ESERCITAZIONI: II PROVA SCRITTA DI INFORMATICA

(1) Una casa editrice desidera archiviare in un database le informazioni riguardanti gli **abbonamenti** alle **riviste** ed ai **giornali** pubblicati tra il 1995 ed il 2006

Per ogni **abbonato** si richiede di memorizzare i dati anagrafici, per ogni **abbonamento** la data ed il periodo di validità (trimestrale, semestrale, annuale). Bisogna considerare che gli abbonati possono avere abbonamenti anche per più pubblicazioni.

Per ogni giornale o rivista occorre archiviare il titolo, la periodicità (quotidiano, settimanale, mensile, il prezzo dell'abbonamento e gli argomenti trattati. Inoltre deve essere mantenuto un indice con i titoli dei principali articoli pubblicati ed a ciascun articolo deve essere associata la pubblicazione in cui è comparso.

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Dato il titolo di una pubblicazione, ricercare gli articoli pubblicati in un determinato anno;

Q2: Dato il titolo di una pubblicazione, ricercare gli abbonati annuali;

Q3: Dato il nominativo di un abbonato, stabilire a quante riviste è abbonato;

Q4: Dato un argomento, elencare le pubblicazioni in cui è trattato;

Q5: Riportare per ogni pubblicazione il numero di abbonamenti;

Q6: Visualizzare i giornali con almeno 5000 abbonati annuali;

Q7: Dati i titoli di due pubblicazioni, visualizzarne gli abbonati comuni;

Q8: Dato il titolo di una pubblicazione, elencare le pubblicazioni che trattano i suoi stessi argomenti.

e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

Premessa: *Una **base di dati** (in inglese **database**) può essere considerata come una raccolta di dati progettati per essere fruiti in maniera ottimizzata da differenti applicazioni e da utenti diversi.*

Una base di dati, per poter essere definita tale, deve essere:

*- **sicura**: ossia deve essere progettata in modo da impedire che essa possa essere danneggiata da eventi accidentali (come crash di sistema) o da accessi non autorizzati;*

*- **integra**: ossia deve essere garantito che le operazioni effettuate da utenti autorizzati non possano provocare una perdita di consistenza dei dati;*

*- **consistente**: ossia i dati in essa contenuti devono essere significativi ed effettivamente utilizzabili nelle applicazioni dell'azienda per cui è stata progettata;*

*- **condivisibile**: ossia applicazioni ed utenti diversi devono poter accedere, secondo opportune modalità, ai dati comuni;*

*- **persistente**: ossia deve avere un tempo di vita che non è limitato a quello delle singole esecuzioni dei programmi che la utilizzano (il contrario dei dati gestiti in memoria centrale)*

*- **efficiente**: l'utilizzo delle risorse deve essere ottimizzato riguardo ai ben noti parametri tempo (efficiente utilizzo della CPU) e spazio (efficiente uso della memoria).*

L'insieme di attività che costituiscono l'attività di progettazione di una base di dati consta di tre distinte attività di progettazione:

*1) **progettazione concettuale**: ha lo scopo di costruire e definire una rappresentazione corretta e completa della realtà di interesse concettuale in modo astratto ed indipendente dal DBMS che si intenderà utilizzare.*

L'input di tale fase è il documento delle specifiche formali (nel nostro caso il testo dell'esercizio)

L'output di tale fase è uno schema concettuale ossia una rappresentazione astratta (indipendente dal DBMS) ed il più possibile formale della realtà (un esempio di output che vedremo in seguito è il diagramma E/R).

*2) **progettazione logica**: ha lo scopo di trasformare lo schema concettuale (ancora astratto e indipendente da un DBMS) in uno schema logico ovvero in una rappresentazione efficiente rispetto alle strutture di un DBMS relativamente ad un ben definito modello dati (un esempio è una descrizione tramite tabelle del modello relazionale).*

L'input di tale fase è il diagramma ER della fase di progettazione concettuale.

L'output di tale fase è uno schema logico riassumibile con relazioni rappresentate da tabelle logiche

*3) **progettazione fisica**: ha lo scopo di implementare lo schema logico definendo tutti gli aspetti fisici di memorizzazione e rappresentazione in memoria di massa.*

L'input di tale fase è lo schema logico (ossia le tabelle logiche) individuate nella fase di progettazione logica.

L'output di tale fase è l'implementazione in memoria di massa di tali tabelle (tabelle fisiche)

Da un'attenta lettura delle specifiche, si evidenzia che sono richieste le seguenti attività:

- gestione degli abbonati e dei loro abbonamenti;
- gestione delle pubblicazioni della casa editrice e degli argomenti trattati;
- raccolta degli articoli pubblicati

Più nel dettaglio volendo evidenziare quali sono le *specifiche sui dati* e quali le *specifiche sulle funzioni* possiamo ipotizzare che sarà necessario raccogliere informazioni relative:

- agli *abbonati* alle pubblicazioni;
- alle *pubblicazioni* della casa editrice (distinguendo tra *giornali* e *riviste*);
- agli *argomenti* trattati;
- agli *articoli* pubblicati.

Abbiamo così ottenuto un primo elenco di entità che dovranno entrare a far parte dello schema concettuale della base dati

A partire da questo elenco, associamo ad ogni entità individuata i corrispondenti attributi, attingendo sempre le informazioni dalle specifiche fornite:

- a ciascun abbonato deve essere associato un *Cognome*, un *Nome*, una *DataNascita*, un *Indirizzo*, un *Cap* ed una *Città* di residenza (anagrafica breve);
- ciascuna pubblicazione deve essere caratterizzata da un *Titolo*, dalla *Periodicità*, dal *Tipo* di pubblicazione (mapping di un'ISA con accorpamento delle entità figlie nell'entità padre – ISA totale), dal *Costo Trimestrale, Costo Semestrale, Costo Annuale* dell'abbonamento;
- ciascun argomento è caratterizzato da una sua *descrizione*;
- a ciascun articolo è associato un *titolo*, un *testo* ed un'eventuale *fotografia.*

Passiamo ora ad esaminare quali sono le associazioni tra le entità ipotizzate, individuando per ciascuna di esse la molteplicità e la totalità/parzialità in base alle caratteristiche di funzionalità evidenziate dalle specifiche nonché gli eventuali attributi:

- tra le entità *Abbonato* e *Pubblicazione* (entità padre dell'ISA) esiste un'associazione *"SiAbbona"* di molteplicità N:N, totale in entrambi i versi, in quanto *"un abbonato deve essere abbonato ad una o più pubblicazioni (supponendo che chi non rinnovi alcun abbonamento non sia più mantenuto nell'entità Abbonato) e viceversa una pubblicazione deve avere uno o più abbonati contemporaneamente";*
- tra le entità *Pubblicazione* ed *Argomento* esiste un'associazione *"Tratta"* di molteplicità N:N, totale in entrambi i versi, in quanto *"una pubblicazione deve trattare uno o più argomenti e viceversa un argomento deve essere trattato in una o più pubblicazioni"*;
- tra le entità *Pubblicazione* ed *Articolo* esiste un'associazione *"Pubblica"* di molteplicità 1:N, totale in entrambi i versi, in quanto *"una pubblicazione deve pubblicare uno o più articoli e viceversa un articolo deve essere pubblicato in una pubblicazione"*;

Sulla base della precedente analisi e delle specifiche rappresentiamo lo schema concettuale attraverso un diagramma E/R mostrato sotto.

Oltre alle specifiche fornite abbiamo introdotto alcune ipotesi aggiuntive:

(*) nello schema concettuale è stata individuata una *chiave primaria* per ogni entità;

(*) l'attributo *Fotografia* (opzionale) dell'entità Articolo conterrà il "path" completo di nome file dell'immagine che eventualmente accompagna il testo;

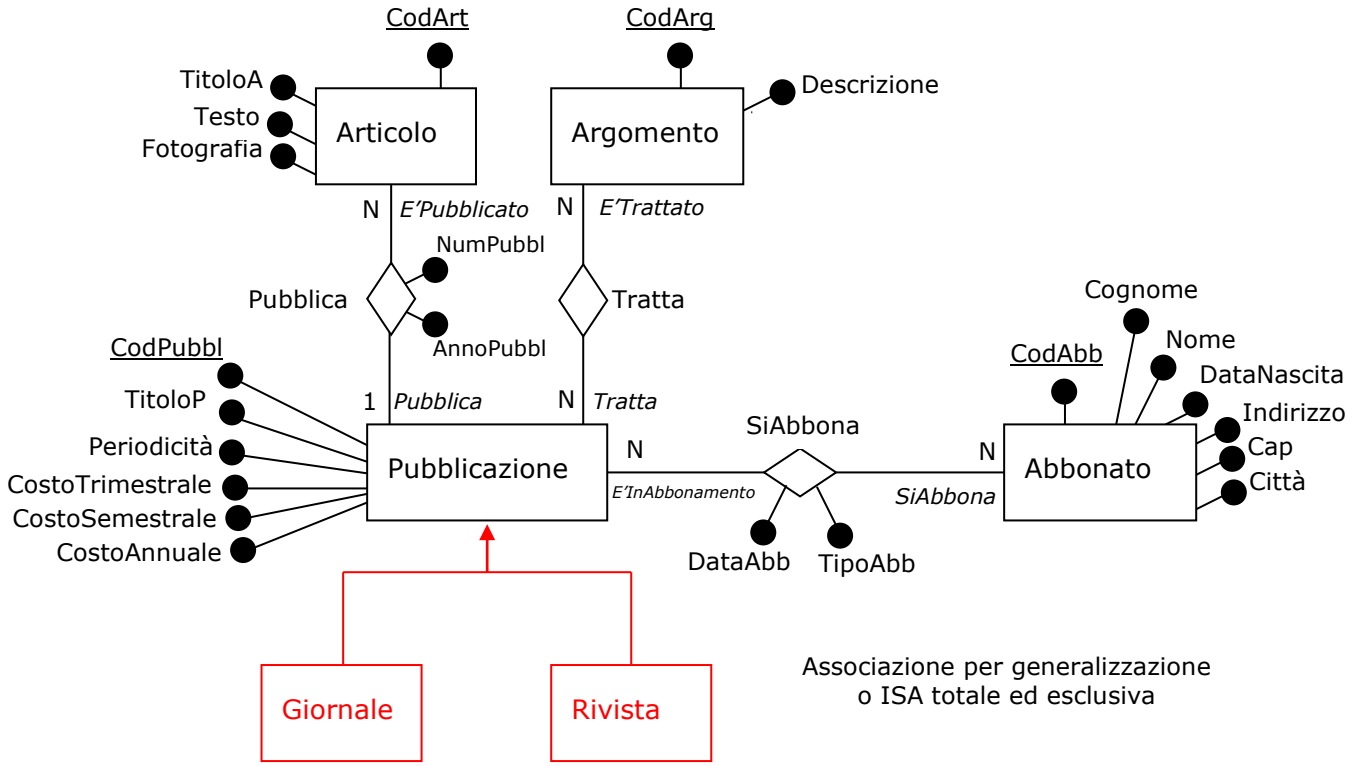
(*) gli attributi *DataAbb* e *TipoAbb* dell'associazione "SiAbbona" permettono di sapere per ogni abbonamento da quando è partito e di che tipo è (trimestrale, semestrale, annuale);

(*) gli attributi *NumPubbl* e *AnnoPubbl* dell'associazione "Pubblica" permettono di conoscere su quale numero della pubblicazione è stato pubblicato l'articolo ed in quale anno, in modo da realizzare *l'indice* degli articoli richiesto dalle specifiche;

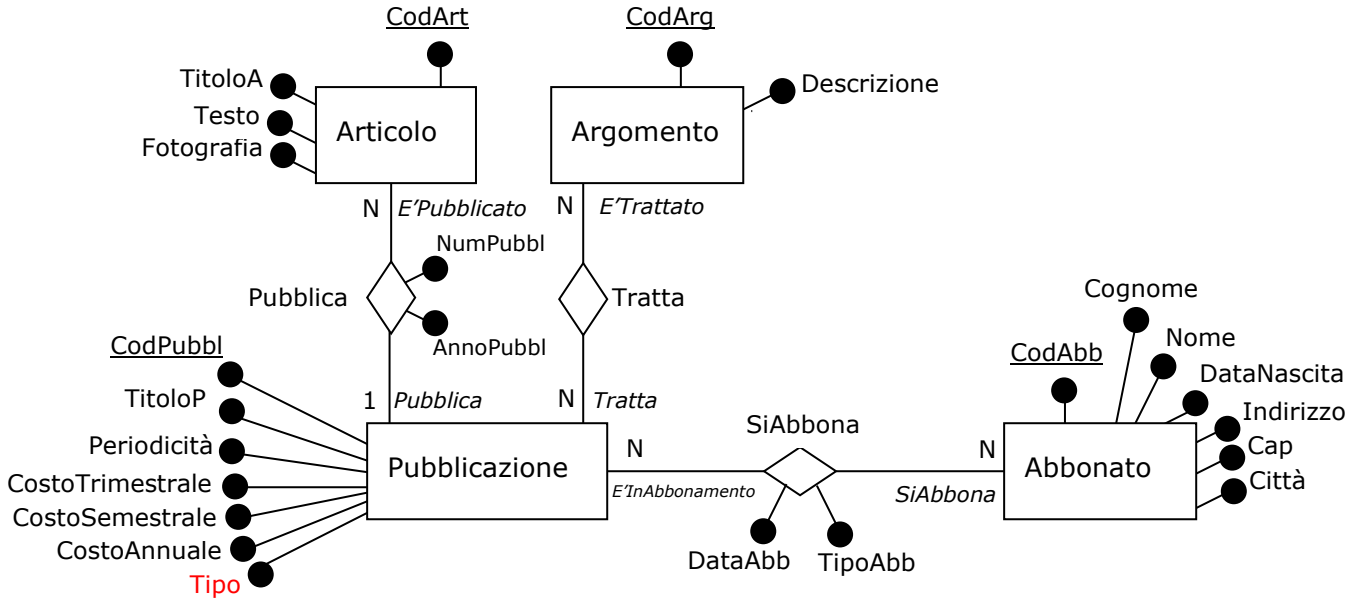
Specifichiamo infine, oltre ai ben noti vincoli impliciti (rappresentati dai vincoli di chiave primaria e dalla totalità delle associazioni dirette e/o inverse) i seguenti vincoli espliciti:

- il dominio dell'attributo *Periodicità* è composto dai valori "Quotidiano", "Settimanale" e "Mensile";
- il dominio dell'attributo *Tipo* è composto dai valori "Giornale" e "Rivista";
- il valore degli attributi *CostoTrimestrale, CostoSemestrale, CostoAnnuale* deve essere positivo
- il dominio dell'attributo *TipoAbb* è composto dai valori "Trimestrale", "Semestrale" e "Annuale";
- il valore dell'attributo *NumAbb* deve essere positivo e corrispondente ad una pubblicazione esistente;
- il valore dell'attributo *DataNascita* di un abbonato deve essere precedente rispetto al valore dell'attributo *DataAbb* del suo abbonamento.

Schema concettuale della realtà di interesse – diagramma E/R con ISA



(a) Schema concettuale della realtà di interesse – diagramma E/R senza ISA
(ottenuto attraverso l'accorpamento delle entità figlie nell'entità padre perché l'ISA è TOTALE)



Vincoli impliciti:

Sono quei vincoli direttamente deducibili dal diagramma E/R in quanto caratterizzati graficamente:

- vincoli di chiave primaria: tutti gli attributi sottolineati
- vincoli di integrità referenziale: totalità delle associazioni dirette e/o inverse (linea continua)

Vincoli espliciti:

Vincoli non deducibili direttamente dal diagramma E/R:

V1: (Pubblicazione.Peridiocita = "Quotidiano") OR (Pubblicazione.Peridiocita = "Settimanale") OR (Pubblicazione.Peridiocita = "Mensile")

V2: (Pubblicazione.Tipo = "Giornale") OR (Pubblicazione.Tipo = "Rivista")

V3: (Pubblicazione.CostoTrimestrale > 0)

V4: (Pubblicazione.CostoSemestrale > 0)

V5: (Pubblicazione.CostoAnnuale > 0)

V6: (SiAbbona.TipoAbb = "Trimestrale") OR (SiAbbona.TipoAbb = "Semestrale") OR (SiAbbona.TipoAbb = "Annuale")

V7: (Pubblica.NumPubbl > 0)

V8: (Pubblica.DataPubb <= "1995") AND (Pubblica.DataPubb <= "2006")

V9: (Abbonato.DataNascita < SiAbbona.DataAbb)

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "SiAbbona" di molteplicità N:N tra le entità "Abbonato" e "Pubblicazione"

Pubblicazione (CodPubbl, TitoloP, Periodicita, CostoTrimestrale, CostoSemestrale, CostoAnnuale, Tipo)

Abbonato (CodAbb, Cognome, Nome, DataNascita, Indirizzo, Cap, Citta)

SiAbbona (CodAbb1, CodPubbl1, DataAbb, TipoAbb)

con "CodAbb1" chiave esterna (foreign key) sull'attributo "CodAbb" della relazione "Abbonato"

con "CodPubbl1" chiave esterna (foreign key) sull'attributo "CodPubbl" della relazione "Pubblicazione"

$VR_{CodAbb1}$ (SiAbbona) $\subseteq$ VR_{CodAbb} (Abbonato) dal mapping relazionale dell'associazione N:N

$VR_{CodPubbl1}$ (SiAbbona) $\subseteq$ $VR_{CodPubbl}$ (Pubblicazione) dal mapping relazionale dell'associazione N:N

VR_{CodAbb} (Abbonato) $\subseteq$ $VR_{CodAbb1}$ (SiAbbona) dalla TOTALITA' dell'associazione diretta "SiAbbona"

$VR_{CodPubbl}$ (Pubblicazione) $\subseteq$ $VR_{CodPubbl1}$ (SiAbbona) dalla TOTALITA' dell'associazione inversa "E'inAbbonamento"

(ii) mapping dell'associazione "Tratta" di molteplicità N:N tra le entità "Pubblicazione" e *"Argomento"*

Pubblicazione già mappata in precedenza

Argomento (CodArg, Descrizione)

Tratta (CodPubbl2, CodArg2)

con "CodPubbl2" chiave esterna (foreign key) sull'attributo "CodPubbl" della relazione "Pubblicazione"

con "CodArg2" chiave esterna (foreign key) sull'attributo "CodArg" della relazione "Argomento"

$VR_{CodPubbl2}$ (Tratta) $\subseteq$ $VR_{CodPubbl}$ (Pubblicazione) dal mapping relazionale dell'associazione N:N

$VR_{CodArg2}$ (Tratta) $\subseteq$ VR_{CodArg} (Argomento) dal mapping relazionale dell'associazione N:N

$VR_{CodPubbl}$ (Pubblicazione) $\subseteq$ $VR_{CodPubbl2}$ (Tratta) dalla TOTALITA' dell'associazione diretta "Tratta"

VR_{CodArg} (Argomento) $\subseteq$ $VR_{CodArg2}$ (Tratta) dalla TOTALITA' dell'associazione inversa "E'Trattato"

(iii) mapping dell'associazione "Pubblica" di molteplicità 1:N tra le entità "Pubblicazione" e *"Articolo"*

Pubblicazione già mappata in precedenza

Articolo (CodArt, TitoloA, Testo, Fotografia, NumPubbl, AnnoPubbl, CodPubbl3)

con "CodPubbl3" chiave esterna (foreign key) sull'attributo" CodPubbl" della relazione "Pubblicazione"

$VR_{CodPubbl}$ (Pubblicazione) $\subseteq$ $VR_{CodPubbl3}$ (Articolo) dalla TOTALITA' dell'associazione diretta "Pubblica"

$VR_{CodPubbl3}$ (Articolo) $\subseteq$ $VR_{CodPubbl}$ (Pubblicazione) dalla TOTALITA' dell'associazione inversa "E'Trattato"

Vincoli di integrità intrarelazionali (o interni) ed interrelazionali (o esterni)

I vincoli di chiave primaria (impliciti nel modello E/R) sono mappati in *vincoli intrarelazionali su più ennuple.*

I vincoli di totalità di un'associazione (impliciti nel modello E/R) sono mappati in *vincoli interrelazionali di tipo referenziale*

I vincoli espliciti del diagramma E/R vengono mappati come segue:

