*Basi di Dati e Laboratorio di Basi di Dati*

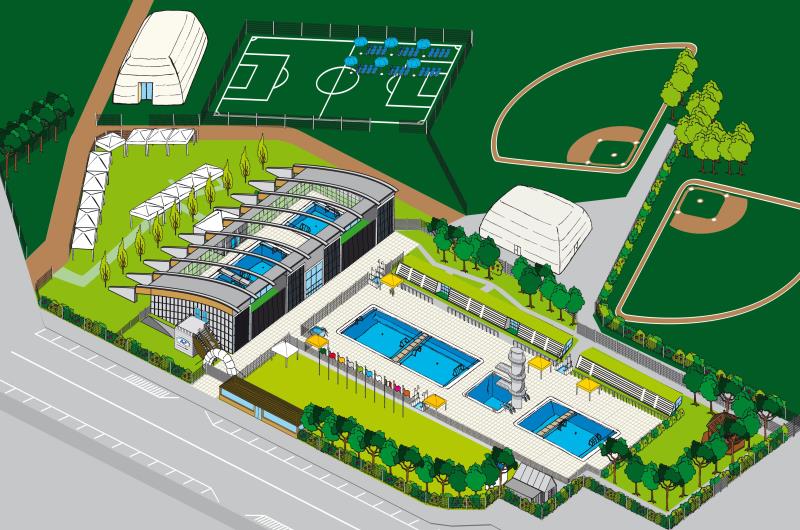
Prof. A. Maratea

*Anno Accademico 2019/2020*



**Progetto d’esame**

*Gestione di una catena di Polisportive*

**

*Capobianco Salvatore  
0124000974  
  
Lomello Carlo  
0124001651*

*Data di consegna:*

# Indice

[**CAPITOLO 1 - Analisi dei requisiti 2**](#_Toc30429255)

[*1.1 - Sintesi dei requisiti 2*](#_Toc30429256)

[*1.2 - Glossario 3*](#_Toc30429257)

[**CAPITOLO 2 - Diagramma Concettuale (EER) & Relazionale 5**](#_Toc30429258)

*Diagramma Concettuale EER ………………………………………………………………………………………………….... 5*

[*2.1 - Analisi entità e loro attributi 6*](#_Toc30429259)

[*2.2 - Analisi associazioni 8*](#_Toc30429260)

*Diagramma Relazionale ………..……………………………………………………………….…………………………….... 10*

[*2.3 - Tabelle 11*](#_Toc30429261)

[*2.4 - Analisi tabelle e modellazione attributi 11*](#_Toc30429262)

[**CAPITOLO 3 - Utenti e loro categorie 14**](#_Toc30429263)

[*3.1 - Categorie Utenti 14*](#_Toc30429264)

[*3.2 Operazioni degli utenti 17*](#_Toc30429265)

[*3.3 - Volumi 21*](#_Toc30429266)

[*3.4 - Vincoli d’integrità statici 22*](#_Toc30429267)

[*3.5 - Vincoli d’integrità dinamici 23*](#_Toc30429268)

[**CAPITOLO 4 - Normalizzazione 25**](#_Toc30429269)

[**CAPITOLO 5 - Possibili estensioni 26**](#_Toc30429270)

[**CAPITOLO 6 - Implementazione 27**](#_Toc30429271)

[*6.1 - Data Definition Language 27*](#_Toc30429273)

[*6.2 - Data Manipulation Language 32*](#_Toc30429274)

[*6.3 - Triggers 49*](#_Toc30429275)

[*6.4 - Procedure 65*](#_Toc30429276)

[*6.5 - Viste 77*](#_Toc30429277)

[*6.6 - Data Control Language 84*](#_Toc30429278)

[*6.7 - Scheduler 87*](#_Toc30429279)

# CAPITOLO 1 - Analisi dei requisiti

## - Sintesi dei requisiti

Si vuole realizzare una base di dati per gestire una catena di polisportive.  
Il database in questione dovrà registrare e gestire le informazioni fondamentali, anche di natura amministrativa, logistica e finanziaria, di ogni sede:

* Personale dipendente e professionisti esterni (istruttori), e relativa turnazione;
* Corsi periodici e clienti abbonati;
* Campi e relative prenotazioni telefoniche dei clienti;
* Utenze;
* Fornitori e relativi ordini di prodotti commerciali o attrezzatura sportiva;
* Vendite effettuate al bar della struttura.

Ogni sede, è aperta dalle 9:00 alle 23:00; ospita un bar e una serie di campi e sale per varie discipline, principalmente calcetto, pallavolo, basket e arti marziali come judo e karate; essi vengono utilizzati sia per i corsi organizzati dalla struttura che per le attività autogestite dei clienti, previa prenotazione telefonica.   
L’azienda si è posta l’obiettivo di costruire e mantenere almeno una sede nelle più popolose città italiane, ma non più di una nella stessa zona/quartiere/indirizzo.

Il calendario dei corsi viene organizzato, di anno in anno, solitamente nei primi giorni di gennaio, dai direttori delle sedi o, più raramente, dal proprietario della catena, e ha la priorità rispetto alle prenotazioni esterne per quanto riguarda l’occupazione dei campi.

Un cliente può sottoscrivere un abbonamento ad un corso per una durata minima di un mese e massima della durata del corso stesso, ed eventualmente rinnovarlo. In ogni caso, abbonamenti e prenotazioni, prevedono il pagamento anticipato, della relativa quota da parte dei clienti, direttamente in segreteria, pertanto non si rende necessario, monitorare lo stato di questi pagamenti, ma solo registrarne gli importi.

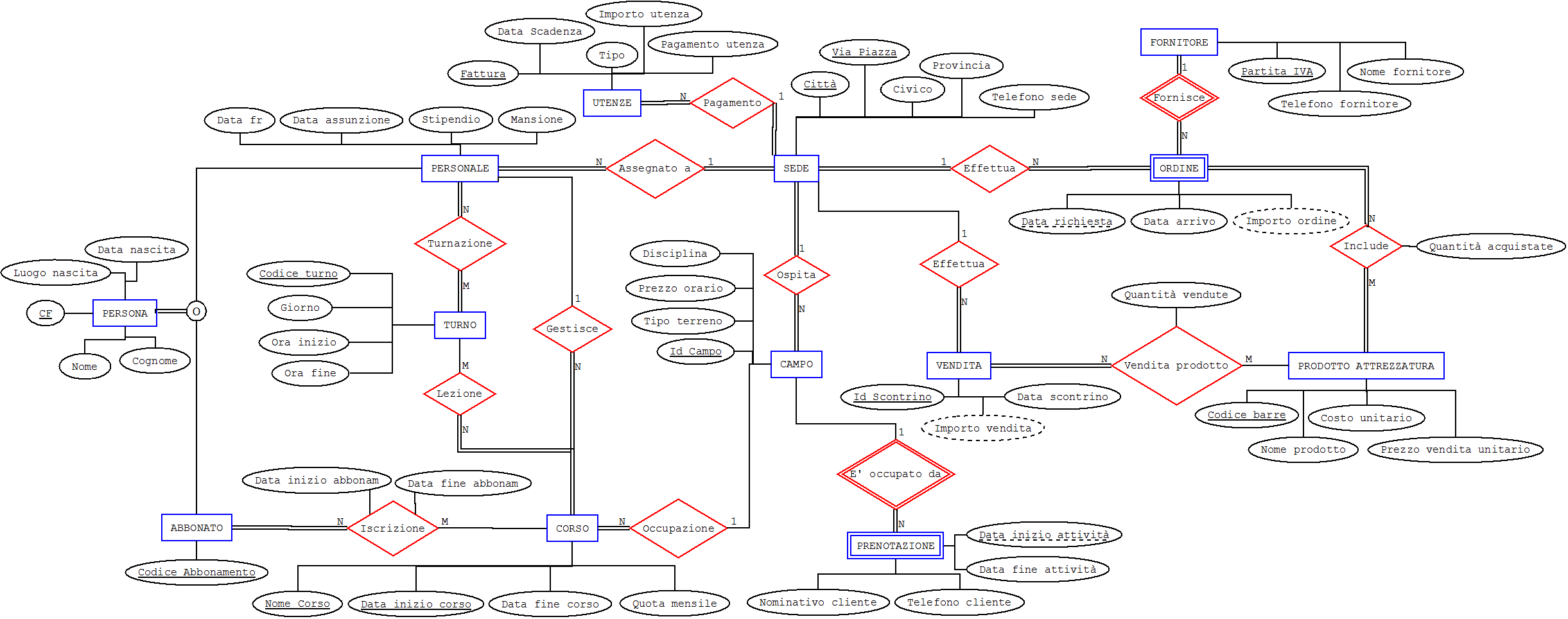
La segreteria è incaricata principalmente dell’accoglienza e gestione dei clienti, quindi delle iscrizioni ai corsi e delle prenotazioni mentre gli addetti al bar si occupano delle vendite, dell’inventario di magazzino, e quindi, degli ordini ai fornitori che vengono pagati alla consegna. Occasionalmente, a seconda delle esigenze della struttura, la direzione potrebbe effettuare degli ordini di attrezzatura.

Custodi, inservienti e clienti non hanno motivo di accedere al database.

## - Glossario

|  |  |  |  |
| --- | --- | --- | --- |
| **Termini** | **Definizione** | **Sinonimi** | **Collegamenti** |
| Sede | Luogo dov’è posta una polisportiva della catena | Struttura, Polisportiva | PERSONALE, UTENZE, CAMPO, ORDINE, VENDITA |
| Cliente | Persona che si iscrive ad un corso o prenota un campo. | Abbonato, Iscritto | ABBONATO, PRENOTAZIONE |
| Istruttore | Professionista esterno all’azienda, ingaggiato per la supervisione di uno o più corsi | Personale, Lavoratore | CORSO, SEDE, TURNAZIONE |
| Personale dipendente | Personale interno all’azienda con diverse mansioni/ruoli | Personale, Lavoratore, Custode, Inserviente, Segreteria, Bar, Direzione | SEDE, TURNAZIONE, |
| Turnazione | Calendario dei turni del personale | - | TURNO, PERSONALE |
| Prodotto commerciale | Prodotto vendibile al bar della struttura | - | INCLUDE, VENDITA\_PRODOTTO |
| Attrezzatura sportiva | Prodotto non vendibile riservato alla struttura | - | INCLUDE |
| Corso | Insieme di lezioni di una determinata disciplina sportiva, tenuta su un campo | - | RELAZIONE, ISCRIZIONE |
| Prenotazione | Atto con cui un cliente prenota un campo | - | CAMPO |
| Ordine | Lista di prodotti richiesti ad un fornitore | - | FORNITORE, INCLUDE |
| Abbonamento | Sottoscrizione di partecipazione ad un corso da parte di un abbonato | Iscrizione, Tesseramento | ABBONATO, CORSO |

# CAPITOLO 2 - Diagramma Concettuale (EER) & Relazionale



## 2.1 - Analisi entità e loro attributi

*LEGENDA:*

**ENTITA’ (DEBOLE/SPECIALIZZAZIONE) CHIAVE** DEBOLE ATTRIBUTO derivato

**SEDE** - strutture della catena di polisportive:

* **CITTA** - città dov’è situata la sede;
* **VIA PIAZZA** - indirizzo della sede;
* CIVICO - numero civico della sede;
* TELEFONO SEDE - contatto telefonico della sede.

**UTENZE** - servizi di cui fruiscono le strutture:

* **FATTURA** - numero identificativo della bolletta;
* TIPO - tipologia del servizio;
* DATA SCADENZA - termine ultimo per il pagamento del servizio senza more;
* IMPORTO UTENZA - costo della bolletta;
* PAGAMENTO UTENZA - stato del pagamento della bolletta.

**PERSONA** - persone (generalità), che per diversi motivi, si rapportano con la polisportiva:

* **CF** - codice fiscale della persona;
* NOME;
* COGNOME;
* LUOGO NASCITA;
* DATA NASCITA.

**PERSONALE (specializzazione di PERSONA) \*** - dipendenti e istruttori esterni delle polisportive:

* MANSIONE - ruolo del lavoratore all’interno della struttura;
* STIPENDIO - paga mensile del lavoratore;
* DATA ASSUNZIONE - data di assunzione del lavoratore;
* DATA FR - data di fine rapporto o contratto del lavoratore.

**ABBONATO** **(specializzazione di PERSONA) \*** - clienti iscritti ai corsi:

* **CODICE ABBONAMENTO** - identificativo della singola sottoscrizione (non viene aggiornata in caso di rinnovo).

**TURNO** - periodi, in cui si svolge un turno lavorativo da parte del personale oppure una lezione di un corso:

* **CODICE TURNO** - identificativo del turno;
* GIORNO - giorno della settimana;
* ORA INIZIO - ora di inizio della fascia oraria;
* ORA FINE - ora di fine della fascia oraria.

**CORSO** - corsi sportivi specializzati, organizzati dalla struttura:

* **NOME CORSO**;
* **DATA INIZIO CORSO** - data in cui si terrà la prima lezione del corso;
* DATA FINE CORSO - data in cui si terrà l’ultima lezione del corso;
* QUOTA MENSILE - costo mensile dell’abbonamento al corso.

**CAMPO** - campi o sale, per attività sportive, delle sedi:

* **ID CAMPO** - identificativo del campo;
* DISCIPLINA - tipologia di attività che è possibile praticare sul campo;
* TIPO TERRENO - tipologia del terreno o della pavimentazione del campo;
* PREZZO ORARIO - costo del fitto da parte dei prenotanti.

**PRENOTAZIONE (debole di CAMPO)** - prenotazioni di fitto dei campi, da parte di clienti esterni:

* DATA INIZIO ATTIVITA’ - data e ora in cui avrà inizio l’attività autogestita;
* DATA FINE ATTIVITA’ - data e ora in cui terminerà l’attività autogestita;
* NOMINATIVO CLIENTE - nome e cognome del prenotante;
* TELEFONO CLIENTE - contatto telefonico del prenotante.

**FORNITORE** - fornitori di prodotti commerciali (bar) e attrezzatura per le polisportive:

* **PARTITA IVA** - numero di partita iva che identifica il fornitore;
* NOME FORNITORE - nome dell’azienda;
* TELEFONO FORNITORE - contatto telefonico del fornitore.

**ORDINE (debole di FORNITORE)** - richieste di un insieme, di una lista di prodotti, ad un fornitore:

* DATA RICHIESTA - data in cui è stata effettuata la richiesta da parte della struttura;
* DATA ARRIVO - data in cui l’ordine è stato consegnato in sede;
* importo ordine - costo complessivo dell’ordine.

**PRODOTTO ATTREZZATURA** - prodotti vendibili al bar della struttura e attrezzatura sportiva:

* **CODICE BARRE** - identificativo del prodotto;
* NOME PRODOTTO;
* COSTO UNITARIO - costo del prodotto, al pezzo, per la polisportiva;
* PREZZO VENDITA UNITARIO - costo del prodotto, al pezzo, per i clienti del bar.

**VENDITA** - vendite di un insieme, di una lista di prodotti, effettuate dal bar della struttura:

* **ID SCONTRINO** - numero identificativo dello scontrino legato alla vendita;
* DATA SCONTRINO - data e ora in cui è stato stampato lo scontrino;
* importo vendita - importo complessivo della vendita.

\**Specializzazione totale sovrapposta (O): Tutte le persone registrate sono lavoratori e/o iscritti ai corsi. Al personale è permesso iscriversi ai corsi organizzati dalla polisportiva stessa, ad ovvia eccezione degli istruttori, che non possono farlo ai corsi che supervisionano.*

## 2.2 - Analisi associazioni

**TURNAZIONE** [CF, CODICE TURNO]

Molteplicità N:M e totalità ambo i lati: Tutti i lavoratori sono associati ad uno o più turni, tutti i turni sono associati ad uno o più lavoratori.

**LEZIONE** [CODICE TURNO, NOME CORSO, DATA INIZIO CORSO]

Molteplicità N:M e totalità lato CORSO: Tutti i corsi sono associati ad uno o più turni, alcuni turni sono associati ad uno o più corsi. Rappresenta la programmazione settimanale dei corsi.

**ISCRIZIONE** [CODICE ABBONAMENTO, NOME CORSO, DATA INIZIO CORSO, DATA INIZIO ABBONAMENTO, DATA FINE ABBONAMENTO]

Molteplicità N:M e totalità lato ABBONATO: Tutti gli abbonati sono associati ad uno o più corsi, alcuni corsi sono associati ad uno o più abbonati (in determinati momenti potrebbero non averne).  
DATA INIZIO ABBONAMENTO e DATA FINE ABBONAMENTO sono gli estremi temporali di validità dell’iscrizione al corso da parte dell’abbonato.

**GESTISCE** [CF, NOME CORSO, DATA INIZIO CORSO]

Molteplicità 1:N e totalità lato CORSO: Alcuni lavoratori (gli istruttori) gestiscono uno o più corsi, tutti i corsi sono gestiti da uno degli istruttori.

**ASSEGNATO A** [CF, CITTA’, VIA PIAZZA]

Molteplicità 1:N e totalità ambo i lati: Tutti i lavoratori sono assegnati ad una delle sedi, tutte le sedi hanno dei lavoratori.

**OCCUPAZIONE** [NOME CORSO, DATA INIZIO CORSO, ID CAMPO]

Molteplicità 1:N e totalità lato CORSO: Tutti i corsi si svolgono su uno dei campi, alcuni campi sono occupati da uno o più corsi in diversi momenti.

**PAGAMENTO** [# FATTURA, CITTA’, VIA PIAZZA]

Molteplicità 1:N e totalità ambo i lati: Tutte le utenze sono associate ad una delle sedi, tutte le sedi pagano una o più utenze.

**OSPITA** [CITTA’, VIA PIAZZA, ID CAMPO]

Molteplicità 1:N e totalità ambo i lati: Tutte le sedi ospitano dei campi, tutti i campi fanno parte di una delle sedi.

**EFFETTUA** [CITTA’, VIA PIAZZA, DATA RICHIESTA]

Molteplicità 1:N e totalità ambo i lati: Tutte le sedi fanno degli ordini, tutti gli ordini vengono richiesti da una delle sedi.

**EFFETTUA** [CITTA’, VIA PIAZZA, DATA RICHIESTA]

Molteplicità 1:N e totalità lato VENDITA: Alcune sedi effettuano delle vendite, tutte le vendite vengono effettuate da una delle sedi.

**È OCCUPATO DA *(identificazione di PRENOTAZIONE da CAMPO)*** [ID CAMPO, DATA INIZIO ATTIVITA’]

Molteplicità 1:N e totalità lato PRENOTAZIONE: Alcuni campi sono impegnati da una o più prenotazioni, tutte le prenotazioni impegnano uno dei campi.

**FORNISCE *(identificazione di ORDINE da FORNITORE)*** [PARTITA IVA, DATA RICHIESTA]

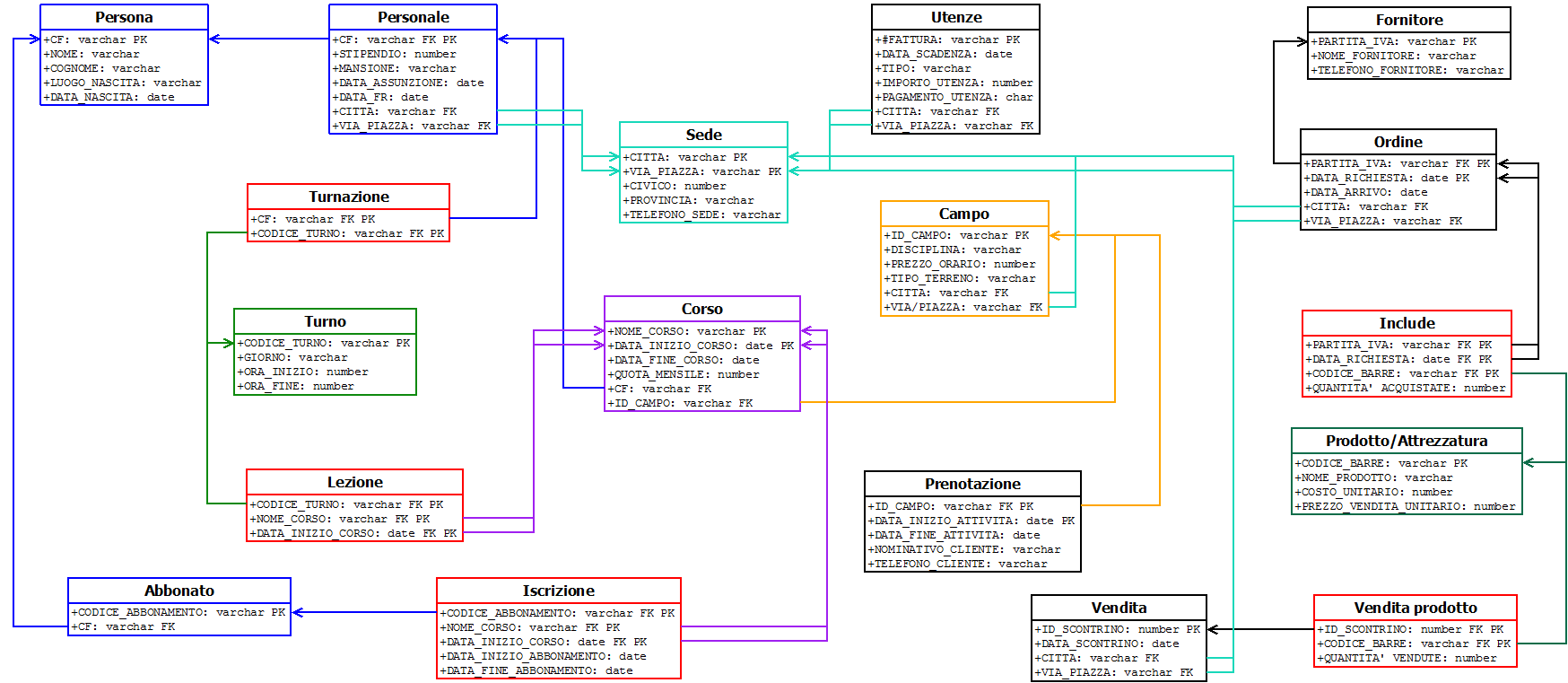
Molteplicità 1:N e totalità ambo i lati: Tutti i fornitori sono incaricati di uno o più ordini, tutti gli ordini sono gestiti da uno dei fornitori.

