Università di Pisa

Facoltà di Ingegneria

Corso di laurea in
Ingegneria Informatica

Documentazione Progetto per il corso di Basi di dati Anno Accademico 2015-2016

Prof. Gigliola Vaglini, Ing. Francesco Pistolesi

Studenti: Ludovica Cocchella, Matteo Castrignano

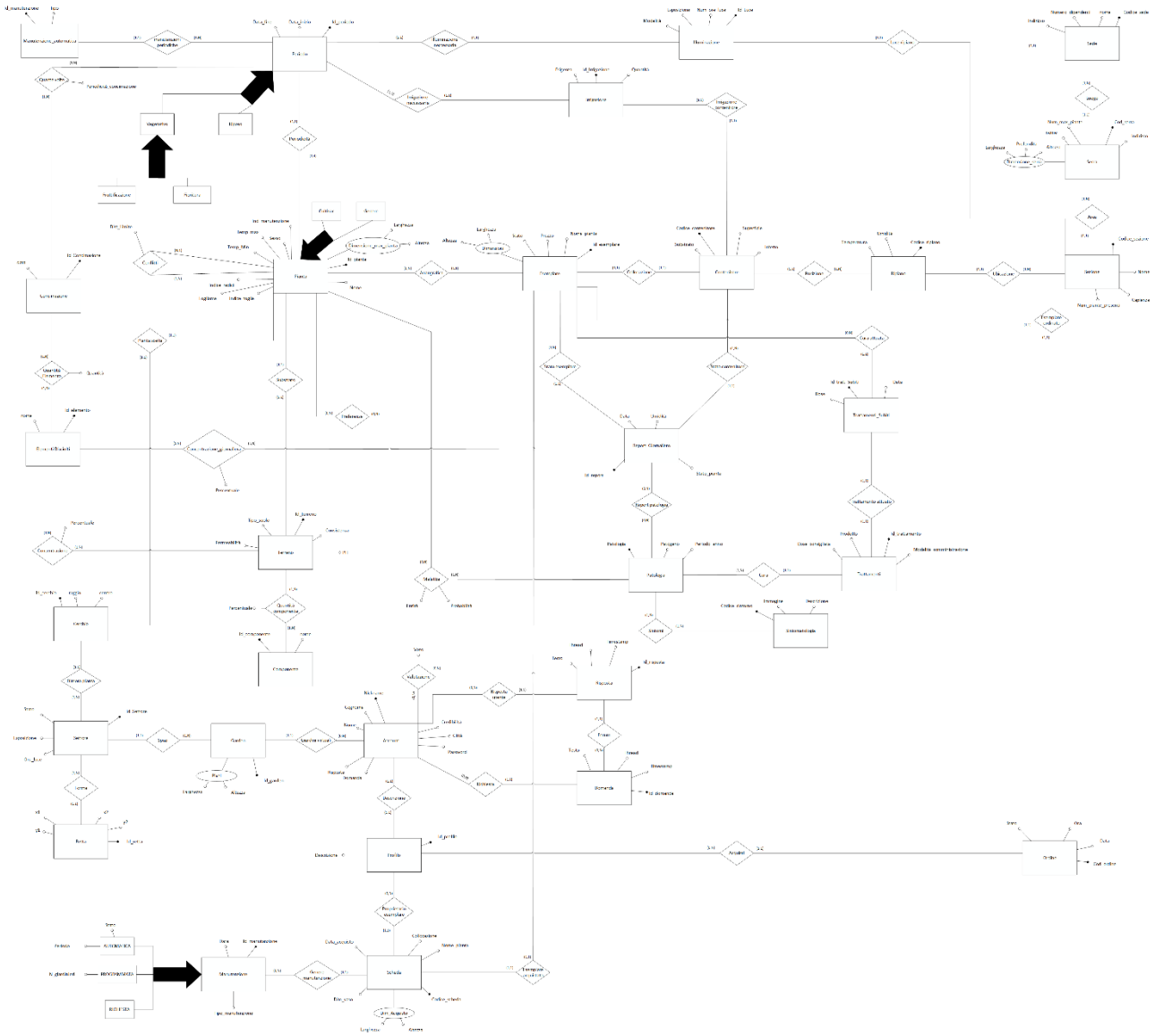
N. Matricola: 532189, 532151

Progettazione concettuale

Nel passaggio da un livello di raffinamento a un altro, lo schema viene modificato adottando alcune trasformazioni elementari che vengono denominate primitive di trasformazione top-down. Il vantaggio che abbiamo potuto trarre dalla strategia top-down è quella di descrivere inizialmente tutte le specifiche dei dati senza preoccuparsi dei dettagli, per poi entrare ad approfondire meglio ogni concetto ed a estrapolare tutti i dettagli necessari per descriverlo al meglio.

Attuando questa tecnica siamo riusciti ad ottenere uno schema concettuale corrispondente ai requisiti dati.

Schema concettuale



Di seguito viene riportata la documentazione riguardo alle entità e relazione che sono state introdotte.

Per ogni entità viene data una breve descrizione, vengono indicati i relativi attributi e ne viene indicata la chiave primaria.

Per quanto riguarda le relazioni, viene data per ogni relazione una descrizione con le relative entità coinvolte e vengono inoltre indicati i relativi attributi.

DOCUMENTAZIONE DELLE ENTITA'

Entità	Descrizione	Attributi	Identificatore
Pianta	Tutte le possibili caratteristiche di una pianta	Id_pianta,nome, Altezza_max, Larghezza_max, Fogliame,Genere,Indice_radice,Indice_foglie, sesso, Cultivar,Costo Ind_manutenzione, Temp_max, Temp_min;	Id_pianta
Esemplare	Identificazione della singola pianta presente all'interno del database	Id_esemplare,Altezza, Larghezza,Nome_pianta, Prezzo	Id_esemplare
Sede	Dislocazione dell'azienda	Codice_Sede, nome, indirizzo,Città Numero_Dipendenti	Codice_Sede
Serra	Compongono le sedi ed è l'area dove vengono coltivate le piante	Cod_serra, indirizzo, Num_max_piante,nome, altezza, lunghezza, larghezza	Cod_serra
Sezione	Area della Serra costituita da ripiani e contenitori	Codice_Sezione, Nome, Num_Piante_Presenti Capienza	Codice_Sezione
Ripiano	Suddivisione delle piante all'interno della sezione	Codice_Ripiano, Umidità, Temperatura	Codice_Ripiano
Contenitore	Collocazione di una pianta ospitata nella serra	Codice_Contenitore, Superficie, infetto	Codice_Contenitore
Illuminazione	Individua i vari tipi di illuminazione che una pianta può richiedere	Id_Luce, Esposizione, Modalità, Num_ore_Luce	Id_Luce

Irrigazione	Fabbisogno idrico di una pianta	Id_Irrigazione, Quantità, Esigenza	Id_Irrigazione
Periodo	Suddivisione dell'anno in determinate periodo e ognuno caratterizzato da determinate caratteristiche che contraddistinguono la piante	Id_Periodo, Mese_inizio, Mese_fine, nome	Id_Periodo
Concimazione	Le Piante necessitano di interventi durante l'anno che variano in base al periodo dell'anno	Id_Concimazione, Data	Id_Concimazione
Terreno	Area dove vengono collocate le piante	Id_Terreno, Tipo_Suolo, PH, Permeabilità, Consistenza	Id_Terreno
Elementi Disciolti	Elementi contenuti nel terreno	Id_elemento, nome	Id_elemento
Componente	Elementi che costituiscono il substrato	Id_Componente, nome	Id_Componente
Patologia	Tutte le possibili malattie della pianta	Id_Patologia, patogeno,	Patologie
Sintomatologia	Indica i sintomi della pianta causati dalla patologia	Codice_sintomo, Descrizione, Immagine	Codice_sintomo
Trattamenti	Possibile Modalità di debellazione del sintomo	Id_trattamento, Modalità_somministrazione, prodotto, Dose_consigliata	Id_trattamento
Trattamenti_Subiti	Tipo di trattamento attuato per debellare il sintomo	Id_tratt_Subiti, Dose, Data	Id_tratt_Subiti
Report_Giornaliero	Raccolta informazioni riguardanti lo stato giornaliero di ogni esemplare e le condizioni del terreno in cui si trovano	Id_report, Data, Umidità, Stato_pianta	Id_report

Profilo	Raccolta preferenze del cliente e delle piante che ha acquistato dall'azienda	Id_profilo, Descrizione	Id_profilo
Scheda	Raccolta delle informazioni riguardanti la pianta acquistata dal cliente	Codice_scheda,Collocazione , Dim_vaso, Data_acquisto, Altezza_acquisto, Larghezza_acquisto	Codice_scheda
Risposta	Risposta relativa a una domanda posta da un altro cliente nel forum	Id_risposta, Timestamp, Thread, Testo	Id_risposta
Domanda	Inserimento da parte di un cliente di una domanda, riguardante informazioni relative alle piante, all'interno del forum	Id_domanda, Timestamp, Thread, Testo	Id_domanda
Ordine	Raccolta informazioni riguardante l'ordine effettuato dal cliente di una nuova pianta	Cod_ordine, Data, Ora, Stato	Cod_ordine
Garden	Giardino virtuale del cliente	Id_garden, pixel_altezza, pixel_larghezza	Id_garden
Settore	Suddivisione dello spazio del giardino e informazioni relative alla pavimentazione del settore	Id_settore, Stato, Esposizione, ore_luce	Id_settore
Cerchio	Forma geometrica utilizzata per realizzare piante, vasi ecc all'interno del giardino virtuale	Id_cerchio, centro, raggio	Id_cerchio
Retta	Ente geometrico utilizzato per realizzare le sezioni nel giardino virtuale	Id_retta, x1,x2,y1,y2	Id_retta
Manutenzione_Automatica	Suggerimento del sistema di interventi di manutenzione relative alle esigenze della pianta	Id_manutenzione_automatica, Tipo_manutenzione, Data,Stato,Periodo	Id_manutenzione

Manutenzione_Programmata		Data, Tipo_manutenzione, Id-man_prog, N_Giardinieri	Id_man_prog
Manutenzione_richiesta	Richiesta da parte del cliente di una manutenzione entro una determinata data	Id_richiesta, Data_Scadenza, Tipo_manutenzione	Id_richiesta
Intervento	Indica in base al periodo gli manutenzioni periodiche necessarie	Id_intervento, Tipo	Id_intervento
Account	Registrazione delle informazioni dell'utente	Nickname, Cognome, Nome, Risposta, Domanda, Credibilità, Città, Password	Nickname

DOCUMENTAZIONE DELLE RELAZIONI

Relazione	Descrizione	Entità coinvolte	Attributi
Anagrafica	Associa ad ogni esemplare le caratteristiche corrispondenti alla rispettiva pianta	Pianta (0,N), Esemplare (1,1)	
Conflitto	Indica in base agli elementi disciolti e alle caratteristiche dell'apparato radicale quale piante non possono essere messe a dimore in prossimità di un'altra pianta	Pianta (1,N), Pianta (N,1)	
Preferenze	Associa ad ogni cliente le piante che preferisce	Pianta (0.N), Account (0,N)	
Luogo	Associa ad ogni Serra la sede in cui si trova	Sede (1,N), Serra (1,1)	

Area	Individua in quale Serra si trova quella determinata sezione	Serra (1,N), Sezione (1,1)	
Ubicazione	Indica in quale sezione si trova il ripiano	Sezione(1,N), Ripiano (1,1)	
Posizione	Indica in quale ripiano si trova quel determinato Contenitore	Ripiano (1,N), Contenitore (1,1)	
Collocazione	Indica in quale Contenitore è ospitata la pianta	Esemplare (0,1), Contenitore (1,1)	
Cura_attuata	Associa ad ogni esemplare che presenta una patologia un determinato trattamento	Esemplare (0,N), Trattamenti_Subiti (1,1)	
trattamento_attuato	Fra tutti i possibili trattamenti indica quelli utilizzati	Trattamenti_Subiti (1,1), Trattamenti (1,1)	
Cura	Indica i possibili trattamenti per quella determinata patologia	Trattamenti (1,1), Patologia (1,N)	
Malattia	Associa ad ogni pianta le possibili patologie	Patologia (1,N), Pianta (0,N)	
Sintomi	Associa ad ogni patologia i possibili sintomi	Patologia1,N) , Sintomatologia (1,N)	
Report_patologia	Indica nel Report_Giornaliero lo stato della pianta	Report_Giornaliero (0,N) , Patologia (0,N)	
Irrigazione contenitore	Indica il fabbisogno idrico necessario per quel determinato contenitore	Irrigazione (1,1), Contenitore (1,1)	
Valutazione	Associa ad ogni cliente la valutazione assegnata alla suo rispota	Account (0,N) , Risposta (0,N)	Voto
Periodicità concimazione	Indica quante volte deve essere effettuata la concimazione in un determinato periodo	Periodo (1,N) , Concimazione (1,N)	Quante_volte

Quantità elemento	Indica la quantità dell'elemento utilizzato nella concimazione	Concimazione (1,N) , Elementi Disciolti (1,N)	Quantità
Concentrazione	Indica la concentrazione degli elementi disciolti nel terreno	Elementi Disciolti (1,N) , Terreno (1,N)	Percentuale
Quantità componente	Indica, in percentuale, la quantità della componente presente nel terreno	Terreno (1,N) , Componente (1,N)	Percentuale
Luce ripiano	Indica il tipo di illuminazione del ripiano	Illuminazione (1,1), Ripiano (1,1)	
Concentrazione_giornaliera	Indica la quantità giornaliera di elementi disciolti presenti nel terreno da inserire nel report	Elementi Disciolti (1,N), Report_Giornaliero (1,N)	Percentuale
Illuminazione necessaria	Associa al periodo il tipo di illuminazione necessaria	Periodo (1,1) , Illuminazione (1,1)	
Irrigazione necessaria	Associa al periodo la quantità di irrigazione necessaria	Periodo (1,1), Irrigazione (1,1)	
Irrigazione contenitore	Indica l'irrigazione necessaria per il contenitore	Irrigazione (1,1) , Contenitore (1,1)	
Periodicità	Associa le caratteristiche della pianta in quel determinato periodo	Periodo (1,1), Pianta (1,4)	
Manutenzioni Periodiche	Indica che manutenzione deve essere effettuata in quel periodo	Intervento (1,1), Periodo (1,N)	
Dimora pianta	Indica in ogni settore dove sono collocate le piante	Cerchio (1,1) , Settore (1,N)	

