ELAB - Laboratorio Elettronico

Progetto Esame Basi di Dati – Alberto Garbui (Mat.561226)

Abstract

Questo progetto ha come scopo principale la gestione della produzione di schede elettroniche della ditta ELAB (Electronic LAB). L'interfaccia web permette ai diversi lavoratori di accedere alla base dati dell'azienda ed apportare modifiche relative al proprio ambito lavorativo. L'idea è di poter migliorare la produzione permettendo una miglior comunicazione tra ufficio tecnico che crea un progetto ed il singolo operaio che assemblerà la scheda, oltre che permettere all'ufficio di amministrazione di avere una visione globale sullo stato della produzione in tempo reale.

Password di accesso

Di seguito sono elencati nomi utenti e password di ogni tipologia utente per permettere l'accesso all'interfaccia web:

Tipologia: ***Operaio*** nome utente: ***matteg9*** password: ***ELAB***

Tipologia: ***Tecnico*** nome utente: ***alberb5*** password: ***ELAB***

Tipologia: ***Venditore*** nome utente: ***ginoz2*** password: ***ELAB***

Tipologia: ***Amministratore*** nome utente: ***admin*** password: ***ELAB***

Analisi dei requisiti

Interfaccia Web che permetta l'accesso di ogni lavoratore nella sua area di competenza.

Quattro tipi di utenti:

- Operai
- Tecnici
- Venditori
- Amministratori

Classe Utenti:

- Id (identifica univocamente l'utente)
- Nome
- Cognome
- Data Assunzione
- Username
- Password

Classe Progetti:

- Id (identifica univocamente il progetto)
- Descrizione
- DataCreazione
- DataUltimaModifica

Ogni progetto è creato e/o modificato da un singolo tecnico. Un progetto può essere un aggiornamento di un progetto esistente. Ogni progetto è caratterizzato da un determinato numero di componenti. La DataCreazione ha una duplice funzione: se dichiarata indica la data di creazione del progetto, se NULL indica che il progetto è in fase di progettazione.

Classe Commesse:

- Id (identifica univocamente la commessa)
- QuantitàDaProdurre
- DataCommessa

Ogni singola commessa è legata ad un singolo progetto, commissionata da un singolo venditore in favore di un singolo cliente.

Classe Clienti:

- Id (identifica univocamente il cliente)
- Nome
- Cognome
- Società
- Livello
- DataPrimoOrdine

Classe Componenti:

