Elaborato per il corso di Basi di Dati

A.A 2021-2022

Progetto di una base di dati per la gestione di una scuola di skateboard

A cura di:

Sara Cappelletti – 0000976977 – sara.cappelletti4@studio.unibo.it  
Rachele Margutti – 0000997560 – rachele.margutti@studio.unibo.it

## Analisi dei requisiti

Intervista 3

Estrazione dei concetti principali 4

## **Progettazione concettuale**

Schema Scheletro 6

Schema finale 9

## **Progettazione logica**

Stima del volume dei dati 10

Descrizione delle operazioni principali e stima della loro frequenza 11

Tabelle degli accessi 11

Raffinamento dello schema 15

Analisi delle ridondanze 16

Traduzione di entità e associazioni in relazioni 17

Schema relazionale finale 19

Traduzione della creazione tabelle in query SQL 20

Traduzione delle operazioni in query SQL 23

## **Progettazione dell’applicazione**

Descrizione dell’architettura dell’applicazione realizzata 27

**ANALISI DEI REQUISITI**

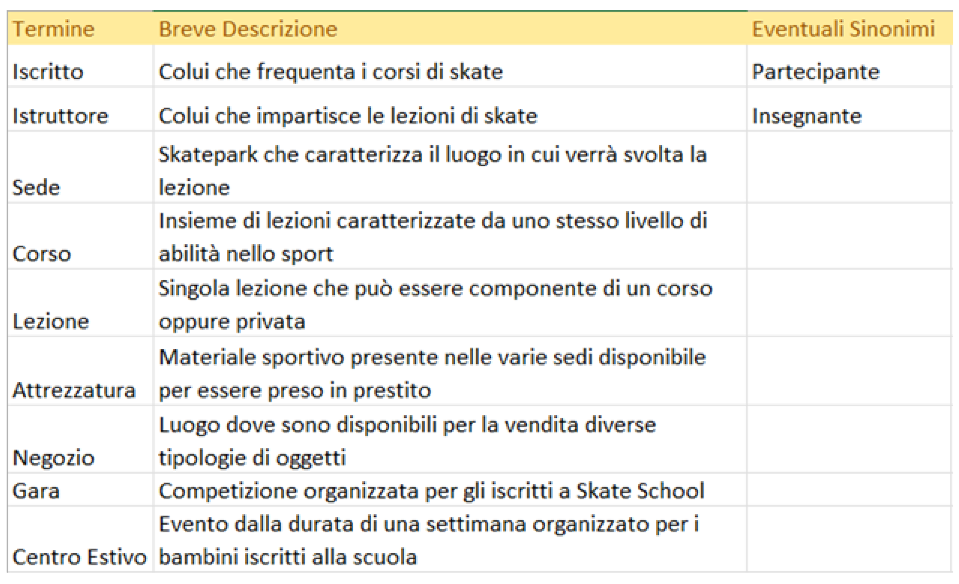
Si vuole realizzare una base di dati volta a supportare e ad automatizzare la gestione di una scuola di skateboard. Per questo, il database dovrà occuparsi della raccolta di informazioni relative agli istruttori, ai clienti, alle differenti tipologie di corsi e a varie attività organizzate da Skate School quali competizioni e centri estivi. Dovrà inoltre occuparsi della gestione del negozio e dei vari tipi di tariffari disponibili e del prestito di attrezzatura sportiva.

## **Intervista**

Ecco un testo ottenuto dall’intervista:

Skate school è una scuola di skate, con vari sedi, istruttori, corsi, gare e partecipanti. Offre anche la possibilità di acquistare materiale sportivo.  
Le sedi sono identificate da un codice (attualmente sono Cesena, Forlì, Cesenatico, San Mauro Pascoli e Ravenna).  
Gli istruttori sono identificati da un codice ma hanno anche nome, cognome, età e telefono.  
Ogni istruttore può insegnare in più sedi e avere più corsi, ma insegnare al massimo a 6 partecipanti.  
I corsi sono per ora baby, base, avanzato, team, cooperativa, per bambini e adulti, ogni corso ha le lezioni con diversi orari che cambiano in base alla sede, un codice identificativo e un massimo di partecipanti legato alla sede in cui si svolge il corso. In aggiunta sono presenti anche le lezioni private a un singolo partecipante, che non hanno un orario fisso.  
Ogni lezione può essere pagata singolarmente (10 euro per la lezione normale e 20 per quella privata), a blocchi di 5 o di 10, con relativo sconto.  
Il negozio è presente in ogni sede, con magazzini indipendenti, ed è fornito di vari oggetti (per esempio: skate, track, ruote, cuscinetti, viti, grip, maglie, felpe, scarpe…) identificati dal codice identificativo e aventi il tipo di oggetto e il prezzo.  
Sono inoltre presenti in ogni sede una tavole e caschi che i partecipanti possono prendere in prestito, identificati da un codice.  
Le gare sono organizzate ogni tanto, senza cadenza periodica, e hanno categorie in base all’età.  
Una volta o più volte l’anno viene organizzato un centro estivo, che è caratterizzato da un codice e ha anche la data di inizio, di fine, ed è legato ai partecipanti e agli istruttori. Per ogni partecipante viene salvato un identificativo e nome, cognome, età, telefono, livello.

## **Estrazione dei concetti principali**



Dopo un’attenta analisi dei requisiti, si procede alla realizzazione di un testo che riassuma i concetti principali e ne fornisca le caratteristiche precise:

Per ogni **iscritto** di Skate School vengono memorizzati nome, cognome, età, numero di telefono e livello di abilità nello sport. Ognuno possiede anche un codice univoco identificativo generato al momento dell’iscrizione. Un iscritto può partecipare ad una lezione esclusivamente previa registrazione alla scuola e pagamento.   
Ogni **corso** è composto da differenti lezioni in base al livello di abilità ed è caratterizzato da una descrizione, il livello e il codice della sede di riferimento.  
Una **lezione** invece è caratterizzata da un codice univoco, il codice dell’istruttore che la terrà, il codice del corso di cui fa parte, il numero massimo di partecipanti possibili e la data in cui verrà impartita.   
Gli **istruttori** si occupano di tenere le lezioni, che siano esse di gruppo oppure private.   
I prezzi delle lezioni sono memorizzati in un **tariffario** che offre vari pacchetti in base al numero di lezioni desiderate.  
Una **sede** è il particolare skatepark nel quale si desidera tenere una lezione, ed è caratterizzata dal codice identificativo, dalla città e una descrizione col nome del Park. In ognuna di esse è presente **l’attrezzatura** disponibile per il prestito, che sarà identificata da un codice specifico, il codice della sede in cui si trova e da una breve descrizione.  
Il **negozio** offre la possibilità di acquistare diversi **oggetti** caratterizzati da un codice identificativo, quantità desiderata, prezzo, una breve descrizione, il codice della sede in cui si trova e il codice del tipo di oggetto in questione (ad esempio maglia, skate, ecc, informazioni contenute in **tipologia oggetto**).   
Infine, vengono organizzati eventi quali **gare**, ossia competizioni dedicate agli iscritti di qualsiasi età divisi in categorie, caratterizzate da un codice, una città e una data e i **centri estivi** caratterizzati da un codice, data di inizio e di fine e descrizione.

