizio 3

$$\pi_{automobile}(Prenotazione) - \pi_{automobile}(\sigma_{età \geq 21}(Guidatori \bowtie_{Codice=Guidatore} Prenotazioni))$$

Esercizio 4

```
select min(Affidabilita)
from Guidatore
where Affidabilita >all (select Affidabilita
                         from Guidatore
                         where Eta <= 20)
```

Esercizio 5

```
select count(*)
from Automobili
     join Prenotazioni on Automobili.Codice = Prenotazioni.Automobile
     join Guidatori on Prenotazioni.Guidatore = Guidatore.Codice
where Eta between 20 and 40
  and (Colore = 'rosso' or Colore is null)
  and Guidatore not in (select Guidatore
                        from Automobili join Prenotazioni
                          on Automobili.Codice = Prenotazioni.Automobile
                        where Colore = 'verde')
```

Esercizio 6

Dati i volumi riportati nel testo del compito, si assume che ogni comune abbia mediamente 100 abitanti.

Costo della soluzione senza ridondanza:

Operazione	frequenza	accessi	costo
Op. 1	100	1S + 1S	600
Op. 2	10	1L + 100L	1010
Totale			**1610**

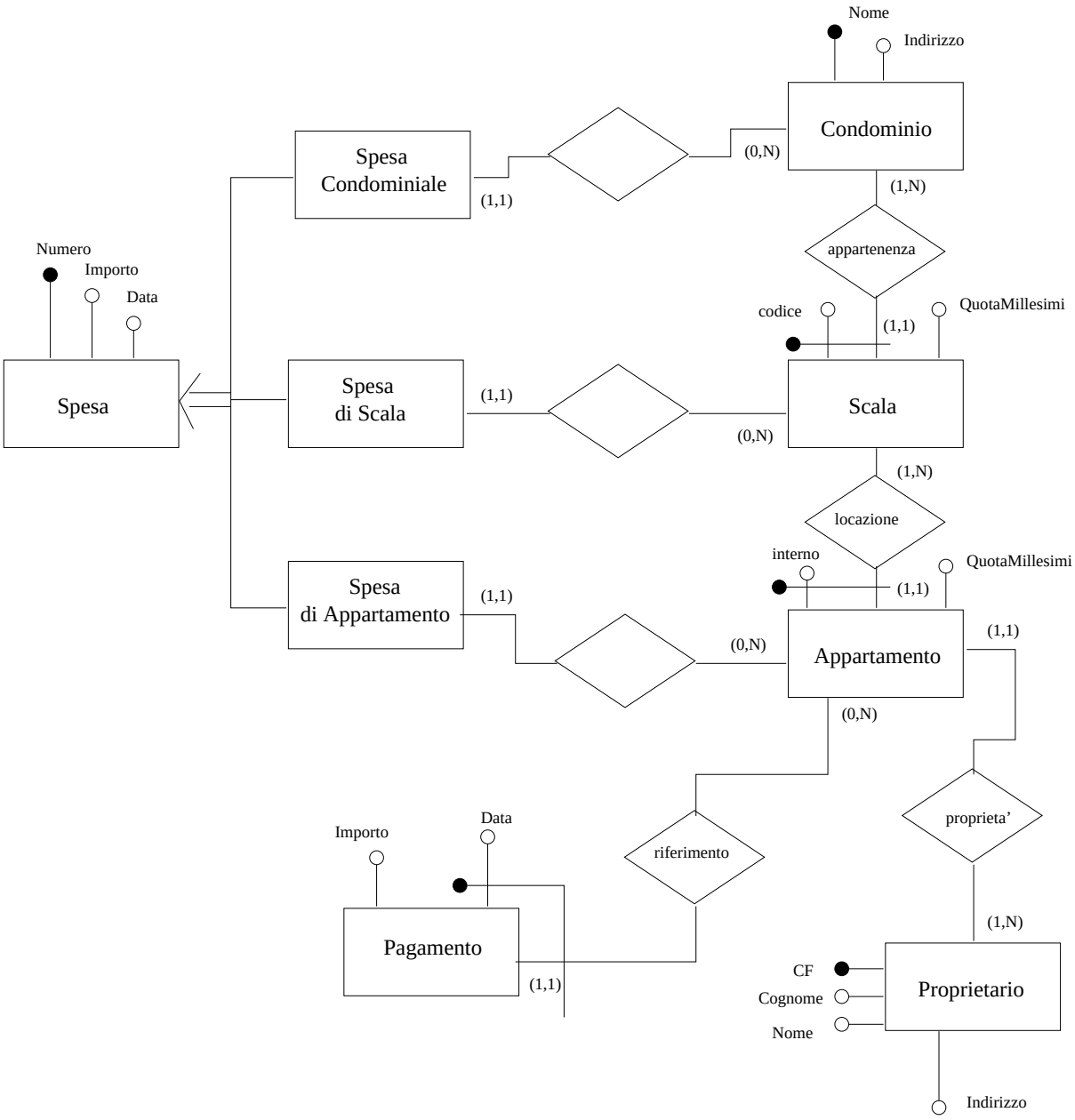
Costo della soluzione con ridondanza:

Operazione	frequenza	accessi	costo
Op. 1	100	1S + 1S + 1L + 1S	1000
Op. 2	10	1L	10
Totale			**1010**

In conclusione, l'analisi suggerisce di mantenere il dato ridondante.

Soluzione del compito del 14 luglio 2003

Esercizio 1



Si noti che la scelta di utilizzare una gerarchia di spese ci impone di inserire un identificatore specifico per l'entità Spesa (che abbiamo chiamato Numero).

Una soluzione alternativa è quella di considerare i tre tipi di spesa come entità indipendenti, cioè senza gerarchia, ciascuna con una identificazione esterna tramite la relazione con il condominio, la scala e l'appartamento, rispettivamente.

Esercizio 2

Eliminazione delle gerarchie

Per eliminare la gerarchia, si sceglie di eliminare l'entità `Spesa` e replicare i suoi attributi su ciascuna delle entità figlie.

Traduzione

Condomini(<u>Nome</u>,Indirizzo)
Scale(<u>Codice,Condominio</u>,QuotaMillesimi)

Appartamenti(Interno,Scala,Condominio,QuotaMillesimi, Proprietario)
Proprietari(CF, Nome, Cognome, Indirizzo)
SpeseCondominiali(Numero,Data, Importo,Condominio)
SpeseDiScala(Numero,Data, Importo,Scala,Condominio)
SpeseDiAppartamento(Numero,Data, Importo,Interno,Scala,Condominio)
Pagamenti(Interno,Scala,Condominio,Data,Importo)

Vincoli:

Scale(Condominio) $\subseteq$ Condomini(Nome)
Appartamenti(Scala,Condominio) $\subseteq$ Scale(Codice,Condominio)
Appartamenti(Proprietario) $\subseteq$ Proprietari(CF)
SpeseCondominiali(Condominio) $\subseteq$ Condomini(Nome)
SpeseDiScala(Scala,Condominio) $\subseteq$ Scale(Codice,Condominio)
SpeseDiAppartamento(Interno,Scala,Condominio) $\subseteq$ Appartamenti(Interno,Codice,Condominio)
Pagamenti(Interno,Scala,Condominio) $\subseteq$ Appartamenti(Interno,Codice,Condominio)

Esercizio 3

Si abbreviano i nomi degli attributi della tabella Appartamenti con le loro iniziali.

$$\pi_{nome,cognome}(Proprietari \bowtie_{CF=P} Appartamenti \bowtie_{CF=P1 \wedge C=C1 \wedge S \neq S1} \rho_{I1,S1,C1,QM1,P1 \leftarrow I,S,C,QM,P} Appartamenti)$$

Esercizio 4

```
select Scala, Condominio, sum(Importo)
from SpeseDiScala
group by Scala, Condominio
```

Esercizio 5

Vi è una inconsistenza nel testo: si parla esplicitamente di pagamenti ma dal contesto si deduce che ci si riferisce alla spese. Interpretiamo il testo sostituendo la parola pagamenti con la parola spese.

```
select SS.Scala, sum(SS.Importo)+QuotaMillesimi*(sum(SC.Importo)/1000) as Totale
from SpeseCondominiali as SC
     join SpeseDiScala as SS on SS.Condominio = SC.Condominio
     join Scale as S on S.Codice = SS.Scala and S.Condominio = SS.Condominio
where SC.Condominio = 'Le Terrazze'
group by Scala, QuotaMillesimi
```