- Id (identifica univocamente il componente)
- SiglaProduttore
- Descrizione

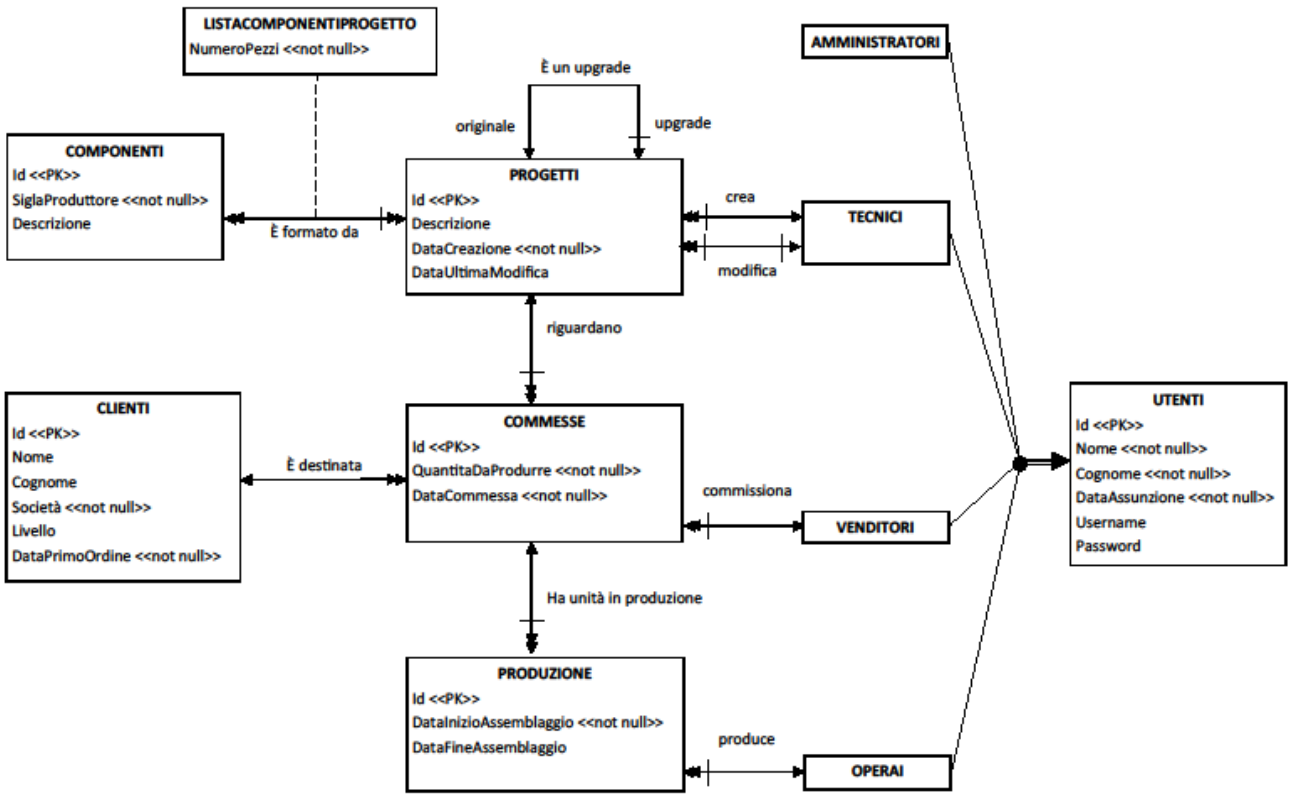
Classe Produzione:

- Id (identifica univocamente la produzione)
- DataInizioAssemblaggio
- DataFineAssemblaggio

Ogni produzione è relativa ad una singola unità (scheda elettronica) assemblata da un singolo operaio per una determinata commessa.

Progettazione Concettuale

Schema Concettuale:



Progettazione Concettuale – Classi

Utenti: rappresenta gli utenti che possono accedere all'interfaccia web ed alla base di dati relativamente al proprio ambito lavorativo.

- Id: int <<PK>>
- Nome: string NOT NULL

- Cognome: string NOT NULL
- Data Assunzione: timestamp NOT NULL
- Username: string
- Password: string

Operai: utenti che possono accedere all'interfaccia web per iniziare un assemblaggio di una scheda oppure per dichiarare la terminazione di un assemblaggio.

Venditori: utenti che possono accedere all'interfaccia web per inserire una nuova commessa nel sistema dopo aver stipulato un contratto con un cliente.

Tecnici: utenti che possono accedere all'interfaccia web per inserire un nuovo progetto oppure per modificarne uno esistente.

Amministratori: utenti che possono accedere all'interfaccia web per controllare la produzione e gestire vari aspetti dell'azienda.

Progetti: rappresentano i progetti creati dai vari tecnici utilizzando vari componenti

- Id: int <<PK>>
- Descrizione: string
- DataCreazione: timestamp
- DataUltimaModifica: timestamp

Commesse: rappresentano le commesse create dai vari venditori per conto di singoli clienti relative ad un determinato progetto

- Id: int <<PK>>
- QuantitàDaProdurre: int NOT NULL
- DataCommessa: timestamp NOT NULL

Clienti: rappresentano i clienti che hanno ordinato almeno una commessa

- Id: int <<PK>>
- Nome: string
- Cognome: string
- Società: string NOT NULL
- DataPrimoOrdine: timestamp NOT NULL

Componenti: rappresentano i componenti utilizzati nei progetti attuali o disponibili per progetti futuri

- Id: string <<PK>>
- SiglaProduttore: string NOT NULL
- Descrizione: string

Produzione: rappresentano le produzioni degli opera relative a singole unità (schede) sia in fase di assemblaggio che terminate per conto di una determinata commessa.

