

18. Modifica delle tabelle in SQL

Uso di SQL come DML

database

modifica delle tabelle in SQL

Abbiamo visto i comandi principali sfruttati da SQL per la definizione degli schemi di un database (sia al livello di database che di tabelle), oltre che per la definizione di vincoli intrarelazionali ed interrelazionali, evidenziandone, dunque, le potenzialità di DDL (*Data Definition Language*).

SQL fornisce anche comandi per la manipolazione (modifica e cancellazione) degli schemi di un database, che permettono di modificare le definizioni di tabelle e vincoli introdotte in precedenza, funzionalità di tipo DML (*Data Manipulation Language*).

modifica delle tabelle in SQL

Il comando **ALTER TABLE** permette di modificare lo schema delle tabelle:

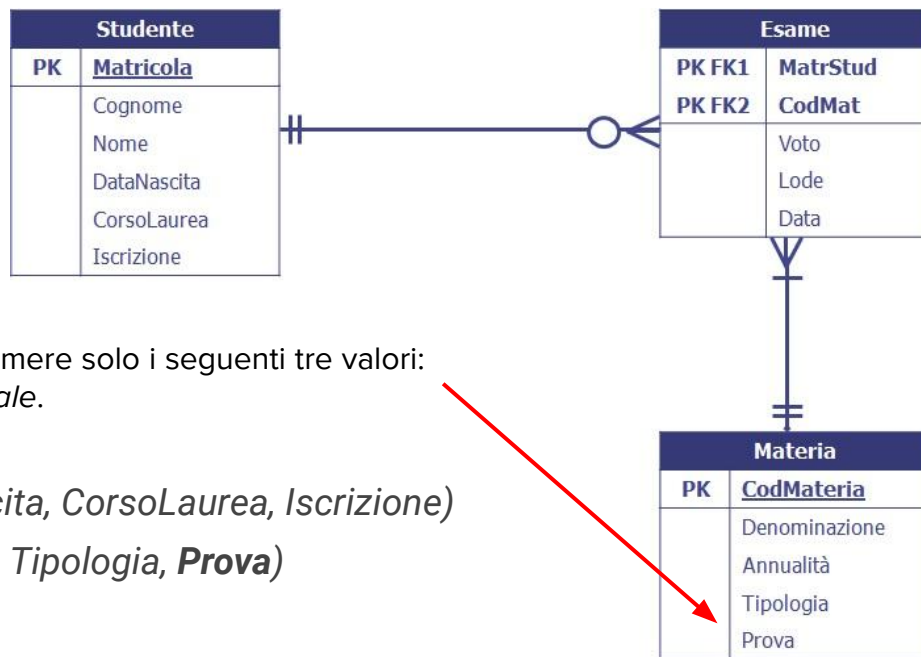
```
ALTER TABLE <NomeTabella> (  
    ALTER/MODIFY COLUMN <NomeAttributo> [SET DEFAULT <NuovoDefault>]  
                                DROP DEFAULT  
  
    ADD CONSTRAINT <DefVincolo>  
    DROP CONSTRAINT <NomeVincolo>  
    ADD COLUMN <DefAttributo>  
    DROP COLUMN <NomeAttributo>  
)
```

Tramite questo comando è possibile aggiungere o eliminare vincoli ed attributi, oppure modificare gli attributi esistenti, tenendo presente che eventuali nuovi vincoli devono comunque essere compatibili con i dati già inseriti.

modifica delle tabelle in SQL

Riprendiamo l'esempio precedente per la registrazione degli esami svolti dagli studenti di un ateneo universitario.

Vogliamo aggiungere l'attributo **Prova** nella tabella **Materia** in modo da indicare la modalità di svolgimento dell'esame.