V1 (Pubblicazione): "(Peridiocità = "Quotidiano") OR (Peridiocità = "Settimanale") OR (Peridiocità = "Mensile")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V2 (Pubblicazione): "(Tipo = "Giornale") OR (Tipo = "Rivista")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V3 (Pubblicazione): "(CostoTrimestrale > 0)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V4 (Pubblicazione): "(CostoSemestrale > 0)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V5 (Pubblicazione): "(CostoAnnuale > 0)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V6 (SiAbbona): (TipoAbb = "Trimestrale") OR (TipoAbb = "Semestrale") OR (TipoAbb = "Annuale")
vincolo intrarelazionale su singola ennupla su singolo attributo

V7 (Articolo): "(NumPubbl > 0)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V8 (Articolo): "(DataPubbl >= "1995") AND (DataPubbl <= "2006")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V9(Abbonato, SiAbbona): "(Abbonato.DataNascita < SiAbbona.DataAbb)"
vincolo interrelazionale di tipo non referenziale

Normalizzazione

*La **teoria della normalizzazione** ha come scopo quello di fornire metodi per progettare basi di dati senza anomalie. La realtà che si intende analizzare tramite una base di dati può avere più schemi, equivalenti fra loro,che la descrivono. Alcuni di questi schemi possono presentare degli inconvenienti (**ridondanze o anomalie** di comportamento) che rendono inadeguato l'utilizzo efficiente delle informazioni.*

*Con **ridondanza** si intende la non necessaria ripetizione della stessa informazione con il conseguente spreco di spazio;*

*Con **anomalie da modifica** si intende la necessità di ripetere, in caso di cambiamento di una informazione, la modifica ovunque tale informazione è, duplicata;*

*Con **anomalia da inserimento** si intende la necessità di inserire ulteriori informazioni, non strettamente necessarie per poter inserire un nuovo dato;*

*Con **anomalie di cancellazion**e si intende l'eliminazione di alcune informazioni in conseguenza della cancellazione di altre.*

*La teoria della normalizzazione si occupa di definire criteri di bontà per schemi relazionali (**forme normali**)*

Una forma normale è una proprietà di uno schema relazionale che ne garantisce la "qualità", cioè l'assenza di determinati difetti) ed eventualmente di determinare metodi algoritmici per ottenere uno schema "migliore" ed equivalente a partire da uno schema non in forma normale (normalizzazione)

In genere per costruire uno schema relazionale si possono utilizzare i seguenti metodi:

1. Si parte da un buon schema E/R e lo si "traduce";

2. Si costruiscono direttamente le relazioni e poi si correggono quelle che presentano "anomalie";

3. Si analizza uno schema relazionale già esistente e lo si modifica/completa;

La normalizzazione

• È particolarmente utile se, per costruire lo schema, si usano i metodi (2) o (3);

• È moderatamente utile anche quando si usa il metodo (1);

***PRIMA FORMA NORMALE:** Ogni campo deve contenere un solo valore: (la 1NF è chiamata anche FORMA ATOMICA).*

***SECONDA FORMA NORMALE:** Una tabella è in 2NF se è in 1NF e tutti gli attributi non chiave dipendono funzionalmente da tutta la chiave*

***TERZA FORMA NORMALE:** Una tabella è in 3NF se è in 2NF e tutti gli attributi non chiave dipendono funzionalmente solo dalla chiave (eliminazione delle dipendenze funzionali transitive).*

N.B. Il nostro schema relazionale è in 3FN

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE CasaEditrice;

USE CasaEditrice;

CREATE DOMAIN MiaPeriodicita AS CHAR(11)
  CHECK (VALUE IN ("Quotidiano", "Settimanale", "Mensile"));      // Vincolo V1

CREATE DOMAIN MioTipo AS CHAR(8)
  CHECK (VALUE IN ("Giornale", "Rivista"));                       // Vincolo V2

CREATE DOMAIN MioAbbonamento AS CHAR(11)
  CHECK (VALUE IN ("Trimestrale", "Semestrale", "Annuale"));      //Vincolo V6

CREATE TABLE Pubblicazione
  (
  CodPubbl            CHAR(4)          NOT NULL,
  TitoloP             CHAR(25)         NOT NULL,
  Periodicita         MiaPeriodicita   NOT NULL,
  Costo Trimestrale   DECIMAL(3,2)     NOT NULL,
  Costo Semestrale    DECIMAL(3,2)     NOT NULL,
  Costo Annuale       DECIMAL(3,2)     NOT NULL,
  Tipo                MioTipo          NOT NULL,
  PRIMARY KEY (CodPubbl),
  UNIQUE (TitoloP),
  CHECK (Costo Trimestrale > 0),                  // Vincolo V3
  CHECK (Costo Semestrale > 0),                   // Vincolo V4
  CHECK (Costo Annuale > 0)                       // Vincolo V5
  );

CREATE TABLE Abbonato
  (
  CodAbb              CHAR(10)         NOT NULL,
  Cognome             CHAR(25)         NOT NULL,
  Nome                CHAR(25)         NOT NULL,
  DataNascita         DATE             NOT NULL,
  Indirizzo           CHAR(100)        NOT NULL,
  Cap                 CHAR(5)          NOT NULL,
  Citta               CHAR(25)         NOT NULL,
  PRIMARY KEY (CodAbb),
  UNIQUE (Cognome, Nome, DataNascita)
  );

CREATE TABLE SiAbbona
  (
  CodAbb1             CHAR(10)         NOT NULL,
  CodPubbl1           CHAR(4)          NOT NULL,
  DataAbb             DATE             NOT NULL,
  TipoAbb             MioAbbonamento   NOT NULL,
  PRIMARY KEY (CodAbb1, CodPubbl1),
  FOREIGN KEY (CodAbb1) REFERENCES ON Abbonato (CodAbb)          // VR di chiave esterna
  ON DELETE CASCADE,  ON UPDATE CASCADE,
  FOREIGN KEY (CodPubbl1) REFERENCES ON Pubblicazione (CodPubbl)     // VR di chiave esterna
  ON DELETE CASCADE,  ON UPDATE CASCADE );
```

```
CREATE TABLE Argomento
  (
  CodArg            CHAR(4)          NOT NULL,
  Descrizione       VARCHAR(500)     NOT NULL,
  PRIMARY KEY (CodArg)
  );

CREATE TABLE Tratta
  (
  CodPubbl2         CHAR(4)          NOT NULL,
  CodArg2           CHAR(4)          NOT NULL,
  PRIMARY KEY (CodPubbl2, CodArg2),
  FOREIGN KEY (CodPubbl2) REFERENCES ON Pubblicazione (CodPubbl)     // VR di chiave esterna
  ON DELETE CASCADE,  ON UPDATE CASCADE,
  FOREIGN KEY (CodArg2) REFERENCES ON Argomento (CodArg)         // VR di chiave esterna
  ON DELETE CASCADE,  ON UPDATE CASCADE
  );

CREATE TABLE Articolo
  (
  CodArt            CHAR(4)          NOT NULL,
  TitoloA           CHAR(25)         NOT NULL,
  Testo             VARCHAR(2000)    NOT NULL,
  Fotografia        CHAR(255),
  NumPubbl          INT              NOT NULL,
  AnnoPubbl         CHAR(4)          NOT NULL,
  CodPubbl3         CHAR(4),
  PRIMARY KEY (CodArt),
  FOREIGN KEY (CodPubbl3) REFERENCES ON Pubblicazione (CodPubbl)     // VR di chiave esterna
  ON DELETE SET NULL,  ON UPDATE CASCADE,
  CHECK (NumPubbl > 0),                                          // Vincolo V7
  CHECK (AnnoPubb BETWEEN "1995" AND "2006")          // Vincolo V8
  );

CREATE ASSERTION V9 CHECK (Abbonato.DataNascita < SiAbbona.DataAbb);
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1: Dato il titolo di una pubblicazione, ricercare gli articoli pubblicati in un determinato anno;

SQL dapprima occorre eseguire la connessione al database e poi

```
USE CasaEditrice;

SELECT TitoloA, Testo, Fotografia, NumPubbl
FROM Pubblicazione, Articolo
WHERE Pubblicazione.CodPubbl = Articolo.CodPubll3
AND (AnnoPubbl = [InserisciAnno]) AND (TitoloP = [InserisciPubblicazione]);
```

N.B. e' possibile ordinarli per numero di pubblicazione

oppure anche

```
SELECT TitoloA, Testo, Fotografia, NumPubbl
FROM Pubblicazione INNER JOIN Articolo ON Pubblicazione.CodPubbl = Articolo.CodPubll3
WHERE (AnnoPubbl = [InserisciAnno]) AND (TitoloP = [InserisciPubblicazione]);
```

Algebra Relazionale

A = { TitoloA, Testo, Fotografia, NumPubbl }

P = { (AnnoPubbl = [InserisciAnno]) AND (TitoloP = [InserisciPubblicazione]) }

$$Q1 = \Pi_{A}(\sigma_{P}(\text{Pubblicazione} \underset{\text{CodPubbl=CodPubbl3}}{\triangleright\triangleleft} \text{Articolo}))$$

oppure anche in modo più efficiente

$$Q1 = \Pi_{A}(\text{Pubblicazione} \underset{\text{CodPubbl=CodPubbl3}}{\triangleright\triangleleft} \sigma_{P}(\text{Articolo}))$$

Q2: Dato il titolo di una pubblicazione, ricercare gli abbonati annuali;

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

SELECT Cognome, Nome, DataNascita, Indirizzo, Cap, Citta
FROM Abbonato, SiAbbona, Pubblicazione
WHERE (Abbonato.CodAbb = SiAbbona.CodAbb1) **AND** (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl)
AND (TitoloP = [InserisciPubblicazione]);

N.B. e' possibile ordinarli per Cognome e Nome

oppure anche

SELECT Cognome, Nome, DataNascita, Indirizzo, Cap, Citta
FROM Abbonato **INNER JOIN** (SiAbbona **INNER JOIN** Pubblicazione **ON** CodPubbl1 = CodPubbl)
ON CodAbb = CodAbb1
WHERE (TitoloP = [InserisciPubblicazione]);

Algebra Relazionale

A = { Cognome, Nome, DataNascita, Indirizzo, Cap, Citta }

P = { (TitoloP = [Inserisci Pubblicazione]) }

$$Q2 = \Pi_{A}(\sigma_{P}(\text{Abbonato} \underset{\text{CodAbb=CodAbb1}}{\triangleright\triangleleft} (\text{SiAbbona} \underset{\text{CodPubbl1=CodPubbl}}{\triangleright\triangleleft} \text{Pubblicazione})))$$

oppure anche in modo più efficiente

$$Q2 = \Pi_{A}(\text{Abbonato} \underset{\text{CodAbb=CodAbb1}}{\triangleright\triangleleft} (\text{SiAbbona} \underset{\text{CodPubbl1=CodPubbl}}{\triangleright\triangleleft} \sigma_{P}(\text{Pubblicazione})))$$

Q3: Dato il nominativo di un abbonato, stabilire a quante riviste è abbonato

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

SELECT COUNT(*) AS NumRiviste
FROM Abbonato, SiAbbona, Pubblicazione
WHERE (Abbonato.CodAbb = SiAbbona.CodAbb1) **AND** (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl)
AND (Cognome = [InserisciCognome]) **AND** (Nome = [InserisciNome]) **AND** (Tipo = "Rivista");

oppure anche

SELECT COUNT(*) AS NumRiviste
FROM Abbonato **INNER JOIN** (SiAbbona **INNER JOIN** Pubblicazione **ON** CodPubbl1 = CodPubbl)
ON CodAbb = CodAbb1
WHERE (Cognome = [InserisciCognome]) **AND** (Nome = [InserisciNome]) **AND** (Tipo = "Rivista");

Algebra Relazionale

A = { COUNT(*) AS NumRiviste } (N.B. un pò forzato quando vengono usate funzioni di aggregazione)

P = { (Cognome = [InserisciCognome]) **AND** (Nome = [InserisciNome]) }

P1 = { (Tipo = "Rivista") }

$$Q3 = \Pi_A(\sigma_P(\text{Abbonato} \underset{\text{CodAbb=CodAbb1}}{\triangleright\!\triangleleft} (\text{SiAbbona} \underset{\text{CodPubbl1=CodPubbl}}{\triangleright\!\triangleleft} \text{Pubblicazione})))$$

oppure anche in modo più efficiente

$$Q3 = \Pi_A(\sigma_P(\text{Abbonato}) \underset{\text{CodAbb=CodAbb1}}{\triangleright\!\triangleleft} (\text{SiAbbona} \underset{\text{CodPubbl1=CodPubbl}}{\triangleright\!\triangleleft} \sigma_{P1}(\text{Pubblicazione})))$$

Q4: Dato un argomento, elencare le pubblicazioni in cui è trattato

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

```
SELECT TitoloP
FROM Pubblicazione, Tratta, Argomento
WHERE (Pubblicazione.CodPubbl = Tratta.CodPubbl2) AND (Tratta.CodArg2 = Argomento.CodArg)
AND (Descrizione = [InserisciDescrizione]);
```

N.B. E' possibile ordinarle per TitoloP

oppure anche

```
SELECT TitoloP
FROM Pubblicazione INNER JOIN (Tratta INNER JOIN Argomento ON CodArg2 =CodArg)
ON CodPubbl = CodPubbl2
WHERE (Descrizione = [InserisciDescrizione]);
```

Algebra Relazionale

A = { TitoloP }

P = { (Descrizione = [InserisciDescrizione]) }

$$Q4 = \Pi_A(\sigma_P(\text{Pubblicazione} \underset{\text{CodPubbl=CodPubbl2}}{\triangleright\!\triangleleft} (\text{Tratta} \underset{\text{CodArg2=CodArg}}{\triangleright\!\triangleleft} \text{Argomento})))$$

oppure anche in modo più efficiente

$$Q4 = \Pi_A(\text{Pubblicazione} \underset{\text{CodPubbl=CodPubbl2}}{\triangleright\!\triangleleft} (\text{Tratta} \underset{\text{CodArg2=CodArg}}{\triangleright\!\triangleleft} \sigma_P(\text{Argomento})))$$

Q5: Riportare per ogni pubblicazione il numero di abbonamenti

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

```
SELECT CodPubbl, TitoloP, COUNT(*) AS NumAbbonati
FROM SiAbbona, Pubblicazione
WHERE (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl)
GROUP BY CodPubbl, TitoloP;
```

N.B. E' possibile ordinarle per TitoloP

oppure anche

```
SELECT CodPubbl, TitoloP, COUNT(*) AS NumAbbonati
FROM SiAbbona INNER JOIN Pubblicazione ON CodPubbl1 = CodPubbl
GROUP BY CodPubbl, TitoloP;
```

Q6: Visualizzare i giornali con almeno 5000 abbonati annuali

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

N.B. E' possibile ordinarle per TitoloP

```
SELECT CodPubbl, TitoloP
FROM SiAbbona, Pubblicazione
WHERE (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl) AND (Tipo = "Giornale")
AND (TipoAbb = "Annuale")
GROUP BY CodPubbl, TitoloP
HAVING COUNT(*) >= 5000;
```

oppure anche

```
SELECT CodPubbl, TitoloP
FROM SiAbbona INNER JOIN Pubblicazione ON CodPubbl1 = CodPubbl)
WHERE (Tipo = "Giornale") AND (TipoAbb = "Annuale")
GROUP BY CodPubbl, TitoloP
HAVING COUNT(*) >= 5000;
```

Q7: Dati i titoli di due pubblicazioni, visualizzarne gli abbonati comuni

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

```
(SELECT CodAbb, Cognome, Nome
FROM Abbonato, SiAbbona, Pubblicazione
WHERE (Abbonato.CodAbb = SiAbbona.CodAbb1) AND (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl)
AND (TitoloP = [InserisciPrimoTitolo]) )

INTERSECT

(SELECT CodAbb, Cognome, Nome
FROM Abbonato, SiAbbona, Pubblicazione
WHERE (Abbonato.CodAbb = SiAbbona.CodAbb1) AND (SiAbbona.CodPubbl1 = Pubblicazione.CodPubbl)
AND (TitoloP = [InserisciSecondoTitolo]) )
```

Q8: Dato il titolo di una pubblicazione, elencare le pubblicazioni che trattano i suoi stessi argomenti

SQL dapprima occorre eseguire la connessione al database e poi
USE CasaEditrice;

N.B. E' possibile ordinarle per TitoloP

```
SELECT TitoloP
FROM Pubblicazione, Tratta
WHERE (Pubblicazione.CodPubbl = Tratta.CodPubbl2)
AND CodArg2 IN (SELECT CodArg
                FROM Pubblicazione, Tratta
                WHERE (Pubblicazione.CodPubbl = Tratta.CodPubbl2)
                AND (TitoloP = [Inserisci Titolo]) ) ;
```

(ESAME DI STATO – SECONDA PROVA SCRITTA ANNO 2004)

(2) Il **Dirigente Scolastico** di una Scuola Secondaria Superiore chiede che si realizzi una base di dati per l'archiviazione e la gestione di informazioni riguardanti le **attività scolastiche ed extrascolastiche** documentate, nonché i risultati scolastici conseguiti da ciascuno studente al fine di produrre, in itinere e/o al termine del corso di studi, un *.portfolio studente*.

L'organizzazione scolastica dell'istituto prevede che:

a) ciascuno studente possa frequentare più corsi di recupero e/o di sostegno e/o di arricchimento dell'offerta formativa;

b) ogni corso abbia un titolo, una descrizione, una data di inizio e di fine, un monte ore definito;

c) ogni studente possa frequentare più corsi esterni alla scuola;

d) i corsi esterni alla scuola hanno un titolo, una descrizione, una data di inizio e di fine, un monte ore definito, un riferimento che indichi l'Ente e/o l'Istituzione che li ha organizzati, un riferimento alla documentazione di accertamento;

e) ogni classe sia individuata univocamente da un numero ordinale progressivo (I, II, III, IV, V) e da una lettera che ne indica la sezione di appartenenza (A, B, C, D, E, F, G,).

In particolare, il Dirigente Scolastico chiede che si possa procedere all'archiviazione dei:

• dati anagrafici degli studenti utili alla loro univoca identificazione;

• dati relativi alla frequenza delle classi del corso di studi (quali classi ciascuno studente ha frequentato in ordine crescente ed in quale anno scolastico);

• dati relativi agli esiti conclusivi di ciascun anno scolastico (promozione sì/no, eventuali debiti formativi ed in quale disciplina) per ciascuno studente;

• dati relativi ai corsi interni;

• dati relativi ai corsi esterni alla scuola purché documentati.

Il candidato, fatte le opportune ipotesi aggiuntive, progetti una base di dati utile alla realizzazione del portfolio studente richiesto dal Dirigente Scolastico, fornendo:

1. uno schema concettuale della base di dati;

2. uno schema logico della base di dati;

3. la definizione delle relazioni della base di dati in linguaggio SQL;

ed inoltre:

4. implementi in linguaggio SQL le seguenti interrogazioni:

Q1: Data una classe ed un anno scolastico, visualizzare quali studenti di quella classe hanno frequentato corsi e di che tipo;

Q2: Dato uno studente, visualizzare quali corsi ha frequentato, di che tipo, per quale monte ore e in quale anno scolastico;

Q3: Dato un anno scolastico, visualizzare quali corsi interni sono stati attivati e da quali studenti sono stati seguiti;

Q4: Dato un corso, visualizzare quali sono i dati relativi ad esso e per quali anni scolastici è stato attivato;

Q5: Dato uno studente, visualizzare quali classi ha frequentato, in quali anni scolastici e con quali esiti finali;

Q6: Per ogni anno scolastico, contare il numero di studenti respinti;

Q7: Dato un anno scolastico, contare il numero totale di ore dei corsi organizzati per l'arricchimento dell'offerta formativa;

Q8: Visualizzare l'elenco degli studenti che non hanno mai frequentato corsi di recupero.

5. Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

Premessa: omessa (vedi svolgimento esercizio precedente)

Da un'attenta lettura delle specifiche si evidenzia che sono richieste le seguenti attività:

- gestione degli studenti e della loro situazione scolastica;
- gestione dei corsi interni ed esterni ala scuola.

Possiamo quindi ipotizzare di avere entità contenenti le informazioni relative:

- agli *studenti*
- alle *classi* da loro frequentate con le risultanza finali anno per anno;
- ai *corsi* da loro frequentati (suddivisi in *interni* ed *esterni*)

Abbiamo quindi ottenuto un primo elenco di entità che dovranno entrare a far parte dello schema concettuale della base di dati che si sta progettando.