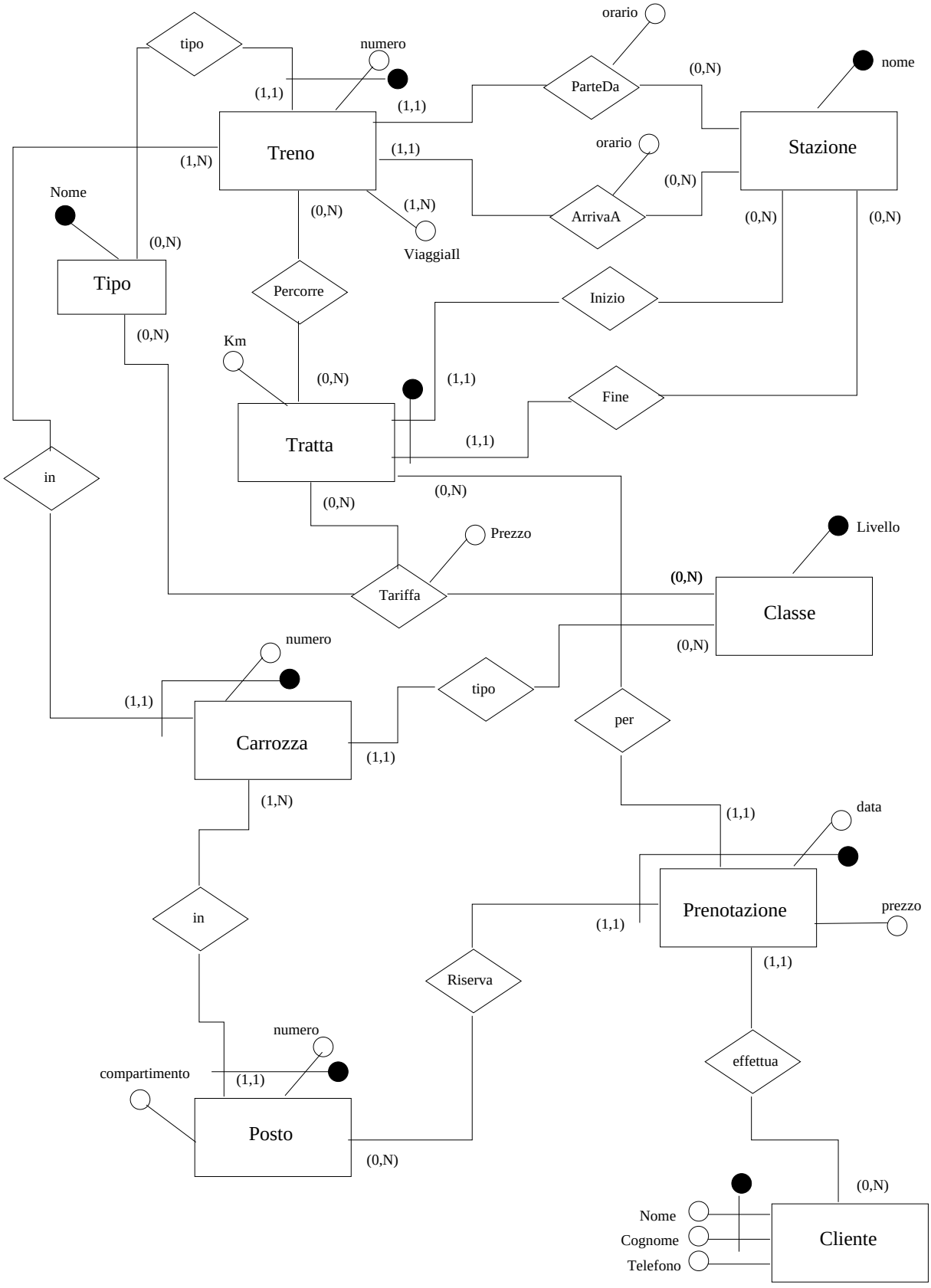
Esercizio 6

```
delete from Pagamento
where Data >= '01-jan-2001' and Data <= '31-dec-2001'
and (Interno,Scala,Condominio) in (select Interno, Scala, Condominio
                                   from appartamenti
                                   where proprietario = 'LCIPLM76A21F555A')
```

Soluzione del compito del 2 settembre 2003

Esercizio 1

E' stato necessario creare l'entità Tipo, invece di considerarlo come attributo di Treno, in quanto nelle specifiche è scritto che la tariffa dipende anche dal tipo di treno.



Esercizio 2

TRENO(Tipo,Numero,ParteDa,ArrivaA,ParteAlle,ArrivaAlle)
TIPOTRENO(Nome)
STAZIONE(Nome)
CLASSE(Livello)

ViaggiaIl(TipoTreno,NumeroTreno,Giorno)
Tratta(Inizio,Fine,Chilometraggio)
Percorre(TipoTreno,NumeroTreno,InizioTratta,FineTratta)
Carrozza(Numero,TipoTreno,NumeroTreno)
Posto(Numero,Carrozza,TipoTreno,NumeroTreno,Compartimento)
Cliente(Nome,Cognome,Telefono)
Prenotazione(InizioTratta,FineTratta,Posto,Carrozza,TipoTreno,NumeroTreno,Data,
NomeCliente,CognomeCliente, Prezzo)
Tariffa(InizioTratta,FineTratta,Classe,TipoTreno,Prezzo)

con i vincoli:

Treno(ParteDa) $\subseteq$ Stazione
Treno(ArrivaA) $\subseteq$ Stazione
Treno(Tipo) $\subseteq$ TipoTreno
ViaggiaIl(TipoTreno,NumeroTreno) $\subseteq$ Treno(Tipo,Numero)
Tratta(Inizio) $\subseteq$ Stazione
Tratta(Fine) $\subseteq$ Stazione
Percorre(TipoTreno,NumeroTreno) $\subseteq$ Treno(Tipo,Numero)
Percorre(InizioTratta,FineTratta) $\subseteq$ Tratta(Inizio,Fine)
Carrozza(TipoTreno,NumeroTreno) $\subseteq$ Treno(Tipo,Numero)
Posto(Carrozza,TipoTreno,NumeroTreno) $\subseteq$ Carrozza
Prenotazione(NomeCliente,CognomeCliente) $\subseteq$ Cliente(Nome,Cognome)
Prenotazione(InizioTratta,FineTratta) $\subseteq$ Tratta(Inizio,Fine)
Prenotazione(Posto,Carrozza,TipoTreno,NumeroTreno) $\subseteq$
Posto(Numero,Carrozza,TipoTreno,NumeroTreno)
Tariffa(InizioTratta,FineTratta) $\subseteq$ Tratta(Inizio,Fine)
Tariffa(TipoTreno) $\subseteq$ TipoTreno
Tariffa(Classe) $\subseteq$ Classe(Livello)

Esercizio 3

$$\pi_{Squadra}(Giocatore \bowtie_{NTessera=Marcatore \wedge Autogol='true'} (Gol))$$

Esercizio 4

Il testo dell'esercizio può essere interpretato in più modi. L'interpretazione che diamo è la seguente: "Trovare i portieri delle squadre che hanno subito *il massimo numero di gol segnati da una squadra* in una sola partita".

```
select NTessera, Nome, Cognome
from Giocatore join Partita on (SqCasa = Squadra
      and GolTrasf >=all (select GolCasa from partita)
      and GolTrasf >=all (select GolTrasf from partita))
         or
     (SqTrasf = Squadra
      and GolCasa >=all (select GolCasa from partita)
      and GolCasa >=all (select GolTrasf from partita))
where ruolo = 'Portiere'
```

Esercizio 5

Creiamo inizialmente una vista che memorizza il numero totale di gol di ciascun giocatore.

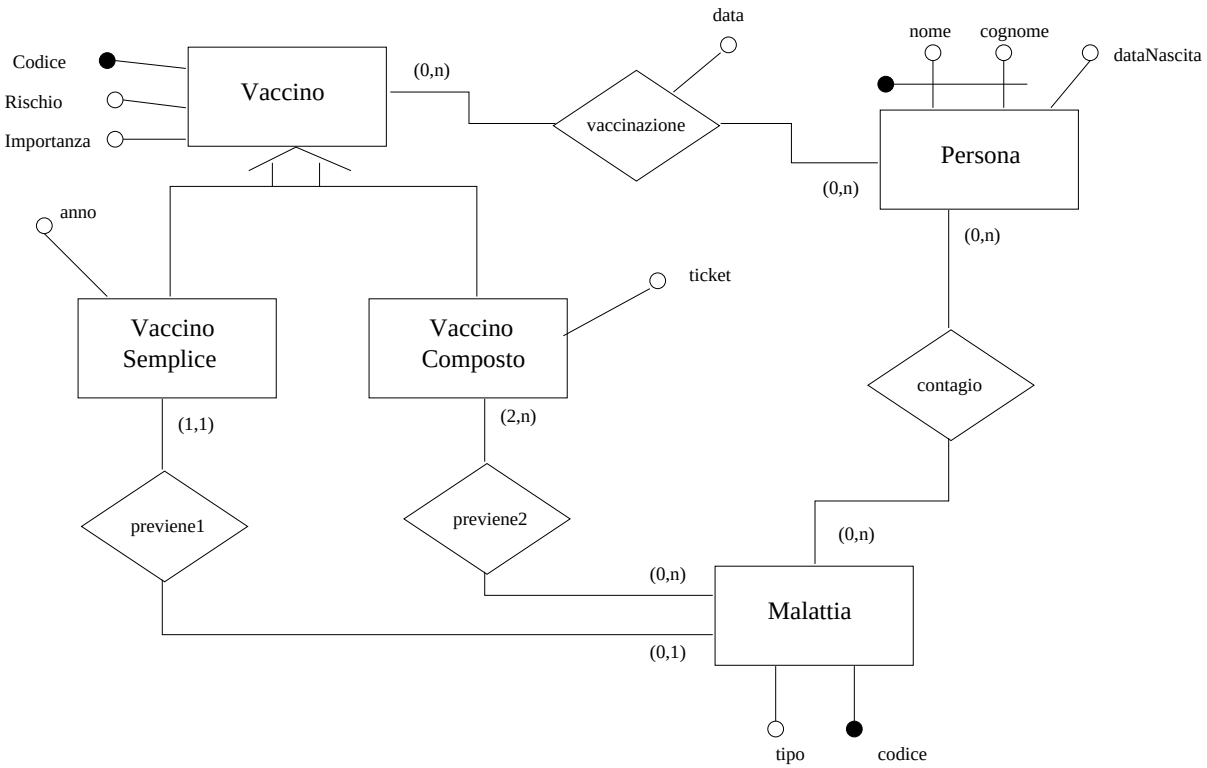
```
create view Cannonieri(Nome,Cognome,NTessera,Squadra,NumGol) as
select Nome, Cognome, NTessera, Squadra, count(*)
from Gol join Giocatore on Marcatore = NTessera
group by Squadra, Marcatore
```