- Id: int <<PK>>
- DataInizioAssemblaggio: timestamp NOT NULL
- DataFineAssemblaggio: timestamp

Progettazione Concettuale – Associazioni

Tecnici-Progetti: crea

- Ogni tecnico crea più progetti o nessuno, ogni progetto è creato da un singolo tecnico
- Molteplicità: M:1
- Totalità: totale da progetti a tecnici, parziale da tecnici a progetti

Tecnici-Progetti: modifica

- Ogni tecnico modifica più progetti o nessuno, ogni progetto è modificato da un singolo tecnico o nessuno
- Molteplicità: M:1
- Totalità: parziale sia da progetti a tecnici che da tecnici a progetti

Venditori-Commesse: commissiona

- Ogni venditore commissiona più commesse o nessuna, ogni commessa è commissionata da un singolo venditore
- Molteplicità: M:1
- Totalità: totale da commesse a venditori, parziale da venditori a commesse

Operai-Produzione: produce

- Ogni operaio produce più unità lavorative o nessuna, ogni unità lavorativa è prodotta da un singolo operaio
- Molteplicità: M:1
- Totalità: totale da produzione a operai, parziale da operai a produzione

Progetti-Progetti: è un upgrade

- Ogni progetto può avere un upgrade o nessuno, ogni upgrade è un aggiornamento di un progetto esistente
- Molteplicità: 1:1
- Totalità: totale da upgrade ad originale, parziale da originale a upgrade

Progetti-Componenti: è formato da

- Ogni progetto è formato da più componenti, ogni componente è utilizzato in più progetti o nessuno
- Molteplicità: M:N
- Totalità: totale da progetti a componenti, parziale da componenti a progetti
- Questa associazione ha come attributo: Numero Pezzi

Commesse-Clienti: è destinata

- Ogni commessa è destinata ad un singolo cliente, ogni cliente ha ordinato più commesse
- Molteplicità: M:1
- Totalità: totale sia da commesse a clienti che da clienti a commesse

Commesse-Progetti: riguardano

- Ogni commessa riguarda un singolo progetto, ogni progetto può essere presente in più commesse o nessuna
- Molteplicità: M:1
- Totalità: totale da commesse a progetti, parziale da progetti a commesse

Commesse-Produzione: ha unità in produzione

- Ogni commessa può avere più unità in produzione o nessuna, ogni unità in produzione è relativa ad una singola commessa
- Molteplicità: 1:M
- Totalità: totale da produzione a commesse, parziale da commesse a produzione

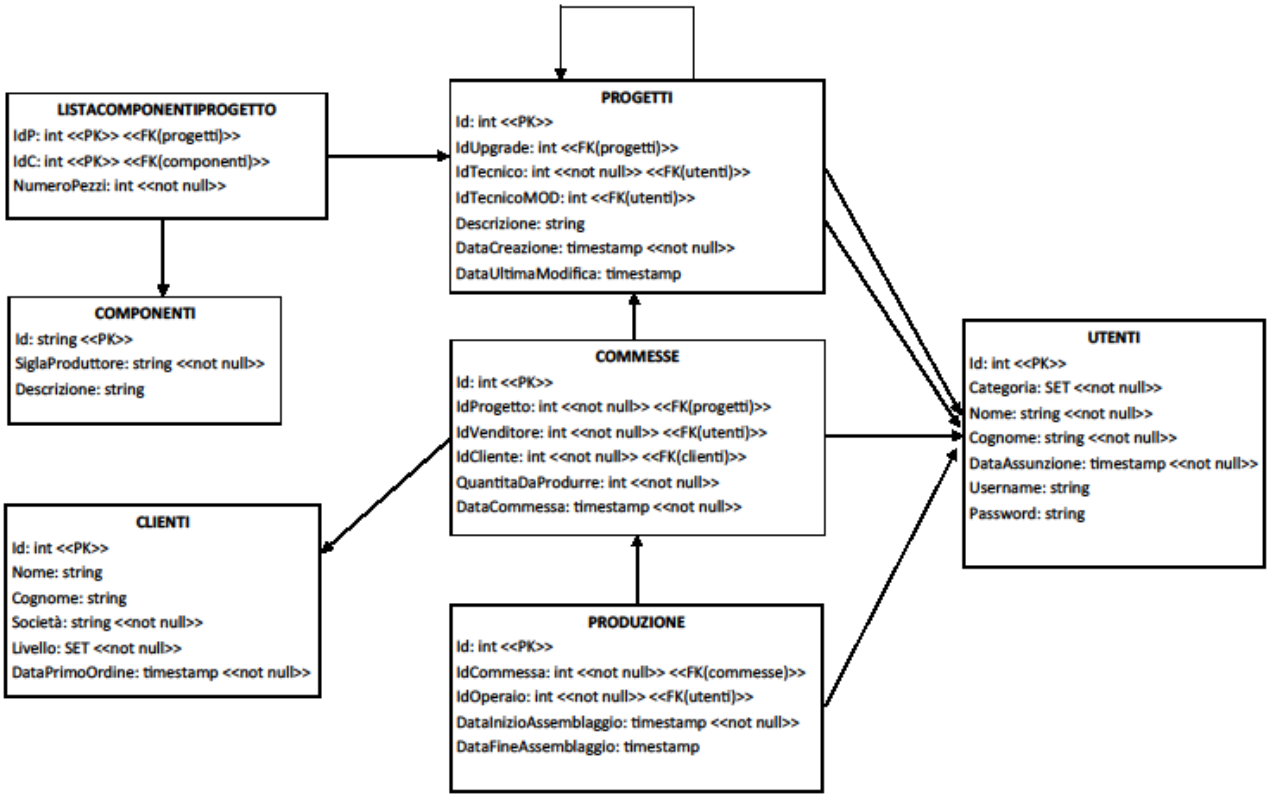
Progettazione Concettuale – Gerarchia delle classi