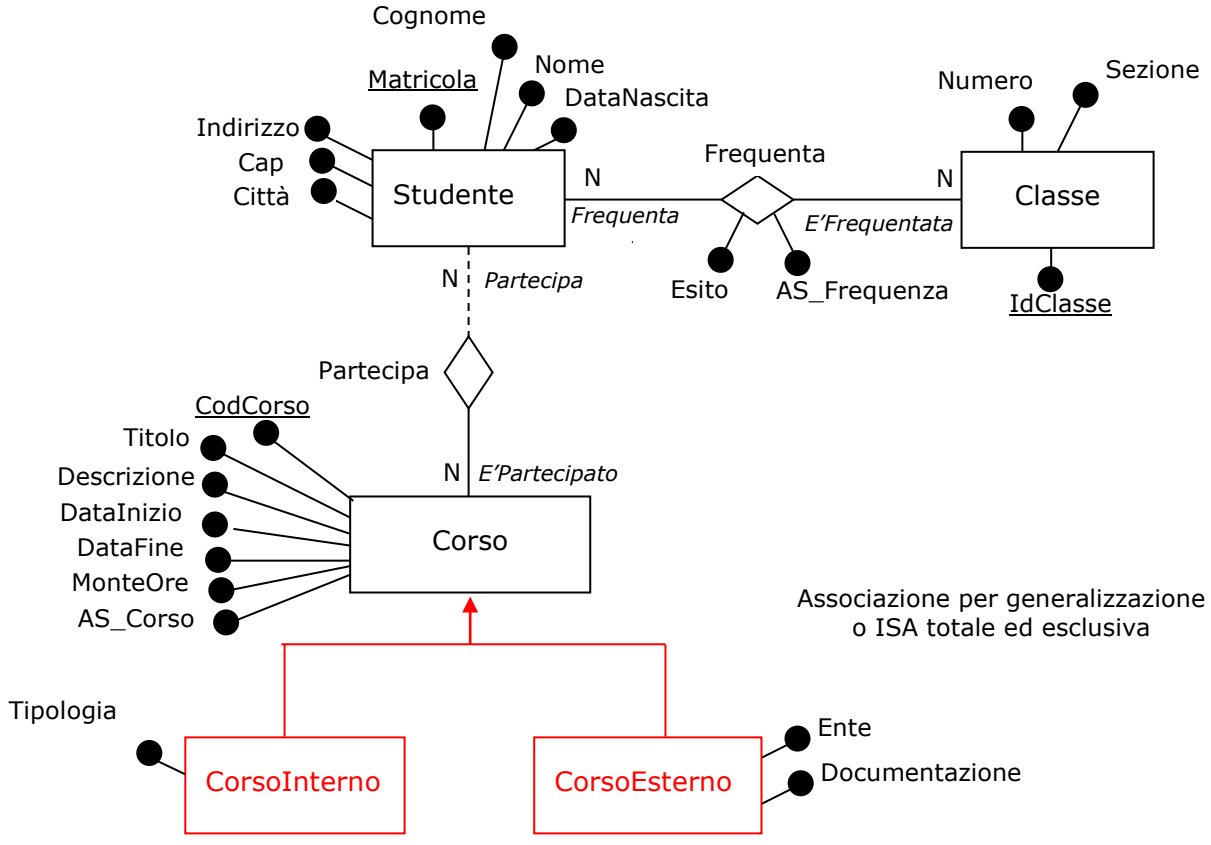
A partire da questo elenco, associamo a ciascuna entità individuata i corrispondenti attributi, attingendo sempre le informazioni dalle specifiche fornite:

- a ciascuno studente devono essere associati una *Matricola*, un *Cognome*, un *Nome*, una *DataNascita*, un *Indirizzo*, un *Cap*, una *Città* (anagrafica breve);
- a ciascuna classe deve essere associata la coppia di valori *numero ordinale classe* e *lettera sezione*;
- ciascun corso è caratterizzato da un *Titolo*, una *Descrizione*, una *DataInizio*, una *DataFine*, un *MonteOre* e dal'*AnnoScolastico* di riferimento in cui stato attivato; ai corsi interni è inoltre associata una *Tipologia* (corso di recupero, di sostegno, di arricchimento) mentre a quelli esterni è associato l'*Ente* organizzatore e la *Documentazione* di accertamento.

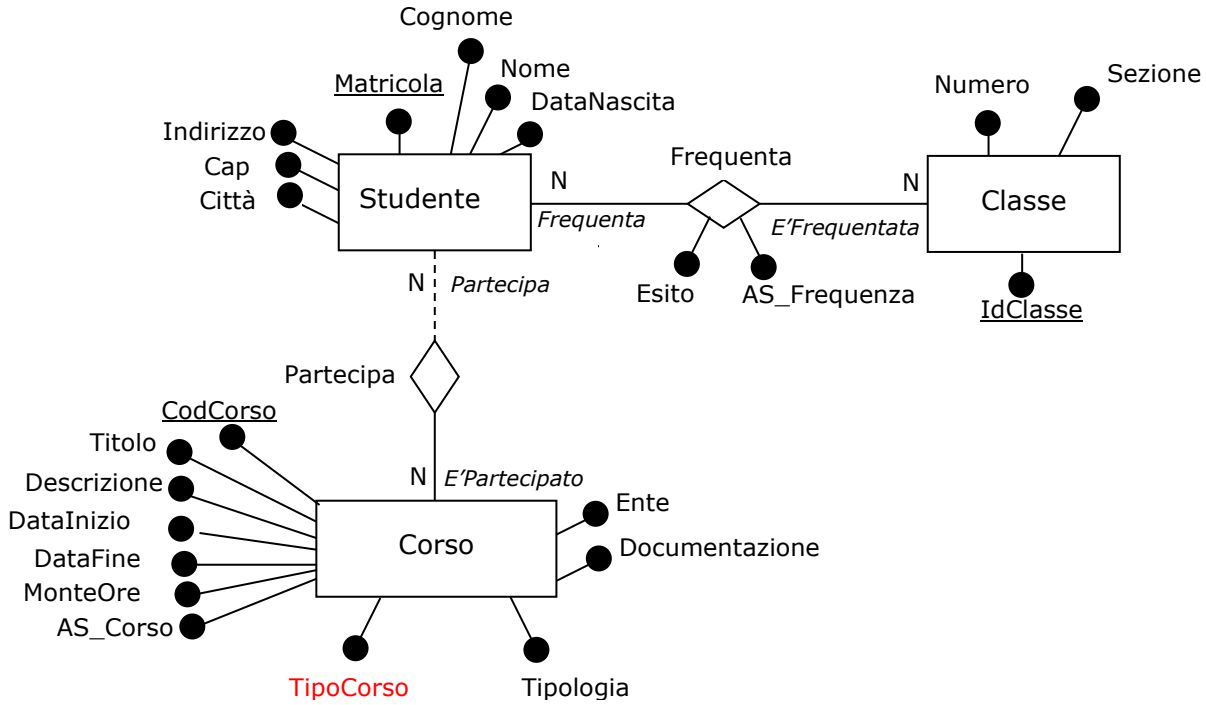
Passiamo ora ad esaminare quali sono le associazioni tra le entità ipotizzate, individuando per ciascuna di esse la molteplicità dell'associazione in base alle caratteristiche di funzionalità evidenziate nelle specifiche più eventuali attributi:

- esiste un'associazione ISA totale ed esclusiva con la funzione di suddividere le istanze dell'entità padre *Corso* nelle entità figlie *CorsoInterno* e *CorsoEsterno*, in modo da poter specificare per ciascuna sottoentità i corrispondenti attributi aggiuntivi;
- tra le entità *Studente* e *Corso* esiste un'associazione "Partecipa" di molteplicità N:N (con diretta parziale ed inversa totale) in quanto *"uno studente può partecipare a nessuno o più corsi e viceversa un corso deve essere partecipato da uno o più studenti"*;
- tra le entità *Studente* e *Classe* esiste un'associazione "Frequenta" di molteplicità N:N (totale in entrambi i versi) in quanto *"uno studente deve frequentare una o più classi* (si tiene conto cos' dello "storico" ossia delle classi da lui frequentate negli anni precedenti) *ed ogni classe deve essere frequentata da uno o più studenti"*; sono inoltre mantenute come attributi dell'associazione, le informazioni sull'Esito (supponendo che possa assumere il valore "Promosso", "Respinto", "Debito") conclusivo di ogni anno scolastico per ciascuno studente nella corrispondente classe frequentata; se si desidera si potrebbe mantenere l'elenco dei debiti formativi (attributo multiplo opzionale) introducendo un'altra entità *Materia* ed un'altra associazione

Schema concettuale della realtà di interesse – diagramma E/R con ISA



(a) <u>Schema concettuale della realtà di interesse – diagramma E/R senza ISA</u>
<u>(ottenuto attraverso l'accorpamento delle entità figlie nell'entità padre perché l'ISA è TOTALE)</u>



<u>Vincoli impliciti:</u>

Sono quei vincoli direttamente deducibili dal diagramma E/R in quanto caratterizzati graficamente:

- vincoli di chiave primaria: tutti gli attributi sottolineati
- vincoli di integrità referenziale: totalità delle associazioni dirette e/o inverse (linea continua)

<u>Vincoli espliciti:</u>

Vincoli non deducibili direttamente dal diagramma E/R:

V1: (Corso.Tipologia="Recupero") OR (Corso.Tipologia="Sostegno") OR (Corso.Tipologia="Arricchimento")

V2: (Corso.Tipologia="Interno") OR (Corso.Tipologia="Esterno")

V3: (Classe.Numero=1) OR (Classe.Numero=2) OR (Classe.Numero=3) OR (Classe.Numero=4) OR (Classe.Numero = 5)

V4: (Classe.Sezione='A') OR (Classe.Sezione='B') OR (Classe.Sezione='C') OR (Classe.Sezione='D') OR (Classe.Sezione=E') OR (Classe.Sezione='F') OR (Classe.Sezione='G)

V5: (Frequenta.Esito="Promosso") OR (Frequenta.Esito="Respinto") OR (Frequenta.Esito="Debito")

V6: (Corso.DataInizio < Corso.DataFine)

V7: (Corso.MonteOre > 0)

V8: (Corso.DataInizio > Studente.DataNascita)

V9: SE (Corso.TipoCorso = "Interno")
ALLORA
Corso.Ente = NULL
Corso.Documentazione = NULL
FINE SE

V10: SE (Corso.TipoCorso = "Esterno")
ALLORA
Corso.Tipologia = NULL
FINE SE

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "SiAbbona" di molteplicità N:N tra le entità "Abbonato" e "Pubblicazione"

Studente (Matricola, Cognome, Nome, DataNascita, Indirizzo, Cap, Citta)

Classe (IdClasse, Numero, Sezione)

Frequenta (Matricola1, IdClasse1, Esito, AS_Frequenza)

con "Matricola1" chiave esterna (foreign key) sull'attributo "Matricola" della relazione "Studente"

con "IdClasse1" chiave esterna (foreign key) sull'attributo "IdClasse" della relazione "Classe"

$VR_{Matricola1}$ (Frequenta) $\subseteq$ $VR_{Matricola}$ (Studente) dal mapping relazionale dell'associazione N:N

$VR_{IdClasse1}$ (Frequenta) $\subseteq$ $VR_{IdClasse}$ (Classe) dal mapping relazionale dell'associazione N:N

$VR_{Matricola}$ (Studente) $\subseteq$ $VR_{Matricola1}$ (Frequenta) dalla TOTALITA' dell'associazione diretta "Frequenta"

$VR_{IdClasse}$ (Classe) $\subseteq$ $VR_{IdClasse1}$ (Frequenta) dalla TOTALITA' dell'associazione inversa "E'Frequentata"

(ii) mapping dell'associazione "Partecipa" di molteplicità N:N tra le entità "Studente" e "Corso"

Studente già mappato in precedenza

Corso (CodCorso, Titolo, Descrizione, DataInizio, DatFine, MonteOre, AS_Corso, Tipologia, Ente, Documentazione, TipoCorso)

Partecipa (Matricola2, CodCorso2)

con "Matricola2" chiave esterna (foreign key) sull'attributo "Matricola" della relazione "Studente"

con "CodCorso2" chiave esterna (foreign key) sull'attributo "CodCorso" della relazione "Corso"

$VR_{Matricola2}$ (Partecipa) $\subseteq$ $VR_{Matricola}$ (Studente) dal mapping relazionale dell'associazione N:N

$VR_{CodCorso2}$ (Partecipa) $\subseteq$ $VR_{CodCorso}$ (Corso) dal mapping relazionale dell'associazione N:N

$VR_{CodCorso}$ (Corso) $\subseteq$ $VR_{CodCorso2}$ (Partecipa) dalla TOTALITA' dell'associazione inversa "E'Partecipato"

Vincoli di integrità intrarelazionali (o interni) ed interrelazionali (o esterni)

I vincoli di chiave primaria (impliciti nel modello E/R) sono mappati in *vincoli intrarelazionali su più ennuple.*

I vincoli di totalità di un'associazione (impliciti nel modello E/R) sono mappati in *vincoli interrelazionali di tipo referenziale*

I vincoli espliciti del diagramma E/R vengono mappati come segue:

V1 (Corso): "(Tipologia="Recupero") OR (Tipologia="Sostegno") OR (Tipologia="Arricchimento")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V2 (Corso): "(Tipologia="Interno") OR (Tipologia="Esterno")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V3 (Classe): "(Numero=1) OR (Numero=2) OR (Numero=3) OR (Numero=4) OR (Numero = 5)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V4 (Classe): "(Sezione='A') OR (Sezione='B') OR (Sezione='C') OR (Sezione='D') OR (Sezione=E') OR (Sezione='F') OR (Sezione='G)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V5 (Frequenta): "(Esito="Promosso") OR (Esito="Respinto") OR (Esito="Debito")" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V6 (Corso): "(DataInizio < DataFine)" *vincolo intrarelazionale su singola ennupla su più di un attributo*

V7 (Corso): "(MonteOre > 0)" *vincolo intrarelazionale su singola ennupla su singolo attributo*

V8 (Corso, Studente): "(Corso.DataInizio > Studente.DataNascita)" *vincolo interrelazionale di tipo non referenziale*

V9 (Corso): SE (TipoCorso = "Interno") *vincolo intrarelazionale su singola ennupla su più di un attributo*

ALLORA

Ente = NULL

Documentazione = NULL

FINE SE

V10 (Corso): SE (TipoCorso = "Esterno") *vincolo intrarelazionale su singola ennupla su più di un attributo*

ALLORA
Tipologia = NULL
FINE SE

Normalizzazione: omessa (vedi svolgimento esercizio precedente)

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE Scuola;

USE Scuola;

CREATE DOMAIN MiaTipologia AS CHAR(13)
  CHECK (VALUE IN ("Recupero", "Sostegno", "Arricchimento"));      // Vincolo V1

CREATE DOMAIN MioTipo AS CHAR(7)
  CHECK (VALUE IN ("Interno", "Esterno"));                         // Vincolo V2

CREATE DOMAIN MioEsito AS CHAR(8)
  CHECK (VALUE IN ("Promosso", "Respinto", "Debito"));             // Vincolo V5

CREATE TABLE Studente
  (
  Matricola          CHAR(10)          NOT NULL,
  Cognome            CHAR(50)          NOT NULL,
  Nome               CHAR(50)          NOT NULL,
  DataNascita        DATE              NOT NULL,
  Indirizzo          CHAR(255)         NOT NULL,
  Cap                CHAR(5)           NOT NULL,
  Citta              CHAR(30)          NOT NULL,
  PRIMARY KEY (Matricola)
  );

CREATE TABLE Classe
  (
  IdClasse           INT               NOT NULL,
  Numero             SMALLINT          NOT NULL,
  Sezione            CHAR(1)           NOT NULL,
  PRIMARY KEY (IdClasse),
  CHECK (Numero VALUE IN (1, 2, 3, 4, 5)),                         //Vincolo V3
  CHECK (Sezione VALUE IN ('A', 'B', 'C', 'D', 'E', 'F', 'G'))     // Vincolo V4
  );

CREATE TABLE Frequenta
  (
  Matricola1         CHAR(10)          NOT NULL,
  IdClasse1          INT               NOT NULL,
  Esito              MioEsito          NOT NULL,
  AS_Frequenta       CHAR(9)           NOT NULL,
  PRIMARY KEY (Matricola1, IdClasse1),
  FOREIGN KEY (Matricola1) REFERENCES ON Studente (Matricola)      // VR di chiave esterna
  [ON DELETE NO ACTION,]  ON UPDATE CASCADE,
  FOREIGN KEY (IdClasse1) REFERENCES ON Classe (IdClasse)          // VR di chiave esterna
  [ON DELETE NO ACTION,] ON UPDATE CASCADE );
```

```
CREATE TABLE Corso
  (
  CodCorso          CHAR(10)          NOT NULL,
  Titolo            CHAR(50)          NOT NULL,
  Descrizione       CHAR(255)         NOT NULL,
  DataInizio        DATE              NOT NULL,
  DataFine          DATE              NOT NULL,
  MonteOre          SMALLINT          NOT NULL,
  AS_Corso          CHAR(9)           NOT NULL,
  Tipologia         MiaTipologia      NOT NULL,
  Esito             MioEsito          NOT NULL,
  Descrizione       VARCHAR(1000)     NOT NULL,
  PRIMARY KEY (CodCorso),
  CHECK (DataInizio < DataFine),                    //Vincolo V6
  CHECK (MonteOre > 0)                              //Vincolo V7
  );

CREATE TABLE Partecipa
  (
  Matricola2        CHAR(10)          NOT NULL,
  CodCorso2         CHAR(10)          NOT NULL,
  PRIMARY KEY (Matricola2, CodCorso2),
  FOREIGN KEY (Matricola2) REFERENCES ON Studente (Matricola)      // VR di chiave esterna
  [ON DELETE NO ACTION,]  ON UPDATE CASCADE,
  FOREIGN KEY (CodCorso2) REFERENCES ON Corso (CodCorso)           // VR di chiave esterna
  [ON DELETE NO ACTION,] ON UPDATE CASCADE
  );

CREATE ASSERTION V8 CHECK (Corso.DataInizio > Studente.DataNascita);
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Data una classe ed un anno scolastico, visualizzare quali studenti di quella classe hanno frequentato corsi e di che tipo

SQL dapprima occorre connettersi al database e poi

```
USE Scuola;

SELECT Cognome, Nome, Titolo, Descrizione, DataInizio, DataFine
FROM Studente, Frequenta, Classe, Partecipa, Corso
WHERE (Studente.Matricola = Frequenta.Matricola1) AND (Frequenta.IdClasse1 = Classe.IdClasse)
AND (Studente.Matricola=Partecipa.Matricola2) AND (Partecipa.CodCorso2 = Corso.CodCorso)
AND (Classe.Numero =[InserisciNumero]) AND (Classe.Sezione =[InserisciSezione])
AND (AS_Frequenta = [InserisciAS]);
```

N.B. e' possibile ordinarli per Cognome e Nome

Q2. Dato uno studente, visualizzare quali corsi ha frequentato, di che tipo, per quale monte ore e in quale anno scolastico

SQL dapprima occorre connettersi al database e poi

USE Scuola;

SELECT Titolo, Descrizione, DataInizio, DataFine, MonteOre, AS_Corso
FROM Partecipa, Corso
WHERE (Partecipa.CodCorso2 = Corso.CodCorso)
AND (Partecipa.Matricola2 = [InserisciMatricola]);

oppure

SELECT Titolo, Descrizione, DataInizio, DataFine, MonteOre, AS_Corso
FROM Studente, Partecipa, Corso
WHERE (Studente.Matricola = Partecipa.Matricola2) **AND** (Partecipa.CodCorso2 = Corso.CodCorso)
AND (Studente.Cognome = [InserisciCognome]) **AND** (Studente.Nome = [InserisciNome]);

Q3. Dato un anno scolastico, visualizzare quali corsi interni sono stati attivati e da quali studenti sono stati seguiti

SQL dapprima occorre connettersi al database e poi

N.B. e' possibile ordinarli per CodCorso

USE Scuola;

SELECT CodCorso, Titolo, Tipologia, Cognome, Nome
FROM Studente, Partecipa, Corso
WHERE (Studente.Matricola=Partecipa.Matricola2) **AND** (Partecipa.CodCorso2 = Corso.CodCorso)
AND (AS_Corso = [InserisciAS]) **AND** (Tipologia = "Interno");

Q4. Dato un corso, visualizzare i dati relativi ad esso e per quali anni scolastici è stato attivato

SQL

SELECT CodCorso, Descrizione, Data Inizio, DataFine, MonteOre, AS_Corso
FROM Corso
WHERE (Titolo = [InserisciTitolo]);

Q5. Dato uno studente, visualizzare quali classi ha frequentato, in quali anni scolastici e con quali esiti finali

SQL dapprima occorre connettersi al database e poi

N.B. e' possibile ordinarli per anno scolastico (AS_frequenta)

USE Scuola;

SELECT AS_Frequenta, Numero, Sezione, Esito
FROM Frequenta, Classe
WHERE (Frequenta.IdClasse1 = Classe.IdClasse)
AND (Frequenta.Matricola1 = [Inserisci Matricola]);

oppure

SELECT AS_Frequenta, Numero, Sezione, Esito
FROM Studente, Frequenta, Classe
WHERE (Studente.Matricola = Frequenta.Matricola1) AND (Frequenta.IdClasse1 = Classe.IdClasse)
AND (Studente.Cognome = [InserisciCognome]) **AND** (Studente.Nome = [InserisciNome]);

Q6. Per ogni anno scolastico, contare il numero di studenti respinti

SQL dapprima occorre connettersi al database e poi

N.B. e' possibile ordinarli per anno scolastico (AS_Frequenta)

USE Scuola;

SELECT AS_Frequenta, COUNT(*) As NumRespinti
FROM Frequenta
WHERE (Esito = "Respinto")
GROUP BY AS_Frequenta;

Q7. Dato un anno scolastico, contare il numero totale di ore dei corsi organizzati per l'arricchimento dell'offerta formativa

SQL dapprima occorre connettersi al database e poi

USE Scuola;

SELECT SUM(MonteOre) AS TotOre
FROM Corso
WHERE (Tipologia = "Arricchimento")
AND (AS_Corso = [InserisciAS]);

Q8. Visualizzare l'elenco degli studenti che non hanno mai frequentato corsi di recupero

SQL dapprima occorre connettersi al database e poi

USE Scuola;

(**SELECT** Matricola, Cognome, Nome
FROM Studente)

MINUS

(**SELECT DISTINCT** Matricola, Cognome, Nome
FROM Studente, Partecipa, Corso
WHERE (Studente.Matricola=Partecipa.Matricola2) **AND** (Partecipa.CodCorso2 = Corso.CodCorso)
AND (Tipologia = "Recupero"));

(3) Una **biblioteca** vuole realizzare una base dati per gestire le sue **attività di classificazione, ricerca e prestito dei libri ai suoi soci**.