Forma	Elemento di costruzione del Settore	Retta (1,1) , Settore (1,N)	
Spazi	Indica la suddivisione del giardino in Settori	Settore (1,1), Garden (1,N)	
Giardini Virtuali	Associa ad un account il giardino virtuale dell'utente	Account (0,N), Garden (1,1)	
Descrizione	Associa ad ogni account un profilo contenente le preferenze dell'utente	Account (1,1), Profilo (1,1)	
Acquisti	Associa ad ogni profilo il rispettivo ordine	Profilo (0,N), Ordine (1,1)	
Esemplare acquistato	Associa una scheda ad una pianta acquistata	Scheda (1,1), Esemplare (1,1)	
Report Patologie	Registrazione giornaliera dello stato della patologia	Report_giornaliero (0,N), Patologia (0,N)	
Pianta Scelta	Associa alla forma cerchio la pianta a cui fa riferimento	Cerchio (1,1), Pianta (1,1)	
Proprietario esemplare	Associa ad un esemplare, presente in una scheda, il rispettivo proprietario	Profilo (0,N), Scheda (1,1)	
Richieste	Indica per quale esemplare il cliente richiede la manutenzione	Manutenzione_richiesta (1,1), Scheda (0,N)	
Manutenzione temporale	Indica quale esemplare è oggetto di una manutenzione nel corso del tempo	Manutenzione_programmata (1,1), Scheda (1,N)	
Stato Esemplare	Indica lo stato giornaliero dell'esemplare e lo memorizza nel Report	Esemplare (1,N), Report_giornaliero (1,1)	
Substrato	Superficie su cui vivono le piante	Pianta(1,1), Terreno(1,N)	

Tipo substrato	Tipologia di substrato presente in un contenitore	Terreno(0,N) Contenitore(1,1)	
Esemplare ordinato	Indica l'esemplare che è stato ordinato da un utente	Esemplare(1,1) Ordine(1,1)	
Riposta utente	Risposta di un utente ad una domanda effettuata da un altro utente	Account(0,N), Risposta(1,1)	
Richiesta	Indica un account che ha effettuato una domanda	Account(0,N), Domanda(1,1)	
Forum	Insieme di risposte relativa a una domanda	Risposta(1,1), Domanda(0,N)	
Automatico	Indica una manutenzione automatica all'interno della scheda	Manutenzione_automatica(1,1), Scheda(1,N)	
Stato contenitore	Indica lo stato giornaliero del contenitore e lo memorizza nel Report	Contenitore (1,N), Report_giornaliero (1,1)	

Regole di Vincolo

Qui di seguito vengono riportate le regole di vincolo

- RV1-La Dimensione della pianta deve essere sempre minore o uguale della dimensione massima raggiungibile.
- RV2-Per le piante infestanti la distanza minima che deve intercorrere fra le altre piante deve essere maggiore della distanza minima prevista nel caso in cui la pianta sia sana.
- RV3-Durante il periodo di riposo l'esigenza di irrigazione sarà sicuramente minore rispetto a quando essa si trova nel periodo vegetativo.
- RV4-Le potature vengono stabilite in base allo stato della pianta
- RV5-Il numero di piante ospitabile in ogni singola sezione deve essere minore del numero di piante ospitabili all'interno della serra.
- RV6-La dimensione del contenitore non deve essere minore della dimensione della pianta contenuta.

- RV7-Il numero di piante effettivamente presenti nella sezione non può essere maggiore del numero di piante ospitabile in quella rispettiva sezione.
- RV8-La dimensione del vaso non deve essere minore della dimensione della pianta al momento dell'acquisto.
- RV9-Una patologia può essere curata esclusivamente attuando un trattamento fra quelli consigliati per quella determinata patologia.
- RV10-La data di una risposta relativa ad una domanda deve essere maggiore della data della domanda stessa.
- RV11-ogni utente può avere solo un account
- RV12-La data di effettuazione della manutenzione automatica deve essere compresa fra la data di inizio e la data di fine del periodo

Ristrutturazione del diagramma E-R

I sistemi tradizionali per la gestione delle basi di dati non consentono di rappresentare direttamente una generalizzazione, è necessario dunque trasformare questo costrutto in altri costrutti del modello E-R per i quali esiste una implementazione naturali che non sono altro che le entità e le relazioni.

Le generalizzazioni presenti nel diagramma E-R che devono essere soggette a una trasformazione sono quelle relative al: periodo, la manutenzione e alla pianta.

Generalizzazione Periodo

Il metodo che abbiamo attuato per eliminare la generalizzazione periodo è l'accorpamento delle entità figlie della generalizzazione nell'entità genitore. Abbiamo appunto eliminato le entità 'Fruttificazione' e 'Fioritura' e le loro proprietà(attributi e partecipazioni ad associazioni e generalizzazioni) sono state aggiunte all'entità genitore 'Vegetativo'. Successivamente abbiamo attuato lo stesso metodo per l'entità 'Vegetativo' e 'Riposo' le cui proprietà sono state aggiunte all'entità 'Periodo'.

Generalizzazione Pianta

La tecnica che viene utilizzata per eliminare la generalizzazione Pianta è l'accorpamento dell'entità figlie della generalizzazione nell'entità padre. Si effettua l'eliminazione delle due entità 'Cultivar' e 'Genere' e le loro proprietà vengono aggiunte all'entità padre che è 'Pianta'. Infine vengono aggiunte all'entità padre gli attributi 'Cultivar' e 'Genere'.

Generalizzazione manutenzione

In questo caso il metodo attuato per eliminare la generalizzazione manutenzione è l'accorpamento del genitore della generalizzazione nelle figlie.
L'entità genitore 'Manutenzione' viene eliminata e tutte le sue proprietà vengono ereditate dall'entità figlie 'Richiesta', 'Programmata' e 'Automatica'

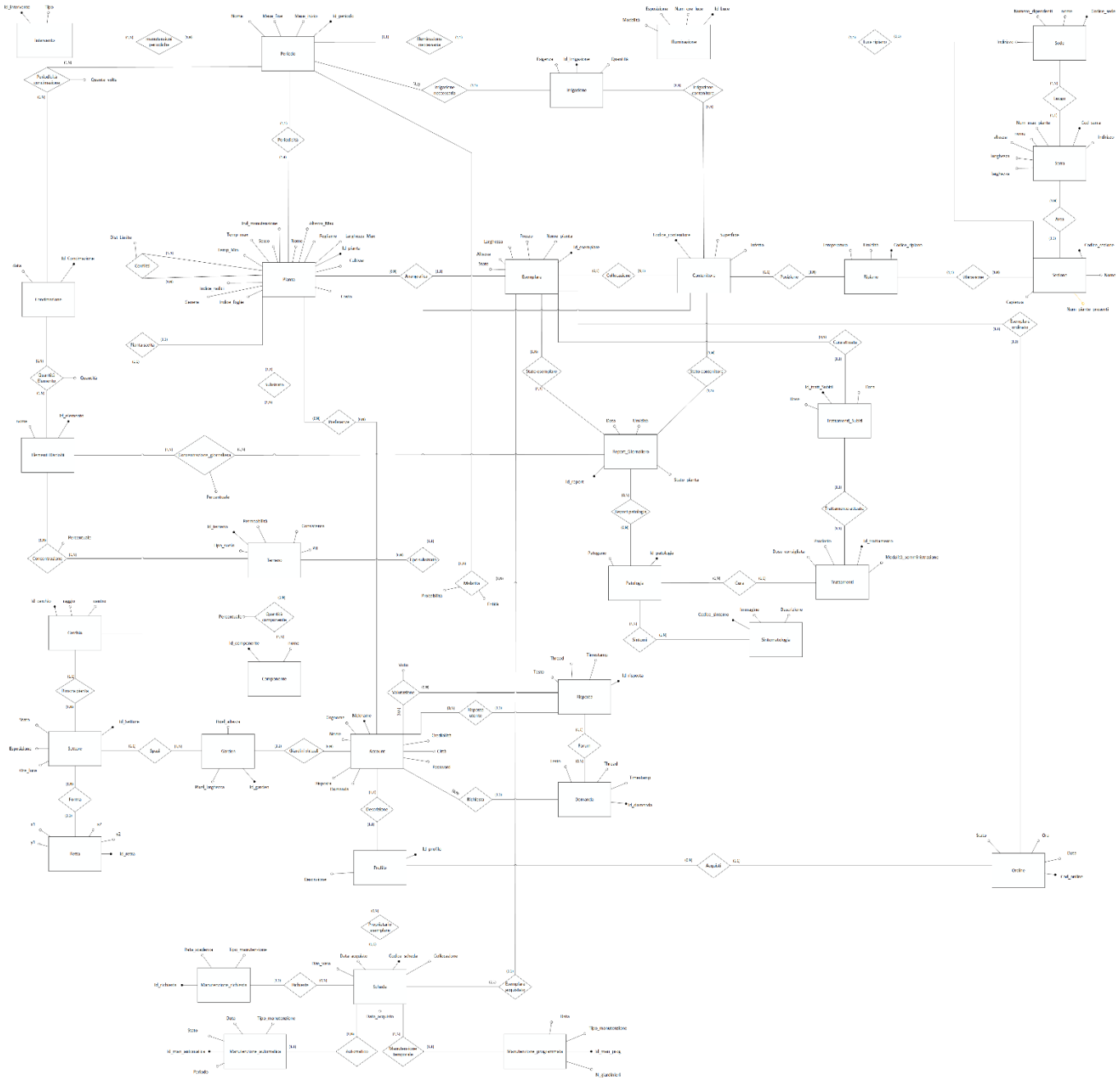
Eliminazione Attributi Multivalore

Come per le generalizzazioni, è opportuno effettuare l'eliminazione degli attributi multivalore poiché il modello relazione non permette di rappresentare in maniera diretta questo tipo di attributo.
Gli attributi multivalore presenti nel diagramma E-R sono: 'pixel', 'Dimensione_max_pianta', 'Dimensione_serra', 'Dim_acquisto'.

Per tutti questi attributi multivalore, viene attuato lo stesso metodo di eliminazione che consiste nel partizionamento dell'attributo multivalore in due attributi distinti.

Schema E-R ristrutturato

Come risultato dell'eliminazione delle generalizzazioni e degli attributi multivalore si ottiene il seguente schema E-R ristrutturato.



Individuazione di operazioni sui dati

Dopo avere completato la realizzazione del digramma E-R ed averne effettuato la ristrutturazione, abbiamo individuato le 8 operazione significative richieste.

Le 8 operazioni individuate sono:

Op1-Gestione dei conflitti

Op2- Calcolo del prezzo

Op3- Numero piante della serra

Op4- Calcolo del voto

Op5- Inserimento esemplare non esistente ma che è stato ordinato

Op6- Gestione manutenzione automatica

Op7- Gestione impiegati

Op8- Gestione Schede

Operazione 1

Descrizione

Questa operazione gestisce i conflitti fra le varie piante, in particolare all'inserimento di una pianta controlla gli eventuali elementi in comune con le altre piante e calcola la distanza minima fra le due piante tenendo in considerazione la loro larghezza massima.

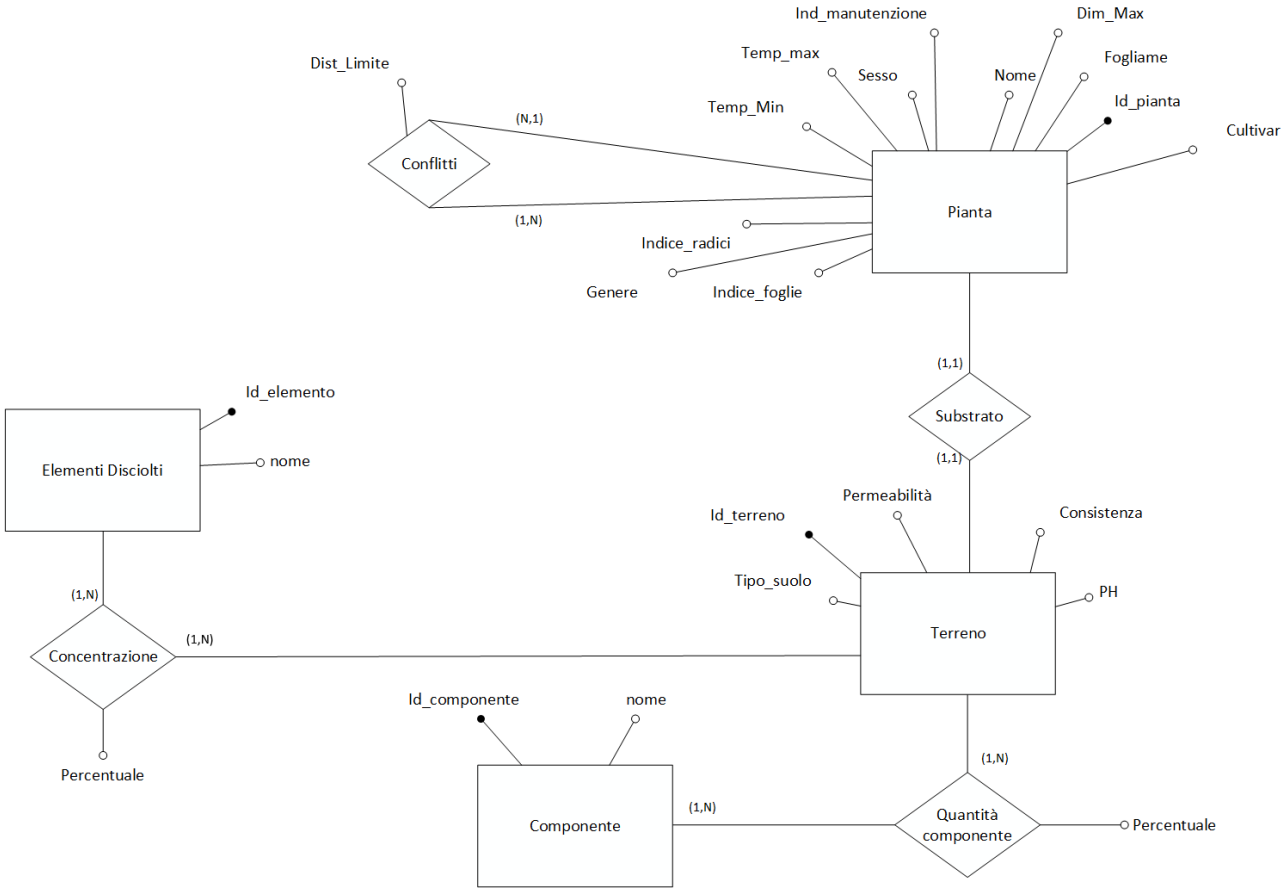


Tavola dei volumi: Operazione 1

Concetto	Tipo	Volume	Motivazione
Pianta	E	5000	Ipotesi iniziale
Conflitti	R	15000	Ogni pianta ha in media 3 conflitti
Elementi Disciolti	E	40	Ipotesi iniziale
Terreno	E	500	Ogni terreno ha in media 10 substrati
Componente	E	100	Ipotesi iniziale
Concentrazione	R	400	Ogni terreno ha in media 4 componenti
Substrato	R	5000	Cardinalità (1,1) con Pianta
Quantità componente	R	2000	Essendo che ogni terreno ha in media 4 componenti ed essendoci 500 istanze di terreno si avrà 4x500=2000

