

Basi di Dati

Esercizi di vincoli, procedure e regole attive in SQL

Gestione magazzino

- ▶ Lo schema che utilizzeremo è il seguente:

MAGAZZINO (COD_PROD, QTA_DISP, QTA_RIORD)

RIORDINO (COD_PROD, DATA, QTA_ORD)

FK: COD_PROD REFERENCES MAGAZZINO

Gestione magazzino

- ▶ Statement SQL per le tabelle relative al database (schema) “ordini”:

CREATE TABLE MAGAZZINO

```
( COD_PROD    NUMERIC(3) NOT NULL,  
  QTA_DISP    NUMERIC(5),  
  QTA_RIORD   NUMERIC(5),  
  PRIMARY KEY (COD_PROD)  
);
```

CREATE TABLE RIORDINO

```
( COD_PROD    NUMERIC(3) NOT NULL,  
  DATA       DATE,  
  QTA_ORD     NUMERIC(5),  
  PRIMARY KEY (COD_PROD,DATA),  
  FOREIGN KEY (COD_PROD) REFERENCES MAGAZZINO (COD_PROD)  
);
```

Gestione magazzino

- ▶ Creare una funzione **prelievo()** per la gestione dei prelievi da magazzino e degli eventuali riordini di merce. Seguire le seguenti specifiche:
 - ▶ L'utente indica un prelievo dando il **codice** del prodotto e la **quantità** da prelevare
 - ▶ Viene eseguito il **prelievo**, modificando la quantità disponibile in magazzino
 - ▶ La funzione restituisce la nuova **quantità disponibile**

Gestione magazzino

- ▶ Statement SQL per la creazione della procedura prelievo() (versione PL/pgSQL per PostgreSQL):

```
CREATE FUNCTION prelievo(PROD INTEGER, QUANT INTEGER) RETURNS INTEGER AS $$
DECLARE
    Q1 INTEGER;
    Q2 INTEGER;
BEGIN
    SELECT QTA_DISP INTO Q1
    FROM MAGAZZINO
    WHERE COD_PROD = PROD;

    Q2 := Q1 - QUANT;

    UPDATE MAGAZZINO
    SET QTA_DISP = Q2
    WHERE COD_PROD = PROD;

    RETURN Q2;
END;
$$ LANGUAGE 'plpgsql';
```

Gestione magazzino

- ▶ Creare un trigger **gestione_riordino** per gestire automaticamente i seguenti aspetti:
 - ▶ Se si tenta di modificare la quantità disponibile con un numero negativo (la quantità disponibile in magazzino non è sufficiente per un prelievo) la procedura si arresta con una **eccezione**
 - ▶ Se la nuova quantità disponibile in magazzino è inferiore alla quantità di riordino si predispone un nuovo **ordine** d'acquisto.

Gestione magazzino

- ▶ Statement SQL per la creazione del trigger (versione PL/pgSQL per PostgreSQL):

```
CREATE FUNCTION riordino() RETURNS trigger AS $$
BEGIN
    IF NEW.QTA_DISP < NEW.QTA_RIORD THEN
        IF NEW.QTA_DISP < 0 THEN
            RAISE EXCEPTION 'Modifica impossibile';
        ELSE
            INSERT INTO RIORDINO
                VALUES(NEW.COD_PROD, CURRENT_DATE, NEW.QTA_RIORD);
        END IF;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE 'plpgsql';
```

```
CREATE TRIGGER GESTIONE_RIORDINO
AFTER UPDATE ON MAGAZZINO
FOR EACH ROW EXECUTE PROCEDURE riordino();
```

Gestione magazzino

- ▶ Per provare la funzione (un esempio):

```
INSERT INTO MAGAZZINO VALUES (1,150,100);
```

```
INSERT INTO MAGAZZINO VALUES (3,130,80);
```

```
INSERT INTO MAGAZZINO VALUES (4,170,50);
```

```
INSERT INTO MAGAZZINO VALUES (5,500,150);
```

```
SELECT prelievo(1,70);
```

```
SELECT prelievo(3,200);
```