Per ogni socio si vogliono registrare i dati anagrafici e per ogni libro si vuole archiviare il titolo, l'autore, l'editore e l'anno di pubblicazione . inoltre si vogliono registrare le informazioni relative alla collocazione del libro nella biblioteca, ai suoi contenuti (attraverso parole chiave) ed ai prestiti del libro ai soci.

Per ogni libro esistono più copie in biblioteca ed un socio può prendere in prestito anche più di un libro contemporaneamente. E' fissato a 15 il numero dei giorni del prestito.

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Dato il nominativo di un autore, visualizzare i libri da lui scritti presenti in biblioteca;

Q2: Dato il nominativo di un socio, visualizzare i libri attualmente in prestito;

Q3: Data una parola chiave, ricercare i libri che la contengono;

Q4: Per ogni autore, determinare il numero dei libri presenti in biblioteca;

Q5: Dato il titolo di un libro, determinare il numero di copie presenti in biblioteca;

Q6: Determinare il numero di copie di libri attualmente in prestito;

Q7: Visualizzare il codice dei soci che attualmente non hanno libri in prestito;

Q8: Visualizzare l'elenco dei libri di cui esistono più copie in biblioteca.

e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

Premessa: omessa (vedi svolgimento esercizio precedente)

Da un'attenta lettura delle specifiche, si evidenzia che sono richieste le seguenti attività:

- gestione dei libri presenti in biblioteca;
- classificazione dei libri in base a parole chiave;
- gestione dei soci in biblioteca;
- gestione dei prestiti ne della restituzione dei libri.

Più in dettaglio, possiamo ipotizzare che sarà necessario raccogliere informazioni relative:

- ai *libri* della biblioteca;
- ai *soci* della biblioteca;
- agli *autori* dei libri;
- agli *editori* dei libri;
- alle *parole chiave* che caratterizzano i libri.

Abbiamo quindi ottenuto un primo elenco di entità che dovranno entrare a far parte dello schema concettuale della base dati.

E' importante notare che di ogni libro possono essere presenti in biblioteca anche più copie.

(Per la gestione di questo problema potrebbe anche essere introdotta una nuova entità chiamata Copia che conterrà le informazioni relative alle copie dei libri fisicamente presenti in biblioteca.

In altre parole, mentre l'entità Libro descrive le informazioni generali sul libro, l'entità Copia ne descrive la copia cartacea presente sullo scaffale che sarà oggetto del prestito al socio.

A questa entità apparterrebbero sia gli attributi Collocazione Data Prestito sia l'associazione "Esiste"tra le entità Libro e Copia di molteplicità 1:N sia l'associazione "PrendeInPrestito"tra le entità Socio e Copia di molteplicità 1:N).

Intendiamo in questo svolgimento come istanza dell'entità Libro ciascuna copia cartacea dello stesso

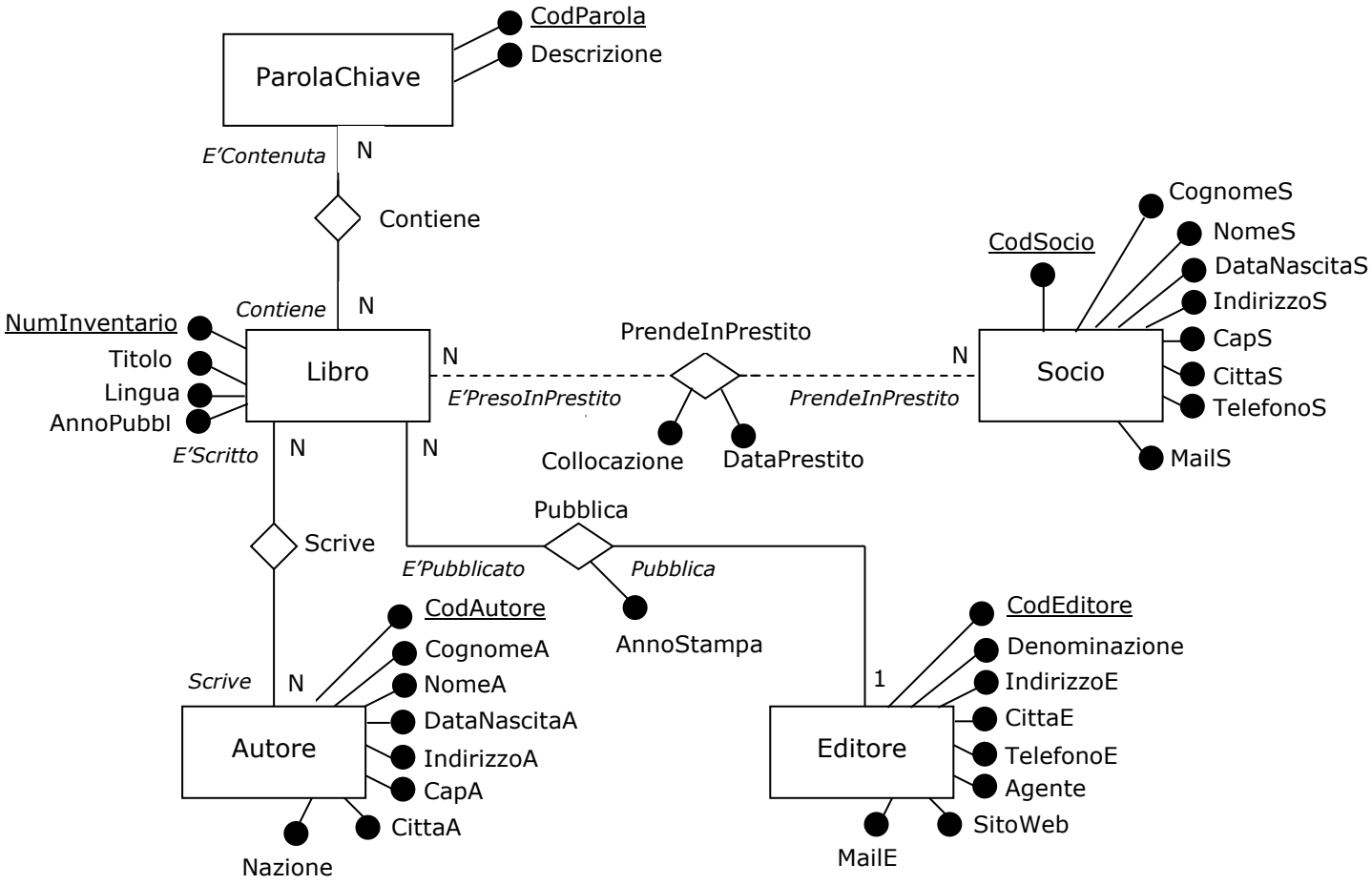
A partire da questo elenco, associamo a ciascuna entità individuata i corrispondenti attributi, attingendo dalle informazioni presenti nelle specifiche:

- a ciascun *Libro* deve essere associato un *Titolo*, un *AnnoPubblicazione* e la *Lingua* nel quale è scritto;
- a ciascun *Socio* deve essere associata un'*anagrafica breve* ed un eventuale *Telefono* e *Mail*
- a ciascun *Autore* deve essere associata un'*anagrafica breve* ed una *Nazione* di riferimento;
- a ciascun *Editore* deve essere associato una *Denominazione*, un *Indirizzo*, un *Agente* di riferimento, un *Telefono*, una *Mail* ed un eventuale indirizzo del *SitoWeb*;
- a ciascuna *ParolaChiave* deve essere associata la sua *Descrizione*

Passiamo ora ad esaminare quali sono le associazioni tra le entità ipotizzate, individuando per ciascuna di esse la molteplicità della associazione in base alle caratteristiche di funzionalità indicate nelle specifiche nonchè i relativi attributi:

- tra le entità *Autore* e *Libro* esiste un'associazione "Scrive" di molteplicità N:N (totale in entrambi i versi), in quanto *"un autore deve scrivere uno o più libri e viceversa un libro deve essere stato scritto da uno o più autori"* (anche "Anonimo" sarà una valida istanza dell'entità Autore per gestire i libri che no n risultassero avere nomi di autori validi);
- tra le entità *Editore* e *Libro* esiste un'associazione "Pubblica" di molteplicità 1:N (totale in entrambi i versi), in quanto *"un editore deve pubblicare uno o più libri e viceversa un libro è pubblicato da un editore"*: viene mantenuto come attributo *AnnoStampa* l'informazione sull'anno di stampa della copia del libro;
- tra le entità *Libro* e *ParolaChiave* esiste un'associazione "Contiene" di molteplicità N:N (totale in entrambi i versi) in quanto *"un libro deve contenere una o più parole chiave e viceversa una parola chiave deve essere contenuta in uno o più libri"*;
- tra le entità *Socio* e *Libro* esiste un'associazione "Prende in Prestito" di molteplicità N:N (parziale in entrambi i versi) in quanto *"un socio può prendere in prestito nessuna o più (copie di) un libro e viceversa un (copia di) libro può essere stata presa in prestito da nessuno o più soci":* a questa associazione appartengono gli attributi *Collocazione/NumeroInventario*, *DataPrestit*o per gestire le informazioni relative alla presa in prestito delle copie cartacee dei vari libri.

(a) Schema concettuale della realtà di interesse – diagramma E/R



Vincoli impliciti:

Sono quei vincoli direttamente deducibili dal diagramma E/R in quanto caratterizzati graficamente:

- vincoli di chiave primaria: tutti gli attributi sottolineati
- vincoli di integrità referenziale: totalità delle associazioni dirette e/o inverse (linea continua)

Vincoli espliciti:

Vincoli non deducibili direttamente dal diagramma E/R:

Nessuno

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Contiene" di molteplicità N:N tra le entità "Libro" e "ParolaChiave"

Libro (NumInventario, Titolo, Lingua, AnnoPubbl, Anno Stampa, CodEditore1)
con "CodEditore1" chiave esterna (foreign key) sull'attributo "CodEditore" della relazione "Editore"

ParolaChiave (CodParola, Descrizione)

Contiene (NumInventario1, CodParola1)
con" NumInventario1" chiave esterna (foreign key) sull'attributo "NumInventario" della relazione "Libro"
con "CodParola1" chiave esterna (foreign key) sull'attributo "CodParola" della relazione "ParolaChiave"

$VR_{NumInventario1}$ (Contiene) $\subseteq$ $VR_{NumInventario}$ (Libro) dal mapping relazionale dell'associazione N:N
$VR_{CodParola1}$ (Contiene) $\subseteq$ $VR_{CodParola}$ (ParolaChiave) dal mapping relazionale dell'associazione N:N

$VR_{NumInventario}$ (Libro) $\subseteq$ $VR_{NumInventario1}$ (Contiene) dalla TOTALITA' dell'associazione diretta "Contiene"
$VR_{CodParola}$ (ParolaChiave) $\subseteq$ $VR_{CodParola1}$ (Contiene) dalla TOTALITA' dell'associazione inversa "E'Contenuta"

(ii) mapping dell'associazione "Scrive" di molteplicità N:N tra le entità "Autore" e "Libro"

Autore (CodAutore, CognomeA, NomeA, DataNascitaA, IndirizzoA, CapA, CittaA, Nazione)

Libro già mappato in precedenza

Scrive (CodAutore2, NumInventario2)
con "CodAutore2" chiave esterna (foreign key) sull'attributo "CodAutore" della relazione "Autore"
con "NumInventario2" chiave esterna (foreign key) sull'attributo "NumInventario" della relazione "Libro"

$VR_{CodAutore2}$ (Scrive) $\subseteq$ $VR_{CodAutore}$ (Autore) dal mapping relazionale dell'associazione N:N
$VR_{NumInventario2}$ (Scrive) $\subseteq$ $VR_{NumInventario}$ (Libro) dal mapping relazionale dell'associazione N:N

$VR_{CodAutore}$ (Autore) $\subseteq$ $VR_{CodAutore2}$ (Scrive) dalla TOTALITA' dell'associazione diretta "Scrive"
$VR_{NumInventario}$ (Libro) $\subseteq$ $VR_{NumInventario2}$ (Scrive) dalla TOTALITA' dell'associazione inversa "E'Scritto"

(iii) mapping dell'associazione "Pubblica" di molteplicità 1:N tra le entità "Editore" e "Libro"

Editore (CodEditore, Denominazione, IndirizzoE, CittaE, TelefonoE, Agente, sito Web, MailE)

Libro già mappato in precedenza

$VR_{CodEditore}$ (Editore) $\subseteq$ $VR_{CodEditore1}$ (Libro) dalla TOTALITA' dell'associazione diretta "Pubblica"
$VR_{CodEditore1}$ (Libro) $\subseteq$ $VR_{CodEditore}$ (Editore) dalla TOTALITA' dell'associazione inversa "E'Pubblicato"

(iv) mapping dell'associazione "PrendeInPrestito" di molteplicità N:N tra le entità "Socio" e "Libro"

Socio (CodSocio, CognomeS, NomeS, DataNascitaS, IndirizzoS, CapS, CittaS, TelefonoS, MailS)

Libro già mappato in precedenza

PrendeInPrestito (CodSocio3, NumInventario3)
con "CodSocio3" chiave esterna (foreign key) sull'attributo "CodSocio" della relazione "Socio"
con "NumInventario3" chiave esterna (foreign key) sull'attributo "NumInventario" della relazione "Libro"

$VR_{CodSocio3}$ (PrendeInPrestito) $\subseteq$ $VR_{CodSocio}$ (Socio) dal mapping relazionale dell'associazione N:N
$VR_{NumInventario3}$ (PrendeInPrestito) $\subseteq$ $VR_{NumInventario}$ (Libro) dal mapping relazionale dell'associazione N:N

Vincoli di integrità intrarelazionali (o interni) ed interrelazionali (o esterni)

I vincoli di chiave primaria (impliciti nel modello E/R) sono mappati in *vincoli intrarelazionali su più ennuple.*

I vincoli di totalità di un'associazione (impliciti nel modello E/R) sono mappati in *vincoli interrelazionali di tipo referenziale*

I vincoli espliciti del diagramma E/R vengono mappati come segue:

NESSUNO

Normalizzazione: omessa (vedi svolgimento esercizio precedente)

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE Biblioteca;

USE Biblioteca;

CREATE TABLE Libro
  (
  NumInventario     CHAR(10)          NOT NULL,
  Titolo            CHAR(50)          NOT NULL,
  Lingua            CHAR(20)          NOT NULL,
  AnnoPubbl         CHAR(4)           NOT NULL,
  AnnoStampa        CHAR(4)           NOT NULL,
  CodEditore1       CHAR(10),
  PRIMARY KEY (NumInventario),
  FOREIGN KEY (CodEditore1) REFERENCES Editore (CodEditore)      // VR di chiave esterna
  [ON DELETE NO ACTION]  ON UPDATE CASCADE
  );

CREATE TABLE ParolaChiave
  (
  CodParola         CHAR(10)          NOT NULL,
  Descrizione       CHAR(50)          NOT NULL,
  PRIMARY KEY (CodParola)
  );

CREATE TABLE Contiene
  (
  NumInventario1    CHAR(10)          NOT NULL,
  CodParola1        CHAR(10)          NOT NULL,
  PRIMARY KEY (NumInventario1, CodParola1),
  FOREIGN KEY (NumInventario1) REFERENCES Libro (NumInventario)     // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (CodParola1) REFERENCES PaolaChiave (CodParola)       // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE Autore
  (
  CodAutore         CHAR(10)          NOT NULL,
  CognomeA          CHAR(50)          NOT NULL,
  NomeA             CHAR(50)          NOT NULL,
  DataNascitaA      DATE              NOT NULL,
  IndirizzoA        CHAR(255)         NOT NULL,
  CapA              CHAR(5)           NOT NULL,
```

```
  CittaA            CHAR(30)          NOT NULL,
  Nazione           CHAR(30),
  PRIMARY KEY (CodAutore)
  );

CREATE TABLE Scrive
  (
  CodAutore2        CHAR(10)          NOT NULL,
  NumInventario2    CHAR(10)          NOT NULL,
  PRIMARY KEY (CodAutore2, NumInventario2),
  FOREIGN KEY (CodAutore2) REFERENCES Autore (CodAutore)      // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  FOREIGN KEY (NumInventario2) REFERENCES Libro (NumInventario)    // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE
  );

CREATE TABLE Editore
  (
  CodEditore        CHAR(10)          NOT NULL,
  Denominazione     CHAR(50)          NOT NULL,
  IndirizzoE        CHAR(255)         NOT NULL,
  CittaE            CHAR(30)          NOT NULL,
  TelefonoE         CHAR(30),
  Agente            CHAR (100)        NOT NULL,
  Sito Web          CHAR(255),
  MailE             CHAR(30),
  PRIMARY KEY (CodEditore)
  );

CREATE TABLE Socio
  (
  CodSocio          CHAR(10)          NOT NULL,
  CognomeS          CHAR(50)          NOT NULL,
  NomeS             CHAR(50)          NOT NULL,
  DataNascitaS      DATE              NOT NULL,
  IndirizzoS        CHAR(255)         NOT NULL,
  CapS              CHAR(5)           NOT NULL,
  CittaS            CHAR(30)          NOT NULL,
  TelefonoS         CHAR(30),
  MailS             CHAR(30),
  PRIMARY KEY (CodSocio)
  );

CREATE TABLE PrendeInPrestito
  (
  CodSocio3         CHAR(10)          NOT NULL,
  NumInventario3    CHAR(10)          NOT NULL,
  PRIMARY KEY (CodSocio3, NumInventario3),
  FOREIGN KEY (CodSocio3) REFERENCES Socio (CodSocio)      // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  FOREIGN KEY (NumInventario3) REFERENCES Libro (NumInventario)    // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE
  );
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Dato il nominativo di un autore, visualizzare i libri da lui scritti presenti in biblioteca

SQL: prima occorre connettersi al database poi

USE Biblioteca;

N.B. e' possibile ordinarli per titolo

```
SELECT Titolo
FROM Autore, Scrive, Libro
WHERE (Autore.CodAutore = Scrive.CodAutore2) AND (Scrive.NumInventario2 = Libro.NumInventario)
AND (CognomeA = [InserisciCognome]) AND (NomeA = [InserisciNome])
GROUP BY Titolo;  //perché di uno stesso libro possono esserci n copie
```

