Basi di Dati

Progetto A.A. 2023/2024

PERSONAL TRAINER DIGITALE

0312790

Domenico Azzarito

Indice

1. Descrizione del Minimondo 2

2. Analisi dei Requisiti 4

3. Progettazione concettuale 7

4. Progettazione logica 11

5. Progettazione fisica 16

1. Descrizione del Minimondo

Si vuole realizzare un'applicazione per consentire, ai personal trainer di una palestra, di gestire le schede degli esercizi dei propri clienti.

Ciascun utente della palestra è associato ad un personal trainer di riferimento. Questi potrà redigere una scheda di allenamento personalizzata per ciascun utente, scegliendo gli esercizi e i macchinari da utilizzare in un insieme definito dal proprietario della palestra. Per ciascun esercizio, il personal trainer indicherà anche il numero di serie e ripetizioni dell'esercizio.

Periodicamente, quando viene redatta una nuova scheda di esercizi, la scheda precedente viene archiviata. Questa sarà ancora consultabile dagli utenti, ma essi potranno "interagire" solo con quella corrente.

Gli utenti della palestra possono infatti accedere all'applicazione e mostrare quali sono gli esercizi che devono svolgere in una sessione di allenamento. Il sistema mostrerà l'esercizio e la serie corrente, permettendo all'utente di contrassegnare un dato esercizio come completato. L'atleta ha la possibilità di saltare un esercizio, anche se cominciato, e passare al successivo.

I personal trainer possono generare un report che mostra, per tutti i clienti a loro assegnati, quanti sono gli allenamenti sostenuti in un intervallo temporale richiesto, qual è la percentuale di completamento delle schede di allenamento in ogni sessione di allenamento e quanto tempo è durato ciascun allenamento.

Analisi dei concetti rilevanti all'interno del minimondo:

Si vuole realizzare un'applicazione per consentire, ai personal trainer di una palestra, di gestire le schede degli esercizi dei propri clienti.
Ciascun utente della palestra è associato ad un personal trainer di riferimento. Questi potrà redigere una scheda di allenamento personalizzata per ciascun utente, scegliendo gli esercizi e i macchinari da utilizzare in un insieme definito dal proprietario della palestra. Per ciascun esercizio, il personal trainer indicherà anche il numero di serie e ripetizioni dell'esercizio.
Periodicamente, quando viene redatta una nuova scheda di esercizi, la scheda precedente viene archiviata. Questa sarà ancora consultabile dagli utenti, ma essi potranno "interagire" solo con quella corrente.
Gli utenti della palestra possono infatti accedere all'applicazione e mostrare quali sono gli esercizi che devono svolgere in una sessione di allenamento. Il sistema mostrerà l'esercizio e la serie corrente, permettendo all'utente di contrassegnare un dato esercizio come completato. L'atleta ha la possibilità di saltare un esercizio, anche se cominciato, e passare al successivo.
I personal trainer possono generare un report che mostra, per tutti i clienti a loro assegnati, quanti sono gli allenamenti sostenuti in un intervallo temporale richiesto, qual è la percentuale di completamento delle schede di allenamento in ogni sessione di allenamento e quanto tempo è durato ciascun allenamento.

A seguito di interviste, si sono ottenute le seguenti delucidazioni:

- L'intervallo di tempo, nella riga 16, si riferisce ad un intervallo di giorni.
- Si possono effettuare più sessioni di allenamento in un giorno.
- Gli esercizi e i macchinari sono stabiliti e inseriti dal proprietario.
- I clienti sono inseriti e associati al Personal Trainer dal proprietario.
- Il Personal Trainer crea una scheda di allenamento personalizzata per ciascun cliente e non può riusare la stessa per altri.
- Gli esercizi possono essere anche a corpo libero.
- Il proprietario può inserire più macchinari della stessa tipologia.
- Gli esercizi hanno un ordine prestabilito nella scheda di allenamento.
- Dopo aver saltato un esercizio non è possibile rieffettuarlo.
- La palestra si trova in un piccolo paese.

2. Analisi dei Requisiti

Identificazione dei termini ambigui e correzioni possibili

Linea	Termine	Nuovo termine	Motivo correzione
2, 7	Scheda di esercizi	Scheda di allenamento	Non c'è differenza semantica nel minimondo.
3,4,8,10	Utente	Cliente	Il termine utente, dove usato, si riferisce ai clienti della palestra e non ai personal trainer o il proprietario
5	Insieme	Insieme di macchinari ed esercizi	Il proprietario stabilisce un insieme di macchinari utilizzabili e gli esercizi eseguibili con o senza i macchinari presenti
8	Consultabile	Visibile	Specifico cosa si intende per "consultabile"
10	Mostrare	Visualizzare	"Mostrare" si riferisce all'applicazione e non ai clienti, che invece possono "visualizzare"
13	Atleta	Cliente	L'atleta, in questo contesto, coincide con il cliente
15, 17	Allenamento	Sessione di allenamento	Il sistema gestisce le istanze degli allenamenti, non il concetto generico di allenamento
16	Temporale	Di giorni	Si intende un intervallo di giorni

Specifica disambiguata

Si vuole realizzare un'applicazione per consentire, ai personal trainer di una palestra, di gestire le schede di allenamento dei propri clienti.

Ciascun cliente della palestra è associato ad un personal trainer di riferimento. Questi potrà redigere una scheda di allenamento personalizzata per ciascun cliente, scegliendo gli esercizi e i macchinari da utilizzare in un insieme di macchinari ed esercizi definito dal proprietario della palestra. Per ciascun esercizio, il personal trainer indicherà anche il numero di serie e ripetizioni dell'esercizio.

Periodicamente, quando viene redatta una nuova scheda di allenamento, la scheda precedente viene archiviata. Questa sarà ancora visibile ai clienti, ma essi potranno "interagire" solo con quella corrente.

I clienti della palestra possono infatti accedere all'applicazione e visualizzare quali sono gli esercizi che devono svolgere in una sessione di allenamento. Il sistema mostrerà l'esercizio e la serie corrente, permettendo al cliente di contrassegnare un dato esercizio come completato. Il cliente ha la possibilità di saltare un esercizio, anche se cominciato, e passare al successivo.

I personal trainer possono generare un report che mostra, per tutti i clienti a loro assegnati, quante sono le sessioni di allenamento sostenute in un intervallo di giorni richiesto, qual è la percentuale di completamento delle schede di allenamento in ogni sessione di allenamento e quanto tempo è durata ciascuna sessione di allenamento.

Glossario dei Termini

Termine	Descrizione	Sinonimi	Collegamenti
Cliente	È colui che si iscrive alla palestra ed effettua le sessioni di allenamento con una determinata scheda di allenamento	Utente, Atleta	Personal Trainer, Sessione di allenamento, Scheda di allenamento
Scheda di allenamento	Descrive gli esercizi da effettuare da un cliente. Può essere utilizzabile o archiviata	Scheda degli esercizi	Cliente, Esercizio
Sessione di allenamento	Descrive il momento in cui il cliente va in palestra per allenarsi	Allenamento	Cliente, Scheda di allenamento
Personal Trainer	È colui che crea le schede di allenamento e le assegna ai propri clienti		Cliente
Esercizio	Indica un esercizio fisico da compiere attraverso un eventuale macchinario		Macchinario, Scheda di allenamento
Macchinario	Indica un attrezzo presente in palestra, utilizzabile per eseguire esercizi fisici		Esercizio

Raggruppamento dei requisiti in insiemi omogenei

Frasi relative a Cliente
Ciascun cliente della palestra è associato ad un personal trainer di riferimento. Questi potrà redigere una scheda di allenamento personalizzata per ciascun cliente. [La scheda archiviata] sarà ancora visibile ai clienti, ma essi potranno “interagire” solo con quella corrente. I clienti della palestra possono infatti accedere all’applicazione e visualizzare quali sono gli esercizi che devono svolgere in una sessione di allenamento. Il cliente ha la possibilità di saltare un esercizio, anche se cominciato, e passare al successivo.

