# Progetto Basi di Dati


## Merola Carlo

## Matricola n.082776


### Testo

Si vuole progettare una base di dati per tenere traccia degli utenti che si sottoscrivono al corso di yoga o che acquistano un tappetino per praticare e merce fidelizzante.

Tutti gli utenti hanno un identificativo. Gli utenti possono essere registrati o non registrati. Nel caso l'utente non sia registrato può acquistare comunque la merce senza che i suoi dati personali vengano registrati. Se l'utente vuole prenotare un abbonamento gli viene chiesto di procedere alla registrazione o di contattare direttamente l'insegnante del corso. In quest'ultimo caso questo non verrà registrato nel database.

Per un utente registrato viene registrato username, password, data di registrazione e numero di volte che ha pagato il corso. Lo username deve essere univoco e la password viene salvata tramite la funzione di hashing, pertanto è necessaria un lunghezza di 255 caratteri. Se è la prima volta che si avvicina all'insegnamento, gli viene offerta una prima lezione di prova gratuita.

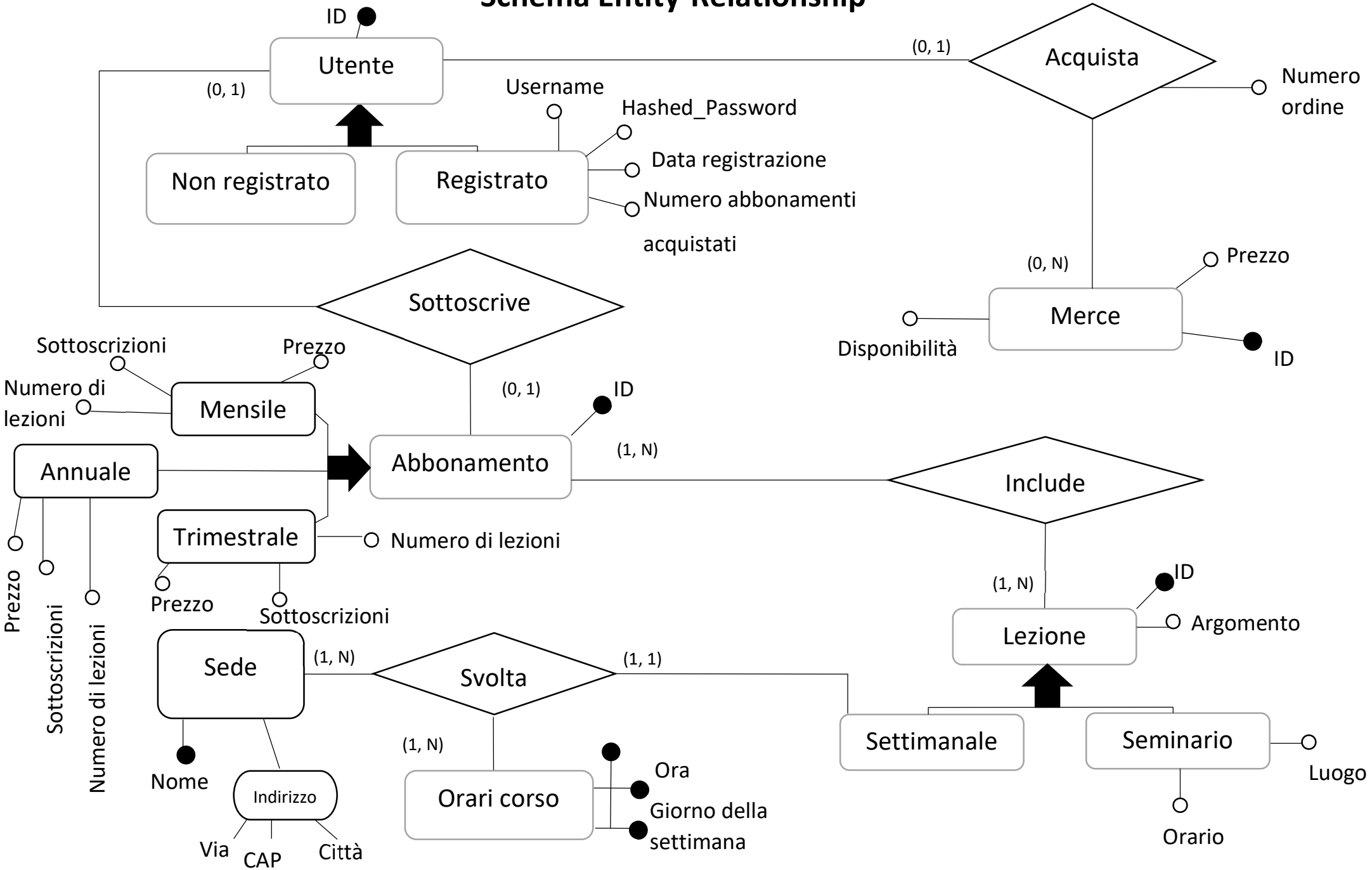
Gli abbonamenti del corso si suddividono in mensile, trimestrale e annuale. Per ogni tipologia di abbonamento viene salvato inoltre il prezzo, il numero di utenti che lo hanno sottoscritto (può essere derivato) e il numero di singole lezioni che comprende (può essere derivato).

