

Gestionale per un'azienda informatica per la vendita e l'assistenza hardware e software

Studenti: Piccinni Marco, Rovinetti Andrea

Progetto d'esame di Basi di Dati

ANNO ACCADEMICO: 2018/2019

Sommario

1	Descrizione	3
1.1	Glossario	5
1.2	Schema scheletro	7
2	Progetto concettuale	8
2.1	Scelte implementative	8
2.1.1	Gestione della gerarchia Magazzino	8
2.1.2	Gestione Dipendenti e Dipartimenti	9
2.1.3	Gestione Ticket e incarichi	10
2.2	Dati derivati	11
2.2.1	Studio su prezzo_totale in ORDINE	11
2.2.2	Studio su "durata_totale" in TICKET	13
2.3	Schema E/R	15
3	Progetto logico	16
3.1	Gestione delle gerarchie	16
3.1.1	Persona	16
3.1.2	Ticket	17
3.2	Schema Logico	18
4	Implementazione - Codice SQL	19
4.1	Vincoli aggiuntivi – Trigger	19
4.1.1	Dati derivati	20
4.1.2	Ordine	23
4.1.3	In_Carico	24
4.1.4	M_Vendita	25
4.2	Operazioni preliminari	27
4.2.1	Creazione dei dati	27
4.2.2	Inserimento dei dati	31
4.3	Operazioni di interrogazione	38
5	Studio degli indici	41
6	Interfaccia grafica	42
6.1	Visualizzazione	42
6.2	Modifica ed Eliminazione	43
6.3	Creazione	43
6.4	Query Tool	44

La seguente relazione descrive la progettazione e la conseguente realizzazione di un database per la gestione di una azienda informatica che si occupa della vendita sia di componenti e di dispositivi hardware, sia di software progettati ad-hoc per il cliente. In questa prima parte sono descritti in dettaglio tutte le richieste e vincoli sulle quali si è voluto porre l'accento, poi sviluppate e argomentate nel dettaglio nei paragrafi successivi.

1 DESCRIZIONE

L'azienda, come citato, si occupa di vendita al dettaglio e di sviluppo software. Inoltre, viene offerto anche uno strumento di supporto e assistenza (sia esterno che ad uso interno) mediante uno sistema di ticketing per la gestione delle richieste e degli interventi svolti.

I **Clienti** sono **Persone** di cui occorre conoscere il CF (Codice Fiscale), cognome e nome. L'indirizzo e-mail e la password invece, sono necessari per poter effettuare l'accesso ai servizi online (tra i quali si potrebbe avere e-commerce, gestione degli ordini, richiesta e gestione ticket e del personale). Sono richiesti anche ulteriori recapiti quali il numero telefonico e la residenza (città, CAP, indirizzo e numero civico).

Per ogni cliente è salvata anche la data di registrazione e, facoltativamente, anche la partita IVA (nel caso a registrarsi siano aziende o professionisti) e il nome dell'azienda (nel caso in cui il cliente che si iscrive lo faccia per conto dell'azienda per cui lavora).

Per i **Dipendenti** invece, rispetto agli attributi delle persone, sono richieste informazioni quali la data di assunzione, il ruolo e lo stipendio. Ogni dipendente è associato ad una **Divisione**, ognuna delle quali si occupa di svolgere compiti specifici all'interno dell'azienda (software, hardware, reparto vendite, direzione del personale, ...).

A capo di ogni divisione deve esistere un "team leader" che funga da Capo Area per il proprio settore di competenza. Tale ruolo può essere ricoperto dal proprietario (livello riconosciuto grazie al campo booleano leader, possono essere anche molteplici, per esempio in caso l'attività sia amministrata in società) o da uno dei dipendenti, che acquisisce la responsabilità di gestire e coordinare la propria area di competenza.

I proprietari dell'azienda devono essere in grado di poter gestire e visualizzare tutte le attività e ticket presi in carico da tutti i dipartimenti.

Il **Magazzino** richiede di conoscere il tipo di prodotto, la marca e il modello. Viene anche aggiunta la data in cui il prodotto è stato portato in magazzino, la sua posizione, ed eventualmente una descrizione. Il magazzino è suddiviso tra i prodotti presi in carico dall'**Assistenza** e i prodotti in **Vendita**, di cui è fornito anche il prezzo di listino.

Per i prodotti in assistenza è importante anche sapere se il dispositivo è stato ritirato dal cliente o se risulta ancora stoccato nel magazzino, per questo è importante conoscere anche la data in cui il prodotto è stato riconsegnato al proprietario.

Per i **Software**, sviluppati e creati internamente da uno o più dipendenti, sono salvati nome, versione, sistema operativo per il quale sono stati ideati e lo stato di sviluppo (in sviluppo, in fase di test, abbandonato, attivo, ...).

Per ogni applicazione è richiesta anche una descrizione ed eventualmente il prezzo di listino nel caso il software sia pronto per essere commercializzato. Inoltre, si vogliono rendere noti i creatori iniziali e la data in cui ciò è avvenuto.

Un **Ordine**, effettuabile sia da un cliente che un dipendente, può richiedere sia componenti e prodotti hardware sia software, di cui si vogliono salvare i prezzi di vendita (possono variare rispetto al prezzo di listino in base a sconti e offerte temporanee) e il prezzo totale.

Per ogni vendita è salvata data e ora e anche il dipendente che ha gestito la vendita, tranne nel caso in cui la richiesta di acquisto sia stata effettuata sul portale online.

Nel gestionale è previsto un sistema di **Ticketing** per le richieste di assistenza e supporto tecnico, e per la gestione del lavoro svolto dai dipendenti. Ogni ticket può essere specializzato in base alla divisione a cui appartiene o a cui è rivolto.

I ticket **Hardware** e **Software** sono casi particolari, in quanto sono riferiti rispettivamente a componenti hardware nel magazzino dell'assistenza e ai prodotti software sviluppati.

Ogni ticket aperto verso uno di queste specializzazioni (dipartimenti) sarà obbligatoriamente collegato rispettivamente al **Magazzino Assistenza** e a **Software**, rispettando il vincolo di un prodotto per ogni ticket aperto.

Ogni ticket contiene informazioni quali la descrizione, lo stato (aperto, chiuso, concluso, rifiutato, ...), la priorità (considerata da 1 a 9, dove 1 è la priorità maggiore e 9 quella minore), la data di creazione, la durata totale del tempo impiegato per chiudere il ticket (calcolato in base agli interventi effettuati), ed eventualmente il prezzo (nel caso il ticket sia stato aperto da parte di un cliente per richiedere assistenza).

Uno o più dipendenti (viene permesso il lavoro in team) possono prendere in carico un ticket (che può richiedere anche molteplici interventi), ma esclusivamente uno alla volta e a patto che essi siano registrati presso il dipartimento per il quale il ticket è stato aperto.

Ogni intervento ha campi come la data e l'ora di inizio e fine dell'operazione. Inoltre, è prevista la possibilità di registrare eventuali note per le operazioni svolte o ancora da concludere.

1.1 Glossario

Nome	Descrizione	Sinonimi	Legame
Persona	<ul style="list-style-type: none"> - Nome - Cognome - Codice Fiscale - E-mail - Password - Telefono - Indirizzo 		<ul style="list-style-type: none"> - Cliente - Dipendente - Ticket - Ordine
Dipendente	<ul style="list-style-type: none"> - Stipendio - Leader - Data Assunzione 	<ul style="list-style-type: none"> - Personale - Impiegato - Direttore - Tecnico - Programmatore - Commesso 	<ul style="list-style-type: none"> - Persona - Divisione - Ticket - Software - Ordine
Cliente	<ul style="list-style-type: none"> - Data Registrazione - Partita IVA - Azienda 	<ul style="list-style-type: none"> - Utente - Privato - Azienda 	<ul style="list-style-type: none"> - Persona
Divisione	<ul style="list-style-type: none"> - Nome - Direttore 	<ul style="list-style-type: none"> - Sezione - Dipartimento - Team - Area 	<ul style="list-style-type: none"> - Dipendente - Ticket
Magazzino	<ul style="list-style-type: none"> - Tipo 	<ul style="list-style-type: none"> - Scaffale 	<ul style="list-style-type: none"> - Ticket

	<ul style="list-style-type: none"> - Marca - Modello - Data - Descrizione - Posizione 	<ul style="list-style-type: none"> - Ripiano 	<ul style="list-style-type: none"> - Ordine - M. Assistenza - M. Vendita
Magazzino assistenza	<ul style="list-style-type: none"> -Data restituzione 	<ul style="list-style-type: none"> - Magazzino Assistenza 	<ul style="list-style-type: none"> - Magazzino - Ticket
Magazzino vendita	<ul style="list-style-type: none"> - Prezzo Listino 	<ul style="list-style-type: none"> - Magazzino vendita 	<ul style="list-style-type: none"> - Magazzino - Ordine
Software	<ul style="list-style-type: none"> - Nome - Versione - Sistema Operativo - Stato - Descrizione - Prezzo Listino 	<ul style="list-style-type: none"> - Programma - Applicazione - Prodotto 	<ul style="list-style-type: none"> - Dipendente - Ticket
Ticket	<ul style="list-style-type: none"> - Descrizione - Stato - Priorità - Prezzo - Data - Durata 	<ul style="list-style-type: none"> - Richiesta - Assistenza - Intervento 	<ul style="list-style-type: none"> - Persona - Dipendente - Divisione - Software - M. Assistenza
Ordine	<ul style="list-style-type: none"> - Data - Prezzo 	<ul style="list-style-type: none"> - Vendita 	<ul style="list-style-type: none"> - M. Vendita - Software - Dipendente - Persona

Tabella 1- Glossario delle entità

1.2 Schema scheletro

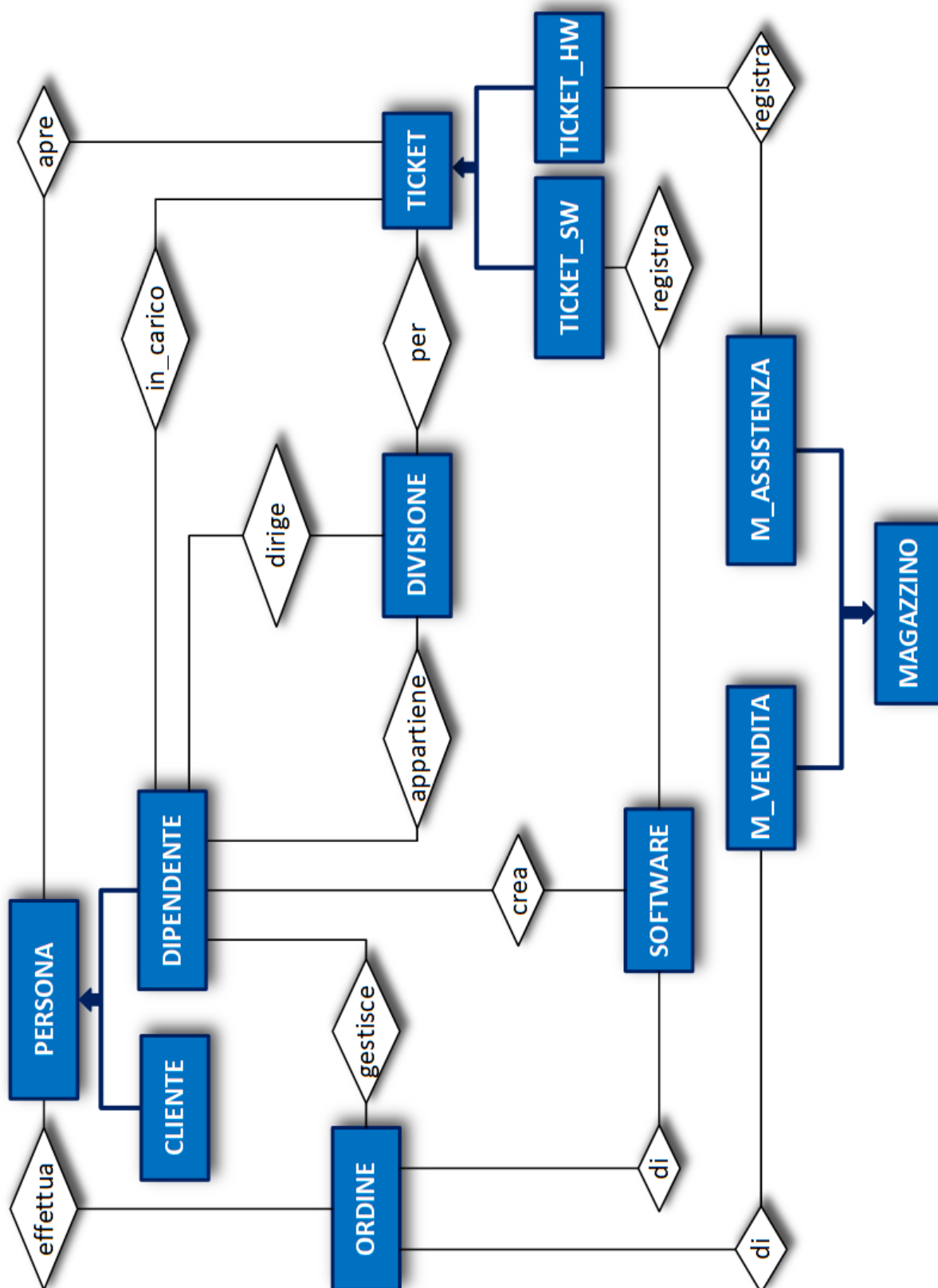


Figura 1 - Schema scheletro del database relazionale

2 PROGETTO CONCETTUALE

2.1 Scelte implementative

Di seguito sono mostrate alcune scelte implementative che, modificando quanto proposto nello schema scheletro, hanno contribuito alla realizzazione dello schema relazionale del progetto ovviando ad alcune criticità riscontrate.

2.1.1 Gestione della gerarchia Magazzino

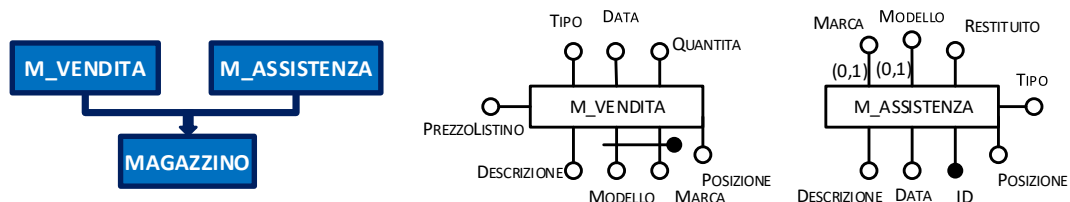


Figura 2 - Gerarchia magazzino

La gerarchia soprastante (a sinistra) è stata modificata concettualmente (a destra) rispetto allo schema scheletro precedentemente esposto. Questa scelta è stata effettuata per gestire alcune problematiche, principalmente legate alle differenti chiavi utilizzate nelle due entità.

Per il **Magazzino Vendita** si è deciso di optare per la coppia Marca, Modello del prodotto in questione, mentre gli stessi attributi sono solo opzionali per il **Magazzino Assistenza**; questo rendere possibile una gestione più flessibile di prodotti e dispositivi che non dispongono di tali campi (es. computer assemblati quindi senza una dicitura precisa di marca e modello).