Frasi relative a Personal Trainer
Si vuole realizzare un’applicazione per consentire, ai personal trainer di una palestra, di gestire le schede di allenamento dei propri clienti. Ciascun cliente della palestra è associato ad un personal trainer di riferimento. [Il personal trainer] potrà redigere una scheda di allenamento personalizzata per ciascun cliente, scegliendo gli esercizi e i macchinari da utilizzare in un insieme di macchinari definito dal proprietario della palestra. Per ciascun esercizio, il personal trainer indicherà anche il numero di serie e ripetizioni dell’esercizio. I personal trainer possono generare un report che mostra, per tutti i clienti a loro assegnati, quanti sono le sessioni di allenamento sostenuti in un intervallo temporale richiesto, qual è la percentuale di

completamento delle schede di allenamento in ogni sessione di allenamento e quanto tempo è durato ciascuna sessione di allenamento.

Frasi relative a Sessione di allenamento

I clienti della palestra possono infatti accedere all'applicazione e visualizzare quali sono gli esercizi che devono svolgere in una sessione di allenamento.

I personal trainer possono generare un report che mostra, per tutti i clienti a loro assegnati, quanti sono le sessioni di allenamento sostenuti in un intervallo temporale richiesto, qual è la percentuale di completamento delle schede di allenamento in ogni sessione di allenamento e quanto tempo è durato ciascuna sessione di allenamento.

Frasi relative a Scheda di allenamento

Si vuole realizzare un'applicazione per consentire, ai personal trainer di una palestra, di gestire le schede di allenamento dei propri clienti.

[Il personal trainer] potrà redigere una scheda di allenamento […] scegliendo gli esercizi e i macchinari da utilizzare .

Periodicamente, quando viene redatta una nuova scheda di allenamento, la scheda precedente viene archiviata. Questa sarà ancora visibile ai clienti, ma essi potranno "interagire" solo con quella corrente.

I personal trainer possono generare un report che mostra […] qual è la percentuale di completamento delle schede di allenamento in ogni sessione di allenamento

Frasi relative a Esercizio

[Il personal trainer sceglie] gli esercizi e i macchinari da utilizzare. Per ciascun esercizio, il personal trainer indicherà anche il numero di serie e ripetizioni dell'esercizio.

I clienti della palestra possono infatti accedere all'applicazione e visualizzare quali sono gli esercizi che devono svolgere. Il sistema mostrerà l'esercizio e la serie corrente, permettendo al cliente di contrassegnare un dato esercizio come completato. Il cliente ha la possibilità di saltare un esercizio, anche se cominciato, e passare al successivo.

Frasi relative a Macchinario

[Il personal trainer sceglie] gli esercizi e i macchinari da utilizzare in un insieme di macchinari e di esercizi definito dal proprietario della palestra.

3. Progettazione concettuale

Costruzione dello schema E-R

Nella costruzione dello schema E-R sono partito da uno schema scheletro (Figura 1) ottenuto tramite i requisiti e il glossario dei termini precedentemente definiti.

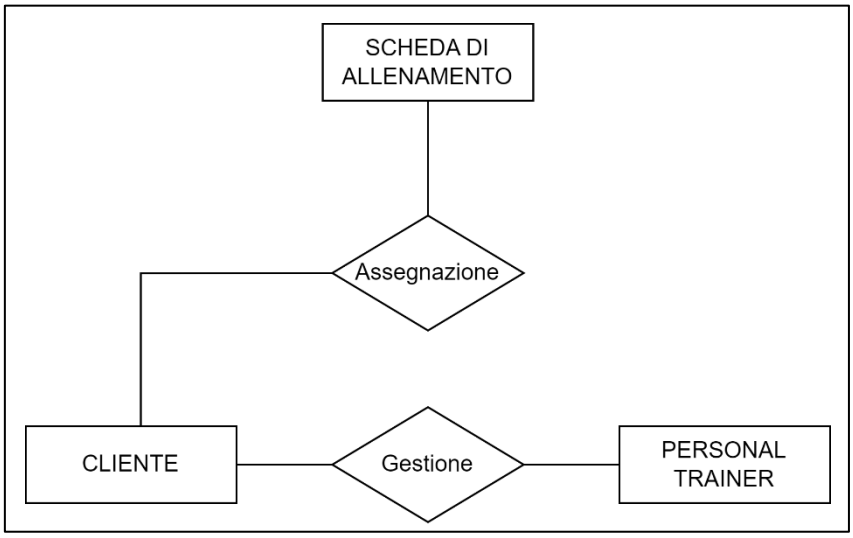

Figura 1: Schema Scheletro

Ho analizzato poi separatamente le specifiche per le tre entità rappresentate nello schema scheletro.
Per quanto riguarda l'entità **Personal Trainer,** ho inserito l'attributo *Codice Fiscale* per identificarla e gli attributi *Nome* e *Cognome* per renderlo riconoscibile.
Per quanto riguarda l'entità **Cliente**, ho inserito l'attributo *Codice Fiscale* per identificarla e gli attributi *Nome* e *Cognome* come per Personal Trainer. Per rappresentare il singolo allenamento effettuato dal cliente ho aggiunto l'entità **Sessione di allenamento**, avente gli attributi *Data*, *Ora Inizio* e *Ora Fine*, i quali rappresentano il giorno in cui si effettua l'allenamento e la sua ora di inizio e fine. Tale entità è identificata dal giorno in cui viene effettuato l'allenamento, dall'ora di inizio e dal cliente che lo effettua. Di conseguenza, specificando la relazione e le cardinalità si ottiene lo schema parziale in Figura 2.

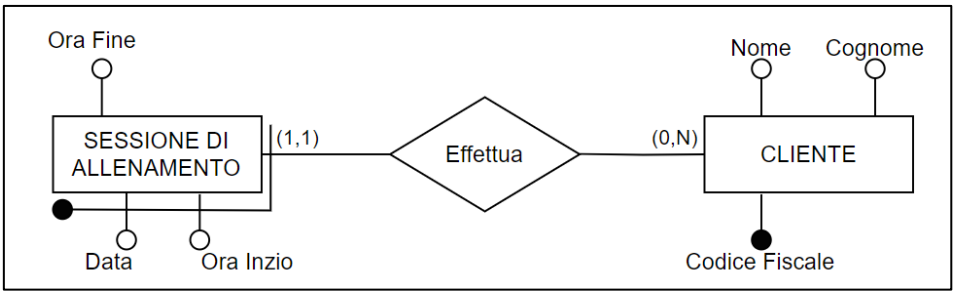

Figura 2: Raffinamento partendo dall'entità Cliente

Per quanto riguarda l'entità **Scheda di allenamento**, essa è composta da un elenco ordinato di esercizi eseguiti con i macchinari disponibili e forniti dal proprietario. Di conseguenza ho aggiunto le entità **Esercizio** e **Macchinario**. La prima ha l'attributo *Nome* che la identifica. Tale entità è legata alla Scheda di allenamento tramite una relazione che ha come attributi *Numero, Serie* e *Ripetizioni*, di cui il primo permette di creare un ordinamento all'interno della Scheda di allenamento mentre gli altri due si riferiscono al numero di serie da compiere e al numero di ripetizioni per ogni serie. Per quanto riguarda l'entità Macchinario, essa ha come attributi *Nome* e *Quantità* che sono rispettivamente il nome del macchinario e quanti ne sono presenti nella palestra. Il numero di macchinari viene conservato in quanto l'applicazione deve permettere al proprietario della palestra di inserire i macchinari posseduti e possono essere eventualmente inseriti più macchinari uguali. Ogni Sessione di allenamento è associata ad una Scheda di allenamento e ne conserva la percentuale di completamento, di conseguenza ho aggiunto l'entità **Sessione di allenamento** con l'attributo *Percentuale* sulla relazione che lega le due entità. Specificando le relazioni e le cardinalità si ottiene lo schema parziale in Figura 3.

