

Università di Pisa

Pettorali Marco, Sabatini Gabriele

Corso di Basi di Dati

A.A. 2016-17

DOCUMENTAZIONE PROGETTO

Progettazione di un database per la gestione di un’impresa di fitness

Sommario

[1- Analisi delle specifiche 4](#_Toc493493991)

[1.1- Descrizione del problema 4](#_Toc493493992)

[1.2- Operazioni significative 4](#_Toc493493993)

[1.3- Glossario 5](#_Toc493493994)

[2- Progettazione concettuale 8](#_Toc493493995)

[2.1- Strategia concettuale 8](#_Toc493493996)

[2.2- Fasi di sviluppo del diagramma ER 8](#_Toc493493997)

[2.2.1- Sviluppo inside-out dell’entità “Centro” 9](#_Toc493493998)

[2.2.2- Sviluppo inside-out dell’entità “Dipendente” 10](#_Toc493493999)

[2.2.3- Sviluppo inside-out dell’entità “Cliente” 10](#_Toc493494000)

[2.2.4- Sviluppo inside-out dell’entità “Contratto” 10](#_Toc493494001)

[2.3- Ristrutturazione del diagramma ER 10](#_Toc493494002)

[2.3.1- Eliminazione della generalizzazione “Dipendente” 10](#_Toc493494003)

[2.3.2- Eliminazione della generalizzazione “Luogo\_allenamento” 10](#_Toc493494004)

[2.3.3- Eliminazione della generalizzazione “Post” 10](#_Toc493494005)

[3- Operazioni sui dati 10](#_Toc493494006)

[3.1- Compilazione tavole dei volumi 10](#_Toc493494007)

[3.1.1- Entità 10](#_Toc493494008)

[3.1.2- Relazioni 10](#_Toc493494009)

[3.1.3- Frequenza giornaliera delle operazioni significative 10](#_Toc493494010)

[3.2- Operazioni 10](#_Toc493494011)

[3.2.1- Operazione 1 10](#_Toc493494012)

[3.2.2- Operazione 2 10](#_Toc493494013)

[3.2.3- Operazione 3 10](#_Toc493494014)

[3.2.4- Operazione 4 10](#_Toc493494015)

[3.2.5- Operazione 5 10](#_Toc493494016)

[3.2.6- Operazione 6 10](#_Toc493494017)

[3.2.7- Operazione 7 10](#_Toc493494018)

[3.2.8- Operazione 8 10](#_Toc493494019)

[3.2.9- Operazione 9 10](#_Toc493494020)

[3.3- Introduzione di ridondanze 10](#_Toc493494021)

[3.3.1- Ridondanza 1: ClientiPresenti 10](#_Toc493494022)

[3.3.2- Ridondanza 2: VisiteMedico 10](#_Toc493494023)

[3.3.3- Ridondanza 3 : TotVoto, TotDurata, TotEsecuzioni 10](#_Toc493494024)

[4- Progettazione logica 10](#_Toc493494025)

[4.1- Traduzione del modello concettuale in modello logico 10](#_Toc493494026)

[4.2- Tabelle: 10](#_Toc493494027)

[4.2- Schema del database 10](#_Toc493494028)

[4.3- Vincoli di integrità referenziale 10](#_Toc493494029)

[4.4- Vincoli di integrità generici 10](#_Toc493494030)

[5- Implementazione su DBMS Oracle MySQL 10](#_Toc493494031)

[5.1- Definizione schema 10](#_Toc493494032)

[5.2- Definizione tabelle 10](#_Toc493494033)

[5.3- Implementazione delle operazioni significative 10](#_Toc493494034)

[5.3.1- Operazione 1 10](#_Toc493494035)

[5.3.2- Operazione 2 10](#_Toc493494036)

[5.3.3- Operazione 3 10](#_Toc493494037)

[5.3.4- Operazione 4 10](#_Toc493494038)

[5.3.5- Operazione 5 10](#_Toc493494039)

[5.3.6- Operazione 6 10](#_Toc493494040)

[5.3.7- Operazione 7 10](#_Toc493494041)

[5.3.8- Operazione 8 10](#_Toc493494042)

[5.3.9- Operazione 9 10](#_Toc493494043)

[5.4- Implementazione delle operazioni di aggiornamento delle ridondanze 10](#_Toc493494044)

[5.4.1- Ridondanza 1 10](#_Toc493494045)

[5.4.2- Ridondanza 2 10](#_Toc493494046)

[5.4.3- Ridondanza 3 10](#_Toc493494047)

[5.5- Implementazione delle funzionalità lato server 10](#_Toc493494048)

[5.5.1- Reporting 10](#_Toc493494049)

[5.5.2- Performance sportiva 10](#_Toc493494050)

[5.5.3- Rotazione del magazzino 10](#_Toc493494051)

[5.6- Implementazione vincoli di integrità generici 10](#_Toc493494052)

[5.6.1- Vincolo di integrità generico 1 10](#_Toc493494053)

[5.6.2- Vincolo di integrità generico 2 10](#_Toc493494054)

[5.6.3- Vincolo di integrità generico 3 10](#_Toc493494055)

[5.6.4- Vincolo di integrità generico 4 10](#_Toc493494056)

[5.6.5- Vincolo di integrità generico 5 10](#_Toc493494057)

[5.6.6- Vincolo di integrità generico 6 10](#_Toc493494058)

[5.6.7- Vincolo di integrità generico 7 10](#_Toc493494059)

[5.6.8- Vincolo di integrità generico 8 10](#_Toc493494060)

# Analisi delle specifiche

## 1.1- Descrizione del problema

Si vuole progettare un database che raccolga e gestisca i dati di un’impresa di fitness che opera a livello nazionale.

Il database deve contenere i dati relativi alla gestione di clienti e dipendenti, dei centri fitness e dei corsi che vi vengono organizzati, dei rapporti con fornitori e istituti finanziari esterni.

L’impresa dispone di un social network aperto ai suoi clienti di cui si devono gestire le funzionalità di pubblicazione di post, amicizia e il meccanismo delle “sfide”.

Si vuole, infine, implementare alcune funzionalità di reporting atte al miglioramento dei servizi offerti dall’impresa.

## 1.2- Operazioni significative

Si vogliono automatizzare le seguenti nove operazioni:

|  |  |
| --- | --- |
| Operazione |  |
| 1 | Inserimento di una scheda di allenamento e aggiunta di un esercizio |
| 2 | Elencare i partecipanti a un determinato corso |
| 3 | Indicare tutti gli esercizi che i clienti di un centro stanno eseguendo in un determinato momento e quanti sono |
| 4 | Visualizzare tutti gli armadietti liberi di un centro |
| 5 | Indicare il numero di visite fatte da un cliente con il suo medico nutrizionista per la scheda di alimentazione attuale. |
| 6 | Modificare l’attributo FrequenzaVisite di Scheda\_alimentazione con la media della frequenza delle visite effettuate da un cliente con il suo medico |
| 7 | Indicare il numero di clienti attualmente presenti in un determinato centro |
| 8 | Inserire un nuovo accesso ad un centro di un cliente se non si è già superato il valore di MaxClienti |
| 9 | Valutare durata media e giudizio della prestazione medio di un esercizio |

Questo argomento sarà trattato a parte all’interno del capitolo 4.

## 1.3- Glossario

|  |  |  |
| --- | --- | --- |
| Termine | Descrizione | Sinonimi |
| Centro | Luogo nel quale si svolgono le attività di fitness. | Sede, centro fitness |
| Sala | Locale all’interno di un centro nella quale sono contenute le apparecchiature. | Ambiente |
| Attrezzatura | Strumento a disposizione dei clienti per l’allenamento. | Macchinario, macchina, attrezzo ginnico, apparecchiatura |
| Cliente | Fruitore dei servizi dell’impresa. | Utente, firmatario |
| Dipendente | Ha una mansione all’interno della gerarchia aziendale. Il suo impiego può variare dal centro in cui lavora. |  |
| Responsabile | Dipendente a cui viene affidata la responsabilità di alcuni sottoposti. |  |
| Direttore | Dipendente che dirige un determinato centro. |  |
| Tutor | Dipendente che assiste il cliente nella sua attività fisica fornendogli schede di allenamento personalizzate. |  |
| Istruttore | Dipendente che organizza corsi nelle sale del centro. |  |
| Medico nutrizionista | Dipendente che visita periodicamente il cliente e gli fornisce schede di alimentazione personalizzate. |  |
| Piano di turnazione | Insieme degli orari e dei giorni in cui un dipendente lavora all’interno di un centro. |  |
| Contratto | Permette l’accesso alle aree del centro. Può essere standard o personalizzato, applicabile ad una o più sedi. |  |
| Istituto finanziario | Ente al quale l’impresa si rivolge per permettere la rateizzazione dei contratti dei clienti. |  |
| Scopo | Obiettivo del cliente scelto al momento della firma del contratto. Permette a tutor e medici nutrizionisti di impostare l’attività del cliente all’interno dei centri. | Obiettivo |
| Interesse | Attività svolta all’interno di un centro alla quale un cliente si definisce interessato al momento dell’iscrizione. Gli interessi di un cliente possono variare nel tempo. |  |
| Scheda di  allenamento | Scheda fornita dal tutor ad un cliente contenente gli esercizi che deve svolgere per raggiungere il suo obiettivo. |  |
| Scheda di alimentazione | Scheda fornita dal medico nutrizionista ad un cliente contenente la dieta che deve seguire per raggiungere il suo obiettivo. |  |
| Accesso | Data e ora in cui il cliente entra in un centro fitness. |  |
| Corso | Attività svolta nelle sale del centro a periodicità fissa. E’ tenuto da un istruttore. |  |
| Allenamento | Insieme di esercizi svolti da un cliente in una certa sessione. | Attività |
| Esercizio | Attività fisica compiuta dal cliente. Può essere aerobico o anaerobico e può prevedere o meno l’utilizzo di apparecchiature. |  |
| Integratore | Preparato commerciale abbinato ad una dieta. | Integratore alimentare |
| Fornitore | Ente esterno all’impresa che fornisce integratori alimentari. |  |
| Ordine | Lista di integratori che l’impresa ordina a un fornitore. |  |
| Magazzino | Insieme degli integratori di cui un centro dispone | Inventario |
| Forum | Servizio online fornito dall’impresa ai suoi clienti per la condivisione delle esperienze svolte all’interno dei centri. | Social network |
| Thread | Area del forum | Area |
| Post | Messaggio, articolo caricato da un utente del forum all’interno di un thread. Può essere di risposta a un altro post. | Messaggio |
| Amicizia | Relazione che lega due utenti. Un utente fa una richiesta di amicizia ad un altro utente che può decidere di accettarla o meno. |  |
| Amico | Utente legato ad una relazione di amicizia ad un altro utente. Se un utente A è amico di un altro utente B, allora anche B è ami-co di A. |  |
| Cerchia | Suddivisione degli amici di un utente definita da quest’ultimo in base agli interessi comuni. |  |
| Sfida | Scopo da raggiungere in un determinato tempo. Un utente (chiamato proponente) lancia una sfida ai propri amici che possono decidere se accettare o meno. |  |

# Progettazione concettuale

## 2.1- Strategia concettuale

Abbiamo scelto di operare secondo una strategia mista top-down/bottom-up.

Dopo aver analizzato ciascuna delle 4 aree in cui sono divise le specifiche, abbiamo individuato uno schema-scheletro generale.

Successivamente, concentrandoci su singole parti del diagramma, abbiamo proceduto a raffinamenti ulteriori fino a identificare le entità e le relazioni del diagramma ER alle quali, in un secondo momento, abbiamo applicato gli attributi e gli identificatori primari.

## 2.2- Fasi di sviluppo del diagramma ER

Abbiamo iniziato individuando le principali entità che compongono il diagramma ER: “Cliente”, “Dipendente”, “Contratto” e “Centro”. Esiste una relazione “Sottoscrizione” che collega l’entità Cliente con Contratto, una relazione “Accesso” che collega Contratto a Centro, una relazione “Impiego” che collega l’entità Dipendente a Centro.

Seppur le informazioni anagrafiche da memorizzare siano le stesse, sia per Cliente che per Dipendente, abbiamo preferito non creare una superclasse “Persona” che raccogliesse entrambe le entità perché queste, sin dai primi momenti della progettazione del database, risulteranno estremamente non correlate tra loro.

Cliente

Contratto

Dipendente

Centro

Sottoscrizione

Accesso

Impiego

Secondo la metodologia inside-out, abbiamo individuato le entità collegate alle quattro principali.

Abbiamo iniziato dall’entità Centro.

### 2.2.1- Sviluppo inside-out dell’entità “Centro”

Innanzitutto abbiamo aggiunto all’entità Centro i seguenti attributi:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodCentro | Codice identificativo |
| Indirizzo | Indirizzo del centro |
| NumeroTelefonico | Numero di telefono |
| Dimensione | Dimensione in metri quadri |
| MaxClienti | Numero massimo di clienti che il centro può ospitare contemporaneamente |

Il centro offre corsi tenuti da istruttori e ai quali i clienti si iscrivono. Abbiamo creato l’entità Corso che raccoglie tutti i dati d’interesse.

Questi sono gli attributi di Corso:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodCorso | Codice identificativo |
| Disciplina | Disciplina praticata |
| Livello | Livello di difficoltà |
| DataInizioCorso | Data in cui il corso ha inizio |
| DataFIneCorso | Data in cui il corso ha fine |
| MaxPartecipanti | Numero massimo di partecipanti ammessi |

Il corso si tiene in una sala di un centro oppure in una sua piscina; più in generale possiamo dire in un luogo d’allenamento. Abbiamo perciò creato l’entità superclasse “Luogo\_allenamento”, collegata a Centro tramite la relazione “Posizione\_Luogo”. Successivamente, abbiamo aggiunto le sottoclassi “Sala” e “Piscina”.

L’entità Luogo\_allenamento ha il seguente attributo:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodLuogo | Codice identificativo del luogo nel quale si svolgono i corsi |

Al quale l’entità Sala aggiunge:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| Nome | Nome della sala |

Mentre Piscina aggiunge:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| Dimensione | Dimensione in metri quadrati della piscina |

Un luogo d’allenamento deve trovarsi necessariamente all’interno di uno e un solo centro. Un centro dispone sicuramente di almeno una sala (altrimenti non sarebbe neanche un edificio), ma è possibile che al momento della creazione sul database del centro non si abbia l’elenco delle sale che lo formano. Per cui abbiamo scelto di mettere la cardinalità lato Centro di Posizione\_Luogo pari a (0,N), mentre lato Luogo\_allenamento pari a (1,1).

Centro

Luogo\_allenamento

Posizione\_Luogo

(0,N)

(1,1)

Sala

Piscina

Per memorizzare gli orari, giorni e luogo nel quale un corso si tiene, abbiamo creato l’entità “Giorno” che contiene tutti i giorni della settimana e la relazione ternaria “Pianificazione\_Corso” che collega Corso, Giorno e Luogo\_allenamento e contiene come attributi gli orari di inizio e fine corso. Si assume che un corso non si possa tenere due volte nello stesso giorno.

L’entità Giorno ha un solo attributo:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| NomeGiorno | Nome dei giorni della settimana |

Mentre la relazione Pianificazione\_Corso ha questi attributi:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| OraInizioCorso | Orario di inizio del corso |
| OraFineCorso | Orario di fine del corso |

Un corso può essere tenuto in fasce orarie diverse a seconda dei giorni, ma sicuramente deve averne almeno una. La cardinalità di Pianificazione\_Corso – lato Corso – è dunque (1,N). In un giorno possono esservi attività in più fasce orarie diverse fra loro, ma potrebbe anche essere giorno di chiusura e non avere corsi in programma per tutta la giornata.

La cardinalità di Pianificazione\_Corso – lato Giorno – è (0,N). Un luogo d’allenamento, può essere utilizzato in più fasce orarie, anche all’interno dello stesso giorno, e non necessariamente deve essere usato. La cardinalità lato Luogo-allenamento è (0,N).

Giorno

Corso

Luogo\_Allenamento

Pianificazione\_Corso

(0,N)

(0,N)

(1,N)

Il centro ha un orario di servizio. Per indicare a che ore apre un centro nei vari giorni della settimana, si crea l’entità “Orario\_servizio”, collegandola a Centro tramite la relazione Orario\_centro e a Giorno tramite Orario\_giorno.

“Orario\_servizio” contiene gli orari di apertura e chiusura del centro, e permette di memorizzare più fasce orarie di apertura di un centro per uno stesso giorno.

Gli attributi dell’entità Orario\_servizio sono dunque:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodOrario | Codice identificativo dell’orario di servizio |
| OrarioApertura | Orario di apertura del centro |
| OrarioChiusura | Orario di chiusura del centro |

Le cardinalità di Orario\_centro e Orario\_giorno dal lato di Centro e Giorno sono (0,N), poiché un Centro può essere sempre chiuso (si pensi nei primi giorni dalla creazione del Centro del database quando ancora non è stato definito il calendario settimanale), e in un giorno della settimana possono essere aperti uno o più centri oppure nessuno.

Poiché un orario corrisponde sempre ad un solo centro e ad un solo giorno della settimana, le cardinalità delle due relazioni dal lato di Orario\_servizio sono entrambe (1,1).

Centro

Orario\_servizio

Orario\_centro

(0,N)

(1,1)

Giorno

Orario\_giorno

(0,N)

(1,1)

Una sala può contenere attrezzature ginniche; si crea dunque l’entità “Attrezzatura” collegandola a Sala tramite la relazione “Posizione\_Attrezzatura”.

Gli attributi dell’entità Attrezzatura sono:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodAttrezzatura | Codice identificativo dell’attrezzo ginnico |
| Tipologia | Categoria di attrezzi ginnici a cui la macchina appartiene |
| ConsumoEnergetico | Valore espresso in Wattora: indica il consumo energetico della macchina |
| Usura | Numero intero compreso tra 0 e 100 che indica il livello di usura dell’atrezzo: 100 indica un completo logoramento. |

In una sala ci possono essere più attrezzature oppure nessuna. Un attrezzo ginnico, invece, appartiene ad una ed una sola sala. La cardinalità di Posizione\_Attrezzatura è dunque (1,1) dal lato di Attrezzatura, mentre è (0,N) dal lato di Sala.

Attrezzatura

Sala

Posizione\_Attrezzatura

(1,1)

(0,N)

Ogni attrezzo dispone di regolazioni che possono essere modificate a seconda dell’esercizio da svolgere. Viene creata dunque l’entità “Regolazione”, collegata a Attrezzatura tramite la relazione “Regolazione\_attrezzatura”.

L’entità Regolazione ha un attributo:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| NomeRegolazione | Nome che indica la regolazione di cui è dotata la macchina. E’ identificatore insieme all’attributo esterno Attrezzatura. |

Poiché un attrezzo può disporre o meno di regolazioni, la cardinalità di Regolazione\_attrezzatura – lato Attrezzatura – è (0,N). Una regolazione, invece, appartiene ad un solo attrezzo, quindi la cardinalità lato Regolazione è (1,1).

Regolazione

Attrezzatura

Regolazione\_attrezzatura

(1,1)

(0,N)

Il centro dispone inoltre di spogliatoi. Viene aggiunta l’entità “Spogliatoio”, collegandola a Centro tramite la relazione “Posizione\_Spogliatoio”.

Gli attributi dell’entità Spogliatoio sono:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodSpogliatoio | Codice identificativo |
| Capienza | Totale di persone che lo spogliatoio può accogliere |
| PostiDisponibili | Differenza tra capienza e numero di attuali utilizzatori |
| Posizione | Punto cardinale che indica la posizione dello spogliatoio all’interno del centro |

In un centro ci possono essere più spogliatoi. Per permettere di inserire nel database un centro senza fornire immediatamente l’elenco degli spogliatoi presenti, si è scelto di porre la cardinalità (0,N) alla relazione Posizione\_spogliatoio lato Centro, mentre dal lato Spogliatoio la cardinalità è (1,1).

Centro

Spogliatoio

Posizione\_spogliatoio

(0,N)

(1,1)

Negli spogliatoi sono collocati degli armadietti, assegnati ai clienti al momento dell’accesso al centro. Viene aggiunta l’entità “Armadietto”, collegandola a Spogliatoio tramite la relazione “Posizione\_armadietto”.

Gli attributi di Armadietto sono:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodArmadietto | Codice identificativo |
| Combinazione | Combinazione numerica che permette lo sblocco dell’armadietto |

Un armadietto è posizionato in uno e un solo spogliatoio. Dal lato di Spogliatoio, dunque, la cardinalità di Posizione\_armadietto è (1,1). Dal lato di Spogliatoio, abbiamo optato per una cardinalità (0,N), in modo tale che appena si aggiunge un record di Spogliatoio nel database, non è necessario inserire immediatamente i record relativi agli armadietti contenuti nello spogliatoio.

Spogliatoio

Armadietto

Posizione\_armadietto

(0,N)

(1,1)

Per rifornirsi di integratori alimentari, il centro ne fa un ordine ad un fornitore. Vengono aggiunte le entità “Fornitore”, “Integratore” e “Ordine”; Centro, Fornitore e Integratore sono collegati a Ordine rispettivamente con le relazioni “Ordine\_centro”, “Ordine\_fornitore”, “Ordine\_integratore”.

Gli attributi di Fornitore sono:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| PartitaIva | Codice identificativo |
| NomeCommerciale | Nome dell’azienda fornitrice di integratori |
| FormaSocietaria | Forma societaria dell’azienda fornitrice (s.p.a, s.a.s., ecc.) |
| Indirizzo | Indirizzo della sede dell’azienda |
| NumeroTelefonico | Numero di telefono della sede dell’azienda |

Gli attributi di Integratore sono invece:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| NomeCommerciale | Nome commerciale del prodotto, funge da identificatore |
| Sostanza | Sostanza contenuta nell’integratore |
| Forma | Forma nella quale è commercializzato l’integratore  (liquida o solida) |
| NumeroPezzi | Numero di porzioni contenute nella confezione |
| Concentrazione | Concentrazione della sostanza in ogni porzione, espressa in milligrammi o in millilitri a seconda della forma  dell’integratore |
| DataScadenza | Data di scadenza dell’integratore |
| PrezzoIngrosso | Prezzo all’ingrosso del prodotto |
| PrezzoDettaglio | Prezzo al dettaglio |

Gli attributi di Ordine sono i seguenti:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodOrdine | Codice identificativo |
| DataEvasione | Data di invio dell’ordine dal centro al fornitore |
| Stato | Stato dell’ordine (Incompleto/evaso) |
| DataConsegnaRichiesta | Data entro la quale la direzione del centro richiede la consegna delle merci |

Alla relazione Ordine\_integratore si aggiunge un attributo:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| Quantita | Numero che esprime le unità di prodotto richieste al fornitore |

Per quanto riguarda la cardinalità delle relazioni “Ordine\_centro”, “Ordine\_fornitore” e “Ordine\_integratore”, abbiamo fatto le seguenti scelte:

Un centro può fare almeno un ordine oppure nessuno, mentre un ordine deve avere obbligatoriamente uno e un solo centro che lo ha commissionato. La cardinalità lato Centro di “Ordine\_centro” è quindi (0,N), mentre dal lato di Ordine è (1,1).

Un fornitore può essere coinvolto in uno o più ordini oppure nessuno, mentre un ordine deve avere obbligatoriamente uno e un solo fornitore che provvede ad esso. La cardinalità lato Fornitore di “Ordine\_fornitore” è quindi (0,N), mentre dal lato di Ordine è (1,1).

Un integratore può essere ordinato zero o più volte. Un ordine, invece, deve necessariamente coinvolgere almeno un prodotto, ma ne può contenere anche più di uno. La cardinalità lato Integratore di “Ordine\_integratore” è quindi (0,N), mentre dal lato di Ordine è (1,N).

Un fornitore ha un insieme di integratori che può commercializzare. Per rappresentare ciò, abbiamo creato la relazione “Commercializzazione” che lega le entità Integratore e Fornitore.

Commercializzazione ha un atrtibuto:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodEsterno | Codice esterno associato da un fornitore ad un prodotto |

Un fornitore può commercializzare nessuno o più integratori (in modo tale da poterne inserire l’elenco in un secondo momento rispetto alla creazione nel database di un fornitore): la cardinalità di Commercializzazione – lato Fornitore – è quindi (0,N). Un integratore, per essere nel database, significa che è stato commercializzato da almeno un fornitore, quindi La cardinalità minima dal lato di Integratore è 1; poiché un prodotto può essere commercializzato da più fornitori, la cardinalità massima è N.

Integratore

Centro

Fornitore

Ordine

(0,N)

(0,N)

(0,N)

Commercializzazione

(0,N)

(1,N)

Ordine\_centro

Ordine\_integratore

Ordine\_fornitore

(1,1)

(1,1)

(1,N)

Infine, il centro si rivolge a istituti finanziari per permettere ai suoi clienti la rateizzazione dell’importo annuale. SI aggiunge quindi l’entità “Istituto\_finanziario”, collegata a Centro tramite la relazione “Convenzione”.

Gli attributi di Istituto\_finanziario sono:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodIstituto | Codice identificativo |
| RagioneSociale | Ragione sociale dell’istituto finanziario |
| TassoInteresse | Tasso d’interesse che l’istituto applica alle rate |

La cardinalità di questa relazione è (0,N) dal lato di Centro, per permettere, al momento dell’inserimento di un record di Centro, di non collegarvi immediatamente alcun istituto finanziario. Un istituto finanziario, per il fatto che è memorizzato nel database, ha sicuramente almeno una convenzione con un centro, e ne può avere anche più di una. La relazione Convenzione, dal lato di Istituto Finanziario, ha dunque cardinalità (1,N).

Centro

Istituto Finanziario

Convenzione

(0,N)

(1,N)

### 2.2.2- Sviluppo inside-out dell’entità “Dipendente”

Dopo esserci concentrati sull’entità Centro, abbiamo focalizzato l’attenzione su chi ha una mansione all’interno del centro: i dipendenti.

Abbiamo aggiunto all’entità Dipendente i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodFiscale | Codice identificativo |
| Nome | Nome anagrafico |
| Cognome | Cognome anagrafico |
| DataNascita | Data di nascita del dipendente |
| Indirizzo | Residenza |
| CodDocumento | Codice di un documento di identità |
| Prefettura | Prefettura di rilascio del documento di identità |

La relazione Impiego (che unisce Dipendente a Centro) rappresenta in senso generale il fatto che un dipendente lavori in un determinato centro. Si aggiunge l’ attributo “Mansione”, grazie al quale è possibile esprimere qualsiasi possibile impiego di un dipendente all’interno di una struttura.

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Mansione | Indica l’impego del dipendente all’interno di un centro |

Poiché un dipendente è tale se e solo se ha un impiego in un centro e può lavorare in centri diversi, la cardinalità di Impiego - lato Dipendente – sarà (1,N). In un centro, possono invece lavorare molti dipendenti, ma al momento della creazione nel database, potrebbe non esservi stato aggiunto alcun dipendente. La cardinalità di Impiego – lato Centro – sarà dunque (0,N).

Dipendente

Centro

Impiego

(0,N)

(1,N)

Per le attività strettamente collegate alle attività della palestra (ad esempio tutor, istruttore) sono state create delle sottoclassi all’entità Dipendente.

Ogni sala è posta sotto la responsabilità di un dipendente; si aggiunge l’entità “Responsabile Sala” come sottoclasse di Dipendente, collegandola tramite la relazione “Gestione\_Sala” a Sala.

Non si aggiungono altri particolari attributi a questa entità.

Un responsabile, per essere tale, deve avere la gestione di almeno una sala, ma ne può avere anche più di una. Una sala, invece, è posta sotto la responsabilità di uno ed un solo responsabile. Per cui la relazione Gestione\_Sala ha cardinalità (1,N) dal lato di Responsabile\_Sala, mentre dal lato di Sala ha cardinalità (1,1).

Dipendente

Sala

Gestione\_Sala

(1,1)

(1,N)

Responsabile\_Sala

Ciascun corso che il centro propone è gestito da un istruttore. Abbiamo dunque creato l’entità “Istruttore” come sottoclasse di Dipendente, connessa all’entità Corso tramite la relazione “Insegnamento”.

Non si aggiungono altri particolari attributi a questa entità.

Un istruttore può insegnare in più corsi oppure può in qualsiasi momento non avere alcun corso da tenere. Un corso, invece, può essere tenuto da uno e un solo istruttore. Per questi motivi, la relazione Insegnamento ha cardinalità (0,N) dal lato di Istruttore e (1,1) dal lato di Corso.

Dipendente

Corso

Insegnamento

(1,1)

(0,N)

Istruttore

Inoltre il centro è diretto da un direttore, che è anch’egli un dipendente. Abbiamo dunque aggiunto, come sottoclasse di Dipendente, l’entità ”Direttore”, collegandola a Centro tramite la relazione “Direzione”

Non si aggiungono altri particolari attributi a questa entità.

Un centro è diretto da uno e un solo direttore, mentre un direttore potrebbe trovarsi a dirigere più centri, ma ne dirigerà sempre almeno uno. La cardinalità di Direzione è quindi (1,1) lato Centro e (1,N) lato Direttore.

Dipendente

Centro

Direzione

(1,1)

(1,N)

Direttore

Ogni cliente, al momento della sottoscrizione del contratto, è affidato ad un tutor. Abbiamo creato l’entità “Tutor”, che viene collegata tramite la relazione “Assegnazione\_tutor” a Cliente.

Non si aggiungono particolari attributi a questa entità.

Ad un cliente viene assegnato uno ed un solo tutor al momento della firma del contratto. Un tutor invece può aver assegnati zero oppure più clienti. La cardinalità di Assegnazione sarà (1,1) dal lato di Cliente, (0,N) dal lato di Tutor.

Dipendente

Cliente

Assegnazione\_tutor

(1,1)

(0,N)

Tutor

Un cliente viene assistito da un consulente nella scelta del contratto migliore per lui. Si aggiunge l’entità Consulente, sottoclasse di Dipendente, collegandola a Cliente tramite la relazione “Consulenza”.

Non si aggiungono particolari attributi a questa relazione. L’entità Contratto verrà analizzata in seguito.

Un consulente può essere assunto senza averlo immediatamente coinvolto nella scelta di un contratto per un cliente ma, allo stesso tempo, può essere il consulente di molti contratti. La cardinalità di Consulenza – lato Consulente – è (0,N). Un contratto, invece, ha uno ed un solo consulente abbinato, per cui la cardinalità di Consulenza, lato Contratto, è (1,1).

Dipendente

Contratto

Consulenza

(1,1)

(0,N)

Consulente

Al momento della firma del contratto, il cliente viene affidato ad un medico nutrizionista. Abbiamo creato l’entità “Medico\_nutrizionista”, collegandolo a Cliente tramite la relazione “Assegnazione\_medico”.

Non si aggiungono particolari attributi a questa entità.

Un medico nutrizionista può essere assunto senza assegnargli immediatamente un cliente da visitare; un medico può però visitare più clienti. Un cliente, invece, ha uno ed un solo medico nutrizionista assegnato. La cardinalità di Assegnazione\_medico è (0,N) dal laro di Medico\_nutrizionista, ed è (1,1) dal lato di Cliente.

Il cliente viene periodicamente visitato da un medico (si suppone che il suo medico assegnato possa essere sostituito da un altro medico per effettuare la visita).

Si crea l’entità “Visita” che collega Medico\_nutrizionista a Cliente tramite le relazioni “Visita\_medico” e “Visita\_cliente”.

L’entità Visita ha questi attributi:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodVisita | Codice identificativo |
| DataVisita | Data in cui è stata effettuata la visita |

La cardinalità lato Cliente di Visita\_cliente è (1,N), in quanto il cliente viene visitato almeno una volta e può essere visitato più volte. La cardinalità lato Medico\_nutrizionista di Visita\_medico è invece (0,N), perché un medico può non aver effettuato ancora alcuna visita oppure ne ha effettuate una o più. Le cardinalità lato Visita di queste due relazioni sono entrambe (1,1), perché ogni visita è relativa ad un unico cliente e ad un unico medico.

Dipendente

Cliente

Assegnazione\_medico

(1,1)

(0,N)

Medico\_nutrizionista

Visita\_medico

(1,N)

(0,N)

Visita\_cliente

Visita

(1,1)

(1,1)

Inoltre, ogni dipendente ha un responsabile che controlla il suo lavoro. Si crea dunque la sotto entità di Dipendente “Resp\_personale”, collegandola con Dipendente con la relazione “Responsabilita”.

Non si aggiungono particolari attributi a questa entità.

Un responsabile del personale può, al momento, non aver assegnato alcun dipendente, mentre un dipendente ha sempre un suo responsabile. La cardinalità di Responsabilita è (0,N) dal lato di Resp\_personale, (1,1) dal lato di Dipendente.

Dipendente

Responsabilita

(1,1)

(0,N)

Resp\_personale

Il sistema deve gestire la turnazione del personale. Abbiamo quindi creato l’entità “Turnazione”, collegata a Centro tramite “Turno\_centro”, a Dipendente tramite “Turno” e a Giorno tramite “Turno\_giorno”.

Turnazione ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributi | Descrizione |
| CodTurnazione | Codice identificativo |
| OraInizioTurno | Orario di inizio del turno |
| OraFIneTurno | Orario di fine del turno |

Si supponga che i turni di lavoro vengano decisi in un secondo momento rispetto alla creazione nel database di centri e dipendenti.

Un Centro, quindi, può avere molti turni di lavoro oppure nessuno. Un piano di turnazione si riferisce ad uno ed un solo centro. La cardinalità di Turno\_centro è (0,N) dal lato di Centro e (1,1) dal lato di Turnazione.

Un dipendente, di norma, ha più orari di lavoro, diversi a seconda dei giorni in cui lavora. Quando si assume un dipendente e si inserisce nel database, non è detto però che si sappia subito il suo piano di lavoro. La cardinalità di Turno è quindi (0,N) dal lato di Dipendente. Una turnazione, invece, coinvolge un solo dipendente alla volta, quindi la cardinalità lato Turnazione è (1,1).

In un giorno ci possono essere molti turni di lavoro oppure nessuno (ad esempio il giorno di riposo). Una turnazione, invece, si riferisce ad un solo giorno lavorativo. La cardinalità di Turno\_giorno è (0,N) dal lato di Giorno, mentre dal lato di Turnazione è (1,1).

### 2.2.3- Sviluppo inside-out dell’entità “Cliente”

Dopo aver analizzato le entità rappresentanti il centro e i dipendenti, ci siamo occupati dei clienti che usufruiscono dei servizi delle strutture.

Si considera Cliente un firmatario di contratto.

Abbiamo dunque creato l’entità Cliente con questi attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodFiscale | Codice identificativo |
| Nome | Nome anagrafico |
| Cognome | Cognome anagrafico |
| DataNascita | Data di nascita del dipendente |
| Indirizzo | Residenza |
| CodDocumento | Codice di un documento di identità |
| Prefettura | Prefettura di rilascio del documento di identità |
| Username | Username del cliente per accedere al forum |
| Password | Password associata al cliente per l’accesso al forum |

Un cliente, al momento della firma del contratto, deve indicare lo scopo che vuole raggiungere grazie alle attività che svolgerà all’interno del centro. Abbiamo creato l’entità Scopo, collegata tramite la relazione “Scelta\_scopo” a Cliente.

Scopo ha un solo attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| NomeScopo | Nome dello scopo, funge da identificativo |

NomeScopo può assumere soltanto uno fra i tre valori: “Potenziamento Muscolare”, “Dimagrimento” oppure “Scopo Ricreativo”.

Poiché un cliente deve scegliere uno ed un solo scopo, la cardinalità di Scelta\_scopo – lato Cliente – è (1,1). Uno scopo può essere invece scelto da nessuno oppure da più clienti, per cui la cardinalità dal lato di Scopo è (0,N).

Cliente

Scopo

Scelta\_scopo

(1,1)

(0,N)

Nel caso in cui un cliente scelga lo scopo “Potenziamento Muscolare”, deve scegliere anche un gruppo di muscoli da potenziare, con relativo livello di approfondimento, misurato tramite una scala di valori (“lieve”, “moderato”, “elevato”).

Abbiamo quindi creato l’entità Muscolo, che raccoglie tutti i muscoli allenabili del corpo umano, collegandolo con la relazione “Muscolo\_target” a Cliente.

L’entità Muscolo ha un solo attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| NomeMuscolo | Nome del muscolo, funge da identificativo |

La relazione “Muscolo\_target” ha l’attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| LivelloPotenziamento | Livello di potenziamento del muscolo che il cliente vuole raggiungere |

Un cliente, se non ha scelto lo scopo “Potenziamento Muscolare”, non deve scegliere un muscolo target (un vincolo di integrità generico gestirà questa situazione); se invece ha scelto quello scopo, può scegliere più di un muscolo. La cardinalità di Muscolo\_target è quindi (0,N) dal lato di cliente. Un muscolo, inoltre, potrebbe non essere stato scelto da nessun cliente, oppure da molti: la cardinalità dal lato di muscolo è quindi (0,N).

Cliente

Muscolo

Muscolo\_target

(0,N)

(0,N)

Ad ogni cliente è associato un tutor che lo accompagna durante la sua permanenza nella palestra fornendogli schede d’allenamento personalizzate. Abbiamo quindi creato l’entità “Scheda\_allenamento”, collegata a Cliente tramite la relazione “Assegnazione\_scheda” e a Tutor tramite “Consegna\_scheda”.

Scheda\_allenamento ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodScheda | Codice identificativo |
| DataInizioScheda | Data di inizio validità della scheda |
| DataFineScheda | Data di fine validità della scheda |

Un cliente può aver assegnate zero o più schede di allenamento. Una scheda di allenamento è invece assegnata ad uno ed un solo cliente. La cardinalità di Assegnazione\_scheda è (0,N) dal lato di Cliente, (1,1) dal lato di Scheda\_allenamento.