Adesso esprimiamo l'interrogazione richiesta utilizzando la vista `Cannonieri`.

```
select Nome, Cognome, NTessera
from Cannonieri as C
where NumGol >=all (select NumGol
                    from Cannonieri
                    where Squadra = C.Squadra)
```

Soluzione del compito del 16 settembre 2003

Esercizio 1



Si noti che (per semplicità) si è supposto che nome e cognome identifichino una persona. In alternativa si sarebbe potuto inglobare anche la data di nascita nell'identificatore, oppure aggiungere un attributo (ad es. il codice fiscale) come identificatore unico.

Esercizio 2

Essendo sia l'entità Vaccino che le sue figlie legate da relazioni con altre entità, la soluzione più generale è quella di sostituire la gerarchia con delle relazioni. Così facendo si ottiene il seguente schema.

VACCINO(Codice,Rischio,Importanza)
VACCINOSEMPLICE(Codice,Anno,MalattiaPrevenuta)
VACCINOCOMPOSTO(Codice,Ticket)
MALATTIA(Codice,Tipo)
PERSONA(Nome,Cognome,DataNascita)
PREVENZIONECOMPOSTA(VaccinoComposto,Malattia)
CONTAGIO(NomePersona,CognomePersona,Malattia)
VACCINAZIONE(NomePersona,CognomePersona,Vaccino,Data)

con i vincoli:

VACCINOSEMPLICE(Codice) $\subseteq$ VACCINO(Codice)
VACCINOSEMPLICE(MalattiaPrevenuta) $\subseteq$ MALATTIA(Codice)
VACCINOCOMPOSTO(Codice) $\subseteq$ VACCINO(Codice)
PREVENZIONECOMPOSTA(VaccinoComposto) $\subseteq$ VACCINOCOMPOSTO(Codice)

PrevenzioneComposta(Malattia) $\subseteq$ Malattia(Codice)
Contagio(NomePersona,CognomePersona) $\subseteq$ Persona(Nome,Cognome)
Contagio(Malattia) $\subseteq$ Malattia(Codice)
Vaccinazione(NomePersona,CognomePersona) $\subseteq$ Persona(Nome,Cognome)
Vaccinazione(Vaccino) $\subseteq$ Vaccino(Codice)

Esercizio 3

```
create table Persona
(
  nome varchar(20),
  cognome varchar(20),
  DataNascita date,
  primary key (nome,cognome)
)

create table Vaccinazione
(
  NomePersona varchar(20),
  CognomePersona varchar(20),
  malattia char(5) references Malattia(codice),
  data date,
  primary key (NomePersona,CognomePersona,malattia),
  foreign key (NomePersona,CognomePersona) references Persona(nome,cognome)
)
```

Esercizio 4

$$\pi_{nome.cognome}(Giocatore \bowtie_{NTessera=Marcatore} (\pi_{Minuto,Marcatore}(\sigma_{Autogol='false'}Gol) -$$

$$\pi_{Minuto,Marcatore}(Gol \bowtie_{Minuto<Mi} \rho_{I,Mi,Ma,A \leftarrow IdPartita,Minuto,Marcatore,Autogol}Gol))$$

Esercizio 5

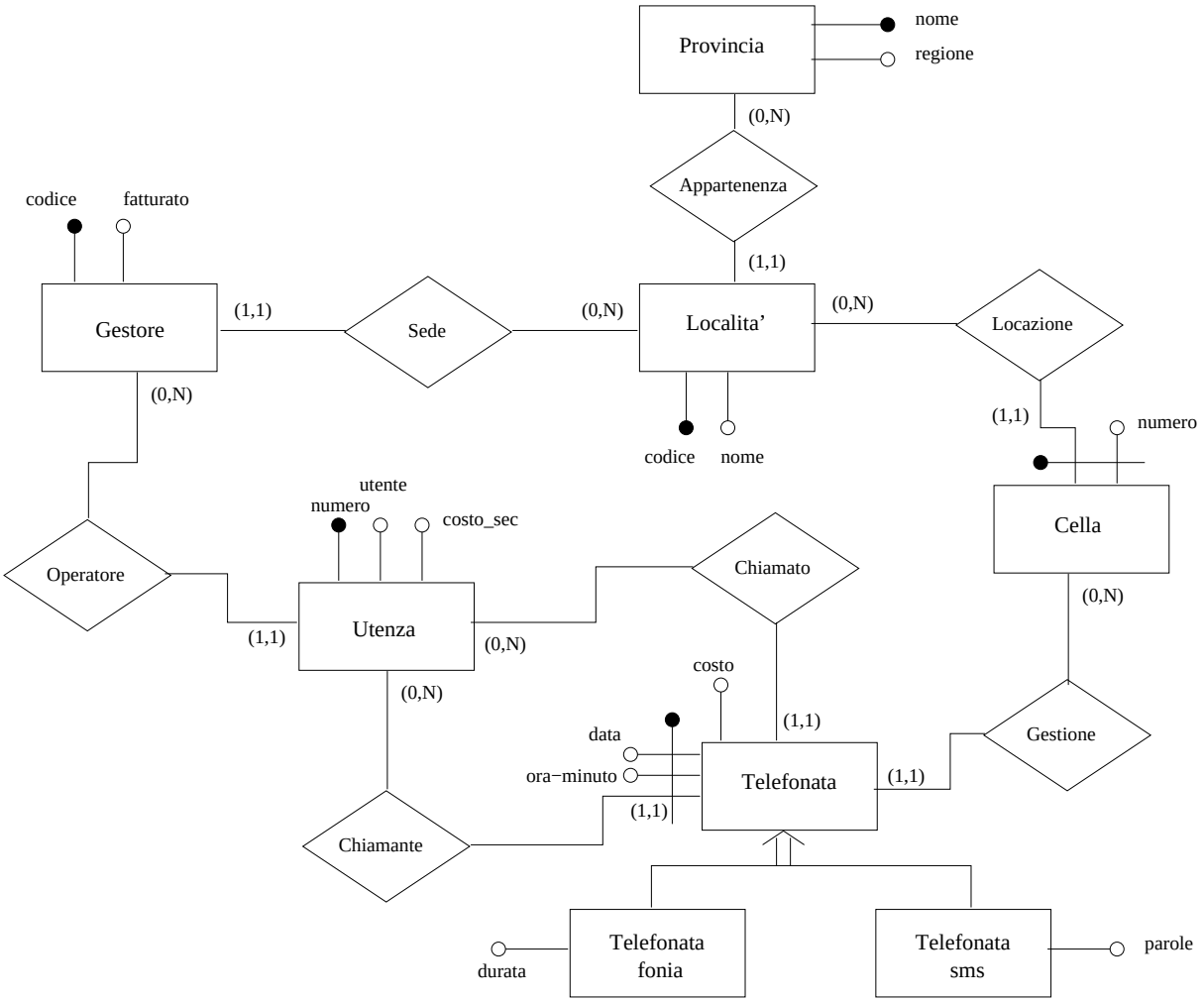
```
select distinct Ruolo
from Giocatore
where Ruolo not in (select Ruolo
                    from Giocatore join Squadra on Giocatore.Squadra = Squadra.Nome
                    where Citta = 'Milano')
```

Esercizio 6

```
select Nome, Cognome, Squadra
from Giocatore as G1
where DataNascita >=all (select DataNascita
                         from Giocatore
                         where Squadra = G1.Squadra)
```

Soluzione del compito del 15 marzo 2006

Esercizio 1



Esercizio 2

Si noti che l'attributo `costo` dell'entità Telefonata è ridondante, in quanto calcolabile a partire da `costo_sec` (entità Utenza) e `parole` o `durata` (entità Telefonata Fonia o `Telefonata Sms`). In mancanza di dati quantitativi, decidiamo di mantenere la ridondanza.