Tavola degli accessi per l'operazione 1

Concetto	Costrutto	Accessi	Tipo
Pianta	E	2	L
Conflitti	R	1	S
Terreno	E	2	L
Concentrazione	R	8	L

Per ogni pianta un terreno ha 4 concentrazioni

Implementazione operazione 1

```
DROP TRIGGER IF EXISTS Conflitti;

DELIMITER $$

CREATE TRIGGER Conflitti
AFTER INSERT ON Pianta FOR EACH ROW
BEGIN

DECLARE finito INTEGER DEFAULT 0;
DECLARE _larghezza int(11);
DECLARE _pianta int(11);
DECLARE _elem int(11);
DECLARE _perc int(11);
DECLARE _distanza int(11);

DECLARE conflitti CURSOR FOR

SELECT DISTINCT P.id_pianta, (P.Larghezza_Max + EP.Larghezza_max)/2
AS DistanzaMinima
FROM Pianta P NATURAL JOIN Terreno T NATURAL JOIN Concentrazione C
     CROSS JOIN
      (SELECT P.Id_pianta, P.Larghezza_max, C.Id_elemento,
C.Percentuale
      FROM Pianta P NATURAL JOIN Terreno T NATURAL JOIN
Concentrazione C
       WHERE P.Id_pianta = NEW.Id_pianta) AS EP
```

```
WHERE C.Id_elemento = EP.Id_elemento AND P.Id_Pianta !=
EP.Id_pianta;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET finito = 1;

OPEN conflitti;

scan: LOOP
     FETCH conflitti INTO _pianta, _distanza;
 IF(finito=1) THEN
        LEAVE scan;
      END IF;

    INSERT INTO Conflitti VALUES ( _pianta,NEW.id_pianta,
_distanza);

END LOOP;

CLOSE conflitti;

END$$
DELIMITER;
```

Operazione 2

Descrizione

All'inserimento di un nuovo esemplare viene calcolato il costo sulla base delle sue dimensioni.

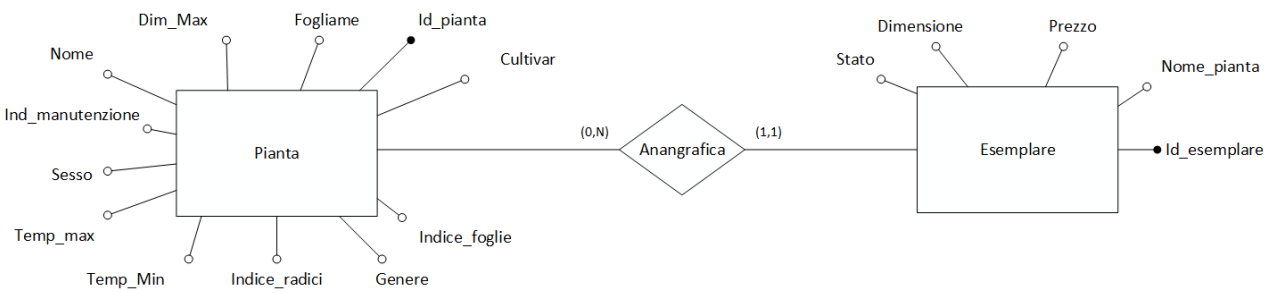


Tavola dei volumi: Operazione 2

Concetto	Tipo	Volume	Motivazione
Pianta	E	5000	Ipotesi iniziale
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari
Anagrafica	R	36000	Cardinalità (1,1) con Esemplare

Tavola degli accessi per l'operazione 2

Concetto	Costrutto	Accessi	Tipo
Pianta	E	1	L
Esemplare	E	1	S

Implementazione operazione 2

```
DROP TRIGGER IF EXISTS CalcoloCosto;

DELIMITER $$

CREATE TRIGGER CalcoloCosto
BEFORE INSERT ON Esemplare FOR EACH ROW
BEGIN

DECLARE CostoPianta int(11);

SELECT Costo INTO CostoPianta
FROM Pianta
WHERE Id_pianta = NEW.Id_pianta;
```

```
SET NEW.Prezzo = NEW.Altezza * NEW.Larghezza * CostoPianta;

END$$
DELIMITER;
```

Operazione 3

Descrizione

Dato il codice della Serra viene effettuato il conteggio degli esemplari al suo interno.

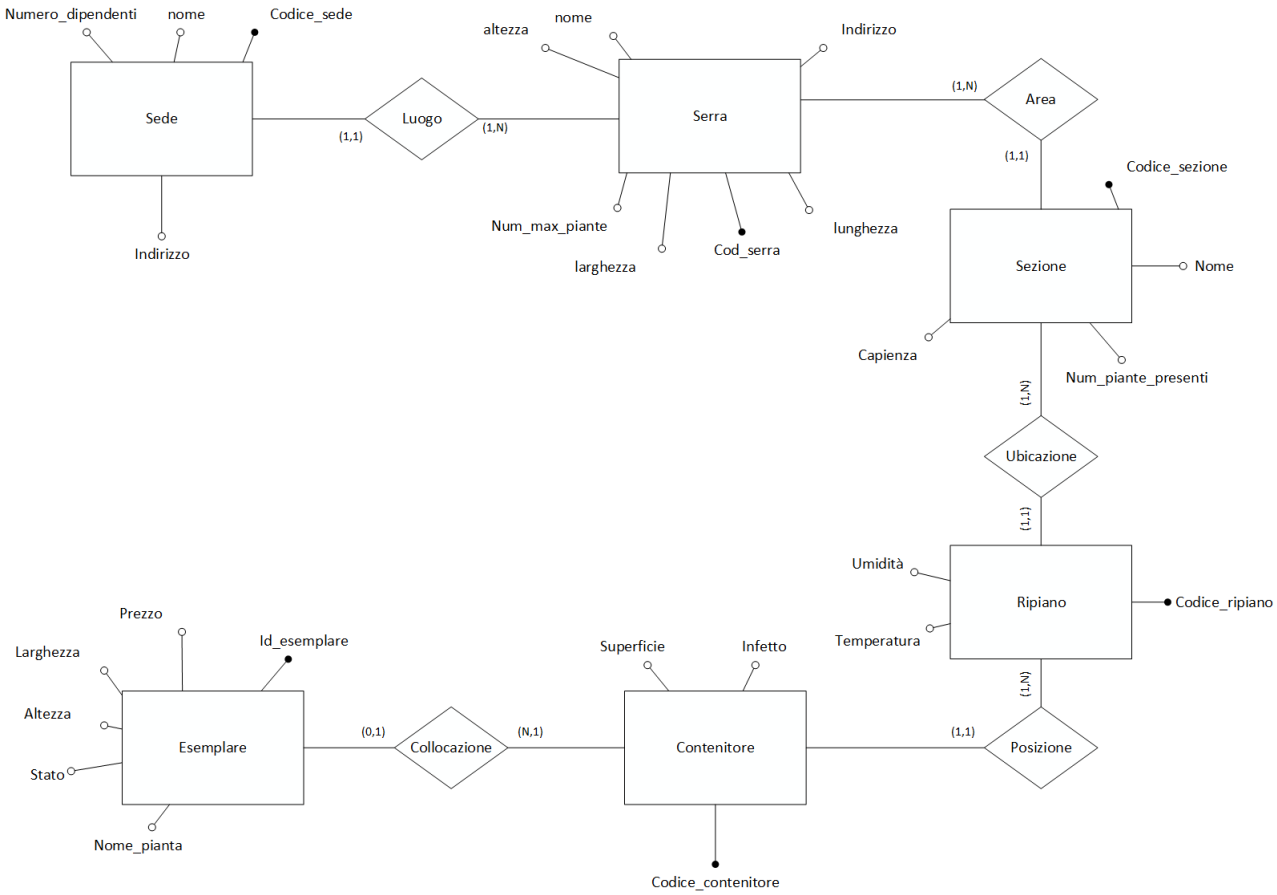


Tavola dei volumi: Operazione 3

Concetto	Tipo	Volume	Motivazione
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari
Collocazione	R	24000	2/3 degli Esemplare ha una collocazione nei vari contenitore
Contenitore	E	100000	In ogni ripiano ci sono in media 5 contenitori
Posizione	R	100000	Cardinalità (1,1) con Contenitore
Ripiano	E	20000	Ogni sezione ha in media 100 ripiani
Ubicazione	R	20000	Cardinalità (1,1) con Ripiano
Serra	E	20	Ipotesi iniziale
Sezione	E	200	In media ogni serra ha 10 sezioni
Area	R	200	Cardinalità (1,1) con Sezione

Luogo	R	5	Cardinalità (1,1) con Sede
Sede	E	5	Ogni sede ha in media 4 serra

Tavole degli accessi per l'operazione 3

Concetto	**Costrutto**	**Accessi**	**Tipo**
Serra	E	2	L
Contenitore	E	5000	L
Ripiano	E	1000	L
Sezione	E	10	L
Esemplare	E	5000	L

Implementazione operazione 3

```
DROP PROCEDURE IF EXISTS NumPiantePresenti;

DELIMITER $$

CREATE PROCEDURE NumPiantePresenti(IN _serra int(11), OUT
_NpiantePresenti int(11))
BEGIN

DECLARE Presenti INTEGER DEFAULT 0;

SELECT COUNT(*) INTO _NpiantePresenti
FROM Esemplare E INNER JOIN Contenitore C ON E.Codice_contenitore =
C.Codice_contenitore
     INNER JOIN Ripiano R ON C.Codice_ripiano = R.Codice_ripiano
     INNER JOIN Sezione SE ON R.Codice_sezione = SE.Codice_sezione
     INNER JOIN Serra SR ON SE.Codice_serra = SR.Codice_serra
WHERE SR.Codice_serra = _serra;

END$$
DELIMITER;
```

Introduzione Ridondanza operazione 3

Abbiamo introdotto l'attributo 'NPiantePresenti' alla tabella 'Serra' e sono stati due trigger per il mantenimento della ridondanza. Qui di seguito verrà riportata la tabella degli accessi e verrà confrontata con la precedente per vedere se è più vantaggioso mantenere la ridondanza oppure no.

Concetto	**Costrutto**	**Accessi**	**Tipo**
Serra	E	1	L
Serra	E	2	S
Contenitore	E	2	L
Ripiano	E	2	L
Sezione	E	2	L
Esemplare	E	1	L

Considerando che le operazioni di scrittura sono più onerose di quelle in lettura, in quanto devono essere eseguite in modo esclusivo e possono richiedere l'aggiornamento di indici, è più conveniente non introdurre la seguente ridondanza poiché in assenza della seguente non sono presenti operazioni di scrittura.

Operazione 4

Descrizione

Dopo l'inserimento di una valutazione viene aggiornata la credibilità dell'utente che ha risposto ad una domanda, effettuando una media matematica tra la somma dei voti e il numero di valutazioni ricevute.

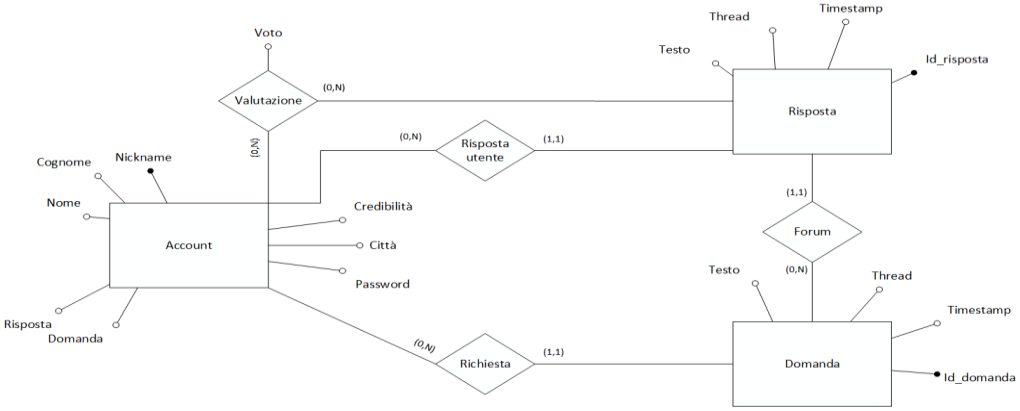


Tavola dei volumi: Operazione 4

Concetto	Tipo	Volume	Motivazione
Account	E	8000	Ipotesi iniziale
Risposta	E	80000	Ogni domanda ha in media 5 risposte
Domanda	E	16000	Un account fa in media 2 domane
Valutazione	R	240000	Ogni risposta ha in media 3 valutazioni
Risposta utente	R	80000	Cardinalità (1,1) con risposta
Richiesta	R	16000	Cardinalità (1,1) con domanda
Forum	R	80000	Cardinalità (1,1) con risposta

Tavola degli accessi per l'operazione 4

Concetto	Costrutto	Accessi	Tipo
Account	E	1	S
Risposta	E	31	L
Valutazione	R	31	L

ogni account risponde in media a 10 domande e ogni domanda ha in media 3 valutazioni= >10*3=30

Implementazione operazione 4

```
DROP TRIGGER IF EXISTS CalcoloVoto;

DELIMITER $$

CREATE TRIGGER CalcoloVoto
AFTER INSERT ON Valutazione FOR EACH ROW
BEGIN

SELECT DISTINCT R1.Nickname INTO @nome
FROM Valutazione V1 inner join Risposta R1  on
R1.Id_risposta=V1.Id_risposta
WHERE R1.Id_risposta=NEW.Id_risposta;

SELECT SUM(V.Voto), COUNT(*) INTO @sommaVoti, @NumRisposte
FROM Valutazione V inner join Risposta R  on
R.Id_risposta=V.Id_risposta
WHERE R.Nickname= @nome;

UPDATE Account
SET Credibilita= @sommaVoti / @NumRisposte
WHERE Nickname= @nome;

END$$
DELIMITER;
```

Introduzione Ridondanza

Concetto	Costrutto	Accessi	Tipo
Account	E	3	S
Risposta	E	1	L
Valutazione	R	1	L

Considerando che le operazioni di scrittura sono più onerose di quelle in lettura, in quanto devono essere eseguite in modo esclusivo e possono richiedere l'aggiornamento di indici, è più conveniente non introdurre la seguente ridondanza poiché in assenza della seguente si avrebbe un solo accesso in scrittura invece di tre.