Si è voluto quindi evitare l'uso della gerarchia, per eliminare campi inutili e inutilizzati da parte di entrambe le entità, diminuendo anche l'ingombro e l'eccessivo uso delle chiavi (es. Nel caso si fosse deciso di mantenere ID come chiave primaria per entrambe le entità, il **M_Vendita** avrebbe ottenuto identificativi che non sarebbero poi stati utilizzati, in quanto utilizzata la coppia marca-modello, privandoli al **M_Assistenza**). Questa scelta implica che i due magazzini siano separati (logicamente e non necessariamente anche fisicamente), migliorando l'individuazione e il posizionamento dei dispositivi stipati.

2.1.2 Gestione Dipendenti e Dipartimenti

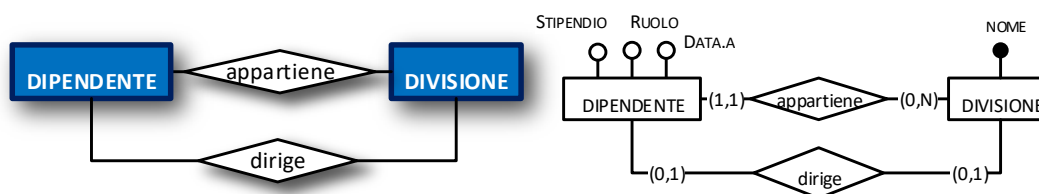


Figura 3 - Relazione Dipendente-Divisione

Come esposto nella descrizione, ogni persona all'interno dell'entità **Dipendente** (indifferentemente dal ruolo che essa ricopre all'interno dell'azienda) dovrebbe appartenere ad una **Divisione** (Software, Hardware, Vendita, Direzione, ...). Il "livello gerarchico" di ogni dipendente è stabilito all'interno dell'attributo leader: variabile booleana impostata per indicare se la persona descritta è proprietario (o uno dei soci) o un sottoposto (tecnico, impiegato, programmatore, addetto alla vendita, ...). Per quanto riguarda la responsabilità del dipendente, nel caso sia anche a capo della propria divisione, viene gestita tramite una relazione in cui un dipendente può, o meno, assumere tale responsabilità e una divisione può, o meno, avere un capo-area. La cardinalità debole nella parte della divisione è stata stabilita per evitare eventuali inconvenienti che si sarebbero potuti riscontrare in fase di creazione (o nel caso in cui la divisione esista ma ancora nessuno le sia stato assegnato).

Da database è permessa la possibilità di richiedere la lettura, la modifica e la creazione di **Ticket**, o di interventi (gestiti grazie alla relazione **In_Carico**), esclusivamente in base alla divisione di appartenenza del singolo. Per consentire ai proprietari la completa autonomia di gestione dei ticket, indifferentemente dal proprio settore di competenza, le interrogazioni alle entità saranno gestite in modo differente nel front-end, avendo a disposizione un numero maggiore di query specifiche per il ruolo che essi rivestono.

In ultima analisi, si è deciso di suddividere chiunque lavori per l'azienda su 3 livelli gerarchici. A capo dell'attività sono presenti i proprietari, al di sotto i dirigenti dei singoli dipartimenti e, per finire, i sottoposti. Non è escluso che un proprietario possa essere anche dirigente di una singola divisione, permettendo quindi una discreta autonomia per la gestione del personale.

2.1.3 Gestione Ticket e incarichi

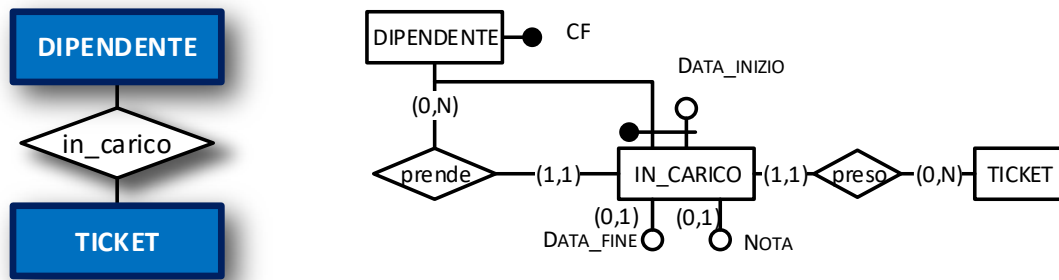


Figura 4 – Relazione Dipendente-Ticket

Nel caso attuale viene considerato il vincolo per il quale più dipendenti possono prendere in carico uno stesso ticket, per permettere anche il team work, ma un dipendente può concentrarsi su un ticket per volta.

In pratica, se un intervento risultasse ancora da concludere (data_fine nulla) non sarebbe possibile inserire, o cominciare, nuovi incarichi. Inoltre, non è possibile avere più incarichi accavallati.

Per ottenere queste limitazioni si è dovuta effettuare un'operazione di *reificazione* sulla relazione in_carico trasformandola in una entità. La chiave della nuova entità **In_Carico** è data dalla coppia (data_inizio e cf del dipendente). Tale soluzione non copre tutti i casi imposti dalle limitazioni descritte, per le quali si è dovuto intervenire creando un trigger descritto successivamente.

2.2 Dati derivati

Nel seguente paragrafo verranno trattati i due dati derivati presenti all'interno del database per discutere della loro effettiva utilità, e se comportano un vantaggio o un costo aggiuntivo in termini di numero di operazioni.

2.2.1 Studio su prezzo totale in ORDINE

Si vuole valutare l'uso del dato derivato prezzo totale, come attributo dell'entità **Ordine**, calcolato come somma degli attributi prezzo vendita moltiplicato per quantita_venduta, sulle relazioni **Di_S** e **Di_H**, per ogni prodotto o dispositivo all'interno dell'ordinazione.

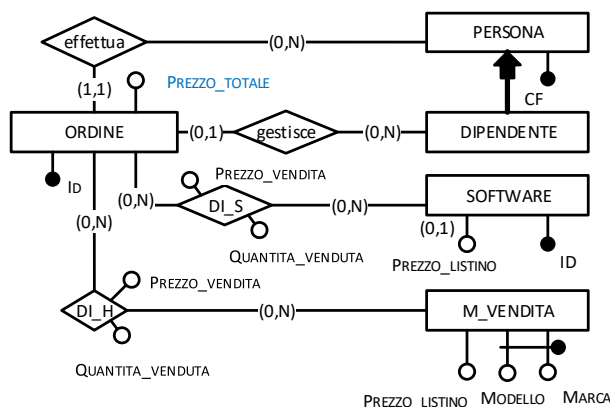


Figura 5 – Estratto di schema E/R per operazioni su ORDINE

Concetto	Tipo	Volume
PERSONA	E	1000
effettua	R	5000 (ordine)
ORDINE	E	5000
DIPENDENTE	E	200
gestisce	R	5000 (ordine)
DI_S	R	1500
SOFTWARE	E	20
DI_H	R	4000
M_VENDITA	E	200

Per la valutazione sarà considerato il caso più completo in cui l'ordine comprende l'acquisto sia di prodotti hardware che software.

Le operazioni considerate sono le seguenti:

1. Creazione di un nuovo ordine (online) contenente 2 prodotti hardware e 1 software (avendo CF di Persona e prodotti noti).
2. Visualizzazione degli ordini gestiti da un dipendente noto (incluso il prezzo totale).

Op.	Tipo	Frequenza
1	I	5/giorno
2	I	2/giorno

Operazione (con dato derivato)	Concetto	Accesso	Tipo
1:	ORDINE	1	S
1 accessi in lettura	DI_H	2*1	S
5 accessi in scrittura	DI_S	1	S
	ORDINE	1	L
11*5 = 55/giorno	ORDINE	1	S
2:	DIPENDENTE	1	L
26 accessi in lettura	ORDINE	25	L
26*2 = 52/giorno			
Totale: 107/giorno			

Operazione (senza dato derivato)	Concetto	Accesso	Tipo
1:	ORDINE	1	S
4 accessi in scrittura	DI_H	2*1	S
	DI_S	1	S
8*5 = 40/giorno			
2:	DIPENDENTE	1	L
53,5 accessi in lettura	ORDINE	25	L
	DI_S	0,3*25=7,5	L
53,5*2 = 107/giorno	DI_H	0,8*25=20	L
Totale: 147/giorno			

Lo studio effettuato mostra che è conveniente utilizzare il dato derivato analizzato, in quanto comporta un minore numero di operazioni al giorno, e pertanto è stato inserito nel database.

2.2.2 Studio su “durata_totale” in TICKET

In questo secondo caso invece, si è voluto analizzare il dato derivato durata_totale, attributo dell’entità **Ordine**, calcolato come somma delle differenze temporali (tra data_fine e data_inizio) di ogni intervento contenuto nell’entità **In_Carico**.

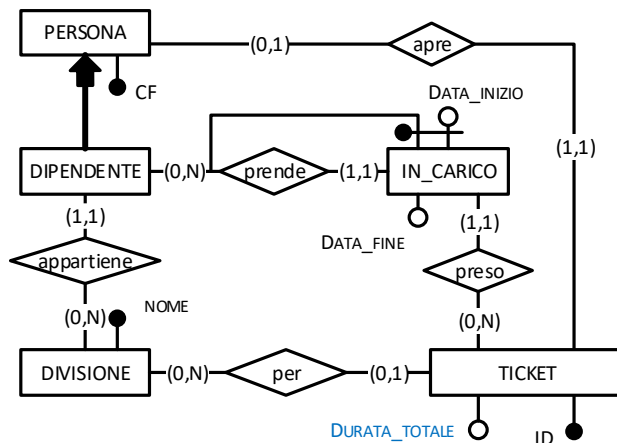


Figura 6 - Estratto di schema E/R per operazioni su TICKET

Concetto	Tipo	Volume
PERSONA	E	1000
apre	R	10000 (ticket)
TICKET	E	8000 (ticket)
DIPENDENTE	E	200
appartiene	R	200 (dipendente)
DIVISIONE	E	4
per	R	10000 (ticket)
prende	R	20000 (in_carico)
IN_CARICO	E	48000
preso	R	20000 (in_carico)

Le operazioni considerate sono le seguenti:

1. Inserimento di un nuovo intervento (ID Ticket e CF Dipendente noti) in In_Carico.
2. Visualizzazione dei dati di un Ticket, compresa la durata_totale (ID Ticket noto).

Op.	Tipo	Frequenza
1	I	25/giorno
2	I	15/giorno

Operazione (con dato derivato)	Concetto	Accesso	Tipo
1:	IN_CARICO	1	S
1 accessi in lettura	prende	1	S
4 accessi in scrittura	preso	1	S
	TICKET	1	L
9*25 = 225/giorno	TICKET	1	S
2:	TICKET	1	L
1 accessi in lettura			
1*15 = 15/giorno			
Totale: 240/giorno			

Operazione (senza dato derivato)	Concetto	Accesso	Tipo
1:	IN_CARICO	1	S
3 accessi in scrittura	prende	1	S
	preso	1	S
6*25 = 150/giorno			
2:	TICKET	1	L
7 accessi in lettura	IN_CARICO	6	L
7*15 = 105/giorno			
Totale: 255/giorno			

Anche in questo caso l'analisi mostra che utilizzare il dato derivato, piuttosto che calcolarlo ad ogni richiesta, dovrebbe permettere un minor numero di operazioni effettuate sul database.

2.3 Schema E/R

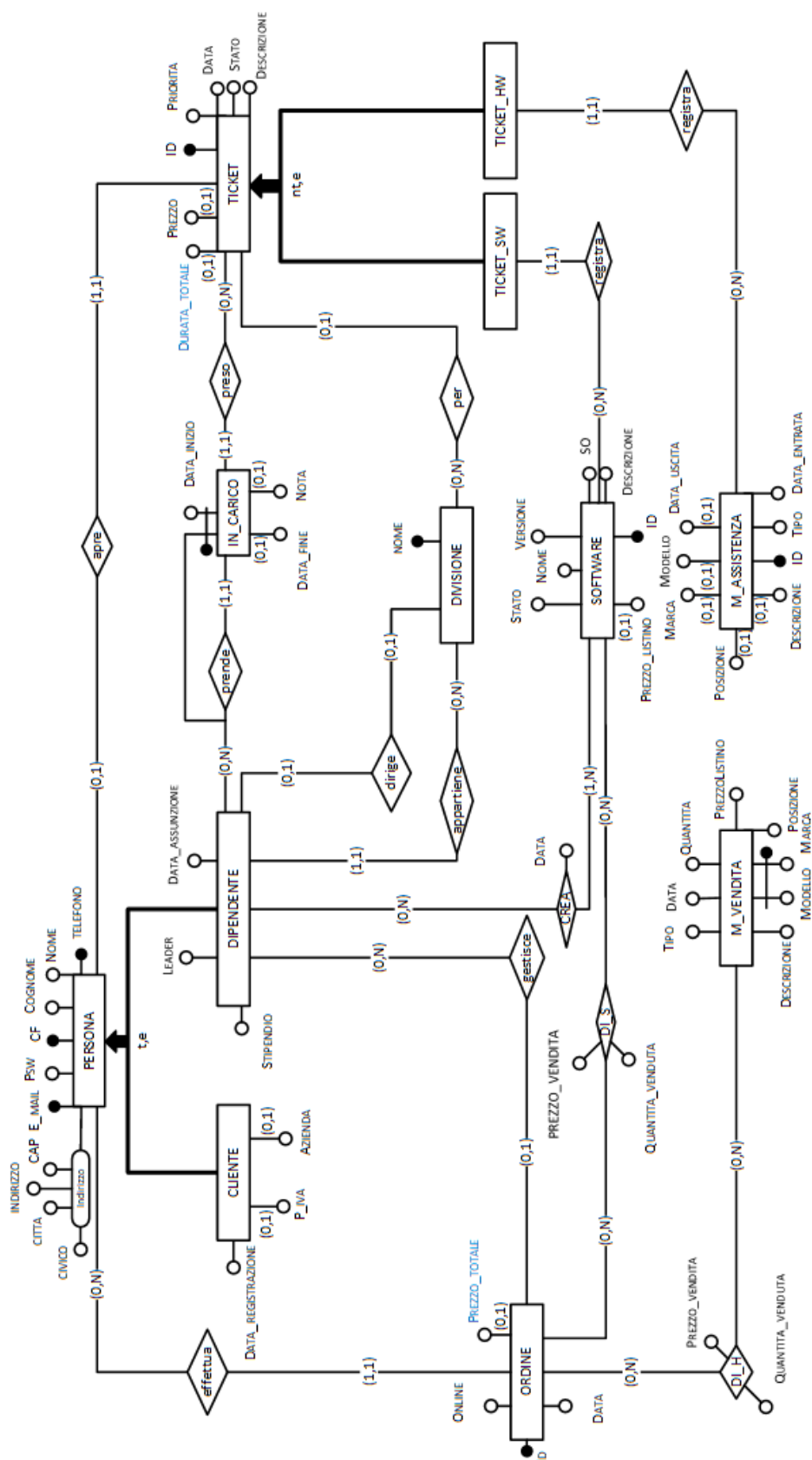


Figura 7 - Schema E/R del database relazionale

3 PROGETTO LOGICO

3.1 Gestione delle gerarchie

3.1.1 Persona

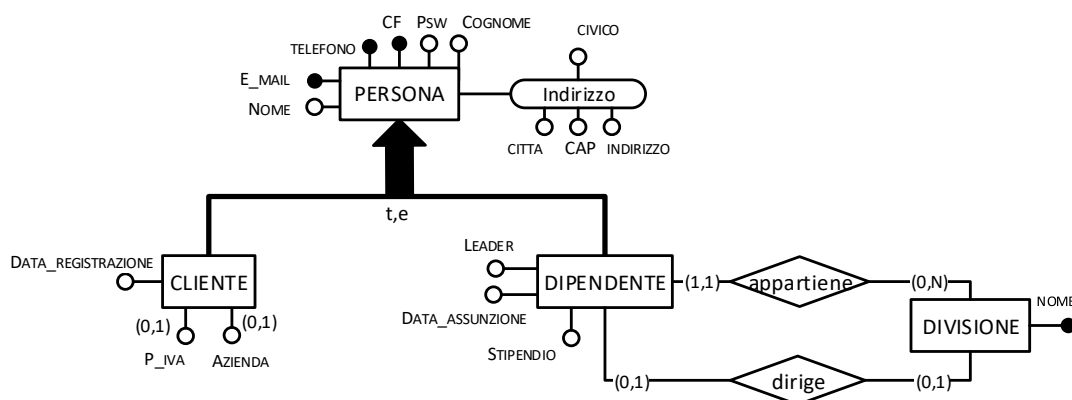


Figura 8 - Estratto di E/R sulla gerarchia di PERSONA e DIVISIONE

La gerarchia soprastante è stata organizzata in modo tale da creare tutte le rispettive tabelle, mantenendo quindi le attuali relazioni. Questa scelta è stata presa in via del fatto che esistono operazioni che possono essere svolte esclusivamente da **Dipendente** e altre che potrebbero essere intraprese da entrambe le entità figlie (per cui è conveniente sfruttare l'entità padre).