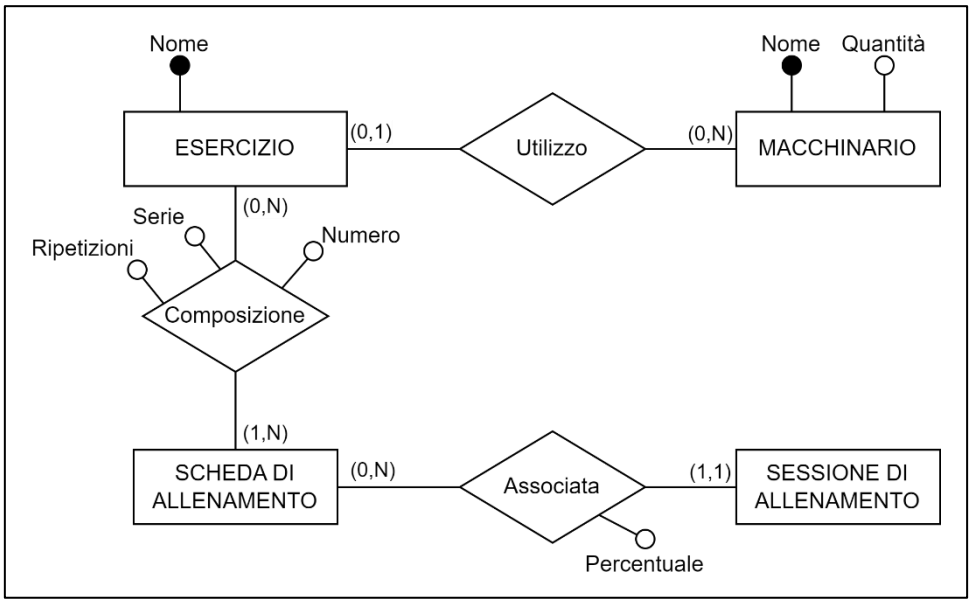

Figura 3: Raffinamento partendo dall'entità Scheda di allenamento

Ogni Scheda di allenamento può essere archiviata e dunque esiste un sottoinsieme di schede di allenamento archiviate e di schede correnti. Per rappresentarle ho usato una generalizzazione totale ed esclusiva tra l'entità padre Scheda di allenamento e le entità figlie **Scheda archiviata** e **Scheda corrente**, inserendo sulla prima l'attributo *Data Fine* e sul padre l'attributo *Data Inizio*.

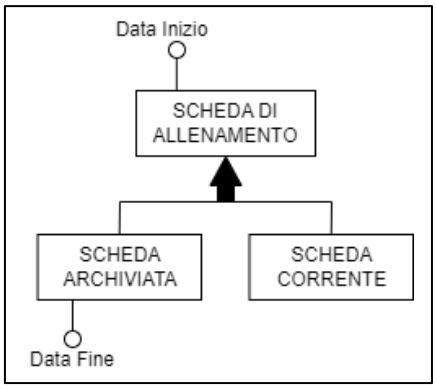

Figura 4: Storicizzazione dell'entità Scheda di allenamento

Integrazione finale

Integrando i vari schemi parziali, ho unito le entità omonime unendo gli attributi. Per quanti riguarda l'identificazione dell'entità **Scheda di allenamento**, ho inserito un identificatore esterno composto da *Data Inizio* e l'associazione Assegnazione. Aggiungendo le cardinalità nello schema in Figura 1 ho ottenuto lo schema finale in Figura 5.

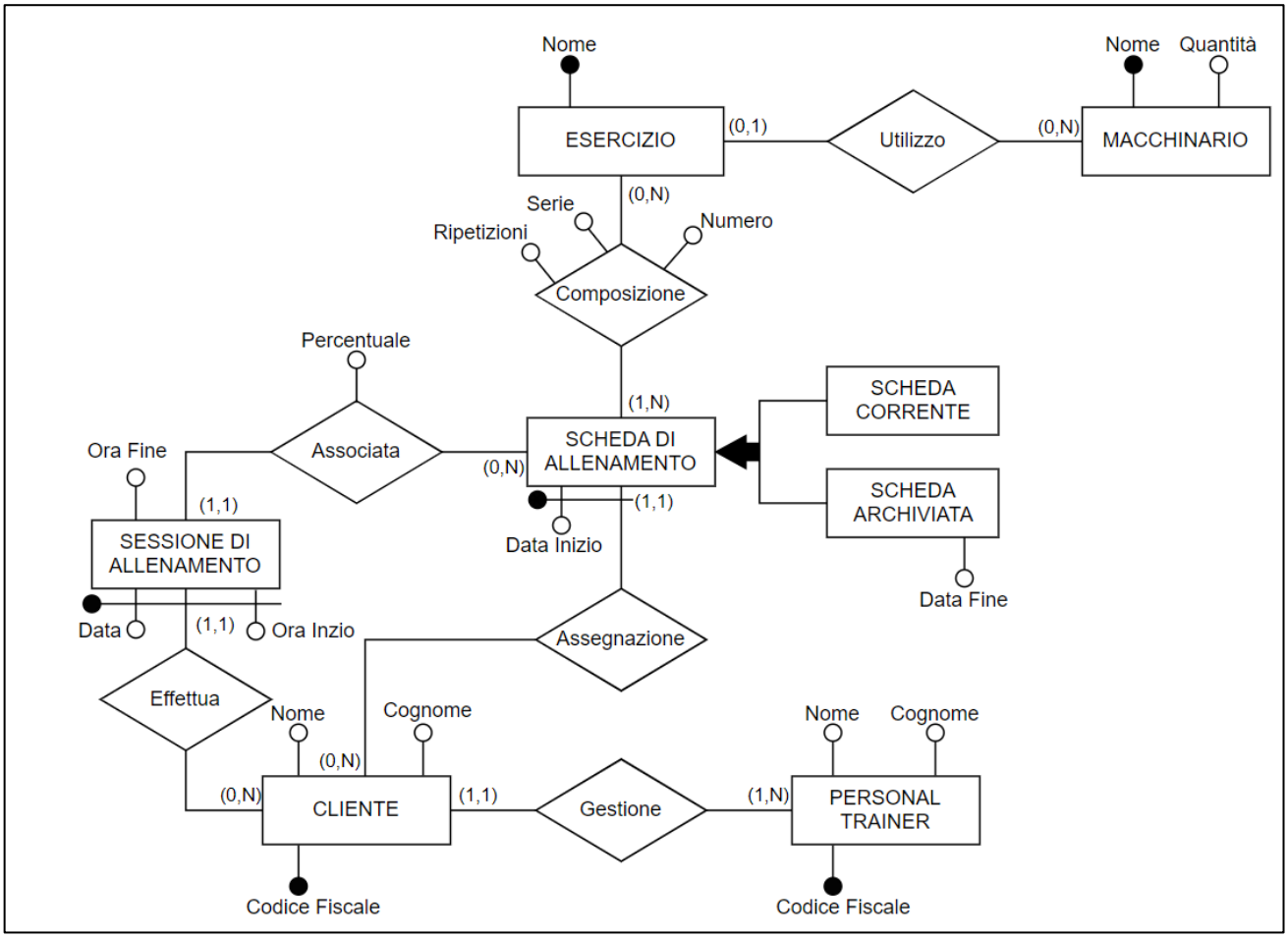

Figura 5: Schema finale

Regole aziendali

Un cliente non può effettuare una nuova sessione di allenamento con una scheda di allenamento archiviata.

Il personal trainer può assegnare una nuova scheda solo ai clienti a lui associati.

In una sessione di allenamento l'ora di inizio deve essere precedente all'ora di fine.

In una sessione di allenamento la percentuale deve essere maggiore di 0 e minore o uguale a 100 .

In una scheda archiviata la data di inizio deve essere precedente alla data di fine.

Il numero di macchinari deve essere maggiore di zero.

Il numero di serie e ripetizioni in Composizione devono essere maggiori di zero.

Dizionario dei dati

Entità	Descrizione	Attributi	Identificatori
Personal Trainer	Professionista deputato alla preparazione di allenamenti personalizzati per i propri clienti	Nome, Cognome	Codice Fiscale
Cliente	Colui che è iscritto alla palestra ed effettua gli allenamenti	Nome, Cognome	Codice Fiscale
Sessione di allenamento	L'effettuazione di un allenamento da parte del cliente	Ora Fine	Data, Ora Inizio, Effettua
Scheda di allenamento	Insieme di esercizi da compiere in una sessione di allenamento		Assegnazione, Data Inizio
Scheda corrente	L'ultima scheda di allenamento assegnata a un cliente.		Assegnazione, Data Inizio
Scheda archiviata	Scheda di allenamento archiviata dopo l'assegnazione di una nuova scheda	Data Fine	Assegnazione, Data Inizio
Esercizio	Esercizio fisico da compiere per allenarsi		Nome
Macchinario	Macchinario da usare in un esercizio fisico	Quantità	Nome

4. Progettazione logica

Volume dei dati

Nell'analisi, si ipotizza che il sistema mantenga i dati relativi:

- ai clienti per un periodo non superiore a cinque anni;
- alle schede di allenamento per un periodo non superiore ad un anno;
- alle sessioni di allenamento per un periodo non superiore a sei mesi;

Si considera che una scheda di allenamento sia composta in media da otto esercizi, i quali sono per l'80% associati a un macchinario. Il proprietario compra annualmente nuovi macchinari, dei quali il 70% sono macchinari già esistenti.

