PROGETTO DavideGiovanni

1. ABSTRACT

Questo progetto punta a sviluppare una base di dati che aiuti la gestione di una palestra nei suoi aspetti manageriali. Verranno gestite in modo coerente e sistematico le informazioni dei dipendenti, in particolar modo dei dipendenti con maggiore attenzione agli istruttori, nonché delle sale, le attrezzature, i corsi e gli abbonamenti messi a disposizione. L'obiettivo di questo lavoro è di fornire un sistema che monitori le risorse a disposizione e che permetta ad abbonati e istruttori di accedere e gestire le prenotazioni ai corsi, verificando la validità degli abbonamenti presentati.
La base di dati proposta permette di assegnare uno o più abbonamenti ai suoi iscritti, ciascuno designato per un corso; corso al quale l'iscritto potrà accedere solo tramite previa prenotazione. Ciascun istruttore potrà essere assegnato ai propri corsi ed avrà la possibilità di gestire in autonomia l'inserimento e la gestione delle lezioni. Ogni corso verrà svolto in una sala che a sua volta ha a disposizione degli attrezzi, che resteranno occupati finché lo sarà anche la sala, cioè per tutta la durata della lezione.
Così facendo, la palestra avrà traccia di tutti i dati delle persone che la frequentano e degli abbonamenti validi e non. Inoltre verrà così ottimizzata la gestione delle lezioni, sapendo quante persone saranno presenti, non ci sarà rischio di avere sovraffolamento né di perdite di tempo da parte degli istruttori nel caso ci fossero poche presenze alla lezione.

2. ANALISI DEI REQUISITI

Abbonamento. Ogni abbonamento è identificato da un codice univoco e contiene le seguenti informazioni:

- UID, il quale distingue in modo univoco ciascun abbonamento
- Data inizio, la data in cui l'abbonamento è stato aperto
- Data fine, la data di termine dell'abbonamento
- Prezzo, l'importo pagato

  In particolare, l'abbonamento classico per l'accesso alla sola sala pesi non avrà alcun corso associato. Per poter accedere ai corsi ci sarà bisogno di un abbonamento apposito per ciascun corso che la persona vuole seguire.

Persona. Ciascuna persona viene riconosciuta tramite il proprio codice fiscale, univoco per definizione. Le informazioni immagazzinate sono le seguenti:

- Codice fiscale, identificativo univoco di una persona
- Nome
- Cognome
- Data di nascita
- Luogo di nascita

In particolare, il dipendente è un caso di persona al quale viene assegnato un ruolo all'interno della palestra:

- IBAN
- Livello

Ruolo. Il ruolo contiene le informazioni di base del lavoro che uno o più dipendenti dovranno svolgere:

- Codice ruolo
- Denominazione
- Stipendio base
- Ore

In particolare, un ruolo puó essere di due categorie:

- Staff, aggiunge informazione sull'area operativa
- Istruttore, specifica le certificazioni tecniche ed il livello di formazione richiesti per insegnare, corsi avanzati richiederanno istruttori con piú certificazioni ed un livello di formazione piú alto

Corso. Ogni corso è identificato da un codice univoco e contiene le seguenti informazioni:

- Codice Corso
- Denominazione
- Numero lezioni

Lezione. Ciascuna lezione è collegata ad un corso e contiene i dettagli che seguono:

- Codice Lezione, univoco
- Giorno
- Orario inizio
- Durata

Prenotazione. Per accedere ad una lezione è necessario essere iscritti al corso tramite abbonamento ed effettuare una prenotazione. Questa prenotazione potrà essere modificata o cancellata:

- Codice Lezione
- Data Registrazione

Sala. Ogni lezione verrà svolta in una predeterminata sala, in base anche alla necessità di specifici attrezzi e della sua capienza:

- Codice Sala
- Denominazione
- Capienza Max

Attrezzo. A ciascun attrezzo verrà assegnata una sala, vengono salvata le informazioni come:

- Codice Seriale, univoco
- Marca
- Anno Acquisto

Tipologia. Ogni attrezzo appartiene ad una determinata tipologia, la quale individua un insieme di attrezzi che allenano una serie di muscoli, è rappresentata da:

- UID, codice univoco tipologia
- Denominazione
- Muscoli interessati

3. PROGETTAZIONE CONCETTUALE

— Lista entità

Abbonamento)
- UID : Varchar (20)
- Prezzo : int
- Data_fine : date
- Data_inizio : date
- UID