La merce comprende un identificativo di 6 caratteri, il prezzo e la disponibilità. Le prime due lettere dell'identificativo si riferiscono alla tipologia del prodotto. Pertanto non è necessario registrare un attributo per il tipo di oggetto.

Ogni lezione è compresa di data e di argomento trattato (es. Kriya per la gestione dell'ansia ecc.). Inoltre può trattarsi di una lezione ordinaria nel caso si svolga nei giorni della settimana usuali, o in alternativa dei seminari che si svolgono in giorni non prestabiliti, o in luoghi non ordinari.

Vengono registrati gli orari e i giorni in cui si svolgono le lezioni, insieme alla sede in cui tale giorno si svolge il corso (sono disponibili 3 diverse sedi in base al giorno della settimana). Questi dati vengono spesso acceduti in riferimenti diversi rispetto agli argomenti delle lezioni. Per ogni sede viene inoltre registrato l'indirizzo (via, CAP, città).

# Schema Entity-Relationship



# Ristrutturazione (in ottica di semplificazione)

## Analisi delle ridondanze

È possibile rimuovere il numero di sottoscrizioni, il numero di lezioni e il numero di abbonamenti acquistati da un utente in quanto attributi derivabili.

## Eliminazione delle gerarchie

Rimozione delle gerarchie totali tramite accorpamento. Aggiungere relazioni non è necessario in quanto non ci sono operazioni che fanno distinzioni tra entità padre ed entità figlie.

Per le entità “lezione settimanale” e “seminario”, viene sostituita la generalizzazione con una relazione, in quanto delle operazioni fanno distinzione tra le due entità figlie.

Invece, in ottica di semplificazione dello schema, per le entità “abbonamento” e “utente” viene effettuato l’accorpamento delle figlie nel padre, nonostante si tratti di generalizzazione totale. Vengono introdotti degli attributi per la distinzione nel primo caso, e la possibilità di avere attributi con valori nulli nel secondo caso.

## Accorpamento delle entità

Operazione svolta insieme all’eliminazione delle gerarchie.

## Eliminazione degli attributi composti

Scompongo l’attributo indirizzo.