Si suppone che un cliente si alleni mediamente tre volte a settimana, riceva una nuova scheda ogni otto settimane e che soltanto l'80% dei clienti si alleni regolarmente. Inoltre, almeno una volta al mese i clienti visualizzano una scheda archiviata.

Si stabilisce che ogni personal trainer gestisca fino a un massimo di venticinque clienti e che le sue informazioni siano fisse nel tempo, data la locazione della palestra in un piccolo paese. Infine, si suppone che ogni personal trainer effettui almeno un report al mese per ciascun cliente, e che la maggior parte dei report riguardi gli allenamenti effettuati nell'ultimo mese.

Concetto nello schema	Tipo	Volume atteso
Personal Trainer	E	10
Cliente	E	200
Sessione di allenamento	E	15.600
Scheda di allenamento	E	1.400
Scheda corrente	E	200
Scheda archiviata	E	1.200
Esercizio	E	180
Macchinario	E	30
Gestione	R	200
Assegnazione	R	1400
Effettua	R	15.600
Associata	R	15.600
Composizione	R	11.200
Utilizzo	R	150

Tavola delle operazioni

Cod.	Descrizione	Frequenza attesa
CL1	Visualizzazione scheda corrente per eseguire sessione di allenamento	80/Giorno
CL2	Registrazione sessione di allenamento dopo averla eseguita	80/Giorno

CL3	Visualizzazione della lista delle schede archiviate in un intervallo di tempo dato	200/Mese
CL4	Visualizzazione di una scheda archiviata	200/Mese
P1	Inserimento esercizio	16/Anno
P2	Inserimento macchinario	8/Anno
P3	Visualizzazione lista macchinari	8/Anno
P4	Inserimento cliente	40/Anno
P5	Inserimento personal trainer	3/Anno
P6	Visualizzazione della lista dei personal trainer assunti con il numero di clienti gestiti da ciascuno	3/Anno
PT1	Creazione scheda allenamento	100/Mese
PT2	Report sessioni di allenamento	200/Mese
PT3	Lista dei clienti assegnati	100/Mese
PT4	Visualizzazione lista esercizi	48/Anno
L1	Login	90/Giorno
L2	Aggiorna Password	50/Anno

Costo delle operazioni

Si considera il costo di accesso in scrittura pari al doppio di quelli in lettura.

CL1 | Visualizzazione scheda corrente

- Cliente: 1/Lettura
- Assegnazione: 1/Lettura
- Scheda di allenamento: 1/Lettura
- Scheda corrente: 1/Lettura
- Composizione: 8/Lettura
- Esercizio: 8/Lettura
- Utilizzo: 6/Lettura
- Macchinario: 6/Lettura

Costo totale: 32

Accessi/Giorno: 2.560

CL2 | Registrazione sessione di allenamento

- Cliente: 1/Lettura
- Effettua: 1/Scrittura
- Sessione di allenamento: 1/Scrittura
- Associata: 1/Scrittura
- Scheda di allenamento: 1/Lettura

Costo totale: 8

Accessi/Giorno: 640

CL3 | Lista schede archiviate

- Cliente: 1/Lettura
- Assegnazione: 1/Lettura
- Scheda di allenamento: 1/Lettura
- Scheda archiviata: 1/Lettura

Costo totale: 4

Accessi/Mese: 800

CL4 | Visualizzazione scheda archiviata

- Cliente: 1/Lettura
- Assegnazione: 1/Lettura
- Scheda di allenamento: 1/Lettura
- Scheda archiviata: 1/Lettura
- Composizione: 8/Lettura
- Esercizio: 8/Lettura
- Utilizzo: 6/Lettura
- Macchinario: 6/Lettura

Costo totale: 32

Accessi/Mese: 6.400

P1 | Inserimento macchinario

- Macchinario: 1/Lettura
- Macchinario: 1/Scrittura

Costo totale: 3

Accessi/Anno: 24

P2 | Inserimento esercizio

Siccome l’80% degli esercizi è associato a un macchinario, si ha che l’80% delle esecuzioni di P2 avrà i seguenti accessi:

- Esercizio: 1/Scrittura
- Macchinario: 1/Lettura
- Utilizzo: 1/Scrittura

Costo totale: 5

Mentre il 20% il seguente accesso:

- Esercizio: 1/Scrittura

Costo totale: 2

Dunque, si ha:

Accessi/Anno: 112

P3 | Visualizzazione lista macchinari

- Macchinario: 1/Lettura

Costo totale: 1

Accessi//Anno: 3

P4 | Inserimento cliente

(Riferimento allo schema logico finale)

- Cliente: 1/Scrittura
- Utente: 1/Scrittura

Costo totale: 4

Accessi/Anno: 160

P5 | Inserimento personal trainer

(Riferimento allo schema logico finale)

- Personal Trainer: 1/Scrittura
- Utente: 1/Scrittura

Costo totale: 4

Accessi/Anno: 12

P6 | Lista personal trainer

- Personal Trainer: 1/Lettura
- Gestione: 20/Lettura
- Cliente: 20/Letture

Costo totale: 41

Accessi/Anno: 123/Anno

PT1 | Creazione scheda di allenamento

- Personal Trainer: 1/Lettura
- Cliente: 1/Lettura
- Scheda di allenamento: 1/Scrittura
- Scheda corrente: 1/Scrittura
- Scheda archiviata: 1/Scrittura
- Esercizio: 8/Letture
- Gestione: 1/Lettura
- Assegnazione: 1/Scrittura
- Composizione: 8/Scrittura

Costo totale: 35

Accessi/Mese: 3.500

PT2 | Report sessioni di allenamento

- Personal Trainer: 1/Lettura
- Cliente: 1/Lettura
- Sessioni di allenamento: 12/Lettura
- Scheda di allenamento: 12/Lettura
- Associata: 12/Lettura
- Gestione:1/Lettura
- Effettua: 12/Lettura

Costo totale: 51

Accesso/Mese: 10.200

PT3 | Lista clienti assegnati

- Personal Trainer: 1/Lettura
- Gestione: 20/Lettura
- Cliente: 20/Lettura

Costo totale: 43

Accessi/Mese: 4.300

PT4 | Lista esercizi

- Esercizio: 1/Lettura
- Utilizzo: 1/Lettura
- Macchinario: 1/Lettura

Costo totale: 3

Accessi//Anno: 144

L1 | Login

(Riferimento allo schema logico finale)

- Utente: 1/Lettura
- Cliente: 1/Lettura o Personal Trainer: 1/Lettura

Costo totale: 2

Accessi/Giorno: 180

L2 | Registra password

(Riferimento allo schema logico finale)

- Utente: 1/Scrittura

Costo totale: 2

Accessi/Anno: 100/

Ristrutturazione dello schema E-R

Per quanto riguarda l'unica generalizzazione presente si è scelto di accorpare le entità figlie **Scheda archiviata** e **Scheda corrente** nell'entità padre **Scheda di allenamento**, aggiungendo a quest'ultima l'attributo *Data Fine,* che può assumere valori nulli nel caso in cui la scheda sia corrente, e l'attributo *Tipo* per distinguere tra Corrente o Archiviata. In questo modo, a discapito di un piccolo numero di valori nulli, diminuisce il numero di accessi nelle operazioni CL1,CL3,CL4 e PT1, il cui costo diventa rispettivamente di 2.480/Giorno, 600/Mese, 6.200/Mese e 3.300/Mese. Si potrebbe anche non aggiungere l'attributo *Tipo* distinguendo tra chi ha un valore di *Data Fine* e chi valore nullo, ma ciò renderebbe il valore nullo indefinito.