Operazione 5

Descrizione

Attraverso una procedura viene inserito un esemplare in stato pendente nel caso in cui questo non sia presente nel database.

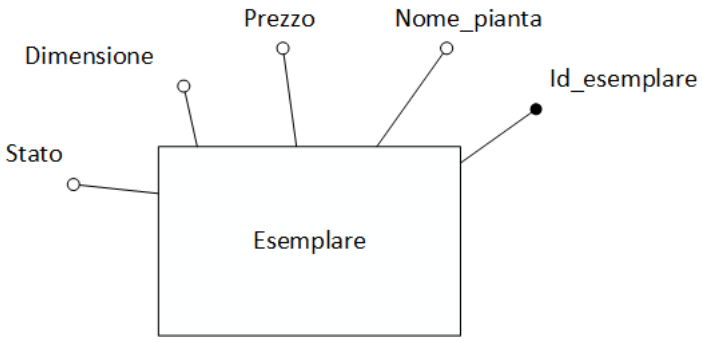


Tavola dei volumi: Operazione 5

Concetto	Tipo	Volume	Motivazione
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari

Tavola degli accessi per l'operazione 5

Concetto	Costrutto	Accessi	Tipo
Esemplare	E	1	S
Esemplare	E	1	L

Implementazione operazione 5

```
DROP PROCEDURE IF EXISTS GestioneOrdine;

DELIMITER $$

CREATE PROCEDURE GestioneOrdine(IN _Altezza int(11), IN _Larghezza
int(11), IN _Nome varchar(250), IN _Id_Pianta int(11), IN
_Contenitore int(11), IN _Id_esemplare int(11))
BEGIN

SELECT COUNT(*) INTO @esiste
FROM Esemplare
WHERE Id_esemplare = _Id_esemplare;

IF @esiste = 0 THEN

INSERT INTO Esemplare VALUES (_Altezza, _Larghezza,
NULL,'Pendente', _Nome, _Id_Pianta, _Contenitore, _Id_Esemplare );

ELSE

SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'ERRORE CREAZIONE ORDINE PENDENTE';

END IF;

END$$
DELIMITER;
```

Operazione 6

Descrizione

All'inserimento di una nuova scheda vengono associate tutte le manutenzioni automatiche da effettuare in ogni periodo.

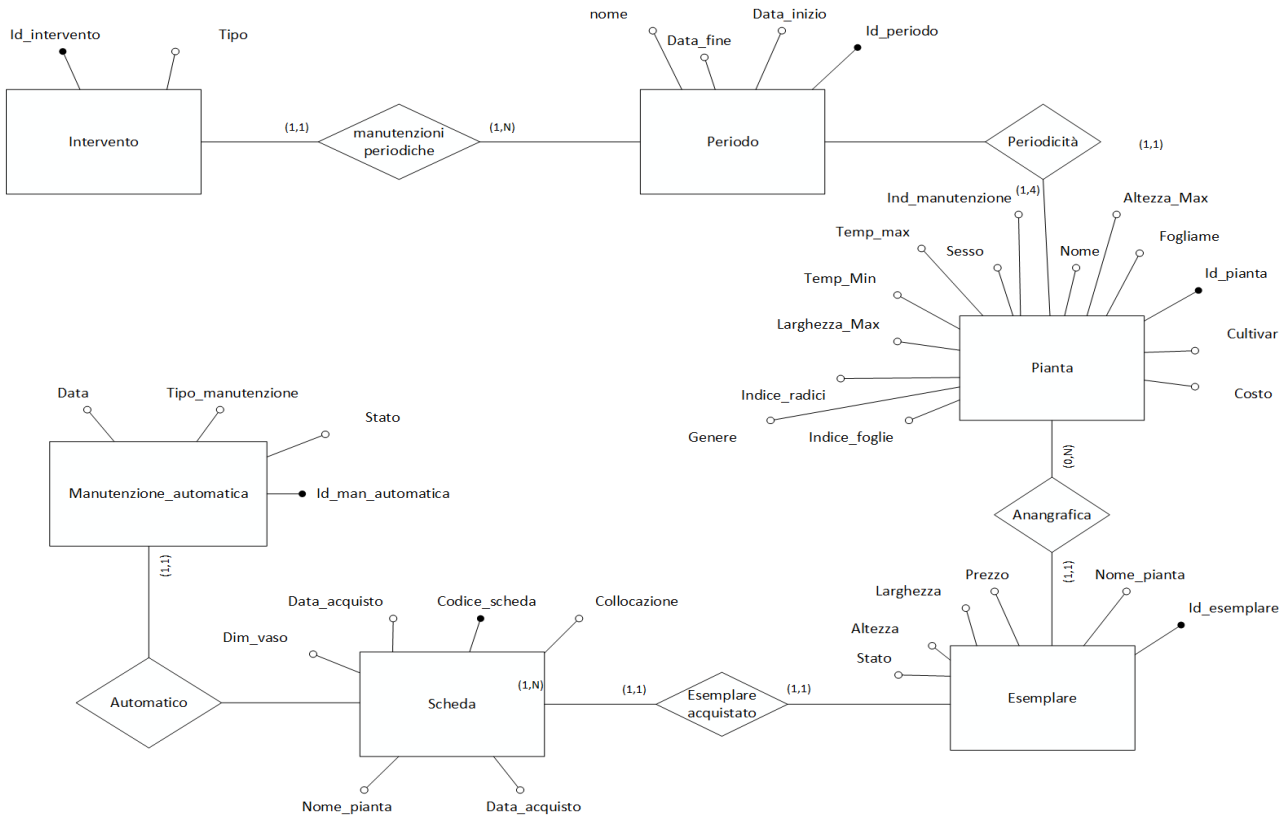


Tavola dei volumi: Operazione 6

Concetto	Tipo	Volume	Motivazione
Manutenzione Automatica	E	360000	Ogni esemplare ha in media 10 manutenzioni automatiche
Automatico	R	360000	Cardinalità (1,1) Manutenzione automatica

Scheda	E	36000	Cardinalità (1,1) con Esemplare
Esemplare acquistato	R	36000	Cardinalità (1,1) con Esemplare
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari
Anagrafica	R	36000	Cardinalità (1,1) con Esemplare
Pianta	E	5000	Ipotesi iniziale
Periodicità	R	20000	Ogni pianta può avere al massimo 4 periodicità
Periodo	E	20000	Cardinalità (1,1) con Periodicità
Manutenzioni periodiche	R	40000	In ogni periodo ci sono in media 2 interventi
Intervento	E	30	Ipotesi

Tavola degli accessi per l'operazione 6

Concetto	**Costrutto**	**Accessi**	**Tipo**
Manutenzione Automatica	E	8	S
Esemplare	E	1	L
Pianta	E	1	L
Periodo	E	4	L
Manutenzioni periodiche	R	8	L
Intervento	E	4	L

In ogni periodo ci sono in media 2 interventi e i periodi sono 4

Implementazione operazione 6

```
DROP TRIGGER IF EXISTS GestioneManutenzioniAutomatiche;

DELIMITER $$

CREATE TRIGGER GestioneManutenzioniAutomatiche
AFTER INSERT ON Scheda FOR EACH ROW
BEGIN

DECLARE finito INTEGER DEFAULT 0;
DECLARE _tipo VARCHAR (250);
DECLARE _periodo VARCHAR (250);
DECLARE i INTEGER DEFAULT 0;

DECLARE manutenzioni CURSOR FOR
SELECT PD.Nome,I.Tipo_intervento
FROM Esemplare E INNER JOIN Pianta PI ON E.Id_pianta = PI.Id_pianta
     INNER JOIN Periodo PD ON PI.Id_pianta = PD.Id_pianta
     INNER JOIN manutenzioni_periodiche MP ON PD.Id_periodo =
MP.Id_periodo
     INNER JOIN Intervento I ON MP.Id_intervento = I.Id_intervento
WHERE E.Id_esemplare= NEW.Id_Esemplare;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET finito=1;

SELECT IF(MAX(Id_man_automatica) IS NULL, 0,
MAX(Id_man_automatica)) INTO i
FROM Manutenzione_automatica ;

OPEN manutenzioni;

scan: LOOP

      FETCH manutenzioni INTO _periodo, _tipo ;

      IF(finito=1) THEN
        LEAVE scan;
      END IF;

     SET i=i+1;

     INSERT INTO Manutenzione_Automatica VALUES (_tipo, NULL,'DA
ESEGUIRE', _periodo, i, NEW.Codice_scheda);
```

```
END LOOP;

CLOSE manutenzioni;

END$$

DELIMITER;
```

Operazione 7

Descrizione

La seguente operazione ricava il numero totale di dipendenti di una serra e il numero di persone che servono per tutti gli interventi del mese. In base a ciò viene calcolato il numero di persone, se necessarie, da assumere.

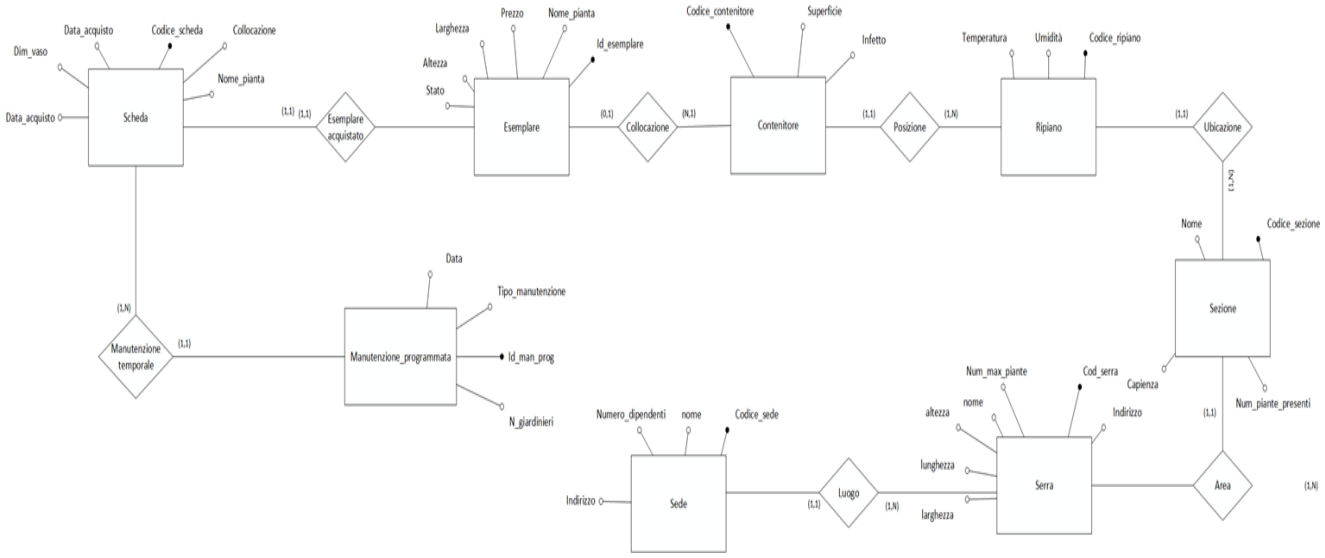


Tavola dei volumi: Operazione 7

Concetto	Tipo	Volume	Motivazione
Manutenzione programmata	E	72000	Ogni Esemplare ha in media 2 manutenzioni

Manutenzione temporale	R	72000	Cardinalità (1,1) con Manutenzione programmata
Scheda	E	36000	Cardinalità (1,1) con Esemplare
Esemplare acquistato	R	36000	Cardinalità (1,1) con Esemplare
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari
Collocazione	R	24000	2/3 degli Esemplare ha una collocazione nei vari contenitore
Contenitore	E	100000	In ogni ripiano ci sono in media 5 contenitori
Posizione	R	100000	Cardinalità (1,1) con Contenitore
Ripiano	E	20000	Ogni sezione ha in media 100 ripiani
Ubicazione	R	20000	Cardinalità (1,1) con Ripiano
Serra	E	20	Ipotesi iniziale

Sezione	E	200	In media ogni serra ha 10 sezioni
Area	R	200	Cardinalità (1,1) con Sezione
Luogo	R	5	Cardinalità (1,1) con Sede
Sede	E	5	Ogni sede ha in media 4 serra

Tavola degli accessi per l'operazione 7

Concetto	**Costrutto**	**Accessi**	**Tipo**
Manutenzione programmata	E	40000	L
Scheda	E	20000	L
Esemplare	E	20000	L
Contenitore	E	20000	L
Ripiano	E	4000	L
Serra	E	4	L
Sezione	E	40	L
Sede	E	1	L

Ogni scheda ha in media due manutenzioni

Implementazione operazione 7

```
DROP PROCEDURE IF EXISTS GestioneImpiegati;


DELIMITER $$

CREATE PROCEDURE GestioneImpiegati(IN _mese int(11), IN _anno
int(11), IN _sede varchar(250), OUT _DipNecessari int(11))
BEGIN

DECLARE _DipendentiNecesssari INTEGER DEFAULT 0;
DECLARE _dipPresenti INTEGER DEFAULT 0;
DECLARE finito INTEGER DEFAULT 0;

SELECT SUM(MP.N_Giardinieri), SD.Numero_Dipendenti INTO
_DipendentiNecesssari, _dipPresenti
FROM Manutenzione_programmata MP INNER JOIN Scheda S ON
MP.Codice_scheda = S.Codice_scheda
     INNER JOIN Esemplare E ON S.Id_esemplare = E.Id_esemplare
     INNER JOIN Contenitore C ON E.Codice_contenitore =
C.Codice_contenitore
     INNER JOIN Ripiano R ON C.Codice_ripiano = R.Codice_ripiano
     INNER JOIN Sezione SE ON R.Codice_sezione = SE.Codice_sezione
     INNER JOIN Serra SR ON SE.Codice_serra = SR.Codice_serra
     INNER JOIN Sede SD ON SD.Codice_sede = SR.Codice_sede
WHERE E.Stato='Venduto' AND MONTH(MP.Data) = _mese AND
YEAR(MP.Data) = _anno AND SD.Codice_sede=_sede;


SET _DipNecessari = _DipendentiNecesssari - _dipPresenti ;

IF _DipNecessari < 0 THEN
   SET _DipNecessari = 0;
END IF;

END$$
DELIMITER
```

Operazione 8
Descrizione

Successivamente all'aggiornamento di un ordine in stato di evaso, viene generata la scheda sul profilo del cliente del corrispondente esemplare.