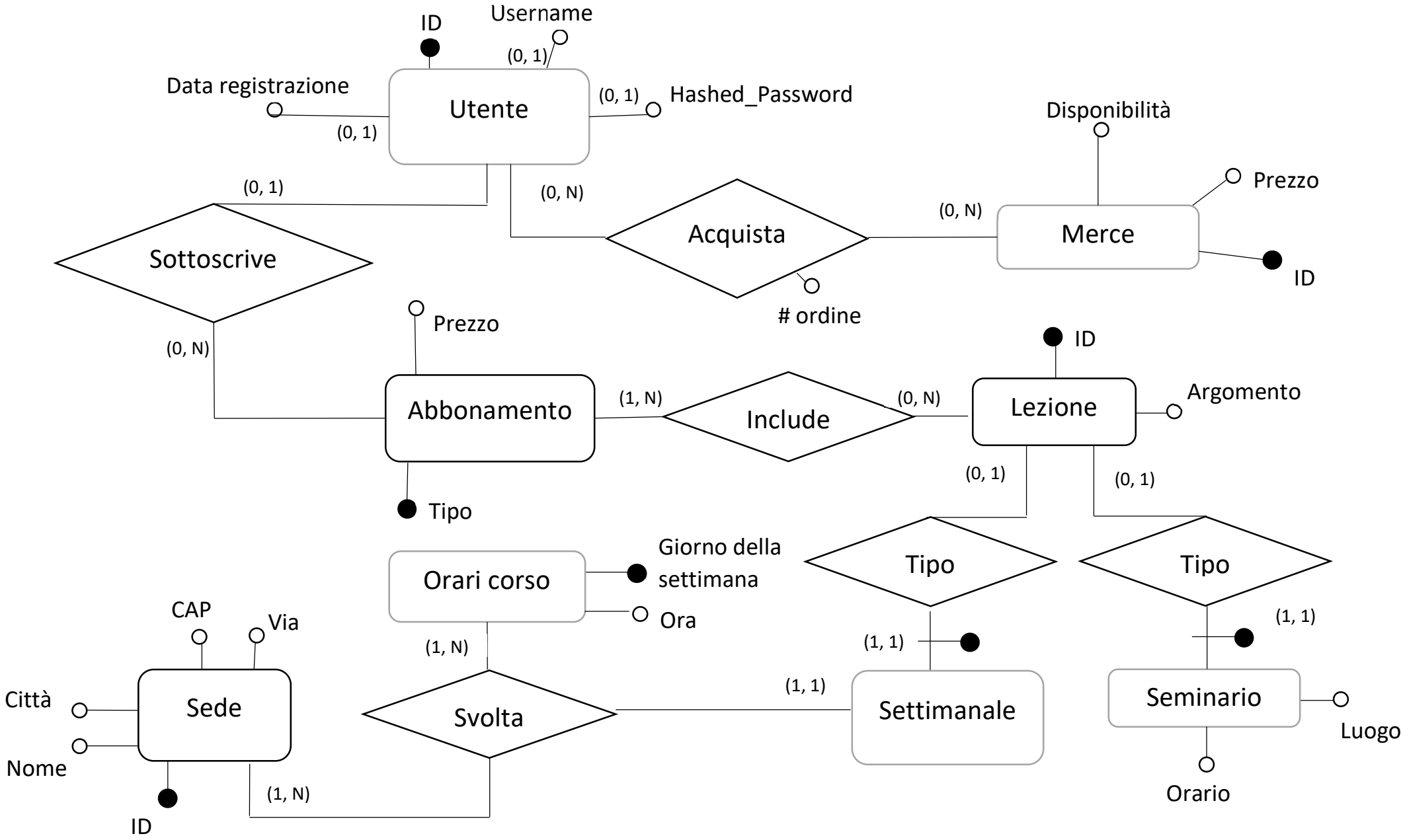
## Scelta degli identificatori principali

Per l’entità “Orari corso”, scelgo “Giorno della settimana” come unico identificatore. Infatti non verranno mai svolte due lezioni nello stesso giorno.

Per l’entità “Abbonamento” scelto l’attributo tipo come chiave primaria.

Aggiunta di un identificativo per l’entità “Sede”, essendo il nome un attributo lungo.

# Schema Ristrutturato



# Traduzione verso il Modello Relazionale

Utente(ID, Username, Hasehd_Password, DataRegistrazione)

Sottoscrive(Utente, TipoAbbonamento)

Acquista(Utente, Merce, NumeroOrdine)

Merce(ID, Disponibilita, Prezzo)

Abbonamento(Tipo, Prezzo)

Include(TipoAbbonamento, Lezione)

Lezione(ID, Argomento)

LezioneSettimanale(IDLezione, GiornoSettimana, Sede)

Seminario(IDLezione, GiornoSettimana, Ora, Luogo)

OrariCorso(GiornoSettimana, Ora)

Sede(ID, Nome, Via, Cap, Citta)