Gerarchia Utenti: Operai/Tecnici/Venditori/Amministratori

- Classi disgiunte
- L'unione è una copertura

Progettazione Logica

Schema Logico:



Rappresentazione Associazioni:

Tecnici-Progetti: crea

- Chiave esterna "IdTecnico" su progetti (not null)

Tecnici-Progetti: modifica

- Chiave esterna “IdTecnicoMOD” su progetti

Venditori-Commesse: commissiona

- Chiave esterna “IdVenditore” su commesse (not null)

Operai-Produzione: produce

- Chiave esterna “IdOperaio” su produzione (not null)

Progetti-Progetti: è un upgrade

- Chiave esterna “IdUpgrade” su progetti

Progetti-Componenti: è formato da

- Nuova relazione “ListaComponentiProgetto”
- Chiave esterna “IdP” a progetti
- Chiave esterna “IdC” a componenti
- Nuovo attributo: Numero Pezzi

Commesse-Clienti: è destinata

- Chiave esterna “IdCliente” su commesse (not null)

Commesse-Progetti: riguardano

- Chiave esterna “IdProgetto” su commesse (not null)

Commesse-Produzione: ha unità in produzione

- Chiave esterna “IdCommessa” su produzione

Gerarchia Utenti: Operai/Venditori/Tecnici/Amministratori

- Tabella Unica -> attributi delle sottoclassi inesistenti
- Attributo multivalore “Categoria” su utenti con valori: “operai, tecnici, venditori, amministratori” (not null)

Query, Trigger e Funzioni

Query 1

Query che calcola il tempo medio di produzione di una singola unità riguardante il progetto ID 1 (CubeX NFC Transceiver) costruite da operai assunti da meno di un mese:

```
SELECT  avg(TempoProduzioneUnita(p.Id)) as MediaOreSingolaScheda
FROM produzione p JOIN commesse c ON p.IdCommessa=c.Id
WHERE IdProgetto=1 AND EXISTS (
                                        SELECT *
                                        FROM utenti u
                                        WHERE p.IdOperaio=u.Id  AND
                                                DataAssunzione>DATE_SUB(CURDATE(), INTERVAL 31 DAY)
                                );
```

OUTPUT(query1):

```
+---------------------------------+
| MediaOreSingolaScheda |
+---------------------------------+
|                      67.0000 |
+---------------------------------+
```

Query 2

Query che calcola il cliente che ha fatto almeno 2 ordini nell'ultimo mese relativi a progetti che utilizzano il componente "TRF7970A" ID IC05:

```
SELECT Societa
FROM commesse c JOIN clienti z ON c.IdCliente=z.Id
WHERE DataCommessa>DATE_SUB(CURDATE(), INTERVAL 31 DAY) AND EXISTS (
                                        SELECT *
                                        FROM progetti p JOIN listacomponentiprogetto l ON p.Id=l.IdP
```

```
                                        WHERE l.IdC='IC05' AND c.IdProgetto=l.IdP
                                    )
    GROUP BY Societa HAVING count(*)>1;
```