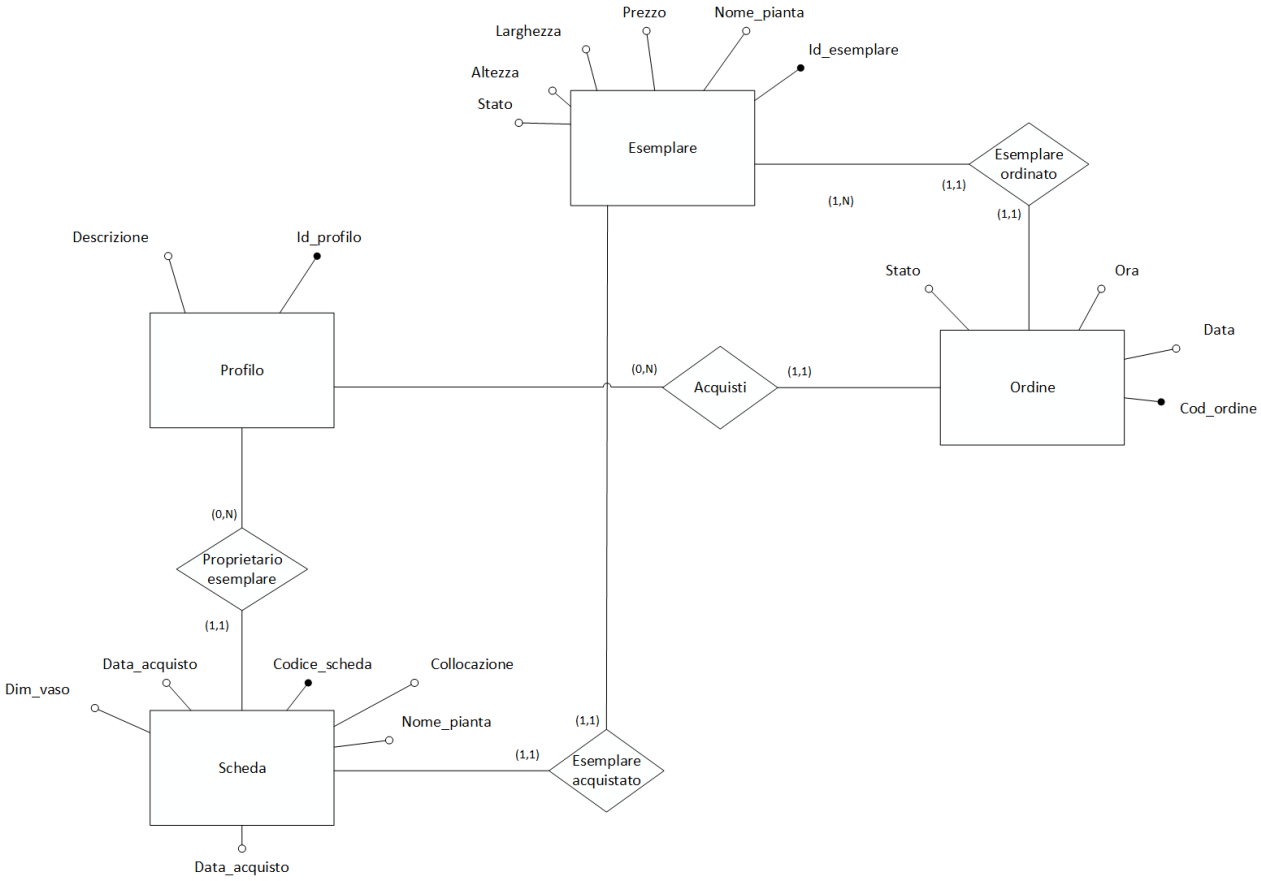


Tavola dei volumi: Operazione 8

Concetto	Tipo	Volume	Motivazione
Esemplare	E	36000	Ogni account ha in media 2 esemplare e ogni serra ha in media 1000 esemplari
Esemplare ordinato	R	36000	Cardinalità (1,1) con Esemplare

Ordine	E	36000	Cardinalità (1,1) con Esemplare ordinato
Acquisti	R	36000	Cardinalità (1,1) con ordine
Profilo	E	8000	Cardinalità (1,1) con account
Proprietario Esemplare	R	36000	Cardinalità (1,1) con Scheda
Scheda	E	36000	Cardinalità (1,1) con Esemplare
Esemplare acquistato	R	36000	Cardinalità (1,1) con Esemplare

Tavola degli accessi per l'operazione 8

Concetto	Costrutto	Accessi	Tipo
Esemplare	E	1	L
Scheda	E	1	S

Implementazione operazione 8

```
DROP TRIGGER IF EXISTS GestioneSchede;

DELIMITER $$

CREATE TRIGGER GestioneSchede
AFTER UPDATE ON ORDINE FOR EACH ROW
BEGIN

DECLARE _nome VARCHAR (250);
DECLARE i INTEGER DEFAULT 0;

IF(NEW.Stato='Evaso') THEN

   SELECT IF(MAX(Codice_Scheda) IS NULL, 0, MAX(Codice_Scheda))
INTO i
   FROM Scheda;
```

```
   SELECT Nome_pianta INTO _nome
   FROM Esemplare E
   WHERE Id_esemplare = NEW.Id_esemplare;

   SET i = i + 1;

   INSERT INTO Scheda VALUE (NULL, NULL, NEW.data,
NEW.Id_esemplare, NEW.Id_profilo, i);

END IF;

END$$

DELIMITER;
```

Progettazione logica

In questa fase abbiamo tradotto il diagramma E-R ristrutturato nel modello logico relazionale che come risultato produce lo schema logico del database.

Pianta (Nome , Sesso, Fogliame, Altezza_max, Larghezza_max, Costo, Ind_manutenzione, Temp_max, Temp_min, Indice_radici, Indice_foglie, Cultivar, Genere, Id_terreno, Id_pianta)

Conflitti (Id_pianta1, Id_pianta2 , Dist_Limite)

Illuminazione (Modalità, Esposizione, Num_ore_luce, Id_illuminazione)

Irrigazione (Esigenza, Quantita, Id_irrigazione)

Concimazione (Data, Id_concimazione)

Periodicita_concimazione (Quante_volte, Id_periodo, Id_concimazione)

Quantita_elemento (Quantita, Id_elemento, Id_concimazione)

Elemento_disciolto (Nome, Id_elemento)

Periodo (Nome, Mese_inzio, Mese_fine, Id_pianta, Id_irrigazione, Id_illuminazione, id_periodo)

Sede (Nome, Indirizzo, Citta, Numero_dipendenti, Codice_sede)

Serra (Nome, Indirizzo, Citta, Altezza, Larghezza, Lunghezza, Num_max_piante, Codice_sede, Codice_serra)

Sezione (Nome, Capienza, Num_max_presenti, Id_illuminazione, Codice_serra, Codice_sezione)

Ripiano (Temperatura, Umidita, Codice_sezione, Codice_ripiano)

Contenitore (Superficie, Infetto, Id_irrigazione, Codice_ripiano, Codice_contenitore)

Esemplare (Altezza, Larghezza, Prezzo, Stato, Nome_pianta, Id_pianta, Codice_contenitore, Id_esemplare)

Terreno (Consistenza, PH, Permeabilita, Tipo_suolo, Id_terreno)

Concentrazione (Percentuale, Id_elemento, Id_terreno)

Quantita_componente (Percentuale, Id_componente, Id_terreno)

Componente (Nome, Id_componente)

Patologia (Patogeno, Id_patologia,)

Sintomatologia (Descrizione, Immagine, Codice_sintomo)

Sintomo (Id_patologia, Codice_sintomo)

Malattia (Entita, Probabilita, Id_patologia, Id_periodo)

Intervento (Tipo_intervento, Id_intervento)

Manutenzioni_periodiche (Id_intervento, Id_periodo)

Trattamento (Prodotto, Modalita_somministrazione, Dose_consigliata, Id_trattamento)

Cura (Id_patologia, Id_trattamento)

Trattamenti_subiti (Data, Dose, Id_trattamento, Id_esemplare, Id_trattamento_subito)

Report_giornaliero (Data, Umidita, Stato_pianta, Id_contenitore, Id_esemplare, Id_report)

Report_patologie (Stato_malattia, Id_patologia, Id_report)

Concentrazione_giornaliera (Percentuale, Id_elemento, Id_report)

Account (Nome, Cognome, Sesso, Credibilita, Password, Domanda, Risposta, Nickname)

Preferenze (Id_pianta, Nickname)

Domanda (Thread, Testo, Timestamp , Nickname, Id_domanda)

Risposta (Testo, Timestamp , Id_domanda, Nickname, Id_risposta)

Valutazione (Voto, Id_risposta, Nickname)

Profilo (Descrizione, Id_profilo, Nickname)

Ordine (Ora, Data, Stato, Id_esemplare, Id_profilo, Cod_ordine)

Scheda (Dim_vaso, Collocazione, Data_acquisto, Altezza_acquisto, Larghezza_acquisto, Id_esemplare, Id_profilo, Codice_scheda)

Manutenzione_automatica (Tipo_manutenzione, Data, Stato, Periodo, Codice_scheda, Id_man_automatica)

Manutenzione_richiesta (Tipo_manutenzione, Data_scadenza, Codice_scheda, Id_richiesta)

Manutenzione_programmata (Tipo_manutenzione, N_giardinieri, Data, Codice_scheda, Id_man_programmata)

Giardino (Pixel_Altezza, Pixel_Larghezza, Id_profilo, Id_garden)

Settore (Ore_ luce, Esposizione, Stato, Id_garden, Id_settore)

Retta (X1, X2, Y1, Y2, Id_settore, Id_retta)

Cerchio (Raggio, X-Centro, Y-Centro, Id_pianta, Id_settore, Id_cerchio)

Specifica dei vincoli di integrità referenziale

Precedentemente abbiamo individuato le tabelle della base di dati a partire dallo schema E-R. Tali tabelle per poter contenere informazioni plausibile devo necessariamente verificare una serie di vincoli di integrità referenziale.

Qui di seguito verranno elencati tutti i vincoli di integrità referenziale presenti nella base di dati che abbiamo creato.

- Esiste un vincolo di integrità referenziale tra l'attributo *Id_terreno* della tabella *Terreno* e l'attributo *Id_terreno* della tabella *Pianta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_pianta1* della tabella *Conflitti* e l'attributo *Id_pianta* della tabella *Pianta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_pianta2* della tabella *Conflitti* e l'attributo *Id_pianta* della tabella *Pianta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_Concimazione* della tabella *Periodicità_Concimazione* e l'attributo *Id_Concimazione* della tabella *Concimazione.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_periodo* della tabella *Periodicità_Concimazione* e l'attributo *Id_Periodo* della tabella *Periodo.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_elemento* della tabella *Quantità_elemento* e l'attributo *Id_elemento* della tabella *Elemento_Disciolto.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_Concimazione* della tabella *Quantità_elemento* e l'attributo *Id_Concimazione* della tabella *Concimazione.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_pianta* della tabella *Periodo* e l'attributo *Id_pianta* della tabella *Pianta.*

- Esiste un vincolo di integrità referenziale tra l'attributo *Id_irrigazione* della tabella *Periodo* e l'attributo *Id_irrigazione* della tabella *Irrigazione.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_illuminazione* della tabella *Periodo* e l'attributo *Id_illuminazione* della tabella *Illuminazione.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_sede* della tabella *Serra* e l'attributo *Codice_sede* della tabella *Sede.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_sezione* della tabella *Ripiano* e l'attributo *Codice_sezione* della tabella *Sezione.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_ripiano* della tabella *Contenitore* e l'attributo *Codice_ripiano* della tabella *Ripiano.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_contenitore* della tabella *Esemplare* e l'attributo *Codice_Contenitore* della tabella *Contenitore.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_pianta* della tabella *Esemplare* e l'attributo *Id_pianta* della tabella *Pianta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_terreno* della tabella *Concentrazione* e l'attributo *Id_terreno* della tabella *Terreno.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_elemento* della tabella *Concentrazione* e l'attributo *Id_elemento* della tabella *Elemento_Disciolto.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_terreno* della tabella *Quantità_Componente* e l'attributo *Id_terreno* della tabella *Terreno.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_componente* della tabella *Quantità_componente* e l'attributo *Id_componente* della tabella *Componente.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_patologia* della tabella *Sintomo* e l'attributo *Id_patologia* della tabella *Patologia.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_sintomo* della tabella *Sintomo* e l'attributo *Codice_sintomo* della tabella *Sintomatologia.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_patologia* della tabella *Malattia* e l'attributo *Id_patologia* della tabella *Patologia.*