Trasformazione di attributi e identificatori

Per quanto riguarda gli identificatori esterni, si ha l'entità Scheda di allenamento e l'entità Sessione di allenamento. La prima è identificata da Assegnazione e Data Inizio, il quale risulta un identificatore pesante che deve essere usato per rappresentare le associazioni Composizione e Associata con molte occorrenze. Di conseguenza risulta utile introdurre l'identificatore e attributo Codice che può essere presentato con non più di 2 byte, riducendo lo spreco di memoria. Riguardo l'entità Sessione di allenamento, l'identificatore composto da Ora Inizio, Data e l'associazione Effettua risulta rilevante nell'operazione PT2 e si è deciso quindi di mantenerlo.
Lo schema E-R da tradurre risulta essere il seguente:

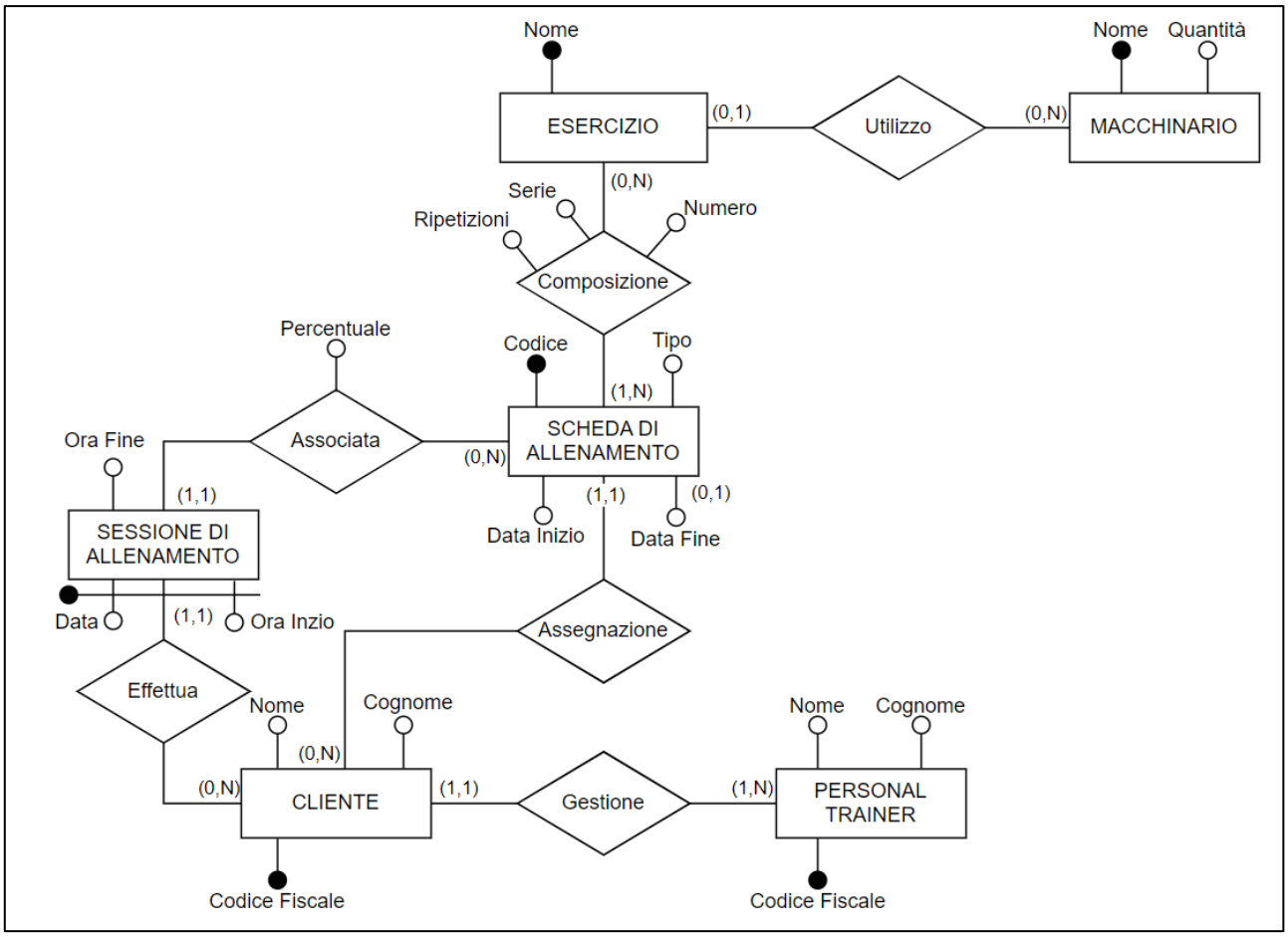

Figura 6: Schema E-R dopo la ristrutturazione

Traduzione di entità e associazioni

Per effettuare la traduzione dello schema E-R esegui i seguenti passaggi:

1. Trasformare le entità forti che partecipano con cardinalità massima pari a N a tutte le associazioni ad esse collegate:

   PersonalTrainer(Codice Fiscale, Nome, Cognome)

   Macchinario(Nome, Quantità)

2. Trasformare le entità forti che partecipano con cardinalità (1,1) ad almeno una delle associazioni ad esse collegate:

   Cliente(Codice Fiscale, Nome, Cognome, PersonalTrainer)

   SchedaDiAllenamento(Codice, DataInizio, DataFine*, Tipo, Cliente)

   con i seguenti vincoli di referenziazione:

   Cliente(Personal Trainer) ⊆ PersonalTrainer(Codice Fiscale)

   SchedaDiAllenamento(Cliente) ⊆ Cliente(Codice Fiscale)

3. Trasformare le entità forti che partecipano con cardinalità (0,1) ad almeno una delle associazioni ad esse collegate:

   Esercizio(Nome, Macchinario*)

   con i seguenti vincoli di referenziazione:

   Esercizio(Macchinario) ⊆ Macchinario(Nome)

   In questo caso, siccome la maggior parte degli esercizi è effettuata con i macchinari, ho scelto di includere la relazione Utilizzo in Esercizio invece di creare uno schema a sé; in questo modo si diminuisce di 960/Giorno il numero di accessi nell'operazione CL1 a discapito di circa 35 valori nulli attesi.
4. Trasformare le entità deboli:

   SessioneDiAllenamento(Cliente, Data, OraInizio, OraFine, Scheda, Percentuale)

   con i seguenti vincoli di referenziazione:

   SessioneDiAllenamento(Cliente) ⊆ Cliente(Codice Fiscale)

   SessioneDiAllenamento(Scheda) ⊆ SchedaDiAllenamento(Codice)
5. Trasformare tutte le associazioni rimaste:

   EserciziScheda(Scheda, Esercizio, Numero, Serie, Ripetizioni)

   con i seguenti vincoli di referenziazione:

   EserciziScheda (Scheda) ⊆ SchedaDiAllenamento(Codice)

   EserciziScheda (Esercizio) ⊆ Esercizio(Nome)

Si ottiene così lo schema logico in Figura 7.