**INCLUDE** [PARTITA IVA, DATA RICHIESTA, CODICE BARRE, QUANTITA’ ACQUISTATE]

Molteplicità N:M e totalità ambo i lati: Tutti gli ordini includono uno o più prodotti, tutti i prodotti sono associati ad uno o più ordini.  
QUANTITA’ ACQUISTATE indica il numero per prodotto, acquistato dalla struttura.

**VENDITA PRODOTTO** [ID SCONTRINO, CODICE BARRE, QUANTITA’ VENDUTE]

Molteplicità N:M e totalità lato VENDITA: Tutte le vendite sono associate ad uno o più prodotti, alcuni prodotti sono associati ad una o più vendite.  
QUANTITA’ VENDUTE indica il numero per prodotto, venduto dal bar della struttura.



## 2.3 - Tabelle

Il DB presenta 18 tabelle di cui:

* 9 entità
* 2 entità deboli
* 2 sotto-entità figlie
* 5 relazioni N:M

## - Analisi tabelle e modellazione attributi

* **PERSONA:   
  CF** è una stringa di massimo 16 caratteri, nonché **chiave primaria** della tabella;  
  NOME, COGNOME e LUOGO\_NASCITA sono stringhe, rispettivamente di massimo 32, 32 e 64 caratteri, e sono campi obbligatori, ai fini delle registrazioni di componenti del personale e degli abbonati ai corsi;  
  DATA\_NASCITA è un attributo di tipo data ed è anch’esso obbligatorio per gli stessi motivi.
* **ABBONATO:**  
  **CODICE\_ABBONAMENTO** è un numero intero, nonché **chiave primaria** artificiale auto-incrementata della tabella;  
  ***CF*** è ***chiave esterna*** ereditata dall’entità madre PERSONA;
* **PERSONALE:**  
  ***CF*** è ***chiave esterna*** ereditata dall’entità madre PERSONA, nonché **chiave primaria** della tabella;  
  MANSIONE è una stringa di massimo 64 caratteri, obbligatoria;  
  STIPENDIO è un numero decimale di 8 cifre (7 interi, 2 decimali), obbligatorio;  
  DATA\_ASSUNZIONE è una data obbligatoria;  
  DATA\_FR è un attributo di tipo data;  
  ***CITTA*** e ***VIA\_PIAZZA*** sono ***chiavi esterne*** riferite agli omonimi attributi della tabella SEDE.
* **SEDE:   
  CITTA** e **VIA\_PIAZZA** sono stringhe, rispettivamente di massimo 64 e 128 caratteri, e la loro combinazione costituisce la **chiave primaria** della tabella;  
  CIVICO è un numero intero di massimo 8 cifre, obbligatorio;  
  PROVINCIA e TELEFONO\_SEDE sono stringhe, rispettivamente di massimo 2 e 16 caratteri, obbligatorie.
* **UTENZE:   
  FATTURA** è una stringa di massimo 16 caratteri, nonché **chiave primaria** della tabella;  
  DATA\_SCADENZA è una data obbligatoria;  
  TIPO è una stringa di massimo 64 caratteri, obbligatoria;  
  IMPORTO\_UTENZA è un numero decimale di 7 cifre (5 interi, 2 decimali), obbligatorio;  
  PAGAMENTO\_UTENZA è un carattere obbligatorio;  
  ***CITTA*** e ***VIA\_PIAZZA*** sono ***chiavi esterne*** riferite agli omonimi attributi della tabella SEDE.
* **TURNO:   
  CODICE\_TURNO** è una stringa di massimo 8 caratteri, nonché **chiave primaria** della tabella;  
  GIORNO è una stringa di massimo 10 caratteri, obbligatoria;  
  ORA\_INIZIO e ORA\_FINE sono numeri interi di massimo 2 cifre, obbligatori.
* **CORSO:   
  NOME\_CORSO** e **DATA\_INIZIO\_CORSO** sono, rispettivamente, una stringa di massimo 64 caratteri, e una data, e la loro combinazione costituisce la **chiave primaria** della tabella;  
  DATA\_FINE\_CORSO è una data obbligatoria;  
  QUOTA\_MENSILE è un numero decimale di 5 cifre (3 interi, 2 decimali), obbligatorio;  
  ***CF*** e ***ID\_CAMPO*** sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle PERSONA e CAMPO;
* **CAMPO:   
  ID\_CAMPO** è una stringa di massimo 8 caratteri, nonché **chiave primaria** della tabella;  
  DISCIPLINA è una stringa di massimo 64 caratteri, obbligatoria;  
  PREZZO\_ORARIO è un numero decimale di 5 cifre (3 interi, 2 decimali), obbligatorio;  
  TIPO\_TERRENO è una stringa di massimo 64 caratteri, obbligatoria;  
  ***CITTA*** e ***VIA\_PIAZZA*** sono ***chiavi esterne*** riferite agli omonimi attributi della tabella SEDE.
* **PRENOTAZIONE:   
  *ID\_CAMPO*** è ***chiave esterna (forte)*** ereditata dall’entità forte CAMPO;  
  **DATA\_INIZIO\_ATTIVITA** è una data, nonché **chiave debole**: la combinazione di chiave forte e chiave debole costituisce la **chiave primaria** della tabella;  
  DATA\_FINE\_ATTIVITA è una data obbligatoria;  
  NOMINATIVO\_CLIENTE e TELEFONO\_CLIENTE sono stringhe, rispettivamente di massimo 64 e 16 caratteri, obbligatorie.
* **FORNITORE:   
  PARTITA\_IVA** è una stringa di massimo 11 caratteri, nonché **chiave primaria** della tabella;  
  NOME\_FORNITORE e TELEFONO\_FORNITORE sono stringhe, rispettivamente di massimo 64 e 16 caratteri, obbligatoria.
* **ORDINE:   
  *PARTITA\_IVA*** è ***chiave esterna (forte)*** ereditata dall’entità forte FORNITORE;  
  **DATA\_RICHIESTA** è una data, nonché **chiave debole**: la combinazione di chiave forte e chiave debole costituisce la **chiave primaria** della tabella;  
  DATA\_ARRIVO è un attributo di tipo data;

***CITTA*** e ***VIA\_PIAZZA*** sono ***chiavi esterne*** riferite agli omonimi attributi della tabella SEDE.

* **PRODOTTO\_ATTREZZATURA:   
  CODICE\_BARRE** è una stringa di massimo 64 caratteri, nonché **chiave primaria** della tabella;  
  NOME\_PRODOTTO è una stringa di massimo 64 caratteri, obbligatoria;  
  COSTO\_UNITARIO e PREZZO\_VENDITA\_UNITARIO sono numeri decimali di 8 cifre (6 interi, 2 decimali), di cui, solo il primo obbligatorio.
* **VENDITA:   
  ID\_SCONTRINO** è un numero intero, nonché **chiave primaria** artificiale auto-incrementata della tabella;  
  DATA\_SCONTRINO è una data, con valore di default pari alla data e ora di sistema;  
  ***CITTA*** e ***VIA\_PIAZZA*** sono ***chiavi esterne*** riferite agli omonimi attributi della tabella SEDE.
* **TURNAZIONE**:  
  CF e CODICE\_TURNO sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle PERSONALE e TURNO, e la loro combinazione costituisce la **chiave primaria** della tabella.
* **LEZIONE**:  
  CODICE\_TURNO, NOME\_CORSO, DATA\_INIZIO\_CORSO sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle TURNO e CORSO, e la loro combinazione costituisce la **chiave primaria** della tabella.
* **ISCRIZIONE**:  
  CODICE\_ABBONAMENTO, NOME\_CORSO, DATA\_INIZIO\_CORSO sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle ABBONATO e CORSO, e la loro combinazione costituisce la **chiave primaria** della tabella;  
  DATA\_INIZIO\_ABBONAMENTO e DATA\_FINE\_ABBONAMENTO sono date obbligatorie.
* **INCLUDE**:  
  PARTITA\_IVA, DATA\_RICHIESTA e CODICE\_BARRE sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle FORNITORE, ORDINE e PRODOTTO\_ATTREZZATURA, e la loro combinazione costituisce la **chiave primaria** della tabella;  
  QUANTITA\_ACQUISTATE è un numero intero di massimo 8 cifre.
* **VENDITA\_PRODOTTO**:  
  ID\_SCONTRINO e CODICE\_BARRE sono ***chiavi esterne*** riferite agli omonimi attributi, rispettivamente, delle tabelle VENDITA e PRODOTTO\_ATTREZZATURA, e la loro combinazione costituisce la **chiave primaria** della tabella;  
  QUANTITA\_VENDUTE è un numero intero di massimo 8 cifre.

# CAPITOLO 3 - Utenti e loro categorie

## 3.1 - Categorie Utenti

Le categorie di utenti del DB sono tre: Direzione, Segreteria e Bar.   
Per ognuno di questi ruoli, ci sono tre utenti (verosimilmente uno per sede), più il proprietario della catena che fa parte della “Direzione” e l’amministratore della base di dati.  
Nella seguente tabella è rappresentata la **tavola degli utenti**; successivamente i privilegi di oggetto.

|  |  |  |  |
| --- | --- | --- | --- |
| **RUOLO** | **UTENTI** | **TIPO** | **VOLUME** |
| - | Admin\_Polisportive | Amministratore | 1 |
| Direzione | dir1, dir2, dir3, proprieta | Comune | 4 |
| Segreteria | seg1, seg2, seg3 | Comune | 3 |
| Bar | bar1, bar2, bar3 | Comune | 3 |

**PERMESSI**

*Admin\_Polisportive:*

GRANT ALL PRIVILEGES TO Admin\_Polisportive;

*Direzione:*

GRANT SELECT, INSERT, UPDATE ON persona TO Direzione;

GRANT SELECT, INSERT, UPDATE ON personale TO Direzione;

GRANT SELECT ON vw\_bilancio TO Direzione;

GRANT SELECT ON vw\_info\_ordini\_aperti TO Direzione;

GRANT SELECT ON vw\_info\_ordini\_ricevuti TO Direzione;

GRANT SELECT ON vw\_abbonamenti\_in\_corso TO Direzione;

GRANT SELECT ON vw\_abbonamenti\_terminati TO Direzione;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Direzione;

GRANT SELECT ON vw\_info\_corsi\_in\_atto TO Direzione;

GRANT SELECT ON vw\_info\_corsi\_terminati TO Direzione;

GRANT SELECT ON vw\_info\_prenotazioni\_in\_corso TO Direzione;

GRANT SELECT ON vw\_info\_storico\_prenotazioni TO Direzione;

GRANT SELECT ON vw\_vendite\_ultimo\_mese TO Direzione;

GRANT SELECT ON vw\_storico\_vendite TO Direzione;

GRANT SELECT ON vw\_storico\_utenze\_non\_pagate TO Direzione;

GRANT SELECT ON vw\_storico\_utenze\_pagate TO Direzione;

GRANT SELECT, INSERT, UPDATE ON turno TO Direzione;

GRANT SELECT, INSERT, UPDATE ON turnazione TO Direzione;

GRANT SELECT, INSERT, UPDATE ON lezione TO Direzione;

GRANT SELECT, INSERT, UPDATE ON corso TO Direzione;

GRANT SELECT, INSERT, UPDATE ON utenze TO Direzione;

GRANT SELECT, INSERT, UPDATE ON sede TO Direzione;

GRANT SELECT, INSERT, UPDATE ON campo TO Direzione;

GRANT SELECT, INSERT, UPDATE ON fornitore TO Direzione;

GRANT UPDATE, DELETE ON ordine TO Direzione;

GRANT SELECT, INSERT, UPDATE ON prodotto\_attrezzatura TO Direzione;

GRANT SELECT ON vw\_importo\_ordine TO Direzione;

GRANT SELECT ON vw\_importo\_vendita TO Direzione;

GRANT EXECUTE ON effettua\_ordine TO Direzione;  
GRANT EXECUTE ON ordine\_consegnato TO Direzione;

*Segreteria:*

GRANT SELECT, INSERT, UPDATE ON persona TO Segreteria;

GRANT SELECT, UPDATE ON abbonato TO Segreteria;

GRANT SELECT ON personale TO Segreteria;

GRANT SELECT ON vw\_abbonamenti\_in\_corso TO Segreteria;

GRANT SELECT ON vw\_abbonamenti\_terminati TO Segreteria;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Segreteria;

GRANT SELECT ON vw\_info\_corsi\_in\_atto TO Segreteria;

GRANT SELECT ON vw\_info\_corsi\_terminati TO Segreteria;

GRANT SELECT ON vw\_info\_prenotazioni\_in\_corso TO Segreteria;

GRANT SELECT ON vw\_info\_storico\_prenotazioni TO Segreteria;

GRANT DELETE ON prenotazione TO Segreteria;

GRANT SELECT ON vw\_storico\_utenze\_non\_pagate TO Segreteria;

GRANT SELECT ON vw\_storico\_utenze\_pagate TO Segreteria;

GRANT SELECT, INSERT, UPDATE ON utenze TO Segreteria;

GRANT SELECT ON sede TO Segreteria;

GRANT SELECT ON campo TO Segreteria;

GRANT SELECT ON vw\_importo\_ordine TO Segreteria;

GRANT SELECT ON vw\_importo\_vendita TO Segreteria;

GRANT EXECUTE ON prenotazione\_campo TO Segreteria;

GRANT EXECUTE ON iscrizione\_corso TO Segreteria;

*Bar:*

GRANT SELECT ON sede TO Bar;

GRANT SELECT ON vw\_info\_ordini\_aperti TO Bar;

GRANT SELECT ON vw\_info\_ordini\_ricevuti TO Bar;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Bar;

GRANT SELECT ON vw\_vendite\_ultimo\_mese TO Bar;

GRANT SELECT ON vw\_storico\_vendite TO Bar;

GRANT SELECT ON fornitore TO Bar;

GRANT SELECT, INSERT, UPDATE ON prodotto\_attrezzatura TO Bar;

GRANT SELECT ON vw\_importo\_ordine TO Bar;

GRANT SELECT ON vw\_importo\_vendita TO Bar;

GRANT EXECUTE ON effettua\_ordine TO Bar;

GRANT EXECUTE ON ordine\_consegnato TO Bar;

GRANT EXECUTE ON vendita\_prodotti TO Bar;

*Creazione utenti e assegnazione ruoli:*

CREATE USER bar1 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg1 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir1 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER bar2 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg2 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir2 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER bar3 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg3 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir3 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER proprieta IDENTIFIED BY proprieta DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

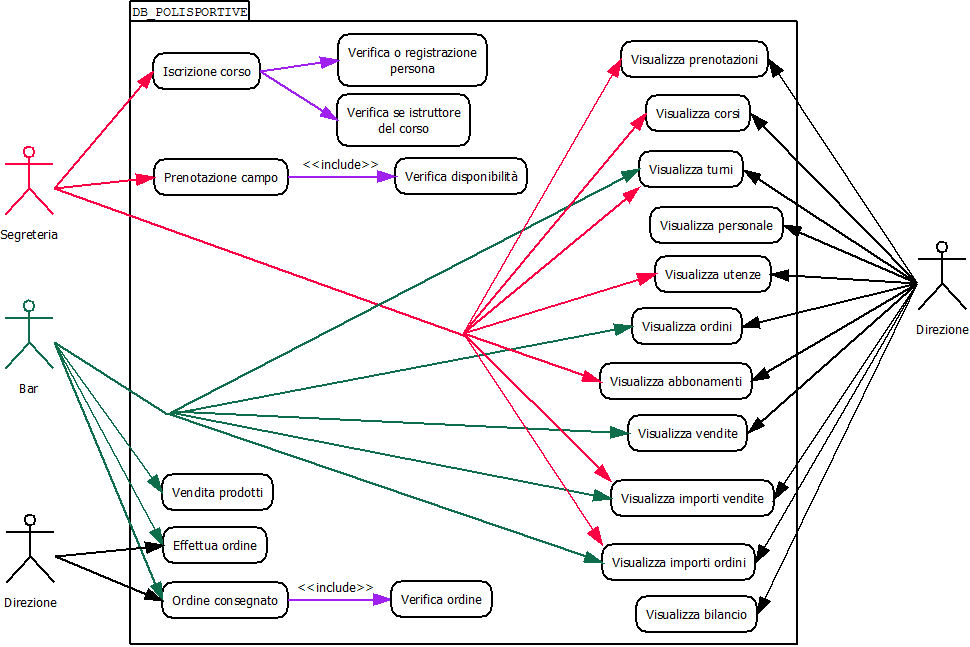
GRANT Direzione TO dir1, dir2, dir3, proprieta;

GRANT Segreteria TO seg1, seg2, seg3;

GRANT Bar TO bar1, bar2, bar3;

## 3.2 Operazioni degli utenti

Oltre alle operazioni di base come inserimenti, aggiornamenti e cancellazioni, queste categorie di utenti hanno i permessi necessari, sia per visualizzare determinati dati attraverso le view, sia eseguire operazioni più complesse, mediante le procedure. Entrambi sono, infatti, degli strumenti molto potenti, rispetto ai semplici comandi DML, sia per progettare le operazioni più frequenti in maniera automatizzata e “*user-friendly”,* sia per limitare le possibilità di far danni da parte degli stessi utenti. Tra le tante possibili si è scelto di implementare le operazioni degli utenti più frequenti nel contesto di una polisportiva, cercando un buon compromesso con i requisiti del progetto d’esame, ovvero: le richieste di ordini ai fornitori, la registrazione della consegna di questi ultimi, le iscrizioni dei clienti ai corsi, le prenotazioni dei campi e le vendite effettuate al bar della struttura. Nel seguente **diagramma dei casi d’uso** sono rappresentate, in maniera generale e in linguaggio naturale, le operazioni degli utenti implementate e i relativi privilegi. Seguono le **schede descrittore operazione** delle suddette.



|  |  |
| --- | --- |
| OPERAZIONE: | Effettua ordine |
| SCOPO: | registrare un ordine e la relativa lista di prodotti |
| ARGOMENTI: | partita iva, data richiesta, città, via piazza, codice barre 1, quantità 1,  [… codice barre 10, quantità 10] |
| RISULTATO: | ordine effettuato/errore |
| ERRORI: | partita iva errata  citta errata  via piazza errata |
| USA: | ordine, include, prodotto attrezzatura, fornitore |
| INSERIMENTO/MODIFICA: | ordine, include |
| PRIMA: | non è registrato un ordine e la relativa lista di prodotti |
| POI: | è registrato un ordine e la relativa lista di prodotti |

|  |  |
| --- | --- |
| OPERAZIONE: | Ordine consegnato |
| SCOPO: | registrare la data di consegna di un ordine già effettuato |
| ARGOMENTI: | partita iva, data richiesta, [data consegna] |
| RISULTATO: | registrazione data consegna/errore |
| ERRORI: | ordine inesistente |
| USA: | ordine |
| MODIFICA: | ordine |
| PRIMA: | l’ordine non è ancora stato consegnato |
| POI: | l’ordine è stato consegnato in data consegna o data odierna |

|  |  |
| --- | --- |
| OPERAZIONE: | Iscrizione corso |
| SCOPO: | Iscrivere un abbonato ad un corso [registrare una persona] |
| ARGOMENTI: | nome corso, data inizio corso, data inizio abbonamento, numero mesi (durata abbonamento), codice fiscale, [nome, cognome, luogo nascita, data nascita] |
| RISULTATO: | registrazione persona e/o abbonamento /errore |
| ERRORI: | istruttore che si iscrive allo stesso corso che supervisiona  persona non registrata (codice fiscale non trovato e argomenti insufficienti) |
| USA: | corso, persona, abbonato, iscrizione |
| INSERIMENTO/MODIFICA: | persona, abbonato, iscrizione |
| PRIMA: | non c’è un abbonamento e/o una persona registrata |
| POI: | c’è un nuovo abbonamento e/o persona registrata |

|  |  |
| --- | --- |
| OPERAZIONE: | Prenotazione campo |
| SCOPO: | prenotare un campo (scelta automatizzata in base alla disciplina) per un’attività autogestita da parte dei clienti |
| ARGOMENTI: | disciplina, città, via piazza, data inizio attività, durata ore attività, nominativo cliente, telefono cliente |
| RISULTATO: | campo prenotato/errore |
| ERRORI: | nessun campo disponibile per la disciplina |
| USA: | campo, prenotazione, corso, lezione, turno |
| INSERIMENTO/MODIFICA: | prenotazione |
| PRIMA: | non c’è una prenotazione |
| POI: | c’è un nuova prenotazione |

|  |  |
| --- | --- |
| OPERAZIONE: | Vendita prodotti |
| SCOPO: | registrare una vendita e la relativa lista di prodotti |
| ARGOMENTI: | città, via piazza, codice barre 1, quantità 1,  [… codice barre 10, quantità 10] |
| RISULTATO: | vendita effettuata/errore |
| ERRORI: | citta errata  via piazza errata |
| USA: | vendita prodotto, prodotto attrezzatura, vendita |
| INSERIMENTO/MODIFICA: | vendita prodotto, vendita |
| PRIMA: | non è registrato una vendita e la relativa lista di prodotti |
| POI: | è registrato una vendita e la relativa lista di prodotti |

\*[ARGOMENTI OPZIONALI]

Segue la **tavola delle operazioni** (stime dei volumi complessivi di tutte le attuali sedi):

|  |  |  |  |
| --- | --- | --- | --- |
| **OPERAZIONE** | **TIPO** | **VOLUME** | **PERIODO** |
| Effettua ordine | B | 150 | anno |
| Ordine consegnato | B | 150 | anno |
| Iscrizione corso | B | 900 | anno |
| Prenotazione campo | B | 75000 | anno |
| Vendita prodotti | B | 300 | giorno |

## 3.3 - Volumi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TABELLE** | **TIPO** | **VOLUME** | **INCREMENTO** | **PERIODO** |
| Abbonato | E(S) | 450 | 150 | Anno |
| Campo | E | 30 | 10 | Anno |
| Corso | E | 30 | 10 | Anno |
| Fornitore | E | 30 | 0 | Anno |
| Include | A | 750 | 250 | Anno |
| Iscrizione | A | 900 | 300 | Anno |
| Ordine | ED | 150 | 50 | Anno |
| Persona | E | 495 | 165 | Anno |
| Personale | E(S) | 45 | 15 | Anno |
| Prenotazione | ED | 75000 | 25000 | Anno |
| Prodotto Attrezzatura | E | 150 | 0 | Anno |
| Lezione | A | 30 | 10 | Anno |
| Sede | E | 3 | 1 | Anno |
| Turnazione | A | 300 | 100 | Anno |
| Turno | E | 50 | 0 | Anno |
| Utenze | E | 72 | 24 | Anno |
| Vendita | E | 300 | 100 | Giorno |
| Vendita prodotto | A | 1500 | 500 | Giorno |

*Note sul calcolo dei volumi:*

Questa tabella dei volumi fa riferimento alla catena di polisportive, costituita da 3 sedi.