- Esiste un vincolo di integrità referenziale tra l'attributo *Id_periodo* della tabella *Malattia* e l'attributo *Id_periodo* della tabella *Periodo.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_intervento* della tabella *Manutenzione_periodiche* e l'attributo *Id_intervento* della tabella *Intervento.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_periodo* della tabella *Manutenzione_periodiche* e l'attributo *Id_periodo* della tabella *Periodo.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_patologia* della tabella *Cura* e l'attributo *Id_patologia* della tabella *Patologia.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_trattamento* della tabella *Cura* e l'attributo *Id_trattamento* della tabella *Trattamento.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_trattamento* della tabella *Trattamenti_subiti* e l'attributo *Id_trattamento* della tabella *Trattamento.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_Esemplare* della tabella *Trattamenti_subiti* e l'attributo *Id_esemplare* della tabella *Esemplare.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_contenitore* della tabella *Report_giornaliero* e l'attributo *Id_contenitore* della tabella *Contenitore.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_Esemplare* della tabella *Report_giornaliero* e l'attributo *Id_Esemplare* della tabella *Esemplare.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_patologia* della tabella *Report_patologie* e l'attributo *Id_patologia* della tabella *Patologia.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_report* della tabella *Report_patologie* e l'attributo *Id_report* della tabella *Report_giornaliero.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_elemento* della tabella *Concentrazione_giornaliera* e l'attributo *Id_elemento* della tabella *Elemento_Disciolto.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_report* della tabella *Concentrazione_giornaliera* e l'attributo *Id_report* della tabella *Report_Giornaliero.*

- Esiste un vincolo di integrità referenziale tra l'attributo *Nickname* della tabella *Preferenze* e l'attributo *Nickname* della tabella *Account.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_pianta* della tabella *Preferenze* e l'attributo *Id_pianta* della tabella *Pianta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Nickname* della tabella *Domanda* e l'attributo *Nickname* della tabella *Account.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Nickname* della tabella *Risposta* e l'attributo *Nickname* della tabella *Account.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_domanda* della tabella *Risposta* e l'attributo *Id_domanda* della tabella *Domanda.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_risposta* della tabella *Valutazione* e l'attributo *Id_risposta* della tabella *Risposta.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Nickname* della tabella *Valutazione* e l'attributo *Nickname* della tabella *Account.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Nickname* della tabella *Profilo* e l'attributo *Nickname* della tabella *Account.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_esemplare* della tabella *Ordine* e l'attributo *Id_esemplare* della tabella *Esemplare.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_profilo* della tabella *Profilo* e l'attributo *Id_profilo* della tabella *Ordine.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_esemplare* della tabella *Scheda* e l'attributo *Id_esemplare* della tabella *Esemplare.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_profilo* della tabella *Profilo* e l'attributo *Id_profilo* della tabella *Scheda.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_scheda* della tabella *Manutenzione_automatica* e l'attributo *Codice_scheda* della tabella *Scheda.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_scheda* della tabella *Manutenzione_richiesta* e l'attributo *Codice_scheda* della tabella *Scheda.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Codice_scheda* della tabella *Manutenzione_programmata* e l'attributo *Codice_scheda* della tabella *Scheda.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_profilo* della tabella *Profilo* e l'attributo *Id_profilo* della tabella *Giardino.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_garden* della tabella *Settore* e l'attributo *Id_garden* della tabella *Giardino.*

- Esiste un vincolo di integrità referenziale tra l'attributo *Id_settore* della tabella *Retta* e l'attributo *Id_settore* della tabella *Settore*.
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_settore* della tabella *Cerchio* e l'attributo *Id_settore* della tabella *Settore.*
- Esiste un vincolo di integrità referenziale tra l'attributo *Id_piante* della tabella *Pianta* e l'attributo *Id_pianta* della tabella *Cerchio.*

Analisi delle dipendenze funzionali e normalizzazione

Vediamo adesso quali sono le dipendenze funzionali che valgono sulle tabelle che abbiamo individuato.

TABELLA PIANTA

Id_pianta → Nome , Sesso, Fogliame, Altezza_max, Larghezza_max, Costo, Ind_manutenzione, Temp_max, Temp_min, Indice_radici, Indice_foglie, Cultivar, Genere, Id_terreno

TABELLA CONFLITTI

Id_pianta1, Id_pianta2 → Dist_Limite

TABELLA ILLUMINAZIONE

Id_illuminazione → Modalità, Esposizione, Num_ore_luce

TABELLA IRRIGAZIONE

Id_irrigazione → Esigenza, Quantita

TABELLA CONCIMAZIONE

Id_concimazione → Data

TABELLA PERIODICITA_CONCIMAZIONE

Id_periodo, Id_concimazione → Quante_volte

TABELLA QUANTITA_ELEMENTO

Id_elemento, Id_concimazione → Quantita

TABELLA ELEMENTO_DISCIOLTO

Id_elemento → Nome

TABELLA PERIODO

id_periodo → Nome, Mese_inzio, Mese_fine, Id_pianta, Id_irrigazione, Id_illuminazione

TABELLA SEDE

Codice_sede → Nome, Indirizzo, Citta, Numero_dipendenti

TABELLA SERRA

Codice_serra → Nome, Indirizzo, Citta, Altezza, Larghezza, Lunghezza, Num_max_piante, Codice_sede

TABELLA SEZIONE

Codice_sezione → Nome, Capienza, Num_max_presenti, Id_illuminazione, Codice_serra

TABELLA RIPIANO

Codice_ripiano → Temperatura, Umidita, Codice_sezione

TABELLA CONTENITORE

Codice_contenitore → Superficie, Infetto, Id_irrigazione, Codice_ripiano

TABELLA ESEMPLARE

Id_esemplare → Altezza, Larghezza, Prezzo, Stato, Nome_pianta, Id_pianta, Codice_contenitore

TABELLA TERRENO

Id_terreno → Consistenza, PH, Permeabilita, Tipo_suolo

TABELLA CONCENTRAZIONE

Id_elemento, Id_terreno → Percentuale

TABELLA QUANTITA COMPONENTE

Id_componente, Id_terreno → Percentuale

TABELLA COMPONENTE

Id_componente → Nome

TABELLA PATOLOGIA

Id_patologia → Patogeno

TABELLA SINTOMATOLOGIA

Codice_sintomo → Descrizione, Immagine

TABELLA SINTOMO

Id_patologia , Codice_sintomo

TABELLA MALATTIA

Id_patologia, Id_periodo → Entita, Probabilita

TABELLA INTERVENTO

Id_intervento → Tipo_intervento

TABELLA MANUTENZIONI_PERIODICHE

Id_periodo , Id_intervento

TABELLA TRATTAMENTO

Id_trattamento → Prodotto, Modalita_somministrazione, Dose_consigliata

TABELLA CURA

Id_patologia , Id_trattamento

TABELLA TRATTAMENTI_SUBITI

Id_trattamento_subito → Data, Dose, Id_trattamento, Id_esemplare

TABELLA REPORT_GIORNALIERO

Id_report → Data, Umidita, Stato_pianta, Id_contenitore, Id_esemplare

TABELLA REPORT_PATOLOGIE

Id_patologia, Id_report → Stato_malattia

TABELLA CONCENTRAZIONE_GIORNALIERA

Id_elemento, Id_report → Percentuale

TABELLA ACCOUNT

Nickname → Nome, Cognome, Sesso, Credibilita, Password, Domanda, Risposta

TABELLA PREFERENZE

Nickname , Id_pianta

TABELLA DOMANDA

Id_domanda → Thread, Testo, Timestamp , Nickname

TABELLA RISPOSTA

Id_risposta → Testo, Timestamp , Id_domanda, Nickname

TABELLA VALUTAZIONE

Id_risposta, Nickname → Voto

TABELLA PROFILO

Id_profilo → Nickname, Descrizione

TABELLA ORDINE

Cod_ordine → Ora, Data, Stato, Id_esemplare, Id_profilo

TABELLA SCHEDA

Codice_scheda → Dim_vaso, Collocazione, Data_acquisto, Altezza_acquisto, Larghezza_acquisto, Id_esemplare, Id_profilo

TABELLA MANUTENZIONE_AUTOMATICA

Id_man_automatica → Tipo_manutenzione, Data, Stato, Periodo, Codice_scheda

TABELLA MANUTENZIONE_RICHIESTA

Id_richiesta → Tipo_manutenzione, Data_scadenza, Codice_scheda

TABELLA MANUTENZIONE_PROGRAMMATA

Id_man_programmata → Tipo_manutenzione, N_giardinieri, Data, Codice_scheda

TABELLA GIARDINO

Id_garden → Pixel_Altezza, Pixel_Larghezza, Id_profilo

TABELLA SETTORE

Id_settore → Ore_luce, Esposizione, Stato, Id_garden

TABELLA RETTA

Id_retta → X1, X2, Y1, Y2, Id_settore

TABELLA CERCHIO

Id_cerchio → Raggio, X-Centro, Y-Centro, Id_pianta, Id_settore

Area analytics

Qui di seguito vengono riportate le funzionalità che permettono di analizzare i dati.

Funzionalità 1

La prima funzionalità riguarda lo smart design. Una volta passato il costo massimo, l'indice di manutenzione e il numero ore di luce del settore cerca nel database tutte le piante che hanno un costo minore del costo massimo e il massimo numero di ore luce uguale al numero di ore luce del settore. Infine se l'indice di manutenzione di tutte le piante è minore o uguale a quello desiderato le restituisce attraverso la tabella.

Implementazione funzionalità 1

```
DROP TABLE IF EXISTS PianteTarghet;
DROP PROCEDURE IF EXISTS RiprogettaSettore;
CREATE TABLE PianteTarghet LIKE Pianta;
DELIMITER $$

CREATE PROCEDURE RiprogettaSettore(IN _costo int(11), IN _indiceMan
int(11), IN _oreLuce int(11))
BEGIN

TRUNCATE TABLE PianteTarghet;

INSERT INTO PianteTarghet
SELECT P.*
FROM Pianta P INNER JOIN Periodo PD ON P.id_pianta = PD.id_pianta
     INNER JOIN Illuminazione I ON I.id_illuminazione =
PD.id_illuminazione
WHERE P.costo < _costo
GROUP BY P.id_pianta
HAVING MAX(Num_ore_luce)= _oreLuce;

SELECT AVG (PT.Ind_manutenzione) INTO @media
FROM PianteTarghet PT;

IF(@media > _indiceMan ) THEN
   TRUNCATE TABLE PianteTarghet;
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'ERRORE CREAZIONE ORDINE PENDENTE';
```

```
END IF;

END $$
DELIMITER ;
```

Funzionalità 2

La seconda funzionalità riguarda il reporting delle patologie e delle vendite. Passato il numero minimo di vendite e il numero massimo di esordi tollerabili vengono restituite rispettivamente le piante che hanno avuto un volume di vendite inferiore a quello specificato e le piante, con il relativo patogeno, che hanno avuto un numero di esordi superiore a quanto previsto

Implementazione funzionalità 2

```
DROP TABLE IF EXISTS PianteAmmalate;
DROP TABLE IF EXISTS PiantePocoVendute;
DROP PROCEDURE IF EXISTS ReportPiante;

CREATE TABLE IF NOT EXISTS PiantePocoVendute
(
 Pianta int(11) NOT NULL,
 Nome varchar(50) NOT NULL,
 Quantita int(11) NOT NULL,
 PRIMARY KEY(Pianta)
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE IF NOT EXISTS PianteAmmalate
(
 Pianta int(11) NOT NULL,
 Patogeno varchar(50) NOT NULL,
 N_casi int(11) NOT NULL,
 PRIMARY KEY(Pianta,Patogeno)
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DELIMITER $$

CREATE PROCEDURE ReportPiante(IN _venditeMin int(11), IN _Esordi
int(11))
BEGIN

TRUNCATE TABLE PianteAmmalate;
TRUNCATE TABLE PiantePocoVendute;
```

```
INSERT INTO PiantePocoVendute
SELECT P.id_pianta, P.Nome , COUNT(*) as quantita
FROM (Esemplare E INNER JOIN Ordine O ON E.id_esemplare =
O.id_esemplare) RIGHT JOIN Pianta P on P.id_pianta = E.id_pianta
GROUP BY P.id_pianta
HAVING COUNT(*) < _vediteMin;

INSERT INTO PianteAmmalate
SELECT E.id_pianta, P.Patogeno, COUNT(*) as casi
FROM Esemplare E INNER JOIN Report_Giornaliero RG ON E.id_esemplare
= RG.id_esemplare
                                                    INNER JOIN
Report_Patologie RP ON RG.id_report = RP.id_report
                                                    INNER JOIN
Patologia P ON RP.id_patologia = P.id_patologia
GROUP BY E.id_pianta, P.Patogeno
HAVING COUNT(*) > _Esordi ;

END $$
DELIMITER;
```

Funzionalità 3

La funzionalità 3 riguarda l'investigazione, in base alla storia dei dati memorizzati nel database delle serre, delle condizioni che hanno causato un danno alle piante. Passato un patogeno restituisce per ogni periodo i sintomi e il numero di casi rilevati.