Q2. Dato il nominativo di un socio, visualizzare i libri attualmente in prestito

SQL: prima occorre connettersi al database poi

USE Biblioteca;

N.B. e' possibile ordinarli per titolo

```
SELECT Titolo
FROM Socio, PrendeInPrestito, Libro
WHERE (Socio.CodSocio = PrendeInPrestito.CodSocio3) AND (PrendeInPrestito.NumInventario3 = Libro.NumInventario)
AND (CognomeS = [InserisciCognome]) AND (NomeS = [InserisciNome]);
```

Q3. Data una parola chiave, ricercare i libri che la contengono

SQL: prima occorre connettersi al database poi

USE Biblioteca;

N.B. e' possibile ordinarli per titolo

```
SELECT Titolo
FROM Libro, Contiene, ParolaChiave
WHERE (Libro.NumInventario = Contiene.NumInventario1) AND (Contiene.CodParola1 = ParolaChiave.Codparola)
AND (Descrizione = [InserisciParola])
GROUP BY Titolo;  //perché di uno stesso libro possono esserci n copie
```

Q4. Per ogni autore, determinare il numero dei libri presenti in biblioteca

SQL: prima occorre connettersi al database poi

USE Biblioteca;

N.B. e' possibile ordinarli per cognome e nome autore

```
SELECT CognomeA, NomeA, COUNT(*) AS NumLibri
FROM Autore, Scrive
WHERE (Autore.CodAutore = Scrive.CodAutore2)
GROUP BY CognomeA, NomeA;  //perché di uno stesso libro possono esserci n copie
```

Q5. Dato il titolo di un libro, determinare il numero di copie presenti in biblioteca

SQL: prima occorre connettersi al database poi

USE Biblioteca;

N.B. e' possibile ordinarli per titolo

```
SELECT Titolo, COUNT(*) AS NumCopie
FROM Libro
WHERE (Titolo = [InserisciTitolo])
```

GROUP BY Titolo; //perché di uno stesso libro possono esserci n copie

Q6. Determinare il numero di copie di libri attualmente in prestito

SQL: prima occorre connettersi al database poi

USE Biblioteca;

SELECT COUNT(*) AS NumPrestiti
FROM PrendeInPrestito;
WHERE (DataPrestito IS NOT NULL);

Q7. Visualizzare il codice dei soci che attualmente non hanno libri in prestito

SQL: prima occorre connettersi al database poi

USE Biblioteca;

(**SELECT** CodSocio
FROM Socio)

MINUS

(**SELECT DISTINCT** Socio.CodSocio
FROM Socio, PrendeInPrestito
WHERE (Socio.CodSocio = PrendeInprestito. CodSocio3) AND (Dataprestito IS NOT NULL));

Q8. Visualizzare l'elenco dei libri di cui esistono più copie in biblioteca

SQL: prima occorre connettersi al database poi

USE Biblioteca;

SELECT Titolo, COUNT(*) AS NumCopie
FROM Libro
GROUP BY Titolo; //perché di uno stesso libro possono esserci n copie
HAVING COUNT(*) >= 2;

N.B. e' possibile ordinarli in senso decrescente per numero di copie

(4) In un **istituto scolastico** si vogliono organizzare degli **scambi** tra un gruppo di studenti dell'istituto ed un gruppo di studenti stranieri al fine di migliorare la conoscenza delle lingue.

Per realizzare gli scambi ed al fine di scegliere un'opportuna sistemazione presso una famiglia straniera, si raccolgono presso gli studenti interessati i seguenti dati: cognome, nome, data di nascita, classe e sezione frequentata, numero di fratelli e sorelle, tipo di professione esercitata dal padre e dalla madre.

Ogni scambio è individuato da un codice, la scuola straniera collegata, la nazione di appartenenza, il numero di studenti coinvolti, la data di inizio e di fine, l'anno scolastico in cui avviene e l'elenco degli studenti che vi partecipano.

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Elenco degli studenti che hanno effettuato un determinato scambio con una data scuola;

Q2: Elenco di tutti gli scambi effettuati nello stesso anno scolastico;

Q3: Elenco degli studenti il cui padre esercita una determinata professione;

Q4: Cognome e nome di tutti gli studenti che hanno partecipato ad uno scambio con scuole del Portogallo.

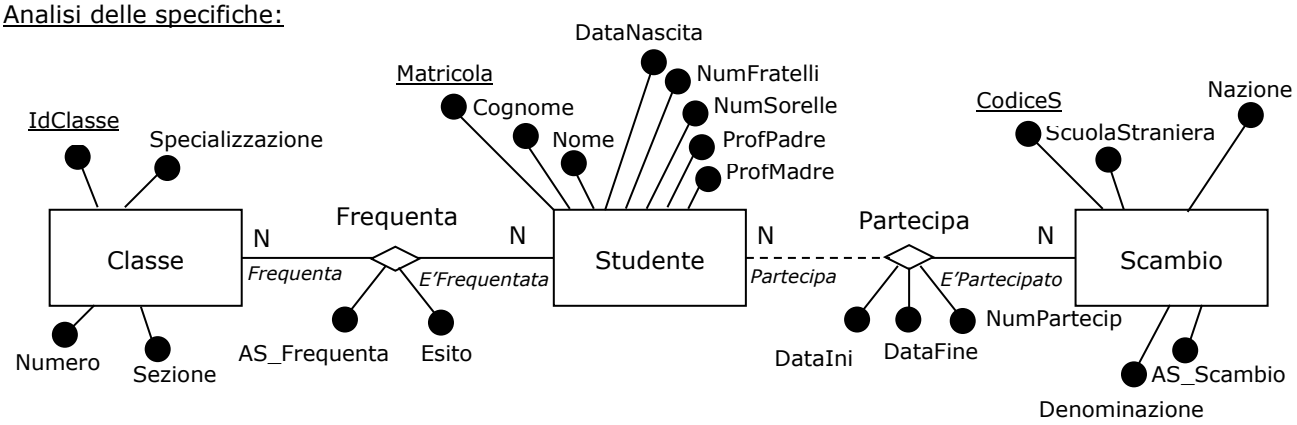
e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

(a) Schema concettuale della realtà di interesse – diagramma E/R

Premessa: omessa (vedi svolgimento esercizio precedente)

Analisi delle specifiche:



Vincoli impliciti:

Vincoli espliciti:

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Frequenta" di molteplicità N:N tra le entità "Classe" e "Studente"

Classe (IdClasse, Numero, Sezione, Specializzazione)
Studente (Matricola, Cognome, Nome, DataNascita, NumFratelli, NumSorelle, ProfPadre, ProfMadre)
Frequenta (Matricola1, IdClasse1, AS_Frequenta, Esito)

con "Matricola1" chiave esterna (foreign key) sull'attributo "Matricola" della relazione "Studente"
con "IdClasse1" chiave esterna (foreign key) sull'attributo "IdClasse" della relazione "Classe"

$VR_{Matricola1}$ (Frequenta) ⊆ $VR_{Matricola}$ (Studente) dal mapping relazionale dell'associazione N:N
$VR_{IdClasse1}$ (Frequenta) ⊆ $VR_{IdClasse}$ (Classe) dal mapping relazionale dell'associazione N:N

$VR_{Matricola}$ (Studente) ⊆ $VR_{Matricola1}$ (Frequenta) dalla TOTALITA' dell'associazione diretta "Frequenta"
$VR_{IdClasse}$ (Classe) ⊆ $VR_{IdClasse1}$ (Frequenta) dalla TOTALITA' dell'associazione inversa "E'Frequentata"

(ii) mapping dell'associazione "Partecipa" di molteplicità N:N tra le entità "Studente" e "Scambio"

Studente già mappato in precedenza
Scambio (CodiceS, Denominazione, ScuolaStraniera, Nazione, AS_Scambio)
Partecipa (Matricola2, CodiceS2, DataIni, DataFine, NumPart)
con "Matricola2" chiave esterna (foreign key) sull'attributo "Matricola" della relazione "Studente"
con "CodiceS2" chiave esterna (foreign key) sull'attributo "CodiceS" della relazione "Scambio"

$VR_{Matricola2}$ (Partecipa) ⊆ $VR_{Matricola}$ (Studente) dal mapping relazionale dell'associazione N:N
$VR_{CodiceS2}$ (Partecipa) ⊆ $VR_{CodiceS}$ (Scambio) dal mapping relazionale dell'associazione N:N

$VR_{Matricola}$ (Studente) ⊆ $VR_{Matricola2}$ (Partecipa) dalla TOTALITA' dell'associazione diretta "Partecipa"
$VR_{CodiceS}$ (Scambio) ⊆ $VR_{CodiceS2}$ (Partecipa) dalla TOTALITA' dell'associazione inversa "E'Partecipato"

Mapping dei vincoli nel modello relazionale:

Normalizzazione: omessa (vedi svolgimento esercizio precedente)

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE Scambi Culturali;
USE ScambiCulturali;

CREATE DOMAIN MioEsito AS CHAR(8)
  CHECK (VALUE IN ("Promosso", "Respinto", "Debito"));

CREATE TABLE Classe
  (
  IdClasse          CHAR(10)        NOT NULL,
  Numero            SMALLINT        NOT NULL,
  Sezione           CHAR(1)         NOT NULL,
  Specializzazione  CHAR(50),
  PRIMARY KEY (IdClasse)
  );

CREATE TABLE Studente
  (
  Matricola         CHAR(10)        NOT NULL,
  Cognome           CHAR(50)        NOT NULL,
  Nome              CHAR(50)        NOT NULL,
  DataNascita       DATE            NOT NULL,
  NumFratelli       SMALLINT        DEFAULT 0,
  NumSorelle        SMALLINT        DEFAULT 0,
  ProfPadre         CHAR(50)        NOT NULL,
  ProfMadre         CHAR(50),
  PRIMARY KEY (Matricola),
  CHECK (Numero VALUE IN (1, 2, 3, 4, 5)),
  CHECK (Sezione VALUE IN ('A', 'B', 'C', 'D', 'E', 'F', 'G')),
  CHECK (NumFratelli >= 0),
  CHECK (NumSorelle >= 0)
  );
```

```
CREATE TABLE Frequenta
  (
  Matricola1          CHAR(10)          NOT NULL,
  IdClasse1           CHAR(10)          NOT NULL,
  AS_Frequenta        CHAR(4)           NOT NULL,
  Esito               MioEsito          NOT NULL,
  PRIMARY KEY (Matricola1, IdClasse1),
  FOREIGN KEY (Matricola1) REFERENCES Studente (Matricola)     // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (IdClasse1) REFERENCES IdClasse (Classe)        // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE Scambio
  (
  CodiceS             CHAR(10)          NOT NULL,
  Denominazione       CHAR(50)          NOT NULL,
  ScuolaStraniera     CHAR(50)          NOT NULL,
  Nazione             CHAR(50)          NOT NULL,
  AS_Scambio          CHAR(4)           NOT NULL,
  PRIMARY KEY (CodiceS)
  );

CREATE TABLE Partecipa
  (
  Matricola2          CHAR(10)          NOT NULL,
  CodiceS2            CHAR(10)          NOT NULL,
  PRIMARY KEY (Matricola2, CodiceS2),
  FOREIGN KEY (Matricola2) REFERENCES Studente (Matricola)     // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (CodiceS2) REFERENCES CodiceS (Scambio)        // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  );
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Elenco degli studenti che hanno effettuato un determinato scambio con una data scuola

SQL: prima occorre connettersi al database poi

USE ScambiCulturali;

SELECT AS_Scambio, Cognome, Nome
FROM Studente, Partecipa, Scambio
WHERE (Studente.Matricola = Partecipa.Matricola2) **AND** (Partecipa.CodiceS2 = Scambio.CodiceS)
AND (ScuolaStraniera = [InserisciScuola]);
[**ORDER BY** AS_Scambio]

oppure

SELECT AS_Scambio, Cognome, Nome
FROM Studente, Partecipa, (**SELECT** * **FROM** Scambio **WHERE** (ScuolaStraniera = [InserisciScuola]) AS t1)
WHERE (Studente.Matricola = Partecipa.Matricola2) **AND** (Partecipa.CodiceS2 = t1.CodiceS);
[**ORDER BY** AS_Scambio]

Algebra Relazionale

A = { AS_Scambio, Cognome, Nome }

P = { (ScuolaStraniera = [InserisciScuola]) }

$$Q1 = \Pi_A (\sigma_P (\text{Studente} \underset{\text{Matricola = Matricola2}}{\triangleright\triangleleft} (\text{Partecipa} \underset{\text{CodiceS2 = CodiceS}}{\triangleright\triangleleft} \text{Scambio})))$$

oppure

$$Q1 = \Pi_A (\text{Studente} \underset{\text{Matricola = Matricola2}}{\triangleright\triangleleft} (\text{Partecipa} \underset{\text{CodiceS2 = CodiceS}}{\triangleright\triangleleft} \sigma_P (\text{Scambio})))$$

Q2. Elenco di tutti gli scambi effettuati nello stesso anno scolastico

USE ScambiCulturali;

SELECT AS_Scambio, Denominazione
FROM Scambio;
[ORDER BY AS_Scambio;]

Algebra Relazionale

A = { AS_Scambio, Denominazione }

$$Q2 = \Pi_A (\text{Scambio})$$

Q3. Elenco degli studenti il cui padre esercita una determinata professione

USE ScambiCulturali;

SELECT Cognome, Nome
FROM Studente
WHERE (ProfPadre = [InserisciProfessione]);

Algebra Relazionale

A = { Cognome, Nome }

P = { (ProfPadre = [InserisciProfessione]) }

$$Q3 = \Pi_{A}(\sigma_{P}(\text{Studente}))$$

Q4. Cognome e nome di tutti gli studenti che hanno partecipato ad uno scambio con scuole del Portogallo

USE ScambiCulturali;

SELECT Cognome, Nome
FROM Studente, Partecipa, Scambio
WHERE (Studente.Matricola = Partecipa.Matricola2) **AND** (Partecipa.CodiceS2 = Scambio.CodiceS)
AND (Nazione = "PORTOGALLO");

oppure

SELECT Cognome, Nome
FROM Studente, Partecipa, (**SELECT** * **FROM** Scambio **WHERE** (Nazione = "PORTOGALLO") AS t1)
WHERE (Studente.Matricola = Partecipa.Matricola2) **AND** (Partecipa.CodiceS2 = t1.CodiceS);

Algebra Relazionale

A = { Cognome, Nome }

P = { (Nazione = "PORTOGALLO") }

$$Q4 = \Pi_{A}(\sigma_{P}(\text{Studente} \underset{\text{Matricola = Matricola2}}{\triangleright\!\triangleleft} (\text{Partecipa} \underset{\text{CodiceS2 = CodiceS}}{\triangleright\!\triangleleft} \text{Scambio}))$$

oppure

$$Q4 = \Pi_{A}(\text{Studente} \underset{\text{Matricola = Matricola2}}{\triangleright\!\triangleleft} (\text{Partecipa} \underset{\text{CodiceS2 = CodiceS}}{\triangleright\!\triangleleft} \sigma_{P}(\text{Scambio})))$$

(5) Le informazioni relative alle **attività sportive studentesche** devono essere organizzate in una base dati. Gli **studenti**, dei quali si conservano le informazioni anagrafiche, frequentano gli **istituti superiori**, e possono partecipare ad una o più **manifestazioni sportive** (specialità sportive diverse, giornate diverse, campionati che durano mesi o gare di un giorno).

Per ogni attività sportiva le scuole indicano un **professore** che svolge la funzione di riferimento e di allenatore: ogni professore segue una sola manifestazione, ma una stessa manifestazione può essere seguita da professori diversi di scuole diverse.

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Numero degli studenti che partecipano ad una determinata manifestazione sportiva;

Q2: Elenco anagrafico degli allenatori di un'attività sportiva;

Q3: Elenco delle scuole (denominazione) con il numero di studenti che partecipano alle attività sportive;

Q4: Elenco delle scuole (con denominazione, indirizzo, telefono) con studenti che partecipano ad una determinata manifestazione sportiva;

Q5: Elenco allenatori (cognome e nome) e scuole (denominazione) di appartenenza in ordine alfabetico;

Q6: Numero degli studenti partecipanti ad una determinata scuola per ciascuna delle manifestazioni sportive.

e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

Premessa: omessa (vedi svolgimento esercizio precedente)

Da un'attenta lettura delle specifiche si evidenzia che sono richieste le seguenti attività:

- gestione delle attività sportive studentesche;
- gestione degli abbinamenti dei professori-allenatori con le rispettive manifestazioni.

Possiamo quindi ipotizzare di avere entità contenenti le informazioni relative:

- agli *studenti* delle diverse scuole che partecipano alle manifestazioni sportive;
- ai *professori* per rappresentare i docenti che svolgono la funzione di referente-allenatore;
- agli *istituti* di appartenenza di studenti e professori;
- alle *manifestazioni* (sportive) per rappresentare le varie tipologie di gare studentesche.

Abbiamo quindi ottenuto un primo elenco di entità che dovranno entrare a far parte dello schema concettuale della base di dati che si sta progettando.

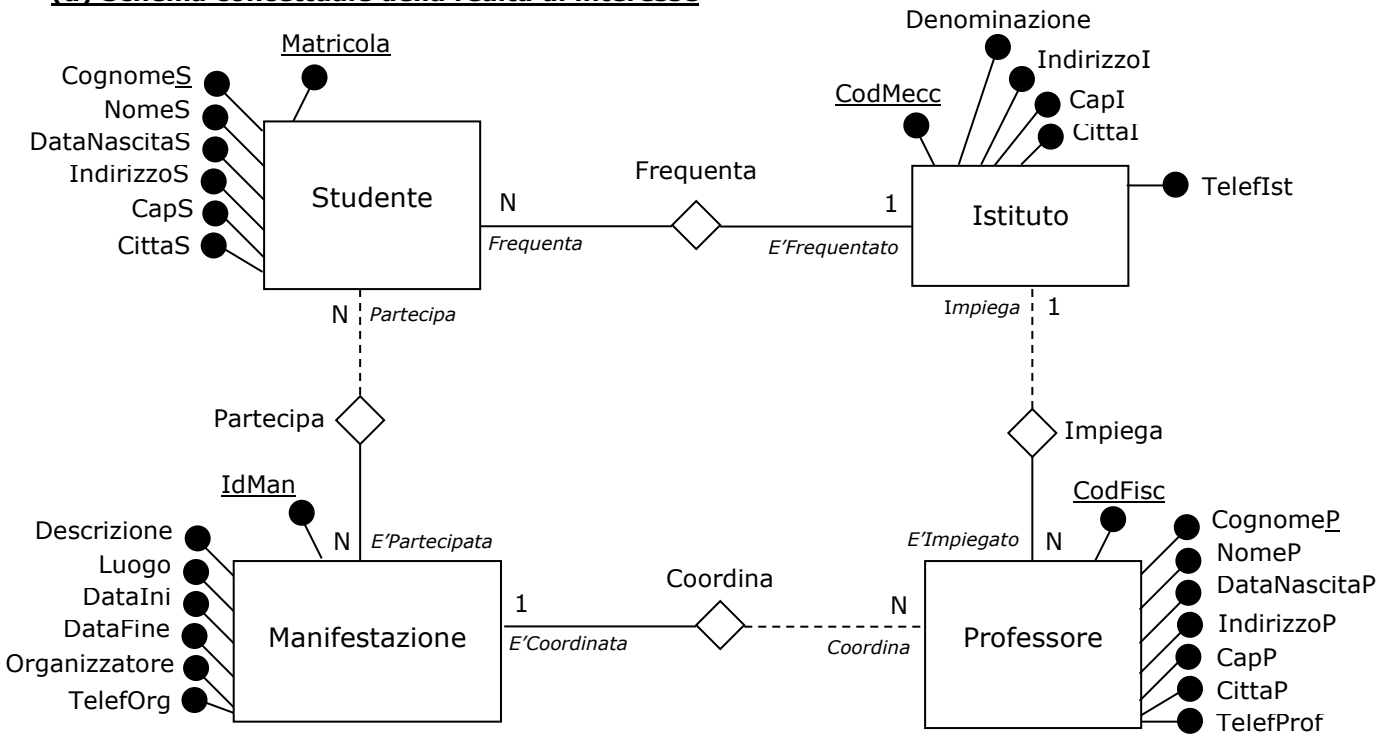
A partire da questo elenco, associamo a ciascuna entità individuata i corrispondenti attributi, attingendo sempre le informazioni dalle specifiche fornite:

- a ciascuno studente devono essere associati una *Matricola*, un *Cognome*, un *Nome*, una *DataNascita*, un *Indirizzo*, un *Cap*, una *Città* (anagrafica breve);
- a ciascun professore devono essere associati una *CodiceFiscale*, un *Cognome*, un *Nome*, una *DataNascita*, un *Indirizzo*, un *Cap*, una *Città* (anagrafica breve), un *Telefono*;
- a ciascun istituto deve essere associata un *CodiceMeccanografico*, una *Denominazione*, un *Indirizzo* ed un *Telefono*;
- a ciascuna manifestazione devono essere associati un *Identificativo* della manifestazione, una *Descrizione*, un *Luogo*, una *DataInizio*, una *DataFine*, un *Organizzatore*, un T*elefono* dell'organizzatore

Passiamo ora ad esaminare quali sono le associazioni tra le entità ipotizzate, individuando per ciascuna di esse la molteplicità dell'associazione in base alle caratteristiche di funzionalità evidenziate nelle specifiche più eventuali attributi:

- tra le entità *Studente* e *Istituto* esiste un'associazione "Frequenta" di molteplicità N:1 (totale in entrambi i versi) in quanto *"uno studente deve frequentare un istituto e viceversa un istituto deve essere frequentato da uno o più studenti"*;
- tra le entità *Istituto* e *Professore* esiste un'associazione "Impiega" di molteplicità 1:N (con diretta parziale ed inversa totale) in quanto *"un istituto può impiegare nessuno o più professori e viceversa un professore deve essere impiegato da un istituto"*;
- tra le entità *Professore* e *Manifestazione* esiste un'associazione "Coordina" di molteplicità N:1 (con diretta parziale ed inversa totale) in quanto *"un professore può coordinare nessuna o una manifestazione e viceversa una manifestazione deve essere coordinata da uno o più professori"*;
- tra le entità *Studente* e *Manifestazione* esiste un'associazione "Partecipa" di molteplicità N:N (con diretta parziale ed inversa totale) in quanto *"uno studente può partecipare a nessuna o a più manifestazioni e viceversa una manifestazione deve essere partecipata da uno o più studenti*;

(a) Schema concettuale della realtà di interesse



Vincoli impliciti:

Vincoli espliciti:

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Partecipa" di molteplicità N:N tra le entità "Studente" e "Manifestazione"

Studente (Matricola, CognomeS, NomeS, DataNascitaS, IndirizzoS, CapS, CittaS, CodMecc1)
con "CodMecc1" chiave esterna (foreign key) sull'attributo "CodMecc" della relazione "Istituto"

Manifestazione (IdMan, Descrizione, Luogo, DataIni, DataFine, Organizzatore, TelefOrg)

Partecipa (Matricola1, IdMan1)
con "Matricola1" chiave esterna (foreign key) sull'attributo "Matricola" della relazione "Studente"
con "IdMan1" chiave esterna (foreign key) sull'attributo "IdMan" della relazione "Manifestazione"

$VR_{Matricola1}$ (Partecipa) $\subseteq$ $VR_{Matricola}$ (Studente) dal mapping relazionale dell'associazione N:N
VR_{IdMan1} (Partecipa) $\subseteq$ VR_{IdMan} (Manifestazione) dal mapping relazionale dell'associazione N:N

VR_{IdMan} (Manifestazione) $\subseteq$ VR_{IdMan1} (Partecipa) dalla TOTALITA' dell'associazione inversa "E'Partecipata"

(ii) mapping dell'associazione "Coordina" di molteplicità N:1 tra le entità "Professore" e "Manifestazione"

Professore (CodFisc, CognomeP, NomeP, DataNascitaP, IndirizzoP, CapP, CittaP, TelefProf, IdMan2, CodMecc2)

con "IdMan2" chiave esterna (foreign key) sull'attributo "IdMan" della relazione "Manifestazione"
con "CodMecc2" chiave esterna (foreign key) sull'attributo "CodMecc" della relazione "Istituto"

Manifestazione già mappata in precedenza

VR_{IdMan} (Manifestazione) $\subseteq$ VR_{IdMan2} (Professore) dalla TOTALITA' dell'associazione inversa "E'Coordinata"

(iii) mapping dell'associazione "Impiega" di molteplicità 1:N tra le entità "Istituto" e "Professore"

Professore già mappato in precedenza
Istituto (CodMecc, Denominazione, IndirizzoI, CapI, CittaI, TelefIst)

$VR_{CodMecc2}$ (Professore) $\subseteq$ $VR_{CodMecc}$ (Istituto) dalla TOTALITA' dell'associazione inversa "E'Impiegato"

(iv) mapping dell'associazione "Frequenta" di molteplicità N:1 tra le entità "Studente" e "Istituto"

Studente già mappato in precedenza
Istituto già mappato in precedenza

$VR_{CodMecc2}$ (Studente) $\subseteq$ $VR_{CodMecc}$ (Istituto) dalla TOTALITA' dell'associazione diretta "Frequenta"
$VR_{CodMecc}$ (Istituto) $\subseteq$ $VR_{CodMecc2}$ (Studente) dalla TOTALITA' dell'associazione inversa "E'Frequentato"

Mapping dei vincoli nel modello relazionale:

Normalizzazione: omessa (vedi svolgimento esercizio precedente)

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE GareSportive;

USE GareSportive;

CREATE TABLE Studente
  (
  Matricola          CHAR(10)          NOT NULL,
  CognomeS           CHAR(50)          NOT NULL,
  NomeS              CHAR(50)          NOT NULL,
  DataNascitaS       DATE              NOT NULL,
  IndirizzoS         CHAR(255),
  CapS               CHAR(5),
  CittaS             CHAR(30),
  CodMecc1           CHAR(10),
  PRIMARY KEY (Matricola),
  FOREIGN KEY (CodMecc1) REFERENCES CodMecc (Istituto)     // VR di chiave esterna
  ON DELETE SET NULL  ON UPDATE CASCADE,
  );

CREATE TABLE Manifestazione
  (
  IdMan              CHAR(10)          NOT NULL,
  Descrizione        CHAR(50)          NOT NULL,
  Luogo              CHAR(50)          NOT NULL,
  DataIni            DATE              NOT NULL,
```

```
  DataFine          DATE              NOT NULL,
  Organizzatore     CHAR(50),
  TelefOrg          CHAR(30),
  PRIMARY KEY (IdMan),
  CHECK (DataIni <= DataFine)
  );

CREATE TABLE Partecipa
  (
  Matricola1        CHAR(10)          NOT NULL,
  IdMan1            CHAR(10)          NOT NULL,
  PRIMARY KEY (Matricola1, IdMan1),
  FOREIGN KEY (Matricola1) REFERENCES Studente (Matricola)     // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (IdMan1) REFERENCES IdMan (Manifestazione)       // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE TABLE Professore
  (
  CodFisc           CHAR(16)          NOT NULL,
  CognomeP          CHAR(50)          NOT NULL,
  NomeP             CHAR(50)          NOT NULL,
  DataNascitaP      DATE              NOT NULL,
  IndirizzoP        CHAR(255),
  CapP              CHAR(5),
  CittaP            CHAR(30),
  TelefProf         CHAR(30),
  IdMan2            CHAR(10)
  CodMecc2          CHAR(10)
  PRIMARY KEY (Matricola),
  FOREIGN KEY (IdMan2) REFERENCES IdMan (Manifestazione)     // VR di chiave esterna
  ON DELETE SET NULL  ON UPDATE CASCADE,
  FOREIGN KEY (CodMecc2) REFERENCES CodMecc (Istituto)     // VR di chiave esterna
  ON DELETE SET NULL  ON UPDATE CASCADE
  );

CREATE TABLE Istituto
  (
  CodMecc           CHAR(10)          NOT NULL,
  Denominazione     CHAR(50)          NOT NULL,
  IndirizzoI        CHAR(255)         NOT NULL,
  CapI              CHAR(5),
  CittaI            CHAR(30)          NOT NULL,
  TelefOrg          CHAR(30),
  PRIMARY KEY (CodMecc)
  );
```

CREATE ASSERTION V_x **CHECK** (Professore.DataNascitaP < Studente.DataNascitaS);
CREATE ASSERTION V_y **CHECK** (Manifestazione.DataIni > Studente.DataNascitaS);
CREATE ASSERTION V_z **CHECK** (Manifestazione.DataIni > Professore.DataNascitaP);

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Numero degli studenti che partecipano ad una determinata manifestazione sportiva

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT COUNT(*) AS NumPartecipanti
FROM Partecipa, Manifestazione
WHERE (Partecipa.IdMan1 = Manifestazione.IdMan)
AND (Descrizione = [InserisciNomeGara]);

Q2. Elenco anagrafico degli allenatori di un'attività sportiva

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT CognomeP, NomeP
FROM Professore, Manifestazione
WHERE (Professore.IdMan2 = Manifestazione.IdMan) **AND** (Descrizione = [InserisciNomeGara])
ORDER BY CognomeP, NomeP;

Q3. Elenco delle scuole (denominazione) con il numero di studenti che partecipano alle attività sportive

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT Denominazione, COUNT(DISTINCT Matricola) AS Numpartecipanti
FROM Istituto, Studente, Partecipa
WHERE (Istituto.CodMecc = Studente.CodMecc1) **AND** (Studente.Matricola = Partecipa.Matricola1)
GROUP BY Denominazione;

Q4. Elenco delle scuole (con denominazione, indirizzo, telefono) con studenti che partecipano ad una determinata manifestazione sportiva

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT Denominazione, IndirizzoI, TelefIst
FROM Istituto, Studente, Partecipa, Manifestazione
WHERE (Istituto.CodMecc = Studente.CodMecc1) **AND** (Studente.Matricola = Partecipa.Matricola1)
AND (Partecipa.IdMan1 = Manifestazione.IdMan) **AND** (Descrizione = [InserisciNomeGara])
GROUP BY Denominazione, IndirizzoI, TelefIst;

Q5. Elenco allenatori (cognome e nome) e scuole (denominazione) di appartenenza in ordine alfabetico

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT CognomeP, NomeP, Denominazione
FROM Istituto, Professore
WHERE (Istituto.CodMecc = Professore.CodMecc2)
ORDER BY Cognome P, Nome P;

Q6. Numero degli studenti partecipanti ad una determinata scuola per ciascuna delle manifestazioni sportive

SQL: prima occorre connettersi al database poi

USE GareSportive;

SELECT Descrizione, COUNT(*) AS NumPartecipanti
FROM Istituto, Studente, Partecipa, Manifestazione
WHERE (Istituto.CodMecc = Studente.CodMecc1) **AND** (Studente.Matricola = Partecipa.Matricola1)
AND (Partecipa.IdMan1 = Manifestazione.IdMan) **AND** (Denominazione = [InserisciNomeIstituto])
GROUP BY Descrizione;

(6) Una **galleria d'arte** ha deciso di creare un sistema che **consenta via web ai suoi clienti registrati** di consultare **il catalogo completo dei quadri** in listino ed ad un proprio utente amministratore di inserire, modificare o cancellare le informazioni relative agli stessi.

Per ogni quadro presente in galleria è compilata una scheda che riporta l'autore, il titolo, la tipologia tecnica realizzativa (olio, tempera, carboncino, litografia, etc.), le dimensioni, il prezzo, nonché l'immagine illustrativa dell'opera.

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Elencare tutte le opere realizzate con una determinata tecnica pittorica;

Q2: Elencare tutte le opere realizzate da un determinato artista il cui prezzo è inferiore ai 300 euro;

Infine:

e) si provi a progettare ed a sviluppare in dettaglio utilizzando un linguaggio di programmazione lato server una delle seguenti funzioni:

(-) pagina di accesso all'area riservata ai clienti registrati;

(-) pagina riservata ai clienti per la consultazione del catalogo e delle singole schede delle opere;

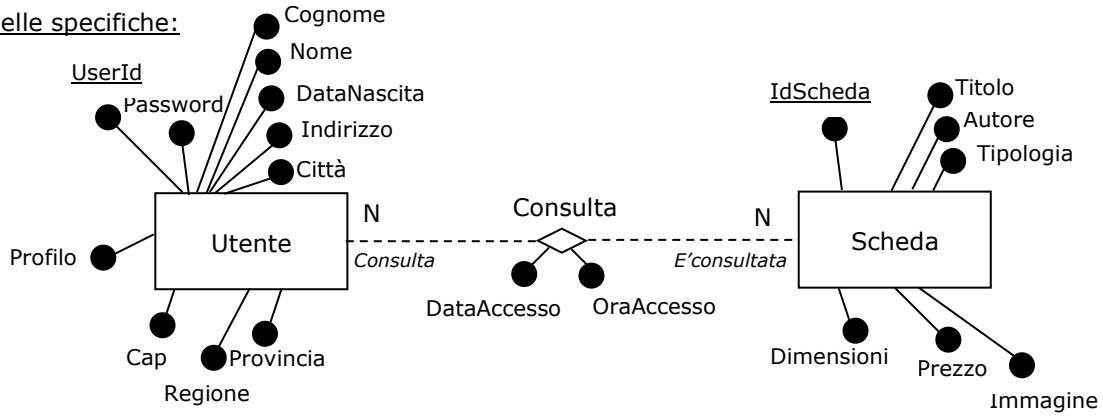
(-) pagina riservata all'amministratore per la gestione completa dei dati presenti in catalogo.

Svolgimento

(a) Schema concettuale della realtà di interesse – diagramma E/R

Premessa:

Analisi delle specifiche:



Vincoli impliciti:

Vincoli espliciti:

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Consulta" di molteplicità N:N tra le entità "Utente" e "Scheda"

Utente (UserId, Password, Profilo, Cognome, Nome, DataNascita, Indirizzo, Cap, Città, Provincia, Regione)

Scheda (IdScheda, Titolo, Autore, Tipologia, Dimensioni, Prezzo, Immagine)

Consulta (UserId1, IdScheda1, DataAccesso, OraAccesso)

con "UserId1" chiave esterna (foreign key) sull'attributo "UserId" della relazione "Utente"

con "IdScheda1" chiave esterna (foreign key) sull'attributo "IdScheda" della relazione "Scheda"

VR_{UserId} (Consulta) $\subseteq$ VR_{UserId} (Utente) dal mapping relazionale dell'associazione N:N

$VR_{IdScheda1}$ (Consulta) $\subseteq$ $VR_{IdScheda}$ (Scheda) dal mapping relazionale dell'associazione N:N

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE GalleriaArte;

USE GalleriaArte;

CREATE DOMAIN MiaTipologia AS CHAR(11)
  CHECK (VALUE IN ("Olio", "Tempera", "Carboncino", "Litografia", "Acquarello"));

CREATE DOMAIN MioProfilo AS CHAR(6)
  CHECK (VALUE IN ("User", "Admin", "Ospite"));

CREATE TABLE Utente
  (
  UserId          CHAR(10)        NOT NULL,
  Password        CHAR(10)        NOT NULL,
  Profilo         MioProfilo      NOT NULL,
  Cognome         CHAR(50)        NOT NULL,
  Nome            CHAR(50)        NOT NULL,
  DataNascita     DATE            NOT NULL,
  Indirizzo       CHAR(255)       NOT NULL,
  Cap             CHAR(5)         NOT NULL,
  Citta           CHAR(30)        NOT NULL,
  Provincia       CHAR(2)         NOT NULL,
  Regione         CHAR(50)        NOT NULL,
  PRIMARY KEY (UserId)
  );

CREATE TABLE Scheda
  (
  IdScheda        CHAR(10)        NOT NULL,
  Titolo          CHAR(30)        NOT NULL,
  Autore          CHAR(50)        NOT NULL,
  Tipologia       MiaTipologia    NOT NULL,
  Dimensioni      CHAR(10)        NOT NULL,
  Prezzo          DECIMAL(8,2)    NOT NULL,
  Immagine        CHAR(255)       NOT NULL,
  PRIMARY KEY (idScheda),
  CHECK (Prezzo > 0)
  );

CREATE TABLE Consulta
  (
  UserId1         CHAR(10)        NOT NULL,
  IdScheda1       CHAR(10)        NOT NULL,
  DataAccesso     DATE            NOT NULL,
  OraAccesso      TIME            NOT NULL,
  PRIMARY KEY (UserId1, IdScheda1),
  FOREIGN KEY (UserId1) REFERENCES UserId (Utente)          // VR di chiave esterna
  [ON DELETE NO ACTION]  ON UPDATE CASCADE,
  FOREIGN KEY (IdScheda1) REFERENCES IdScheda (Scheda)      // VR di chiave esterna
  [ON DELETE NO ACTION] ON UPDATE CASCADE
  );
```

CREATE ASSERTION V_x **CHECK** (Consulta.DataAccesso > Utente.DataNascita);

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Elencare tutte le opere realizzate con una determinata tecnica pittorica

SQL: prima occorre connettersi al database poi

USE GalleriaArte;

SELECT Autore, Titolo, Dimensioni, Prezzo
FROM Scheda
WHERE (Tipologia = [InserisciTipologia])
ORDER BY Autore;

Q2. Elencare tutte le opere realizzate da un determinato artista il cui prezzo è inferiore ai 300 euro

SQL

SELECT Titolo, Dimensioni, Prezzo
FROM Scheda
WHERE (Artista = [InserisciArtista]) **AND** (Prezzo <= 300,00);

(e) Progettazione e sviluppo in dettaglio utilizzando un linguaggio di programmazione lato server di una delle seguenti funzioni:

(-) pagina di accesso all'area riservata ai clienti registrati (profilo "User")

Premessa:

Dato che viene richiesto un catalogo on line, il database deve essere disponibile su un server on line. Per organizzare il sito di presentazione della galleria d'arte si può pensare di utilizzare per generare la parte grafica uno dei tanti tool presenti sul mercato e per il codice, uno dei tanti linguaggi attualmente esistente per la realizzazione di pagine dinamiche.

Molte sono le opzioni disponibili per lo sviluppo della componente server-side. Nella soluzione che presentiamo, abbiamo deciso di utilizzare il linguaggio di programmazione PHP che consente di arricchire le pagine Web di codice script che sarà eseguito direttamente sul server.

In particolare, PHP consente di implementare un motore di scripting server side molto diffuso e multipiattaforma con un buon supporto della connettività verso database diversi (ad es. dBase, Oracle, MySQL) attraverso componenti standard.

Nello specifico ipotizzeremo di interfacciarsi verso un DBMS MySQL, anch'esso multipiattaforma e piuttosto semplice da utilizzare

Ovviamente, per potere utilizzare PHP è necessario aver installato sul proprio sistema un Web Server (come ad esempio Apache).

Riepilogando si è deciso di utilizzare quindi per lo sviluppo del portale e del relativo servizo Web una piattaforma **WAMP** basata su

Sistema Operativo.: **W**indows 2000 o 2003 server

Web Server: **A**PACHE

Database Server: **M**YSQL

Linguaggio di programmazione lato server: **P**HP

mentre si utilizzerà il **linguaggio SQL** per la creazione delle tabelle e per le interrogazioni.