Sedi: 3, prevista 1 nuova apertura all’anno (da cui gli incrementi appena descritti).

Abbonati: ogni corso può avere al massimo 30 iscritti (30 iscritti \* 10 corsi \* 3 sedi = 900 massimi, 450 medi).

Campi: Ogni sede ospita circa una decina campi.

Corsi: Ogni sede organizza circa 10 corsi l’anno.

Ordini: Ogni sede richiede circa 50 ordini l’anno.

Include: ogni ordine può essere composto da al massimo 10 prodotti diversi (non è la quantità di prodotti): 10 prodotti \* 50 ordini \* 3 sedi = 1500 massimi, 750 medi.

Iscrizioni: si basa sul numero di abbonati a cui bisogna aggiungere una quantità che dipende dal numero e la durata dei rinnovi, che dipendono a loro volta dalla durata del corso; supponendo verosimilmente che tutti gli abbonati rinnovino l’iscrizione e la sottoscrivano in due trance, otteniamo: 450\*2 = 900.

Personale: ogni sede ha circa 10 dipendenti a cui vanno aggiunti circa 5 istruttori (nel peggiore dei casi, uno per corso, ovvero 10).

Persona: è la somma di abbonati e staff.

Prenotazioni: basate sul numero dei campi (30) e le fasce orarie disponibili dalle 9:00 alle 23:00 (14); supponendo siano tutte della durata minima (1 ora), il massimo possibile è circa 150.000 l’anno, quindi in media 75.000.

Lezioni: è pari al numero di corsi.

Utenze: supponendo che le principali tipologie siano solo 4 e che il periodo di fatturazione sia bimestrale, otteniamo: 6 fatture \* 4 tipologie \* 3 sedi = 72 l’anno.

Vendite e Vendita prodotto: supponiamo 100 vendite al giorno per ogni sede; ognuna può includere al massimo 10 prodotti diversi (non è la quantità di prodotti): 3000 massimo, 1500 in media.

Turnazione: dipende dalle specifiche esigenze e scelte dell’azienda (cambio calendario turni del personale) e dal numero del personale.

Prodotto attrezzatura, Fornitore e Turno: dipendono dalle specifiche esigenze e scelte dell’azienda. Verosimilmente non ci saranno grandi cambiamenti.

## 3.4 - Vincoli d’integrità statici

I vincoli d’integrità statici limitano il dominio degli attributi **indipendentemente dal tempo**.  
Nel seguente elenco sono riportati i principali vincoli d’integrità statici, sia naturali, sia individuati nel contesto delle polisportive e del loro *modus operandi,* ovvero, delle loro regole di business. Non vengono riportati in quanto ovvi, i vincoli di chiave primaria ed esterna, ma per esempio, i vincoli di obbligatorietà (NOT NULL) e di dominio (CHECK):

* Campo: è obbligatorio registrarne la disciplina, il prezzo ad ora, il tipo di terreno o pavimentazione;  
  il prezzo può oscillare tra i 50 e i 200 euro;
* Corso: è obbligatorio registrarne la data di fine e la quota mensile;  
  quest’ultima oscilla tra i 20 e i 200 euro;  
  la data di fine dev’essere posteriore alla data di inizio;
* Fornitore: è obbligatorio registrarne il nome e il contatto telefonico;  
  la partita iva è composta da 11 caratteri;  
  un numero di telefono è composto da un minimo di 9 caratteri, fino ad un massimo di 16;
* Include: le quantità acquistate sono maggiori di zero;
* Iscrizione: è obbligatorio registrarne le date di inizio e fine abbonamento;  
  la prima è anteriore alla seconda;
* Ordine: la data di consegna è posteriore alla data della richiesta;
* Persona: è obbligatorio registrarne nome, cognome, luogo e data di nascita;  
  il codice fiscale è composto da 16 caratteri;
* Personale: è obbligatorio registrarne stipendio, mansione e data assunzione;  
  lo stipendio oscilla tra gli 800 e i 10.000 euro;  
  se esiste una data di fine rapporto/contratto questa è posteriore alla data di assunzione;
* Prenotazione: è obbligatorio registrarne la data di fine attività, il nominativo e il contatto telefonico del cliente;  
  un numero di telefono è composto da un minimo di 9 caratteri, fino ad un massimo di 16;
* Prodotto Attrezzatura: è obbligatorio registrarne nome, costo d’acquisto e prezzo di vendita ai clienti;  
  questi ultimi sono maggiori di zero;  
  un codice a barre è composto da un minimo di 5 caratteri;
* Sede: è obbligatorio registrarne civico, provincia e contatto telefonico;  
  un numero di telefono è composto da un minimo di 9 caratteri, fino ad un massimo di 16;
* Turno: è obbligatorio registrarne giorno, ora di inizio e ora di fine;  
  giorno è un giorno della settimana;  
  le ore oscillano dalle 9 alle 23 e quella di fine è posteriore a quella di inizio;
* Utenze: è obbligatorio registrarne data di scadenza, tipo, importo e stato del pagamento;  
  l’importo è maggiore di zero;  
  lo stato del pagamento è “pagato” o “non pagato” (booleano user-friendly);
* Vendita: la data dello scontrino è di default quella odierna;
* Vendita prodotto: le quantità vendute sono maggiori di zero.

\*tutti i campi obbligatori (NOT NULL) sono sempre “*disponibili*” nella realtà.

## 3.5 - Vincoli d’integrità dinamici

I vincoli d’integrità dinamici, invece, riguardano gli attributi che variano **in funzione del tempo** o che vengono **combinati, con dati di natura temporale**.  
Di seguito, sono elencati, i soli vincoli dinamici gestiti tramite triggers; non vengono riportati in quanto ovvi quelli che gestiscono gli auto-incrementi delle chiavi artificiali:

* Durata prenotazione: il campo può essere occupato da un’attività autogestita dello stesso cliente, per una durata minima di un’ora fino ad un massimo di quattro;
* Età persona: vengono registrate solo persone tra i 6 e i 100 anni;
* Iscrizione in corso: il periodo di validità dell’abbonamento deve rientrare nel periodo del corso;
* Massime iscrizioni ad un corso: un corso può avere al più 30 iscritti;
* Massime ore lavorative giornaliere: per legge è pari a 13 (comprende straordinari);
* Massime ore lavorative settimanali: per legge è pari a 48;
* Massimo staff per sede: ogni sede ha al più 50 lavoratori;
* Massime quantità vendute: i prodotti venduti non possono essere maggiori di quelli disponibili;
* Personale futuro: la data di assunzione non può essere posteriore a quella odierna;
* Personale maggiorenne: per legge i lavoratori devono avere 18 anni di età;
* Sovra-Iscrizioni: un iscritto non può avere più abbonamenti allo stesso corso con periodi di validità sovrapposti;
* Sovrapposizioni tra corsi: non si possono tenere più corsi sullo stesso campo nello stesso momento;
* Sovrapposizioni tra prenotazioni: non si possono prenotare campi già prenotati per lo stesso periodo;
* Sovrapposizioni prenotazioni a corsi: non si possono prenotare campi già occupati da corsi, nello stesso momento;
* Prenotazione passata: non si possono prenotare campi, in periodi passati.

# CAPITOLO 4 - Normalizzazione

Le **forme normali** permettono di valutare la qualità di uno schema nonché la ridondanza che può generare.

1. Uno schema è detto in ***prima forma normale*** se **tutti i suoi attributi sono atomici**, quindi se non presenta attributi multi-valore, composti o loro combinazioni. In altre parole, essa richiede che il dominio di un attributo comprenda solo valori semplici, indivisibili e che il valore di qualsiasi attributo, sia un valore singolo nel suo dominio.  
   Il rispetto di questi requisiti è implicito dal momento che si tratta di uno schema *flat-relational model*.
2. La ***seconda forma normale*** riguarda le chiavi multi-attributo: **tutti gli attributi non primi dipendono in maniera completa dalla chiave**. Cioè tutti gli attributi che non fanno parte della chiave e che non sono univoci (non UNIQUE), dipendono funzionalmente, da tutti quelli che compongono la chiave, ovvero non ci sono dipendenze parziali con la suddetta. Si basa, quindi, sul concetto di **dipendenza funzionale completa**.  
   *Una dipendenza funzionale X da Y, in uno schema R, è una dipendenza funzionale completa se la rimozione di qualsiasi attributo A∈X da X, comporta che la dipendenza non sussista più.   
   Una dipendenza funzionale X da Y è una dipendenza funzionale parziale se si possono rimuovere da X certi attributi A∈X e la dipendenza continua a sussistere.*   
   Nel nostro schema le uniche due entità ad avere una chiave multi-attributo sono SEDE e CORSO ma, in entrambi i casi, gli attributi non primi dipendono funzionalmente, in maniera completa, dalla chiave (concettualmente, potrebbe esserci una dipendenza parziale tra CITTA’ e PROVINCIA ma non è da ritenersi tale dal momento che questi collegamenti non sono noti a priori, nel contesto di una base di dati, che non rappresenta tale informazione e, inoltre, potrebbero esserci città omonime in province diverse).
3. La ***terza forma normale*** richiede che **tutte le dipendenze funzionali siano determinate da chiavi oppure su attributi primi.** Essa non garantisce la conservazione delle dipendenze, ma solo quella dei dati, infatti è sempre raggiungibile. Si basa sul concetto di **dipendenza funzionale transitiva**.  
   *Una dipendenza funzionale X da Y, in uno schema R, è una dipendenza transitiva se esiste un insieme di attributi Z, che non è né una chiave candidata né un sottoinsieme di una chiave di R, per cui valgono contemporaneamente X da Z e Z da Y.*   
   Uno schema è in terza forma normale quando soddisfa la seconda forma normale e quando nessun attributo non-primo dipende in modo transitivo dalla chiave primaria.  
   Nel nostro caso abbiamo che, oltre a rispettare la seconda forma normale, il nostro schema non presenta attributi non-primi che dipendono in modo transitivo dalla chiave primaria.
4. **La BCNF (Boyce-Cod Normal Form)** richiede, invece, che **tutte le dipendenze funzionali siano determinate da chiavi**; è più restrittiva ma garantisce anche la conservazione delle dipendenze, infatti non è sempre raggiungibile.  
   Anche questi requisiti sono rispettati nel nostro schema, poiché tutte le dipendenze funzionali sono determinate da chiavi.

# CAPITOLO 5 - Possibili estensioni

Oltre a procedure per inserimenti e cancellazioni per tutte le tabelle, per migliorarne l’accessibilità, tra le estensioni possibili del DB, abbiamo individuato:

* viene richiesto automaticamente un ordine, quando la quantità in magazzino, di un determinato prodotto, scende al di sotto di una specifica soglia;
* non è permessa l’assunzione di una persona con una specifica mansione, quando, per quel ruolo è già stato raggiunto il limite massimo di personale (dipende dalle regole di business);
* vengono registrati i nominativi dei clienti che hanno disdetto una prenotazione il giorno stesso dell’attività (lista nera);
* viene calcolato un indice di qualità dei fornitori, sulla base del tempo tra, le richieste di ordini e la loro consegna;
* si applicano sconti sulla quota di una prenotazione, se effettuata da un abbonato;
* stilare il bilancio annuale specifico delle varie sedi.

# CAPITOLO 6 - Implementazione

## Programmazione

L’ambiente di sviluppo usato è *Oracle 11g Express Edition.*

È consigliabile effettuare i seguenti passi:

1. Creare l’utente Amministratore con tutti i privilegi e connettersi;
2. Creare le tabelle e le sequenze per le chiavi artificiali;
3. Creare i triggers per l’auto-incremento delle suddette sequenze;
4. Popolare le tabelle (ordine numerato);
5. Inserire triggers, procedure, views e schedulers;
6. Creare gli utenti con i relativi privilegi di ruolo.

## 6.1 - Data Definition Language

-- CREAZIONE TABELLE E RELATIVI VINCOLI

CREATE TABLE sede (

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  civico NUMBER(8) NOT NULL,

  provincia VARCHAR(2) NOT NULL,

  telefono\_sede VARCHAR(16) NOT NULL CHECK (LENGTH(telefono\_sede) >= 9),

  CONSTRAINT pk\_sede PRIMARY KEY (citta, via\_piazza)

);

CREATE TABLE utenze (

  fattura VARCHAR(16) PRIMARY KEY,

  data\_scadenza DATE NOT NULL,

  tipo VARCHAR(64) NOT NULL,

  importo\_utenza NUMBER(7,2) NOT NULL CHECK( importo\_utenza > 0),

  pagamento\_utenza VARCHAR(16) NOT NULL CHECK (LOWER(pagamento\_utenza) IN ('pagato', 'non pagato')),

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  CONSTRAINT fk\_sede\_utenze FOREIGN KEY (citta, via\_piazza)

    REFERENCES sede (citta, via\_piazza) ON DELETE CASCADE

);

CREATE TABLE campo(

  id\_campo VARCHAR(8) PRIMARY KEY,

  disciplina VARCHAR(64) NOT NULL,

  prezzo\_orario NUMBER(5,2) NOT NULL CHECK(prezzo\_orario BETWEEN 50 AND 200),

  tipo\_terreno VARCHAR(64) NOT NULL,

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  CONSTRAINT fk\_sede\_cam FOREIGN KEY (citta,via\_piazza)

    REFERENCES sede (citta, via\_piazza) ON DELETE CASCADE

);

CREATE TABLE persona (

  cf VARCHAR(16) PRIMARY KEY CHECK(LENGTH(cf) > 15),

  nome VARCHAR(32) NOT NULL,

  cognome VARCHAR(32) NOT NULL,

  luogo\_nascita VARCHAR(64) NOT NULL,

  data\_nascita DATE NOT NULL

);

CREATE TABLE abbonato(

  codice\_abbonamento NUMBER PRIMARY KEY,

  cf VARCHAR(16),

  CONSTRAINT fk\_pers\_abb FOREIGN KEY (cf)

    REFERENCES persona (cf) ON DELETE CASCADE

);

CREATE TABLE personale (

  cf VARCHAR(16) PRIMARY KEY,

  stipendio NUMBER(7,2) NOT NULL CHECK (stipendio BETWEEN 800 AND 10000),

  mansione VARCHAR(64) NOT NULL,

  data\_assunzione DATE NOT NULL,

  data\_fr DATE,

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  CONSTRAINT fk\_pers\_prsnl FOREIGN KEY (cf)

    REFERENCES persona(cf),

  CONSTRAINT fk\_sede\_prsnl FOREIGN KEY (citta,via\_piazza)

    REFERENCES sede (citta,via\_piazza),

  CONSTRAINT chk\_date\_contratto CHECK ((data\_fr IS NULL) OR (data\_assunzione < data\_fr))

);

CREATE TABLE corso(

  nome\_corso VARCHAR(64),

  data\_inizio\_corso DATE,

  data\_fine\_corso DATE NOT NULL,

  quota\_mensile NUMBER(5,2) NOT NULL CHECK(quota\_mensile BETWEEN 20 AND 200),

  cf VARCHAR(16),

  id\_campo VARCHAR(8),

  CONSTRAINT pk\_corso PRIMARY KEY (nome\_corso, data\_inizio\_corso),

  CONSTRAINT fk\_istruttore FOREIGN KEY (cf)

    REFERENCES personale(cf),

  CONSTRAINT fk\_cam\_cor FOREIGN KEY (id\_campo)

    REFERENCES campo (id\_campo),

  CONSTRAINT chk\_date\_corso CHECK (data\_inizio\_corso < data\_fine\_corso)

);

CREATE TABLE iscrizione (

  codice\_abbonamento NUMBER,

  nome\_corso VARCHAR(64),

  data\_inizio\_corso DATE,

  data\_inizio\_abbonamento DATE NOT NULL,

  data\_fine\_abbonamento DATE NOT NULL,

  CONSTRAINT pk\_iscrizione PRIMARY KEY (codice\_abbonamento, nome\_corso, data\_inizio\_corso),

  CONSTRAINT fk\_abb\_iscr FOREIGN KEY (codice\_abbonamento)

    REFERENCES abbonato(codice\_abbonamento) ON DELETE CASCADE,

  CONSTRAINT fk\_cor\_iscr FOREIGN KEY (nome\_corso,data\_inizio\_corso)

    REFERENCES corso (nome\_corso, data\_inizio\_corso) ON DELETE CASCADE,

  CONSTRAINT chk\_date\_iscrizione CHECK (data\_inizio\_abbonamento <= ADD\_MONTHS(data\_fine\_abbonamento, -1))

);

CREATE TABLE prenotazione (

  id\_campo VARCHAR(8),

  data\_inizio\_attivita DATE,

  data\_fine\_attivita DATE NOT NULL,

  nominativo\_cliente VARCHAR(64) NOT NULL,

  telefono\_cliente VARCHAR(16) NOT NULL CHECK (LENGTH(telefono\_cliente) >= 9),

  CONSTRAINT pk\_pre PRIMARY KEY (id\_campo, data\_inizio\_attivita),

  CONSTRAINT fk\_cam\_pre FOREIGN KEY (id\_campo)

    REFERENCES campo(id\_campo) ON DELETE CASCADE

);

CREATE TABLE turno (

  codice\_turno VARCHAR (8) PRIMARY KEY,

  giorno VARCHAR (10) NOT NULL,

  ora\_inizio NUMBER(2) NOT NULL,

  ora\_fine NUMBER (2) NOT NULL,

  CONSTRAINT chk\_weekday CHECK (LOWER(giorno) IN ('domenica', 'lunedì', 'martedì', 'mercoledì', 'giovedì', 'venerdì', 'sabato')),

  CONSTRAINT chk\_hours CHECK (ora\_inizio >= 9 AND ora\_inizio <= 23 AND ora\_fine >= 9 AND ora\_fine <= 23 AND ora\_inizio < ora\_fine)

);

CREATE TABLE prodotto\_attrezzatura (

  codice\_barre VARCHAR(64) PRIMARY KEY CHECK(codice\_barre > 5),

  nome\_prodotto VARCHAR(64) NOT NULL,

  costo\_unitario NUMBER(8,2) NOT NULL CHECK(costo\_unitario > 0),

  prezzo\_vendita\_unitario NUMBER(8,2) CHECK(prezzo\_vendita\_unitario > 0)

);

CREATE TABLE fornitore (

  partita\_iva VARCHAR(11) PRIMARY KEY CHECK(LENGTH(partita\_iva) > 10),

  nome\_fornitore VARCHAR(64) NOT NULL,

  telefono\_fornitore VARCHAR(16) NOT NULL CHECK(LENGTH(telefono\_fornitore) >= 9)

);

CREATE TABLE ordine (

  partita\_iva VARCHAR(11),

  data\_richiesta DATE,

  data\_consegna DATE,

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  CONSTRAINT pk\_ordine PRIMARY KEY (partita\_iva, data\_richiesta),

  CONSTRAINT fk\_forn\_ord FOREIGN KEY (partita\_iva)

    REFERENCES fornitore (partita\_iva) ON DELETE CASCADE,

  CONSTRAINT fk\_sede\_ord FOREIGN KEY (citta,via\_piazza)

    REFERENCES sede(citta, via\_piazza) ON DELETE CASCADE,

  CONSTRAINT chk\_date\_ordine CHECK (data\_richiesta < data\_consegna)

);

CREATE TABLE include (

  partita\_iva VARCHAR(11),

  data\_richiesta DATE,

  codice\_barre VARCHAR(64),

  quantita\_acquistate NUMBER(8) NOT NULL CHECK(quantita\_acquistate > 0),

  CONSTRAINT pk\_include PRIMARY KEY (partita\_iva, data\_richiesta, codice\_barre),

  CONSTRAINT fk\_ord\_inc FOREIGN KEY (partita\_iva, data\_richiesta)

    REFERENCES ordine (partita\_iva, data\_richiesta) ON DELETE CASCADE,

  CONSTRAINT fk\_prd\_inc FOREIGN KEY (codice\_barre)