OUTPUT(query2):

```
+------------------------------+
| Societa                      |
+------------------------------+
| Security Systems s.p.a. |
+------------------------------+
```

Query 3

Query che ritorna il nome, cognome e numero di schede prodotte nell'ultimo mese dei primi 3 operai che hanno prodotto più unità, dando la precedenza a quelli più "anziani" (presenti in azienda da più tempo):

```
    SELECT Nome, Cognome,(      SELECT count(*)
                                 FROM produzione pp
                                WHERE pp.IdOperaio=u.Id
                            ) as tot
    FROM utenti u JOIN produzione p ON u.Id=p.IdOperaio JOIN commesse c ON c.Id=p.IdCommessa
    WHERE DataCommessa>DATE_SUB(CURDATE(), INTERVAL 31 DAY)
     GROUP BY tot DESC,DataAssunzione DESC LIMIT 3;
```

OUTPUT(query3):

```
+-------------+----------------+------+
| Nome      | Cognome    | tot  |
+-------------+----------------+------+
| Matteo    | Giacomazzi |    3 |
| Alessio   | Bixio      |    2 |
| Giacomo | Petrin       |    2 |
+-------------+-----------------+------+
```

Query 4

Query che ritorna nome, cognome e tempo di produzione di una singola scheda dei 3 operai che hanno impiegato più di 3giorni per l'assemblaggio di una singola unità nell'ultimo mese:

```
    SELECT DISTINCT Nome,Cognome, TempoProduzioneUnita(p.Id) as tempo
    FROM produzione p JOIN utenti u ON p.IdOperaio=u.Id
    WHERE (
                SELECT TIMESTAMPDIFF(HOUR,px.DataInizioAssemblaggio,px.DataFineAssemblaggio)
                FROM produzione px
                WHERE px.Id=p.Id
          )> 72
```

```
        AND (p.DataInizioAssemblaggio>DATE_SUB(CURDATE(), INTERVAL 31 DAY))
    GROUP BY tempo DESC LIMIT 3;
```

OUTPUT(query4):

```
+----------+-------------+----------+
| Nome    | Cognome | tempo |
+----------+-------------+----------+
| Lorella  | Paolin      |   290    |
+----------+-------------+----------+
```

Query 5

Query che ritorna il nome, cognome ed il numero di progetti del tecnico che ha creato più progetti nell'ultima settimana con almeno 20 componenti:

```
    SELECT Nome,Cognome, (
                        SELECT count(*)
                        FROM progetti px
                        WHERE px.IdTecnico=u.Id AND  (
                            SELECT count(*)
                            FROM progetti p JOIN listacomponentiprogetto l ON p.Id=l.IdP
                            WHERE px.Id=p.Id
                            )>19
                        ) as nprogetti
    FROM progetti pp JOIN utenti u ON pp.IdTecnico=u.Id
    WHERE pp.DataCreazione>DATE_SUB(CURDATE(), INTERVAL 7 DAY)
    GROUP BY nprogetti DESC;
```

OUTPUT(query5):

```
+----------+-------------+---------------+
| Nome    | Cognome  | nprogetti  |
+----------+-------------+---------------+
| Michele | Guidolin   |             1 |
+----------+-------------+---------------+
```

Query 6

Query che ritorna nome, cognome ed il numero di commesse dei venditori che hanno firmato almeno 2 contratti con clienti relativamente nuovi nell'ultima settimana per un progetto sostanzioso (> 20componenti):

```
    SELECT Nome,Cognome, (SELECT count(*)
```

```
                        FROM commesse cc
                        WHERE cc.IdVenditore=u.Id
                                AND DataCommessa>DATE_SUB(CURDATE(), INTERVAL 7 DAY)
                        ) as numeroCommesse
FROM commesse c JOIN utenti u ON u.Id=c.IdVenditore
WHERE DataCommessa>DATE_SUB(CURDATE(), INTERVAL 7 DAY)
        AND EXISTS (
                        SELECT *
                        FROM clienti
                        WHERE Id=c.IdCliente AND Livello='new'
                    )
        AND  (
                SELECT count(*)
                FROM progetti p JOIN listacomponentiprogetto l ON p.Id=l.IdP
                WHERE c.IdProgetto=p.Id
              )>20
GROUP BY numeroCommesse DESC HAVING count(*)>1;
```

