**Università Degli Studi Di Napoli**

**“Parthenope”**



**Progetto di Basi di Dati**

**“palestre Gym power”**



*Gennaro Traino 0124001105*

*Pierluigi Scotto 0124001277*

*Anno Accademico 2018/2019*

Indice

[**Progettazione** 3](#_Toc531518985)

[**Sintesi dei requisiti** 3](#_Toc531518986)

[**Glossario** 5](#_Toc531518987)

[**Sintesi EE/R** 6](#_Toc531518988)

[**Modellazione Concettuale** 6](#_Toc531518989)

[**Gestione Degli Iscritti** 6](#_Toc531518990)

[**Gestione Degli Istruttori** 8](#_Toc531518991)

[**Gestione Delle Schede** 9](#_Toc531518992)

[**Diagramma EE/R** 10](#_Toc531518993)

[**Modellazione Logica** 11](#_Toc531518994)

[**Traduzione: Gestione Degli Iscritti** 11](#_Toc531518995)

[**Traduzione: Gestione Degli Istruttori** 14](#_Toc531518996)

[**Traduzione: Gestione Delle Schede** 16](#_Toc531518997)

[**Diagramma Relazionale** 18](#_Toc531518998)

[**Definizione Utenti** 19](#_Toc531518999)

[**Permessi Utenti** 19](#_Toc531519000)

[**Tabella Dei Volumi** 19](#_Toc531519001)

[**Diagrammi Dei Casi D’Uso** 20](#_Toc531519002)

[**Tabelle Operazioni Degli Utenti** 20](#_Toc531519003)

[**Vincoli** 22](#_Toc531519004)

[**Vincoli Statici** 22](#_Toc531519005)

[**Vincoli Dinamici** 23](#_Toc531519006)

[**Normalizzazione** 24](#_Toc531519007)

[**Implementazione** 24](#_Toc531519008)

[**DCL: Creazione & Permessi Degli Utenti** 24](#_Toc531519009)

[**DDL: Creazioni Delle Tabelle** 25](#_Toc531519010)

[**DML: Popolamento** 28](#_Toc531519011)

[**Trigger** 34](#_Toc531519012)

[**Trigger Scheda Esercizi** 34](#_Toc531519013)

[**Trigger Auto-Id Scheda** 34](#_Toc531519014)

[**Trigger Controllo Turni** 35](#_Toc531519015)

[**Trigger Auto-Badge Corso** 35](#_Toc531519016)

[**Trigger Inserimento Corso** 36](#_Toc531519017)

[**Trigger Abbonamenti** 37](#_Toc531519018)

[**Trigger Istruttori Maggiorenni** 38](#_Toc531519019)

[**Trigger Iscritti Maggiorenni** 38](#_Toc531519020)

[**Trigger Eventi Massimi** 38](#_Toc531519021)

[**Procedure** 39](#_Toc531519022)

[**Creazione Scheda** 39](#_Toc531519023)

[**Sottoscrizione Abbonamento** 40](#_Toc531519024)

[**Stila Informazioni Iscritto** 41](#_Toc531519025)

[**Stila Lista Turni** 43](#_Toc531519026)

[**Assegna Scheda** 44](#_Toc531519027)

[**Assegna Corso** 45](#_Toc531519028)

[**Assumi Istruttore** 46](#_Toc531519029)

[**Licenzia Istruttore** 47](#_Toc531519030)

[**Crea Evento** 47](#_Toc531519031)

[**Viste** 47](#_Toc531519032)

[**Lezioni ultimo mese** 48](#_Toc531519033)

[**Lezioni ultimo mese** 48](#_Toc531519034)

[**Catalogo Corsi** 48](#_Toc531519035)

[**Eventi in Programma** 48](#_Toc531519036)

[**Iscritti ad Eventi** 48](#_Toc531519037)

[**Scheduler** 48](#_Toc531519038)

# **Progettazione**

Al solo scopo didattico per questo progetto faremo riferimento ad una fittizia società, che chiameremo "***Gym Power***", che possiede diversi centri fitness in alcune città italiane.

Faremo riferimento a nomi, codici fiscali e più in generale ad informazioni personali del tutto fittizie al solo scopo di poter creare ed operare su di un database.

Allo stesso modo abbiamo fatto per attributi come nomi degli eventi, indirizzi e nomi di esercizi; dati resi anch'essi fittizi per solo scopo didattico.

## **Sintesi dei requisiti**

Con il progetto che mostreremo si vuole creare un database per la gestione dell'organizzazione di tutte le strutture fitness della "Gym Power".

Il sistema tiene traccia di:

* iscritti che hanno sottoscritto abbonamenti
* strutture nelle varie città italiane
* corsi e lezioni nelle varie strutture
* eventi svolti nelle varie strutture
* schede esercizi sviluppate dagli istruttori e assegnate ai vari iscritti
* turni degli istruttori nelle varie strutture

Iscritti

Per quanto riguarda gli utenti si vogliono registrare gli iscritti **maggiorenni** con alcune loro informazioni personali di base, tra cui il **codice fiscale**, che useremo come **codice univoco**.

Abbonamenti

Gli abbonamenti saranno assegnati agli iscritti con un **badge** e una durata. Un abbonamento, una volta scaduto, può essere **rinnovato** conservando lo stesso badge. Ogni abbonamento prevede dei corsi, che vengono scelti dagli iscritti e aggiunti all’abbonamento, si possono scegliere un **massimo di due corsi** per abbonamento ed entrambi devono **appartenere ad una sola struttura**: non è possibile scegliere due corsi in diverse strutture. L’abbonamento di un iscritto è unico e valido in tutte le strutture: l’iscritto può infatti accedere a tutte le sedi della Gym Power per il semplice allenamento sala, ma può seguire unicamente i corsi stabiliti in fase di sottoscrizione dell’abbonamento presso le sedi scelte in quella fase, sarà quindi da memorizzare questo aspetto.

Strutture

Per quanto riguarda le sedi si registra: dove sono ubicate e la capienza massima di persone. Ogni struttura offre vari **corsi** (es. Aerobica, Karate, Pilates, ecc.) **diversi** per ogni struttura. Una struttura può offrire ai suoi iscritti un massimo di **9 corsi**.

Corsi

Abbiamo precedentemente accennato ai corsi: La Gym Power offre dei corsi a scelta ai propri iscritti per un programma di allenamento di gruppo seguito e monitorato da un istruttore esperto. Dei corsi interessa conoscere le **date** di inizio, di fine, gli orari e i giorni programmati, interessa registrare però anche le lezioni singole effettive svolte con **date**, ora di inizio, fine e numero stanza. I corsi sono tenuti da un singolo istruttore e non è possibile che un istruttore diverso segua un corso assegnato ad un altro istruttore, poiché l’assegnazione di un istruttore ad un corso è stabilita in base alle abilità e conoscenze dello stesso.

Eventi

La “Gym Power”, saltuariamente, organizza eventi o manifestazioni, a scopo promozionale e non, in cui gli iscritti possono **partecipare** gratuitamente con previa registrazione e senza bisogno di avere un abbonamento valido: sono invitati anche ex clienti e semplici utenti che andranno però registrati. Degli eventi si registrano il nome, la data e la struttura in cui viene svolto l’evento. In uno **stesso** **giorno** non possono verificarsi più di **4 eventi** in una singola struttura. Gli eventi sono diversi per ogni struttura.

Istruttori

Gli istruttori hanno il compito di essere presenti in sala ed essere presenti alle lezioni del corso che tengono, avendo come orario lavorativo un proprio turno personale. Degli istruttori interessa registrare **informazioni personali di base**, **data** e orario dei **turni** svolti presso una struttura e il **corso** al quale lavorano. Un istruttore non può svolgere più di **due turni al giorno** e non più di **27 turni in un mese solare**. Un istruttore non può lavorare a più di **4 corsi contemporaneamente.** L’istruttore per essere assunto deve avere un’età maggiore di **25 anni.**

Scheda Esercizi

La scheda esercizi viene scritta da un **singolo** istruttore e registrata nel database, con la possibilità di specificare la difficoltà. Di essa interessano gli **esercizi**, con serie, ripetizioni e livello di difficoltà, consigliati da svolgere nell’allenamento libero (seduta in sala attrezzi). Una singola scheda può essere assegnata a **più iscritti**. Quando viene assegnata, la scheda ha una **durata** che viene scelta arbitrariamente dall’istruttore. La scheda non può contenere più di **6 esercizi**. La difficoltà della scheda viene stabilita dall’istruttore.

### **Glossario**

Di seguito riportiamo un piccolo glossario per chiarire meglio alcuni termini ed espressioni usate nella relazione:

|  |  |  |
| --- | --- | --- |
| TERMINE | DESCRIZIONE | SINONIMI |
| Iscritto | Persona che ha registrato i propri dati personali presso una struttura per sottoscrivere un abbonamento. | Utente – Individuo – Cliente –Persona |
| Scheda Esercizi | Programma di allenamento registrato. | Scheda – Lista esercizi |
| Corso | Programmazioni dei corsi fitness da presentare ai clienti come programma in fase di sottoscrizione abbonamento. | Programma allenamento fitness |
| Lezione | Singola seduta di allenamento e lezione effettiva di un determinato corso. | Seduta di allenamento |
| Allenamento libero | Semplice allenamento libero in sala attrezzi (Diverso da quello svolto durante la lezione di un corso). | Seduta di allenamento – allenamento in Sala attrezzi |
| Evento | Giornate dedicate a competizioni, prove di attrezzi e incontri con esperti per promuovere il marchio. | Manifestazioni – Gare – Festa |
| Turno | Strettamente correlato ad un istruttore, rappresenta l’effettiva presenza di quest’ultimo nella data e orari indicati. | Presenza |
| Struttura | Singola sede della catena di palestre “Gym Power”. | Sede – Palestra |
| Livello Scheda | Indica la difficoltà della scheda: Alto, Medio, Basso. | Difficoltà |
| Istruttore | Persona che lavora presso la struttura. Svolge i turni e lavora ai corsi. | Dipendente |
| Esercizio | Esercizi che devono essere eseguiti in una sala di attrezzi. Un insieme di esercizi crea un programma di allenamento (scheda esercizi). | Allenamento libero |
| Serie | Si intende un gruppo di esercizi svolti consecutivamente e seguiti da un tempo di recupero che definisce il termine della sequenza effettuata. |  |
| Ripetizioni | Sono il numero degli esercizi compiuti in una serie. |  |

## **Sintesi EE/R**

Individuiamo 3 macro-aree di interesse per definire lo schema della struttura, le quali verranno riprese anche nei paragrafi successivi.

1. **Gestione degli Iscritti**

Quest’area riguarda la gestione degli iscritti, più in particolare la sottoscrizione degli abbonamenti e la partecipazione degli iscritti agli eventi.

1. **Gestione degli Istruttori**

Quest’area riguarda la gestione degli istruttori, più in particolare i turni svolti dagli istruttori e l’assegnazione dei corsi agli istruttori.

1. **Gestione delle Schede**

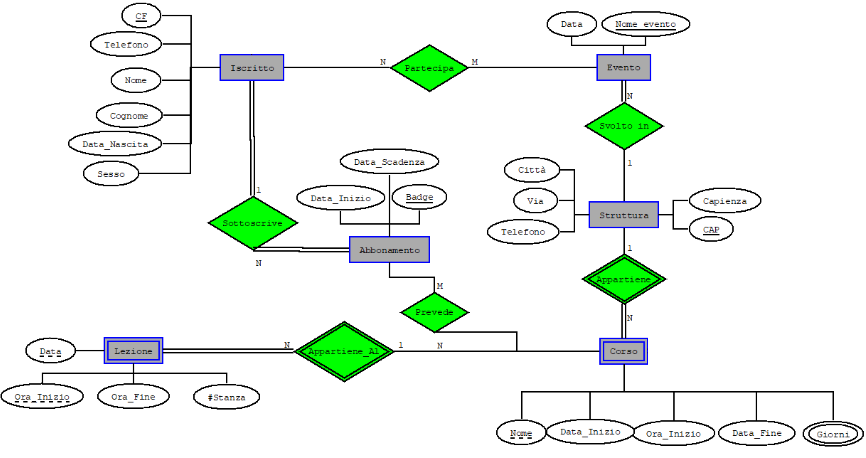
Quest’area riguarda la gestione delle schede, più in particolare un istruttore scrive una scheda che contiene vari esercizi e questa viene assegnata a più utenti.

## **Modellazione Concettuale**

In questa sezione viene spiegato lo schema *EE/R* che è stato creato, in base ai requisiti raccolti, per la rappresentazione concettuale della base di dati.

Le entità e le associazioni dello schema saranno analizzate dividendole nelle tre "macro-aree" individuate nella sezione precedente.

### **Gestione Degli Iscritti**



La prima macro-area è composta dalle seguenti entità:

* **Iscritto**
* **Evento**
* **Struttura**
* **Abbonamento**
* **Corso**
* **Lezione**

L’entità di partenza di questa macro-area è l’entità iscritto che può sottoscrivere abbonamenti e partecipare ad eventi. Entità fondamentale è anche quella di Struttura che è il punto di partenza per la gestione delle sedi.

#### **Sottoscrizione Abbonamenti**

Se l’iscritto sceglie di sottoscrivere un abbonamento, viene rilasciato un abbonamento con una durata e con una lista di massimo 2 corsi che appartengono ad una sola struttura.

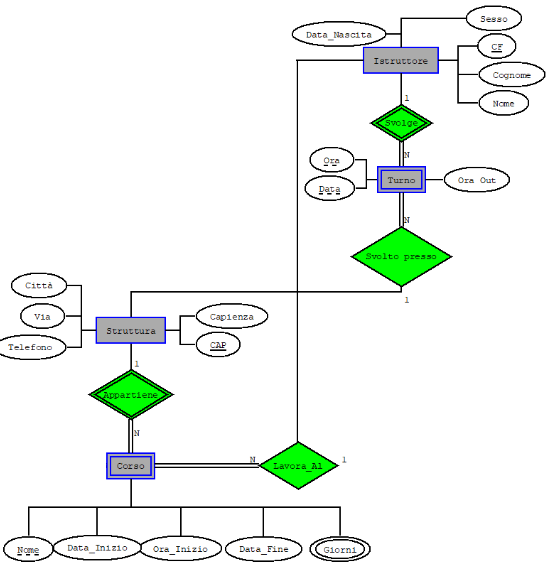
Per questioni di leggibilità si è deciso di colorare di verde le **associazioni** e di blu le **entità** **deboli** in questa e nei successivi paragrafi

* **Iscritto**: entità nel quale si registrano le persone che sottoscrivono o partecipano agli eventi, contiene le informazioni di base di contatto; la chiave è il **codice fiscale**.
* **Sottoscrive**: associazione che collega l’entità **iscritto** e l’entità **abbonamento**. Serve per rappresentare gli iscritti che sottoscrivono gli abbonamenti. Si noti che c’è la totalità da ambo i membri poiché tutti gli iscritti sottoscrivono degli abbonamenti. Un iscritto sottoscrive N abbonamenti.
* **Abbonamento**: ad ogni iscritto viene fatto sottoscrivere un abbonamento che ha una durata mensile, bimensile, trimestrale, semestrale ed annuale. La chiave è **il numero di badge** che serve ad indentificare ogni badge con un iscritto.
* **Prevede**: associazione che collega l’entità **abbonamento** e l’entità **corso**. Serve per memorizzare tutti i corsi compresi in un abbonamento. Si noti che non è presente totalità da ambo i membri poiché non tutti gli iscritti scelgono dei corsi da aggiungere all’abbonamento (solo allenamento in sala) e non tutti i corsi sono scelti in abbonamenti: potrebbe esserci un corso che non è stato scelto da nessuno degli iscritti.
* **Corso**: I corsi hanno una durata e sono identificati da un **nome**. È un’entità debole poiché l’esistenza dell’entità corso dipende dall’entità struttura: un corso “aerobica” in una struttura sarà scollegato da un corso “aerobica” in un'altra struttura.
* **Lezione**: Serve per rappresentare la lezione fisica del corso: mentre il corso rappresenta l’intero corso con giorni e orari teorici, la lezione fisica rappresenta la sessione di allenamento effettiva tenutasi di quel corso. Si registrano la **data** e la **durata**. È un’entità debole poiché l’esistenza dell’entità lezione dipende dall’entità corso che a sua volta dipende dal Cap della struttura.
* **Appartiene Al**: associazione che collega l’entità **corso** con l’entità **lezione**. Tutte le lezioni appartengono ad un solo corso ma non tutti i corsi avranno lezioni certe.
* **Struttura**: entità nella quale vengono registrati le informazioni in cui sono ubicate le varie strutture. La chiave scelta è il **CAP** poiché non si prevedono nei progetti della società 2 strutture così ravvicinate che rientrino in un solo CAP.

#### **Partecipazione Eventi**

* **Evento**: entità nella quale vengono registrati gli eventi. La chiave scelta è il **nome**: I nomi degli eventi non saranno mai uguali poiché nel nome rientrerà la numerazione o l’anno dell’evento (es IV GARA BODYBULDING). Si registra anche la data in cui viene svolto.
* **Partecipa**: associazione che collega l’entità **iscritto** e l’entità **evento**. Serve a rappresentare gli iscritti che partecipano ai corsi. Un iscritto partecipa ad N eventi e ad un evento partecipano M iscritti. Non ci sono totalità.
* **Svolto In:** associazione che collega l’entità **evento** e l’entità **struttura**. Serve a rappresentare gli eventi che vengono svolti in una struttura. Tutti gli eventi sono svolti in strutture ma non tutte le strutture ospitano eventi.

### **Gestione Degli Istruttori**



La seconda macro-area è composta dalle seguenti entità:

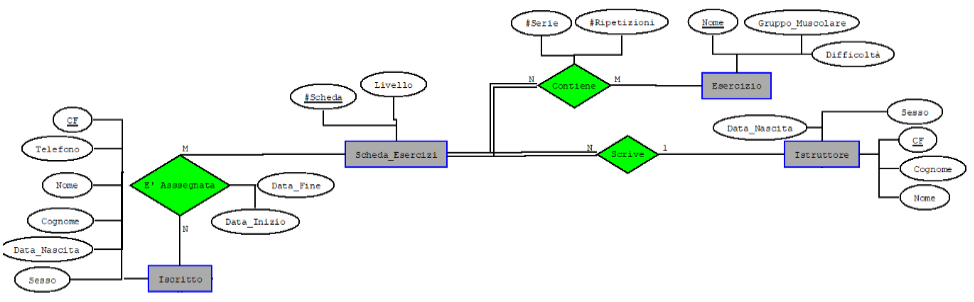
* Istruttore
* Turno
* Corso
* Struttura

L’entità primaria di questa macro-area è l’entità istruttore.

In questa macro-area interessano quali turni e corsi effettuano gli istruttori.