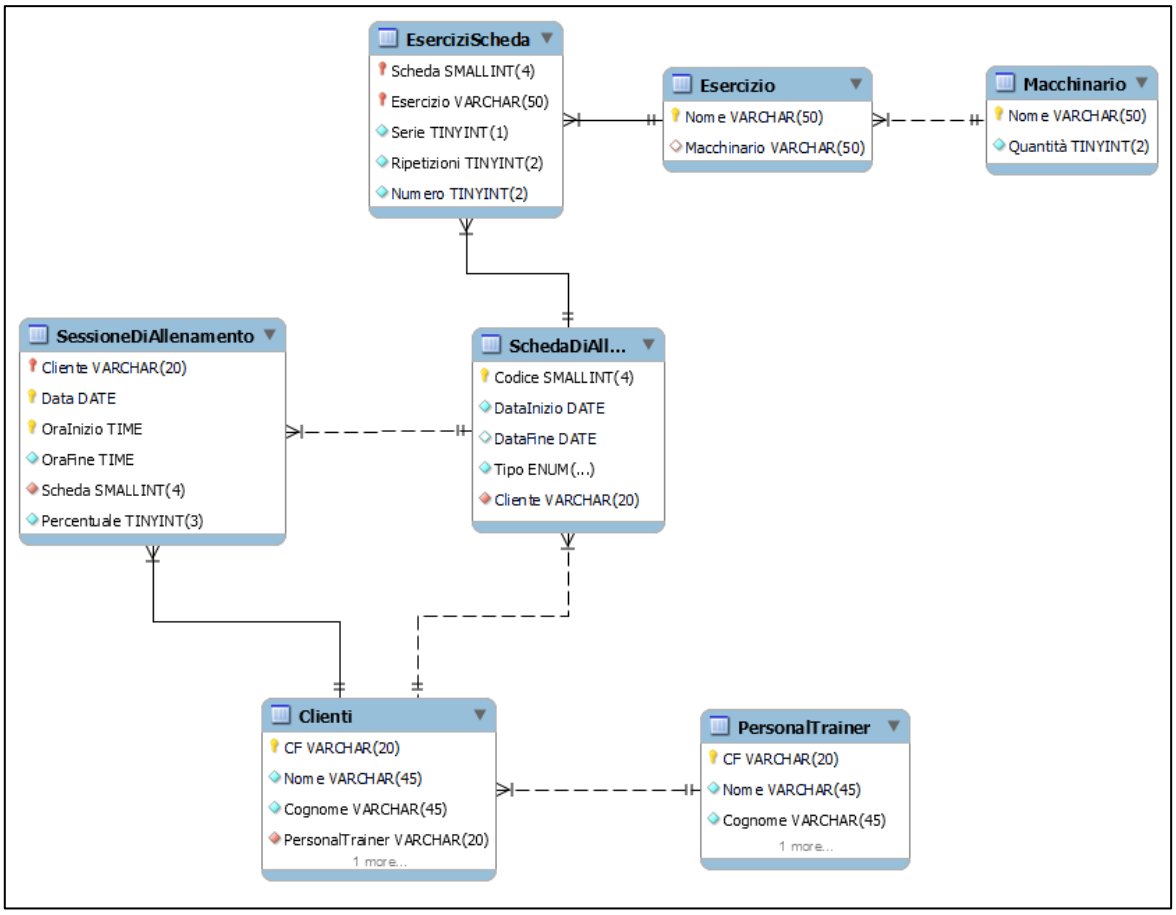

Figura 7: Schema logico della base di dati analizzata

Normalizzazione del modello relazionale

La base di dati risulta essere in terza forma normale perché per ogni relazione si ha che ogni dipendenza funzionale non banale $X \rightarrow Y$ verifica almeno una delle seguenti condizioni:

- X contiene almeno una chiave K di r
- ogni attributo di Y appartiene ad almeno una chiave di r

5. Progettazione fisica

Utenti e privilegi

Si prevedono tre ruoli, per implementare il Principle of Least Privilege:

- Login:

  Grant in esecuzione sull'operazione L1 e L2

- Cliente:

  Grant in esecuzione sulle operazioni CL1, CL2, CL3 e CL4

- Personal:

  Grant in esecuzione sulle operazioni PT1, PT2, PT3 e PT4

- Proprietario:

  Grant in esecuzione sulle operazioni P1, P2, P3, P4, P5 e P6

Per identificare gli utenti si introduce una tabella Utenti per mantenere le credenziali. Essa è costituita dagli attributi *Username*, *Password* e *Ruolo*. Per permettere il recupero delle informazioni del cliente o personal trainer che ha effettuato l'accesso si aggiunge l'attributo *Username* alle tabelle Clienti e PersonalTrainer con un vincolo di integrità referenziale alla tabella Utenti (Figura 8).

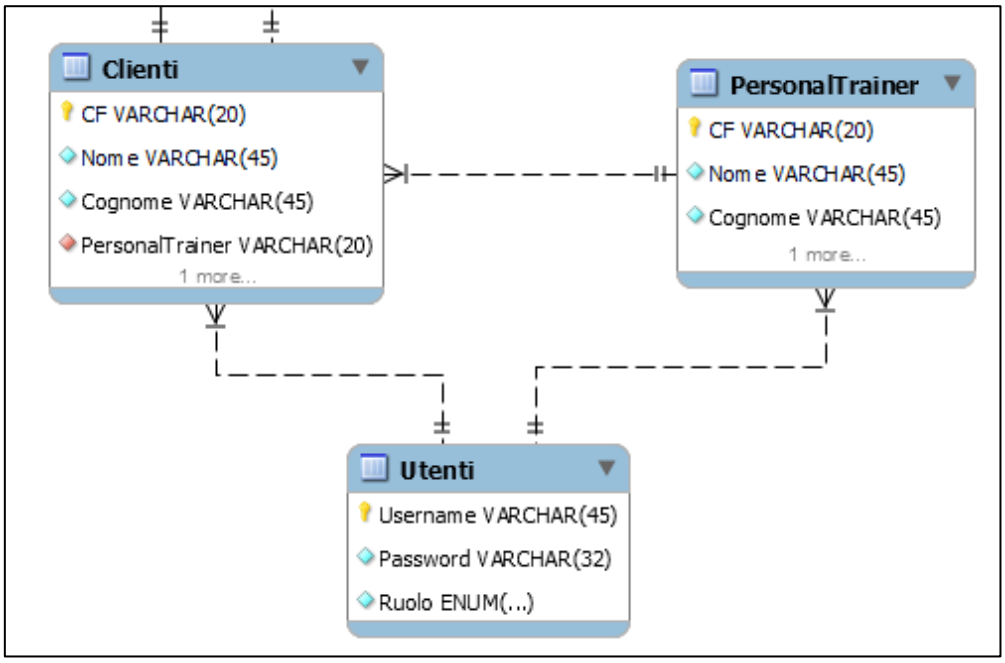

Figura 8: Porzione schema logico dopo l'introduzione della tabella Utenti

Strutture di memorizzazione

Tabella Utente		
Colonna	**Tipo di dato**	**Attributi**[1]
Username	VARCHAR(45)	PK, NN
Password	VARCHAR(32)	NN
Ruolo	ENUM(‘Cliente’, ‘PersonalTrainer’, ‘Proprietario’)	NN

Tabella Macchinario		
Colonna	**Tipo di dato**	**Attributi**
Nome	VARCHAR(50)	PK, NN
Quantità	UNSIGNED TINYINT(2)	NN

Tabella Esercizio		
Colonna	**Tipo di dato**	**Attributi**
Nome	VARCHAR(50)	PK, NN
Macchinario	VARCHAR(50)	

Tabella EserciziScheda		
Colonna	**Tipo di dato**	**Attributi**
Scheda	SMALLINT(4)	PK, NN
Esercizio	VARCHAR(50)	PK, NN
Numero	UNSIGNED TINYINT(2)	NN
Serie	UNSIGNED TINYINT(1)	NN
Ripetizioni	UNSIGNED TINYINT(2)	NN

Tabella SessioneDiAllenamento		
Colonna	**Tipo di dato**	**Attributi**
Cliente	VARCHAR(20)	PK, NN
Data	DATE	PK, NN
OraInizio	TIME	PK, NN
OraFine	TIME	NN
Scheda	SMALLINT(4)	NN
Percentuale	TINYINT(3)	NN

Tabella SchedaDiAllenamento		
Colonna	**Tipo di dato**	**Attributi**
Codice	SMALLINT(4)	PK, NN, AI
DataInizio	DATE	NN
DataFine	DATE	
Tipo	ENUM(‘Corrente’, ‘Archiviata’)	NN

[1] PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna.

Cliente	VARCHAR(20)	NN

Tabella Cliente		
Colonna	**Tipo di dato**	**Attributi**
Codice Fiscale	VARCHAR(20)	PK, NN, AI
Nome	VARCHAR(45)	NN
Cognome	VARCHAR(45)	NN
PersonalTrainer	VARCHAR(20)	NN
Username	VARCHAR(45)	NN

Tabella PersonalTrainer		
Colonna	**Tipo di dato**	**Attributi**
Codice Fiscale	VARCHAR(20)	PK, NN, AI
Nome	VARCHAR(45)	NN
Cognome	VARCHAR(45)	NN
Username	VARCHAR(45)	NN

Indici

Si è introdotto l'indice unique_giorno per assicurare che non vengano assegnate più schede in uno stesso giorno allo stesso cliente.

Tabella SchedaDiAllenamento	
Indice unique_giorno	**Tipo[2]:**
DataInizio, Cliente	UQ

Si è introdotto l'indice unique_ordine per assicurare che non ci siano ripetizioni del numero di un esercizio all'interno di una scheda nella tabella EserciziScheda.

Tabella EserciziScheda	
Indice unique_ordine	**Tipo[3]:**
Scheda, Numero	UQ

Si è introdotto l'indice intervallo_schede per migliorare le prestazioni dell'operazione CL3.

Tabella SchedaDiAllenamento	
Indice intervallo_schede	**Tipo:**
Cliente, DataInizio	IDX

[2] IDX = index, UQ = unique, FT = full text, PR = primary.

[3] IDX = index, UQ = unique, FT = full text, PR = primary.