    REFERENCES prodotto\_attrezzatura (codice\_barre) ON DELETE CASCADE

);

CREATE TABLE lezione (

  codice\_turno VARCHAR(8),

  nome\_corso VARCHAR(64),

  data\_inizio\_corso DATE,

  CONSTRAINT pk\_lez PRIMARY KEY (codice\_turno, nome\_corso, data\_inizio\_corso),

  CONSTRAINT fk\_turno\_lez FOREIGN KEY (codice\_turno)

    REFERENCES turno (codice\_turno) ON DELETE CASCADE,

  CONSTRAINT fk\_cor\_lez FOREIGN KEY (nome\_corso, data\_inizio\_corso)

    REFERENCES corso (nome\_corso, data\_inizio\_corso) ON DELETE CASCADE

);

CREATE TABLE turnazione (

  cf VARCHAR(16),

  codice\_turno VARCHAR(8),

  CONSTRAINT pk\_trnzn PRIMARY KEY (cf,codice\_turno),

  CONSTRAINT fk\_prsnl\_trnzn FOREIGN KEY (cf)

    REFERENCES personale (cf) ON DELETE CASCADE,

  CONSTRAINT fk\_trnzn\_turno FOREIGN KEY (codice\_turno)

    REFERENCES turno (codice\_turno) ON DELETE CASCADE

);

CREATE TABLE vendita (

  id\_scontrino NUMBER PRIMARY KEY,

  data\_scontrino DATE DEFAULT CURRENT\_TIMESTAMP,

  citta VARCHAR(64),

  via\_piazza VARCHAR(128),

  CONSTRAINT fk\_sede\_vendita FOREIGN KEY (citta,via\_piazza)

    REFERENCES sede (citta, via\_piazza) ON DELETE CASCADE

);

CREATE TABLE vendita\_prodotto (

  id\_scontrino NUMBER,

  codice\_barre VARCHAR(64),

  quantita\_vendute NUMBER(8) NOT NULL CHECK(quantita\_vendute > 0),

  CONSTRAINT pk\_vp PRIMARY KEY (id\_scontrino,codice\_barre),

  CONSTRAINT fk\_vendita\_vp FOREIGN KEY (id\_scontrino)

    REFERENCES vendita (id\_scontrino) ON DELETE CASCADE,

  CONSTRAINT fk\_prd\_vp FOREIGN KEY (codice\_barre)

    REFERENCES prodotto\_attrezzatura (codice\_barre) ON DELETE CASCADE

);

-- Creazione sequenze per chiavi artificiali

CREATE SEQUENCE cod\_abb\_seq START WITH 1;

CREATE SEQUENCE id\_scontrino\_seq START WITH 1;

## 6.2 - Data Manipulation Language

*Per evitare dimensioni spropositate, vengono riportate solo alcune righe, del popolamento effettivo del DB (comunque accessibile attraverso i file .sql), a titolo esemplificativo:*

-- POPOLAMENTO ABBONATO

INSERT INTO abbonato (cf) VALUES ('KOFYYT17S38U638U');

INSERT INTO abbonato (cf) VALUES ('KOFYYT17S38U638U');

INSERT INTO abbonato (cf) VALUES ('NLHIXB62S32Y775K');

INSERT INTO abbonato (cf) VALUES ('JKYKSM74N83S975Q');

INSERT INTO abbonato (cf) VALUES ('KXLGXC56Y75P629J');

INSERT INTO abbonato (cf) VALUES ('ZEYICX55K63G324J');

INSERT INTO abbonato (cf) VALUES ('MWYJND99N34E339N');

INSERT INTO abbonato (cf) VALUES ('HXWLZA07N98T897X');

INSERT INTO abbonato (cf) VALUES ('MACPOG67H92G694Y');

INSERT INTO abbonato (cf) VALUES ('CWTUIO18G87D467T');

INSERT INTO abbonato (cf) VALUES ('MVQPMG17O50Z911K');

INSERT INTO abbonato (cf) VALUES ('XQILFH60T93K325T');

INSERT INTO abbonato (cf) VALUES ('HTXITV20A02P731J');

INSERT INTO abbonato (cf) VALUES ('EFYYUI98M60M985I');

INSERT INTO abbonato (cf) VALUES ('KXLGXC56Y75P629J');

INSERT INTO abbonato (cf) VALUES ('OKEASQ06Q16N152Y');

INSERT INTO abbonato (cf) VALUES ('NJCJKE60D13O802G');

INSERT INTO abbonato (cf) VALUES ('MWYJND99N34E339N');

INSERT INTO abbonato (cf) VALUES ('GVJHAU27Q32N837Y');

INSERT INTO abbonato (cf) VALUES ('NGMLDP73Z18B213P');

INSERT INTO abbonato (cf) VALUES ('GPVATN29S13Y054D');

INSERT INTO abbonato (cf) VALUES ('EPDJQO69D06Z640H');

INSERT INTO abbonato (cf) VALUES ('OSBTTP08I81O197K');

INSERT INTO abbonato (cf) VALUES ('XUWECR40J59J366M');

INSERT INTO abbonato (cf) VALUES ('AOXAGG69B07A485B');

INSERT INTO abbonato (cf) VALUES ('HTOLJO06B26T598N');

INSERT INTO abbonato (cf) VALUES ('NGMLDP73Z18B213P');

INSERT INTO abbonato (cf) VALUES ('HXHZJE80P66S207J');

INSERT INTO abbonato (cf) VALUES ('AOXAGG69B07A485B');

INSERT INTO abbonato (cf) VALUES ('KDTTCD26U39W036E');

INSERT INTO abbonato (cf) VALUES ('IGWSSK77H62B335M');

INSERT INTO abbonato (cf) VALUES ('WONDHG91B60R291V');

INSERT INTO abbonato (cf) VALUES ('VYCSDG06O09L210T');

INSERT INTO abbonato (cf) VALUES ('QXZUTJ60Y63J106D');

INSERT INTO abbonato (cf) VALUES ('EIVKNT21K28N459I');

-- POPOLAMENTO CAMPO

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('S-01', 'Judo e Karate', 100, 'Tatami', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('S-02', 'Judo e Karate', 60, 'Tatami', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-01', 'Calcetto', 60, 'Terreno', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-02', 'Calcetto', 60, 'Terreno', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-03', 'Calcetto', 50, 'Erba sintetica', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-04', 'Calcetto', 60, 'Sintetico', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('P-01', 'Pallavolo', 70, 'Sintetico', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('P-02', 'Pallavolo', 70, 'Sintetico', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-01', 'Basket', 70, 'Sintetico', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-02', 'Basket', 80, 'Parquet', 'Napoli', 'Corso Umberto');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('S-20', 'Judo e Karate', 90, 'Tatami', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('S-21', 'Judo e Karate', 80, 'Tatami', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-20', 'Basket', 70, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-21', 'Basket', 70, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-22', 'Basket', 90, 'Parquet', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('B-23', 'Basket', 90, 'Parquet', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('P-20', 'Pallavolo', 80, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('P-21', 'Pallavolo', 90, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('P-22', 'Pallavolo', 70, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-21', 'Calcetto', 70, 'Sintetico', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-20', 'Calcetto', 80, 'Erba sintetica', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-22', 'Calcetto', 80, 'Terreno', 'Milano', 'Piazza Duomo');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('S-40', 'Judo e Karate', 50, 'Tatami', 'Portici', 'Corso Garibaldi');

INSERT INTO campo (id\_campo, disciplina, prezzo\_orario, tipo\_terreno, citta, via\_piazza) VALUES ('C-40', 'Calcetto', 50, 'Erba sintetica', 'Portici', 'Corso Garibaldi');

-- POPOLAMENTO CORSO

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-jr-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'MVQPMG17O50Z911K','B-22');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-jr-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'VCHGSK89E29G238D','C-21');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('PALLAVOLO20-jr-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'LGRQSR76I22X607Q','P-22');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'MVQPMG17O50Z911K','B-22');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'VCHGSK89E29G238D','C-21');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('PALLAVOLO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'LGRQSR76I22X607Q','P-22');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-jr-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'GPVATN29S13Y054D','B-01');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-jr-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'CAKZQH40R12B796K','C-03');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('JUDO20-jr-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'HJEMQE85I87J391B','S-01');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'GPVATN29S13Y054D','B-01');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'CAKZQH40R12B796K','C-03');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('JUDO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'HJEMQE85I87J391B','S-01');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('KARATE20-jr-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'APRRPJ58E87T744L','S-40');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-jr-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'WONDHG91B60R291V','C-41');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-jr-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'WSJCQY00A64N338W','B-40');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('KARATE20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'APRRPJ58E87T744L','S-40');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('CALCIO20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'WONDHG91B60R291V','C-41');

INSERT INTO corso (nome\_corso, data\_inizio\_corso, data\_fine\_corso, quota\_mensile, cf, id\_campo) VALUES ('BASKET20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'),TO\_DATE('30/07/2020','dd/mm/yyyy'),50,'WSJCQY00A64N338W','B-40');

-- POPOLAMENTO FORNITORE

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('64859643436', 'Greenfelder and Sons', '080395498');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('74171634394', 'Schuster and Sons', '080474968');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('06801141467', 'Schultz and Sons', '085080131');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('96961378459', 'Lindgren LLC', '081936355');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('11303492335', 'O''Connell LLC', '084731054');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('49890484517', 'King, Johns and Johnson', '081432378');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('22893008666', 'Lockman-Swift', '080788761');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('90229628520', 'Hyatt Group', '084935904');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('83416970499', 'Sauer LLC', '084966338');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('66557816501', 'Donnelly-Blick', '088657823');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('68000188608', 'Schaefer-Breitenberg', '088902841');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('59320853867', 'Bernhard, Cummerata and Simonis', '083778332');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('24333780071', 'Buckridge LLC', '086945158');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('82137224488', 'Mayer, Marks and Stracke', '083784761');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('79435947012', 'Corwin Inc', '086889214');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('89464091226', 'Conroy-Bednar', '086997099');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('09085928022', 'Nolan-Johnston', '081528551');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('68535126960', 'Quigley-Morar', '080146267');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('58827778673', 'Abbott-Rohan', '081461131');

INSERT INTO fornitore (partita\_iva, nome\_fornitore, telefono\_fornitore) VALUES ('03839015489', 'Roberts and Sons', '080409125');

-- POPOLAMENTO INCLUDE

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('11303492335', TO\_DATE('20/10/2019','dd/mm/yyyy'), '8586004022', 49);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('22893008666', TO\_DATE('16/05/2019','dd/mm/yyyy'), '8603359913', 6);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('11303492335', TO\_DATE('04/06/2019','dd/mm/yyyy'), '8586004022', 42);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('11303492335', TO\_DATE('04/06/2019','dd/mm/yyyy'), '1564812969', 42);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('11303492335', TO\_DATE('04/06/2019','dd/mm/yyyy'), '5734168667', 8);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('90229628520', TO\_DATE('10/05/2019','dd/mm/yyyy'), '5775275022', 29);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('24333780071', TO\_DATE('06/06/2019','dd/mm/yyyy'), '8603359912', 21);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('64859643436', TO\_DATE('08/08/2019','dd/mm/yyyy'), '8603359912', 38);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('59320853867', TO\_DATE('21/04/2019','dd/mm/yyyy'), '8603359912', 24);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('22893008666', TO\_DATE('16/05/2019','dd/mm/yyyy'), '1564812969', 4);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('68000188608', TO\_DATE('31/03/2019','dd/mm/yyyy'), '7035223416', 9);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('89464091226', TO\_DATE('13/05/2019','dd/mm/yyyy'), '8586004022', 49);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('24333780071', TO\_DATE('24/04/2019','dd/mm/yyyy'), '5734168667', 5);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('68535126960', TO\_DATE('04/01/2019','dd/mm/yyyy'), '8603359913', 17);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('24333780071', TO\_DATE('24/04/2019','dd/mm/yyyy'), '5734168666', 32);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('82137224488', TO\_DATE('30/01/2019','dd/mm/yyyy'), '1564812969', 46);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('90229628520', TO\_DATE('02/09/2019','dd/mm/yyyy'), '5734168666', 34);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('74171634394', TO\_DATE('28/03/2019','dd/mm/yyyy'), '5734168667', 34);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('79435947012', TO\_DATE('27/07/2019','dd/mm/yyyy'), '8586004022', 21);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('24333780071', TO\_DATE('01/10/2019','dd/mm/yyyy'), '5189744043', 46);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('68535126960', TO\_DATE('22/09/2019','dd/mm/yyyy'), '8603359912', 31);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('82137224488', TO\_DATE('30/01/2019','dd/mm/yyyy'), '5734168666', 48);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('68000188608', TO\_DATE('04/04/2019','dd/mm/yyyy'), '5184572554', 4);

INSERT INTO include (partita\_iva, data\_richiesta, codice\_barre, quantita\_acquistate) VALUES ('89464091226', TO\_DATE('10/08/2019','dd/mm/yyyy'), '5184572554', 17);

-- POPOLAMENTO ISCRIZIONE

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (1, 'BASKET20-PO', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('01/01/2020' ,'dd/mm/yyyy'), TO\_DATE('01/02/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (2, 'BASKET20-jr-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('16/04/2020' ,'dd/mm/yyyy'), TO\_DATE('16/05/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (3, 'CALCIO20-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('11/01/2020' ,'dd/mm/yyyy'), TO\_DATE('11/02/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (4, 'BASKET20-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('26/04/2020' ,'dd/mm/yyyy'), TO\_DATE('26/05/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (5, 'CALCIO20-jr-PO', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('29/05/2020' ,'dd/mm/yyyy'), TO\_DATE('29/06/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (6, 'PALLAVOLO20-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('17/03/2020' ,'dd/mm/yyyy'), TO\_DATE('17/04/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (7, 'BASKET20-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('25/05/2020' ,'dd/mm/yyyy'), TO\_DATE('25/06/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (8, 'CALCIO20-jr-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('19/02/2020' ,'dd/mm/yyyy'), TO\_DATE('19/03/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (9, 'CALCIO20-jr-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('19/04/2020' ,'dd/mm/yyyy'), TO\_DATE('19/05/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (10, 'BASKET20-NA', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('30/03/2020' ,'dd/mm/yyyy'), TO\_DATE('30/04/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (11, 'JUDO20-jr-NA', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('27/05/2020' ,'dd/mm/yyyy'), TO\_DATE('27/06/2020' ,'dd/mm/yyyy'));

INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento) VALUES (12, 'BASKET20-jr-MI', TO\_DATE('01/01/2020','dd/mm/yyyy'), TO\_DATE('29/05/2020' ,'dd/mm/yyyy'), TO\_DATE('29/06/2020' ,'dd/mm/yyyy'));

-- POPOLAMENTO LEZIONE

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','BASKET20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','BASKET20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','BASKET20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','CALCIO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','CALCIO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','CALCIO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','PALLAVOLO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','PALLAVOLO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','PALLAVOLO20-MI',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','BASKET20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','BASKET20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','BASKET20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','CALCIO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','CALCIO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','CALCIO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','JUDO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','JUDO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','JUDO20-NA',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','BASKET20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','BASKET20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('V13','BASKET20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('L13','CALCIO20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'));

INSERT INTO lezione (codice\_turno, nome\_corso, data\_inizio\_corso) VALUES ('Me13','CALCIO20-PO',TO\_DATE('01/01/2020','dd/mm/yyyy'));

-- POPOLAMENTO ORDINE

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('24333780071', TO\_DATE('21/05/2019', 'dd/mm/yyyy'), TO\_DATE('11/06/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('90229628520', TO\_DATE('10/05/2019', 'dd/mm/yyyy'), TO\_DATE('01/06/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('90229628520', TO\_DATE('02/09/2019', 'dd/mm/yyyy'), TO\_DATE('03/09/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('96961378459', TO\_DATE('23/02/2019', 'dd/mm/yyyy'), TO\_DATE('21/03/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('74171634394', TO\_DATE('26/07/2019', 'dd/mm/yyyy'), TO\_DATE('02/08/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('74171634394', TO\_DATE('28/03/2019', 'dd/mm/yyyy'), TO\_DATE('21/04/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('03839015489', TO\_DATE('22/02/2019', 'dd/mm/yyyy'), TO\_DATE('21/03/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('68000188608', TO\_DATE('21/06/2019', 'dd/mm/yyyy'), TO\_DATE('21/07/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('68535126960', TO\_DATE('04/01/2019', 'dd/mm/yyyy'), TO\_DATE('21/01/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('68000188608', TO\_DATE('31/03/2019', 'dd/mm/yyyy'), TO\_DATE('21/04/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('68535126960', TO\_DATE('12/08/2019', 'dd/mm/yyyy'), TO\_DATE('21/08/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('83416970499', TO\_DATE('26/09/2019', 'dd/mm/yyyy'), TO\_DATE('21/10/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('68535126960', TO\_DATE('22/09/2019', 'dd/mm/yyyy'), TO\_DATE('21/10/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('79435947012', TO\_DATE('27/07/2019', 'dd/mm/yyyy'), TO\_DATE('21/08/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('11303492335', TO\_DATE('04/06/2019', 'dd/mm/yyyy'), TO\_DATE('21/06/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('03839015489', TO\_DATE('26/08/2019', 'dd/mm/yyyy'), TO\_DATE('21/09/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('74171634394', TO\_DATE('30/03/2019', 'dd/mm/yyyy'), TO\_DATE('21/04/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('22893008666', TO\_DATE('16/05/2019', 'dd/mm/yyyy'), TO\_DATE('21/05/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('74171634394', TO\_DATE('17/03/2019', 'dd/mm/yyyy'), TO\_DATE('21/03/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('82137224488', TO\_DATE('04/09/2019', 'dd/mm/yyyy'), TO\_DATE('21/09/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('24333780071', TO\_DATE('02/05/2019', 'dd/mm/yyyy'), TO\_DATE('21/05/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('59320853867', TO\_DATE('21/04/2019', 'dd/mm/yyyy'), TO\_DATE('21/05/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('24333780071', TO\_DATE('06/06/2019', 'dd/mm/yyyy'), TO\_DATE('21/06/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO ordine (partita\_iva, data\_richiesta, data\_consegna, citta, via\_piazza) VALUES ('24333780071', TO\_DATE('24/04/2019', 'dd/mm/yyyy'), TO\_DATE('21/05/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

-- POPOLAMENTO PERSONA

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('EVQXXV35O88Z845F', 'Vivyan', 'Tucsell', 'Xingfu', TO\_DATE('24/06/1992', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('PPXWDU31I04B263C', 'Juieta', 'Bernardino', 'Sorinomo', TO\_DATE('20/12/1947', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('PYPODQ34D32C117F', 'Randee', 'Salomon', 'Qizili', TO\_DATE('04/04/1951', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('NTSRUC90H06I924Y', 'Andy', 'Blackhurst', 'Nanxing', TO\_DATE('30/12/1960', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('EPDJQO69D06Z640H', 'Reinhard', 'Dedmam', 'København', TO\_DATE('17/02/1974', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('OSBTTP08I81O197K', 'Nathalie', 'Bolens', 'Hroznětín', TO\_DATE('10/08/1993', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('KPPRTH66L84J377O', 'Morena', 'Drayton', 'Dulyapino', TO\_DATE('02/08/2000', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('PKXSMJ51K19P701U', 'Ernest', 'Abarough', 'Rosh Pinna', TO\_DATE('29/09/2008', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('CPALLK07D82R196H', 'Orelie', 'Belson', 'Dundee', TO\_DATE('27/02/1970', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('WGLZTK42G48Y967X', 'Ingmar', 'Bouda', 'Asíni', TO\_DATE('02/02/1942', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('CGIUJV94K42C915D', 'Val', 'Gillbanks', 'Dallas', TO\_DATE('17/08/1950', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('JXQYCE43W47S283H', 'Barbi', 'Minot', 'Gaopi', TO\_DATE('01/08/1958', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('EFYYUI98M60M985I', 'Codi', 'Aitchinson', 'Panambi', TO\_DATE('29/07/2003', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('SXAZNH21T96D805U', 'Ronni', 'Isworth', 'Kadugannawa', TO\_DATE('13/03/1962', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('HCKYHW00M78W508U', 'Richard', 'McEntee', 'Hovd', TO\_DATE('19/08/1971', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('WXIGWA03J07T275J', 'Marylynne', 'Heims', 'Upi', TO\_DATE('23/07/1983', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('NJCJKE60D13O802G', 'Robbin', 'Basnall', 'Astghadzor', TO\_DATE('11/09/1979', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('HXWLZA07N98T897X', 'Roman', 'Guslon', 'Fenyan', TO\_DATE('31/10/1985', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('DSHSRL32Q69A666R', 'Romy', 'Vaskov', 'Aqaba', TO\_DATE('13/12/1989', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('KXLGXC56Y75P629J', 'Sam', 'Harsnipe', 'Esperanza', TO\_DATE('04/10/1963', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('UZGMWH06O34B499Q', 'Karie', 'Croke', 'Arys', TO\_DATE('06/08/1956', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('NGMLDP73Z18B213P', 'Deanne', 'Tempest', 'Kishi', TO\_DATE('02/01/1946', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('NMYMCN76J64E546E', 'Toma', 'Swenson', 'Karangpao', TO\_DATE('13/02/1954', 'dd/mm/yyyy'));

INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita) VALUES ('ANJUPQ16N94O866V', 'Ninon', 'Capps', 'Medvezh’yegorsk', TO\_DATE('29/09/1953', 'dd/mm/yyyy'));