* **Istruttore**: entità nella quale si registrano i dipendenti che lavorano nelle strutture ed ai corsi, contiene le informazioni di base di contatto; la chiave è il **codice fiscale**.
* **Svolge**: associazione che collega l’entità **istruttore** e l’entità **turno**. Serve per rappresentare quali turni effettua un istruttore. Si noti che c’è la totalità solo dal lato turni poiché tutti i turni vengono svolti da un istruttore.
* **Turno**: ogni istruttore esegue un turno che viene svolto presso una struttura. I turni hanno una **durata** e una **data**. È entità debole poiché esistenza dell’entità **turno** dipende dall’esistenza **istruttore**. Le chiavi deboli sono **data** ed **ora**.
* **Svolto Presso**:associazione che collega l’entità **turno** e l’entità **struttura**. Serve per rappresentare i turni svolti presso una struttura. Si noti che la totalità dal lato di turno poiché tutti i turni sono svolti presso una struttura. Un turno è svolto presso una sola struttura, che invece ospita più turni.
* **Lavora Al**: associazione che collega l’entità **istruttore** e l’entità **corso**. Serve per rappresentare gli istruttori che lavorano ai corsi. Un istruttore lavora a più corsi, un corso è gestito da un solo istruttore. Tutti i corsi sono gestiti da istruttori ma non tutti gli istruttori lavorano un corso.

### **Gestione Delle Schede**



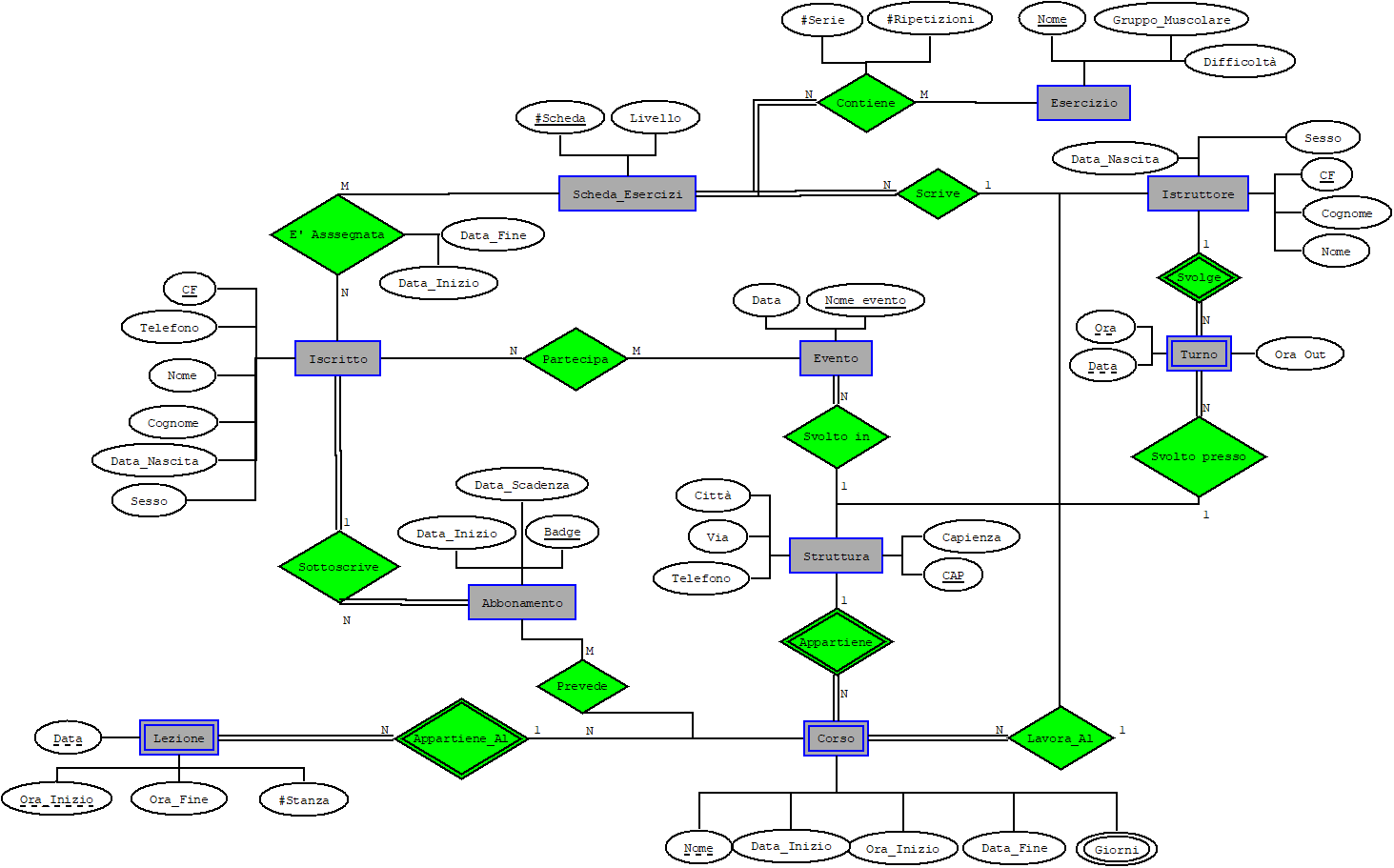
La seconda macro-area è composta dalle seguenti entità:

* Istruttore
* Scheda Esercizi
* Iscritto
* Esercizio

In questa macro-area interessano quali schede esercizi scrivono gli istruttori ed a quali iscritti vengono assegnate.

* **Scrive**: associazione che collega l’entità **istruttore** e l’entità **scheda esercizi**. Serve per rappresentare quali schede esercizi vengono scritte da un istruttore. Si noti che c’è la totalità solo dal lato turni poiché tutte le schede vengono scritte da un istruttore. Un istruttore scrive N schede
* **Scheda Esercizi**: entità nel quale si registrano le schede che vengono scritte dagli istruttori, contiene il livello e il **numero della scheda**, scelta come la chiave.
* **Contiene**: associazione che collega l’entità **scheda esercizi** e l’entità **esercizio**. Serve per rappresentare quali esercizi contengono le schede. Si noti che c’è la totalità solo dal lato della scheda poiché tutte le schede contengono almeno due esercizi. Si noti anche che l’associazione ha due attributi **numero serie e ripetizioni** che stanno ad indicare come bisogna svolgere un esercizio.
* **Esercizio**: ogni scheda contiene almeno due esercizi. Gli esercizi hanno un **nome**, scelto come chiave, il gruppo muscolare interessato e la difficolta.
* **È assegnata**:associazione che collega l’entità **scheda esercizi** e l’entità **iscritto**. Serve per rappresentare le schede che vengono assegnate agli iscritti. Si noti che l’associazione ha due attributi che stanno ad indicare la durata della scheda. Non vi sono totalità e un iscritto può avere N schede e la stessa scheda può essere assegnata ad M iscritti.

### **Diagramma EE/R**

****

## **Modellazione Logica**

In questo capitolo sarà descritto il passaggio dalla modellazione concettuale a quella logica, passando dal diagramma *EE/R* a quello *Relazione*.

Questo diagramma si occupa di definire i vari soggetti del database e le interazioni fra di essi.

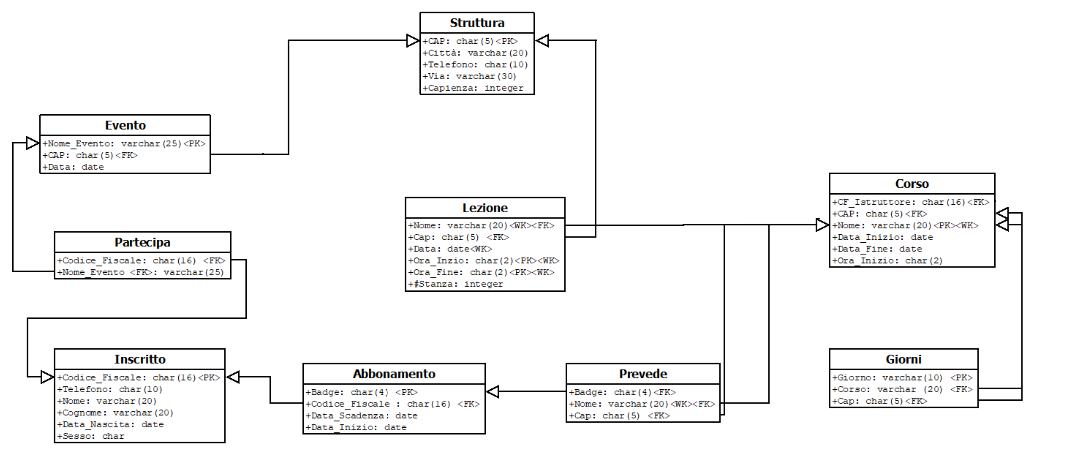
Si definisce infatti come **Schema Relazionale** la terna , rispettivamente:

* i-**R**elazioni : consistono nelle tabelle rappresentanti le entità;
* j-**I**ntegrity **C**ondition: consistono nei vari vincoli che vengono specificati per garantire l'integrità della base di dati;
* k-**F**unctional **D**ipendence: sono delle particolari dipendenze di attributi in funzione di altri;

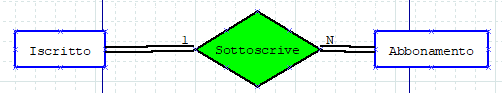
Sarà quindi effettuata una vera e propria traduzione mediante le regole studiate durante il corso per rappresentare gli elementi dello schema relazionale sopra descritti.

Anche in questo caso l'analisi sarà divisa nelle tre macro-aree già viste nella [modellazione concettuale](#_Modellazione_Concettuale).

### **Traduzione: Gestione Degli Iscritti**

**Sottoscrizione Abbonamenti**

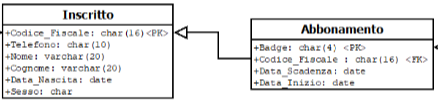
**Iscritto – Abbonamento**



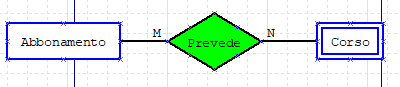
Sono due entità forti con l’associazione 1:N, viene tradotta trasformando le entità in due tabelle.

Si aggiunge alla tabella che si trova dal lato N come chiave esterna, la chiave primaria della tabella che si trova al lato 1.

Quindi **Abbonamento** contiene come chiave esterna, la chiave primaria di **Iscritto**.



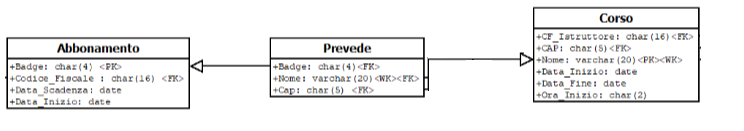
**Abbonamento – Prevede – Corso**



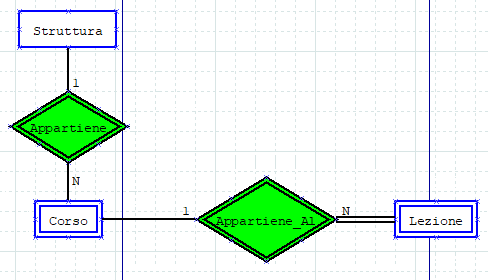
Sono due entità forti con l’associazione M:N. Ciò comporta una tabella di transizione.

La tabella sarà caratterizzata dal fatto di avere come chiave primaria, la combinazione delle chiavi primarie delle tabelle di cui rappresenta l'associazione. Inoltre, gli attributi dell'associazione vanno a finire proprio su questa tabella.

Quindi la tabella **Prevede** contiene la chiave di **Abbonamento** e la chiave di **Corso.**

****

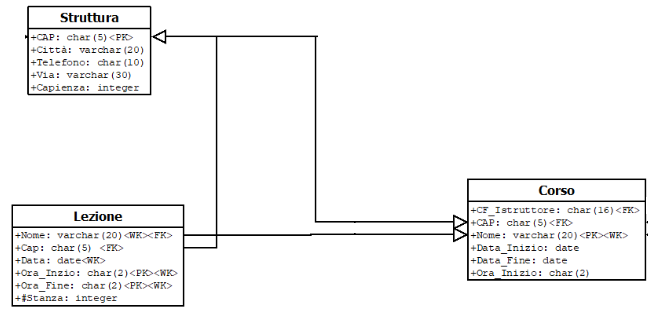
**Struttura – Corso – Lezione**



L’entità **Struttura** è un entità **forte** mentre l’entità **Corso** e **Lezione** sono entità **deboli**, quindi la chiave primaria dell'entità forte diventa un membro della chiave primaria delle entità deboli.

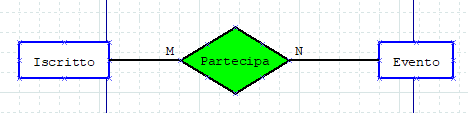
La tabella **Corso** avrà come chiave esterna, la chiave primaria di **Struttura.**

La tabella Lezione avrà come chiave esterna, la chiave di **Corso** e la chiave di **Struttura.**

****

#### **Partecipazione Eventi**

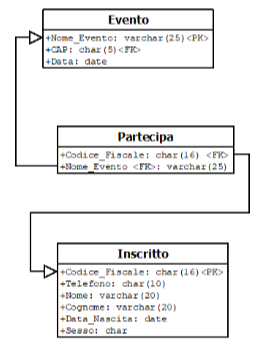
**Iscritto – Partecipa – Evento**



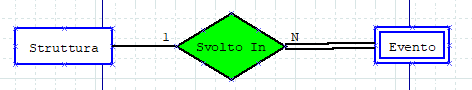
Sono due entità forti con l’associazione M:N. Ciò comporta una tabella di transizione.

La tabella sarà caratterizzata dal fatto di avere come chiave primaria, la combinazione delle chiavi primarie delle tabelle di cui rappresenta l'associazione. Inoltre, gli attributi dell'associazione vanno a finire proprio su questa tabella.

Quindi la tabella **Partecipa** contiene la chiave di **Iscritto** e la chiave di **Evento.**



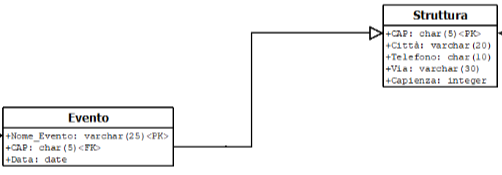
**Evento – Struttura**



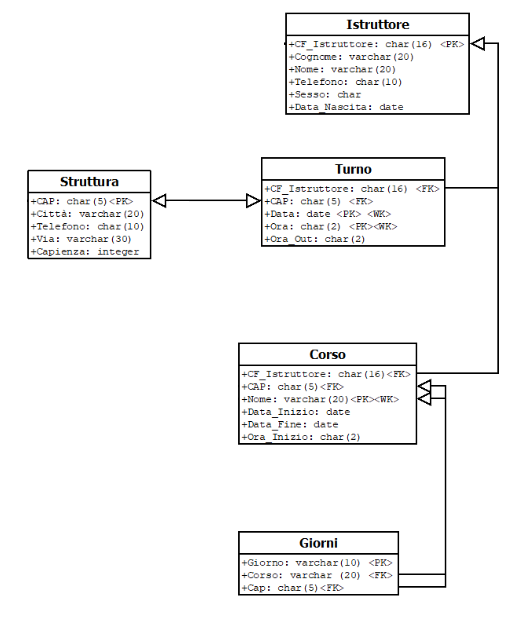
Sono due entità forti con l’associazione 1:N, viene tradotta trasformando le entità in due tabelle.

Si aggiunge alla tabella che si trova dal lato N come chiave esterna, la chiave primaria della tabella che si trova al lato 1.

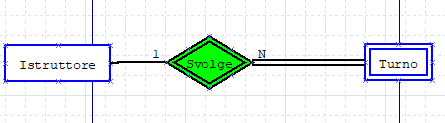
Quindi **Evento** contiene come chiave esterna, la chiave primaria di **Struttura**.



### **Traduzione: Gestione Degli Istruttori**

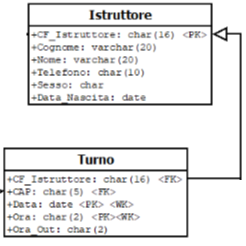


**Istruttore – Turno**

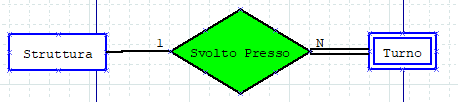


L’entità **Istruttore** è un entità **forte** mentre l’entità **Turno** è un’entità **debole**, quindi la chiave primaria dell'entità forte diventa un membro della chiave primaria dell'entità debole.

La tabella **Turno** avrà come chiave esterna, la chiave primaria di **Istruttore.**

****

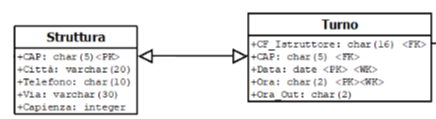
**Struttura – Turno**



Sono due entità forti con l’associazione 1:N, viene tradotta trasformando le entità in due tabelle.

Si aggiunge alla tabella che si trova dal lato N come chiave esterna, la chiave primaria della tabella che si trova al lato 1.

Quindi **Turno** contiene come chiave esterna, la chiave primaria di **Struttura**.



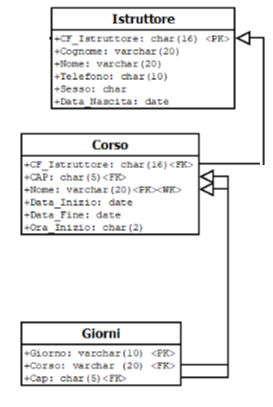
**Istruttore – Corso**



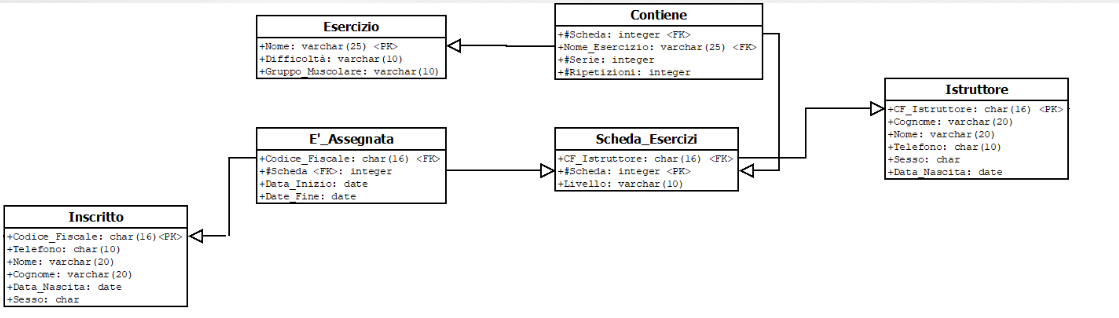
Sono due entità forti con l’associazione 1:N, viene tradotta trasformando le entità in due tabelle.

Si aggiunge alla tabella che si trova dal lato N come chiave esterna, la chiave primaria della tabella che si trova al lato 1.

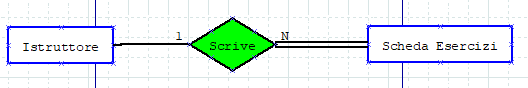
Quindi **Corso** contiene come chiave esterna, la chiave primaria di **Istruttore**.

****

### **Traduzione: Gestione Delle Schede**



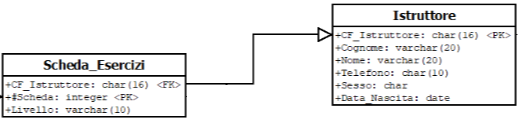
**Istruttore – Scheda Esercizi**



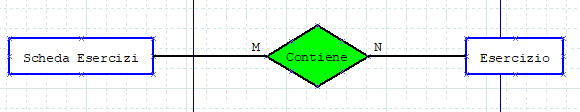
Sono due entità forti con l’associazione 1:N, viene tradotta trasformando le entità in due tabelle.