Riguardo all'eliminazione della gerarchia, le specifiche ci suggeriscono di mantenere le entità figlie e di eliminare l'entità genitore.

Province(Nome, Regione)
Località(Codice, Nome, Provincia)
Operatori(Codice, Fatturato, Sede)
Utenze(Numero, Utente, Operatore, CostoSec)
Celle(Numero, Località)
ChiamateFonia(Chiamante, Data, Orario, Chiamato, Costo, Durata, NumCella, LocalitàCella)
ChiamateSms(Chiamante, Data, Orario, Chiamato, Costo, NumParole, NumCella, LocalitàCella)

con i vincoli:

Località(Provincia) ⊆ Province(Nome)
Operatori(Sede) ⊆ Località(Codice)
Utenze(Operatore) ⊆ Operatori(Codice)
Celle(Località) ⊆ Località(Codice)
ChiamateFonia(Chiamante) ⊆ Utenze(Numero)
ChiamateFonia(Chiamato) ⊆ Utenze(Numero)

ChiamateFonia(NumCella, LocalitàCella) ⊆ Celle
ChiamateSms(Chiamante) ⊆ Utenze(Numero)
ChiamateSms(Chiamato) ⊆ Utenze(Numero)
ChiamateSms(NumCella, LocalitàCella) ⊆ Celle

Esercizio 3

$$\pi_{Nome}(AUTORE) - \pi_{Nome}(AUTORE \bowtie_{Libro=Codice} \sigma_{Prezzo>20} LIBRO)$$

Esercizio 4

```
select Nome, count(*)
from Autore join Prestito on Autore.Libro = Prestito.Libro
where DataFine is not null
  and DataFine >= DataInizio + 3
group by Nome
```

Esercizio 5

Per questo esercizio presentiamo due soluzioni alternative basate su costruzioni diverse.

Soluzione 1: basata sull'uso di `having` e `count`.

```
select Utente.Nome, Utente.Telefono
from Utente
     join Prestito on Utente.Tessera = Prestito.Utente
     join Autore on Prestito.Libro = Autore.Libro
where Autore.Nome = 'Alessandro Manzoni'
group by Utente.Nome, Utente.Telefono
having count(*) = (select count(*)
                   from Autore
                   where Nome = 'Alessandro Manzoni')
```

Soluzione 2: basata su interrogazioni annidate con negazione.

```
select Nome, Telefono
from Utente
where not exists (select *
                  from Autore
                  where Nome = 'Alessandro Manzoni'
                  and Libro not in (select Libro
                                    from Prestito
                                    where Utente = Utente.Tessera))
```

Esercizio 6

Dipendenze:

- ID_Corso → NomeCorso, Docente, CorsoDiStudi
- NomeCorso, CorsoDiStudi → ID_Corso
- Data, Aula → ID_Corso

Chiavi:

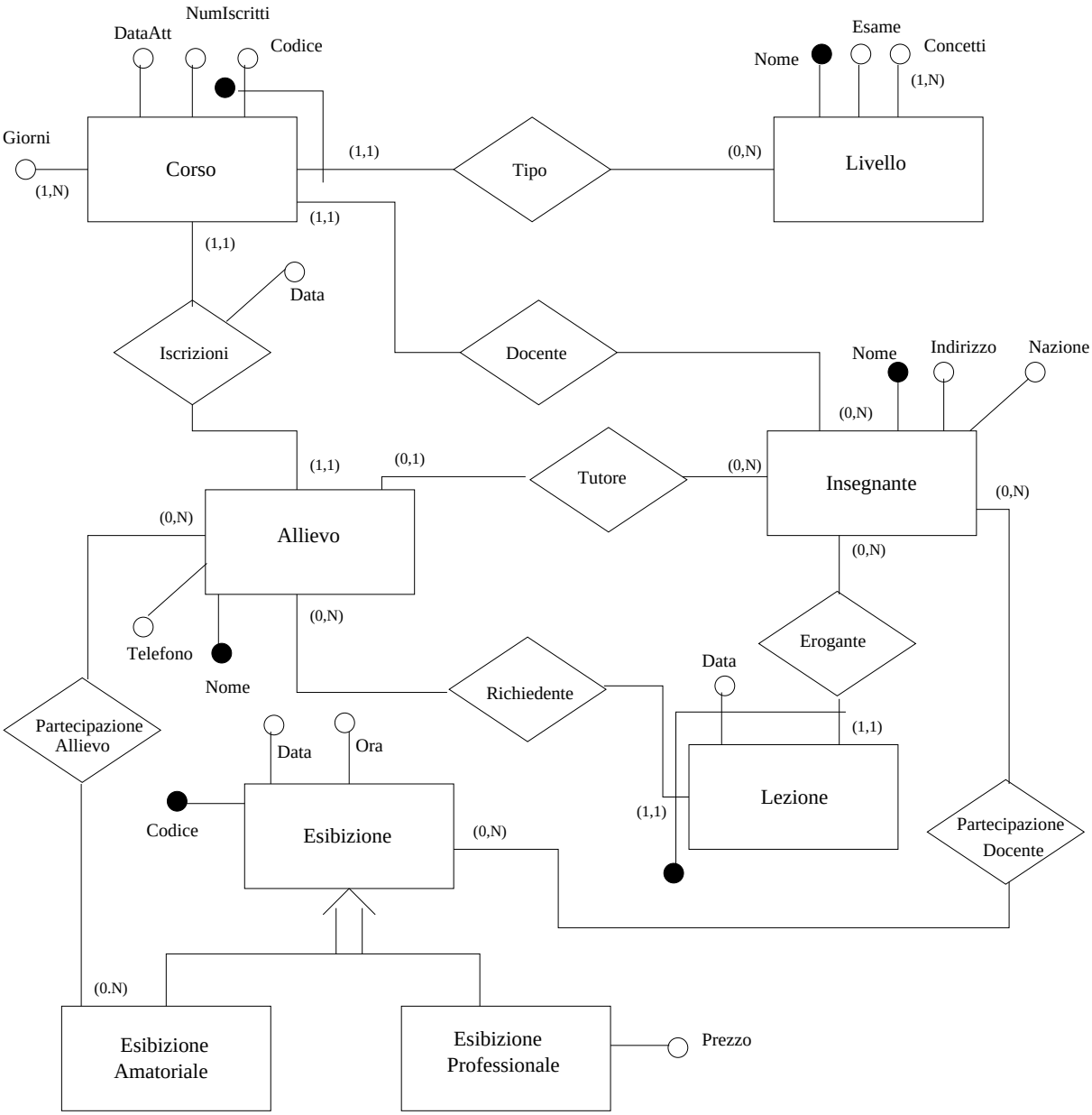
- {Data, Aula}

Tabelle:

- Esami(ID_Corso, Data, Aula)
- Corsi(ID, Nome, Docente, CorsoDiStudi)

Soluzione del compito del 21 marzo 2007

Esercizio 1



Esercizio 2

L'attributo NumIscritti dell'entità Corso è ridondante. In mancanza di informazioni quantitative, abbiamo deciso di mantenere la ridondanza. Per l'eliminazione delle gerarchia, utilizziamo la soluzione federalista.

LIVELLI(Nome, Esame)
INSEGNATI(Nome, Indirizzo, Nazione)
CORSI(Codice, Livello, NumIscritti, DataAtt, Docente)
GIORNILEZIONE(Corso, Livello,Giorno)
CONCETTI(Livello,Concetto)
ALLIEVI(Nome, Telefono, Corso, Livello)
TUTORI(Allievo, Tutore)
LEZIONI(Allievo, Insegnante, Data, Motivazione)
ESIBIZIONE(Codice, Data, Ora)
ESIBIZIONEAMATORIALE(Codice)
ESIBIZIONEPROFESSIONALE(Codice, Prezzo)
PARTECIPAZIONEALLIEVO(Allievo,Esibizione)

PartecipazioneDocente(Docente, Esibizione)

con i vincoli:

Corsi(Livello) ⊆ Livelli(Nome)
GiorniLezione(Corso, Livello) ⊆ Corsi(Codice, Livello)
Concetti(Livello) ⊆ Livelli(Nome)
Allievi(Corso, Livello) ⊆ Corsi(Codice, Livello)
Corsi(Docente) ⊆ Insegnati(Nome)
Tutori(Allievo) ⊆ Allievi(Nome)
Tutori(Tutore) ⊆ Insegnati(Nome)
Lezioni(Allievo) ⊆ Allievi(Nome)
Lezioni(Insegnante) ⊆ Insegnati(Nome)
EsibizioneAmatoriale(Codice) ⊆ Esibizione(Codice)
EsibizioneProfessionale(Codice) ⊆ Esibizione(Codice)
PartecipazioneAllievo(Allievo) ⊆ Allievi(Nome)
PartecipazioneAllievo(Esibizione) ⊆ EsibizioneAmatoriale(Codice)
PartecipazioneDocente(Docente) ⊆ Insegnati(Nome)
PartecipazioneDocente(Esibizione) ⊆ Esibizione(Codice)

Esercizio 3

La soluzione più semplice di questo esercizio comporta l'uso di join naturale.

$$\pi_{Bevitore,Birra}\sigma_{Prezzo<2}Piace \bowtie Frequenta \bowtie Vende$$

Esercizio 4

```
select Bevitore.Nome, count(distinct Birra.Nome)
from Bevitore
     join Frequenta on Bevitore.Nome = Frequenta.Bevitore
     join Vende on Frequenta.bar = Vende.Bar
     join Birra on Birra.Nome = Vende.Birra
where Birra.Nazione = Bevitore.Nazionalita
group by Bevitore.Nome
```

Esercizio 5

Riformuliamo l'interrogazione come: "tutti i bevitori esclusi coloro che frequentano un bar per cui non esiste una birra che esso vende e a loro piace".