-- POPOLAMENTO PERSONALE

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione)          VALUES ('YVQNLG13Y95L654M', 1800, 'Dirigente', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2017', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('KXLGXC56Y75P629J', 1500, 'Custode', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2010', 'dd/mm/yyyy'), TO\_DATE('29/09/2025', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('LHDWMA91Y89R464L', 1800, 'Bar', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2018', 'dd/mm/yyyy'), TO\_DATE('29/09/2031', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('HOZKHB25M69L179W', 1800, 'Segreteria', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2018', 'dd/mm/yyyy'), TO\_DATE('29/09/2031', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('MVQPMG17O50Z911K', 1200, 'Istruttore', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), TO\_DATE('29/09/2021', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('VCHGSK89E29G238D', 1500, 'Istruttore', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), TO\_DATE('29/09/2021', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('LGRQSR76I22X607Q', 1500, 'Istruttore', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), TO\_DATE('29/09/2021', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('EVQXXV35O88Z845F', 1500, 'Inserviente', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2018', 'dd/mm/yyyy'), TO\_DATE('29/09/2020', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('ANJUPQ16N94O866V', 1500, 'Inserviente', 'Milano', 'Piazza Duomo', TO\_DATE('10/10/2018', 'dd/mm/yyyy'), TO\_DATE('29/09/2020', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione)          VALUES ('WXIGWA03J07T275J', 1800, 'Dirigente', 'Napoli', 'Corso Umberto', TO\_DATE('10/10/2009', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('AOXAGG69B07A485B', 1500, 'Custode', 'Napoli', 'Corso Umberto', TO\_DATE('10/10/1976', 'dd/mm/yyyy'), TO\_DATE('29/09/2020', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('YYSAOA37L17T806M', 1500, 'Bar', 'Napoli', 'Corso Umberto', TO\_DATE('14/10/2014', 'dd/mm/yyyy'), TO\_DATE('29/09/2022', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('PYYJZM70U78S551D', 1500, 'Segreteria', 'Napoli', 'Corso Umberto', TO\_DATE('14/10/2014', 'dd/mm/yyyy'), TO\_DATE('29/09/2022', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('GPVATN29S13Y054D', 1200, 'Istruttore', 'Napoli', 'Corso Umberto', TO\_DATE('14/11/2012', 'dd/mm/yyyy'), TO\_DATE('19/03/2015', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('CAKZQH40R12B796K', 1000, 'Istruttore', 'Napoli', 'Corso Umberto', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), TO\_DATE('29/09/2021', 'dd/mm/yyyy'));

INSERT INTO personale (cf, stipendio, mansione, citta, via\_piazza, data\_assunzione, data\_fr) VALUES ('HJEMQE85I87J391B', 1000, 'Istruttore', 'Napoli', 'Corso Umberto', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), TO\_DATE('29/09/2021', 'dd/mm/yyyy'));

-- POPOLAMENTO PRENOTAZIONE

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('C-41', TO\_DATE('30/06/2020 17:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('30/06/2020 18:00', 'dd/mm/yyyy HH24:mi'), 'Cesaro Povey', '8282639040');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('P-01', TO\_DATE('11/04/2020 19:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('11/04/2020 20:00', 'dd/mm/yyyy HH24:mi'), 'Blondy Harper', '9662180639');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('S-40', TO\_DATE('27/12/2019 19:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('27/12/2019 20:00', 'dd/mm/yyyy HH24:mi'), 'Irvin Ivanov', '2686400155');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('S-02', TO\_DATE('01/02/2020 20:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('01/02/2020 21:00', 'dd/mm/yyyy HH24:mi'), 'Conny Bootland', '2996518791');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('P-22', TO\_DATE('07/05/2020 22:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('07/05/2020 23:00', 'dd/mm/yyyy HH24:mi'), 'Krysta Lambirth', '2274371665');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('B-20', TO\_DATE('17/06/2020 21:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('17/06/2020 22:00', 'dd/mm/yyyy HH24:mi'), 'Shaylynn Falck', '6562608133');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('P-02', TO\_DATE('08/07/2020 21:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('08/07/2020 22:00', 'dd/mm/yyyy HH24:mi'), 'Aloysius McNirlin', '2789355002');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('B-40', TO\_DATE('16/12/2019 18:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('16/12/2019 19:00', 'dd/mm/yyyy HH24:mi'), 'Orland Kebbell', '2794988443');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('B-41', TO\_DATE('20/04/2020 9:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('20/04/2020 10:00', 'dd/mm/yyyy HH24:mi'), 'Gussi Shafe', '5154994416');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('C-41', TO\_DATE('04/01/2020 13:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('04/01/2020 14:00', 'dd/mm/yyyy HH24:mi'), 'Ivar Boyles', '5745091913');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('C-43', TO\_DATE('23/09/2020 15:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('23/09/2020 16:00', 'dd/mm/yyyy HH24:mi'), 'Jenilee Adamczyk', '4984931589');

INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente, telefono\_cliente) VALUES ('B-21', TO\_DATE('04/01/2020 14:00', 'dd/mm/yyyy HH24:mi'), TO\_DATE('04/01/2020 15:00', 'dd/mm/yyyy HH24:mi'), 'Shanan Dwelly', '1805476082');

-- POPOLAMENTO PRODOTTO\_ATTREZZATURA

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5184572554', 'scala svedese', 50.33, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5616121125', 'tappetino fitness', 10.75, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5734168666', 'tavolo bar', 33.04, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5734168667', 'sedia bar', 13.04, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('1002473331', 'panca', 36.56, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5189744043', 'tabellone', 142.24, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('7819603166', 'canestro basket', 63.42, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('1564812969', 'sedile tribuna', 38.66, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('0027149555', 'porta calcetto', 50.02, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('7035223416', 'pallone basket', 16.83, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5775275022', 'pallone calcetto', 11.77, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8603359912', 'divisa squadra basket', 16.58, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8586004022', 'divisa squadra calcetto', 18.2, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8603359913', 'borsone squadra basket', 16.58, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8586004023', 'borsone squadra calcetto', 18.2, NULL);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('9744922467', 'Caviar - Salmon', 1.93, 5.76);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8127078013', 'Tequila - Sauza Silver', 4.5, 4.79);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('9881529773', 'Ecolab - Medallion', 2.02, 5.56);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('3540285966', 'Beans - Kidney, Canned', 4.17, 8.71);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('5493464449', 'Cup - 4oz Translucent', 2.93, 7.34);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('4912168396', 'Wine - Cahors Ac 2000, Clos', 1.26, 5.28);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('1679802686', 'Lid - 10,12,16 Oz', 4.72, 6.75);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('7466532744', 'Olives - Kalamata', 2.17, 7.48);

INSERT INTO prodotto\_attrezzatura (codice\_barre, nome\_prodotto, costo\_unitario, prezzo\_vendita\_unitario) VALUES ('8961329850', 'Cheese - Brick With Pepper', 0.4, 0.76);

-- POPOLAMENTO SEDE

INSERT INTO sede (citta, via\_piazza, civico, provincia, telefono\_sede) VALUES ('Napoli', 'Corso Umberto', '10', 'NA', '081565721');

INSERT INTO sede (citta, via\_piazza, civico, provincia, telefono\_sede) VALUES ('Milano', 'Piazza Duomo', '3', 'MI', '086765428');

INSERT INTO sede (citta, via\_piazza, civico, provincia, telefono\_sede) VALUES ('Portici', 'Corso Garibaldi', '151', 'NA', '0817754321');

-- POPOLAMENTO TURNAZIONE

INSERT INTO turnazione (cf, codice\_turno) VALUES ('APRRPJ58E87T744L', 'S02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('UDJISE18C58B844F', 'M01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CGIUJV94K42C915D', 'S03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'D03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WSJCQY00A64N338W', 'M03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'S01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WSJCQY00A64N338W', 'V03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('ALAPYD31J47A233B', 'D02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CGIUJV94K42C915D', 'G03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WONDHG91B60R291V', 'S02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CGIUJV94K42C915D', 'M01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WONDHG91B60R291V', 'V02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'M03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CGIUJV94K42C915D', 'V03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'V03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WSJCQY00A64N338W', 'M01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'L01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'M01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'S03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('UDJISE18C58B844F', 'D02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('UDJISE18C58B844F', 'G01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'M02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'D02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'M03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('NJCJKE60D13O802G', 'L02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'D03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('NJCJKE60D13O802G', 'L01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WONDHG91B60R291V', 'L02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WSJCQY00A64N338W', 'S01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'D01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('ALAPYD31J47A233B', 'Me01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('WONDHG91B60R291V', 'M03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'Me01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CGIUJV94K42C915D', 'L02');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('APRRPJ58E87T744L', 'M01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('IKPYMQ51F96Y769N', 'G01');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('CWTUIO18G87D467T', 'S03');

INSERT INTO turnazione (cf, codice\_turno) VALUES ('UDJISE18C58B844F', 'V01');

-- POPOLAMENTO TURNO

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L01','Lunedì',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L02','Lunedì',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L03','Lunedì',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L11','Lunedì',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L12','Lunedì',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('L13','Lunedì',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M01','Martedì',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M02','Martedì',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M03','Martedì',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M11','Martedì',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M12','Martedì',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('M13','Martedì',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me01','Mercoledì',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me02','Mercoledì',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me03','Mercoledì',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me11','Mercoledì',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me12','Mercoledì',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('Me13','Mercoledì',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G01','Giovedì',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G02','Giovedì',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G03','Giovedì',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G11','Giovedì',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G12','Giovedì',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('G13','Giovedì',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V01','Venerdì',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V02','Venerdì',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V03','Venerdì',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V11','Venerdì',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V12','Venerdì',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('V13','Venerdì',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S01','Sabato',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S02','Sabato',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S03','Sabato',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S11','Sabato',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S12','Sabato',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('S13','Sabato',22,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D01','Domenica',9,14);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D02','Domenica',14,20);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D03','Domenica',20,23);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D11','Domenica',11,12);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D12','Domenica',18,19);

INSERT INTO turno (codice\_turno, giorno, ora\_inizio, ora\_fine) VALUES ('D13','Domenica',22,23);

-- POPOLAMENTO UTENZE

-- PAGATE

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('9014484171', TO\_DATE('22/03/2019', 'dd/mm/yyyy'), 'Luce', 113.64, 'pagato', 'Milano', 'Piazza Duomo');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('3160728976', TO\_DATE('30/07/2018', 'dd/mm/yyyy'), 'Luce', 380.44, 'pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('1298619950', TO\_DATE('21/03/2018', 'dd/mm/yyyy'), 'Gas', 798.55, 'pagato', 'Napoli', 'Corso Umberto');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('9263859521', TO\_DATE('17/08/2018', 'dd/mm/yyyy'), 'Luce', 704.03, 'pagato', 'Napoli', 'Corso Umberto');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('8522078647', TO\_DATE('19/12/2018', 'dd/mm/yyyy'), 'Telefono e Internet', 224.9, 'pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('2992352022', TO\_DATE('02/08/2019', 'dd/mm/yyyy'), 'Luce', 939.06, 'pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('3436221607', TO\_DATE('20/04/2018', 'dd/mm/yyyy'), 'Telefono e Internet', 678.98, 'pagato', 'Milano', 'Piazza Duomo');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('7608929413', TO\_DATE('13/07/2019', 'dd/mm/yyyy'), 'Gas', 477.08, 'pagato', 'Napoli', 'Corso Umberto');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('6003114722', TO\_DATE('18/06/2019', 'dd/mm/yyyy'), 'Gas', 413.78, 'pagato', 'Milano', 'Piazza Duomo');

-- NON PAGATE

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('9577539200', TO\_DATE('08/11/2019', 'dd/mm/yyyy'), 'Gas', 1000.13, 'non pagato', 'Milano', 'Piazza Duomo');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('4205386530', TO\_DATE('06/11/2019', 'dd/mm/yyyy'), 'Acqua', 813.36, 'non pagato', 'Napoli', 'Corso Umberto');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('5729597400', TO\_DATE('11/10/2019', 'dd/mm/yyyy'), 'Luce', 717.6, 'non pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('5729597402', TO\_DATE('11/11/2019', 'dd/mm/yyyy'), 'Gas', 717.6, 'non pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('7163918880', TO\_DATE('10/10/2019', 'dd/mm/yyyy'), 'Gas', 496.88, 'non pagato', 'Napoli', 'Corso Umberto');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('0970473916', TO\_DATE('10/11/2019', 'dd/mm/yyyy'), 'Telefono e Internet', 118.09, 'non pagato', 'Portici', 'Corso Garibaldi');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('8549212460', TO\_DATE('26/11/2019', 'dd/mm/yyyy'), 'Telefono e Internet', 440.63, 'non pagato', 'Milano', 'Piazza Duomo');

INSERT INTO utenze (fattura, data\_scadenza, tipo, importo\_utenza, pagamento\_utenza, citta, via\_piazza) VALUES ('9577539201', TO\_DATE('08/11/2019', 'dd/mm/yyyy'), 'Luce', 1000.13, 'non pagato', 'Milano', 'Piazza Duomo');

-- POPOLAMENTO VENDITA

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('30/07/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('13/09/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('02/12/2018', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('24/12/2018', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('15/04/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('07/07/2018', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('29/09/2018', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('20/01/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('28/11/2018', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('06/05/2018', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('30/05/2018', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('20/09/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('24/08/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('19/02/2018', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('23/04/2019', 'dd/mm/yyyy'), 'Napoli', 'Corso Umberto');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('14/03/2019', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('06/05/2018', 'dd/mm/yyyy'), 'Portici', 'Corso Garibaldi');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('28/12/2018', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

INSERT INTO vendita (data\_scontrino, citta, via\_piazza) VALUES (TO\_DATE('31/03/2019', 'dd/mm/yyyy'), 'Milano', 'Piazza Duomo');

-- POPOLAMENTO VENDITA\_PRODOTTO

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (27, '1108756768', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (14, '4518810352', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (19, '4204182439', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (5, '4755391358', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (9, '6334613044', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (45, '4912168396', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (26, '1679802686', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (8, '6215779754', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (45, '4492924103', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (19, '6957247219', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (45, '8317983568', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (22, '9979747282', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (8, '9744922467', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (19, '3574906044', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (29, '8999374580', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (31, '0539388229', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (34, '6867834376', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (40, '4518810352', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (44, '5373049315', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (39, '7499938906', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (29, '4518810352', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (29, '2630971287', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (27, '6334613044', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (50, '3574906044', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (29, '0946243893', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (33, '7499938906', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (10, '5555975470', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (24, '3976397678', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (29, '5160276869', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (21, '2247995796', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (33, '1093329176', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (14, '6105477377', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (45, '8398042351', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (25, '1427136722', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (49, '5035630371', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (13, '5046638114', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (2, '4100226330', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (13, '4755391358', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (33, '5516348160', 3);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (30, '5046638114', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (22, '5493464449', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (33, '7220212366', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (30, '1261743358', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (11, '4492924103', 4);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (7, '7710548912', 1);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (45, '5555975470', 5);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (47, '7732968889', 2);

INSERT INTO vendita\_prodotto (id\_scontrino, codice\_barre, quantita\_vendute) VALUES (15, '8410844025', 3);

## 6.3 - Triggers

I trigger DML sono utili principalmente per il controllo di vincoli dinamici in fase di immissione,

cancellazione o aggiornamento dei dati. È possibile talvoltautilizzarli anche per implementare

*regole di business* o per calcolare, generare o sovrascrivere il valore di alcuni campi.

**Auto-incremento codice abbonamento**

Questo trigger ha lo scopo di incrementare il valore della sequenza cod\_abb\_seq per poterlo poi assegnare, come codice\_abbonamento, nel momento della registrazione, di un nuovo abbonato.

CREATE OR REPLACE TRIGGER auto\_increm\_cod\_abb

BEFORE INSERT ON abbonato

FOR EACH ROW

BEGIN

  SELECT cod\_abb\_seq.NEXTVAL

  INTO :NEW.codice\_abbonamento

  FROM DUAL;

END;

/

**Auto-incremento id scontrino**

Questo trigger ha lo scopo di incrementare il valore della sequenza id\_scontrino\_seq per poterlo poi assegnare, come id\_scontrino, nel momento della registrazione, di una nuova vendita.

CREATE OR REPLACE TRIGGER auto\_increm\_id\_scontr

BEFORE INSERT ON vendita

FOR EACH ROW

BEGIN

  SELECT id\_scontrino\_seq.NEXTVAL

  INTO :NEW.id\_scontrino

  FROM DUAL;

END;

/

**Durata prenotazione**

Questo trigger controlla che, nel tentativo di inserimento o di aggiornamento di una prenotazione, questa non occupi il campo per più di quattro ore o meno di un’ora (regola di business). Nel caso, lancia l’eccezione.

CREATE OR REPLACE TRIGGER durata\_prenotazione

BEFORE INSERT OR UPDATE ON prenotazione

FOR EACH ROW

DECLARE

  errore\_durata\_prenotazione EXCEPTION;

BEGIN

  IF( (:NEW.data\_inizio\_attivita + INTERVAL '4' HOUR) < :NEW.data\_fine\_attivita OR

   (:NEW.data\_fine\_attivita - INTERVAL '1' HOUR) < :NEW.data\_inizio\_attivita ) THEN

        RAISE errore\_durata\_prenotazione;

    END IF;

EXCEPTION

WHEN errore\_durata\_prenotazione THEN

    RAISE\_APPLICATION\_ERROR('-20005','ERRORE DURATA ATTIVITA: il campo non può essere occupato per più di 4 ore o meno di 1 ora!');

END;

/

**Età persona**

Questo trigger viene attivato, all’inserimento o modifica dei dati, di una persona e verifica che non abbia un’età minore di sei anni o maggiore di 100 anni (regola di business).  
In caso contrario viene mostrato un messaggio d’errore.

CREATE OR REPLACE TRIGGER eta\_persona

BEFORE INSERT OR UPDATE ON PERSONA

FOR EACH ROW

DECLARE

    errore\_eta\_6 EXCEPTION;

    errore\_eta\_100 EXCEPTION;

BEGIN

    IF(MONTHS\_BETWEEN(SYSDATE,:NEW.data\_nascita)<72) THEN

        RAISE errore\_eta\_6;

    ELSIF (MONTHS\_BETWEEN(SYSDATE,:NEW.data\_nascita)>1200) THEN

        RAISE errore\_eta\_100;

    END IF;

EXCEPTION

WHEN errore\_eta\_6 THEN

    RAISE\_APPLICATION\_ERROR('-20001','ERRORE ETA: la persona ha meno di 6 anni!');

WHEN errore\_eta\_100 THEN

    RAISE\_APPLICATION\_ERROR('-20002','ERRORE ETA: la persona ha più di 100 anni!');

END;

/

**Iscrizione in corso**

Questo trigger ha lo scopo di verificare, che l’abbonamento che sta per essere sottoscritto o modificato, sia valido, entro le date, di inizio e fine del corso. In caso contrario viene mostrato un messaggio d’errore.

CREATE OR REPLACE TRIGGER iscrizione\_in\_corso

BEFORE INSERT OR UPDATE ON iscrizione

FOR EACH ROW

DECLARE

  data\_ic DATE;

  data\_fc DATE;

  errore\_iscrizione\_in\_corso EXCEPTION;

BEGIN

  /\* seleziona le date di inizio e fine

  del corso a cui si sta iscrivendo l'abbonato \*/

  SELECT data\_inizio\_corso, data\_fine\_corso INTO data\_ic, data\_fc

  FROM corso

  WHERE nome\_corso = :NEW.nome\_corso AND data\_inizio\_corso = :NEW.data\_inizio\_corso;

 /\* se il periodo di validità del nuovo abbonamento non rientra nelle suddette date

  lancia eccezione \*/

  IF (:NEW.data\_inizio\_abbonamento NOT BETWEEN data\_ic AND data\_fc

     OR

     :NEW.data\_fine\_abbonamento NOT BETWEEN data\_ic AND data\_fc) THEN

    RAISE errore\_iscrizione\_in\_corso;

  END IF;

EXCEPTION

WHEN errore\_iscrizione\_in\_corso THEN

  RAISE\_APPLICATION\_ERROR('-20008','ERRORE DATE ISCRIZIONE AL CORSO: Le date dell iscrizione non rientrano nella durata del corso!');

END;/

**Massime iscrizioni al corso**

Questo trigger viene attivato nel momento in cui, una persona si iscrive ad un corso, e verifica, che non venga superato il limite massimo di iscritti per quest’ultimo (trenta persone – regola di business).

CREATE OR REPLACE TRIGGER max\_iscritti\_corso

BEFORE INSERT OR UPDATE ON iscrizione

FOR EACH ROW

DECLARE

    contatore NUMBER;

    errore\_max\_iscritti EXCEPTION;

BEGIN

    /\* conta gli iscritti al corso,

    con abbonamento ancora valido \*/

    SELECT COUNT(\*) INTO contatore

    FROM iscrizione

    WHERE nome\_corso = :NEW.nome\_corso AND

    data\_inizio\_corso = :NEW.data\_inizio\_corso AND

    data\_fine\_abbonamento > :NEW.data\_inizio\_abbonamento;

    -- se maggiore di 30, lancia eccezione

    IF(contatore >= 30) THEN

        RAISE errore\_max\_iscritti;

    END IF;

EXCEPTION

WHEN errore\_max\_iscritti THEN

    RAISE\_APPLICATION\_ERROR('-20007','ERRORE ETA: Il corso non può avere più di 30 iscritti!');

END;

/

**Massime ore lavorative giornaliere**

Questo trigger tutela i dipendenti, dal superare il limite di ore lavorative giornaliere (tredici ore comprendendo i possibili straordinari).  
Il trigger viene attivato, nel momento in cui, viene assegnato un turno ad un dipendente: controlla la quantità di ore assegnategli già in precedenza, e quindi, se aggiungendo le nuove ore di lavoro si supera il suddetto limite, le modifiche non vengono apportate e viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER max\_ore\_gg\_personale

BEFORE INSERT OR UPDATE ON turnazione

FOR EACH ROW

DECLARE

  ore\_tot NUMBER;

  ore\_new NUMBER;

  errore\_max\_ore\_gg EXCEPTION;

BEGIN

  -- somma le ore di lavoro, già accumulate nella giornata, dal lavoratore

  SELECT SUM(turno.ora\_fine - turno.ora\_inizio) INTO ore\_tot

  FROM turno

  NATURAL JOIN turnazione

  WHERE turnazione.cf = :NEW.cf AND turno.giorno = (SELECT giorno

                                          FROM turno

                                          WHERE codice\_turno = :NEW.codice\_turno);

  -- seleziona le ore di lavoro che si stanno aggiungendo

  SELECT (ora\_fine - ora\_inizio) INTO ore\_new

  FROM turno

  WHERE codice\_turno = :NEW.codice\_turno;

  -- se la somma delle due è maggiore di 13, lancia eccezione

  IF( (ore\_tot + ore\_new) > 13) THEN

    RAISE errore\_max\_ore\_gg;

  END IF;

EXCEPTION

WHEN errore\_max\_ore\_gg THEN

  RAISE\_APPLICATION\_ERROR('-20010','ERRORE MAX ORE GG: Il personale non può lavorare più di 13 ore al giorno!');

END;

/

**Massime ore lavorative settimanali**

Questo trigger tutela i dipendenti, dal superare il limite di ore lavorative settimanali (quarantotto ore comprendendo i possibili straordinari).  
Il funzionamento è simile a quello visto in precedenza.