Un tutor può assegnare zero o più schede d’allenamento. Una scheda d’allenamento è invece assegnata da un solo tutor. La cardinalità di Consegna\_scheda è (0,N) dal lato di Tutor, (1,1) dal lato di Scheda\_allenamento.

Cliente

Scheda\_allenamento

Assegnazione\_scheda

(0,N)

(1,1)

Tutor

Consegna\_scheda

(0,N)

(1,1)

Una scheda d’allenamento comprende uno o più esercizi da ripetersi quotidianamente per tutta la sua durata. Si crea dunque l’entità “Esercizio” che raccolga le informazioni relative agli esercizi. Abbiamo collegato Esercizio a Scheda\_allenamento tramite la relazione “Composizione”.

Poiché gli esercizi possono essere di due tipi, aerobico o anaerobico, abbiamo creato due omonime sotto-entità di Esercizio.

Esercizio ha questi attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodEsercizio | Codice identificativo |
| Nome | Nome dell’esercizio |
| DispendioEnergetico | Dispendio energetico misurato in kilocalorie/ora impiegato nell’esecuzione dell’esercizio |

Ai quali l’entità Aerobico aggiunge:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Durata | Durata in secondi dell’esercizio |

Mentre l’entità Anaerobico aggiunge:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Ripetizioni | Numero di ripetizioni previste dall’esercizio |
| Recupero | Tempo di recupero in minuti |

Una scheda d’allenamento è composta da almeno un esercizio, mentre un esercizio può essere integrato in più schede d’allenamento. La cardinalità di Composizione è quindi (1,N) dal lato di Scheda\_allenamento, mentre è (0,N) dal lato di Esercizio.

Scheda\_allenamento

Esercizio

Composizione

(1,N)

(0,N)

Aerobico

Anaerobico

Un esercizio può prevedere l’utilizzo di una determinata apparecchiatura, e questa può essere usata con una particolare regolazione. Abbiamo quindi creato l’entità “Configurazione\_es”, collegata a Esercizio con la relazione “Configurazione”, a Attrezzatura con “Attrezzo\_es” e a Regolazione con la relazione “Regolazione\_es”.

L’entità Configurazione\_es ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodConfigurazione | Codice identificativo |

La relazione Regolazione\_es ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Valore | Valore di regolazione da impostare su una macchina ai fini dell’esercizio |

Non tutti gli esercizi hanno una attrezzatura da utilizzare, quindi non tutti gli esercizi hanno una propria configurazione. Allo stesso modo, una configurazione può essere riutilizzata per esercizi diversi, ma appartiene obbligatoriamente ad almeno un esercizio. La cardinalità di Configurazione è (0,N) dal lato di Esercizio, mentre è (1,N) dal lato di Configurazione\_es.

Una configurazione d’esercizio è tale perché prevede l’utilizzo di una ed una sola atrtezzatura. Un’attrezzatura, invece, può non essere coinvolta in nessuna configurazione. La cardinalità di Attrezzo\_es è (1,1) dal lato di Configurazione\_es e (0,N) dal lato di Attrezzatura.

Una configurazione può prevedere l’utilizzo di nessuna, una o più regolazioni della macchina alla quale è associata. Una regolazione può essere coinvolta in una configurazione oppure no. La cardinalità di Regolazione\_es è (0,N) da entrambi i lati.

Esercizio

Configurazione\_es

Configurazione

(0,N)

(1,N)

Attrezzo\_es

Regolazione\_es

Attrezzatura

Regolazione

(0,N)

(1,1)

(0,N)

(0,N)

Una ripetizione di un esercizio potrebbe coinvolgere un’ulteriore apparecchiatura con una particolare configurazione. Si aggiungono quindi l’entità “Ripetizione” collegandola a Anaerobico tramite la relazione “Ripetizione\_es” e la relazione “Configurazione\_rip” che collega Ripetizione a Configurazione\_es.

Ripetizione ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| NumeroRipetizione | Indica il numero della ripetizione presa in considerazione. E’ identificatore insieme all’attributo esterno Esercizio |

Un esercizio anaerobico può prevedere o meno ripetizioni, ma una ripetizione è sempre associata ad un solo esercizio. La cardinalità di Ripetizione\_es è dunque (0,N) dalla parte di Anaerobico e (1,1) dal lato di Ripetizione.

Una ripetizione può prevedere zero o più regolazioni diverse di attrezzature, e non è detto che una configurazione sia coinvolta in una ripetizione di un esercizio. La cardinalità di Regolazione\_rip è (0,N) da entrambi i lati.

Esercizio

Configurazione

Aerobico

Anaerobico

Ripetizione

Ripetizione\_es

Configurazione\_rip

(0,N)

(0,N)

(1,1)

(0,N)

Ciascun cliente è dotato di un dispositivo RFID che monitora la sua effettiva attività fisica durante le sessioni d’allenamento. Per raccogliere queste informazioni, abbiamo creato l’entità “Prestazione\_esercizio”, collegandola a Cliente tramite la relazione “Prestazione\_Cliente”, a Regolazione tramite “Regolazione\_reale” e a Esercizio tramite “Esercizio\_reale”.

Prestazione esercizio ha questi attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodPrestazione | Codice identificativo della prestazione |
| TimestampInizio | Timestamp di inizio esercizio |
| TimestampFine | Timestamp di fine esercizio |
| ValutazionePrestazione | Valutazione in decimi della prestazione, misurata tramite funzionalità di back-end (vedi paragrafo 5.5.2) |

Regolazione\_reale ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Valore | Indica il valore della regolazione che l’utente ha impostato sulla macchina |

Esercizio\_reale ha questi attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Durata | Durata effettiva dell’esercizio eseguito dal cliente. Ha valore NULL se l’esercizio è anaerobico |
| TempoRecupero | Tempo di recupero effettivo eseguito dal cliente. Ha valore NULL se l’esercizio è aerobico |
| NumeroRipetizioni | Numero di ripetizioni effettivamente eseguite dal cliente. Ha valore NULL se l’esercizio è aerobico |

Se l’esercizio non necessita di apparecchiature, non ci saranno regolazioni particolari alle quali il cliente si dovrà attenere. Inoltre, una regolazione, può non essere coinvolta in alcun esercizio, quindi la cardinalità di Regolazione\_reale è (0,N) da entrambi i lati.

Un esercizio presente nel database potrebbe non essere mai stato eseguito dai clienti, ma una prestazione di un cliente si riferisce ad uno ed un solo esercizio. La cardinalità di Esercizio\_reale è (0,N) dal lato di Esercizio e (1,1) dal lato di Prestazione\_esercizio.

Un cliente può non aver eseguito alcun esercizio, mentre una prestazione si riferisce ad uno ed un solo cliente. La cardinalità di Prestazione\_cliente è (0,N) dal lato di Cliente, e (1,1) dal lato di Prestazione\_esercizio

Esercizio

Regolazione

Cliente

Ripetizione\_es

Regolazione\_rip

(0,N)

(0,N)

(1,1)

(0,N)

Ripetizione\_es

Prestazione\_esercizio

(1,1)

(0,N)

Un cliente, quando sottoscrive il contratto, viene visitato immediatamente dal suo medico nutrizionista, che ne misura altezza, peso e altri dati relativi alla massa corporea. Queste misurazioni sono effettuate settimanalmente. Abbiamo creato dunque l’entità “Misurazione”, contenente le statistiche fisiche relative al cliente, e l’abbiamo collegata a Cliente tramite la relazione “Controllo”.

Gli attributi di Misurazione sono:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| DataMisurazione | Data in cui è effettuata la misurazione. Costituisce l’identificatore dell’entità insieme all’identificatore esterno Cliente |
| Peso | Indica il peso del cliente in Kilogrammi |
| Altezza | Indica l’altezza del cliente in centimetri |
| Stato | Indica lo stato del cliente (sottopeso/normopeso/sovrappeso) in base al calcolo dell’Indice di Massa Corporea |
| IMC | Attributo derivato; valore dell’ indice di massa corporea |
| AcquaTotale | Percentuale di acqua totale |
| MassaMagra | Percentuale di massa magra |
| MassaGrassa | Percentuale di massa grassa |

Poiché un cliente viene visitato appena dopo aver firmato il contratto, e generalmente più di una volta, la relazione Controllo ha cardinalità (1,N) dal lato di Cliente.

Una misurazione, invece, appartiene ad uno ed un solo cliente, per cui la cardinalità lato Misurazione è (1,1).

Cliente

Misurazione

Controllo

(1,N)

(1,1)

Un medico nutrizionista elabora una scheda di alimentazione personalizzata per ciascuno dei clienti a lui assegnati. Abbiamo creato l’entità “Scheda\_alimentazione”, collegata a Cliente tramite la relazione “Alimentazione\_cliente”, e a Medico tramite la relazione “Consiglio”.

L’entità Scheda\_alimentazione ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodScheda | Codice identificativo della scheda |
| Obiettivo | Obiettivo indicato dal paziente al medico nutrizionista (campo testuale) |
| DataInizioScheda | Data di inizio validità della scheda |
| DataFIneScheda | Data di fine validità della scheda |
| FrequenzaVisita | Frequenza in giorni con il quale il medico decide di visitare il paziente |

N.B. l’attributo “Obiettivo” presente in questa entità non rappresenta lo stesso concetto espresso dall’entità “Scopo”.

- Obiettivo infatti indica un’informazione aggiuntiva che il cliente dà al medico nutrizionista per aiutarlo a scegliere la scheda più adatta per lui.

- Scopo indica invece l’obbiettivo prefissato dal cliente al momento della firma del contratto, e può essere soltanto “Scopo ricreativo”, “Potenziamento Muscolare” oppure “Dimagrimento”.

Un cliente può avere zero o più schede di alimentazione (una attuale e le altre scadute), mentre una scheda di alimentazione appartiene ad uno ed un solo cliente. La cardinalità di Alimentazione\_cliente è (0,N) dal lato di Cliente, (1,1) dal lato di Scheda\_alimentazione.

Un medico può consigliare zero oppure molte schede di alimentazione complessivamente ai suoi clienti. Una scheda di alimentazione è invece consigliata da uno ed un solo medico. La cardinalità di Consiglio è (0,N) dal lato di Medico e (1,1) dal lato di Scheda\_alimentazione.

Medico

Cliente

(0,N)

(1,1)

Scheda\_alimentazione

Consiglio

Alimentazione\_cliente

(1,1)

(0,N)

In una scheda d’alimentazione è presente la dieta che il cliente deve seguire per raggiungere l’obbiettivo. Si aggiunge l’entità Dieta, congiunta a Scheda\_alimentazione tramite la relazione “Composizione\_S\_alim”.

L’entità Dieta comprende gli attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodDieta | Codice identificativo |
| ApportoCalorico | Apporto calorico in kilocalorie a cui il cliente deve attenersi per rispettare la dieta |
| NumeroPasti | Attributo derivato che indica il totale di pasti da cui è composta la dieta |

In una scheda di alimentazione vi è una ed una sola dieta, mentre una dieta può essere presente in nessuna o in più schede d’alimentazione. La cardinalità di “Composizione\_S\_alim” è dunque (1,1) dal lato di Scheda\_alimentazione, (0,N) dal lato di Dieta.

Dieta

Scheda\_alimentazione

Composizione\_S\_alim

(0,N)

(1,1)

Una dieta è formata da più pasti. Abbiamo aggiunto l’entità “Pasto”, collegandola a Dieta tramite la relazione “Composizione\_dieta”.

L’entità Pasto ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Nome | Nome del pasto, funge da identificatore dell’entità |
| Composizione | Campo testuale che indica la composizione del pasto |

Poiché in una dieta ci possono essere più pasti, ma sempre e comunque almeno uno, la cardinalità di Composizione\_dieta – lato Dieta – è (1,N). Un pasto può però non essere incluso in alcuna dieta oppure in una o più diete. La cardinalità dal lato di Pasto è dunque (0,N).

Pasto

Dieta

Composizione\_dieta

(0,N)

(1,N)

Ad una Scheda di alimentazione possono essere abbinati degli integratori alimentari. Si crea dunque la relazione “Abbinamento” che leghi Scheda\_alimentazione ad Integratore.

Poiché una scheda di alimentazione può anche non contenere integratori, la cardinalità di Abbinamento – lato Scheda\_alimentazione – è (0,N).

Un integratore, inoltre, può non essere ancora stato abbinato ad una scheda oppure in più di una, per cui la cardinalità – lato Integratore – è ancora (0,N).

Scheda\_alimentazione

Integratore

Abbinamento

(0,N)

(0,N)

I clienti possono acquistare gli integratori direttamente da uno dei centri dell’azienda, supponiamo solamente dal centro in cui ha deposto il contratto attuale. Abbiamo dunque creato l’entità “Acquisto” che raccolga i dati relativi all’acquisto di integratori da parte dei clienti. Acquisto è collegata a integratore tramite la relazione “Acquisto\_int” e a Cliente tramite “Acquisto\_cli”.

L’entità Acquisto ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodAcquisto | Codice identificativo |
| DataAcquisto | Data dell’acquisto |
| Quantita | Quantità acquistata dal cliente |
| Importo | Importo effettivamente speso dal cliente per acquistare il prodotto |

Poiché un cliente può decidere di non effettuare acquisti oppure ne può fare molti, la cardinalità di Acquisto\_cli dal lato di Cliente è (0,N). Un integratore può essere acquistato da molti clienti oppure da nessuno, quindi la cardinalità di Acquisto\_int dal lato di Integratore è (0,N). Poiché un acquisto è riferito ad un solo cliente e ad un solo integratore, le cardinalità di Acquisto\_cli e Acquisto\_int dal lato di Acquisto sono entrambe (1,1).

Cliente

Acquisto

Acquisto\_cli

(0,N)

(1,1)

Integratore

Acquisto\_int

(0,N)

(1,1)

Per quanto riguarda il forum, tutti i clienti ne hanno l’accesso con le credenziali scelte al momento dell’iscrizione (gli attributi “Username” e “Password” dell’entità Cliente).

Il cliente sceglie un insieme di interessi relativi ad attività sportive. Abbiamo creato quindi un’entità “Attivita” e l’abbiamo collegata a Cliente tramite la relazione “Interesse”.

L’entità Attivita ha un unico attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| NomeAttivita | Nome dell’attività, funge da identificatore dell’entità |

Poiché un cliente può scegliere di non inserire nel suo profilo social i suoi interessi oppure può inserirne molti, la cardinalità di Interesse è (0, N) dal lato di Cliente. Un’attività può essere scelta da zero o più clienti, quindi la cardinalità dal lato di Attivita è (0,N).

Cliente

Attivita

Interesse

(0,N)

(0,N)

Inoltre il cliente (che per il resto del capitolo chiameremo “user” o “utente”) può inviare richieste d’amicizia agli altri user; questi possono decidere di accettare o rifiutare.

Per memorizzare tutte le richieste di amicizia, accettate, rifiutate oppure ancora senza risposta, abbiamo creato la relazione “Amicizia”, relazione ricorsiva su Cliente, con ruoli “Richiedente” e “Ricevente”.

Amicizia ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Stato | Stato dell’amicizia, può assumere uno tra questi tre valori: “accettata”, “rifiutata”, “in attesa” |

Per sapere se un utente A è amico di un utente B, bisognerà controllare se esiste una richiesta d’amicizia da parte di A (o B) nei confronti di B (o A) con l’attributo Stato pari a “accettata”.

Poiché un utente può inviare o ricevere richieste di amicizia da zero o più utenti, la cardinalità di questa relazione è (0,N) da entrambi i lati.

Cliente

Amicizia

(0,N)

(0,N)

Ricevente

Richiedente

Uno user può decidere di dividere i suoi amici in gruppi di utenti, le “cerchie”, in base agli interessi comuni.

Abbiamo creato l’entità Cerchia, collegata a Cliente tramite le due relazioni “Creazione”, che indica il creatore della cerchia, e “Appartenenza”, che indica chi appartiene a una determinata cerchia.

Gli attributi di Cerchia sono:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodCerchia | Codice identificativo della cerchia |
| Nome | Nome attribuito dal creatore alla cerchia |

Poiché un utente può creare nessuna oppure più cerchie, e può appartenere a nessuna o più cerchie, le cardinalità di Creazione e Appartenenza, dal lato di Cliente, sono entrambe (0,N). Una cerchia, invece, ha un unico creatore, quindi la cardinalità di Creazione – lato Cerchia – è (1,1). Inoltre, una cerchia può essere vuota oppure esserci più amici: la cardinalità dal lato di Cerchia di Appartenenza è quindi (0,N).

Cliente

Cerchia

Creazione

(0,N)

(0,N)

Appartenenza

(0,N)

(1,1)

All’interno del forum, gli utenti possono aprire nuovi thread nei quali discutere delle varie attività della palestra e condividere le proprie impressioni.

Il creatore del thread, imposta il “topic”, ossia carica il post che determina l’argomento dell’intero thread. Gli altri utenti possono rispondere a questo post, e possono valutare i post di risposta degli altri utenti con una scala di giudizio da una a cinque stelle.

Abbiamo quindi creato l’entità “Thread”, collegata a Cliente tramite la relazione “Creazione\_thread”.

L’entità Thread ha un solo attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| TitoloThread | Titolo del thread attribuito dal suo creatore, funge anche da identificatore |

Un utente può decidere di creare nessuno o più thread, mentre un thread è creato necessariamente da uno ed un solo user. La cardinalità di Creazione\_thread è quindi (0,N) dal lato di Cliente, e (1,1) dal lato di Thread.

Cliente

Thread

Creazione\_thread

(0,N)

(1,1)

Abbiamo inoltre creato l’entità “Post”, che comprende le due sottoclassi “Topic” e “Post\_risposta”. Post è collegata a Cliente tramite la relazione “Pubblicazione”, e a Thread tramite “Thread\_post”.

Post ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodPost | Codice identificativo |
| Testo | Contenuto del post |

Topic e Post\_risposta non aggiungono ulteriori attributi.

La relazione Pubblicazione ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Timestamp | Campo testuale che contiene data e ora della pubblicazione del post |

Un utente può pubblicare nessuno o più post, mentre un post è pubblicato sempre da un unico utente. La cardinalità di Pubblicazione è (0,N) dal lato di Cliente e (1,1) dal lato di Post.

Un post appartiene ad un unico thread, mentre un thread contiene almeno un post (il topic) e ne può contenere molti. La cardinalità di Thread\_post è (1,1) dal lato di Post, (1,N) dal lato di Thread.

Cliente

Post

Pubblicazione

(0,N)

(1,1)

Thread

Thread\_post

(1,N)

(1,1)

Topic

Post\_risposta

Un post può includere anche dei link a pagine web esterne. Abbiamo quindi creato l’entità “Collegamento”, che rappresenta l’associazione di un link ad un post. Collegamento è collegata a Post tramite la relazione “Inclusione\_link”.

Collegamento ha un unico attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| URL | Indirizzo web del link, insieme all’attributo esterno Post funge da identificatore dell’entità |

Poiché un post può avere nessuna o più istanze di inclusione di un link, la cardinalità di Inclusione\_link è (0,N) dal lato di Post. Un’istanza di Collegamento (ossia un abbinamento link-post) è relativa ad un solo post, quindi la cardinalità – lato Collegamento – è (1,1).

Post

Collegamento

Inclusione\_link

(0,N)

(1,1)

Gli utenti del forum possono valutare i post di risposta con un giudizio da una a cinque stelle. Abbiamo creato la relazione “Valutazione\_post” che unisce Cliente e Post\_risposta.

Valutazione\_post ha un attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Giudizio | Numero da 1 a 5 che esprime il giudizio dato ad un post di risposta da parte di un utente |

Un utente può valutare nessuno oppure più post, e un post può essere valutato da nessuno o più utenti: la cardinalità di Valutazione\_post è da entrambi i lati (0,N).

Cliente

Valutazione\_post

(0,N)

(0,N)

Post

Topic

Post\_risposta

Il sistema deve gestire il meccanismo delle “sfide”: esse vengono lanciate da un proponente, gli amici del proponente possono partecipare alla sfida e, insieme al proponente, valutano quotidianamente le proprie condizioni psico-fisiche raggiunte al termine dell’allenamento.

Viene creata l’entità “Sfida”, collegata a Cliente tramite le relazioni “Proposta” e “Partecipazione”. Viene anche aggiunta l’entità Giorno\_sfida, collegata a Sfida tramite la relazione “Composizione\_sfida”, e a Cliente tramite la relazione “Valutazione\_sfida”.

Sfida ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodSfida | Codice identificativo |
| Scopo | Obiettivo da raggiungere sotto forma di campo testuale |
| DataLancio | Data di lancio della sfida da parte del proponente |
| DataInizio | Data d’inizio sfida |
| DataFine | Data di fine sfida |

Giorno\_sfida ha un unico attributo:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| Giorno | Numero che indica un giorno della sfida. Insieme all’ attributo esterno Sfida è identificatore dell’entità |

La relazione Valutazione\_sfida ha due attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| GiudizioPsichico | Valore da 1 a 5 che rappresenta la valutazione data da un utente alla sua condizione psichica in un determinato giorno della sfida |
| GiudizioFisico | Valore da 1 a 5 che rappresenta la valutazione data da un utente alla sua condizione fisica in un determinato giorno della sfida |

Un utente può proporre zero o più sfide, mentre una sfida è proposta sempre da un unico utente. La relazione Proposta ha cardinalità (0,N) dal lato di Cliente e (1,1) dal lato di Sfida.

Un utente può partecipare a zero o più sfide e ad una sfida possono partecipare zero o più utenti. La relazione Partecipazione ha cardinalità (0,N) da entrambi i lati.

Una sfida è solitamente composta da più giorni, ma sicuramente da almeno uno. Un giorno di una sfida appartiene esclusivamente ad una sfida. La cardinalità di Composizione\_sfida è (1,N) dal lato di Sfida e (1,1) dal lato di Giorno\_sfida.

Un utente è tenuto a valutare quotidianamente le proprie prestazioni durante i vari giorni della sfida, ma potrebbe non farlo a causa di numerose motivazioni. La cardinalità di Valutazione\_sfida, dal lato di Cliente, ha dunque cardinalità (0,1). Un giorno di una sfida può essere valutato da nessun utente oppure da tutti i suoi partecipanti, in generale più di uno. La cardinalità lato Giorno\_sfida è quindi (0,N).

Cliente

Valutazione\_sfida

(0,N)

(1,1)

Sfida

Giorno\_sfida

Partecipazione

Proposta

Composizione\_sfida

(0,N)

(0,N)

(1,N)

(1,1)

(0,N)

(0,1)

Il proponente può aggiungere una scheda d’alimentazione correlata alla sfida. Abbiamo creato la relazione “Abbinamento\_alimentazione” che collega Sfida con Scheda\_alimentazione. Poiché l’aggiunta di una scheda d’alimentazione è facoltativa, la cardinalità lato Sfida è (0,1), mentre dal lato di Scheda\_alimentazione è (0,N), in quanto una scheda d’alimentazione può essere abbinata a zero o più sfide.

Sfida

Scheda\_alimentazione

Abbinamento\_alimentazione

(0,1)

(0,N)

Ad ogni giorno della sfida il proponente deve associare una scheda di allenamento dalla validità giornaliera, contenente gli esercizi da eseguire quel determinato giorno. Se si vuole applicare una stessa scheda di allenamento a tutti i giorni della sfida, si dovrà utilizzare un’apposita opzione sul programma che gestisce il database.

Abbiamo dunque creato la relazione “Abbinamento\_scheda” che collega Giorno\_sfida con Scheda\_allenamento. Ogni giorno prevede una ed una sola scheda di Allenamento, quindi la cardinalità lato Giorno\_sfida è (1,1). Una scheda d’allenamento può essere abbinata a zero oppure a più giorni di una sfida o di altre sfide, quindi la cardinalità lato Scheda\_allenamento è (0,N).

Giorno\_sfida

Scheda\_allenamento

Abbinamento\_scheda

(1,1)

(0,N)

### 2.2.4- Sviluppo inside-out dell’entità “Contratto”

Un cliente sottoscrive contratti con l’azienda. Abbiamo quindi creato l’entità “Contratto” con i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodContratto | Codice identificativo del contratto |
| DataSottoscrizione | Data di sottoscrizione del contratto da parte del Cliente |
| Durata | Durata della validità del contratto dal momento della sottoscrizione, espressa in mesi |
| Tipologia | Tipologia di contratto. Può essere “standard”, “personalizzato” o “multisede” |
| ModalitaPagamento | Modalità di pagamento del contratto. Può assumere i valori “rateizzato” o “pagamento unico” |
| Importo | Importo che il cliente si impegna a pagare al momento della firma del contratto. E’ un attributo derivato vedi regole derivazione) |

L’entità Contratto è collegata a Cliente tramite la relazione “Sottoscrizione”.

Poiché un Cliente è tale se sottoscrive almeno un contratto, e un cliente può sottoscrivere più contratti, la cardinalità di Sottoscrizione dalla parte di Cliente è (1,N) mentre dal lato di Contratto, poiché un contratto può essere sottoscritto da un unico cliente, la cardinalità è (1,1).

Cliente

Contratto

Sottoscrizione

(1,N)

(1,1)

Ogni contratto è depositato in un unico centro. Si crea quindi la relazione “Deposito” che collega Contratto a Cliente.

Poiché un centro può avere o meno contratti depositati al suo interno, la cardinalità di Deposito dal lato di Centro è (0,N), mentre dal lato di Contratto è (1,1).

Contratto

Centro

Deposito

(1,1)

(1,1)

Un contratto contiene un’offerta, scelta dal cliente, che gli permette di accedere a determinate sale, piscine e corsi. In base alla tipologia del contratto, le offerte possono essere le tre standard “silver”, “gold” oppure “platinum”, oppure personalizzate dal cliente in base alle sue esigenze. Le offerte standard non sono uguali per tutti i centri dell’azienda: alcuni centri, infatti, potrebbero avere sale specializzate non presenti in altri centri. Ad esempio, se il centro di Milano possiede una sala per la sauna, non presente nelle altre sedi, la dirigenza potrebbe decidere di integrare l’accesso alla sauna nel contratto personalizzato “platinum”. Inoltre, l’offerta permette l’accesso unicamente al centro alla quale è legata.

Per i contratti multisede, infatti, si prevede la sottoscrizione di un unico contratto con la scelta di tre offerte relative a centri diversi: l’importo totale che il firmatario di un contratto multisede dovrà versare all’azienda, sarà la somma dell’importo totale delle singole offerte alla quale sarà applicato uno speciale sconto (ad es. il 20%). In questo modo, un cliente può decidere di applicare offerte diverse ai vari centri scelti, siano esse offerte standard o personalizzate.

Per rappresentare questi concetti, abbiamo creato l’entità Offerta, collegandola a Contratto tramite la relazione “Inclusione\_offerta”, e a Corso, Luogo\_allenamento e Centro tramite le relazioni “Offerta\_corso” e “Offerta\_luogo” e “Offerta\_centro”.

L’entità Offerta ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodOfferta | Codice identificativo |
| TipoOfferta | Tipologia dell’offerta. Può assumere i valori “silver”, “gold” e “platinum” (offerte standard) oppure “personalizzata” |
| MaxIngressi | Numero massimo di ingressi permessi settimanalmente dall’offerta |
| AccessiPiscine | Numero massimo di accessi alle piscine permessi mensilmente dall’offerta |
| CostoMensile | Costo mensile dell’offerta |

Un’offerta può dunque essere inclusa in più contratti, oppure potrebbe non essere stata scelta da nessun cliente. Un contratto, invece, può contenere un’offerta (in caso di contratto standard o personalizzato) oppure più offerte (relative al massimo a tre centri, in caso di contratto multisede) ma sempre e comunque almeno una. La cardinalità di Inclusione\_offerta è quindi (1,N) dal lato di Contratto e (0,N) dal lato di Offerta.

Ciascuna offerta permette l’accesso ad uno ed un solo centro, mentre un centro potrebbe essere incluso in molte oppure nessuna offerta. La cardinalità di Offerta\_centro è quindi (1,1) dal lato di Offerta e (0,N) dal lato di Centro.

Inoltre, un’offerta può permettere zero oppure più accessi a corsi e luoghi d’allenamento e, allo stesso tempo, un corso può essere incluso in nessuna o più offerte, così come un luogo d’allenamento. Le cardinalità di Offerta\_corso e Offerta\_luogo sono entrambe (0,N) da entrambi i lati.

Contratto

Offerta

Inclusione\_offerta

(1,N)

(0,N)

Centro

Corso

Luogo\_allenamento

Offerta\_centro

Offerta\_corso

Offerta\_luogo

(1,1)

(0,N)

(0,N)

(0,N)

(0,N)

(0,N)

Se la modalità di pagamento del contratto è “rateizzato”, il cliente si impegna a versare mensilmente delle rate a un istituto finanziario convenzionato con l’azienda. Viene creata l’entità “Rata”, collegata a Cliente tramite “Rateizzazione\_cliente” e a Istituto\_finanziario tramite “Rateizzazione”. Quando si inserisce nel database un nuovo contratto con ModalitaPagamento uguale a “rateizzato”, un trigger inserirà automaticamente nel database tutte le rate associate.

L’entità Rata ha i seguenti attributi:

|  |  |
| --- | --- |
| Attributo | Descrizione |
| CodRata | Codice identificativo |
| Importo | Importo della rata |
| DataScadenza | Data di scadenza |
| Stato | Stato di pagamento della rata. Può assumere i valori “eseguito”, “non ancora dovuto”, “scaduto”. |

Poiché un cliente che non ha scelto la modalità di pagamento rateizzata non ha rate da pagare, la cardinalità di Rateizzazione\_cliente è (0,N) dal lato di Cliente. Una rata, invece, è relativa ad uno ed un solo cliente, quindi la cardinalità dal lato di Rata è (1,1).

Una rata, inoltre, è emessa da un unico istituto finanziario; un istituto, però, può emettere zero o più rate, quindi la cardinalità di Rateizzazione è (1,1) dal lato di Rata, (0,N) dal lato di Istituto\_finanziario.

Istituto\_finanziario

Rata

Rateizzazione

(0,N)

(1,1)

Cliente

Rateizzazione\_cliente

(1,1)

(0,N)

## 2.3- Ristrutturazione del diagramma ER

Poiché nel diagramma ER finora progettato sono state inserite delle generalizzazione, ci occuperemo in questa fase della progettazione della loro eliminazione.

Abbiamo optato per le generalizzazioni “Dipendente” e “Luogo\_allenamento” per il mantenimento delle entità, collegate alla vecchia classe madre con associazioni.

Per la generalizzazione “Post”, abbiamo invece optato per il collasso verso l’alto, eliminando dunque le entità “Topic” e “Post\_risposta” e aggiungendo un attributo booleano a Post.

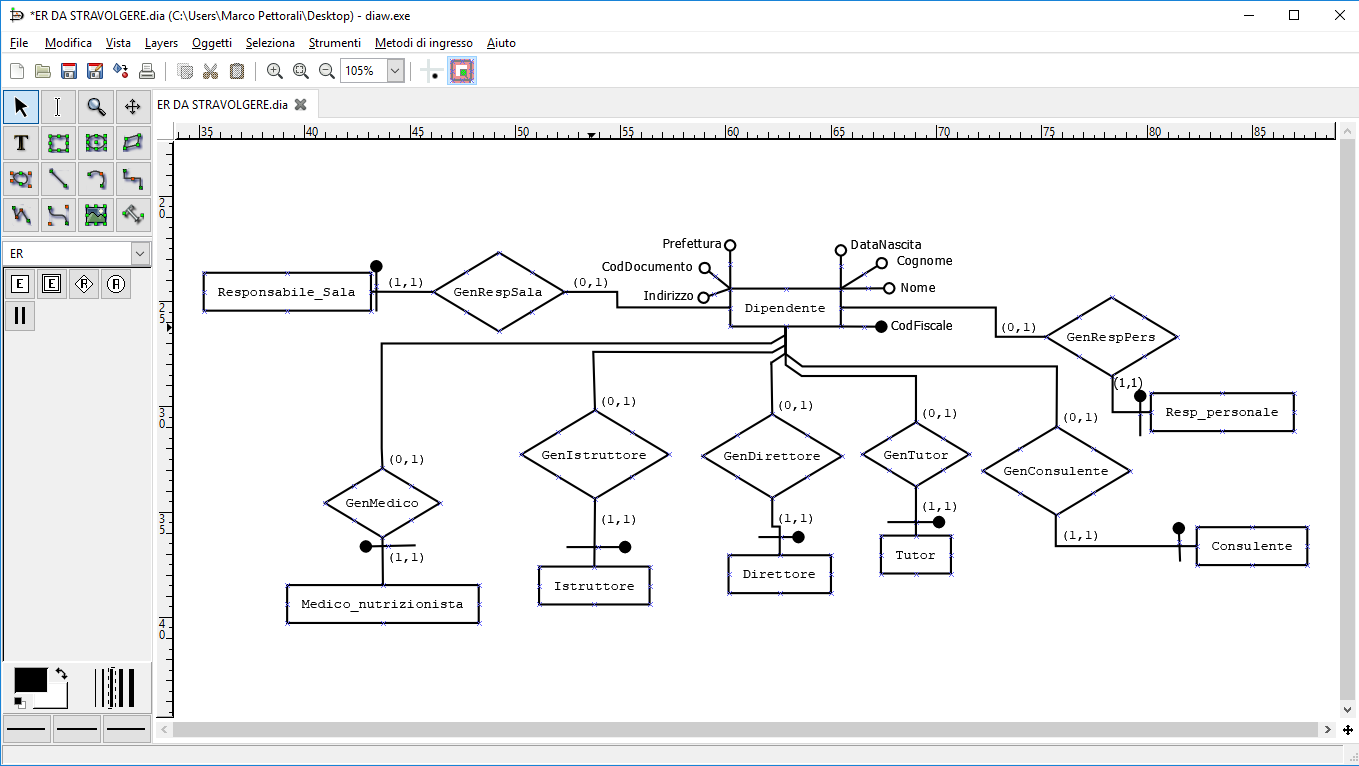
Ecco in dettaglio le varie fasi della ristrutturazione:

### 2.3.1- Eliminazione della generalizzazione “Dipendente”

Come accennato in precedenza, abbiamo optato per il mantenimento delle sotto-entità “Responsabile\_sala”, “Istruttore”, “Direttore”, “Tutor”, “Consulente”, “Medico\_nutrizionista” e “Resp\_Personale”, collegandole a Dipendente tramite, rispettivamente, le relazioni “GenRespSala”, “GenIstruttore”, “GenDirettore”, “GenTutor”, “GenConsulente”, “GenMedico” e “GenRespPers”.

In questo modo, è possibile preservare tutte le relazioni associate ai diversi ruoli che può assumere un dipendente: con la tecnica del collasso verso il basso, infatti, avremmo appesantito le sotto-entità aggiungendo gli attributi anagrafici del dipendente, e se avessimo voluto una lista di tutti i dipendenti, sarebbe stato molto laborioso trovarli tutti, in quanto sarebbe mancata un elemento che unisce tutti i vari ruoli. Con il collasso verso l’alto, invece, avremmo dovuto aggiungere degli attributi sull’entità Dipendente che ne indicassero il ruolo. Questo, però, non dipende soltanto dal dipendente, ma anche dal centro in cui lavora, e non avremmo potuto dunque gestire questa situazione.

La porzione di diagramma ER ristrutturato è questa:



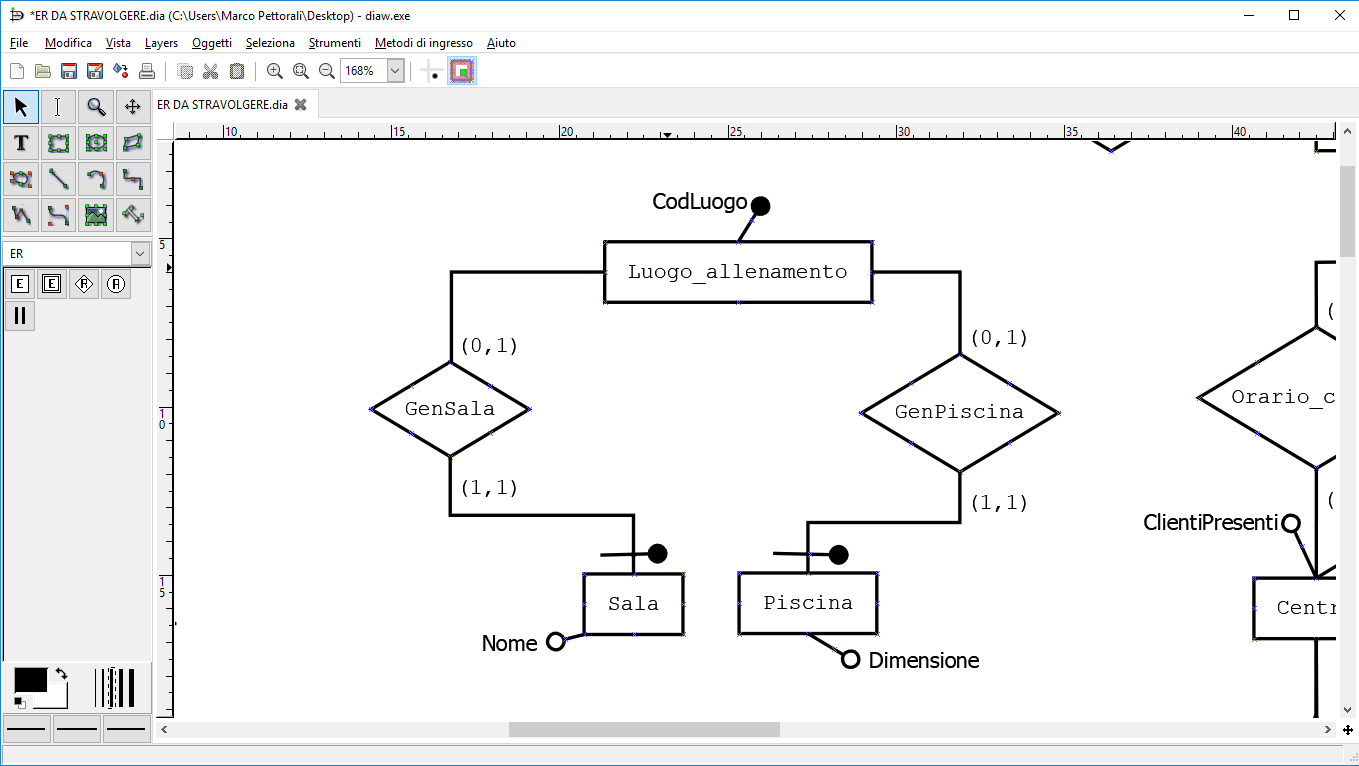
Le entità aggiunte allo schema sono:

|  |  |  |
| --- | --- | --- |
| Entità | Attributi | Descrizione |
| Responsabile\_sala | / | La chiave è l’attributo esterno “CodDipendente” |
| Istruttore | / | La chiave è l’attributo esterno “CodDipendente” |
| Direttore | / | La chiave è l’attributo esterno “CodDipendente” |
| Tutor | / | La chiave è l’attributo esterno “CodDipendente” |
| Consulente | / | La chiave è l’attributo esterno “CodDipendente” |
| Medico\_nutrizionista | / | La chiave è l’attributo esterno “CodDipendente” |
| Resp\_personale | / | La chiave è l’attributo esterno “CodDipendente” |

### 2.3.2- Eliminazione della generalizzazione “Luogo\_allenamento”

Per preservare le relazioni già esistenti con Luogo\_allenamento, abbiamo deciso di utilizzare la strategia del mantenimento delle entità con la conseguente instaurazione di nuove relazioni tra le sottoclassi e la classe madre. Le entità Sala e Piscina mantengono i loro attributi, e la chiave è in entrambe l’attributo esterno “CodLuogo”.