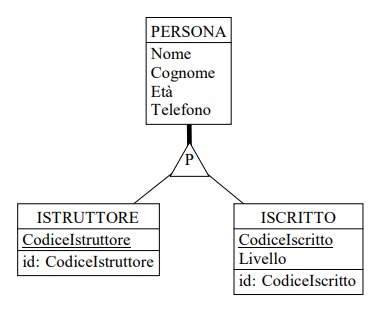
Di seguito, un elenco delle principali azioni richieste:

1. Inserire un istruttore
2. Inserire un nuovo iscritto
3. Inserire una lezione
4. Verificare il pagamento di una lezione
5. Iscrivere un partecipante ad una lezione
6. Visualizzare tutti gli iscritti ad una lezione
7. Visualizzare il tariffario più venduto
8. Controllare se l’attrezzatura prestata è stata restituita
9. Acquistare n T-Shirt
10. Visualizzare le sedi di un corso in base al livello di un iscritto
11. Visualizzare l’elenco degli istruttori disponibili in una certa data
12. Prenotare una lezione privata
13. Visualizzare gli iscritti ad un centro estivo
14. Visualizzare il guadagno totale per anno
15. Aggiornare il prezzo di un oggetto

**PROGETTAZIONE CONCETTUALE**

**Schema Scheletro**

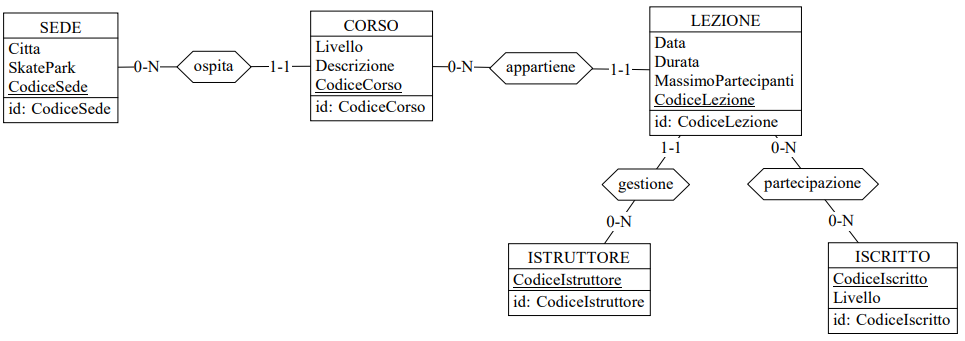
Per la rappresentazione delle entità **istruttore** e **iscritto** si è scelto di usare la gerarchia **persona** in modo da generalizzare gli attributi in comune che tutte le persone hanno.  
Le due vengono poi distinte tramite un codice identificativo (rispettivamente, *codiceIstruttore* e *codiceIscritto*). L’iscritto ha inoltre il *Livello*.



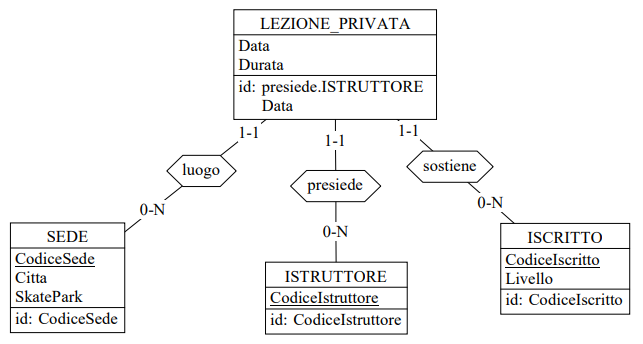
Una **sede** può ospitare più corsi in una giornata e varie lezioni.

Un **corso** si divide in **lezioni** ed ha uno specifico livello, più lezioni dello stesso corso possono verificarsi in contemporanea se il numero di **iscritti** supera quello dei partecipanti massimi ammessi, però in tal caso sarà necessaria la partecipazione di un ulteriore **istruttore** (verrà considerata come una **lezione** differente).

Ogni **lezione** è tenuta esclusivamente da un **istruttore**, il quale può tenere più lezioni di corsi differenti.

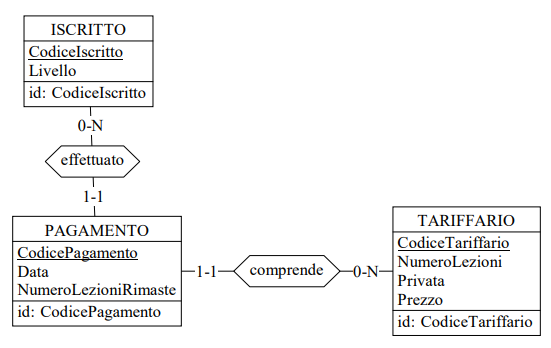


Una **lezione privata** può essere effettuata solamente per un unico **iscritto** alla volta e non ha vincolo di livello, viene fissata in base alla disponibilità dell’**istruttore** e può essere effettuata in qualsiasi **sede**.

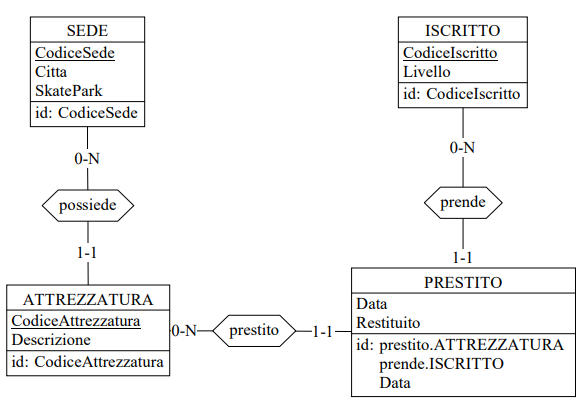


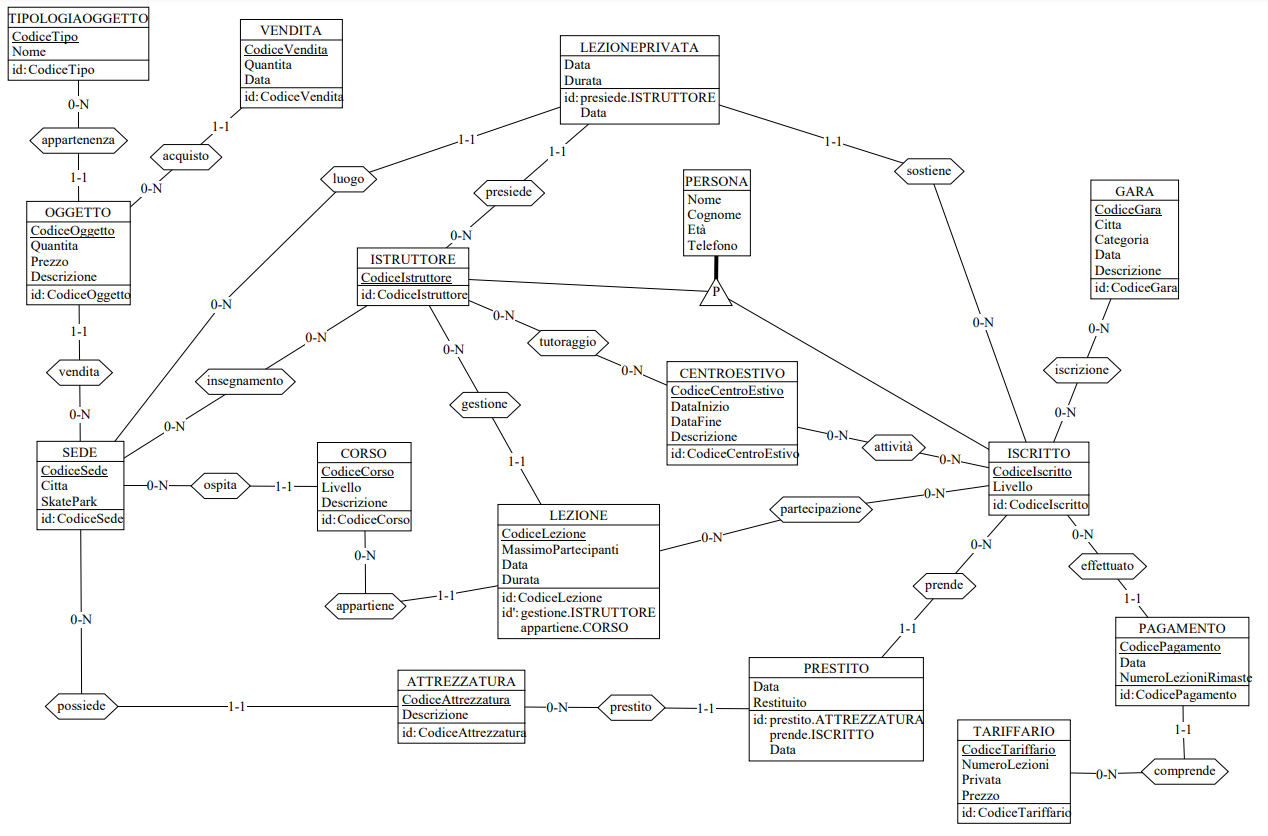
In **tariffario** è presente *NumeroLezioni* che indica il numero di lezioni che si intende comprare assieme e il relativo *Prezzo*. È inoltre presente l’attributo *Privata*, di tipo boolean, che indica se la lezione comprata è privata o meno; se lo fosse, l’iscritto ha un nuovo pagamento per le lezioni private, se no ha il pagamento per le lezioni regolari dei corsi.