## Vincoli di integrità referenziale

Sottoscrive.Utente → Utente.ID

Sottoscrive.TipoAbbonamento → Abbonamento.Tipo

Acquista.Utente → Utente.ID

Acquista.Merce → Merce.ID

Include.TipoAbbonamento → Abbonamento.Tipo

Include.Lezione → Lezione.ID

Seminario.IDLezione → Lezione.ID

LezioneSettimanale.IDLezione → Lezione.ID

LezioneSettimanale.GiornoSettimana → OrariCorso.GiornoSettimana

LezioneSettimanale.Sede → Sede.ID

# Creazione Base di Dati

```
Create Table Utente(
        ID serial primary key not null,
        Username varchar(20) UNIQUE,
        Hashed_Password varchar(255),
        DataRegistrazione date,
        NumeroAbbonamenti integer,
)

Create Table Abbonamento(
        Tipo varchar(11) primary key not null,
        Prezzo numeric(5,2) not null check(Prezzo>0)
)
```

```
Create Table Sottoscrive(
        Utente integer not null references Utente(ID),
        TipoAbbonamento varchar(11) not null references Abbonamento(tipo),
        Primary Key (Utente, TipoAbbonamento)
)

Create Table Merce(
        ID char(6) primary key not null,
        Disponibilita int not null  check(Disponibilita >= 0),
        Prezzo numeric(4,2) not null check(Prezzo>0)
)


Create Table Acquista(
        ID int not null references Utente(ID),
        Merce char(6) not null references Merce(ID),
```

```
        NumeroOrdine serial,
        primary key(ID, Merce)
)

Create Table Lezione(
        ID serial not null primary key,
        Argomento varchar(50)
)

Create Table Include(
        TipoAbbonamento varchar(11) not null references Abbonamento(Tipo),
        Lezione int not null references Lezione(ID),
        primary key(TipoAbbonamento, Lezione)
)
```

```
Create Table LezioneSettimanale(
        IDLezione int not null references Lezione(ID),
        GiornoSettimana varchar(10) not null references OrariCorso(GiornoSettimana),
        Sede int not null references Sede(ID),
        primary key (IDLezione)
)

Create Table Seminario(
        IDLezione int not null references Lezione(ID),
        Orario timestamp not null,
        Luogo varchar(20) not null,
        primary key (IDLezione)
)
```

```
Create Table OrariCorso(
        GiornoSettimana varchar(10) not null primary key,
        Ora time not null
)



Create Table Sede(
        ID serial not null primary key,
        Nome varchar(26) not null,
        Citta varchar(8) not null,
        CAP int not null,
        Via varchar(30) not null
)
```

# Interrogazioni SQL

1. Trovare gli orari del corso.

   ```
   SELECT GiornoSettimana, Ora
   FROM OrariCorso
   ```

2. Visualizzare tutti i seminari programmati  compresi di luogo, data e gli argomenti che tratteranno.

   ```
   SELECT seminario.IDLezione, seminario.Luogo, seminario.Orario, lezione.Argomento
   FROM seminario JOIN lezione ON seminario.IDLezione = lezione.ID
   ```

3. Restituire solo la merce disponibile ordinata per prezzo.

   ```
   SELECT Merce.ID, Merce.prezzo
   FROM Merce
   WHERE Disponibilita > 0
   ```

```
ORDER BY Prezzo ASC
```

4. Trovare il nome della sede dove si svolge la lezione del martedì.

```
SELECT Sede.Nome
FROM LezioneSettimanale JOIN Sede ON LezioneSettimanale.Sede = Sede.ID
WHERE LezioneSettimanale.GiornoSettimana = 'Martedi'
```

5. Trovare il numero di lezioni incluse per tipo di abbonamento.

```
SELECT abbonamento.tipo, count(*) AS numero_lezioni
FROM abbonamento JOIN include ON abbonamento.tipo = include.tipoabbonamento
group by abbonamento.tipo
```

6. Trovare le informazioni identificative e il numero di abbonamenti sottoscritti dagli utenti. Nel caso in cui gli utenti non abbiano sottoscritto nessun abbonamento vengono comunque inseriti nei risultati.

```
SELECT utente.id, utente.username, utente.DataRegistrazione, count(sottoscrive.utente) AS
numero_abbonamenti
FROM utente LEFT JOIN sottoscrive ON sottoscrive.utente = utente.id
```

```
group by utente.id, utente.username, utente.DataRegistrazione
```

7. Visualizzare il costo dell'abbonamento annuale e il risparmio che si ottiene rispetto alla situazione in cui si sottoscrivono lo stesso numero di lezioni tramite rinnovo di abbonamenti mensili. Si ottenga il la sia in valore assoluto che in percentuale.

```
CREATE VIEW mensile_annuale (tipo_abbonamento, prezzo)
AS SELECT tipo, abbonamento.prezzo * 12
        FROM Abbonamento
        WHERE tipo = 'Mensile'

SELECT abbonamento.tipo, abbonamento.prezzo,
(mensile_annuale.prezzo - abbonamento.prezzo) AS sconto_annuale,
(100 - ((abbonamento.prezzo * 100)/mensile_annuale.prezzo)) AS percentuale
FROM Abbonamento, mensile_annuale
WHERE abbonamento.tipo = 'Annuale'
```