La porzione di diagramma ER ristrutturato è questa:



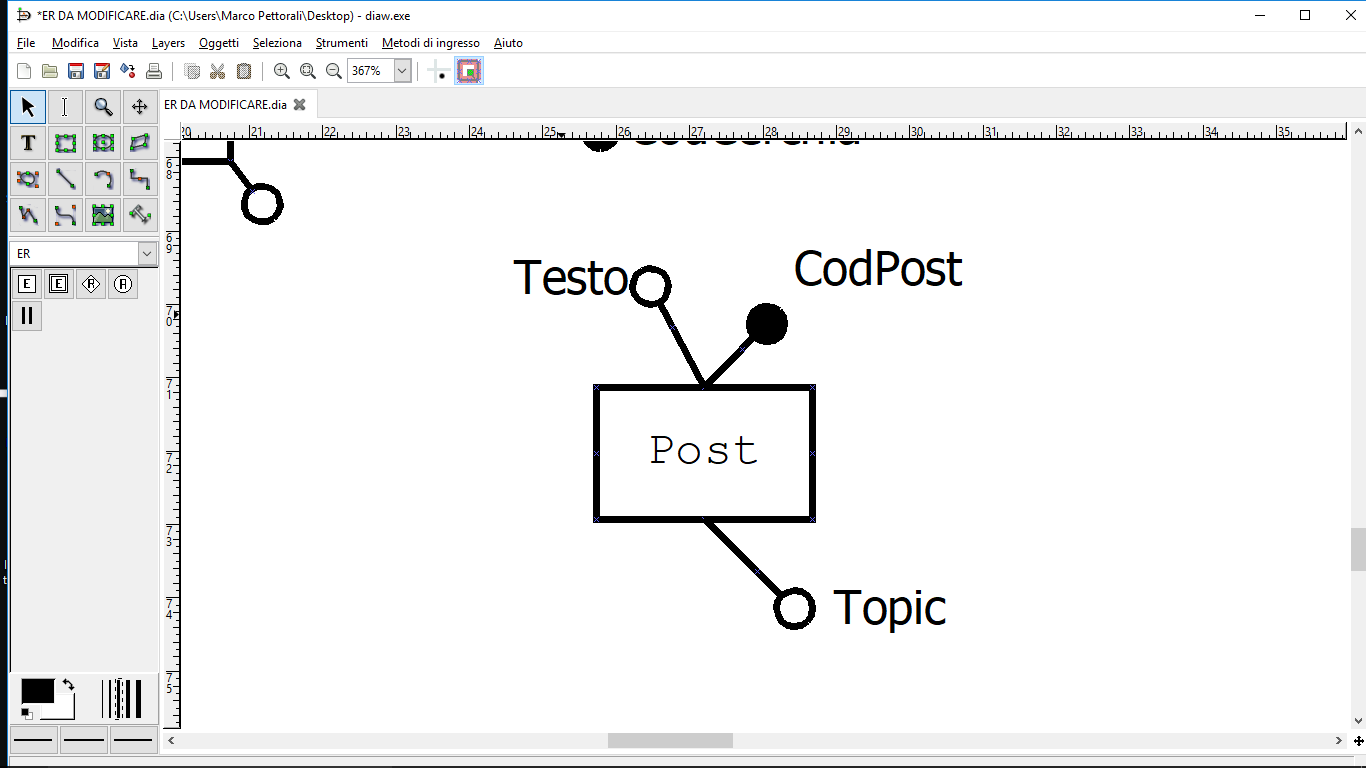
Le entità aggiunte allo schema sono:

|  |  |  |
| --- | --- | --- |
| Entità | Attributi | Descrizione |
| Sala | Nome | Nome della sala. La chiave è l’attributo esterno “CodLuogo” |
| Piscina | Dimensione | Dimensione in metri quadri della piscina. La chiave è l’attributo esterno “CodLuogo” |

### 2.3.3- Eliminazione della generalizzazione “Post”

Per eliminare questa generalizzazione, abbiamo deciso di collassare le sottoclassi verso la classe madre: un attributo booleano “Topic” indicherà, infatti, se il post è il topic del thread. In caso contrario, il post sarà un post di risposta, e potrà dunque essere valutato.

La porzione di diagramma ER ristrutturato è questa:



Le entità modificate dello schema sono:

|  |  |  |
| --- | --- | --- |
| Entità | Attributi | Descrizione |
| Post | CodPost | Codice identificativo del post |
|  | Testo | Contenuto del post |
|  | Topic | Attributo booleano che indica se il post è topic o post di risposta |

# Operazioni sui dati

## 3.1- Compilazione tavole dei volumi

### 3.1.1- Entità

L’azienda di fitness è proprietaria di più centri fitness su territorio nazionale, supponiamo con una media di un centro in ogni provincia italiana. Poiché le province sono 93, l’azienda gestisce circa 100 centri.

Supponiamo che ogni centro sia aperto tutti i giorni per due fasce orarie distinte una mattutina e una pomeridiana, ad eccezione della domenica che comprende un’unica fascia oraria. L’entità Orario\_servizio ha dunque 6\*2 + 1 = 13 istanze per centro. In totale sono dunque 13 \* 100 centri = 1 300 istanze.

Ogni centro offre in media 30 tipologie diverse di corsi, ognuna affrontata in tre livelli di difficoltà diversi. 30 corsi \* 3 livelli di difficoltà \* 100 centri = 9 000 corsi totali

Ogni centro ha in media 20 luoghi d’allenamento, tra sale e piscine, con una media di 18 sale e 2 piscine. In totale, nel database saranno memorizzati 20 \* 100 centri = 2 000 luoghi d’allenamento, di cui 1 800 sale e 200 piscine.

Ogni centro comprende in media 5 spogliatoi (ad esempio uno maschile ed uno femminile per due piani più uno spogliatoio per disabili). Gli spogliatoi totali saranno quindi 100 centri \* 5 = 500 spogliatoi.

Ogni spogliatoio contiene in media 200 armadietti, quindi in totale saranno memorizzati nel database 200 \* 500 spogliatoi = 100 000.

Alcune sale, come quelle nelle quali si praticano arti marziali, saranno sprovviste di attrezzi ginnici, mentre altre ne conterranno molte, considerando una media di 30 apparecchi a sala. Le sale con attrezzature sono in media la metà delle sale totali, quindi sono 9 per centro. Le attrezzature presenti in un centro sono dunque 9 \* 30 = 270 attrezzature ginniche. In tutti i centri ci saranno 100 centri \* 270 = 27 000 attrezzi ginnici.

Ogni attrezzo ginnico, in media, ha una regolazione: alcune non ne hanno (ad esempio la spalliera o il quadro svedese), altre ne hanno più di una (tapis-roulant, spinners). Poiché le attrezzature sono 27 000, anche le regolazioni sono 27 000.

In media, una palestra è grande circa 2000 mq. Statisticamente, le palestre italiane accolgono indicativamente 1,5 clienti a metro quadro. 1.5\*2000 = 3000 clienti che in media vengono accolti da una palestra. Poiché i centri sono 100 \* 3000 clienti = 300 000 clienti totali.

Ogni cliente è tale perché ha firmato almeno un contratto. Poiché nel database sono memorizzati tutti i contratti effettuati dai vari clienti, ogni cliente potrebbe risultare firmatario di più contratti oramai scaduti. Supponendo che quando un cliente lascia la palestra, i suoi dati all’interno del database vengono cancellati, ci saranno almeno 300 000 contratti nel sistema più quelli dei clienti che erano iscritti gli anni passati, ma che hanno rinnovato il contratto. Supponiamo che essi siano la metà, 150 000, e che i restanti siano clienti che dopo un anno decidono di abbandonare la palestra. Nel database avremo quindi 300 000 + 150 000 = 450 000 contratti.

I contratti contengono delle offerte: esse sono standard o personalizzate. Le standard sono di 3 tipi diversi (silver, gold, platinum) e sono diverse da centro a centro. 100 centri \* 3 tipologie di offerta = 300 offerte standard.

Inoltre, alcuni clienti hanno deciso di usufruire di offerte personalizzate. Supponiamo essi siano 10 000, complessivamente, poiché normalmente i contratti standard offrono già una discreta libertà di movimento nelle strutture a prezzi vantaggiosi. Abbiamo dunque 10 000 offerte personalizzate memorizzate nel database. Complessivamente, l’entità Offerta avrà circa 10 300 istanze.

Molti clienti, inoltre, avranno scelto la modalità di pagamento rateizzato, dato che è la soluzione più comoda per pagare l’abbonamento ai servizi dell’azienda: i clienti che rateizzano l’importo saranno circa 200 000. Poiché le rate, solitamente, hanno scadenza mensile e il periodo medio di iscrizione a una palestra è di 6 mesi, le rate presenti nel database saranno 200 000 clienti \* 6 rate per cliente = 1 200 000 rate.

Per permettere la rateizzazione dell’importo, l’azienda si appoggia a istituti finanziari. Supponiamo che, per venire incontro alle esigenze dei clienti, l’azienda abbia convenzioni con almeno un istituto finanziario per ogni regione, quindi con 20 istituti finanziari.

Un cliente può scegliere uno tra 3 tipi di scopo. Scopo avrà quindi sempre 3 istanze.

Le categorie di muscoli che i clienti possono scegliere di allenare sono circa 10. Muscolo ha circa 10 istanze.

Un cliente, insieme al suo medico nutrizionista, deve effettuare delle misurazioni con cadenza settimanale. 24 settimane (considerando i 6 mesi come durata media di un contratto) \* 300 000 = 7 200 000 schede di misurazione vengono annualmente aggiunte nel database.

I clienti che intendono seguire una dieta, effettuano delle visite con i propri medici nutrizionisti per fare degli accertamenti. Solitamente, un cliente viene visitato una volta alla settimana. In questo modo, contestualmente alla visita, il medico riesce a compilare anche le schede di misurazione del cliente, ottimizzando così i tempi. Considerando ancora una volta una media di permanenza nelle strutture dell’azienda di 6 mesi (quindi 24 settimane), il cliente verrà visitato annualmente 24 volte. 150 000 clienti \* 24 = 3 600 000 visite saranno memorizzate annualmente.

Ogni cliente può ricevere schede di alimentazione. Supponendo che i clienti che vogliano seguire una dieta siano la metà del totale, e che ciascuno di loro riceve annualmente due schede di alimentazione, saranno memorizzate nel database 300 000 schede d’alimentazione ogni anno.

Ogni centro fa anche degli ordini per acquistare integratori alimentari da abbinare alle schede d’alimentazione. Supponendo che la metà delle schede d’alimentazione abbiano abbinato un integratore, ci sarà bisogno di ordinare almeno 150 000 integratori all’anno, suddivisi su tutto il territorio nazionale. Ogni centro dovrà quindi ordinare circa 150 000/100 = 1 500 integratori all’anno. Si supporre che l’ordine venga fatto a stock di 500 integratori; saranno necessari quindi 3 ordini annuali per ogni centro, quindi un totale di 300 ordini saranno memorizzati annualmente nel database.

Supponendo che quasi tutti gli integratori (90%) che vengono ordinati siano acquistati dai clienti, e sapendo che per ogni ordine ci sono circa 10 integratori, si può stimare che le istanze di Acquisto siano meno di 0,9 \* 10 \* 300 = 2 700/anno.

Si suppone inoltre che il catalogo di integratori contenuto nel database comprenda circa 150 prodotti e che i fornitori ai quali l’azienda si rivolge per la consegna dei prodotti siano uno per ogni tre regioni italiane, quindi in totale 7.

Nel database sono memorizzate inoltre alcune diete che i medici assegnano ai propri clienti. Le diete più seguite nel mondo sono raccolte in una lista di 60 elementi, supponendo che nel database siano registrate altre diete meno conosciute, oppure diete create dal team di medici nutrizionisti dell’azienda, l’entità Dieta avrà circa 100 istanze.

L’entità Pasto, invece, raccoglie le ricette dei cibi che compongono la dieta. Le combinazioni, ovviamente, sono infinite. Un famoso sito web, però, offre un ricettario dietetico di 1000 elementi, per cui si suppone che anche nel database dell’azienda siano memorizzate un numero paragonabile di ricette.

Ogni cliente riceve in media una scheda d’allenamento diversa ogni 2 mesi. Considerando che un cliente va in palestra in media per 6 mesi l’anno, egli riceverà annualmente 3 schede. Annualmente, nel database si aggiungono 300 000 \* 3 = 900 000 schede d’allenamento.

Consideriamo inoltre che nel database siano memorizzati circa 300 esercizi diversi (100 tipologie ognuna con tre livelli di difficoltà, differenziati da durata o numero di ripetizioni diverse), di cui la metà anaerobici. Ogni esercizio anaerobico ha in media tre ripetizioni. L’entità Ripetizione avrà quindi circa 150 esercizi anaerobici \* 3 ripetizioni = 450 ripetizioni totali.

Ogni cliente è tenuto a rispettare la propria scheda d’allenamento. Un cliente, in media, va in palestra una volta ogni tre giorni. Poiché una scheda d’allenamento, in media, comprende 15 esercizi, ciascun cliente eseguirà quindi 15 esercizi ogni tre giorni. Dunque, statisticamente parlando, un cliente svolge 5 esercizi a giorno. L’entità Prestazione\_esercizio, che contiene i dati relativi alle prestazioni dei clienti mentre fanno esercizio, subirà un incremento giornaliero di 300.000 clienti \* 5 = 1 500 000. In un anno, considerando un periodo di attività del cliente di 6 mesi (quindi 180 giorni), saranno effettuati e registrati 1 500 000 \* 180 giorni = 270 000 000 esercizi.

Dunque, ogni cliente accede al centro ogni tre giorni. In media, ogni giorno accedono ai vari centri dell’azienda circa 100 000 persone. L’entità Accesso ha dunque circa 100 000 istanze giornaliere. In un anno, considerando un periodo di attività di 6 mesi (180 giorni) ci saranno 18 000 000 di accessi.

Dato che chi frequenta la palestra sarà quasi sicuramente iscritto ad almeno un social network, si suppone che i clienti utilizzino di rado il forum della palestra.

Un cliente, sul suo profilo social del forum dell’azienda, può scegliere alcune attività preferite. Considerando come attività ogni tipologia di corso e aggiungen-do altre attività non effettuate all’interno della struttura, ma sempre riferibili al mon-do sportivo, si può considerare un elenco di almeno 50 attività fra le quali l’utente può scegliere le sue preferite.

Un cliente posta sul forum dei messaggi. Considerando che una piccola parte di clienti sfrutterà il social network interno all’azienda (supponiamo 5 000 in totale), e supponendo che ognuno di essi pubblichi in media un post settimanalmente (magari per condividere le sue esperienze durante quella settimana), in un mese saranno presenti 5 000 \* 4 = 20 000 post. In un anno con una frequentazione statistica di sei mesi ci saranno dunque 20.000 \* 6 = 120 000 post.

Ogni post può contenere link. Considerando che, in media, il 10% dei post contengono almeno un link, e solitamente non più di uno, ci saranno circa 0,1 \* 120.000 = 12 000 istanze di Collegamento ogni anno.

Supponiamo, che dei 5 000 utenti del forum, la metà abbia lanciato almeno una sfida ai loro amici, possiamo supporre con una media di una all’anno. Ogni anno dunque l’entità Sfida raccoglierà circa 2 500 sfide.

Per motivi fisiologici, per quanto una persona si impegni, i primi risultati significativi di un allenamento mirato a migliorare una particolare parte del proprio corpo si ottengono dopo almeno due mesi. Una sfida, in media, avrà la durata di 2 mesi. Nel database saranno quindi memorizzate informazioni relative a 2 500 sfide annuali \* 60 = 150 000 giorni di varie sfide ogni anno.

Considerando che ad ogni sfida viene associato un thread sul forum, e supponendo che altri utenti del forum (5 utenti per centro) abbiano creato thread per condividere opinioni sulle attività sportive, nel database saranno memorizzati circa 3 000 thread.

Un utente avrà maggiormente come amici (sul social network aziendale) altri utenti suoi coetanei e concittadini. In media, possiamo supporre 15 amici per utente.

Possiamo quindi supporre che ogni cliente crei 2 cerchie per suddividere i suoi amici in base agli interessi comuni. In totale saranno quindi memorizzate nel database 5 000 utenti \* 2 = 10 000 cerchie.

In ogni centro si assume ci siano circa 50 dipendenti (considerando circa 20 istruttori, 10 responsabili di cui 8 delle sale e 2 del personale, 10 medici, 10 tutor e 2 consulenti e 1 direttore). 100 centri \* 50 dipendenti = 5 000 dipendenti totali, di cui 2.000 istruttori, 800 responsabili delle sale, 200 responsabili del personale, 1 000 medici , 1 000 tutor, 200 consulenti finanziari e 100 direttori.

Ogni dipendente lavora 6 giorni a settimana. Considerando che la palestra è aperta tutto l’anno tranne due mesi per la sosta estiva (ovvero per un totale di 300 giorni) e che ogni dipendente ha diritto ad alcuni giorni di ferie, si può supporre che un dipendente lavori effettivamente circa 250 giorni all’anno. Quindi l’entità Turnazione raccoglie circa 250 \* 5 000 dipendenti = 1 250 000 piani di turnazione.

Infine, l’entità Giorno, che raccoglie i 7 giorni della settimana, avrà appunto 7 istanze.

|  |  |
| --- | --- |
| Entità | Volume |
| Centro | 100 |
| Orario\_servizio | 1 300 |
| Corso | 9 000 |
| Luogo\_allenamento | 2 000 |
| Sala | 1 800 |
| Piscina | 200 |
| Spogliatoio | 500 |
| Armadietto | 100 000 |
| Attrezzatura | 27 000 |
| Regolazione | 27 000 |
| Cliente | 300 000 |
| Contratto | 450 000 |
| Offerta | 10 300 |
| Rata | 1 200 000 |
| Istituto\_finanziario | 20 |
| Scopo | 3 |
| Muscolo | 10 |
| Misurazione | 7 200 000/anno |
| Visita | 3 600 000/anno |
| Scheda\_alimentazione | 300 000/anno |
| Ordine | 300/anno |
| Acquisto | 2 700/anno |
| Integratore | 150 |
| Fornitore | 7 |
| Dieta | 100 |
| Pasto | 1 000 |
| Scheda\_allenamento | 900 000/anno |
| Esercizio | 300 |
| Ripetizione | 450 |
| Prestazione\_esercizio | 270 000 000/anno |
| Accesso | 18 000 000/anno |
| Attivita | 50 |
| Post | 120 000/anno |
| Sfida | 2 500/anno |
| Giorno\_sfida | 150 000/anno |
| Thread | 3 000 |
| Cerchia | 10 000 |
| Collegamento | 12 000/anno |
| Dipendente | 5 000 |
| Istruttore | 2 000 |
| Responsabile\_sala | 1 800 |
| Resp\_personale | 200 |
| Tutor | 1 000 |
| Medico\_nutrizionista | 1 000 |
| Consulente | 200 |
| Direttore | 100 |
| Turnazione | 1 250 000 |
| Giorno | 7 |

### 3.1.2- Relazioni

Ogni dipendente ha almeno un impiego in un centro. Poiché solo una piccola parte dei dipendenti lavora in più centri, supponiamo 1 000, la relazione Impiego conterrà circa 6 000 istanze.

Le relazioni GenIstruttore, GenRespSala, GenRespPers, GenDirettore, GenTutor, GenMedico e GenConsulente hanno un numero di istanze pari alle entità alle quali sono collegate tramite cardinalità (1,1), quindi hanno rispettivamente 2 000, 800, 200, 100, 1 000, 1 000 e 200 istanze.

Ad ogni istanza di Turnazione è associato un centro, un dipendente e un giorno. Le relazioni Turno\_centro, Turno e Turno\_giorno hanno dunque un numero di istanze pari a quelle dell’entità Turnazione, ossia 1 250 000.

Allo stesso modo, poiché ad ogni luogo d’allenamento e ad ogni spogliatoio corrisponde un unico centro, le istanze di Posizione\_luogo e Posizione\_spogliatoio saranno rispettivamente 2 000 e 500.

Poiché in totale ci sono circa 1 800 sale e 200 piscine, le istanze di GenSala e GenPiscina saranno anch’esse 1 800 per la prima e 200 per la seconda.

Ogni armadietto è contenuto in un unico spogliatoio. Posizione\_armadietto ha dunque un numero di istanze pari a quello dell’entità Armadietto, ossia 100 000.

Ogni attrezzo ginnico è contenuto in un unico luogo d’allenamento. Posizione\_attrezzatura ha dunque un numero d’istanze pari a quello dell’entità Attrezzatura, ovvero 27 000.

Ogni regolazione corrisponde ad un’unica attrezzatura. Ci saranno dunque 27 000 istanze di Regolazione\_attrezzatura.

Ad ogni accesso sono associati esattamente un centro, un cliente e un armadietto. Accesso\_centro, Accesso\_cliente e Accesso\_armadietto hanno quindi un numero di istanze uguale a quello di Accesso, ovvero 18 000 000/anno.

Ogni centro è diretto da un unico direttore. La relazione Direzione ha un numero di istanze pari a quello di Centro, ovvero 100.

Si può supporre che, nonostante la cardinalità di Convenzione sia (0,N) dal lato di Centro, tutti i centri abbiano almeno una convenzione con un istituto finanziario. Supponiamo quindi che ogni centro abbia in media una convenzione con almeno 3 istituti finanziari. Le istanze di convenzione saranno dunque 100 centri \* 3 = 300.

Le relazioni Orario\_giorno e Orario\_centro hanno un numero di istanze pari a quelle di Orario, quindi 1 300.

Ad ogni ordine corrisponde un centro, un fornitore ed uno o più integratori. Le istanze di Ordine\_centro e Ordine\_fornitore sono dunque entrambe pari al numero di istanze di Ordine, ovvero 300/anno. In un ordine, spesso, vengono effettuati acquisti di integratori diversi, supponiamo con una media di 10 integratori a ordine. Le istanze di Ordine\_integratore sono dunque 300 ordini/anno \* 10 integratori = 3.000/anno.

Ogni sala è gestita da un responsabile. Poiché le sale in totale sono circa 1.800, Gestione\_sala avrà circa 1 800 istanze.

Ogni corso è gestito da un istruttore. Le istanze di Corso sono uguali in numero a quelle di Insegnamento, ovvero sono 9 000.

Ad ogni contratto è associato un consulente. Le istanze di Consulenza sono dunque pari a quelle di Contratto, ovvero 450 000.

Ogni cliente è associato ad un medico nutrizionista. Le istanze di Assegnazione\_medico sono dunque pari a quelle di Cliente, ovvero 300 000.

Ogni misurazione è associata ad un unico cliente, quindi la relazione Controllo ha un numero di istanze pari a quelle di Misurazione, ovvero 7 200 000/anno.

Ogni istanza di Visita corrisponde ad una della relazione Visita\_cliente ed una di Visita\_medico, quindi queste relazioni hanno un numero di istanze pari a 3 600 000 ad anno.

Ogni scheda di alimentazione è consigliata da un solo medico nutrizionista. Le istanze di Consiglio saranno in numero uguali a quelle di Scheda\_alimentazione, ovvero 300 000/anno.

Ogni scheda di alimentazione contiene una sola dieta. Composizione\_S\_alim avrà quindi un numero di istanze pari a quelle di Scheda\_alimentazione, ovvero 300.000/anno.

Ogni scheda d’alimentazione è abbinata ad un solo cliente. Alimentazione\_cliente avrà quindi un numero di istanze pari a quelle di Scheda\_alimentazione, ovvero 300 000/anno.

Ogni dieta è composta da almeno un pasto. Normalmente, in una dieta, ci sono più pasti da alternare settimanalmente. Supponiamo che ogni giorno si possa scegliere uno tra quattro primi piatti e uno tra quattro secondi piatti. Complessivamente, Composizione\_dieta avrà 7 giorni \* 8 piatti al giorno per dieta \* 100 diete = 5 600 istanze.

Come abbiamo supposto in precedenza, soltanto metà delle schede di alimentazione hanno associato un integratore alimentare. Le istanze di Abbinamento sono dunque 150 000/anno.

Ogni integratore è commercializzato da almeno un fornitore. Supponiamo che tutti i fornitori forniscono l’80% degli integratori, e il restante 20% e affidato in media a due fornitori per integratore. Gli integratori commercializzati da tutti i fornitori sono dunque 0,8 \* 150 = 120 integratori. Le istanze di Commercializzazione sono dunque circa 120 \* 7 + 30 \* 2 = 1 740.

Le relazioni Acquisto\_cli e Acquisto\_int hanno un numero di istanze pari a quelle di Acquisto, quindi sono 2 700/anno.

Ogni contratto è sottoscritto da un unico cliente ed è depositato in un unico centro. Le istanze di Sottoscrizione e di Deposito saranno in numero pari a quelle di Contratto, ossia 450 000.

Ad ogni contratto corrisponde una o tre offerte, in caso di scelta di contratto multisede. Supponendo che a scegliere il contratto multisede sia il 5% dei clienti, ci saranno 0,95 \* 300 000 contratti = 285 000 istanze + 0,05 \* 300 000 contratti \* 3 offerte = 330 000 istanze complessive di Inclusione\_offerta.

Ogni offerta permette l’accesso ad un unico centro, quindi le istanze di Offerta\_centro sono uguali in numero a quelle di Offerta, ossia 10 300.

Supponiamo che le offerte silver permettono l’accesso all’80% dei corsi, le gold e le personalizzate al 90% e le platinum e le al 100%. Offerta\_corso avrà dunque: (100 offerte silver \* 0,8 \* 9 000 corsi) + (10 100 offerte gold e personalizzate \* 0,9 \* 9 000 corsi) + (100 offerte platinum \* 1 \* 9 000 corsi) = 834 300 000 istanze.

Supponiamo che le offerte silver e le personalizzate permettono l’accesso al 50% dei luoghi d’allenamento, le gold al 70% e le platinum e le al 90%. Offerta\_luogo avrà dunque: (10 100 offerte silver e personalizzate\* 0,5 \* 2 000 luoghi) + (100 offerte gold \* 0,7 \* 2 000 luoghi) + (100 offerte platinum \* 0.9 \* 2 000 luoghi) = 10 420 000 istanze.

Ad ogni rata corrisponde un istituto di riferimento e un cliente. Le istanze di Rateizzazione e di Rateizzazione\_cliente saranno dunque pari a quelle di Rata, ovvero 1 200 000 ciascuna.

Ogni cliente può iscriversi ai corsi che la palestra offre. Supponendo che la metà dei clienti seguano almeno un corso, e in media due corsi, Iscrizione avrà 150.000 clienti \* 2 corsi a cliente = 300 000 istanze.

Ogni corso, in media, ha una pianificazione giornaliera ad esclusione del giorno di riposo. Le istanze di Pianificazione\_corso sono quindi 9 000 corsi \* 6 giorni = 54 000.

Ad ogni cliente viene assegnato un tutor. Assegnazione avrà dunque 300 000 istanze.

Ad ogni scheda d’allenamento corrisponde un cliente e un tutor. Le relazioni Assegnazione\_scheda e Consegna\_scheda hanno dunque un numero di istanze pari a quelle di Scheda\_allenamento, ovvero 900 000/anno.

Come supposto in precedenza, ogni scheda è composta da 15 esercizi. La relazione Composizione avrà dunque 900 000 schede/anno \* 15 esercizi = 13.500.000 istanze/anno.

La cardinalità di Regolazione\_esercizio lato Esercizio è (1,N), quindi ogni esercizio prevede l’uso di almeno una regolazione e quindi almeno una macchina. Supponiamo che un quarto degli esercizi preveda l’utilizzo di almeno due regolazioni. Le istanze di Regolazione\_esercizio sono dunque 0.75 \* 300 esercizi \* 1 regolazione + 0.25 \* 300 esercizi \* 2 regolazioni = 375 istanze.

Ogni ripetizione è associata ad un solo esercizio. Le istanze di Ripetizione\_es sono dunque pari a quelle di Ripetizione, ovvero 450.

Ogni ripetizione può sfruttare attrezzature ulteriori a quelle che prevede l’esercizio. Supponiamo che solamente l’ultima ripetizione di ogni esercizio anaerobico le preveda. Gli esercizi anaerobici, come stimato in precedenza, sono la metà di tutti gli esercizi, ovvero 150: ci saranno dunque 150 istanze di Regolazione\_rip nel database.

Ogni prestazione è riferita ad un solo esercizio e ad un solo cliente. Le istanze di Esercizio\_reale e Prestazione\_cliente sono in numero uguali a quelle di Prestazione\_esercizio, ovvero 270 000 000/anno.

Nel database vengono memorizzate le informazioni delle regolazioni impostate dai clienti. Poiché vengono memorizzate le informazioni di ogni esercizio che prevede l’uso di regolazioni, ed essi sono la metà di tutti gli esercizi, supponendo che vengano svolti il 75% di esercizi di anaerobici e il restante di aerobici, Regolazione\_reale avrà circa 0,75 \* 270 000 000 = 202 500 000 istanze/anno.

Ogni cliente deve scegliere uno scopo al momento dell’iscrizione. La relazione Scelta\_scopo ha dunque un numero di istanze pari a quelle di Cliente, ossia 300 000.

Supponendo che chi ha scelto lo scopo Potenziamento muscolare sia un terzo di tutti i clienti, ad un terzo dei clienti è chiesto di scegliere almeno un muscolo da potenziare. Supponiamo che in media i clienti scelgano 2 muscoli: la relazione Target ha dunque 100 000 \* 2 = 200 000 istanze.

Ogni utente del forum sceglie un gruppo di attività di interesse. Come già supposto in precedenza, gli utenti effettivi del forum sono circa 5 000. Ogni cliente sceglie in media 5 attività. La relazione Interesse ha dunque 5 000 \* 5 = 25 000 istanze.

Come supposto in precedenza, ogni utente ha in media 15 amici. Le istanze di Amicizia sono dunque circa 5 000 \* 15 = 75 000.

Ogni cerchia è creata da un unico utente. Le istanze di Creazione sono dunque uguali in numero alle istanze di Cerchia, ovvero 10 000.

Poiché ogni cliente ha in media 15 amici e crea 2 cerchie, ogni cerchia avrà mediamente 8 persone. In totale, la relazione Appartenenza avrà 8 \* 10 000 cerchie = 80 000 istanze.

Ogni sfida ha un proponente. La relazione Proposta ha quindi un numero di istanze pari a quelle di Sfida, ovvero 2 500/anno.

Ad ogni sfida possono partecipare soltanto gli amici del proponente, supponiamo un terzo. Ad ogni sfida parteciperanno così 5 persone più il proponente. In totale, la relazione Partecipazione avrà 2 500 sfide/anno \* 6 persone per sfida = 15.000 istanze/anno.

Ogni sfida può avere abbinata una scheda d’alimentazione, supponiamo metà delle sfide. Nel database ci saranno quindi 1 250 istanze/anno di Abbinamento\_alimentazione.

Ogni sfida è composta da almeno un giorno. Poiché in media una sfida dura 2 mesi, quindi 60 giorni, ci saranno 60 \* 2 500 sfide/anno = 150 000 istanze di Composizione\_sfida nel database.

Ogni giorno della sfida ha abbinata una scheda d’allenamento. Le istanze di Abbinamento\_scheda sono dunque, in numero, uguali a quelle di Giorno\_sfida, ovvero 150 000/anno.

Ogni giorno della sfida può essere valutato dai suoi partecipanti. Poiché i partecipanti ad una sfida sono in media 6, supponendo che due terzi dei partecipanti voti effettivamente le proprie prestazioni, per ogni giorno della sfida ci saranno 4 valutazioni. La relazione Valutazione\_sfida avrà quindi circa 150 000/anno \* 4 = 600.000 istanze.

Ogni post è pubblicato da un solo utente ed è incluso in un solo thread. Le istanze di Pubblicazione e Appartenenza\_thread sono quindi pari al numero di istanze di Post, ovvero 120 000/anno.

Ogni thread è creato da un unico utente, quindi le istanze della relazione Creazione\_thread sono pari a quelle di Thread, ovvero 3 000.

Un post di risposta può essere valutato dagli utenti. Supponendo che un terzo dei post riceva almeno un giudizio, e in media almeno 2 giudizi, sapendo che i post di risposta sono indicativamente 117 000 post di risposta (120 000 post complessivi a cui vanno sottratti i topic che sono pari al numero di thread, ovvero 3 000), si può concludere che le istanze di Valutazione\_post sono circa 0,3 \* 117 000 post/anno \* 2 = 70 200/anno.

Ogni abbinamento link-post è relativo ad un unico post, quindi le istanze di Inclusione\_link sono pari a quelle di Collegamento, ovvero 12 000/anno.

|  |  |
| --- | --- |
| Relazione | Volume |
| Impiego | 6 000 |
| GenIstruttore | 2 000 |
| GenRespSala | 800 |
| GenRespPers | 200 |
| GenDirettore | 100 |
| GenTutor | 1 000 |
| GenMedico | 1 000 |
| GenConsulente | 200 |
| Turno\_centro | 1 250 000 |
| Turno | 1 250 000 |
| Turno\_giorno | 1 250 000 |
| Posizione\_luogo | 2 000 |
| GenSala | 1 800 |
| GenPiscina | 200 |
| Posizione\_spogliatoio | 500 |
| Posizione\_armadietto | 100 000 |
| Posizione\_attrezzatura | 27 000 |
| Accesso\_centro | 18 000 000/anno |
| Accesso\_cliente | 18 000 000/anno |
| Accesso\_armadietto | 18 000 000/anno |
| Direzione | 100 |
| Convenzione | 300 |
| Orario\_giorno | 1 300 |
| Oraio\_centro | 1 300 |
| Regolazione\_attrezzatura | 27 000 |
| Ordine\_integratore | 3 000/anno |
| Gestione\_sala | 1 800 |
| Insegnamento | 9 000 |
| Consulenza | 450 000 |
| Assegnazione\_medico | 300 000 |
| Controllo | 7 200 000/anno |
| Visita\_cliente | 3 600 000/anno |
| Visita\_medico | 3 600 000/anno |
| Consiglio | 300 000/anno |
| Composizione\_S\_alim | 300 000/anno |
| Alimentazione\_cliente | 300 000/anno |
| Composizione\_dieta | 5 600 |
| Abbinamento | 150 000/anno |
| Commercializzazione | 1 740 |
| Acquisto\_cli | 2 700/anno |
| Acquisto\_int | 2 700/anno |
| Sottoscrizione | 450 000 |
| Deposito | 450 000 |
| Inclusione\_offerta | 330 000 |
| Offerta\_centro | 10 300 |
| Offerta\_corso | 834 300 000 |
| Offerta\_luogo | 10 420 000 |
| Rateizzazione | 1 200 000 |
| Rateizzazione\_cliente | 1 200 000 |
| Iscrizione | 300 000 |
| Pianificazione\_corso | 54 000 |
| Assegnazione | 300 000 |
| Assegnazione\_scheda | 900 000/anno |
| Consegna\_scheda | 900 000/anno |
| Composizione | 13 500 000/anno |
| Regolazione\_esercizio | 375 |
| Ripetizione\_es | 450 |
| Regolazione\_rip | 150 |
| Prestazione\_cliente | 270 000 000/anno |
| Esercizio\_reale | 270 000 000/anno |
| Regolazione\_reale | 202 500 000/anno |
| Scelta\_scopo | 300 000 |
| Target | 200 000 |
| Interesse | 25 000 |
| Amicizia | 75 000 |
| Cerchia | 10 000 |
| Appartenenza | 80 000 |
| Proposta | 2 500/anno |
| Partecipazione | 15 000/anno |
| Abbinamento\_alimentazione | 1 250/anno |
| Composizione\_sfida | 150 000/anno |
| Abbinamento\_scheda | 150 000/anno |
| Valutazione\_sfida | 600 000/anno |
| Pubblicazione | 120 000/anno |
| Appartenenza\_thread | 120 000/anno |
| Creazione\_thread | 3 000 |
| Valutazione\_post | 70 200/anno |
| Inclusione\_link | 12 000/anno |

### 3.1.3- Frequenza giornaliera delle operazioni significative

Poiché ciascun cliente esegue una scheda d’allenamento diversa ogni 2 mesi, e considerando che la durata media di un contratto è di 6 mesi, vengono annualmente distribuite dai tutor ai clienti dell’azienda 900 000 schede d’allenamento. Ogni giorno, dunque, si aggiungeranno in media 900 000 / 365 ≈ 2 500 schede.