Da notare la dipendenza tra **Dipendente** e **Divisione**: un dipendente appartiene ad una divisione e una divisione è diretta da un dipendente. Per ovviare al problema delle dipendenze in fase di creazione si è dovuto intervenire separando la creazione della tabella **Dipendente** e la dichiarazione della chiave esterna, ponendola successivamente alla creazione della tabella **Divisione**.

Per ovviare al problema si sarebbe potuto intervenire ponendo l'attributo identificativo della leadership della divisione come attributo booleano di **Dipendente**, ma avrebbe sollevato la possibilità di più capi di dipartimento per ciascuno, richiedendo un ulteriore controllo durante le operazioni di inserimento o modifica. Pertanto, si è deciso di lasciare l'attributo come riferimento a **Dipendente**, in **Divisione** in modo tale da essere unico per ogni entry della tabella.

AK: e_mail
AK: telefono

CLIENTE (CF, data_registrazione, p_iva, azienda)
FK: CF REFERENCES PERSONA

DIPENDENTE (CF, data_assunzione, stipendio, leader, divisione)
FK: CF REFERENCES PERSONA
FK: divisione REFERENCES DIVISIONE

DIVISIONE (nome, CF)
FK: CF REFERENCES DIPENDENTE

3.1.2 Ticket

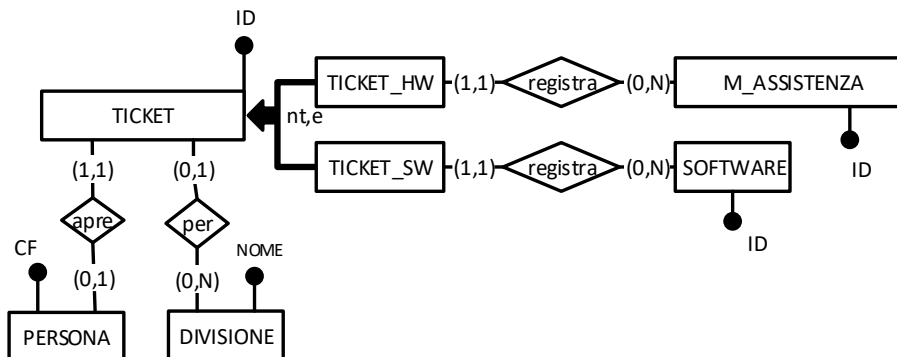


Figura 9 - Estratto E/R sulla gerarchia TICKET

Per la gerarchia si   effettuato un collasso verso l'alto, eliminando quindi le entit  figlie. La scelta   stata condizionata da fattori come l'assenza di attributi sulle entit  figlie (non fornivano alcuna informazione aggiuntiva) e la gestione delle alternative (Ticket generico, Ticket SW, Ticket HW) legata alla **Divisione**, ampliando le possibilit  (Ticket legato ad una divisione) offrendo una migliore gestione e flessibilit  nel caso di una differente organizzazione aziendale da quella da noi proposta.

Il vincolo per cui un Ticket Software e un Ticket Hardware devono avere associato un prodotto (rispettivamente **Software** o **Magazzino Assistenza**) sar  gestita tramite trigger in fase di creazione della relazione o modifica dell'attributo di chiave esterna su **Divisione**.

TICKET (ID, data, stato, descrizione, priorit , prezzo, durata_totale, CF, ID_sw, ID_hw, divisione)
FK: CF REFERENCES PERSONA
FK: ID_sw REFERENCES SOFTWARE
FK: ID_hw REFERENCES M_ASSISTENZA
FK: divisione REFERENCES DIVISIONE

3.2 Schema Logico

Di seguito è infine esposto lo schema logico completo.

PERSONA (CF, nome, cognome, e_mail, psw, telefono, citta, cap, indirizzo, civico)

AK: e_mail

AK: telefono

CLIENTE (CF, data_registrazione, p_iva, azienda)

FK: CF REFERENCES PERSONA

DIPENDENTE (CF, data_assunzione, stipendio, leader, divisione)

FK: CF REFERENCES PERSONA

FK: divisione REFERENCES DIVISIONE

SOFTWARE (ID, nome, versione, stato, SO, descrizione, prezzo)

M_ASSISTENZA (ID, data_entrata, data_uscita, posizione, tipo, descrizione, marca, modello)

M_VENDITA (marca, modello, data, quantita, posizione, tipo, prezzo_listino, descrizione)

CREA (ID, CF, data)

FK: ID REFERENCES SOFTWARE

FK: CF REFERENCES DIPENDENTE

DIVISIONE (nome, CF)

FK: CF REFERENCES DIPENDENTE

ORDINE (ID, data, prezzo_totale, online, dipendente, cliente)

FK: dipendente REFERENCES DIPENDENTE

FK: cliente REFERENCES PERSONA

DI_S (ordine, software, quantita_venduta, prezzo_vendita)

FK: ordine REFERENCES ORDINE

FK: software REFERENCES SOFTWARE

DI_H (ordine, m_vendita, quantita_venduta, prezzo_vendita)

FK: ordine REFERENCES ORDINE

FK: m_vendita REFERENCES M_VENDITA

TICKET (ID, data, stato, descrizione, priorit , prezzo, durata_totale, CF, ID_sw, ID_hw, divisione)

FK: CF REFERENCES PERSONA

FK: ID_sw REFERENCES SOFTWARE

FK: ID_hw REFERENCES M_ASSISTENZA

FK: divisione REFERENCES DIVISIONE

IN_CARICO (CF, data_inizio, data_fine, nota, ID)

FK: CF REFERENCES DIPENDENTE

FK: ID REFERENCES TICKET

4 IMPLEMENTAZIONE - CODICE SQL

4.1 Vincoli aggiuntivi – Trigger

Poiché il database non gestisce direttamente alcuni vincoli è stato necessario implementarli mediante l'uso di trigger e funzioni:

- 1) Il dato prezzo_totale in **Ordine** dev'essere aggiornato in seguito a operazioni effettuate sulle relazioni **di_h** e **di_s**, ovvero all'aggiunta, modifica o eliminazione di prodotti ad un ordine.
- 2) Il dato durata_totale in **Ticket** dev'essere aggiornato in seguito a operazioni effettuate sulla relazione **In_Carico**, ovvero all'aggiunta, modifica o eliminazione di un intervento su di un ticket.
- 3) Solo i ticket associati al dipartimento 'Software' e 'Hardware' posseggono esclusivamente una relazione con la relativa tabella (rispettivamente **Software** e **M_Assistenza**), altrimenti i campi id_hw e id_sw devono essere nulli.
- 4) Solo gli acquisti effettuati presso l'attività commerciale hanno un dipendente del reparto vendite che gestisce la transazione. Altrimenti, se l'**ordine** di acquisto è effettuato online, il campo dev'essere nullo.
- 5) Per ogni **ticket**, solo i dipendenti registrati nel dipartimento dichiarato, possono eseguire interventi.
- 6) Un dipendente può incaricarsi l'intervento solo su di un singolo ticket per volta.
- 7) Il campo contenente la quantità dei prodotti in **M_Vendita** dev'essere aggiornato in seguito a qualsiasi operazione in **di_h**, ovvero all'aggiunta, modifica o eliminazione di prodotti hardware ad un ordine.

4.1.1 Dati derivati

1) Aggiornamento del dato prezzo_totale in Ordine.

```
CREATE FUNCTION prezzo_ordine() RETURNS TRIGGER AS $$
BEGIN
    IF (TG_OP = 'INSERT' OR TG_OP = 'UPDATE') THEN
        UPDATE ORDINE
        SET PREZZO_TOTALE = (SELECT ROUND(CAST(SUM(PREZZO) AS
                                NUMERIC), 2)
                            FROM (SELECT SUM(PREZZO_VENDITA *
                                QUANTITA_VENDUTA) AS PREZZO
                                FROM DI_H
                                WHERE DI_H.ORDINE = NEW.ORDINE
                                UNION
                                SELECT SUM(PREZZO_VENDITA *
                                QUANTITA_VENDUTA) AS PREZZO
                                FROM DI_S
                                WHERE DI_S.ORDINE = NEW.ORDINE)
                            AS X)
        WHERE ORDINE.ID = NEW.ORDINE;
    END IF;
    IF (TG_OP = 'DELETE' OR TG_OP = 'UPDATE') THEN
        UPDATE ORDINE
        SET PREZZO_TOTALE = (SELECT ROUND(CAST(SUM(PREZZO) AS
                                NUMERIC), 2)
                            FROM (SELECT SUM(PREZZO_VENDITA *
                                QUANTITA_VENDUTA) AS PREZZO
                                FROM DI_H
                                WHERE DI_H.ORDINE = OLD.ORDINE
                                UNION
                                SELECT SUM(PREZZO_VENDITA *
                                QUANTITA_VENDUTA) AS PREZZO
                                FROM DI_S
                                WHERE DI_S.ORDINE = OLD.ORDINE)
                            AS X)
        WHERE ORDINE.ID = OLD.ORDINE;
        IF (SELECT PREZZO_TOTALE FROM ORDINE WHERE ORDINE.ID =
            OLD.ORDINE) IS NULL THEN
            DELETE FROM ORDINE WHERE ORDINE.ID = OLD.ORDINE;
        END IF;
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER prezzo_ordine_di_h
AFTER INSERT OR UPDATE OR DELETE
ON DI_H
FOR EACH ROW
EXECUTE PROCEDURE prezzo_ordine();

CREATE TRIGGER prezzo_ordine_di_s
AFTER INSERT OR UPDATE OR DELETE
ON DI_S
FOR EACH ROW
EXECUTE PROCEDURE prezzo_ordine();
```

2) Aggiornamento del dato durata_totale in Ticket.

```
CREATE FUNCTION durata_ticket() RETURNS TRIGGER AS $$
BEGIN
    IF (TG_OP = 'INSERT' OR TG_OP = 'UPDATE') THEN
        UPDATE TICKET
        SET DURATA_TOTALE = (SELECT SUM(IN_CARICO.DATA_FINE -
                                     IN_CARICO.DATA_INIZIO) AS
                             DURATA_TOTALE
                             FROM IN_CARICO
                             WHERE IN_CARICO.ID = NEW.ID)
        WHERE TICKET.ID = NEW.ID;
    END IF;
    IF (TG_OP = 'UPDATE' AND NEW.ID <> OLD.ID) THEN
        UPDATE TICKET
        SET DURATA_TOTALE = (SELECT SUM(IN_CARICO.DATA_FINE -
                                     IN_CARICO.DATA_INIZIO) AS
                             DURATA_TOTALE
                             FROM IN_CARICO
                             WHERE IN_CARICO.ID = OLD.ID)
        WHERE TICKET.ID = OLD.ID;
    END IF;
    IF (TG_OP = 'DELETE') THEN
        IF (SELECT COUNT(*) FROM IN_CARICO WHERE OLD.ID =
            IN_CARICO.ID) > 0 THEN
            UPDATE TICKET
            SET DURATA_TOTALE = (SELECT SUM(IN_CARICO.DATA_FINE -
                                     IN_CARICO.DATA_INIZIO) AS
                                DURATA_TOTALE
                                FROM IN_CARICO
                                WHERE IN_CARICO.ID = OLD.ID)
            WHERE TICKET.ID = OLD.ID;
        ELSE
            UPDATE TICKET
            SET DURATA_TOTALE = NULL
            WHERE TICKET.ID = OLD.ID;
        END IF;
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER durata_ticket
AFTER INSERT OR UPDATE OR DELETE
ON IN_CARICO
FOR EACH ROW
EXECUTE PROCEDURE durata_ticket();
```

3) Verifica che i dati relativi alla **Divisione**, e gli indici a prodotti software o hardware inseriti in un **Ticket**, siano consistenti.

```
CREATE FUNCTION dipartimento_ticket() RETURNS TRIGGER AS $$
BEGIN
    IF (NEW.Divisione = 'Software') THEN
        IF (NEW.ID_SW IS NULL) THEN
            RAISE EXCEPTION 'Ticket Software must have a Software
                ID';
        END IF;
        IF (NEW.ID_HW IS NOT NULL) THEN
            RAISE EXCEPTION 'Ticket Software must not have a
                M_Assistenza ID';
        END IF;
    ELSE
        IF (NEW.Divisione = 'Hardware') THEN
            IF (NEW.ID_HW IS NULL) THEN
                RAISE EXCEPTION 'Ticket Hardware must have a
                    M_Assistenza ID';
            END IF;
            IF (NEW.ID_SW IS NOT NULL) THEN
                RAISE EXCEPTION 'Ticket Hardware must not have a
                    Software ID';
            END IF;
        ELSE
            IF (NEW.ID_SW IS NOT NULL OR NEW.ID_HW IS NOT NULL)
            THEN
                RAISE EXCEPTION 'This ticket must not have a
                    Software or M_Assistenza ID';
            END IF;
        END IF;
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER dipartimento_ticket
    BEFORE INSERT OR UPDATE
    ON TICKET
    FOR EACH ROW
EXECUTE PROCEDURE dipartimento_ticket();
```