Si aggiunge alla tabella che si trova dal lato N come chiave esterna, la chiave primaria della tabella che si trova al lato 1.

Quindi **Scheda Esercizi** contiene come chiave esterna, la chiave primaria di **Istruttore**.

****

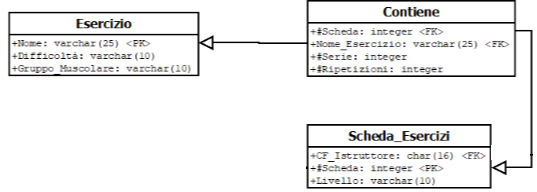
**Scheda Esercizi – Contiene – Esercizio**



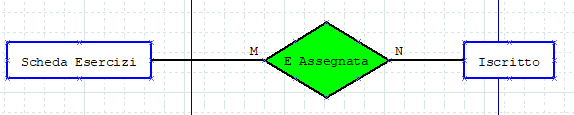
Sono due entità forti con l’associazione M:N. Ciò comporta una tabella di transizione.

La tabella sarà caratterizzata dal fatto di avere come chiave primaria, la combinazione delle chiavi primarie delle tabelle di cui rappresenta l'associazione. Inoltre, gli attributi dell'associazione vanno a finire proprio su questa tabella.

Quindi la tabella **Contiene** la chiave di **Scheda Esercizi** e la chiave di **Esercizio.**

****

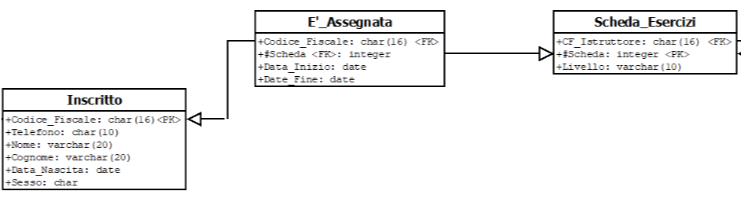
**Scheda Esercizi – Assegnata – Iscritto**



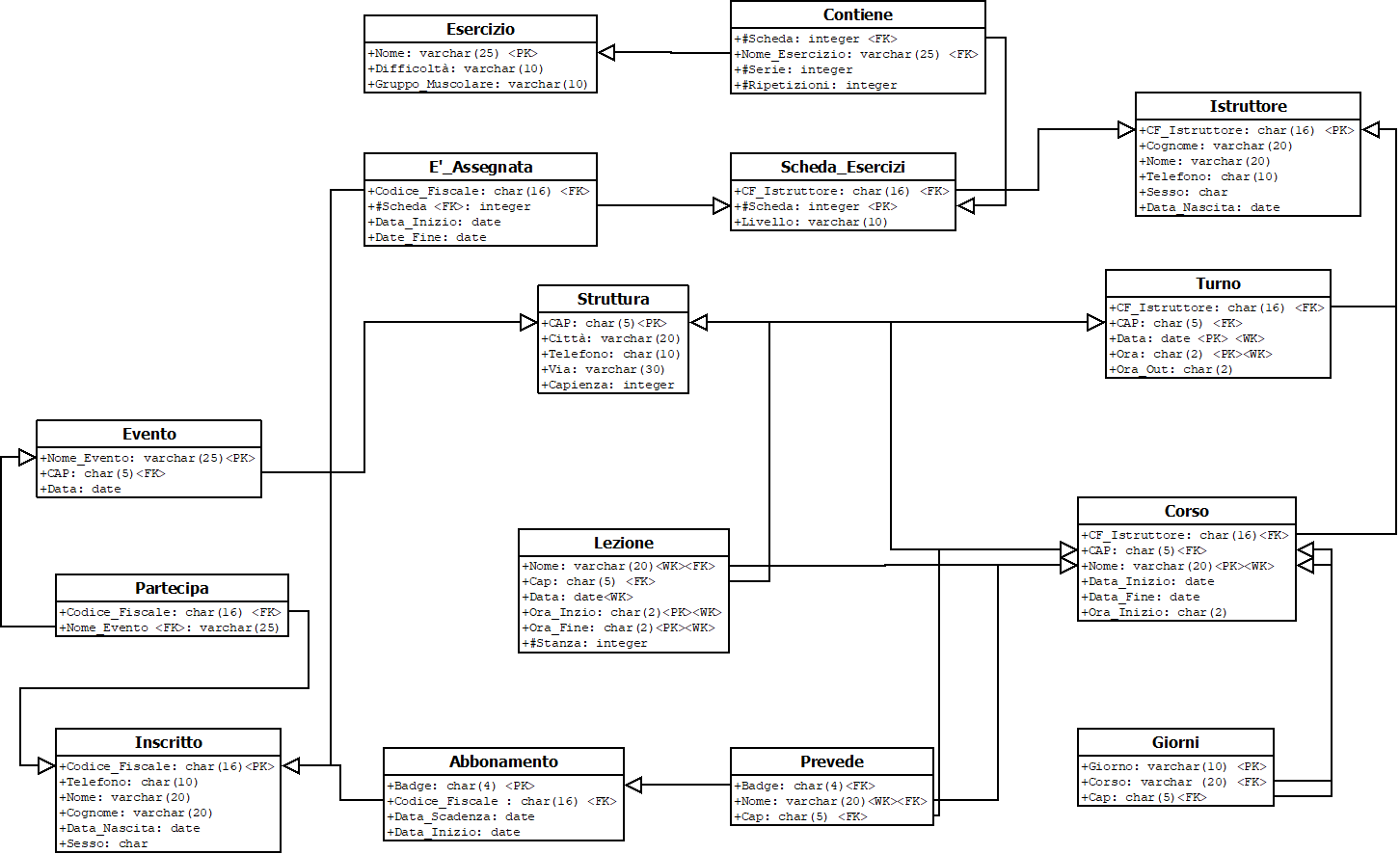
Sono due entità forti con l’associazione M:N. Ciò comporta una tabella di transizione.

La tabella sarà caratterizzata dal fatto di avere come chiave primaria, la combinazione delle chiavi primarie delle tabelle di cui rappresenta l'associazione. Inoltre, gli attributi dell'associazione vanno a finire proprio su questa tabella.

Quindi la tabella **E** **Assegnata** contiene la chiave di **Scheda Esercizi** e la chiave di **Iscritto.**



### **Diagramma Relazionale**



## **Definizione Utenti**

In questo paragrafo si definiscono tutti gli utenti che accedono al database e le operazioni possibili.

Il sistema permette l’accesso al database a utenti differenti:

* **Direttore**: è il **proprietario** ed **amministratore** del database. Ha permessi su tutto.
* **Segretario**: si occupa dei **clienti**, dei loro **abbonamenti** e della gestione degli **eventi**.
* **Istruttore**: può accedere al database per **scrivere** ed **assegnare** **schede** ai **clienti**.

### **Permessi Utenti**

Di seguito si mostra una tabella riassuntiva dei permessi degli utenti.

|  |  |  |
| --- | --- | --- |
| UTENTE | TIPO | PERMESSI |
| Direttore | Amministratore | ALL  EXECUTE ON LISTA\_TURNI\_ISTRUTTORE  EXECUTE ON ASSUMI\_ISTRUTTORE  EXECUTE ON LICENZIA\_ISTRUTTORE  EXECUTE ON CREA\_EVENTO  EXECUTE ON INFO\_ISCRITTO  EXECUTE ON SOTTOSCRIVI\_ABBONAMENTI  SELECT ON LEZIONI  SELECT ON ABBONAMENTI\_SCADUTI |
| Segretario | Comune | CONNECT  CREATE SESSION  ALL ON ISCRITTO  ALL ON ABBONAMENTO  ALL ON PARTECIPA  ALL ON EVENTO  EXECUTE ON INFO\_ISCRITTO  EXECUTE ON SOTTOSCRIVI\_ABBONAMENTO  EXECUTE ON ASSEGNA\_CORSO  SELECT ON ABBONAMENTI\_SCADUTI  SELECT ON CORSI\_ISTRUTTORI  SELECT ON EVENTI\_PROGRAMMA  SELECT ON ISCRITTO\_EVENTO |
| Istruttore | Comune | CONNECT  CREATE SESSION  ALL ON SCHEDA\_ESERCIZI  ALL ON CONTIENE  ALL ON ESERCIZI  ALL ON E\_ASSEGNATA  EXECUTE ON CREA\_SCHEDA  EXECUTE ON ASSEGNA\_SCHEDA |

### **Tabella Dei Volumi**

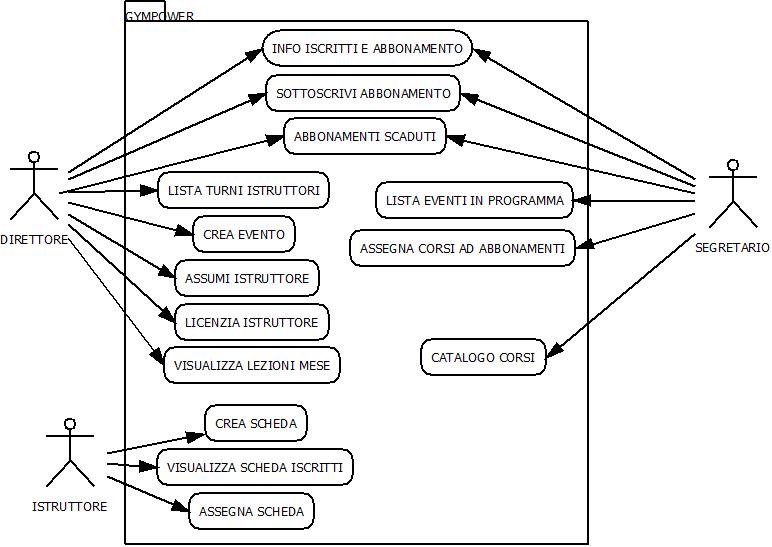
La tabella dei volumi ha lo scopo di dare un’idea di quanti dati sono circa presenti nel database e un ipotesi su quanti circa ne arriveranno nel lungo periodo.

|  |  |  |  |
| --- | --- | --- | --- |
| TABELLA | QUANTITA | INCREMENTO | PERIODO |
| STRUTTURA | 8 | 2 | BIENNIO |
| ISCRITTO | 100 | 100 | ANNUALE |
| ESERCIZIO | 40 | 3 | ANNUALE |
| ISTRUTTORE | 30 | 5 | ANNUALE |
| SCHEDA\_ESERCIZI | 30 | 20 | ANNUALE |
| EVENTI | 25 | 20 | ANNUALE |
| ABBONAMENTI | 100 | 100 | ANNUALE |
| CORSI | 70 | 10 | ANNUALE |
| LEZIONI | 120 | 100 | ANNUALE |

### **Diagrammi Dei Casi D’Uso**

Di seguito vengono elencate e spiegate in dettaglio le operazioni automatizzate, complesse e di base, che possono eseguire i vari utenti del database.

* **Il segretario** può stilare informazioni su iscritti e abbonamenti, sottoscrivere abbonamenti, listare gli abbonamenti scaduti, listare eventi in programma, assegnare corsi ad abbonamenti e listare il catalogo corsi.
* **Il direttore** può anche lui stilare informazioni su iscritti e abbonamenti, sottoscrivere abbonamenti, listare gli abbonamenti scaduti, ma può anche listare i turni svolti dagli istruttori o le lezioni svolte nell’ultimo mese, creare evento, assumere istruttori o licenziarli.
* **L’istruttore** può creare una scheda, visualizzare le schede degli iscritti assegnare schede

****

### **Tabelle Operazioni Degli Utenti**

Le tabelle delle operazioni degli utenti rappresentano le operazioni che possono effettuare gli utenti.

Di seguito vengono descritte in dettaglio le varie operazioni:

|  |  |
| --- | --- |
| **OPERAZIONE** | CREA SCHEDA |
| **INPUT** | Codice fiscale istruttore che crea la scheda, la difficoltà è nome esercizio, serie e ripetizioni per 6 volte (massimo numero esercizi da poter immettere nella scheda) |
| **SCOPO** | Lo scopo è la creazione di una nuova scheda esercizio che abbia minimo 2 esercizi e massimo 6. per ogni esercizio si assegnano numero sere e ripetizioni. se viene inserito in input è immessa anche la difficoltà |
| **ERRORI** | Se viene inserito in input un solo esercizio |
| **TABELLE** | Contiene – Scheda\_esercizi – esercizio |

|  |  |
| --- | --- |
| **OPERAZIONE** | SOTTOSCRIVI ABBONAMENTO |
| **INPUT** | Codice fiscale iscritto e numero di mesi per cui si vuole sottoscrivere l’abbonamento |
| **SCOPO** | Lo scopo è la creazione o il rinnovo di un abbonamento per un iscritto di validità 1-3-6 o 12 mesi (in caso di abbonamento per la prima volta una sequenza automatica genererà il badge da inserire nell’abbonamento) |
| **ERRORI** | Se viene inserito in input una durata di mesi diversa da 1,3,6 o 12 genera un errore |
| **TABELLE** | abbonamento – iscritto |

|  |  |
| --- | --- |
| **OPERAZIONE** | INFORMAZIONI ISCRITTO |
| **INPUT** | Codice fiscale iscritto |
| **SCOPO** | Lo scopo è la creazione di un listato in cui compaiano:  - dati anagrafici  - informazioni abbonamento  - corsi a cui è iscritto  - scheda esercizio con esercizi |
| **ERRORI** | Il codice fiscale dell’iscritto non è presente nel database e un warning se l’iscritto non possiede una scheda esercizi |
| **TABELLE** | abbonamento – iscritto – scheda esercizi – contiene – prevede |

|  |  |
| --- | --- |
| **OPERAZIONE** | LISTA TURNI ISTRUTTORE |
| **INPUT** | Codice fiscale istruttore |
| **SCOPO** | Lo scopo è la creazione di un listato in cui compaiano:  - dati anagrafici  - conteggio turni totali e ultimo mese  - elenco turni  - corsi che segue |
| **ERRORI** | L’istruttore non riporta turni eseguiti |
| **TABELLE** | istruttore – corso – turno |

|  |  |
| --- | --- |
| **OPERAZIONE** | ASSEGNA SCHEDA |
| **INPUT** | Codice fiscale iscritto, numero scheda e data scadenza della scheda |
| **SCOPO** | Lo scopo è l’assegnazione di una scheda esercizi presente nel database ad un iscritto, mostrando a video la variazione di livello se c’è stata |
| **ERRORI** | Warning: la vecchia scheda dell’iscritto non era ancora scaduta, sarà automaticamente fatta scadere istantaneamente |
| **TABELLE** | scheda\_esercizi – e\_assegnata - iscritto |

|  |  |
| --- | --- |
| **OPERAZIONE** | ASSEGNA CORSO |
| **INPUT** | Codice fiscale iscritto, nome corso 1 e nome corso 2, cap dei corsi |
| **SCOPO** | Lo scopo è assegnare ad un abbonamento di un iscritto preso in input due corsi. |
| **ERRORI** | * errore se l’abbonamento già comprende due corsi scelti * errore nell’inserire il secondo corso se il primo corso rappresenta il secondo corso dell’abbonamento |
| **TABELLE** | corso – abbonamento – iscritto |

Invece di seguito vengono mostrare le **operazioni** **di** **base** e le **viste**.

|  |  |
| --- | --- |
| OPERAZIONE | SCOPO |
| CREA EVENTO | Genera un nuovo evento. |
| ASSUMI ISTRUTTORE | Inserisce un nuovo istruttore nella tabella istruttori. |
| LICENZIA ISTRUTTORE | Cancella un istruttore dalla tabella. |
| ABBONAMENTI SCADUTI | Vista che mostra tutti gli abbonamenti scaduti. |
| CATALOGO CORSI | Vista che mostra tutti i corsi disponibili con gli istruttori che li tengono |
| ISCRITTI AD EVENTI | Vista che mostra tutti gli iscritti a tutti gli eventi che hanno partecipato |
| EVENTI IN PROGRAMMA | Vista che mostra tutti gli eventi in programma. |
| LEZIONI SVOLTE ULTIMO MESE | Vista che mostra le lezioni svolte nell’ultimo mese |

## **Vincoli**

In questo paragrafo vengono elencati i vincoli **statici** e **dinamici.** I vincoli *statici* sono implementati nella base di dati tramite istruzioni DDL, invece, quelli *dinamici*sono implementati tramite **Trigger**.

### **Vincoli Statici**

|  |  |  |  |
| --- | --- | --- | --- |
| TABELLA | ATTRIBUTO | VINCOLO | DESCRIZIONE |
| STRUTTURA | TELEFONO | UNIQUE | Deve essere unico |
| ISCRITTO | TELEFONO  SESSO  CF\_CONSENTITO | UNIQUE  CHECK(SESSO IN ('M','F'))  CHECK(REGEXP\_LIKE(CF\_ISCRITTO, '^[A-Z]{6}\d{2}[A-Z]\d{2}[A-Z]\d{3}[A-Z]$')) | Deve essere unico  Deve assumere solo due valori M e F rispettivamente maschio e femmina  Controlla se il codice fiscale è corretto |
| ESERCIZO | DIFFICOLTA  GRUPPO MUSCOLARE | CHECK (DIFFICOLTA IN ('ALTA','MEDIA','BASSA'))  CHECK (GRUPPO\_MUSCOLARE IN ('PETTO','DORSO','BRACCIA','  ADDOME','GAMBE','SPALLE')) | Deve assumere valori alta, media, bassa  Deve assumere solo valori tra i seguenti: petto, dorso, braccia, addome,  gambe, spalle |
| ISTRUTTORE | SESSO\_ISTRUTTORE  CF\_ISTRUTTORE\_  CONSENTITO | CHECK(SESSO IN ('M','F'))  CHECK(REGEXP\_LIKE(CF\_ISCRITTO, '^[A-Z]{6}\d{2}[A-Z]\d{2}[A-Z]\d{3}[A-Z]$')) | Deve assumere solo due valori M e F rispettivamente maschio e femmina  Controlla se il codice fiscale è corretto |
| SCHEDA\_ESERCIZI | LIVELLO | CHECK(LIVELLO IN ('ALTO','MEDIO','BASSO',NULL)) | Deve assumere valori alto, medio, basso |
| CONTIENE | N\_SERIE  N\_RIPETIZIONI | CHECK(N\_SERIE <= 5 AND N\_SERIE > 0)  CHECK(N\_RIPETIZIONI <= 15 AND N\_RIPETIZIONI > 0) | Il valore deve essere compreso tra 0 escluso e 5 incluso  Il valore deve essere compreso tra 0 escluso e 15 incluso |

### **Vincoli Dinamici**

|  |  |  |
| --- | --- | --- |
| TRIGGER | OPERA SULLA TABELLA | DESCRIZIONE |
| controllo\_esercizi\_scheda | CONTIENE | Il numero degli esercizi in una scheda non sia maggiore di 6. |
| controllo\_turno | TURNO | * Un istruttore non deve svolgere più di 2 turni in un giorno * Un istruttore non svolgere più di 27 turni in un mese solare. |
| inserimento\_corso | CORSO | * Un corso non si può inserire con una data iniziale minore della data corrente * Non si può assegnare un corso ad un istruttore se quest’ultimo lavora a più di 4 corsi in contemporanea * Non si può inserire un corso in una struttura che abbia già più di 9 corsi. |
| controllo\_abbonamento | PREVEDE | * Il numero massimo dei corsi che può prevedere un abbonamento è 2 * Non si possono inserire 2 corsi che si tengano in sedi diverse in uno stesso abbonamento. |
| ISTRUTTORE\_ABILITATO | ISTRUTTORE | Un istruttore deve avere più di 25 anni per essere assunto. |
| CONTROLLO\_EVENTI | EVENTO | La struttura non può svolgere più di 4 eventi in contemporanea in un giorno. |
| ISCRITTO\_MINORENNE | ISCRITTO | Un iscritto per essere inserito nel database deve essere maggiorenne. |

## **Normalizzazione**

Con le seguenti tre forme si effettua una valutazione teorica della qualità della base di dati e dello schema relazionale. Verificheremo con i tre test le tre forme normali.