L’operazione 1 (inserimento di una scheda di allenamento) sarà eseguita con una frequenza di circa 2 500 volte al giorno.

Può essere estremamente comodo per gli istruttori sapere chi sono gli iscritti al proprio corso, in modo tale da poter impostare al meglio il corso in base all’età dei clienti e della loro preparazione atletica. Abbiamo quindi pensato di automatizzare l’operazione che elenca i partecipanti ad un determinato corso. Supponendo che ogni istruttore esegui questa operazione settimanalmente per ogni suo corso, in una settimana l’operazione viene eseguita 9 000 volte, con una media quotidiana di 9 000/7 ≈ 1 300 volte.

L’operazione 2 (elencare i partecipanti a un determinato corso) sarà eseguita con una frequenza di circa 1 300 volte al giorno.

L’operazione 3 è una funzionalità di reporting che può essere utile per evidenziare eventuali problematiche relative alla mancanza di attrezzature oppure alla cattiva progettazione delle schede di allenamento. Infatti, si vuole sapere quali esercizi stanno svolgendo i clienti di un centro in un determinato momento e quanti clienti lo stanno svolgendo: se, ad esempio, moltissimi clienti stanno eseguendo contemporaneamente lo stesso esercizio, e questa situazione si verifica di frequente, potrebbe essere necessario comprare nuove attrezzature che permettono lo svolgimento di quel tale esercizio, oppure consigliare ai tutor di compilare schede più variegate.

Supponendo che questa operazione venga eseguita nelle fasce di massima affluenza (tipicamente tra le 17 e le 21) ogni mezz’ora, quotidianamente un centro effettua quest’operazione 10 volte. In totale, questa operazione è eseguita 100 centri \* 10 = 1 000 volte al giorno.

Il dispositivo RFID assegna al cliente un armadietto nel momento in cui attraversa il tornello d’ingresso. Per visualizzare quali armadietti sono liberi in un determinato centro, si esegue l’operazione 4 (visualizzare tutti gli armadietti liberi di un centro).

Questa operazione viene eseguita ad ogni accesso di un cliente: in media un cliente accede al centro ogni tre giorni, quindi ogni giorno accedono ai vari centri dell’azienda circa 100 000 persone.

L’operazione 4 viene dunque eseguita 100 000 volte al giorno.

L’operazione 5 (Indicare il numero di visite fatte da un cliente con il suo medico nutrizionista per la scheda di alimentazione attuale), invece, è utile per intuire se un cliente ha necessità di cambiare medico curante. Se infatti egli preferisce essere visitato da un medico che non è quello assegnatogli al momento dell’iscrizione, il cliente eseguirà poche visite col suo medico nutrizionista. Supponendo che questa operazione venga eseguita ogni mese per tutti i clienti, in totale si avrà una frequenza giornaliera di 300 000/30 = 10 000 volte.

L’operazione 6 serve per automatizzare la scelta della frequenza delle visite di un paziente, impostando come valore la frequenza media delle visite dall’inizio della scheda d’alimentazione. Ovvero se un paziente, dall’inizio della scheda d’alimentazione un mese fa è stato visitato da quel medico soltanto 3 volte, l’operazione imposta nel database l’attributo FrequenzaVisite dell’entità Scheda\_alimentazione pari a 10 giorni. Questa operazione viene eseguita dal medico ogni quattro visite di un paziente per calcolare automaticamente quando fare nuovamente la visita. Poiché, in media, ogni giorno vengono visitati 10 000 clienti, ma soltanto una volta su quattro viene eseguita l’operazione, la frequenza giornaliera di questa operazione è 10 000/ 4 = 2 500.

Poiché i centri possono accogliere contemporaneamente soltanto un nume-ro limitato di clienti, è necessario monitorare costantemente quanti clienti sono presenti in un determinato centro. L’operazione 7 (indicare il numero di clienti attualmente presenti in un determinato centro) viene dunque eseguita ad ogni accesso, quindi in media 100 000 volte al giorno.

L’operazione 8 (inserire un nuovo accesso ad un centro di un cliente se non si è già superato il valore di MaxClienti) è eseguita ogni volta che il dispositivo RFID segnala l’ingresso di un nuovo utente in un determinato centro. Poiché in media ogni giorno 100 000 clienti fanno accesso a un centro, l’operazione sarà eseguita 100 000 volte al giorno.

L’operazione 9 consiste nel valutare durata media e giudizio della prestazione medio di un esercizio, in modo tale da stabilire il giudizio di un cliente che ha appena svolto quell’esercizio. Questa operazione va in esecuzione ogni volta che un cliente termina un esercizio, quindi circa 740 000 volte al giorno.

|  |  |
| --- | --- |
| Operazione | Frequenza giornaliera |
| Op. 1 | 2 500/giorno |
| Op. 2 | 1 300/giorno |
| Op. 3 | 1 000/giorno |
| Op. 4 | 100 000/giorno |
| Op. 5 | 10 000/giorno |
| Op. 6 | 2 500/giorno |
| Op. 7 | 100 000/giorno |
| Op. 8 | 100 000/giorno |
| Op. 9 | 740 000/giorno |

## 3.2- Operazioni

### 3.2.1- Operazione 1

#### 3.2.1.1- Descrizione

Inserimento di una scheda di allenamento e aggiunta di un esercizio alla scheda

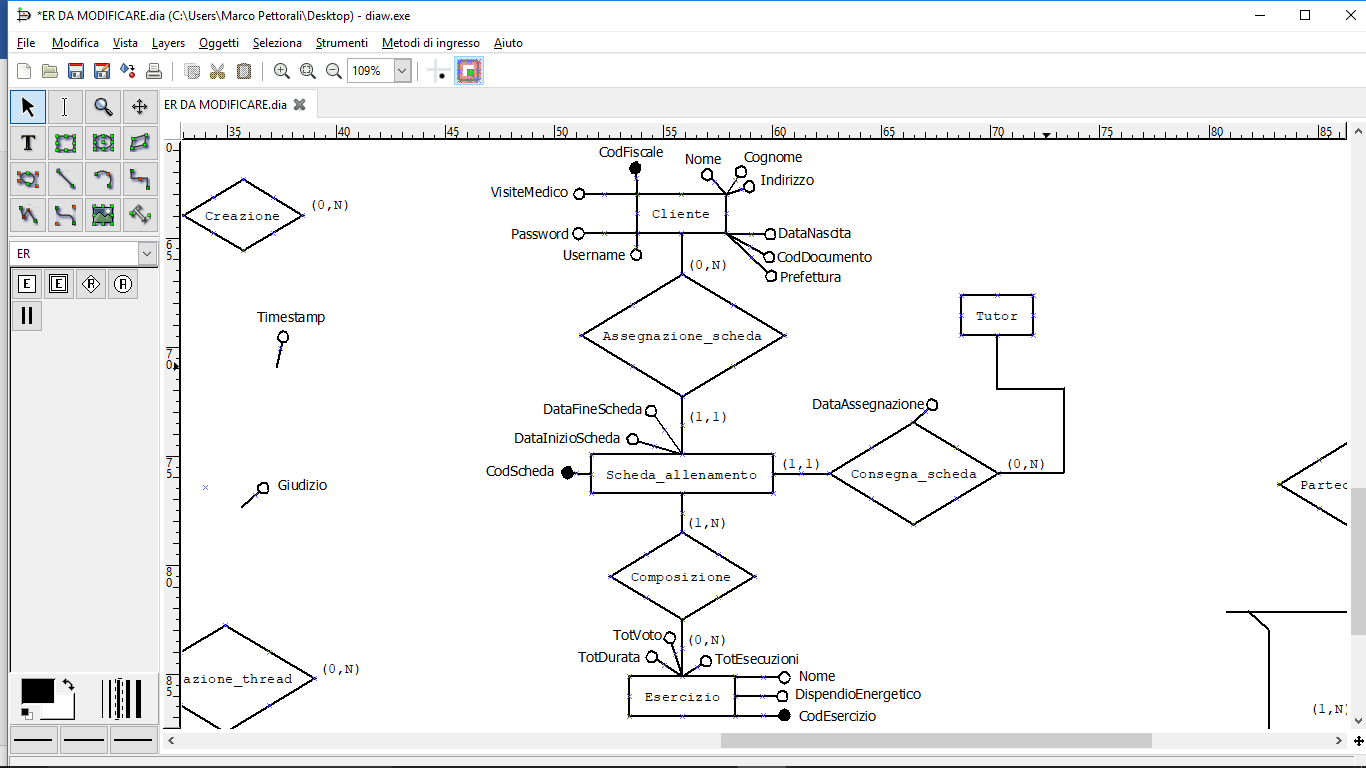
#### 3.2.1.2- Input

Cliente, data di assegnazione della scheda, data di inizio e fine validità, composizione della scheda (esercizi compresi nella scheda).

#### 3.2.1.3- Output

Nuova istanza di “Scheda\_allenamento”, di Assegnazione\_scheda, Consegna\_scheda e Composizione con i valori di input.

#### 3.2.1.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.1.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Scheda\_allenamento | 900 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Assegnazione\_scheda | 900 000/anno |
| Consegna\_scheda | 900 000/anno |
| Composizione | 13 500 000/anno |

#### 3.2.1.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | W | E | Scheda\_allenamento | Scrittura di un nuovo record in Scheda\_allenamento |
| 1 | W | R | Assegnazione\_scheda | Scrittura di un nuovo record in Assegnazione\_scheda |
| 1 | W | R | Consegna\_scheda | Scrittura di un nuovo record in Consegna\_scheda |
| 1 | W | R | Composizione | Scrittura di un nuovo record in Composizione\_scheda |

N.B. Nella colonna ‘Tipo operazione’ la lettera ‘R’ indica un’ operazione di lettura, mentre la lettera ‘W’ indica un’ operazione di scrittura.

### 3.2.2- Operazione 2

#### 3.2.2.1- Descrizione

Elencare i partecipanti a un determinato corso

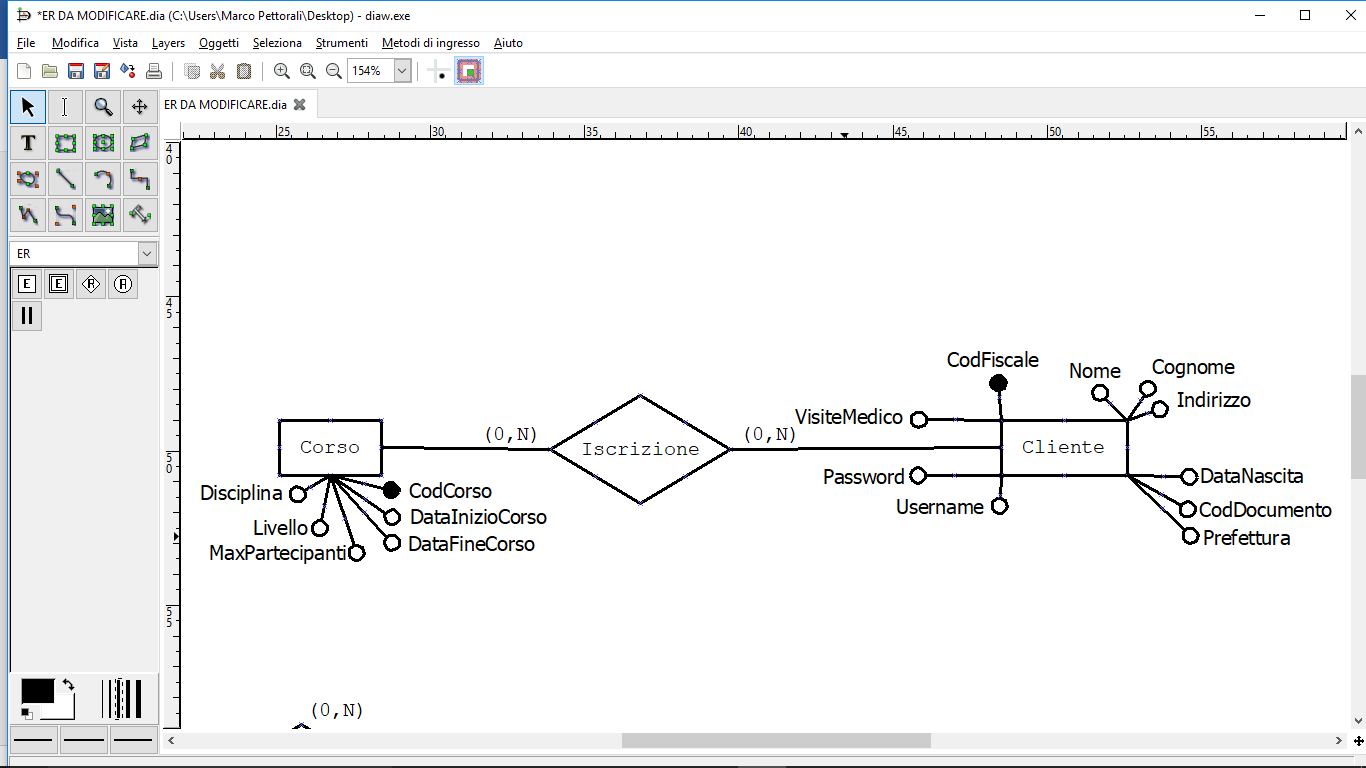
#### 3.2.2.2- Input

Codice identificativo del corso

#### 3.2.2.3- Output

Dati anagrafici dei partecipanti al corso

#### 3.2.2.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.2.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Corso | 9 000 |
| Cliente | 300 000 |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Iscrizione | 300 000 |

#### 3.2.2.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 30 | R | R | Iscrizione | In media, ogni cliente è iscritto ad un corso. Quindi ogni corso ha in media 300 000 clienti / 9 000 corsi ≈ 30 iscritti. |
| 30 | E | E | Cliente | Ogni istanza di Iscrizione corrisponde ad una di Cliente |

### 3.2.3- Operazione 3

#### 3.2.3.1- Descrizione

Indicare tutti gli esercizi che i clienti di un centro stanno eseguendo in un determinato momento e quanti sono

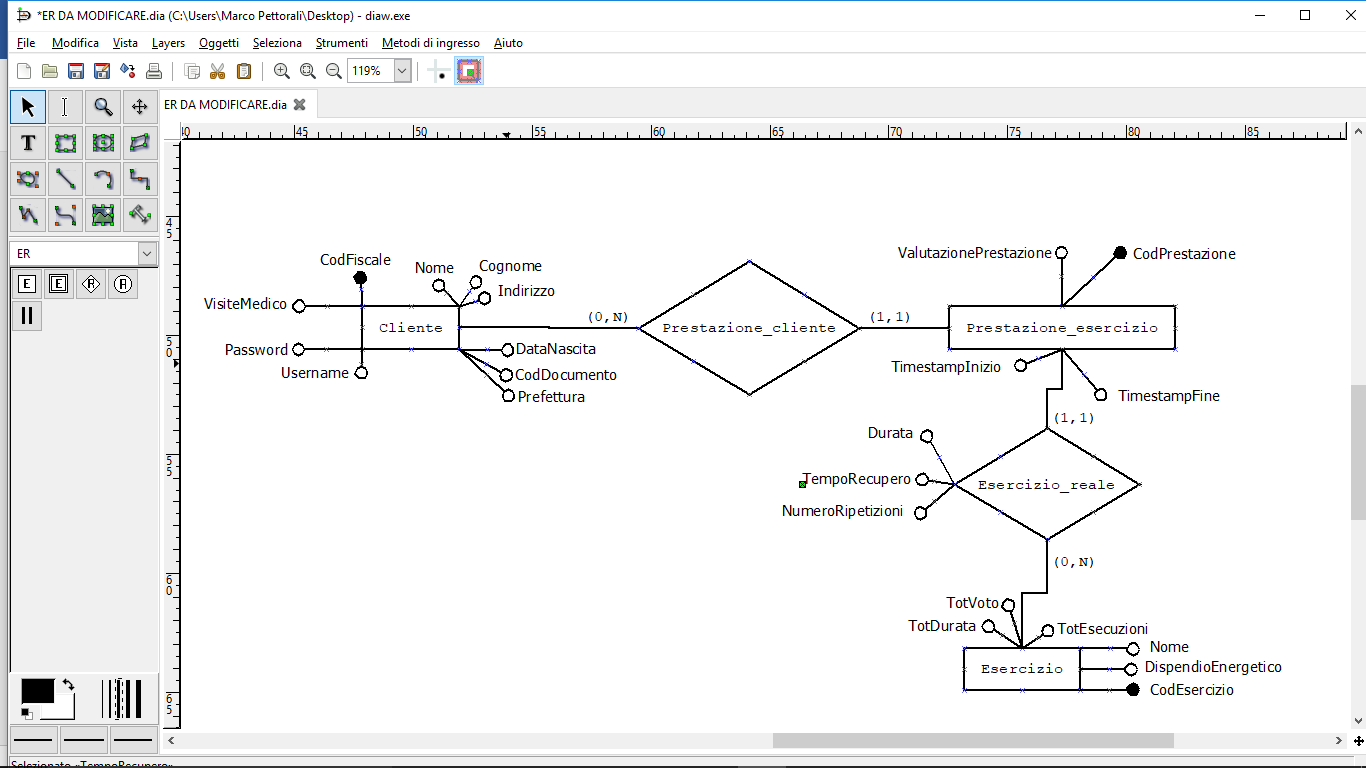
#### 3.2.3.2- Input

Dati contenuti in Prestazione\_esercizio, Esercizio\_reale e Prestazione\_cliente

#### 3.2.3.3- Output

Codice dell’esercizio, numero di esecutori dell’esercizio in un determinato momento

#### 3.2.3.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.3.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Prestazione\_esercizio | 270 000 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Esercizio\_reale | 270 000 000/anno |
| Prestazione\_cliente | 270 000 000/anno |

#### 3.2.3.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 200 | R | E | Prestazione\_esercizio | Ogni minuto vengono eseguiti circa 1 200 esercizi |
| 1 200 | R | R | Esercizio\_reale | Ogni istanza di Prestazione\_esercizio corrisponde ad una di Esercizio\_reale |
| 1 200 | R | R | Prestazione\_cliente | Ogni istanza di Prestazione\_esercizio corrisponde ad una di Prestazione\_cliente |

### 3.2.4- Operazione 4

#### 3.2.4.1- Descrizione

Visualizzare tutti gli armadietti liberi di un centro

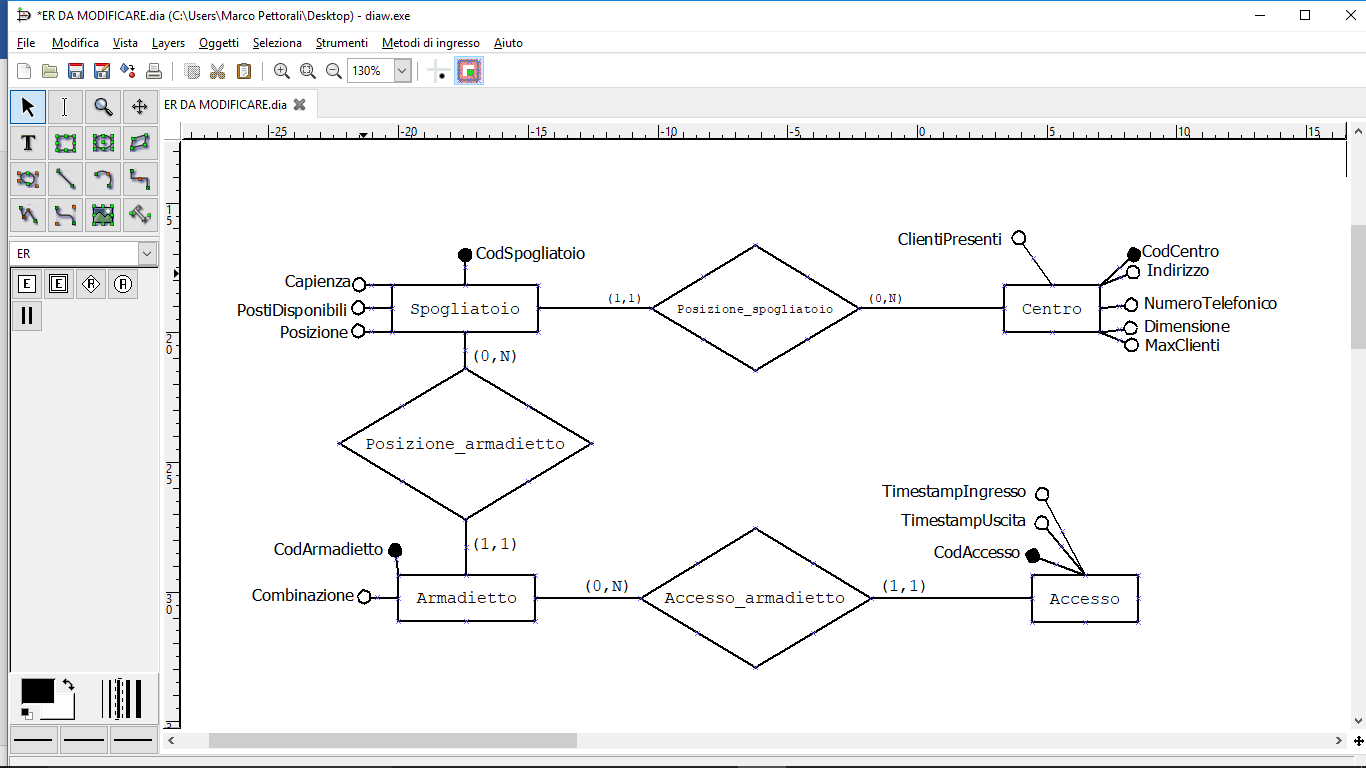
#### 3.2.4.2- Input

Codice centro, dati delle entità Accesso, Armadietto, Spogliatoio e delle relazioni, Accesso\_centro, Posizione\_armadietto e Accesso\_armadietto

#### 3.2.4.3- Output

Dati degli armadietti non associate ad un accesso e spogliatoi che contengono questi armadietti

#### 3.2.4.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.4.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Accesso | 18 000 000/anno |
| Spogliatoio | 500 |
| Armadietto | 100 000 |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Accesso\_centro | 18 000 000/anno |
| Accesso\_armadietto | 18 000 000/anno |
| Posizione\_armadietto | 100 000 |
| Posizione\_spogliatoio | 500 |

#### 3.2.4.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 900 | R | R | Accesso\_centro | Considero un periodo di permanenza nelle strutture di circa 2 ore e un orario di apertura di 10 ore. Contemporaneamente, in un centro, ci sono sempre circa 1.000 clienti, ai quali corrispondono 900 accessi. |
| 900 | R | E | Accesso | Accessi dei clienti a quel centro |
| 900 | R | R | Accesso\_armadietto | Ad ogni istanza di Accesso ne corrisponde una di Accesso\_armadietto. |
| 100 | R | E | Armadietto | Si prendono gli armadietti che non hanno un accesso collegato |
| 100 | R | R | Posizione\_armadietto | Ad ogni istanza di Armadietto ne corrisponde una di Posizione\_armadietto. |
| 100 | R | E | Spogliatoio | Spogliatoi con armadietti target |

### 3.2.5- Operazione 5

#### 3.2.5.1- Descrizione

Indicare il numero di visite fatte da un cliente con il suo medico nutrizionista per la scheda di alimentazione attuale.

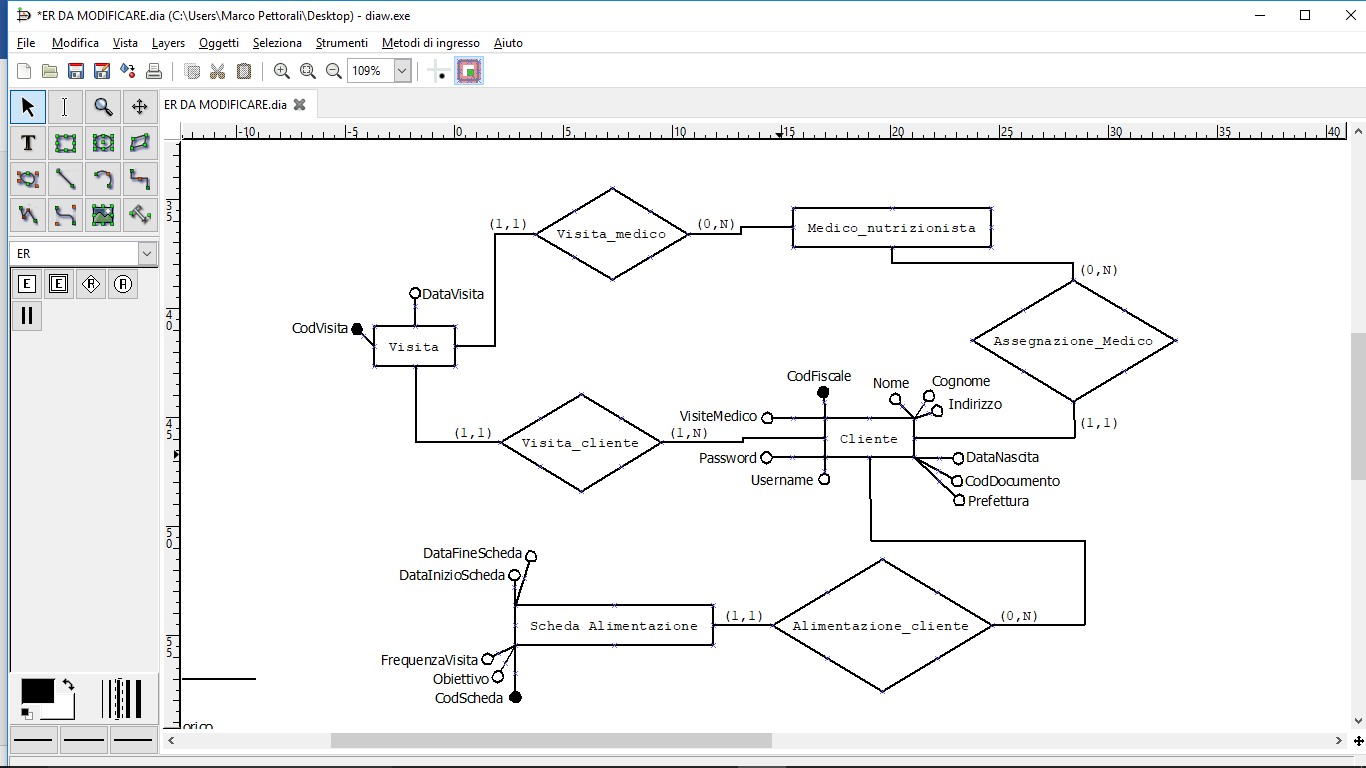
#### 3.2.5.2- Input

Codice del cliente

#### 3.2.5.3- Output

Numero di visite fatte dal cliente col suo medico

#### 3.2.5.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.5.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Medico\_nutrizionista | 1 000 |
| Scheda\_alimentazione | 300 000/anno |
| Visita | 3 600 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Visita\_cliente | 3 600 000/anno |
| Visita\_medico | 3 600 000/anno |
| Assegnazione\_medico | 300 000 |
| Alimentazione\_cliente | 300 000/anno |

#### 3.2.5.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Assegnazione\_medico | Ad ogni cliente viene associato un solo medico nutrizionista |
| 1 | R | E | Medico | Medico nutrizionista del cliente |
| 1 | R | R | Alimentazione\_cliente | Si trovano tutte le schede d’alimentazione del cliente |
| 1 | R | E | Scheda\_alimentazione | Dalla scheda d’alimentazione attuale, si legge la data di inizio per scartare le visite precedenti |
| 3 600 | R | R | Visita\_medico | Un medico, in media, effettua 3 600 visite all’anno |
| 24 | R | R | Visita\_cliente | Un cliente in media viene visitato 24 volta all’anno |
| 18 | R | E | Visita | Ogni cliente, in media, riceve una scheda all’anno. Supponiamo che, mediamente, il 75% delle visite totali sia effettuato dal proprio medico. |

### 3.2.6- Operazione 6

#### 3.2.6.1- Descrizione

Modificare l’attributo FrequenzaVisite di Scheda\_alimentazione con la media della frequenza delle visite effettuate da un cliente con il suo medico

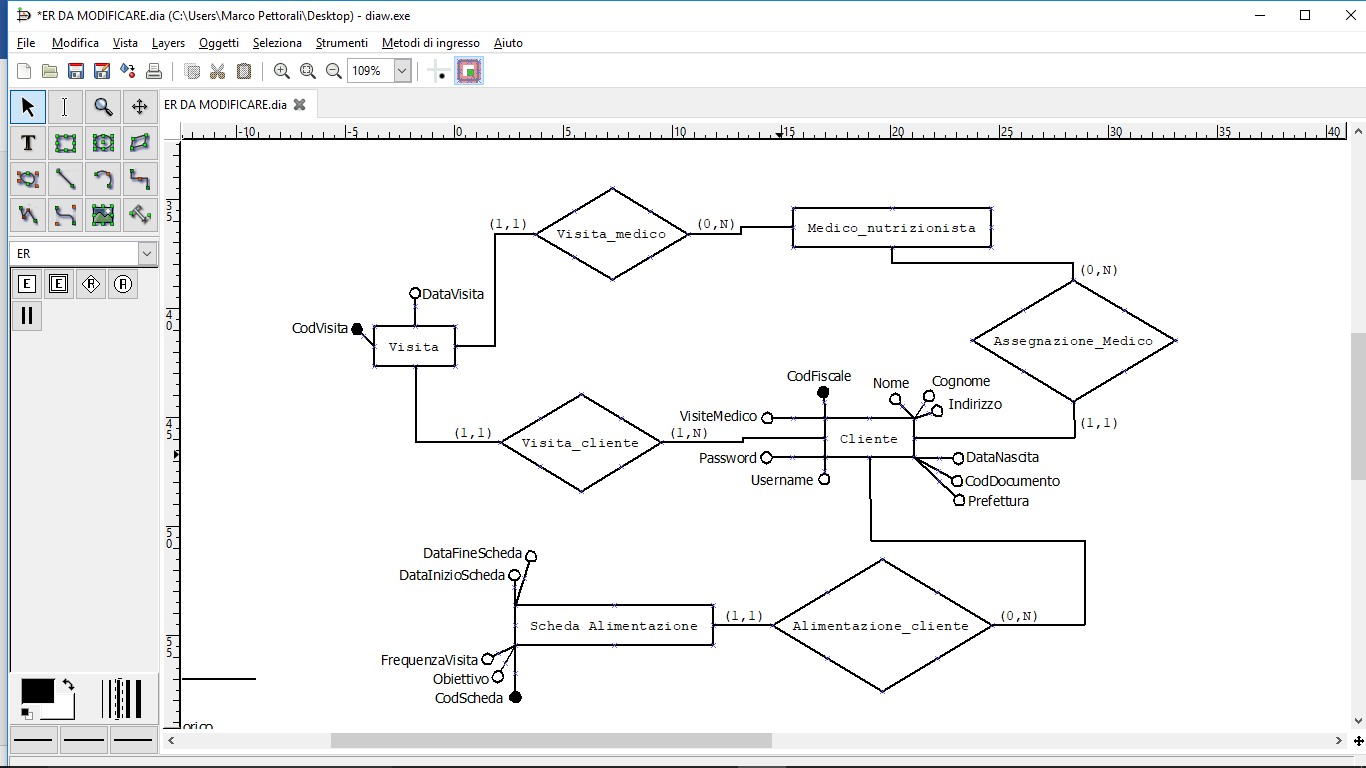
#### 3.2.6.2- Input

Codice di un cliente

#### 3.2.6.3- Output

Modifica dell’attributo FrequenzaVisite

#### 3.2.6.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.6.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Medico\_nutrizionista | 1 000 |
| Scheda\_alimentazione | 300 000/anno |
| Visita | 3 600 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Visita\_cliente | 3 600 000/anno |
| Visita\_medico | 3 600 000/anno |
| Assegnazione\_medico | 300 000 |
| Alimentazione\_cliente | 300 000/anno |
| Consiglio | 300 000/anno |

#### 3.2.6.6- Tavola degli accessi

NB. Si suppone che una scheda d’alimentazione sia assegnata ad un cliente soltanto dal proprio medico.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Assegnazione\_medico | Ad ogni cliente viene associato un solo medico nutrizionista |
| 1 | R | E | Medico | Medico nutrizionista del cliente |
| 1 | R | R | Alimentazione\_cliente | Si trovano tutte le schede d’alimentazione del cliente |
| 1 | R | E | Scheda\_alimentazione | Dalla scheda di alimentazione attuale, si legge la data di inizio per la durata in giorni della scheda di alimentazione fino al giorno dell’esecuzione dell’operazione. |
| 1 | R | E | Scheda\_alimentazione | Dalla scheda di alimentazione attuale, si legge la data di inizio per scartare le visite precedenti |
| 3 600 | R | R | Visita\_medico | Un medico, in media, effettua 3 600 visite all’anno |
| 24 | R | R | Visita\_cliente | Un cliente in media viene visitato 24 volta all’anno |
| 1 | W | R | Scheda\_alimentazione | Modifica dell’attributo FrequenzaVIsite |

### 3.2.7- Operazione 7

#### 3.2.7.1- Descrizione

Indicare il numero di clienti attualmente presenti in un determinato centro

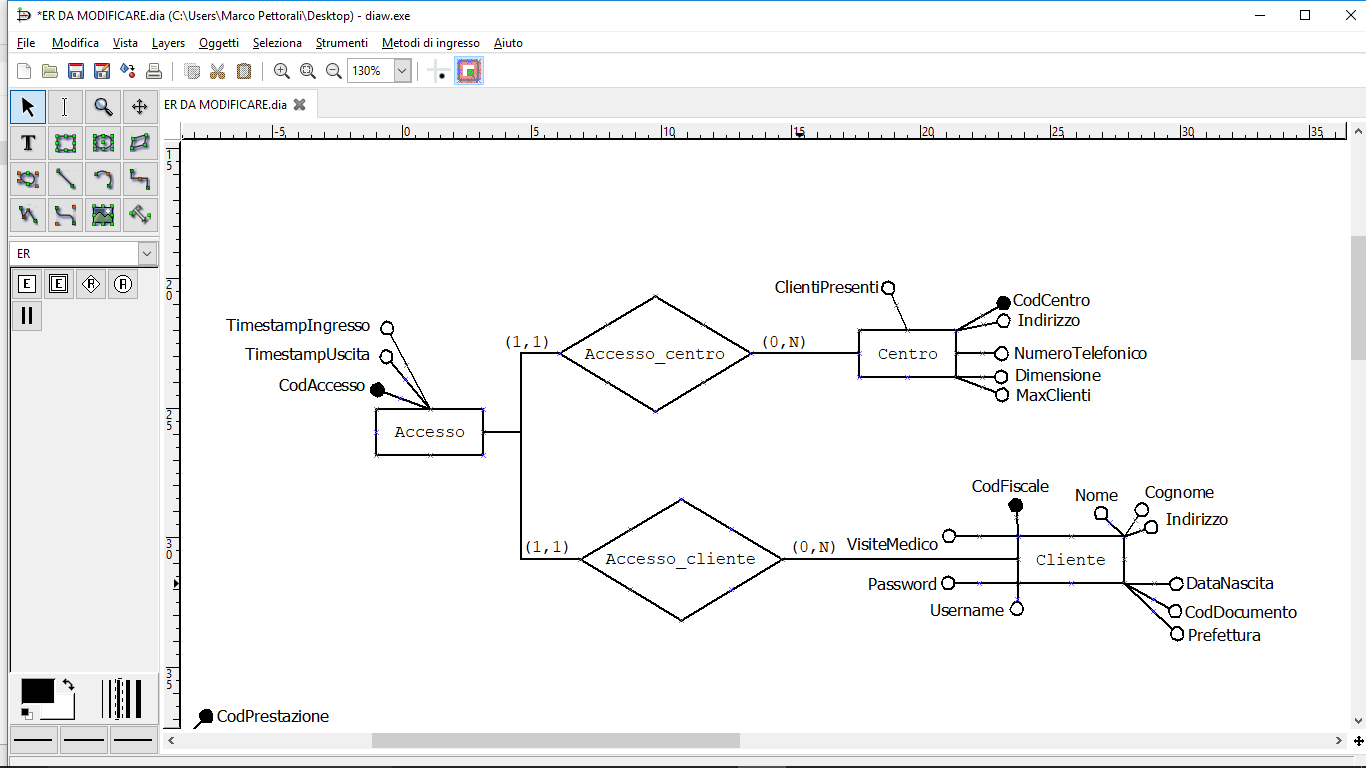
#### 3.2.7.2- Input

Codice del centro

#### 3.2.7.3- Output

Numero di clienti presenti nel centro

#### 3.2.7.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.7.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Accesso | 18 000 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Accesso\_centro | 18 000 000/anno |

#### 3.2.7.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 180 000 | R | R | Accesso\_centro | In media, in un centro acce-dono all’anno 18 000 000 /100=180.000 persone |
| 200 | R | E | Accesso | Accessi contemporanei medi in un centro |

### 3.2.8- Operazione 8

#### 3.2.8.1- Descrizione

Inserire un nuovo accesso ad un centro di un cliente se non si è già superato il valore di MaxClienti