4.1.2 Ordine

4) Verifica che gli ordini online non abbiano un dipendente e viceversa.

```
CREATE FUNCTION online_ordine() RETURNS TRIGGER AS $$
BEGIN
    IF (NEW.ONLINE = TRUE AND NEW.DIPENDENTE IS NOT NULL) THEN
        RAISE EXCEPTION 'If online must not have a Dipendente';
    ELSE
        IF (NEW.ONLINE = FALSE) THEN
            IF (NEW.DIPENDENTE IS NULL) THEN
                RAISE EXCEPTION 'If not online must have Dipendente';
            ELSE
                IF (SELECT DIVISIONE
                     FROM DIPENDENTE
                     WHERE CF = NEW.DIPENDENTE) <> 'Vendita' THEN
                    RAISE EXCEPTION 'Dipendente must be in Vendita';
                END IF;
            END IF;
        END IF;
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER online_ordine
BEFORE INSERT OR UPDATE
ON ORDINE
FOR EACH ROW
EXECUTE PROCEDURE online_ordine();
```

4.1.3 In_Carico

- 5) Verifica che un dipendente non possa intervenire su un ticket non appartenente alla sua area lavorativa.

```
CREATE FUNCTION divisione_in_carico() RETURNS TRIGGER AS $$
BEGIN
    IF (SELECT DIVISIONE
        FROM TICKET
        WHERE NEW.ID = TICKET.ID) <>
        (SELECT DIVISIONE
        FROM DIPENDENTE
        WHERE NEW.CF = DIPENDENTE.CF) THEN
        RAISE EXCEPTION 'Only Dipendente in % can works on the
            selected ticket',
            (SELECT DIVISIONE
            FROM TICKET
            WHERE NEW.ID = TICKET.ID);
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER divisione_in_carico
    BEFORE INSERT OR UPDATE
    ON IN_CARICO
    FOR EACH ROW
EXECUTE PROCEDURE divisione_in_carico();
```

- 6) Verifica che durante l'inserimento di un nuovo incarico da parte di un dipendente non devono esistere interventi a suo nome senza data fine (non ancora portati a termine) o la cui data inizio è compresa tra le date (inizio e fine) di interventi già completati (dallo stesso dipendente).

```
CREATE FUNCTION singolo_in_carico() RETURNS TRIGGER AS $$
BEGIN
    IF (SELECT COUNT(*)
        FROM IN_CARICO
        WHERE CF = NEW.CF
        AND DATA_FINE IS NULL) > 0
        OR ( SELECT COUNT(*)
            FROM IN_CARICO
            WHERE CF = NEW.CF
            AND (NEW.DATA_INIZIO BETWEEN DATA_INIZIO AND
                DATA_FINE
                OR NEW.DATA_FINE BETWEEN DATA_INIZIO AND
                DATA_FINE)) > 0 THEN
        RAISE EXCEPTION 'ONLY ONE INTERVENTION IS ALLOWED AT A
            TIME';
    END IF;
    RETURN NEW;
END
$$ LANGUAGE 'plpgsql';
```

```

CREATE TRIGGER singolo_in_carico
BEFORE INSERT OR UPDATE
ON IN_CARICO
FOR EACH ROW
EXECUTE PROCEDURE singolo_in_carico();

```

4.1.4 M_Vendita

- 7) Viene aggiornata la quantità dei prodotti ancora disponibili in magazzino. Sono considerati tutti i casi possibili : inserimento, eliminazione e modifica della tupla di un prodotto in un ordine.
- Nell’inserimento quantita in **M_Vendita** è diminuita se l’operazione non porta ad avere una disponibilità negativa.
- L’eliminazione della tupla aumenta la disponibilità del prodotto eliminato dall’ordine del quantitativo indicato
- La modifica deve invece considerare due possibili eventi: la modifica della sola quantità del prodotto, e/o la modifica del prodotto stesso. Nel primo caso è necessario verificare che la nuova disponibilità (calcolata come quantità in M_Vendita + quantità di **di_h** prima della modifica – quantità di **di_h** dopo la modifica) sia non-negativa. Nel secondo caso invece bisogna controllare che la tupla dopo la modifica porti ad avere ancora una quantità non-negativa per il prodotto considerato; in tal caso è possibile aggiornare il **M_Vendita** come nel caso di inserimento (sul prodotto selezionato in seguito alla modifica) e di una eliminazione (sul prodotto selezionato prima della modifica).

```

CREATE FUNCTION quantita_m_vendita() RETURNS TRIGGER AS $$
BEGIN
    IF (TG_OP = 'INSERT') THEN
        IF (SELECT MV.QUANTITA
            FROM M_VENDITA MV
            WHERE MV.MARCA = NEW.MARCA
            AND MV.MODELLO = NEW.MODELLO) -
            NEW.QUANTITA_VENDUTA >= 0 THEN
            UPDATE M_VENDITA MV
            SET QUANTITA = QUANTITA - NEW.QUANTITA_VENDUTA
            WHERE MV.MARCA = NEW.MARCA
            AND MV.MODELLO = NEW.MODELLO;
        ELSE
            RAISE EXCEPTION 'The selected quantity isn't
                             available for this product';
        END IF;
    ELSEIF (TG_OP = 'DELETE') THEN
        UPDATE M_VENDITA MV
        SET QUANTITA = QUANTITA + OLD.QUANTITA_VENDUTA
        WHERE MV.MARCA = OLD.MARCA
        AND MV.MODELLO = OLD.MODELLO;
    ELSEIF (TG_OP = 'UPDATE') THEN

```

```

IF (NEW.MARCA = OLD.MARCA
    AND NEW.MODELLO = OLD.MODELLO
    AND NEW.QUANTITA_VENDUTA <> OLD.QUANTITA_VENDUTA)
THEN
    IF (SELECT MV.QUANTITA
        FROM M_VENDITA MV
        WHERE MV.MARCA = NEW.MARCA
            AND MV.MODELLO = NEW.MODELLO) +
        (OLD.QUANTITA_VENDUTA -
        NEW.QUANTITA_VENDUTA) >= 0 THEN
        UPDATE M_VENDITA MV
        SET QUANTITA = QUANTITA + (OLD.QUANTITA_VENDUTA
            - NEW.QUANTITA_VENDUTA)
        WHERE MV.MARCA = NEW.MARCA
            AND MV.MODELLO = NEW.MODELLO;
    ELSE
        RAISE EXCEPTION 'The selected quantity isn''t
            available for this product';
    END IF;
ELSE
    UPDATE M_VENDITA MV
    SET QUANTITA = QUANTITA + OLD.QUANTITA_VENDUTA
    WHERE MV.MARCA = OLD.MARCA
        AND MV.MODELLO = OLD.MODELLO;
    UPDATE M_VENDITA MV
    SET QUANTITA = QUANTITA - NEW.QUANTITA_VENDUTA
    WHERE MV.MARCA = NEW.MARCA
        AND MV.MODELLO = NEW.MODELLO;
    END IF;
END IF;
RETURN NEW;
END
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER quantita_m_vendita
BEFORE INSERT OR UPDATE OR DELETE
ON DI_H
FOR EACH ROW
EXECUTE PROCEDURE quantita_m_vendita();

```

4.2 Operazioni preliminari

4.2.1 Creazione dei dati

Sono stati usati i due seguenti tipi enumerativi per le colonne che richiedono uno “status”, ovvero le tabelle **Software** e **Ticket**, per fare in modo che non sia possibile utilizzare valori differenti.

```
CREATE TYPE s_status AS ENUM ('Proposed', 'Rejected', 'Approved', 'In planning', 'In development', 'In testing', 'Quality control', 'Issue', 'Active', 'Need view', 'At risk', 'Closed', 'Suspended');
```

```
CREATE TYPE t_status AS ENUM ('On hold', 'Rejected', 'Approved', 'Scheduled', 'In progress', 'Completed');
```

Mentre di seguito sono elencate le query di creazione delle tabelle usate nel database.

```
CREATE TABLE persona
(
    cf          character varying(16) NOT NULL,
    nome        text                  NOT NULL,
    cognome     text                  NOT NULL,
    e_mail      text UNIQUE           NOT NULL,
    psw         text                  NOT NULL,
    telefono    text UNIQUE           NOT NULL,
    citta       text                  NOT NULL,
    cap         text                  NOT NULL,
    indirizzo   text                  NOT NULL,
    civico      text                  NOT NULL,
    PRIMARY KEY (cf)
);
```

```
CREATE TABLE software
(
    id          serial   NOT NULL,
    nome        text     NOT NULL,
    versione    text     NOT NULL,
    stato       s_status NOT NULL,
    so          text     NOT NULL,
    descrizione  text     NOT NULL,
    prezzo_listino double precision,
    PRIMARY KEY (id),
    CHECK (prezzo_listino > 0 OR NULL)
);
```

```

CREATE TABLE m_assistenza
(
    id                bigserial                NOT NULL,
    data_entrata      timestamp without time zone NOT NULL,
    data_uscita       timestamp without time zone,
    posizione         text,
    tipo              text                    NOT NULL,
    descrizione       text,
    marca             text,
    modello           text,
    PRIMARY KEY (id),
    CHECK (data_entrata < data_uscita OR data_uscita IS NULL)
);

CREATE TABLE m_vendita
(
    marca             text                    NOT NULL,
    modello           text                    NOT NULL,
    data              timestamp without time zone NOT NULL,
    quantita          integer                 NOT NULL,
    posizione         text,
    tipo              text                    NOT NULL,
    prezzo_listino    double precision        NOT NULL,
    descrizione       text,
    PRIMARY KEY (marca, modello),
    CHECK (prezzo_listino > 0 )
);

CREATE TABLE cliente
(
    cf                character varying(16)    NOT NULL,
    data_registrazione timestamp without time zone NOT NULL,
    p_iva             text,
    azienda           text,
    PRIMARY KEY (cf),
    FOREIGN KEY (cf) REFERENCES persona (cf) ON UPDATE CASCADE ON
DELETE NO ACTION
);

CREATE TABLE dipendente
(
    cf                character varying(16)    NOT NULL,
    data_assunzione   timestamp without time zone NOT NULL,
    stipendio          double precision         NOT NULL,
    leader            boolean                  NOT NULL,
    divisione         text                    NOT NULL,
    PRIMARY KEY (cf),
    FOREIGN KEY (cf) REFERENCES persona (cf) ON UPDATE CASCADE ON
DELETE NO ACTION
);

```

```

CREATE TABLE divisione
(
    nome text NOT NULL,
    cf character varying(16),
    PRIMARY KEY (nome),
    FOREIGN KEY (cf) REFERENCES dipendente (cf) ON UPDATE CASCADE
ON DELETE CASCADE
);
ALTER TABLE ONLY dipendente
    ADD FOREIGN KEY (divisione) REFERENCES divisione (nome) ON
UPDATE CASCADE ON DELETE SET NULL;

CREATE TABLE crea
(
    id serial NOT NULL,
    cf character varying(16) NOT NULL,
    data timestamp without time zone NOT NULL,
    PRIMARY KEY (id, cf),
    FOREIGN KEY (cf) REFERENCES dipendente (cf) ON UPDATE CASCADE
ON DELETE NO ACTION,
    FOREIGN KEY (id) REFERENCES software (id) ON UPDATE CASCADE ON
DELETE NO ACTION
);

CREATE TABLE ordine
(
    id bigserial NOT NULL,
    data timestamp without time zone NOT NULL,
    prezzo_totale double precision,
    online boolean NOT NULL,
    dipendente character varying(16),
    cliente character varying(16),
    PRIMARY KEY (id),
    FOREIGN KEY (cliente) REFERENCES persona (cf) ON UPDATE
CASCADE ON DELETE NO ACTION,
    FOREIGN KEY (dipendente) REFERENCES dipendente (cf) ON UPDATE
CASCADE ON DELETE NO ACTION,
    CHECK (prezzo_totale > 0 OR NULL)
);

CREATE TABLE di_h
(
    ordine bigint NOT NULL,
    marca text NOT NULL,
    modello text NOT NULL,
    quantita_venduta integer NOT NULL,
    prezzo_vendita double precision NOT NULL,
    PRIMARY KEY (ordine, marca, modello),
    FOREIGN KEY (marca, modello) REFERENCES m_vendita (marca,
modello) ON UPDATE NO ACTION ON DELETE NO ACTION,
    FOREIGN KEY (ordine) REFERENCES ordine (id) ON UPDATE CASCADE
ON DELETE CASCADE,
    CHECK (prezzo_vendita > 0 OR NULL)
);

```

```

CREATE TABLE di_s
(
    ordine          bigint          NOT NULL,
    software        integer         NOT NULL,
    quantita_venduta integer         NOT NULL,
    prezzo_vendita  double precision NOT NULL,
    PRIMARY KEY (ordine, software),
    FOREIGN KEY (ordine) REFERENCES ordine (id) ON UPDATE CASCADE
ON DELETE CASCADE,
    FOREIGN KEY (software) REFERENCES software (id) ON UPDATE
CASCADE ON DELETE NO ACTION,
    CHECK (prezzo_vendita > 0 OR NULL)
);

CREATE TABLE ticket
(
    id              bigserial        NOT NULL,
    data            timestamp without time zone NOT NULL,
    stato           t_status         NOT NULL,
    descrizione     text              NOT NULL,
    priorita        smallint         NOT NULL,
    prezzo          double precision,
    durata_totale   interval,
    cf              character varying(16) NOT NULL,
    id_sw           integer,
    id_hw           bigint,
    divisione       text,
    PRIMARY KEY (id),
    FOREIGN KEY (cf) REFERENCES persona (cf) ON UPDATE CASCADE ON
DELETE NO ACTION,
    FOREIGN KEY (divisione) REFERENCES divisione (nome) ON UPDATE
CASCADE ON DELETE SET NULL,
    FOREIGN KEY (id_hw) REFERENCES m_assistenza (id) ON UPDATE
CASCADE ON DELETE CASCADE,
    FOREIGN KEY (id_sw) REFERENCES software (id) ON UPDATE CASCADE
ON DELETE CASCADE,
    CHECK (priorita BETWEEN 1 AND 9 ),
    CHECK (prezzo > 0 OR NULL)
);

CREATE TABLE in_carico
(
    cf              text              NOT NULL,
    data_inizio     timestamp without time zone NOT NULL,
    data_fine       timestamp without time zone,
    nota            text,
    id              bigint            NOT NULL,
    PRIMARY KEY (cf, data_inizio),
    FOREIGN KEY (cf) REFERENCES dipendente (cf) ON UPDATE NO
ACTION ON DELETE NO ACTION,
    FOREIGN KEY (id) REFERENCES ticket (id) ON UPDATE CASCADE ON
DELETE CASCADE,
    CHECK (data_inizio < data_fine)
);

```

4.2.2 Inserimento dei dati

Di seguito è riportato il codice per l'inserimento dei dati per il popolamento delle tabelle del database. Si fa notare che a causa dell'eccessiva dimensione è riportato, per ogni tabella, solamente un estratto del codice realmente utilizzato per il testing.

Lo script completo, contenente quindi tutte le informazioni, è possibile recuperarlo, ed eventualmente utilizzarlo, nel file "insert_script.sql" allegato a questa relazione.

Nella prima parte del file è presente anche tutto il necessario per la distruzione e la creazione delle tabelle (e dei trigger) e di tutti gli elementi impiegati nel database.