In **pagamento** è presente l’attributo *NumeroLezioniRimaste* il cui scopo è tenere traccia del numero di lezioni già pagate dall’iscritto e non ancora effettuate.   
Un iscritto può avere più pagamenti.



Una **sede** possiede diverse **attrezzature** che vengono messe a disposizione degli **iscritti** nel caso in cui non dovessero disporre di determinati materiali sportivi. In questo caso viene effettuato un **prestito**, che presenta un attributo *Restituito* di tipo boolean che andrà ad indicare se l’oggetto in questione è stato riconsegnato o meno.



**Schema finale**

**PROGETTAZIONE LOGICA**

**Stima del volume dei dati**

**Immagine che contiene tavolo

Descrizione generata automaticamente**

Il numero delle partecipazioni si ottiene tramite la moltiplicazione della media delle lezioni per corso in un anno per la media degli iscritti per corso in un anno.

**Descrizione delle operazioni principali e stima della loro frequenza**

Nella precedente fase di analisi sono già state elencate le operazioni principali da effettuare; in questa tabella sono riportate insieme alla relativa frequenza:

Immagine che contiene tavolo

Descrizione generata automaticamente

**Tabelle degli accessi**

Di seguito, sono presenti le tabelle degli accessi delle operazioni.

**OP 1 – Inserire un istruttore**

In questa e nelle seguenti due operazioni sarà solamente necessario effettuare un accesso all’entità desiderata.

****

**OP 2 – Aggiungere un nuovo iscritto**

**Immagine che contiene testo

Descrizione generata automaticamente**

**OP 3 – Inserire una lezione**

**Immagine che contiene testo

Descrizione generata automaticamente**

**OP 4 – Verificare il pagamento di una lezione dato un iscritto**

Prima di effettuare l’iscrizione ad una lezione bisogna controllare tutti i pagamenti a nome di un iscritto per trovarne uno con il *numeroLezioniRimaste* maggiore di 0, questo comporta leggere tutti i pagamenti del cliente che in media sono 18,5.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 5 – Iscrivere un partecipante ad una lezione**

Per verificare il numero di partecipanti già iscritti prendo come valore medio 3, essendo per ora 6 il numero massimo di partecipanti.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 6 – Visualizzare tutti gli iscritti ad una lezione**

Come nell’operazione precedente, considero come valore medio dei partecipanti 3.

**Immagine che contiene testo

Descrizione generata automaticamente**

**OP 7 – Visualizzare il tariffario più venduto**

Per questa operazione vanno letti tutti i pagamenti e i tre tipi diversi di tariffari in modo da poter determinare quale di essi sia il più venduto.

Immagine che contiene tavolo

Descrizione generata automaticamente

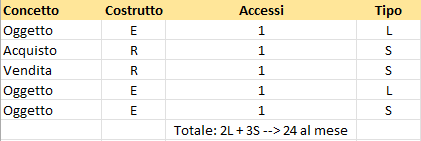
**OP 8 – Controllare se l’attrezzatura prestata è stata restituita**

In quest’operazione è necessario un unico accesso in lettura per controllare l’attributo “restituito”, di tipo boolean.

****

**OP 9 – Acquistare n T-Shirt**

In quest’operazione è necessario cercare l’oggetto desiderato, registrare un nuovo acquisto e aggiornare il numero di oggetti rimasti.

****

**OP 10 – Visualizzare le sedi di un corso in base al livello di un iscritto**

È necessario visualizzare le lezioni a cui può partecipare l’iscritto in modo da individuarne il corso e le relative sedi.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 11 – Visualizzare l’elenco degli istruttori disponibili in una certa data**

Per questa operazione, ricevuta in input una data, sarà necessario visualizzare sia le lezioni normali che quelle private per assicurarsi che l’istruttore desiderato non sia già impegnato in una di queste.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 12 – Prenotazione di una lezione privata**

Supponiamo di conoscere il codice identificativo dell’iscritto e dell’istruttore scelto da quest’ultimo per la lezione, oltre a quello della sede destinata all’attività; basta effettuare una scrittura dell’entità Lezione Privata.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 13 – Visualizzare gli iscritti ad un centro estivo**

All’inizio, conoscendo già l’identificativo del centro estivo, effettuo 30 letture sulla relazione Attività che lega tale entità con Iscritto; di conseguenza dovrò effettuare altrettante letture su Iscritto.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 14 – Visualizzare il guadagno totale dell’anno**

Similmente all’operazione 7, si dovrà accedere a tutti i pagamenti effettuati e ai diversi tipi di tariffario per poter riuscire a visualizzare il guadagno totale annuo.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**OP 15 – Aggiornare il prezzo di un oggetto**

Qui è necessario effettuare un update, perciò verrà svolto solo un accesso in lettura e uno in scrittura per modificare l’attributo desiderato.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

**Raffinamento dello schema**

**Eliminazione di gerarchie**

Per l’eliminazione della gerarchia persona, abbiamo scelto di adottare l’approccio del collasso verso il basso, copiando gli attributi in istruttore e iscritto. Abbiamo preso questa decisione poiché la differenza tra istruttore e iscritto era solo l’attributo del livello; quindi, l’istruttore sarebbe stata un’entità senza attributi oltre al codice identificativo.

**Eliminazione delle chiavi composte**

Per la maggior parte delle entità si è scelto di utilizzare un codice identificativo univoco, mentre per lezione privata la chiave è data da istruttore e data e per il prestito invece da attrezzatura, iscritto e data.