Studente(Matricola, Cognome, Nome, DataNascita, CorsoLaurea, Iscrizione)

*Materia(CodMateria, Denominazione, Annualità, Tipologia, **Prova**)*

Esame(MatrStud, CodMat, Voto, Lode, Data)

modifica delle tabelle in SQL

Possiamo modificare la struttura della tabella `Esame` con il seguente comando:

```
ALTER TABLE Materia
```

← Nome della tabella della quale modificare lo schema.

```
ADD COLUMN Prova ENUM('Scritto', 'Orale', 'Scritto\Orale')
```

← Aggiungiamo la colonna `Prova` di tipo `ENUM` con i valori indicati

Il nuovo campo può essere rinominato da `Prova` in `TipoProva`:

```
ALTER TABLE Materia
```

```
CHANGE Prova TipoProva CHAR(12)
```

← Oltre al vecchio nome `Prova` ed al nuovo nome `TipoProva`, bisogna anche indicare il tipo (che ovviamente può essere quello precedente).

Ed eliminato usando il comando:

```
ALTER TABLE Materia
```

```
DROP COLUMN TipoProva
```

← Elimina la colonna `TipoProva` dallo schema della tabella.

modifica delle tabelle in SQL

Proviamo adesso a modificare lo schema originale della tabella `Esame` partendo dalla definizione iniziale senza i vincoli intrarelazionali sul voto e senza le chiavi esterne.

```
CREATE TABLE Esame(  
    MatrStud INT,  
    CodMat CHAR(5),  
    Voto INT NOT NULL,  
    Lode BOOLEAN DEFAULT NULL,  
    Data DATE DEFAULT NULL,  
    PRIMARY KEY(MatrStud, CodMat)           Esame(MatrStud, CodMat, Voto, Lode, Data)  
)
```

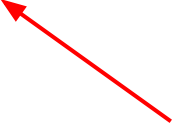
Possiamo imporre 18 come voto di default (**DROP DEFAULT** per eliminarlo):

```
ALTER TABLE Esame  
    ALTER COLUMN Voto SET DEFAULT 18
```

modifica delle tabelle in SQL

Possiamo inserire un vincolo di tupla per imporre un voto che sia compreso tra 18 e 30:

```
ALTER TABLE Esame  
  MODIFY COLUMN Voto INT NOT NULL CHECK (Voto>=18 AND Voto<=30)
```



NB: Alcuni RDBMS (come MySql) non ammettono l'uso del comando `ALTER` per modificare la struttura di una colonna; in questi casi è meglio utilizzare il comando `MODIFY`.

Ed un secondo vincolo di tupla per evitare di registrare la lode se il voto è diverso da 30:

```
ALTER TABLE Esame  
  ADD CONSTRAINT ChkLode CHECK ((Voto=30 AND Lode=1) OR  
  (Voto>=18 AND Voto<=30 AND Lode=0))
```

modifica delle tabelle in SQL

Mentre il comando seguente imposta una chiave esterna sul campo `MatrStud` della tabella `Esame` collegandolo alla chiave primaria `Matricola` della tabella `Studente`:

```
ALTER TABLE Esame
  ADD FOREIGN KEY (MatrStud) REFERENCES Studente(Matricola)
  ON UPDATE NO ACTION ON DELETE CASCADE
```

Il comando seguente imposta la seconda chiave esterna dichiarandola come vincolo:

```
ALTER TABLE Esame
  ADD CONSTRAINT FK_Materia
  FOREIGN KEY(CodMat) REFERENCES Materia(CodMateria)
  ON UPDATE NO ACTION ON DELETE CASCADE
```


modifica delle tabelle in SQL

Il comando **DROP** permette di rimuovere componenti dal database, ad es. per rimuovere l'intera tabella `Esame` possiamo lanciare semplicemente il comando:

```
DROP TABLE Esame
```

Il comando **DROP** può rimuovere anche l'intero database:

```
DROP DATABASE RegistroEsami
```