Si vuole far notare che il codice sottostante è solo una versione ridotta del reale database popolato utilizzato al solo scopo dimostrativo. Per un elenco più dettagliato fare riferimento al file di creazione citato.

```
INSERT INTO SOFTWARE (nome, versione, stato, so, descrizione, prezzo_listino)
VALUES ('Stratosphere', '15.98', 'Issue', 'ChromeOS', 'Il software per tutte le vostre esigenze.',
'3.5'),
('Connections', '165.32', 'Need view', 'OsX', 'Il miglior prodotto valutato da Hardware
Upgrade', '10'),
('Drivers', '1.1', 'Need view', 'Windows 10', 'Un prodotto completo e semplice', '10'),
('GameFan', '2.12.435', 'In testing', 'IOs', 'Un prodotto completo e semplice', NULL),
('BestRecording', '2.12.435', 'Quality control', 'Android', 'Il software per tutte le vostre
esigenze.', '40'),
('OfficeManager', '76.4.23', 'At risk', 'OsX', 'Il software per tutte le vostre esigenze.', '25'),
('SuperPhotoEditor', '76.4.23', 'Issue', 'ChromeOS', 'Il software per tutte le vostre esigenze.',
'10'),
('3D Designer', '0.4.1254', 'Quality control', 'Android', 'Il software per tutte le vostre esigenze.',
'15'),
('VirusGuardX', '76.4.23', 'In development', 'Linux', 'Un prodotto completo e semplice', '3.5');

INSERT INTO PERSONA (cf, nome, cognome, e_mail, psw, telefono, citta, cap, indirizzo, civico)
VALUES ('MARBECZZ0JW5S43M', 'Marco', 'Becco', 'marco.becco@aol.com',
'c0wi1gnaybf3me7japx9e4f9', '3517994975', 'Modena', '17171', 'piazza Università', '200'),
('ALEFER7TUZ0YN99Z', 'Alessio', 'Ferrari', 'alessio.ferrari@goowy.com',
'qvlln33ehfbbmlijbanfc5vz', '3421120206', 'Livorno', '89920', 'largo Augusto', '90'),
('ERISALPTEI59POMB', 'Erica', 'Sala', 'erica.sala@outlook.it', 'ycmxt2i23doslc5sbne249o3',
'3100913053', 'Parma', '00792', 'vicolo Corto', '100'),
('ALBSERU2U6F2HF3Z', 'Alberto', 'Serra', 'alberto.serra@yandex.com',
'snfp5yidqaatvbqqohiwdmo0', '3973240120', 'Modena', '94836', 'via Accademia', '33'),
('ALEROSK4BUL185UF', 'Alessia', 'Rossi', 'alessia.rossi@hush.com',
'f4evdqfmmw5r78hbyt7fd0uih', '3214490702', 'Bologna', '81494', 'corso Ateneo', '21'),
('FRAMENPGV7QO2S6Z', 'Franco', 'Menta', 'franco.menta@goowy.com',
'2crskvxnq737e2hm7mzm63sw', '3085405958', 'Bologna', '63163', 'vicolo Stretto', '27'),
('GIOLEOW2DWHUE9SH', 'Giovanni', 'Leone', 'giovanni.leone@outlook.it',
'aacq34bdcyv4umw44ablk7q3', '3020495775', 'Roma', '97876', 'piazza Baglioni', '61'),
('GIOBEC4F477QA6NC', 'Giorgio', 'Becco', 'giorgio.becco@fastmail.fm',
```

'jqsr839koe0edc46e7xzdhc', '3035524200', 'Parma', '10546', 'Corso Raffaello', '131'),
 ('ANGLOMZNUABIOBSI', 'Angelo', 'Lombardi', 'angelo.lombardi@live.com',
 '9e2r8vyx1o85zmc8z9fwa5i', '3119223441', 'Livorno', '65896', 'piazza Università', '91'),
 ('GLOBOTV6GPM9PDH6', 'Giovanni', 'Botta', 'giovanni.botta@yahoo.com',
 'v08ip4h2vg9ng32c8tk10g1k', '3547425090', 'Livorno', '51396', 'viale dei Giardini', '17'),
 ('GIOINNRDDY7UJQB9', 'Giorgio', 'Innocenti', 'giorgio.innocenti@shortmail.com',
 '003r8vbn9tqd6hfn7d6rwpb7', '3936639936', 'Bologna', '44113', 'viale dei Giardini', '190'),
 ('ENRLOMMHNE0GDWIN', 'Enrico', 'Lombardi', 'enrico.lombardi@hushmail.com',
 '5igs8ujhde75v7ps802dsu42', '3864074298', 'Livorno', '04581', 'Piazza Dante', '95'),
 ('MARR0M2L9C78R4R6', 'Mario', 'Romano', 'mario.romano@lycos.com',
 'eyg7ir82bnx1l74u39keryle', '3852112814', 'Livorno', '32945', 'viale Vesuvio', '70'),
 ('MASSEREVB32IPZP', 'Massimo', 'Serra', 'massimo.serra@unimore.it',
 'w3ys99aj4uz42yz0jvbb7dm', '3984208155', 'Roma', '58960', 'parco della Vittoria', '96'),
 ('FRAFONRG60NZ2GPH', 'Francesca', 'Fontana', 'francesca.fontana@inbox.com',
 '5x9qufvpwp4spn5alwhy8fmg', '3640926529', 'Parma', '69889', 'vicolo Stretto', '144'),
 ('MARFONTLSSMGVN1', 'Mario', 'Fontana', 'mario.fontana@shortmail.com',
 'd68k6pwwji5rrd6un7nhmhho', '3700758058', 'Torino', '18718', 'via Marco Polo', '21'),
 ('GIOGENXB6S7EX28H', 'Giorgio', 'Gentile', 'giorgio.gentile@live.it',
 '9ye3c254k5gw8gmlijrw269f', '3776277734', 'Bologna', '71461', 'viale Italia', '141'),
 ('LORGRENSNZ7V93VK', 'Lorenzo', 'Greco', 'lorenzo.greco@fastmail.fm',
 'nbvy68s2rdm2a7c1txwv9hyd', '3975061878', 'Bologna', '59211', 'corso Impero', '33'),
 ('GIOLOMP2S399T882', 'Giovanni', 'Lombardi', 'giovanni.lombardi@fastmail.fm',
 'zmfztked9xkd79dr9gek9m78', '3506042731', 'Livorno', '90319', 'via Accademia', '155'),
 ('MARMONIQHMAZYB0ZY', 'Marianna', 'Monti', 'marianna.monti@hushmail.com',
 '45lzi9l1evlpk7ugz2taldmi', '3268177533', 'Modena', '89935', 'piazza Baglioni', '155'),
 ('ENRMILJ0609QFBEZ', 'Enrico', 'Milano', 'enrico.milano@unimore.it',
 '0e2b6sze46yfpt7e4pl1subi', '3666375556', 'Modena', '13933', 'viale Traiano', '40'),
 ('CHIMON4MMIFWEM6D', 'Chiara', 'Monti', 'chiara.monti@yandex.com',
 'zdrcc8yix6sxfehrweu47ska', '3077517138', 'Torino', '94816', 'viale Vesuvio', '78'),
 ('MARGREPHE2ZZI8P', 'Maria', 'Greco', 'maria.greco@inbox.com',
 'frkm4ji1rwzgcuehnhuxddkf', '3143239986', 'Torino', '10738', 'Piazza Dante', '32'),
 ('LORBIA1EGE04UDZL', 'Lorenzo', 'Bianchi', 'lorenzo.bianchi@goowy.com',
 'wxr3csb2k2ewoifhmt1i9e9', '3321701307', 'Modena', '55444', 'via Bach', '160'),
 ('ANGFONAM9SEFWK0L', 'Angelo', 'Fontana', 'angelo.fontana@yahoo.com',
 'ptf70wwwvjz1rrgmpzxzg80', '3127760725', 'Torino', '47988', 'vicolo Stretto', '101');

INSERT INTO M_ASSISTENZA (data_entrata, data_uscita, posizione, tipo, descrizione, marca, modello)

VALUES ('2018-08-31 01:24:50', '2019-05-08 04:01:15', 'nggrwz', 'Netbook', NULL, 'Asus', 'Air'),
 ('2018-03-03 18:43:20', '2019-04-03 02:40:14', 'lsblxd', 'Laptop', NULL, 'Netgear', 'Galaxy'),
 ('2018-09-26 05:19:19', '2019-05-23 14:19:02', 'ma8n88', 'Fotocamera', NULL, 'D-Link', 'Air'),
 ('2018-10-23 01:18:15', '2019-04-15 00:57:04', 'ouv43x', 'Monitor', NULL, 'Asus', 'Jet3480'),
 ('2018-10-31 05:18:17', NULL, 'sizz73', 'Desktop', NULL, 'Lenovo', 'QL17'),
 ('2018-04-01 14:22:32', '2019-04-19 14:25:38', '45mjb8', 'Fotocamera', NULL, 'D-Link', '600D'),
 ('2019-03-21 13:52:35', '2019-05-15 01:33:00', 'knjt8i', 'Fotocamera', NULL, 'Sony', 'K400'),
 ('2018-06-05 00:55:54', '2019-03-12 09:21:27', 'kl9hy7', 'Smartphone', NULL, 'Wiko', 'Alfa'),
 ('2019-02-11 18:20:04', NULL, '954a0h', 'Access Point', NULL, 'Wiko', 'S'),
 ('2018-05-02 15:32:55', '2019-04-05 07:47:34', '12rqza', 'Laptop', NULL, 'Huawei', 'Pro Max'),
 ('2018-09-21 18:54:45', '2019-01-23 12:56:52', 'a47vqx', 'Monitor', NULL, 'Netgear', 'Evo'),
 ('2018-04-23 11:56:32', NULL, 'q9k3uo', 'Access Point', NULL, 'Netgear', '20'),
 ('2019-06-06 19:47:35', '2019-06-09 02:17:11', 'x0pvog', 'Netbook', NULL, 'Acer', 'Seven'),
 ('2018-06-07 10:05:11', NULL, '4xz8jq', 'Desktop', NULL, 'Wiko', 'Seven'),
 ('2018-01-15 09:40:19', '2018-04-23 22:19:12', '2bh7t5', 'Netbook', NULL, 'Nikon', 'Pro 6');

INSERT INTO M_VENDITA (marca, modello, data, quantita, posizione, tipo, prezzo_listino, descrizione)

VALUES ('Apple', 'Aspire', '2018-07-12 19:37:28', '5', '57cdyr', 'Smartphone', '1078.78', NULL),
 ('Nikon', 'KRT', '2018-03-03 02:50:26', '26', '1glz7o', 'Access Point', '194.85',


```

'La massima velocità nelle tue mani'),
('Lenovo', 'S', '2018-07-19 05:13:17', '2', 'qgwwym', 'Smartphone', '1301.46', NULL),
('Netgear', 'Delta', '2018-04-19 22:44:37', '29', 'vesj70', 'Monitor', '570.54', NULL),
('Sony', 'Max', '2019-02-10 22:40:19', '6', '6q5zmp', 'Laptop', '745.28',
'8GB RAM 128GB Memoria Schermo 6.1 pollici'),
('Nikon', '20', '2018-07-23 05:44:38', '22', 'k63rpu', 'Stampante', '1244.4', NULL),
('Dell', 'Air', '2018-10-30 23:18:09', '36', 'waqphh', 'Netbook', '1286.51', NULL),
('Acer', 'S', '2018-05-01 00:07:21', '19', 'jz0k6f', 'Access Point', '1003.65', NULL),
('Lenovo', 'Pro Max', '2018-11-23 19:13:45', '22', '7f2xnp', 'Modem', '20.67', NULL),
('D-Link', 'Gemini', '2018-05-23 09:58:19', '5', 'x7a10o', 'Desktop', '985.83', NULL),
('Wacom', 'Alfa', '2019-01-02 05:41:45', '3', 'j5v814', 'Stampante', '1175.54', NULL),
('Dell', 'Pro Max', '2018-06-14 07:40:37', '26', 'fy0pww', 'Smartphone', '990.75', NULL),
('Cisco', 'Galaxy', '2018-07-14 12:13:55', '14', '3nk1r6', 'Stampante', '768.83', NULL),
('Wiko', 'Intus', '2018-08-29 01:30:41', '29', 'tme0be', 'Modem', '824.86', NULL),
('IBM', 'R', '2018-11-09 11:54:43', '8', 'upm9wv', 'Laptop', '1392.8', NULL),
('Cisco', 'Gemini', '2019-05-21 08:24:48', '17', 'k29vcc', 'Smartphone', '1070.04', NULL),
('HP', 'Alfa', '2018-12-27 06:16:54', '30', 'kzk93m', 'Stampante', '165.95', NULL),
('Lenovo', 'Pro 1', '2019-05-19 14:08:07', '4', '6rh03m', 'Modem', '396.48',
'La massima velocità nelle tue mani'),
('Netgear', 'Bravo', '2018-06-14 12:41:34', '18', 'fe18lp', 'Smartphone', '866.92', NULL),
('IBM', '600D', '2018-07-18 10:23:42', '26', '2hnzes', 'Access Point', '477.39',
'8GB RAM 128GB Memoria Schermo 6.1 pollici');

```

```

INSERT INTO CLIENTE (cf, data_registrazione, p_iva, azienda)
VALUES ('GIOGENXB657EX28H', '2018-03-18 19:32:12', NULL, NULL),
('MARFONTLSSMGVN1', '2018-05-31 00:17:08', NULL, NULL),
('ERISALPTEI59POMB', '2018-04-14 00:11:54', NULL, NULL),
('ENRMILJ0609QFBEZ', '2018-02-14 10:37:44', NULL, NULL),
('MARGREPHE2ZZI8P', '2018-07-19 19:42:43', 'mi6lxznex69', 'Eppol'),
('CHIMON4MMIFWEM6D', '2019-04-01 01:06:11', NULL, NULL),
('ENRLOMMHNE0GDWIN', '2018-10-10 22:33:50', NULL, NULL),
('GIOINNRDDY7UJQB9', '2018-12-29 23:56:58', NULL, NULL),
('ANGFONAM95EFWK0L', '2018-08-20 10:47:50', NULL, NULL),
('ALEROSK4BUL185UF', '2018-04-07 05:08:31', NULL, NULL),
('LORGRENSNZ7V93VK', '2018-02-27 03:48:34', NULL, NULL),
('GIOLOMP2S399T882', '2018-01-21 15:58:13', NULL, NULL),
('GIBOTV6PM9PDH6', '2019-05-20 09:52:39', NULL, NULL),
('ANGLOMZNUABI0BSI', '2019-06-10 03:31:05', NULL, NULL),
('GIOLEOW2DWHUE9SH', '2019-06-07 17:22:34', 'kccpc04bf5m', 'CalTec'),
('FRAMENPGV7QO2S6Z', '2018-04-18 19:51:46', NULL, NULL),
('ALBSERU2U6F2HF3Z', '2018-01-12 03:24:23', NULL, NULL),
('MASSEREVBC32IPZP', '2019-01-11 23:23:52', NULL, NULL),
('MARROM2L9C78R4R6', '2018-07-06 14:45:28', '6zvoizg4gb4', 'CatCo'),
('LORBIA1EGE04UDZL', '2019-04-22 11:32:25', NULL, NULL);