Persona)
- CF : Varchar (16)
- Nome : Varchar (15)
- Cognome : Varchar (15)
- Data_nascita : date
- Luogo_nascita : Varchar (20)

Corso)
- Codice_corso : Varchar (10)
- Denominazione : Varchar (15)
- Capienza_massima : int

Attrezzo)
- Codice_serie : Varchar (7)
- Denominazione : Varchar (10)

Lezione)
- Codice_lezione : Varchar (10)
- Giorno : Date
- Durata : Time
- Orario inizio : Time

Prenotazione)
- Codice_lezione : Varchar (10)
- Data_registrazione : Date

Dipendente)

- Livello : int
- IBAN : Varchar (24)

— Lista relazioni

1) lezione - sala “svolta in”
- Una lezione si svolge in una sala (1:1)
- In una sala si possono svolgere piu lezioni (1:N)

2) prenotazione - lezione “accede”
- Una prenotazione accede a una lezione
- A una lezione accedono piu prenotazioni

3) persona - abbonamento “iscrizione”
- Una persona si iscrive tramite uno o più abbonamenti
- Un abbonamento è di una persona

4) abbonamento - corso “comprende”
- Un abbonamento è legato a un corso
- A un corso possono essere associati piu abbonamenti
-

5) ruolo - corso “istruttore”

6) sala - attrezzo “contiene”

7) attrezzo - tipologia “appartiene”

8) persona - prenotazione “registra”

9)

10)

4. PROGETTAZIONE LOGICA

- **Analisi delle rindondanze**

Facendo un’analisi dello schema E-R, osserviamo che l’attributo numero_lezioni dell’entità corso potrebbe risultare ridondante, in quanto è possibile calcolarlo dinamicamente come conteggio delle righe della tabella lezione che hanno come valore dell’attributo codice_corso uguale al codice del corso considerato.

In particolare, se nella tabella lezione ogni lezione memorizza il corso a cui appartiene tramite un attributo codice_corso, allora il numero totale di lezioni per ciascun corso può essere ricavato con una semplice query COUNT(*) filtrando per codice_corso.

Tuttavia, questa informazione potrebbe essere salvata direttamente nella tabella corso, come campo numero_lezioni, per ottimizzare le prestazioni delle letture, a scapito di una maggiore complessità in scrittura.

Operazioni considerate

Per valutare l'effettiva convenienza o meno di mantenere la ridondanza, analizziamo le operazioni più frequenti che coinvolgono questo dato:

- **Operazione 1 (250/settimana)**: inserimento di una nuova lezione
- **Operazione 2 (40/settimana)**: visualizzazione del numero di lezioni di un corso

Alle operazioni di scrittura viene assegnato un costo doppio rispetto a quelle di lettura.

Con ridondanza

Nel caso in cui si scelga di memorizzare numero_lezioni all'interno della tabella corso, ogni volta che si aggiunge una nuova lezione sarà necessario aggiornare anche il valore corrispondente nella tabella corso. La lettura del numero di lezioni sarà invece immediata e diretta.

Concetto	Costrutto	Operazione di accesso DB	Tipo	Operazioni/settima
lezione	Entità	1	S	250
corso	Entità	1	S	250
corso	Entità	1	L	40

Costo settimanale totale con ridondanza = (250 × 2) + (40 × 1) = **540 accessi**

Senza ridondanza

Nel caso in cui non venga mantenuta la ridondanza, non è necessario aggiornare nulla nella tabella corso quando si aggiunge una lezione. Tuttavia, ogni volta che si vuole conoscere il numero di lezioni per un determinato corso, è necessario contare dinamicamente le righe nella tabella lezione che lo riguardano.

Assumiamo che in media ogni corso sia composto da 5 lezioni. Quindi ogni lettura richiede circa 5 accessi alla tabella lezione.

Concetto	Costrutto	Operazione di accesso DB	Tipo	Operazioni/settima
lezione	Entità	1	S	250
lezione	Entità	5	L	40

Costo settimanale totale senza ridondanza = (250 × 2) + (40 × 5) = **700 accessi**

Confrontando i costi settimanali:

Configurazione	Scritture (peso 2)	Letture (peso 1)	Totale
Con ridondanza	250 × 2 = 500	40	**540**
Senza ridondanza	250 × 2 = 500	40 × 5 = 200	**700**

Constatiamo quindi che l'attributo numero_lezioni è logicamente ridondante, ma mantenerlo consente un risparmio consistente in termini di accessi totali, specialmente se le letture sono frequenti. Perciò, in questo caso, conviene mantenere la ridondanza, aggiornando il valore numero_lezioni ogni volta che si aggiunge o rimuove una lezione.