Trigger

È stato implementato il seguente trigger per controllare che i nomi degli esercizi non contengano caratteri numerici o speciali.

```
CREATE DEFINER = CURRENT_USER TRIGGER
`palestra`.`Esercizio_BEFORE_INSERT` BEFORE INSERT ON `Esercizio` FOR EACH
ROW
BEGIN
   IF NOT NEW.Nome REGEXP  '^[a-zA-Z]*( [a-zA-Z]*)?$' THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Il nome contiene caratteri numerici o speciali.';
END IF;
END
```

È stato implementato il seguente trigger per controllare che i nomi dei macchinari non contengano caratteri numerici o speciali e il numero di macchinari sia sempre maggiore e diverso da zero.

```
CREATE DEFINER = CURRENT_USER TRIGGER
`palestra`.`Macchinario_BEFORE_INSERT` BEFORE INSERT ON `Macchinario` FOR
EACH ROW
BEGIN
   IF NOT NEW.Nome REGEXP  '^[a-zA-Z]*( [a-zA-Z]*)?$' THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Il nome contiene caratteri numerici o speciali.';
   ELSEIF NEW.Quantità <= 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'La quantità non è valida.';
   END IF;
END
```

I seguenti due trigger controllano che il CF e il Nome e Cognome dei clienti e personal trainer siano validi.

- ```
  CREATE DEFINER = CURRENT_USER TRIGGER
  `palestra`.`Cliente_BEFORE_INSERT` BEFORE INSERT ON `Cliente` FOR EACH ROW
  BEGIN
      IF NOT NEW.CF REGEXP  '^[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z] $'
  THEN
  ```
  ```

```
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Il CF non è valido.';
      ELSEIF NOT NEW.Nome REGEXP '^[a-zA-Z]*( [a-zA-Z]*)?$' OR NOT
    NEW.Cognome REGEXP '^[a-zA-Z]*( [a-zA-Z]*)?$' THEN
        SIGNAL SQLSTATE '4500'
        SET MESSAGE_TEXT = 'Il nome o il cognome contiene caratteri numerici o speciali.';
    END IF;
    END
```

- ```
  CREATE DEFINER = CURRENT_USER TRIGGER
  `palestra`.`PersonalTrainer_BEFORE_INSERT` BEFORE INSERT ON `PersonalTrainer`
  FOR EACH ROW
  BEGIN
    IF NOT NEW.CF REGEXP  '^[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z] $'
  THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Il CF non è valido.';
      ELSEIF NOT NEW.Nome REGEXP '^[a-zA-Z]*( [a-zA-Z]*)?$' OR NOT
  NEW.Cognome REGEXP '^[a-zA-Z]*( [a-zA-Z]*)?$' THEN
        SIGNAL SQLSTATE '4500'
        SET MESSAGE_TEXT = 'Il nome o il cognome contiene caratteri numerici o speciali.';
  END IF;
  END
  ```

Ho aggiunto il seguente trigger per far si che in una sessione di allenamento non si possa avere una percentuale maggiore del 100% o minore o uguale del 0% e che non si possa avere l'orario di inizio coincidente o minore con l'orario di fine.

```
CREATE DEFINER = CURRENT_USER TRIGGER
`palestra`.`SessioneDiAllenamento_BEFORE_INSERT` BEFORE INSERT ON
`SessioneDiAllenamento` FOR EACH ROW
BEGIN
  IF NEW.Percentuale > 100 OR NEW.Percentuale <= 0 THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'La percentuale non è valida.';
  ELSEIF NEW.OraFine < = NEW.OraInizio THEN
    SIGNAL SQLSTATE '45000'
```
```

```
    SET MESSAGE_TEXT = 'Ora di fine non può essere minore di ora di inizio.';
    END IF;
END
```

Ho aggiunto il seguente trigger per far si che il numero dell'esercizio e il numero di serie e ripetizioni in EserciziScheda sia sempre maggiore e diverso da zero.

```
CREATE DEFINER = CURRENT_USER TRIGGER
`palestra`.`EserciziScheda_BEFORE_INSERT` BEFORE INSERT ON `EserciziScheda`
FOR EACH ROW
BEGIN
        IF NEW.Numero <= 0 OR New.Serie <=0 OR New.Ripetizioni <=0 THEN
    SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Una quantità tra numero, serie e ripetizione non è
valida.';
        END IF;
END
```

Eventi

Sono stati utilizzati due eventi per ripulire il database in modo regolare dalle sessioni e le schede di allenamento non più consultate. Si hanno così i seguenti eventi:

- **rimozione_sessioni_vecchie** elimina le sessioni di allenamento eseguite più di sei mesi fa rispetto la data corrente. Tale evento viene ripetuto ogni mese a partire dal sesto mese successivo all'inserimento dell'evento nella base di dati, in quanto nei primi sei mesi non si avranno sessioni da eliminare. Il codice è il seguente:

```
CREATE EVENT IF NOT EXISTS rimozione_sessioni_vecchie
ON SCHEDULE EVERY 1 MONTH
STARTS CURRENT_TIMESTAMP + INTERVAL 6 MONTH
ON COMPLETION PRESERVE
COMMENT 'Rimozione vecchie sessioni di allenamento'
DO
BEGIN
        DELETE FROM SessioneDiAllenamento
        WHERE 'Data' < DATE_SUB(NOW(), INTERVAL 6 MONTH);
END
```

- **rimozione_schede_vecchie** elimina le schede di allenamento archiviate assegnate più di un anno fa rispetto la data corrente. Tale evento viene ripetuto ogni mese a partire dall'anno successivo all'inserimento dell'evento nella base di dati, in quanto nei primi dodici mesi non si avranno schede archiviate da eliminare. Il codice è il seguente:

```
CREATE EVENT IF NOT EXISTS rimozione_schede_vecchie
ON SCHEDULE EVERY 1 MONTH
STARTS CURRENT_TIMESTAMP + INTERVAL 12 MONTH
ON COMPLETION PRESERVE
COMMENT 'Rimozione vecchie schede di allenamento'
DO
BEGIN
        DELETE FROM SchedaDiAllenamento
        WHERE 'DataInizio' < DATE_SUB(NOW(), INTERVAL 12 MONTH)
                AND 'Tipo' = 'Archiviata';
END
```

Per rendere gli eventi utilizzabili bisogna prima eseguire la seguente riga:

```
SET GLOBAL event_scheduler = ON;
```

Viste

Non sono state utilizzate viste.

Stored Procedures e transazioni

Operazione CL1

```
CREATE PROCEDURE `scheda_corrente` (in var_cliente varchar(20), out var_scheda smallint(4))
BEGIN
   declare exit handler for sqlexception
 begin
   rollback;
   resignal;
 end;

 set transaction isolation level read committed;
 set transaction read only;
 start transaction;

   select Codice
   from SchedaDiAllenamento
```

```
   where Cliente = var_cliente and Tipo = 'Corrente'
   into var_scheda;

   select Numero, Esercizio, Serie, Ripetizioni, Macchinario
   from EserciziScheda join Esercizio on Esercizio.Nome = EserciziScheda.Esercizio
   where EserciziScheda.Scheda = var_scheda
   order by Numero;

commit;
END
```

Operazione CL2

```
CREATE PROCEDURE `registra_sessione` (in var_cliente varchar(20), in var_data date, in var_ora_inizio time, in var_ora_fine time, in var_scheda smallint(4), percentuale tinyint(3))
BEGIN
  insert into SessioneDiAllenamento (Cliente, Data, OraInizio, OraFine, Scheda, Percentuale) values (var_cliente, var_data, var_ora_inizio, var_ora_fine, var_scheda, percentuale);

END
```

Operazione CL3

```
CREATE PROCEDURE `lista_archiviate` (in var_cliente varchar(20), in var_data_inizio date, in var_data_fine date)
BEGIN

   declare exit handler for sqlexception
  begin
   rollback;
   resignal;
  end;

  set transaction isolation level read committed;
  set transaction read only;
  start transaction;

   select Codice, DataInizio, DataFine
   from SchedaDiAllenamento
    where Cliente = var_cliente and Tipo = 'Archiviata' and DataInizio >= var_data_inizio and DataInizio<= var_data_fine;

commit;
END
```

Operazione CL4

```
CREATE PROCEDURE `scheda_archiviata` (in var_cliente varchar(20), in var_scheda smallint(4))
BEGIN
   declare var_count int;
```

```
  declare exit handler for sqlexception
 begin
  rollback;
  resignal;
 end;

 set transaction isolation level read committed;
 set transaction read only;
 start transaction;

  select count(*)
  from SchedaDiAllenamento
  where Codice = var_scheda and Tipo = 'Archiviata' and Cliente = var_cliente
  into var_count;

  if var_count = 0 then
   signal sqlstate '45000'
   set message_text = 'La scheda non esiste o non è archiviata.';
  else
   select Numero, Esercizio, Serie, Ripetizioni, Macchinario
   from EserciziScheda join Esercizio on Esercizio.Nome = EserciziScheda.Esercizio
   where EserciziScheda.Scheda = var_scheda
   order by Numero;
  end if;

commit;
END
```