**Eliminazione degli identificatori esterni**

Rispetto allo schema E/R sono state eliminate le seguenti relazioni:

* Tiene, importando codiceSede in Oggetto
* Appartenenza, importando codiceTipo in Oggetto
* Vendita, importando codiceOggetto in Vendita
* Possiede, importando codiceSede in Attrezzatura
* Prestito, importando codiceAttrezzatura in Prestito
* Prende, importando codiceIscritto in Prestito
* Effettuato, importando codiceIscritto in Pagamento
* Comprende, importando numeroLezioni in Pagamento
* Luogo, importando codiceSede in LezionePrivata
* Presiede, importando codiceIstruttore in LezionePrivata
* Sostiene, importando codiceIscritto in LezionePrivata
* Gestione, importando codiceIstruttore in Lezione
* Ospita, importando codiceSede in Corso
* Appartiene, importando codiceCorso in Lezione

**Analisi delle ridondanze**

La ridondanza è stata introdotta grazie all’utilizzo dell’attributo NumeroLezioniRimaste, che ci permette di semplificare le operazioni relative al calcolo di questo numero.

**OP 4 – Verificare il pagamento di una lezione dato un iscritto**

In questa situazione avere una ridondanza semplifica notevolmente il calcolo di questo parametro utile alla gestione dei pagamenti.

**Immagine che contiene tavolo

Descrizione generata automaticamente**

Senza ridondanza si dovrebbe per forza controllare le partecipazioni dell’iscritto per ricavare il numero delle lezioni svolte.

Immagine che contiene tavolo

Descrizione generata automaticamente

Notiamo quindi come convenga mantenere la ridondanza in quanto presenti un vantaggio risparmiando quasi un centinaio di accessi quotidiani.

**Traduzione di entità e associazioni in relazioni**

ATTREZZATURA(codiceAttrezzatura, descrizione)  
FK: codiceSede REFERENCES SEDE

CENTROESTIVO(codiceCentroEstivo, dataInizio, dataFine, descrizione)

CORSO(codiceCorso, livello, descrizione)  
FK: codiceSede REFERENCES SEDE

GARA(codiceGara, città, categoria, data, descrizione)

INSEGNAMENTO(codiceIstruttore, codiceSede)  
FK: codiceSede REFERENCES SEDE  
FK: codiceIstruttore REFERENCES ISTRUTTORE

ISCRITTO(codiceIscritto, nome, cognome, eta, telefono, livello)

ISCRIZIONE(codiceIscritto, codiceLezione)  
FK: codiceLezione REFERENCES LEZIONE  
FK: codiceIscritto REFERENCES ISCRITTO

ISCRIZIONECENTROESTIVO(codiceCentroEstivo, codiceIscritto)  
FK: codiceIscritto REFERENCES ISCRITTO  
FK: codiceCentroEstivo REFERENCES CENTRO\_ESTIVO

ISCRIZIONEGARA(codiceIscritto, codiceGara)  
FK: codiceIscritto REFERENCES ISCRITTO  
FK: codiceGara REFERENCES GARA

ISTRUTTORE(codiceIstruttore, nome, cognome, eta, telefono)

LEZIONE(codiceLezione, codiceIstruttore, codiceCorso, data, durata, massimoPartecipanti)  
FK: codiceIstruttore REFERENCES ISTRUTTORE  
FK: codiceCorso REFERENCES CORSO

LEZIONEPRIVATA(data, durata, codiceIstruttore)  
FK: codiceIstruttore REFERENCES ISTRUTTORE  
FK: codiceSede REFERENCES SEDE  
FK: codiceIscritto REFERENCES ISCRITTO

OGGETTO(codiceOggetto, descrizione, prezzo, quantita)  
FK: codiceSede REFERENCES SEDE  
FK: codiceTipo REFERENCES TIPOLOGIA\_OGGETTO

PAGAMENTO(codicePagamento, data, numeroLezioniRimaste, codiceIscritto, codiceTariffario)  
FK: codiceIscritto REFERENCES ISCRITTO  
FK: codiceTariffario REFERENCES TARIFFARIO

PRESTITO(codiceAttrezzatura, codiceIscritto, data, restituito)  
FK: codiceAttrezzatura REFERENCES ATTREZZATURA  
FK: codiceIscritto REFERENCES ISCRITTO

SEDE(codiceSede, citta, skatepark)

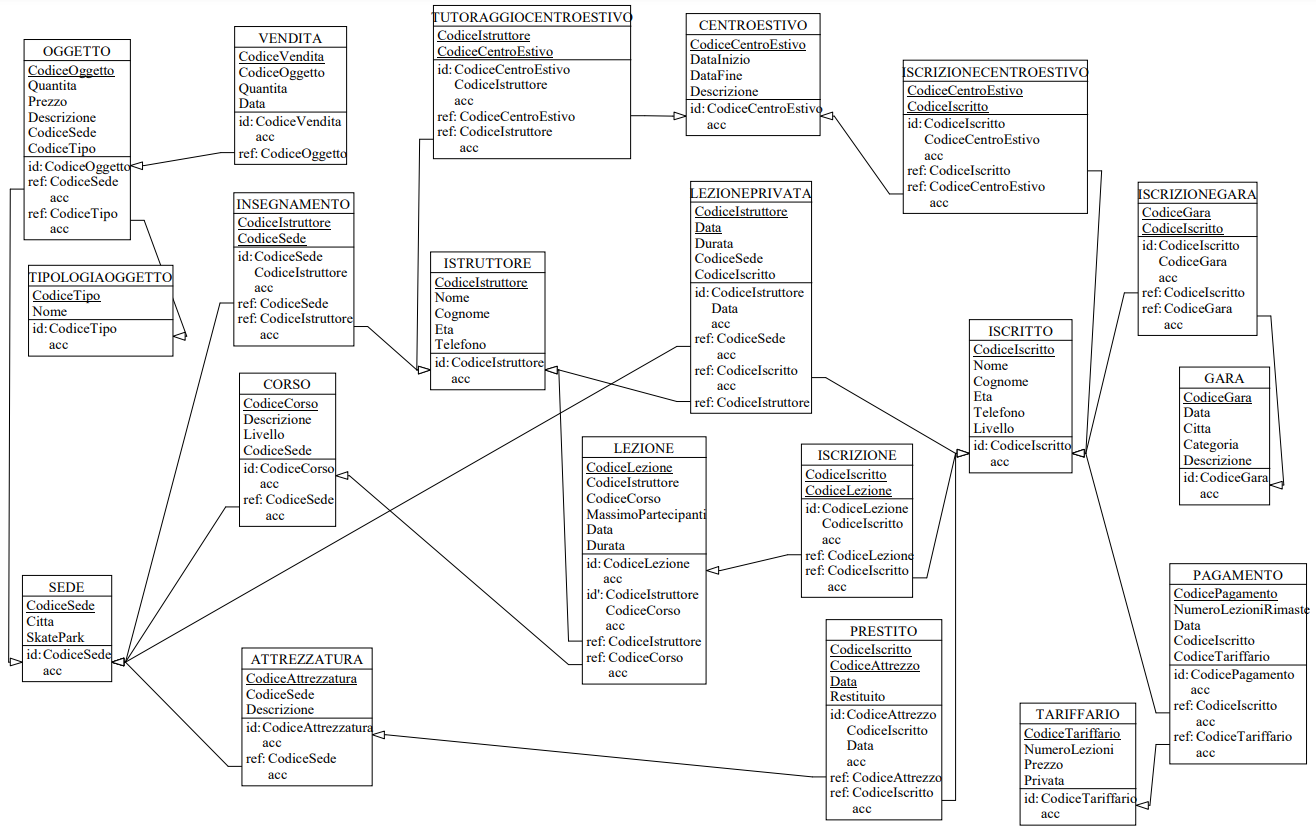
TARIFFARIO(codiceTariffario, numeroLezioni, prezzo, privata)

TIPOLOGIAOGGETTO(codiceTipo, nome)