* **Prima forma normale:** Richiede che tutti gli **attributi** siano ***atomici***, quindi la prima forma normale è rispettata automaticamente in tutti i modelli relazionali.
* **Seconda forma normale:** Uno schema è in seconda forma normale se e solo se tutti gli **attributi** **non-primi** dipendono completamente da ogni chiave (cioè non dipendono solamente da una parte della chiave). Questa seconda forma normale quindi riguarda le **chiavi** **multi-attributi**. Nel nostro caso specifico le tabelle con chiavi multi-attributi quali *Lezioni*, *Corso*, *Turno*, *Prevede*, *E Assegnata*, *Contiene* e *Partecipa* hanno **attributi** **non-primi** che dipendono funzionalmente dalle chiavi: nessun attributo dipende in maniera parziale dalle chiavi. Quindi la seconda forma normale è rispettata.
* **Terza Forma Normale:** Uno schema è in terza forma normale se per ogni dipendenza funzionale , è una **chiave** oppure è **primo**. Dopo un attento test, notiamo che le tabelle Struttura ed Iscritto presentano un attributo Telefono che viola la Terza Forma Normale. Lo schema risulta non essere quindi in 3NF.

# **Implementazione**

In questo capitolo si tratterà dell’implementazione del codice *PL/SQL,* nel dettaglio si vedranno le operazioni *DCL, DDL* e *DML*.

## **DCL: Creazione & Permessi Degli Utenti**

In questo paragrafo si implementano gli utenti ed i relativi permessi.

1. -- Creazione Utenti
3. **CREATE** USER DIRETTORE IDENTIFIED **BY** admin;
4. **CREATE** USER SEGRETARIO IDENTIFIED **BY** segretario;
5. **CREATE** USER ISTRUTTORE IDENTIFIED **BY** istruttore;

8. -- Assegnazione permessi
10. --DIRETTORE
11. **GRANT** ALL **PRIVILEGES** **TO** DIRETTORE;
12. **GRANT** **EXECUTE** **ON** LISTA\_TURNI\_ISTRUTTORE **TO** DIRETTORE;
13. **GRANT** **EXECUTE** **ON** ASSUMI\_ISTRUTTORE **TO** DIRETTORE;
14. **GRANT** **EXECUTE** **ON** LICENZIA\_ISTRUTTORE **TO** DIRETTORE;
15. **GRANT** **EXECUTE** **ON** CREA\_EVENTO **TO** DIRETTORE;
16. **GRANT** **EXECUTE** **ON** INFO\_ISCRITTO **TO** DIRETTORE;
17. **GRANT** **EXECUTE** **ON** SOTTOSCRIVI\_ABBONAMENTO **TO** DIRETTORE;
18. **GRANT** **SELECT** **ON** LEZIONI **TO** DIRETTORE;
19. **GRANT** **SELECT** **ON** ABBONAMENTI\_SCADUTI **TO** DIRETTORE;

22. --SEGRETARIO
23. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** ISCRITTO **TO** SEGRETARIO;
24. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** ABBONAMENTO **TO** SEGRETARIO;
25. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** PARTECIPA **TO** SEGRETARIO;
26. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** EVENTO **TO** SEGRETARIO;
27. **GRANT** **EXECUTE** **ON** INFO\_ISCRITTO **TO** SEGRETARIO;
28. **GRANT** **EXECUTE** **ON** SOTTOSCRIVI\_ABBONAMENTO **TO** SEGRETARIO;
29. **GRANT** **EXECUTE** **ON** ASSEGNA\_CORSO **TO** SEGRETARIO;
30. **GRANT** **SELECT** **ON** ABBONAMENTI\_SCADUTI **TO** SEGRETARIO;
31. **GRANT** **SELECT** **ON** CORSI\_ISTRUTTORI **TO** SEGRETARIO;
32. **GRANT** **SELECT** **ON** EVENTI\_PROGRAMMA **TO** SEGRETARIO;
33. **GRANT** **SELECT** **ON** ISCRITTO\_EVENTO **TO** SEGRETARIO;
35. --ISTRUTTORE
36. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** SCHEDA\_ESERCIZI **TO** ISTRUTTORE;
37. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** CONTIENE **TO** ISTRUTTORE;
38. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** ESERCIZIO **TO** ISTRUTTORE;
39. **GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** E\_ASSEGNATA **TO** ISTRUTTORE;
40. **GRANT** **EXECUTE** **ON** CREA\_SCHEDA **TO** ISTRUTTORE;
41. **GRANT** **EXECUTE** **ON** ASSEGNA\_SCHEDA **TO** ISTRUTTORE;

## **DDL: Creazioni Delle Tabelle**

In questo paragrafo si implementano le tabelle.

1. **DROP** **TABLE** STRUTTURA;
2. **DROP** **TABLE** ISCRITTO;
3. **DROP** **TABLE** CORSO;
4. **DROP** **TABLE** ABBONAMENTO;
5. **DROP** **TABLE** TURNO;
6. **DROP** **TABLE** EVENTO;
7. **DROP** **TABLE** SCHEDA\_ESERCIZI;
8. **DROP** **TABLE** ISTRUTTORE;
9. **DROP** **TABLE** ESERCIZIO;
10. **DROP** **TABLE** GIORNI;
11. **DROP** **TABLE** LEZIONE;
12. **DROP** **TABLE** E\_ASSEGNATA;
13. **DROP** **TABLE** PARTECIPA;
14. **DROP** **TABLE** CONTIENE;
15. **DROP** **TABLE** PREVEDE;

18. **CREATE** **TABLE** STRUTTURA
19. (
20. CAP                     **CHAR**(5)         **PRIMARY** **KEY**,
21. CAPIENZA                **INT**             NOT NULL,
22. TELEFONO                **CHAR**(10)        **UNIQUE**,
23. VIA                     **VARCHAR**(30)     NOT NULL,
24. CITTA                   **VARCHAR**(20)     NOT NULL
25. );
27. **CREATE** **TABLE** ISCRITTO
28. (
29. CF\_ISCRITTO             **CHAR**(16)        **PRIMARY** **KEY**,
30. TELEFONO                **CHAR**(10)        **UNIQUE**,
31. NOME                    **VARCHAR**(20)     NOT NULL,
32. COGNOME                 **VARCHAR**(20)     NOT NULL,
33. SESSO                   **CHAR**            NOT NULL,
34. DATA\_NASCITA            **DATE**            NOT NULL,
35. **CONSTRAINT** SESSO\_ISCRITTO **CHECK**(SESSO IN ('M','F')),
36. **CONSTRAINT** CFConsentito **CHECK**(REGEXP\_LIKE(CF\_ISCRITTO, '^[A-Z]{6}\d{2}[A-Z]\d{2}[A-Z]\d{3}[A-Z]$'))
37. );
39. **CREATE** **TABLE** ESERCIZIO
40. (
41. NOME                    **VARCHAR**(25)     **PRIMARY** **KEY**,
42. DIFFICOLTA              **VARCHAR**(10)     NOT NULL,
43. GRUPPO\_MUSCOLARE        **VARCHAR**(10)     NOT NULL,
44. **CONSTRAINT** DIFFICOLTA **CHECK** (DIFFICOLTA IN ('ALTA','MEDIA','BASSA')),
45. **CONSTRAINT** GRUPPO\_MUSCOLARE **CHECK** (GRUPPO\_MUSCOLARE IN ('PETTO','DORSO','BRACCIA','ADDOME','GAMBE','SPALLE'))
46. );
48. **CREATE** **TABLE** ISTRUTTORE
49. (
50. CF\_ISTRUTTORE           **CHAR**(16)        **PRIMARY** **KEY**,
51. TELEFONO                **CHAR**(10)        **UNIQUE**,
52. NOME                    **VARCHAR**(20)     NOT NULL,
53. COGNOME                 **VARCHAR**(20)     NOT NULL,
54. SESSO                   **CHAR**            NOT NULL,
55. DATA\_NASCITA            **DATE**            NOT NULL,
56. **CONSTRAINT** SESSO\_ISTRUTTORE **CHECK**(SESSO IN ('M','F')),
57. **CONSTRAINT** CFConsentito\_Istruttore **CHECK**(REGEXP\_LIKE(CF\_ISTRUTTORE, '^[A-Z]{6}\d{2}[A-Z]\d{2}[A-Z]\d{3}[A-Z]$'))
58. );
60. **CREATE** **TABLE** SCHEDA\_ESERCIZI
61. (
62. CF\_ISTRUTTORE\_SCHEDA    **CHAR**(16)        NOT NULL,
63. N\_SCHEDA                **INT**             **PRIMARY** **KEY**,
64. LIVELLO                 **VARCHAR**(10),
65. **CONSTRAINT** LIVELLO **CHECK**(LIVELLO IN ('ALTO','MEDIO','BASSO',NULL)),
66. **FOREIGN** **KEY**(CF\_ISTRUTTORE\_SCHEDA) **REFERENCES** ISTRUTTORE(CF\_ISTRUTTORE) **ON** **DELETE** **CASCADE**
67. );
69. **CREATE** **TABLE** CONTIENE
70. (
71. N\_SCHEDA\_FK             **INT**,
72. NOME\_ESERCIZIO          **VARCHAR**(25),
73. N\_SERIE                 **INT**             **CHECK**(N\_SERIE <= 5 AND N\_SERIE > 0),
74. N\_RIPETIZIONI           **INT**             **CHECK**(N\_RIPETIZIONI <= 15 AND N\_RIPETIZIONI > 0),
75. **FOREIGN** **KEY**(N\_SCHEDA\_FK) **REFERENCES** SCHEDA\_ESERCIZI(N\_SCHEDA) **ON** **DELETE** **CASCADE**,
76. **FOREIGN** **KEY**(NOME\_ESERCIZIO) **REFERENCES** ESERCIZIO(NOME) **ON** **DELETE** **CASCADE**
77. );
79. **CREATE** **TABLE** E\_ASSEGNATA
80. (
81. CF\_ISCRITTO\_FK          **CHAR**(16),
82. N\_SCHEDA\_FK             **INT**,
83. DATA\_INIZIO             **DATE**            NOT NULL,
84. DATA\_FINE               **DATE**            NOT NULL,
85. **PRIMARY** **KEY**(CF\_ISCRITTO\_FK,N\_SCHEDA\_FK),
86. **FOREIGN** **KEY**(CF\_ISCRITTO\_FK) **REFERENCES** ISCRITTO(CF\_ISCRITTO) **ON** **DELETE** **CASCADE**,
87. **FOREIGN** **KEY**(N\_SCHEDA\_FK) **REFERENCES** SCHEDA\_ESERCIZI(N\_SCHEDA) **ON** **DELETE** **CASCADE**
88. );
90. **CREATE** **TABLE** TURNO
91. (
92. CF\_ISTRUTTORE\_TURNO     **CHAR**(16)        NOT NULL,
93. CAP\_TURNO               **CHAR**(5),
94. DATA\_TURNO              **DATE**,
95. ORA\_TURNO               **CHAR**(2),
96. ORA\_OUT\_TURNO           **CHAR**(2)         NOT NULL,
97. **PRIMARY** **KEY**(DATA\_TURNO,ORA\_TURNO,CAP\_TURNO),
98. **FOREIGN** **KEY**(CF\_ISTRUTTORE\_TURNO) **REFERENCES** ISTRUTTORE(CF\_ISTRUTTORE) **ON** **DELETE** **CASCADE**,
99. **FOREIGN** **KEY**(CAP\_TURNO) **REFERENCES** STRUTTURA(CAP) **ON** **DELETE** **CASCADE**
100. );
102. **CREATE** **TABLE** EVENTO
103. (
104. NOME\_EVENTO             **VARCHAR**(25)     **PRIMARY** **KEY**,
105. CAP\_EVENTO              **CHAR**(5)         NOT NULL,
106. DATA                    **DATE**            NOT NULL,
107. **FOREIGN** **KEY**(CAP\_EVENTO) **REFERENCES** STRUTTURA(CAP) **ON** **DELETE** **CASCADE**
108. );
110. **CREATE** **TABLE** PARTECIPA
111. (
112. CF\_ISCRITTO\_PARTECIPA   **CHAR**(16),
113. NOME\_PARTECIPA          **VARCHAR**(25),
114. **PRIMARY** **KEY**(CF\_ISCRITTO\_PARTECIPA,NOME\_PARTECIPA),
115. **FOREIGN** **KEY**(CF\_ISCRITTO\_PARTECIPA) **REFERENCES** ISCRITTO(CF\_ISCRITTO) **ON** **DELETE** **CASCADE** ,
116. **FOREIGN** **KEY**(NOME\_PARTECIPA) **REFERENCES** EVENTO(NOME\_EVENTO) **ON** **DELETE** **CASCADE**
117. );
119. **CREATE** **TABLE** ABBONAMENTO
120. (
121. BADGE                   **CHAR**(4)         **PRIMARY** **KEY**,
122. CF\_ISCRITTO\_ABBONAMENTO **CHAR**(16)        NOT NULL,
123. DATA\_INIZIO             **DATE**            NOT NULL,
124. DATA\_SCADENZA           **DATE**            NOT NULL,
125. **FOREIGN** **KEY**(CF\_ISCRITTO\_ABBONAMENTO) **REFERENCES** ISCRITTO(CF\_ISCRITTO) **ON** **DELETE** **CASCADE**
126. );
128. **CREATE** **TABLE** CORSO
129. (
130. CF\_ISTRUTTORE\_CORSO     **CHAR**(16)        NOT NULL,
131. CAP\_CORSO               **CHAR**(5) ,
132. NOME\_CORSO              **VARCHAR**(20),
133. DATA\_INIZIO\_CORSO       **DATE**            NOT NULL,
134. DATA\_FINE\_CORSO         **DATE**            NOT NULL,
135. ORA\_INIZIO\_CORSO        **CHAR**(2)         NOT NULL,
136. **PRIMARY** **KEY**(NOME\_CORSO,CAP\_CORSO),
137. **FOREIGN** **KEY**(CF\_ISTRUTTORE\_CORSO) **REFERENCES** ISTRUTTORE(CF\_ISTRUTTORE) **ON** **DELETE** **SET** NULL,
138. **FOREIGN** **KEY**(CAP\_CORSO) **REFERENCES** STRUTTURA(CAP) **ON** **DELETE** **CASCADE**
139. );
141. **CREATE** **TABLE** PREVEDE
142. (
143. BADGE\_FK                **CHAR**(4),
144. CAP\_CORSO\_FK            **CHAR**(5),
145. NOME\_CORSO\_FK           **VARCHAR**(20),
146. **PRIMARY** **KEY**(BADGE\_FK,NOME\_CORSO\_FK,CAP\_CORSO\_FK),
147. **FOREIGN** **KEY**(BADGE\_FK) **REFERENCES** ABBONAMENTO(BADGE) **ON** **DELETE** **CASCADE**,
148. **FOREIGN** **KEY**(NOME\_CORSO\_FK,CAP\_CORSO\_FK) **REFERENCES** CORSO(NOME\_CORSO,CAP\_CORSO) **ON** **DELETE** **CASCADE**
149. );
151. **CREATE** **TABLE** LEZIONE
152. (
153. CAP\_LEZIONE             **CHAR**(5)         NOT NULL,
154. NOME\_LEZIONE            **VARCHAR**(20)     NOT NULL,
155. DATA\_LEZIONE            **DATE**            NOT NULL,
156. ORA\_INIZIO\_LEZIONE      **CHAR**(2)         NOT NULL,
157. ORA\_FINE\_LEZIONE        **CHAR**(2)         NOT NULL,
158. N\_STANZA                **INT**             NOT NULL,
159. **PRIMARY** **KEY**(DATA\_LEZIONE,ORA\_INIZIO\_LEZIONE,ORA\_FINE\_LEZIONE,NOME\_LEZIONE,CAP\_LEZIONE),
160. **FOREIGN** **KEY**(NOME\_LEZIONE,CAP\_LEZIONE) **REFERENCES** CORSO(NOME\_CORSO,CAP\_CORSO) **ON** **DELETE** **CASCADE**
161. );
163. **CREATE** **TABLE** GIORNI
164. (
165. GIORNO\_CORSO            **VARCHAR**(10),
166. CAP\_STRUTTURA\_FK        **CHAR**(5),
167. NOME\_CORSO\_GIORNO       **VARCHAR**(20),
168. **PRIMARY** **KEY**(GIORNO\_CORSO,NOME\_CORSO\_GIORNO,CAP\_STRUTTURA\_FK),
169. **FOREIGN** **KEY**(NOME\_CORSO\_GIORNO,CAP\_STRUTTURA\_FK) **REFERENCES** CORSO(NOME\_CORSO,CAP\_CORSO) **ON** **DELETE** **CASCADE**
170. );

## **DML: Popolamento**

Lo script popolamento completo contiene oltre 1500 righe di Inserimento, portarle tutte nella relazione comporterebbe uno spreco di spazio e disordine generale agli occhi di chi la legge: saranno quindi mostrate per ogni tabella solo alcuni esempi di inserimento.