```

```

INSERT INTO DIVISIONE (nome, cf)
VALUES ('Software', NULL),
('Hardware', NULL),
('Vendita', NULL),
('Direzione', NULL);

```

```

INSERT INTO DIPENDENTE (cf, data_assunzione, stipendio, leader, divisione)
VALUES ('FRAFONRG60NZ2GPH', '2018-05-08 03:21:11', '1568.88', True, 'Hardware'),
('ALEFER7TUZOYN99Z', '2019-03-19 21:33:22', '2771.13', True, 'Hardware'),
('MARBECZZ0JW5S43M', '2018-06-07 06:53:15', '910.02', True, 'Vendita'),
('MARMONIQHMYB0ZY', '2018-12-29 19:00:54', '1073.07', True, 'Software'),
('GIOBEC4F477QA6NC', '2018-09-29 18:55:59', '2292.31', False, 'Direzione');

```

```

UPDATE DIVISIONE SET cf = 'MARMONIQHMAYBOZY' WHERE nome = 'Software';
UPDATE DIVISIONE SET cf = 'ALEFER7TUZOYN99Z' WHERE nome = 'Hardware';
UPDATE DIVISIONE SET cf = 'MARBECZZ0JW5S43M' WHERE nome = 'Vendita';
UPDATE DIVISIONE SET cf = 'GIOBEC4F477QA6NC' WHERE nome = 'Direzione';

```

```

INSERT INTO CREA (id, cf, data)

```

```

VALUES ('3', 'MARMONIQHMAYBOZY', '2018-06-16 13:27:49'),
      ('9', 'MARMONIQHMAYBOZY', '2018-08-31 20:08:45'),
      ('4', 'MARMONIQHMAYBOZY', '2018-03-28 18:00:57'),
      ('5', 'MARMONIQHMAYBOZY', '2019-05-11 08:11:01'),
      ('6', 'MARMONIQHMAYBOZY', '2018-11-22 05:43:29'),
      ('1', 'MARMONIQHMAYBOZY', '2018-02-07 11:26:51'),
      ('8', 'MARMONIQHMAYBOZY', '2018-06-05 14:34:44'),
      ('2', 'MARMONIQHMAYBOZY', '2018-08-09 07:01:18'),
      ('7', 'MARMONIQHMAYBOZY', '2018-02-24 20:36:00');

```

```

INSERT INTO ORDINE (data, online, dipendente, cliente)

```

```

VALUES ('2019-01-05 22:11:38', False, 'MARBECZZ0JW5S43M', 'MARBECZZ0JW5S43M'),
      ('2018-02-06 03:52:39', True, NULL, 'ALEFER7TUZOYN99Z'),
      ('2018-07-25 21:33:07', True, NULL, 'ENRLOMMHNEOGDWIN'),
      ('2018-05-22 08:54:20', False, 'MARBECZZ0JW5S43M', 'ERISALPTEI59POMB'),
      ('2018-05-31 15:16:50', True, NULL, 'MARGREPHE2ZZII8P'),
      ('2019-05-17 16:17:33', False, 'MARBECZZ0JW5S43M', 'LORGRENSNZ7V93VK'),
      ('2019-05-04 13:16:08', True, NULL, 'MARBECZZ0JW5S43M'),
      ('2018-06-22 20:27:02', False, 'MARBECZZ0JW5S43M', 'LORBIA1EGE04UDZL'),
      ('2018-05-25 18:09:37', True, NULL, 'LORGRENSNZ7V93VK'),
      ('2018-11-28 13:59:30', True, NULL, 'ERISALPTEI59POMB'),
      ('2019-05-20 02:53:47', False, 'MARBECZZ0JW5S43M', 'ALEROSK4BUL185UF'),
      ('2018-10-29 17:14:23', False, 'MARBECZZ0JW5S43M', 'ERISALPTEI59POMB'),
      ('2018-08-30 06:32:34', True, NULL, 'ENRLOMMHNEOGDWIN'),
      ('2019-05-07 23:11:21', False, 'MARBECZZ0JW5S43M', 'MASSEREVBC32IPZP'),
      ('2018-03-15 08:18:25', False, 'MARBECZZ0JW5S43M', 'GIOLEOW2DWHUE9SH');

```

```

INSERT INTO DI_H (ordine, marca, modello, quantita_venduta, prezzo_vendita)

```

```

VALUES ('3', 'Nikon', '20', '10', '171.32'),
      ('8', 'IBM', 'R', '8', '757.45'),
      ('4', 'Netgear', 'Bravo', '1', '558.91'),
      ('6', 'Acer', 'S', '2', '475.61'),
      ('1', 'Sony', 'Max', '3', '232.68'),
      ('10', 'Acer', 'S', '4', '930.16'),
      ('11', 'Lenovo', 'Pro Max', '3', '21.79'),
      ('13', 'HP', 'Alfa', '4', '102.14'),
      ('10', 'Lenovo', 'S', '1', '507.05'),
      ('12', 'IBM', 'R', '5', '1131.22'),
      ('2', 'Lenovo', 'Pro Max', '7', '5.88'),
      ('9', 'Netgear', 'Delta', '9', '308.86'),
      ('5', 'Wacom', 'Alfa', '8', '952.21'),
      ('5', 'Acer', 'S', '4', '112.78'),
      ('10', 'IBM', 'R', '8', '336.14');

```

```

INSERT INTO DI_S (ordine, software, quantita_venduta, prezzo_vendita)

```

```

VALUES ('14', '2', '6', '3.14'),
      ('12', '7', '1', '11.31'),
      ('3', '1', '3', '10.0'),
      ('4', '7', '2', '5.4'),
      ('7', '6', '7', '8.72'),
      ('15', '3', '6', '7.23');

```

('3', '8', '2', '13.33');

```
INSERT INTO TICKET (data, stato, descrizione, priorit , prezzo, durata_totale, cf, id_sw, id_hw,
divisione)
VALUES ('2018-01-15 09:40:19', 'Scheduled', 'Sostituzione batteria', '1', NULL, NULL,
'MARROM2L9C78R4R6', NULL, '15',
'Hardware'),
('2018-02-07 11:26:51', 'On hold', 'Miglioramento delle prestazioni nel ciclo parallelo', '3',
NULL, NULL,
'MARMONIQHMAYBOZY', '1', NULL, 'Software'),
('2018-02-22 19:06:44', 'In progress', 'Errore in lettura da file', '6', NULL, NULL,
'MARMONIQHMAYBOZY', '1',
NULL, 'Software'),
('2018-02-24 20:36:00', 'Scheduled', 'Miglioramento delle prestazioni nel ciclo parallelo', '3',
NULL, NULL,
'MARMONIQHMAYBOZY', '7', NULL, 'Software'),
('2018-03-03 18:43:20', 'Rejected', 'Sostituzione batteria', '9', NULL, NULL,
'ENRLOMMHNE0GDWIN', NULL, '2',
'Hardware'),
('2018-03-23 19:03:04', 'Approved', 'Bug fixed on lib/iostream/', '5', NULL, NULL,
'MARMONIQHMAYBOZY', '7', NULL,
'Software'),
('2018-03-28 18:00:57', 'Completed', 'Errore in lettura da file', '7', NULL, NULL,
'MARMONIQHMAYBOZY', '4', NULL,
'Software'),
('2018-04-01 14:22:32', 'Completed', 'Upgrade Ram', '1', NULL, NULL, 'MARGREPHE2ZZI8P',
NULL, '6', 'Hardware'),
('2018-04-07 22:25:06', 'In progress', 'Modifica alla grafica di visualizzazione', '5', NULL, NULL,
'MARMONIQHMAYBOZY', '4', NULL, 'Software'),
('2018-04-20 21:47:07', 'Approved', 'Modifica della mission aziendale', '5', NULL, NULL,
'GIOBEC4F477QA6NC',
NULL, NULL, 'Direzione'),
('2018-04-23 11:56:32', 'Rejected', 'Schermo da sostituire', '7', NULL, NULL,
'CHIMON4MMIFWEM6D', NULL, '12',
'Hardware'),
('2018-04-24 09:29:17', 'On hold', 'Apertura assunzioni reparto Software', '9', NULL, NULL,
'GIOBEC4F477QA6NC',
NULL, NULL, 'Direzione'),
('2018-05-02 15:32:55', 'In progress', 'Sostituzione scheda madre', '2', NULL, NULL,
'MARFONTLLS5MGVN1', NULL,
'10', 'Hardware'),
('2018-06-02 02:37:46', 'In progress', 'Analisi di bilancio mensile', '7', NULL, NULL,
'GIOBEC4F477QA6NC', NULL,
NULL, 'Direzione'),
('2018-06-05 00:55:54', 'Completed', 'Sostituzione scheda madre', '5', NULL, NULL,
'MARGREPHE2ZZI8P', NULL, '8',
'Hardware'),
('2018-06-05 14:34:44', 'Completed', 'Errore in lettura da file', '4', NULL, NULL,
'MARMONIQHMAYBOZY', '8', NULL,
'Software'),
('2018-06-07 10:05:11', 'Rejected', 'Upgrade Ram', '9', NULL, NULL, 'FRAFONRG60NZ2GPH',
NULL, '14', 'Hardware'),
('2018-06-16 13:27:49', 'Rejected', 'Modifica alla grafica di visualizzazione', '6', NULL, NULL,
'MARMONIQHMAYBOZY', '3', NULL, 'Software'),
('2018-06-16 16:53:56', 'On hold', 'Modifica alla grafica di visualizzazione', '8', NULL, NULL,
'MARMONIQHMAYBOZY', '8', NULL, 'Software'),
('2018-06-19 16:50:12', 'On hold', 'Update README.md', '9', NULL, NULL,
'MARMONIQHMAYBOZY', '3', NULL,
```

'Software'),
 ('2018-07-27 05:06:49', 'Approved', 'Calo delle prestazioni in assistenza', '3', NULL, NULL,
 'GIOBEC4F477QA6NC',
 NULL, NULL, 'Direzione'),
 ('2018-08-09 07:01:18', 'In progress', 'Modifica alla grafica di visualizzazione', '1', NULL, NULL,
 'MARMONIQHMAYBOZY', '2', NULL, 'Software'),
 ('2018-08-31 01:24:50', 'Scheduled', 'Upgrade Ram', '8', NULL, NULL, 'ALEFER7TUZ0YN99Z',
 NULL, '1', 'Hardware'),
 ('2018-08-31 20:08:45', 'Completed', 'Bug fixed on lib/iostream/', '2', NULL, NULL,
 'MARMONIQHMAYBOZY', '9',
 NULL, 'Software'),
 ('2018-09-05 04:38:15', 'In progress', 'Modifica alla grafica di visualizzazione', '2', NULL, NULL,
 'MARMONIQHMAYBOZY', '2', NULL, 'Software'),
 ('2018-09-18 17:18:09', 'Approved', 'Update README.md', '4', NULL, NULL,
 'MARMONIQHMAYBOZY', '9', NULL,
 'Software'),
 ('2018-09-21 18:54:45', 'Rejected', 'Sostituzione scheda madre', '1', NULL, NULL,
 'ANGFONAM95EFWK0L', NULL, '11',
 'Hardware'),
 ('2018-09-26 05:19:19', 'Approved', 'Schermo da sostituire', '2', NULL, NULL,
 'MARROM2L9C78R4R6', NULL, '3',
 'Hardware'),
 ('2018-10-09 21:42:56', 'In progress', 'Formazione del personale alle vendite', '8', NULL, NULL,
 'MARBECZZ0JW5S43M', NULL, NULL, 'Vendita'),
 ('2018-10-23 01:18:15', 'Rejected', 'Sostituzione lettore cd', '6', NULL, NULL,
 'LORBIA1EGE04UDZL', NULL, '4',
 'Hardware'),
 ('2018-10-31 05:18:17', 'On hold', 'Sostituzione lettore cd', '1', NULL, NULL,
 'ALBSERU2U6F2HF3Z', NULL, '5',
 'Hardware'),
 ('2018-11-17 11:37:04', 'Approved', 'Modifica della gestione dei rischi', '2', NULL, NULL,
 'GIOBEC4F477QA6NC',
 NULL, NULL, 'Direzione'),
 ('2018-11-22 05:43:29', 'Scheduled', 'Update README.md', '8', NULL, NULL,
 'MARMONIQHMAYBOZY', '6', NULL,
 'Software'),
 ('2018-12-09 12:27:42', 'In progress', 'Richiesta sostituzione per cassiere Sabato', '6', NULL,
 NULL,
 'MARBECZZ0JW5S43M', NULL, NULL, 'Vendita'),
 ('2018-12-13 11:10:46', 'Approved', 'Miglioramento delle prestazioni nel ciclo parallelo', '7',
 NULL, NULL,
 'MARMONIQHMAYBOZY', '6', NULL, 'Software'),
 ('2019-02-11 18:20:04', 'Approved', 'Sostituzione batteria', '6', NULL, NULL,
 'ERISALPTEI59POMB', NULL, '9',
 'Hardware'),
 ('2019-03-21 13:52:35', 'Scheduled', 'Sostituzione scheda madre', '3', NULL, NULL,
 'MARBECZZ0JW5S43M', NULL, '7',
 'Hardware'),
 ('2019-04-20 12:42:16', 'On hold', 'Sostituzione del logo', '6', NULL, NULL,
 'GIOBEC4F477QA6NC', NULL, NULL,
 'Direzione'),
 ('2019-04-23 13:20:07', 'Completed', 'Scaffale portatili da aggiornare', '6', NULL, NULL,
 'MARBECZZ0JW5S43M',
 NULL, NULL, 'Vendita'),
 ('2019-05-11 08:11:01', 'In progress', 'Miglioramento delle prestazioni nel ciclo parallelo', '2',
 NULL, NULL,
 'MARMONIQHMAYBOZY', '5', NULL, 'Software'),
 ('2019-06-05 03:18:00', 'Rejected', 'Miglioramento delle prestazioni nel ciclo parallelo', '1',

```

NULL, NULL,
'MARMONIQHMAYBOZY', '5', NULL, 'Software'),
('2019-06-06 19:47:35', 'Approved', 'Sostituzione batteria', '4', NULL, NULL,
'ENRLOMMHNEOGDWIN', NULL, '13',
'Hardware');