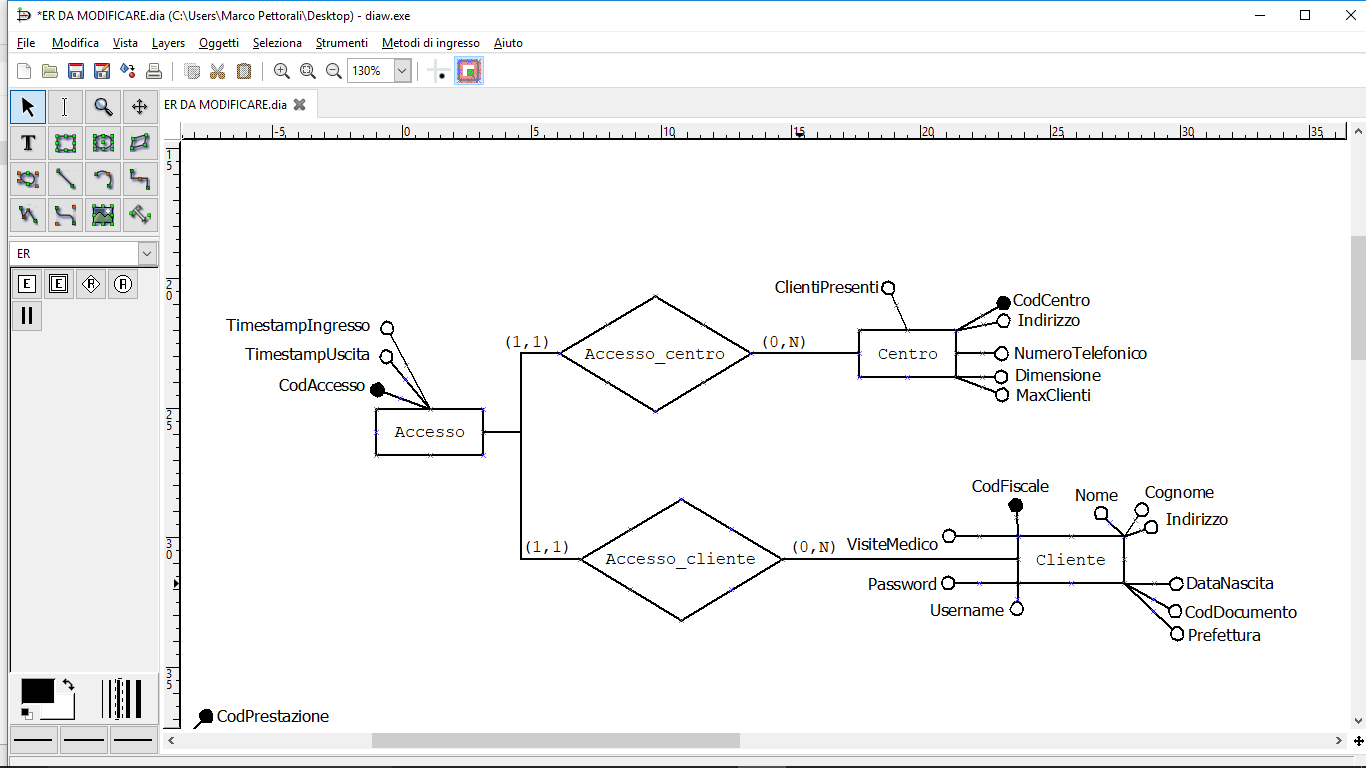
#### 3.2.8.2- Input

Codice di cliente e centro

#### 3.2.8.3- Output

Inserimento di un nuovo accesso se la condizione è rispettata

#### 3.2.8.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.8.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Centro | 100 |
| Accesso | 18 000 000/anno |

Relazioni:

|  |  |
| --- | --- |
| Relazione | Volume |
| Accesso\_cliente | 18 000 000/anno |

#### 3.2.8.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | E | Centro | Esiste un solo centro con il codice di input. Si legge il valore di MaxClienti |
| 180 000 | R | R | Accesso\_centro | In ogni centro accedono circa 180 000 clienti all’anno |
| 200 | R | E | Accesso | Accessi contemporanei medi in un centro |
| 1 | W | E | Accesso | Se l’operazione va a buon fine, si aggiunge il nuovo accesso |
| 1 | W | R | Accesso\_centro | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |
| 1 | W | R | Accesso\_cliente | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |
| 1 | W | R | Accesso\_armadietto | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |

### 3.2.9- Operazione 9

#### 3.2.9.1- Descrizione

Valutare durata media e giudizio della prestazione medio di un esercizio

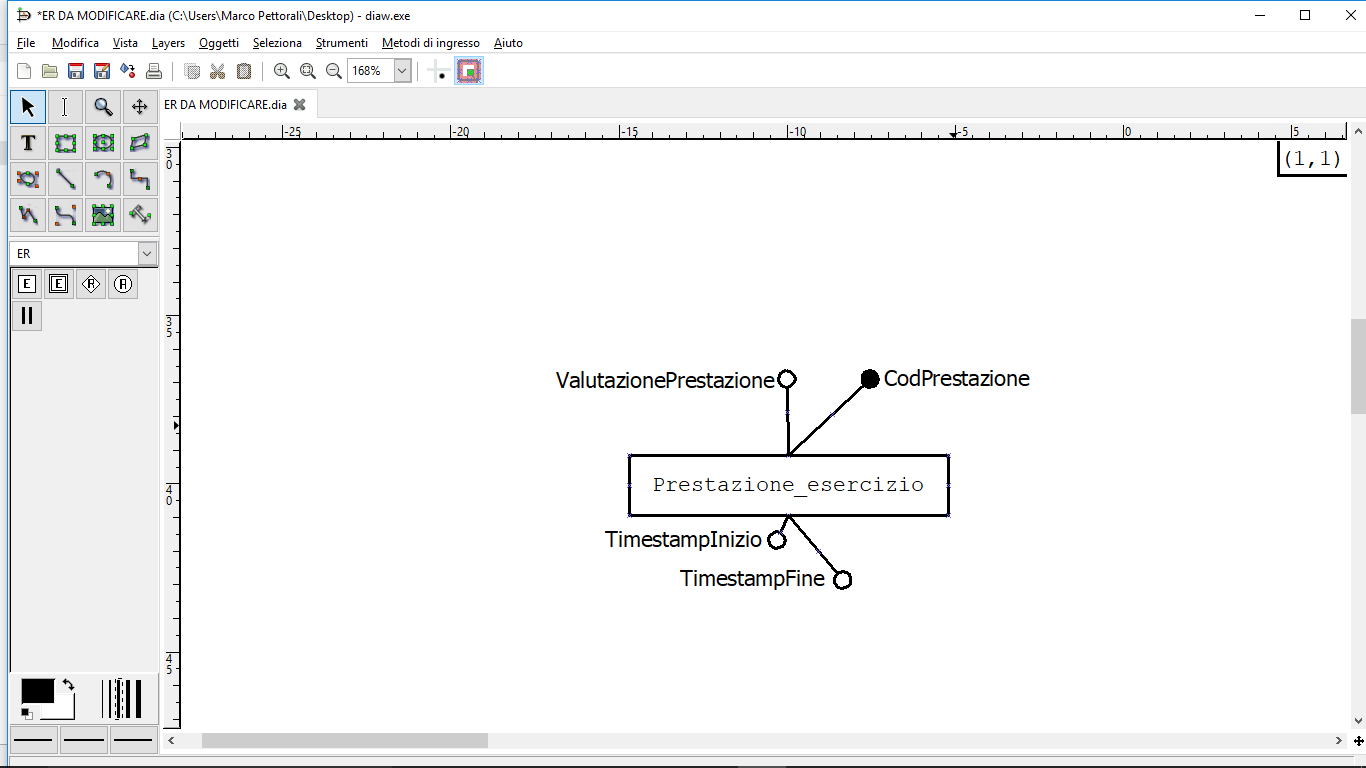
#### 3.2.9.2- Input

Codice di esercizio

#### 3.2.9.3- Output

Durata media e giudizio medio delle prestazioni dei clienti che hanno svolto quell’esercizio.

#### 3.2.9.4- Porzione di diagramma ER interessata dall’operazione



#### 3.2.9.5- Porzione di tavola dei volumi interessata dall’operazione

Entità:

|  |  |
| --- | --- |
| Entità | Volume |
| Prestazione\_esercizio | 270 000 000/anno |

#### 3.2.9.6- Tavola degli accessi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 900 000 | R | E | Prestazione\_esercizio | Ogni esercizio, in media, è stato eseguito 270 000 000/300 = 900 000 volte all’anno. Leggo quante occorrenze ci sono e per ognuna leggo ValutazionePrestazione, TimestampInizio e TimestampFine. |

## 3.3- Introduzione di ridondanze

Analizzando le operazioni 5, 6, 7 e 8 abbiamo pensato di aggiungere delle ridondanze che ci permettessero di eseguire queste operazioni in maniera più efficiente. Abbiamo quindi considerato l’inserimento dell’attributo ClientiPresenti sull’entità Centro, contenente il numero totale di clienti presenti nel centro in ogni momento, e dell’attributo VisiteMedico sull’entità Cliente, che memorizzi quante visite sono state effettuate da quel cliente con il suo medico nutrizionista.

Abbiamo aggiunto anche delle ridondanze relative all’operazione 9, che risulterà molto utile ai fini delle analytics sulle prestazioni dei clienti.

### 3.3.1- Ridondanza 1: ClientiPresenti

Le operazioni 7 e 8 possono essere sensibilmente rese più efficienti utilizzando un contatore, aggiornato in tempo reale, che rappresenti il numero totale di clienti presenti all’interno delle strutture: per tenerlo aggiornato, si incrementa ogni volta che un cliente accede ad un determinato centro e si decrementa ogni volta che un cliente esce.

#### 3.3.1.1- Tavola degli accessi operazione 7 in presenza di ridondanza

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Centro | Basta controllare il contatore ClientiAttuali associato al centro |

#### 3.3.1.2- Codice operazione di aggiornamento della ridondanza

Vedere paragrafo 5.4.1.

#### 3.3.1.3- Tavola degli accessi operazione di aggiornamento

Questa operazione viene effettuata in tempo reale ogni volta che un cliente passa il tornello d’entrata. Sapendo che mediamente al giorno accedono 100 000 clienti, questa operazione è effettuata 100 000 volte al giorno.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | E | Accesso | Si controlla se il cliente è entrato o uscito, conoscendo il codice di accesso |
| 1 | R | R | Accesso\_centro | Si controlla dove il cliente ha eseguito l’accesso |
| 1 | W | E | Centro | Si aggiorna la ridondanza incrementandola o decrementandola |

#### 3.3.1.4- Calcolo convenienza della ridondanza

L’operazione target (Op. 7 ha una frequenza giornaliera stimata fT pari a 100 000/giorno. Il totale delle operazioni di lettura e scrittura effettuate da questa operazione in assenza di ridondanza oT è 180 200. Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione:

nT = fT ∙ oT = 100 000 ∙ 180 200 = 18 020 000 000.

In presenza di ridondanza, l’operazione effettua un totale di operazioni elementari oTRID pari a 1. Il totale di operazioni elementari effettuate nTRID si ottiene moltiplicando fT e oTRID. Sarà dunque

nTRID = fT ∙ oTRID = 100 000 ∙ 1 = 100 000.

Per ottenere il totale delle operazioni elementari comportate dall’introduzione della ridondanza, è necessario però sommare il numero di operazioni elementari effettuate giornalmente dalla procedura che aggiorna la ridondanza.

La frequenza giornaliera della procedura gA è pari a 100 000/giorno. Il numero di operazioni elementari necessari alla sua esecuzione oA è 3.

Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione di aggiornamento:

nA = gA ∙ oA = 100 000 ∙ 3 = 300 000.

Dunque, in presenza di ridondanza si effettuano giornalmente nA + nTRID = 100 000 + 300 000 = 400 000 operazioni.

Poiché nA + nTRID ≤ nT, allora si può concludere che l’introduzione della ridondanza in analisi, è conveniente, pertanto la ridondanza può essere mantenuta in quanto migliora le performance dell’operazione 7.

#### 3.3.1.5- Tavola degli accessi operazione 8 in presenza di ridondanza

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 2 | R | E | Centro | Esiste un solo centro con il codice di input. Si leggono i valori di MaxClienti e ClientiAttuali |
| 1 | W | E | Accesso | Se l’operazione va a buon fine, si aggiunge il nuovo accesso |
| 1 | W | R | Accesso\_centro | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |
| 1 | W | R | Accesso\_cliente | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |
| 1 | W | R | Accesso\_armadietto | Si aggiungono le informazioni correlate all’accesso relative a centro, cliente e armadietto |

#### 3.3.1.6- Calcolo convenienza della ridondanza

L’operazione target (Op. 8) ha una frequenza giornaliera stimata fT pari a 100 000/giorno. Il totale delle operazioni di lettura e scrittura effettuate da questa operazione in assenza di ridondanza oT è 180 205. Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione:

nT = fT ∙ oT = 100 000 ∙ 180 205 = 18 020 500 000.

In presenza di ridondanza, l’operazione effettua un totale di operazioni elementari oTRID pari a 6. Il totale di operazioni elementari effettuate nTRID si ottiene moltiplicando fT e oTRID. Sarà dunque

nTRID = fT ∙ oTRID = 100 000 ∙ 6 = 600 000.

Per ottenere il totale delle operazioni elementari comportate dall’introduzione della ridondanza, è necessario però sommare il numero di operazioni elementari effettuate giornalmente dalla procedura che aggiorna la ridondanza.

La frequenza giornaliera della procedura gA è pari a 100 000/giorno. Il numero di operazioni elementari necessari alla sua esecuzione oA è 3.

Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione di aggiornamento:

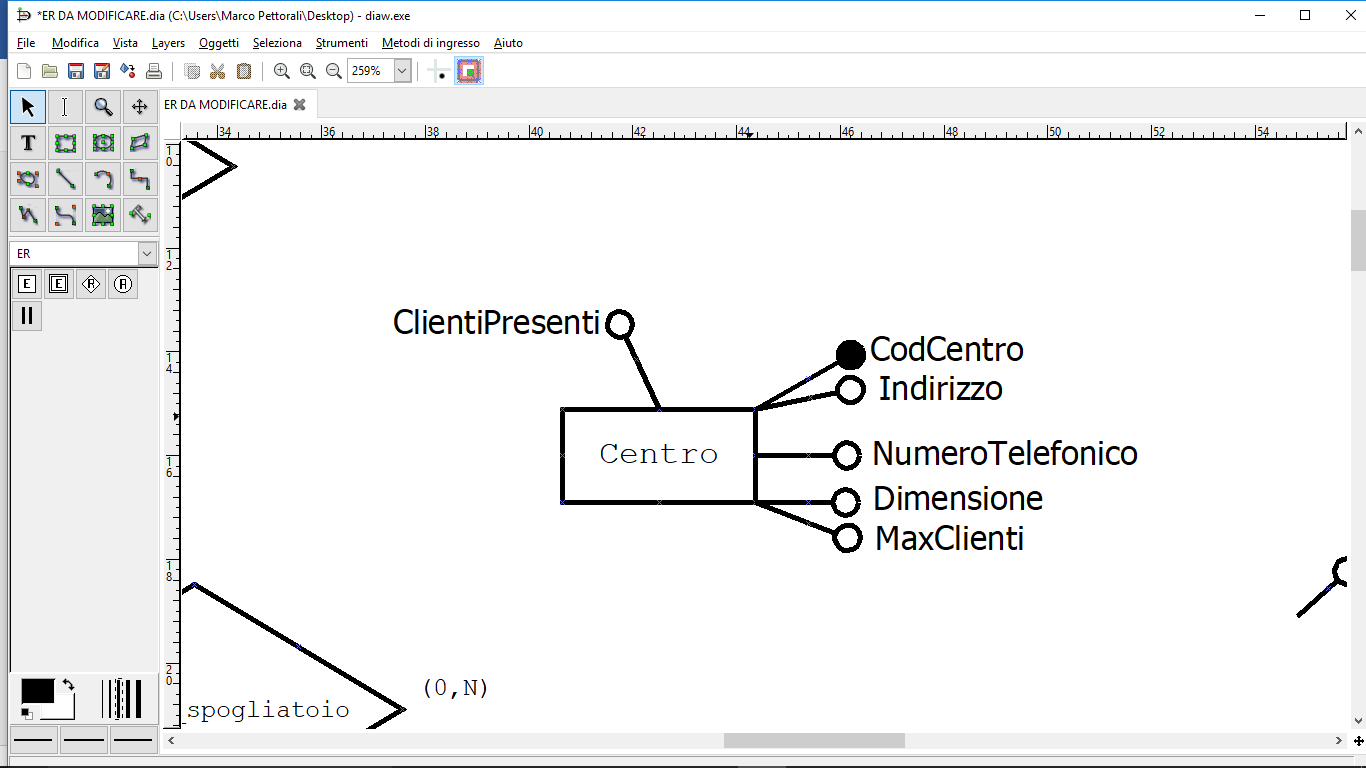
nA = gA ∙ oA = 100 000 ∙ 3 = 300 000.

Dunque, in presenza di ridondanza si effettuano giornalmente nA + nTRID = 600 000 + 300 000 = 900 000 operazioni.

Poiché nA + nTRID ≤ nT, allora si può concludere che l’introduzione della ridondanza in analisi, è conveniente, pertanto la ridondanza può essere mantenuta in quanto migliora le performance dell’operazione 8.

#### 3.3.1.7- Modifiche al diagramma ER

Poiché le operazioni 7 e 8, in presenza di ridondanza, sono molto più efficenti, la ridondanza è stata mantenuta. Si apportano le seguenti modifiche al diagramma ER:



### 3.3.2- Ridondanza 2: VisiteMedico

Le operazioni 5 e 6 possono essere rese più efficienti inserendo, per ogni cliente, un contatore delle visite effettuate dal medico assegnatogli al momento della firma del contratto. Ogni volta che verrà inserita una nuova visita nel database, il contatore si incrementerà, e verrà resettato ogni volta che il cliente termina la scheda di alimentazione, iniziandone così una nuova.

#### 3.3.2.1- Tavola degli accessi operazione 5 in presenza di ridondanza

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Cliente | Conoscendo il codice del cliente si risale immediatamente al valore di VisiteMedico ad esso collegato |

#### 3.3.2.2- Codice operazione di aggiornamento della ridondanza

Vedere paragrafo 5.4.2.

#### 3.3.2.3- Tavola degli accessi operazione di aggiornamento

Questa operazione viene effettuata ogni volta che si aggiunge al database una visita, quindi in media 10 000 volte al giorno.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Assegnazione\_medico | Sapendo il codice del cliente, si risale al medico assegnatogli |
| 1 | R | E | Medico | Si controlla se il medico che visita è il medico assegnato al cliente |
| 1 | R | R | Cliente | Sapendo il codice del cliente, si risale immediatamente a VisiteMedico e lo si incrementa se il medico è quello giusto |

#### 3.3.2.4- Calcolo convenienza della ridondanza

L’operazione target (Op. 5) ha una frequenza giornaliera stimata fT pari a 10 000/giorno. Il totale delle operazioni di lettura e scrittura effettuate da questa operazione in assenza di ridondanza oT è 3646. Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione:

nT = fT ∙ oT = 10 000 ∙ 3646 = 36 460 000.

In presenza di ridondanza, l’operazione effettua un totale di operazioni elementari oTRID pari a 1. Il totale di operazioni elementari effettuate nTRID si ottiene moltiplicando fT e oTRID. Sarà dunque

nTRID = fT ∙ oTRID = 10 000 ∙ 1 = 10 000.

Per ottenere il totale delle operazioni elementari comportate dall’introduzione della ridondanza, è necessario però sommare il numero di operazioni elementari effettuate giornalmente dalla procedura che aggiorna la ridondanza.

La frequenza giornaliera della procedura gA è pari a 10 000/giorno. Il numero di operazioni elementari necessari alla sua esecuzione oA è 3.

Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione di aggiornamento:

nA = gA ∙ oA = 10 000 ∙ 3 = 30 000.

Dunque, in presenza di ridondanza si effettuano giornalmente nA + nTRID = 10 000 + 30 000 = 40 000 operazioni.

Poiché nA + nTRID ≤ nT, allora si può concludere che l’introduzione della ridondanza in analisi, è conveniente, pertanto la ridondanza può essere mantenuta in quanto migliora le performance dell’operazione 5.

#### 3.3.2.5- Tavola degli accessi operazione 6 in presenza di ridondanza

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | E | Cliente | Sapendo il codice del cliente, si legge il suo valore di VisiteMedico |
| 1 | R | R | Alimentazione\_cliente | Si trovano tutte le schede d’alimentazione del cliente |
| 1 | R | E | Scheda\_alimentazione | Dalla scheda d’alimentazione attuale, si legge la data di inizio per ricavare la durata della scheda fino ad oggi. |
| 1 | W | E | Scheda\_alimentazione | Si scrive sull’attributo FrequenzaVisite |

#### 3.3.2.6- Calcolo convenienza della ridondanza

L’operazione target (Op. 6) ha una frequenza giornaliera stimata fT pari a 2 500/giorno. Il totale delle operazioni di lettura e scrittura effettuate da questa operazione in assenza di ridondanza oT è 3 630. Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione:

nT = fT ∙ oT = 2 500 ∙ 3 630 = 9 075 000.

In presenza di ridondanza, l’operazione effettua un totale di operazioni elementari oTRID pari a 4. Il totale di operazioni elementari effettuate nTRID si ottiene moltiplicando fT e oTRID. Sarà dunque

nTRID = fT ∙ oTRID = 2 500 ∙ 4 = 10 000.

Per ottenere il totale delle operazioni elementari comportate dall’introduzione della ridondanza, è necessario però sommare il numero di operazioni elementari effettuate giornalmente dalla procedura che aggiorna la ridondanza.

La frequenza giornaliera della procedura gA è pari a 10 000/giorno. Il numero di operazioni elementari necessari alla sua esecuzione oA è 3.

Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione di aggiornamento:

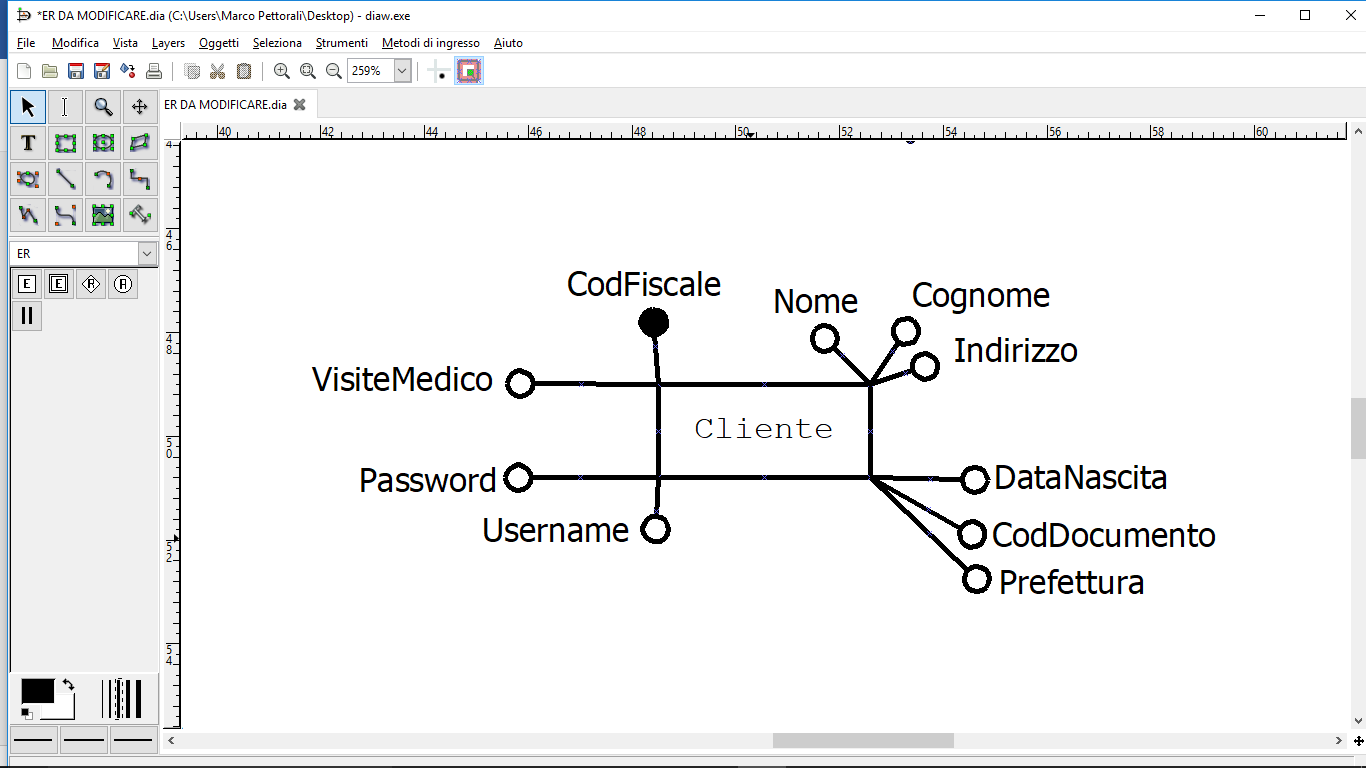
nA = gA ∙ oA = 10 000 ∙ 3 = 30 000.

Dunque, in presenza di ridondanza si effettuano giornalmente nA + nTRID = 10 000 + 30 000 = 40 000 operazioni.

Poiché nA + nTRID ≤ nT, allora si può concludere che l’introduzione della ridondanza in analisi, è conveniente, pertanto la ridondanza può essere mantenuta in quanto migliora le performance dell’operazione 6.

#### 3.3.2.7- Modifiche al diagramma ER

Poiché le operazioni 5 e 6, in presenza di ridondanza, sono molto più efficenti, la ridondanza è stata mantenuta. Si apportano le seguenti modifiche al diagramma ER:



### 3.3.3- Ridondanza 3 : TotVoto, TotDurata, TotEsecuzioni

Questi tre attributi ridondanti sull’entità “Esercizio” raccolgono rispettivamente le informazioni relative alla somma delle votazioni ottenute dai clienti, alla somma delle durate delle esecuzioni e del numero di esecuzioni di un esercizio. Sarà dunque possibile calcolare il giudizio medio e la durata media dividendo TotVoto e TotDurata per TotEsecuzioni.

#### 3.3.3.1- Tavola degli accessi operazione 9 in presenza di ridondanza

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | E | Esercizio | Basta leggere i valori delle tre ridondanze relative al record dell’esercizio target |

#### 3.3.3.2- Codice operazione di aggiornamento della ridondanza

Vedere paragrafo 5.4.3.

#### 3.3.3.3- Tavola degli accessi operazione di aggiornamento

Questa operazione va in esecuzione ogni volta che un cliente termina un esercizio, quindi in media 740 000 volte al giorno.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numero operazioni elementari | Tipo operazione | Tipo costrutto | Nome costrutto | Descrizione |
| 1 | R | R | Esercizio\_reale | Una volta inserita una prestazione, e conoscendo quindi il codice dell’esercizio svolto, si accede al record corrispondente dell’entità Esercizio |
| 1 | W | E | Esercizio | Si incrementa TotEsecuzioni, si somma a TotVoto il giudizio collegata alla prestazione, e si somma a TotDurata la durata della prestazione |

#### 3.3.3.4- Calcolo convenienza della ridondanza

L’operazione target (Op. 9) ha una frequenza giornaliera stimata fT pari a 740 000/giorno. Il totale delle operazioni di lettura e scrittura effettuate da questa operazione in assenza di ridondanza oT è 900 000. Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione:

nT = fT ∙ oT = 740 000 ∙ 900 000 = 666 000 000 000.

In presenza di ridondanza, l’operazione effettua un totale di operazioni elementari oTRID pari a 1. Il totale di operazioni elementari effettuate nTRID si ottiene moltiplicando fT e oTRID. Sarà dunque

nTRID = fT ∙ oTRID = 740 000 ∙ 1 = 740 000.

Per ottenere il totale delle operazioni elementari comportate dall’introduzione della ridondanza, è necessario però sommare il numero di operazioni elementari effettuate giornalmente dalla procedura che aggiorna la ridondanza.

La frequenza giornaliera della procedura gA è pari a 740 000/giorno. Il numero di operazioni elementari necessari alla sua esecuzione oA è 2.

Si può dunque ricavare il totale di operazioni elementari effettuate giornalmente dall’operazione di aggiornamento:

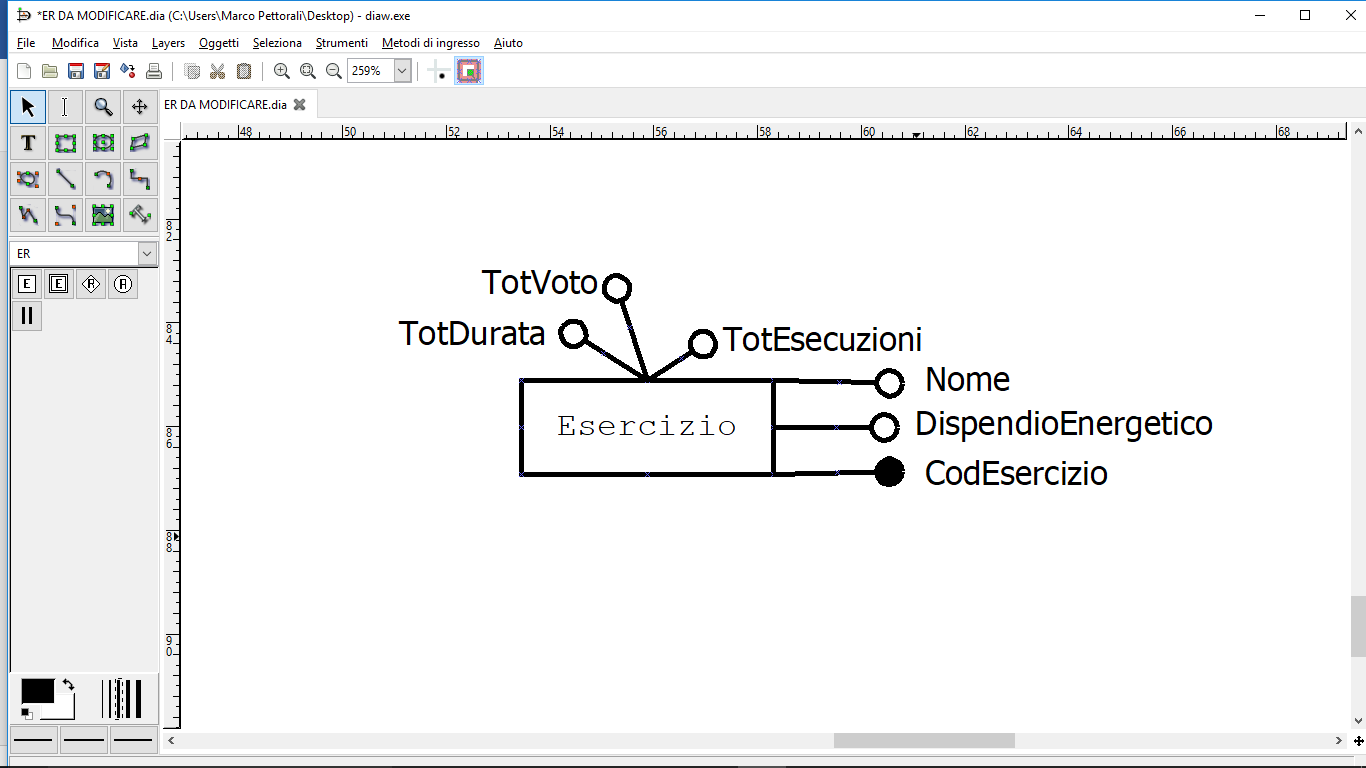
nA = gA ∙ oA = 740 000 ∙ 2 = 1 480 000.

Dunque, in presenza di ridondanza si effettuano giornalmente nA + nTRID = 740 000 + 1 480 000 = 2 220 000 operazioni.

Poiché nA + nTRID  ≤ nT, allora si può concludere che l’introduzione della ridondanza in analisi, è conveniente, pertanto la ridondanza puo’ essere mantenutain quanto migliora le performance dell’operazione 9.

#### 3.3.3.5- Modifiche al diagramma ER

Poiché la ridondanza è stata mantenuta, si apportano le seguenti modifiche al diagramma ER:



# Progettazione logica

## 4.1- Traduzione del modello concettuale in modello logico

Per la traduzione del diagramma ER in modello logico in tabelle, abbiamo associato ad ogni entità e ad ogni relazione che non abbia da alcun lato la cardinalità (1,1) una tabella; per le restanti relazioni, abbiamo eseguito degli accorpamenti alle entità collegate alle relazioni tramite cardinalità (1,1).

Segue l’elenco di tutti gli schemi delle tabelle e delle considerazioni su dipendenze funzionali, normalizzazione e vincoli di integrità:

## 4.2- Tabelle:

**Tabella CENTRO:**

CENTRO(CodCentro, Indirizzo, NumeroTelefonico, Dimensione, MaxClienti, ClientiPresenti, Direttore)

* CodCentro -> Indirizzo, NumeroTelefonico, Dimensione, MaxClienti, ClientiPresenti, Direttore

Poiché la parte sinistra è superchiave, CENTRO è in BCNF.

ClientiPresenti non deve superare MaxClienti

Esiste un vincolo di integrità referenziale tra l’attributo Direttore della tabella CENTRO e l’attributo CodFiscale della tabella DIRETTORE

**Tabella TURNAZIONE:**

TURNAZIONE(CodTurnazione, OraInizioTurno, OraFineTurno, Centro, Giorno, Dipendente)

* CodTurnazione -> OraInizioTurno, OraFineTurno, Centro, Giorno, Dipendente

Poiché la parte sinistra è superchiave, TURNAZIONE è in BCNF.

Non si possono inserire due turnazioni di un dipendente con orari sovrapposti

Un dipendente deve avere un orario di lavoro giornaliero di massimo 8 ore

Esiste un vincolo di integrità referenziale tra l’attributo CodCentro della tabella CENTRO e l’attributo Centro della tabella TURNAZIONE

Esiste un vincolo di integrità referenziale tra l’attributo Giorno della tabella TURNAZIONE e l’attributo Nome della tabella GIORNO

Esiste un vincolo di integrità referenziale tra l’attributo Dipendente della tabella TURNAZIONE e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella RATA:**

RATA(CodRata, DataScadenza, Importo, Stato, Cliente, IstitutoFinanziario)

* CodRata -> DataScadenza, Importo, Stato, Cliente, IstitutoFinanziario

Poiché la parte sinistra è superchiave, RATA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella RATA e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo IstitutoFinanziario della tabella RATA e l’attributo CodIstituto della tabella ISTITUTO FINANZIARIO

**Tabella ISTITUTO FINANZIARIO:**

ISTITUTO FINANZIARIO(CodIstituto, RagioneSociale, TassoInteresse)

* CodIstituto -> RagioneSociale, TassoInteresse

Poiché la parte sinistra è superchiave, ISTITUTO FINANZIARIO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Istituto della tabella CONVENZIONE e l’attributo CodIstituto della tabella ISTITUTO FINANZIARIO

**Tabella CONVENZIONE:**

CONVENZIONE(Istituto, Centro)

Poiché non esistono dipendenze funzionali, CONVENZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella CONVENZIONE e l’attributo CodCentro della tabella CENTRO

**Tabella ORARIO DI SERVIZIO:**

ORARIO DI SERVIZIO(CodOrario, OrarioApertura, OrarioChiusura, Centro, Giorno)

* CodOrario -> OrarioApertura, OrarioChiusura, Centro, Giorno

Poiché la parte sinistra è superchiave, ORARIO DI SERVIZIO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella ORARIO DI SERVIZIO e l’attributo CodCentro della tabella CENTRO

Esiste un vincolo di integrità referenziale tra l’attributo Giorno della tabella ORARIO DI SERVIZIO e l’attributo Nome della tabella GIORNO

**Tabella GIORNO:**

GIORNO(Nome)

Poiché non esistono dipendenze funzionali, GIORNO è in BCNF.