Risolviamo prima la sottointerrogazione: " Trovare i bevitori che frequentano un bar per cui non esiste una birra che esso vende e a loro piace"

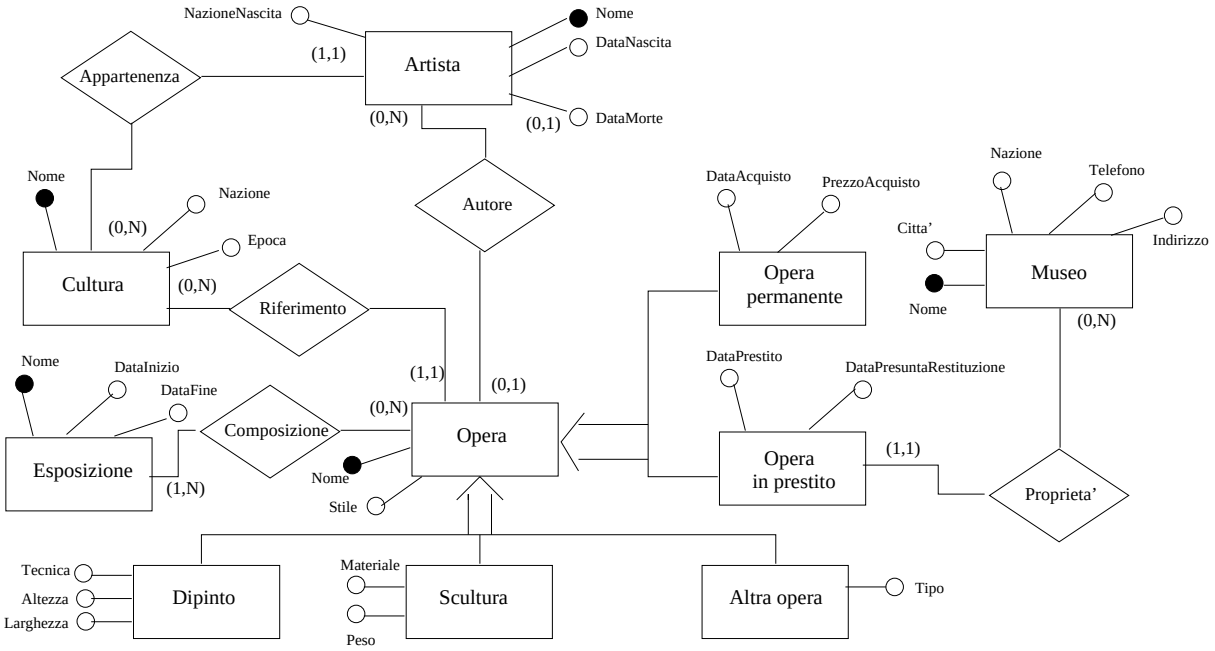
```
select Bevitore
from Frequenta
where not exists (select Piace.Birra
                  from Piace natural join Vende
                  where Frequenta.Bar = Vende.Bar
                    and Frequenta.Bevitore = Piace.Bevitore))
```

L'interrogazione richiesta si ottiene aggiungendo una interrogazione esterna come di seguito (o in alternativa utilizzando `except`.

```
select Nome
from Bevitore
where Nome not in (select Bevitore
                   from Frequenta
                   where not exists (select Piace.Birra
                                     from Piace natural join Vende
                                     where Frequenta.Bar = Vende.Bar
                                       and Frequenta.Bevitore = Piace.Bevitore))
```

Soluzione del compito del 20 aprile 2007

Esercizio 1



Esercizio 2

Entrambe le gerarchie vengono trasformate in relazioni per evitare il proliferare di valori nulli. Solo l'entità `Altra opera` viene eliminata, in quanto l'attributo `Tipo` può essere attribuito anche ai dipinti e alle sculture.

OPERE(Nome, Stile, Tipo, Cultura)
DIPINTI(Nome, Tecnica, Altezza, Larghezza)
SCULTURE(Nome, Peso, Materiale)
OPEREPERMANENTI(Nome, DataAcquisto, PrezzoAcquisto)
OPEREINPRESTITO(Nome, DataPrestito, DataPresuntaRestituzione, Museo)
CULTURE(Nome, Nazione, Epoca)
ARTISTI(Nome, DataNascita, DataMorte*, Cultura)
AUTORI(Opera, Artista)
MUSEI(Nome, Città, Nazione, Telefono, Email)
ESPOSIZIONI(Nome, DataInizio, DataFine)
COMPOSIZIONEESPOSIZIONI(Esposizione, Opera)

con i vincoli:

DIPINTI(Nome) $\subseteq$ OPERE(Nome)
SCULTURE(Nome) $\subseteq$ OPERE(Nome)
OPEREPERMANENTE(Nome) $\subseteq$ OPERE(Nome)
OPEREINPRESTITO(Nome) $\subseteq$ OPERE(Nome)
OPEREINPRESTITO(Museo) $\subseteq$ MUSEI(Nome)
OPERE(Cultura) $\subseteq$ CULTURE(Nome)
ARTISTI(Cultura) $\subseteq$ CULTURE(Nome)
AUTORI(Opera) $\subseteq$ OPERE(Nome)
AUTORI(Artista) $\subseteq$ ARTISTE(Nome)
COMPOSIZIONEESPOSIZIONI(Esposizione) $\subseteq$ ESPOSIZIONI(Nome)
COMPOSIZIONEESPOSIZIONI(Opera) $\subseteq$ OPERE(Nome)

Esercizio 3

$$\pi_{Nome}(Giocatore) - \pi_{Nome}\sigma_{DataNascita>DA}(Giocatore \bowtie_{Nome=Vincente} Partita$$

$\bowtie_{Nome=Perdente}\ \rho_{No,Na,DN \leftarrow Nome,Nazionalita,DataNascita} Giocatore)$

Esercizio 4

Se un giocatore ha vinto una partita ha certamente vinto un set, quindi per cercare coloro che non hanno vinto set è sufficiente cercare coloro che non hanno vinto partite e che non hanno vinto set *solo nelle partite perse*.

```
select Nome
from Giocatore
where Nome not in (select Vincitore
                   from Partita)
  and Nome not in (select Perdente
                   from Partita
                   join SetPartita on Codice = Partita
                   where Punti2 > Punti1)
```

Esercizio 5

Per ottenere le due informazioni richieste in una sola interrogazione risulta comodo definire due viste. Il risultato dell'interrogazione si ottiene successivamente da un join tra le viste e la tabella GIOCATORI. Si noti che è necessario utilizzare un join esterno (sinistro) in quanto altrimenti si perderebbero le tuple dei giocatori che non compaiono in una delle due viste. Nel risultato finale compaiono dei valori nulli che rappresentano il valore 0.