CREATE OR REPLACE TRIGGER max\_ore\_personale

BEFORE INSERT OR UPDATE ON turnazione

FOR EACH ROW

DECLARE

    ore\_tot NUMBER;

    ore\_new NUMBER;

    errore\_max\_ore EXCEPTION;

BEGIN

    -- somma le ore di lavoro, già accumulate nella settimana, dal lavoratore

    SELECT SUM(turno.ora\_fine - turno.ora\_inizio) INTO ore\_tot

    FROM turno

    NATURAL JOIN turnazione

    WHERE turnazione.cf = :NEW.cf;

    -- seleziona le ore di lavoro che si stanno aggiungendo

    SELECT (ora\_fine - ora\_inizio) INTO ore\_new

    FROM turno

    WHERE codice\_turno = :NEW.codice\_turno;

    -- se la somma delle due è maggiore di 48, lancia eccezione

    IF( (ore\_tot + ore\_new) > 48) THEN

        RAISE errore\_max\_ore;

    END IF;

EXCEPTION

WHEN errore\_max\_ore THEN

    RAISE\_APPLICATION\_ERROR('-20009','ERRORE MAX ORE SETTIMANALI: Il personale non può lavorare più di 48 ore a settimana!');

END;

/

**Massimo personale per sede**

Questo trigger controlla che ogni sede, abbia al più, cinquanta componenti nello staff (istruttori compresi).  
Il trigger si attiva all’inserimento di un nuovo dipendente, e se il personale è già al completo per quella sede, questo non viene inserito e viene stampato un messaggio.

CREATE OR REPLACE TRIGGER max\_personale\_sede

BEFORE INSERT ON personale

FOR EACH ROW

DECLARE

    contatore NUMBER;

    errore\_max\_personale EXCEPTION;

BEGIN

    -- conta il personale della sede

    SELECT COUNT(\*) INTO contatore

    FROM personale

    WHERE citta = :NEW.citta AND via\_piazza = :NEW.via\_piazza;

    -- se è maggiore di 50, lancia eccezione

    IF(contatore >= 50) THEN

        RAISE errore\_max\_personale;

    END IF;

EXCEPTION

WHEN errore\_max\_personale THEN

    RAISE\_APPLICATION\_ERROR('-20006','ERRORE ETA: La sede non può avere più di 50 persone nello staff!');

END;

/

**Massime quantità vendute**

Questo trigger controlla che non vengano venduti prodotti non disponibili in magazzino.  
Calcola le quantità acquistate, di uno specifico prodotto, tramite ordini ai fornitori, nella variabile tot\_acq; mentre le vendite effettuate in precedenza, dello stesso prodotto, in tot\_ven.  
Successivamente vengono sommate quest’ultima e le (ipotetiche) quantità appena vendute: se si supera il valore in tot\_acq, la vendita non viene effettuata e viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER max\_prodotti\_venduti

BEFORE INSERT ON vendita\_prodotto

FOR EACH ROW

DECLARE

    tot\_acq NUMBER;

    tot\_ven NUMBER;

    errore\_max\_prodotti\_venduti EXCEPTION;

BEGIN

    -- calcola le quantità acquistate del prodotto

    SELECT SUM(quantita\_acquistate) INTO tot\_acq

    FROM include

    WHERE codice\_barre = :NEW.codice\_barre;

    -- calcola le quantità già vendute del prodotto

    SELECT SUM(quantita\_vendute) INTO tot\_ven

    FROM vendita\_prodotto

    WHERE codice\_barre = :NEW.codice\_barre;

    /\* se la somma delle quantità già vendute e appena vendute

    è maggiore di quella acquistata, lancia eccezione \*/

    IF(tot\_ven + :NEW.quantita\_vendute >= tot\_acq) THEN

        RAISE errore\_max\_prodotti\_venduti;

    END IF;

EXCEPTION

WHEN errore\_max\_prodotti\_venduti THEN

    RAISE\_APPLICATION\_ERROR('-20015','ERRORE VENDITA: Quantità prodotto non disponibile in magazzino o non registrata!');

END;

/

**Personale futuro**

Questo trigger previene gli errori di inserimento o modifica, della data di assunzione, di un addetto al personale. Difatti, se questa supera la data odierna, viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER personale\_futuro

BEFORE INSERT OR UPDATE ON personale

FOR EACH ROW

DECLARE

    errore\_personale\_futuro EXCEPTION;

BEGIN

    IF(:NEW.data\_assunzione > SYSDATE) THEN

        RAISE errore\_personale\_futuro;

    END IF;

EXCEPTION

WHEN errore\_personale\_futuro THEN

    RAISE\_APPLICATION\_ERROR('-20016','ERRORE DATA ASSUNZIONE PERSONALE: La data di assunzione di questo membro dello staff è nel futuro!');

END;

/

**Personale maggiorenne**

Il trigger previene l’inserimento di un nuovo addetto al personale se è minorenne; in tal caso viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER personale\_maggiorenne

BEFORE INSERT OR UPDATE ON personale

FOR EACH ROW

DECLARE

    data\_nascita\_p DATE;

    data\_assunzione\_p DATE;

    errore\_eta\_p EXCEPTION;

BEGIN

    -- seleziona data di nascita e assunzione del lavoratore

    SELECT data\_nascita, :NEW.data\_assunzione INTO data\_nascita\_p,data\_assunzione\_p

    FROM persona

    WHERE cf = :NEW.cf;

    -- se la differenza tra le due è minore di 18 anni (216 mesi), lancia eccezione

    IF(MONTHS\_BETWEEN(data\_assunzione\_p, data\_nascita\_p)<216) THEN

        RAISE errore\_eta\_p;

    END IF;

EXCEPTION

WHEN errore\_eta\_p THEN

    RAISE\_APPLICATION\_ERROR('-20003','ERRORE ETA: Il personale non può essere minorenne al momento dell assunzione!');

END;

/

**Sovra-iscrizioni**

Questo trigger previene eventuali errori nei rinnovi degli abbonamenti (inserimento di un nuovo abbonamento ma legato alla stessa persona): se i periodi di validità, del vecchio e del nuovo abbonamento si sovrappongono, viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER sovra\_iscrizioni

BEFORE INSERT ON iscrizione

FOR EACH ROW

DECLARE

  verifica NUMBER;

  errore\_periodi\_iscrizioni EXCEPTION;

  cod\_fis VARCHAR(16);

BEGIN

  /\* seleziona il codice fiscale della persona che si sta iscrivendo

  (non è presente in iscrizione, ma in abbonato) \*/

  SELECT cf INTO cod\_fis

  FROM abbonato

  WHERE codice\_abbonamento = :NEW.codice\_abbonamento;

  /\* verifica se la persona possiede già, un abbonamento al corso valido,

  ovvero, se le date del preesistente e del nuovo abbonamento si sovrappongono \*/

  SELECT COUNT (cf) INTO verifica

  FROM iscrizione

  NATURAL JOIN abbonato

  WHERE cf = cod\_fis AND

  nome\_corso = :NEW.nome\_corso AND

  data\_inizio\_corso = :NEW.data\_inizio\_corso AND (

    data\_inizio\_abbonamento BETWEEN :NEW.data\_inizio\_abbonamento AND :NEW.data\_fine\_abbonamento

    OR

    data\_fine\_abbonamento BETWEEN :NEW.data\_inizio\_abbonamento AND :NEW.data\_fine\_abbonamento

  );

  -- nel caso, lancia eccezione

  IF (verifica > 0) THEN

    RAISE errore\_periodi\_iscrizioni;

  END IF;

EXCEPTION

WHEN errore\_periodi\_iscrizioni THEN  
    RAISE\_APPLICATION\_ERROR('-20011','ERRORE DATE ISCRIZIONE AL CORSO: Il tesserato risulta già iscritto, a questo corso, in questo periodo!');

END;

/

**Sovrapposizioni corsi**

Questo trigger previene l’inserimento di un nuovo corso, le cui lezioni si svolgono nello stesso giorno, orario e campo, di quelle di altri corsi.

CREATE OR REPLACE TRIGGER sovrapposizione\_corsi

BEFORE INSERT ON lezione

FOR EACH ROW

DECLARE

  verifica NUMBER;

  errore\_corso\_sovrapposto EXCEPTION;

BEGIN

/\* verifica se sullo stesso campo, nello stesso giorno e nella stessa fascia  oraria, del nuovo corso, se ne svolge già uno \*/

   SELECT COUNT (\*) INTO verifica

   FROM (-- seleziona giorno e fasce orarie, dei corsi che si sovrappongono, a quelle del  corso nuovo

            SELECT giorno,ora\_inizio,ora\_fine

            FROM (-- seleziona tutti i corsi che si svolgono sullo stesso campo, di quello nuovo

                  SELECT \*

                  FROM corso

                  NATURAL JOIN lezione

                  NATURAL JOIN turno

                  WHERE id\_campo = (-- campo del nuovo corso

                                    SELECT id\_campo

                                    FROM corso

                                    WHERE nome\_corso = :NEW.nome\_corso AND

data\_inizio\_corso = :NEW.data\_inizio\_corso

                                    )

                  )

            WHERE giorno = (-- giorno della nuova lezione

                              SELECT giorno

                              FROM turno

                              WHERE codice\_turno = :NEW.codice\_turno) AND

                              -- l'ora di inizio turno è tra:

                              (ora\_inizio BETWEEN(

-- ora inizio e ora fine della lezione del nuovo corso

                                          SELECT ora\_inizio

                                          FROM turno

                                          WHERE codice\_turno = :NEW.codice\_turno)  AND

                                          (SELECT ora\_fine

                                          FROM turno

                                          WHERE codice\_turno = :NEW.codice\_turno)

                              -- oppure l'ora di fine turno è tra:

                              OR ora\_fine BETWEEN

                                          (

-- ora inizio e ora fine della lezione del nuovo corso

                                          SELECT ora\_inizio

                                          FROM turno

                                          WHERE codice\_turno = :NEW.codice\_turno)  AND

                                          (SELECT ora\_fine

                                          FROM turno

                                          WHERE codice\_turno = :NEW.codice\_turno)

                              )

            );

      -- in caso affermativo, lancia eccezione

      IF (verifica > 0) THEN

            RAISE errore\_corso\_sovrapposto;

      END IF;

EXCEPTION

WHEN errore\_corso\_sovrapposto THEN

      RAISE\_APPLICATION\_ERROR('-20013','ERRORE CORSO: Questo campo è già occuppato da un corso nella fascia oraria indicata!');

END;

/

**Sovrapposizioni prenotazioni a corsi**

Da politica aziendale, per l’occupazione di un campo, si dà maggiore priorità ai corsi, rispetto alle prenotazioni telefoniche e questo trigger previene tale sovrapposizione.  
Difatti, nel momento in cui viene effettuata una prenotazione, vengono interrogate le tabelle corso, lezione e turno, per verificare che nella fascia oraria richiesta, il campo non sia già occupato da un corso. In caso contrario viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER sovrapposizione\_pre\_corso

BEFORE INSERT ON prenotazione

FOR EACH ROW

DECLARE

  verifica NUMBER;

  pre\_sovrapposta\_corso EXCEPTION;

BEGIN

      /\* verifica se sullo stesso campo, nello stesso giorno e nella stessa fascia  oraria, della nuova prenotazione, si svolge già un corso \*/

  SELECT COUNT (\*) INTO verifica

  FROM(

-- seleziona giorno e fasce orarie,

dei corsi che si svolgono sul campo della nuova prenotazione

       SELECT giorno,ora\_inizio,ora\_fine

       FROM (SELECT \*

             FROM corso

             NATURAL JOIN lezione

             NATURAL JOIN turno

             WHERE id\_campo = :NEW.id\_campo

             )

            WHERE giorno = -- giorno della prenotazione

                              TO\_CHAR(:NEW.data\_inizio\_attivita, 'Day') AND

--fasce orarie delle lezioni che si sovrappongono

  a quelle della nuova prenotazione

    (ora\_inizio BETWEEN TO\_NUMBER(TO\_CHAR(:NEW.data\_inizio\_attivita,'HH24,MI')) AND

                              TO\_NUMBER(TO\_CHAR(:NEW.data\_fine\_attivita,'HH24,MI'))

    OR ora\_fine BETWEEN TO\_NUMBER(TO\_CHAR(:NEW.data\_inizio\_attivita,'HH24,MI')) AND

                              TO\_NUMBER(TO\_CHAR(:NEW.data\_fine\_attivita,'HH24,MI'))

            )

  );

  -- in caso affermativo, lancia eccezione

  IF (verifica > 0) THEN

     RAISE pre\_sovrapposta\_corso;

  END IF;

EXCEPTION

WHEN pre\_sovrapposta\_corso THEN

      RAISE\_APPLICATION\_ERROR('-20014','ERRORE PRENOTAZIONE: Questo campo è già occuppato da un corso nella fascia oraria indicata!');

END;

/

**Sovrapposizioni prenotazioni a prenotazioni**

Concettualmente simile al precedente, questo trigger, previene le sovrapposizioni tra prenotazioni (stesso campo, fasce orarie sovrapposte).

CREATE OR REPLACE TRIGGER sovrapposizione\_prenotazioni

BEFORE INSERT OR UPDATE ON prenotazione

FOR EACH ROW

DECLARE

  verifica NUMBER;

  prenot\_sovrapposta EXCEPTION;

BEGIN

  /\* verifica se il campo è già prenotato nello stesso periodo \*/

  SELECT COUNT (\*) INTO verifica

  FROM prenotazione

  WHERE id\_campo = :NEW.id\_campo AND

    -- date e ore della prenotazione sovrapposte a quelle nuove

(data\_inizio\_attivita BETWEEN :NEW.data\_inizio\_attivita AND :NEW.data\_fine\_attivita

    OR

data\_fine\_attivita BETWEEN :NEW.data\_inizio\_attivita AND :NEW.data\_fine\_attivita

);

  -- in caso affermativo, lancia eccezione

  IF (verifica > 0) THEN

    RAISE prenot\_sovrapposta;

  END IF;

EXCEPTION

WHEN prenot\_sovrapposta THEN

  RAISE\_APPLICATION\_ERROR('-20012','ERRORE PRENOTAZIONE: Questo campo è già stato prenotato nella fascia oraria indicata!');

END;

/

**Prenotazioni passate**

Questo trigger previene errori di inserimento, nella data di una prenotazione. Se quest’ultima è precedente a quella odierna, viene lanciata un’eccezione.

CREATE OR REPLACE TRIGGER tempo\_prenotazioni

BEFORE INSERT ON prenotazione

FOR EACH ROW

DECLARE

  errore\_prenotazione EXCEPTION;

BEGIN

    IF(SYSDATE > :NEW.data\_inizio\_attivita ) THEN

        RAISE errore\_prenotazione;

    END IF;

EXCEPTION

WHEN errore\_prenotazione THEN

    RAISE\_APPLICATION\_ERROR('-20004','ERRORE PRENOTAZIONE: La prenotazione non può essere effettuata nel passato!');

END;

/

## 6.4 - Procedure

Le procedure sono la parte più strettamente legata alla *logica di business* e all’automazione

della base di dati. Se nel DDL e nel DML il problema principale sta nel decidere la rappresentazione

ottima dei dati e garantirne il rispetto dei vincoli di integrità, in questa fase l’obiettivo è sfruttare al meglio tale rappresentazione, per automatizzare la gestione e trarne il massimo profitto.

**Effettua ordine**

Questa procedura, ha come argomenti delle informazioni sull’ordine: la partita iva del fornitore, la data (passata come stringa per agevolare l’utente), la sede della richiesta e la lista di prodotti; quest’ultima è composta da almeno un prodotto (gli altri 9 sono opzionali, ovvero NULL di default).  
Utilizza un cursore per la stampa della fattura, contenente una serie di informazioni, quali: il codice a barre, il nome, la quantità e il costo del prodotto.   
Una volta registrato l’ordine nella relativa tabella, verranno inseriti in “Include” tutti i prodotti appartenenti alla lista.  
Seguono le stampe del fornitore dell’ordine, della fattura e, infine, del costo totale.

CREATE OR REPLACE PROCEDURE effettua\_ordine(

    partita\_ivaX VARCHAR,

    data\_richiesta VARCHAR,

    citta VARCHAR,

    via\_piazza VARCHAR,

    codice\_barre\_prod1 VARCHAR,

    quantita\_prod1 NUMBER,

    codice\_barre\_prod2 VARCHAR := NULL,

    quantita\_prod2 NUMBER := NULL,

    codice\_barre\_prod3 VARCHAR := NULL,

    quantita\_prod3 NUMBER := NULL,

    codice\_barre\_prod4 VARCHAR :=  NULL ,

    quantita\_prod4 NUMBER :=  NULL,

    codice\_barre\_prod5 VARCHAR := NULL,

    quantita\_prod5 NUMBER := NULL,

    codice\_barre\_prod6 VARCHAR := NULL,

    quantita\_prod6 NUMBER := NULL,

    codice\_barre\_prod7 VARCHAR := NULL,

    quantita\_prod7 NUMBER := NULL,

    codice\_barre\_prod8 VARCHAR := NULL,

    quantita\_prod8 NUMBER := NULL,

    codice\_barre\_prod9 VARCHAR := NULL,

    quantita\_prod9 NUMBER := NULL,

    codice\_barre\_prod10 VARCHAR := NULL,

    quantita\_prod10 NUMBER := NULL

)

IS

    nome\_fornitoreX VARCHAR(64);

    tot\_ordine NUMBER := 0;

    data\_richiestaX DATE := TO\_DATE(data\_richiesta, 'dd/mm/yyyy');

    -- Cursore contenente la lista di prodotti (stampa)

    CURSOR C

    IS(

        SELECT codice\_barre AS cod\_prd, nome\_prodotto AS nome, quantita\_acquistate AS qnt,

        (costo\_unitario \* quantita\_acquistate) AS costo

        FROM include

        NATURAL JOIN prodotto\_attrezzatura

        WHERE partita\_iva = partita\_ivaX AND data\_richiesta = data\_richiestaX);

BEGIN

    -- inserimento dell'ordine

    INSERT INTO ordine (citta, via\_piazza, partita\_iva, data\_richiesta)

    VALUES (citta, via\_piazza, partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'));

    SELECT nome\_fornitore INTO nome\_fornitoreX FROM fornitore WHERE partita\_iva = partita\_ivaX;

    -- inserimento dei prodotti

    INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

    VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod1,quantita\_prod1);

    IF ((codice\_barre\_prod2 IS NOT NULL) AND (quantita\_prod2 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod2,quantita\_prod2);

    END IF;

    IF ((codice\_barre\_prod3 IS NOT NULL) AND (quantita\_prod3 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod3,quantita\_prod3);

    END IF;

    IF ((codice\_barre\_prod4 IS NOT NULL) AND (quantita\_prod4 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod4,quantita\_prod4);

    END IF;

    IF ((codice\_barre\_prod4 IS NOT NULL) AND (quantita\_prod5 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod5,quantita\_prod5);

    END IF;

    IF ((codice\_barre\_prod5 IS NOT NULL) AND (quantita\_prod6 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod6,quantita\_prod6);

    END IF;

    IF ((codice\_barre\_prod6 IS NOT NULL) AND (quantita\_prod7 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod7,quantita\_prod7);

    END IF;

    IF ((codice\_barre\_prod7 IS NOT NULL) AND (quantita\_prod8 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod8,quantita\_prod8);

    END IF;

    IF ((codice\_barre\_prod8 IS NOT NULL) AND (quantita\_prod9 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod9,quantita\_prod9);

    END IF;

    IF ((codice\_barre\_prod9 IS NOT NULL) AND (quantita\_prod10 IS NOT NULL)) THEN

        INSERT INTO include (partita\_iva,data\_richiesta,codice\_barre,quantita\_acquistate)

        VALUES (partita\_ivaX, TO\_DATE(data\_richiesta, 'dd/mm/yyyy'), codice\_barre\_prod10,quantita\_prod10);

    END IF;

    DBMS\_OUTPUT.PUT\_LINE('Ordine effettuato presso: ' || nome\_fornitoreX || ' - partita iva: ' || partita\_ivaX || ' - in data: '|| data\_richiesta);

    FOR rec IN C

    LOOP

        DBMS\_OUTPUT.PUT\_LINE('cod\_prd: '||rec.cod\_prd ||' - nome: ' ||rec.nome ||' - qnt: '||rec.qnt || ' - subtot: ' || rec.costo || '€');

        tot\_ordine := tot\_ordine + rec.costo;

    END LOOP;

    DBMS\_OUTPUT.PUT\_LINE('TOTALE ORDINE: ' || tot\_ordine || '€');

END;

/

**Iscrizione corso**

Questa procedura può essere utilizzata, sia per le nuove iscrizioni ai corsi sia per i rinnovi degli abbonamenti già esistenti (per come è stato strutturato il DB, sono entrambi nuovi abbonamenti).  
Prende in input i dati del corso, il periodo di validità del nuovo abbonamento sotto forma di numero di mesi ed il codice fiscale dell’iscritto; se vengono compilati i campi opzionali (nome, cognome luogo e data di nascita), si provvede anche alla registrazione del nuovo cliente (persona), qualora non fosse inserito.  
Siccome è consentito, agli stessi dipendenti, di iscriversi ad un corso, viene effettuato un controllo per evitare, che si iscriva ad un corso, il suo stesso istruttore.  
Qualora la procedura sia andata a buon fine, viene stampato un messaggio di conferma, altrimenti può essere lanciata, o l’eccezione relativa agli istruttori, oppure quella relativa ai dati mancanti, del nuovo cliente.