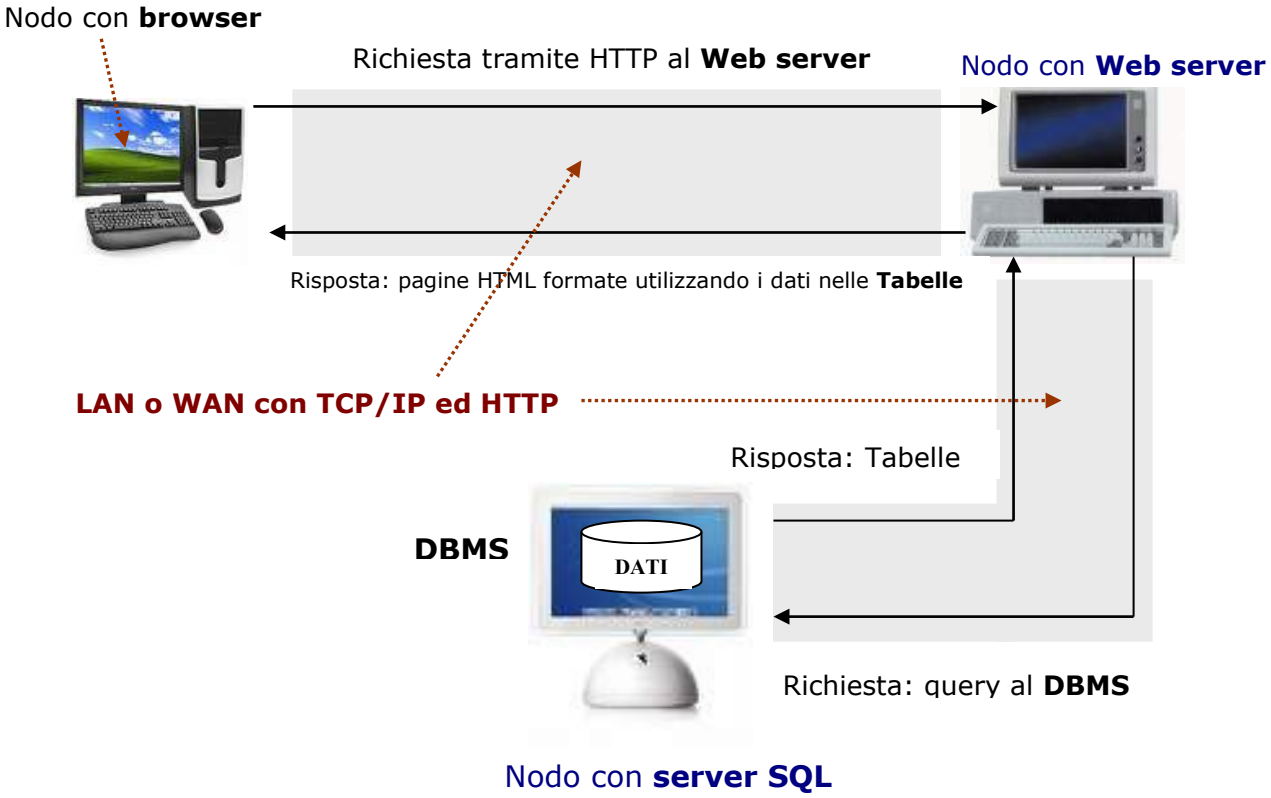
Esistono piattaforme complete open-source assolutamente gratuite che contengono il web server APACHE, il linguaggio di scripting lato server PHP ed il database relazionale MYSQL, (Ad esempio EasyPHP, PHPMyAdmin oppure Apache2triad) sia per ambiente Windows sia per ambiente Linux.

Naturalmente l'uso di una piattaforma **LAMP** (ossia con un operativo server anch'esso open source come **L**inux) sebbene da un lato porti l'indiscutibile vantaggio della totale gratuità rispetto agli alti costi di gestione della Microsoft relativamente alla risoluzione di problematiche connesse all'eventuale risoluzione di problematiche di installazione e/o configurazione, dall'altro può creare a tutti quegli utenti "non addetti ai lavori" che decidano di farne uso e che non abbiano un grosso livello di competenza tecnica informatica

specifica più di qualche problemino di configurazione, di customizzazione e di utilizzo non sempre compatibili con le ragioni di efficienza "aziendale"

Per quanto riguarda l'architettura di riferimento da utilizzare per accedere un database in rete per questa soluzione implementativa facciamo le seguente ipotesi di partenza:

- consideriamo il modello concettuale di networking (ovvero tutto ciò che riguarda la comunicazione tra reti diverse) semplificato a 4 livelli;
- consideriamo i protocolli di comunicazione della famiglia TCP/IP, il browser come client universale ed il Web server come server universale



Per semplicità implementativa, in caso di una quantità di dati non eccessivamente numerosa, possiamo implementare sia il Web server (ad esempio APACHE) sia il Database server (ad esempio MYSQL) sullo stesso nodo.
Sempre per semplicità e chiarezza possiamo fare in modo di utilizzare l'SQL come linguaggio standard di interrogazione e manipolazione di un database ed utilizzeremo anche un server SQL relazionale (ad esempio MYSQL).
Dato l'ambiente client-server così ipotizzato in un'architettura di protocolli TCP/IP ed HTTP possiamo utilizzare per sviluppare tutte le funzionalità previste dall'applicazione complessiva, la programmazione lato server ossia possiamo sviluppare una serie di programmi applicativi che andranno in esecuzione prevalentemente sul server, accettando le richieste dal client e fornendo a quest'ultimo i risultati dell'elaborazione sottoforma di pagine HTML;

Tra i linguaggi di programmazione lato server possiamo utilizzare il PHP ossia un linguaggio di scripting lato server (come PERL ed ASP). Un linguaggio di scripting si differenzia dai linguaggi di programmazione veri e propri lato server perché non ha vita autonoma ma può essere utilizzato esclusivamente in quel contesto.

Il PHP è un linguaggio interpretato lato server: infatti un file di comandi PHP è un classico esempio di programma interpretato lato server poiché infatti il Web server associa a tale file l'interprete PHP che deve essere mandato in esecuzione per poterne eseguire i comandi.
Per questo motivo è possibile avere comandi PHP all'interno di pagine HTML oppure pagine PHP pure.

File: logon.htm

```
<html>
  <head> <title>Form di logon utente</title>
  </head>
  <body>
  <form name="logon" action="logon.php" method="POST" >
      <label>UserId: <input type="text" name="USERID" tabindex="1"><br>
      </label>
      <label>Password: <input type="password" name="PSWD" tabindex="2" ><br>
      </label>
      <select name="PROFILI" tabindex="3">
      <option value="prima voce" selected="selected"> User  </option>
      <option value="seconda voce"> Admin </option>
      <option value="terza voce"> Ospite </option>
       <input type="submit" value="Invia i dati" tabindex="4">
       <input type="reset"  value="Cancella i dati" tabindex="5">
  </form>
</body>
<html>
```

SCRIPT: logon.php SCHEMA DI MASSIMA (senza eccessivi dettagli HTML embedded)

```
<?php
session_start();
// Valorizzazione parametri del database
$db_host = "localhost";   // oppure l'IP address dove risiede il DB Server
$db_user = "root";         // oppure un utente precedentemente creato con privilegi adeguati
$db_pswd = "xxxxx";      // anche eventualmente criptata
$db_name = "GalleriaArte";
// in alternativa è possibile inserire i dati di connessione al DB in un file da includere
//include ("include\parameter.php");
// Connessione al database
$conn = mysqli_connect ($db_host, $db_user, $db_pswd, $db_name)
                OR die("Connessione non riuscita" . " " . mysqli_error() . " " . mysqli_errno());
$userid = $_POST["USERID"];
$pswd = $_POST["PSWD"];
// Esecuzione di una query SQL
$query = "SELECT * FROM Utente WHERE UserId = '". $userid . "' AND Password ='" . $pswd ."'
AND Profilo='User';";
$risultato = mysqli_query ($conn, $query)
                OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));

// Calcolo del numero di occorrenze
$num_righe=mysqli_num_rows($risultato);
if ($num_righe == 1)
        {
        $_SESSION['USER']=$userid;
        $_SESSION['PSWD']=$pswd;
        $_SESSION['PROFILE']='User';
        }
else if ($num_righe == 0)
        {
        echo "Utente non presente nel DB ";
        }
else
        {
        echo "UserID e Password attribuite a più utenti!!! ";
        }
// Chiusura della connessione
mysqli_close($conn)  OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));

}
?>
```

Nel primo script HTML (logon.htm) abbiamo creato il form nel quale l'utente può inserire (e far controllare) le proprie credenziali di accesso (userid e password) al catalogo on line.
Ricordiamo che attraverso il **metodo POST** utilizzato**,** l'invio dei dati avviene in due step distinti: prima viene contattata la pagina sul server che deve processare i dati (logon.php), e poi vengono inviati i dati stessi. Per questo motivo i parametri non compaiono nella "stringa di interrogazione" o query string.
Quindi se si desidera la segretezza della trasmissione di parametri utente "sensibili" questo metodo è senza dubbio preferibile all'altro **metodo GET** con il quale invece, in un unico step, viene contattata la pagina di risposta e vengono inviati i dati. Nell'URL della pagina di risposta è possibile dunque vedere tutti i parametri nella barra degli indirizzi nella query string tenendo presente che per alcuni server essa non può contenere più di 255 caratteri.

Nel secondo script PHP (*logon.php*) innanzitutto abbiamo inizializzato una sessione di lavoro con la funzione **session_start()** in modo di verificare una volta sola le credenziali di accesso dell'utente registrato che verranno poi utilizzate successivamente, in caso di bisogno, nel corso della navigazione senza dover effettuare un ulteriore accesso alla base dati.
Per collegarsi al database abbiamo utilizzato la funzione che riceve come parametro il nome dell'host, il nome dell'utente e la password ossia *$conn* = ***mysqli_connect****($host, $user, $pswd, $db)*
che permette di effettuare il collegamento con un host su cui gira un'istanza di MySQL. e selezionare l'istanza corretta del database sul quale si intende lavorare (essenziale in quanto una stessa istanza del database server può contenere più di un database).
Dopo aver selezionato il database, è possibile effettuare query su qualsiasi tabella tramite le funzione *$risultato* = ***mysqli_query****([$conn], $query).*A questo punto l'oggetto *$risultato* contiene il risultato della query.
Per calcolare il numero delle ennuple presenti nella tabella utente abbiamo usato la funzione $num_righe=**mysqli_num_rows**($risultato)
che restituisce il numero di righe in un risultato. Questo comando è valido solo per le istruzioni SELECT. Per recuperare il numero di righe coinvolte dalle query INSERT, UPDATE o DELETE, usare **mysqli_affected_rows().**
(N.B. Per estrarre eventualmente dei dati da una select di sola LETTURA si usa invece la funzione ***mysqli_fetch_array*** *($risultato, MYSQLI_NUM | MYSQLI_ASSOC | MYSQLI_BOTH);*
Il risultato della chiamata a mysqli_fetch_array ( ) è un array:
- *solo numerico (se il 2° parametro è = MYSQLI_NUM),*
- *solo associativo (se il 2° parametro è = MYSQLI_ASSOC),*
- *oppure entrambi (se il 2° parametro è = MYSQLI_BOTH che è poi il default),*
- *oppure NULL quando è terminato il result set*

Una volta ottenuto l'array relativo all'istanza si potrennao ottenere i valori che restituisce i contenuti di una cella da un risultato MySQL dove l'argomento campo può essere l'indice della riga contenente tutti i dati)
La query da noi effettuata verifica la presenza all'interno della tabella "Utente" di un record avente determinate userid , password asociato ad un determinato profilo (verifica l'esattezza delle credenziali di accesso dell'utente registrato al catalogo on-line).
Per terminare correttamente una sessione di interrogazione ad MYSQL dobbiamo utilizzare la funzione ***mysqli_close***($conn) che chiude la connessione al server MySQL associata all'identificativo di connessione *$conn* specificato permettendo eventualmente ad un altro client di utilizzarla.

NOTA FINALE

Per garantire un discreto margine di sicurezza, sarebbe opportuno far sì che le informazioni riservate, come ad esempio le password, non compaiano nel sorgente html della pagina, e che inoltre, tramite opportuni meccanismi, viaggino criptate attraverso la rete.

Per eseguire query diverse da quella nello script proposto, è sufficiente inserire il codice nella query che deve essere eseguita e, quindi

$query = " select ... ";

Inoltre, per questioni di sicurezza e per garantire la consistenza dei dati, potrebbe essere opportuno definire diverse almeno tre tipologie di utenti del catalogo on line (ad esempio utenti semplici, utenti registrati, amministratori del catalogo, etc.), definendo a livello di database appropriate modalita' di accesso ai dati. Determinare i livelli di accesso e i gruppi di utenti riconosciuti e' compito dell'amministratore della base dati (Data Base Administrator); la regola di massima è quella di concedere i permessi di accesso strettamente necessari. Di norma, tali privilegi vengono assegnati attraverso il comando GRANT, disponibile in MySQL.

Tramite questo comando

GRANT SELECT ON scheda.* TO pippo@localhost IDENTIFIED BY 'password' WITH GRANT OPTION

Si concede all'utente l'autorizzazione all'esecuzione di query di estrazione dati, ma non la possibilita' di modificarli. Il permesso di inserire nuovi record viene, ad esempio, concesso, invece, tramite il comando GRANT INSERT.

(-) pagina riservata ai clienti per la consultazione del catalogo e delle singole schede delle opere

Restano valide le premesse fatte al punto precedente.

Supponiamo che il cliente registrato abbia provveduto a completare con esito positivo la fase di logon con profilo utente "User" necessario per accedere alla consultazione del catalogo on line.

SCRIPT: consulta.php SCHEMA DI MASSIMA (senza eccessivi dettagli HTML embedded)

```
<?php
session_start();
// Prelevo le credenziali utente precedentemente salvati
$userid = $_SESSION['USER'];
$pswd = $_SESSION['PSWD'];
$profilo = $_SESSION['PROFILE'];
// Valorizzazione parametri del database
vedi esempio precedente
// Connessione al database
$conn = mysqli_connect ($db_host, $db_user, $db_pswd, $db_name)
                OR die("Connessione non riuscita" . " " . mysqli_error() . " " . mysqli_errno());
// Esecuzione di una query SQL
$query = "SELECT * FROM Scheda;";
$risultato = mysqli_query ($conn, $query)
        OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
// Calcolo del numero di occorrenze
$num_righe=mysqli_num_rows($risultato);
// controllo il numero di quadri presenti in catalogo
if ($num_righe != 0 )
        {
        // Intestazione della tabella che conterrà lista delle schede dei quadri
        echo "
        <table align='center'>
         <tr>
           <td align='center'><b>Titolo</b></td>
           <td align='center'><b>Autore</b></td>
           <td align='center'><b>Prezzo</b></td>
           <td align='center'><b>Visualizza dettagli</b></td>
         </tr>
        ";
        for ($i=1;$i<=$num_righe;$i++)
                {
                // Prelevamento dati di ciascun record in tabella
                $riga = mysqli_fetch_array ($risultato, MYSQLI_ASSOC);
                $idscheda=$riga['IdScheda'];
                $titolo=$riga['Titolo'];
                $autore=$riga['Autore'];
                $prezzo=$riga['Prezzo'];
                echo "
                        <tr bgcolor='ffffff'>
                        <td>$titolo</td>
                        <td>$autore</td>
                        <td>$prezzo</td>
                        <td>
                                <form action='consulta_dettagli.php' method='POST'>
                                <input type='hidden' name='id' value='$idscheda'>
                                <input type='image' border='0' src='images/dettagli.jpg'
                                name='immagine' title='Clicca per la scheda del quadro>
                                </form>
                        </td>
```

```
                        </tr>
                ";
                }
        echo "</table>";
        }
else
        {
        echo "La tabella non contiene alcun record!”;
        }
// Chiusura della connessione
mysqli_close($conn)  OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
}
?>
```

SCRIPT: consulta_dettagli.php SCHEMA DI MASSIMA (senza dettagli HTML embedded)

```
<?php
session_start();
// Prelevo le credenziali utente precedentemente salvati
$userid=$_SESSION['USER'];
$pswd=$_SESSION['PSWD'];
$profilo = $_SESSION['PROFILE'];
// Valorizzazione parametri del database
vedi esempio precedente
// Connessione al database
$conn = mysqli_connect ($db_host, $db_user, $db_pswd, $db_name)
                OR die("Connessione non riuscita" . " " . mysqli_error() . " " . mysqli_errno());
// Recupero informazioni record da visualizzare attraverso l’secuzione di una query SQL
$query = "SELECT * FROM Scheda WHERE IdScheda = $_POST[`$idscheda’]";
//echo "<br>Stringa di query  <br>" . $query;
$risultato = mysqli_query ($conn, $query)
        OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
// Calcolo del numero di occorrenze
$numrighe=mysqli_num_rows($risultato)
                OR die("<b>Impossibile ottenere le righe dalla query</b>");
if ($numrighe == 1)
        {
        //Recupero e Stampa Dettagli record
        $riga = mysqli_fetch_array ($risultato, MYSQLI_ASSOC);
        $titolo=$riga[’Titolo’];
        $autore=$riga[’’Autore’];
        $tipologia=$riga[’’Tipologia’];
        $dimensione=$riga[’’Dimensione’];
        $prezzo=$riga[’’Prezzo’];
        $fileimmagine=$riga[’’Immagine’];
        echo "
        <table border='1' bgcolor='ffffff' cellpadding='0' cellspacing='0' width='650'>
        <tr>
                <td>Identificativo: </td> <td> $idscheda</td>
        </tr>
        <tr>
                <td> Titolo: </td> <td> $titolo</td>
        </tr>
        <tr>
                <td> Autore: </td> <td> $autore</td>
        </tr>
        <tr>
                <td> Tipologia: </td> <td> $tipologia</td>
        </tr>
        <tr>
                <td> Dimensione: </td> <td> $dimensione</td>
        </tr>
        <tr>
                <td> Prezzo: </td> <td> $prezzo</td>
```

```
        </tr>
        <tr>
                <td> Immagine: </td> <td> <IMG SRC=$fileimmagine > </td>
        </tr>
        </table>
        ";
        }
else
        {
        echo "Errore";
        }
// Chiusura della connessione
mysqli_close($conn)  OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
}
?>
```

(-) pagina riservata all'amministratore per la gestione completa dei dati presenti in catalogo

Restano valide le premesse fatte al punto precedente.

Supponiamo che l'utente individuato come amministratore del catalogo on-line abbia provveduto a completare con esito positivo la fase di logon con profilo utente "Admin" necessario per accedere alla consultazione del catalogo on line.

SCRIPT: consulta_admin.php SCHEMA DI MASSIMA (senza eccessivi dettagli HTML embedded)

```
<?php
session_start();
// Prelevo le credenziali utente precedentemente salvati
$userid=$_SESSION['USER'];
$pswd=$_SESSION['PSWD'];
$profilo = $_SESSION['PROFILE'];
// Valorizzazione parametri del database
vedi esempio precedente
// Connessione al database
$conn = mysqli_connect ($db_host, $db_user, $db_pswd, $db_name)
                OR die("Connessione non riuscita" . " " . mysqli_error() . " " . mysqli_errno());
// Esecuzione di una query SQL
$query = "SELECT * FROM Scheda;";
$risultato = mysqli_query ($conn, $query)
        OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
// Calcolo del numero di occorrenze
$num_righe=mysqli_num_rows($risultato);
// controllo il numero di quadri presenti in catalogo
if ($num_righe != 0 )
        {
        // Intestazione della tabella che conterrà lista delle schede dei quadri
        echo "
        <table align='center'>
         <tr>
           <td align='center'><b>Titolo</b></td>
           <td align='center'><b>Autore</b></td>
           <td align='center'><b>Prezzo</b></td>
           <td align='center'><b>Visualizza dettagli</b></td>
         </tr>
        ";
        for ($i=1;$i<=$num_righe;$i++)
                {
                // Prelevamento dati di ciascun record in tabella
                $riga = mysqli_fetch_array ($risultato, MYSQLI_ASSOC);
                $idscheda=$riga['IdScheda'];
                $titolo=$riga['Titolo'];
                $autore=$riga['Autore'];
                $prezzo=$riga['Prezzo'];
                echo "
```

```
                        <tr bgcolor='ffffff'>
                        <td>$titolo</td>
                        <td>$autore</td>
                        <td>$prezzo</td>
                        <td>
                                <form action=consulta_dettagli.php' method='POST'>
                                <input type='hidden' name='id' value='$idscheda'>
                                <input type='image' border='0' src='images/dettagli.jpg'
                                name='immagine' title='Clicca per la scheda del quadro>
                                </form>
                        </td>
                        <td>
                                <form action='update_scheda.php' method='POST'>
                                <input type='hidden' name='id' value='$idscheda'>
                                <input type='image' border='0' src='images/update.jpg'
                                name='immagine' title='Clicca per aggiornarla scheda del quadro >
                                </form>
                        </td>
                        <td>
                                <form action='delete_scheda.php' method='POST'>
                                <input type='hidden' name='id' value='$idscheda'>
                                <input type='image' border='0' src='images/delete.jpg' name='immagine'
                                title='Clicca per cancellare la scheda del quadro>
                                </form>
                        </td>

                        </tr>
                ";
                }
        echo "</table>";
        }
else
        {
        echo "La tabella non contiene alcun record!”;
        }
// Chiusura della connessione
mysqli_close($conn)  OR die("Query fallita" . " " . mysqli_error($conn) . " " . mysqli_errno($conn));
?>
```

(7) Il nuovo **direttore finanziario** di una **società** intende automatizzare **la gestione dei pagamenti dei crediti erogati ai propri clienti**. Ogni credito è identificato con codice, denominazione del cliente, indirizzo, provincia e regione di appartenenza, modalità di pagamento (pronta cassa, 30 giorni, 60 giorni o 90 giorni), ammontare del credito stesso, data di concessione del credito, data del pagamento prestabilita, esito del pagamento (non effettuato perché ancora non si è raggiunti la data di pagamento prevista, andato a buon fine perché pagato entro i limiti, non andato a buon fine se il cliente non ha pagato entro la data prestabilita).