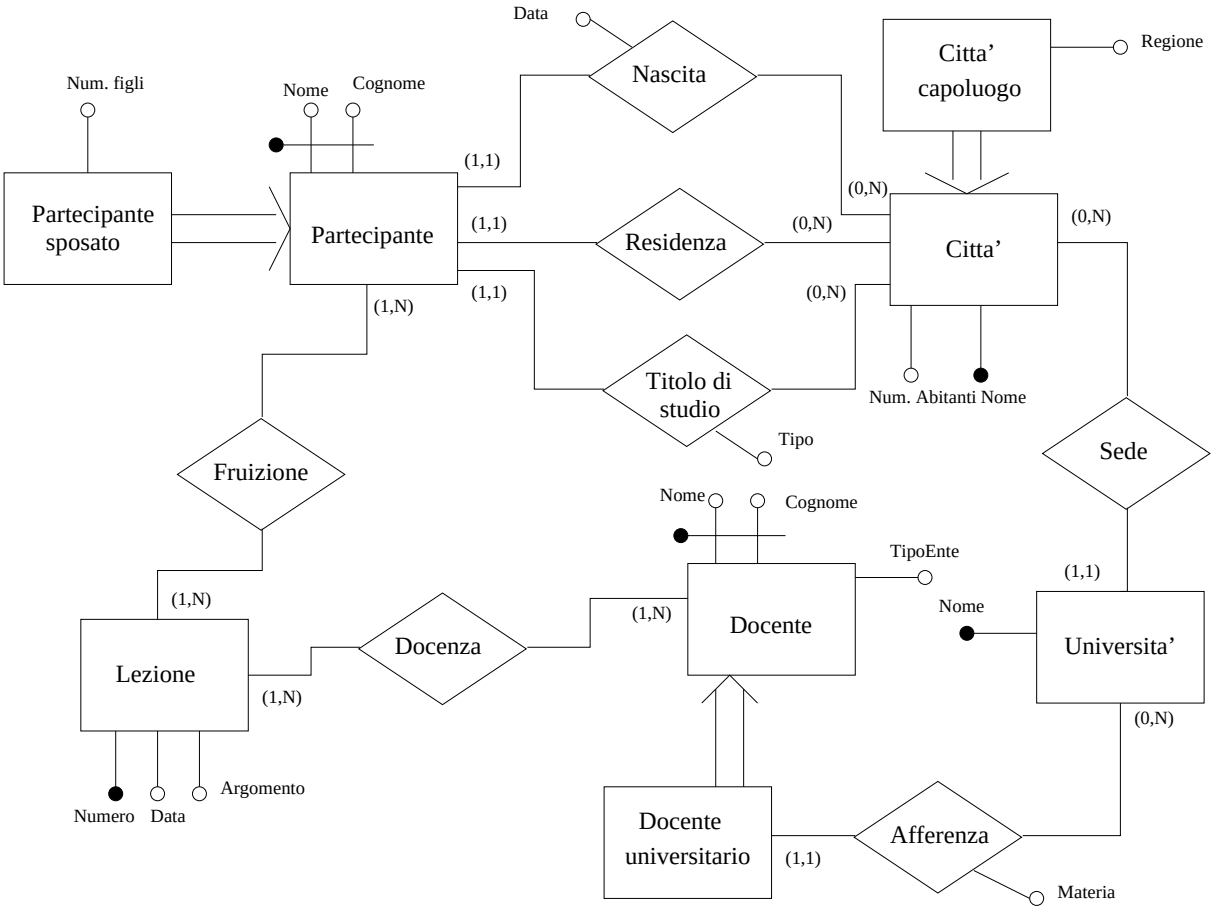
```
create View SetVincente(Giocatore,Set) as
select Vincitore, count(*)
from Partita join SetPartita on Codice = Partita
where Punti1 > Punti2
group by Vincitore

create View SetPerdente(Giocatore,Set) as
select Perdente, count(*)
from Partita join SetPartita on Codice = Partita
where Punti2 > Punti1
group by Perdente

select distinct Nome, SetVincente.Set, SetPerdente.Set
from Giocatore left join SetVincente on Nome = SetVincente.Giocatore
               left join SetPerdente on Nome = SetPerdente.Giocatore
```

Soluzione del compito del 20 marzo 2008

Esercizio 1



Esercizio 2

Dovendo evitare valori nulli, tutte e tre le gerarchie presenti vengono eliminate trasformandole in relazioni. Lo schema relazionale risultante è il seguente.

PARTECIPANTI(Nome,Cognome, DataNascita, CittàNascita, CittaResidenza,UltimoTitolo, CittàTitolo)
PARTECIPANTISPOSATI(Nome,Cognome,NumeroFigli)
LEZIONI(Numero,Data,Argomento)
FRUIZIONE(Lezione,NomePartecipante,CognomePartecipante)
DOCENTI(Nome,Cognome,TipoEnte)
DOCENTIUNIVERSITARI(Nome,Cognome,Università,Materia)
DOCENZA(Lezione,NomeDocente,CognomeDocente)
UNIVERSITÀ(Nome,Sede)
CITTÀ(Nome,NumAbitanti)
CAPOLUOGO(Nome,Regione)

con i vincoli:

PARTECIPANTI(CittàNascita) $\subseteq$ CITTÀ(Nome)
PARTECIPANTI(CittaResidenza) $\subseteq$ CITTÀ(Nome)
PARTECIPANTI(CittàTitolo) $\subseteq$ CITTÀ(Nome)
FRUIZIONE(Lezione) $\subseteq$ LEZIONI(Numero)
FRUIZIONE(NomePartecipante,CognomePartecipante) $\subseteq$ PARTECIPANTI(Nome,Cognome)
DOCENZA(Lezione) $\subseteq$ LEZIONI(Numero)
DOCENZA(NomeDocente,CognomeDocente) $\subseteq$ DOCENTI(Nome,Cognome)

PartecipantiSposati(Nome,Cognome) ⊆ Partecipanti(Nome,Cognome)

DocentiUniversitari(Nome,Cognome) ⊆ Docenti(Nome,Cognome)
DocentiUniversitari(Università) ⊆ Università(Nome)
Università(Sede) ⊆ Città(Nome)
Capoluogo(Nome) ⊆ Città(Nome)

Esercizio 3

Per questo esercizio esistono due soluzioni alternative.
Soluzione 1: utilizzando gli operatori di aggregazione */

```
select Persona
from Cura join Cani on Cane = Nome
where età = 5
group by Persona
having count(*) = (select count(*)
                   from Cani
                   where età = 5)
```

Soluzione 2: usando le interrogazioni annidate

```
select Nome
from Persone
where not exists (select *
                  from Cani
                  where Età = 5
                  and Nome not in (select Cane
                                   from Cura
                                   where Persona = Persone.Nome))
```

Esercizio 4

```
select Persone.Età, avg(Cani.Età)
from Persone join Cura on Persone.nome = Persona
             join Cani on Cane = Cani.Nome
where Età < 18
group by Persone.Età
```

Si noti che nella soluzione proposta se un cane viene curato da più persone la sua età contribuisce alla media più volte. Una soluzione che conta ciascun cane una volta sola necessita della definizione di una vista composta dall'età del padrone, quella del cane e il nome di quest'ultimo. La sua realizzazione viene lasciata per esercizio allo studente.

Esercizio 5

a	b
1	5

Esercizio 6

Le dipendenze funzionali sono:

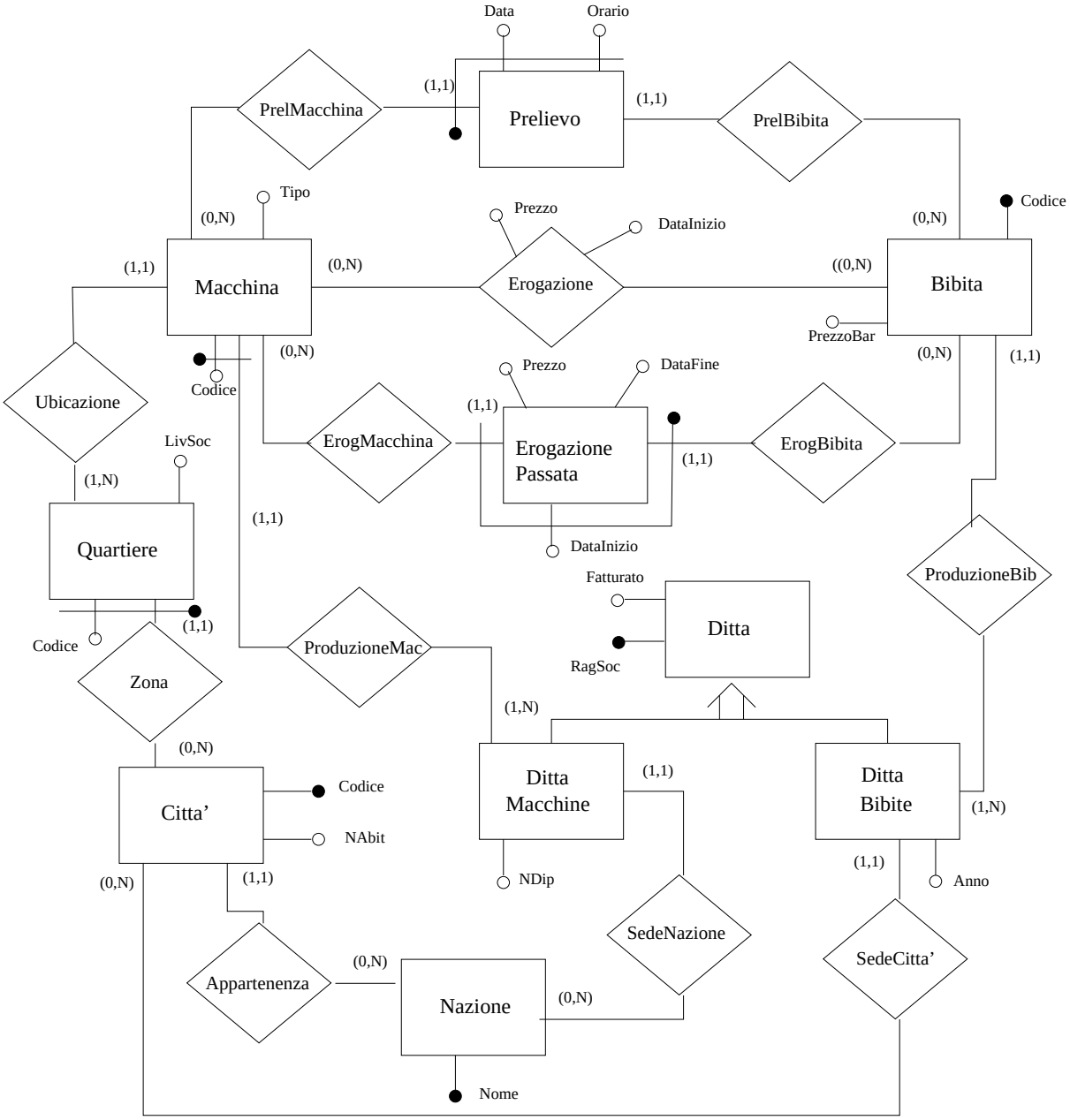
- Progetto, Pezzo → Codice
- Dipartimento, Fornitore → Pezzo
- Progetto → Fornitore

La prima dipendenza ha a sinistra una chiave, quindi non richiede decomposizioni. Effettuiamo quindi due decomposizioni successive sulla base degli attributi delle altre due dipendenze. Lo schema risultante è il seguente.