OUTPUT(query6):

```
+---------+---------------+-------------------------+
| Nome  | Cognome | numeroCommesse |
+---------+---------------+-------------------------+
| Fabio  | Grosso      |                       2 |
+---------+---------------+-------------------------+
```

Funzione 1

Funzione che calcola le ore necessarie alla produzione di una singola unità (scheda):

```
CREATE FUNCTION TempoProduzioneUnita(IdP SMALLINT)
RETURNS INT
BEGIN
DECLARE intervallo INT;
DECLARE inizio TIMESTAMP;
DECLARE fine TIMESTAMP;

SELECT DataFineAssemblaggio, DataInizioAssemblaggio INTO fine, inizio
FROM produzione
WHERE Id=IdP;

SELECT TIMESTAMPDIFF(HOUR,inizio,fine) INTO intervallo;

RETURN intervallo;
END$
```

Funzione 2

Funzione che calcola la data dell'ultimo lavoro riguardante una commessa (se la commessa è terminata la data coincide con la data di termine della commessa stessa):

```
CREATE FUNCTION DataFineUltimoLavoroCommessa(Idcomm SMALLINT)
RETURNS TIMESTAMP
BEGIN
DECLARE dataUltimoLavoro TIMESTAMP;

SELECT DataFineAssemblaggio INTO dataUltimoLavoro
FROM produzione p JOIN commesse c ON c.Id=p.IdCommessa
WHERE DataFineAssemblaggio IS NOT NULL AND IdCommessa=Idcomm
ORDER BY p.Id DESC LIMIT 1;

RETURN dataUltimoLavoro;

END$
```

Funzione 3

Funzione che calcola il numero di pezzi prodotti per una determinata commessa:

```
CREATE FUNCTION PezziProdotti(IdC SMALLINT)
RETURNS INT
BEGIN
DECLARE Pezzi INT;

SELECT COUNT(*) INTO Pezzi
FROM produzione
WHERE IdCommessa=IdC AND DataFineAssemblaggio IS NOT NULL;

RETURN Pezzi;
END$
```

Funzione 4

Funzione che calcola il numero di pezzi di una determinata commessa che sono attualmente in assemblaggio:

```
CREATE FUNCTION PezziInAssemblaggio(IdC SMALLINT)
RETURNS INT
BEGIN
DECLARE Pezzi INT;

SELECT COUNT(*) INTO Pezzi
FROM produzione
WHERE IdCommessa=IdC AND DataFineAssemblaggio IS NULL;

RETURN Pezzi;
END$
```

Funzione 5

Funzione che calcola il numero di pezzi ancora da produrre per poter terminare una determinata commessa:

```
CREATE FUNCTION PezziDaProdurre(IdC SMALLINT)
```

```
RETURNS INT
BEGIN
DECLARE PezziRichiesti INT;

SELECT QuantitaDaProdurre INTO PezziRichiesti
FROM commesse
WHERE Id=IdC;

SET PezziRichiesti=PezziRichiesti-(PezziProdotti(Idc)-PezziInAssemblaggio(Idc));

RETURN PezziRichiesti;
END$
```

Funzione 6

Funzione che calcola se un operaio è in assemblaggio su più di un determinato numero di lavori (produzioni). Funzione utilizzata per permettere ad un operaio di iniziare un assemblaggio solamente se ha terminato l'assemblaggio precedente.

```
CREATE FUNCTION UtenteInAssemblaggio(IdU SMALLINT,numeroLavori SMALLINT)
RETURNS BOOL
BEGIN
DECLARE LavoriInCorso INT;
DECLARE statoOperaio BOOL;

SELECT COUNT(*) INTO LavoriInCorso
FROM produzione
WHERE IdOperaio=IdU AND DataFineAssemblaggio IS NULL;

IF(LavoriInCorso>numeroLavori)
THEN
  SET statoOperaio=1;
ELSE
  SET statoOperaio=0;
END IF;

RETURN statoOperaio;
END$
```