**Tabella PIANIFICAZIONE CORSO:**

PIANIFICAZIONE CORSO(Giorno, Luogo, Corso, OraInizioCorso, OraFineCorso)

* Giorno, Luogo, Corso -> OraInizioCorso, OraFineCorso

Poiché la parte sinistra è superchiave, PIANIFICAZIONE CORSO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Giorno della tabella PIANIFICAZIONE CORSO e l’attributo Nome della tabella GIORNO

Esiste un vincolo di integrità referenziale tra l’attributo Luogo della tabella PIANIFICAZIONE CORSO e l’attributo CodLuogo della tabella LUOGO DI ALLENAMENTO

Esiste un vincolo di integrità referenziale tra l’attributo Corso della tabella PIANIFICAZIONE CORSO e l’attributo CodCorso della tabella CORSO

**Tabella LUOGO DI ALLENAMENTO:**

LUOGO DI ALLENAMENTO(CodLuogo, Centro)

* CodLuogo -> Centro

Poiché la parte sinistra è superchiave, LUOGO DI ALLENAMENTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella LUOGO DI ALLENAMENTO e l’attributo CodCentro della tabella CENTRO

Esiste un vincolo di integrità referenziale tra l’attributo Luogo della tabella SALA e l’attributo CodLuogo della tabella LUOGO DI ALLENAMENTO

**Tabella SALA:**

SALA(Luogo, Nome, ResponsabileSala)

* Luogo -> Nome, ResponsabileSala

Poiché la parte sinistra è superchiave, SALA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo ResponsabileSala della tabella SALA e l’attributo CodFiscale della tabella RESPONSABILE SALA

**Tabella PISCINA:**

PISCINA(Luogo, Dimensione)

* Luogo -> Dimensione

Poiché la parte sinistra è superchiave, PISCINA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Luogo della tabella PISCINA e l’attributo CodLuogo della tabella LUOGO DI ALLENAMENTO

**Tabella DIPENDENTE:**

DIPENDENTE(CodFiscale, Nome, Cognome, DataNascita, Prefettura, CodDocumento, Indirizzo, ResponsabilePersonale)

* CodFiscale -> Nome, Cognome, DataNascita, Prefettura, CodDocumento, Indirizzo, ResponsabilePersonale

Poiché la parte sinistra è superchiave, DIPENDENTE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo ResponsabilePersonale della tabella DIPENDENTE e l’attributo CodFiscale della tabella RESPONSABILE PERSONALE

**Tabella IMPIEGO:**

IMPIEGO(Centro, Dipendente, Mansione)

* Centro, Dipendente -> Mansione

Poiché la parte sinistra è superchiave, IMPIEGO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella IMPIEGO e l’attributo CodCentro della tabella CENTRO

Esiste un vincolo di integrità referenziale tra l’attributo Dipendente della tabella IMPIEGO e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella ATTREZZATURA:**

ATTREZZATURA(CodAttrezzatura, Tipologia, ConsumoEnergetico, Usura, Sala)

* CodAttrezzatura -> Tipologia, ConsumoEnergetico, Usura, Sala

Poiché la parte sinistra è superchiave, ATTREZZATURA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Sala della tabella ATTREZZATURA e l’attributo Luogo della tabella SALA

**Tabella REGOLAZIONE:**

REGOLAZIONE(Attrezzatura, NomeRegolazione)

Poiché non esistono dipendenze funzionali, REGOLAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Attrezzatura della tabella REGOLAZIONE e l’attributo CodAttrezzatura della tabella ATTREZZATURA

**Tabella CONFIGURAZIONE ESERCIZIO:**

CONFIGURAZIONE ESERCIZIO(CodConfigurazione, Attrezzatura)

* CodConfigurazione -> Attrezzatura

Poiché la parte sinistra è superchiave, CONFIGURAZIONE ESERCIZIO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Attrezzatura della tabella CONFIGURAZIONE ESERCIZIO e l’attributo CodAttrezzatura della tabella ATTREZZATURA

**Tabella REGOLAZIONE ESERCIZIO:**

REGOLAZIONE ESERCIZIO(Configurazione, Attrezzatura, NomeRegolazione, Valore)

* Configurazione, Attrezzatura, NomeRegolazione -> Valore

Poiché la parte sinistra è superchiave, REGOLAZIONE ESERCIZIO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Configurazione della tabella REGOLAZIONE ESERCIZIO e l’attributo CodConfigurazione della tabella CONFIGURAZIONE ESERCIZIO

Esiste un vincolo di integrità referenziale tra l’attributo Attrezzatura della tabella REGOLAZIONE ESERCIZIO e l’attributo CodAttrezzatura della tabella ATTREZZATURA

Esiste un vincolo di integrità referenziale tra l’attributo NomeRegolazione della tabella REGOLAZIONE ESERCIZIO e l’attributo NomeRegolazione della tabella REGOLAZIONE

**Tabella SPOGLIATOIO:**

SPOGLIATOIO(CodSpogliatoio, Capienza, PostiDisponibili, Posizione, Centro)

* CodSpogliatoio -> Capienza, PostiDisponibili, Posizione, Centro

Poiché la parte sinistra è superchiave, SPOGLIATOIO è in BCNF.

PostiDisponibili non deve superare Capienza

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella SPOGLIATOIO e l’attributo CodCentro della tabella CENTRO

**Tabella ARMADIETTO:**

ARMADIETTO(CodArmadietto, Combinazione, Spogliatoio)

* CodArmadietto -> Combinazione, Spogliatoio

Poiché la parte sinistra è superchiave, ARMADIETTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Spogliatoio della tabella ARMADIETTO e l’attributo CodSpogliatoio della tabella SPOGLIATOIO

**Tabella ACCESSO:**

ACCESSO(CodArmadietto, TimestampIngresso, TimestampUscita, Centro, Cliente)

* CodArmadietto -> TimestampIngresso, TimestampUscita, Centro, Cliente

Poiché la parte sinistra è superchiave, ACCESSO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Armadietto della tabella ACCESSO e l’attributo CodArmadietto della tabella ARMADIETTO

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella ACCESSO e l’attributo CodCentro della tabella CENTRO

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella ACCESSO e l’attributo CodFiscale della tabella CLIENTE

**Tabella MEDICO:**

MEDICO(CodFiscale)

Poiché non esistono dipendenze funzionali, MEDICO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella MEDICO e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella VISITA:**

VISITA(CodVisita, DataVisita, Medico, Cliente)

* CodVisita -> DataVisita, Medico, Cliente

Poiché la parte sinistra è superchiave, VISITA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Medico della tabella VISITA e l’attributo CodFiscale della tabella MEDICO

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella VISITA e l’attributo CodFiscale della tabella CLIENTE

**Tabella CLIENTE:**

CLIENTE(CodFiscale, Nome, Cognome, Indirizzo, DataNascita, CodDocumento, Prefettura, Username, Password, VisiteMedico, Medico, Tutor, Scopo)

* CodFiscale -> Nome, Cognome, Indirizzo, DataNascita, CodDocumento, Prefettura, Username, Password, VisiteMedico, Medico, Tutor, Scopo

Poiché la parte sinistra è superchiave, CLIENTE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Medico della tabella CLIENTE e l’attributo CodFiscale della tabella MEDICO

Esiste un vincolo di integrità referenziale tra l’attributo Tutor della tabella CLIENTE e l’attributo CodFiscale della tabella TUTOR

Esiste un vincolo di integrità referenziale tra l’attributo Scopo della tabella CLIENTE e l’attributo NomeScopo della tabella SCOPO

**Tabella CORSO:**

CORSO(CodCorso, DataInizioCorso, DataFineCorso, Livello, MaxPartecipanti, Istruttore, Disciplina)

* CodCorso -> DataInizioCorso, DataFineCorso, Livello, MaxPartecipanti, Istruttore, Disciplina

Poiché la parte sinistra è superchiave, CORSO è in BCNF.

DataFineCorso deve essere maggiore di DataInizioCorso.

Esiste un vincolo di integrità referenziale tra l’attributo Istruttore della tabella CORSO e l’attributo CodFiscale della tabella ISTRUTTORE

**Tabella ISTRUTTORE:**

ISTRUTTORE(CodFiscale)

Poiché non esistono dipendenze funzionali, ISTRUTTORE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella ISTRUTTORE e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella RESPONSABILE PERSONALE:**

RESPONSABILE PERSONALE(CodFiscale)

Poiché non esistono dipendenze funzionali, RESPONSABILE PERSONALE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella RESPONSABILE PERSONALE e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella DIRETTORE:**

DIRETTORE(CodFiscale)

Poiché non esistono dipendenze funzionali, DIRETTORE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella DIRETTORE e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella RESPONSABILE SALA:**

RESPONSABILE SALA(CodFiscale)

Poiché non esistono dipendenze funzionali, RESPONSABILE SALA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella RESPONSABILE SALA e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella TUTOR:**

TUTOR(CodFiscale)

Poiché non esistono dipendenze funzionali, TUTOR è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella TUTOR e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella CONSULENTE:**

CONSULENTE(CodFiscale)

Poiché non esistono dipendenze funzionali, CONSULENTE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodFiscale della tabella CONSULENTE e l’attributo CodFiscale della tabella DIPENDENTE

**Tabella ISCRIZIONE:**

ISCRIZIONE(Cliente, Corso)

Poiché non esistono dipendenze funzionali, ISCRIZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella ISCRIZIONE e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Corso della tabella ISCRIZIONE e l’attributo CodCorso della tabella CORSO

**Tabella CONTRATTO:**

CONTRATTO(CodContratto, Durata, Tipologia, DataSottoscrizione, Importo, ModalitaDiPagamento,

Cliente, Consulente, Centro)

* CodContratto -> Durata, Tipologia, DataSottoscrizione, Importo, ModalitaDiPagamento, Cliente, Consulente, Centro

Poiché la parte sinistra è superchiave, CONTRATTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella CONTRATTO e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Consulente della tabella CONTRATTO e l’attributo CodFiscale della tabella CONSULENTE

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella CONTRATTO e l’attributo CodCentro della tabella CENTRO

**Tabella OFFERTA:**

OFFERTA(CodOfferta, TipoOfferta, MaxIngressi, AccessiPiscine, CostoMensile, Centro)

* CodOfferta -> TipoOfferta, MaxIngressi, AccessiPiscine, CostoMensile, Centro

Poiché la parte sinistra è superchiave, OFFERTA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella OFFERTA e l’attributo CodCentro della tabella CENTRO

**Tabella INCLUSIONE OFFERTA:**

INCLUSIONE OFFERTA(Contratto, Offerta)

Poiché non esistono dipendenze funzionali, INCLUSIONE OFFERTA è in BCNF.

Un contratto può includere offerte relative al massimo a 3 centri diversi.

Esiste un vincolo di integrità referenziale tra l’attributo Contratto della tabella INCLUSIONE OFFERTA e l’attributo CodContratto della tabella CONTRATTO

Esiste un vincolo di integrità referenziale tra l’attributo Offerta della tabella INCLUSIONE OFFERTA e l’attributo CodOfferta della tabella OFFERTA

**Tabella OFFERTA LUOGO:**

OFFERTA LUOGO(Offerta, Luogo)

Poiché non esistono dipendenze funzionali, OFFERTA LUOGO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Offerta della tabella OFFERTA LUOGO e l’attributo CodOfferta della tabella OFFERTA

Esiste un vincolo di integrità referenziale tra l’attributo Luogo della tabella OFFERTA LUOGO e l’attributo CodLuogo della tabella LUOGO DI ALLENAMENTO

**Tabella OFFERTA CORSO:**

OFFERTA CORSO(Offerta, Corso)

Poiché non esistono dipendenze funzionali, OFFERTA CORSO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Offerta della tabella OFFERTA CORSO e l’attributo CodOfferta della tabella OFFERTA

Esiste un vincolo di integrità referenziale tra l’attributo Corso della tabella OFFERTA CORSO e l’attributo CodCorso della tabella CORSO

**Tabella SCOPO:**

SCOPO(NomeScopo)

Poiché non esistono dipendenze funzionali, SCOPO è in BCNF.

**Tabella MUSCOLO:**

MUSCOLO(NomeMuscolo)

Poiché non esistono dipendenze funzionali, MUSCOLO è in BCNF.

**Tabella TARGET:**

TARGET(Muscolo, Cliente, LivelloPotenziamento)

* Muscolo, Cliente -> LivelloPotenziamento

Poiché la parte sinistra è superchiave, TARGET è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Muscolo della tabella TARGET e l’attributo NomeMuscolo della tabella MUSCOLO

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella TARGET e l’attributo CodFiscale della tabella CLIENTE

**Tabella AMICIZIA:**

AMICIZIA(Ricevente, Richiedente, Stato)

* Ricevente, Richiedente -> Stato

Poiché la parte sinistra è superchiave, AMICIZIA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Richiedente della tabella AMICIZIA e l’attributo Username della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Ricevente della tabella AMICIZIA e l’attributo Username della tabella CLIENTE

**Tabella MISURAZIONE:**

MISURAZIONE(Cliente, DataMisurazione, AcquaTotale, MassaMagra, MassaGrassa, Peso, Altezza, IMC, Stato)

* Cliente, DataMisurazione -> AcquaTotale, MassaMagra, MassaGrassa, Peso, Altezza, IMC, Stato

Poiché la parte sinistra è superchiave, MISURAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella MISURAZIONE e l’attributo CodFiscale della tabella CLIENTE

**Tabella ATTIVITA:**

ATTIVITA(Nome)

Poiché non esistono dipendenze funzionali, ATTIVITA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Attivita della tabella INTERESSE e l’attributo Nome della tabella ATTIVITA

**Tabella INTERESSE:**

INTERESSE(Attivita, Cliente)

Poiché non esistono dipendenze funzionali, INTERESSE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella INTERESSE e l’attributo CodFiscale della tabella CLIENTE

**Tabella SCHEDA ALIMENTAZIONE:**

SCHEDA ALIMENTAZIONE(CodScheda, DataInizioScheda, DataFineScheda, FrequenzaVisita, Obiettivo, Cliente, Medico, Dieta)

* CodScheda -> DataInizioScheda, DataFineScheda, FrequenzaVisita, Obiettivo, Cliente, Medico, Dieta

Poiché la parte sinistra è superchiave, SCHEDA ALIMENTAZIONE è in BCNF.

DataFineScheda deve essere maggiore di DataInizioScheda

Non Possono esistere due schede d’alimentazione attive contemporaneamente per uno stesso cliente.

Esiste un vincolo di integrità referenziale tra l’attributo Obiettivo della tabella SCHEDA DI ALIMENTAZIONE e l’attributo NomeScopo della tabella SCOPO

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella SCHEDA DI ALIMENTAZIONE e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Medico della tabella SCHEDA DI ALIMENTAZIONE e l’attributo CodFiscale della tabella MEDICO

Esiste un vincolo di integrità referenziale tra l’attributo Dieta della tabella SCHEDA DI ALIMENTAZIONE e l’attributo CodDieta della tabella DIETA

**Tabella DIETA:**

DIETA(CodDieta, NumeroPasti, ApportoCalorico)

* CodDieta -> NumeroPasti, ApportoCalorico

Poiché la parte sinistra è superchiave, DIETA è in BCNF.

**Tabella PASTO:**

PASTO(Nome, Composizione)

* Nome -> Composizione

Poiché la parte sinistra è superchiave, PASTO è in BCNF.

**Tabella COMPOSIZIONE DIETA:**

COMPOSIZIONE DIETA(Dieta, Pasto)

Poiché non esistono dipendenze funzionali, COMPOSIZIONE DIETA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Dieta della tabella COMPOSIZIONE DIETA e l’attributo CodDieta della tabella DIETA

Esiste un vincolo di integrità referenziale tra l’attributo Pasto della tabella COMPOSIZIONE DIETA e l’attributo Nome della tabella PASTO

**Tabella INTEGRATORE:**

INTEGRATORE(CodIntegratore, NomeCommerciale, Sostanza, Concentrazione, DataScadenza, NumeroPezzi, Forma, PrezzoIngrosso, PrezzoDettaglio)

* NomeCommerciale -> Sostanza, Concentrazione, DataScadenza, NumeroPezzi, Forma, PrezzoIngrosso, PrezzoDettaglio

Poiché la parte sinistra è superchiave, INTEGRATORE è in BCNF.

**Tabella ABBINAMENTO:**

ABBINAMENTO(SchedaAlimentazione, Integratore)

Poiché non esistono dipendenze funzionali, ABBINAMENTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo SchedaAlimentazione della tabella ABBINAMENTO e l’attributo CodScheda della tabella SCHEDA DI ALIMENTAZIONE

Esiste un vincolo di integrità referenziale tra l’attributo Integratore della tabella ABBINAMENTO e l’attributo NomeCommerciale della tabella INTEGRATORE

**Tabella ACQUISTO:**

ACQUISTO(CodAcquisto, DataAcquisto, Quantita, Integratore, Cliente, Importo)

* CodAcquisto -> DataAcquisto, Quantita, Integratore, Cliente, Importo

Poiché la parte sinistra è superchiave, ACQUISTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Integratore della tabella ACQUISTO e l’attributo NomeCommerciale della tabella INTEGRATORE

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella ACQUISTO e l’attributo CodFiscale della tabella CLIENTE

**Tabella ORDINE:**

ORDINE(CodOrdine, Stato, DataEvasione, DataConsegna, Centro, Fornitore)

* CodOrdine -> Stato, DataEvasione, DataConsegna, Centro, Fornitore

Poiché la parte sinistra è superchive, ORDINE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Centro della tabella ORDINE e l’attributo CodCentro della tabella CENTRO

Esiste un vincolo di integrità referenziale tra l’attributo Fornitore della tabella ORDINE e l’attributo PartitaIva della tabella FORNITORE

**Tabella ORDINE INTEGRATORE:**

ORDINE INTEGRATORE(Integratore, Ordine, Quantita)

* Integratore, Ordine -> Quantita

Poiché la parte sinistra è superchiave, ORDINE INTEGRATORE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Integratore della tabella ORDINE INTEGRATORE e l’attributo NomeCommerciale della tabella INTEGRATORE

Esiste un vincolo di integrità referenziale tra l’attributo Ordine della tabella ORDINE INTEGRATORE e l’attributo CodOrdine della tabella ORDINE

**Tabella FORNITORE:**

FORNITORE(PartitaIva, NomeCommerciale, NumeroTelefonico, Indirizzo, FormaSocietaria)

* PartitaIva -> NomeCommerciale, NumeroTelefonico, Indirizzo, FormaSocietaria

Poiché la parte sinistra è superchiave, FORNITORE è in BCNF.

**Tabella COMMERCIALIZZAZIONE:**

COMMERCIALIZZAZIONE(Fornitore, Integratore, CodEsterno)

* Fornitore, Integratore -> CodEsterno

Poiché la parte sinistra è superchiave, COMMERCIALIZZAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Fornitore della tabella COMMERCIALIZZAZIONE e l’attributo PartitaIva della tabella FORNITORE

Esiste un vincolo di integrità referenziale tra l’attributo Integratore della tabella COMMERCIALIZZAZIONE e l’attributo NomeCommerciale della tabella INTEGRATORE

**Tabella CERCHIA:**

CERCHIA(CodCerchia, Nome, Cliente)

* CodCerchia -> Nome, Cliente

Poiché la parte sinistra è superchiave, CERCHIA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella CERCHIA e l’attributo CodFiscale della tabella CLIENTE

**Tabella APPARTENENZA:**

APPARTENENZA(Cerchia, Cliente)

Poiché non esistono dipendenze funzionali, APPARTENENZA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cerchia della tabella APPARTENENZA e l’attributo CodCerchia della tabella CERCHIA

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella APPARTENENZA e l’attributo Username della tabella CLIENTE

**Tabella POST:**

POST(CodPost, Testo, Topic, Timestamp, Cliente, Thread)

* CodPost -> Testo, Topic, Timestamp, Cliente, Thread

Poichè la parte sinistra è superchiave, POST è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella POST e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Thread della tabella POST e l’attributo TitoloThread della tabella THREAD

**Tabella COLLEGAMENTO:**

COLLEGAMENTO(URL, Post)

Poiché non esistono dipendenze funzionali, COLLEGAMENTO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Post della tabella COLLEGAMENTO e l’attributo CodPost della tabella POST

**Tabella THREAD:**

THREAD(TitoloThread, Cliente)

* TitoloThread -> Cliente

Poiché la parte sinistra è superchiave, THREAD è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella THREAD e l’attributo CodFiscale della tabella cliente

**Tabella VALUTAZIONE POST:**

VALUTAZIONE POST(Cliente, Post, Giudizio)

* Cliente, Post -> Giudizio

Poiché la parte sinistra è superchiave, VALUTAZIONE POST è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella VALUTAZIONE POST e l’attributo Username della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Post della tabella VALUTAZIONE POST e l’attributo CodPost della tabella POST

**Tabella SCHEDA ALLENAMENTO:**

SCHEDA ALLENAMENTO(CodScheda, DataInizioScheda, DataFineScheda, Cliente, DataAssegnazione, Tutor)

* CodScheda -> DataInizioScheda, DataFineScheda, Cliente, DataAssegnazione, Tutor

Poiché la parte sinistra è superchiave, SCHEDA ALLENAMENTO è in BCNF.

DataFineScheda deve essere maggiore di DataInizioScheda

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella SCHEDA DI ALLENAMENTO e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Tutor della tabella SCHEDA DI ALLENAMENTO e l’attributo CodFiscale della tabella TUTOR

**Tabella ESERCIZIO:**

ESERCIZIO(CodEsercizio, Nome, DispendioEnergetico, TotEsecuzioni, TotDurata, TotVoto)

* CodEsercizio -> Nome, DispendioEnergetico, TotEsecuzioni, TotDurata, TotVoto

Poiché la parte sinistra è superchiave, ESERCIZIO è in BCNF.

**Tabella COMPOSIZIONE:**

COMPOSIZIONE(Scheda, Esercizio)

Poiché non esistono dipendenze funzionali, COMPOSIZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Scheda della tabella COMPOSIZIONE e l’attributo CodScheda della tabella SCHEDA DI ALLENAMENTO

Esiste un vincolo di integrità referenziale tra l’attributo Esercizio della tabella COMPOSIZIONE e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella CONFIGURAZIONE:**

CONFIGURAZIONE(CodiceConf, Esercizio)

Poiché non esistono dipendenze funzionali, CONFIGURAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Esercizio della tabella CONFIGURAZIONE e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella AEROBICO:**

AEROBICO(CodEsercizio, Durata)

* CodEsercizio -> Durata

Poiché la parte sinistra è superchiave, AEROBICO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodEsercizio della tabella AEROBICO e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella ANAEROBICO:**

ANAEROBICO(CodEsercizio, TempoRecupero, NumeroRipetizioni)

* CodEsercizio -> TempoRecupero, NumeroRipetizioni

Poiché la parte sinistra è superchiave, ANAEROBICO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodEsercizio della tabella ANAEROBICO e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella RIPETIZIONE:**

RIPETIZIONE(CodEsercizio, NumeroRipetizione)

Poiché non esistono dipendenze funzionali, RIPETIZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodEsercizio della tabella RIPETIZIONE e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella CONFIGURAZIONE RIPETIZIONE:**

CONFIGURAZIONE RIPETIZIONE(CodConf, CodEsercizio, NumeroRipetizione)

Poiché non esistono dipendenze funzionali, CONFIGURAZIONE RIPETIZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodConf della tabella CONFIGURAZIONE RIPETIZIONE e l’attributo CodiceConf della tabella CONFIGURAZIONE

Esiste un vincolo di integrità referenziale tra l’attributo CodEsercizio della tabella CONFIGURAZIONE RIPETIZIONE e l’attributo CodEsercizio della tabella ESERCIZIO

Esiste un vincolo di integrità referenziale tra l’attributo NumeroRipetizione della tabella CONFIGURAZIONE RIPETIZIONE e l’attributo NumeroRipetizione della tabella RIPETIZIONE

**Tabella SFIDA:**

SFIDA(CodSfida, DataInizio, DataFine, DataLancio, Scopo, Cliente)

* CodSfida -> DataInizio, DataFine, DataLancio, Scopo, Cliente

Poiché la parte sinistra è superchiave, SFIDA è in BCNF.

DataFine deve essere maggiore di DataInizio.

Esiste un vincolo di integrità referenziale tra l’attributo Scopo della tabella SFIDA e l’attributo NomeScopo della tabella SCOPO

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella SFIDA e l’attributo CodFiscale della tabella CLIENTE

**Tabella PARTECIPAZIONE:**

PARTECIPAZIONE(Sfida, Cliente)

Poiché non esistono dipendenze funzionali, PARTECIPAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Sfida della tabella PARTECIPAZIONE e l’attributo CodSfida della tabella SFIDA

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella PARTECIPAZIONE e l’attributo CodFiscale della tabella CLIENTE

**Tabella ABBINAMENTO ALIMENTAZIONE:**

ABBINAMENTO ALIMENTAZIONE(Sfida, SchedaAlimentazione)

Poiché non esistono dipendenze funzionali, ABBINAMENTO ALIMENTAZIONE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Sfida della tabella ABBINAMENTO ALIMENTAZIONE e l’attributo CodSfida della tabella SFIDA

Esiste un vincolo di integrità referenziale tra l’attributo SchedaAlimentazione della tabella ABBINAMENTO ALIMENTAZIONE e l’attributo CodScheda della tabella SCHEDA DI ALIMENTAZIONE

**Tabella GIORNO SFIDA:**

GIORNO SFIDA(CodSfida, Giorno, SchedaAllenamento)

* CodSfida, Giorno -> SchedaAllenamento

Poiché la parte sinistra è superchiave, GIORNO SFIDA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodSfida della tabella GIORNO SFIDA e l’attributo CodSfida della tabella SFIDA

Esiste un vincolo di integrità referenziale tra l’attributo SchedaAllenamento della tabella GIORNO SFIDA e l’attributo CodScheda della tabella SCHEDA DI ALLENAMENTO

**Tabella VALUTAZIONE SFIDA:**

VALUTAZIONE SFIDA(Cliente, CodSfida, Giorno, GiudizioPsichico, GiudizioFisico)

* Cliente, CodSfida, Giorno -> GiudizioPsichico, GiudizioFisico

Poiché la parte sinistra è superchiave, VALUTAZIONE SFIDA è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella VALUTAZIONE SFIDA e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo CodSfida della tabella VALUTAZIONE SFIDA e l’attributo CodSfida della tabella SFIDA

Esiste un vincolo di integrità referenziale tra l’attributo Giorno della tabella VALUTAZIONE SFIDA e l’attributo Giorno della tabella GIORNO SFIDA

**Tabella PRESTAZIONE ESERCIZIO:**

PRESTAZIONE ESERCIZIO(CodPrestazione, ValutazionePrestazione, TimestampInizio, TimestampFine, Cliente, Esercizio, Durata, TempoRecupero, NumeroRipetizioni)

* CodPrestazione -> ValutazionePrestazione, TimestampInizio, TimestampFine, Cliente, Esercizio, Durata, TempoRecupero, NumeroRipetizioni

Poiché la parte sinistra è superchiave, PRESTAZIONE ESERCIZIO è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo Cliente della tabella PRESTAZIONE ESERCIZIO e l’attributo CodFiscale della tabella CLIENTE

Esiste un vincolo di integrità referenziale tra l’attributo Esercizio della tabella PRESTAZIONE ESERCIZIO e l’attributo CodEsercizio della tabella ESERCIZIO

**Tabella REGOLAZIONE REALE:**

REGOLAZIONE REALE(CodPrestazione, NomeRegolazione, Attrezzatura)

Poiché non esistono dipendenze funzionali, REGOLAZIONE REALE è in BCNF.

Esiste un vincolo di integrità referenziale tra l’attributo CodPrestazione della tabella REGOLAZIONE REALE e l’attributo CodPrestazione della tabella PRESTAZIONE ESERCIZIO

Esiste un vincolo di integrità referenziale tra l’attributo NomeRegolazione della tabella REGOLAZIONE REALE e l’attributo NomeRegolazione della tabella REGOLAZIONE

Esiste un vincolo di integrità referenziale tra l’attributo Attrezzatura della tabella REGOLAZIONE REALE e l’attributo CodAttrezzatura della tabella ATTREZZATURA

## 4.2- Schema del database

Questo è dunque lo schema del database:

CENTRO(CodCentro, Indirizzo, NumeroTelefonico, Dimensione, MaxClienti, ClientiAttualmentePresenti, Direttore)

TURNAZIONE(CodTurnazione, OraInizioTurno, OraFineTurno, Centro, Giorno, Dipendente)

RATA(CodRata, DataScadenza, Importo, Stato, Cliente, IstitutoFinanziario)

ISTITUTO FINANZIARIO(CodIstituto, RagioneSociale, TassoInteresse)

CONVENZIONE(Istituto, Centro)

ORARIO DI SERVIZIO(CodOrario, OrarioApertura, OrarioChiusura, Centro, Giorno)

GIORNO(Nome)

PIANIFICAZIONE CORSO(Giorno, Luogo, Corso, OraInizioCorso, OraFineCorso)

LUOGO DI ALLENAMENTO(CodLuogo, Centro)

SALA(Luogo, Nome, ResponsabileSala)

PISCINA(Luogo, Dimensione)

DIPENDENTE(CodFiscale, Nome, Cognome, DataNascita, Prefettura, CodDocumento, Indirizzo, ResponsabilePersonale)

IMPIEGO(Centro, Dipendente, Mansione)

ATTREZZATURA(CodAttrezzatura, Tipologia, ConsumoEnergetico, Usura, Sala)

REGOLAZIONE(Attrezzatura, NomeRegolazione)

CONFIGURAZIONE ESERCIZIO(CodConfigurazione, Attrezzatura)

REGOLAZIONE ESERCIZIO(Configurazione, Attrezzatura, NomeRegolazione, Valore)

SPOGLIATOIO(CodSpogliatoio, Capienza, PostiDisponibili, Posizione, Centro)

ARMADIETTO(CodArmadietto, Combinazione, Spogliatoio)

ACCESSO(CodAccesso, TimestampIngresso, TimestampUscita, Centro, Cliente, Armadietto)

MEDICO(CodFiscale)

VISITA(CodVisita, DataVisita, Medico, Cliente)

CLIENTE(CodFiscale, Nome, Cognome, Indirizzo, DataNascita, CodDocumento, Prefettura, Username, Password, VisiteMedico, Medico, Tutor, Scopo)

CORSO(CodCorso, DataInizioCorso, DataFineCorso, Livello, MaxPartecipanti, Istruttore, Disciplina)

ISTRUTTORE(CodFiscale)

RESPONSABILE PERSONALE(CodFiscale)

DIRETTORE(CodFiscale)

RESPONSABILE SALA(CodFiscale)

TUTOR(CodFiscale)

CONSULENTE(CodFiscale)

ISCRIZIONE(Cliente, Corso)

CONTRATTO(CodContratto, Durata, Tipologia, DataSottoscrizione, Importo, ModalitaDiPagamento,

Cliente, Consulente, Centro)

OFFERTA(CodOfferta, TipoOfferta, MaxIngressi, AccessiPiscine, CostoMensile, Centro)

INCLUSIONE OFFERTA(Contratto, Offerta)

OFFERTA LUOGO(Offerta, Luogo)

OFFERTA CORSO(Offerta, Corso)

SCOPO(NomeScopo)

MUSCOLO(NomeMuscolo)

TARGET(Muscolo, Cliente, LivelloPotenziamento)

AMICIZIA(Ricevente, Richiedente, Stato)

MISURAZIONE(Cliente, DataMisurazione, AcquaTotale, MassaMagra, MassaGrassa, Peso, Altezza, IMC, Stato)

ATTIVITA(Nome)

INTERESSE(Attivita, Cliente)

SCHEDA ALIMENTAZIONE(CodScheda, DataInizioScheda, DataFineScheda, FrequenzaVisita, Obiettivo, Cliente, Medico, Dieta)

DIETA(CodDieta, NumeroPasti, ApportoCalorico)

PASTO(Nome, Composizione)

COMPOSIZIONE DIETA(Dieta, Pasto)

INTEGRATORE(NomeCommerciale, Sostanza, Concentrazione, DataScadenza, NumeroPezzi, Forma, PrezzoIngrosso, PrezzoDettaglio)

ABBINAMENTO(SchedaAlimentazione, Integratore)

ACQUISTO(CodAcquisto, DataAcquisto, Quantita, Integratore, Cliente, Importo)

ORDINE(CodOrdine, Stato, DataEvasione, DataConsegna, Centro, Fornitore)

ORDINE INTEGRATORE(Integratore, Ordine, Quantita)

FORNITORE(PartitaIva, NomeCommerciale, NumeroTelefonico, Indirizzo, FormaSocietaria)

COMMERCIALIZZAZIONE(Fornitore, Integratore)

CERCHIA(CodCerchia, Nome, Cliente)

APPARTENENZA(Cerchia, Cliente)

POST(CodPost, Testo, Topic, Timestamp, Cliente, Thread)

COLLEGAMENTO(URL, Post)

THREAD(TitoloThread, Cliente)

VALUTAZIONE POST(Cliente, Post, Giudizio)

SCHEDA ALLENAMENTO(CodScheda, DataInizioScheda, DataFineScheda, Cliente, DataAssegnazione, Tutor)

ESERCIZIO(CodEsercizio, Nome, DispendioEnergetico, TotEsecuzioni, TotDurata, TotVoto)

COMPOSIZIONE(Scheda, Esercizio)

CONFIGURAZIONE(CodiceConf, Esercizio)

AEROBICO(CodEsercizio, Durata)

ANAEROBICO(CodEsercizio, TempoRecupero, NumeroRipetizioni)

RIPETIZIONE(CodEsercizio, NumeroRipetizione)

CONFIGURAZIONE RIPETIZIONE(CodConf, CodEsercizio, NumeroRipetizione)

SFIDA(CodSfida, DataInizio, DataFine, DataLancio, Scopo, Cliente)

PARTECIPAZIONE(Sfida, Cliente)

ABBINAMENTO ALIMENTAZIONE(Sfida, SchedaAlimentazione)

GIORNO SFIDA(CodSfida, Giorno, SchedaAllenamento)

VALUTAZIONE SFIDA(Cliente, CodSfida, Giorno, GiudizioPsichico, GiudizioFisico)

PRESTAZIONE ESERCIZIO(CodPrestazione, ValutazionePrestazione, TimestampInizio, TimestampFine, Cliente, Esercizio, Durata, TempoRecupero, NumeroRipetizioni)

REGOLAZIONE REALE(CodPrestazione, NomeRegolazione, Attrezzatura)