1. **ALTER** SESSION **SET** NLS\_DATE\_FORMAT = 'DD/MM/YYYY';

-- POPOLAMENTO STRUTTURA

2. **INSERT** **INTO** STRUTTURA **VALUES** ('80144',300,'0817565543','VIA NAPOLI 64','NAPOLI');
3. **INSERT** **INTO** STRUTTURA **VALUES** ('80053',300,'0811955543','VIA CAMPANIA 63','ERCOLANO');
4. **INSERT** **INTO** STRUTTURA **VALUES** ('20156',300,'0211955543','VIA DEGLI ALIANTI 23','MILANO');
5. **INSERT** **INTO** STRUTTURA **VALUES** ('20138',300,'0233955543','VIA EUGENIO BELLOSIO 10','MILANO');
6. **INSERT** **INTO** STRUTTURA **VALUES** ('70100',300,'0883955543','VIA VITTORIO VENETO 47','BARI');
7. **INSERT** **INTO** STRUTTURA **VALUES** ('10121',300,'0113955543','VIA VITTORIO EMANUELE 35','TORINO');
8. **INSERT** **INTO** STRUTTURA **VALUES** ('80010',400,'0519555433','VIA MARITTIMA 89','BOLOGNA');
9. **INSERT** **INTO** STRUTTURA **VALUES** ('00010',400,'0653955543','VIA DEL CORSO 12','ROMA');
10. **INSERT** **INTO** STRUTTURA **VALUES** ('81010',300,'0813955543','VIA MONTEBELLO 55','CASERTA');

-- POPOLAMENTO ISCRITTO

2. **INSERT** **INTO** ISCRITTO **VALUES** ('KMZNIW02J50D385L','3188493717','ANTONIO','ESPOSITO','M','17/10/1990');
3. **INSERT** **INTO** ISCRITTO **VALUES** ('ZGACTV06L44M056S','3774420139','LUCA','ANTONINI','M','23/11/1987');
4. **INSERT** **INTO** ISCRITTO **VALUES** ('KQXBSW82D68L433D','3594879773','ARISA','COPPETTI','F','31/03/1986');
5. **INSERT** **INTO** ISCRITTO **VALUES** ('CJKSDG43J50B477X','3305263953','MASSIMO','FRAGOLA','M','23/03/1946');
6. **INSERT** **INTO** ISCRITTO **VALUES** ('JAWVMH55A14O606A','3078973964','GENNARO','TRAINO','M','24/08/1995');
7. **INSERT** **INTO** ISCRITTO **VALUES** ('MVAKFZ55S29A648X','3354350007','PIERLUIGI','SCOTTO','M','15/07/1950');
8. **INSERT** **INTO** ISCRITTO **VALUES** ('UJHVEW14A98S577B','3191845188','SIMONA','CARPENITO','F','25/08/1987');
9. **INSERT** **INTO** ISCRITTO **VALUES** ('WUXHQK92Q44L711M','3268602599','FEDERICA','ISOLA','F','12/02/1950');
10. **INSERT** **INTO** ISCRITTO **VALUES** ('OAWILV32Y85R021K','3358969430','LOREDANA','CARAFFA','F','01/06/1969');
11. **INSERT** **INTO** ISCRITTO **VALUES** ('QESWVR92G10B017F','3765052137','VITTORIO','CAPONE','M','05/11/1987');
12. **INSERT** **INTO** ISCRITTO **VALUES** ('ASCGWQ31R44Z342I','3542114257','ANNA','RICCIOLO','F','20/12/1991');
13. **INSERT** **INTO** ISCRITTO **VALUES** ('LNBCVX77B53X278C','3720498324','EUGENIA','CATTIVO','F','17/10/1965');
14. **INSERT** **INTO** ISCRITTO **VALUES** ('BEJMCI13W11F237L','3868820921','ALDO','BRUSCIANO','M','14/02/1961');
15. **INSERT** **INTO** ISCRITTO **VALUES** ('FZLBNR04B93Q545M','3125274245','DEBORA','ROSSI','F','10/05/1993');
16. **INSERT** **INTO** ISCRITTO **VALUES** ('ATDQPI45H28W807X','3464560455','MICHELE','ANGELILLO','M', '07/01/1969');
17. **INSERT** **INTO** ISCRITTO **VALUES** ('XQDZVK54Y91F149G','3515567709','CHIARA','COZZOLINO','F', '15/07/1943');
18. **INSERT** **INTO** ISCRITTO **VALUES** ('OLCDGA16P83E695I','3853512083','ANDREA','SCOTTO','M', '30/09/1947');

-- POPOLAMENTO ESERCIZIO

1. **INSERT** **INTO** ESERCIZIO **VALUES** ('PANCA PIANA','ALTA','PETTO');
2. **INSERT** **INTO** ESERCIZIO **VALUES** ('PANCA ALTA','ALTA','PETTO');
3. **INSERT** **INTO** ESERCIZIO **VALUES** ('PECTORAL MACHINE','ALTA','PETTO');
4. **INSERT** **INTO** ESERCIZIO **VALUES** ('REMATORI MACCHINA','ALTA','DORSO');
5. **INSERT** **INTO** ESERCIZIO **VALUES** ('CURL MANUBRI','ALTA','BRACCIA');
6. **INSERT** **INTO** ESERCIZIO **VALUES** ('CURL BILANCIERE','ALTA','BRACCIA');
7. **INSERT** **INTO** ESERCIZIO **VALUES** ('CYCLETTE MACHINE','BASSA','GAMBE');
8. **INSERT** **INTO** ESERCIZIO **VALUES** ('ROW MACHINE','MEDIA','DORSO');
9. **INSERT** **INTO** ESERCIZIO **VALUES** ('REMATORI CORDA','MEDIA','DORSO');
10. **INSERT** **INTO** ESERCIZIO **VALUES** ('CRUNCH','MEDIA','ADDOME');
11. **INSERT** **INTO** ESERCIZIO **VALUES** ('CURL PESO LEGGERO','BASSA','BRACCIA');
12. **INSERT** **INTO** ESERCIZIO **VALUES** ('SQUAT LIBERO','BASSA','GAMBE');
13. **INSERT** **INTO** ESERCIZIO **VALUES** ('AFFONDI LIBERI','BASSA','GAMBE');

-- POPOLAMENTO ISTRUTTORE

2. **INSERT** **INTO** ISTRUTTORE **VALUES** ('DOKWIA13R15P994B', '3080774312', 'ANTONIO', 'BERARDI', 'M', '20/12/1982');
3. **INSERT** **INTO** ISTRUTTORE **VALUES** ('JHKSGQ83S05D455J', NULL, 'ROMEO', 'LAVEGNO', 'F', '13/09/1945');
4. **INSERT** **INTO** ISTRUTTORE **VALUES** ('XPKQMJ90D67H047X', '3389991455', 'CRISTINA', 'DIEGO', 'F', '30/04/1977');
5. **INSERT** **INTO** ISTRUTTORE **VALUES** ('HAVEKN72V54A537M', '3086078330', 'LINDA', 'FELICE', 'F', '30/07/1979');
6. **INSERT** **INTO** ISTRUTTORE **VALUES** ('RWMULV46D82C327V', NULL, 'SIMONA', 'GRAZIATA', 'F', '28/05/1999');
7. **INSERT** **INTO** ISTRUTTORE **VALUES** ('YNAVDC04T09B897H', '3646716201', 'CARLA', 'ESPOSITO', 'F', '10/05/1946');
8. **INSERT** **INTO** ISTRUTTORE **VALUES** ('TUMLPD88X61G115W', '3010521160', 'ELENA', 'CORRADO', 'F', '08/08/1997');
9. **INSERT** **INTO** ISTRUTTORE **VALUES** ('AKDZCT42H20V702G', '3157122092', 'MARCO', 'PRINCI', 'M', '09/02/1967');
10. **INSERT** **INTO** ISTRUTTORE **VALUES** ('NIDGYC56V10P953I', '3023359662', 'RENATA', 'CASO', 'F', '27/11/1983');
11. **INSERT** **INTO** ISTRUTTORE **VALUES** ('KDPFAO60P32T846I', '3618563068', 'SARA', 'CARE', 'F', '12/10/1977');
12. **INSERT** **INTO** ISTRUTTORE **VALUES** ('GQVRLB24U11X980D', '3498753667', 'MARIA', 'VIOLA', 'F', '09/09/1994');
13. **INSERT** **INTO** ISTRUTTORE **VALUES** ('CJKZGU71R40Z530T', '3993470505', 'GIANLUCA', 'PINZO', 'M', '03/01/1988');

--POPOLAMENTO SCHEDA ESERCIZI

2. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','1','ALTO');
3. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('JGANDL90S24C290L','2','BASSO');
4. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('HGDREQ86Z85D099A','3','MEDIO');
5. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','4','ALTO');
6. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','5','ALTO');
7. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','6','BASSO');
8. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','7','ALTO');
9. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('JGANDL90S24C290L','8','BASSO');
10. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('DOKWIA13R15P994B','9','MEDIO');
11. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('JGANDL90S24C290L','10','BASSO');
12. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('XPKQMJ90D67H047X','11','ALTO');
13. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('XPKQMJ90D67H047X','12','BASSO');
14. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('KYFLAM82X62J244F','13','ALTO');
15. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES** ('KYFLAM82X62J244F','14','MEDIO');

--POPOLAMENTO CONTIENE

2. **INSERT** **INTO** CONTIENE **VALUES** ('1','PANCA PIANA',3,10);
3. **INSERT** **INTO** CONTIENE **VALUES** ('1','REMATORI MACCHINA',3,10);
4. **INSERT** **INTO** CONTIENE **VALUES** ('1','CRUNCH CON PESO',4,10);
5. **INSERT** **INTO** CONTIENE **VALUES** ('1','SQUAT BILANCIERE',4,8);
6. **INSERT** **INTO** CONTIENE **VALUES** ('1','CURL MANUBRI',3,10);
7. **INSERT** **INTO** CONTIENE **VALUES** ('2','SPINTE MACCHINA',3,10);
8. **INSERT** **INTO** CONTIENE **VALUES** ('2','AFFONDI LIBERI',3,10);
9. **INSERT** **INTO** CONTIENE **VALUES** ('2','ABDOMINAL MACHINE',3,10);
10. **INSERT** **INTO** CONTIENE **VALUES** ('2','CUFF MACHINE',3,10);
11. **INSERT** **INTO** CONTIENE **VALUES** ('2','CURL PESO LEGGERO',3,8);
12. **INSERT** **INTO** CONTIENE **VALUES** ('2','LAT MACHINE LEGGERA',3,10);
13. **INSERT** **INTO** CONTIENE **VALUES** ('3','ALZATE LATERALI MACCHINA',3,10);
14. **INSERT** **INTO** CONTIENE **VALUES** ('3','CURL MACCHINA',3,10);
15. **INSERT** **INTO** CONTIENE **VALUES** ('3','CROCI MACCHINA',4,8);
16. **INSERT** **INTO** CONTIENE **VALUES** ('3','AFFONDI MANUBRI',4,10);

-- POPOLAMENTO E\_ASSEGNATA

2. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('KMZNIW02J50D385L', '1', '01/07/2018','01/09/2018');
3. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('ZGACTV06L44M056S', '2', '02/07/2018', '02/09/2018');
4. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('KQXBSW82D68L433D', '3', '03/07/2018', '03/09/2018');
5. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('CJKSDG43J50B477X', '4', '04/07/2018', '04/09/2018');
6. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('JAWVMH55A14O606A', '5', '05/07/2018', '05/09/2018');
7. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('MVAKFZ55S29A648X', '6', '06/07/2018', '06/09/2018');
8. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('UJHVEW14A98S577B', '7', '07/07/2018', '07/09/2018');
9. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('WUXHQK92Q44L711M', '8', '08/07/2018', '08/09/2018');
10. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('OAWILV32Y85R021K', '9', '09/07/2018', '09/09/2018');
11. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('QESWVR92G10B017F', '10', '10/07/2018', '10/09/2018');
12. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('ASCGWQ31R44Z342I', '11', '11/07/2018', '11/09/2018');
13. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('LNBCVX77B53X278C', '12', '12/07/2018', '12/09/2018');
14. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('BEJMCI13W11F237L', '13', '13/07/2018', '13/09/2018');
15. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('FZLBNR04B93Q545M', '14', '14/07/2018', '14/09/2018');
16. **INSERT** **INTO** E\_ASSEGNATA **VALUES** ('ATDQPI45H28W807X', '15', '15/07/2018', '15/09/2018');

--POPOLAMENTO TURNO

2. **INSERT** **INTO** TURNO **VALUES** ('DOKWIA13R15P994B','80144','15/07/2018','08','15');
3. **INSERT** **INTO** TURNO **VALUES** ('DOKWIA13R15P994B','80144','16/07/2018','08','15');
4. **INSERT** **INTO** TURNO **VALUES** ('DOKWIA13R15P994B','80144','17/07/2018','08','15');
5. **INSERT** **INTO** TURNO **VALUES** ('DOKWIA13R15P994B','80144','18/07/2018','08','15');
6. **INSERT** **INTO** TURNO **VALUES** ('DNZXOC14C04L989E','80053','15/07/2018','08','15');
7. **INSERT** **INTO** TURNO **VALUES** ('DNZXOC14C04L989E','80053','16/07/2018','08','15');
8. **INSERT** **INTO** TURNO **VALUES** ('DNZXOC14C04L989E','80053','17/07/2018','08','15');
9. **INSERT** **INTO** TURNO **VALUES** ('DNZXOC14C04L989E','80053','18/07/2018','08','15');
10. **INSERT** **INTO** TURNO **VALUES** ('EAPWRB22P77L607X','20138','18/07/2018','15','22');
11. **INSERT** **INTO** TURNO **VALUES** ('KDPFAO60P32T846I','20138','19/07/2018','08','15');
12. **INSERT** **INTO** TURNO **VALUES** ('KDPFAO60P32T846I','20138','20/07/2018','08','15');
13. **INSERT** **INTO** TURNO **VALUES** ('KYFLAM82X62J244F','70100','20/07/2018','08','15');
14. **INSERT** **INTO** TURNO **VALUES** ('KYFLAM82X62J244F','70100','21/07/2018','08','15');
15. **INSERT** **INTO** TURNO **VALUES** ('KYFLAM82X62J244F','70100','22/07/2018','08','15');

--POPOLAMENTO EVENTO

2. **INSERT** **INTO** EVENTO **VALUES** ('I GIORNATA FITNESS','80144','02/07/2018');
3. **INSERT** **INTO** EVENTO **VALUES** ('II GIORNATA FITNESS','80053','15/07/2018');
4. **INSERT** **INTO** EVENTO **VALUES** ('III GIORNATA FITNESS','20156','20/07/2018');
5. **INSERT** **INTO** EVENTO **VALUES** ('IV GIORNATA FITNESS','20138','29/07/2018');
6. **INSERT** **INTO** EVENTO **VALUES** ('V GIORNATA FITNESS','70100','05/08/2018');
7. **INSERT** **INTO** EVENTO **VALUES** ('VI GIORNATA FITNESS','10121','01/09/2018');
8. **INSERT** **INTO** EVENTO **VALUES** ('I GARA KARATE','80010','12/09/2018');
9. **INSERT** **INTO** EVENTO **VALUES** ('II GARA KARATE','00010','19/09/2018');
10. **INSERT** **INTO** EVENTO **VALUES** ('III GARA KARATE','81010','26/09/2018');
11. **INSERT** **INTO** EVENTO **VALUES** ('IV GARA KARATE','80144','04/10/2018');
12. **INSERT** **INTO** EVENTO **VALUES** ('V GARA KARATE','80053','09/10/2018');
13. **INSERT** **INTO** EVENTO **VALUES** ('VI GARA KARATE','20156','17/10/2018');
14. **INSERT** **INTO** EVENTO **VALUES** ('I GARA FIT BOXE','20138','20/10/2018');
15. **INSERT** **INTO** EVENTO **VALUES** ('II GARA FIT BOXE','70100','25/10/2018');
16. **INSERT** **INTO** EVENTO **VALUES** ('III GARA FIT BOXE','10121','29/10/2018');
17. **INSERT** **INTO** EVENTO **VALUES** ('IV GARA FIT BOXE','80010','01/11/2018');
18. **INSERT** **INTO** EVENTO **VALUES** ('V GARA FIT BOXE','00010','03/11/2018');

--POPOLAMENTO PARTECIPA

2. **INSERT** **INTO** PARTECIPA **VALUES** ('KMZNIW02J50D385L', 'I GIORNATA FITNESS');
3. **INSERT** **INTO** PARTECIPA **VALUES** ('ZGACTV06L44M056S', 'III GARA BODYBULDING');
4. **INSERT** **INTO** PARTECIPA **VALUES** ('KQXBSW82D68L433D', 'V GARA KARATE');
5. **INSERT** **INTO** PARTECIPA **VALUES** ('CJKSDG43J50B477X', 'III TROFEO NIKE');
6. **INSERT** **INTO** PARTECIPA **VALUES** ('JAWVMH55A14O606A', 'II TROFEO NIKE');
7. **INSERT** **INTO** PARTECIPA **VALUES** ('MVAKFZ55S29A648X', 'I TROFEO ADIDAS');
8. **INSERT** **INTO** PARTECIPA **VALUES** ('UJHVEW14A98S577B', 'IV GARA FIT BOXE');
9. **INSERT** **INTO** PARTECIPA **VALUES** ('WUXHQK92Q44L711M', 'V GIORNATA FITNESS');
10. **INSERT** **INTO** PARTECIPA **VALUES** ('OAWILV32Y85R021K', 'VI GARA BODYBULDING');
11. **INSERT** **INTO** PARTECIPA **VALUES** ('QESWVR92G10B017F', 'V TROFEO ADIDAS');
12. **INSERT** **INTO** PARTECIPA **VALUES** ('ASCGWQ31R44Z342I', 'II TROFEO ADIDAS');
13. **INSERT** **INTO** PARTECIPA **VALUES** ('LNBCVX77B53X278C', 'VI TROFEO ADIDAS');
14. **INSERT** **INTO** PARTECIPA **VALUES** ('BEJMCI13W11F237L', 'I GIORNATA FITNESS');
15. **INSERT** **INTO** PARTECIPA **VALUES** ('FZLBNR04B93Q545M', 'II GIORNATA FITNESS');
16. **INSERT** **INTO** PARTECIPA **VALUES** ('ATDQPI45H28W807X', 'III GIORNATA FITNESS');
17. **INSERT** **INTO** PARTECIPA **VALUES** ('XQDZVK54Y91F149G', 'IV GIORNATA FITNESS');
18. **INSERT** **INTO** PARTECIPA **VALUES** ('OLCDGA16P83E695I', 'II GARA FIT BOXE');

--POPOLAMENTO ABBONAMENTO

2. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0001','KMZNIW02J50D385L','01/07/2018','01/07/2019');
3. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0002','ZGACTV06L44M056S','02/07/2018','02/07/2019');
4. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0003','KQXBSW82D68L433D','03/07/2018','03/07/2019');
5. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0004','CJKSDG43J50B477X','04/07/2018','04/07/2019');
6. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0005','JAWVMH55A14O606A','05/07/2018','05/07/2019');
7. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0006','MVAKFZ55S29A648X','06/07/2018','06/07/2019');
8. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0007','UJHVEW14A98S577B','07/07/2018','07/07/2019');
9. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0008','WUXHQK92Q44L711M','08/07/2018','08/07/2019');
10. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0009','OAWILV32Y85R021K','09/07/2018','09/07/2019');
11. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0010','QESWVR92G10B017F','10/07/2018','10/07/2019');
12. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0011','ASCGWQ31R44Z342I','11/07/2018','11/07/2019');
13. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0012','LNBCVX77B53X278C','12/07/2018','12/07/2019');
14. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0013','BEJMCI13W11F237L','13/07/2018','13/07/2019');
15. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0014','FZLBNR04B93Q545M','14/07/2018','14/07/2019');
16. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0015','ATDQPI45H28W807X','15/07/2018','15/07/2019');
17. **INSERT** **INTO** ABBONAMENTO **VALUES** ('0016','XQDZVK54Y91F149G','16/07/2018','16/07/2019');