Funzione 7

Funzione che calcola se una determinata commessa è terminata o comunque in fase di terminazione:

```
CREATE FUNCTION commessaTerminata(IdC SMALLINT)
RETURNS BOOL
BEGIN
DECLARE terminata BOOL;

SELECT count(*) INTO terminata
FROM commesse
WHERE Id=IdC AND (PezziProdotti(Id)+PezziInAssemblaggio(Id))=QuantitaDaProdurre;
```

```
RETURN terminata;
END$
```

Funzione 8

Funzione che calcola il numero di commesse ordinate da un determinato utente (utilizzata nel trigger 1):

```
CREATE FUNCTION NumeroCommesseCliente(IdC SMALLINT)
RETURNS INT
BEGIN
DECLARE numeroCommesse INT;

SELECT count(*) into numeroCommesse
FROM commesse
WHERE IdCliente=IdC;

RETURN numeroCommesse;
END$
```

Trigger 1

Trigger che aggiorna il livello del cliente al raggiungimento della quota successiva (oro>100, argento>50, bronzo>25)

```
CREATE TRIGGER bonusClienti AFTER INSERT ON commesse
FOR EACH ROW
BEGIN
    IF(NumeroCommesseCliente(new.IdCliente)>100)
    THEN
        UPDATE clienti
        SET Livello='oro'
        WHERE Id=new.IdCliente;
    ELSE IF (NumeroCommesseCliente(new.IdCliente)>50)
        THEN
            UPDATE clienti
            SET Livello='argento'
            WHERE Id=new.IdCliente;
        ELSE IF (NumeroCommesseCliente(new.IdCliente)>25)
            THEN
                UPDATE clienti
                SET Livello='bronzo'
                WHERE Id=new.IdCliente;
            END IF;
        END IF;
    END IF;
END$
```

Trigger 2

Trigger che nel caso venga creato un progetto “upgrade” di un progetto in fase assemblaggio chiude la commessa esistente e crea una nuova commessa relativa al nuovo progetto Upgrade con i pezzi rimanenti da produrre.

```
CREATE TRIGGER upgradeCommessa AFTER UPDATE ON progetti
FOR EACH ROW
BEGIN
DECLARE OLDcommessa SMALLINT;
DECLARE venditore SMALLINT;
DECLARE cliente SMALLINT;
DECLARE quantita INT;
DECLARE pezziProdotti INT;
DECLARE NEWquantita INT;

DECLARE flagFineFetch INT DEFAULT FALSE;
DECLARE cursoreCommesse CURSOR FOR
                SELECT Id,IdVenditore,IdCliente,QuantitaDaProdurre
                FROM commesse
                WHERE IdProgetto=new.IdUpgrade AND
(PezziProdotti(Id)+PezziInAssemblaggio(Id))<QuantitaDaProdurre;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET flagFineFetch=TRUE;

IF (old.IdUpgrade>=1 AND old.DataCreazione IS NULL AND new.Descrizione!="- in progettazione -" AND
commessaInCorso(new.IdUpgrade))
        THEN
                OPEN cursoreCommesse;
                loopAggiornaCommesse: LOOP

                        FETCH cursoreCommesse INTO OLDcommessa,venditore,cliente,quantita;
                        IF flagFineFetch THEN
                          LEAVE loopAggiornaCommesse;
                        END IF;

                        UPDATE commesse
                        SET
QuantitaDaProdurre=PezziProdotti(OLDcommessa)+PezziInAssemblaggio(OLDcommessa)
                        WHERE Id=OLDcommessa;

                        SET
pezziProdotti=PezziProdotti(OLDcommessa)+PezziInAssemblaggio(OLDcommessa);
                        SET NEWquantita=quantita-pezziProdotti;
                        INSERT INTO commesse (IdProgetto,IdVenditore,IdCliente,QuantitaDaProdurre)
                        VALUES (new.Id,venditore,cliente,NEWquantita);

                END LOOP;
                CLOSE cursoreCommesse;
        END IF;
END$
```