```

INSERT INTO IN_CARICO (cf, data_inizio, data_fine, nota, id)

```

VALUES ('MARMONIQHMAYBOZY', '2018-08-05 00:41:53', '2018-08-05 10:08:22', NULL, 9),
('MARMONIQHMAYBOZY', '2018-10-22 18:37:35', '2018-10-23 07:26:12', NULL, 24),
('MARMONIQHMAYBOZY', '2018-08-06 10:32:46', '2018-08-06 17:44:06', NULL, 7),
('MARMONIQHMAYBOZY', '2019-05-02 22:06:41', '2019-05-03 00:39:38', NULL, 33),
('MARMONIQHMAYBOZY', '2018-10-02 14:41:48', '2018-10-03 08:27:38', NULL, 25),
('MARMONIQHMAYBOZY', '2018-12-26 03:05:21', '2018-12-26 16:11:58', NULL, 7),
('MARMONIQHMAYBOZY', '2018-11-19 11:22:18', '2018-11-20 00:13:09', NULL, 20),
('MARMONIQHMAYBOZY', '2019-05-31 11:54:51', '2019-06-01 07:39:56', NULL, 4),
('MARMONIQHMAYBOZY', '2018-09-08 08:37:59', '2018-09-08 11:25:08', NULL, 22),
('MARMONIQHMAYBOZY', '2018-05-14 01:31:48', '2018-05-14 07:54:51', NULL, 7),
('MARMONIQHMAYBOZY', '2019-01-13 02:45:11', '2019-01-13 10:58:47', NULL, 35),
('MARMONIQHMAYBOZY', '2019-06-09 21:48:57', '2019-06-10 18:01:04', NULL, 41),
('MARMONIQHMAYBOZY', '2018-09-21 13:51:32', '2018-09-22 02:25:18', NULL, 6),
('MARMONIQHMAYBOZY', '2018-06-03 20:53:34', '2018-06-04 20:15:21', NULL, 7),
('MARMONIQHMAYBOZY', '2018-12-22 03:39:25', '2018-12-22 14:54:19', NULL, 3),
('MARMONIQHMAYBOZY', '2019-02-23 04:38:15', '2019-02-23 08:47:51', NULL, 35),
('MARMONIQHMAYBOZY', '2018-12-22 19:21:49', '2018-12-23 03:26:29', NULL, 35),
('MARMONIQHMAYBOZY', '2018-09-10 12:10:42', '2018-09-10 18:23:14', NULL, 3),
('MARMONIQHMAYBOZY', '2018-06-08 02:56:33', '2018-06-08 16:32:44', NULL, 3),
('MARMONIQHMAYBOZY', '2019-06-07 04:51:57', '2019-06-07 10:54:05', NULL, 41),
('MARMONIQHMAYBOZY', '2019-03-04 06:08:03', '2019-03-05 00:37:52', NULL, 18),
('MARMONIQHMAYBOZY', '2019-03-19 15:53:39', '2019-03-19 19:09:57', NULL, 7),
('MARMONIQHMAYBOZY', '2019-03-11 10:18:03', '2019-03-11 11:59:50', NULL, 2),
('MARMONIQHMAYBOZY', '2019-03-16 02:48:52', '2019-03-16 19:45:08', NULL, 26),
('MARMONIQHMAYBOZY', '2018-11-14 13:26:55', '2018-11-15 10:27:38', NULL, 26),
('MARMONIQHMAYBOZY', '2018-12-04 18:40:05', '2018-12-05 00:24:03', NULL, 7),
('MARMONIQHMAYBOZY', '2019-04-16 16:29:33', '2019-04-17 06:29:00', NULL, 22),
('MARMONIQHMAYBOZY', '2019-03-06 18:45:18', '2019-03-07 10:37:02', NULL, 35),
('MARMONIQHMAYBOZY', '2019-03-11 21:43:32', '2019-03-12 17:37:38', NULL, 33),
('MARMONIQHMAYBOZY', '2018-11-27 15:44:06', '2018-11-28 08:06:09', NULL, 16),
('MARMONIQHMAYBOZY', '2018-09-11 12:38:30', '2018-09-12 09:03:11', NULL, 18),
('MARMONIQHMAYBOZY', '2018-12-01 13:23:31', '2018-12-02 07:09:31', NULL, 3),
('MARMONIQHMAYBOZY', '2019-02-24 04:30:54', '2019-02-24 21:56:46', NULL, 22),
('MARMONIQHMAYBOZY', '2019-01-20 18:21:59', '2019-01-20 19:46:08', NULL, 24),
('MARMONIQHMAYBOZY', '2019-05-19 17:12:08', '2019-05-19 18:04:49', NULL, 19),
('MARMONIQHMAYBOZY', '2018-07-17 11:29:03', '2018-07-17 17:24:39', NULL, 18),
('MARMONIQHMAYBOZY', '2019-05-20 13:29:50', '2019-05-21 07:21:51', NULL, 2),
('MARMONIQHMAYBOZY', '2018-08-19 00:58:21', '2018-08-19 13:26:45', NULL, 2),
('MARMONIQHMAYBOZY', '2019-06-04 05:38:52', '2019-06-05 01:50:02', NULL, 33),
('MARMONIQHMAYBOZY', '2019-05-17 20:51:28', '2019-05-18 14:32:40', NULL, 40),
('MARMONIQHMAYBOZY', '2019-05-12 02:35:26', '2019-05-12 04:04:40', NULL, 40),
('MARMONIQHMAYBOZY', '2018-08-12 14:36:59', '2018-08-13 00:01:20', NULL, 16),
('MARMONIQHMAYBOZY', '2019-03-24 01:43:17', '2019-03-24 19:57:10', NULL, 35),
('MARMONIQHMAYBOZY', '2019-03-06 06:41:32', '2019-03-06 08:18:03', NULL, 16),
('MARMONIQHMAYBOZY', '2018-07-08 16:43:37', '2018-07-09 08:22:29', NULL, 4),
('MARMONIQHMAYBOZY', '2018-09-27 10:31:40', '2018-09-27 18:23:39', NULL, 9),
('MARMONIQHMAYBOZY', '2018-10-07 04:36:26', '2018-10-07 08:54:39', NULL, 24),
('MARMONIQHMAYBOZY', '2018-11-29 08:30:13', '2018-11-29 21:01:21', NULL, 33),
('MARMONIQHMAYBOZY', '2019-05-28 04:11:01', '2019-05-29 00:35:28', NULL, 22),
('MARMONIQHMAYBOZY', '2018-11-12 02:11:53', '2018-11-13 01:54:53', NULL, 9),
('MARMONIQHMAYBOZY', '2019-02-20 12:10:12', '2019-02-20 19:59:11', NULL, 33),

```



```
( 'MARMONIQHMAYBOZY', '2018-08-11 01:03:51', '2018-08-11 20:20:22', NULL, 7),
( 'MARMONIQHMAYBOZY', '2019-01-25 06:26:08', '2019-01-26 03:51:56', NULL, 3),
( 'MARMONIQHMAYBOZY', '2019-05-05 16:02:50', '2019-05-06 12:17:50', NULL, 16),
( 'MARMONIQHMAYBOZY', '2019-01-18 22:16:13', '2019-01-19 14:29:55', NULL, 24),
( 'MARMONIQHMAYBOZY', '2018-09-05 11:29:09', '2018-09-05 22:07:51', NULL, 3),
( 'MARMONIQHMAYBOZY', '2019-05-11 11:23:17', '2019-05-11 23:30:15', NULL, 22),
( 'MARMONIQHMAYBOZY', '2019-04-20 12:27:39', '2019-04-21 00:18:38', NULL, 35),
( 'MARMONIQHMAYBOZY', '2019-04-10 04:55:31', '2019-04-10 19:09:31', NULL, 2),
( 'MARMONIQHMAYBOZY', '2018-11-08 13:59:53', '2018-11-09 04:37:31', NULL, 19),
( 'MARMONIQHMAYBOZY', '2019-01-08 05:07:29', '2019-01-09 04:27:00', NULL, 9),
( 'MARMONIQHMAYBOZY', '2018-12-27 22:21:56', '2018-12-28 10:28:13', NULL, 22),
( 'MARMONIQHMAYBOZY', '2018-11-21 00:00:08', '2018-11-21 20:43:52', NULL, 26),
( 'MARMONIQHMAYBOZY', '2018-11-05 13:11:32', '2018-11-05 19:37:33', NULL, 7),
( 'MARMONIQHMAYBOZY', '2019-04-09 00:35:40', '2019-04-09 06:01:18', NULL, 7),
( 'MARMONIQHMAYBOZY', '2019-03-26 05:56:58', '2019-03-26 14:15:12', NULL, 18),
( 'MARMONIQHMAYBOZY', '2019-05-04 06:51:50', '2019-05-04 18:49:11', NULL, 6),
( 'MARMONIQHMAYBOZY', '2019-06-08 17:58:13', '2019-06-08 22:24:07', NULL, 41),
( 'MARMONIQHMAYBOZY', '2018-07-26 04:51:59', '2018-07-27 04:31:34', NULL, 19),
( 'MARMONIQHMAYBOZY', '2019-02-14 16:39:33', '2019-02-15 13:59:15', NULL, 35),
( 'MARMONIQHMAYBOZY', '2018-04-02 01:04:11', '2018-04-02 14:24:39', NULL, 2),
( 'MARMONIQHMAYBOZY', '2019-01-05 19:10:27', '2019-01-06 09:06:30', NULL, 6),
( 'MARMONIQHMAYBOZY', '2018-11-28 09:02:42', '2018-11-28 20:52:55', NULL, 24),
( 'MARMONIQHMAYBOZY', '2019-03-25 07:10:55', '2019-03-25 11:18:49', NULL, 26),
( 'MARMONIQHMAYBOZY', '2018-12-14 18:51:14', '2018-12-15 11:03:58', NULL, 19),
( 'MARMONIQHMAYBOZY', '2019-05-26 13:16:53', '2019-05-27 08:54:32', NULL, 26),
( 'MARMONIQHMAYBOZY', '2018-12-09 02:41:31', '2018-12-09 09:53:07', NULL, 25),
( 'MARMONIQHMAYBOZY', '2018-04-13 08:17:01', '2018-04-13 21:22:10', NULL, 6),
( 'MARMONIQHMAYBOZY', '2018-09-28 16:27:07', '2018-09-29 06:42:57', NULL, 16),
( 'MARMONIQHMAYBOZY', '2018-07-14 13:49:40', '2018-07-14 17:32:06', NULL, 4),
( 'MARMONIQHMAYBOZY', '2018-10-16 02:34:19', '2018-10-16 09:19:17', NULL, 18),
( 'MARMONIQHMAYBOZY', '2018-11-09 12:32:07', '2018-11-10 03:25:38', NULL, 20),
( 'MARMONIQHMAYBOZY', '2019-03-31 17:07:03', '2019-04-01 05:33:54', NULL, 18),
( 'MARMONIQHMAYBOZY', '2019-01-15 07:48:49', '2019-01-16 05:06:10', NULL, 33),
( 'MARMONIQHMAYBOZY', '2019-04-12 19:57:32', '2019-04-13 08:49:37', NULL, 6);
```

I valori di inserimento sono stati generati automaticamente grazie ad un software, appositamente creato, scritto in Python 3.7: **InsertSQL.py**. Per la versione originale multiplatforma, e il codice sorgente dello script (compresa la versione solo per Windows: **InsertSQL.exe**, che non necessita dell'interprete Python) si può trovare tutto il necessario nei file allegati a questa relazione. Per l'esecuzione da terminale utilizzare i seguenti comandi dopo essersi recati nella directory contenente il file:

```
python3 InsertSQL.py
```

4.3 Operazioni di interrogazione

Sono qui proposte alcune possibili query utilizzabili nel database

1. Selezionare tutte le persone che hanno ordinato tutti i software acquistati anche da una determinata persona.

Questa interrogazione non è possibile eseguirla direttamente perché consiste in un'operazione di *divisione*, pertanto dev'essere riscritta come: "Selezionare tutte le persone per le quali non esiste alcun software che una determinata persona abbia acquistato, e che loro stessi non abbiano comprato".

Questo genere di interrogazioni produce query annidate, con tre livelli, separate dalla clausola *NOT EXISTS*.

```
SELECT CF
FROM PERSONA P
WHERE CF <> 'VITBOTVQ5T3BB004'
AND NOT EXISTS(SELECT *
                FROM ORDINE O, DI_S D
                WHERE O.ID = D.ORDINE
                AND O.CLIENTE = 'VITBOTVQ5T3BB004'
                AND NOT EXISTS(SELECT *
                                FROM ORDINE O2, DI_S D2
                                WHERE O2.ID = D2.ORDINE
                                AND O2.CLIENTE = P.CF
                                AND D.SOFTWARE = D2.SOFTWARE
                                )
                );
```

2. Mostrare tutti i ticket in cui, nei relativi interventi, sono presenti tutti i dipendenti della divisione di appartenenza del ticket.

```
SELECT *
FROM TICKET T1
WHERE (SELECT COUNT(DISTINCT I2.CF)
       FROM IN_CARICO I2
       WHERE I2.ID = T1.ID) = (SELECT COUNT(*)
                                FROM DIPENDENTE D3
                                WHERE D3.DIVISIONE = T1.DIVISIONE)
```

3. Selezionare il dipendente addetto alla vendita che ha registrato il guadagno mensile maggiore tra tutti i dipendenti.

La funzione *to_char* permette di utilizzare solo il formato della data interessato, nel caso specifico la coppia anno-mese

```
SELECT SUM(O.PREZZO_TOTALE) AS GUADAGNO_MENSILE, P.NOME,  
       P.COGNOME, P.CF, to_char(O.DATA, 'YYYY-MM') AS PERIODO  
FROM DIPENDENTE D, ORDINE O, PERSONA P  
WHERE O.DIPENDENTE = D.CF  
      AND D.CF = P.CF  
GROUP BY 4, 5  
ORDER BY 1 DESC  
LIMIT 1;
```

4. Prodotti in vendita e relative quantità presenti in magazzino in ordine crescente di disponibilità.

```
SELECT MARCA, MODELLO, QUANTITA  
FROM M_VENDITA  
ORDER BY 3,1,2
```

5. Ticket da chiudere ordinati per priorità e data.

```
SELECT *  
FROM TICKET  
WHERE STATO NOT IN ('Completed', 'Rejected')  
ORDER BY PRIORITA, DATA
```

6. Numero di ordini per ogni dipendente addetto alla vendita, in un determinato periodo (in questo esempio tra il 1 Gennaio 2019 e il 31 Gennaio 2019).

```
SELECT D.CF, P.NOME, P.COGNOME, COUNT(*) AS NUMERO_ORDINI  
FROM DIPENDENTE D, ORDINE O, PERSONA P  
WHERE D.CF = O.DIPENDENTE  
      AND D.CF = P.CF  
      AND O.DATA BETWEEN '2019-1-1' AND '2019-1-31'  
GROUP BY 1, 2, 3  
ORDER BY 4 DESC
```

5 STUDIO DEGLI INDICI

Si è deciso di non creare indici aggiuntivi per il seguente database. Tale scelta è stata valutata considerando diversi aspetti.

Generalizzando è possibile notare che le tabelle possono essere suddivise in due macro-categorie:

1. tabelle con molti dati e molte operazioni di inserimento e modifica;
2. tabelle con operazioni quasi esclusivamente di visita e con pochi elementi.

Nel primo caso la dimensione della tabella potrebbe essere un ottimo punto di partenza per la creazione di un buon indice che possa comportare un miglioramento durante le query. Tuttavia, le numerose operazioni, di inserimento o modifica, peggiorerebbero notevolmente le prestazioni del database a causa del frequente aggiornamento dell'indice stesso.

Nel secondo caso invece la scarsa frequenza di operazioni “dannose”, per il mantenimento dell'indice, è un vantaggio per una sua eventuale implementazione. Ma potrebbe essere inutile, e nel peggiore dei casi anche negativo, a causa dello scarso popolamento di tali tabelle, per le quali una ricerca sequenziale potrebbe risultare addirittura migliore.