--POPOLAMENTO CORSO

2. **INSERT** **INTO** CORSO **VALUES** ('DOKWIA13R15P994B','80144','AEROBICA','16/07/2018','15/07/2019','08');
3. **INSERT** **INTO** CORSO **VALUES** ('DOKWIA13R15P994B','80144','PILATES','19/07/2018','15/07/2019','10');
4. **INSERT** **INTO** CORSO **VALUES** ('JHKSGQ83S05D455J','80144','ZUMBA','18/07/2018','15/07/2019','16');
5. **INSERT** **INTO** CORSO **VALUES** ('JHKSGQ83S05D455J','80144','KARATE','17/07/2018','15/07/2019','18');
6. **INSERT** **INTO** CORSO **VALUES** ('XPKQMJ90D67H047X','80144','TOTAL BODY','17/07/2018','15/07/2019','08');
7. **INSERT** **INTO** CORSO **VALUES** ('XPKQMJ90D67H047X','80144','YOGA','18/07/2018','15/07/2019','10');
8. **INSERT** **INTO** CORSO **VALUES** ('HAVEKN72V54A537M','80144','CROSSFIT','16/07/2018','15/07/2019','16');
9. **INSERT** **INTO** CORSO **VALUES** ('HAVEKN72V54A537M','80144','FIT BOXE','16/07/2018','15/07/2019','18');
10. **INSERT** **INTO** CORSO **VALUES** ('DNZXOC14C04L989E','80053','AEROBICA','16/07/2018','15/07/2019','08');
11. **INSERT** **INTO** CORSO **VALUES** ('DNZXOC14C04L989E','80053','PILATES','19/07/2018','15/07/2019','10');
12. **INSERT** **INTO** CORSO **VALUES** ('RWMULV46D82C327V','80053','ZUMBA','18/07/2018','15/07/2019','16');
13. **INSERT** **INTO** CORSO **VALUES** ('RWMULV46D82C327V','80053','FIT BOXE','16/07/2018','15/07/2019','18');
14. **INSERT** **INTO** CORSO **VALUES** ('APVJKM81T10J191V','80053','KARATE','17/07/2018','15/07/2019','18');
15. **INSERT** **INTO** CORSO **VALUES** ('APVJKM81T10J191V','80053','TOTAL BODY','17/07/2018','15/07/2019','10');
16. **INSERT** **INTO** CORSO **VALUES** ('RWMULV46D82C327V','80053','YOGA','18/07/2018','15/07/2019','16');
17. **INSERT** **INTO** CORSO **VALUES** ('RWMULV46D82C327V','80053','CROSSFIT','16/07/2018','15/07/2019','18');
18. **INSERT** **INTO** CORSO **VALUES** ('YNAVDC04T09B897H','20156','AEROBICA','16/07/2018','15/07/2019','08');

-- POPOLAMENTO PREVEDE

2. **INSERT** **INTO** PREVEDE **VALUES** ('0001','80144','AEROBICA');
3. **INSERT** **INTO** PREVEDE **VALUES** ('0001','80144','FIT BOXE');
4. **INSERT** **INTO** PREVEDE **VALUES** ('0002','80144','FIT BOXE');
5. **INSERT** **INTO** PREVEDE **VALUES** ('0003','80144','YOGA');
6. **INSERT** **INTO** PREVEDE **VALUES** ('0003','80144','CROSSFIT');
7. **INSERT** **INTO** PREVEDE **VALUES** ('0004','80053','AEROBICA');
8. **INSERT** **INTO** PREVEDE **VALUES** ('0005','80053','ZUMBA');
9. **INSERT** **INTO** PREVEDE **VALUES** ('0005','80053','TOTAL BODY');
10. **INSERT** **INTO** PREVEDE **VALUES** ('0006','80053','FIT BOXE');
11. **INSERT** **INTO** PREVEDE **VALUES** ('0006','80053','CROSSFIT');
12. **INSERT** **INTO** PREVEDE **VALUES** ('0007','20156','AEROBICA');
13. **INSERT** **INTO** PREVEDE **VALUES** ('0008','20156','ZUMBA');
14. **INSERT** **INTO** PREVEDE **VALUES** ('0009','20156','FIT BOXE');
15. **INSERT** **INTO** PREVEDE **VALUES** ('0009','20156','KARATE');
16. **INSERT** **INTO** PREVEDE **VALUES** ('0010','20156','TOTAL BODY');
17. **INSERT** **INTO** PREVEDE **VALUES** ('0011','20156','YOGA');

--POPOLAMENTO LEZIONE

2. **INSERT** **INTO** LEZIONE **VALUES** ('80144','AEROBICA','16/07/2018','08','10',1);
3. **INSERT** **INTO** LEZIONE **VALUES** ('80144','AEROBICA','18/07/2018','08','10',1);
4. **INSERT** **INTO** LEZIONE **VALUES** ('80144','AEROBICA','20/07/2018','08','10',1);
5. **INSERT** **INTO** LEZIONE **VALUES** ('80144','PILATES','19/07/2018','10','12',3);
6. **INSERT** **INTO** LEZIONE **VALUES** ('80144','FIT BOXE','16/07/2018','18','20',1);
7. **INSERT** **INTO** LEZIONE **VALUES** ('80144','FIT BOXE','18/07/2018','18','20',1);
8. **INSERT** **INTO** LEZIONE **VALUES** ('80144','FIT BOXE','20/07/2018','18','20',1);
9. **INSERT** **INTO** LEZIONE **VALUES** ('80144','ZUMBA','18/07/2018','16','18',2);
10. **INSERT** **INTO** LEZIONE **VALUES** ('80144','KARATE','17/07/2018','18','20',3);
11. **INSERT** **INTO** LEZIONE **VALUES** ('80144','KARATE','19/07/2018','18','20',3);
12. **INSERT** **INTO** LEZIONE **VALUES** ('80144','TOTAL BODY','17/07/2018','08','10',1);
13. **INSERT** **INTO** LEZIONE **VALUES** ('80144','TOTAL BODY','19/07/2018','08','10',1);
14. **INSERT** **INTO** LEZIONE **VALUES** ('80144','YOGA','18/07/2018','10','12',2);
15. **INSERT** **INTO** LEZIONE **VALUES** ('80144','CROSSFIT','16/07/2018','16','18',2);
16. **INSERT** **INTO** LEZIONE **VALUES** ('80144','CROSSFIT','19/07/2018','16','18',2);

-- POPOLAMENTO GIORNI

2. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','80144','AEROBICA');
3. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','80144','AEROBICA');
4. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','80144','AEROBICA');
5. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','80053','AEROBICA');
6. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','80053','AEROBICA');
7. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','80053','AEROBICA');
8. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','20156','AEROBICA');
9. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','20156','AEROBICA');
10. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','20156','AEROBICA');
11. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','20138','AEROBICA');
12. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','20138','AEROBICA');
13. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','20138','AEROBICA');
14. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','70100','AEROBICA');
15. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','70100','AEROBICA');
16. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','70100','AEROBICA');
17. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','10121','AEROBICA');
18. **INSERT** **INTO** GIORNI **VALUES**  ('MERCOLEDI','10121','AEROBICA');
19. **INSERT** **INTO** GIORNI **VALUES**  ('VENERDI','10121','AEROBICA');
20. **INSERT** **INTO** GIORNI **VALUES**  ('LUNEDI','80010','AEROBICA');

## **Trigger**

Di seguito verranno mostrati i codici dei **Trigger** implementati per rappresentare i vincoli dinamici precedenti.

### **Trigger Scheda Esercizi**

Il seguente trigger controlla, prima di un inserimento sulla tabella **CONTIENE**, il numero di **esercizi** presenti nella scheda in cui si sta inserendo che non sia maggiore di 6. Se sono maggiori allora lancia un errore.

1. --TRIGGER PER CONTROLLARE CHE IL NUMERO DEGLI ESERCIZI IN UNA SCHEDA NON SIA MAGGIORE DI 6
2. **CREATE** OR REPLACE **TRIGGER** CONTROLLO\_ESERCIZI\_SCHEDA
3. BEFORE **INSERT** **ON** CONTIENE
4. **FOR** EACH ROW
5. **DECLARE**
6. NUM\_ES NUMBER;
7. TROPPI\_ESERCIZI EXCEPTION;
9. **BEGIN**
11. **SELECT** COUNT(\*) **INTO** NUM\_ES
12. **FROM** CONTIENE
13. **WHERE** N\_SCHEDA\_FK = :NEW.N\_SCHEDA\_FK;
15. IF (NUM\_ES > 5) **THEN** RAISE TROPPI\_ESERCIZI;
16. **END** IF;
18. EXCEPTION
19. **WHEN** TROPPI\_ESERCIZI **THEN**
20. RAISE\_APPLICATION\_ERROR(-20530,'TROPPI ESERCIZI IN QUESTA SCHEDA');
21. **END**;
22. /

### **Trigger Auto-Id Scheda**

Il trigger usa una sequenza chiamata “**seq\_n\_scheda**” per generare di volta in volta il nuovo **id scheda** e lo inserirà ogni volta che un operazione di insert sulla tabella **SCHEDA\_ESERCIZI** avrà il valore **N\_SCHEDA NULL**.

1. --SEQUENCE PER INSERIMENTO AUTOMATICO NUMERO SCHEDA
2. **DROP** **SEQUENCE** SEQ\_N\_SCHEDA;
4. **CREATE** **SEQUENCE** SEQ\_N\_SCHEDA
5. START **WITH** 1
6. INCREMENT **BY** 1
7. NOMAXVALUE
8. NOCYCLE;
10. --INSERIMENTO AUTOMATICO NUMERO SCHEDA
11. **CREATE** OR REPLACE **TRIGGER** N\_SCHEDA
12. BEFORE **INSERT** **ON** SCHEDA\_ESERCIZI
13. **FOR** EACH ROW
14. **WHEN** (NEW.N\_SCHEDA **IS** NULL)
15. --------------
16. **DECLARE**
17. N\_S NUMBER := SEQ\_N\_SCHEDA.NEXTVAL;
18. **BEGIN**
19. :NEW.N\_SCHEDA := N\_S;
20. **END**;
21. /

### **Trigger Controllo Turni**

Il trigger parte prima dell’inserimento sulla tabella **TURNO**, verifica che i turni di un istruttore in un giorno non siano due: in caso affermativo lancia un errore quando si sta inserendo un 3° turno. Verifica anche che i turni in un **mese** non siano più di **27**, in caso affermativo lancia un errore quando si sta inserendo il 28° turno.

1. --TRIGGER CONTROLLO TURNI DI UN ISTRUTTORE
2. **CREATE** OR REPLACE **TRIGGER** CONTROLLO\_TURNO
3. BEFORE **INSERT** **ON** TURNO
4. **FOR** EACH ROW
6. **DECLARE**
7. TROPPI\_TURNI EXCEPTION;
8. TURNI NUMBER;
9. TROPPI\_TURNI\_MESE EXCEPTION;
10. TURNI\_M NUMBER;
12. **BEGIN**
13. --VERIFICA TURNI DI UN ISTRUTTORE
14. **SELECT** COUNT(\*) **INTO** TURNI
15. **FROM** TURNO
16. **WHERE** DATA\_TURNO = :NEW.DATA\_TURNO;
18. --SE SVOLGE PIU' DI UN TURNO LANCIA UN ERRORE
19. IF (TURNI > 1) **THEN** RAISE TROPPI\_TURNI;
20. **END** IF;

23. --VERIFICA TURNI DI UN ISTRUTTORE IN UN MESE SOLARE
24. **SELECT** COUNT(\*) **INTO** TURNI\_M
25. **FROM** TURNO
26. **WHERE** TO\_CHAR(DATA\_TURNO,'MON-YEAR') = TO\_CHAR(:NEW.DATA\_TURNO,'MON-YEAR');
28. --SE SVOLGE PIU DI 27 TURNI AL MESE LANCIA ERRORE
29. IF (TURNI\_M > 26) **THEN** RAISE TROPPI\_TURNI\_MESE;
30. **END** IF;
32. EXCEPTION
33. **WHEN** TROPPI\_TURNI **THEN**
34. RAISE\_APPLICATION\_ERROR(-20456,'TROPPI TURNI PER QUESTO ISTRUTTORE');
35. **WHEN** TROPPI\_TURNI\_MESE **THEN**
36. RAISE\_APPLICATION\_ERROR(-20543,'QUESTO ISTRUTTORE HA FATTO GIA 27 TURNI QUESTO MESE');
37. **END**;
38. /

### **Trigger Auto-Badge Corso**

Il Trigger utilizza una sequenza per l’inserimento automatico di un codice **badge** nell’inserimento di abbonamento: se durante un inserimento sulla tabella **ABBONAMENTO** il valore **badge** sarà **NULL**, allora il trigger prenderà il numero badge dalla sequenza e lo convertirà in char. Si noti che il badge sarà diviso per 10000 e saranno prese le ultime 4 cifre decimale con SUBSTR per non perdere lo zero iniziale.

1. --SEQUENCE PER INSERIMENTO AUTOMATICO NUMERO SCHEDA
2. **DROP** **SEQUENCE** SEQ\_N\_BADGE;
4. **CREATE** **SEQUENCE** SEQ\_N\_BADGE
5. START **WITH** 1
6. INCREMENT **BY** 1
7. NOMAXVALUE
8. NOCYCLE;
10. --INSERIMENTO AUTOMATICO NUMERO BADGE
11. **CREATE** OR REPLACE **TRIGGER** N\_BADGE
12. BEFORE
13. **INSERT** **ON** ABBONAMENTO
14. **FOR** EACH ROW
15. **WHEN** (NEW.BADGE **IS** NULL)
16. --------------
17. **DECLARE**
18. N\_S NUMBER := SEQ\_N\_BADGE.NEXTVAL;
19. **BEGIN**
20. -- DIVIDO IL NUMERO PER 10000 E PRENDO SOLO LA PARTE DECIAMELE CONVERTITA IN CHAR PER NON PERDERE LO 0 INIZIALE!
21. N\_S := N\_S/10000;
23. :NEW.BADGE := SUBSTR(TO\_CHAR(N\_S,'9999.9999'),7,4);
24. **END**;
25. /

### **Trigger Inserimento Corso**

Il triggerparte prima di un inserimento nella tabella **CORSO**, controlla che un corso abbia una data iniziale maggiore della data corrente, che l’istruttore che tiene il corso non lavori a già 4 corsi, e che non ci siano più di 9 corsi attivi in una struttura. Se una di queste condizioni è violata si lancia un errore corrispondente.

1. -- TRIGGER PER CONTROLLARE L'INSERIMENTO DI UN CORSO
2. **CREATE** OR REPLACE **TRIGGER** INSERIMENTO\_CORSO
3. BEFORE **INSERT** **ON** CORSO
4. **FOR** EACH ROW
6. **DECLARE**
7. N\_CORSI\_ISTRUTTORE  NUMBER;
8. N\_CORSI\_STRUTTURA  NUMBER;
9. DATA\_INFERIORE  EXCEPTION;
10. TROPPO\_LAVORO  EXCEPTION;
11. TROPPI\_CORSI  EXCEPTION;

14. **BEGIN**
15. -- NON SI PUO' INSERIRE UN CORSO CHE ABBIA DATA INIZIALE MINORE DELLA DATA CORRENTE
16. IF(:NEW.DATA\_INIZIO\_CORSO < SYSDATE)
17. **THEN** RAISE DATA\_INFERIORE;
18. **END** IF;
20. -- VERIFICA QUANTI CORSI STA TENENDO L'ISTRUTTORE IN CONTEMPORANEA
21. -- SE LAVORA A PIU DI 4 CORSI ALLORA LANCIA UN ECCEZIONE
22. **SELECT** COUNT(\*) **INTO** N\_CORSI\_ISTRUTTORE
23. **FROM** ISTRUTTORE JOIN CORSO  **ON** CF\_ISTRUTTORE = CF\_ISTRUTTORE\_CORSO
24. **WHERE** CF\_ISTRUTTORE\_CORSO = :NEW.CF\_ISTRUTTORE\_CORSO
25. AND DATA\_FINE\_CORSO > SYSDATE;
27. IF(N\_CORSI\_ISTRUTTORE > 3)
28. **THEN** RAISE TROPPO\_LAVORO;
29. **END** IF;

32. -- VERIFICA QUANTI CORSI TIENE UNA STRUTTURA CONTEMPORANEAMENTE,
33. -- SE CI SONO PIU' DI 8 CORSI LANCIA UN ECCEZIONE
34. **SELECT** COUNT(\*) **INTO** N\_CORSI\_STRUTTURA
35. **FROM** CORSO JOIN STRUTTURA **ON** CAP\_CORSO = CAP
36. **WHERE** CAP\_CORSO = :NEW.CAP\_CORSO
37. AND DATA\_FINE\_CORSO > SYSDATE;
39. IF(N\_CORSI\_STRUTTURA > 8)
40. **THEN** RAISE TROPPI\_CORSI;
41. **END** IF;

44. EXCEPTION
45. **WHEN** DATA\_INFERIORE **THEN**
46. RAISE\_APPLICATION\_ERROR(-20505,'E'' STATA INSERITA UNA DATA INIZIO CORSO PRECEDENTE A QUELLA DI OGGI');
47. **WHEN** TROPPO\_LAVORO **THEN**
48. RAISE\_APPLICATION\_ERROR(-20506,'ISTRUTTORE STA SEGUENDO GIA 4 CORSI');
49. **WHEN** TROPPI\_CORSI **THEN**
50. RAISE\_APPLICATION\_ERROR(-20507,'TROPPI CORSI PER QUESTA STRUTTURA');
52. **END**;
53. /

### **Trigger Abbonamenti**

Il trigger parte prima di inserire sulla tabella prevede per l’assegnazione di un corso ad un abbonamento.  
Controlla che nell’abbonamento non vi siano corsi di sedi diverse e che non vi siano più di due corsi in un abbonamento.

1. **CREATE** OR REPLACE **TRIGGER** CONTROLLO\_ABBONAMENTO
2. BEFORE **INSERT** **ON** PREVEDE
3. **FOR** EACH ROW
5. **DECLARE**
6. CORSI\_SEDE\_DIV      NUMBER;
7. CORSO\_SEDE\_DIVERSA  EXCEPTION;
8. N\_CORSI NUMBER;
9. TROPPI\_CORSI EXCEPTION;
11. **BEGIN**
13. -- CONTROLLA CON UNA VARIABILE NUMERICA SE L’ABBONAMENTO HA GIA CORSI CON SEDE DIVERSA
14. **SELECT** COUNT(\*) **INTO** CORSI\_SEDE\_DIV
15. **FROM** PREVEDE
16. **WHERE** CAP\_CORSO\_FK <> :NEW.CAP\_CORSO\_FK;
18. IF(CORSI\_SEDE\_DIV > 0) **THEN** RAISE CORSO\_SEDE\_DIVERSA;
19. **END** IF;
21. -- VERIFICA QUANTI CORSI CI SONO IN UN ABBONAMENTO
22. -- SE PIU' DI 2 CORSI LANCIA ECCEZIONE
23. **SELECT** COUNT(\*) **INTO** N\_CORSI
24. **FROM** PREVEDE
25. **WHERE** BADGE\_FK = :NEW.BADGE\_FK;
27. IF(N\_CORSI > 1)
28. **THEN** RAISE TROPPI\_CORSI;
29. **END** IF;

32. EXCEPTION
34. **WHEN** CORSO\_SEDE\_DIVERSA **THEN**
35. RAISE\_APPLICATION\_ERROR(-20543,'IL CORSO HA UNA SEDE DIVERSA RISPETTO AGLI ALTRI GIA PRESENTI NELL ABBONAMENTO');
36. **WHEN** TROPPI\_CORSI **THEN**
37. RAISE\_APPLICATION\_ERROR(-21001, 'TROPPI CORSI PER QUESTO ABBONAMENTO');
38. **END**;
39. /

### **Trigger Istruttori Maggiorenni**

Il trigger parte prima dell’inserimento sulla tabella ISTRUTTORE.  
Controlla semplicemente che l’istruttore appena assunto non abbia un età minore di 25 anni. Altrimenti lancia un errore.