## 4.3- Vincoli di integrità referenziale

|  |  |
| --- | --- |
| RVIR | Descrizione |
| 1 | CodCentro di CENTRO e Centro di TURNAZIONE |
| 2 | Direttore di CENTRO e CodFiscale di DIRETTORE |
| 3 | Giorno di TURNAZIONE e Nome di GIORNO |
| 4 | Dipendente di TURNAZIONE e CodFiscale di DIPENDENTE |
| 5 | Cliente di RATA e CodFiscale di CLIENTE |
| 6 | IstitutoFinanziario di RATA e CodIstituto di ISTITUTO FINANZIARIO |
| 7 | Istituto di CONVENZIONE e CodIstituto di ISTITUTO FINANZIARIO |
| 8 | Centro di CONVENZIONE e CodCentro di CENTRO |
| 9 | Centro di ORARIO DI SERVIZIO e CodCentro di CENTRO |
| 10 | Giorno di ORARIO DI SERVIZIO e Nome di GIORNO |
| 11 | Giorno di PIANIFICAZIONE CORSO e Nome di GIORNO |
| 12 | Luogo di PIANIFICAZIONE CORSO e CodLuogo di LUOGO DI ALLENAMENTO |
| 13 | Corso di PIANIFICAZIONE CORSO e CodCorso di CORSO |
| 14 | Centro di LUOGO DI ALLENAMENTO e CodCentro di CENTRO |
| 15 | Luogo di SALA e CodLuogo di LUOGO DI ALLENAMENTO |
| 16 | ResponsabileSala di SALA e CodFiscale di RESPONSABILE SALA |
| 17 | Luogo di PISCINA e CodLuogo di LUOGO DI ALLENAMENTO |
| 18 | ResponsabilePersonale di DIPENDENTE e CodFiscale di RESPONSABILE PERSONALE |
| 19 | Centro di IMPIEGO e CodCentro di CENTRO |
| 20 | Dipendente di IMPIEGO e CodFiscale di DIPENDENTE |
| 21 | Sala di ATTREZZATURA e Luogo di SALA |
| 22 | Attrezzatura di REGOLAZIONE e CodAttrezzatura di ATTREZZATURA |
| 23 | Attrezzatura di CONFIGURAZIONE ESERCIZIO e CodAttrezzatura di ATTREZZATURA |
| 24 | Configurazione di REGOLAZIONE ESERCIZIO e CodConfigurazione di CONFIGURAZIONE ESERCIZIO |
| 25 | Attrezzatura di REGOLAZIONE ESERCIZIO e CodAttrezzatura di ATTREZZATURA |
| 26 | NomeRegolazione di REGOLAZIONE ESERCIZIO e NomeRegolazione di REGOLAZIONE |
| 27 | Centro di SPOGLIATOIO e CodCentro di CENTRO |
| 28 | Spogliatoio di ARMADIETTO e CodSpogliatoio di SPOGLIATOIO |
| 29 | Armadietto di ACCESSO e Armadietto di ARMADIETTO |
| 30 | Centro di ACCESSO e CodCentro di CENTRO |
| 31 | Cliente di ACCESSO e CodFiscale di CLIENTE |
| 32 | CodFiscale di MEDICO e CodFiscale di DIPENDENTE |
| 33 | Medico di VISITA e CodFiscale di MEDICO |
| 34 | Cliente di VISITA e CodFiscale di CLIENTE |
| 35 | Medico di CLIENTE e CodFiscale di MEDICO |
| 36 | Tutor di CLIENTE e CodFiscale di TUTOR |
| 37 | Scopo di CLIENTE e NomeScopo di SCOPO |
| 38 | Istruttore di CORSO e CodFiscale di ISTRUTTORE |
| 39 | CodFiscale di ISTRUTTORE e CodFiscale di DIPENDENTE |
| 40 | CodFiscale di RESPONSABILE PERSONALE e CodFiscale di DIPENDENTE |
| 41 | CodFiscale di DIRETTORE e CodFiscale di DIPENDENTE |
| 42 | CodFiscale di RESPONSABILE SALA e CodFiscale di DIPENDENTE |
| 43 | CodFiscale di TUTOR e CodFiscale di DIPENDENTE |
| 44 | CodFiscale di CONSULENTE e CodFiscale di DIPENDENTE |
| 45 | Cliente di ISCRIZIONE e CodFiscale di CLIENTE |
| 46 | Corso di ISCRIZIONE e CodCorso di CORSO |
| 47 | Cliente di CONTRATTO e CodFiscale di CLIENTE |
| 48 | Consulente di CONTRATTO e CodFiscale di CONSULENTE |
| 49 | Centro di CONTRATTO e CodCentro di CENTRO |
| 50 | Centro di OFFERTA e CodCentro di CENTRO |
| 51 | Contratto di INCLUSIONE OFFERTA e CodContratto di CONTRATTO |
| 52 | Offerta di INCLUSIONE OFFERTA e CodOfferta di OFFERTA |
| 53 | Offerta di ACCESSO LUOGO e CodOfferta di OFFERTA |
| 54 | Luogo di ACCESSO LUOGO e CodLuogo di LUOGO DI ALLENAMENTO |
| 55 | Offerta di ACCESSO CORSO e CodOfferta di OFFERTA |
| 56 | Corso di ACCESSO CORSO e CodCorso di CORSO |
| 57 | Muscolo di TARGET e NomeMuscolo di MUSCOLO |
| 58 | Cliente di TARGET e CodFiscale di CLIENTE |
| 59 | Richiedente di AMICIZIA e Username di CLIENTE |
| 60 | Ricevente di AMICIZIA e Username di CLIENTE |
| 61 | Cliente di MISURAZIONE e CodFiscale di CLIENTE |
| 62 | Attivita di INTERESSE e Nome di ATTIVITA |
| 63 | Cliente di INTERESSE e CodFiscale di CLIENTE |
| 64 | Obiettivo di SCHEDA DI ALIMENTAZIONE e NomeScopo di Scopo |
| 65 | Cliente di SCHEDA DI ALIMENTAZIONE e CodFiscale di CLIENTE |
| 66 | Medico di SCHEDA DI ALIMENTAZIONE e CodFiscale di MEDICO |
| 67 | Dieta di SCHEDA DI ALIMENTAZIONE e CodDieta di DIETA |
| 68 | Dieta di COMPOSIZIONE DIETA e CodDieta di DIETA |
| 69 | Pasto di COMPOSIZIONE DIETA e Nome di PASTO |
| 70 | SchedaAlimentazione di ABBINAMENTO e CodScheda di SCHEDA DI ALIMENTAZIONE |
| 71 | Integratore di ABBINAMENTO e NomeCommerciale di INTEGRATORE |
| 72 | Integratore di ACQUISTO e NomeCommerciale di INTEGRATORE |
| 73 | Cliente di ACQUISTO e CodFiscale di CLIENTE |
| 74 | Centro di ORDINE e CodCentro di CENTRO |
| 75 | Fornitore di ORDINE e PartitaIva di FORNITORE |
| 76 | Integratore di ORDINE INTEGRATORE e NomeCommerciale di INTEGRATORE |
| 77 | Ordine di ORDINE INTEGRATORE e CodOrdine di ORDINE |
| 78 | Fornitore di COMMERCIALIZZAZIONE e PartitaIva di FORNITORE |
| 79 | Integratore di COMMERCIALIZZAZIONE e NomeCommerciale di INTEGRATORE |
| 80 | Cliente di CERCHIA e CodFiscale di CLIENTE |
| 81 | Cerchia di APPARTENENZA e CodCerchia di CERCHIA |
| 82 | Cliente di APPARTENENZA e Username di CLIENTE |
| 83 | Cliente di POST e CodFiscale di CLIENTE |
| 84 | Thread di POST e TitoloThread di THREAD |
| 85 | Post di COLLEGAMENTO e CodPost di POST |
| 86 | Cliente di THREAD e CodFiscale di cliente |
| 87 | Cliente di VALUTAZIONE POST e Username di CLIENTE |
| 88 | Post di VALUTAZIONE POST e CodPost di POST |
| 89 | Cliente di SCHEDA DI ALLENAMENTO e CodFiscale di CLIENTE |
| 90 | Tutor di SCHEDA DI ALLENAMENTO e CodFiscale di TUTOR |
| 91 | Scheda di COMPOSIZIONE e CodScheda di SCHEDA DI ALLENAMENTO |
| 92 | Esercizio di COMPOSIZIONE e CodEsercizio di ESERCIZIO |
| 93 | Esercizio di CONFIGURAZIONE e CodEsercizio di ESERCIZIO |
| 94 | CodEsercizio di AEROBICO e CodEsercizio di ESERCIZIO |
| 95 | CodEsercizio di ANAEROBICO e CodEsercizio di ESERCIZIO |
| 96 | CodEsercizio di Ripetizione e CodEsercizio di ESERCIZIO |
| 97 | CodConf di CONFIGURAZIONE RIPETIZIONE e CodiceConf di CONFIGURAZIONE |
| 98 | CodEsercizio di CONFIGURAZIONE RIPETIZIONE e CodEsercizio di ESERCIZIO |
| 99 | NumeroRipetizione di CONFIGURAZIONE RIPETIZIONE e NumeroRipetizione di RIPETIZIONE |
| 100 | Scopo di SFIDA e NomeScopo di SCOPO |
| 101 | Cliente di SFIDA e CodFiscale di CLIENTE |
| 102 | Sfida di PARTECIPAZIONE e CodSfida di SFIDA |
| 103 | Cliente di partecipazione e CodFiscale di CLIENTE |
| 104 | Sfida di ABBINAMENTO ALIMENTAZIONE e CodSfida di SFIDA |
| 105 | SchedaAlimentazione di ABBINAMENTO ALIMENTAZIONE e CodScheda di SCHEDA DI ALIMENTAZIONE |
| 106 | CodSfida di GIORNO SFIDA e CodSfida di SFIDA |
| 107 | Giorno di GIORNO SFIDA e Nome di GIORNO |
| 108 | SchedaAllenamento di GIORNO SFIDA e CodScheda di SCHEDA DI ALLENAMENTO |
| 109 | Cliente di VALUTAZIONE SFIDA e CodFiscale di CLIENTE |
| 110 | CodSfida di VALUTAZIONE SFIDA e CodSfida di SFIDA |
| 111 | Giorno di VALUTAZIONE SFIDA e Nome di GIORNO |
| 112 | Cliente di PRESTAZIONE ESERCIZIO e CodFiscale di CLIENTE |
| 113 | Esercizio di PRESTAZIONE ESERCIZIO e CodEsercizio di ESERCIZIO |
| 114 | CodPrestazione di REGOLAZIONE REALE e CodPrestazione di PRESTAZIONE ESERCIZIO |
| 115 | NomeRegolazione di REGOLAZIONE REALE e NomeRegolazione di REGOLAZIONE |
| 116 | Attrezzatura di REGOLAZIONE REALE e CodAttrezzatura di ATTREZZATURA |

## 4.4- Vincoli di integrità generici

|  |  |
| --- | --- |
| RVIG | Descrizione |
| 1 | Non si possono inserire due turnazioni di un dipendente con orari sovrapposti. |
| 2 | Un dipendente deve avere un orario di lavoro giornaliero di massimo 8 ore. |
| 3 | DataFineCorso deve essere maggiore di DataInizioCorso. |
| 4 | Un contratto può includere offerte relative al massimo a 3 centri diversi. |
| 5 | DataFineScheda deve essere maggiore di DataInizioScheda (allenamento) |
| 6 | Non possono esistere due schede d’alimentazione attive contemporaneamente per uno stesso cliente. |
| 7 | DataFineScheda deve essere maggiore di DataInizioScheda (alimentazione) |
| 8 | DataFine deve essere maggiore di DataInizio (sfida). |

# Implementazione su DBMS Oracle MySQL

## 5.1- Definizione schema

SET NAMES latin1;

BEGIN;

CREATE DATABASE IF NOT EXISTS fitness;

COMMIT;

USE fitness;

## 5.2- Definizione tabelle

SET NAMES latin1;

BEGIN;

CREATE DATABASE IF NOT EXISTS fitness;

COMMIT;

USE fitness;

CREATE TABLE Centro (

CodCentro INTEGER NOT NULL AUTO\_INCREMENT,

Indirizzo VARCHAR(255) NOT NULL,

NumeroTelefonico INTEGER NOT NULL,

Dimensione INTEGER NOT NULL,

MaxClienti INTEGER NOT NULL,

ClientiPresenti INTEGER NOT NULL,

Direttore VARCHAR(255) NOT NULL,

PRIMARY KEY (CodCentro)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Turnazione (

CodTurnazione INTEGER NOT NULL AUTO\_INCREMENT,

OraInizioTurno TIME NOT NULL,

OraFineTurno TIME NOT NULL,

Centro INTEGER NOT NULL,

Giorno VARCHAR(255) NOT NULL,

Dipendente VARCHAR(255) NOT NULL,

PRIMARY KEY (CodTurnazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Rata (

CodRata INTEGER NOT NULL AUTO\_INCREMENT,

DataScadenza DATE NOT NULL,

Importo DOUBLE NOT NULL,

Stato ENUM('Eseguito','Non ancora dovuto', 'Scaduto'),

Cliente VARCHAR(255) NOT NULL,

IstitutoFinanziario INTEGER NOT NULL,

PRIMARY KEY (CodRata)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE IstitutoFinanziario (

CodIstituto INTEGER NOT NULL AUTO\_INCREMENT,

RagioneSociale VARCHAR(255) NOT NULL,

TassoInteresse DOUBLE NOT NULL,

PRIMARY KEY (CodIstituto)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Convenzione (

Istituto INTEGER NOT NULL,

Centro INTEGER NOT NULL,

PRIMARY KEY (Istituto , Centro)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE OrarioDiServizio (

CodOrario INTEGER NOT NULL AUTO\_INCREMENT,

OrarioApertura TIME NOT NULL,

OrarioChiusura TIME NOT NULL,

Centro INTEGER NOT NULL,

Giorno VARCHAR(255) NOT NULL,

PRIMARY KEY (CodOrario)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Giorno (

Nome VARCHAR(255) NOT NULL,

PRIMARY KEY (Nome)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE PianificazioneCorso (

Giorno VARCHAR(255) NOT NULL,

Luogo INTEGER NOT NULL,

Corso INTEGER NOT NULL,

OraInizioCorso TIME NOT NULL,

OraFineCorso TIME NOT NULL,

PRIMARY KEY (Giorno , Luogo , Corso)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE LuogoAllenamento (

CodLuogo INTEGER NOT NULL AUTO\_INCREMENT,

Centro INTEGER NOT NULL,

PRIMARY KEY (CodLuogo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Sala (

Luogo INTEGER NOT NULL AUTO\_INCREMENT,

Nome VARCHAR(255) NOT NULL,

ResponsabileSala VARCHAR(255) NOT NULL,

PRIMARY KEY (Luogo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Piscina (

Luogo INTEGER NOT NULL AUTO\_INCREMENT,

Dimensione DOUBLE NOT NULL,

PRIMARY KEY (Luogo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Dipendente (

CodFiscale VARCHAR(255) NOT NULL,

Nome VARCHAR(255) NOT NULL,

Cognome VARCHAR(255) NOT NULL,

DataNascita DATE NOT NULL,

Prefettura VARCHAR(255) NOT NULL,

CodDocumento INTEGER NOT NULL,

Indirizzo VARCHAR(255) NOT NULL,

ResponsabilePersonale VARCHAR(255),

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Impiego (

Centro INTEGER NOT NULL,

Dipendente VARCHAR(255) NOT NULL,

Mansione VARCHAR(255) NOT NULL,

PRIMARY KEY (Centro , Dipendente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Attrezzatura (

CodAttrezzatura INTEGER NOT NULL AUTO\_INCREMENT,

Tipologia VARCHAR(255) NOT NULL,

ConsumoEnergetico INTEGER NOT NULL,

Usura INTEGER NOT NULL,

Sala INTEGER NOT NULL,

PRIMARY KEY (CodAttrezzatura)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Regolazione (

Attrezzatura INTEGER NOT NULL,

NomeRegolazione VARCHAR(255) NOT NULL,

PRIMARY KEY (Attrezzatura , NomeRegolazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ConfigurazioneEsercizio (

CodConfigurazione INTEGER NOT NULL AUTO\_INCREMENT,

Esercizio INTEGER NOT NULL,

Attrezzatura INTEGER NOT NULL,

PRIMARY KEY (CodConfigurazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE RegolazioneEsercizio (

Configurazione INTEGER NOT NULL,

Attrezzatura INTEGER NOT NULL,

NomeRegolazione VARCHAR(255) NOT NULL,

Valore INTEGER NOT NULL,

PRIMARY KEY (Configurazione , Attrezzatura , NomeRegolazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Spogliatoio (

CodSpogliatoio INTEGER NOT NULL AUTO\_INCREMENT,

Capienza INTEGER NOT NULL,

PostiDisponibili INTEGER NOT NULL,

Posizione ENUM('N', 'S', 'E', 'W') NOT NULL,

Centro INTEGER NOT NULL,

PRIMARY KEY (CodSpogliatoio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Armadietto (

CodArmadietto INTEGER NOT NULL AUTO\_INCREMENT,

Combinazione INTEGER NOT NULL,

Spogliatoio INTEGER NOT NULL,

PRIMARY KEY (CodArmadietto)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Accesso (

CodAccesso INTEGER NOT NULL AUTO\_INCREMENT,

TimestampIngresso DATETIME NOT NULL,

TimestampUscita DATETIME,

Centro INTEGER NOT NULL,

Cliente VARCHAR(255) NOT NULL,

Armadietto INTEGER NOT NULL,

PRIMARY KEY (CodAccesso)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Medico (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Visita (

CodVisita INTEGER NOT NULL AUTO\_INCREMENT,

DataVisita DATE NOT NULL,

Medico VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (CodVisita)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Cliente (

CodFiscale VARCHAR(255) NOT NULL,

Nome VARCHAR(255) NOT NULL,

Cognome VARCHAR(255) NOT NULL,

Indirizzo VARCHAR(255) NOT NULL,

DataNascita DATE NOT NULL,

CodDocumento INTEGER NOT NULL,

Prefettura VARCHAR(255) NOT NULL,

Username VARCHAR(255) NOT NULL,

Password VARCHAR(255) NOT NULL,

VisiteMedico INTEGER NOT NULL,

Medico VARCHAR(255) NOT NULL,

Tutor VARCHAR(255) NOT NULL,

Scopo ENUM('Dimagrimento', 'PotenziamentoMuscolare', 'Ricreativo') NOT NULL,

PRIMARY KEY (CodFiscale),

UNIQUE(Username)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Corso (

CodCorso INTEGER NOT NULL AUTO\_INCREMENT,

DataInizioCorso DATE NOT NULL,

DataFineCorso DATE NOT NULL,

Livello ENUM('Principiante', 'Avanzato', 'Esperto', 'Campione', 'Agonista') NOT NULL,

MaxPartecipanti INTEGER NOT NULL,

Istruttore VARCHAR(255) NOT NULL,

Disciplina VARCHAR(255) NOT NULL,

PRIMARY KEY (CodCorso)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Istruttore (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ResponsabilePersonale (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Direttore (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ResponsabileSala (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Tutor (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Consulente (

CodFiscale VARCHAR(255) NOT NULL,

PRIMARY KEY (CodFiscale)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Iscrizione (

Cliente VARCHAR(255) NOT NULL,

Corso INTEGER NOT NULL,

PRIMARY KEY (Cliente , Corso)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Contratto (

CodContratto INTEGER NOT NULL AUTO\_INCREMENT,

Durata INTEGER NOT NULL,

Tipologia ENUM('Standard', 'Personalizzato','Multisede') NOT NULL,

DataSottoscrizione DATE NOT NULL,

Importo DOUBLE NOT NULL,

ModalitaDiPagamento ENUM('UnicaSoluzione', 'Rateale') NOT NULL,

Cliente VARCHAR(255) NOT NULL,

Consulente VARCHAR(255) NOT NULL,

Centro INTEGER NOT NULL,

PRIMARY KEY (CodContratto)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Offerta (

CodOfferta INTEGER NOT NULL AUTO\_INCREMENT,

TipoOfferta VARCHAR(255) NOT NULL,

MaxIngressi INTEGER NOT NULL,

AccessiPiscine INTEGER NOT NULL,

CostoMensile DOUBLE NOT NULL,

Centro INTEGER NOT NULL,

PRIMARY KEY (CodOfferta)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE InclusioneOfferta (

Contratto INTEGER NOT NULL,

Offerta INTEGER NOT NULL,

PRIMARY KEY (Contratto , Offerta)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE OffertaLuogo (

Offerta INTEGER NOT NULL,

Luogo INTEGER NOT NULL,

PRIMARY KEY (Offerta , Luogo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE OffertaCorso (

Offerta INTEGER NOT NULL,

Corso INTEGER NOT NULL,

PRIMARY KEY (Offerta , Corso)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Scopo (

NomeScopo ENUM('Dimagrimento', 'PotenziamentoMuscolare', 'Ricreativo') NOT NULL,

PRIMARY KEY (NomeScopo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Muscolo (

NomeMuscolo VARCHAR(255) NOT NULL,

PRIMARY KEY (NomeMuscolo)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Target (

Muscolo VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

LivelloPotenziamento ENUM('lieve', 'moderato', 'elevato') NOT NULL,

PRIMARY KEY (Muscolo , Cliente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Amicizia (

Ricevente VARCHAR(255) NOT NULL,

Richiedente VARCHAR(255) NOT NULL,

Stato ENUM ('Accettata', 'Rifiutata', 'In attesa'),

PRIMARY KEY (Ricevente , Richiedente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Misurazione (

Cliente VARCHAR(255) NOT NULL,

DataMisurazione DATE NOT NULL,

AcquaTotale DOUBLE NOT NULL,

MassaMagra DOUBLE NOT NULL,

MassaGrassa DOUBLE NOT NULL,

Peso DOUBLE NOT NULL,

Altezza DOUBLE NOT NULL,

IMC DOUBLE NOT NULL,

Stato ENUM('Sovrappeso', 'Normopeso', 'Sottopeso'),

PRIMARY KEY (Cliente , DataMisurazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Attivita (

Nome VARCHAR(255) NOT NULL,

PRIMARY KEY (Nome)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Interesse (

Attivita VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (Attivita , Cliente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE SchedaAlimentazione (

CodScheda INTEGER NOT NULL AUTO\_INCREMENT,

DataInizioScheda DATE NOT NULL,

DataFineScheda DATE,

FrequenzaVisita INTEGER NOT NULL,

Obiettivo VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

Medico VARCHAR(255) NOT NULL,

Dieta INTEGER NOT NULL,

PRIMARY KEY (CodScheda)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Dieta (

CodDieta INTEGER NOT NULL AUTO\_INCREMENT,

NumeroPasti INTEGER NOT NULL,

ApportoCalorico DOUBLE NOT NULL,

PRIMARY KEY (CodDieta)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Pasto (

Nome VARCHAR(255) NOT NULL,

Composizione ENUM('Proteine', 'Grassi', 'Carboidrati') NOT NULL,

PRIMARY KEY (Nome)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ComposizioneDieta (

Dieta INTEGER NOT NULL,

Pasto VARCHAR(255) NOT NULL,

PRIMARY KEY (Dieta , Pasto)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Integratore (

CodIntegratore INTEGER NOT NULL AUTO\_INCREMENT,

NomeCommerciale VARCHAR(255) NOT NULL,

Sostanza VARCHAR(255) NOT NULL,

Concentrazione DOUBLE NOT NULL,

DataScadenza DATE NOT NULL,

NumeroPezzi INTEGER NOT NULL,

Forma ENUM('Compressa', 'Liquido', 'Polvere') NOT NULL,

PrezzoIngrosso DOUBLE NOT NULL,

PrezzoDettaglio DOUBLE NOT NULL,

PRIMARY KEY (CodIntegratore)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Abbinamento (

SchedaAlimentazione INTEGER NOT NULL,

Integratore INTEGER NOT NULL,

PRIMARY KEY (SchedaAlimentazione , Integratore)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Acquisto (

CodAcquisto INTEGER NOT NULL AUTO\_INCREMENT,

DataAcquisto DATE NOT NULL,

Quantita INTEGER NOT NULL,

Integratore INTEGER NOT NULL,

Cliente VARCHAR(255) NOT NULL,

Importo DOUBLE NOT NULL,

PRIMARY KEY (CodAcquisto)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Ordine (

CodOrdine INTEGER NOT NULL AUTO\_INCREMENT,

Stato ENUM('Incompleto', 'Evaso'),

DataEvasione DATE,

DataConsegna DATE NOT NULL,

Centro INTEGER NOT NULL,

Fornitore INTEGER NOT NULL,

PRIMARY KEY (CodOrdine)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE OrdineIntegratore (

Integratore INTEGER NOT NULL,

Ordine INTEGER NOT NULL,

Quantita INTEGER NOT NULL,

PRIMARY KEY (Integratore, Ordine)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Fornitore (

PartitaIva INTEGER NOT NULL,

NomeCommerciale VARCHAR(255) NOT NULL,

NumeroTelefonico INTEGER NOT NULL,

Indirizzo VARCHAR(255) NOT NULL,

FormaSocietaria VARCHAR(255) NOT NULL,

PRIMARY KEY (PartitaIva)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Commercializzazione (

Fornitore INTEGER NOT NULL,

Integratore INTEGER NOT NULL,

CodEsterno VARCHAR(255) NOT NULL,

PRIMARY KEY (Fornitore , Integratore)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Cerchia (

CodCerchia INTEGER NOT NULL AUTO\_INCREMENT,

Nome VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (CodCerchia)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Appartenenza (

Cerchia INTEGER NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (Cerchia , Cliente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Post (

CodPost INTEGER NOT NULL AUTO\_INCREMENT,

Testo VARCHAR(255) NOT NULL,

Topic VARCHAR(255) NOT NULL,

Timestamp DATETIME NOT NULL,

Cliente VARCHAR(255) NOT NULL,

Thread VARCHAR(255) NOT NULL,

PRIMARY KEY (CodPost)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Collegamento (

URL VARCHAR(255) NOT NULL,

Post INTEGER NOT NULL,

PRIMARY KEY (URL , Post)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Thread (

TitoloThread VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (TitoloThread)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ValutazionePost (

Cliente VARCHAR(255) NOT NULL,

Post INTEGER NOT NULL,

Giudizio ENUM('Positivo', 'Negativo') NOT NULL,

PRIMARY KEY (Cliente , Post)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE SchedaAllenamento (

CodScheda INTEGER NOT NULL AUTO\_INCREMENT,

DataInizioScheda DATE NOT NULL,

DataFineScheda DATE NOT NULL,

Cliente VARCHAR(255) NOT NULL,

DataAssegnazione DATE NOT NULL,

Tutor VARCHAR(255) NOT NULL,

PRIMARY KEY (CodScheda)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Esercizio (

CodEsercizio INTEGER NOT NULL AUTO\_INCREMENT,

Nome VARCHAR(255) NOT NULL,

DispendioEnergetico DOUBLE NOT NULL,

TotEsecuzioni INTEGER NOT NULL,

TotDurata INTEGER NOT NULL,

TotVoto INTEGER NOT NULL,

PRIMARY KEY (CodEsercizio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Composizione (

Scheda INTEGER NOT NULL,

Esercizio INTEGER NOT NULL,

PRIMARY KEY (Scheda , Esercizio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Configurazione (

CodiceConf INTEGER NOT NULL,

Esercizio INTEGER NOT NULL,

PRIMARY KEY (CodiceConf , Esercizio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Aerobico (

CodEsercizio INTEGER NOT NULL AUTO\_INCREMENT,

Durata INTEGER NOT NULL,

PRIMARY KEY (CodEsercizio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Anaerobico (

CodEsercizio INTEGER NOT NULL AUTO\_INCREMENT,

TempoRecupero INTEGER NOT NULL,

NumeroRipetizioni INTEGER NOT NULL,

PRIMARY KEY (CodEsercizio)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Ripetizione (

CodEsercizio INTEGER NOT NULL,

NumeroRipetizione INTEGER NOT NULL,

PRIMARY KEY (CodEsercizio , NumeroRipetizione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ConfigurazioneRipetizione (

CodConf INTEGER NOT NULL,

NumeroRipetizione INTEGER NOT NULL,

Esercizio INTEGER NOT NULL,

PRIMARY KEY (CodConf, Esercizio, NumeroRipetizione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Sfida (

CodSfida INTEGER NOT NULL AUTO\_INCREMENT,

DataInizio DATE NOT NULL,

DataFine DATE NOT NULL,

DataLancio DATE NOT NULL,

Scopo VARCHAR(255) NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (CodSfida)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE Partecipazione (

Sfida INTEGER NOT NULL,

Cliente VARCHAR(255) NOT NULL,

PRIMARY KEY (Sfida , Cliente)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE AbbinamentoAlimentazione (

Sfida INTEGER NOT NULL,

SchedaAlimentazione INTEGER NOT NULL,

PRIMARY KEY (Sfida , SchedaAlimentazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE GiornoSfida (

CodSfida INTEGER NOT NULL,

Giorno INTEGER NOT NULL,

SchedaAllenamento INTEGER NOT NULL,

PRIMARY KEY (CodSfida , Giorno)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE ValutazioneSfida (

Cliente VARCHAR(255) NOT NULL,

CodSfida INTEGER NOT NULL,

Giorno INTEGER NOT NULL,

GiudizioPsichico ENUM('Positivo', 'Negativo') NOT NULL,

GiudizioFisico ENUM('Positivo', 'Negativo') NOT NULL,

PRIMARY KEY (Cliente , CodSfida , Giorno)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE PrestazioneEsercizio (

CodPrestazione INTEGER NOT NULL AUTO\_INCREMENT,

ValutazionePrestazione DOUBLE,

TimestampInizio DATETIME NOT NULL,

TimestampFine DATETIME,

Cliente VARCHAR(255) NOT NULL,

Esercizio INTEGER NOT NULL,

Durata INTEGER,

TempoRecupero INTEGER,

NumeroRipetizioni INTEGER,

PRIMARY KEY (CodPrestazione)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

CREATE TABLE RegolazioneReale (

CodPrestazione INTEGER NOT NULL,

NomeRegolazione VARCHAR(255) NOT NULL,

Attrezzatura INTEGER NOT NULL,

PRIMARY KEY (CodPrestazione , NomeRegolazione , Attrezzatura)

) ENGINE=INNODB DEFAULT CHARSET=LATIN1;

ALTER TABLE Centro

ADD FOREIGN KEY(Direttore)

REFERENCES DIRETTORE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Turnazione

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Giorno)

REFERENCES GIORNO(Nome)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Dipendente)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Rata

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(IstitutoFinanziario)

REFERENCES ISTITUTOFINANZIARIO(CodIstituto)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Convenzione

ADD FOREIGN KEY(Istituto)

REFERENCES ISTITUTOFINANZIARIO(CodIstituto)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE OrarioDiServizio

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Giorno)

REFERENCES GIORNO(Nome)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE PianificazioneCorso

ADD FOREIGN KEY(Giorno)

REFERENCES GIORNO(Nome)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Luogo)

REFERENCES LUOGOALLENAMENTO(CodLuogo)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Corso)

REFERENCES CORSO(CodCorso)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE LUOGOAllenamento

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Sala

ADD FOREIGN KEY(Luogo)

REFERENCES LUOGOALLENAMENTO(CodLuogo)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(ResponsabileSala)

REFERENCES RESPONSABILESALA(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Piscina

ADD FOREIGN KEY(Luogo)

REFERENCES LUOGOALLENAMENTO(CodLuogo)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Dipendente

ADD FOREIGN KEY(ResponsabilePersonale)

REFERENCES RESPONSABILEPERSONALE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Impiego

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Dipendente)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Attrezzatura

ADD FOREIGN KEY(Sala)

REFERENCES SALA(Luogo)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Regolazione

ADD FOREIGN KEY(Attrezzatura)

REFERENCES ATTREZZATURA(CodAttrezzatura)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ConfigurazioneEsercizio

ADD FOREIGN KEY(Esercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Attrezzatura)

REFERENCES ATTREZZATURA(CodAttrezzatura)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE RegolazioneEsercizio

ADD FOREIGN KEY(Configurazione)

REFERENCES CONFIGURAZIONEESERCIZIO(CodConfigurazione)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Attrezzatura)

REFERENCES ATTREZZATURA(CodAttrezzatura)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Spogliatoio

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Armadietto

ADD FOREIGN KEY(Spogliatoio)

REFERENCES Spogliatoio(CodSpogliatoio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Accesso

ADD FOREIGN KEY(Armadietto)

REFERENCES ARMADIETTO(CodArmadietto)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Medico

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Visita

ADD FOREIGN KEY(Medico)

REFERENCES Medico(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Cliente

ADD FOREIGN KEY(Medico)

REFERENCES MEDICO(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Tutor)

REFERENCES TUTOR(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Scopo)

REFERENCES SCOPO(NomeScopo)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Corso

ADD FOREIGN KEY(Istruttore)

REFERENCES ISTRUTTORE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Istruttore

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ResponsabilePersonale

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Direttore

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ResponsabileSala

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Tutor

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Consulente

ADD FOREIGN KEY(CodFiscale)

REFERENCES DIPENDENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Iscrizione

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Corso)

REFERENCES CORSO(CodCorso)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Contratto

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Consulente)

REFERENCES CONSULENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Offerta

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE InclusioneOfferta

ADD FOREIGN KEY(Contratto)

REFERENCES CONTRATTO(CodContratto)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Offerta)

REFERENCES OFFERTA(CodOfferta)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE OffertaLuogo

ADD FOREIGN KEY(Offerta)

REFERENCES OFFERTA(CodOfferta)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Luogo)

REFERENCES LUOGOALLENAMENTO(CodLuogo)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE OffertaCorso

ADD FOREIGN KEY(Offerta)

REFERENCES OFFERTA(CodOfferta)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Corso)

REFERENCES CORSO(CodCorso)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Target

ADD FOREIGN KEY(Muscolo)

REFERENCES MUSCOLO(NomeMuscolo)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Amicizia

ADD FOREIGN KEY(Ricevente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Richiedente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Misurazione

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Interesse

ADD FOREIGN KEY(Attivita)

REFERENCES ATTIVITA(Nome)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE SchedaAlimentazione

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Medico)

REFERENCES MEDICO(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Dieta)

REFERENCES DIETA(CodDieta)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ComposizioneDieta

ADD FOREIGN KEY(Dieta)

REFERENCES DIETA(CodDieta)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Pasto)

REFERENCES PASTO(Nome)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Abbinamento

ADD FOREIGN KEY(SchedaAlimentazione)

REFERENCES SCHEDAALIMENTAZIONE(CodScheda)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Integratore)

REFERENCES INTEGRATORE(CodIntegratore)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Acquisto

ADD FOREIGN KEY(Integratore)

REFERENCES INTEGRATORE(CodIntegratore)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Ordine

ADD FOREIGN KEY(Centro)

REFERENCES CENTRO(CodCentro)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Fornitore)

REFERENCES FORNITORE(PartitaIva)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE OrdineIntegratore

ADD FOREIGN KEY(Integratore)

REFERENCES INTEGRATORE(CodIntegratore)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Ordine)

REFERENCES ORDINE(CodOrdine)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Commercializzazione

ADD FOREIGN KEY(Fornitore)

REFERENCES FORNITORE(PartitaIva)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Integratore)

REFERENCES INTEGRATORE(CodIntegratore)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Cerchia

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Appartenenza

ADD FOREIGN KEY(Cerchia)

REFERENCES CERCHIA(CodCerchia)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Post

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Thread)

REFERENCES THREAD(TitoloThread)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Collegamento

ADD FOREIGN KEY(Post)

REFERENCES POST(CodPost)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Thread

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ValutazionePost

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Post)

REFERENCES POST(CodPost)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE SchedaAllenamento

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Tutor)

REFERENCES TUTOR(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Composizione

ADD FOREIGN KEY(Scheda)

REFERENCES SCHEDAALLENAMENTO(CodScheda)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Esercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Configurazione

ADD FOREIGN KEY(Esercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Aerobico

ADD FOREIGN KEY(CodEsercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Anaerobico

ADD FOREIGN KEY(CodEsercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Ripetizione

ADD FOREIGN KEY(CodEsercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ConfigurazioneRipetizione

ADD FOREIGN KEY(CodConf)

REFERENCES CONFIGURAZIONEESERCIZIO(CodConfigurazione)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Sfida

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE Partecipazione

ADD FOREIGN KEY(Sfida)

REFERENCES SFIDA(CodSfida)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE AbbinamentoAlimentazione

ADD FOREIGN KEY(Sfida)

REFERENCES SFIDA(CodSfida)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(SchedaAlimentazione)

REFERENCES SCHEDAALIMENTAZIONE(CodScheda)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE GiornoSfida

ADD FOREIGN KEY(CodSfida)

REFERENCES SFIDA(CodSfida)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(SchedaAllenamento)

REFERENCES SCHEDAALLENAMENTO(CodScheda)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE ValutazioneSfida

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(CodSfida)

REFERENCES SFIDA(CodSfida)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE PrestazioneEsercizio

ADD FOREIGN KEY(Cliente)

REFERENCES CLIENTE(CodFiscale)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Esercizio)

REFERENCES ESERCIZIO(CodEsercizio)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE RegolazioneReale

ADD FOREIGN KEY(CodPrestazione)

REFERENCES PRESTAZIONEESERCIZIO(CodPrestazione)

ON UPDATE NO ACTION

ON DELETE NO ACTION,

ADD FOREIGN KEY(Attrezzatura)

REFERENCES ATTREZZATURA(CodAttrezzatura)

ON UPDATE NO ACTION

ON DELETE NO ACTION;

## 5.3- Implementazione delle operazioni significative

### 5.3.1- Operazione 1

DELIMITER $$

DROP PROCEDURE IF EXISTS NuovaSchedaAllenamento$$

CREATE PROCEDURE NuovaSchedaAllenamento (IN data\_inizio DATE,

IN data\_fine DATE,

IN cliente VARCHAR(255),

IN tutor VARCHAR(255),

IN esercizio INTEGER)

BEGIN

DECLARE cod\_scheda INTEGER DEFAULT 0;

DECLARE data\_odierna varchar(255) DEFAULT '';

SET data\_odierna = CURRENT\_DATE;

INSERT INTO SchedaAllenamento(DataInizioScheda,

DataFineScheda,

Cliente,

DataAssegnazione,

Tutor)

VALUES (data\_inizio, data\_fine, cliente, data\_odierna, tutor);

SELECT CodScheda INTO cod\_scheda

FROM SchedaAllenamento

WHERE DataAssegnazione = data\_odierna

AND Cliente = cliente

AND Tutor = tutor

AND DataInizioScheda = data\_inizio

AND DataFineScheda = data\_fine;

INSERT INTO Composizione

VALUES (cod\_scheda, esercizio);

END$$

DELIMITER ;

### 5.3.2- Operazione 2

DELIMITER $$

DROP PROCEDURE IF EXISTS IscrittiCorso$$

CREATE PROCEDURE IscrittiCorso (IN corso INTEGER)

BEGIN

SELECT C.\*

FROM Iscrizione I

INNER JOIN Cliente C ON I.Cliente = C.CodFiscale

WHERE I.Corso = corso;

END$$

DELIMITER ;

### 5.3.3- Operazione 3

DELIMITER $$

DROP PROCEDURE IF EXISTS EserciziSvolti$$

CREATE PROCEDURE EserciziSvolti ()

BEGIN

SELECT PE.Esercizio, COUNT(\*)

FROM PrestazioneEsercizio PE

WHERE PE.TimestampFine IS NULL

GROUP BY PE.Esercizio;

END$$

DELIMITER ;

### 5.3.4- Operazione 4

DELIMITER $$

DROP PROCEDURE IF EXISTS ArmadiettiLiberi$$

CREATE PROCEDURE ArmadiettiLiberi (IN centro INTEGER)

BEGIN

SELECT AR.Spogliatoio, AR.CodArmadietto

FROM Armadietto AR

INNER JOIN Spogliatoio SP

ON AR.Spogliatoio = SP.CodSpogliatoio

WHERE SP.Centro = centro

AND NOT EXISTS (

SELECT \*

FROM Accesso AC

WHERE AC.TimestampUscita IS NULL

AND AC.Armadietto=AR.CodArmadietto

)

ORDER BY AR.Spogliatoio, AR.CodArmadietto;

END$$

DELIMITER ;

### 5.3.5- Operazione 5

#### 5.3.5.1- Senza ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS TotVisite$$

CREATE PROCEDURE TotVisite (IN cliente VARCHAR(255))

BEGIN

DECLARE medico VARCHAR(255) DEFAULT 0;

DECLARE inizio\_scheda VARCHAR(255) DEFAULT "";

SELECT C.Medico INTO medico

FROM Cliente C

WHERE C.CodFiscale = cliente;

SELECT SA.DataInizioScheda INTO inizio\_scheda

FROM SchedaAlimentazione SA

WHERE SA.Cliente = cliente

AND CURRENT\_DATE BETWEEN SA.DataInizioScheda

AND SA.DataFineScheda;

IF inizio\_scheda IS NOT NULL THEN

SELECT COUNT(\*) AS TotaleVisite

FROM Visita V

WHERE V.Cliente = cliente

AND V.Medico = medico

AND V.DataVisita > inizio\_scheda;

ELSE

SELECT 0 AS TotaleVisite;

END IF;

END$$

DELIMITER ;

#### 5.3.5.2- Con ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS TotVisite\_Rid$$

CREATE PROCEDURE TotVisite\_Rid (IN cliente VARCHAR(255))

BEGIN

SELECT C.VisiteMedico

FROM Cliente C

WHERE C.CodFiscale = cliente;

END$$

DELIMITER ;

### 5.3.6- Operazione 6

#### 5.3.6.1- Senza ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS ModificaFrequenzaVisite$$

CREATE PROCEDURE ModificaFrequenzaVisite (IN cliente VARCHAR(255))

BEGIN

DECLARE medico VARCHAR(255) DEFAULT 0;

DECLARE inizio\_scheda VARCHAR(255) DEFAULT '';

DECLARE totale\_visite INTEGER DEFAULT 0;

SELECT C.Medico INTO medico

FROM Cliente C

WHERE C.CodFiscale = cliente;

SELECT SA.DataInizioScheda INTO inizio\_scheda

FROM SchedaAlimentazione SA

WHERE SA.Cliente = cliente

AND(

(CURRENT\_DATE BETWEEN SA.DataInizioScheda

AND SA.DataFineScheda)

OR

(CURRENT\_DATE > SA.DataInizioScheda

AND SA.DataFineScheda IS NULL)

);

IF inizio\_scheda IS NOT NULL THEN

SELECT COUNT(\*) INTO totale\_visite

FROM Visita V

WHERE V.Cliente = cliente

AND V.Medico = medico

AND V.DataVisita > inizio\_scheda;

UPDATE SchedaAlimentazione SA

SET SA.FrequenzaVisita = ROUND(

DATEDIFF(CURRENT\_DATE, inizio\_scheda)

/totale\_visite

)

WHERE SA.Cliente = cliente

AND(

(CURRENT\_DATE BETWEEN SA.DataInizioScheda

AND SA.DataFineScheda)

OR

(CURRENT\_DATE > SA.DataInizioScheda

AND SA.DataFineScheda IS NULL)

);

ELSE

SELECT "Impossibile aggiornare FrequenzaVisite.