CREATE OR REPLACE PROCEDURE iscrizione\_corso(

    nome\_corso\_x VARCHAR,

    data\_inizio\_corso VARCHAR,

    data\_inizio\_abbonamento VARCHAR,

    num\_mesi\_x NUMBER,

    cf\_x VARCHAR,

    nome\_x VARCHAR := NULL,

    cognome\_x VARCHAR := NULL,

    luogo\_nascita\_x VARCHAR := NULL,

    data\_nascita VARCHAR := NULL)

IS

    data\_inizio\_corso\_x DATE := TO\_DATE(data\_inizio\_corso, 'dd/mm/yyyy');

    data\_inizio\_abbonamento\_x DATE := TO\_DATE(data\_inizio\_abbonamento, 'dd/mm/yyyy');

    data\_fine\_abbonamento\_x DATE := ADD\_MONTHS(data\_inizio\_abbonamento\_x, num\_mesi\_x);

    data\_nascita\_x DATE := TO\_DATE(data\_nascita, 'dd/mm/yyyy');

    cod\_abb NUMBER := 0;

    persona\_non\_reg NUMBER := 0;

    istruttore\_iscritto NUMBER := 0;

    error\_persona\_non\_reg EXCEPTION;

    error\_istrutt\_iscritto EXCEPTION;

    nome\_temp VARCHAR(64);

    cognome\_temp VARCHAR(64);

BEGIN

    -- verifica se ad iscriversi è l'istruttore del corso stesso e nel caso lancia l'eccezione

    SELECT COUNT(cf) INTO istruttore\_iscritto

    FROM corso

    WHERE cf=cf\_x;

    IF (istruttore\_iscritto > 0) THEN

        RAISE error\_istrutt\_iscritto;

    END IF;

    -- verifica se ad iscriversi è una persona non registrata nel DB e nel caso lancia l'eccezione

    SELECT COUNT(cf) INTO persona\_non\_reg

    FROM persona

    WHERE cf=cf\_x;

    -- se la persona non è registrata

    IF (persona\_non\_reg = 0) THEN

        -- se sono stati inseriti tutti i parametri per farlo

        IF (nome\_x IS NOT NULL AND

            cognome\_x IS NOT NULL AND

            luogo\_nascita\_x IS NOT NULL AND

            data\_nascita\_x IS NOT NULL) THEN

                -- registra persona

                INSERT INTO persona (cf, nome, cognome, luogo\_nascita, data\_nascita)

                VALUES (cf\_x, nome\_x, cognome\_x, luogo\_nascita\_x, data\_nascita\_x);

                -- selezione per stampa

                SELECT nome, cognome INTO nome\_temp, cognome\_temp

                FROM persona

                WHERE cf = cf\_x;

                DBMS\_OUTPUT.PUT\_LINE('REGISTRAZIONE PERSONA: '|| nome\_temp || ' ' || cognome\_temp || ' EFFETTUATA!');

        --altrimenti lancia eccezione

        ELSE

            RAISE error\_persona\_non\_reg;

        END IF;

    END IF;

    -- registrazione abbonato e relativo codice abbonamento

    INSERT INTO abbonato (cf)

    VALUES (cf\_x);

    SELECT COD\_ABB\_SEQ.CURRVAL INTO cod\_abb FROM DUAL;

    -- registrazione iscrizione al corso

    INSERT INTO iscrizione (codice\_abbonamento, nome\_corso, data\_inizio\_corso, data\_inizio\_abbonamento, data\_fine\_abbonamento)

    VALUES (cod\_abb, nome\_corso\_x, data\_inizio\_corso\_x, data\_inizio\_abbonamento\_x, data\_fine\_abbonamento\_x);

    DBMS\_OUTPUT.PUT\_LINE('ISCRIZIONE AVVENUTA CON SUCCESSO!');

    EXCEPTION

    WHEN error\_istrutt\_iscritto THEN

        RAISE\_APPLICATION\_ERROR('-20021','ERRORE: L ISTRUTTORE DEL CORSO NON PUO ISCRIVERSI AL CORSO CHE SUPERVISIONA!');

    WHEN error\_persona\_non\_reg THEN

        RAISE\_APPLICATION\_ERROR('-20022','ERRORE: PERSONA NON REGISTRATA E DATI INSERITI INSUFFICIENTI PER UNA NUOVA REGISTRAZIONE!');

END;

/

**Ordine Consegnato**

Questa procedura può essere utilizzata per registrare la consegna di un ordine.  
Prevede in input la partita iva del fornitore e la data in cui è stata effettuata la richiesta, per identificare l’ordine in questione.  
La data di consegna può essere inserita manualmente, oppure, in caso contrario, verrà inserita automaticamente la data odierna.

CREATE OR REPLACE PROCEDURE ordine\_consegnato(

  partita\_ivaX VARCHAR,

  data\_richiesta VARCHAR,

  data\_consegna VARCHAR := NULL)

IS

  no\_ordine\_exception EXCEPTION;

  data\_richiestaX DATE := TO\_DATE(data\_richiesta, 'dd/mm/yyyy');

  data\_consegnaX DATE := TO\_DATE(data\_consegna, 'dd/mm/yyyy');

  counter NUMBER;

BEGIN

  -- verifica se l'ordine è stato registrato

  SELECT COUNT(\*) INTO counter

  FROM ordine

  WHERE partita\_iva = partita\_ivaX AND data\_richiesta = data\_richiestaX;

  IF(counter = 1) THEN

    -- se è stata passata una data di consegna, inseriscila

    IF(data\_consegnaX IS NOT NULL) THEN

      UPDATE ordine SET data\_consegna = data\_consegnaX WHERE partita\_iva = partita\_ivaX AND data\_richiesta = data\_richiestaX;

      DBMS\_OUTPUT.PUT\_LINE('Ordine: ' || partita\_ivaX || ' richiesto in data: '|| data\_richiestaX || ' consegnato in data: ' || data\_consegnaX);

    -- altrimenti inserisci la data odierna

    ELSE

      UPDATE ordine SET data\_consegna = SYSDATE WHERE partita\_iva = partita\_ivaX AND data\_richiesta = data\_richiestaX;

      DBMS\_OUTPUT.PUT\_LINE('Ordine: ' || partita\_ivaX || ' richiesto in data: '|| data\_richiestaX || ' consegnato in data: ' || SYSDATE);

    END IF;

  --altrimenti lancia eccezione

  ELSE

    RAISE no\_ordine\_exception;

  END IF;

EXCEPTION

WHEN no\_ordine\_exception THEN

  RAISE\_APPLICATION\_ERROR('-20020','ERRORE: ordine non registrato! Verificare la partita iva e la data richiesta inserite...');

END;

/

**Prenotazione Campo**

Tale procedura ha lo scopo di prenotare automaticamente, il primo campo disponibile per la disciplina desiderata dal cliente.  
Vengono passati in input la disciplina, i dati della sede, la data e la durata dell’attività sotto forma di numero di ore ed i dati del cliente (per identificarlo e/o contattarlo in caso di problemi).  
Tramite un cursore vengono listati i campi adeguati, della sede in questione, disponibili nella fascia oraria della prenotazione (quindi quelli in cui non si tengono corsi e che non sono già stati prenotati). Se la lista è non vuota, viene effettuata la prenotazione sul primo campo disponibile e stampato un messaggio con l’identificativo del campo, altrimenti viene lanciata l’eccezione.

CREATE OR REPLACE PROCEDURE prenotazione\_campo(

  disciplinaX VARCHAR,

  cittaX VARCHAR,

  via\_piazzaX VARCHAR,

  data\_inizio\_attivita VARCHAR,

  durata\_ore\_attivitaX number,

  nominativo\_clienteX VARCHAR,

  telefono\_clienteX VARCHAR)

IS

  data\_inizio\_attivitaX DATE := TO\_DATE(data\_inizio\_attivita, 'dd/mm/yyyy hh24:mi');

  data\_fine\_attivitaX DATE := TO\_DATE(TO\_CHAR(data\_inizio\_attivitaX + durata\_ore\_attivitaX/24,'dd/mm/yyyy hh24:mi'),

    'dd/mm/yyyy hh24:mi');

  giornoX VARCHAR(10) := TO\_CHAR(TO\_DATE(data\_inizio\_attivita, 'dd/mm/yyyy hh24:mi'), 'Day');

  ora\_inizioX NUMBER := TO\_NUMBER(TO\_CHAR(data\_inizio\_attivitaX,'HH24,MI'));

  ora\_fineX NUMBER:= TO\_NUMBER(TO\_CHAR(data\_fine\_attivitaX,'HH24,MI'));

  idC campo.id\_campo%TYPE;

  campi\_esauriti EXCEPTION;

  -- Cursore contenente tutti i campi liberi

  CURSOR C

  IS(

    (SELECT id\_campo FROM campo WHERE citta=cittaX AND via\_piazza=via\_piazzaX AND disciplina=disciplinaX)

    MINUS -- Campi occupati da prenotazioni

      (SELECT id\_campo

      FROM prenotazione

      WHERE(

      (data\_inizio\_attivita BETWEEN data\_inizio\_attivitaX AND data\_fine\_attivitaX)

      OR

      (data\_fine\_attivita BETWEEN data\_inizio\_attivitaX AND data\_fine\_attivitaX)

      ))

    MINUS -- Campi occupati da lezioni dei corsi

    SELECT id\_campo

    FROM (SELECT \*

          FROM corso NATURAL JOIN lezione NATURAL JOIN turno

          WHERE giorno = giornoX AND

            (ora\_inizio BETWEEN ora\_inizioX AND ora\_fineX

            OR ora\_fine BETWEEN ora\_inizioX AND ora\_fineX)

          )

);

BEGIN

  OPEN C;

  FETCH C INTO idC;

  -- se ci sono campi liberi (il cursore contiene elementi)

  IF idC IS NOT NULL THEN

    -- inserimento prenotazione

    INSERT INTO prenotazione (id\_campo, data\_inizio\_attivita, data\_fine\_attivita, nominativo\_cliente , telefono\_cliente)

    VALUES (idC, data\_inizio\_attivitaX, data\_fine\_attivitaX, nominativo\_clienteX, telefono\_clienteX);

    DBMS\_OUTPUT.PUT\_LINE('CAMPO '|| idC || ' prenotato!');

  -- altrimenti lancia eccezione

  ELSE

    RAISE campi\_esauriti;

  END IF;

  CLOSE C;

EXCEPTION

  WHEN campi\_esauriti THEN

  RAISE\_APPLICATION\_ERROR('-20022','ERRORE: NESSUN CAMPO DISPONIBILE per la sede e la disciplina indicate!');

END;

/

**Vendita prodotti**

Questa procedura, ha come argomenti le informazioni relative alla vendita: la sede e la lista dei prodotti; quest’ultima è composta da almeno un prodotto (gli altri 9 sono opzionali, ovvero NULL di default). Utilizza un cursore per la stampa dello scontrino contenente una serie di informazioni, quali: il codice a barre, il nome, la quantità e il costo del prodotto.  
Una volta registrata la vendita nella relativa tabella, verranno inseriti in “Vendita\_prodotto” tutti i prodotti appartenenti alla lista.  
Segue la stampa dello scontrino e, quindi, della lista dei prodotti venduti e l’importo totale.

CREATE OR REPLACE PROCEDURE vendita\_prodotti(

    citta VARCHAR,

    via\_piazza VARCHAR,

    codice\_barre\_prod1 VARCHAR,

    quantita\_prod1 NUMBER,

    codice\_barre\_prod2 VARCHAR := NULL,

    quantita\_prod2 NUMBER := NULL,

    codice\_barre\_prod3 VARCHAR := NULL,

    quantita\_prod3 NUMBER := NULL,

    codice\_barre\_prod4 VARCHAR :=  NULL ,

    quantita\_prod4 NUMBER :=  NULL,

    codice\_barre\_prod5 VARCHAR := NULL,

    quantita\_prod5 NUMBER := NULL,

    codice\_barre\_prod6 VARCHAR := NULL,

    quantita\_prod6 NUMBER := NULL,

    codice\_barre\_prod7 VARCHAR := NULL,

    quantita\_prod7 NUMBER := NULL,

    codice\_barre\_prod8 VARCHAR := NULL,

    quantita\_prod8 NUMBER := NULL,

    codice\_barre\_prod9 VARCHAR := NULL,

    quantita\_prod9 NUMBER := NULL,

    codice\_barre\_prod10 VARCHAR := NULL,

    quantita\_prod10 NUMBER := NULL

)

IS

    cod\_scontrino NUMBER := 0;

    tot\_scontrino NUMBER := 0;

    -- Cursore contenente la lista di prodotti (stampa)

    CURSOR C

    IS (SELECT vendita\_prodotto.codice\_barre AS cod\_prd, prodotto\_attrezzatura.nome\_prodotto AS nome,

        vendita\_prodotto.quantita\_vendute AS qnt,

        (prodotto\_attrezzatura.prezzo\_vendita\_unitario \* vendita\_prodotto.quantita\_vendute) AS costo

        FROM vendita\_prodotto

        JOIN prodotto\_attrezzatura ON vendita\_prodotto.codice\_barre = prodotto\_attrezzatura.codice\_barre

        WHERE vendita\_prodotto.id\_scontrino = cod\_scontrino);

BEGIN

    -- inserimento della vendita

    INSERT INTO vendita (citta, via\_piazza)

    VALUES (citta, via\_piazza);

    SELECT ID\_SCONTRINO\_SEQ.CURRVAL INTO cod\_scontrino FROM DUAL;

    -- inserimento dei prodotti

    INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

    VALUES (cod\_scontrino, codice\_barre\_prod1,quantita\_prod1);

    IF ((codice\_barre\_prod2 IS NOT NULL) AND (quantita\_prod2 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod2,quantita\_prod2);

    END IF;

    IF ((codice\_barre\_prod3 IS NOT NULL) AND (quantita\_prod3 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod3,quantita\_prod3);

    END IF;

    IF ((codice\_barre\_prod4 IS NOT NULL) AND (quantita\_prod4 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod4,quantita\_prod4);

    END IF;

    IF ((codice\_barre\_prod4 IS NOT NULL) AND (quantita\_prod5 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod5,quantita\_prod5);

    END IF;

    IF ((codice\_barre\_prod5 IS NOT NULL) AND (quantita\_prod6 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod6,quantita\_prod6);

    END IF;

    IF ((codice\_barre\_prod6 IS NOT NULL) AND (quantita\_prod7 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod7,quantita\_prod7);

    END IF;

    IF ((codice\_barre\_prod7 IS NOT NULL) AND (quantita\_prod8 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod8,quantita\_prod8);

    END IF;

    IF ((codice\_barre\_prod8 IS NOT NULL) AND (quantita\_prod9 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod9,quantita\_prod9);

    END IF;

    IF ((codice\_barre\_prod9 IS NOT NULL) AND (quantita\_prod10 IS NOT NULL)) THEN

        INSERT INTO vendita\_prodotto (id\_scontrino,codice\_barre,quantita\_vendute)

        VALUES (cod\_scontrino, codice\_barre\_prod10,quantita\_prod10);

    END IF;

    DBMS\_OUTPUT.PUT\_LINE('Stampato lo scontrino numero: ' || cod\_scontrino);

    FOR rec IN C

    LOOP

        DBMS\_OUTPUT.PUT\_LINE('cod\_prd: '||rec.cod\_prd ||' - nome: ' ||rec.nome ||' - qnt: '||rec.qnt || ' - subtot: ' || rec.costo || '€');

        tot\_scontrino := tot\_scontrino + rec.costo;

    END LOOP;

    DBMS\_OUTPUT.PUT\_LINE('TOTALE SCONTRINO: ' || tot\_scontrino || '€');

END;

/

## 6.5 - Viste

Le viste sono relazioni virtuali utili a regolamentare l’accesso ed aumentare la fruibilità dei dati,

per i vari utenti. È possibile (e auspicabile) usare le viste per mostrare porzioni dei dati (i dati

più recenti, meno recenti, quelli di una determinata area etc.), calcolare gli attributi derivati e

applicare funzioni più o meno complesse.

**Prenotazioni ancora valide**

Questa view, mostra le prenotazioni ancora attive, in particolare la sede e il campo coinvolti e la fascia oraria in cui si terrà l’attività.

CREATE OR REPLACE VIEW vw\_info\_prenotazioni\_in\_corso AS

SELECT citta, via\_piazza, id\_campo, disciplina, data\_inizio\_attivita, data\_fine\_attivita,

    nominativo\_cliente, telefono\_cliente, prezzo\_orario, tipo\_terreno

FROM prenotazione NATURAL JOIN campo

WHERE (data\_inizio\_attivita >= SYSDATE)

ORDER BY data\_inizio\_attivita;

**Storico prenotazioni**

Questa view, mostra le prenotazioni non più attive, in particolare la sede e il campo coinvolti e la fascia oraria in cui si è tenuta l’attività.

CREATE OR REPLACE VIEW vw\_info\_storico\_prenotazioni AS

SELECT citta, via\_piazza, id\_campo, disciplina, data\_inizio\_attivita, data\_fine\_attivita,

    nominativo\_cliente, telefono\_cliente, prezzo\_orario, tipo\_terreno

FROM prenotazione NATURAL JOIN campo

WHERE (data\_inizio\_attivita < SYSDATE)

ORDER BY data\_inizio\_attivita;

**Corsi in atto**

Questa view mostra: i corsi ancora attivi con le sue informazioni, il campo sulla quale si svolge l’attività, l’istruttore che lo gestisce ed il numero di persone abbonate.

CREATE OR REPLACE VIEW vw\_info\_corsi\_in\_atto AS

SELECT corso.nome\_corso, corso.data\_inizio\_corso, corso.data\_fine\_corso,  count(DISTINCT abbonato.cf) AS abbonati, corso.quota\_mensile,

sede.citta, sede.via\_piazza, campo.id\_campo, persona.nome AS nome\_istruttore, persona.cognome AS cognome\_istruttore

FROM corso

JOIN iscrizione ON iscrizione.nome\_corso = corso.nome\_corso AND iscrizione.data\_inizio\_corso = corso.data\_inizio\_corso

JOIN abbonato ON iscrizione.codice\_abbonamento = abbonato.codice\_abbonamento

JOIN persona ON persona.cf = corso.cf

JOIN campo ON campo.id\_campo = corso.id\_campo

JOIN sede ON campo.citta = sede.citta AND campo.via\_piazza = sede.via\_piazza

WHERE corso.data\_fine\_corso >= SYSDATE

GROUP BY corso.nome\_corso, corso.data\_inizio\_corso, corso.data\_fine\_corso, corso.quota\_mensile,

sede.citta, sede.via\_piazza, campo.id\_campo, persona.nome, persona.cognome;

**Corsi terminati**

Questa view mostra i corsi non più attivi con le sue informazioni, il campo sulla quale si è svolta l’attività, l’istruttore che lo ha gestito ed il numero di persone che vi hanno partecipato.

CREATE OR REPLACE VIEW vw\_info\_corsi\_terminati AS

SELECT corso.nome\_corso, corso.data\_inizio\_corso, corso.data\_fine\_corso,  count(DISTINCT abbonato.cf) AS abbonati, corso.quota\_mensile,

sede.citta, sede.via\_piazza, campo.id\_campo, persona.nome AS nome\_istruttore, persona.cognome AS cognome\_istruttore

FROM corso

JOIN iscrizione ON iscrizione.nome\_corso = corso.nome\_corso AND iscrizione.data\_inizio\_corso = corso.data\_inizio\_corso

JOIN abbonato ON iscrizione.codice\_abbonamento = abbonato.codice\_abbonamento

JOIN persona ON persona.cf = corso.cf

JOIN campo ON campo.id\_campo = corso.id\_campo

JOIN sede ON campo.citta = sede.citta AND campo.via\_piazza = sede.via\_piazza

WHERE corso.data\_fine\_corso < SYSDATE

GROUP BY corso.nome\_corso, corso.data\_inizio\_corso, corso.data\_fine\_corso, corso.quota\_mensile,

sede.citta, sede.via\_piazza, campo.id\_campo, persona.nome, persona.cognome;

**Turni dipendenti**

Questa view mostra le informazioni relative alla turnazione dei dipendenti, quindi i dati anagrafici e le fasce orarie di lavoro.