6 INTERFACCIA GRAFICA

L'elaborato dispone di un'interfaccia grafica dimostrativa in JAVA che simula alcuni possibili casi d'uso:

1. Visualizzazione dei Ticket presenti con possibilità di filtrare per:
 - a. Stato
 - b. Divisione
 - c. Data
 - d. Priorità
2. Modifica ed Eliminazione di un Ticket già presente
3. Creazione di un nuovo Ticket
4. Un Query Tool integrato con la possibilità di utilizzare anche query preinserite.

6.1 Visualizzazione

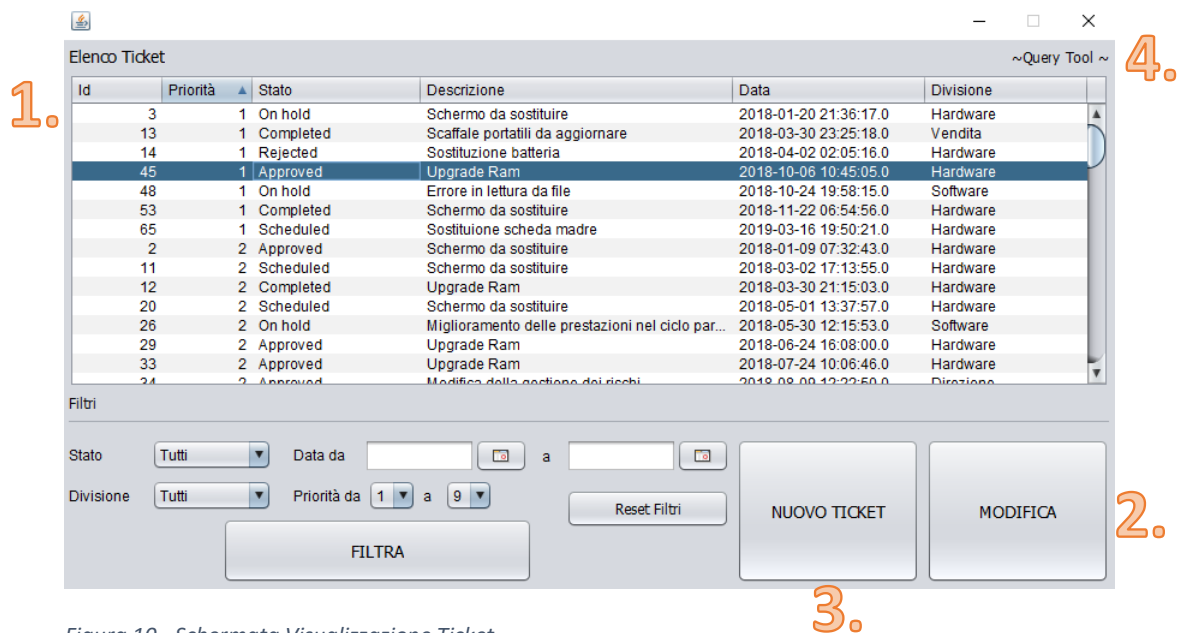


Figura 10 - Schermata Visualizzazione Ticket

La schermata di visualizzazione dei Ticket (punto 1) permette l'ordinamento in base alle colonne disponibili (ID, Priorità, Stato, Descrizione, Data e Divisione tramite click sulla colonna interessata) e presenta la possibilità di filtrare per Stato, Divisione, un intervallo di Date e/o un intervallo di Priorità.

6.2 Modifica ed Eliminazione

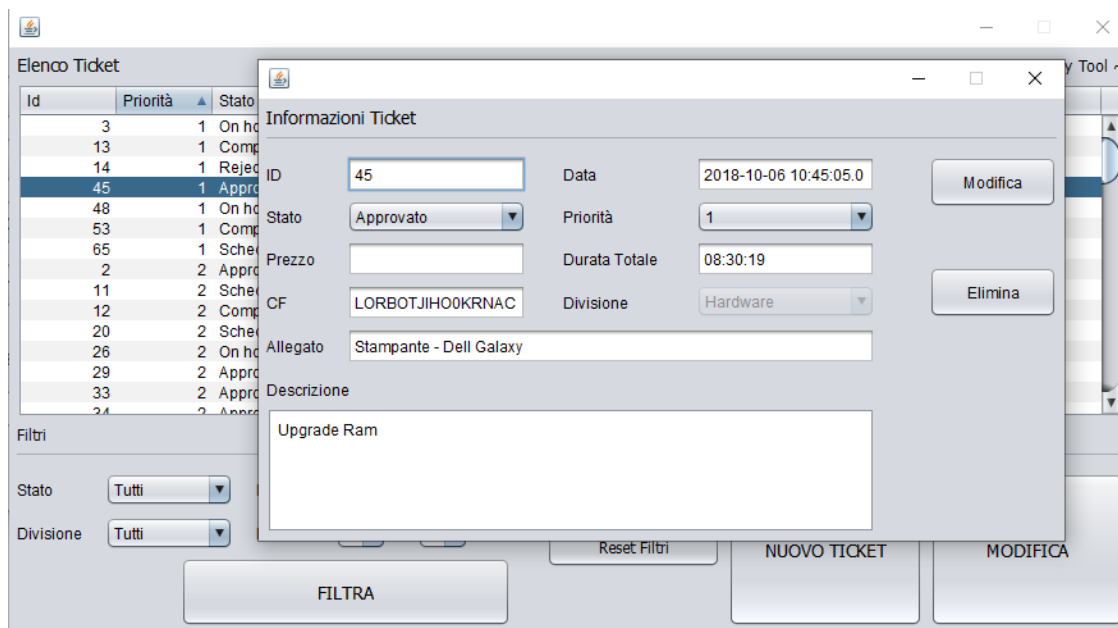


Figura 11 - Schermata Modifica ed Eliminazione Ticket

Selezionando un Ticket fra i disponibili verrà resa disponibile la possibilità di modifica (punto 2) attraverso il tasto apposito. Verrà aperta una schermata con maggiori informazioni riguardanti il suddetto Ticket dove sarà possibile modificare unicamente i campi variabili (Stato, Priorità, Prezzo e Descrizione). Nella stessa schermata viene data la possibilità di eliminare il Ticket aperto.

6.3 Creazione

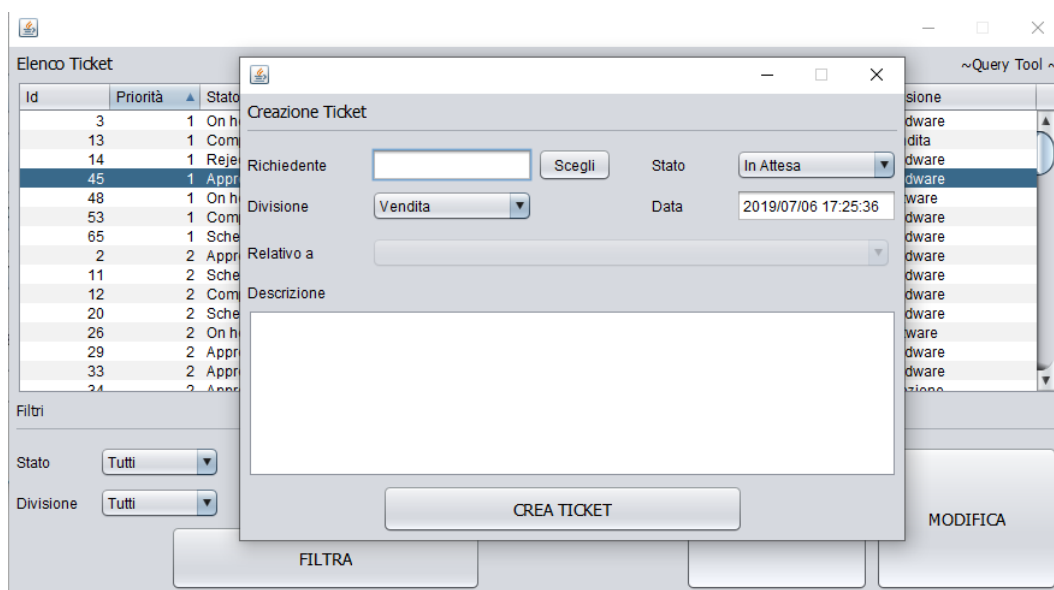
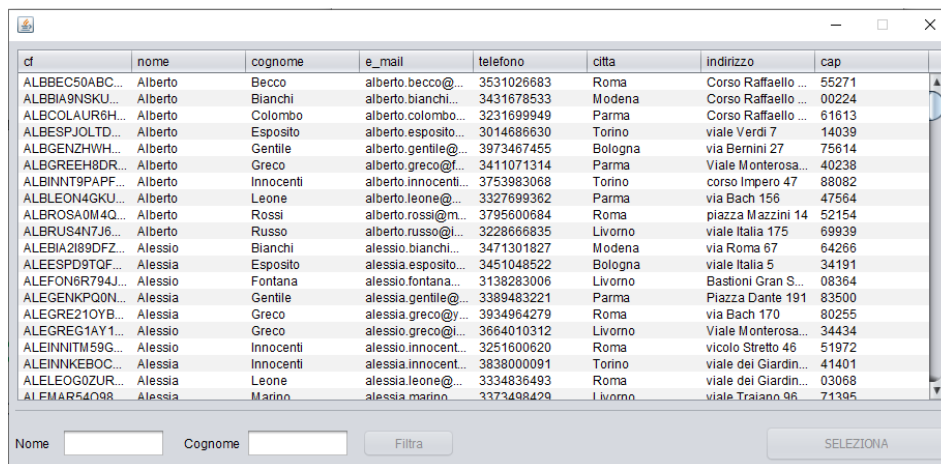


Figura 12 - Schermata Creazione Ticket

Tramite l'apposito bottone sarà possibile creare un nuovo Ticket (punto 3) tramite un'interfaccia apposita. Per la creazione sarà necessario inserire un Richiedente, uno Stato, la Divisione di appartenenza ed una Descrizione.



cf	nome	cognome	e_mail	telefono	città	indirizzo	cap
ALBBEC50ABC...	Alberto	Becco	alberto.becco@...	3531026683	Roma	Corso Raffaello ...	55271
ALBBIA9NSKU...	Alberto	Bianchi	alberto.bianchi...	3431678533	Modena	Corso Raffaello ...	00224
ALBCOLAUR6H...	Alberto	Colombo	alberto.colombo...	3231699949	Parma	Corso Raffaello ...	61613
ALBESPJOLTD...	Alberto	Esposito	alberto.esposito...	3014686630	Torino	viale Verdi 7	14039
ALBGENZHWH...	Alberto	Gentile	alberto.gentile@...	3973467455	Bologna	via Bernini 27	75614
ALBGREH8DI...	Alberto	Greco	alberto.greco@f...	3411071314	Parma	Viale Monterosa...	40238
ALBININT9PAP...	Alberto	Innocenti	alberto.innocenti...	3753983068	Torino	corso Impero 47	88082
ALBLEON4GKU...	Alberto	Leone	alberto.leone@...	3327699362	Parma	via Bach 156	47564
ALBROSA0M4Q...	Alberto	Rossi	alberto.rossi@...	3795600684	Roma	piazza Mazzini 14	52154
ALBRUS4N7J6...	Alberto	Russo	alberto.russo@...	3228666835	Livorno	viale Italia 175	69939
ALEBIA2I89DFZ...	Alessio	Bianchi	alessio.bianchi...	3471301827	Modena	via Roma 67	64266
ALEESP09TQF...	Alessio	Esposito	alessio.esposito...	3451048522	Bologna	viale Italia 5	34191
ALEFON6R794J...	Alessio	Fontana	alessio.fontana...	3138283006	Livorno	Bastioni Gran S...	08364
ALEGENKP00N...	Alessia	Gentile	alessia.gentile@...	3389483221	Parma	Piazza Dante 191	83500
ALEGRE21OYB...	Alessia	Greco	alessia.greco@y...	3934964279	Roma	via Bach 170	80255
ALEGREG1AY1...	Alessio	Greco	alessio.greco@...	3664010312	Livorno	Viale Monterosa...	34434
ALEINNTM59G...	Alessio	Innocenti	alessio.innocent...	3251600620	Roma	vicolo Stretto 46	51972
ALEINNKBOC...	Alessia	Innocenti	alessia.innocent...	3838000091	Torino	viale dei Giardin...	41401
ALELEOG0ZUR...	Alessia	Leone	alessia.leone@...	3334836493	Roma	viale dei Giardin...	03068
ALEMAR54O9R...	Alessia	Marino	alessia.marino...	3373498429	Livorno	viale Traiano 96	71395

Figura 13 - Schermata selezione Richiedente in Creazione

L'inserimento del Richiedente è gestito tramite un'apposita tabella (in cui poter filtrare per Nome e/o Cognome) per evitare problemi durante l'INSERT (nel caso in cui la persona scelta non esista nel database, o dati inseriti non corretti).

6.4 Query Tool

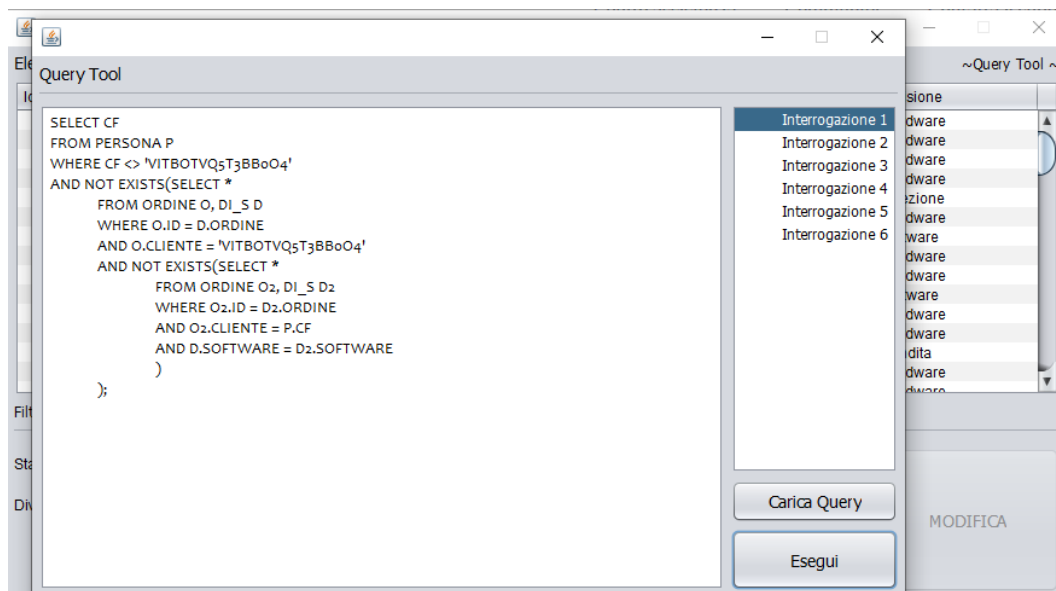


Figura 14 - Schermata Query Tool

Attraverso la scritta nell'angolo in alto a destra è possibile aprire un Query Tool integrato (punto 4) in cui è possibile provare liberamente delle Query sul database.

In aggiunta, viene resa disponibile una lista di Query preimpostate (e se necessario modificabili al momento) per verificare casi d'uso più avanzati.