Non esiste una scheda per il paziente.";

END IF;

END$$

DELIMITER ;

#### 5.3.6.2- Con ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS ModificaFrequenzaVisite\_Rid$$

CREATE PROCEDURE ModificaFrequenzaVisite\_Rid (IN cliente

VARCHAR(255))

BEGIN

DECLARE inizio\_scheda VARCHAR(255) DEFAULT '';

DECLARE totale\_visite INTEGER DEFAULT 0;

SELECT SA.DataInizioScheda INTO inizio\_scheda

FROM SchedaAlimentazione SA

WHERE SA.Cliente = cliente

AND(

(CURRENT\_DATE BETWEEN SA.DataInizioScheda

AND SA.DataFineScheda)

OR

(CURRENT\_DATE > SA.DataInizioScheda

AND SA.DataFineScheda IS NULL)

);

IF inizio\_scheda IS NOT NULL THEN

SELECT C.VisiteMedico INTO totale\_visite

FROM Cliente C

WHERE C.CodFiscale = cliente;

UPDATE SchedaAlimentazione SA

SET SA.FrequenzaVisita = ROUND(

DATEDIFF(CURRENT\_DATE, inizio\_scheda)

/totale\_visite)

WHERE SA.Cliente = cliente

AND(

(CURRENT\_DATE BETWEEN SA.DataInizioScheda

AND SA.DataFineScheda)

OR

(CURRENT\_DATE > SA.DataInizioScheda

AND SA.DataFineScheda IS NULL)

);

ELSE

SELECT "Impossibile aggiornare FrequenzaVisite.

Non esiste una scheda per il paziente.";

END IF;

END$$

DELIMITER ;

### 5.3.7- Operazione 7

#### 5.3.7.1- Senza ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS ClientiPresenti$$

CREATE PROCEDURE ClientiPresenti (IN centro INTEGER)

BEGIN

SELECT COUNT(\*) AS ClientiPresenti

FROM Accesso A

WHERE A.Centro = centro

AND A.TimestampUscita IS NULL;

END$$

DELIMITER ;

#### 5.3.7.2- Con ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS ClientiPresenti\_Rid$$

CREATE PROCEDURE ClientiPresenti\_Rid (IN centro INTEGER)

BEGIN

SELECT C.ClientiPresenti

FROM Centro C

WHERE C.CodCentro = centro;

END$$

DELIMITER ;

### 5.3.8- Operazione 8

#### 5.3.8.1- Senza ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS AggiungiAccesso$$

CREATE PROCEDURE AggiungiAccesso (IN timestamp\_ingresso DATETIME,

IN timestamp\_uscita DATETIME,

IN centro INTEGER,

IN cliente VARCHAR(255),

IN armadietto INTEGER)

BEGIN

DECLARE clienti\_presenti INTEGER DEFAULT 0;

DECLARE max\_clienti INTEGER DEFAULT 0;

SELECT COUNT(\*) INTO clienti\_presenti

FROM Accesso A

WHERE A.Centro = centro

AND A.TimestampUscita IS NULL;

SELECT C.MaxClienti INTO max\_clienti

FROM Centro C

WHERE C.CodCentro = centro;

IF max\_clienti > clienti\_presenti THEN

INSERT INTO `Accesso` (`TimestampIngresso`,

`TimestampUscita`,

`Centro`,

`Cliente`,

`Armadietto`)

VALUES (timestamp\_ingresso,

timestamp\_uscita,

centro,

cliente,

armadietto);

END IF;

END$$

DELIMITER ;

#### 5.3.8.2- Con ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS AggiungiAccesso\_Rid$$

CREATE PROCEDURE

AggiungiAccesso\_Rid (IN timestamp\_ingresso DATETIME,

IN timestamp\_uscita DATETIME,

IN centro INTEGER,

IN cliente VARCHAR(255),

IN armadietto INTEGER)

BEGIN

DECLARE clienti\_presenti INTEGER DEFAULT 0;

DECLARE max\_clienti INTEGER DEFAULT 0;

SELECT C.ClientiPresenti INTO clienti\_presenti

FROM Centro C

WHERE C.CodCentro = centro;

SELECT MaxClienti INTO max\_clienti

FROM Centro C

WHERE C.CodCentro = centro;

IF max\_clienti > clienti\_presenti THEN

INSERT INTO `Accesso` (`TimestampIngresso`,

`TimestampUscita`,

`Centro`,

`Cliente`,

`Armadietto`)

VALUES (timestamp\_ingresso,

timestamp\_uscita,

centro,

cliente,

armadietto);

END IF;

END$$

DELIMITER ;

### 5.3.9- Operazione 9

#### 5.3.9.1- Senza ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS PrestazioneMedia$$

CREATE PROCEDURE PrestazioneMedia (IN esercizio INTEGER)

BEGIN

SELECT AVG(D.DurataC) AS DurataMedia,

AVG(D.ValutazionePrestazione) AS GiudizioMedio

FROM (

SELECT PE.\*,

TIME\_TO\_SEC(TIMEDIFF(PE.TimestampFine,

PE.TimestampInizio)

) AS DurataC

FROM PrestazioneEsercizio PE

WHERE PE.Esercizio = esercizio

) AS D;

END$$

DELIMITER ;

#### 5.3.9.1- Con ridondanza:

DELIMITER $$

DROP PROCEDURE IF EXISTS PrestazioneMedia\_Rid$$

CREATE PROCEDURE PrestazioneMedia\_Rid (IN esercizio INTEGER)

BEGIN

SELECT (E.TotDurata / E.TotEsecuzioni) AS DurataMedia,

(E.TotVoto / E.TotEsecuzioni) AS GiudizioMedio

FROM Esercizio E

WHERE E.CodEsercizio = esercizio;

END$$

DELIMITER ;

## 5.4- Implementazione delle operazioni di aggiornamento delle ridondanze

### 5.4.1- Ridondanza 1

DELIMITER $$

-- Nuovo accesso di un cliente

DROP TRIGGER IF EXISTS Aggiorna\_Accesso\_Entrata$$

CREATE TRIGGER Aggiorna\_Accesso\_Entrata

AFTER INSERT ON Accesso

FOR EACH ROW

BEGIN

DECLARE presenti INTEGER DEFAULT 0;

DECLARE massimi INTEGER DEFAULT 0;

SELECT C.MaxClienti INTO massimi

FROM Centro C

WHERE C.CodCentro = NEW.Centro;

SELECT C.ClientiPresenti INTO presenti

FROM Centro C

WHERE C.CodCentro = NEW.Centro;

IF presenti = massimi THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT='Impossibile accedere a questo centro:

numero massimo di clienti raggiunto';

END IF;

UPDATE Centro

SET ClientiPresenti=ClientiPresenti + 1

WHERE CodCentro=NEW.Centro;

END$$

-- Cliente che esce

CREATE TRIGGER Aggiorna\_Accesso\_Uscita

AFTER UPDATE ON Accesso

FOR EACH ROW

BEGIN

IF(NEW.TimestampUscita <> OLD.TimestampUscita) THEN

UPDATE Centro

SET ClientiPresenti=ClientiPresenti - 1

WHERE CodCentro=NEW.Centro;

END IF;

END$$

DELIMITER ;

### 5.4.2- Ridondanza 2

DELIMITER $$

DROP TRIGGER IF EXISTS Aggiorna\_VisiteMedico$$

CREATE TRIGGER Aggiorna\_VisiteMedico

AFTER INSERT ON Visita

FOR EACH ROW

BEGIN

DECLARE Medico VARCHAR(255) DEFAULT 0;

DECLARE Counter INTEGER DEFAULT 0;

SELECT COUNT(\*) INTO Counter

FROM schedaalimentazione SA

WHERE SA.Cliente = NEW.Cliente

AND SA.Medico = NEW.Medico

AND SA.DataInizioScheda < NEW.DataVisita

AND SA.DataFineScheda IS NULL;

SELECT C.Medico INTO Medico

FROM Cliente C

WHERE C.CodFiscale = NEW.Cliente;

IF Counter <> 0 THEN

IF Medico = NEW.Medico THEN

UPDATE Cliente C

SET C.VisiteMedico = C.VisiteMedico + 1

WHERE C.CodFiscale = NEW.Cliente;

END IF;

ELSE

UPDATE Cliente C

SET C.VisiteMedico = 0

WHERE C.CodFiscale = NEW.Cliente;

END IF;

END$$

DELIMITER ;

### 5.4.3- Ridondanza 3

DELIMITER $$

DROP TRIGGER IF EXISTS Aggiorna\_TotEsecuzioni$$

CREATE TRIGGER Aggiorna\_TotEsecuzioni

AFTER UPDATE ON PrestazioneEsercizio

FOR EACH ROW

BEGIN

IF (NEW.TimestampFine IS NOT NULL

AND OLD.TimestampFine IS NULL) THEN

UPDATE Esercizio

SET TotEsecuzioni=TotEsecuzioni + 1

WHERE CodEsercizio=NEW.Esercizio;

UPDATE Esercizio

SET TotVoto=TotVoto + NEW.ValutazionePrestazione

WHERE CodEsercizio=NEW.Esercizio;

UPDATE Esercizio

SET TotDurata=TotDurata +

TIME\_TO\_SEC(TIMEDIFF(NEW.TimestampFine,

NEW.TimestampInizio)

)

WHERE CodEsercizio=NEW.Esercizio;

END IF;

END$$

DELIMITER ;

## 5.5- Implementazione delle funzionalità lato server

### 5.5.1- Reporting

#### 5.5.1.1- Corsi poco frequentati

Per conoscere quali corsi di un centro sono poco frequentati è stata implementata una stored procedure che identifica i corsi attualmente attivati col minor rapporto iscritti/massimo iscrizioni. In questo modo è possibile applicare politiche di promozione relative a questi corsi abbassandone i prezzi oppure rendendoli più disponibili nei contratti standard.

Il codice dell’operazione è il seguente:

DELIMITER $$

DROP PROCEDURE IF EXISTS CorsiPocoFrequentati$$

CREATE PROCEDURE CorsiPocoFrequentati (IN centro INTEGER)

BEGIN

SELECT \*

FROM(

SELECT C.CodCorso,

C.Disciplina,

C.Livello,

(COUNT(\*)/C.MaxPartecipanti) AS

RapportoIscrittiMaxIscritti

FROM iscrizione I

INNER JOIN corso C ON I.Corso = C.CodCorso

INNER JOIN PianificazioneCorso PC

ON PC.Corso=C.CodCorso

INNER JOIN LuogoAllenamento L

ON L.CodLuogo=PC.Luogo

WHERE L.Centro = centro

AND C.DataFineCorso > CURRENT\_DATE

GROUP BY C.CodCorso, C.Disciplina, C.Livello

) AS D

ORDER BY D.RapportoIscrittiMaxIscritti;

END$$

DELIMITER ;

#### 5.5.1.2- Fasce orarie poco frequentate

Abbiamo deciso, per ogni fascia oraria e per ogni giorno della settimana, di individuare quanti clienti accedono ad un centro. Per fare ciò, è necessario controllare la tabella Accesso relativa ad un qualsiasi centro (tramite il codice) e, cliente per cliente, analizzare in quale fasce orarie ricade la sua permanenza al centro. Il codice dell’operazione è il seguente:

DELIMITER $$

DROP PROCEDURE IF EXISTS FasceOrarie$$

CREATE PROCEDURE FasceOrarie (IN centro INTEGER)

BEGIN

DROP TABLE IF EXISTS FasceOrarie;

CREATE TEMPORARY TABLE FasceOrarie(

OraInizio INTEGER NOT NULL,

OraFine INTEGER NOT NULL,

PRIMARY KEY (OraInizio)

) ENGINE = InnoDB DEFAULT CHARSET = latin1;

INSERT INTO FasceOrarie

VALUES(8,9),(9,10),(10,11),(11,12),(12,13),(13,14),(14,15),

(15,16),(16,17),(17,18),(18,19),(19,20),(20,21),(21,22);

SELECT FO.OraInizio,

FO.OraFine,

DAYOFWEEK(A.TimestampIngresso) AS GiornoSettimana,

COUNT(\*) AS Accessi

FROM accesso A

CROSS JOIN FasceOrarie FO

WHERE A.Centro = centro

AND(

(HOUR(TimestampIngresso)=FO.OraInizio)

OR

(

(HOUR(TimestampIngresso)<FO.OraInizio)

AND

TimestampUscita IS NOT NULL

AND

HOUR(TimestampUscita)>FO.OraFine

)

OR

(

(HOUR(TimestampIngresso)<FO.OraInizio)

AND

TimestampUscita IS NULL

)

)

GROUP BY FO.OraInizio, FO.OraFine, DAYOFWEEK(A.TimestampIngresso);

END$$

DELIMITER ;

#### 5.5.1.3- Corsi e luoghi d’allenamento da includere in contratti standard

Per capire quali corsi, sale e piscine sono da considerare per un eventuale inserimento in un contratto standard di un centro, abbiamo pensato di controllare quanti clienti firmatari di contratti personalizzati hanno richiesto l’accesso a quel corso (oppure a quel luogo d’allenamento), confrontandoli con i clienti che vi hanno accesso tramite contratti standard. Se questi ultimi sono inferiori, ad esempio, al 10% del totale dei clienti che hanno accesso al centro dove si tiene il corso, e se primi sono in un numero considerevole, forse vale la pena di includere quel corso anche in altri contratti standard.

Il codice dell’operazione è il seguente (relativo al centro con codice XXXX):

DELIMITER $$

DROP PROCEDURE IF EXISTS CorsiDaIncludere$$

CREATE PROCEDURE CorsiDaIncludere (IN centro INTEGER)

BEGIN

SELECT DISTINCT C.\*

FROM Corso C

INNER JOIN Pianificazionecorso PC

ON C.CodCorso = PC.Corso

INNER JOIN LuogoAllenamento LA ON PC.Luogo = LA.CodLuogo

INNER JOIN OffertaCorso AC ON AC.Corso = C.CodCorso

INNER JOIN Offerta O ON AC.Offerta = O.CodOfferta

INNER JOIN InclusioneOfferta IO

ON O.CodOfferta = IO.Offerta

WHERE O.TipoOfferta = 'personalizzata'

AND LA.Centro = centro

AND 0.1 \*(

SELECT COUNT(\*)

FROM Contratto CO

INNER JOIN InclusioneOfferta IO1

ON IO1.Contratto=CO.CodContratto

INNER JOIN Offerta O1

ON O1.CodOfferta = IO1.Offerta

WHERE O1.Centro = centro

)>(

SELECT COUNT(\*)

FROM Corso C1

INNER JOIN OffertaCorso AC1

ON AC1.Corso = C1.CodCorso

INNER JOIN Offerta O2

ON AC1.Offerta=O2.CodOfferta

INNER JOIN InclusioneOfferta IO2

ON O2.CodOfferta=IO2.Offerta

WHERE O2.Centro = 7

AND O2.TipoOfferta <> 'personalizzata'

AND C1.CodCorso = C.CodCorso

);

END$$

DELIMITER ;

Per quanto riguarda i luoghi di allenamento invece:

DELIMITER $$

DROP PROCEDURE IF EXISTS LuoghiDaIncludere$$

CREATE PROCEDURE LuoghiDaIncludere (IN centro INTEGER)

BEGIN

SELECT DISTINCT L.\*

FROM Luogoallenamento L

INNER JOIN Offertaluogo OL ON OL.Luogo = L.CodLuogo

INNER JOIN Offerta O ON OL.Offerta = O.CodOfferta

INNER JOIN InclusioneOfferta IOF ON O.CodOfferta = IOF.Offerta

WHERE O.TipoOfferta = 'personalizzata'

AND L.Centro = centro

AND 0.1 \*

(

SELECT COUNT(\*)

FROM Contratto CO

INNER JOIN InclusioneOfferta IO1

ON IO1.Contratto=CO.CodContratto

INNER JOIN Offerta O1

ON O1.CodOfferta = IO1.Offerta

WHERE O1.Centro = centro

)>(

SELECT COUNT(\*)

FROM luogoallenamento L1

INNER JOIN OffertaLuogo OL1

ON OL1.Luogo = L1.CodLuogo

INNER JOIN Offerta O2

ON OL1.Offerta=O2.CodOfferta

INNER JOIN InclusioneOfferta IOF2

ON O2.CodOfferta=IOF2.Offerta

WHERE O2.Centro = centro

AND O2.TipoOfferta<>'personalizzata'

AND L1.CodLuogo = L.CodLuogo

);

END$$

DELIMITER ;

#### 5.5.1.4- Tempi medi di attesa delle sale

Per riuscire a calcolare i tempi medi di attesa delle sale in modo da trovare quelle che creano più problemi a livello logistico, ci è bastato controllare tra le prestazioni degli esercizi la durata tra la fine di un esercizio e l’inizio di un altro svolto nella stessa sala dallo stesso cliente, tale che non esista un altro esercizio di quel cliente compreso tra i due. Il codice di implementazione è il seguente:

DELIMITER $$

DROP PROCEDURE IF EXISTS AttesaSale$$

CREATE PROCEDURE AttesaSale (IN centro INTEGER)

BEGIN

SELECT D.Sala, AVG(D.Attesa) as AttesaMedia

FROM(

SELECT PE1.CodPrestazione as Prestazione,

S1.Luogo as Sala,

TIME\_TO\_SEC(TIMEDIFF(PE2.TimestampInizio,

PE1.TimestampFine))AS Attesa

FROM PrestazioneEsercizio PE1

INNER JOIN PrestazioneEsercizio PE2 USING (Cliente)

INNER JOIN Esercizio E1

ON PE1.Esercizio=E1.CodEsercizio

INNER JOIN ConfigurazioneEsercizio C1

ON C1.Esercizio=E1.CodEsercizio

INNER JOIN Attrezzatura A1

ON C1.Attrezzatura=A1.CodAttrezzatura

INNER JOIN Sala S1 ON A1.Sala = S1.Luogo

INNER JOIN LuogoAllenamento L1

ON S1.Luogo=L1.CodLuogo

INNER JOIN Esercizio E2

ON PE2.Esercizio = E2.CodEsercizio

INNER JOIN ConfigurazioneEsercizio C2

ON C2.Esercizio=E2.CodEsercizio

INNER JOIN Attrezzatura A2

ON C2.Attrezzatura=A2.CodAttrezzatura

INNER JOIN Sala S2 ON A2.Sala = S2.Luogo

INNER JOIN LuogoAllenamento L2

ON S2.Luogo=L2.CodLuogo

WHERE S2.Luogo=S1.Luogo

AND L1.Centro = centro

AND PE2.TimestampInizio > PE1.TimestampFine

AND PE2.Cliente = PE1.Cliente

AND NOT EXISTS(

SELECT \*

FROM PrestazioneEsercizio PE3

WHERE PE3.Cliente = PE1.Cliente

AND PE3.TimestampInizio >

PE1.TimestampInizio

AND PE3.TimestampInizio <

PE2.TimestampInizio

)

) AS D

GROUP BY D.Sala;

END$$

DELIMITER ;

### 5.5.2- Performance sportiva

Per costruire una metrica grazie alla quale poter valutare le prestazioni dei clienti, ci siamo concentrati inizialmente sulle due tipologie di esercizi che è possibile eseguire, ossia gli esercizi anaerobici e quelli aerobici. A seconda del tipo di esercizio svolto, infatti, le variabili che influiscono sulla prestazione possono variare o assumere significati diversi; ad esempio una maggior durata totale dell’esecuzione sarà un elemento identificatore di una miglior prestazione in caso di esercizi aerobici, mentre l’esatto opposto in caso di esercizi anaerobici.

Per valutare un esercizio aerobico, abbiamo analizzato dunque quali variabili giocano a favore e quali a sfavore della prestazione.

Come precedentemente affermato, la durata dell’esercizio è un parametro che, nel caso dell’esercizio aerobico, più si avvicina al valore consigliato nell’esercizio, più significa che il cliente esecutore dell’esercizio si è trovato a suo agio. Viceversa, se un cliente dovesse terminare l’esercizio prima della durata indicata, significa che non è abbastanza allenato da poter eseguirlo, per cui la valutazione della prestazione non potrà essere alta.

Inoltre, per far sì che si ottengano giudizi negativi se l’esecuzione termina prima della metà della durata standard dell’esercizio, abbiamo usato una scala parabolica al posto di una scala lineare, in particolare la semplice y=x2. Se la durata d’esecuzione supera quella consigliata nella descrizione dell’esercizio, il voto non sale ma resta costantemente 10.

Ecco dunque la formula per calcolare il giudizio in decimi di un esercizio aerobico:

Per un esercizio anaerobico, invece si aggiungono i fattori tempo di riposo medio effettivo tra le ripetizioni (più alto è più significa che il cliente si è stancato di più e quindi il giudizio cala) e il fattore numero di ripetizioni effettive (più è basso, più la valutazione, ovviamente, scende). Questi parametri sono messi a confronto con il tempo di recupero e il numero di ripetizioni indicati dall’esercizio.

In più per gli esercizi anaerobici, poiché non è prevista una durata media indicata sulla scheda dell’esercizio, si considera la media delle tempistiche effettuate da tutti gli altri clienti che l’hanno svolto. Questa volta, però, più la durata è alta, più significa che il cliente ha faticato per terminare l’esercizio, quindi il giudizio, inevitabilmente, scende.

Anche per questa metrica abbiamo utilizzato una scala parabolica.

La formula per il calcolo del giudizio in decimi di un esercizio aerobico è la seguente:

Il codice dell’implementazione è il seguente:

DELIMITER $$

DROP TRIGGER IF EXISTS ValutazionePrestazione$$

CREATE TRIGGER ValutazionePrestazione

BEFORE UPDATE ON PrestazioneEsercizio

FOR EACH ROW

BEGIN

DECLARE giudizio DOUBLE DEFAULT 0;

DECLARE counter INTEGER DEFAULT 0;

IF(OLD.TimestampFine IS NULL

AND NEW.TimestampFine IS NOT NULL) THEN

SELECT IFNULL(COUNT(\*),0) INTO counter

FROM Aerobico A

WHERE A.CodEsercizio = NEW.Esercizio;

IF counter = 1 THEN

SELECT POW(TIMEDIFF(PE.TimestampFine,

PE.TimestampInizio)/AE.Durata

,2)\*10 INTO giudizio

FROM prestazioneesercizio PE

INNER JOIN Esercizio E

ON PE.Esercizio = E.CodEsercizio

INNER JOIN Aerobico AE

ON AE.CodEsercizio = E.CodEsercizio

WHERE PE.Esercizio = NEW.Esercizio;

ELSE

SELECT POW(

((E.TotDurata/E.TotEsecuzioni)/

TIME\_TO\_SEC(TIMEDIFF(NEW.TimestampFine,

NEW.TimestampInizio)))

\*(AN.TempoRecupero/PE.TempoRecupero)

\*(AN.NumeroRipetizioni/

PE.NumeroRipetizioni)

,2)\*10 INTO giudizio

FROM prestazioneesercizio PE

INNER JOIN Esercizio E

ON PE.Esercizio = E.CodEsercizio

INNER JOIN Anaerobico AN

ON AN.CodEsercizio = E.CodEsercizio

WHERE PE.Esercizio = NEW.Esercizio;

END IF;

SELECT IF(giudizio > 10, 10, giudizio) INTO giudizio;

SET NEW.ValutazionePrestazione = giudizio;

END IF;

END$$

DELIMITER ;

### 5.5.3- Rotazione del magazzino

#### 5.5.3.1- Vendite degli integratori e presenze nel magazzino

Per controllare quanti integratori sono presenti in un magazzino di un centro, banalmente bisogna contare quanti integratori sono stati ordinati e quanti clienti hanno acquistato integratori da quel centro (considerando che un cliente può acquistare integratori soltanto dal centro in cui ha stipulato il contratto).

Si produrrà dunque un report con indicati, per ogni integratore, ordini, acquisti e presenze in magazzino. Il codice dell’operazione è il seguente:

DELIMITER $$

DROP PROCEDURE IF EXISTS RimanenzeMagazzino$$

CREATE PROCEDURE RimanenzeMagazzino (IN centro INTEGER)

BEGIN

DROP TEMPORARY TABLE IF EXISTS OrdiniIntegratori;

REATE TEMPORARY TABLE OrdiniIntegratori(

Integratore INTEGER NOT NULL,

TotaleOrdini INTEGER NOT NULL

) ENGINE = InnoDB DEFAULT CHARSET = latin1;

DROP TEMPORARY TABLE IF EXISTS AcquistiIntegratori;

CREATE TEMPORARY TABLE AcquistiIntegratori(

Integratore INTEGER NOT NULL,

TotaleAcquisti INTEGER NOT NULL

) ENGINE = InnoDB DEFAULT CHARSET = latin1;

INSERT INTO OrdiniIntegratori

SELECT OI.Integratore, SUM(OI.Quantita)

FROM Ordine O

INNER JOIN OrdineIntegratore OI ON O.CodOrdine = OI.Ordine

WHERE O.Centro = centro

GROUP BY OI.Integratore;

INSERT INTO AcquistiIntegratori

SELECT A.Integratore, SUM(A.Quantita) AS TotaleAcquisti

FROM Cliente C

INNER JOIN Contratto CO on C.CodFiscale = CO.Cliente

INNER JOIN Acquisto A on C.CodFiscale = A.Cliente

WHERE CO.Centro = centro

GROUP BY A.Integratore;

SELECT OI.Integratore,

OI.TotaleOrdini,

IFNULL(AI.TotaleAcquisti,0) AS TotaleAcquisti,

(OI.TotaleOrdini-IFNULL(AI.TotaleAcquisti,0))AS Rimanenze

FROM OrdiniIntegratori OI

LEFT OUTER JOIN AcquistiIntegratori AI USING (Integratore);

END$$

DELIMITER ;

#### 5.5.3.2- Promozioni per integratori invenduti

Per calcolare le promozioni da applicare agli integratori che restano a lungo invenduti nel magazzino, abbiamo considerato diverse variabili e valutato alcune considerazioni per cercare di fornire promozioni il più possibile convenienti sia per i consumatori ma soprattutto per il centro di vendita.

Abbiamo innanzitutto stabilito uno sconto massimo da applicare a un integratore, oltre il quale il centro va in perdita. Infatti, lo sconto massimo ScontoMAX deve essere quello che, se applicato ad un prodotto, imposta il prezzo di vendita al dettaglio pari al prezzo all’ingrosso pagato dal centro al fornitore.

Impostando l’equazione:

Ricavo ScontoMAX:

Di questo sconto, viene applicata una percentuale che varia da prodotto a prodotto in base al rapporto acquisti/ordinazioni. Un prodotto che è stato ordinato in notevoli quantità ma le cui vendite sono estremamente scarse, subirà uno sconto molto vicino al 100% dello sconto massimo applicabile. Viceversa, un prodotto le cui vendite raggiungono quasi le sue ordinazioni da parte del centro, non subirà sconti notevoli.

Abbiamo dunque deciso di applicare la seguente formula:

per calcolare la percentuale da applicare allo sconto massimo di un prodotto.

Il risultato finale sarà la formula:

che indica la percentuale di sconto da applicare al prodotto per ottenere la promozione migliore.

Ad esempio, se un prodotto X ha un prezzo all’ingrosso di € 1 e uno al dettaglio di € 5, e ne sono stati ordinati 70 pezzi di cui soltanto 15 venduti, lo sconto da applicare al prodotto sarà del:

Notare che se il prodotto non fosse stato mai acquistato, si sarebbe applicato lo sconto massimo pari al:

con questo sconto, infatti, il prodotto al dettaglio ha un prezzo finale di: € 5 - € 5 \* 0,8 = € 1, ovvero il prezzo all’ingrosso. Questo è il codice dell’implementazione:

DELIMITER $$

DROP PROCEDURE IF EXISTS PromozioniIntegratoriInvenduti$$

CREATE PROCEDURE PromozioniIntegratoriInvenduti(IN centro INTEGER)

BEGIN

DROP TEMPORARY TABLE IF EXISTS OrdiniIntegratori;

CREATE TEMPORARY TABLE OrdiniIntegratori(

Integratore INTEGER NOT NULL,

TotaleOrdini INTEGER NOT NULL

) ENGINE = InnoDB DEFAULT CHARSET = latin1;

DROP TEMPORARY TABLE IF EXISTS AcquistiIntegratori;

CREATE TEMPORARY TABLE AcquistiIntegratori(

Integratore INTEGER NOT NULL,

TotaleAcquisti INTEGER NOT NULL

) ENGINE = InnoDB DEFAULT CHARSET = latin1;

INSERT INTO OrdiniIntegratori

SELECT OI.Integratore, SUM(OI.Quantita)

FROM Ordine O

INNER JOIN OrdineIntegratore OI on O.CodOrdine=OI.Ordine

WHERE O.Centro = centro

GROUP BY OI.Integratore;

INSERT INTO AcquistiIntegratori

SELECT A.Integratore, SUM(A.Quantita) AS TotaleAcquisti

FROM Cliente C

INNER JOIN Contratto CO on C.CodFiscale = CO.Cliente

INNER JOIN Acquisto A on C.CodFiscale = A.Cliente

WHERE CO.Centro = centro

GROUP BY A.Integratore;

SELECT OI.Integratore,

OI.TotaleOrdini,

IFNULL(AI.TotaleAcquisti,0) AS TotaleAcquisti,

100\*

(1-(IFNULL(AI.TotaleAcquisti,0)/OI.TotaleOrdini))\*

(1-(I.PrezzoIngrosso/I.PrezzoDettaglio)) AS Promozione

FROM OrdiniIntegratori OI

LEFT OUTER JOIN AcquistiIntegratori AI USING(Integratore)

INNER JOIN Integratore I

ON OI.Integratore = I.CodIntegratore;

END$$

DELIMITER ;

#### 5.5.3.3- Promozioni per integratori in via di scadenza

Esattamente come nel paragrafo precedente, si impone sempre lo sconto massimo applicabile in funzione dei prezzi all’ingrosso e al dettaglio, ma cambia il criterio in base al quale si applica la percentuale: più un prodotto è vicino alla data di scadenza rispetto alla data dell’ordine, più questo va venduto velocemente.

Non sarebbe sufficiente, infatti, controllare solamente se un prodotto è vicino alla data di scadenza, ma va messo in relazione con il tempo totale di rimanenza nel magazzino dal momento dell’ordine. Infatti, un prodotto potrebbe avere una data di scadenza molto vicina alla data dell’ordine, per cui per quel prodotto sarebbe normale avere rimanenze di magazzino a pochi giorni dalla data di scadenza. Un prodotto ordinato anni fa, invece, se resta ancora invenduto a poche settimane prima della data di scadenza, è da considerarsi come prodotto da vendere immediatamente per evitare di doverlo gettare.

Il calcolo della percentuale da applicare allo sconto massimo sarà dunque:

La formula per lo sconto finale sarà dunque:

Ad esempio, un prodotto Y che costà € 1 all’ingrosso e € 5 al dettaglio, che è stato ordinato 400 giorni fa e la data di scadenza è fra 7 giorni, subirà una promozione del:

percentuale molto prossima allo sconto massimo applicabile dell’80% (vedi paragrafo precedente). Del codice precedente va sostituita soltanto la select clause: questo è il codice dell’implementazione:

SELECT I.CodIntegratore,

I.DataScadenza,

100\*

(1-(DATEDIFF(I.DataScadenza, CURRENT\_DATE)/

DATEDIFF(I.DataScadenza, O.DataEvasione)))\*

(1-(I.PrezzoIngrosso/I.PrezzoDettaglio)) AS Promozione

FROM Integratore I

INNER JOIN OrdineIntegratore OI

ON OI.Integratore=I.CodIntegratore

INNER JOIN Ordine O ON OI.Ordine=O.CodOrdine

WHERE O.Centro = centro;

## 5.6- Implementazione vincoli di integrità generici

### 5.6.1- Vincolo di integrità generico 1

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_Turnazione $$

CREATE TRIGGER BR\_Turnazione

BEFORE INSERT ON Turnazione

FOR EACH ROW

BEGIN

DECLARE counter INTEGER DEFAULT 0;

SELECT COUNT(\*) INTO counter

FROM Turnazione T

WHERE T.Dipendente = NEW.Dipendente

AND (T.OraFineTurno > NEW.OraInizioTurno)

AND (T.OraInizioTurno < NEW.OraFineTurno);

IF counter <> 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Piano di turnazione non valido:

collisione con altri turni.';

END IF;

END$$

DELIMITER ;

### 5.6.2- Vincolo di integrità generico 2

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_OrarioOttoOre $$

CREATE TRIGGER BR\_OrarioOttoOre

BEFORE INSERT ON Turnazione

FOR EACH ROW

BEGIN

DECLARE somma\_secondi INTEGER DEFAULT 0;

DECLARE somma\_secondi\_attuale INTEGER DEFAULT 0;

DECLARE inizio TIME DEFAULT '';

DECLARE fine TIME DEFAULT '';

DECLARE finito INTEGER DEFAULT 0;

DECLARE curTurni CURSOR FOR

SELECT T.OraInizioTurno, T.OraFineTurno

FROM Turnazione T

WHERE T.Dipendente = NEW.Dipendente

AND T.Giorno = NEW.Giorno;

DECLARE CONTINUE HANDLER FOR

NOT FOUND SET finito = 1;

OPEN curTurni;

preleva:LOOP

FETCH curTurni INTO inizio, fine;

IF finito = 1 THEN

LEAVE preleva;

END IF;

SET somma\_secondi = somma\_secondi +

TIME\_TO\_SEC(TIMEDIFF(fine, inizio));

END LOOP;

SET somma\_secondi\_attuale = somma\_secondi +

TIME\_TO\_SEC(TIMEDIFF(NEW.OraFineTurno,

NEW.OraInizioTurno));

IF (somma\_secondi\_attuale/3600) > 8 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Piano di turnazione non valido:

orario complessivo di lavoro oltre le 8 ore.';

END IF;

END$$

DELIMITER ;

### 5.6.3- Vincolo di integrità generico 3

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_DataCorso $$

CREATE TRIGGER BR\_DataCorso

BEFORE INSERT ON Corso

FOR EACH ROW

BEGIN

IF(NEW.DataInizioCorso > NEW.DataFineCorso) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Impossibile inserire questo corso:

la data di fine è precedente a quella

di inizio.';

END IF;

END$$

DELIMITER ;

### 5.6.4- Vincolo di integrità generico 4

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_Contratto $$

CREATE TRIGGER BR\_Contratto

BEFORE INSERT ON InclusioneOfferta

FOR EACH ROW

BEGIN

DECLARE counter INTEGER DEFAULT 0;

SELECT COUNT(DISTINCT D.Centro) INTO counter

FROM(

SELECT O.Centro

FROM InclusioneOfferta IOF

INNER JOIN Offerta O ON IOF.Offerta = O.CodOfferta

WHERE IOF.Contratto = NEW.Contratto

UNION

SELECT O1.Centro

FROM Offerta O1

WHERE O1.CodOfferta = NEW.Offerta

) as D;

IF counter > 3 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Impossibile aggiungere questa offerta

a questo contratto: consentito accesso

a massimo 3 sedi.';

END IF;

END$$

DELIMITER ;

### 5.6.5- Vincolo di integrità generico 5

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_DataAllenamento $$

CREATE TRIGGER BR\_DataAllenamento

BEFORE INSERT ON SchedaAllenamento

FOR EACH ROW

BEGIN

IF(NEW.DataInizioScheda > NEW.DataFineScheda) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Impossibile inserire questa scheda:

la data di fine è precedente a quella

di inizio.';

END IF;

END$$

DELIMITER ;

### 5.6.6- Vincolo di integrità generico 6

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_SchedaAlimentazione $$

CREATE TRIGGER BR\_SchedaAlimentazione

BEFORE INSERT ON SchedaAlimentazione

FOR EACH ROW

BEGIN

DECLARE counter INTEGER DEFAULT 0;

SELECT COUNT(\*) INTO counter

FROM SchedaAlimentazione SA

WHERE SA.Cliente = NEW.Cliente

AND (SA.DataFineScheda > NEW.DataInizioScheda)

AND (SA.DataInizioScheda < NEW.DataFineScheda);

IF counter <> 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Scheda di alimentazione non valida:

collisione con altre schede.';

END IF;

END$$

DELIMITER ;

### 5.6.7- Vincolo di integrità generico 7

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_DataAlimentazioneData $$

CREATE TRIGGER BR\_DataAlimentazioneData

BEFORE INSERT ON SchedaAlimentazione

FOR EACH ROW

BEGIN

IF(NEW.DataInizioScheda > NEW.DataFineScheda) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Impossibile inserire questa scheda:

la data di fine è precedente a

quella di inizio.';

END IF;

END$$

DELIMITER ;

### 5.6.8- Vincolo di integrità generico 8

DELIMITER $$

DROP TRIGGER IF EXISTS BR\_DataSfida $$

CREATE TRIGGER BR\_DataSfida

BEFORE INSERT ON Sfida

FOR EACH ROW

BEGIN

IF(NEW.DataInizio > NEW.DataFine) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Impossibile inserire questa sfida:

la data di fine è precedente a

quella di inizio.';

END IF;

END$$

DELIMITER ;