1. **CREATE** OR REPLACE **TRIGGER** ISTRUTTORE\_ABILITATO
2. BEFORE **INSERT** **ON** ISTRUTTORE
3. **FOR** EACH ROW
4. **DECLARE**
5. NO\_ABILITATO EXCEPTION;
7. **BEGIN**
9. IF (((SYSDATE -:NEW.DATA\_NASCITA)/365) < 25) **THEN** RAISE NO\_ABILITATO;
10. **END** IF;
12. EXCEPTION
13. **WHEN** NO\_ABILITATO **THEN**
14. RAISE\_APPLICATION\_ERROR(-20504,'ISTRUTTORE NON ABILITATO');
15. **END**;
16. /

### **Trigger Iscritti Maggiorenni**

Il Trigger parte prima dell’inserimento sulla tabella **Iscritto** e controlla che l’iscritto non sia **minorenne**. Genera un errore in caso contrario.

1. **CREATE** OR REPLACE **TRIGGER** ISCRITTO\_MINORENNE
2. BEFORE **INSERT** **ON** ISCRITTO
3. **FOR** EACH ROW
4. **DECLARE**
5. MINORENNE EXCEPTION;
7. **BEGIN**
8. -- GLI ISCRITTI DEVONO ESSERE TUTTI MAGGIORENNI
9. IF (((SYSDATE -:NEW.DATA\_NASCITA)/365) < 18) **THEN** RAISE MINORENNE;
10. **END** IF;
12. EXCEPTION
13. **WHEN** MINORENNE **THEN**
14. RAISE\_APPLICATION\_ERROR(-20530,'ISCRITTO MINORENNE');
15. **END**;
16. /

### **Trigger Eventi Massimi**

Il Trigger parte prima dell’inserimento sulla tabella **Evento**, controlla che il **numero** **di** **eventi** in una struttura non sia maggiore di **4** altrimenti genera un errore.

1. **CREATE** OR REPLACE **TRIGGER** CONTROLLO\_EVENTI
2. BEFORE **INSERT** **ON** EVENTO
3. **FOR** EACH ROW
4. **DECLARE**
5. N\_EVENTI NUMBER;
6. TROPPI\_EVENTI EXCEPTION;
8. **BEGIN**
9. -- CONTROLLA IL NUMERO DI EVENTI IN UNA STRUTTURA
10. -- NON DEVONO ESSERE PIU DI 4 IN UN GIORNO
11. **SELECT** COUNT(\*) **INTO** N\_EVENTI
12. **FROM** EVENTO
13. **WHERE** CAP\_EVENTO = :NEW.CAP\_EVENTO AND DATA = :NEW.DATA;
15. IF (N\_EVENTI > 3 AND :NEW.DATA > SYSDATE)
16. **THEN** RAISE TROPPI\_EVENTI;
17. **END** IF;
19. EXCEPTION
20. **WHEN** TROPPI\_EVENTI **THEN**
21. RAISE\_APPLICATION\_ERROR(-20510,'TROPPI EVENTI NELLA STESSA DATA');
23. **END**;
24. /

## **Procedure**

Verranno mostrate ora tutte le procedure e operazioni di base descritte precedentemente.

### **Creazione Scheda**

La procedura prenderà in input **Codice Fiscale** dell’**istruttore**, Il **livello** (se immesso) e per ogni **esercizio**, per un **massimo** di 6, **Nome**, **Serie** e **Ripetizioni** (L’istruttore inserirà null in quei valori riferiti agli esercizi mancanti ma non può inserirne **meno** di 2).

1. **CREATE** OR REPLACE **PROCEDURE** CREA\_SCHEDA
2. (
3. CF          SCHEDA\_ESERCIZI.CF\_ISTRUTTORE\_SCHEDA%TYPE,
4. DIFFICOLTA      SCHEDA\_ESERCIZI.LIVELLO%TYPE,
5. ESERCIZIO\_1         CONTIENE.NOME\_ESERCIZIO%TYPE,
6. SERIE\_1             CONTIENE.N\_SERIE%TYPE,
7. RIPETIZIONI\_1       CONTIENE.N\_RIPETIZIONI%TYPE,
8. ESERCIZIO\_2         CONTIENE.NOME\_ESERCIZIO%TYPE,
9. SERIE\_2             CONTIENE.N\_SERIE%TYPE,
10. RIPETIZIONI\_2       CONTIENE.N\_RIPETIZIONI%TYPE,
11. ESERCIZIO\_3         CONTIENE.NOME\_ESERCIZIO%TYPE,
12. SERIE\_3             CONTIENE.N\_SERIE%TYPE,
13. RIPETIZIONI\_3       CONTIENE.N\_RIPETIZIONI%TYPE,
14. ESERCIZIO\_4         CONTIENE.NOME\_ESERCIZIO%TYPE,
15. SERIE\_4             CONTIENE.N\_SERIE%TYPE,
16. RIPETIZIONI\_4       CONTIENE.N\_RIPETIZIONI%TYPE,
17. ESERCIZIO\_5         CONTIENE.NOME\_ESERCIZIO%TYPE,
18. SERIE\_5             CONTIENE.N\_SERIE%TYPE,
19. RIPETIZIONI\_5       CONTIENE.N\_RIPETIZIONI%TYPE,
20. ESERCIZIO\_6         CONTIENE.NOME\_ESERCIZIO%TYPE,
21. SERIE\_6             CONTIENE.N\_SERIE%TYPE,
22. RIPETIZIONI\_6       CONTIENE.N\_RIPETIZIONI%TYPE
23. )
25. **IS**
26. NUMERO             NUMBER;
27. FLAG               NUMBER;
28. ESERCIZIO\_EXC EXCEPTION;
30. **BEGIN**
32. -- CONTA GLI ESERCIZI DA INSERIRE
33. **SELECT** COUNT(\*) **INTO** FLAG
34. **FROM** ESERCIZIO
35. **WHERE** NOME = ESERCIZIO\_1 OR NOME = ESERCIZIO\_2 OR
36. NOME = ESERCIZIO\_3 OR NOME = ESERCIZIO\_4 OR
37. NOME = ESERCIZIO\_5 OR NOME = ESERCIZIO\_6;
39. -- SE MINORI DI 2 LANCIA ECCEZIONE
40. IF(FLAG < 2)
41. **THEN** RAISE ESERCIZIO\_EXC;
42. **END** IF;
44. -- CONTA IL NUMERO MASSIMO DI ID SCHEDA ASSEGNATO PER CALCOLARE E INSERIRE L'ID DELLA SUCCESSIVA
45. **SELECT** **MAX**(N\_SCHEDA) **INTO** NUMERO
46. **FROM** SCHEDA\_ESERCIZI;
48. NUMERO := NUMERO + 1;

51. -- CREA LA SCHEDA
52. **INSERT** **INTO** SCHEDA\_ESERCIZI **VALUES**(CF,NUMERO,DIFFICOLTA);
53. -- INSERISCE I PRIMI 2 ESERCIZI
54. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_1,SERIE\_1,RIPETIZIONI\_1);
55. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_2,SERIE\_2,RIPETIZIONI\_2);
56. -- CONTROLLA SE C'E' UN ALTRO ESERCIZIO LO INSERISCE
57. IF (ESERCIZIO\_3 **IS** NOT NULL) **THEN**
58. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_3,SERIE\_3,RIPETIZIONI\_3);
59. **END** IF;
61. IF (ESERCIZIO\_4 **IS** NOT NULL) **THEN**
62. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_4,SERIE\_4,RIPETIZIONI\_4);
63. **END** IF;
65. IF (ESERCIZIO\_5 **IS** NOT NULL) **THEN**
66. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_5,SERIE\_5,RIPETIZIONI\_5);
67. **END** IF;
69. IF (ESERCIZIO\_6 **IS** NOT NULL) **THEN**
70. **INSERT** **INTO** CONTIENE **VALUES**(NUMERO,ESERCIZIO\_6,SERIE\_6,RIPETIZIONI\_6);
71. **END** IF;
73. EXCEPTION
74. **WHEN** ESERCIZIO\_EXC **THEN**
75. RAISE\_APPLICATION\_ERROR(-20451, 'NON PUOI INSERIRE UN SOLO ESERCIZIO');

78. **END** CREA\_SCHEDA;
79. /

### **Sottoscrizione Abbonamento**

La Procedura prenderà in input **Codice** **Fiscale** **istruttore** e il **numero** dei **mesi** di validità del nuovo **abbonamento** (1 mese, 3 mesi, 6 mesi o 12 mesi).  
Recupera il **Badge** **dell’iscritto** che aveva in precedenza e se non lo ha immetterà **null** in modo tale che la sequenza auto-badge vista in precedenza dia un nuovo **Badge**.

1. **CREATE** OR REPLACE **PROCEDURE** SOTTOSCRIVI\_ABBONAMENTO
2. (
3. CF\_I            ISCRITTO.CF\_ISCRITTO%TYPE,
4. N\_MESI          NUMBER
5. )
7. **IS**
8. BADGE\_I             ABBONAMENTO.BADGE%TYPE;
9. DURATA\_NON\_VALIDA   EXCEPTION;
11. **BEGIN**
13. -- CONTROLLA CHE LA RICHIESTA DELL ABBONAMENTO NON SIA DIVERSA DI 1,3,6 O 12 MESI
14. IF(N\_MESI <> 1 AND N\_MESI <> 3 AND N\_MESI <> 6 AND N\_MESI <> 12)
15. **THEN** RAISE DURATA\_NON\_VALIDA;
16. **END** IF;
18. -- SELEZIONA IL BADGE CHE STA USANDO L'ISCRITTO DELL'ISCRITTO PER RINNOVARLO
19. -- SE L'ISCRITTO NON HA BADGE O E' LA PRIMA VOLTA CHE SOTTOSCRIVE UN ABBONAMENTO BADGE SARA' NULL
20. -- E SARA' GENERATO DAL TRIGGER
21. **SELECT** **MAX**(BADGE) **INTO** BADGE\_I
22. **FROM** ISCRITTO JOIN ABBONAMENTO **ON** CF\_ISCRITTO = CF\_ISCRITTO\_ABBONAMENTO
23. **WHERE** CF\_I = CF\_ISCRITTO;

26. **DELETE** **FROM** ABBONAMENTO **WHERE** BADGE = BADGE\_I;
27. **INSERT** **INTO** ABBONAMENTO **VALUES** (BADGE\_I,CF\_I,SYSDATE,SYSDATE+(30 \* N\_MESI));


31. EXCEPTION
32. **WHEN** DURATA\_NON\_VALIDA **THEN**
33. RAISE\_APPLICATION\_ERROR(-20564,'LA DURATA DELL ABBONAMENTO PUO ESSERE DI 1,2,3,6 O 12 MESI');
34. **WHEN** NO\_DATA\_FOUND **THEN**
35. BADGE\_I := NULL;
37. **END** SOTTOSCRIVI\_ABBONAMENTO;
38. /

### **Stila Informazioni Iscritto**

La procedura prende in input il **Codice** **Fiscale** dell’**iscritto** e genera un listato di informazioni:  
- I corsi  
- Lista esercizi  
- Info Personali  
- Info Abbonamenti e Corsi

Genera un **errore** se l’iscritto **non** **esiste** e un **warning** se non ha ancora una scheda esercizi assegnata.

1. **CREATE** OR REPLACE **PROCEDURE** INFO\_ISCRITTO
2. (
3. CF\_I            ISCRITTO.CF\_ISCRITTO%TYPE
4. )
5. **IS**
7. USER                ISCRITTO%ROWTYPE;
8. BADGE\_IN            ABBONAMENTO.BADGE%TYPE;
9. DATA\_IN             ABBONAMENTO.DATA\_INIZIO%TYPE;
10. DATA\_FN             ABBONAMENTO.DATA\_SCADENZA%TYPE;
11. LIVELLO\_IN          SCHEDA\_ESERCIZI.LIVELLO%TYPE;
12. CORSI               PREVEDE%ROWTYPE;
13. LISTA               CONTIENE%ROWTYPE;
14. NO\_SCHEDA           EXCEPTION;
15. **CURSOR** CORSI\_SCHEDA **IS**
16. -- MOSTRA I CORSI A CUI è ISCRITTO L'UTENTE
17. **SELECT** BADGE\_FK,CAP\_CORSO\_FK,NOME\_CORSO\_FK
18. **FROM** ISCRITTO JOIN ABBONAMENTO  **ON** CF\_ISCRITTO = CF\_ISCRITTO\_ABBONAMENTO
19. JOIN PREVEDE **ON** BADGE\_FK = BADGE
20. **WHERE** CF\_ISCRITTO = CF\_I;
22. **CURSOR** LISTA\_ES **IS**
23. -- LA LISTA DI ESERCIZI NELLA SCHEDA
24. **SELECT** E.N\_SCHEDA\_FK,C.NOME\_ESERCIZIO,C.N\_SERIE,C.N\_RIPETIZIONI
25. **FROM** SCHEDA\_ESERCIZI S JOIN CONTIENE C **ON** S.N\_SCHEDA = C.N\_SCHEDA\_FK
26. JOIN E\_ASSEGNATA E **ON** E.N\_SCHEDA\_FK = C.N\_SCHEDA\_FK
27. **WHERE** CF\_ISCRITTO\_FK = CF\_I;


31. **BEGIN**
33. **SELECT** CF\_ISCRITTO,TELEFONO,NOME,COGNOME,SESSO,DATA\_NASCITA **INTO** USER
34. **FROM** ISCRITTO
35. **WHERE** CF\_ISCRITTO = CF\_I;
37. -- MOSTRA INFORMAZIONI BASE DELL'ISCRITTO
38. DBMS\_OUTPUT.PUT\_LINE('NOME: '       || USER.NOME);
39. DBMS\_OUTPUT.PUT\_LINE('COGNOME: '    || USER.COGNOME);
40. DBMS\_OUTPUT.PUT\_LINE('SESSO: '      || USER.SESSO);
41. DBMS\_OUTPUT.PUT\_LINE('TELEFONO: '   || USER.TELEFONO);
43. -- INFORMAZIONI SU ABBONAMENTO
44. **SELECT** BADGE, DATA\_INIZIO, DATA\_SCADENZA **INTO** BADGE\_IN,DATA\_IN,DATA\_FN
45. **FROM** ISCRITTO JOIN ABBONAMENTO  **ON** CF\_ISCRITTO = CF\_ISCRITTO\_ABBONAMENTO
46. **WHERE** CF\_ISCRITTO = CF\_I;
48. DBMS\_OUTPUT.PUT\_LINE('BADGE ABBONAMENTO: '      || BADGE\_IN);
49. DBMS\_OUTPUT.PUT\_LINE('DATA INIZIO: '            || DATA\_IN);
50. DBMS\_OUTPUT.PUT\_LINE('DATA SCADENZA: '          || DATA\_FN);
51. DBMS\_OUTPUT.PUT\_LINE('DURATA RIMANENTE: '       || TRUNC(TO\_NUMBER(DATA\_FN-SYSDATE)) || ' GIORNI');

54. **OPEN** CORSI\_SCHEDA;
56. IF CORSI\_SCHEDA%NOTFOUND **THEN**
57. **CLOSE** CORSI\_SCHEDA;
58. **END** IF;
60. LOOP
61. **FETCH** CORSI\_SCHEDA **INTO** CORSI;
62. EXIT **WHEN** CORSI\_SCHEDA%NOTFOUND;
63. DBMS\_OUTPUT.PUT\_LINE('CORSO: ' || CORSI.NOME\_CORSO\_FK || CORSI.CAP\_CORSO\_FK);
64. **END** LOOP;
65. **CLOSE** CORSI\_SCHEDA;
67. **SELECT** LIVELLO **INTO** LIVELLO\_IN
68. **FROM** ISCRITTO JOIN E\_ASSEGNATA **ON** CF\_ISCRITTO = CF\_ISCRITTO\_FK
69. JOIN SCHEDA\_ESERCIZI **ON** N\_SCHEDA = N\_SCHEDA\_FK
70. **WHERE** CF\_ISCRITTO = CF\_I;
71. DBMS\_OUTPUT.PUT\_LINE('DETTAGLIO SCHEDA ALLENAMENTO');
72. DBMS\_OUTPUT.PUT\_LINE('LIVELLO SCHEDA: '         || LIVELLO\_IN);
73. DBMS\_OUTPUT.PUT\_LINE('ESERCIZI: ');

76. -- MOSTRA SCHEDA UTENTE CON ESERCIZI
77. **OPEN** LISTA\_ES;
79. IF LISTA\_ES%NOTFOUND **THEN**
80. RAISE NO\_SCHEDA;
81. **CLOSE** LISTA\_ES;
82. **END** IF;
84. LOOP
85. **FETCH** LISTA\_ES **INTO** LISTA;
86. EXIT **WHEN** LISTA\_ES%NOTFOUND;
87. DBMS\_OUTPUT.PUT\_LINE(' ');
88. DBMS\_OUTPUT.PUT\_LINE('NOME: ' || LISTA.NOME\_ESERCIZIO || 'SERIE: ' || LISTA.N\_SERIE || ' x ' || LISTA.N\_RIPETIZIONI);
89. **END** LOOP;
90. **CLOSE** LISTA\_ES;



95. EXCEPTION
96. **WHEN** NO\_DATA\_FOUND **THEN**
97. RAISE\_APPLICATION\_ERROR(-20543,'ISCRITTO NON ESISTENTE');
98. **WHEN** NO\_SCHEDA **THEN**
99. DBMS\_OUTPUT.PUT\_LINE('L ISCRITTO NON POSSIEDE UNA SCHEDA ESERCIZI');
101. **END** INFO\_ISCRITTO;
102. /

### **Stila Lista Turni**

La procedura prende in input il **Codice** **Fiscale** di un **Istruttore** e genera un listato con:  
- Turni totali istruttori  
- Conteggio turni ultimo mese e li mostra con data e orari  
- I corsi a cui lavora l’istruttore  
Genera un **errore** se l’istruttore **non** **ha** **svolto** **turni**.