TUTORAGGIOCENTROESTIVO(codiceIstruttore, codiceCentroEstivo)  
FK: codiceIstruttore REFERENCES ISTRUTTORE  
FK: codiceCentroEstivo REFERENCES CENTRO\_ESTIVO

VENDITA(codiceVendita, codiceOggetto, data, quantita)  
FK: codiceOggetto REFERENCES OGGETTO

**Schema relazionale finale**



**Traduzione delle operazioni in query SQL**

**CREAZIONE TABELLE**

**CREATE TABLE** Attrezzatura (  
 CodiceAttrezzatura **INT NOT NULL IDENTITY**,  
 CodiceSede **INT NOT NULL**,  
 Descrizione **VARCHAR**(200) **NOT NULL**,  
 **CONSTRAINT** ID\_ATTREZZATURA\_ID **PRIMARY KEY** (CodiceAttrezzatura)  
);

**CREATE TABLE** CentroEstivo (  
 CodiceCentroEstivo **INT NOT NULL IDENTITY**,  
 DataInizio **DATE NOT NULL**,  
 DataFine **DATE NOT NULL**,  
 Descrizione **VARCHAR**(200) **NOT NULL**,  
 **CONSTRAINT** ID\_CENTROESTIVO\_ID **PRIMARY** **KEY** (CodiceCentroEstivo)  
);

**CREATE TABLE** Corso (  
 CodiceCorso **INT NOT NULL IDENTITY**,  
 Descrizione **VARCHAR**(200) **NOT NULL**,  
 Livello **VARCHAR**(20) **NOT NULL**,  
 CodiceSede **INT NOT NULL**,  
 **CONSTRAINT** ID\_CORSO\_ID **PRIMARY** **KEY** (CodiceCorso)  
);

**CREATE TABLE** Gara (  
 CodiceGara **INT NOT NULL IDENTITY**, Data **DATETIME2 NOT NULL**,  
 Citta **VARCHAR**(30) **NOT NULL**,  
 Categoria **VARCHAR**(30) **NOT NULL**,  
 Descrizione **VARCHAR**(200) **NOT NULL**,  
 **CONSTRAINT** ID\_GARA\_ID **PRIMARY KEY** (CodiceGara)  
);

**CREATE TABLE** Insegnamento (  
 CodiceIstruttore **INT NOT NULL**,  
 CodiceSede **INT NOT NULL**,  
 **CONSTRAINT** ID\_insegnamento\_ID **PRIMARY KEY** (CodiceSede, CodiceIstruttore)  
);

**CREATE TABLE** Iscritto (  
 CodiceIscritto **INT NOT NULL IDENTITY**,  
 Nome **VARCHAR**(30) **NOT NULL**,  
 Cognome **VARCHAR**(30) **NOT NULL**,  
 Eta **INT NOT NULL**,  
 Telefono **VARCHAR**(20) **NOT NULL**,  
 Livello **VARCHAR**(20) **NOT** **NULL**,  
 **CONSTRAINT** FKPER\_ISC\_ID **PRIMARY** **KEY** (CodiceIscritto)  
);

**CREATE TABLE** Iscrizione (  
 CodiceIscritto **INT NOT NULL**,  
 CodiceLezione **INT NOT NULL**,  
 **CONSTRAINT** ID\_partecipazione\_ID **PRIMARY** **KEY** (CodiceLezione, CodiceIscritto)  
);

**CREATE TABLE** IscrizioneCentroEstivo (  
 CodiceCentroEstivo **INT NOT NULL**,  
 CodiceIscritto **INT NOT NULL**,  
 **CONSTRAINT** ID\_attivita\_ID **PRIMARY** **KEY** (CodiceIscritto, CodiceCentroEstivo)  
);

**CREATE** **TABLE** IscrizioneGara (  
 CodiceGara **INT NOT NULL**,  
 CodiceIscritto **INT NOT** **NULL**,  
 **CONSTRAINT** ID\_iscrizione\_ID **PRIMARY** **KEY** (CodiceIscritto, CodiceGara)  
);

**CREATE TABLE** Istruttore (  
 CodiceIstruttore **INT NOT NULL IDENTITY**,  
 Nome **VARCHAR**(30) **NOT NULL**,  
 Cognome **VARCHAR**(30) **NOT NULL**,  
 Eta **INT NOT NULL**,  
 Telefono **VARCHAR**(20) **NOT NULL**,  
 **CONSTRAINT** FKPER\_IST\_ID **PRIMARY KEY** (CodiceIstruttore)  
);

**CREATE TABLE** Lezione (  
 CodiceLezione **INT NOT NULL IDENTITY**,  
 CodiceIstruttore **INT NOT NULL**,  
 CodiceCorso **INT NOT NULL**,  
 MassimoPartecipanti **INT NOT NULL**,  
 Data **DATETIME2 NOT NULL**,  
 Durata **INT NOT NULL**,  
 **CONSTRAINT** ID\_LEZIONE\_ID **PRIMARY KEY** (CodiceLezione),  
 **CONSTRAINT** SID\_LEZIONE\_ID **UNIQUE** (CodiceIstruttore, CodiceCorso)  
);

**CREATE TABLE** LezionePrivata (  
 CodiceIstruttore **INT NOT NULL**,  
 Data **DATETIME2 NOT NULL**,  
 Durata **INT NOT NULL**,  
 CodiceSede **INT NOT NULL**,  
 CodiceIscritto **INT NOT** **NULL**,  
 **CONSTRAINT** ID\_LEZIONE\_PRIVATA\_ID **PRIMARY KEY** (CodiceIstruttore, Data)  
);

**CREATE TABLE** Oggetto (  
 CodiceOggetto **INT NOT NULL IDENTITY**,  
 Quantita **INT NOT NULL**,  
 Prezzo **REAL NOT NULL**,  
 Descrizione **VARCHAR**(200) **NOT NULL**,  
 CodiceSede **INT NOT NULL**,  
 CodiceTipo **INT NOT NULL**,  
 **CONSTRAINT** IDOGGETTO **PRIMARY KEY** (CodiceOggetto)  
);

**CREATE TABLE** Pagamento (  
 CodicePagamento **INT NOT NULL IDENTITY**,  
 NumeroLezioniRimaste **INT NOT NULL**,  
 Data **DATETIME2 NOT NULL**,  
 CodiceIscritto **INT NOT NULL**,  
 CodiceTariffario **INT NOT NULL**,  
 **CONSTRAINT** ID\_PAGAMENTO\_ID **PRIMARY KEY** (CodicePagamento)  
);

**CREATE TABLE** Prestito (  
 CodiceIscritto **INT NOT NULL**,  
 CodiceAttrezzatura **INT NOT NULL**,  
 Data **DATETIME2 NOT NULL**,  
 Restituito **BIT NOT NULL**,  
 **CONSTRAINT** ID\_PRESTITO\_ID **PRIMARY** **KEY** (CodiceAttrezzatura, CodiceIscritto, Data)  
);

**CREATE TABLE** Sede (  
 CodiceSede **INT NOT NULL IDENTITY**,  
 Citta **VARCHAR**(30) **NOT NULL**,  
 SkatePark **VARCHAR**(30) **NOT NULL**,  
 **CONSTRAINT** ID\_SEDE\_ID **PRIMARY** **KEY** (CodiceSede)  
);

**CREATE TABLE** Tariffario (  
 CodiceTariffario **INT NOT NULL IDENTITY**,  
 NumeroLezioni **INT NOT NULL**,  
 Prezzo **REAL NOT NULL**,  
 Privata **BIT NOT NULL**,  
 **CONSTRAINT** ID\_TARIFFARIO\_ID **PRIMARY** **KEY** (CodiceTariffario)  
);