Operazione PT1

```
CREATE PROCEDURE `inserisci_scheda` (in var_personal varchar(20), in var_cliente varchar(20),
in var_len INT, in var_esercizi JSON)
BEGIN
      declare var_count int;
      declare var_scheda smallint(4);

      declare i int DEFAULT 0;
      declare numero tinyint(2);
      declare nome varchar(50);
      declare serie tinyint(1);
      declare ripetizioni tinyint(2);

      declare exit handler for sqlexception
  begin
      rollback;
      resignal;
  end;

  set transaction isolation level read committed;
```

```
   start transaction;

   select count(*)
   from Clienti
   where CF = var_cliente and PersonalTrainer = var_personal
   into var_count;

   if var_count = 0 then
      signal sqlstate '45000'
      set message_text = 'Il cliente non esiste o non è da lei gestito.';
   end if;

   update SchedaDiAllenamento
   set Tipo = 'Archiviata', DataFine = curdate()
   where Cliente = var_cliente and Tipo = 'Corrente';

      insert into SchedaDiAllenamento (Cliente,DataInizio,Tipo) values (var_cliente, curdate(), 'Corrente');

  set var_scheda=last_insert_id();

   while i < var_len do
         set numero = JSON_EXTRACT(var_esercizi, CONCAT('$[', i, '].numero'));
         set nome = JSON_UNQUOTE(JSON_EXTRACT(var_esercizi, CONCAT('$[', i, '].nome')));
         set serie = JSON_EXTRACT(var_esercizi, CONCAT('$[', i, '].serie'));
         set ripetizioni = JSON_EXTRACT(var_esercizi, CONCAT('$[', i, '].ripetizioni'));

         insert into EserciziScheda(Scheda, Esercizio, Numero, Serie, Ripetizioni) values (var_scheda, nome, numero, serie, ripetizioni);

         set i = i + 1;
   end while;

commit;
END
```

Operazione PT2

```
CREATE PROCEDURE `report_sessioni` (in var_personal varchar(20), in var_data_inizio date, in var_data_fine date)
BEGIN

      declare var_count int;

   declare exit handler for sqlexception
  begin
   rollback;
   resignal;
  end;
```

```
 set transaction isolation level read committed;
 set transaction read only;
 start transaction;

        select Nome, Cognome, Data, OraInizio, timediff(OraFine,OraInizio) as Durata, Scheda, Percentuale
      from SessioneDiAllenamento join Clienti on CF = Cliente
        where Cliente = var_personal  and PersonalTrainer = var_personal and Data between var_data_inizio and var_data_fine;

commit;

        if not found_rows() then
                signal sqlstate '45000'
                set message_text = 'Nessuna sessione di allenamento trovata.';
        end if;

END
```

Operazione PT3

```
CREATE PROCEDURE `lista_clienti` (in var_personal varchar(20))
BEGIN

                declare exit handler for sqlexception
        begin
                rollback;
                resignal;
        end;

        set transaction isolation level read committed;
        set transaction read only;
        start transaction;

   select CF, Nome, Cognome
   from Clienti
   where PersonalTrainer = var_personal;

commit;
END
```

Operazione PT4

```
CREATE PROCEDURE `lista_esercizi` ()
BEGIN

        declare exit handler for sqlexception
        begin
                rollback;
```

```
            resignal;
        end;

        set transaction isolation level read committed;
        set transaction read only;
        start transaction;

            select *
            from Esercizio;

        commit;
END
```

Operazione P1

```
CREATE PROCEDURE `inserisci_esercizio` (in var_nome varchar(50), in var_macchinario varchar(50))
BEGIN

  insert into Esercizio (Nome, Macchinario) values (var_nome, var_macchinario);

END
```

Operazione P2

```
CREATE PROCEDURE `inserisci_macchinario` (in var_nome varchar(50), in var_quantità tinyint(2))
BEGIN
        declare exit handler for sqlexception
        begin
            rollback;
            resignal;
        end;

        set transaction isolation level read committed;
        start transaction;

        if exists (select * from Macchinario where Nome = var_nome) then
            update Macchinario set Quantità = Quantità + var_quantità where Nome = var_nome;
        else
            insert into Macchinario (Nome, Quantità) values (var_nome, var_quantità);
        end if;

        commit;
END
```

Operazione P3

```
CREATE PROCEDURE `lista_macchinari` ()
BEGIN
```

```
        declare exit handler for sqlexception
        begin
                rollback;
                resignal;
        end;

        set transaction isolation level read committed;
        set transaction read only;
        start transaction;

        select *
        from Macchinario;

        commit;
END
```

Operazione P4

```
CREATE PROCEDURE `nuovo_cliente` (in var_cliente VARCHAR(20), in var_nome VARCHAR(45),in var_cognome VARCHAR(45), in var_personal VARCHAR(20), in var_nome_utente VARCHAR(45))
BEGIN

   declare exit handler for sqlexception

   begin
        rollback;
        resignal;
    end;

set transaction isolation level read uncommitted;
start transaction;

   insert into Utenti (Username, Password, Ruolo) values (var_nome_utente, md5('Nuova'), 'Cliente');

   insert into Clienti (CF, Nome, Cognome, PersonalTrainer, Username) values (var_cliente, var_nome, var_cognome,var_personal, var_nome_utente);

commit;
END;
```

Operazione P5

```
CREATE PROCEDURE `nuovo_personal` (in var_personal VARCHAR(20), in var_nome VARCHAR(45),in var_cognome VARCHAR(45), in var_nome_utente VARCHAR(45))
BEGIN
   declare exit handler for sqlexception

   begin
```

```
            rollback;
            resignal;
        end;

        set transaction isolation level read uncommitted;
        start transaction;

    insert into Utenti (Username, Password, Ruolo) values (var_nome_utente, md5('Nuova'), 'PersonalTrainer');

    insert into PersonalTrainer (CF, Nome, Cognome, Username) values (var_personal, var_nome, var_cognome, var_nome_utente);

commit;
END;
```

Operazione P6

```
CREATE PROCEDURE `lista_personal` ()
BEGIN

        declare exit handler for sqlexception
            begin
            rollback;
            resignal;
        end;

        set transaction isolation level read committed;
        set transaction read only;
        start transaction;

            select PersonalTrainer.CF, PersonalTrainer.Nome, PersonalTrainer.Cognome, Count(Clienti.CF) as NumClienti
              from PersonalTrainer left join Clienti on PersonalTrainer.CF = Clienti.PersonalTrainer
                group by PersonalTrainer.CF;

        commit;

END
```

Operazione L1

```
CREATE PROCEDURE `login` (in var_username varchar(45), in var_pass varchar(45), out var_role INT, out var_CF varchar(20))
BEGIN
        declare var_user_role ENUM('Cliente', 'PersonalTrainer', 'Proprietario');

        declare exit handler for sqlexception
        begin
            rollback;
```

```
            resignal;
        end;

        set transaction isolation level read committed;
        set transaction read only;
        start transaction;

                    select `Ruolo` from `Utenti`
                    where `Username` = var_username and `Password` = md5(var_pass)
                    into var_user_role;

                    if var_user_role = 'Cliente' then
                        set var_role = 1;
                        select `CF` from `Clienti`
                        where `Username` = var_username
                        into var_CF;
                    elseif var_user_role = 'PersonalTrainer' then
                        set var_role = 2;
                        select `CF` from `PersonalTrainer`
                        where `Username` = var_username
                        into var_CF;
                    elseif var_user_role = 'Proprietario' then
                        set var_role = 3;
                    else
                        set var_role = 0;
                    end if;
        commit;
END
```

Operazione L2

```
CREATE PROCEDURE `update_password` (in var_username VARCHAR(45), in var_password
VARCHAR(32))
BEGIN
      if strcmp(var_password, 'Nuova') != 0 then
      update Utenti
      set Password = md5(var_password)
      where Username = var_username;
   elseif strcmp(var_password, 'Nuova') = 0 then
            signal sqlstate '45000'
      set message_text = 'Password non valida';
   end if;
END;
```