ELIMINAZIONE DELLE GENERALIZZAZIONI

Persona → Dipendente
La generalizzazione è stata eliminata creando due entità distinte: Persona e Dipendente. La tabella Dipendente è collegata alla tabella Persona tramite una relazione uno-a-uno (1:1) sul codice fiscale. In questo modo si evita l'uso di attributi nulli per le persone che non sono dipendenti.
Questo approccio permette di mantenere la separazione tra i dati anagrafici generici (Persona) e quelli legati all'impiego (Dipendente), migliorando la chiarezza del modello e riducendo le ridondanze.

Ruolo → Istruttore, Staff
Anche questa generalizzazione è stata eliminata mantenendo l'entità Ruolo e creando due entità figlie (Istruttore e Staff) collegate tramite relazione uno-a-uno al Ruolo.
In questo modo si evita di gestire attributi non applicabili ad alcuni tipi di ruolo, come ad esempio Area_operativa per un istruttore o Certificazioni_tecniche per uno staff.
È stato introdotto un vincolo logico per assicurare l'esclusività: ogni ruolo può appartenere o alla categoria Istruttore o a Staff, ma non entrambe.

PARTIZIONAMENTO/ACCORPAMENTO DI ENTITA E RELATIONSHIP

Gli attributi generici di Persona, come Nome, Cognome, Data_nascita, Luogo_nascita, sono stati accorpati nell'entità Persona, separata logicamente dall'entità Dipendente, che contiene solo informazioni relative all'impiego.

Analogamente, gli attributi comuni a Ruolo (come Ore e Stipendio_base) sono stati mantenuti nella tabella Ruolo, mentre quelli specifici per i sottotipi (Certificazioni_tecniche, Livello_formazione, Area_operativa) sono stati distribuiti nelle entità Istruttore e Staff.

Le generalizzazioni sono state partizionate logicamente in entità figlie, collegate alle rispettive entità madri con relazioni uno-a-uno, in linea con la prassi seguita per evitare valori nulli e mantenere integrità semantica.

SCELTA DI INDETIFICATORI PRIMARI

- Persona: Il codice fiscale (CF) è l'identificatore primario.

- Dipendente: L’identificatore primario è il codice fiscale, che funge anche da chiave esterna a Persona.

- Ruolo: Il codice (Codice_ruolo) identifica univocamente ciascun ruolo.

- Istruttore, Staff: Entrambe queste entità ereditano la chiave primaria da Ruolo, mantenendo Codice_ruolo come chiave esterna e primaria.

NOTE varie

```
 /* nota : vanno messe in ordine di eredità
nota ER : va aggiunto cf istruttore a entità “corso” */

CREATE TABLE Persona (
        CF VARCHAR(16) NOT NULL PRIMARY KEY,
        Nome VARCHAR(50) NOT NULL,
        Cognome VARCHAR(50) NOT NULL,
        Data_nascita DATE NOT NULL,
        Luogo_nascita VARCHAR(50) NOT NULL
 ) ;

CREATE TABLE Abbonamento (
        UID VARCHAR(20) NOT NULL PRIMARY KEY,
        prezzo INT NOT NULL,
        Data_fine DATE NOT NULL,
        Data_inizio DATE NOT NULL,
) ;

CREATE TABLE Dipendente (
        cf_dip VARCHAR(16) NOT NULL PRIMARY KEY,
        IBAN VARCHAR(27) NOT NULL,
        Livello INT NOT NULL,

        FOREIGN KEY (cf_dip) REFERENCES Persona(CF)
) ;

/*  foreign key è il vincolo integrità referenziale, riferito alla generalizzazione parziale di cui è
parte, eredita il codice fiscale */

CREATE TABLE Corso (
        Codice_corso VARCHAR(10) NOT NULL PRIMARY KEY,
        Denominazione VARCHAR(20) NOT NULL,
        Numero_lezioni INT(10) NOT NULL
) ;

CREATE TABLE Lezione (
        Codice_lezione VARCHAR(10) NOT NULL PRIMARY KEY,
        Durata INT(10) NOT NULL,
        Orario_inizio TIMESTAMP NOT NULL,
        Giorno DATE NOT NULL,

        FOREIGN KEY (Codice_lezione) REFERENCES Prenotazione (Codice_lezione)
        ) ;

CREATE TABLE Sala (

) ;
```