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Elenco dei clienti che devono ancora effettuare i pagamenti per ogni provincia e per ogni regione;

Q2: Elenco dei clienti che devono effettuare il pagamento in un certo giorno;

Q3: Per ogni tipo di pagamento, elenco dei clienti che hanno scelto quel tipo di pagamento;

Q4: Elenco dei clienti che hanno ricevuto un credito superiore a 100.000 euro e non hanno effettuato il pagamento entro la data stabilita (cattivi pagatori).

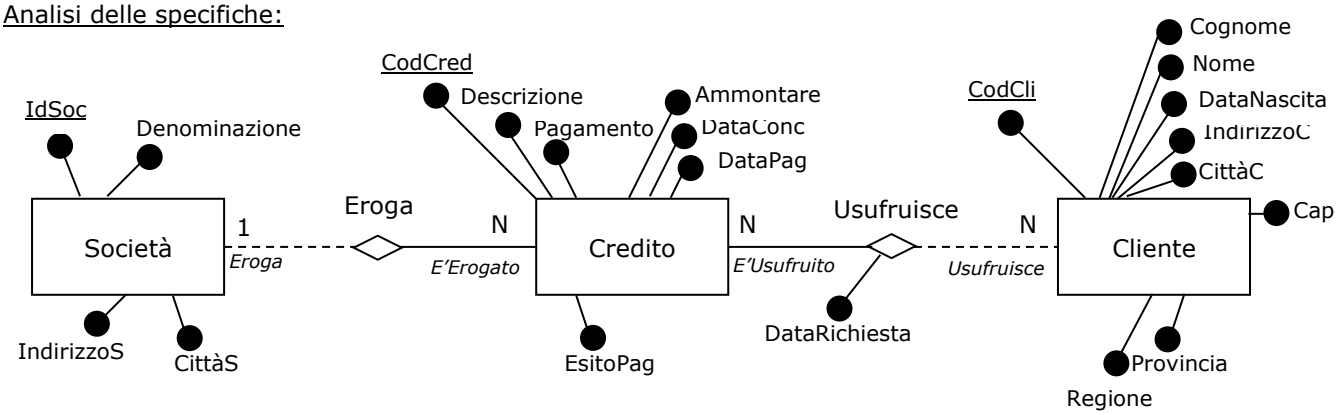
e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

(a) Schema concettuale della realtà di interesse – diagramma E/R

Premessa:

Analisi delle specifiche:



Vincoli impliciti:

Vincoli espliciti:

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Eroga" di molteplicità 1:N tra le entità "Società" e "Credito"

Società (IdSoc, Denominazione, IndirizzoS, CittàS)

Credito (CodCred, Descrizione, Pagamento, Ammontare, DataConc, DataPag, EsitoPag, IdSoc1)

con "IdSoc1" chiave esterna (foreign key) sull'attributo "IdSoc" della relazione "Società"

VR_{IdSoc1} (Credito) $\subseteq$ VR_{IdSoc} (Società) dalla TOTALITA' dell'associazione inversa "E'Erogato"

(ii) mapping dell'associazione "Usufruisce" di molteplicità N:N tra le entità "Cliente" e "Credito"

Cliente (CodCli, Cognome, Nome, DataNascita, Indirizzo, Cap, CittàC, Provincia, Regione)
Credito già mappato in precedenza
Usufruisce (CodCli1, CodCred1, DataRichiesta)
con "CodCli1" chiave esterna (foreign key) sull'attributo "CodCli" della relazione "Cliente"
con "CodCred1" chiave esterna (foreign key) sull'attributo "CodCred" della relazione "Credito"

$VR_{CodCli1}$ (Usufruisce) $\subseteq$ VR_{CodCli} (Cliente) dal mapping relazionale dell'associazione N:N
$VR_{CodCred1}$ (Usufruisce) $\subseteq$ $VR_{CodCred}$ (Credito) dal mapping relazionale dell'associazione N:N

VR_{CodCli} (Cliente) $\subseteq$ $VR_{CodCli1}$ (Usufruisce) dalla TOTALITA' dell'associazione diretta "Usufruisce"
$VR_{CodCred}$ (Credito) $\subseteq$ $VR_{CodCred1}$ (Usufruisce) dalla TOTALITA' dell'associazione inversa "E'Usufruito"

Mapping dei vincoli nel modello relazionale:

Normalizzazione:

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE Gestione Crediti;

USE GestioneCrediti;

CREATE DOMAIN MioModoPag AS CHAR(12)
  CHECK (VALUE IN ("Pronta Cassa", "A 30 giorni", "A 60 giorni", "A 90 giorni"));

CREATE DOMAIN MioEsitoPag AS CHAR(15)
  CHECK (VALUE IN ("Non effettuato", "A buon fine", "Non a buon fine"));

CREATE TABLE Societa
  (
  IdSoc            CHAR(10)          NOT NULL,
  Denominazione    CHAR(50)          NOT NULL,
  IndirizzoS       CHAR(255)         NOT NULL,
  CittaS           CHAR(50),
  PRIMARY KEY (IdSoc)
  );

CREATE TABLE Credito
  (
  CodCred          CHAR(10)          NOT NULL,
  Descrizione      CHAR(50)          NOT NULL,
  Pagamento        MioModpag         NOT NULL,
  Ammontare        DECIMAL(8,2)      NOT NULL,
  DataConc         DATE              NOT NULL,
  DataPag          DATE              NOT NULL,
  EsitoPag         MioEsitoPag       NOT NULL,
  IdSoc1           CHAR(10),
  PRIMARY KEY (CodCred),
  FOREIGN KEY (IdSoc1) REFERENCES IdSoc (Societa)     // VR di chiave esterna
  ON DELETE SET NULL  ON UPDATE CASCADE,
  CHECK (DataConc < DataPag),
  CHECK (Ammontare >= 0)
  );
```

```
CREATE TABLE Cliente
  (
  CodCli          CHAR(10)        NOT NULL,
  Cognome         CHAR(50)        NOT NULL,
  Nome            CHAR(50)        NOT NULL,
  DataNascita     DATE            NOT NULL,
  IndirizzoC      CHAR(255)       NOT NULL,
  Cap             CHAR(5)         NOT NULL,
  CittaC          CHAR(30)        NOT NULL,
  Provincia       CHAR(2)         NOT NULL,
  Regione         CHAR(50)        NOT NULL,
  PRIMARY KEY (CodCli)
  );

CREATE TABLE Usufruisce
  (
  CodCli1         CHAR(10)        NOT NULL,
  CodCred1        CHAR(10)        NOT NULL,
  DataRichiesta   DATE            NOT NULL,
  PRIMARY KEY (CodCli1, CodCred1),
  FOREIGN KEY (CodCli1) REFERENCES CodCli (Cliente)          // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (CodCred1) REFERENCES CodCred (Credito)        // VR di chiave esterna
  ON DELETE CASCADE ON UPDATE CASCADE
  );

CREATE ASSERTION Vx CHECK (Usufruisce.DataRichiesta <= Credito.DataConc);
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Elenco dei clienti che devono ancora effettuare i pagamenti per ogni provincia e per ogni regione

SQL: prima occorre connettersi al database poi

```
USE GestioneCrediti;

SELECT Cognome, Nome
FROM Cliente, Usufruisce, Credito
WHERE (Cliente.CodCli = Usufruisce.CodCli1) AND (Usufruisce.CodCred1 = Credito.CodCred)
AND (Pagamento = "Non effettuato");
```

oppure

```
SELECT Cognome, Nome
FROM Cliente, Usufruisce, (SELECT * FROM Pagamento WHERE (Pagamento = "Non effettuato") AS t1)
WHERE (Cliente.CodCli = Usufruisce.CodCli1) AND (Usufruisce.CodCred1 = t1.CodCred)
```

Algebra Relazionale

A = { Cognome, Nome }

P = { (Pagamento = "Non effettuato") }

$$Q1 = \Pi_A (\sigma_P (\text{Cliente} \underset{\text{CodCli = CodCli1}}{\triangleright\!\triangleleft} (\text{Usufruisce} \underset{\text{CodCred1 = CodCred}}{\triangleright\!\triangleleft} \text{Credito})))$$

oppure

$$Q1 = \Pi_{A} (\text{Cliente} \underset{\text{CodCli = CodCli1}}{\triangleright\triangleleft} (\text{Usufruisce} \underset{\text{CodCred1 = CodCred}}{\triangleright\triangleleft} \sigma_{P} (\text{Credito})))$$

Q2. Elenco dei clienti che devono effettuare il pagamento in un certo giorno

SQL: prima occorre connettersi al database poi

USE GestioneCrediti;

SELECT Cognome, Nome
FROM Cliente, Usufruisce, Credito
WHERE (Cliente.CodCli = Usufruisce.CodCli1) **AND** (Usufruisce.CodCred1 = Credito.CodCred)
AND (DataPag = [InserisciData]);

oppure

SELECT Cognome, Nome
FROM Cliente, Usufruisce, (**SELECT** * **FROM** Pagamento **WHERE** (DataPag = [InserisciData]) AS t1)
WHERE (Cliente.CodCli = Usufruisce.CodCli1) **AND** (Usufruisce.CodCred1 = t1.CodCred)

Algebra Relazionale

A = { Cognome, Nome }

P = { (DataPag = [InserisciData]) }

$$Q2 = \Pi_{A} (\sigma_{P} (\text{Cliente} \underset{\text{CodCli = CodCli1}}{\triangleright\triangleleft} (\text{Usufruisce} \underset{\text{CodCred1 = CodCred}}{\triangleright\triangleleft} \text{Credito})))$$

oppure

$$Q2 = \Pi_{A} (\text{Cliente} \underset{\text{CodCli = CodCli1}}{\triangleright\triangleleft} (\text{Usufruisce} \underset{\text{CodCred1 = CodCred}}{\triangleright\triangleleft} \sigma_{P} (\text{Credito})))$$

Q3. Per ogni tipo di pagamento, elenco dei clienti che hanno scelto quel tipo di pagamento

SQL: prima occorre connettersi al database poi

USE GestioneCrediti;

SELECT Cognome, Nome
FROM Cliente, Usufruisce, Credito
WHERE (Cliente.CodCli = Usufruisce.CodCli1) **AND** (Usufruisce.CodCred1 = Credito.CodCred)
ORDER BY Pagamento, Cognome, Nome;

Q4. Elenco dei clienti che hanno ricevuto un credito superiore a 100.000 euro e non hanno effettuato il pagamento entro la data stabilita (cattivi pagatori)

SQL: prima occorre connettersi al database poi

USE GestioneCrediti;

SELECT Cognome, Nome
FROM Cliente, Usufruisce, Credito
WHERE (Cliente.CodCli = Usufruisce.CodCli1) **AND** (Usufruisce.CodCred1 = Credito.CodCred)
AND (Pagamento = "Non a buon fine") **AND** (Ammontare < 10000,00);

oppure

SELECT Cognome, Nome

FROM Cliente, Usufruisce, (**SELECT** * **FROM** Pagamento **WHERE** (Pagamento = "Non a buon fine") **AND** (Ammontare < 10000,00) AS t1)

WHERE (Cliente.CodCli = Usufruisce.CodCli1) **AND** (Usufruisce.CodCred1 = t1.CodCred);

(8) Un'**azienda** vuole gestire **gli ordini dei propri clienti relativi ai prodotti** trattati in modo automatizzato. Ogni ordine è individuato da codice, data, valore complessivo in euro, denominazione cliente, indirizzo, comune e provincia ed è costituito dall'intenzione di acquisto in quantità variabili di determinati prodotti.

Ogni cliente è individuato dalla sua anagrafica.

Ogni prodotto è individuato da codice, descrizione, prezzo di listino

Si realizzino, fatte le ipotesi aggiuntive del caso,

a) Uno schema concettuale della realtà di interesse attraverso la produzione del diagramma E/R (scrivendo esplicitamente le conseguenti regole di lettura);

b) lo schema logico della realtà di interesse ottenuto attraverso il mapping relazionale dello schema concettuale (diagramma E/R) ottenuto al punto precedente;

c) la definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL.

Ed inoltre

d) si implementino, dapprima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL, le seguenti interrogazioni:

Q1: Visualizzare il numero di ordini effettuati da ogni cliente;

Q2: Visualizzare il numero di ordini effettuati complessivamente da tutti i clienti;

Q3: Visualizzare il valore medio degli ordini di ciascun cliente;

Q4: Elenco dei clienti che hanno un valore complessivo degli ordini al di sotto della media.

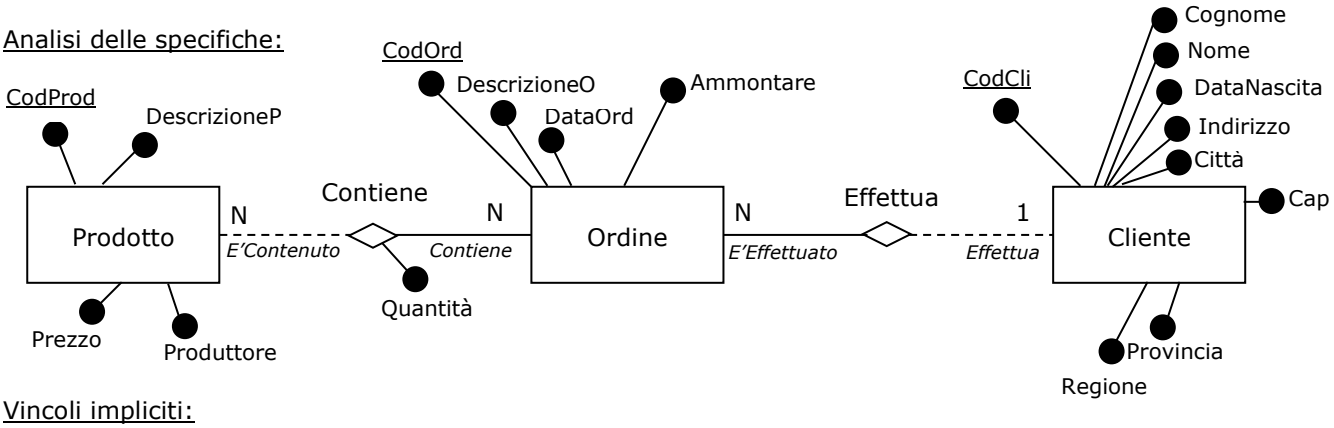
e) Descrivere l'interfaccia utente che si intende proporre per interagire con la base di dati e codificare, in un linguaggio di programmazione a scelta, un segmento significativo del progetto realizzato (esempio una query).

Svolgimento

(a) Schema concettuale della realtà di interesse – diagramma E/R

Premessa:

Analisi delle specifiche:



Vincoli impliciti:

Vincoli espliciti:

(b) Schema logico della realtà di interesse - mapping relazionale del diagramma E/R

(i) mapping dell'associazione "Contiene" di molteplicità N:N tra le entità "Ordine" e "Prodotto"

Ordine (CodOrd, DescrizioneO, DataOrd, Ammontare, CodCli1)

con "CodCli1" chiave esterna (foreign key) sull'attributo "CodCli" della relazione "Cliente"

Prodotto (CodProd, DescrizioneP, Prezzo, Produttore)

Contiene (CodOrd1, CodProd1, Quantità)
con "CodOrd1" chiave esterna (foreign key) sull'attributo "CodOrd" della relazione "Ordine"
con "CodProd1" chiave esterna (foreign key) sull'attributo "CodProd" della relazione "Prodotto"

$VR_{CodOrd1}$ (Contiene) $\subseteq$ VR_{CodOrd} (Ordine) dal mapping relazionale dell'associazione N:N
$VR_{CodProd1}$ (Contiene) $\subseteq$ $VR_{CodProd}$ (Prodotto) dal mapping relazionale dell'associazione N:N

VR_{CodOrd} (Ordine) $\subseteq$ $VR_{CodOrd1}$ (Contiene) dalla TOTALITA' dell'associazione diretta "Contiene"

(ii) mapping dell'associazione "Effettua" di molteplicità 1:N tra le entità "Cliente" e "Ordine"

Cliente (CodCli, Cognome, Nome, DataNascita, Indirizzo, Cap, Città, Provincia, Regione)
Ordine già mappato in precedenza

$VR_{CodCli1}$ (Ordine) $\subseteq$ VR_{CodCli} (Cliente) dalla TOTALITA' dell'associazione inversa "E'effettuato"

Mapping dei vincoli nel modello relazionale:

Normalizzazione:

(c) Definizione delle relazioni della base dati ottenute al punto precedente in linguaggio SQL

```
CREATE DATABASE Ordini;

USE Ordini;

CREATE TABLE Ordine
  (
  CodOrd          CHAR(10)        NOT NULL,
  DescrizioneO    CHAR(50)        NOT NULL,
  DataOrd         DATE            NOT NULL,
  Ammontare       DECIMAL(8,2)    NOT NULL,
  CodCli1         CHAR(10),
  PRIMARY KEY (CodOrd),
  FOREIGN KEY (CodCli1) REFERENCES CodCli (Cliente)     // VR di chiave esterna
  ON DELETE SET NULL  ON UPDATE CASCADE,
  CHECK (Ammontare > 0)
  );

CREATE TABLE Prodotto
  (
  CodProd         CHAR(10)        NOT NULL,
  DescrizioneP    CHAR(50)        NOT NULL,
  Prezzo          DECIMAL(8,2)    NOT NULL,
  Produttore      CHAR(50),
  PRIMARY KEY (CodProd),
  CHECK (Prezzo > 0)
  );

CREATE TABLE Contiene
  (
  CodOrd11        CHAR(10)        NOT NULL,
  CodProd1        CHAR(10)        NOT NULL,
  Quantita        SMALLINT        DEFAULT 1,
  PRIMARY KEY (CodOrd1, CodProd1),
```

```
  FOREIGN KEY (CodOrd1) REFERENCES CodOrd (Ordine)          // VR di chiave esterna
  ON DELETE CASCADE  ON UPDATE CASCADE,
  FOREIGN KEY (CodProd1) REFERENCES CodProd (Prodotto)      // VR di chiave esterna
  [ON DELETE NO ACTION] ON UPDATE CASCADE,
  CHECK (Quantita >= 1)
  );

CREATE TABLE Cliente
  (
  CodCli          CHAR(10)        NOT NULL,
  Cognome         CHAR(50)        NOT NULL,
  Nome            CHAR(50)        NOT NULL,
  DataNascita     DATE            NOT NULL,
  Indirizzo       CHAR(255)       NOT NULL,
  Cap             CHAR(5)         NOT NULL,
  Citta           CHAR(30)        NOT NULL,
  Provincia       CHAR(2)         NOT NULL,
  Regione         CHAR(50)        NOT NULL,
  PRIMARY KEY (CodCli)
  );
```

(d) Svolgimento delle query prima utilizzando gli operatori dell'algebra relazionale (se possibile) poi usando il linguaggio SQL

Q1. Visualizzare il numero di ordini effettuati da ogni cliente

SQL: prima occorre connettersi al database poi

```
USE Ordini;

SELECT CodCli, Cognome, Nome, COUNT(*) AS NumOrdini
FROM Cliente, Ordine
WHERE (Cliente.CodCli = Ordine.CodCli1)
GROUP BY CodCli, Cognome, Nome;
```

Q2. Visualizzare il numero di ordini effettuati complessivamente da tutti i clienti

SQL: prima occorre connettersi al database poi

```
USE Ordini;

SELECT COUNT(*) AS NumTotOrdini
FROM Ordine;
```

Q3. Visualizzare il valore medio degli ordini di ciascun cliente

SQL: prima occorre connettersi al database poi

```
USE Ordini;

SELECT CodCli, Cognome, Nome, AVG(Ammontare) AS MediaOrdini
FROM Cliente, Ordine
WHERE (Cliente.CodCli = Ordine.CodCli1)
GROUP BY CodCli, Cognome, Nome;
```

Q4. Elenco dei clienti che hanno un valore complessivo degli ordini al di sotto della media

SQL: prima occorre connettersi al database poi

```
USE Ordini;

SELECT CodCli, Cognome, Nome, SUM(Ammontare) AS ComplOrdini
FROM Cliente, Ordine
WHERE (Cliente.CodCli = Ordine.CodCli1)
AND (ComplOrdini <  (SELECT AVG(Ammontare)
                     FROM Ordine )
GROUP BY CodCli, Cognome, Nome;
```