CREATE OR REPLACE VIEW vw\_info\_turni\_dipendenti AS

SELECT persona.cognome, persona.nome, personale.mansione, personale.citta, personale.via\_piazza, turno.codice\_turno, turno.giorno, turno.ora\_inizio, turno.ora\_fine

FROM turnazione inner JOIN persona ON turnazione.cf=persona.cf

inner JOIN personale ON turnazione.cf=personale.cf

inner JOIN turno ON turnazione.codice\_turno=turno.codice\_turno

ORDER BY turnazione.codice\_turno;

**Ordini ricevuti**

Questa view mostra le informazioni relative agli ordini già ricevuti, nonché dei prodotti che ne fanno parte.

CREATE OR REPLACE VIEW vw\_info\_ordini\_ricevuti AS

SELECT ordine.partita\_iva, ordine.data\_richiesta, ordine.data\_consegna, ordine.citta, ordine.via\_piazza, include.codice\_barre, prodotto\_attrezzatura.nome\_prodotto, include.quantita\_acquistate, (prodotto\_attrezzatura.costo\_unitario \* include.quantita\_acquistate) AS costo

FROM include JOIN ordine ON include.partita\_iva = ordine.partita\_iva AND include.data\_richiesta = ordine.data\_richiesta

JOIN prodotto\_attrezzatura ON prodotto\_attrezzatura.codice\_barre = include.codice\_barre

WHERE ordine.data\_consegna <= SYSDATE AND ordine.data\_consegna IS NOT NULL

ORDER BY ordine.data\_richiesta;

**Ordini aperti**

Questa view mostra le informazioni relative agli ordini non ancora consegnati, nonché dei prodotti che ne fanno parte.

CREATE OR REPLACE VIEW vw\_info\_ordini\_aperti AS

SELECT ordine.partita\_iva, ordine.data\_richiesta, ordine.data\_consegna, ordine.citta, ordine.via\_piazza, include.codice\_barre, prodotto\_attrezzatura.nome\_prodotto, include.quantita\_acquistate, (prodotto\_attrezzatura.costo\_unitario \* include.quantita\_acquistate) AS costo

FROM include JOIN ordine ON include.partita\_iva = ordine.partita\_iva AND include.data\_richiesta = ordine.data\_richiesta

JOIN prodotto\_attrezzatura ON include.codice\_barre = prodotto\_attrezzatura.codice\_barre

WHERE ordine.data\_consegna IS NULL

ORDER BY ordine.data\_richiesta;

**Abbonamenti ancora validi**

Questa view mostra i dati delle persone con un abbonamento ad un corso, ancora valido.

CREATE OR REPLACE VIEW vw\_abbonamenti\_in\_corso AS

SELECT abb.nome, abb.cognome, iscrizione.nome\_corso, iscrizione.data\_inizio\_abbonamento, iscrizione.data\_fine\_abbonamento, corso.quota\_mensile,

campo.citta, campo.via\_piazza

FROM iscrizione NATURAL JOIN (SELECT abbonato.codice\_abbonamento, persona.nome, persona.cognome  FROM  abbonato NATURAL JOIN persona)abb

JOIN corso ON iscrizione.nome\_corso = corso.nome\_corso AND iscrizione.data\_inizio\_corso=corso.data\_inizio\_corso

JOIN campo ON corso.id\_campo= campo.id\_campo

WHERE iscrizione.data\_fine\_abbonamento > SYSDATE

ORDER BY campo.citta, campo.via\_piazza, iscrizione.nome\_corso, iscrizione.data\_inizio\_abbonamento;

**Abbonamenti scaduti**

Questa view mostra i dati delle persone, i cui abbonamenti sono scaduti.

CREATE OR REPLACE VIEW vw\_abbonamenti\_terminati AS

SELECT abb.nome, abb.cognome, iscrizione.nome\_corso, iscrizione.data\_inizio\_abbonamento, iscrizione.data\_fine\_abbonamento, corso.quota\_mensile,

campo.citta, campo.via\_piazza

FROM iscrizione NATURAL JOIN (SELECT abbonato.codice\_abbonamento, persona.nome, persona.cognome  FROM  abbonato NATURAL JOIN persona)abb

JOIN corso ON iscrizione.nome\_corso = corso.nome\_corso AND iscrizione.data\_inizio\_corso=corso.data\_inizio\_corso

JOIN campo ON corso.id\_campo= campo.id\_campo

WHERE iscrizione.data\_fine\_abbonamento <= SYSDATE

ORDER BY campo.citta, campo.via\_piazza, iscrizione.nome\_corso, iscrizione.data\_inizio\_abbonamento;

**Vendite ultime mese**

Questa view mostra le informazioni relative ai prodotti venduti nell’ultimo mese, ordinati a partire dai più venduti.

CREATE OR REPLACE VIEW vw\_vendite\_ultimo\_mese AS

SELECT vendita\_prodotto.codice\_barre, prodotto\_attrezzatura.nome\_prodotto, SUM(vendita\_prodotto.quantita\_vendute) AS quantita\_vendute\_tot,  vendita.citta, vendita.via\_piazza

FROM vendita\_prodotto JOIN prodotto\_attrezzatura ON vendita\_prodotto.codice\_barre=prodotto\_attrezzatura.codice\_barre

JOIN vendita ON vendita\_prodotto.id\_scontrino=vendita.id\_scontrino

WHERE vendita.data\_scontrino >= ADD\_MONTHS(SYSDATE, -1)

GROUP BY vendita\_prodotto.codice\_barre, prodotto\_attrezzatura.nome\_prodotto, vendita.citta, vendita.via\_piazza

ORDER BY  quantita\_vendute\_tot DESC;

**Storico vendite**

Questa view contiene le informazioni inerenti ai prodotti venduti, ordinati per scontrino e sede.

CREATE OR REPLACE VIEW vw\_storico\_vendite AS

SELECT vendita.id\_scontrino, vendita\_prodotto.codice\_barre, prodotto\_attrezzatura.nome\_prodotto, vendita\_prodotto.quantita\_vendute,  vendita.citta, vendita.via\_piazza

FROM vendita\_prodotto JOIN prodotto\_attrezzatura ON vendita\_prodotto.codice\_barre=prodotto\_attrezzatura.codice\_barre

JOIN vendita ON vendita\_prodotto.id\_scontrino=vendita.id\_scontrino

ORDER BY vendita.citta, vendita.via\_piazza, vendita.id\_scontrino;

**Utenze pagate**

Questa view mostra i dettagli delle utenze, ordinate per sede, il cui pagamento è già stato effettuato.

CREATE OR REPLACE VIEW vw\_storico\_utenze\_pagate AS

SELECT fattura, data\_scadenza, tipo, importo\_utenza, citta, via\_piazza

FROM utenze WHERE pagamento\_utenza = 'pagato'

ORDER BY citta, via\_piazza, data\_scadenza DESC;

**Utenze non pagate**

Tale view mostra i dettagli delle utenze il cui pagamento non è ancora stato effettuato.   
Sono ordinate in base alla sede e alla data di scadenza.

CREATE OR REPLACE VIEW vw\_storico\_utenze\_non\_pagate AS

SELECT fattura, data\_scadenza, tipo, importo\_utenza, citta, via\_piazza

FROM utenze WHERE pagamento\_utenza = 'non pagato'

ORDER BY citta, via\_piazza, data\_scadenza DESC;

**Personale contratto determinato**

Tale view restituisce i dettagli inerenti al personale con contratto a tempo determinato e mostra principalmente il costo totale del lavoratore, dalla data di assunzione alla data di fine rapporto.

CREATE OR REPLACE VIEW vw\_personale\_contratto\_determ AS

SELECT cf, ROUND((SUM(stipendio) \* MONTHS\_BETWEEN(data\_fr, data\_assunzione))) AS stipendi\_totali , data\_assunzione, data\_fr , citta, via\_piazza FROM personale

WHERE data\_fr IS NOT NULL

GROUP BY cf, data\_assunzione, data\_fr, citta, via\_piazza

ORDER BY citta, via\_piazza;

**Personale contratto indeterminato**

Tale view restituisce i dettagli inerenti al personale assunto con contratto a tempo indeterminato e mostra principalmente il costo totale del lavoratore.

CREATE OR REPLACE VIEW vw\_personale\_contratto\_indet AS

SELECT cf, ROUND((SUM(stipendio)) \* MONTHS\_BETWEEN(SYSDATE, data\_assunzione)) AS stipendi\_totali , data\_assunzione, citta, via\_piazza FROM personale

WHERE data\_fr IS NULL

GROUP BY cf, data\_assunzione, citta, via\_piazza

ORDER BY citta, via\_piazza;

**Importo ordine**

Questa view viene utilizzata principalmente per calcolare il costo complessivo dei singoli ordini (attributo derivabile di ordine).

CREATE OR REPLACE VIEW vw\_importo\_ordine AS

SELECT partita\_iva, data\_richiesta, citta, via\_piazza, SUM(quantita\_acquistate\*costo\_unitario) AS importo\_ordine

FROM ordine NATURAL JOIN include NATURAL JOIN prodotto\_attrezzatura

GROUP BY partita\_iva, data\_richiesta, citta, via\_piazza;

**Importo vendita**

Questa view viene utilizzata principalmente per calcolare l’importo complessivo delle singole vendite (attributo derivabile di vendita).

CREATE OR REPLACE VIEW vw\_importo\_vendita AS

SELECT id\_scontrino, citta, via\_piazza, SUM(quantita\_vendute\*prezzo\_vendita\_unitario) AS importo\_vendita

FROM vendita\_prodotto NATURAL JOIN prodotto\_attrezzatura NATURAL JOIN vendita

GROUP BY id\_scontrino, citta, via\_piazza;

**Bilancio**

Questa view mostra gli importi totali delle varie entrate ed uscite delle singole sedi, per poi calcolarne un bilancio storico (non annuale).

CREATE OR REPLACE VIEW vw\_bilancio AS

SELECT citta, via\_piazza, utenze, ordini, stipendi\_fr, stipendi\_indeterminati, vendite, prenotazioni, abbonamenti\_in\_corso, abbonamenti\_terminati,

(u.utenze + o.ordini + sfr.stipendi\_fr + sind.stipendi\_indeterminati) AS uscite,

(v.vendite + p.prenotazioni + aic.abbonamenti\_in\_corso + at.abbonamenti\_terminati) AS entrate,

((v.vendite + p.prenotazioni + aic.abbonamenti\_in\_corso + at.abbonamenti\_terminati) - (u.utenze + o.ordini + sfr.stipendi\_fr + sind.stipendi\_indeterminati)) AS bilancio

FROM(SELECT citta, via\_piazza, SUM(importo\_utenza) AS utenze

    FROM vw\_storico\_utenze\_pagate

    GROUP BY citta, via\_piazza) u

    NATURAL JOIN(SELECT citta, via\_piazza, SUM(importo\_ordine) AS ordini

        FROM vw\_importo\_ordine

        GROUP BY citta, via\_piazza) o

        NATURAL JOIN (SELECT citta, via\_piazza, SUM(stipendi\_totali) AS stipendi\_fr

            FROM vw\_personale\_contratto\_determ

            GROUP BY citta, via\_piazza) sfr

            NATURAL JOIN (SELECT citta, via\_piazza, SUM(stipendi\_totali) AS stipendi\_indeterminati

                FROM vw\_personale\_contratto\_indet

                GROUP BY citta, via\_piazza) sind

                NATURAL JOIN (SELECT citta, via\_piazza, SUM(vw\_importo\_vendita.importo\_vendita) AS vendite

                     FROM vw\_importo\_vendita

                     GROUP BY citta, via\_piazza) v

                     NATURAL JOIN (SELECT citta, via\_piazza, SUM(((data\_fine\_attivita - data\_inizio\_attivita) \* 24) \* prezzo\_orario) AS prenotazioni

                         FROM vw\_info\_storico\_prenotazioni

                         GROUP BY citta, via\_piazza) p

                         NATURAL JOIN (SELECT citta, via\_piazza, ROUND(SUM(MONTHS\_BETWEEN(data\_fine\_abbonamento, data\_inizio\_abbonamento) \* quota\_mensile), 2) AS abbonamenti\_in\_corso

                             FROM vw\_abbonamenti\_in\_corso

                             GROUP BY citta, via\_piazza) aic

                             NATURAL JOIN (SELECT citta, via\_piazza, ROUND(SUM(MONTHS\_BETWEEN(data\_fine\_abbonamento, data\_inizio\_abbonamento) \* quota\_mensile), 2) AS abbonamenti\_terminati

                                 FROM vw\_abbonamenti\_terminati

                                 GROUP BY citta, via\_piazza) at;

## 6.6 - Data Control Language

-- CREAZIONE RUOLI E UTENTI

-- Creazione Amministratore e relativa assegnazione privilegi

CREATE USER Admin\_Polisportive IDENTIFIED by oracle

DEFAULT TABLESPACE USERS TEMPORARY

TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

GRANT ALL PRIVILEGES TO Admin\_Polisportive;

-- Creazione Ruoli

CREATE ROLE Direzione;

CREATE ROLE Segreteria;

CREATE ROLE Bar;

-- Privilegi di sistema

GRANT CONNECT, CREATE SESSION TO Direzione;

GRANT CONNECT, CREATE SESSION TO Segreteria;

GRANT CONNECT, CREATE SESSION TO Bar;

-- Privilegi di oggetto

GRANT SELECT, INSERT, UPDATE ON persona TO Segreteria;

GRANT SELECT, UPDATE ON abbonato TO Segreteria;

GRANT SELECT ON personale TO Segreteria;

GRANT SELECT ON vw\_abbonamenti\_in\_corso TO Segreteria;

GRANT SELECT ON vw\_abbonamenti\_terminati TO Segreteria;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Segreteria;

GRANT SELECT ON vw\_info\_corsi\_in\_atto TO Segreteria;

GRANT SELECT ON vw\_info\_corsi\_terminati TO Segreteria;

GRANT SELECT ON vw\_info\_prenotazioni\_in\_corso TO Segreteria;

GRANT SELECT ON vw\_info\_storico\_prenotazioni TO Segreteria;

GRANT DELETE ON prenotazione TO Segreteria;

GRANT SELECT ON vw\_storico\_utenze\_non\_pagate TO Segreteria;

GRANT SELECT ON vw\_storico\_utenze\_pagate TO Segreteria;

GRANT SELECT, INSERT, UPDATE ON utenze TO Segreteria;

GRANT SELECT ON sede TO Segreteria;

GRANT SELECT ON campo TO Segreteria;

GRANT SELECT ON vw\_importo\_ordine TO Segreteria;

GRANT SELECT ON vw\_importo\_vendita TO Segreteria;

GRANT EXECUTE ON prenotazione\_campo TO Segreteria;

GRANT EXECUTE ON iscrizione\_corso TO Segreteria;

GRANT SELECT ON sede TO Bar;

GRANT SELECT ON vw\_info\_ordini\_aperti TO Bar;

GRANT SELECT ON vw\_info\_ordini\_ricevuti TO Bar;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Bar;

GRANT SELECT ON vw\_vendite\_ultimo\_mese TO Bar;

GRANT SELECT ON vw\_storico\_vendite TO Bar;

GRANT SELECT ON fornitore TO Bar;

GRANT SELECT, INSERT, UPDATE ON prodotto\_attrezzatura TO Bar;

GRANT SELECT ON vw\_importo\_ordine TO Bar;

GRANT SELECT ON vw\_importo\_vendita TO Bar;

GRANT EXECUTE ON effettua\_ordine TO Bar;

GRANT EXECUTE ON ordine\_consegnato TO Bar;

GRANT EXECUTE ON vendita\_prodotti TO Bar;

GRANT SELECT, INSERT, UPDATE ON persona TO Direzione;

GRANT SELECT, INSERT, UPDATE ON personale TO Direzione;

GRANT SELECT ON vw\_bilancio TO Direzione;

GRANT SELECT ON vw\_info\_ordini\_aperti TO Direzione;

GRANT SELECT ON vw\_info\_ordini\_ricevuti TO Direzione;

GRANT SELECT ON vw\_abbonamenti\_in\_corso TO Direzione;

GRANT SELECT ON vw\_abbonamenti\_terminati TO Direzione;

GRANT SELECT ON vw\_info\_turni\_dipendenti TO Direzione;

GRANT SELECT ON vw\_info\_corsi\_in\_atto TO Direzione;

GRANT SELECT ON vw\_info\_corsi\_terminati TO Direzione;

GRANT SELECT ON vw\_info\_prenotazioni\_in\_corso TO Direzione;

GRANT SELECT ON vw\_info\_storico\_prenotazioni TO Direzione;

GRANT SELECT ON vw\_vendite\_ultimo\_mese TO Direzione;

GRANT SELECT ON vw\_storico\_vendite TO Direzione;

GRANT SELECT ON vw\_storico\_utenze\_non\_pagate TO Direzione;

GRANT SELECT ON vw\_storico\_utenze\_pagate TO Direzione;

GRANT SELECT, INSERT, UPDATE ON turno TO Direzione;

GRANT SELECT, INSERT, UPDATE ON turnazione TO Direzione;

GRANT SELECT, INSERT, UPDATE ON lezione TO Direzione;

GRANT SELECT, INSERT, UPDATE ON corso TO Direzione;

GRANT SELECT, INSERT, UPDATE ON utenze TO Direzione;

GRANT SELECT, INSERT, UPDATE ON sede TO Direzione;

GRANT SELECT, INSERT, UPDATE ON campo TO Direzione;

GRANT SELECT, INSERT, UPDATE ON fornitore TO Direzione;

GRANT UPDATE, DELETE ON ordine TO Direzione;

GRANT SELECT, INSERT, UPDATE ON prodotto\_attrezzatura TO Direzione;

GRANT SELECT ON vw\_importo\_ordine TO Direzione;

GRANT SELECT ON vw\_importo\_vendita TO Direzione;

GRANT EXECUTE ON effettua\_ordine TO Direzione;

GRANT EXECUTE ON ordine\_consegnato TO Direzione;

-- Creazione utenti e relativa assegnazione ruoli

CREATE USER bar1 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg1 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir1 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER bar2 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg2 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir2 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER bar3 IDENTIFIED BY bar DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER seg3 IDENTIFIED BY segreteria DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER dir3 IDENTIFIED BY direzione DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

CREATE USER proprieta IDENTIFIED BY proprieta DEFAULT TABLESPACE USERS

TEMPORARY TABLESPACE TEMP QUOTA 24M ON USERS PROFILE DEFAULT;

GRANT Direzione TO dir1, dir2, dir3, proprieta;

GRANT Segreteria TO seg1, seg2, seg3;

GRANT Bar TO bar1, bar2, bar3;

## 6.7 - Scheduler

**Cancellazione iscrizioni**

-- JOB CHE OGNI PRIMO DEL MESE CANCELLA TUTTE LE ISCRIZIONI SCADUTE DA DUE ANNI

BEGIN

DBMS\_SCHEDULER.CREATE\_JOB (

   job\_name             =>      'Cancellazione\_Iscrizioni',

   job\_type             =>      'PLSQL\_BLOCK',

   job\_action           =>      'BEGIN

                                    DELETE FROM Iscrizione

                                    WHERE data\_fine\_abbonamento < SYSDATE-730;

                                END;',

   start\_date           =>      TO\_DATE('01-GEN-2017','DD-MON-YYYY'),

   repeat\_interval      =>      'FREQ=MONTHLY',

   enabled              =>      TRUE,

   comments             =>      'Cancellazione delle vecchie iscrizioni');

END;

**Cancellazione prenotazioni**

-- JOB CHE OGNI PRIMO DEL MESE CANCELLA TUTTE LE PRENOTAZIONI DI DUE ANNI PRIMA

BEGIN

DBMS\_SCHEDULER.CREATE\_JOB (

   job\_name                 =>      'Cancellazione\_Prenotazioni',

   job\_type                 =>      'PLSQL\_BLOCK',

   job\_action               =>      'BEGIN

                                        DELETE FROM prenotazione

                                        WHERE data\_fine\_attivita < SYSDATE-730;

                                    END;',

   start\_date               =>      TO\_DATE('01-GEN-2017','DD-MON-YYYY'),

   repeat\_interval          =>      'FREQ=MONTHLY',

   enabled                  =>      TRUE,

   comments                 =>      'Cancellazione delle vecchie prenotazioni');

END;

**Cancellazione corsi senza iscritti per due mesi**

/\* JOB CHE OGNI PRIMO DEL MESE CANCELLA TUTTI I CORSI (E LE RELATIVE LEZIONI  SETTIMANALI), CHE NON HANNO MAI AVUTO ISCRITTI, A DUE MESI DALL'AVVIO \*/

BEGIN

DBMS\_SCHEDULER.CREATE\_JOB (

   job\_name               =>   'Cancellazione\_Corsi\_Inattivi',

   job\_type               =>   'PLSQL\_BLOCK',

   job\_action             =>   'BEGIN

                                  DELETE FROM lezione

                                  WHERE data\_inizio\_corso <= ADD\_MONTHS(SYSDATE,-2)

                                  AND (nome\_corso, data\_inizio\_corso) IN

                                      (SELECT nome\_corso, data\_inizio\_corso

                                       FROM corso

                                       MINUS

                                       SELECT nome\_corso, data\_inizio\_corso

                                       FROM iscrizione

                                       );

                                       DELETE FROM corso

                                  WHERE data\_inizio\_corso <= ADD\_MONTHS(SYSDATE,-2)

                                  AND (nome\_corso, data\_inizio\_corso) IN

                                     (SELECT nome\_corso, data\_inizio\_corso

                                       FROM corso

                                       MINUS

                                       SELECT nome\_corso, data\_inizio\_corso

                                       FROM iscrizione

                                       );

                               END;',

   start\_date             =>  TO\_DATE('01-GEN-2017','DD-MON-YYYY'),

   repeat\_interval        =>  'FREQ=MONTHLY',

   enabled                =>  TRUE,

   comments               =>  'Cancellazione dei corsi e delle lezioni inattivi');

END;