Implementazione funzionalità 3

```
DROP TABLE IF EXISTS IndaginePatologica;
DROP PROCEDURE IF EXISTS IndaginePatologia;

CREATE TABLE IF NOT EXISTS IndaginePatologica
(
 Periodo varchar(50) NOT NULL,
 Cod_sintomo int(11) NOT NULL,
 Descrizione varchar(50) NOT NULL,
 NumCasi int(11) NOT NULL,
 PRIMARY KEY(Periodo,Cod_sintomo)
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
DELIMITER $$

CREATE PROCEDURE IndaginePatologia(IN _pat varchar(250))
BEGIN

TRUNCATE TABLE IndaginePatologica;

INSERT INTO IndaginePatologica
SELECT PD.Nome, SM.Codice_sintomo, SM.Descrizione, COUNT(*) as
num_casi
FROM Report_Patologie RP INNER JOIN Patologia P ON RP.id_patologia =
P.id_patologia
     INNER JOIN Malattia M ON M.id_patologia = P.id_patologia
     INNER JOIN Periodo PD ON PD.id_periodo = M.id_periodo
     INNER JOIN Sintomo SO ON SO.id_patologia = P.id_patologia
     INNER JOIN Sintomatologia SM ON SM.Codice_sintomo =
SO.Codice_sintomo
WHERE P.patogeno=_pat
GROUP BY PD.Nome, SM.Codice_sintomo;

END $$
DELIMITER;
```