1. **CREATE** OR REPLACE **PROCEDURE** LISTA\_TURNI\_ISTRUTTORE
2. (
3. CF  TURNO.CF\_ISTRUTTORE\_TURNO%TYPE
4. )
5. **IS**
7. CONTA\_TURNI     NUMBER;
8. NOT\_TURNO       EXCEPTION;
9. NOME\_IS         ISTRUTTORE.NOME%TYPE;
10. COGNOME\_IS      ISTRUTTORE.COGNOME%TYPE;
11. LISTA\_IN        TURNO%ROWTYPE;
12. CORSO\_I         CORSO%ROWTYPE;
14. **CURSOR** LISTA **IS**
15. **SELECT** CF\_ISTRUTTORE\_TURNO,CAP\_TURNO,DATA\_TURNO,ORA\_TURNO,ORA\_OUT\_TURNO
16. **FROM** TURNO
17. **WHERE** CF\_ISTRUTTORE\_TURNO = CF;
19. **CURSOR** CORSO\_ISTRUTTORE **IS**
20. **SELECT** CF\_ISTRUTTORE\_CORSO,CAP\_CORSO,NOME\_CORSO,DATA\_INIZIO\_CORSO,DATA\_FINE\_CORSO,ORA\_INIZIO\_CORSO
21. **FROM** ISTRUTTORE JOIN CORSO **ON** CF\_ISTRUTTORE = CF\_ISTRUTTORE\_CORSO
22. **WHERE** CF\_ISTRUTTORE = CF;

25. **BEGIN**
26. -- CONTA I TURNI TOTALI E PRENDE NOME E COGNOME
27. **SELECT** COUNT(\*),**MAX**(NOME),**MAX**(COGNOME) **INTO** CONTA\_TURNI,NOME\_IS,COGNOME\_IS
28. **FROM** TURNO JOIN ISTRUTTORE **ON** CF\_ISTRUTTORE\_TURNO = CF\_ISTRUTTORE
29. **WHERE** CF\_ISTRUTTORE\_TURNO = CF;
31. IF (CONTA\_TURNI < 1)
32. **THEN** RAISE NOT\_TURNO;
33. **END** IF;
35. DBMS\_OUTPUT.PUT\_LINE('ISTRUTTORE: ' || NOME\_IS || ' ' || COGNOME\_IS );
36. DBMS\_OUTPUT.PUT\_LINE('TOTALE TURNI SVOLTI: ' || CONTA\_TURNI);
37. -- CONTA I TURNI DELL'ULTIMO MESE E LI MOSTRA
38. **SELECT** COUNT(\*) **INTO** CONTA\_TURNI
39. **FROM** TURNO JOIN ISTRUTTORE **ON** CF\_ISTRUTTORE\_TURNO = CF\_ISTRUTTORE
40. **WHERE** CF\_ISTRUTTORE\_TURNO = CF AND DATA\_TURNO > SYSDATE - 30;
41. DBMS\_OUTPUT.PUT\_LINE('TOTALE TURNI ULTMIMO MESE: ' || CONTA\_TURNI);
43. **OPEN** LISTA;
44. IF LISTA%NOTFOUND **THEN** **CLOSE** LISTA;
45. **END** IF;
47. LOOP
48. **FETCH** LISTA **INTO** LISTA\_IN;
49. EXIT **WHEN** LISTA%NOTFOUND;
50. DBMS\_OUTPUT.PUT\_LINE('DATA: ' || LISTA\_IN.DATA\_TURNO || ' ORA: ' || LISTA\_IN.ORA\_TURNO || ':00 - ' || LISTA\_IN.ORA\_OUT\_TURNO || ':00 ');
51. **END** LOOP;
52. **CLOSE** LISTA;
54. -- SCORRIAMO I CORSI DELL'ISTRUTTORE
55. **OPEN** CORSO\_ISTRUTTORE;
56. IF CORSO\_ISTRUTTORE%NOTFOUND **THEN** **CLOSE** CORSO\_ISTRUTTORE;
57. **END** IF;
59. LOOP
60. **FETCH** CORSO\_ISTRUTTORE **INTO** CORSO\_I;
61. EXIT **WHEN** CORSO\_ISTRUTTORE%NOTFOUND;
62. DBMS\_OUTPUT.PUT\_LINE('CORSO TENUTO: ' || CORSO\_I.NOME\_CORSO || ' PRESSO: ' || CORSO\_I.CAP\_CORSO);
63. **END** LOOP;
64. **CLOSE** CORSO\_ISTRUTTORE;


68. EXCEPTION
69. **WHEN** NOT\_TURNO **THEN**
70. RAISE\_APPLICATION\_ERROR(-20005,'NESSUN TURNO PER QUESTO ISTRUTTORE');
72. **END** LISTA\_TURNI\_ISTRUTTORE;
73. /

### **Assegna Scheda**

La procedura prende in input il **Codice** **Fiscale** dell’**iscritto**, **numero** **scheda** da assegnare (**NULL** se bisogna rinnovare la vecchia) e **data** **scadenza** della **nuova** **scheda**.  
Produce un **warning** per avvisare se la vecchia non era ancora scaduta e uno per avvisare il cambiamento di difficoltà della scheda.

1. **CREATE** OR REPLACE **PROCEDURE** ASSEGNA\_SCHEDA
2. (
3. CF          ISCRITTO.CF\_ISCRITTO%TYPE,
4. SCHEDA      E\_ASSEGNATA.N\_SCHEDA\_FK%TYPE,
5. DATA\_F      E\_ASSEGNATA.DATA\_FINE%TYPE
6. )
7. **IS**
9. WARNING     EXCEPTION;
10. CAMBIAMENTO EXCEPTION;
11. DATAENDOLD  **DATE**;
12. OLDSCHEDA   NUMBER;
13. LIV         SCHEDA\_ESERCIZI.LIVELLO%TYPE;
14. LIVNEW      SCHEDA\_ESERCIZI.LIVELLO%TYPE;
16. **BEGIN**
17. -- PRENDIAMO IL NUMERO DELLA SCHEDA PRECEDENTE E LA DATA DI SCADENZA
19. **SELECT** N\_SCHEDA\_FK,DATA\_FINE **INTO** OLDSCHEDA,DATAENDOLD
20. **FROM** E\_ASSEGNATA
21. **WHERE** CF\_ISCRITTO\_FK = CF
22. AND DATA\_FINE = (**SELECT** **MAX**(DATA\_FINE) **FROM** E\_ASSEGNATA **WHERE** CF\_ISCRITTO\_FK = CF);
24. -- SE ANCORA DEVE SCADERA LANCIA WARNING CHE AGGIORNA LA DATA SCADENZA AD OGGI E AVVISA L'UTENTE
25. IF (DATAENDOLD > SYSDATE AND SCHEDA **IS** NOT NULL) **THEN** RAISE WARNING;
26. **END** IF;
28. -- SE NON E STATA INSERITA LA SCHEDA PROLUNGA LA VECCHIA
29. IF (SCHEDA **IS** NULL) **THEN**
30. DBMS\_OUTPUT.PUT\_LINE('AGGIORNAMENTO DELLA SCHEDA');
31. **UPDATE** E\_ASSEGNATA **SET** DATA\_FINE = DATA\_F **WHERE** N\_SCHEDA\_FK = OLDSCHEDA;
32. **END** IF;
34. -- RECUPERIAMO IL LIVELLO DELLA VECCHIA SCHEDA
35. **SELECT** LIVELLO **INTO** LIV
36. **FROM** SCHEDA\_ESERCIZI
37. **WHERE** N\_SCHEDA = OLDSCHEDA;
39. -- RECUPERIAMO IL LIVELLO DELLA NUOVA
40. **SELECT** LIVELLO **INTO** LIVNEW
41. **FROM** SCHEDA\_ESERCIZI
42. **WHERE** N\_SCHEDA = SCHEDA;

45. IF (LIV **IS** NOT NULL AND LIV <> LIVNEW) **THEN** RAISE CAMBIAMENTO;
46. **END** IF;
48. -- ASSEGNA LA NUOVA SCHEDA
49. **INSERT** **INTO** E\_ASSEGNATA **VALUES** (CF,SCHEDA,SYSDATE,DATA\_F);
51. EXCEPTION
52. **WHEN** WARNING **THEN**
53. DBMS\_OUTPUT.PUT\_LINE('LA VECCHIA SCHEDA ASSEGNATA NON ERA ANCORA SCADUTA: SARA IMPOSTATA COME DATA DI SCADENZA OGGI' );
54. **UPDATE** E\_ASSEGNATA **SET** DATA\_FINE = SYSDATE **WHERE** N\_SCHEDA\_FK = OLDSCHEDA;
55. **WHEN** CAMBIAMENTO **THEN**
56. DBMS\_OUTPUT.PUT\_LINE('LA VECCHIA SCHEDA ASSEGNATA ERA DI LIVELLO ' || LIV || ' LA NUOVA DI LIVELLO ' || LIVNEW);
57. **END** ASSEGNA\_SCHEDA;
58. /

### **Assegna Corso**

La procedura prende in input il **Nome** dei due **Corsi** (**NULL** il secondo se solo uno), il **CAP** di dove si devono svolgere e il **Codice** **Fiscale** dell’**iscritto**.  
Recupera da sola il badge dell’abbonamento valido dell’iscritto e assegnerà uno o due corsi a quell’abbonamento. Genererà errori se si tenta si inserire più corsi di due.

1. **CREATE** OR REPLACE **PROCEDURE** ASSEGNA\_CORSO
2. (
3. CAP\_IN          CORSO.CAP\_CORSO%TYPE,
4. NOME\_1          CORSO.NOME\_CORSO%TYPE,
5. NOME\_2          CORSO.NOME\_CORSO%TYPE,
6. CF\_I            ISCRITTO.CF\_ISCRITTO%TYPE
7. )
9. **IS**
10. CONTROLLO\_NUMERO\_CORSI  NUMBER;
11. TROPPI                  EXCEPTION;
12. BADGEI      ABBONAMENTO.BADGE%TYPE;
14. **BEGIN**
16. -- CONTIAMO IL NUMERO DI CORSI GIA PRESENTI NELL ABBONAMENTO
17. **SELECT** COUNT(\*) **INTO** CONTROLLO\_NUMERO\_CORSI
18. **FROM** PREVEDE JOIN ABBONAMENTO **ON** BADGE\_FK = BADGE
19. **WHERE** CF\_ISCRITTO\_ABBONAMENTO = CF\_I;
21. -- SE HA GIA DUE CORSI LANCIA ERRORE
22. IF CONTROLLO\_NUMERO\_CORSI > 1
23. **THEN** RAISE TROPPI;
24. **END** IF;
26. --RECUPERIAMO IL BADGE DELL'ISCRITTO
27. **SELECT** BADGE **INTO** BADGEI
28. **FROM** ABBONAMENTO
29. **WHERE** CF\_ISCRITTO\_ABBONAMENTO = CF\_I
30. AND DATA\_SCADENZA = (**SELECT** **MAX**(DATA\_SCADENZA) **FROM** ABBONAMENTO **WHERE** CF\_ISCRITTO\_ABBONAMENTO = CF\_I);
32. --INSERISCI NELL'ABBONAMENTO
33. IF (NOME\_1 **IS** NOT NULL) **THEN**
34. **INSERT** **INTO** PREVEDE **VALUES**(BADGEI,CAP\_IN,NOME\_1);
35. DBMS\_OUTPUT.PUT\_LINE('PRIMO CORSO INSERITO');
36. CONTROLLO\_NUMERO\_CORSI := CONTROLLO\_NUMERO\_CORSI + 1;
37. **END** IF;
39. -- RIESEGUI IL CONTROLLO
40. IF CONTROLLO\_NUMERO\_CORSI > 1
41. **THEN** RAISE TROPPI;
42. **END** IF;
44. --INSERISCI IL SECONDO CORSO
45. IF (NOME\_2 **IS** NOT NULL) **THEN**
46. **INSERT** **INTO** PREVEDE **VALUES**(BADGEI,CAP\_IN,NOME\_2);
47. DBMS\_OUTPUT.PUT\_LINE('SECONDO CORSO INSERITO');
48. **END** IF;

51. EXCEPTION
52. **WHEN** NO\_DATA\_FOUND **THEN**
53. CONTROLLO\_NUMERO\_CORSI := 0;
54. **WHEN** TROPPI **THEN**
55. RAISE\_APPLICATION\_ERROR(-20567,'NON PUOI ATTIVARE PIU DI DUE CORSI');

58. **END** ASSEGNA\_CORSO;
59. /

### **Assumi Istruttore**

La procedura è molto semplice e inserisce una tupla nella tabella **Istruttore** prendendo in input tutte le info utili.

1. **CREATE** OR REPLACE **PROCEDURE** ASSUMI\_ISTRUTTORE
2. (
3. CF          ISTRUTTORE.CF\_ISTRUTTORE%TYPE,
4. **NO**          ISTRUTTORE.NOME%TYPE,
5. CO          ISTRUTTORE.COGNOME%TYPE,
6. CELL        ISTRUTTORE.TELEFONO%TYPE,
7. DATA        ISTRUTTORE.DATA\_NASCITA%TYPE,
8. SESSO       ISTRUTTORE.SESSO%TYPE
9. )
10. **IS**

13. **BEGIN**
15. **INSERT** **INTO** ISTRUTTORE **VALUES**(CF,CELL,**NO**,CO,SESSO,DATA);
17. **END** ASSUMI\_ISTRUTTORE;
18. /

### **Licenzia Istruttore**

La procedura è molto semplice ed elimina una tupla nella tabella **Istruttore** prendendo in input solo il **Codice** **Fiscale**.

1. **CREATE** OR REPLACE **PROCEDURE** LICENZIA\_ISTRUTTORE
2. (
3. CF          ISTRUTTORE.CF\_ISTRUTTORE%TYPE
4. )
5. **IS**
7. **BEGIN**
8. **DELETE** **FROM** ISTRUTTORE **WHERE** CF\_ISTRUTTORE = CF;
10. **END** LICENZIA\_ISTRUTTORE;
11. /

### **Crea Evento**

La procedura è molto semplice ed inserisce una tupla nella tabella **Evento** prendendo in input solo nome dell’evento, il **CAP** e la **data**.

1. **CREATE** OR REPLACE **PROCEDURE** CREA\_EVENTO
2. (
3. NOME        EVENTO.NOME\_EVENTO%TYPE,
4. CAP         EVENTO.CAP\_EVENTO%TYPE,
5. DATA        EVENTO.DATA%TYPE
6. )
7. **IS**
9. DOMENICA EXCEPTION;
11. **BEGIN**
13. IF (UPPER(TO\_CHAR(DATA,'DY')) = 'SUN') **THEN** RAISE DOMENICA;
14. **END** IF;
16. **INSERT** **INTO** EVENTO **VALUES** (NOME,CAP,DATA);
18. EXCEPTION
19. **WHEN** DOMENICA **THEN**
20. RAISE\_APPLICATION\_ERROR(-20567,'NON PUOI ORGANIZZARE EVENTI DI DOMENICA!');
21. **END** CREA\_EVENTO;
22. /

## **Viste**

Infine, riportiamo le viste.

### **Lezioni ultimo mese**

La vista seguente mostra una tutte le lezioni svoltesi nell’ultimo mese ordinate per **data** e **CAP**.

1. --VISUALIZZA TUTTE LE LEZIONI SVOLTESI NELL'ULTIMO MESE
2. **CREATE** **VIEW** LEZIONI
3. **SELECT** NOME\_LEZIONE **AS** CORSO, CAP\_LEZIONE **AS** CAP,DATA\_LEZIONE **AS** DATA,ORA\_INIZIO\_LEZIONE **AS** ORA, ORA\_FINE\_LEZIONE **AS** ORA\_FINE,N\_STANZA **AS** STANZA
4. **FROM** LEZIONE
5. **WHERE** DATA\_LEZIONE > SYSDATE - 30
6. **ORDER** **BY** DATA,CAP **DESC**;

### **Lezioni ultimo mese**

La vista genera una tabella con tutti gli **Abbonamenti** **scaduti** e il **Codice** **Fiscale** dell’**iscritto** a cui appartengono.

1. --VISUALIZZA GLI ISCRITTI A CUI E' SCADUTO L'ABBONAMENTO
2. **CREATE** **VIEW** ABBONAMENTI\_SCADUTI **AS**
3. **SELECT** CF\_ISCRITTO, COGNOME, BADGE
4. **FROM** PREVEDE JOIN ABBONAMENTO **ON** BADGE\_FK = BADGE
5. JOIN ISCRITTO **ON** CF\_ISCRITTO\_ABBONAMENTO = CF\_ISCRITTO
6. **WHERE** DATA\_SCADENZA < SYSDATE;

### **Catalogo Corsi**

La Vista mostra un catalogo corsi con gli **istruttori** che li tengono, le **date** e il **CAP** dove si svolge

1. ---VISTA DIRETTORE PER CATALOGO CORSI
3. **CREATE** **VIEW** CORSI\_ISTRUTTORI **AS**
4. **SELECT** NOME\_CORSO  **AS** CORSO,CAP\_CORSO **AS** CAP,CITTA,COGNOME **AS** ISTRUTTORE,DATA\_INIZIO\_CORSO **AS** INIZIO,ORA\_INIZIO\_CORSO **AS** ORARIO,DATA\_FINE\_CORSO **AS** FINE
5. **FROM** ISTRUTTORE JOIN CORSO **ON** CF\_ISTRUTTORE\_CORSO = CF\_ISTRUTTORE
6. JOIN STRUTTURA **ON** CAP\_CORSO = CAP;

### **Eventi in Programma**

La Vista mostra un catalogo di eventi in programma.

1. **CREATE** **VIEW** EVENTI\_PROGRAMMA **AS**
2. **SELECT** E.NOME\_EVENTO,S.CITTA,S.CAP,S.VIA,E.DATA
3. **FROM** EVENTO E JOIN STRUTTURA S **ON** E.CAP\_EVENTO = S.CAP
4. **WHERE** DATA > SYSDATE;

### **Iscritti ad Eventi**

La Vista mostra una lista di iscritti che hanno partecipato e parteciperanno ad eventi.

1. **CREATE** **VIEW** ISCRITTO\_EVENTO **AS**
2. **SELECT** NOME, COGNOME, NOME\_EVENTO, DATA
3. **FROM** ISCRITTO JOIN PARTECIPA **ON** CF\_ISCRITTO\_PARTECIPA = CF\_ISCRITTO
4. JOIN EVENTO **ON** NOME\_PARTECIPA = NOME\_EVENTO;

## **Scheduler**

Il codice che segue implementerà un job che controllerà mensilmente gli iscritti che hanno un abbonamento scaduto da più di 6 mesi e li rimuove dal database.

1. -- JOB CHE CANCELLA ISCRITTI CHE NON HANNO RINNOVATO L'ABBONAMENTO DA PIU DI 6 MESI
2. **BEGIN**
3. DBMS\_SCHEDULER.CREATE\_JOB (
4. job\_name            => 'Cancella\_Iscritto',
5. job\_type            => 'PLSQL\_BLOCK',
6. job\_action          => '**BEGIN**
7. **DELETE** **FROM** ISCRITTO
8. **WHERE** CF\_ISCRITTO IN (
9. **SELECT** CF\_ISCRITTO
10. **FROM** ISCRITTO JOIN ABBONAMENTO **ON** CF\_ISCRITTO = CF\_ISCRITTO\_ABBONAMENTO
11. **WHERE** DATA\_SCADENZA < SYSDATE - (30 \* 6) );
12. **END**;',
13. start\_date          =>  TO\_DATE('01/12/2018', 'DD/MM/YYYY'),
14. repeat\_interval     => 'FREQ=MONTHLY',
15. enabled             =>  **TRUE**,
16. comments            => 'Cancella iscritti con abbonamenti non rinnovti da più di 6 mesi');
17. **END**;
18. /