CONTRATTI(Codice, Progetto, Dipartimento, Qta, Valore)
FORNITURAPEZZI(Dipartimento, Fornitore, Pezzo)
FORNITUREPROGETTI(Progetto, Fornitore)

Soluzione del compito del 9 aprile 2008

Esercizio 1



Esercizio 2

La gerarchia riguardante le ditte viene risolta eliminando l'entità Ditta e spostando tutte gli attributi sulle entità figlie.

Macchine(Codice,Produttore,Tipo,Quartiere,Città)
Bibite(Codice, PrezzoBar, Produttore)
ErogazioniAttuali(Macchina,ProdMacchina,Bibita,DataInizio,Prezzo)
ErogazioniPassate(Macchina,ProdMacchina,Bibita,DataInizio,DataFine,Prezzo)
Prelievo(Macchina,ProdMacchina,Data,Orario,Bibita)
DitteMacchine(RagSociale,Fatturato,NDip,Nazione)
DitteBibite(RagSociale,Fatturato,Anno,Sede)
Quartieri(Codice,Città,LivSoc)
Città(Codice,NAbitanti,Nazione)
Nazioni(Nome)

con i vincoli:

Macchine(Quartiere,Città) ⊆ Quartieri(Codice,Città)
Macchine(Produttore) ⊆ DitteMacchine(RagSociale)
Bibite(Produttore) ⊆ DitteBibite(RagSociale)
ErogazioniAttuali(Macchina,ProdMacchina) ⊆ Macchine(Codice,Produttore)
ErogazioniAttuali(Bibita) ⊆ Bibite(Codice)
ErogazioniPassate(Macchina,ProdMacchina) ⊆ Macchine(Codice,Produttore)
ErogazioniPassate(Bibita) ⊆ Bibite(Codice)
Prelievo(Macchina,ProdMacchina) ⊆ Macchine(Codice,Produttore)
Prelievo(Bibita) ⊆ Bibite(Codice)
DitteMacchine(Nazione) ⊆ Nazioni(Nome)
DitteBibite(Sede) ⊆ Città(Codice)
Quartieri(Città) ⊆ Città(Codice)
Città(Nazione) ⊆ Nazioni(Nome)

Esercizio 3

$\pi_{CV,Artista}\sigma_{AV \geq Anno+10}(BranoMusicale \bowtie_{Codice=Brano} Associazione \bowtie_{Video=CV} \rho_{CV,AV \leftarrow Codice,Anno} Video)$

Esercizio 4

```
select Codice
from BranoMusicale
where Codice not in (select Brano
                     from Associazione join Video on Video = Codice
                     where Artista != Regista)
```

Esercizio 5

```
select Artista, count(distinct Codice), count(distinct Video)
from BranoMusicale left join Associazione on Codice = Brano
group by Artista
```

Si noti l'utilizzo del join esterno (sinistro) che permette di selezionare anche i brani che non hanno video associati. Si noti inoltre la necessità dell'uso di `distinct` per evitare di contare più volte brani con due o più video associati a due o più brani.

Esercizio 6

Dalle condizioni riportate nel testo si deducono le seguenti dipendenze funzionali:

- Gabbia → CodAddetto, GiornoPulizia
- GiornoSpettacolo → OraSpettacolo

Da ulteriori considerazioni sul domino si possono dedurre le seguenti dipendenze aggiuntive:

- CodAddetto → NomeAddetto

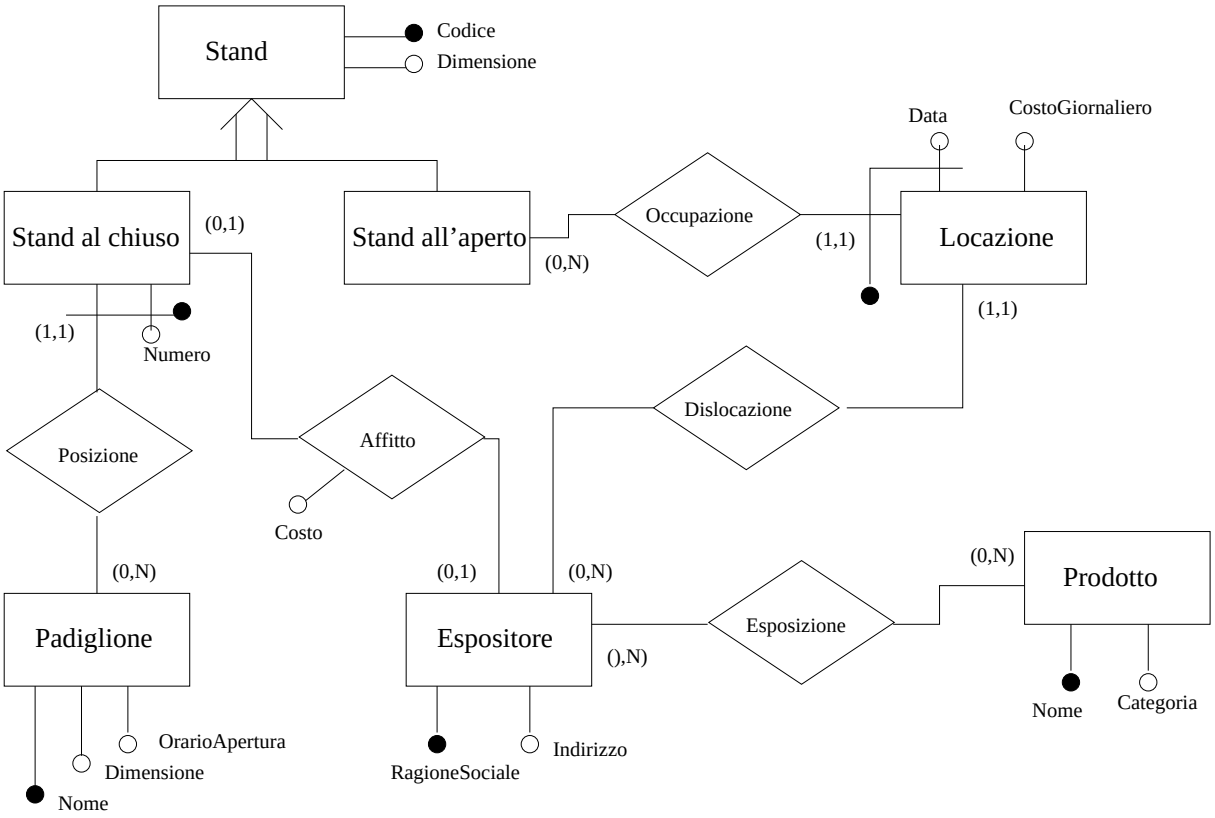
- CodAnimale → GenereAnimale, Gabbia

Una decomposizione senza perdita in BCNF che preserva le dipendenze è la seguente:

ANIMALI(CodAnimale,GenereAnimale,Gabbia)
PULIZIAGABBIE(Gabbia,CodAddetto,GiornoPulizia)
ADDETTIPULIZIA(CodAddetto,NomeAddetto)
ORARISPETTACOLI(GiornoSpettacolo,OraSpettacolo)
PARTECIPAZIONESPETTACOLI(CodAnimale,GiornoSpettacolo)

Soluzione del compito del 18 dicembre 2009

Esercizio 1



Esercizio 2

La gerarchia riguardante gli stand viene eliminata trasformandola in relazioni. Per la tabella STANDALCHIUSO scegliamo come chiave primaria il codice, essndo questo composto da un solo attributo.

Per evitare valori nulli, la relazione affitto esclusivo è tradotta con una tabella a parte. Come chiave primaria scegliamo il codice delle stand.