Implementazione Database

```
SET NAMES latin1;
SET FOREIGN_KEY_CHECKS = 0;
BEGIN;
CREATE DATABASE IF NOT EXISTS `GiardinoVirtuale`;
COMMIT;

USE `GiardinoVirtuale`;

-- ----------------------------
--  Table structure for `Pianta`
-- ----------------------------
DROP TABLE IF EXISTS `Pianta`;
CREATE TABLE `Pianta` (
  `Nome` varchar(50) NOT NULL,
  `Sesso` varchar(50) NOT NULL,
  `Fogliame` varchar(50) NOT NULL,
  `Altezza_max` double NOT NULL,
  `Larghezza_max` double NOT NULL,
  `Costo` double NOT NULL,
  `Ind_mauntenzione` int(11) NOT NULL,
  `Temp_max` int(11) NOT NULL,
  `Temp_min` int(11) NOT NULL,
  `Indice_radici` double NOT NULL,
  `Indice_foglie` double NOT NULL,
  `Cultivar` varchar(50) NOT NULL,
  `Genere` varchar(50) NOT NULL,
  `Id_terreno` int(11) NOT NULL,
  `Id_pianta` int(11) NOT NULL,
  PRIMARY KEY (`Id_Pianta`),
  CONSTRAINT FK_Terreno1 FOREIGN KEY (Id_terreno) REFERENCES
Terreno(Id_terreno) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


-- ----------------------------
--  Table structure for `Conflitti`
-- ----------------------------
DROP TABLE IF EXISTS `Conflitti`;
CREATE TABLE `Conflitti` (
  `Id_pianta1` int(11) NOT NULL,
  `Id_pianta2` int(11) NOT NULL,
```

```
  `Dist_Limite` int(11) NOT NULL,
  PRIMARY KEY (`Id_pianta1`,`Id_pianta2`),
  CONSTRAINT FK_Pianta1 FOREIGN KEY (Id_Pianta1) REFERENCES
Pianta(Id_Pianta) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Pianta2 FOREIGN KEY (Id_Pianta2) REFERENCES
Pianta(Id_Pianta) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Illuminazione`
-- ----------------------------
DROP TABLE IF EXISTS `Illuminazione`;
CREATE TABLE `Illuminazione` (
  `Modalità` varchar(50) NOT NULL,
  `Esposizione` varchar(50) NOT NULL,
  `Num_ore_luce` int(11) NOT NULL,
  `Id_illuminazione` int(11) NOT NULL,
  PRIMARY KEY (`Id_illuminazione`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Irrigazione`
-- ----------------------------
DROP TABLE IF EXISTS `Irrigazione`;
CREATE TABLE `Irrigazione` (
  `Esigenza` int(11) NOT NULL,
  `Quantita` int(11) NOT NULL,
  `Id_irrigazione` int(11) NOT NULL,
  PRIMARY KEY (`Id_irrigazione`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Concimazione`
-- ----------------------------
DROP TABLE IF EXISTS `Concimazione`;
CREATE TABLE `Concimazione` (
  `Data` date NOT NULL,
  `Id_concimazione` int(11) NOT NULL,
  PRIMARY KEY (`Id_concimazione`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Periodicità`
```

```
-- ----------------------------
DROP TABLE IF EXISTS `Periodicita_Concimazione`;
CREATE TABLE `Periodicita_Concimazione` (
  `Quante_volte` int(11) NOT NULL,
  `Id_periodo` int(11) NOT NULL,
  `Id_concimazione` int(11) NOT NULL,
  PRIMARY KEY (`Id_periodo`,`id_concimazione`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Quantita_elemento`
-- ----------------------------
DROP TABLE IF EXISTS `Quantita_elemento`;
CREATE TABLE `Quantita_elemento` (
  `Quantita` int(11) NOT NULL,
  `Id_elemento` int(11) NOT NULL,
  `Id_concimazione` int(11) NOT NULL,
  PRIMARY KEY (`Id_elemento`,`id_concimazione`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Elemento_disciolto`
-- ----------------------------
DROP TABLE IF EXISTS `Elemento_disciolto`;
CREATE TABLE `Elemento_disciolto` (
  `Nome` varchar(50) NOT NULL,
  `Id_elemento` int(11) NOT NULL,
  PRIMARY KEY (`Id_elemento`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Periodo`
-- ----------------------------
DROP TABLE IF EXISTS `Periodo`;
CREATE TABLE `Periodo` (
  `Nome` varchar(50) NOT NULL,
  `Mese_inzio` int(11) NOT NULL,
  `Mese_fine` int(11) NOT NULL,
  `Id_pianta` int(11) NOT NULL,
  `Id_irrigazione` int(11) NOT NULL,
  `Id_illuminazione` int(11) NOT NULL,
  `Id_periodo` int(11) NOT NULL,
  PRIMARY KEY (`Id_periodo`),
```

```
  CONSTRAINT FK_Pianta3 FOREIGN KEY (Id_Pianta) REFERENCES
Pianta(Id_Pianta) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Illuminazione1 FOREIGN KEY (Id_illuminazione)
REFERENCES Illuminazione(Id_illuminazione) ON UPDATE NO ACTION ON
DELETE NO ACTION,
  CONSTRAINT FK_Irrigazione1 FOREIGN KEY (Id_irrigazione)
REFERENCES Iirrigazione(Id_irrigazione) ON UPDATE NO ACTION ON
DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Sede`
-- ----------------------------
DROP TABLE IF EXISTS `Sede`;
CREATE TABLE `Sede` (
  `Nome` varchar(50) NOT NULL,
  `Indirizzo` varchar(50) NOT NULL,
  `Citta` varchar(50) NOT NULL,
  `Numero_dipendenti` int(11) NOT NULL,
  `Codice_sede` int(11) NOT NULL,
  PRIMARY KEY (`Codice_sede`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Serra`
-- ----------------------------
DROP TABLE IF EXISTS `Serra`;
CREATE TABLE `Serra` (
  `Nome` varchar(50) NOT NULL,
  `Indirizzo` varchar(50) NOT NULL,
  `Citta` varchar(50) NOT NULL,
  `Altezza` int(11) NOT NULL,
  `Larghezza` int(11) NOT NULL,
  `Lunghezza` int(11) NOT NULL,
  `Num_max_piante` int(11) NOT NULL,
  `Codice_serra` int(11) NOT NULL,
  `Codice_sede` int(11) NOT NULL,
  PRIMARY KEY (`Codice_serra`),
  CONSTRAINT FK_Sede FOREIGN KEY (Codice_sede) REFERENCES
Sede(Codice_sede) ON UPDATE NO ACTION ON DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
```

```
--  Table structure for `Sezione`
-- ----------------------------
DROP TABLE IF EXISTS `Sezione`;
CREATE TABLE `Sezione` (
  `Nome` varchar(50) NOT NULL,
  `Capienza` int(11) NOT NULL,
  `Num_max_presenti` int(11) NOT NULL,
  `Id_illuminazione` int(11) NOT NULL,
  `Codice_serra` int(11) NOT NULL,
  `Codice_sezione` int(11) NOT NULL,
  PRIMARY KEY (`Codice_sezione`),
  CONSTRAINT FK_Serra FOREIGN KEY (Codice_serra) REFERENCES
Serra(Codice_serra) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Illuminazione2 FOREIGN KEY (Id_illuminazione)
REFERENCES Illuminazione(Id_illuminazione) ON UPDATE NO ACTION ON
DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Ripiano`
-- ----------------------------
DROP TABLE IF EXISTS `Ripiano`;
CREATE TABLE `Ripiano` (
  `Temperatura` int(11) NOT NULL,
  `Umidita` int(11) NOT NULL,
  `Codice_sezione` int(11) NOT NULL,
  `Codice_ripiano` int(11) NOT NULL,
  PRIMARY KEY (`Codice_ripiano`),
  CONSTRAINT FK_Sezione FOREIGN KEY (Codice_sezione) REFERENCES
Sezione(Codice_sezione) ON UPDATE NO ACTION ON DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Contenitore`
-- ----------------------------
DROP TABLE IF EXISTS `Contenitore`;
CREATE TABLE `Contenitore` (
  `Superficie` int(11) NOT NULL,
  `Infetto` varchar(50) NULL,
  `Id_irrigazione` int(11) NOT NULL,
  `Codice_ripiano` int(11) NOT NULL,
  `Codice_contenitore` int(11) NOT NULL,
  PRIMARY KEY (`Codice_contenitore`),
```

```
  CONSTRAINT FK_Ripiano FOREIGN KEY (Codice_ripiano) REFERENCES
Ripiano(Codice_ripiano) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Irrigazione2 FOREIGN KEY (Id_irrigazione)
REFERENCES Irrigazione(Id_irrigazione) ON UPDATE NO ACTION ON
DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Esemplare`
-- ----------------------------
DROP TABLE IF EXISTS `Esemplare`;
CREATE TABLE `Esemplare` (
  `Altezza` double NOT NULL,
  `Larghezza` double NOT NULL,
  `Prezzo` int(11) NULL,
  `Stato` varchar(50) NOT NULL,
  `Nome_pianta` varchar(50) NOT NULL,
  `Id_pianta` int(11) NOT NULL,
  `Codice_contenitore` int(11) NULL,
  `Id_esemplare` int(11) NOT NULL,
  PRIMARY KEY (`Id_esemplare`),

  CONSTRAINT FK_Pianta4 FOREIGN KEY (Id_pianta) REFERENCES
Pianta(Id_pianta) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Contenitore1 FOREIGN KEY (Codice_contenitore)
REFERENCES Contenitore (Codice_contenitore) ON UPDATE NO ACTION ON
DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Terreno`
-- ----------------------------
DROP TABLE IF EXISTS `Terreno`;
CREATE TABLE `Terreno` (
  `Consistenza` double NOT NULL,
  `PH` int(11) NOT NULL,
  `Permeabilita` double NOT NULL,
  `Tipo_suolo` varchar(50) NOT NULL,
  `Id_terreno` int(11) NOT NULL,
  PRIMARY KEY (`Id_terreno`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
```

```
--  Table structure for `Concentrazione`
-- ----------------------------
DROP TABLE IF EXISTS `Concentrazione`;
CREATE TABLE `Concentrazione` (
  `Percentuale` int(11) NOT NULL,
  `Id_elemento` int(11) NOT NULL,
  `Id_terreno` int(11) NOT NULL,
  PRIMARY KEY (`Id_terreno`, `Id_elemento`),
  CONSTRAINT FK_Elemento_disciolto FOREIGN KEY (Id_elemento)
REFERENCES Elemento_disciolto(Id_elemento) ON UPDATE NO ACTION ON
DELETE NO ACTION,
  CONSTRAINT FK_Terreno2 FOREIGN KEY (Id_terreno) REFERENCES
Terreno(Id_terreno) ON UPDATE NO ACTION ON DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Quantita_componente`
-- ----------------------------
DROP TABLE IF EXISTS `Quantita_componente`;
CREATE TABLE `Quantita_componente` (
  `Percentuale` int(11) NOT NULL,
  `Id_componente` int(11) NOT NULL,
  `Id_terreno` int(11) NOT NULL,
  PRIMARY KEY (`Id_terreno`, `Id_componente`),
  CONSTRAINT FK_Terreno3 FOREIGN KEY (Id_terreno) REFERENCES
Terreno(Id_terreno) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Componente FOREIGN KEY (Id_componente) REFERENCES
Componente(Id_componente) ON UPDATE NO ACTION ON DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Componente`
-- ----------------------------
DROP TABLE IF EXISTS `Componente`;
CREATE TABLE `Componente` (
  `Nome` varchar(50) NOT NULL,
  `Id_componente` int(11) NOT NULL,
  PRIMARY KEY (`Id_componente`)
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Patologia`
-- ----------------------------
```

```
DROP TABLE IF EXISTS `Patologia`;
CREATE TABLE `Patologia` (
  `Patogeno` varchar(50) NOT NULL,
  `Id_patologia` int(11) NOT NULL,
  PRIMARY KEY (`Id_patologia`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Sintomatologia`
-- ----------------------------
DROP TABLE IF EXISTS `Sintomatologia`;
CREATE TABLE `Sintomatologia` (
  `Descrizione` varchar(50) NOT NULL,
  `Immagine` varchar(50) NULL,
  `Codice_sintomo` int(11) NOT NULL,
  PRIMARY KEY (`Codice_sintomo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Sintomo`
-- ----------------------------
DROP TABLE IF EXISTS `Sintomo`;
CREATE TABLE `Sintomo` (
  `Id_patologia` int(11) NOT NULL,
  `Codice_sintomo` int(11) NOT NULL,
  PRIMARY KEY (`Codice_sintomo`,`Id_patologia`),
  CONSTRAINT FK_Patologia1 FOREIGN KEY (Id_patologia) REFERENCES
Patologia(Id_patologia) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Sintomatologia1 FOREIGN KEY (Codice_sintomo)
REFERENCES Sintomatologia(Codice_sintomo) ON UPDATE NO ACTION ON
DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Malattia`
-- ----------------------------
DROP TABLE IF EXISTS `Malattia`;
CREATE TABLE `Malattia` (
  `Entita` varchar(50) NOT NULL,
  `Probabilita` int(11) NOT NULL,
  `Id_patologia` int(11) NOT NULL,
  `Id_periodo` int(11) NOT NULL,
  PRIMARY KEY (`Id_periodo`,`Id_patologia`),
```

```
  CONSTRAINT FK_Patologia2 FOREIGN KEY (Id_patologia) REFERENCES
Patologia(Id_patologia) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Periodo1 FOREIGN KEY (Id_periodo) REFERENCES
Periodo(Id_periodo) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Intervento`
-- ----------------------------
DROP TABLE IF EXISTS `Intervento`;
CREATE TABLE `Intervento` (
  `Tipo_intervento` varchar(50) NOT NULL,
  `Id_intervento` int(11) NOT NULL,
  PRIMARY KEY (`Id_intervento`)
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Manutenzioni_periodiche`
-- ----------------------------
DROP TABLE IF EXISTS `Manutenzioni_periodiche`;
CREATE TABLE `Manutenzioni_periodiche` (
  `Id_intervento` int(11) NOT NULL,
  `Id_periodo` int(11) NOT NULL,
  PRIMARY KEY (`Id_intervento`,`Id_periodo`),
  CONSTRAINT FK_Intervento FOREIGN KEY (Id_intervento) REFERENCES
Intervento(Id_intervento) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Periodo FOREIGN KEY (Id_periodo) REFERENCES
Periodo(Id_periodo) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Trattamento`
-- ----------------------------
DROP TABLE IF EXISTS `Trattamento`;
CREATE TABLE `Trattamento` (
  `Prodotto` varchar(50) NOT NULL,
  `Modalita_somministrazione` varchar(50) NOT NULL,
  `Dose_consigliata` double NOT NULL,
  `Id_trattamento` int(11) NOT NULL,
  PRIMARY KEY (`Id_trattamento`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
```

```
--  Table structure for `Cura`
-- ----------------------------
DROP TABLE IF EXISTS `Cura`;
CREATE TABLE `Cura` (
  `Id_patologia` int(11) NOT NULL,
  `Id_trattamento` int(11) NOT NULL,
  PRIMARY KEY (`Id_patologia`,`Id_trattamento`),
  CONSTRAINT FK_Patologia3 FOREIGN KEY (Id_patologia) REFERENCES
Patologia(Id_patologia) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Trattamento1 FOREIGN KEY (Id_trattamento)
REFERENCES Trattamento(Id_trattamento) ON UPDATE NO ACTION ON
DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Trattamenti_subiti`
-- ----------------------------
DROP TABLE IF EXISTS `Trattamenti_subiti`;
CREATE TABLE `Trattamenti_subiti` (
  `Data`date NOT NULL,
  `Dose` int(11) NOT NULL,
  `Id_trattamento` int(11) NOT NULL,
  `Id_esemplare` int(11) NOT NULL,
  `Id_trattamento_subito` int(11) NOT NULL,
  PRIMARY KEY (`Id_trattamento`),
  CONSTRAINT FK_Esemplare1 FOREIGN KEY (Id_esemplare) REFERENCES
Esemplare(Id_esemplare) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Trattamento2 FOREIGN KEY (Id_trattamento)
REFERENCES Trattamento(Id_trattamento) ON UPDATE NO ACTION ON
DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Report_giornaliero`
-- ----------------------------
DROP TABLE IF EXISTS `Report_giornaliero`;
CREATE TABLE `Report_giornaliero` (
  `Data`date NOT NULL,
  `Umidita` int(11) NOT NULL,
  `Stato_pianta` varchar(50) NOT NULL,
  `Id_contenitore` int(11) NOT NULL,
  `Id_esemplare` int(11) NOT NULL,
  `Id_report` int(11) NOT NULL,
```

```
  PRIMARY KEY (`Id_report`),
  CONSTRAINT FK_Esemplare2 FOREIGN KEY (Id_esemplare) REFERENCES
Esemplare(Id_esemplare) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Contenitore2 FOREIGN KEY (Id_contenitore)
REFERENCES Contenitore(Codice_contenitore) ON UPDATE NO ACTION ON
DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Report_patologie`
-- ----------------------------
DROP TABLE IF EXISTS `Report_patologie`;
CREATE TABLE `Report_patologie` (
  `Stato_malattia` varchar(50) NOT NULL,
  `Id_patologia` int(11) NOT NULL,
  `Id_report` int(11) NOT NULL,
  PRIMARY KEY (`Id_report`,`Id_patologia`),
  CONSTRAINT FK_Patologia4 FOREIGN KEY (Id_patologia) REFERENCES
Patologia(Id_patologia) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Report1 FOREIGN KEY (Id_report) REFERENCES
Report_giornaliero(Id_report) ON UPDATE NO ACTION ON DELETE NO
ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Concentrazione_giornaliera`
-- ----------------------------
DROP TABLE IF EXISTS `Concentrazione_giornaliera`;
CREATE TABLE `Concentrazione_giornaliera` (
  `Percentuale` int(11) NOT NULL,
  `Id_elemento` int(11) NOT NULL,
  `Id_report` int(11) NOT NULL,
  PRIMARY KEY (`Id_report`, `Id_elemento`),
  CONSTRAINT FK_Report2 FOREIGN KEY (Id_report) REFERENCES
Report_giornaliero(Id_report) ON UPDATE NO ACTION ON DELETE NO
ACTION,
  CONSTRAINT FK_Elemento_disciolto2 FOREIGN KEY (Id_elemento)
REFERENCES Elemento_disciolto(Id_elemento) ON UPDATE NO ACTION ON
DELETE NO ACTION
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Account`
```

```
-- ----------------------------
DROP TABLE IF EXISTS `Account`;
CREATE TABLE `Account` (
  `Nome` varchar(50) NOT NULL,
  `Cognome` varchar(50) NOT NULL,
  `Sesso` varchar(50) NOT NULL,
  `Credibilita` int(11)  NULL,
  `Password` varchar(50) NOT NULL,
  `Domanda` varchar(50) NOT NULL,
  `Risposta` varchar(50) NOT NULL,
  `Nickname` varchar(50)NOT NULL,
  PRIMARY KEY (`Nickname`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Preferenze`
-- ----------------------------
DROP TABLE IF EXISTS `Preferenze`;
CREATE TABLE `Preferenze` (
  `Id_pianta` int(11) NOT NULL,
  `Nickname` varchar(50) NOT NULL,
  PRIMARY KEY (`Nickname`,`Id_pianta`),
  CONSTRAINT FK_Pianta6 FOREIGN KEY (Id_pianta) REFERENCES
Pianta(Id_pianta) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Account1 FOREIGN KEY (Nickname) REFERENCES Account
(Nickname) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Domanda`
-- ----------------------------
DROP TABLE IF EXISTS `Domanda`;
CREATE TABLE `Domanda` (
  `Thread` varchar(50) NOT NULL,
  `Testo` varchar(250) NOT NULL,
  `Timestamp` timestamp NOT NULL,
  `Id_domanda` int(11) NOT NULL,
  `Nickname` varchar(50) NOT NULL,
  PRIMARY KEY (`Id_domanda`),
  CONSTRAINT FK_Account2 FOREIGN KEY (Nickname) REFERENCES
Account(Nickname) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
-- ----------------------------
--  Table structure for `Risposta`
-- ----------------------------
DROP TABLE IF EXISTS `Risposta`;
CREATE TABLE `Risposta` (
  `Testo` varchar(250) NOT NULL,
  `Timestamp` timestamp NOT NULL,
  `Id_domanda` int(11) NOT NULL,
  `Id_risposta` int(11) NOT NULL,
  `Nickname` varchar(50) NOT NULL,
  PRIMARY KEY (`Id_risposta`),
  CONSTRAINT FK_Domanda FOREIGN KEY (Id_domanda) REFERENCES
Domanda(Id_domanda) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Account3 FOREIGN KEY (Nickname) REFERENCES
Account(Nickname) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Valutazione`
-- ----------------------------
DROP TABLE IF EXISTS `Valutazione`;
CREATE TABLE `Valutazione` (
  `Voto` int(11) NOT NULL,
  `Id_risposta` int(11) NOT NULL,
  `Nickname` varchar(50) NOT NULL,
  PRIMARY KEY (`Id_risposta`,`Nickname`),
  CONSTRAINT FK_Risposta FOREIGN KEY (Id_risposta) REFERENCES
Risposta(Id_risposta) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Account4 FOREIGN KEY (Nickname) REFERENCES
Account(Nickname) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Profilo`
-- ----------------------------
DROP TABLE IF EXISTS `Profilo`;
CREATE TABLE `Profilo` (
  `Descrizione` varchar(250) NOT NULL,
  `Id_profilo` int(11) NOT NULL,
  `Nickname` varchar(50) NOT NULL,
  PRIMARY KEY (`Id_profilo`),
  CONSTRAINT FK_Account5 FOREIGN KEY (Nickname) REFERENCES
Account(Nickname) ON UPDATE NO ACTION ON DELETE NO ACTION
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Ordine`
-- ----------------------------
DROP TABLE IF EXISTS `Ordine`;
CREATE TABLE `Ordine` (
  `Ora` time NOT NULL,
  `Data` date NOT NULL,
  `Stato` varchar(50) NOT NULL,
  `Id_esemplare` int(11) NOT NULL,
  `Id_profilo` int(11) NOT NULL,
  `Cod_ordine` int(11) NOT NULL,
  PRIMARY KEY (`Cod_ordine`),
  CONSTRAINT FK_Esemplare4 FOREIGN KEY (Id_esemplare) REFERENCES
Esemplare(Id_esemplare) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Profilo1 FOREIGN KEY (Id_profilo) REFERENCES
Profilo(Id_profilo) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

 -- ----------------------------
--  Table structure for `Scheda`
-- ----------------------------
DROP TABLE IF EXISTS `Scheda`;
CREATE TABLE `Scheda` (
  `Dim_vaso` double NULL,
  `Collocazione` varchar(50) NOT NULL,
  `Data_acquisto` date NOT NULL,
  `Id_esemplare` int(11) NOT NULL,
  `Id_profilo` int(11) NOT NULL,
  `Codice_scheda` int(11) NOT NULL,
  PRIMARY KEY (`Codice_scheda`),
  CONSTRAINT FK_Esemplare5 FOREIGN KEY (Id_esemplare) REFERENCES
Esemplare(Id_esemplare) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Profilo2 FOREIGN KEY (Id_profilo) REFERENCES
Profilo(Id_profilo) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Manutenzione_automatica`
-- ----------------------------
DROP TABLE IF EXISTS `Manutenzione_automatica`;
CREATE TABLE `Manutenzione_automatica` (
```

```
  `Tipo_manutenzione` varchar(50) NOT NULL,
  `Data` date NOT NULL,
  `Stato` varchar(50) NOT NULL,
  `Periodo` varchar(50) NOT NULL,
  `Id_man_automatica` int(11) NOT NULL,
  `Codice_scheda` int(11) NOT NULL,
  PRIMARY KEY (`Id_man_automatica`),
  CONSTRAINT FK_Scheda1 FOREIGN KEY (Codice_scheda) REFERENCES
Scheda(Codice_scheda) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Manutenzione_richiesta`
-- ----------------------------
DROP TABLE IF EXISTS `Manutenzione_richiesta`;
CREATE TABLE `Manutenzione_richiesta` (
  `Tipo_manutenzione` varchar(50) NOT NULL,
  `Data_scadenza` date NOT NULL,
  `Id_richiesta` int(11) NOT NULL,
  `Codice_scheda` int(11) NOT NULL,
  PRIMARY KEY (`Id_richiesta`),
  CONSTRAINT FK_Scheda2 FOREIGN KEY (Codice_scheda) REFERENCES
Scheda(Codice_scheda) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Manutenzione_programmata`
-- ----------------------------
DROP TABLE IF EXISTS `Manutenzione_programmata`;
CREATE TABLE `Manutenzione_programmata` (
  `Tipo_manutenzione` varchar(50) NOT NULL,
  `Data` date NOT NULL,
  `N_giardinieri` int(11) NOT NULL,
  `Id_man_programmata` int(11) NOT NULL,
  `Codice_scheda` int(11) NOT NULL,
  PRIMARY KEY (`Id_man_programmata`),
  CONSTRAINT FK_Scheda3 FOREIGN KEY (Codice_scheda) REFERENCES
Scheda(Codice_scheda) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Giardino`
-- ----------------------------
```

```
DROP TABLE IF EXISTS `Giardino`;
CREATE TABLE `Giardino` (
  `Pixel_Altezza` int(11) NOT NULL,
  `Pixel_larghezza` int(11) NOT NULL,
  `Id_profilo` int(11) NOT NULL,
  `Id_garden` int(11) NOT NULL,
  PRIMARY KEY (`Id_garden`),
  CONSTRAINT FK_Profilo4 FOREIGN KEY (Id_profilo) REFERENCES
Profilo(Id_profilo) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Settore`
-- ----------------------------
DROP TABLE IF EXISTS `Settore`;
CREATE TABLE `Settore` (
  `Ore_luce` int(11) NOT NULL,
  `Num_Piante_Presenti` int(11) NOT NULL,
  `Esposizione`  varchar(50) NOT NULL,
  `Stato`  varchar(50) NOT NULL,
  `Id_garden` int(11) NOT NULL,
  `Id_settore` int(11) NOT NULL,
  PRIMARY KEY (`Id_settore`),
  CONSTRAINT FK_Giardino FOREIGN KEY (Id_garden) REFERENCES
Giardino(Id_garden) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- ----------------------------
--  Table structure for `Retta`
-- ----------------------------
DROP TABLE IF EXISTS `Retta`;
CREATE TABLE `Retta` (
  `X1` double NOT NULL,
  `X2` double NOT NULL,
  `Y1` double NOT NULL,
  `Y2` double NOT NULL,
  `Id_settore` int(11) NOT NULL,
  `Id_retta` int(11) NOT NULL,
  PRIMARY KEY (`Id_retta`),
  CONSTRAINT FK_Settore1 FOREIGN KEY (Id_settore) REFERENCES
Settore(Id_settore) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
-- ----------------------------
--  Table structure for `Cerchio`
-- ----------------------------
DROP TABLE IF EXISTS `Cerchio`;
CREATE TABLE `Cerchio` (
  `Raggio` int(11) NOT NULL,
  `X-Centro` int(11) NOT NULL,
  `Y-Centro` int(11) NOT NULL,
  `Id_pianta` int(11) NOT NULL,
  `Id_settore` int(11) NOT NULL,
  `Id_cerchio` int(11) NOT NULL,
  PRIMARY KEY (`Id_cerchio`),
  CONSTRAINT FK_Settore2 FOREIGN KEY (Id_settore) REFERENCES
Settore(Id_settore) ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT FK_Pianta7 FOREIGN KEY (Id_pianta) REFERENCES
Pianta(Id_pianta) ON UPDATE NO ACTION ON DELETE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```