**CREATE TABLE** TipologiaOggetto (  
 CodiceTipo **INT NOT NULL IDENTITY**,  
 Nome **VARCHAR**(30) **NOT NULL**,  
 **CONSTRAINT** ID\_TIPOLOGIA\_OGGETTO\_ID **PRIMARY** **KEY** (CodiceTipo)  
);

**CREATE** **TABLE** TutoraggioCentroEstivo (  
 CodiceIstruttore **INT NOT NULL**,  
 CodiceCentroEstivo **INT NOT NULL**,  
 **CONSTRAINT** ID\_tutoraggio\_ID **PRIMARY** **KEY** (CodiceCentroEstivo, CodiceIstruttore)  
);

**CREATE TABLE** Vendita (  
 CodiceVendita **INT NOT NULL IDENTITY**,  
 CodiceOggetto **INT NOT NULL**,  
 Quantita **INT NOT NULL**,  
 Data **DATETIME2 NOT NULL**, **CONSTRAINT** ID\_VENDITA\_ID **PRIMARY** **KEY** (CodiceVendita)  
);

**OPERAZIONI**

**OP 1 - INSERIRE UN ISTRUTTORE**

**INSERT INTO** istruttore (nome, cognome, eta, telefono)  
**VALUES** (?, ?, ?, ?)

**OP 2 - INSERIRE UN NUOVO ISCRITTO**

**INSERT INTO** iscritto (nome, cognome, eta, telefono, livello)  
**VALUES** (?, ?, ?, ?, ?)

**OP 3 - INSERIRE UNA LEZIONE**

**INSERT INTO** lezione (data, durata, massimoPartecipanti, codiceIstruttore, codiceCorso)  
**VALUES** (?, ?, ?, ?, ?)

**OP 4 – VERIFICARE PAGAMENTO LEZIONE**

Controllo tutti i pagamenti di un iscritto per verificare che esista almeno un pagamento con numeroLezioniRimaste maggiore di zero

**SELECT** \*  
**FROM** pagamento  
**WHERE** codiceIscritto = ?  
**AND** numeroLezioniRimaste > 0

**OP 5 – ISCRIVERE UN PARTECIPANTE AD UNA LEZIONE**

Se la condizione della query precedente risulta soddisfatta, un iscritto può partecipare a una lezione e il numero di lezioni rimaste viene diminuito di 1. Per potersi iscrivere bisogna inoltre controllare che il numero di partecipanti sia minore del numero massimo di partecipanti.

**SELECT COUNT (\*) AS** partecipanti **FROM** iscrizione I, lezione L **WHERE** I.codiceLezione= ?  
**AND** L.codiceLezione = I.codiceLezione

**GROUP BY** L.codiceLezione, L.massimoPartecipanti

**HAVING COUNT (\*)** <L.massimoPartecipanti

**INSERT INTO** iscrizione (codiceIscritto, codiceLezione)  
**VALUES** (?, ?)

**UPDATE** pagamento  
**SET** numeroLezioniRimaste = numeroLezioniRimaste - 1  
**WHERE** codiceIscritto = ?

**OP 6 – VISUALIZZARE TUTTI GLI ISCRITTI A UNA LEZIONE**

**SELECT** nome, cognome  
**FROM** iscrizione I  
**JOIN** iscritto ISC **ON** I.codiceIscritto = ISC.codiceIscritto   
**WHERE** codiceLezione = ?  
**ORDER BY** cognome

**OP 7 – VISUALIZZARE IL TARIFFARIO PIU’ VENDUTO**

**SELECT TOP (1) COUNT(**\***) AS** quantita,T.codiceTariffario  
**FROM** pagamento P, tariffario T  
**WHERE** P. codiceTariffario = T. codiceTariffario  
**GROUP BY** P. codiceTariffario  
**ORDER BY COUNT (**\***) DESC**

**OP 8 – CONTROLLARE SE L’ATTREZZATURA PRESTATA E’ STATA RESTITUITA**   
**SELECT** restituito   
**FROM** prestito  
**WHERE** codiceAttrezzatura = ?

**OP 9 – ACQUISTO DI N T-SHIRT**

Innanzitutto, bisogna controllare che gli oggetti desiderati siano disponibili.

**SELECT** quantita   
**FROM** oggetto  
**WHERE** codiceOggetto = ?  
**AND** quantita >= ?

Una volta controllato che l’oggetto sia disponibile, procedo all’acquisto aggiungendo una nuova vendita.

**INSERT INTO** vendita (data, quantita, codiceOggetto)  
**VALUES** (**GETDATE**(), ?, ?)

A questo punto bisogna diminuire la quantità dell’oggetto.

**UPDATE** oggetto  
**SET** quantita = quantita - ?  
**WHERE** codiceOggetto = ?

**OP 10 – VISUALIZZARE LE SEDI DI UN CORSO IN BASE AL LIVELLO DI UN ISCRITTO**

**SELECT** S.\*  
**FROM** corso C   
**JOIN** sede S **ON** C.codiceSede = S.codiceSede  
**WHERE** C.livello = ?

**OP 11 – VISUALIZZARE L’ELENCO DEGLI ISTRUTTORI DISPONIBILI IN UNA CERTA DATA**

Il dato da inserire in questa query è la data con orario per cui si vuole controllare la disponibilità degli istruttori

**SELECT** nome, cognome  
**FROM** istruttore  
**WHERE** codiceIstruttore **NOT IN** (**SELECT** codiceIstruttore  
 **FROM** lezione L  
 **WHERE** L.data **BETWEEN** ? **AND** DATEADD(mi, L.durata, ?)  
 **UNION** **SELECT** codiceIstruttore  
 **FROM** lezionePrivata LP  
 **WHERE** LP.data **BETWEEN** ? **AND** DATEADD(mi, LP.durata, ?))

**OP 12 – PRENOTAZIONE DI UNA LEZIONE PRIVATA**

Dopo aver controllato che l’istruttore richiesto sia libero, posso procedere alla prenotazione di una lezione privata.

**INSERT INTO** lezionePrivata(codiceIstruttore, data, durata, codiceSede, codiceIscritto)  
**VALUES** (?, ?, ?, ?, ?)

**OP 13 – VISUALIZZARE GLI ISCRITTI AD UN CENTRO ESTIVO**

**SELECT** I.\*  
**FROM** iscrizioneCentroEstivo C   
**JOIN** iscritto I **ON** C.codiceIscritto = I.codiceIscritto  
**WHERE** codiceCentroEstivo = ?

**OP 14 – VISUALIZZARE GUADAGNO TOTALE PER ANNO**

I dati da inserire sono la data iniziale e la data finale fra cui visualizzare i guadagni (estremi inclusi), la costante 105 indica il formato italiano per DateTime (gg-mm-aaaa)

**SELECT SUM**(prezzo) **AS** guadagno  
**FROM** tariffario T  
**JOIN** pagamento P **ON** T.codiceTariffario = P.codiceTariffario  
**WHERE** P.data **BETWEEN** **CONVERT**(**DATETIME**, ?, 105) **AND** **CONVERT**(**DATETIME**, ?, 105)+1

Nel caso in cui si voglia inserire solo l’anno (es: 2022) la query si può semplificare in questo modo

**SELECT SUM**(prezzo) **AS** guadagno  
**FROM** tariffario T  
**JOIN** pagamento P **ON** T.codiceTariffario = P.codiceTariffario  
**WHERE** **DATEPART**(yyyy, P.data) = ?