STAND(Codice,Dimensione)
PADOGLIONI(Nome,Dimensione,OrarioApertura)
STANDALCHIUSO(Codice,Numero,Padiglione)
STANDALLAPERTO(Codice)
ESPOSITORI(RagioneSociale,Indirizzo)
PRODOTTI(Nome,Categoria)
LOCAZIONI(Stand,Data,Espositore,CostGiornaliero)
AFFITTI(Stand,Espositore,Costo)

Esposizioni(Espositore,Prodotto)

con i vincoli:

StandAlChiuso(Codice) ⊆ Stand(Codice)
StandAllAperto(Codice) ⊆ Stand(Codice)
StandAlChiuso(Padiglione) ⊆ Padoglioni(Nome)
Locazioni(Stand) ⊆ StandAllAperto(Codice)
Locazioni(Espositore) ⊆ Espositori(RagioneSociale)
Affitti(Stand) ⊆ StandAlChiuso(Codice)
Affitti(Espositore) ⊆ Espositori(RagioneSociale)
Esposizioni(Espositore) ⊆ Espositori(RagioneSociale)
Esposizioni(Prodotto) ⊆ Prodotti(Nome)

Esercizio 3

$$\Pi_{CF,Nome,Cognome}\sigma_{Codice \neq Reparto}(Reparti \bowtie_{Direttore=CF} Impiegati)$$

Esercizio 4

Per questo esercizio proponiamo due soluzioni alternative

```
/* Soluzione n. 1 */
select Codice, Budget
from Reparti
where Codice not in (select I1.Reparto
                     from Impiegati as I1 join Impiegati as I2 on I1.Reparto = I2.Reparto
                     where I1.Genere != I2.Genere)

/* Soluzione n. 2 */
select Codice, Budget
from Reparti
group by Codice, Budget
having count(distinct Genere) = 1
```

Esercizio 5

```
select Reparto, count(*)
from Impiegati
where Stipendio > 30000
and Genere = 'Femminile'
and Reparto in (select Reparto
                from Impiegati
                group by Reparto
                having count(*) > 5)
group by Reparto
```

Esercizio 6

Dalle condizioni riportate nel testo si deducono le seguenti dipendenze funzionali:

- IdTerreno → Comune, NumTerreno, Dimensione, PrezzoTotale
- Comune, NumTerreno → IdTerreno
- Comune, Dimensione → PrezzoTotale
- Comune, PrezzoTotale → Dimensione
- Proprietario → IdTerreno

L'unica chiave è Proprietario. Una decomposizione in BCNF (non richiesta per il compito) è la seguente:

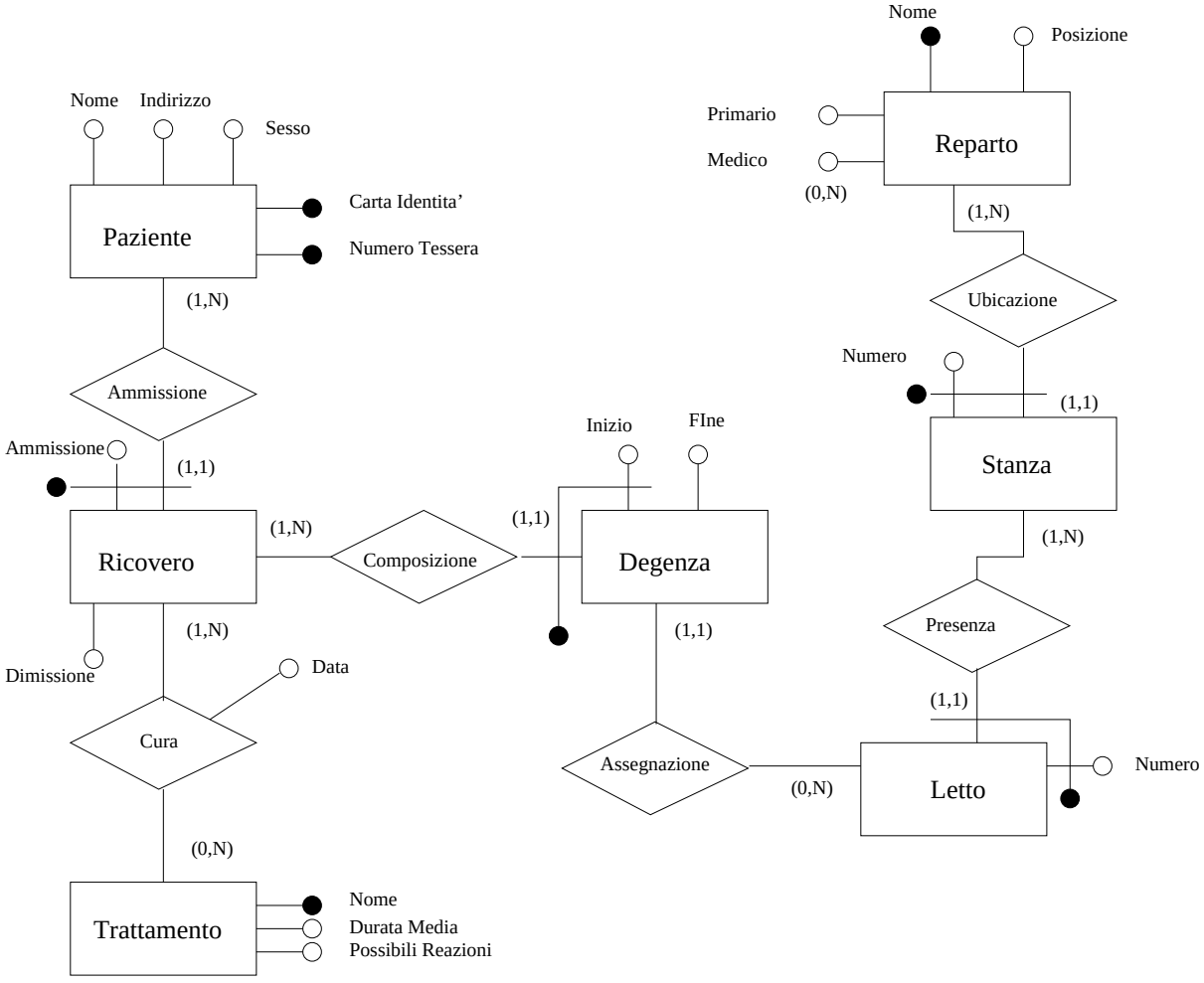
PROPRIETÀ(Proprietario,IdTerreno)
TERRENI(IdTerreno, Comune, NumTerreno, Dimensione)
PREZZI(Comune, Dimensione, Prezzo)

Soluzione del compito del 27 febbraio 2013

Esercizio 1

Si è scelto di separare in entità distinte il ricovero completo (entità Ricovero) dalle degenze (entità Degenza), che rappresentano le permanenze nei singoli letti. Questo comporta che la data di fine della singola degenza è in realtà ridondante, in quanto si può desumere dalla data di inizio della degenza successiva (o dalla data di dimissione).

Si noti che l'entità Paziente ha due identificatori. Viene scelto come primario il numero di tessera, in qualto più pertinente al dominio di interesse.



Esercizio 2

Non essere presenti gerarchie, la progettazione logiche non presenta particolari difficoltà.

PAZIENTI(NumeroTessera,CartaIdentità,Nome, Indirizzo, Sesso)
REPARTI(Nome,Posizione, NomePrimario)
MEDICI(Nome,Reparto)
STANZE(Numero, Reparto)

LETTI(Numero, Stanza, Reparto)
RICOVERI(Paziente,DataAmmissione, DataDimissione)
DEGENZE(Paziente, AmmissionePaziente, Inizio, Fine, Letto, Stanza Reparto)
TRATTAMENTI(Nome,DurataMedia, PossibiliReazioni)
CURE(Paziente, AmmissionePaziente, Trattamento,Data)

con i vincoli:

MEDICI(Reparto) $\subseteq$ REPARTI(Nome)
STANZE(Reparto) $\subseteq$ REPARTI(Nome)
LETTI(Stanza, Reparto) $\subseteq$ STANZE(Numero, Reparto)
RICOVERI(Paziente) $\subseteq$ PAZIENTI(NumeroTessera)
DEGENZE(Paziente, AmmissionePaziente) $\subseteq$ RICOVERI(Paziente,DataAmmissione)
DEGENZE(Letto, Stanza, Reparto) $\subseteq$ LETTI(Numero, Stanza, Reparto)
CURE(Paziente, AmmissionePaziente) $\subseteq$ RICOVERI(Paziente,DataAmmissione)
CURE(Trattamento) $\subseteq$ TRATTAMENTI(Nome)

Esercizio 3

Per area della tesi, intendiamo l'area del relatore (senza considerare quella dell'eventuale correlatore). Sotto questa ipotesi, la soluzione è la seguente.

$\pi_{Area}(Dipartimenti) - \pi_{Area}(\sigma_{Voto<78} Dipartimenti \bowtie_{Sigla=Dipartimento} Docenti \bowtie_{Matricola=Relatore} Tesi)$

Esercizio 4

```
select Studenti.Matricola, Studenti.Nome, Studenti.Cognome
from Studenti
     join Tesi on Studenti.Matricola = Studente
     join Docenti on Relatore = Docenti.Matricola
     join Dipartimenti on Dipartimento = Sigla
where Data between '01-jan-2011' and '31-dec-2011'
  and Area = 'Ingegneria'
  and (Correlatore is null
       or Correlatore in (select Matricola
                          from Docenti join Dipartimenti on Dipartimento = Sigla
                          where Area = 'Ingegneria'))
```

Esercizio 5

```
Select Nome, Cognome
from Docenti
where Matricola not in (select Matricola
                        from Docenti join Tesi on Matricola = Relatore
                        where Correlatore is null
                          and voto < 100)
```