NB: Con il comando **TRUNCATE** possiamo eliminare tutte le righe di una tabella senza rimuovere la tabella stessa che continua a mantenere la struttura originale:

```
TRUNCATE TABLE Studente
```

mentre il comando **RENAME TABLE** per cambiarne il nome:

```
RENAME TABLE Studente TO Allievo
```

modifica delle tabelle in SQL

Il comando di inserimento di nuovi record nel DB presenta due sintassi alternative, la prima delle quali permette di inserire singole righe all'interno delle tabelle:

```
INSERT INTO <NomeTabella> [ListaAttributi]  
                VALUES (ListaDiValori) | SelectSQL
```

La prima forma è quella tipicamente utilizzata all'interno dei programmi per riempire una tabella con i dati forniti direttamente dall'utente e normalmente è associata al riempimento di un *form* di input sul monitor.

modifica delle tabelle in SQL

Volendo ad es. inserire i dati di un nuovo studente (*11111, Verdi, Ausonia, 2003-02-01, Matematica, 850.00*) possiamo usare il comando seguente:

```
INSERT INTO studente(Matricola, Cognome, Nome, DataNascita, CorsoLaurea, Iscrizione)  
          VALUES('11111', 'Verdi', 'Ausonia', 2003-02-01, 'Matematica', 850.00)
```

NB: Se in un inserimento non vengono specificati i valori di tutti gli attributi della tabella, agli attributi mancanti viene assegnato il valore di default, o in assenza di questo il valore NULL.

Se un inserimento viola un vincolo definito su un attributo, l'inserimento viene rifiutato.

modifica delle tabelle in SQL

La seconda forma del comando permette invece di aggiungere insiemi di righe estratte da altre tabelle dello stesso database, mediante l'uso del comando SELECT.

Ad es. dopo aver creato la tabella `studentimatematica` per registrare i soli studenti iscritti alla facoltà di matematica, possiamo riempire la nuova tabella col comando:

```
INSERT INTO studentimatematica  
    (SELECT *  
     FROM studente  
     WHERE CorsoLaurea = 'Matematica')
```

modifica delle tabelle in SQL

Il comando di cancellazione consente l'eliminazione dei record da una tabella mediante la sintassi seguente:

```
DELETE FROM <NomeTabella> [WHERE Condizione]
```

Ad es. per eliminare dalla tabella `studente` tutti gli studenti iscritti al corso di laurea in Matematica, possiamo usare il comando:

```
DELETE FROM studente  
WHERE CorsoLaurea = 'Matematica'
```

NB: Quando la condizione argomento della clausola `WHERE` non viene specificata, il comando cancella tutte le righe della tabella, altrimenti vengono rimosse le sole righe che soddisfano la condizione.

modifica delle tabelle in SQL

Il comando di modifica `update` presenta la sintassi seguente:

```
UPDATE <NomeTabella>  
    SET <Attributo1 = Espressione | SelectSQL | NULL | DEFAULT>,  
    SET <Attributo2 = Espressione | SelectSQL | NULL | DEFAULT>, ...  
    [WHERE Condizione]
```

Il comando `UPDATE` aggiorna uno o più campi della tabella che soddisfano l'eventuale condizione; se assente la clausola `WHERE`, la modifica si esegue su tutte le righe.

Il nuovo valore cui viene posto l'attributo può essere: il risultato di un'espressione sugli attributi della tabella (che può anche fare riferimento al valore corrente dell'attributo che verrà modificato dal comando stesso); il risultato di una generica interrogazione SQL; il valore nullo; il valore di default.

modifica delle tabelle in SQL

Ad es. per aumentare del 10% il costo di iscrizione (inizialmente di €. 450.00) per tutti gli iscritti al corso di laurea in Matematica, si può sfruttare il comando:

```
UPDATE studente  
  SET Iscrizione = Iscrizione+Iscrizione*0.1  
  WHERE CorsoLaurea = 'Matematica'
```

NB: L'operatore di assegnamento = ha un comportamento analogo a quello dei normali linguaggi di programmazione, per cui `Iscrizione` sul lato destro dell'operatore rappresenta il vecchio valore dell'attributo, valutato per ogni riga sulla quale deve essere applicato l'aggiornamento.