**OP 15 – AGGIORNARE IL PREZZO DI UN OGGETTO**

**UPDATE** oggetto  
**SET** prezzo = ?  
**WHERE** codiceOggetto = ?

**PROGETTAZIONE DELL’APPLICAZIONE**

**ARCHITETTURA**

Lo sviluppo dell’applicazione è stato svolto in C#, WPF e SQL Server, sfruttando gli strumenti messi a disposizione da Visual Studio. In particolare usiamo una versione leggera di SQL Server apposita per lo sviluppo, chiamata LocalDB, che permette di creare e modificare il database direttamente dentro a Visual Studio. Per generare il modello usato dal programma per accedere ai dati è stato utilizzato Entity Framework.

**VIEW**

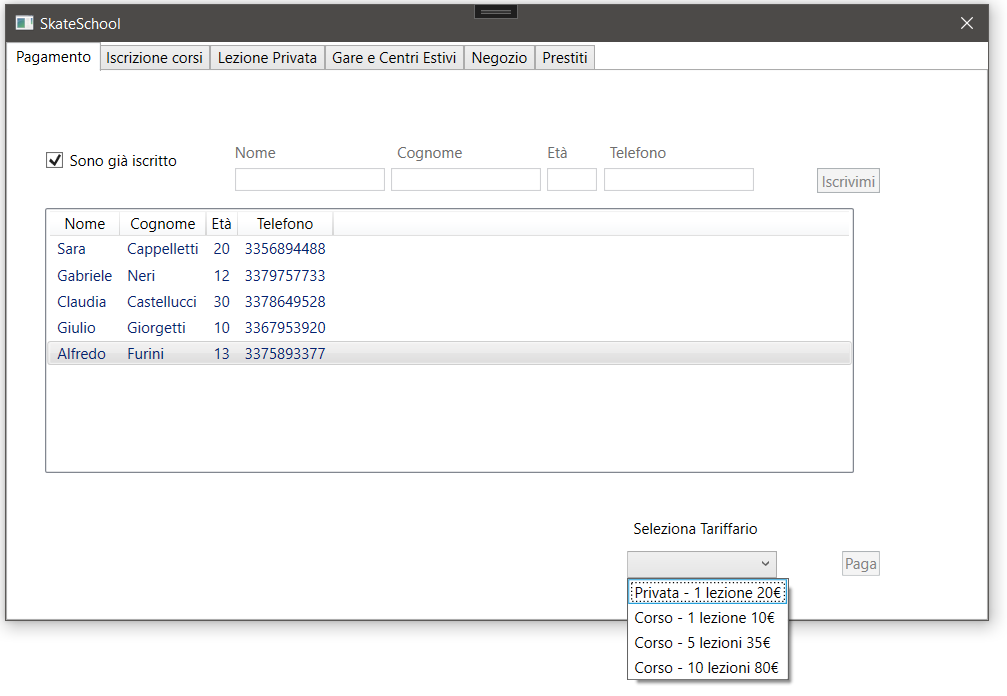
Per verificare la correttezza dei dati inseriti sono stati introdotti dei check che mirano a verificare che l’inserimento dei dati sia opportuno, ad esempio nel caso della registrazione di un utente che i tipi corrispondano; oppure che una persona non possa iscriversi a più lezioni uguali o alla stessa gara.

Figura 1 - Schermata principale per iscrizione e pagamento

In questa pagina è possibile registrare un utente inserendo i relativi dati; se invece la persona in questione dovesse già risultare iscritta, si può proseguire al pagamento di una lezione selezionando il tariffario desiderato tra le diverse tipologie di pacchetti.

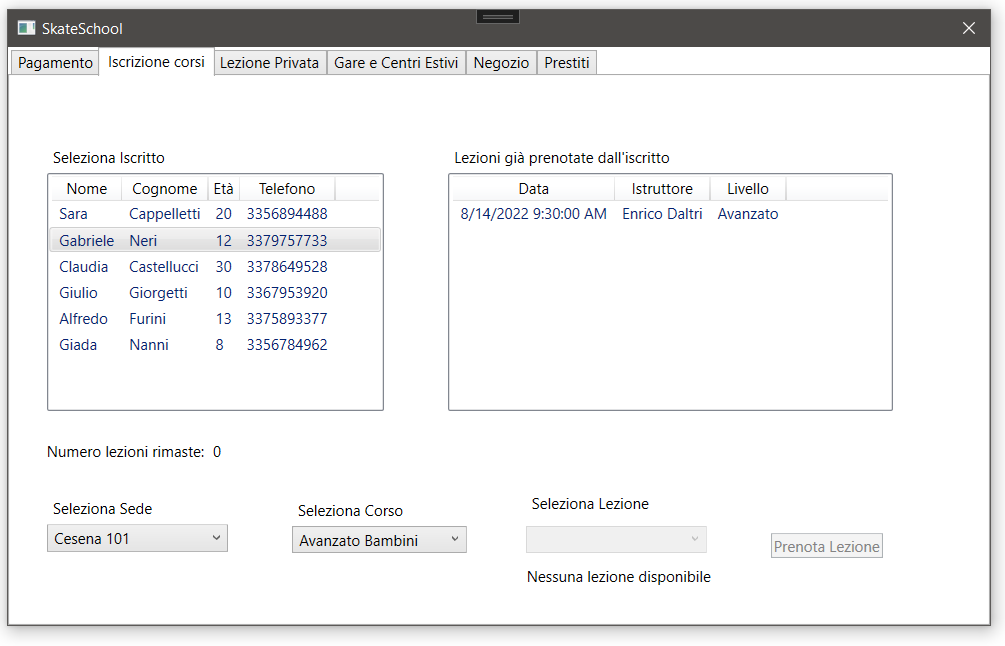
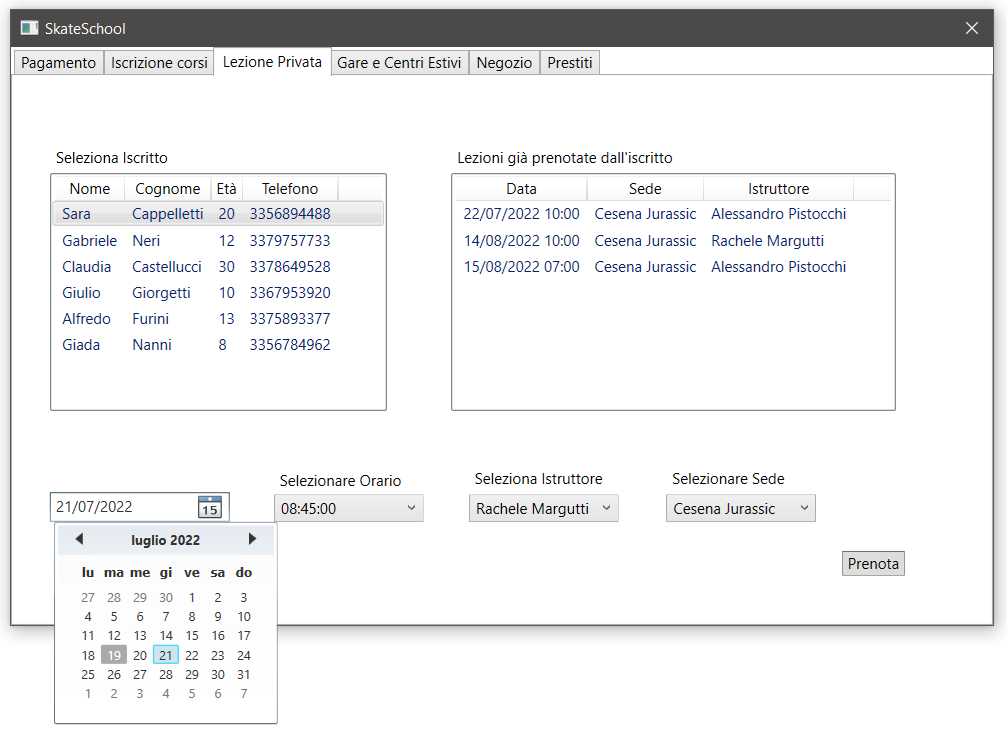


Figura 2 - Schermata per l'iscrizione ad una lezione

In questa pagina sarà possibile procedere all’iscrizione di un cliente alla lezione desiderata, filtrando per sede e corso. Il “Numero lezioni rimaste” indica quante lezioni sono state precedentemente acquistate dal partecipante e, una volta iscritti alla lezione desiderata, questo parametro decrementerà di uno.

Nella tabella destra compaiono le lezioni dei corsi già prenotate da un iscritto, non appariranno le lezioni private possedute.

Se non sono presenti lezioni in quella sede o corso comparirà la scritta “Nessuna lezione disponibile”. Lo stesso messaggio comparirà anche se l’iscritto si è già prenotato in tutte le lezioni disponibili per quella sede e corso.

Figura 3- Schermata per l'iscrizione ad una lezione privata

In questa pagina sarà possibile prenotare una lezione privata, filtrando per data, orario, istruttore e sede. Anche qui è necessario aver già effettuato il pagamento.

Nella tabella destra compaiono tutte le lezioni dei corsi e private già prenotate da un iscritto.

Dopo aver selezionato data e orario sarà possibile selezionare anche l’istruttore dalla lista di quelli disponibili.

Immagine che contiene testo

Descrizione generata automaticamente

Figura 4 - Schermata per l'iscrizione a gara o centro estivo

In questa pagina è possibile, selezionando un iscritto, effettuare l’iscrizione ad un centro estivo o ad una gara e visualizzare gli eventi già prenotati.

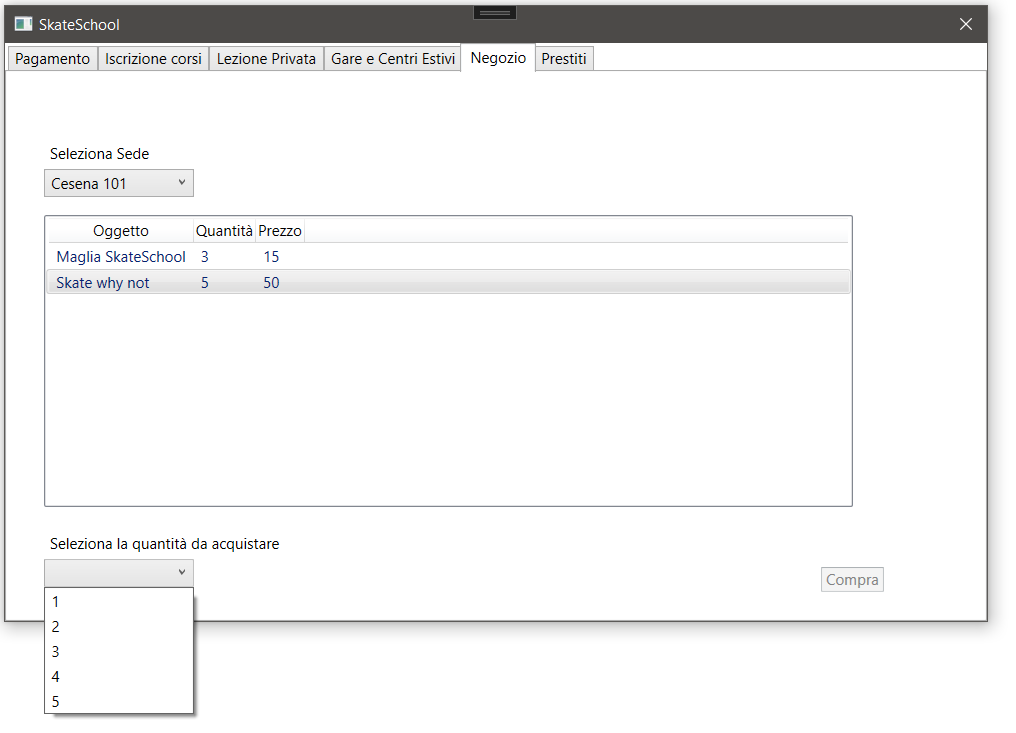


Figura 5 - Schermata per il negozio

Questa pagina si occupa di gestire gli acquisti svolti nei negozi delle varie sedi di Skate School. Selezionando la sede desiderata, sarà possibile visualizzare tutti gli oggetti disponibili alla vendita e la relativa quantità; dalla tendina sottostante si potrà scegliere la quantità da acquistare e tramite il pulsante “Compra” terminare l’acquisto, aggiornando in seguito i dati della tabella del negozio.

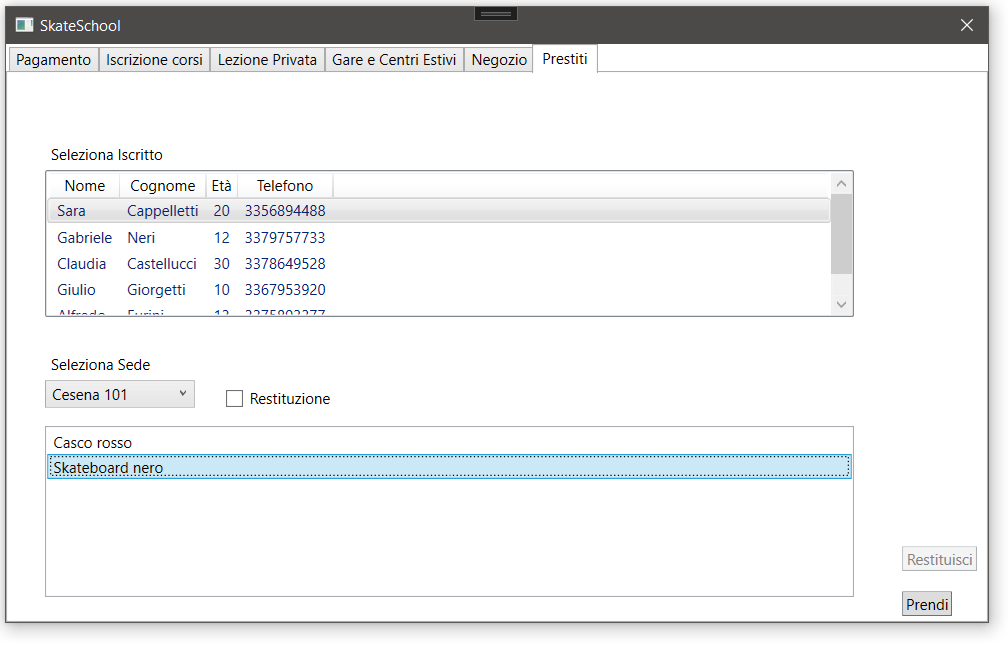


Figura 6 - Schermata per la gestione dei prestiti

In questa schermata sono invece gestiti i prestiti di attrezzatura sportiva utile alle lezioni.

Una volta selezionati l’iscritto e la sede, si procederà alla scelta dell’articolo desiderato e premendo il bottone “Prendi” esso verrà spostato nella lista degli oggetti da restituire.

Selezionando la CheckBox “Restituzione” comparirà la lista degli oggetti presi e il processo sarà l’inverso.